Universidad Nacional del Sur

Tesis de Doctor en Ciencias de la Computación

Desarrollo de técnicas de computación evolutiva para soporte en minería de datos y texto

Rocío L. Cecchini

Bahía Blanca Argentina

2010

Prefacio

Esta Tesis se presenta como parte de los requisitos para optar al grado Académico de Doctor en Ciencias de la Computación, de la Universidad Nacional del Sur y no ha sido presentada previamente para la obtención de otro título en esta Universidad u otra. La misma contiene los resultados obtenidos en investigaciones llevadas a cabo en el ámbito del Departamento de Ciencias e Ingeniería de la Computación durante el período comprendido entre el 1 de abril de 2006 y el 13 de abril de 2010, bajo la dirección de la Dra. Nélida B. Brignole, Profesora del Departamento de Ciencias e Ingeniería de la Computación e investigadora independiente del CONICET y del Dr. Gustavo E. Vazquez, Asistente del Departamento de Ciencias e Ingeniería de la Computación e investigador asistente del CONICET.

.....

Rocío L. Cecchini
rlc@cs.uns.edu.ar
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Bahía Blanca, 13 de abril de 2010

UNIVERSIDAD NACIONAL DEL SUR
Secretaría General de Posgrado y Educación Continua

La presente tesis ha sido aprobada el .../.../..., mereciendo la calificación de .....(.....)

Agradecimientos

Quiero expresar mi agradecimiento a la Dra. Nélida Beatriz Brignole y al Dr. Gustavo Esteban Vazquez, las dos personas que me mostraron el portal y el camino hacia el mundo de la investigación durante estos años. Ellos me han animado a dar pasos muy importantes en este sendero y me han permitido aprender mucho sobre la investigación.

También quiero expresar un agradecimiento especial a la Dra. Ana Gabriela Maguitman y al Dr. Ignacio Ponzoni por su ejemplo, su paciencia y su apoyo invaluable en sus áreas de conocimiento. Tener la oportunidad de trabajar con ellos ha sido un privilegio y una experiencia enriquecedora.

Además, me gustaría darles las gracias a mis compañeros de trabajo de cada día, en especial a los chicos de la sala de becarios y a Telma, Clara y Dana, por brindarme su amistad y lograr que este trabajo sea aún más reconfortante. En particular, quiero agradecerle a Axel por ser, además, un colaborador inquebrantable y compañero en estos pasos hacia la investigación.

Por otro lado, le estoy muy agradecida al Departamento de Ciencias e Ingeniería de la Computación, a la universidad Nacional de Sur y al Concejo Nacional de Investigaciones Científicas y Técnicas, por ofrecerme los medios y herramientas con los cuales hacer esto posible.

Finalmente le doy las gracias a mi familia y afectos, tanto a los que están aquí como a los que están lejos, por brindarme su aliento, su cariño y su compañía que son mis pilares más fuertes. En especial le agradezco a Kcho, por ser un colega y compañero sin igual tanto en el trabajo como en la vida misma y por el amor, la amistad y el tiempo compartidos.

Rocío L. Cecchini

Resumen

La obtención de información a partir de un conjunto de datos o minería de datos es una tarea compleja que involucra varias etapas, tal como sucede en la minería de texto. Esta puede ser considerada como un caso particular de minería de datos donde los datos contemplan la incorporación de texto. Ambos procesos de minería se vuelven aun más complejos cuando nos encontramos ante grandes cúmulos de datos o texto. Es común encontrar conjuntos de datos grandes, complejos y ricos en información en áreas como medicina, comercio, ingeniería y ciencias de la computación. Simultáneamente, los avances tecnológicos han dado lugar a la acumulación de sustanciosas cantidades de documentos, artículos y texto; el ejemplo más contundente de esta clase de material es la Web, la cual se estima que alcanza más de 8.05 billones de páginas. La propuesta de esta tesis es el uso de herramientas evolutivas mono- y multi-objetivo como un soporte para algunas de las etapas de este proceso. En particular, las etapas que implican optimización y búsqueda dentro de estos grandes espacios en los cuales otros métodos serían inviables. A lo largo de la investigación se desarrollaron, evaluaron y compararon algoritmos evolutivos mono- y multi-objetivo tanto para la rama de minería de datos como para la rama de minería de texto. Como caso particular dentro de minería de datos, se contempló el problema de encontrar las relaciones más relevantes entre variables dentro de distintos conjuntos de datos. Dichas relaciones, no son visibles para un experto cuando se encuentra frente a la base de datos original cruda, la cual puede contemplar miles de variables y miles de instancias. Para resolver este problema se propuso una metodología de dos fases. Los algoritmos desarrollados en este contexto se integraron a la primera fase de la arquitectura y fueron exitosamente utilizados como mecanismo de búsqueda masiva. Por otra parte, en el caso de minería de texto se abordó el problema de recuperar información relacionada y novedosa con respecto a un tópico de interés. Para este problema se propuso, implementó y evaluó una arquitectura que, partiendo de una descripción para el tópico de interés, evoluciona varios conjuntos de términos hacia conjuntos que logren obtener mejores documentos con respecto a dicho tema de interés y con respecto a los objetivos propuestos (por ejemplo: similitud, precisión, cobertura). Dentro de las técnicas evolutivas multi-objetivo propuestas, se diseñaron adaptaciones de los algoritmos basados en Pareto más prometedores reportados por la literatura y se propusieron versiones multi-objetivo agregativas. Ambos enfoques, los basados en Pareto y los agregativos, demostraron ser claramente competentes tanto para minería de datos como para minería de texto.

Abstract

Data mining comprises the capture of information from data, which is a complex task that involves many stages. The same applies to text mining that can be considered as a special case of data mining where the data include text. As data and text sets increase, both mining processes become even more complicated. Large, complex and rich information data sets arise in many common research fields like medicine, commerce, engineering and computer science. Simultaneously, technological advances have led to the accumulation of substantial amounts of documents, articles and text; the clearest example of this kind of material is the Web, which is estimated to have reached more than 8.05 billion pages. This thesis proposes the use of mono- and multi-objective evolutionary tools as support in some of the stages of the data and text mining processes. In particular, those stages which imply optimization and search in wide search spaces where other methods could be unfeasible. In this research work, several mono- and multi-objective evolutionary algorithms were developed, evaluated and compared for both, data and text mining research areas. As a particular case in data mining, the problem of finding the most relevant relationship among variables from the data was considered. These relations, are not obvious for experts when they are faced with the original raw database, which can include thousands of variables and thousand of samples. In order to solve this problem, a two-phase methodology was proposed. In this context, the developed algorithms were integrated into the first phase and were succesfully used as massive search mechanisms. On the other hand, as a particular case of the text mining research area, the problem of retrieving novel material that is related to a search context was considered. In order to overcome this problem, an architecture was proposed, implemented and evaluated. Starting from a description for the topic of interest, this architecture evolves several sets of terms towards sets which can obtain better documents with respect to both, the topic of interest and the proposed objectives (e.g., similarity, precision, recall). Among the proposed multi-objetive evolutionary techniques, adaptations of the more promising reported Pareto-based evolutionary algorithms were designed and new multi-objective aggregative schemes were proposed. Both approaches- i.e., the Pareto-based strategy and the aggregative techniques- proved to be clearly competent for both research areas: data and text mining.

Índice

1 Introducción **1**

1.1 Minería de datos y texto 1

1.2 Objetivos y alcance 3

1.3 Revisión sobre algoritmos evolutivos propuestos 7

1.3.1 Algoritmos evolutivos desarrollados en minería de datos 7

1.3.2 Algoritmos evolutivos desarrollados en minería de texto 10

1.4 Estructura de la tesis 13

2 Conceptos básicos de optimización **15**

2.1 Optimización mono y multi-objetivo 16

2.2 Problema de optimización mono-objetivo 18

2.3 Problema de optimización multi-objetivo 18

2.3.1 Concepto de dominancia 20

2.3.2 Optimalidad de Pareto 22

3 Algoritmos evolutivos **25**

3.1 Teoría evolutiva básica (un poco de biología) 25

3.1.1 Supervivencia del más apto 26

3.1.2 Evolución 27

3.2 Algoritmos evolutivos 28

3.2.1 Analogía entre el concepto natural y el artificial 28

3.2.2 Principales ejemplos de algoritmos evolutivos 30

3.2.3 Definición formal de algoritmo evolutivo básico 31

3.2.4 Una definición formal de algoritmo evolutivo multi-objetivo 32

4 Algoritmos evolutivos multi-objetivo **35**

4.1 Clasificación de técnicas para resolver problemas multi-objetivo 35

4.2 Uso de algoritmos evolutivos 37

4.3 Enfoques evolutivos 38

4.4 MOEAs no-Pareto: reseña histórica 38

4.4.1 Esquemas agregativo-escalares 39

4.4.2 Esquemas orden-agregativos 40

4.5 MOEAs basados en Pareto: reseña histórica 42

4.5.1 Fitness Sharing Clásico - Concepto 42

4.5.2 Primera Generación 43

4.5.3 Segunda generación 46

5 Algoritmos evolutivos desarrollados para minería de datos **55**

5.1 Selección de características (Feature Selection) 56

5.2 Antecedentes en selección de características 61

5.3 Escenarios de alcance 63

5.4 Uso de algoritmos evolutivos en FS 65

5.5 Infraestructura de dos fases propuesta para selección de características . . 67

5.5.1 Objetivos generales de investigación 67

5.5.2 Arquitectura propuesta 68

5.5.3 Alcances de esta tesis dentro de la infraestructura propuesta 69

5.5.4 Estructura interna del wrapper 69

5.6 Primera implementación: versión mono-objetivo 71

5.6.1 Cuestiones de investigación 71

5.6.2 Generación de la población inicial 72

5.6.3 Operadores genéticos 72

5.6.4 Función de aptitud 72

5.6.5 Evaluación del algoritmo 75

5.7 Segunda implementación: versión multi-objetivo 83

5.7.1 Cuestiones de investigación 84
5.7.2 Algoritmos evolutivos multi-objetivo utilizados 85
5.7.3 Generación de la población inicial 85
5.7.4 Operadores genéticos 85
5.7.5 Evaluación de los subconjuntos 86
5.7.6 Ubicación de las nuevas consideraciones 87
5.7.7 Evaluación del algoritmo 88

6 Algoritmos evolutivos desarrollados para minería de texto 113

6.1 Recuperación de información temática 113
6.1.1 Antecedentes en búsqueda temática basada en contexto 115
6.1.2 Adaptabilidad de consultas y búsqueda temática 118
6.1.3 Escenarios de alcance 119
6.2 Uso de algoritmos evolutivos en recuperación de información 122
6.3 Generación de consultas temáticas como problema de optimización 125
6.3.1 Precisión y cobertura 125
6.3.2 Modelo de ponderación TF-IDF 126
6.3.3 Similitud 128
6.4 ¿Por qué usar algoritmos evolutivos para generación de consultas de alta calidad? 129
6.5 Infraestructura evolutiva propuesta para generación de consultas temáticas 131
6.5.1 Cuestiones de investigación generales 131
6.5.2 Arquitectura propuesta 132
6.6 Primera implementación: versión mono-objetivo 135
6.6.1 Cuestiones de investigación 135
6.6.2 Selección 136
6.6.3 Función de aptitud basada en similitud por coseno 136
6.6.4 Desempeño de la arquitectura 138
6.6.5 Incorporación de elitismo y función de aptitud basada en similitud novedosa 145

6.7 Segunda implementación: versión multi-objetivo 150
6.7.1 Cuestiones de investigación 151
6.7.2 Esquema evolutivo multi-objetivo para evolución de consultas temáticas 152
6.7.3 Selección 152
6.7.4 Evaluación de las consultas 152
6.7.5 Rendimiento del algoritmo 154

7 Conclusiones **169**
7.1 Revisión 169
7.2 Principales contribuciones 170
7.2.1 Aproximación mono-objetivo para selección de características 171
7.2.2 Aproximación multi-objetivo para selección de características 172
7.2.3 Aproximación mono-objetivo para recuperación de información temática 174
7.2.4 Aproximación multi-objetivo para recuperación de información temática 175
7.3 Trabajo futuro 176

Lista de figuras **179**

Lista de tablas **183**

Bibliografía **191**

Capítulo 1

Introducción

El propósito general de esta tesis es el desarrollo y la evaluación de herramientas evolutivas que sean capaces de extraer información útil a partir de datos o texto. El término *información* se aplica a uno de los primeros niveles en la escala del entendimiento humano [Sha04, Row07]. De acuerdo a esta jerarquía, los *datos* constituyen el nivel más elemental y, entre otras muchas definiciones, pueden verse como entidades, símbolos o números que por sí solos no ofrecen un significado útil. En un segundo nivel de la jerarquía se encuentra la *información*, la cual es extraída a partir de los datos y posee un grado más elevado de abstracción, dado que es capaz de responder a ciertas cuestiones, por ejemplo, la relación existente entre determinadas variables de un conjunto de datos. La extracción de información a partir de datos no es una tarea simple y presenta diferentes desafíos, algunos de los cuales son dependientes del problema en cuestión.

1.1 Minería de datos y texto

La tecnología ha facilitado la captura y el almacenamiento de enormes cantidades de datos y texto. Además, gran porcentaje de esta información se encuentra en forma de datos crudos o texto no organizado. Encontrar información útil dentro de estos cúmulos de datos (tales como tendencias, patrones, relaciones o anomalías) y resumirla en un modelo simple, constituye un importante desafío. Desde hace años, los datos han sido analizados para extraer información útil por medio de diferentes herramientas. Más aun, llevar a cabo esta tarea sobre grandes repositorios ha dado origen a un campo de investigación particular, la *minería de datos* (o *datamining*). Hoy en día, ésta disciplina es un área de investigación consolidada y es responsable del estudio y desarrollo de herramientas basadas en métodos estadísticos, aprendizaje automatizado, teoría de la información y

computación, destinadas a convertir grandes cantidades de datos en información más satisfactoria para ser interpretada por personas. De este modo, la minería de datos juega un rol importante en las etapas que conllevan a la obtención de conocimiento. Es importante notar que el problema de elucidar o estimar dependencias, o descubrir nuevos datos o nueva información a partir de los datos originales, solamente es una parte del proceso general completo a partir del cual los usuarios pueden arribar a conclusiones finales. El proceso general involucra al menos los siguientes pasos:

- **Establecer el problema y formular hipótesis**. La mayoría de los estudios de modelamiento basados en datos se realizan en un dominio de aplicación particular. En esta tesis hemos procurado cubrir problemas de gran generalidad, tanto para el caso de minería de datos como para el caso de minería de texto, apuntando a aquellas situaciones en las que los algoritmos evolutivos fueran consideradas como una herramienta adecuada.
- **Recolectar datos**. Los datos utilizados en los distintos experimentos realizados fueron datos extraídos de fuentes reales, es decir datos verdaderos existentes para los casos de estudio particulares en los que se probaron los distintos algoritmos evolutivos.
- **Preprocesar los datos**. Si bien los datos fueron obtenidos de fuentes especializadas, en cada caso de estudio particular, se hicieron trabajos de preprocesamiento previo a la aplicación de los algoritmos evolutivos con el objetivo de mejorar la confiabilidad de los resultados finales. Por ejemplo, en el caso de minería de datos, se eliminaron variables linealmente dependientes, se realizaron normalizaciones y se seleccionaron características de acuerdo con el conocimiento previo del conjunto de datos. En el caso de minería de texto, se realizaron pasos de preprocesamiento como la eliminación de stopwords y se eligieron conjuntos de páginas que cumplieran determinadas restricciones que aseguraran su calidad (por ejemplo, cuando se evaluó precisión y cobertura se tomaron, para las pruebas, tópicos que contuvieran al menos 100 documentos).
- **Estimar el modelo (minería)**. La principal tarea en este paso es aplicar la técnica automática de minería que se considera adecuada. Esta tesis está centrada en la aplicación de algoritmos evolutivos como herramienta de soporte dentro de la tarea de minería, tanto en datos como en texto. Si bien, cuando fue necesario se realizaron algunas tareas concernientes a las demás etapas, el enfoque central de

esta tesis se vuelca al desarrollo de herramientas evolutivas mono y multi-objetivo como soporte para las técnicas de minería de datos, en particular proponemos el uso de una herramienta alternativa de búsqueda y optimización en minería cuando el conjunto de datos o el corpus de documentos manifiestan importante magnitud.

- **Interpretar el modelo y sacar conclusiones**. En un sentido práctico, los métodos de minería deberían ser de ayuda en la tarea de toma de decisiones, por lo tanto, los modelos obtenidos deberían ser interpretables. El problema de la representación de los resultados obtenidos es una etapa importante, ya que es lo que van a apreciar los usuarios finales. Debido a la complejidad de esta etapa, algunos textos de minería de datos contemplan dentro de su alcance los dos pasos previos y reservan esta última etapa a libros más específicos [Han06]. En el contexto de esta tesis, en todos los experimentos se hizo un análisis de resultados y se extrajeron conclusiones. Sin embargo, dado que esta tesis no está enfocada en esta última etapa, los estudios realizados se centran en la evaluación de los algoritmos propuestos.

1.2 Objetivos y alcance

Nuestra habilidad para analizar y entender grandes conjuntos de datos y grandes corpus de documentos, no es tan amplia como nuestra habilidad para almacenarlos. Existen grandes conjuntos de datos y documentos útiles dispersos por muchas computadoras alrededor del mundo. Los instrumentos científicos son capaces de generar miles de miles de bytes electrónicos y almacenarlos en una computadora en poco tiempo y algo similar ocurre con los documentos de texto. La era de la información ha logrado un crecimiento exponencial en la capacidad de los medios de almacenamiento. El problema que se plantea a partir de estos hechos, es que el tamaño y la dimensionalidad de los datos y la desorganización y diversidad de los documentos son demasiado grandes para poder ser analizados e interpretados de manera manual. Sin embargo, los métodos de minería de datos y texto deberían resultar favorecidos con estas características que parecen poco favorables, dado que pueden sacar provecho de estos enormes cúmulos de información no explícita. Esto se debe a que los grandes conjuntos de datos o los grandes corpus de documentos, tienen el potencial de conducirnos a información más valiosa.

La propuesta de esta investigación es el diseño y evaluación de algoritmos evolutivos como mecanismo alternativo heurístico para sobrellevar la exploración dentro de estos grandes espacios de búsqueda, no necesariamente como única herramienta a utilizar dentro del

proceso de minería, sino también como un paso previo de refinamiento para reducir la complejidad del problema.

Desde un punto de vista general, los datos (tanto datos como texto) pueden clasificarse en *estructurados*, *semi-estructurados* o *no estructurados*. Un ejemplo de dato estructurado son las tablas constituidas por campos bien definidos, con valores numéricos o alfanuméricos. Por otro lado, los reportes (por ejemplo: médicos o comerciales) suelen ser tomados como datos semi-estructurados. Por último, los documentos de texto son un ejemplo clásico de dato no estructurado. Los datos estructurados suelen ser referidos como el tipo de dato tradicional, en tanto que los semi-estructurados y los no estructurados suelen ser mencionados como los tipos de datos no tradicionales [Kan03, ABS00].

Muchos de los métodos de minería de datos y las herramientas comerciales suelen aplicarse a datos tradicionales. Sin embargo, las herramientas e interfaces para lidiar con datos no tradicionales están en rápida evolución. En particular, los algoritmos evolutivos desarrollados a lo largo de esta tesis están dirigidos a conjuntos de datos estructurados y a documentos de texto. Sin embargo, pueden ser extendidos para procesar otros tipos de conjuntos de datos.

El modelo estándar para datos estructurados en minería de datos es una colección de casos. Se especifican medidas potenciales, denominadas *características* (o *features* en inglés), y para cada una de ellas se realizan mediciones en forma uniforme. La representación usual en problemas de minería de datos para los datos estructurados es una tabla, en la cual las columnas representan *características* de los objetos y las filas representan valores o *muestras* de dichas características para distintas situaciones.

Existen varias formas de analizar *características*. Una de las principales es determinar si existe relación entre las *características* (variables dependientes) del sistema con respecto a un valor objetivo (variables independientes). En este contexto, nuestra propuesta es emplear algoritmos evolutivos para determinar cuáles *conjuntos de características* pueden estar relacionadas con respecto a cierto valor objetivo, empleando diferentes métricas útiles para predicción. Esta tarea de selección de características presenta múltiples instancias de aplicación en problemas reales complejos como: *análisis de niveles de expresión de genes* (usando microarrays) [MM04, GST$^+$99], *extracción de genes para diagnóstico de cáncer* por medio de eliminación recursiva de características (RFE por sus siglas en inglés de Recursive Feature Elimination) [RTR$^+$01], *análisis de relación cualitativa y cuantitativa de propiedades para diseño de drogas* [LLYY08], *filtrado de texto* [BEYT$^+$03] o *reconocimiento de caras* [GBNT04]. Los principales aportes de los algoritmos evolutivos

propuestos, los cuales serán vistos en detalle en el capítulo 5, radican en formar parte de un proceso de dos fases, el cual, en la primera etapa se sirve de los algoritmos evolutivos para realizar una búsqueda de nivel grueso que no sobre-ajuste al conjunto de datos. Más adelante, en el capítulo 5, se detallará la metodología de dos fases que enmarca nuestra propuesta.

Dado que los algoritmos evolutivos constituyen una herramienta poderosa en problemas de optimización duros, han sido considerados como uno de los métodos alternativos dentro de la minería de datos [WF05, Kan03] y texto. Minaei-Bidgoli *et. al* proponen el uso de algoritmos genéticos para clasificación de estudiantes utilizando un número de características [MBP03]. Sin embargo, la mayoría de estos trabajos están orientados a clasificación. En nuestro trabajo, en el caso de datos estructurados, el objetivo es la extracción de información a partir de características de tal forma que se lleven a cabo de manera automática dos tareas principales: deducir *cuáles* son los descriptores más relevantes y estimar *cuál es el número* de descriptores aproximado que puede llevar a un modelo aceptable, procurando que dicho número sea mínimo. Cuando nos encontramos ante un gran número de descriptores (n) y poco conocimiento acerca de cuáles y cuántos son necesarios (k) para construir un modelo para el conjunto de datos dado, utilizar un mecanismo de selección exhaustivo, requiere intentar $\binom{n}{1} + \binom{n}{2} + ... + \binom{n}{k}$ veces para encontrar el conjunto de descriptores más relevantes, con lo cual, el tiempo de complejidad es de $O(2^n)$. Se ha demostrado que el problema de Selección de Descriptores (FS por sus siglas en inglés de Feature Selection) es NP-completo [DR94]; por lo tanto, el mecanismo computacional que se utilice debe seguir una alternativa heurística para lograr encontrar un subconjunto de variables razonablemente bueno en un período de tiempo aceptable. En base a estas características, los algoritmos evolutivos constituyen una herramienta adecuada para la elaboración de algoritmos inteligentes capaces de colaborar en la resolución del problema. Nuestra propuesta consiste en incorporar una etapa evolutiva como primera fase de refinamiento dentro de una arquitectura más compleja. Esta arquitectura posee una segunda etapa, basada en el uso de redes neuronales, que mejora los resultados brindados por la primera. A través de este proceso de dos etapas se logra alcanzar resultados de alta calidad con un algoritmo de complejidad razonable.

En segundo término, para el caso de datos no estructurados conformados por documentos de texto, nuestra atención se centrará en el problema de *recuperación temática automatizada*. El crecimiento veloz y las cantidades enormes de datos y texto en forma electrónica

han aumentado la importancia y complejidad de este tipo de recuperación de información. Como una definición general, podemos decir que la búsqueda basada en tópico o búsqueda temática es un proceso que partiendo de grandes corpus de documentos permite la recuperación de información respectiva a un tema de interés. Dicho tema de interés puede ser, por ejemplo, el entorno actual de un usuario utilizando una computadora, un documento inicial a partir del cual queremos obtener más información o una descripción conformada por un conjunto de palabras. A su vez, la *recuperación temática automatizada* es la realización de una búsqueda sobre un tópico por medio de un mecanismo computacional. El proceso de recuperación temática puede llevarse a cabo mediante dos pasos básicos. Primero, formulando consultas relevantes con respecto a dicho contexto temático. Segundo, presentando dichas consultas a un motor de búsqueda para recuperar documentos relevantes. A partir de estos dos pasos, es razonable que la calidad del material recuperado sea altamente dependiente de las consultas que se presentan. Este aspecto hace que la generación inteligente de consultas sea un problema de investigación importante para el área de minería de texto. En este contexto, proponemos el uso de una arquitectura que incorpore un ciclo evolutivo para permitir la recuperación de documentos relevantes con respecto a un tópico de interés. Los algoritmos evolutivos han demostrado su capacidad en el área de minería de texto, tanto para tareas de clasificación, como para tareas de recuperación temática y recomendación.

A diferencia de los esquemas anteriores, nuestra propuesta consiste en el uso de algoritmos evolutivos para evolucionar conjuntos de consultas que logren simultáneamente dos cosas: la selección de las palabras más relevantes con respecto al tópico de interés y el aprendizaje de nuevos e importantes términos no contenidos en el contexto original. Este último requisito, conjuntamente con las medidas utilizadas puede permitirnos la obtención de material novedoso de tal forma que enriquezca el vocabulario original sin salirse de contexto. La versatilidad de la propuesta permite que sea útil en importantes campos de aplicación, por ejemplo: *búsqueda basada en la tarea del usuario* [BHB01, LBMW00], *sistemas de recuperación para portales web* [CvdBD99, MPS04], *acceso a la web oculta* [KSS97, NZC05], *recopilación de información persistente* [SH04] y *soporte en gestión del conocimiento* [LMR$^+$03, MLR05].

1.3 Revisión sobre algoritmos evolutivos propuestos

1.3.1 Algoritmos evolutivos desarrollados en minería de datos

Selección de Características (FS) es el nombre común utilizado para todos los métodos que seleccionan o reducen el conjunto de variables o características para describir alguna situación o actividad en un conjunto de datos. Algunos autores distinguen *variables* de *características*, sin embargo, variables, características y descriptores serán tratados sin distinciones en esta disertación.

Los métodos de FS pueden ser aplicados de dos formas principales, en base a si las variables son evaluadas en forma individual o global. De acuerdo a la primera, se trabaja categorizando cada variable de forma aislada, por ejemplo, de acuerdo a su poder descriptivo individual. Sin embargo, una variable que es poco útil por sí sola, podría ser mejor si se la considerara en conjunto con otras [GE03]. Por esta razón, se logran modelos de aprendizaje más poderosos cuando el modelo de FS selecciona conjuntos de variables que conjuntamente tienen buena capacidad descriptiva.

Si aplicamos una división más sutil a estos últimos tipos de métodos de FS, podremos ver que suelen distinguirse entre filtros, wrappers y métodos integrados. Cuando las variables se seleccionan de acuerdo a características de los datos (p. ej. baja varianza o variables correlacionadas) se trata de un método de FS tipo filtro. Los wrappers utilizan una técnica de aprendizaje automatizado como caja negra, como paso de preprocesamiento para notar conjuntos de variables en términos de su habilidad predictiva. Finalmente, los métodos embebidos llevan a cabo el proceso de FS en la etapa de entrenamiento de un método de aprendizaje automatizado y usualmente se los adapta al método de aprendizaje aplicado [GE03, DGWC07]. Los algoritmos desarrollados a lo largo de esta investigación caen en la segunda categoría.

Un método de FS basado en wrappers, en general, consiste de dos partes principales: la de *evaluación del subconjunto de características*, la cual a su vez puede ser un método de aprendizaje (ya sea para regresión o para clasificación) y la de *búsqueda de subconjuntos de características*, la cual selecciona las variables a ser evaluadas por la función de evaluación. A su vez, los resultados del método de evaluación de subconjuntos de características se utilizan para guiar el proceso de búsqueda de subconjuntos. En consecuencia, el proceso de selección está estrechamente vinculado con el algoritmo de aprendizaje utilizado para la evaluación, tanto con respecto a la calidad de las selecciones que se obtienen como con respecto al tiempo de ejecución invertido. Por ejemplo se pueden obtener diferentes

combinaciones de calidad y tiempo de ejecución si usamos modelos de regresión lineales que si usamos modelos de regresión no lineales [DGWC07].
Existen algunos trabajos en los que se han utilizado algoritmos genéticos para evaluar distintas funciones de regresión o evaluación de subconjuntos de descriptores [LWDP01, DGWC07, LLYW03]. Algunos trabajos, han usado redes neuronales como función de evaluación logrando exitosos resultados [DGWC07, SK96, FTCC05]. Sin embargo, si alguna de estas técnicas se utiliza para desarrollar el método completo de FS, tienen la desventaja de consumir una importante cantidad de tiempo [GOCS06]. En particular, si la cantidad de posibles combinaciones es muy grande, el tiempo de ejecución será prohibitivo. Por un lado, los algoritmos evolutivos son una técnica iterativa, la cual, para este problema particular, exigirá un gran número de generaciones para alcanzar una buena precisión, y a su vez, en cada generación se deberá utilizar un método de regresión para la evaluación de cada subconjunto seleccionado. Por otro lado, las redes neuronales consumirán mucho tiempo en el entrenamiento cuando se encuentren frente a un gran conjunto de descriptores. Debido a esto, nuestra propuesta consiste en un esquema híbrido de dos etapas, de tal forma que para la búsqueda gruesa se aprovecha la velocidad de los algoritmos evolutivos en la búsqueda y optimización, combinada con funciones de regresión conocidas para guiar la búsqueda, en tanto que para la segunda etapa, una vez disminuido el número de descriptores, se utilizan redes neuronales para alcanzar un proceso de refinado más sutil.
Durante esta investigación los algoritmos evolutivos se desarrollaron de forma gradual en complejidad. Como un paso inicial, el primer esquema evolutivo asume como conocido el número aproximado de descriptores necesarios para construir el modelo y sólo intenta optimizar el error de predicción obtenido por los subconjuntos de descriptores seleccionados. Para la evaluación del error de predicción, la función objetivo implementada utiliza cuatro tipos de métodos de predicción. El primero basado en árboles de decisión, el segundo en el esquema de los k vecinos más cercanos, el tercero en regresión lineal y el cuarto en regresión no lineal. Finalmente, se abordó un esquema evolutivo multi-Objetivo que incorporó la minimización del número de descriptores seleccionados como un segundo objetivo. Dentro de este enfoque se estudiaron distintos esquemas basados en técnicas Pareto de conocida eficacia y eficiencia, y se propuso un enfoque agregativo que logró resultados igualmente exitosos.
Los algoritmos implementados se incorporaron como primera fase de una arquitectura que implementa ambas fases. Para la evaluación del comportamiento de la arquitectura

se tomó el problema de análisis de las relaciones cuantitativas estructura-propiedad y estructura-actividad en compuestos químicos. En la industria farmacéutica, estos tipos de análisis son sumamente importantes para el diseño de medicamentos. Una compañía farmacéutica puede tener hasta 100 investigadores trabajando en el proyecto de diseño de una droga, el cual puede tomar de 2 a 10 años para llegar al punto de pruebas clínicas y sobre animales. Aun con todos los recursos disponibles, las compañías farmacéuticas más exitosas logran llevar al mercado uno de cada diez proyectos de los que inician. Algunas de estas bajas se deben a que el esfuerzo del desarrollo podría nunca cubrirse con las ventas. En otros casos, dejando de lado el mercadeo, los compuestos activos terminan resultando tóxicos, no disponibles o extremadamente costosos de fabricar. Los sistemas biológicos son probablemente los sistemas bajo estudio más complejos del planeta. Las drogas generalmente no están compuestas por moléculas simples, la mayoría están constituidos por ciclos cerrados compuestos de diferentes tipos de átomos (son heterocíclicos), de peso molecular moderado y contienen múltiples grupos funcionales. Por esta razón, los desafíos de la síntesis orgánica son tan grandes como los desafíos para hallar los compuestos que deben ser sintetizados.

Las estimaciones acerca del costo de lograr que una droga salga al mercado han reportado un rango que va desde los U$D 300 millones a los U$D 1.7 billones. Dentro de este gasto, los ensayos por computadora (o *in silico*) son los menos costosos, abarcando una mínima parte del costo total. Estas técnicas proveen otras opciones para el entendimiento de los sistemas químicos, conduciendo a los expertos a información que es difícil e incluso imposible de obtener por medio de análisis en laboratorio.

En este proceso de experimentación *in sílico*, no existe una *mejor* técnica computacional. Se usan diferentes técnicas en diferentes etapas. Al principio del proyecto, se emplean métodos quemioinformáticos para la selección de compuestos de fuentes disponibles para ser analizados. A medida que se van identificando grandes colecciones de compuestos, los químicos se valen de distintas técnicas cada vez más específicas para realizar el análisis requerido. Entre los ejemplos más contundentes e importantes, se encuentran los métodos empleados para realizar el análisis de relaciones cuantitativas estructura-propiedad y el de relaciones cuantitativas estructura-actividad, denominados comúnmente como QSPR y QSAR, por sus siglas en inglés de Quantitative Structure-Property Relationships y Quantitative Structure-Activity Relationships. Por medio de este tipo de análisis se pueden correlacionar parámetros de estructura química (conocidos como descriptores) con determinadas propiedades o actividades biológicas (p. ej. hidrofobicidad). Este tipo de análisis

constituye un problema combinatorial duro que requiere la evaluación de *relaciones complejas* y, por lo tanto, el uso de modelos complejos, para lograr determinar la relevancia de los subconjuntos seleccionados. Los algoritmos desarrollados durante la tesis se probaron con varias bases de datos reales concernientes a esta aplicación bioinformática.

1.3.2 Algoritmos evolutivos desarrollados en minería de texto

La reformulación y extensión de consultas ha sido reconocida como una tarea importante en los sistemas de recuperación de información. En particular los usuarios presentan cierta dificultad para articular sus necesidades de información. Por un lado, esto se debe a que las mejores palabras para caracterizar cierto tema no son obvias. Por otro lado, también se debe a la tendencia natural de las personas a ofrecer pocas palabras en su descripción de búsqueda [LB08]. Además, si esta tarea logra realizarse de una manera totalmente automática muchas aplicaciones de recuperación de información y sistemas de conocimiento pueden favorecerse recuperando material relevante para si mismos.
La reformulación o extensión de consultas se logra mediante técnicas de recuperación de información que utilizan documentos a partir de los cuales pueden obtenerse nuevos términos. La tarea de agregar términos a una consulta puede hacerse de manera manual, automática o asistida por el usuario. Si es manual, el usuario agrega términos a la consulta inicial. Mientras que es automática, por ejemplo, cuando un algoritmo calcula los pesos de todos los términos en los resultados más relevantes obtenidos y agrega a la consulta inicial aquellas palabras que obtuvieron mayor puntaje. Por último, se trata de una extensión asistida por el usuario cuando este selecciona cuáles términos serán usados para la expansión. Esta selección se realiza sobre una lista que el sistema previamente calcula y le presenta. Existen algunos sistemas que soportan refinamiento de consultas y muchos estudios han demostrado los beneficios de este tipo de mecanismos de reformulación [Kli01, Chu02, RKC$^+$07]. Sin embargo, la mayoría de estos sistemas proveen una interfaz de navegación que requiere la intervención explícita de los usuarios. Existen situaciones en las que se puede contar con información útil para guiar la reformulación. Los algoritmos evolutivos para recuperación temática desarrollados en esta tesis, están pensados para aprovechar tal información a fin de producir consultas que, al ser presentadas a motores de búsqueda o índices, obtengan resultados relevantes para el tópico de interés. Teniendo en cuenta que un contexto temático puede contener una enorme cantidad de términos y que los motores de búsqueda convencionales suelen limitar la longitud de consulta, es sumamente importante realizar una selección útil de los términos.

Si pensamos en cada consulta como un vector en el cual cada palabra representa una dimensión particular, podemos notar que estamos tratando con un *espacio altamente dimensional*. Esto se ve agravado por el hecho de que dichas consultas tendrán como objetivo la *exploración de grandes corpus* de documentos. Es decir, cada conjunto de palabras que se genere será presentado a una interfaz de búsqueda, la cual retornará los documentos recuperados por dicha consulta, y es en base a este conjunto de documentos que será calificada. Por otro lado, se sabe que en recuperación de información *resultados que no sean precisamente los óptimos*, pueden ser altamente calificados e importantes. En este sentido, también cabe destacar que pueden existir numerosos conjuntos de resultados relevantes (documentos relevantes), es decir varias consultas pueden permitirnos arribar a *distintos conjuntos de documentos pero similarmente relevantes*. Por último, debe tenerse en cuenta que para poder encontrar buenas combinaciones de términos, será necesario *explorar* el corpus en distintas direcciones y *explotar* las regiones prometedoras. Todas estas características indican que los algoritmos evolutivos constituyen una herramienta sumamente adecuada para la solución del problema.
El problema de encontrar un conjunto de términos de forma automática, de tal forma que por medio de ellos podamos recuperar material relevante con respecto a un tema de interés, puede pensarse como un problema de optimización. En todo problema de optimización debemos definir:

- El *espacio búsqueda*, el cual en este caso estará constituido por todas las posibles combinaciones de palabras incluyendo posiblemente palabras repetidas, ya que las palabras repetidas podrían indicar un mayor nivel de importancia para dichos términos.
- El *espacio objetivo*, el cual dependerá de lo que estamos intentando obtener y estará definido por la función objetivo.
- La *función objetivo*, la cual nos permitirá mapear cada elemento del espacio de búsqueda en un vector del espacio de decisión. Por ejemplo, en este problema podemos querer recuperar documentos similares al tópico en cuanto a frecuencia de palabras, o conjuntos de documentos que abarquen el máximo número posible de documentos relevantes y simultáneamente mínima cantidad de documentos no relevantes. La función objetivo será un reflejo de estas aspiraciones y nos permitirá determinar que tan bueno es un subconjunto de palabras para alcanzarlas.

Esta tesis propone el uso de una infraestructura evolutiva compuesta por una *representación interna del tópico* de interés, un mecanismo de *generación de consultas iniciales*,

un *ciclo evolutivo* (conformado por una población de consultas, los operadores genéticos, el proceso de selección y el módulo para evaluación del fitness), un *reservorio de vocabulario nuevo*, la herramienta para extracción de términos de dicho reservorio y un *motor de búsqueda* como medio de comunicación entre el sistema y la colección de documentos. Esta infraestructura y los algoritmos evolutivos desarrollados se explican en detalle en el capítulo 6.

De forma similar a los algoritmos evolutivos desarrollados en esta tesis para minería de datos, hemos desarrollado los algoritmos evolutivos para minería de texto en forma gradual, comenzando por una versión mono-objetivo destinada a la evaluación del comportamiento de los algoritmos evolutivos para este problema de recuperación en especial. En particular se investigó como podemos evolucionar consultas cuando intentamos recuperar material *similar* al tópico de interés, se propuso una metodología para evaluar dichas consultas con el fin de poder ordenarlas de acuerdo a un orden de importancia, se analizó si los resultados obtenidos por medio del algoritmo evolutivo fueron estadísticamente mejores que los iniciales y se estudió como afectan diferentes tasas de mutación (mutación nula, mutación normal e hipermutación) y la aplicación de elitismo al comportamiento del sistema. Además, dentro de esta implementación se propuso una nueva métrica basada en similitud por coseno, denominada *similitud novedosa* destinada a promover la obtención de documentos con términos novedosos relacionados con el tópico de interés. Las distintas implementaciones de la versión mono-objetivo, fueron utilizadas sobre la web para distintos tópicos, alcanzando resultados exitosos que demuestran la efectividad de la propuesta.

A partir de la primera versión del núcleo evolutivo de la arquitectura propuesta, en base al buen comportamiento logrado y como parte de la evolución natural de dicha infraestructura, la segunda instancia está orientada a cumplir el objetivo general de recuperar material temáticamente relacionado con un tópico de interés procurando recuperar la mayor cantidad posible de material relacionado y la menor cantidad posible de material no relacionado. Cuando nos basamos sólo en la métrica de *similitud* para determinar la efectividad de una consulta, no evaluamos cuántos de los *documentos relevantes* existentes estamos recuperando ni tampoco cuántos *documentos no relevantes* recuperamos. Para evaluar estos aspectos utilizamos las conocidas métricas de *precisión* (precision) y *cobertura* (recall) [BYRN99], intentando maximizar ambas medidas. Las definiciones de las métricas precisión y cobertura exigen el conocimiento total del corpus de documentos, por lo tanto, a diferencia del esquema evolutivo cuyo único objetivo es similitud, un es-

quema que considere estas dos métricas no puede ser directamente utilizado sobre la web. Por esta razón, la metodología propuesta consiste en el entrenamiento del algoritmo sobre un corpus de documentos y el testeo de los resultados en un corpus diferente, asegurando de esta manera, la generalidad de las consultas evolucionadas. Durante la implementación de esta segunda versión se estudiaron los siguientes aspectos principales:

- las posibles formas de evolucionar y establecer un rango a las consultas cuando se persiguen múltiples objetivos,
- las diferencias entre observar varios objetivos simultánea e independientemente uno de otro en un esquema Pareto y mirarlos conjuntamente dentro de un esquema agregativo,
- la adaptación de métricas conocidas de recuperación de información (IR) para ser contempladas dentro de los objetivos,
- la mejora de los resultados obtenidos luego de la evolución con respecto a los resultados alcanzados por las consultas iniciales y con respecto a métodos similares usados en IR, finalmente,
- la utilidad de las consultas generadas en la etapa de entrenamiento cuando se las aplica más allá de dicha etapa. Es decir, se analizó si las mejores consultas obtenidas por el algoritmo evolutivo son efectivas cuando se las testea para el mismo tópico pero dentro de un nuevo corpus de documentos.

Para la evaluación de la arquitectura con los distintos algoritmos evolutivos multi-objetivo, recolectamos más de 400 tópicos del directorio DMOZ (Open Directory Proyect - ODP), alcanzando más de 350.000 páginas, las cuales fueron separadas adecuadamente para poder llevar a cabo el entrenamiento del algoritmo y el posterior testeo de los resultados.

1.4 Estructura de la tesis

El capítulo 1 permite comprender la ubicación de los algoritmos e infraestructuras desarrollados en esta tesis dentro del área de minería de datos y texto. Ofrece una revisión de los objetivos propuestos y estudiados dentro de cada una de las dos ramas, así como los alcances de los problemas abordados. Finalmente, presenta un resumen general de los algoritmos evolutivos propuestos y desarrollados.
El capítulo 2 presenta una breve revisión de los conceptos de optimización mono- y multi-objetivo básicos utilizados durante la investigación.

En los capítulos 3 y 4 se realiza una revisión de las definiciones y fundamentos básicos sobre algoritmos evolutivos, y se resumen las técnicas evolutivas multi-objetivo más utilizadas dando un muestreo de su complejidad, tanto en tiempo de ejecución como en implementación, a partir del cual se fundamenta el uso de las mismas dentro de las arquitecturas desarrolladas.
En el capítulo 5, se explican en detalle los algoritmos evolutivos propuestos e implementados para minería de datos dentro del problema de selección de características, la arquitectura de dos fases propuesta para su resolución, los casos de estudio abordados en el problema de análisis de relaciones cuantitativas estructura-propiedad y relaciones cuantitativas estructura-actividad (QSPR y QSAR), y los resultados obtenidos por la metodología.
En el capítulo 6 se presentan detalladamente la arquitectura propuesta e implementada para búsqueda temática en minería de texto y los distintos algoritmos evolutivos mono y multi-objetivo contemplados. A medida que se explican las distintas versiones evolutivas, se presentan las métricas de evaluación propuestas y los resultados obtenidos.
Finalmente, el capítulo 7 expone las conclusiones alcanzadas mediante esta investigación y se perfilan posibles lineamientos de trabajo futuro.

Capítulo 2

Conceptos básicos de optimización

Los métodos desarrollados para minería en esta tesis están fuertemente orientados hacia un problema tradicional más general, el de la optimización combinatoria. El concepto de *optimización* refiere a encontrar una o más soluciones factibles respecto de valores extremos correspondientes a uno o más objetivos. Una parte importante de la investigación en este área contempla problemas donde la calidad de las soluciones se evalúa en términos de un único objetivo. Sin embargo, la mayoría de los problemas de optimización del mundo real involucran, en general, varios objetivos. Esta última instancia conlleva a lo que se conoce como *Optimización Multi-Objetivo*. Además de la complejidad asociada a la optimización en sí, es natural que estos múltiples objetivos estén en conflicto entre sí, lo cual hace de la optimización multi-objetivo un desafío sumamente interesante y complejo. Durante esta investigación se afrontaron ambos niveles de complejidad dentro de las áreas de investigación específicas de *selección de características* y *recuperación temática*. En este sentido, los aportes principales de la tesis conciernen al planteo y resolución de ambos problemas por medio del diseño de herramientas inteligentes basadas en computación evolutiva. Los enfoques mono-objetivo contemplados, apuntaron principalmente a la evaluación del funcionamiento de la metodología planteada cuando se la aplicaba a versiones más simples de cada problema. Por otra parte, los enfoques multi-objetivo desarrollados, tuvieron como finalidad el análisis de las metodologías sobre los problemas multi-objetivo que se abordaron en esta investigación, los cuales caen dentro del grupo de problemas reales en los cuales hay múltiples objetivos contrapuestos. Por ejemplo, en el caso de *selección de características*, tener menor cantidad de descriptores seleccionados puede aumentar las posibilidades de que la capacidad descriptiva del subconjunto sea más pobre. En el caso de *recuperación temática*, una mejor cobertura puede conducir a peor precisión y viceversa. Cuando estamos tratando con este tipo de problemas, suele no haber una única

solución que pueda llamarse "óptima" de acuerdo a los múltiples objetivos contrapuestos contemplados, por lo que la optimización multi-objetivo puede conducirnos a *varias* soluciones "óptimas" que sufren una suerte de negociación entre los objetivos involucrados. En este capítulo veremos brevemente los conceptos y definiciones fundamentales sobre optimización mono y multi-objetivo, que serán empleados como base de los algoritmos evolutivos desarrollados durante esta tesis.

2.1 Optimización mono y multi-objetivo

Cuando un problema involucra un único objetivo, la tarea de encontrar una o más soluciones óptimas se denomina *optimización mono-objetivo* (mono-O). Mientras que, cuando el problema comprende más de un objetivo, la tarea de encontrar una o más soluciones óptimas se denomina *optimización multi-objetivo* (multi-O). La mayoría de los problemas de optimización y búsqueda reales implica múltiples objetivos. En este caso, el clásico principio de quedarnos con la solución extrema (en el sentido de "lo mejor"), utilizado para resolver los problemas mono-objetivo, no puede ser aplicado a uno de los múltiples objetivos, debido a que el resto de los objetivos son también importantes. Generalmente, una solución que es mejor con respecto a uno de los objetivos implicará un compromiso en otros. Existen muchos algoritmos destinados a optimización mono y multi-objetivo. Dado que los primeros parecen ser un caso particular de los segundos, algunas estrategias consisten en transformar los múltiples objetivos requeridos por el problema multi-objetivo en una función que los agregue de manera adecuada, en tanto que otras técnicas optimizan los múltiples objetivos en forma simultánea pero independiente unos de otros. Durante esta investigación se emplearon los dos tipos de enfoques. Sin embargo, cuando los múltiples objetivos fueron agregados dentro de una única función, el análisis final se hizo sobre los objetivos originales en forma separada.

En problemas multi-O, siempre que los objetivos sean contrapuestos, usualmente no habrá una solución que sea mejor que todas en todos los objetivos. Sin información extra, no podemos decidir cuál solución es la mejor. Por otra parte, suministrar tal información no es una tarea trivial en la mayoría de los problemas de optimización. En los problemas multi-O se nos presentan *un conjunto* de soluciones óptimas, muchas de las cuales son importantes. En tanto que en problemas mono-O hay una única solución óptima. En ambas instancias de optimización tendremos presentes los siguientes conceptos generales:

Variables de decisión. Las variables de decisión son cantidades numéricas cuyos valores deben ser elegidos en un problema de optimización. Estas cantidades se denotan como x_j, $j = 1, 2, \ldots, n$. El vector $\mathbf{x}$ de n variables de decisión se representa de la siguiente manera:

$$\mathbf{x} = [\ x_1,\ \ x_2,\ \ \cdots,\ \ x_n\ ]^T \tag{2.1}$$

Vemos que este vector pertenece al espacio euclidiano *n-dimensional*, ya que cada una de las variables de decisión conforma un eje coordenado.

Restricciones. En la mayoría de los problemas de optimización, las características particulares del entorno o los recursos disponibles (por ejemplo tiempo, espacio, etc.), imponen ciertas restricciones. Para que una solución sea considerada válida o aceptable debe cumplir con dichas restricciones. Las restricciones describen dependencias entre las variables de decisión y los parámetros específicos del problema y se escriben en forma de desigualdades matemáticas como:

$$g_i(\mathbf{x}) \leq 0 \qquad i = 1, \ldots, m \tag{2.2}$$

o en la forma de igualdades como:

$$h_j(\mathbf{x}) = 0 \qquad j = 1, \ldots, p \tag{2.3}$$

donde $p \leq n$ ya que de otro modo el problema no tendría grados de libertad.
Estas restricciones se presentarán en forma *explícita*, es decir dispondremos de una fórmula para realizar el cálculo o se presentarán en forma *implícita*, en cuyo caso se contará con un algoritmo que nos facilite dichos cálculos.

Función objetivo. Las funciones objetivo se expresan como: $f_1(x), f_2(x), \ldots, f_k(x)$, donde k es el número de funciones objetivo que se desea resolver dentro del problema. Claramente, el conjunto de funciones objetivo forman un vector *k-dimensional* como el siguiente:

$$\mathbf{f}(\mathbf{x}) = [\ f_1(\mathbf{x}),\ \ f_2(\mathbf{x}),\ \ \cdots,\ \ f_k(\mathbf{x})\ ]^T \tag{2.4}$$

Para completar los conceptos de optimización mono y multi-objetivo definiremos las bases de los problemas que se resuelven mediante cada uno de los enfoques.

2.2 Problema de optimización mono-objetivo

Este es el tipo de problema de optimización clásico ampliamente estudiado en la literatura y una de las definiciones que podemos encontrar es la siguiente:

Definición 1 (**Problema de optimización mono-objetivo (mono-OOP)** [CLV07])**.** *Un problema de optimización mono-objetivo se define como minimizar (o maximizar)* $f(\mathbf{x})$, $\mathbf{x} \in \Omega$, *sujeto a:*

$$g_i(\mathbf{x}) \leq 0 \qquad i = 1, \ldots, m \tag{2.5}$$

y

$$h_j(\mathbf{x}) = 0 \qquad j = 1, \ldots, p \tag{2.6}$$

En este caso, una solución minimiza (o maximiza) el escalar $f(\mathbf{x})$ *donde* $\mathbf{x}$ *es un vector n-dimensional de variables de decisión,* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ *de algún universo* Ω.

Es decir, $g_i(\mathbf{x}) \leq 0$ y $h_j(\mathbf{x}) = 0$ representan restricciones que deben lograrse al mismo tiempo que se optimiza $f(\mathbf{x})$. Por otro lado, Ω contendrá todos los posibles $\mathbf{x}$ que puedan ser utilizados para satisfacer una evaluación de $f(\mathbf{x})$ y las correspondientes restricciones. El método para encontrar el óptimo global de una función se denomina **Optimización Global**. Sin pérdida de generalidad, podemos decir que esta optimización será una minimización y su correspondiente definición suele ser la siguiente:

Definición 2 (**Optimización mono-objetivo de mínimo global** [Bäc96])**.** *Dada una función* $f : \Omega \subseteq \mathbb{R}^n \to \mathbb{R}$, $\Omega \neq \emptyset$, *para* $\mathbf{x} \in \Omega$, *el valor* $f^* \triangleq f(\mathbf{x}^*) > -\infty$ *es un* **mínimo global** *si y sólo si*

$$\forall \quad \mathbf{x} \in \Omega : f(\mathbf{x}^*) \leq f(\mathbf{x}) \tag{2.7}$$

donde $\mathbf{x}^*$ *es, por definición, una solución mínima global, f es la función objetivo, y el conjunto* Ω *es la región factible para* $\mathbf{x}$. *La tarea de determinar la solución(es) mínima(s) global(es) se denomina* **problema de optimización global**, *y en este caso será para un problema mono-objetivo.* □

2.3 Problema de optimización multi-objetivo

El problema de optimización multi-objetivo (multi-OOP) puede pensarse informalmente como el problema de encontrar un vector de variables de decisión que satisfaga

las restricciones y optimice una función vectorial cuyos elementos representan a las funciones objetivo. Estas funciones forman una descripción matemática de distintos criterios de performance que, usualmente, estarán en conflicto unos con otros. Por lo tanto, el término *optimizar* significa encontrar una solución tal que daría valores aceptables para el tomador de decisiones (DM por Decision Maker) para todas las funciones objetivos [Osy85].

Los problemas multi-objetivo son aquellos donde la meta es optimizar k funciones objetivo en forma simultánea. Esto implica *minimizar* las k funciones, *maximizar* las k funciones o minimizar algunas al mismo tiempo que se maximizan las demás. Una definición formal general para problemas de optimización multi-objetivo es la que ofrecen Coello y Veldhuizen:

Definición 3 (**Problema de optimización multi-objetivo general (multi-OOP)** [Vel99, CLV07])**.** *Un problema de optimización multi-objetivo general se define como la tarea de minimizar (o maximizar):*

$$F(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \tag{2.8}$$

$\mathbf{x} \in \Omega$ *sujeto a:*

$$g_i(\mathbf{x}) \leq 0 \qquad i = 1, \ldots, m \tag{2.9}$$

y

$$h_j(\mathbf{x}) = 0 \qquad j = 1, \ldots, p \tag{2.10}$$

Una solución para el multi-OOP minimiza (o maximiza) los componentes del vector $F(\mathbf{x})$, donde $\mathbf{x}$ es un vector n-dimensional de variables de decisión, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ de algún universo Ω. Observemos que $g_i(\mathbf{x}) \leq 0$ y $h_j(\mathbf{x}) = 0$ representan restricciones que deben ser cometidas en forma simultánea con la minimización (maximización) de $F(\mathbf{x})$ y Ω contiene todos los posibles $\mathbf{x}$ que pueden ser usados para satisfacer $F(\mathbf{x})$.

Según Deb [Deb04] puede existir una tercer restricción, en base a la cual, para cada variable de decisión i existe un límite establecido por el problema tal que $x_i^{(L)} \leq x_i \leq x_i^{(U)}$. Estos límites conforman lo que se denomina *espacio de las variables de decisión, D*, o simplemente espacio de decisión. Se incorporan aquí los conceptos de *solución factible* y *solución no factible*, ya que una solución $\mathbf{x}$ que no cumple con las $(m + p)$ restricciones funcionales (correspondientes a g's y h's), ni con las $2n$ restricciones de límite, se denomina *solución no factible*. Mientras que, si $\mathbf{x}$ es una solución que sí cumple dichas restricciones, entonces es una *solución factible*. A partir de esto se puede deducir que el conjunto de todas soluciones factibles dentro del espacio de decisión conformará la *región factible*, o

S. Teniendo en cuenta esto, si denominamos Z al espacio multidimensional constituido por las funciones objetivo, para cada vector $\mathbf{x}$ en D existe un punto $\mathbf{z}$ en Z tal que: $f(\mathbf{x}) = \mathbf{z} = (z_1, z_2, \ldots, z_k)$. Este mapeo tiene lugar de un espacio *n-dimensional* a un espacio *k-dimensional* como se muestra en la figura 2.1.

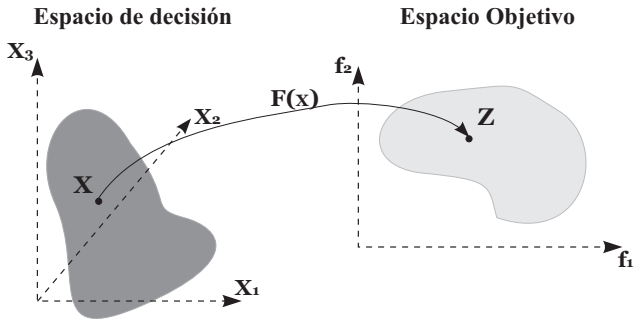

Figura 2.1: Representación del mapeo entre un espacio de decisión *3-dimensional* y un espacio objetivo *2-dimensional* llevado a cabo por $F(\mathbf{x})$.

Multi-OOP lineal vs. multi-OOP no lineal. Si todas las funciones objetivo y todas las funciones que establecen las restricciones son lineales, el problema se denomina *Problema Multi-Objetivo Lineal* (MOLP). Este tipo de problemas tienen propiedades que facilitan las pruebas de convergencia para las técnicas que intentan resolverlos. En cambio, si alguna de las funciones es no lineal, estamos ante un *Problema Multi-Objetivo No Lineal* (MONLP). Desafortunadamente, la mayoría de los problemas reales son MONLP, por lo que las técnicas que intentan solucionarlos, en general no tienen pruebas de convergencia [Deb04].

2.3.1 Concepto de dominancia

En el contexto de problemas mono-objetivo el concepto de dominación se resume en decir que la solución $\mathbf{x}_1$ 'domina' a o 'es mejor que' $\mathbf{x}_2$ si $f(\mathbf{x}_1) < f(\mathbf{x}_2)$, cuando se busca minimizar f, o si $f(\mathbf{x}_1) > f(\mathbf{x}_2)$ cuando se busca maximizar f. Es decir, podemos ver fácilmente cual de las dos soluciones es la mejor, mirando cual cumple mejor el objetivo. En cambio, cuando se trata de problemas multi-objetivo no existe un "orden completo". La noción de óptimo cambia cuando hay varios objetivos que a su vez se contraponen. Muchas nociones de optimalidad en multi-OOP fueron generalizadas por Vilfredo Pareto, un controvertido sociólogo y economista italiano, que conformó estas nociones dentro de un conjunto de teorías socio-económicas conocidas como *sistema Paretiano de equilibrio*

general. En éstas, Pareto afirmaba que la sociedad llegaba al límite de su bienestar cuando no podían lograrse mejoras en algún punto sin empeorarse simultáneamente otros [Par96]. De aquí surge el nombre *óptimo de Pareto* por el cual es conocida la noción de "óptimo" cuya definición formal veremos más adelante en esta sección.

Definición 4 (**Dominancia de Pareto** [Coe02]). *Se dice que un vector* $\mathbf{u} = (u_1, u_2, \ldots, u_k)$ *domina a otro vector* $\mathbf{v} = (v_1, v_2, \ldots, v_k)$ *y se nota '*$\mathbf{u} \preceq \mathbf{v}$*' si y sólo si* $\forall i \in \{1, \ldots, k\}$, $u_i \leq v_i$, *y además* $\exists i \in \{1, \ldots, k\} : u_i < v_i$, *(lo cual equivale a decir que* $\mathbf{u}$ *es parcialmente menor que* $\mathbf{v}$*).*

A partir de esta definición, es intuitivo pensar en el concepto de *no dominación*, el cual es sumamente utilizado por muchos de los algoritmos evolutivos basados en Pareto.

Definición 5 (**No dominancia de Pareto**). *Se dice que un vector* $\mathbf{u} = (u_1, u_2, \ldots, u_k) \in \Omega$ *es no dominado sobre la región* Ω*, si no existe otro vector* $\mathbf{v} = (v_1, v_2, \ldots, v_k) \in \Omega$ *tal que* $\mathbf{v} \preceq \mathbf{u}$.

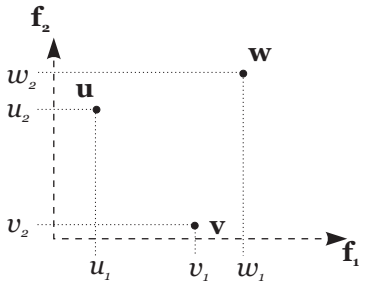

Figura 2.2: Ejemplo de dominancia: $\mathbf{u} \preceq \mathbf{w}$ y $\mathbf{v} \preceq \mathbf{w}$, pero $\mathbf{u} \npreceq \mathbf{v}$ y $\mathbf{v} \npreceq \mathbf{u}$.

La figura 2.2 ilustra la definición anterior. Podemos ver que $\mathbf{u} \preceq \mathbf{w}$ dado que: $u_1 < w_1$ y $u_2 < w_2$. También $\mathbf{v} \preceq \mathbf{w}$ pues se verifica que: $v_1 < w_1$ y $v_2 < w_2$. Por otro lado, $\mathbf{u} \npreceq \mathbf{v}$ ya que la relación "$<$" no se verifica para todas las componentes del vector $\mathbf{u}$ respecto de las correspondientes componentes del vector $\mathbf{v}$.

Propiedades de la relación dominancia

A partir de la definición 4 pueden discutirse algunas propiedades referentes a la relación binaria *dominancia de Pareto*.

Reflexiva. La relación de *dominancia* no es reflexiva, ya que cualquier solución no

se puede dominar a sí misma. Esto ocurre debido a que no se verifica que $\exists\ i \in \{1,\ldots,k\} : u_i < u_i$. La relación de *dominancia* es irreflexiva.

Simétrica. La relación de *dominancia* no es simétrica ya que el hecho de que p domine a q, no implica que q domine a p. De hecho se verifica la siguiente propiedad.

Asimétrica. Partiendo de la definición, podemos ver que la relación de *dominancia* es asimétrica, dado que si $p \preceq q$ entonces $q \npreceq p$.

Antisimétrica. Esta propiedad establece que si $p \preceq q$ y $q \preceq p$ entonces $p = q$. Pero como vimos antes, si $p \preceq q$ entonces $q \npreceq p$, es decir, no existe ningún par de elementos p y q tal que $p \preceq q$ y $q \preceq p$, por lo que podemos decir que la relación de dominancia es (vacuamente) antisimétrica.

Transitiva. La relación de *dominancia* es transitiva ya que si $p \preceq q$ y $q \preceq r$ entonces $p \preceq r$.

Debido a las condiciones anteriores, la relación de dominancia califica como una relación de orden, más precisamente, de *orden parcial estricto*. Esto es porque es irreflexiva, asimétrica y transitiva [GHK$^+$03].

2.3.2 Optimalidad de Pareto

Debido a que en la rama de optimización que ataca los problemas multi-objetivo nos encontramos con *conjuntos de soluciones* más que con soluciones únicas, diversos autores como Coello y Deb han desarrollado definiciones que abarcan conjuntos de elementos.

Definición 6 (**Optimalidad de Pareto** [Coe02])**.** *Se dice que una solución* $\mathbf{x} \in \Omega$ *es* ***óptimo de Pareto*** *con respecto a* Ω*, si y sólo si* $\nexists\ \mathbf{x}^* \in \Omega$ *tal que* $\mathbf{v} = F(\mathbf{x}^*) = (f_1(\mathbf{x}^*),\ldots,f_k(\mathbf{x}^*))$ *domine a* $\mathbf{u} = F(\mathbf{x}) = (f_1(\mathbf{x}),\ldots,f_k(\mathbf{x}))$.

Notemos que la frase *óptimo de Pareto* refiere al espacio de decisión, ya que Ω es el posible universo en el que conviven las soluciones $\mathbf{x}$.

El concepto de óptimo de Pareto suele usarse en dos niveles de rigurosidad, en el nivel menos estricto se encuentra el *óptimo de Pareto Débil* y en un nivel más riguroso el *óptimo de Pareto Estricto.*

Definición 7 (**Óptimo de Pareto Débil**)**.** *Un punto* $\mathbf{x}^* \in \Omega$ *es* ***óptimo de Pareto Débil*** *si* $\nexists\ \mathbf{x} \in \Omega$ *tal que* $f_i(\mathbf{x}) < f_i(\mathbf{x}^*)$ *para* $i = 1,\ldots,k$.

Definición 8 (**Óptimo de Pareto Estricto**). *Un punto* $\mathbf{x}^* \in \Omega$ *es* ***óptimo de Pareto Estricto*** *si* $\nexists\ \mathbf{x} \in \Omega$, $\mathbf{x} \neq \mathbf{x}^*$, *tal que* $f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*)$ *para* $i = 1, \ldots, k$.

Definición 9 (**Conjunto óptimo de Pareto** [Coe02]). *El* ***Conjunto óptimo de Pareto****,* $\mathcal{P}^*$*, para un multi-OOP, cuya función es* $F(\mathbf{x})$ *se define como:*

$$\mathcal{P}^* = \{\ \mathbf{x} \in \Omega\ \ :\ \ \nexists\ \ \mathbf{x}^* \in \Omega\ \ \ y\ \ \ F(\mathbf{x}^*) \preceq F(\mathbf{x})\ \} \tag{2.11}$$

En palabras, este conjunto de soluciones es el conjunto de todas las posibles soluciones pertenecientes al espacio de decisión cuya evaluación para todas las funciones objetivo (un vector en el espacio objetivo) resulta no ser dominada por la evaluación de ninguna otra posible solución. Se trata de un conjunto que mira la dominancia a nivel global ya que toma todos los $\mathbf{x}$ en Ω. El conjunto de vectores en el espacio objetivo conformado por los vectores que resultan de evaluar cada solución perteneciente al conjunto óptimo de Pareto, se denomina **Frente de Pareto** y su definición es la siguiente:

Definición 10 (**Frente de Pareto**). *El* ***Frente de Pareto****,* $\mathcal{PF}^*$*, para un dado multi-OOP, cuya función es* $F(\mathbf{x})$ *y cuyo conjunto óptimo de Pareto es* $\mathcal{P}^*$*, se define como:*

$$\mathcal{PF}^* = \{\ \mathbf{u} = F(\mathrm{x})\ \ :\ \ \mathbf{x} \in \mathcal{P}^*\ \} \tag{2.12}$$

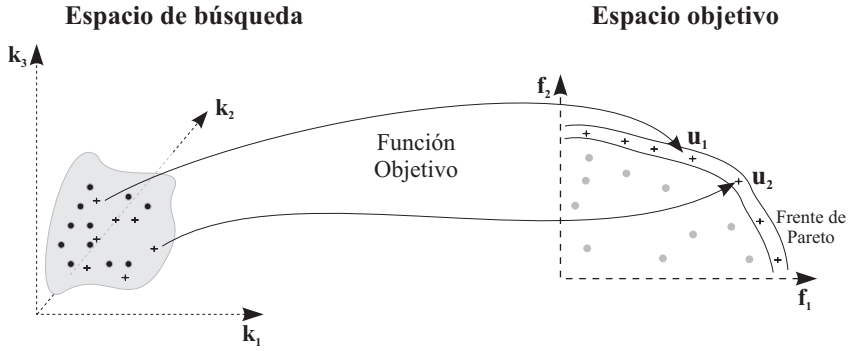

Figura 2.3: Ejemplo de frente de Pareto y conjunto óptimo de Pareto. Los vectores en el espacio de búsqueda cuya evaluación de la función objetivo dan como resultado un vector en el frente de Pareto constituyen el conjunto óptimo de Pareto. Ambos conjuntos se encuentran representados por cruces en cada espacio.

En base a la definición 10, los vectores *no dominados* del espacio objetivo se conocen en forma colectiva como el *frente de Pareto*. Cabe notar que $\mathcal{P}^*$ es un subconjunto que

contiene algunas de las posibles soluciones del *espacio de búsqueda*, en tanto que sus vectores objetivo (las evaluaciones sobre esas soluciones) forman $\mathcal{PF}^*$, un subconjunto del *espacio objetivo* en el cual cada vector es no dominado con respecto a todos los demás vectores producidos por las evaluaciones de cada solución en Ω.

La figura 2.3 muestra un ejemplo de *frente de Pareto*, el cual está formado por los resultados de la evaluación de la función objetivo sobre un subconjunto de soluciones y se encuentra en el *espacio objetivo* (representado por cruces). Las cruces en el *espacio de búsqueda* conforman el *conjunto óptimo de Pareto* de este ejemplo. Dicho conjunto está formado por soluciones cuya evaluación de la función objetivo conduce a un vector perteneciente al frente de Pareto en el espacio objetivo, mientras que las demás soluciones y sus correspondientes evaluaciones están representadas por puntos negros y grises respectivamente. Como ilustra la figura, la pertenencia o no al frente de Pareto no implica un orden o distribución para las correspondientes soluciones en el espacio de búsqueda.

Capítulo 3

Algoritmos evolutivos

Natural selection acts only by taking advantage of slight, successive variations. She can never take a great and sudden leap, but must advance by short and sure, though slow steps.[1]

Charles Darwin.

Algoritmos Evolutivos (EAs) es el término genérico empleado para referir al grupo de técnicas que simulan computacionalmente el proceso evolutivo natural. Dicha simulación se utiliza para resolver determinado problema a través de un proceso que se basa en los conceptos evolutivos de biología. En este capítulo veremos algunos conceptos básicos de biología, los cuales constituyen la base para la analogía que subyace bajo esta imitación. Además, resumiremos los conceptos y las técnicas principales que conciernen a los algoritmos evolutivos.

3.1 Teoría evolutiva básica (un poco de biología)

La ***evolución*** es un proceso de transformación en el cual las especies sufren cambios a lo largo de las generaciones. La teoría de la Evolución de Darwin [Dar59] ofrece una explicación a la diversidad biológica y sus mecanismos subyacentes. Veremos algunos de los conceptos fundamentales de forma absolutamente resumida para comprender mejor como se aplican en computación.

[1] *La selección natural trabaja beneficiándose solamente de variaciones sucesivas leves. Ella no puede dar un salto enorme y repentino, sino que debe avanzar sobre pasos cortos y seguros, aunque sean lentos.*

3.1.1 Supervivencia del más apto

La teoría de Darwin abarca dos aspectos principales. Por un lado, resume las evidencias en favor de que todos los organismos descendieron de un ancestro común encontradas por el científico. Por otro lado, defiende a la *selección natural* como un mecanismo para la evolución. Dado un entorno determinado de supervivencia y un conjunto de individuos o *población*, la selección natural favorece a los individuos que compiten de forma más efectiva por los recursos disponibles. Es decir, resultan favorecidos aquellos individuos que se adaptan o se ajustan mejor a las condiciones del entorno. Pero ¿qué significa que un individuo es *favorecido*? Significa que tendrá mayores probabilidades de sobrevivir y reproducirse. Este fenómeno se conoce como *supervivencia del más apto*. Es natural a partir de lo anterior, pensar en el individuo como la *unidad de selección*. Los *rasgos fenotípicos* son características físicas y de comportamiento del individuo que afectan en forma directa la respuesta del individuo al entorno (el cual incluye a los demás individuos) y, por lo tanto, determinan su capacidad de adaptación o ***fitness***. Cada individuo representa una única combinación de rasgos fenotípicos que el entorno somete a evaluación. Si la evaluación resulta favorable, entonces esta combinación será propagada por medio de los descendientes del individuo. En caso contrario, desaparecerá al morir el individuo sin descendientes.

La selección bajo condiciones de competencia es uno de los puntos claves del proceso evolutivo. Otro de los aspectos fundamentales surge a partir de las variaciones fenotípicas entre miembros de la población. Darwin descubrió que, durante el proceso de reproducción de generación en generación, ocurren pequeñas variaciones en los rasgos fenotípicos de los individuos, un fenómeno conocido como *mutación*. Estas ocasionales mutaciones permiten la aparición de nuevos individuos en la competencia. Así, a medida que el tiempo transcurre, se producen cambios constitutivos en la población y el proceso de evolución progresa. Por lo tanto, podemos ver a la población como la *unidad de evolución*.

La incorporación de nuevos conocimientos en genética poblacional, biología, paleontología y secuenciamiento de ADN en los estudios sobre la evolución, han conducido a la *teoría Neo-Darwiniana* que reconoce la importancia de los operadores de mutación y recombinación. Esta segunda etapa de la teoría evolutiva involucra varias premisas principales. Por un lado, que la evolución es el cambio en la frecuencia de los genes en el conjunto de genes comprendido por una población a lo largo de muchas generaciones. Por otro lado, que las especies (y sus conjuntos de genes) se encuentran aisladas entre sí. En tercer

lugar, que un individuo cuenta solamente con una porción del conjunto total de genes, la cual proviene de dos padres diferentes, y que dicha porción es diferente en cada individuo. Además, los genes recibidos por un individuo están sujetos a posibles mutaciones y recombinaciones. Finalmente, que la selección natural favorecerá a algunos individuos, cuyos genes contribuirán en mayor proporción al conjunto de genes de la siguiente generación. Por lo tanto, si bien la selección natural es uno de los principales mecanismos evolutivos, la mutación y la recombinación son también importantes.

3.1.2 Evolución

Una de las fundamentales observaciones dentro de la disciplina de genética molecular es que cada individuo en la naturaleza es una entidad *dual*, debido a que sus propiedades fenotípicas (las que vemos por fuera) son representadas a un menor nivel genotípico (en forma interna). Es decir, el *genotipo* del individuo codifica a su *fenotipo*. A su vez, dentro del genotipo las unidades funcionales de herencia encargadas de tal codificación, son las que conocemos como *genes*. Todos estos conceptos han sido y siguen siendo ampliamente estudiados. Así, hoy sabemos que todos los organismos vivientes en la tierra se apoyan en la base del ADN (Ácido Desoxirribonucléico), la famosa doble hélice que codifica a los organismos y en la cual podemos encontrar tripletas llamadas codones. Los *genes* son grandes estructuras dentro de esta doble hélice conteniendo muchos codones, que transportan el código de las proteínas. Según define el National Human Genome Research Institute cada una de las variantes para un gen en una posición determinada dentro del cromosoma (locus) se denomina alelo. Diferentes alelos producen variación en las características heredadas tales como color del cabello o grupo sanguíneo. Los cambios en el material genético de una población sólo pueden aparecer por variaciones en los genes y por el proceso de selección natural y no por el aprendizaje del individuo a lo largo de su vida. Es decir, todas las variaciones (mutaciones y recombinaciones) ocurren a nivel genotípico. Sin embargo, el proceso de selección se basa en el desempeño del individuo en un entorno determinado, lo cual ocurre a nivel fenotípico.

¿Qué es entonces *evolución*? Una de las definiciones breves aceptadas por el ámbito de las ciencias biológicas es la siguiente:

Definición 11 (**Evolución** [Ash06])**.** *La evolución es la variación en la frecuencia de los alelos dentro de la población, a través del tiempo de generación en generación.*

En esta definición, el significado de *frecuencia* es el mismo que se utiliza en estadística. Es decir, cuando un individuo abandona la población, la frecuencia global de cada tipo de alelo en la población sufre una modificación y lo mismo ocurre cuando un individuo nace. Estas ideas son las que han sido aplicadas para el desarrollo de los algoritmos evolutivos.

3.2 Algoritmos evolutivos

Históricamente el desarrollo de herramientas (algoritmos) que permitan resolver problemas ha sido y es uno de los temas centrales en ciencias de la computación. El campo de investigación en algoritmos evolutivos (EAs, por sus siglas en Inglés) tiene un gran número de colaboradores e impulsores. Se pueden encontrar revisiones concisas y extendidas sobre la historia de los EAs en la literatura [Bäc96, BHS97, Fog98, Whi01, ES03, Ash06].

3.2.1 Analogía entre el concepto natural y el artificial

Los algoritmos evolutivos pueden verse como una metáfora respecto de los conceptos reales de evolución natural, en la cual, cada concepto de un EA intenta simular a uno biológico asociado a él. La figura 3.1 muestra algunas de las asociaciones más comunes. El concepto de *evolución* intenta ser imitado por la búsqueda de la solución óptima, dado que se intenta encontrar una resolución cada vez mejor (más evolucionada) para el problema. Cada solución candidata representa a un *individuo* de la población. La noción de

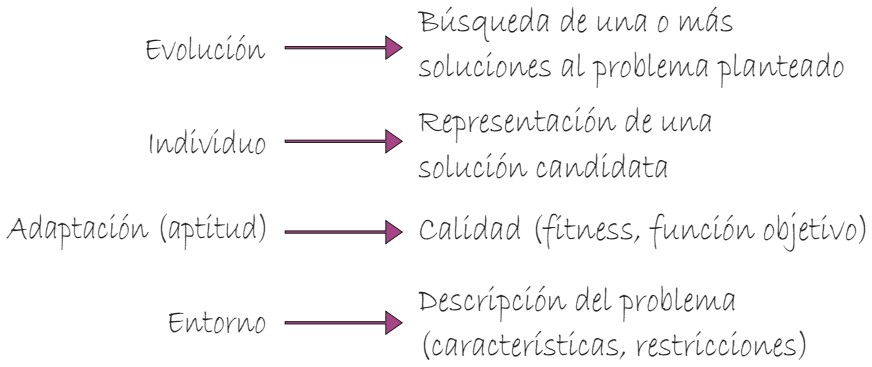

Figura 3.1: Algunas analogías entre el concepto natural de evolución y el esquema artificial en el que se basan los EAs.

adaptación biológica se asocia con la calidad de la solución. En tanto que el *entorno* en el cual se desarrollan los organismos biológicos está ligado a la descripción del problema

bajo estudio. Dentro de esta descripción se contemplan las características particulares del problema, tales como las entidades que participan en él y lo que se espera de ellas, es decir, los puntos de partida y las metas que desean alcanzarse. A su vez, esta descripción puede definir características adicionales que constituyen lo que conocemos como *restricciones*. La idea de resolver un determinado problema a partir de esta técnica es casi mágica y simple. Sin embargo, llevar a cabo la traducción no es tan sencillo. Esto se debe a que es necesario abstraerse en todos los aspectos que describen al problema, dejando de lado las cuestiones que no aportan nada a la solución. Cuando estamos comenzando a desarrollar la metáfora evolutiva para determinado problema, no siempre es trivial separar claramente los aspectos que afectan de los que no. Por eso, el desarrollo de un algoritmo evolutivo suele ser una tarea iterativa en la cual el paso inicial es un modelo simple. Luego, este modelo se va mejorando simultáneamente con nuestra familiaridad tanto con el problema como con las soluciones que van siendo encontradas. Si bien podemos ver cierta dependencia con el problema que estamos intentando resolver, hay pasos básicos que debemos dar para lograr una abstracción del problema y reflejarla en un enfoque evolutivo.
Existen muchas variantes de algoritmos evolutivos, pero la idea que gobierna a todas es la misma: dada una población de individuos, las condiciones de entorno provocan una selección natural (supervivencia del más apto), lo cual a su vez se traduce en una mejora global en el fitness de la población. Dada una función de calidad a ser optimizada (ya sea un máximo o un mínimo), podemos comenzar creando una población de individuos de forma aleatoria a partir de elementos que pertenezcan al dominio de la misma. Luego, basándonos en la función de calidad, algunos de los mejores candidatos se eligen para obtener la próxima generación a través de operadores de recombinación y/o mutación. El proceso de evaluación, selección, recombinación y/o mutación y establecimiento de la nueva población puede repetirse hasta que se alcanza determinada condición de terminación (p. ej.: uno o más individuos logran suficiente calidad, se superó el límite de recursos computacionales destinados, etc). A partir de este proceso podemos identificar varias fuerzas fundamentales que forman la base del sistema evolutivo:

- El método de selección (que determina el curso en la supervivencia del más apto).
- Los operadores de variación (recombinación y mutación).
- La función de calidad (encargada de medir el fitness de cada individuo).

3.2.2 Principales ejemplos de algoritmos evolutivos

Existen múltiples instancias de algoritmos evolutivos que surgen a partir de diferentes variaciones. Algunas de las más conocidas son los Algoritmos Genéticos (AGs), las Estrategias Evolutivas (EEs), la Programación Evolutiva (EP) y la Programación Genética (PG).

- Los *algoritmos genéticos* fueron descritos originalmente por Holland [Hol62, Hol73]. Se caracterizan por resaltar la recombinación (*crossover*) como el operador más importante aplicando *mutación* con una probabilidad sumamente baja, lo cual lo deja como un operador de segundo plano. Además, emplea un operador de selección probabilístico (*selección proporcional*) y comúnmente utiliza una *representación binaria* para los individuos.
- Las *estrategias evolutivas* (ESs) surgieron en forma contemporánea a los GAs. Sus primeros creadores y contribuyentes fueron Rechenberg y Schwefel [Rec73, Sch65]. A diferencia de los AGs, las ESs tienden a enfatizar el operador de mutación por encima del operador de recombinación. Además, las ESs en general se caracterizan por utilizar poblaciones de menor tamaño y para la representación suelen emplear vectores valuados reales. Los dos tipos básicos de ESs se conocen como la (μ, λ)-ES y la $(\mu + \lambda)$-ES. En las cuales μ es el tamaño de la población de padres y λ es el tamaño de la población de descendientes antes de que se aplique el operador de selección. En la (μ, λ)-ES, los descendientes reemplazan a los padres. En cambio en la $(\mu + \lambda)$-ES, la nueva generación se forma seleccionando individuos padres e individuos hijos.
- Las primeras menciones al término *Programación Evolutiva* (EP) son contemporáneas a los comienzos de los AGs y las ESs, ya que fueron introducidas por Larry Fogel en los años 60's [FOW66]. El concepto clásico de EP contempla la evolución de autómatas finitos (máquinas de estados finitos). Sin embargo, en años posteriores, David Fogel (hijo del primero) desarrolló variantes para la optimización de vectores valuados reales [Fog92, Fog95]. Estas últimas se han vuelto más populares y se han convertido en la *EP estándar*. La EP estándar es considerada similar a las ESs, ya que utiliza vectores valuados reales para la representación y una mutación dirigida por introducción de *ruido* del mismo estilo que la empleada en ESs. Por otro lado, la EP se destaca por no poseer operador de recombinación. Luego de comparar resultados entre algoritmos con y sin recombinación, Fogel y Atmar concluyeron que se obtiene mejor performance en la versión sin recombinación [FA90].

- La *Programación Genética* (GP) es una forma de algoritmo evolutivo que se distingue por su representación arbórea. Entre las características principales de la GP, Koza destaca que las estructuras que se manejan en GP son programas de computadora, por lo cual la evaluación del fitness consiste en ejecutar dichos programas [Koz92]. Existen varios operadores genéticos que han sido aplicados en GP. Los operadores de primer grado son el de *reproducción*, el cual opera sobre un único padre, y el de *recombinación*, que a partir de dos padres produce dos nuevos hijos. Mientras que entre los operadores de segundo grado se encuentran la *mutación*, la cual introduce cambios dentro de los nodos de la estructura, y la *permutación*, la cual en vez de modificar el contenido de un nodo, modifica la ubicación de dicho nodo dentro del árbol.

3.2.3 Definición formal de algoritmo evolutivo básico

Muchos de los componentes de este proceso son claramente estocásticos. Por ejemplo, durante la selección los mejores individuos tienen más probabilidades para ser seleccionados; sin embargo, también pueden ser elegidos algunos de los individuos menos aptos. La suerte de ser recombinados también es un evento al azar y lo mismo ocurre con la mutación. El esquema general de un algoritmo evolutivo es el siguiente:

Definición 12 (**Algoritmo Evolutivo** [BFM97])**.** *Sea el espacio arbitrario de individuos* I, $I \neq \emptyset$ *y una función de aptitud para los individuos valuada real* $F : I \to \mathbb{R}$. *Sean* μ *y* λ *los tamaños de las poblaciones de padres y descendientes respectivamente. Sea* $P(t) = (a_1(t), \ldots, a_\mu(t)) \in I^\mu$ *la población en la generación t. Consideremos los operadores de crossover, mutación y selección denotados como* $c : I^\mu \to I^k$, $m : I^k \to I^\lambda$ *y* $s : I^\lambda \to I^\mu$. *Sean* Θ_c, Θ_m *y* Θ_s *los conjuntos de parámetros típicos para los respectivos operadores de crossover, mutación y selección. Por último, sea* Θ_t *el conjunto de parámetros para el criterio de terminación. Un algoritmo evolutivo básico se reduce al siguiente ciclo de recombinación-mutación-selección:*

Entrada: $\mu, \lambda, \Theta_t, \Theta_c, \Theta_m, \Theta_s$
Salida: *a^*, el mejor individuo encontrado durante la corrida o*
P^*, la mejor población encontrada durante la corrida
Comienzo
1 $t \leftarrow 0$
2 $P(t) \leftarrow inicializar(\mu)$
3 $F(t) \leftarrow evaluar(P(t), \mu)$
mientras no $(terminacion(P(t), \Theta_t))$ **hacer**
4 $P'(t) \leftarrow recombinar(P(t), \Theta_r)$
5 $P''(t) \leftarrow mutar(P'(t), \Theta_m)$
6 $F(t) \leftarrow evaluar(P''(t), \lambda)$
7 $P(t+1) \leftarrow seleccionar(P''(t), F(t), \mu, \Theta_s)$
8 $t \leftarrow t+1$
9 $P^* \leftarrow P(t+1)$
10 $a^* \leftarrow mejorIndividuo(P^*)$
Fin

Describir los operadores a nivel de población permite alcanzar una perspectiva de alto nivel suficientemente general como para cubrir diferentes instancias de algoritmos evolutivos clásicos. Si bien la definición anterior abarca a los algoritmos evolutivos clásicos, puede ser útil para definir algoritmos evolutivos multi-objetivo.

3.2.4 Una definición formal de algoritmo evolutivo multi-objetivo

En la literatura existen pseudocódigos que representan a grandes rasgos los pasos a seguir por un MOEA [CLV07], los cuales pueden ser tomados como una definición. A continuación se presentará una nueva definición que hace explícito el concepto de *multi-objetivo* partiendo de la Definición 12.

Definición 13 (**Algoritmo Evolutivo Multi-Objetivo**). *Sea el espacio arbitrario de individuos I, $I \neq \emptyset$ y una función de aptitud para los individuos $F : I \rightarrow \mathbb{R}^n$. Sean μ y λ los tamaños de las poblaciones de padres y descendientes respectivamente. Sea $P(t) = (a_1(t), \ldots, a_\mu(t)) \in I^\mu$ la población en la generación t. Consideremos los operadores de crossover, mutación y selección denotados como $c : I^\mu \rightarrow I^k$, $m : I^k \rightarrow I^\lambda$ y $s : I^\lambda \rightarrow I^\mu$.*

Sean Θ_c, Θ_m, Θ_{sp} y Θ_{ss} los conjuntos de parámetros típicos para los respectivos operadores de crossover, mutación, selección de padres y selección de sobrevivientes. Por último, sea Θ_t el conjunto de parámetros para el criterio de terminación. Un algoritmo evolutivo multi-objetivo se reduce al siguiente ciclo de recombinación-mutación-selección:

Entrada: $\mu, \lambda, \Theta_t, \Theta_c, \Theta_m, \Theta_{sp}, \Theta_{ss}$
Salida: $frente^*$, *el mejor frente encontrado durante la corrida*
P^*, *la última población de la corrida*

Comienzo

1 $t \leftarrow 0$
2 $P(t) \leftarrow inicializar(\mu)$
3 $ObjectiveValues(t) \leftarrow evaluar(P(t), F, \mu)$
mientras no $(terminacion(P(t), \Theta_t))$ **hacer**
4 $P'(t) \leftarrow seleccionarPadres(P(t), ObjectiveValues(t), \mu, \Theta_{sp})$
5 $P''(t) \leftarrow recombinar(P'(t), \Theta_r)$
6 $P'''(t) \leftarrow mutar(P''(t), \Theta_m)$
7 $ObjectiveValues(t) \leftarrow evaluar(P'''(t), F, \lambda)$
8 $P^{iv}(t) \leftarrow jerarquizar(P'''(t) \cup P(t), \mu, \lambda)$
9 $P(t+1) \leftarrow seleccionarSobrevivientes(P^{iv}(t), ObjectiveValues(t), \mu, \Theta_{ss})$
10 $t \leftarrow t+1$
11 $P^* \leftarrow P(t)$
12 $frente^* \leftarrow mejorFrente(P(t))$

Fin

Dentro de los pasos básicos mencionados en la Definición 13 cabe destacar la presencia de dos procesos de selección diferentes (líneas *4* y *9* del algoritmo). El primero está destinado a la selección de los individuos que serán sometidos a los operadores de variación. En cambio el segundo, está destinado a completar la población $P(t+1)$. En el caso de los algoritmos elitistas, es en el segundo paso de selección donde se aplica el elitismo. En tanto que el primer proceso de selección, si bien favorece a los mejores individuos, puede o no ser elitista. Otro paso destacable es el *8* (*jerarquizar*), en el cual se establece una jerarquía u orden parcial sobre la *población global*. Dicha población está compuesta por la unión de los individuos de la población antes de la aplicación de los operadores ($P(t)$) y los descendientes que surgieron luego de aplicar a dicha población los operadores de

selección, cruzamiento y mutación ($P'''(t)$). Contemplar simultáneamente ambas poblaciones en cada generación (es decir, la original y la que surge a partir de modificaciones genéticas realizadas sobre la primera) permite realizar de forma más directa un proceso elitista. Para llevar a cabo el orden parcial, los algoritmos basados en Pareto usan el concepto de dominancia de Pareto. Sin embargo, esta definición no está acotada sólo a este tipo de MOEAs.

Existe también un MOEA General denominado Algoritmo Evolutivo Multi-Objetivo General (GENMOP por sus siglas en Inglés), el cual fue diseñado en el US Air Force Institute of Technology [GHGL99, KGLH03]. A diferencia de la Definición 13, este diseño permite incluir una variedad de operadores evolutivos y posterga hasta el momento de la ejecución la elección de cuáles de estos operadores va a utilizar. A medida que avanza el proceso de búsqueda, el algoritmo prefiere a los operadores que han producido mejores individuos. Sin embargo, en el contexto de esta tesis, se utilizó la Definición 13 para el desarrollo de los distintos módulos.

Capítulo 4

Algoritmos evolutivos multi-objetivo

Los algoritmos evolutivos multi-objetivo (MOEAs) son una extensión natural de los algoritmos evolutivos mono-objetivo. Esta extensión consiste en el uso de algoritmos evolutivos para la resolución de problemas de optimización en el cual intervienen varios objetivos. En este capítulo veremos una síntesis sobre clasificación de técnicas generales para resolver problemas multi-objetivo. Luego, se mostrarán dos de las posibles clasificaciones para algoritmos evolutivos multi-objetivo y dentro de cada categoría, se presentará una revisión de los algoritmos y conceptos asociados más conocidos.

4.1 Clasificación de técnicas para resolver problemas multi-objetivo

Existe una amplia variedad de métodos destinados a optimización y búsqueda. Muchos de estos algoritmos comienzan el proceso con una solución al azar y luego, a partir de ciertas reglas determinísticas, el algoritmo sugiere una dirección de búsqueda. En base a esta dirección se realiza una búsqueda con el fin de encontrar una mejor solución, la cual se convierte en un nuevo punto de búsqueda a partir del cual el proceso se vuelve a repetir. Estos métodos son denominados determinísticos. En la literatura puede encontrarse una variedad de ejemplos de dichos esquemas [BB88, AH84, Pea84]. Cada método se desenvuelve mejor bajo ciertas circunstancias y muchos han sido exitosamente empleados para resolver una amplia variedad de problemas [BB88, Gol89]. Sin embargo, muchos problemas de optimización multi-objetivo (MOPs) se caracterizan por tener muchas dimensiones, ser discontinuos o ser NP-Completos. Se sabe que los métodos determinísticos

suelen ser poco efectivos a la hora de resolver este tipo de problemas. Este hecho se debe a que se encuentran limitados al momento de aplicar una heurística para poder dirigir la búsqueda [MF04, Gol89, Fog99, TAV99]. Además, algunas de las técnicas determinísticas (como los métodos directos o los basadas en gradiente) tienen algunas dificultades conocidas, por ejemplo, la convergencia a una solución óptima puede presentar una dependencia de la solución inicial seleccionada, o un algoritmo eficiente para cierto problema de optimización puede no serlo para un problema de optimización diferente, o pueden ser ineficientes al manejar espacios de búsqueda discretos.
Muchos problemas reales presentan las cualidades mencionadas antes; en consecuencia, se han desarrollado esquemas o aproximaciones alternativas a los determinísticos con el fin de abordar estos problemas más efectivamente. Algunos ejemplos son los métodos de Monte Carlo [Fis96] y los algoritmos evolutivos [Mic94, Gol89]. Estos esquemas pertenecen a un grupo de algoritmos de optimización y búsqueda denominados estocásticos, los cuales requieren algún método para determinar la aptitud (fitness) de las soluciones que se van encontrando. Si bien existen varios esquemas de clasificación para las técnicas destinadas a resolver problemas de optimización multi-objetivo, uno de los más conocidos es el utilizado por Hwang y Masud [HM79], quienes clasifican los métodos de acuerdo a la forma en que participa el responsable de la toma la decisiones (Decision Maker) en el proceso de solución. En base a esto se pueden distinguir cuatro clases:

1. Métodos en los que no se usa información de preferencia o *métodos sin preferencias*.
2. Métodos donde previamente se utiliza cierta información de preferencia o *métodos a priori*.
3. Métodos en los cuales se utiliza cierta información de preferencia en forma posterior o *métodos a posteriori*.
4. Métodos que usan información de manera progresiva o *métodos interactivos*.

En los **métodos sin preferencias**, donde no se toma en cuenta la opinión del Decision Maker (DM), el problema de optimización multi-objetivo se resuelve usando algún método relativamente simple y se presenta al Decision Maker la solución obtenida. A partir de ahí, él podrá aceptarla o rechazarla.
En los **métodos a priori**, el Decision Maker especifica sus preferencias, esperanzas y opiniones antes de que comience el proceso. La dificultad principal de estos enfoques radica en que no siempre se conocen estos datos de forma anticipada.
En los **métodos a posteriori**, se genera un conjunto de soluciones que forma un conjunto

óptimo de Pareto para el problema en cuestión, este conjunto se presenta al DM quien selecciona la alternativa más adecuada. Las dificultades de estos esquemas radican en el costo computacional y la complejidad del proceso de generación.
Por último, los **métodos interactivos** son aquellos en los que el DM trabaja junto con un analista o con un programa de computación interactivo. El analista (o el programa) intentará determinar la estructura de preferencia de manera interactiva. Luego de cada iteración, se le entregará alguna información al DM y se evaluará su reacción para modificar las preferencias de los objetivos de manera acorde. Luego de cierto número (finito) de iteraciones el método debe producir una solución aceptable para el decision maker o al menos mostrar que no hay mejora posible.
En el mundo de la implementación de técnicas para resolver MOP lo más general es una división en tres clases: las que optimizan los objetivos según un orden de importancia, las que combinan todos los objetivos dentro de una única fórmula y las que procuran encontrar el frente de Pareto óptimo.

4.2 Uso de algoritmos evolutivos

Según Coello y sus colegas [CLV07] el uso de algoritmos evolutivos para resolver problemas de optimización multi-objetivo data de los 60s. Aunque destaca que la primera implementación fue propuesta por David Schaffer en su tesis doctoral [Sch84] a mediados de los 80s. Schaffer propuso el Vector Evaluation Genetic Algorithm (VEGA), el cual estuvo dirigido principalmente a resolver problemas de *machine learning* [Sch85]. El algoritmo trabajaba eficientemente por algunas generaciones pero en ocasiones sufría cierta tendencia a encontrar super individuos (se polarizaba). Si bien el objetivo de llevar a cabo una optimización multi-objetivo se realizaba con éxito, no se lograba mantener del todo una buena dispersión entre los individuos de la población. Aunque durante la siguiente década no se vieron grandes cantidades de aportes en esta área, en investigaciones posteriores han surgido varias generaciones de algoritmos evolutivos multi-objetivo.
Una de las diferencias entre la mayoría de los métodos de búsqueda y optimización clásicos y los algoritmos evolutivos es que, estos últimos, procesan un *conjunto* de posibles soluciones en cada iteración en lugar de una sola. Esto le otorga a los EAs una gran ventaja al momento de intentar resolver multi-OOPs, ya que la búsqueda tiene más probabilidades de encontrar óptimos globales. Además, gracias a que un EA trabaja con una población de soluciones, en teoría, podríamos intentar capturar varias soluciones pertenecientes al

frente óptimo de Pareto de nuestro problema.
Las técnicas utilizadas para representar las soluciones y llevar a cabo la variación genética de los individuos en algoritmos evolutivos mono-objetivo pueden ser utilizados para intentar resolver problemas de optimización multi-objetivo. En general, no existen consideraciones especiales para elegir la codificación o diseñar los operadores de recombinación y mutación. Las diferencias más radicales al momento de realizar optimización multi-objetivo se encuentran en la función objetivo, ya que las diferencias en éstas afectan el diseño de la función de fitness y del operador de selección. Cabe aclarar que los esquemas basados en Pareto (algunos de los cuales se explican en las siguientes secciones), además de las consideraciones especiales en el diseño de la función de fitness, suelen presentar consideraciones especiales en el proceso de selección.

4.3 Enfoques evolutivos

Existe una gran cantidad de algoritmos evolutivos para optimización multi-objetivo y no hay una única manera de clasificarlos. Por un lado, tal como las otras técnicas de optimización multi-objetivo, los esquemas evolutivos pueden ser métodos *a priori* (los que llevan a cabo las decisiones antes de llevar a cabo la búsqueda), *a posteriori* (realizan primero la búsqueda y luego la toma de decisiones) o *interactivos* (los que realizan un proceso que alterna entre ambas cosas). Por otro lado, desde el punto de vista de las técnicas evolutivas, una de las posibles clasificaciones, divide a los algoritmos evolutivos multi-objetivo en dos grandes grupos:

- *Esquemas no-Pareto*: entre los cuales se contemplan los esquemas agregativos (en donde los objetivos se combinan numéricamente en una única función objetivo a ser optimizada) y los esquemas basados en la población (donde los diferentes objetivos afectan la selección o des-selección de diferentes partes de la población).
- *Esquemas basados en Pareto*: en los cuales la población se clasifica de acuerdo con la definición de dominancia de Pareto.

4.4 MOEAs no-Pareto: reseña histórica

En el libro *Handbook of Evolutionary Computation* [BFM97] Horn destaca dos enfoques agregativos principales dentro de los esquemas a priori: el Agregativo-Escalar y el Orden-Agregativo. Este énfasis coincide también con autores como Coello [CLV07].

4.4.1 Esquemas agregativo-escalares

Los esquemas *agregativo-escalares* combinan los distintos objetivos en una única función $T(\overline{a})$ valuada-escalar, tal que $T : \mathbb{R}^k \rightarrow \mathbb{R}$ refleja las preferencias particulares de un DM. En este esquema se puede utilizar una composición de funciones $To F$ como función de fitness para el MOEA. En particular, ciertos métodos de selección como los *proporcionales al fitness* (por ej. ruleta), requieren una función de fitness escalar, en tanto que otros requieren sólo un orden completo o aun menos, un orden parcial.
El ejemplo más simple de función agregativa escalar son las *Funciones Agregativas Lineales*, que son sumas con pesos (weighted sums) de la forma:

$$T(\overline{a}) = w_0 a_0 + w_1 a_1 + ... + w_{k-1} a_{k-1} = \sum_{i=1}^{k} w_i a_i \tag{4.1}$$

donde los w_i son coeficientes constantes o pesos, cuya suma usualmente verifica $\sum_{i=1}^{k} w_i = 1$. La ecuación anterior suele usarse para computar el fitness de cada individuo $\mathbf{x}$, tomando $a_i = f_i(\mathbf{x})$, es decir, a_i es el valor del objetivo i para $\mathbf{x}$. Una desventaja de este esquema es que sirve solo para relaciones lineales entre objetivos. La figura 4.1 muestra líneas

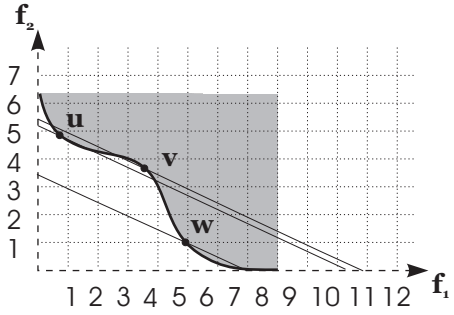

Figura 4.1: Esquema agregativo $w_1 f_1 + w_2 f_2$ para un problema con dos objetivos

paralelas que pasan por algunos de los puntos pertenecientes al frente óptimo de Pareto, las líneas responden a una función pesada del tipo $w_1 a_1 + w_2 a_2$. El punto **u** representa un valor mínimo para la función y pertenece al frente óptimo; sin embargo, el punto **v**, que también pertenece al frente óptimo, será descartado debido a que la función agregativa con esos pesos encontró un valor menor en el punto **u**. Observemos además que diferentes pesos otorgarán una pendiente diferente a las rectas y éstas interceptarán al frente óptimo de Pareto en diferentes puntos. Este ejemplo nos permite ver que una función agregativa

lineal no siempre encontrará a todos los puntos del frente. Estos puntos suelen definirse como *puntos no soportados*. Sin embargo, pueden intentar optimizarse varias funciones distintas agregadas linealmente para armar un conjunto final de soluciones con los óptimos encontrados en cada optimización.
Otro esquema agregativo escalar muy conocido son las *funciones agregativas no lineales*. Los ejemplos más comunes de este esquema son las que trabajan con restricciones y las que miden distancia a un objetivo (target). En las primeras, las restricciones modelan ciertas particularidades no lineales del problema, como las que aparecen cuando un DM se maneja por medio de umbrales (cotas máximas o mínimas). Una forma de resolver estas restricciones es imponer una gran penalización a la solución cuando sobrepasa alguno de los umbrales. En la literatura pueden encontrarse diversos artículos referentes al manejo de restricciones. Uno de estos esquemas, sugerido por Richardson *et al.* [RPLH89], consiste en medir la distancia Euclídea entre una solución $\mathbf{x}$ y la región factible, es decir, una suma lineal de las distancias en las que la solución sobrepasó cada umbral (o la suma de las distancias elevadas a cierto exponente). En los esquemas de distancia a un objetivo, también conocidos como *target-vector* o *distance-to-target* se elige un vector T como solución ideal. Luego las soluciones que se van encontrando se evalúan midiendo su distancia al objetivo. Tanto el vector objetivo como la forma de medir la distancia serán determinados de antemano por el DM. Se pueden emplear distintas métricas y hay varias implementaciones en la literatura.

4.4.2 Esquemas orden-agregativos

Los MOEAs basados en *orden-agregativo* son agregaciones que no entregan como resultado un valor escalar. Por ejemplo, el esquema de *orden lexicográfico* realiza un ordenamiento total de todas las soluciones. Esto implica que el DM realice previamente un orden de importancia de las funciones objetivo. Luego, las soluciones se ordenan en base a la prioridad de cada objetivo. A partir de esto se obtiene la solución óptima minimizando las funciones objetivo en forma secuencial, empezando con la primera en el orden de importancia y continuando con ese orden hasta la última función objetivo, es decir, de forma similar al orden de los items en un diccionario. Una clara dificultad del esquema es que no siempre es posible conocer *a priori* la importancia de cada objetivo. Sin embargo, si no se conoce la prioridad, es posible seleccionar los objetivos a optimizar al azar [Fou85]. Coello marca como principal debilidad de este esquema que cuando el problema presenta muchos objetivos, tiende a favorecer más a algunos de ellos debido a la

aleatoriedad involucrada en el problema. Por otro lado, sus fortalezas son la simplicidad y la eficiencia computacional. Las técnicas lexicográficas parecen ser más útiles cuando se conoce claramente la prioridad de cada objetivo.

Dentro de los enfoques no-Pareto cabe mencionar al *Vector Evaluated Genetic Algorithm* (VEGA) [Sch85]. Este algoritmo es considerado por muchos autores como el primer esquema evolutivo implementado para resolver multi-OOPs [BFM97, CLV07, Vel99]. Para un problema con k objetivos, el VEGA genera k sub-poblaciones de tamaño n/k. Donde n es el tamaño total de la población. Cada sub-población utiliza una sola de las k funciones objetivo para asignar fitness. La figura 4.2 muestra los pasos básicos del VEGA. En el primer paso los individuos de la población se someten a una selección proporcional

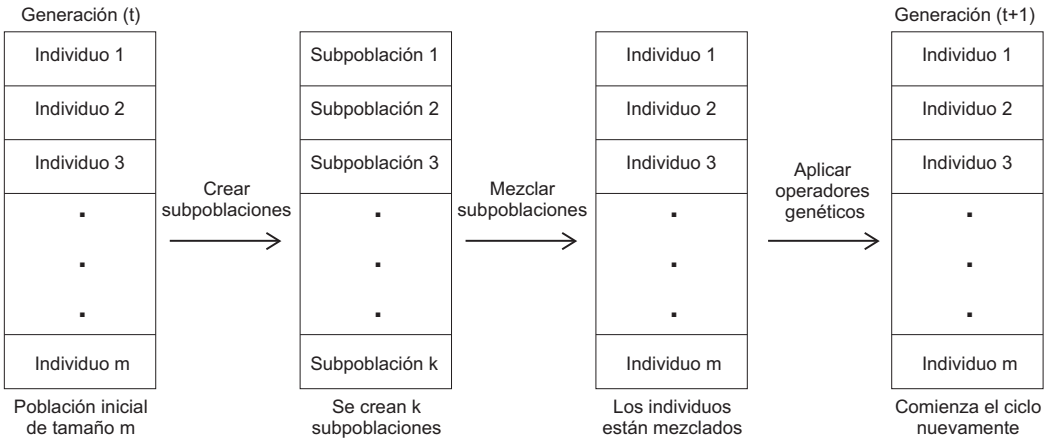

Figura 4.2: Pasos básicos del mecanismo de selección del Vector Evaluated Genetic Algorithm (VEGA)

por separado para cada uno de los k objetivos, dando origen de esta manera a las k subpoblaciones. El segundo paso es unir (mezclar) dichas sub-poblaciones para formar una población de posibles padres. En el tercer paso se aplican los operadores genéticos a los individuos de la población de padres y se genera la generación $i+1$. Schaffer notó que las soluciones generadas eran no dominadas localmente, ya que la no dominancia se acotaba a la población actual. Si bien un individuo dominado localmente es dominado globalmente, un individuo no dominado localmente puede ser dominado globalmente. Además, descubrió que debido a que la técnica selecciona individuos muy buenos respecto de los demás en cierta dimensión (para uno de los objetivos), pero no necesariamente en las otras dimensiones, se puede producir un efecto no deseado denominado *especiación*. Este efecto consiste básicamente en la formación de especies que intentan evolucionar en dis-

tintas direcciones. Con este esquema de selección no sobrevivirán individuos que sean relativamente buenos en todas las direcciones en forma simultánea, los cuales pueden ser muy buenos para soluciones de compromiso como las que buscamos. Otro de los problemas con esta técnica es que, cuando el frente óptimo de Pareto sea cóncavo, el algoritmo fallará en encontrar algunos de sus puntos, lo cual se debe a que la forma de trabajar del VEGA es equivalente a una combinación lineal de los objetivos [BFM97, RPLH89]. Tanto los métodos agregativos como el VEGA suelen ser considerados parte de la primera generación de algoritmos evolutivos pero que no incorporan dominancia de Pareto.

4.5 MOEAs basados en Pareto: reseña histórica

Como era de esperar los algoritmos evolutivos también han experimentado una suerte de evolución. Una primera generación se caracterizó por el uso del concepto de dominancia de Pareto. En tanto que la segunda generación incorporó el uso de elitismo. La idea de utilizar dominancia de Pareto para asignar el valor de fitness a cada individuo fue introducida por David Goldberg [Gol89] quien analizó el comportamiento del VEGA y propuso un esquema de selección basado en optimalidad de Pareto.

4.5.1 Fitness Sharing Clásico - Concepto

En la segunda Conferencia Internacional sobre algoritmos genéticos Goldberg y Richardson propusieron el uso de una función de aptitud cuyo concepto se conoce como *Fitness Sharing* [GR87]. El objetivo de dicha función es distribuir a la población, P, entre los diferentes vértices del espacio de búsqueda, de tal forma que cada vértice reciba una porción de la población. Para lograr esto, la función de sharing degrada el *fitness objetivo* f_i de cada individuo en base a un valor de *nicho* m_i calculado para el individuo. La degradación se obtiene definiendo una nueva función de fitness f'_i, como el cociente entre ambos valores, es decir $f'_i = f_i/m_i$. Mientras que el valor de nicho m_i se calcula como:

$$m_i = \sum_{j \in P} Sh(d_{ij}) \tag{4.2}$$

El valor d_{ij} es la distancia entre el individuo i y el individuo j y $Sh(d)$ es la siguiente *sharing function*:

$$sh(d) = \begin{cases} 1 - \left(\frac{d}{\sigma_{share}}\right)^{\alpha}, & si \quad d \leq \sigma_{share} \\ 0, & si \quad d > \sigma_{share} \end{cases} \tag{4.3}$$

El parámetro σ_{share} es el radio del nicho fijado por el usuario para estimar un valor mínimo de separación entre nichos. Los individuos que se encuentran dentro de una distancia σ_{share} se degradan mutuamente por pertenecer al mismo nicho.

4.5.2 Primera Generación

Existen muchos algoritmos evolutivos que clasifican en este grupo. En esta sección veremos algunos de los principales.

- **Multi Objective Genetic Algorithm: MOGA.** Este algoritmo fue propuesto por Fonseca y Fleming en el año 1993 [FF93]. Su importancia radica en que los autores lograron aplicar el concepto de individuos no-dominados y en forma simultánea atacaron el problema de la diversidad entre dichos individuos. Para entender su funcionamiento veamos en forma resumida los pasos que sigue el algoritmo:
  Paso 1: Calcular la puntuación para el i-ésimo individuo como $r_i = 1+q_i$, donde q_i es el coeficiente de dominancia equivalente a la cantidad de individuos que dominan a i en la generación actual. Incrementar en 1 el contador de puntuaciones, c, en la posición r_i, $c(r_i) = c(r_i) + 1$.
  Paso 2: Ordenar a la población de acuerdo con su puntuación de menor a mayor. Cabe notar que los individuos de igual puntuación quedan consecutivos.
  Paso 3: Asignar un valor de fitness, por ejemplo, interpolando los números de n a 1, de forma correspondiente con los individuos mejor ($r_i = 1$) a peor ($r_i = max$). Es decir, $f = n$ para el primer individuo en el orden, $f = n - 1$ para el segundo y así sucesivamente. Promediar los valores de fitness para aquellos individuos que pertenecen al mismo nivel de puntuación.
  Paso 4: Para cada solución con puntuación r calcular su valor de nicho.
  El último paso es el que marca la diferencia principal entre las técnicas anteriores y el MOGA de Fonseca y Fleming, ya que ellos propusieron utilizar el parámetro de partición σ en el espacio objetivo en lugar de utilizarlo en el dominio de las variables de decisión.
  Desventajas: El parámetro de partición puede provocar que la presión selectiva no se aplique de forma adecuada. Esto se debe a que si en un determinado momento existen muchas buenas soluciones dentro del mismo nicho, todas resultarán afectadas y su fitness compartido será bajo. Con lo cual una solución peor que ellas pero que se encuentre aislada resultará favorecida. Su rendimiento depende fuertemente de

una correcta elección del parámetro σ.
Ventajas: Simplicidad y eficiencia [Coe96].
Observación: El tiempo de ejecución es de $O(kn^2)$.

- **Niched Pareto Genetic Algorithm: NPGA.** Este algoritmo, propuesto por Horn *et al.* [HNG94], trabaja con un esquema de selección basado en dominancia de Pareto. Dados dos individuos u_i, u_j y una población P, la selección se lleva a cabo de la siguiente forma:
Paso 1: Elegir una subpoblación T_{ij} de tamaño t_{dom}.
Paso 2: Calcular α_i como el número de soluciones en T_{ij} que dominan a u_i. Calcular α_j como el número de soluciones en T_{ij} que dominan a u_j.
Paso 3: Si $\alpha_i = 0$ y $\alpha_j > 0$ entonces u_i gana. Si $\alpha_j = 0$ y $\alpha_i > 0$ entonces u_j gana.
Paso 4: Si hay un empate se aplica *sharing function* como se explicó en la sección 4.5.1.
Desventajas: Requiere una cuidadosa elección de dos parámetros importantes: σ_{share} y t_{dom}, ya que el rendimiento se ve afectado por ambos.
Ventajas: Si el valor t_{dom} se mantiene tan grande como para obtener valores de comparación significativos pero bastante más pequeño que el tamaño de P, la complejidad del algoritmo puede no depender fuertemente de la cardinalidad del conjunto de funciones objetivo. Con lo cual puede lograrse eficiencia computacional.
Observación: El tiempo de ejecución es de $O(mn^2)$ si t_{dom} es del orden de n, donde m es la cantidad de objetivos. Sin embargo se puede lograr $O(n^2)$ si t_{dom} es mucho menor que n.

- **Non-dominated Sorting Genetic Algorithm: NSGA.** Es una variante del MOGA propuesta por Srinivas y Deb en 1994 [SD94]. Este algoritmo realiza una clasificación agrupando individuos no dominados en base a su nivel de "no dominación". Una vez que todos los individuos fueron clasificados, los que pertenecen a niveles más bajos (los menos dominados) tienen más probabilidades de ser elegidos en el proceso de selección. Mientras tanto, una función de *Sharing* ayuda a mantener cierto grado de diversidad en la población. El NSGA varía de un algoritmo genético simple en los pasos previos al proceso de selección. Durante dichos pasos la población se clasifica de a cuerdo a su nivel de "no dominación" de la siguiente forma:
Paso 1: Se identifican los individuos no dominados de la población actual, los

cuales pasan a formar parte del *primer* frente no dominado en la población y a los cuales se les asigna un valor de fitness ficticio. Este valor es el mismo para todos los individuos del frente a fin de mantener la misma posibilidad de reproducción entre ellos.

Paso 2: Para lograr diversidad se emplea una función de *sharing* similar a la vista en la sección 4.5.1. El valor de *sharing* entre dos individuos del frente actual se calcula con la siguiente ecuación:

$$sh(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\sigma_{share}}\right)^2, & \text{si} \quad d_{ij} < \sigma_{share} \\ 0, & \text{en caso contrario} \end{cases} \tag{4.4}$$

En la ecuación anterior, el parámetro d_{ij} es la distancia fenotípica entre el individuo i y el j en el frente actual. El parámetro σ_{share} es la máxima distancia fenotípica permitida entre dos individuos cualesquiera del frente para que puedan formar parte del mismo nicho.

Paso 3: Se calcula la *cuenta de nicho* como la sumatoria de los valores obtenidos en la función de sharing para todos los individuos del frente.

Paso 4: Para cada individuo se obtiene un *fitness final* calculado como el fitness ficticio del individuo dividido por el valor de nicho del individuo.

Paso 5: Una vez que el sharing y el fitness final fueron calculados para el frente actual, estos individuos se ignoran temporalmente y se procesa el resto de la población de forma similar repitiendo los pasos 1 a 4. Es decir en un segundo ciclo se identificarían los individuos pertenecientes al *segundo frente no dominado* y se les asignaría un valor de fitness ficticio. Cabe aclarar que en cada ciclo el valor de fitness ficticio asignado es menor que el mínimo fitness final obtenido para el frente anterior.

Cuando se termina de calcular el fitness de todos los individuos de la población se realiza el proceso de selección y el resto de los pasos clásicos de un algoritmo evolutivo. Cabe destacar que el método de sharing funciona con los operadores proporcionales al fitness, en particular el NSGA utiliza el operador *Muestreo Universal Estocástico* conocido en Inglés como *Stochastic Universal Sampling* o *Stochastic Remainder roulette-wheel*.

Desventajas: El esquema obliga a especificar el parámetro σ_{share}. Además, algunos investigadores han reportado que es más sensible a este parámetro que el MOGA.

Ventajas: La principal ventaja es asignar el fitness en base a conjuntos de individ-

uos no dominados junto con que el proceso de sharing dentro del espacio de decisión permite diversidad fenotípica. En un trabajo comparativo, Zitzler [ZT98] reportó que el NSGA trabaja bien en términos de *cobertura* del frente de Pareto, es decir, la población se distribuye de manera razonablemente uniforme sobre el frente.
Observación: El tiempo de ejecución es de $O(mn^3)$ para m funciones objetivo.

4.5.3 Segunda generación

La segunda generación de algoritmos evolutivos se caracteriza por el uso de elitismo. Esta noción significa que los individuos 'elite' no pueden ser expulsados de la población activa a cambio de individuos peores. En el caso de algoritmos evolutivos mono objetivo, el elitismo implica llevar el rastro del mejor individuo obtenido durante toda la ejecución. Dicha solución puede verse como la mejor aproximación a la solución óptima. El análogo multi-objetivo es almacenar todas las soluciones no dominadas por ninguna otra solución durante toda la ejecución. En este caso, dicho conjunto de soluciones representa la mejor aproximación al frente óptimo de Pareto.
Existen varias cuestiones a tener en cuenta al momento de implementar elitismo. Por ejemplo, podemos usar una población elitista secundaria o podemos implementar elitismo implícito a través de un método de selección elitista (*'plus' selection*). En el caso de que se implemente el elitismo a través de una población secundaria ¿Cómo será actualizada dicha población? ¿Cuándo se llevan a cabo el acopio y la reinserción de dichos individuos? Estas y otras decisiones deben ser tomadas al momento de implementar un MOEA elitista. En los algoritmos de esta sección podremos ver como fueron abordadas algunas alternativas.

- **Strength Pareto Evolutionary Algorithm: SPEA.** Este enfoque propuesto por Zitzler y Thiele en 1999 [ZT99] combina varias características de enfoques evolutivos anteriores. Por ejemplo, almacena las soluciones no dominadas en una población externa que es constantemente actualizada. El fitness de un individuo depende de cuántos individuos de la población externa lo dominan. Intenta preservar la diversidad en la población utilizando la relación de dominancia. Una de las particularidades de este algoritmo es el esquema de clustering que utiliza para que la población externa no se desborde más allá de cierto límite.
  El algoritmo comienza con una población inicial $\mathcal{P}_0$ cuyo tamaño es n y con una población externa vacía $\overline{\mathcal{P}}_0$ cuya capacidad máxima es $\overline{n}$. En toda generación t, las soluciones no dominadas de $\mathcal{P}_t$ se copian a la población externa $\overline{\mathcal{P}}_t$. Luego de

esta copia se determinan cuáles son los individuos no dominados de la población externa (mirando sólo los individuos en $\overline{\mathcal{P}}_t$). Todos los individuos no dominados se mantienen y los demás se eliminan. De esta forma los individuos no dominados pueden sobrevivir de generación en generación. Por otro lado, el tamaño de la población externa se limita a $\overline{n}$ para evitar que crezca de manera excesiva. Esto significa que no pueden mantenerse en la población externa todos los individuos no dominados encontrados cuando se sobrepasa este umbral. Para resolver esta situación los autores implementaron un método de clustering que procura mantener a los individuos de las clases menos pobladas. Una vez seleccionadas las clases (o élites), el algoritmo usa los operadores genéticos para hallar una nueva población. En forma resumida los pasos que ejecuta el SPEA son los siguientes:

Paso 1: Generar una población inicial $\mathcal{P}_t$ y crear la población externa vacía $\overline{\mathcal{P}}_t$ para $t = 0$.

Paso 2: Hallar el conjunto de individuos no dominados de $\mathcal{P}_t$, $\mathcal{F}_1(\mathcal{P}_t)$. Copiar esas soluciones a la población externa $\overline{\mathcal{P}}_t$ ($\overline{\mathcal{P}}_t = \overline{\mathcal{P}}_t \cup \mathcal{F}_1(\mathcal{P}_t)$).

Paso 3: Hallar el conjunto de individuos no dominados de $\overline{\mathcal{P}}_t$, $\mathcal{F}_1(\overline{\mathcal{P}}_t)$. Eliminar las soluciones dominadas de $\overline{\mathcal{P}}_t$.

Paso 4: Si $|\overline{\mathcal{P}}_t| <= \overline{n}$, mantener $\overline{\mathcal{P}}_t$ sin cambios. Si no, usar la siguiente técnica de clustering para reducir la población de $\overline{\mathcal{P}}_t$ a tamaño $\overline{n}$:

1. Inicialmente cada solución pertenece a un cluster o clase diferente, es decir $\mathcal{C}_k = \{k\}$. Por lo tanto, si tenemos $\overline{n}'$ individuos en $\overline{\mathcal{P}}_t$ el conjunto de todos los cluster será $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{\overline{n}'}\}$
2. Si $|\mathcal{C}| <= \overline{n}$, ir al punto 5. Si no, ir al punto 3.
3. Para cada par de clusters $\mathcal{C}_k$ y $\mathcal{C}_s$ calcular su distancia-cluster de la siguiente manera:
$$D_{ks} = \frac{1}{|\mathcal{C}_k||\mathcal{C}_s|} \sum_{i \in \mathcal{C}_k, j \in \mathcal{C}_s} d(i,j) \tag{4.5}$$
4. Encontrar el par ($\mathcal{C}_p$,$\mathcal{C}_q$) que haya obtenido menor distancia-cluster. Unir $\mathcal{C}_p$ con $\mathcal{C}_q$ y formar un solo cluster entre ambos. Esto reduce el número de clusters en uno. Regresar al punto 2.
5. Elegir una solución de cada cluster para mantener y eliminar a las demás. La solución que tiene menor distancia a las demás soluciones del cluster puede ser elegida como representante del cluster.

Dado que en este último punto eliminamos a todas menos una solución por cada cluster la población externa tendrá a lo sumo $\overline{n}$ individuos. La población resultante de este paso pasa a ser $\overline{\mathcal{P}}_{t+1}$.

Paso 5: Asignar un valor de fitness a cada solución i de la población externa $\overline{\mathcal{P}}_{t+1}$ utilizando la ecuación:

$$S_i = \frac{n_i}{n+1} \tag{4.6}$$

donde n_i es el número de individuos $j \in \mathcal{P}_t$ que son dominados por i.

Asignar un valor de fitness, S_i, a cada individuo j de $\mathcal{P}_t$ con la siguiente ecuación:

$$\mathcal{F}_j = 1 + \sum_{i \in \overline{\mathcal{P}}_t \wedge i \preceq j} S_i \tag{4.7}$$

Al sumar 1 en este cálculo de fitness estamos asegurando que cualquier individuo de $\mathcal{P}_t$ posea un valor de $\mathcal{F}$ mayor que los valores S_i de sus correspondientes dominadores en $\overline{\mathcal{P}}_t$. Nótese que menores valores de $\mathcal{F}_j$ corresponden a mejores individuos; por lo tanto, el valor de fitness debe ser utilizado de tal forma que menores valores correspondan a mayores probabilidades de reproducción.

Paso 6: Aplicar sobre la población $\overline{\mathcal{P}}_{t+1} \cup \mathcal{P}_t$, de tamaño $\overline{n} + n$, un operador de selección, un operador de cruzamiento y un operador de mutación para generar la nueva población $\mathcal{P}_{t+1}$ de tamaño n. El operador de selección aplicado será "torneo binario" en base a los valores de fitness asignados en el paso 5. La principal diferencia con respecto al concepto de *fitness sharing clásico* es que, al usar el mecanismo de asignación de fitness propuesto en el paso 5, los nichos están definidos en términos de dominancia de Pareto y no en términos de distancia.

Desventajas: El algoritmo exige especificar un parámetro extra, $\overline{n}$. Si es muy grande comparado con n se perderá la diferencia entre tratar con dos poblaciones (la normal y la externa) y tratar con una. Por otro lado, si es muy chico, se perderá el efecto de elitismo (ya que muchas soluciones buenas pueden tener que ser eliminadas para lograr el pequeño cupo) además muchas soluciones de la población normal no serán dominadas por ninguna solución de la población externa. Los investigadores han usado una tasa de 1 a 4 entre la población externa y la normal (4 veces más grande la normal que la externa). Finalmente, los individuos dominados por el mismo número de miembros de la población externa tienen el mismo fitness. En particular si la población externa tiene un solo individuo, todos los miembros

de la población tienen la misma categoría independientemente de si uno domina a otro o no dentro de la población interna. Como consecuencia de esto el proceso de selección se comporta como un algoritmo de búsqueda aleatorio. Si hay muchos individuos que no son dominados por otros dentro de la población interna (esto es un gran frente conteniendo muchos individuos) se debería usar información de densidad para decidir con cuales quedarnos, lo cual se utiliza en la población externa a través del clustering pero no en la interna.

Ventajas: Podemos notar que si se encuentra una solución perteneciente al frente óptimo de Pareto, quedará guardada en la población externa a menos que otra solución perteneciente al frente óptimo conduzca a una mejor distribución dentro de las soluciones dominantes. El algoritmo de clustering permite lograr una buena distribución entre las soluciones no dominadas y no requiere parámetros. Para que el esquema encuentre un conjunto de soluciones más diverso se puede fijar cada solución extrema para que permanezcan en un cluster independiente.

Observación: El tiempo de ejecución de la parte de clustering puede llegar a ser $O(m\overline{n}'^2)$ si se programa con sumo cuidado y si $\overline{n}' = n$ la complejidad de ejecución total puede ser $O(mn^2)$.

- **Strength Pareto Evolutionary Algorithm 2: SPEA2.** Este algoritmo es una mejora propuesta por Zitzler *et al.* para el SPEA [ZLT01]. A diferencia de la primera versión, el SPEA2 utiliza una estrategia de asignación de fitness más refinada que incorpora información sobre la densidad. Además, en SPEA2 el tamaño de la población externa no varía como en SPEA, si no que cuando la cantidad de individuos no dominados no es suficiente para llenar la población externa, se la completa con individuos dominados. Otra diferencia es que el método de clustering, utilizado en SPEA cuando la población de individuos no dominados excede el número máximo de individuos permitidos en la población externa, se reemplazó por un método de reducción similar pero que no pierde puntos extremos. Finalmente, en el proceso de selección para reproducción de este esquema solamente participan miembros de la población externa. Los pasos básicos que realiza el algoritmo son los siguientes:

  Paso 1: Generar una población inicial $\mathcal{P}_t$ y crear la población externa vacía $\overline{\mathcal{P}}_t$ para $t = 0$.

  Paso 2: Calcular el fitness de cada individuo. Para lo cual para cada individuo j

en $\mathcal{P}_t$ y $\overline{\mathcal{P}}_t$ se calcula:

$$S(j) = |\{q | q \in \mathcal{P}_t \cup \overline{\mathcal{P}}_t \wedge j \prec q\}| \tag{4.8}$$

En palabras $\mathcal{S}(j)$ es la cantidad de individuos q pertenecientes a $\mathcal{P}_t \cup \overline{\mathcal{P}}_t$ que son dominados por j. En base a este valor para cada individuo i se calcula:

$$R(i) = \sum_{j \in \mathcal{P}_t \cup \overline{\mathcal{P}}_t, j \prec i} S(j) \tag{4.9}$$

Podemos ver que $R(i)$ es una medida que aumenta con los individuos que dominan a i y con la cantidad de individuos dominados por los individuos que dominan a i. Para evitar el problema que presenta la primera versión del SPEA cuando la mayoría de los individuos de $\mathcal{P}_t \cup \overline{\mathcal{P}}_t$ no dominan a otro compañero, el algoritmo SPEA2 incorpora en el fitness una medida de densidad con la siguiente adaptación del método de k-ésimos vecinos más cercanos:

1. Para cada individuo i en $\mathcal{P}_t \cup \overline{\mathcal{P}}_t$ calcular y ordenar crecientemente las distancias a todos los individuos j en $\mathcal{P}_t \cup \overline{\mathcal{P}}_t$.
2. El k-ésimo elemento de la lista para el individuo i es una estimación de densidad para i, la cual se nota σ_i^k (k generalmente se usa igual a la raíz cuadrada del tamaño de la muestra, en este caso $\sqrt{n + \overline{n}}$).
3. Con el valor σ_i^k calcular la densidad como: $D(i) = \frac{1}{\sigma_i^k + 2}$. En base a esta medida, cuanto más cerca esté el k-ésimo vecino de i más densa será la zona que rodea a i, ya que si $D(i)$ es más cercano a $1/2$ entonces σ_i^k es más chico.

El fitness de cada individuo será:

$$F(i) = R(i) + D(i) \tag{4.10}$$

Paso 3: Copiar los individuos no dominados en $\mathcal{P}_t \cup \overline{\mathcal{P}}_t$ a $\overline{\mathcal{P}}_{t+1}$.
Si $|\overline{\mathcal{P}}_{t+1}| < \overline{n}$, se copian a $\overline{\mathcal{P}}_{t+1}$ los mejores '$\overline{n} - |\overline{\mathcal{P}}_{t+1}|$' individuos dominados en $\mathcal{P}_t \cup \overline{\mathcal{P}}_t$. Estos individuos se pueden obtener ordenando $\mathcal{P}_t \cup \overline{\mathcal{P}}_t$ en base al valor $F(i)$ de los individuos y tomando los primeros '$\overline{n} - |\overline{\mathcal{P}}_{t+1}|$' con valor $F(i) \geq 1$.
Si $|\overline{\mathcal{P}}_{t+1}| > \overline{n}$, se elige para eliminar al individuo i tal que $i \leq_d j$ para todo $j \in \overline{\mathcal{P}}_{t+1}$ con

$$i \leq_d j : \Leftrightarrow \quad \forall\ 0 < k < |\overline{\mathcal{P}}_{t+1}| : \sigma_i^k = \sigma_j^k \quad \vee$$
$$\exists\ 0 < k < |\overline{\mathcal{P}}_{t+1}| : [(\ \forall\ 0 < l < k : \sigma_i^l = \sigma_j^l\ ) \wedge \sigma_i^k < \sigma_j^k]$$

σ_i^k es la distancia de i al k-ésimo vecino más cercano en $\overline{\mathcal{P}}_{t+1}$. En resumen esta definición significa que se elige al individuo que tiene la menor distancia a otro vecino. Si hay varios individuos con igual distancia mínima, se desempata teniendo en cuenta la segunda menor distancia. El proceso de selección para eliminación se repite hasta que $|\overline{\mathcal{P}}_{t+1}| = \overline{n}$.

Paso 4: Realizar los testeos de convergencia.

Paso 5: Aplicar el operador de selección por torneo sobre los individuos de $\overline{\mathcal{P}}_{t+1}$. Aplicar recombinación y mutación a los individuos seleccionados y asignar la población resultante a $\mathcal{P}_{t+1}$. Aumentamos t en 1 y repetimos desde ***Paso 2***.

Desventajas: Al igual que en la primera versión tenemos el parámetro extra $\overline{n}$.

Ventajas: Este algoritmo ha presentado mejor comportamiento ante problemas altamente dimensionales comparado con otros algoritmos tales como su antecesor SPEA, PESA e incluso en algunos problemas de prueba clásicos ha funcionado mejor que el NSGA-II. Las mejoras más rescatables tienen que ver con la distribución del último frente de Pareto y con la convergencia hacia mejores soluciones. Asegura la supervivencia de las mejores soluciones al mismo tiempo que mantiene una buena diversidad por medio del algoritmo de reducción de la población externa, el cual no requiere parámetros adicionales. No se presenta el problema de comportamiento simil-random cuando hay pocos individuos no dominados.

Observación: La complejidad del tiempo de ejecución puede llegar a ser del $O(m^3)$, para $m = n + \overline{n}$ y estará dado por el método de reducción de la población externa. Sin embargo, los autores aseguran que el tiempo promedio será del $O(m^2 log(m))$ ya que los individuos difieren de los demás con respecto al segundo o tercer vecino más cercano.

- **Non-dominated Sorting Genetic Algorithm II: NSGA-II.** El NSGA-II fue sugerido por Deb *et al.* en el año 2000 [DAPM00] como una mejora del NSGA. Ellos destacan que si bien no existe demasiada similitud entre las dos versiones decidieron mantener el nombre. En el NSGA-II se crea una población de descendientes ($\mathcal{Q}_t$) a partir de la población de padres ($\mathcal{P}_t$) y luego ambas poblaciones se unen y forman una población de tamaño $2n$ ($\mathcal{R}_t$). Los individuos de $\mathcal{R}_t$ se ordenan en base a los frentes de no-dominación. Una vez terminado el orden de $\mathcal{R}_t$ por frentes, se comienza

a armar la nueva población ($\mathcal{P}_{t+1}$) completándola con individuos de $\mathcal{R}_t$ comenzando por el primer frente, siguiendo con el segundo y así hasta alcanzar su tamaño de n individuos. Ya que el tamaño de $\mathcal{R}_t$ es $2n$ no encontraremos lugar para todos esos individuos en $\mathcal{P}_{t+1}$ cuyo tamaño es n. Más precisamente, el último frente que intente incorporarse antes de que ya no haya más espacio en $\mathcal{P}_{t+1}$ probablemente tenga una cantidad de individuos mayor a la cantidad de lugares libres aún en $\mathcal{P}_{t+1}$. Esta situación implica que debemos elegir qué individuos de dicho frente debemos incorporar y cuáles debemos descartar. A diferencia de la estrategia de *sharing* utilizada en el algoritmo NSGA la cual requiere especificar el parámetro σ, el NSGA-II utiliza una técnica de distancia que mide el agolpamiento de los individuos de ese frente (denominada por los autores *crowding distance*). Otra diferencia del NSGA-II con la primera versión es que el método de selección empleado es un operador de selección por torneo modificado para que tenga en cuenta el atributo de crowding distance. Los pasos que sigue el algoritmo son los siguientes:

Paso 1: Crear una población inicial $\mathcal{P}_t$ de tamaño n. Identificar los frentes o niveles de no-dominación. Asignar a cada individuo un fitness igual a su número de frente donde el número 1 es el frente de no dominados de toda la población.

Paso 2: Generar la población de hijos $\mathcal{Q}_t$ utilizando torneo binario. Seleccionar dos individuos, i y j, si i tiene un número de frente de no-dominación menor, entonces gana i. Si ambos pertenecen al mismo frente de no-dominación, gana la solución con mayor *crowding distance*. La *crowding distance* de un frente $\mathcal{F}$ se calcula en estos dos pasos:

1. Para cada $i \in \mathcal{F}$ setear $d_i = 0$. Para cada función objetivo f_k $(k = 1, 2, \ldots, m)$ ordenar a los individuos de mejor a peor (para que sea más eficiente es conveniente llevar, para cada objetivo, un orden de los índices de los individuos y no un orden de individuos).
2. Para cada objetivo asignar una distancia infinita a las soluciones de los extremos de la lista correspondiente.

Paso 3: Crear una población, $\mathcal{R}_t$, combinando $\mathcal{P}_t$ con $\mathcal{Q}_t$, es decir $\mathcal{R}_t = \mathcal{P}_t \cup \mathcal{Q}_t$.

Paso 4: Ordenar $\mathcal{R}_t$ de acuerdo al orden de no-dominación. Las soluciones del mejor frente, $\mathcal{F}_1$, serán las mejores soluciones de $\mathcal{P}_t \cup \mathcal{Q}_t$. Notemos que teniendo en cuenta tanto la población de padres como la de hijos, estamos asegurando el elitismo.

Paso 5: Si el tamaño del frente $\mathcal{F}_1$ es menor que n, todos los miembros de $\mathcal{F}_1$ pasan a formar parte de la nueva población, $\mathcal{P}_{t+1}$. El resto de los miembros de $\mathcal{P}_{t+1}$ se eligen consecutivamente de los frentes de no-dominados siguientes, es decir, primero todos los individuos del frente $\mathcal{F}_2$, luego los de $\mathcal{F}_3$ y así hasta que ya no entren más frentes. Sin embargo, si $\mathcal{F}_j$ es el último frente del cual estamos considerando los individuos y no entra completamente en los huecos disponibles, debemos decidir cuáles de sus individuos serán elegidos para formar parte de la nueva población. Para tomar esta decisión, se usa un criterio basado en el operador de comparación de *crowding distance*, por medio del cual se favorece a las regiones menos densamente pobladas. Luego de finalizar este paso, se aumenta t en 1 y se repite a partir del ***Paso 2***.

Desventajas: Si bien es elitista, cuando el tamaño del primer frente se torna mayor que el tamaño de la población, perdemos algunas buenas soluciones. No es tan fácil de implementar como otros algoritmos evolutivos.

Ventajas: La ventaja principal es que la medida de densidad utilizada entre los vecinos de un mismo frente no requiere el seteo de ningún parámetro extra (que sí hace falta en otros como NSGA, MOGA o NPGA). Otra ventaja es que se preservan las mejores soluciones.

Observación: El tiempo de ejecución es del $O(mn^2)$ para efectuar el orden de R_t en el primer paso. Y gracias a que los demás pasos pueden implementarse con un tiempo menor, este es el orden que gobierna al algoritmo.

	no-Pareto (Agregativos)	
Priori	Basados en orden	Lexicográfico
	Escalares	Lineales
		No lineales
	Pareto	
Posteriori	Primera generación	MOGA
		NPGA
		NSGA
	Segunda generación	SPEA
		SPEA2
		NSGA-II

Tabla 4.1: Resumen de técnicas vistas en el presente capítulo y su ubicación dentro de las distintas clasificaciones. Los algoritmos cuyos nombres se encuentran en negrita son los que se utilizaron dentro de las arquitecturas implementadas en la presente tesis.

Existen otras aproximaciones de esta segunda generación de algoritmos evolutivos. Por ejemplo, otros algoritmos evolutivos conocidos son el Pareto Archived Evolution Strategy (PAES) [KC99] y el Indicated-Based Evolutionary Algorithm (IBEA) [ZK04]. Sin embargo, en este capítulo nos limitamos a ver las técnicas que se consideraron básicas para entender las infraestructuras implementadas durante la presente tesis. La tabla 4.1 muestra un resumen de los algoritmos mencionados, su ubicación dentro de las clases de técnicas destinadas a resolver problemas de optimización y su clasificación según Pareto o no-Pareto. Los algoritmos resaltados en negrita son los que se utilizaron dentro de las infraestructuras desarrolladas durante esta tesis.

Capítulo 5

Algoritmos evolutivos desarrollados para minería de datos

Durante el presente capítulo se presentan los algoritmos evolutivos desarrollados en el área de minería de datos estructurados. Más específicamente, se explican las propuestas diseñadas durante la tesis para abordar el problema de *selección de características* (FS por sus siglas en Inglés de Feature Selection), el cual es uno de los importantes desafíos en el área de minería de datos. Dado que los algoritmos evolutivos se distinguen por ser buscadores eficientes y poderosos, se propone el uso de los mismos en la primera etapa de una infraestructura más compleja, la cual consiste en un proceso de dos grandes etapas. La primera de ellas está destinada a realizar una tarea de búsqueda preliminar que permite disminuir el enorme espacio de búsqueda que suelen presentar este tipo de problemas. Para realizar dicha búsqueda, esta etapa utiliza un método de wrapper en el cual se consideran distintos algoritmos de búsqueda y distintos criterios para la evaluación de la capacidad descriptiva, mientras que la segunda etapa, está destinada a mejorar los resultados encontrados por la etapa previa. La incorporación de los algoritmos evolutivos a la primera etapa se realiza de forma gradual, considerando como primera instancia una versión simplificada del problema original en la cual se emplean algoritmos evolutivos mono-objetivo. En esta primera versión, se pretende encontrar subconjuntos de descriptores de tamaño fijo que posean un error de predicción suficientemente cercano al óptimo. Posteriormente se aborda una versión más compleja en la cuál, además del error de predicción, se intenta optimizar el número de descriptores a seleccionar. A partir de esto, el primer objetivo de este capítulo es el diseño, desarrollo y evaluación de algoritmos

evolutivos para colaborar con la primera etapa de la infraestructura, incorporando dichos algoritmos como parte del método de wrapper. La evaluación de los distintos algoritmos evolutivos implementados se realiza desde el punto de vista del desempeño de los mismos al alcanzar la etapa final de la infraestructura. Es decir, los algoritmos no se evalúan, por ejemplo, con respecto a diversidad genética, sino que se evalúan con respecto a la calidad que logra el proceso completo de dos fases al utilizar los resultados que ellos generan en la primera etapa. Por otra parte, el objetivo práctico final de esta etapa de la tesis es ayudar a descubrir cuáles son los descriptores más relevantes para modelar una determinada propiedad y, simultáneamente, determinar cuál es el número aproximado de descriptores que se necesitan para lograr un modelo aceptable en un problema de gran interés, el de análisis de relaciones cuantitativas estructura-propiedad y el de análisis de relaciones cuantitativas estructura-actividad (QSPR y QSAR respectivamente por sus siglas en Inglés). Este es uno de los problemas más representativos de las dificultades que puede involucrar la tarea de selección de características.

5.1 Selección de características (Feature Selection)

Típicamente, los científicos desean estudiar los efectos de determinados factores con respecto a una variable objetivo e incluso predecir el valor que tendrá dicha variable en base a los factores observados. Para esto, generalmente, se utiliza una representación en forma matricial para los datos experimentales, donde las columnas significan características observadas y las filas representan los valores obtenidos para dichas características en las muestras o entidades observadas. Como ejemplo simple, la figura 5.1 muestra una representación gráfica simplificada de un conjunto de muestras (filas) y los respectivos valores para las características consideradas (columnas). La tarea de selección de características se puede ver como un proceso por medio del cual se puede reducir el número de características o variables presentes en un conjunto de datos. En esencia el objetivo es remover aquellas características que resultan irrelevantes o aquellas que resultan redundantes para modelar determinado comportamiento. El número de características que participan es crucial al modelar una actividad, dado que influye en la complejidad del modelo. Dicho modelo nos permitirá predecir el comportamiento de nuevos valores experimentales sin necesidad de poseer el valor real de la propiedad o comportamiento estudiado. Reducir el número de características presentes en el modelo nos permite disminuir la cantidad de tiempo computacional requerido al usar el modelo, construir un modelo más general

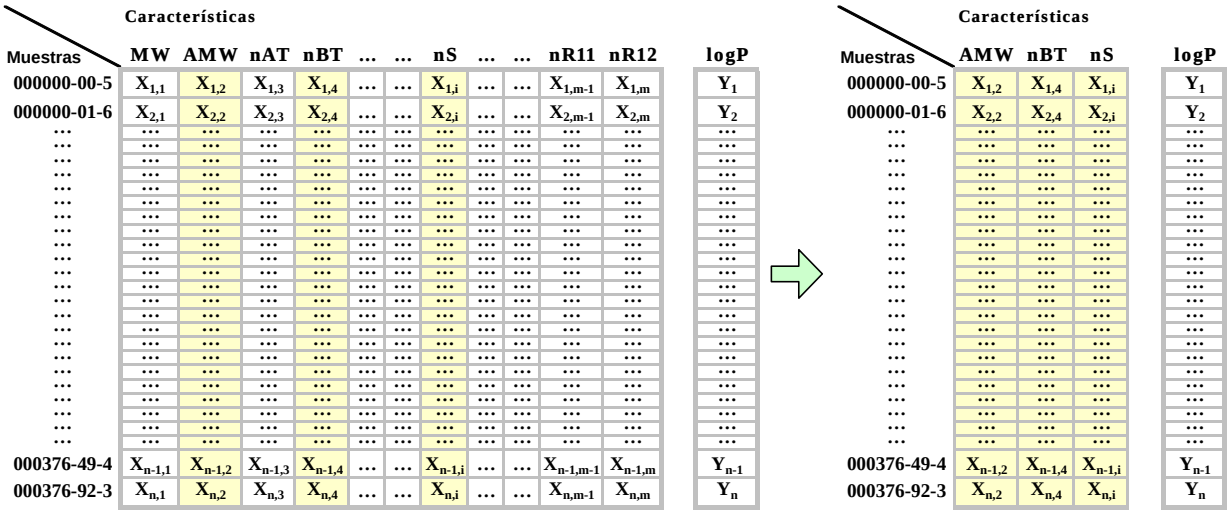

Figura 5.1: Ejemplo simple de un proceso de selección de características sobre una base de datos.

y generar modelos más simples de comprender, permitiendo que los especialistas identifiquen con menos dificultad las relaciones entre dichas características y la actividad o situación bajo estudio.

Típicamente, en un método de selección de características se pueden identificar cuatro pasos básicos:

1. el *procedimiento de generación* de los subconjuntos de características,
2. la *función de evaluación* que determina el desempeño de cada subconjunto bajo análisis,
3. el *criterio de detención* que decide cuando detener la búsqueda, y
4. un *procedimiento de validación* que estudia la validez del modelo final.

La figura 5.2 muestra estos cuatro pasos. El procedimiento de generación es un proceso de búsqueda que, básicamente, genera subconjuntos de características los cuales serán ponderados por la función de evaluación. Este primer paso puede comenzar con: un conjunto vacío, el conjunto total de características o un conjunto de características aleatorio que sea subconjunto del conjunto completo. En los primeros dos casos, la generación de subconjuntos se realiza agregando o quitando respectivamente características en forma iterativa y determinista, mientras que en el tercer caso pueden tanto agregarse como quitarse características y las mismas se pueden seleccionar de manera aleatoria. La función de evaluación determina la calidad de un subconjunto de características que devuelve el proceso de generación. Este valor de calidad permite realizar la comparación de dicho

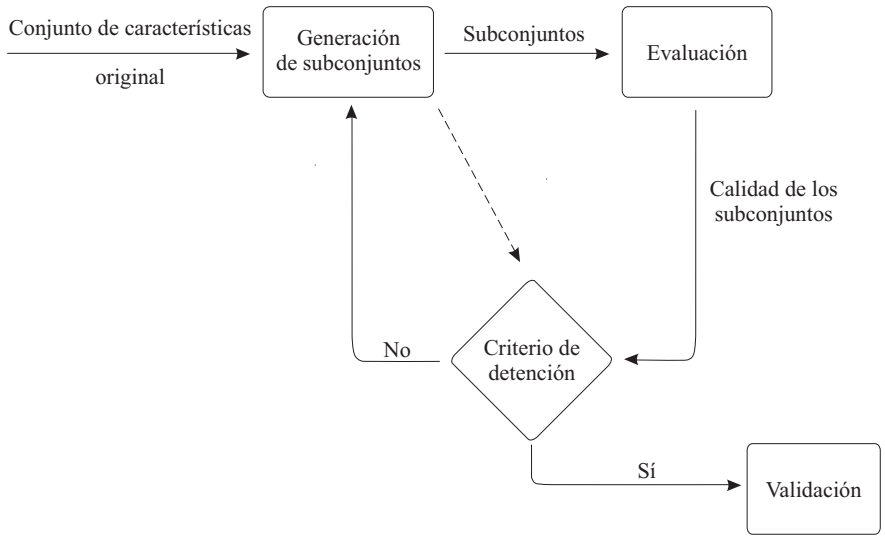

Figura 5.2: Pasos básicos identificados en la tarea de selección de características. El trazo discontinuo representa la posible incidencia del proceso de generación de subconjuntos en el criterio de detención.

subconjunto de características con otros subconjuntos a fin de determinar cuál de ellos es preferible. Dado este proceso iterativo de generación, evaluación y comparación, se requiere un criterio de detención que permita identificar la culminación del ciclo. El proceso de generación y la función de evaluación pueden tener gran influencia sobre este criterio. Un criterio de detención basado en el procedimiento de generación puede contemplar, por ejemplo, si se alcanzó un número de características seleccionadas predefinido o si se alcanzó cierto número de generaciones. Por otra parte, un criterio de detención basado en la función de evaluación puede contemplar, por ejemplo, si luego de cierto número de iteraciones no se logró una mejora en la calidad de los subsecuentes subconjuntos de características o si se alcanzó algún subconjunto que sea óptimo de acuerdo con alguna de las funciones de evaluación. Una vez que el criterio de detención se cumple, el proceso entrega como resultado uno o varios subconjuntos candidatos. Finalmente, el procedimiento de validación no es parte del proceso de selección de características en si mismo sino que es un paso posterior que permite la validación del método. En este último paso se realizan distintos análisis, pruebas y comparaciones con resultados previos o con resultados obtenidos por otros métodos de FS.

En base a lo anterior, la *selección de características* puede verse como el proceso por medio del cual se selecciona un subconjunto del conjunto original de variables tal que sirva para construir un buen predictor. Algunas de las cuestiones importantes dentro de

este proceso son:

- *El ámbito en el cual se aplica el proceso (supervisado o no supervisado).* En un ambiente supervisado se utilizan datos de entrenamiento para inferir un modelo y luego se aplica el modelo obtenido sobre datos de prueba con el fin de determinar su desempeño, mientras que en un ambiente no supervisado, no se cuenta con datos de entrenamiento para inferir el modelo por lo cual debe ser construido en base a otros criterios. En la presente investigación el trabajo se desarrolló en un ambiente supervisado.
- *El tipo de método de selección de características utilizado.* Los métodos de FS suelen dividirse en filtros, wrappers y métodos integrados. Los filtros realizan el proceso de selección de acuerdo a características de los datos (p. ej. baja varianza o variables correlacionadas). Los wrappers utilizan una técnica de aprendizaje automatizado como caja negra, como paso de preprocesamiento para notar conjuntos de variables en términos de su habilidad predictiva. Estos tipos de métodos pueden dividirse internamente en: (1) búsqueda de características y (2) evaluación del subconjunto de características. El primer componente es responsable de realizar la búsqueda combinatoria entre los posibles subconjuntos de características, mientras que el segundo componente evalúa la eficacia del subconjunto seleccionado y en base a esto guía la búsqueda de características. Finalmente, los métodos integrados llevan a cabo el proceso de FS en la etapa de entrenamiento de un método de aprendizaje automatizado y, usualmente, son específicos para dicho método de aprendizaje [GE03, DGWC07]. Como se dijo en el primer párrafo del presente capítulo, los métodos de selección de características utilizados en esta tesis caen dentro de los métodos de wrapper. En particular, se utilizan distintos algoritmos evolutivos para realizar la búsqueda y distintos métodos para evaluar la capacidad predictiva de cada subconjunto que resulta seleccionado.
- *La estrategia de búsqueda utilizada.* La tarea de selección de características puede verse como un problema de búsqueda en el cual el espacio de búsqueda estará conformado por todos los posibles subconjuntos de características. Por ejemplo para un conjunto de tres variables $\{a_1, a_2, a_3\}$, tenemos las siguientes posibilidades: $\{\}$, $\{a_1\}$, $\{a_2\}$, $\{a_3\}$, $\{a_1, a_2\}$, $\{a_1, a_3\}$, $\{a_2, a_3\}$, $\{a_1, a_2, a_3\}$, es decir, $8 = 2^3$ o 2^N (con $N = 3$) posibilidades. Este problema es relativamente trivial si el espacio de búsqueda es pequeño, ya que podemos analizar todos los subconjuntos en cualquier orden por medio de una búsqueda exhaustiva en poco tiempo. Sin embargo, es

común que el número de características involucradas sea mayor, en los casos de estudio abordados durante esta tesis el número de características es $N = 73$ y $N = 1502$. Si bien el número de posibles subconjuntos a ser evaluados es finito, de todas formas es una cantidad sumamente grande como para realizar un análisis iterativo a través de todos los casos. Debido a esto, la estrategia de búsqueda es muy importante. En general, el proceso de búsqueda exhaustiva se deja de lado y se utiliza en su lugar un proceso heurístico que utilice cierta inteligencia para la selección de los subconjuntos. Los métodos de búsqueda secuencial, en general, usan técnicas greedy. Dadas las características de este problema, en esta tesis se utilizaron como herramienta de búsqueda distintos algoritmos evolutivos.

- *La medida de calidad utilizada para ponderar cada subconjunto de características seleccionado.* La evaluación de cada subconjunto usualmente se lleva a cabo por medio de un método predictor. En los algoritmos desarrollados en esta tesis, se utilizaron varios de los métodos predictores más populares, tales como: árboles de decisión, k-vecinos más cercanos y cuadrados mínimos lineales y no lineales. La combinación del método predictor con el algoritmo evolutivo debe ser tal que no comprometa el costo computacional, que de por sí es elevado; por esta razón, los métodos predictores utilizados para evaluar cada subconjunto durante la evolución del algoritmo de búsqueda fueron métodos relativamente eficientes y no necesariamente muy exigentes con respecto a la capacidad predictiva de cada subconjunto, en parte para no agregar costo computacional y en parte para no caer en un problema de sobreajuste.
- *Evaluación del desempeño del predictor.* Finalmente, una vez encontrado un modelo para el conjunto de datos, se utiliza un criterio de evaluación para determinar la validez del mismo y para compararlo con otros modelos existentes. Es importante notar, que para la evaluación final del predictor, se debe considerar un conjunto de testeo independiente. Por este motivo, para la evaluación de los algoritmos presentados en esta tesis, se realizaron experimentos de validación externa para los modelos obtenidos en los cuales se utilizó un conjunto de datos no superpuesto con el conjunto de entrenamiento.

5.2 Antecedentes en selección de características

Existen muchos trabajos que demuestran la importancia y la variedad de investigaciones realizadas en torno al problema de selección de características. Dos de los algoritmos más conocidos y simples para llevar a cabo FS son, *selección secuencial progresiva* (SFS por sus siglas en Inglés de Secuential Forward Selection) y *selección secuencial inversa* (SBS por sus siglas en inglés de Secuential Backward Selection). En el SFS, introducido por Whitney [Whi71], se comienza con un subconjunto vacío de características al cual se van agregando características de a una por vez (agregando la que más contribuye), si el subconjunto resultante al incorporar una característica es mejor que el subconjunto previo, entonces se reemplaza el peor por el nuevo. En el caso del SBS el proceso es inverso, es decir, se comienza con el conjunto de características completo y se van eliminando características de a una por vez (eliminando la que menos contribuye). Estos algoritmos son simples y eficientes con respecto al costo computacional; sin embargo, son pobres a la hora de detectar interacciones entre características. Estos métodos suelen ser conocidos como métodos de regresión paso a paso (o stepwise regression - SWR). En algunos esquemas primero se determina si alguna de las variables que ha sido seleccionada en los pasos previos puede eliminarse del modelo. Luego, si ninguna de esas variables se puede descartar, se evalúa la posibilidad de agregar una de las variables no incluidas. De esta forma, una variable que fue agregada en un paso puede ser eliminada posteriormente. Uno de los esquemas más conocidos que abordan interacción entre características es Relief, propuesto por Kira y Rendell en 1992 [KR92]. Relief es un algoritmo basado en pesos inspirado en aprendizaje basado en ejemplos. Calcula un valor de peso para cada una de las N características disponibles usando una muestra al azar de un conjunto total de muestras (X). Relief selecciona una muestra de entrenamiento al azar (x_i) y, usando distancia Euclideana, encuentra la muestra de igual clase más próxima (x_i^+) y la muestra de diferente clase más próxima (x_i^-). Estas muestras se usan para actualizar el valor de peso de cada característica usando la diferencia entre x_i, x_i^+ y x_i^-. El procedimiento se repite un número predefinido de veces, luego de lo cual, se consideran como relevantes aquellas características con un peso mayor a cierto umbral. Este algoritmo es eficiente computacionalmente, simple y tolerante a ruido en los datos. Una debilidad reportada por los autores es que Relief no contribuye a eliminar redundancia en los datos. ReliefF [Kon94] extiende el algoritmo original para poder contemplar problemas de más de dos clases.

Otro tipo de algoritmo que se ha usado en FS es Simulated Annealing (SA) [KGV83, MZ06]. Esta técnica se inspira en las técnicas de metalúrgica de calentamiento y paulatino enfriamiento de metales. Este procedimiento está destinado a aprovechar la excitación y desorganización inicial de los átomos en el material caliente para encontrar configuraciones fuertes y estables durante un enfriamiento lento. En esta técnica se definen tres elementos principales:

- el esquema de enfriamiento (la temperatura inicial, la temperatura final y una constante de enfriamiento),
- la función de evaluación de los posibles subconjuntos de características, y
- una función vecino que, dada una temperatura y la solución actual, devuelve una nueva solución "cercana". La función trabaja de tal forma que a mayor temperatura el vecindario de cercanía es más amplio que a menor temperatura. Si la nueva solución es mejor, entonces reemplaza a la anterior.

Los métodos de FS basados en SA caen dentro de la clase de procedimientos de generación aleatorios. Dentro de esta categoría también se encuentran los métodos que utilizan algoritmos genéticos. En este caso, se suele codificar cada posible elemento del espacio de búsqueda (cada subconjunto de descriptores) como un string binario, donde los unos de la cadena representan al subconjunto de características seleccionadas. Además, se debe establecer una función de aptitud a fin de evaluar la calidad de cada uno de dichos subconjuntos. Esta función puede utilizar otras técnicas de aprendizaje automatizado como árboles de decisión, algoritmos de regresión, redes neuronales y muchos otros, implicando posiblemente fases de entrenamiento y testeo dentro de la evaluación de cada individuo. Finalmente, es necesario definir los operadores genéticos que serán utilizados para realizar el intercambio de información entre los individuos de una generación. El primer paso de la evolución consiste en generar de forma aleatoria una población inicial de subconjuntos de descriptores. Posteriormente, se aplican los operadores genéticos definidos y se genera una nueva población. Los nuevos subconjuntos de descriptores se evalúan con la función de aptitud predefinida y los más aptos son seleccionados heurísticamente para poblar la próxima generación. Este ciclo de creación, evaluación y selección de nuevos individuos se repite hasta que se alcanza el criterio de finalización. En la sección 5.4 se reportan algunos de los trabajos que han usado algoritmos evolutivos en selección de características.
Se pueden encontrar revisiones más detalladas sobre distintas estrategias propuestas para el problema de FS en [DL97, LM07, SIL07].

5.3 Escenarios de alcance

Como es de esperar, los problemas en los que es necesario aplicar FS involucran gran cantidad de características y poseen un patrón de comportamiento complejo desconocido. A continuación se mencionan algunos de los escenarios importantes donde se suele aplicar selección de características para encontrar un modelo simplificado del conjunto de datos original.

Análisis de niveles de expresión en datos genómicos

Los microarrays (micro matrices) son chips de silicio que contienen medidas de los niveles de expresión de ARNm (ácido ribonucleico mensajero) asociado a decenas de miles de genes simultáneamente. Usualmente, se miden los niveles de expresión del conjunto de genes bajo estudio tomando diferentes muestras y bajo diferentes condiciones. En un microarray típico cada columna representa un gen y cada fila representa una muestra. En base a esto, cada valor m_{ij} es la medida del nivel de expresión del j-ésimo gen para la i-ésima muestra. Los rasgos distintivos típicos de estas matrices de datos incluyen: una gran cantidad de columnas debido al número de genes bajo estudio, pocas filas debido al bajo número de muestras existentes (en el rango de las decenas o pocos cientos) y mucha redundancia entre genes. Dichas particularidades agregan desafíos extras a la tarea de análisis de niveles de expresión de genes. Esta tarea consiste en el estudio de la información genómica con el fin de determinar relaciones entre genes o conjuntos de genes y determinada situación o enfermedad biológica. Una matriz correspondiente a muchos genes y a pocas muestras, en general, conducirá a muchas hipótesis estadísticamente relevantes que serán igualmente válidas al interpretar el conjunto de datos. Por lo tanto, la selección de un subconjunto pequeño de genes, típicamente, ayudará a identificar genes relevantes para el fenómeno bajo estudio y a simplificar la tarea de los biólogos relacionadas al seguimiento de los genes seleccionados. Las técnicas de FS son claramente aplicables a este problema de gran interés y existen trabajos en la literatura que muestran resultados exitosos [MM04, GHGL99, HK05].

Clasificación de documentos

La clasificación de documentos es la tarea de ordenar documentos de acuerdo a un conjunto de categorías. La selección de características puede usarse como paso previo a la tarea de clasificación, usando las palabras como posibles características e identificando

las que mejor resumen a cada conjunto de documentos [BEYT+03, LM07]. En un ambiente supervisado, un algoritmo de aprendizaje automatizado podría aprender las palabras más representativas de cada subconjunto de documentos etiquetados (en el conjunto de documentos de entrenamiento) y luego utilizar dichas características para clasificar documentos no etiquetados. Por otro lado, en un ambiente no supervisado, un algoritmo de selección de características podría aplicar medidas de frecuencia de aparición de las palabras u otras métricas usadas en recuperación de información para identificar palabras que resuman y agrupen documentos. Algunas de las aplicaciones más interesantes de la clasificación automática es la tarea de identificación de spam, el estudio de mercadeo y la tarea de categorización de documentos para facilitar su búsqueda y navegación.

Reconocimiento de caras

Debido a la disponibilidad de nuevas tecnologías en la última década y a intereses comerciales, la tarea de reconocimiento de rostros ha cobrado importancia dentro del área de análisis de imágenes, un hecho que se ve reflejado en las conferencias que abordan este aspecto como tema de interés y en la aparición de técnicas destinadas a resolver este problema, el cual puede resumirse como una tarea de identificación de rostros dentro de un grupo de imágenes. Además de la imagen, los sistemas cuentan con información extra como edad, raza, género, color de piel o expresión facial. De esta manera, el conjunto inicial de características puede contemplar todas las propiedades y pixels de las imágenes. Una imagen pequeña de 85x60 involucraría inicialmente 5100 pixels, cada uno de los cuales puede ser considerado como una característica. Debido a la presencia de tantas características, la tarea de FS se vuelve tan necesaria como en otros problemas altamente dimensionales [GBNT04, SSM02].

Análisis de relaciones cualitativas y cuantitativas de estructura-propiedad y estructura-actividad (QSPR/QSAR) para diseño de drogas

En general, el proceso que sigue una industria farmacéutica al fabricar una nueva droga consiste en una serie de pasos que involucran: testear los compuestos para determinar su efectividad y analizar la forma en que la droga afecta al cuerpo. El segundo paso implica estudiar las capacidades de absorción, distribución, metabolismo, excreción y toxicidad del compuesto en el cuerpo (conocidas como propiedades ADMET o ADME-Tox). Muchos de los compuestos candidatos fallan en las últimas etapas a causa de las

propiedades ADMET, las cuales están relacionadas con la forma en que la droga interactúa con una gran cantidad de macromoléculas y son la principal causa de falla en el diseño de un medicamento. El principal problema reside en la dificultad de conocer la leyes que gobiernan el comportamiento de las propiedades ADMET en el cuerpo humano. Debido a esto, desde fines de los 90s se ha visto un incremento importante en el desarrollo de modelos de predicción para propiedades ADMET que se conocen como modelos *in-sílico* (desarrollados por medio de computadoras). Si bien estos métodos no tienen como fin reemplazar a los experimentos *in vitro*, se sabe que son menos costosos, más veloces y que algunas técnicas computacionales han demostrado tan buena precisión como ensayos experimentales consolidados.
Dada la importancia del análisis de propiedades ADMET, ha aumentado el interés de la comunidad científica y farmacéutica en el análisis de las relaciones cuantitativas estructura-actividad (QSAR) y estructura-propiedad (QSPR). Ambos esquemas comprenden los métodos por medio de los cuales se correlacionan cuantitativamente parámetros de estructura (descriptores) con determinado proceso o actividad biológica bajo estudio (la variable dependiente). Existen muchos trabajos relacionados directamente con QSAR/QSPR que demuestran la gran importancia e interés del problema [SBT02, Tsy08, GPC08, BW09].

5.4 Uso de algoritmos evolutivos en FS

Los algoritmos evolutivos han sido aplicados de distinta forma en el problema de selección de características. Por ejemplo, Leardi *et al.* mostraron en varios trabajos la utilidad de la aplicación de AGs en FS y lograron obtener buenos subconjuntos de descriptores usando cuadrados mínimos parciales (PLS) como método de regresión. Los autores aplicaron el método en cinco conjuntos de datos en los cuales se utilizaron entre 35 y 175 variables al comienzo del AG [LG98].
El algoritmo SET-Gen, propuesto por Cherkauer y Shavlik [CS96], utiliza una función de aptitud que contempla la exactitud y la comprensibilidad del modelo. Para evaluar la capacidad predictiva de los individuos, el SET-Gen utiliza un algoritmo de entrenamiento basado en árboles de decisión y para evaluar la comprensibilidad del modelo utiliza una fórmula que combina la dimensión del árbol subyacente y el número de características seleccionadas.
Inza *et. al* proponen una algoritmo evolutivo denominado FSS-EBNA (Feature Selection by Estimation of Bayesian Network Algorithm) en el cual se evita el uso de los operadores

de recombinación y mutación clásicos y en su lugar utiliza un esquema más complejo basado en redes Bayesianas [ILES00].

En [LTC+01] los autores proponen un método para clasificación de muestras basadas en datos de niveles de expresión genómica. El esquema combina un algoritmo genético con el método de los k vecinos más cercanos para identificar genes que conjuntamente permiten discriminar entre dos tipos de muestras (normales vs. tumorales). Como primer paso se obtienen varios conjuntos de genes por medio del algoritmo genético, a partir de estos subconjuntos se deduce la importancia relativa de cada gen en base a su frecuencia de selección y se arma un modelo con los descriptores más seleccionados globalmente.

En el ámbito más específico de FS para QSAR/QSPR varias propuestas basadas en la combinación de algoritmos evolutivos y otras técnicas de aprendizaje automatizado han demostrado su buen desempeño. So y Karplus proponen el uso de un método híbrido basado en redes neuronales y algoritmos evolutivos. El algoritmo evolutivo selecciona un subconjunto de características que se entrega a la red neuronal y por medio de la red se realiza un modelo multivariado para el conjunto de datos bajo estudio [SK96]. En [KSM03] los autores proponen el uso de un *algoritmo evolutivo de selección local* dentro de una arquitectura que utiliza el EA para la búsqueda y redes neuronales para la evaluación de los subconjuntos que el EA sugiere en cada generación. En este tipo de esquema, la desventaja principal es el tiempo de cómputo requerido por la red neuronal y por ende la baja escalabilidad de la infraestructura subyacente. Bayram *et. al* [BSH+04] y Lavine *et. al* [LDM02] aplicaron esquemas similares, donde los métodos acoplados al algoritmo genético fueron mapas auto-organizados supervisados (sSOM) y análisis de componentes principales (PCA) respectivamente. En [DGWC07] los autores utilizan un esquema basado en AGs, para la selección de un subconjunto de descriptores simultáneamente óptimo para múltiples tipos de modelos, es decir, el esquema procura identificar un único subconjunto que trata de minimizar la degradación en la capacidad predictiva de diferentes tipos de modelos, comparados con modelos construidos usando subconjuntos de descriptores seleccionados por medio de técnicas tradicionales de FS. Además, el esquema evolutivo es usado tanto para regresión como para clasificación. En [WZ03] Wegner y Zell usan una algoritmo genético modificado, el cual utiliza una técnica denominada *grupos de entropía de Shannon* (SEC) para generar la población inicial. Esta técnica acelera el proceso de optimización evolutiva. Sin embargo, en cada generación el conjunto de descriptores seleccionado es evaluado como modelo de regresión utilizando una red neuronal.

5.5 Infraestructura de dos fases propuesta para selección de características

De acuerdo con las particularidades del problema de selección de características y en base a los trabajos relacionados, podemos sustentar que el uso de algoritmos evolutivos es adecuado para colaborar en el proceso de búsqueda dentro de la tarea de selección de características. Además, gracias a que los algoritmos evolutivos constituyen una herramienta fácilmente combinable con otras herramientas de aprendizaje automatizado, es posible considerar un sistema híbrido que se ajuste a la complejidad y necesidades de las etapas involucradas en el proceso de selección de características.

5.5.1 Objetivos generales de investigación

Como hemos visto hasta aquí, en la tarea de desarrollo de modelos de predicción cada modelo se basa en un subconjunto de descriptores donde la relevancia de cada descriptor depende generalmente de los otros, es decir, un descriptor que es inútil por si solo puede resultar muy bueno cuando se lo considera conjuntamente con otros. La infraestructura propuesta tiene por finalidad realizar de manera automática la selección de las características más relevantes en conjuntos de datos en los que participa un número importante de variables (desde varias decenas hasta miles). En particular, se pretende que la metodología sea útil cuando se están considerando muchos descriptores sin importar la complejidad lineal o no-lineal que expresen los datos. En este sentido, es importante notar que el método empleado para la evaluación de los subconjuntos generados no sólo debe ser capaz de identificar tal tipo de relaciones, sino que también debe implicar bajo costo computacional para lograr una rápida evaluación de los subconjuntos de características seleccionadas. El trabajo desarrollado en esta tesis comienza con una primera implementación que considera únicamente el objetivo de maximización de la capacidad predictiva y luego, en su segunda implementación, contempla también la minimización de la complejidad del modelo de predicción. Como objetivos pragmáticos, se persigue el estudio de distintas combinaciones entre algoritmos evolutivos y distintos métodos de predicción, el análisis del grado de dependencia entre conjuntos de datos de distinta complejidad y dichas combinaciones y el uso de la infraestructura propuesta en el problema de FS para QSAR/QSPR.

5.5.2 Arquitectura propuesta

Para llevar a cabo la tarea de selección de características se propone el uso de una metodología de dos fases. La figura 5.3 muestra los componentes principales de la infraestructura de dos fases propuesta. La primera fase está constituida por un método de wrapper en el cual la búsqueda está a cargo de un algoritmo evolutivo y la evaluación de los subconjuntos está a cargo de un algoritmo de regresión o una función estadística. La segunda fase utiliza los subconjuntos de características seleccionadas generados en la

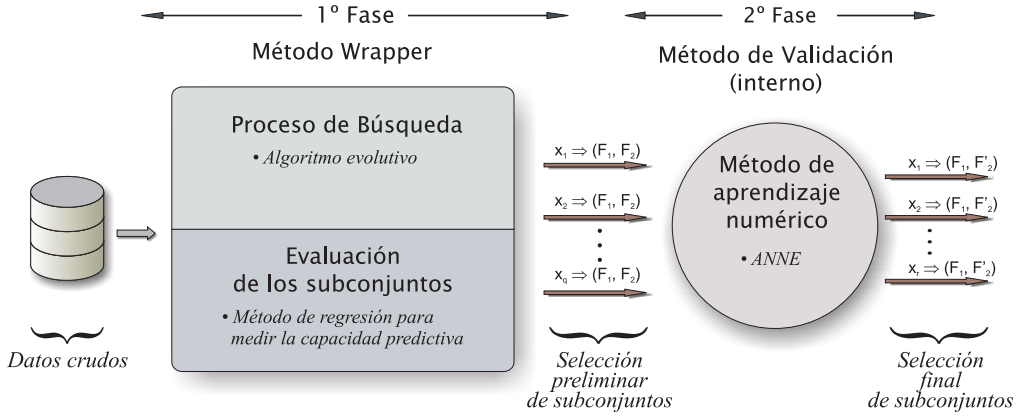

Figura 5.3: Infraestructura de dos fases propuesta.

primera fase y determina cuáles son los más relevantes para predicción. Esta etapa puede considerarse como un *proceso de validación interno* para el método. El algoritmo evolutivo desarrolla su rol dentro de la primera de las dos fases, realizando el barrido del espacio de búsqueda de forma independiente de la función de evaluación de la capacidad predictiva. Esta separación en dos fases permite que la primera fase sea responsable de realizar una búsqueda rápida preliminar y de amplio espectro, logrando abarcar la inmensidad del espacio de búsqueda químico factible. Luego, para lograr una evaluación más estricta de los subconjuntos de características, la salida generada por el wrapper es usada por el método de validación interno, un método de predicción más fuerte y preciso (y tal vez más costoso computacionalmente) que los utilizados dentro del wrapper. Dado que en la segunda fase el espacio de búsqueda es notablemente menor, tenemos la libertad de utilizar un método computacionalmente más riguroso para la evaluación final de los subconjuntos sugeridos por el método de dos fases.

5.5.3 Alcances de esta tesis dentro de la infraestructura propuesta

Dada la complejidad de la arquitectura propuesta, el objetivo específico de esta tesis es el diseño, implementación y análisis de desempeño de los algoritmos evolutivos dentro de la primera fase de la arquitectura. Para llevar a cabo dicha tarea, se realizaron estudios comparativos entre distintos esquemas evolutivos y entre diferentes métodos de evaluación de subconjuntos de características. En este sentido, es importante notar que dado que la segunda fase es la encargada de reportar los subconjuntos finales de características, las comparaciones y el análisis de desempeño debe ser evaluado como un todo y no para cada fase de forma independiente. Por lo tanto, dichos estudios se efectúan en base a los resultados alcanzados por la arquitectura completa y no sólo en base a los alcanzados por la primera fase. Una descripción más detallada de la segunda fase de esta metodología puede ser consultada en [SCVP09].

5.5.4 Estructura interna del wrapper

La figura 5.4 muestra un esquema de la estructura interna del wrapper. Como se puede ver, la primera fase está gobernada internamente por el ciclo del algoritmo evolutivo integrado al proceso de búsqueda del wrapper. Durante este ciclo, el algoritmo evolutivo interactúa permanentemente con el componente de evaluación del wrapper. Este componente permite determinar el valor de aptitud de cada subconjunto a partir del cual tienen lugar los operadores de selección, recombinación y mutación. Durante estas investigaciones se consideraron dos esquemas evolutivos diferentes; el primero de ellos contempla la capacidad predictiva como único objetivo para la evaluación de los subconjuntos, mientras que en el segundo esquema se contempla tanto la capacidad predictiva como la cantidad de características seleccionadas. A continuación se detallan aspectos compartidos por ambas versiones del wrapper y más adelante se detallan los componentes que poseen características particulares para cada versión.

Representación de cromosomas y generación de la población de subconjuntos

Para representar a cada subconjunto de descriptores seleccionados se utiliza una cadena de bits de longitud m, donde m es la cantidad de descriptores considerados a la entrada de la infraestructura. Un valor distinto de cero en la posición i significa que el i-ésimo descriptor ha sido seleccionado dentro de ese subconjunto, mientras que un cero significa que dicha característica no pertenece al subconjunto seleccionado. La figura 5.5

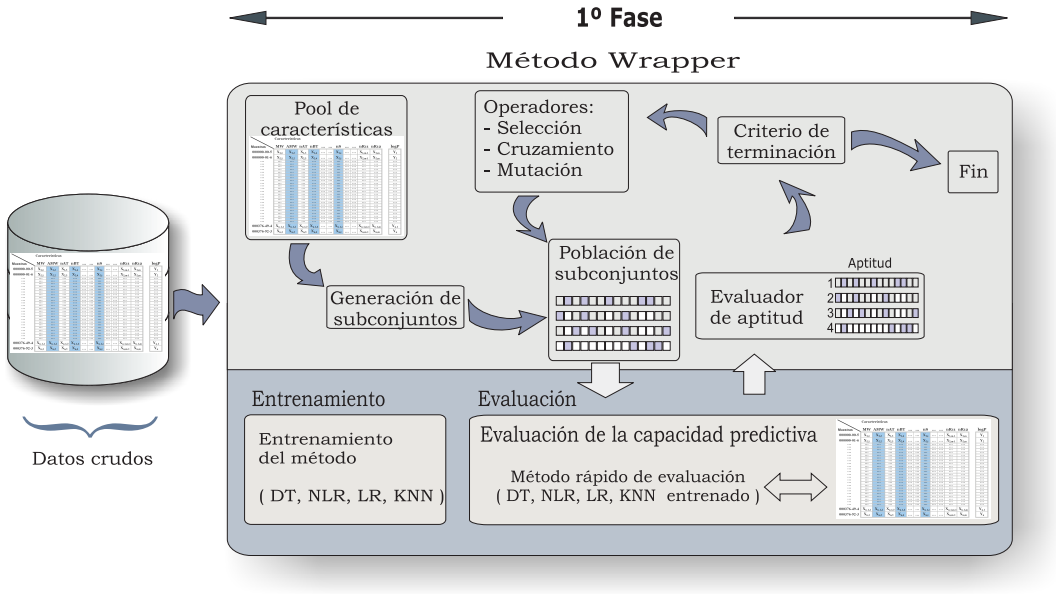

Figura 5.4: Estructura interna de la primera fase de la metodología.

muestra el individuo que representa la selección correspondiente a la figura 5.1.
Para una población de tamaño q, en el primer paso del algoritmo se generan q individuos de manera aleatoria. Cada individuo se construye bit a bit seteando posiciones elegidas al azar y manteniendo las restantes en cero. De esta manera cada individuo es la representación de un subconjunto de descriptores, por lo que en cada generación tendremos q subconjuntos de descriptores seleccionados. Dependiendo del conjunto de datos y de la cantidad de objetivos propuestos, se tuvieron en cuenta situaciones especiales para la generación de cada individuo (p. ej. los descriptores intrínsecamente necesarios fueron seleccionados de forma previa y se restringió el número máximo de características a seleccionar).

0	1	0	1	0	0	1	0	0	0	0
1	2	3	4	...	...	i	...	...	m-1	m

Figura 5.5: Individuo que representa la selección de descriptores de la figura 5.1.

Evaluación de la aptitud de los subconjuntos

Dado que el objetivo principal de todo método de generación de modelos para FS es optimizar la capacidad predictiva, este objetivo fue considerado en ambas implementaciones. Sin embargo, no existe un único método para llevar a cabo tal evaluación. Paralelamente,

para la primera etapa de la arquitectura se busca un método que sea efectivo para determinar la capacidad predictiva y simultáneamente poco costoso a nivel computacional. En base a estas características y con el objetivo de estudiar diferentes métodos de evaluación, se utilizaron cuatro diferentes técnicas de predicción durante la investigación: árboles de decisión (DT), regresión lineal múltiple (MLR), regresión no lineal (NLR) y regresión k-vecinos más cercanos (KNN). Por medio de una de estas técnicas de regresión, el componente de evaluación del wrapper determina el nivel de aptitud de cada subconjunto y en base a este nivel se lleva a cabo la selección de padres dentro del algoritmo evolutivo.

Criterios de terminación

El ciclo evolutivo termina cuando se alcanzó el número máximo de generaciones, g, o cuando el nivel de aptitud más alto en la población no mejora durante una ventana de tiempo de v generaciones, o cuando el nivel de aptitud de la población en promedio alcanza cierto umbral.

5.6 Primera implementación: versión mono-objetivo

En la primera versión, el algoritmo evolutivo contempla la optimización de la capacidad predictiva de subconjuntos de p descriptores. Es decir, pretende encontrar subconjuntos que contengan a las p características más relevantes. Para determinar la capacidad predictiva de cada subconjunto se calcula el error cuadrado medio de predicción cuando se utiliza un método de predicción dado. De esta manera, el wrapper de la primera etapa de la arquitectura está constituido por el mecanismo de búsqueda del algoritmo evolutivo y el mecanismo de evaluación de la capacidad predictiva.

5.6.1 Cuestiones de investigación

- Cuestión 1: ¿Cómo diseñar la primera fase para lograr la evolución apropiada de los subconjuntos de descriptores? ¿Qué consideraciones especiales se deben tener en cuenta? (p.e. cantidad de descriptores a ser seleccionados)
- Cuestión 2: ¿Cómo llevar a cabo la evaluación de la capacidad predictiva de cada subconjunto?
- Cuestión 3: ¿Cuál es la mejor opción entre las funciones de aptitud propuestas para este objetivo si se analizan los resultados obtenidos luego de la segunda fase?

5.6.2 Generación de la población inicial

La generación de los individuos de la población inicial se realiza construyendo cadenas binarias de longitud m. Para poder generar modelos de predicción de p descriptores durante la creación de cada individuo se asegura que haya exactamente p bits en 1 (p bits activos).

5.6.3 Operadores genéticos

- **Selección.** La selección de padres se realiza por medio del operador de *selección por torneo* por medio del cual, k subconjuntos son seleccionados al azar y de ellos se mantiene a los b mejores. En particular, se utilizó la parametrización tradicional para este operador que consiste en $k = 2$ y $b = 1$. Además, como criterio elitista, se mantuvo en todo momento al mejor individuo de la generación.
- **Recombinación.** Como primer paso de este proceso, se utiliza el *operador de recombinación de un punto* de la misma forma que en AGs clásicos, es decir, un padre contribuye con los n primeros bits y el otro con los restantes. Posteriormente, dado que los individuos generados de esta forma pueden resultar con más o menos de p bits activos, se realiza una corrección en esos individuos cambiando de 0 a 1 o de 1 a 0 tantos bits como sean necesarios para lograr que cada individuo tenga p bits activos. Durante este proceso, los bits modificados se seleccionan de forma aleatoria.
- **Mutación.** Debido a que el esquema de recombinación incorpora inherentemente un proceso de mutación resultante del mecanismo de corrección de individuos, en esta primera versión del algoritmo evolutivo nos abstuvimos de aplicar otro proceso de mutación para no incorporar una diversidad excesiva a la población.

5.6.4 Función de aptitud

Teniendo en cuenta que el objetivo del algoritmo es determinar los subconjuntos con los p descriptores más relevantes para predecir la variable dependiente con cierto método de predicción, la función de evaluación debería estimar la precisión que logra dicho método de predicción cuando éste contempla sólo esos p descriptores. Para realizar esta evaluación se utiliza la fórmula presentada en la ecuación 5.1. Esta fórmula calcula el error cuadrado medio de predicción, MSEP, obtenido para la selección correspondiente al $j-$ésimo individuo, s_j, al aplicar el método de predicción $\mathcal{P}_Z$ cuando se utiliza una porción del

conjunto de datos, Z_2. Cabe aclarar que este método fue entrenado con otra porción, Z_1, del conjunto total de datos Z.

$$F_{MSEP}(s_j, \mathcal{P}_{Z_1^j}, Z_2^j) = \frac{1}{m_2}\left[\sum_{(\overline{\mathrm{x}}_i, \mathrm{y}_i)\in Z_2^j} (\mathrm{y}_i - \mathcal{P}_{Z_1^j}(\overline{\mathrm{x}}_i))^2\right] \tag{5.1}$$

Donde:

- Z es una matriz que representa al conjunto total de datos a partir del cual se desea realizar la selección. La última columna de Z almacena la variable dependiente que se quiere modelar. Para el caso de QSAR/QSPR, Z representa los datos de los compuestos donde cada fila corresponde a un compuesto y cada columna, excepto la última, corresponde a una característica. La última columna corresponde a la actividad/propiedad fisicoquímica que se quiere modelar y es denotada como y.
- Z_1 $(m_1 \times n)$ y Z_2 $(m_2 \times n)$ son los conjuntos de datos de los compuestos usados para entrenamiento y validación (interna) respectivamente. Además, $Z_1 \cap Z_2 = \varnothing$ y $Z_1 \cup Z_2 = Z$.
- $\mathcal{P}_Z$ es un método de predicción entrenado con el conjunto de datos Z_1.
- El superíndice j, como el presente en Z_1^j, denota el conjunto de datos Z_1 filtrado de acuerdo al conjunto de características seleccionadas codificado en el j-ésimo individuo. Es decir, Z_1^j contiene solamente las variables de Z_1 cuyo correspondiente bit en el cromosoma del individuo j es un "1".
- $\overline{\mathrm{x}}_i$ es un vector que contiene los valores de los descriptores para el i-ésimo compuesto del conjunto de datos dado. De esta forma, el valor $\mathcal{P}_{Z_1^j}(\overline{\mathrm{x}}_i)$ es el valor de predicción de $\overline{\mathrm{x}}_i$ usando $\mathcal{P}$ entrenado sobre el conjunto de datos Z_1 considerando sólo los descriptores seleccionados por el individuo j.
- y_i es el valor que se quiere obtener para el i-ésimo compuesto del conjunto de datos dado.

Predictores

La ecuación 5.1 es la responsable de guiar la búsqueda del algoritmo evolutivo. Debido a que en cada generación una población de q individuos es evaluada por dicha ecuación, el método contemplado en $\mathcal{P}$ debe ser un método de regresión lo suficientemente veloz para no constituir un esfuerzo computacional excesivo. Dado que no existe un método de regresión ideal para evaluar la capacidad predictiva de un subconjunto de descriptores que

cumpla con estas condiciones se utilizaron tres métodos diferentes. El primero de ellos utiliza árboles de decisión (DT), el segundo método se basa en regresión lineal múltiple (MLR) y el tercero se basa en regresión no lineal (NLR).

DT. Los árboles de decisión se usan como árboles de regresión [BFOS84] sin utilizar ningún tipo de poda. En este tipo de árboles, cada variable es separada en intervalos de acuerdo a los valores de la variable a predecir y se construye un árbol de tal forma que cada nodo es capaz de responder *sí* o *no* e indicar cuál es la dirección a seguir dentro del árbol. En general, el modelo se construye con una porción del conjunto de datos $\mathbf{X}$ e $\mathbf{y}$, digamos $\mathbf{X}$' e $\mathbf{y}$', para el entrenamiento y, posteriormente, se suele obtener el error asociado a una muestra cuando se aplica el modelo en otra porción del conjunto de datos, $\mathbf{X}$" e $\mathbf{y}$". Utilizando los valores de la muestra en $\mathbf{X}$" e $\mathbf{y}$" cada nodo del árbol permite avanzar a través de otros nodos y calcular el error de predicción asociado.

MLR y NLR. Los métodos basados en regresión lineal y no lineal utilizan las ecuaciones 5.2 y 5.3 respectivamente.

$$Y = \beta_0 + \sum_{i=1}^{p} \beta_i X_i \tag{5.2}$$

$$Y = \beta_0 + \sum_{i=1}^{p} \left(\sum_{k=1}^{4} \beta_{ik} X_i^k \right) \tag{5.3}$$

Donde:

- p es la cantidad de descriptores activos,
- X_i corresponde a la $i-$ésima variable del conjunto de variables resultante de realizar el filtro de acuerdo con el $j-$ésimo individuo,
- el coeficiente β_i se encuentra asociado con el término lineal correspondiente a la variable i y representa la pendiente de la relación lineal entre X_i e Y.
- el coeficiente β_{ik} corresponde al coeficiente asociado a la variable i para el término de grado k,
- ambos tipos de coeficientes, β_i y β_{ik}, se ajustan con cuadrados mínimos no lineales por medio del método de Gauss-Newton [MNT04].

5.6.5 Evaluación del algoritmo

Los experimentos realizados en esta primera instancia tienen dos objetivos prácticos: evaluar el desempeño del algoritmo evolutivo en función del desempeño de la infraestructura de dos fases y analizar cuál de los métodos de predicción utilizados en el wrapper resulta más prometedor.

Caso de estudio LogP La *hidrofobicidad* es una de las propiedades fisico-químicas más extensamente modeladas debido a la dificultad de determinar su valor de forma experimental y también por estar directamente relacionada con las propiedades ADMET [TY03, HLT00]. Tradicionalmente, la *hidrofobicidad* se expresa en términos del logaritmo del coeficiente de partición octanol-agua, conocido como LogP. Como se dijo en la sección 5.5.1, uno de los objetivos prácticos finales de la arquitectura es su uso en el *problema de selección de características para QSAR/QSPR*. Más específicamente, la propuesta de estos primeros estudios es encontrar los diez descriptores más relevantes para predicción de LogP de la familia de descriptores constitucionales. Para realizar los experimentos se trabajó con un conjunto de 1200 compuestos (filas) de la *Physical Properties Database* (PHYSPROP) [Phi] y 47 descriptores constitucionales (columnas).

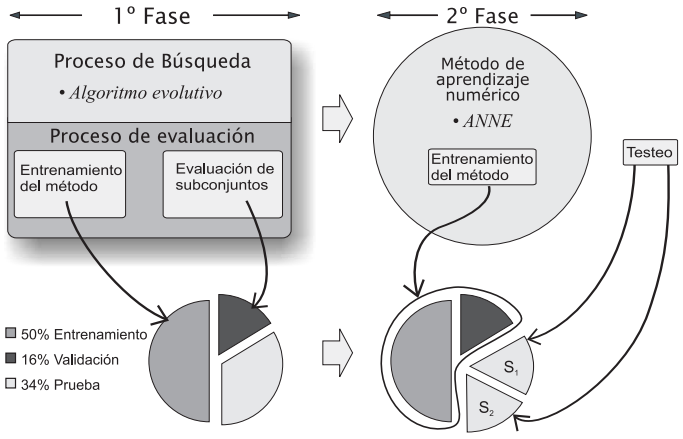

Figura 5.6: El 50% de los datos se reservó para entrenamiento, el 16% se utilizó para la evaluación de los subconjuntos y el 34% restante se dividió en dos conjuntos de testeo (S_1 y S_2).

División del conjuntos de datos. El conjunto de datos se dividió en varios conjuntos no solapados a fin de llevar a cabo el entrenamiento y el testing de las distintas partes de la arquitectura. En esta división, el 50% de los datos se reservó para entrenar los métodos

de predicción utilizados en el wrapper (Z_1), el 16% de los datos se utilizó para realizar la evaluación interna del wrapper (Z_2) (es decir, para otorgar un valor de aptitud a cada individuo generado por el algoritmo evolutivo durante la evolución) y el 34% restante se dividió en dos conjuntos de testeo (S_1 y S_2) que se utilizaron para evaluar el desempeño de los subconjuntos en la segunda etapa de la arquitectura. De esta forma, los resultados obtenidos por el algoritmo evolutivo fueron entregados a un conjunto de redes neuronales (NNE) en la segunda etapa de la arquitectura para ser evaluados más rigurosamente y posteriormente fueron evaluados sobre S_1 y S_2. Esta red fue especialmente diseñada para predicción de LogP y fue entrenada con el 66% (50%+16%, es decir, $Z_1 + Z_2$) de los datos utilizados. La figura 5.6 muestra la división de los datos y la forma en que se utilizaron en cada parte de la arquitectura.

Parámetros del algoritmo evolutivo. Se realizaron 15 réplicas del algoritmo evolutivo sobre el conjunto de datos descripto para cada uno de los métodos de predicción considerados para el wrapper (DT, MLR y NLR). Cada corrida evolucionó una población de *45* individuos durante *45 generaciones*, con una *probabilidad de recombinación de 0.8* y un *tamaño de torneo de 3 individuos*. Como criterio de finalización se utilizó el límite de generaciones y un criterio fenotípico que consiste en dar por terminada la evolución si durante un número de generaciones *t=10* no se observan mejoras en el máximo fitness alcanzado. La longitud de cada individuo, m, debe ser igual a la cantidad de descriptores presentes en el conjunto de datos, por lo que *m=47* para este caso de estudio. De acuerdo con experimentos realizados previamente, el número de características a mantener en cada subconjunto de descriptores fue establecido en $p_m = 10$ (en estos experimentos se analizaron tres tamaños diferentes, lo cual si bien aumentó el tiempo de cómputo en los estudios preliminares, no constituyó una búsqueda exhaustiva).

Experimentos y análisis de resultados

Además de las 15 réplicas del EA para cada método, se consideraron los resultados obtenidos por 15 selecciones aleatorias de subconjuntos de $p = 10$ descriptores y se consideró al subconjunto formado por todas las características. Para cada una de las 15 réplicas del EA se seleccionó al mejor individuo (el cual representa un modelo con un subconjunto de $p = 10$ características) para ser entregado a la segunda etapa de la arquitectura. En la esta etapa se realizaron 7 corridas de la red para tener suficientes réplicas y se determinó

el error de predicción promedio. Estas 7 corridas se realizaron sobre cada uno de los conjuntos: Z_2, S_1 y S_2.

Análisis de resultados promedios. Con los resultados obtenidos durante las réplicas se pudo establecer un error de predicción promedio tomando el mejor resultado de cada corrida del EA (15 subconjuntos) y haciendo las 7 réplicas sobre la red, es decir que en total tenemos $15 \times 7 = 105$ muestras, cuyos promedios para cada método se pueden ver en las columnas *"Prom."* de la tabla 5.1. Por otra parte, también se calculó para cada método el promedio de los 7 resultados obtenidos sobre la red para el *mejor de los mejores 15 subconjuntos*. Estos promedios pueden verse en las columnas *"Mejor"* de la tabla 5.1. Como puede observarse, los resultados obtenidos con los métodos DT y NLR

	Todos	Aleatorio		DT		MLR		NLR	
	-	*Mejor*	*Prom.*	*Mejor*	*Prom.*	*Mejor*	*Prom.*	*Mejor*	*Prom.*
Z_2	1.4430	1.3617	1.5166	1.2118	1.3791	1.3855	1.4066	1.1557	1.2884
S_1	1.3037	1.2860	1.4318	1.1249	1.3398	1.2851	1.3599	1.2849	1.3213
S_2	1.0414	1.0787	1.1571	1.0603	1.1227	1.0331	1.109	1.0255	1.1256

Tabla 5.1: Resultados obtenidos para 15 réplicas del EA y 7 réplicas de la segunda fase de la arquitectura para cada método y para cada porción de los datos. Las columnas *Mejor* muestran el resultado promedio sobre las 7 réplicas de la red ejecutadas para el mejor de los mejores subconjuntos obtenidos en las 15 réplicas del EA. Las columnas *Prom.* muestran el resultado promedio obtenido en las $105 = 15 \times 7$ réplicas de la segunda fase de la arquitectura (7 réplicas de la red para el mejor subconjunto obtenido en cada una de las 15 réplicas del EA).

son mejores que los resultados obtenidos para los conjuntos generados aleatoriamente e incluso en dos de los conjuntos de datos superan a los resultados obtenidos para el conjunto que contempla todos los descriptores. Sin embargo, es necesario realizar otras pruebas estadísticas para determinar si existe una diferencia significativa y si estas diferencias no son producto de la varianza introducida por los métodos. Para esta tarea se suele utilizar el test de ANOVA, el cual permite realizar la comparación de resultados obtenidos por distintos grupos e identificar si las diferencias en los resultados provienen de variaciones intrínsecas de cada grupo o si provienen de variaciones entre los grupos.

Comparación de los métodos en base al mejor de todos los subconjuntos obtenidos por el EA. El test de ANOVA separa los datos en dos fuentes de varianza (*S. of v.*): la *varianza entre* ($M.S._{entre}$) y la *varianza dentro* ($M.S._{dentro}$). La *varianza entre* se calcula sumando los cuadrados de las diferencias entre la media de cada grupo

y la media global, $S.S._{entre}$, y dividiendo dicha suma por los grados de libertad entre grupos (cantidad de grupos menos 1), mientras que la *varianza dentro* de los grupos se calcula dividiendo la sumatoria de los cuadrados de la desviación estándar de cada grupo, $S.S._{dentro}$, por los grados de libertad globales (la cantidad de casos en total menos el número de grupos). Por medio de la relación entre ambas medidas ($M.S._{entre}/M.S._{dentro}$) se calcula una F estadística. Basándonos en la F estadística y en los grados de libertad se puede calcular el valor probabilístico, $p-$value, para determinar si existe una diferencia significativa entre los promedios reportados por cada grupo.

Para determinar si existen diferencias significativas se realizaron dos tests ANOVA, uno

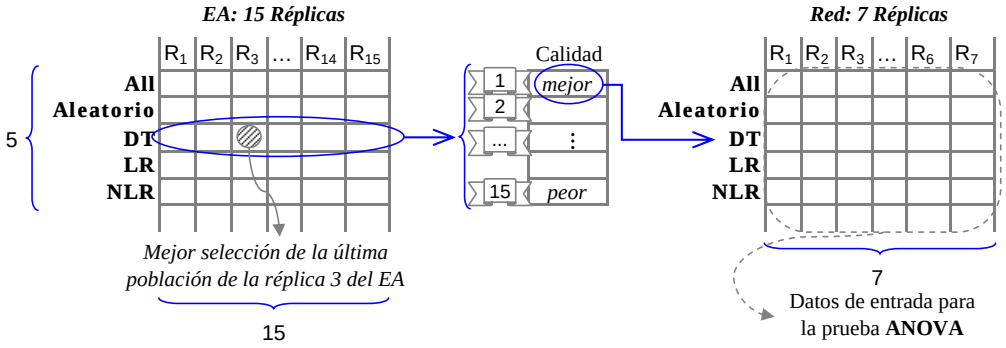

Figura 5.7: Se realizaron 15 réplicas del EA con cada método. Para la mejor selección de obtenida con respecto a todas las réplicas se hicieron 7 réplicas de la red. Al test de ANOVA se ingresaron los valores de las 7 réplicas para cada método (7×5).

sobre los datos del conjunto S_1 y otro sobre los datos del conjunto S_2 (dejamos de lado el conjunto Z_2 porque fue el que se utilizó durante las réplicas de los EAs). La figura 5.7 muestra un esquema del avance de los datos (desde las 15 réplicas realizadas con el wrapper hasta la tabla de entrada para el test ANOVA). Para realizar el test se utilizó el mejor subconjunto de todos los subconjuntos obtenidos en las 15 réplicas del wrapper para cada método. Con este mejor subconjunto se llevaron a cabo 7 réplicas de la segunda fase y se armó la matriz de entrada para el ANOVA. El valor F de la prueba de ANOVA se obtiene de la tabla de la distribución F y depende de la cantidad de grupos (k), el tamaño de cada grupo (n) y los grados de libertad ($d.f.$). En este caso estos valores son:

- $k = 5$ ya que estamos comparando *Todos, Aleatorio, DT, MLR* y *NLR*,
- $n = 7$ son las 7 réplicas realizadas con la red,
- $d.f._{entre} = k - 1 = 5 - 1 = 4$ y
- $d.f._{dentro} = nk - k = 7 * 5 - 5 = 30$

Los valores tabulares de la distribución (para $d.f._{entre}/d.f._{dentro} = v_1/v_2$) son 2.69 y 2.142 para niveles de significación de 5% y 1% respectivamente. Las tablas 5.2 y 5.3 muestran los resultados obtenidos por el test ANOVA para los conjuntos S_1 y S_2 respectivamente. Dado que el $p-$value es menor que 0.05 (el valor para α con el que se calculó F), podemos decir que existe diferencia entre los métodos wrapper para el conjunto S_1 pero no podemos decir lo mismo sobre el conjunto S_2.

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
entre	0,154148444	4	0,038537111	35,9410	0,0001
interna	0,032166989	30	0,001072233		
Total	0,186315433	34			

Tabla 5.2: Resultados de ANOVA para la mejor selección de cada método sobre el conjunto de datos S_1

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
entre	0,013058371	4	0,003264593	1,3541	0,2732
interna	0,072325361	30	0,002410845		
Total	0,085383732	34			

Tabla 5.3: Resultados de ANOVA para la mejor selección de cada método sobre el conjunto de datos S_2

Dado que se pudo establecer que existe cierta diferencia entre las medias reportadas, el siguiente paso es intentar determinar dónde se encuentran dichas diferencias. Para esto se realizó el test de Dunnett utilizando DT como método de referencia. Este test, permite comparar globalmente resultados obtenidos por diferentes grupos (*métodos*) con respecto a un grupo de referencia y determinar si existen diferencias entre todos los demás grupos y el grupo de control elegido. Para esto, se calcula un valor estadístico $t_{observado}$ para cada par (grupo de referencia, grupo i). Luego este valor se compara con el valor crítico, $t_{critico}$, obtenido de la tabla de *Dunnett*, el cual depende del tamaño del grupo (N), el número de grupos comparados con respecto al grupo de referencia (p) y el nivel de significación elegido (α). Para este caso particular, dichos valores son:

- N $= 7$ ya que se hicieron 7 réplicas con la red para cada grupo,
- $p = 4$ dado que se consideraron *NLR, MLR, Random* y *All* para ser comparados con DT y
- $\alpha = 5\%$ y 1% son los errores globales asociados para los cuales se calculó el $t_{critico}$ a partir de las tablas de Dunnett correspondientes.

Además, es necesario calcular los grados de libertad involucrados: $d.f. = (p+1)(\text{N}-1) = (4+1)(7-1) = 30$. Con 30 grados de libertad y 4 grupos a contrastar, la tabla de Dunnett de un sentido da un $t_{critico}$ de $2,25$ y $2,97$ para $\alpha = 5\%$ y $\alpha = 1\%$ respectivamente. Si el valor $t_{observado}$ para el método i es mayor que el $t_{critico}$, el método i difiere del método de referencia. La tabla 5.4 muestra los valores $t_{observado}$ obtenidos para cada método para el conjunto S_1. Como puede apreciarse, todos los valores observados son mayores que ambos valores críticos obtenidos. Esto nos permite concluir que para este conjunto de datos DT se desenvuelve mejor que los demás métodos de regresión (indicado en la tabla con $**$), teniendo el menor error promedio para el mejor de los subconjuntos de características y mostrando diferencias estadísticamente significativas con respecto a los demás métodos. Sin embargo, para el conjunto de datos S_2 no ocurre lo mismo. La tabla 5.5 muestra el $t_{observado}$ obtenido para cada método. Como puede apreciarse, todos los valores son menores a los valores $t_{critico}$ tabulares, por lo que no se detectan diferencias significativas globalmente entre los métodos sobre el conjunto S_2 (ns).

Métodos	DT	NLR	MLR	Aleatorio	Todos
Estadísticas	-	9,1373	9,1476	9,1988	10,2126
Resultados	-	**	**	**	**

Tabla 5.4: Resultados de la prueba de Dunnett para las 7 réplicas de la red sobre el conjunto de datos S_1 con la mejor selección de cada método.

Métodos	NLR	MLR	Todos	DT	Aleatorio
Estadísticas	1.9865	1.5505	1.0759	-	1.0550
Resultados	ns	ns	ns	-	ns

Tabla 5.5: Resultados de la prueba de Dunnett para las 7 réplicas de la red sobre el conjunto de datos S_2 con la mejor selección de cada método.

Comparación de los métodos en base al mejor subconjunto obtenido para cada réplica del EA. Durante el análisis estadístico previo se tuvieron en cuenta los factores de varianza debidos a los métodos considerando varias réplicas en la segunda fase de la arquitectura. Sin embargo, no se tuvo en cuenta la varianza debida a las distintas réplicas efectuadas con el EA para cada método (en este caso 15). Para poder tener en cuenta esta segunda fuente de varianza realizamos una prueba ANOVA anidada considerando el **método** como factor principal y las **réplicas del EA** para cada método como factor anidado. Para poder realizar la prueba se consideró al mejor subconjunto

de características seleccionado para cada réplica del EA (15 subconjuntos: uno por cada réplica). Para cada uno de los 15 subconjuntos se tuvieron en cuenta 7 réplicas de la segunda fase de la arquitectura y se armó la matriz de entrada para la prueba de ANOVA anidada. En esta ocasión debemos utilizar: el número de grupos considerados en el *factor principal* (k_a), el número de grupos considerados en el *factor anidado* (k_b), el tamaño de cada grupo anidado (n) y los grados de libertad ($d.f._{principal}$, $d.f._{anidado}$ y $d.f._{interno}$), cuyos valores son:

- $k_a = 4$ ya que sólo se consideran los grupos para los cuales se cuenta con distintas réplicas, es decir, *DT, MLR, NLR* y *Aleatorio* (se descarta *Todos* debido a que su resultado es único y por lo tanto no introduce varianza a este nivel),
- $k_b = 15$ es el número de selecciones consideradas (la mejor de cada réplica del EA),
- $n = 7$ son las réplicas realizadas con la red,
- $d.f._{principal} = k_a - 1 = 4 - 1 = 3$,
- $d.f._{anidado} = k_a(k_b - 1) = 4 * 14 = 56$ y finalmente
- $d.f._{interno} = k_a k_b(n - 1) = 4 * 15 * 6 = 360$

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
factor principal	0,737076291	3	0,245692097	5,8253	0,0015
factor anidado	2,361903692	56	0,042176851	30,65412	0,0001
interna	0,495322256	360	0,001375895		
Total	3,594302239	419			

Tabla 5.6: Resultados de la prueba ANOVA anidada sobre el conjunto de datos S_1 para las 7 réplicas de la red para la mejor selección de cada réplica del EA para cada método.

Los valores tabulares para la distribución F para el factor principal ($F_{(k_a-1),k_a k_b(n-1)} = F_{3,360}$) son 2,63 y 3,38 para $\alpha = 5\%$ y $\alpha = 1\%$ respectivamente, mientras que para el factor anidado ($(F_{k_a(k_b-1),k_a k_b(n-1)} = F_{56,360})$) son 1,367 y 1,552 para $\alpha = 5\%$ y $\alpha = 1\%$ respectivamente. La tabla 5.6 muestra los resultados obtenidos para dicha prueba de ANOVA anidada sobre el conjunto de datos S_1. El análisis muestra que existen diferencias entre las varianzas ($F_{0_{anidado}} = 30,65412 > 1,552$ y $F_{0_{principal}} = 5.835 > 3,38$).

La tabla 5.7 muestra los resultados obtenidos sobre el conjunto de datos S_2. Para este conjunto de datos no parecen observarse diferencias entre los métodos, es decir, el factor principal no muestra diferencias entre las varianzas ($F_{0_{principal}} = 2,3485 < 2,63$ y

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
factor principal	0,16849875	3	0,05616625	2,3485	0,0823
factor anidado	1,33926114	56	0,023915377	18,38	0,0001
interna	0,468360687	360	0,001301001		
Total	1,976120577	419			

Tabla 5.7: Resultados de la prueba ANOVA anidada sobre el conjunto de datos S_2 para las 7 réplicas de la red para la mejor selección de cada réplica del EA para cada método.

$p = 0,0823 > 0.05$). Sin embargo, el $p-$value para el factor principal ($p = 0,0823$) es suficientemente pequeño como para pensar que pueden haber diferencias a un nivel α menos estricto. Por ejemplo, si se considera $\alpha = 10\%$ el valor tabular para el factor principal ($F_{(k_a-1),k_ak_b(n-1)} = F_{3,360}$) es 2,099 con lo cual se cumple que $F_{0_{principal}} = 2,3485 > 2,099$.

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
entre	0,105296613	3	0,035098871	5,8253	0,0015
interna	0,337414806	56	0,006025264		
Total	0,442711419	59			

Tabla 5.8: Resultados de ANOVA para el promedio de las selecciones de cada método sobre el conjunto de datos S_1

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
entre	0,024071245	3	0,008023748	2,3485	0,0823
interna	0,19132302	56	0,003416482		
Total	0,215394264	59			

Tabla 5.9: Resultados de ANOVA para el promedio de las selecciones de cada método sobre el conjunto de datos S_2

Se puede calcular una última prueba de ANOVA simple basada en los promedios de todas las réplicas anidadas; las tablas 5.8 y 5.9 muestran estos resultados para los conjuntos S_1 y S_2 respectivamente. A partir de los resultados obtenidos se calculó una prueba de Dunnett de comparación múltiple para determinar dónde se encuentran las diferencias y cuán significativas son, las tablas 5.10 y 5.11 muestran estos resultados estadísticos.
En esta ocasión para el conjunto de datos S_1 no se encontraron diferencias significativas a nivel global entre los métodos utilizados por el EA, pero se encontraron diferencias globalmente significativas entre el método de selección *aleatorio* y *árboles de decisión* (DT).

Métodos	NLR	DT	MLR	Aleatorio
Estadísticas	0,6530	-	0,7074	3,2455
Resultados	ns	-	ns	**

Tabla 5.10: Resultados de la prueba de Dunnett para las 7 réplicas de la red sobre el conjunto de datos S_1 considerando la mejor selección de cada réplica de cada método.

Métodos	NLR	DT	MLR	Aleatorio
Estadísticas	0,7688	-	0,1028	1,2127
Resultados	ns	-	ns	ns

Tabla 5.11: Resultados de la prueba de Dunnett para las 7 réplicas de la red sobre el conjunto de datos S_2 considerando la mejor selección de cada réplica de cada método.

Discusión. Los resultados previos nos permiten avalar la hipótesis de que es posible alcanzar una mejora en las predicciones teniendo en cuenta sólo los subconjuntos de características que el EA considera importantes. Las comparaciones se dividieron en dos grupos: la mejor selección de cada método sobre todas las réplicas y el promedio de las 15 diferentes mejores selecciones de cada réplica para cada método. Además, se consideraron los resultados de la predicción cuando se seleccionan conjuntos de características de forma aleatoria y también cuando el grupo total de características es tomado como una selección. En este punto es importante resaltar que cuando todos los descriptores se encuentran seleccionados sólo existe una selección y por lo tanto no se cuenta con estadísticas promedio.

Los resultados obtenidos para los conjuntos de prueba S_1 y S_2 son contradictorios, ya que para el conjunto S_1 las mejores selecciones alcanzadas por el EA (con los tres métodos en evaluación: DT, NLR y MLR) siempre superan al método aleatorio y al grupo con todas las características, mientras que sobre el conjunto S_2 no se encuentran diferencias significativas. Esta discordancia es sumamente interesante dado que es un escenario real posible en el problema de predicción de logP. Es posible que la falta de mejora se deba a que la familia de descriptores constitucionales resulta insuficiente para describir el comportamiento de logP. Sin embargo, es bueno ver que el uso del EA para una selección previa no perjudique la capacidad predictiva resultante en ausencia de descriptores influyentes.

5.7 Segunda implementación: versión multi-objetivo

En la primera versión de la arquitectura de dos fases, se supuso que se conocía de forma aproximada el número de características suficientes para modelar determinado compor-

tamiento. Sin embargo, en algunos problemas de selección de características reales esta suposición no es posible. Por ejemplo, en el caso de QSAR/QSPR, no hay un acuerdo acerca de cuáles y cuántos descriptores son relevantes e influyen sobre el comportamiento de algunas propiedades. Esta es una cuestión importante dado que si se utilizan más características de las necesarias se puede producir un sobreajuste con respecto a los datos de entrenamiento, obteniendo buenos resultados en dichos datos pero malos o pobres resultados en otro grupo de datos. Por otro lado, si se utilizan muy pocas características los modelos obtenidos serán pobres debido a que estarán utilizando menos descriptores de los necesarias. Por lo tanto, en esta segunda versión queremos contemplar simultáneamente dos objetivos: maximizar la capacidad predictiva del subconjunto de características (es decir, minimizar el error de predicción) y minimizar la complejidad del modelo obtenido a través de dicho subconjunto, (es decir, minimizar la cantidad de descriptores seleccionados). Tener en cuenta ambos aspectos en forma conjunta implica ocuparse de un problema de optimización multi-objetivo. Durante la presente sección se explican los detalles de la implementación de una versión más evolucionada de la primera fase de la arquitectura que incorpora de forma automática la selección de un número adecuado de características.

5.7.1 Cuestiones de investigación

- Cuestión 1: ¿Cómo se puede lograr la evolución de subconjuntos y cómo instaurar un rango a dichos subconjuntos cuando se persiguen múltiples objetivos, tales como lograr valores elevados en la capacidad predictiva y un número mínimo o reducido de características? ¿Qué consideraciones se deben tener en cuenta en la estructura interna del wrapper para lograr ambos objetivos?
- Cuestión 2: ¿Resulta mejor observar ambos objetivos pero independientemente uno de otro en un esquema Pareto, que mirarlos conjuntamente dentro de un esquema agregativo? ¿Existe una combinación *"método evolutivo multi-objetivo/método de evaluación de subconjuntos"* que se desempeñe mejor?
- Cuestión 3: ¿Los subconjuntos generados en la etapa de entrenamiento son efectivos cuando se los prueba sobre un nuevo conjunto de datos correspondiente al mismo comportamiento que se quiere modelar?
- Cuestión 4: ¿Los resultados obtenidos son comparables con los reportados en la literatura?

- Cuestión 5: ¿Existe dependencia entre la linealidad o no linealidad de los datos y el desempeño de los métodos de predicción utilizados en la evaluación o los esquemas evolutivos empleados?

5.7.2 Algoritmos evolutivos multi-objetivo utilizados

Para evolucionar los dos objetivos considerados es necesario emplear una estrategia evolutiva capaz de considerar múltiples objetivos en forma simultánea. En base a la revisión desarrollada en el Capítulo 4 se sabe que existen dos esquemas evolutivos multi-objetivo principales: los basados en Pareto y los no-Pareto. Dentro de la primera categoría el **NSGA-II** y el **SPEA2** han reportado muy buenos resultados para la cantidad de objetivos que se quieren considerar en este problema de FS. Por lo tanto, se decidió utilizar estos dos enfoques Pareto y un enfoque agregativo que contempla los objetivos dentro de una única fórmula agregada. De la misma forma que se hizo para el caso mono-objetivo, los esquemas se compararon con respecto al rendimiento alcanzado en las pruebas sobre los resultados reportados por la segunda fase de la arquitectura. En las siguientes secciones se detallan los aspectos particulares para la implementación de la versión multi-objetivo del wrapper.

5.7.3 Generación de la población inicial

Utilizando la representación propuesta inicialmente (cadenas de bits donde un 1 en la posición i significa que el descriptor i está seleccionado y un 0 significa que no), se genera una población inicial de individuos cada uno con una cantidad entre 0 y p_m bits en 1, donde p_m es un valor considerado máximo para la cantidad de características necesarias en el modelo. A diferencia de la versión anterior, en la que todos los individuos tienen la misma cantidad de características seleccionadas, en esta segunda versión los individuos pueden tener una cantidad variable.

5.7.4 Operadores genéticos

- **Selección.** El esquema de selección utilizado depende del algoritmo evolutivo. Para la estrategia agregativa se utilizó el método de selección por torneo descripto anteriormente. Mientras que en los esquemas basados en Pareto, los operadores de selección son los correspondientes los algoritmos NSGA-II y SPEA2 respectivamente, los cuáles fueron explicados en el Capítulo 4. Es importante aclarar que los tres

algoritmos evolutivos desarrollados incluyen elitismo, lo cual protege a los individuos más aptos en cada generación.

- **Recombinación.** Para crear nuevos individuos se utilizó una modificación del operador de recombinación de un punto. Al igual que en el mecanismo de un punto tradicional, se generan dos nuevos individuos copiando los valores de los primeros n bits de uno de los padres y los restantes valores se toman del otro padre. Sin embargo, aprovechando el conocimiento acerca del conjunto de datos que se está estudiando se puede imponer un número máximo a la cardinalidad de los subconjuntos, a la cual denominamos p_m. Por lo tanto, los individuos que superan dicho número son modificados mediante un mecanismo de corrección que selecciona posiciones de manera aleatoria y disminuye la cardinalidad del subconjunto hasta quedar dentro del límite establecido. Cabe destacar, que a diferencia de la versión anterior, este mecanismo de corrección no resulta aplicado tantas veces como para constituir un mecanismo de mutación inherente.
- **Mutación.** Este operador introduce cambios en cada individuo con una probabilidad m. Si un individuo es seleccionado para mutar, cada bit de su cadena tiene una probabilidad m_b de ser modificado de 0 a 1 o de 1 a 0.

5.7.5 Evaluación de los subconjuntos

Durante la evaluación de cada subconjunto se tienen en cuenta la capacidad predictiva y la complejidad del modelo asociado. La capacidad predictiva de cada subconjunto es evaluada mediante la ecuación 5.1, mientras la complejidad del modelo se mide por la cardinalidad del subconjunto F_C. Como se explicó en el Capítulo 4, para evaluar ambos aspectos en forma simultánea, los algoritmos evolutivos basados en Pareto utilizan una representación vectorial e imponen un orden parcial sobre la población. En este problema particular los vectores corresponden a un espacio vectorial $\mathbb{R} \times (\mathbb{N} \cup \{0\})$. El NSGA-II y el SPEA2 establecen un orden parcial sobre conjuntos de vectores pertenecientes a dicho espacio utilizando los conceptos de dominación de Pareto vistos en el Capítulo 2. Por otro lado, el esquema agregativo intenta optimizar ambos objetivos por medio de la función F_{Agg} de la siguiente manera:

$$F_{Agg} = \alpha F_{MSEP} + (1 - \alpha) F_{MSEP} \frac{F_C}{p_m} \tag{5.4}$$

Donde $\alpha \in (0, 1)$ es un parámetro que permite pesar cada objetivo y p_m es el límite

superior para la cardinalidad de cada subconjunto. El primer término de F_{Agg} representa el error de predicción obtenido utilizando un método de predicción dado, $\mathcal{P}$, mientras que el segundo término refleja la relación entre la cardinalidad del subconjunto y la cantidad máxima de descriptores permitidos, escalada por F_{MSEP}. Mediante esta estrategia agregativa se busca un balance entre la complejidad del modelo y la capacidad predictiva, por lo que α nos permite manejar este balance.

Predictores

Se utilizaron cuatro métodos de predicción diferentes: árboles de decisión (***DT***), regresión lineal múltiple (***MLR***), regresión no lineal (***NLR***), los cuales fueron introducidos en la versión mono-objetivo, y regresión basada en los k-vecinos más cercanos, (***k*NN**).

***k*NN.** El algoritmo de los k-vecinos más cercanos suele ser utilizado para clasificación de objetos en base a los ejemplos de entrenamiento más cercanos en el espacio de características dado. Para llevar a cabo esta tarea, el objeto es comparado con las muestras de entrenamiento y se lo asigna a la clase más común entre los k-vecinos más cercanos. Esto también puede aplicarse en problemas de regresión asignándole a la variable imagen del nuevo vector el valor promedio que poseen en dicha variable los k-vecinos más cercanos.

5.7.6 Ubicación de las nuevas consideraciones

La figura 5.8 muestra la ubicación de las nuevas consideraciones en la primera fase de la infraestructura. Por un lado, el mecanismo de búsqueda debe ser un esquema capaz de

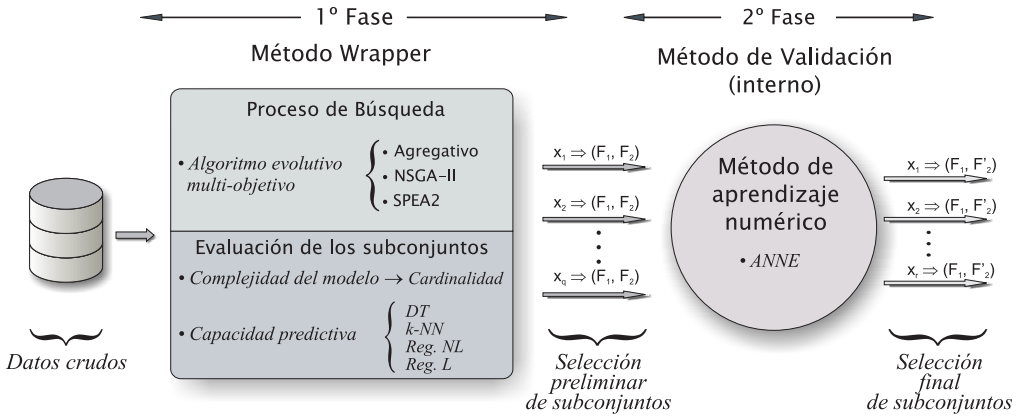

Figura 5.8: Modificaciones en la primera fase de infraestructura de dos fases.

considerar múltiples objetivos tal como el esquema agregativo propuesto o los algoritmos

NSGA-II y SPEA2. Por otro lado, el mecanismo de evaluación del wrapper debe ponderar la capacidad del modelo, por ejemplo: analizando la cardinalidad del subconjunto de descriptores.
Se puede observar que en la primera fase es posible construir varias combinaciones "*método de búsqueda / método de evaluación*", cada una de las cuáles nos permite obtener un wrapper diferente. Al terminar cada réplica del EA luego de cualquier combinación, se arma un frente de individuos no-dominados. Dicho frente puede armarse independientemente de si el método de búsqueda fue o no basado en Pareto, ya que se toma la última población y se analiza cada individuo con respecto a ambos objetivos para determinar si es o no dominado por los demás. De esta manera, los individuos no dominados son considerados como los individuos más importantes para esa réplica del wrapper. Luego, cada uno de los subconjuntos de dicho frente es evaluado por el método de validación (interno). Esto es levemente diferente con respecto a la versión mono-objetivo, ya que gracias a que en dicha versión era posible establecer un orden total sobre la población, durante los experimentos, la segunda fase sólo consideró a un único individuo para cada réplica (el mejor de la población). En la versión multi-objetivo, se aplicó un mecanismo de evaluación más riguroso, en el cual cada subconjunto es evaluado mediante muchas réplicas de una validación cruzada f-fold, donde en cada réplica cada fold es obtenido aleatoriamente. Además, debemos notar que por razones de costo computacional no es posible realizar una estrategia de repetición similar dentro del wrapper, por lo que la segunda fase nos da el escenario adecuado para esta tarea. Para obtener mayores detalles sobre los parámetros y construcción de la fase dos, el lector puede remitirse a [SCVP09]. Finalmente, tal como se hizo en la versión mono-objetivo, se puede aplicar un método de validación externo que evalúe los subconjuntos obtenidos por medio del trabajo conjunto de ambas fases sobre un conjunto de datos totalmente independiente que no haya sido utilizado previamente.

5.7.7 Evaluación del algoritmo

A fin de evaluar el desempeño de la arquitectura bajo este nuevo esquema se recolectaron tres conjuntos de datos diferentes relevantes en QSAR/QSPR:

- *Conjunto de datos* ***logBBB***. La variable objetivo en este conjunto de datos (la propiedad a modelar) es *logBBB*, la cual es una medida de penetración de la barrera hematoencefálica (*BBB* por sus siglas en inglés de *Blood Brain Barrier*). Esta barrera separa al sistema nervioso central restringiendo el paso de muchas sustancias

y permitiendo el paso de otras como el oxígeno y nutrientes. Esta barrera es la que protege al cerebro de muchas posibles infecciones. Sin embargo, superar la dificultad de distribuir agentes farmacéuticos a regiones específicas del cerebro constituye uno de los desafíos más importantes para el tratamiento de enfermedades. Muchos medicamentos pueden no resultar satisfactorios si no logran cruzar la BBB. La medida logBBB es un índice que mide la permeabilidad de esta barrera del cuerpo humano. El conjunto de datos de logBBB, que fue extraído de [KLVHC08], está constituido por **289** compuestos (filas) y **1501** descriptores (columnas) más el descriptor *Iv* el cuál distingue si el índice logBBB ha sido calculado a partir de un ensayo *in vivo* o *in vitro*.

- *Conjunto de datos* ***logHIA***. Los compuestos y descriptores de este conjunto de datos también fueron extraídos de [KLVHC08], donde *logHIA* es la variable objetivo. HIA son las siglas de *Human Intestinal Absortion* (absorción intestinal humana), *logHIA* es otra de las importantes propiedades ADMET, encargada de expresar el grado de absorción intestinal de un medicamento. *LogHIA* es una transformación no lineal de la absorción intestinal expresada como fracción absorbida (% HIA), es decir, el porcentaje que llega al torrente sanguíneo. Este segundo conjunto de datos contiene **127** compuestos y **1499** descriptores.
- *Conjunto de datos* ***logP***. Este conjunto de datos corresponde a la misma propiedad utilizada en el caso de estudio de la versión mono-objetivo. Para recopilar este conjunto de datos se utilizaron 12 descriptores reportados en [YCE+02] y se completó la base de datos con otros 61 descriptores calculados con Dragon [TCMP05]. La base de datos quedó conformada por **442** compuestos pertenecientes a diferentes clases químicas y **73** descriptores.

Antes de la utilización de los conjuntos de datos se aplicó un método de variables linealmente correlacionadas con el fin de descartar redundancias lineales del conjunto de descriptores y facilitar la tarea del wrapper.

Parámetros de los algoritmos evolutivos

Los valores de los parámetros se mantuvieron igual para los tres algoritmos evolutivos multi-objetivo utilizados. Además, aquellos parámetros que no dependen de los datos se mantuvieron iguales para los tres conjuntos de datos. Se utilizó un tamaño de población de *145* individuos, los cuales evolucionaron durante a lo sumo *200 generaciones*. La longitud

de cada individuo depende de la cantidad de características de cada conjunto de datos y se estableció un límite máximo para la cantidad de descriptores seleccionado $p_m = 20$ para los conjuntos ***logBBB*** y ***logHIA***, mientras que para ***logP*** el límite fue $p_m = 50$. Además, se utilizó una *probabilidad de recombinación de 0.75*, una probabilidad de que un individuo sea seleccionado para *mutación* de *0.1* y una *probabilidad de cambio para cada bit de 2/n*. El algoritmo termina cuando se alcanza el número de generaciones máximo o cuando la mejora del fitness promedio poblacional es menor a un nivel de tolerancia $\xi = 10^{-16}$. Esta última característica se analiza durante una ventana de tiempo que en este caso corresponde a $t = 15$. Como característica particular, para el esquema agregativo se utilizó un *tamaño de torneo de 4 individuos*.

Experimentos y análisis de resultados

La propuesta completa comprende 12 diferentes métodos wrapper multi-objetivo, los cuales se obtienen de la combinación de los diferentes métodos evolutivos de búsqueda (*Agregativo, NSGA-II, SPEA2*) con los distintos métodos de evaluación de la capacidad predictiva de los subconjuntos (*DT, MLR, NLR, kNN*). Para cada una de las 12 combinaciones se realizaron 10 ejecuciones del algoritmo evolutivo y para cada una de estas sólo se mantuvieron las soluciones pertenecientes al frente de subconjuntos no dominados formado a partir de la última población. Finalmente, se realizaron 50 validaciones cruzadas f-folds de la segunda fase para cada subconjunto.
Los experimentos se agruparon para analizar cuatro aspectos diferentes:

1. evaluar el *desempeño de los mejores subconjuntos de características obtenidos por el EA* con respecto a ambos objetivos y compararlos con los reportados en los trabajos de Konovalov, [KLVHC08] Yaffe [YCE+02] y Figuereido [Fig03],
2. *analizar y comparar los resultados entre los diferentes wrappers multi-objetivo*,
3. *analizar los subconjuntos de descriptores obtenidos*, y
4. *evaluar la metodología usando validación externa.*

Estos cuatro aspectos se desarrollan en las siguientes subsecciones para cada uno de los conjuntos de datos recopilados.

División de los datos. Con el fin de llevar a cabo los experimentos, nuevamente fue necesario llevar a cabo una división adecuada de los datos. Para poder comparar de

forma justa con el trabajo de Konovalov [Kon94], los conjuntos ***logBBB*** y ***logHIA*** se dividieron en dos grupos, un 50% de los datos se utilizaron para entrenar los métodos de predicción de cada wrapper (Z_1) y el 50% restante se reservó para que dichos métodos realicen la evaluación de la capacidad predictiva de cada subconjunto (Z_2). Además, para estos dos conjuntos de datos se estableció un número de folds $f = 2$ para la segunda etapa (50% de los datos para entrenamiento y 50% para evaluación). Como consideración especial, el descriptor *Iv* correspondiente a ensayo *in vivo/in vitro* (0/1) permaneció siempre seleccionado.

Por otro lado, el conjunto de datos ***logP***, cuyos resultados se cotejaron con el trabajo de Yaffe [YCE+02], se dividió dejando el 80% para entrenar los métodos de predicción de cada wrapper (Z_1) y el 20% restante se reservó para la evaluación de la capacidad predictiva (Z_2). Para la red se estableció un número de folds $f = 5$ (80% de los datos para entrenamiento y 20% para evaluación). Esta división es la utilizada durante los primeros experimentos. Sin embargo, para poder realizar un proceso de validación externa es necesario reservar una porción no solapada de los datos para ser utilizada por el método externo (S_1) (de la misma forma en que se hizo con la versión mono-objetivo). Por lo tanto, para los estudios del punto 4 la división de datos es diferente y será detalla en la subsección ***Análisis de la metodología utilizando validación externa***.

Análisis de los mejores subconjuntos de descriptores

Las tablas 5.12, 5.13 y 5.14 resumen la información de los mejores subconjuntos de descriptores seleccionados, los métodos asociados, la cardinalidad y los errores obtenidos en la validación interna para cada subconjunto. En cada tabla se puede ver el mejor subconjunto reportado por los trabajos de Konovalov [Kon94] y Yaffe [YCE+02] (primera fila de tablas 5.12, 5.13 y 5.14), el mejor subconjunto obtenido con el método de Figuereido [Fig03] (segunda fila), los mejores subconjuntos obtenidos con la infraestructura de dos fases, dos selecciones realizadas por el método agregativo (filas 3 y 4) y dos selecciones realizadas por los EAs basados en Pareto (filas 5 y 6). Las columnas MSEP y R^2 corresponden al error de predicción cuadrado medio y al coeficiente de determinación respectivamente. El coeficiente de determinación es una forma de medir cuan bien puede llegar a resultar la predicción de un modelo. Una de las formas de computar el R^2 es:

$$R^2 = 1 - \frac{\sum_i (Y_i - f_i)^2}{\sum_i (Y_i - \bar{Y})^2} \tag{5.5}$$

En la ecuación anterior Y_i es el valor observado para la muestra i, f_i es el valor predicho por el modelo y $\bar{Y}$ es el promedio de los valores observados. Como puede observarse, a medida que el error en la predicción es menor (numerador) el segundo término en R^2 será más próximo a 0. De esta manera, cuanto más próximo a 1 sea R^2, mejor será la predicción del modelo.

				Método de validación			
	Método de	Función de	Número de	ANNE		MLR	
ID	búsqueda	regresión	descriptores	MSEP	R^2	MSEP	R^2
I	MCVS	MLR	6	0.1265	0.645	0.1225	0.6752
II	Figuereido	MLR	20	0.1302	0.6528	0.121	0.6757
III	$\text{Agr}_{\alpha=0.3}$	MLR	6	0.1205	0.6816	0.1281	0.6525
IV	$\text{Agr}_{\alpha=0.7}$	MLR	15	0.1103	0.7198	0.1113	0.703
V	NSGA-II	MLR	8	0.1140	0.6993	0.1178	0.6727
VI	NSGA-II	MLR	11	0.1052	0.7352	0.1124	0.6821

Tabla 5.12: Comparación de resultados para ***logBBB***. Las columnas MSEP y R^2 corresponden al error cuadrado promedio y al coeficiente de determinación obtenidos sobre el conjunto de validación. El subconjunto I corresponde al mejor subconjunto reportado en [KLVHC08], mientras que el subconjunto II se obtuvo mediante el método de Figuereido [Fig03].

Para el conjunto de datos ***logBBB*** (tabla 5.12) el subconjunto de descriptores III logró mejor capacidad descriptiva (menor MSEP) que el conjunto reportado por Konovalov *et al.* usando el mismo número de descriptores (el mínimo número reportado para este conjunto de datos).

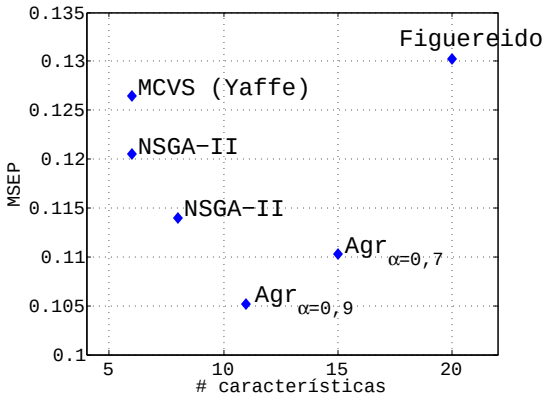

Figura 5.9: Relación *Número de descriptores* vs. *MSEP* para el conjunto de datos ***logBBB***.

Con respecto a MLR, el mejor subconjunto fue el que se obtuvo por medio del método de Figuereido, el subconjunto II, que logró una capacidad descriptiva levemente mejor pero utiliza muchos más descriptores. Finalmente, utilizando una cantidad intermedia

de descriptores (entre 8 y 15) se encontró que los conjuntos IV, V y VI alcanzan una mejor capacidad descriptiva cualquiera sea el método de validación de la segunda fase. Además, estos subconjuntos también alcanzan los más altos coeficientes de determinación. La figura 5.9 muestra gráficamente la relación entre el número de descriptores seleccionados y el error cuadrado medio de predicción para los subconjuntos reportados en la tabla 5.12.

				Método de validación			
	Método de	Función de	Número de	ANNE		MLR	
ID	búsqueda	regresión	descriptores	MSEP	R^2	MSEP	R^2
VII	MCVS [1]	MLR	8	0.1191	0.6813	0.09	0.7532
VIII	Figuereido	MLR	4	0.1715	0.5733	0.138	0.6404
IX	$\text{Agr}_{\alpha=0.1}$	MLR	3	0.0984	0.7421	0.1282	0.65
X	$\text{Agr}_{\alpha=0.3}$	DT	3	0.1055	0.7092	0.1512	0.57
XI	NSGA-II	MLR	7	0.0915	0.6459	0.1112	0.6623
XII	NSGA-II	kNN	2	0.1013	0.6174	0.1374	0.6186

Tabla 5.13: Comparación de resultados para ***logHIA***. Las columnas MSEP y R^2 corresponden al error cuadrado promedio y al coeficiente de determinación obtenidos sobre el conjunto de validación. El subconjunto VII corresponde al mejor subconjunto reportado en [KLVHC08], mientras que el subconjunto VIII se obtuvo mediante el método de Figuereido [Fig03].

La tabla 5.13 muestra los resultados obtenidos sobre el conjunto de datos ***logHIA*** y en la figura 5.10 se observar el número de características seleccionadas con respecto al error cuadrado medio de predicción respectivo para cada subconjunto.

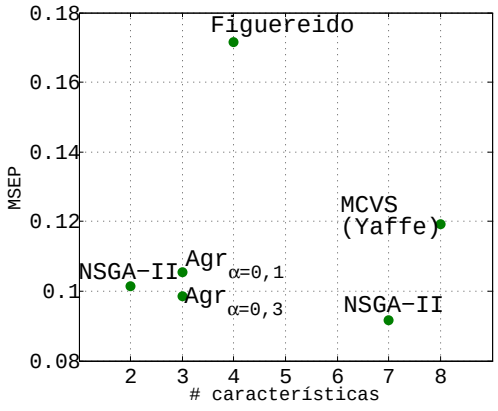

Figura 5.10: Relación *Número de descriptores* vs. *MSEP* para el conjunto de datos ***logHIA***.

Para este conjunto de datos se puede ver que, aunque no se encontraron subconjuntos mejores en cuanto a capacidad predictiva que los reportados por Konovalov (subconjunto

[1] Monte Carlo Variable Selection.

VII), se encontraron subconjuntos muy interesantes. Los subconjuntos IX, X y XII alcanzan una capacidad predictiva levemente peor, pero contienen considerablemente menos descriptores. El subconjunto XI logra una capacidad descriptiva comparable y usa un descriptor menos que el subconjunto VII. Además, cabe aclarar que distinto a la metodología de Konovalov, durante estas investigaciones no se pre-seleccionó ningún otro descriptor (a excepción de *Iv* en ***logBBB***). Por otro lado, se puede notar que el subconjunto obtenido mediante el método de Figuereido fue superado por todos los demás subconjuntos. Finalmente, la mayoría de los subconjuntos tuvieron un poco menos de éxito cuando se usó MLR para el cálculo de la capacidad predictiva.

El conjunto de datos ***logP*** resulta un desafío aún más interesante que los anteriores debido a que la complejidad para modelar la propiedad hidrofobicidad es mayor. En este conjunto de datos, es importante aclarar que las comparaciones con el trabajo de Yaffe son algo complicadas de establecer, ya que se utilizaron diferentes técnicas de predicción y validación. Además, en los experimentos de esta tesis se tuvieron en cuenta otros descriptores que no fueron involucrados en el trabajo de Yaffe [YCE+02]. Sin embargo, con el fin de evaluar los subconjuntos encontrados en los experimentos de esta tesis, se utilizaron los descriptores propuestos por Yaffe (subconjunto XIII) y sus colegas, y se les aplicó el mismo criterio de validación y el mismo método de predicción que se les aplicó a los encontrados por los wrappers propuestos aquí.

				Método de validación			
	Método de	Función de	Número de	ANNE		MLR	
ID	búsqueda	regresión	descriptores	MSEP	R^2	MSEP	R^2
XIII	GA	- [2]	12	0.247	0.884	-	-
XIII	”	”	”	0.29[3]	-	-	-
XIV	Figuereido	MLR	16	0.2052	0.9097	0.2724	0.8804
XV	$\text{Agr}_{\alpha=0.9}$	MLR	24	0.154	0.9297	0.286	0.8795
XVI	$\text{Agr}_{\alpha=0.1}$	MLR	13	0.164	0.9317	0.2617	0.8698
XVII	SPEA2	MLR	15	0.1778	0.9135	0.299	0.8649
XVIII	NSGA-II	MLR	20	0.1696	0.924	0.3426	0.8496

Tabla 5.14: Comparación de resultados para ***logP***. Las columnas MSEP y R^2 corresponden al error cuadrado promedio y al coeficiente de determinación obtenidos sobre el conjunto de validación. El subconjunto XIII corresponde al mejor subconjunto reportado en [YCE+02], mientras que el subconjunto XIV se obtuvo mediante el método de Figuereido [Fig03].

[2] Función de regresión no reportada en el trabajo original.

[3] Resultado reportado en el trabajo original usando un conjunto de validación fijo (sin validación cruzada).

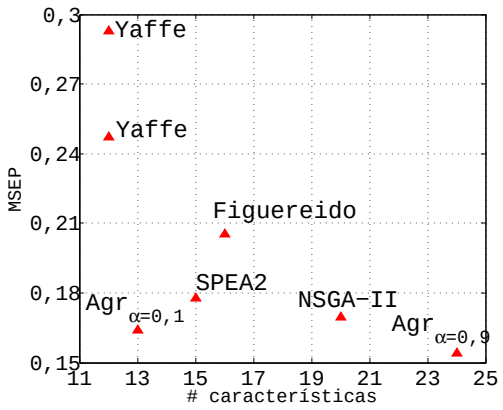

Figura 5.11: Relación *Número de descriptores* vs. *MSEP* para el conjunto de datos ***logP***.

La tabla 5.14 muestra los resultados correspondientes a los mejores subconjuntos reportados para cada método para el conjunto de datos ***logP***. Se puede observar que todos los subconjuntos de descriptores sugeridos por el método de dos fases (subconjuntos XV, XVI, XVII y XVIII) mejoran la capacidad descriptiva del subconjunto XIII. Aunque dichos subconjuntos aumentaron el número de descriptores la diferencia obtenida en el MSEP es considerable. Por otra parte, la capacidad predictiva del subconjunto obtenido con el método de Figuereido es mejor que la del subconjunto XIII, aunque es peor que los subconjuntos propuestos aquí. Este análisis se puede observar gráficamente en la figura 5.11. Finalmente, se puede observar que para este conjunto de datos los MSEP obtenidos usando MLR en la segunda fase son notablemente peores que los obtenidos usando ANNE. Esto evidencia que la relación entre los descriptores y la propiedad ***logP*** es altamente no lineal.

Análisis y comparación entre los diferentes wrappers

Durante estas pruebas se consideró el percentil 50 de los datos con el fin de eliminar los efectos de los subconjuntos de muy baja capacidad predictiva. La tabla 5.15 resume los valores promedio para la capacidad predictiva, es decir, el MSEP promedio del percentil 50 para cada wrapper y para cada conjunto de datos (***logBBB, logHIA y logP***). Se puede observar que para ***logBBB*** el mejor resultado se obtuvo cuando se combinaron el esquema agregativo (con $\alpha = 0.7$) y MLR. Además, MLR parece lograr los mejores resultados sin importar el método de búsqueda. Para el conjunto ***logHIA*** los mejores resultados se logran al combinar SPEA2 y MLR. En ninguno de estos dos conjuntos de datos se observa un esquema de búsqueda que tenga éxito sin importar cuál método de

predicción se utilice. Por último, para ***logP*** el menor MSEP promedio se logra al combinar el esquema agregativo (con $\alpha = 0.9$) con MLR, además se puede apreciar que el esquema agregativo supera a ambos esquemas basados en Pareto, sin importar cuál sea el método de predicción utilizado.

Conjunto	Método de búsqueda	DT	kNN	NLR	MLR
logBBB	Agregativo [4]	0.1504	0.1486	0.1462	0.1261
	NSGA-II	0.1437	0.1382	0.1454	0.1277
	SPEA2	0.1385	0.1361	0.1368	0.1269
logHIA	Agregativo [5]	0.1211	0.1285	0.1212	0.1161
	NSGA-II	0.1049	0.1052	0.105	0.101
	SPEA2	0.1018	0.1104	0.1064	0.0982
logP	Agregativo [6]	0.1881	0.1855	0.1877	0.1787
	NSGA-II	0.2645	0.2592	0.3080	0.2222
	SPEA2	0.2120	0.2067	0.2073	0.1963

Tabla 5.15: Rendimiento promedio de las funciones de regresión: MSEP promedio para el percentil 50 de cada combinación.

Para realizar un análisis estadístico más preciso, el primer paso es realizar una prueba de ANOVA con el fin de evaluar si existen diferencias significativas entre las varianzas de los resultados obtenidos por cada wrapper. Teniendo en mente dos factores de interés, *el método de predicción* y *el algoritmo de búsqueda*, se realizó una prueba de ANOVA de dos direcciones para cada conjunto de datos considerando el 10% de los datos (aproximadamente de 20 a 26 subconjuntos) para cada wrapper y especificando el *método de predicción* como factor A y el *algoritmo de búsqueda* como factor B. Esta prueba permite estudiar si existe diferencia entre las medias con respecto al factor A, si existe diferencia entre las medias con respecto al factor B y si existe interacción entre el factor A y el B. Los cuatro grados de libertad ($d.f._A$, $d.f._B$, $d.f._{AB}$ y $d.f._{Error}$) se calculan en base al número de grupos considerados en el *factor A* (k_a), el número de grupos considerados en el *factor B* (k_b) y la cantidad de observaciones de cada grupo (n) de la siguiente forma:

- $d.f._A = k_a - 1$,
- $d.f._B = k_b - 1$,
- $d.f._{AB} = (k_a - 1)(k_b - 1)$ y finalmente
- $d.f._{Error} = k_a k_b (n - 1)$.

[4] Usando $\alpha = 0.7$.
[5] Usando $\alpha = 0.3$.
[6] Usando $\alpha = 0.9$.

Resultados: conjunto de datos *logBBB*. Para los experimentos con el conjunto ***logBBB*** los valores k_x y los grados de libertad son: $k_a = 4$ (*DT, kNN, MLR* y *NLR*), $k_b = 3$ (*Agregativo, NSGA-II* y *SPEA2*), $n = 26$ (10% superior de los subconjuntos resultantes), $d.f._A = 3$, $d.f._B = 2$, $d.f._{AB} = (k_a - 1)(k_b - 1) = 3 * 2 = 6$ y finalmente $d.f._{Error} = k_a k_b (n - 1) = 4 * 3 * 25 = 300$. La tabla 5.16 muestra los resultados de la

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
Método de regresión	0,082038033	3	0,027346011	279,2476708	0
Algoritmo de búsqueda	7,26E-05	2	3,63E-05	0,370648788	0,690601994
Interacción	0,006339951	6	0,001056659	10,79021865	7,17E-11
Error	0,029378234	300	9,79E-05		
Total	0,117828811	311			

Tabla 5.16: Resultados de la prueba ANOVA de dos direcciones para ***logBBB***.

prueba para el conjunto de datos ***logBBB***. El valor tabular de F para $\alpha = 5\%$ para el factor A ($F_{(k_a-1),k_a k_b(n-1)} = F_{3,300}$) es 2,6347, para el factor B ($F_{(k_b-1),k_a k_b(n-1)} = F_{2,300}$) es 3,0258 y para el factor de interacción AB ($F_{(k_a-1)(k_b-1),k_a k_b(n-1)} = F_{6,300}$) es 2,1288. En base a los valores tabulares se puede deducir que es posible rechazar la hipótesis nula de que no hay diferencias entre las medias con respecto al factor A (*el método de regresión*) con un nivel de significación $\alpha = 5\%$, es decir, existe una diferencia significativa sobre los métodos de regresión. Además, también se puede notar que existe interacción entre ambos factores ($F_{interaccion} = 10,79021865 > 2,1288$).

Una segunda prueba de dos direcciones ANOVA muestra que no se encuentran diferencias significativas entre las medias del factor B (*algoritmos de búsqueda*) cuando se consideran sólo los algoritmos basados en Pareto (sólo NSGA-II y SPEA2) y tampoco pueden hallarse efectos de interacción significativos entre los dos factores. Sin embargo, se aprecian diferencias significativas según qué método de regresión se aplica. Los resultados de esta prueba se encuentran en la tabla 5.17. En este caso los valores tabulares de F para $\alpha = 5\%$ son: $F_{3,200} = 2,649752$ (factor *método de regresión*), $F_{1,200} = 3,888375$ (factor *algoritmo de búsqueda*) y $F_{3,200} = 2,649752$ (interacción).

Dado que ambas pruebas muestran que existen diferencias significativas con respecto al método de regresión utilizado y que en la segunda prueba no se pudo encontrar diferencia significativa entre NSGA-II y SPEA2. Se realizaron dos pruebas de comparación múltiple de Tukey-Kramer para determinar cómo es la diferencia que se detecta entre los métodos de regresión, una considerando los subconjuntos resultantes de la búsqueda de los algoritmos basados en Pareto y otra considerando los subconjuntos encontrados

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
Método de regresión	0,068233635	3	0,022744545	215,8800971	0
Algoritmo de búsqueda	6,30E-06	1	6,30E-06	0,059771489	0,807108114
Interacción	0,000559012	3	0,000186337	1,768621963	0,154397011
Error	0,021071461	200	0,000105357		
Total	0,089870404	207			

Tabla 5.17: Resultados de la prueba ANOVA de dos direcciones para ***logBBB*** considerando sólo los algoritmos de búsqueda NSGA-II y SPEA2.

por el esquema agregativo (con $\alpha = 0,7$, es decir el correspondiente a ***logBBB*** en tabla 5.15). Mediante la prueba de Tukey-Kramer es posible determinar qué pares de grupos muestran diferencias estadísticas entre sí. En esta prueba se calculan las medias de cada grupo y se arman todas las posibles combinaciones de grupos de a dos. De esta manera, si hay k grupos a contrastar habrá $k(k-1)/2$ pares a comparar. En este caso se quieren hacer comparaciones entre los resultados de los cuatro métodos de predicción utilizados en el wrapper ($k = 4$) por lo que se realizan 6 comparaciones. La figura 5.12 muestra los

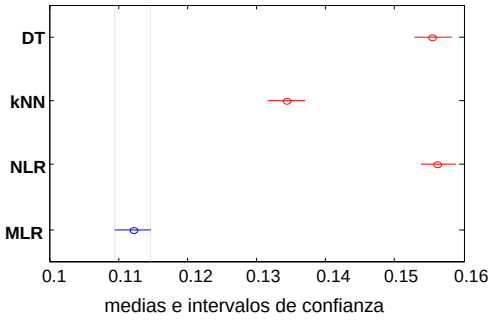

Figura 5.12: Prueba de comparación entre los métodos de regresión para las estrategias basadas en *Pareto* usando la prueba de Tukey-Kramer para $\alpha = 5\%$ para el conjunto ***logBBB***.

resultados de la prueba de Tukey-Kramer sobre los cuatro métodos de regresión para los subconjuntos obtenidos con los algoritmos basados en Pareto para un nivel de confianza $\alpha = 5\%$. Los círculos marcan la media de cada grupo y alrededor de la misma se puede apreciar el intervalo de confianza correspondiente. Se puede apreciar que se forman tres grupos que difieren significativamente entre sí dado que sus intervalos de confianza no se solapan: *"MLR"*, *"kNN"* y *"DT, NLR"*. En particular, se puede deducir que los mejores valores se obtienen utilizando el método MLR. La misma prueba se aplicó sobre los subconjuntos obtenidos con el esquema agregativo. La figura 5.13 muestra estos resultados. Nuevamente se puede ver que el grupo MLR es significativamente diferente de los demás grupos y que logra las mejores predicciones. Luego de los resultados estadísticos obtenidos

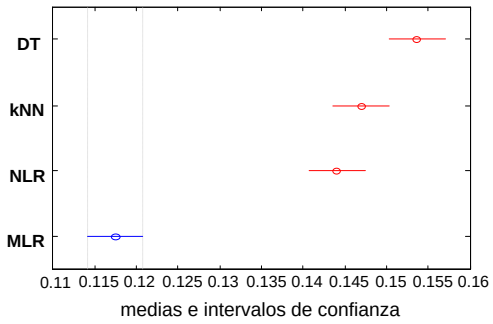

Figura 5.13: Prueba de comparación entre los métodos de regresión para la estrategia **agregativa** ($\alpha = 0.7$) usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logBBB***.

con la prueba de Tukey-Kramer se puede llegar a la conclusión general de que el método de regresión que logra mejor rendimiento sin importar el algoritmo de búsqueda es MLR. Finalmente, se realizó un análisis específico con respecto a los algoritmos de búsqueda utilizando sólo los subconjuntos obtenidos con MLR. Para esto se realizó una prueba de ANOVA de un sentido (tabla 5.18) para las tres estrategias de búsqueda con respecto a MLR para saber si existen diferencias significativas entre las distintas estrategias de búsqueda cuando se usa MLR como método de predicción. El valor tabular de F es $F_{2,75} = 3,119$ para $\alpha = 5\%$, por lo que según la prueba podemos rechazar la hipótesis

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	*p*−value
Algoritmo de búsqueda	0,000527384	2	0,000263692	5,08297517	0,008507902
Error	0,003890815	75	5,19E-05		
Total	0,004418199	77			

Tabla 5.18: Resultados de la prueba ANOVA de un sentido para ***logBBB*** considerando los tres algoritmos de búsqueda con respecto a MLR como método de regresión.

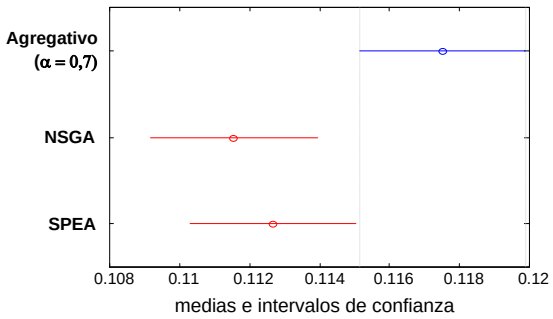

Figura 5.14: Comparación entre los tres algoritmos de búsqueda con respecto a MLR usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logBBB***.

nula de que las medias de los distintos grupos son iguales con un 95% de confianza y concluir que los grupos expresan diferencias significativas. En base a esto se efectuó una última prueba de Tukey-Kramer para detectar cómo se producen tales diferencias, cuyos resultados se muestran en la figura 5.14. Con esta prueba se pudo identificar una diferencia significativa entre la estrategia agregativa y los algoritmos basados en Pareto, resultando estas últimas en un mejor desempeño con respecto a la capacidad predictiva y no significativamente diferentes entre sí.

Resultados: conjunto de datos *logHIA*. Sobre este conjunto de datos se realizaron análisis similares a los realizados sobre el conjunto de datos anterior y los resultados fueron muy similares. Para efectuar la prueba de ANOVA de dos direcciones, los valores de los parámetros y los tabulares de F fueron los mismos que para ***logBBB*** ($F_{3,300} = 2,6347$ para el factor A, $F_{2,300} = 3,0258$ para el factor B y $F_{6,300} = 2,1288$ para el factor de interacción AB).

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
Método de regresión	0,027669711	3	0,009223237	71,76517744	0
Algoritmo de búsqueda	0,01248564	2	0,00624282	48,57482001	0
Interacción	0,005038979	6	0,00083983	6,534639875	1,69E-06
Error	0,038555901	300	0,00012852		
Total	0,083750231	311			

Tabla 5.19: Resultados de la prueba ANOVA de dos direcciones para ***logHIA***.

La tabla 5.19 muestra los resultados obtenidos para $\alpha = 5\%$. Como se puede ver, en esta ocasión las tres hipótesis nulas pueden ser rechazadas con un 95% de confianza, es decir, las medias difieren significativamente con respecto a los *métodos de regresión*, las medias difieren significativamente con respecto a los *algoritmos de búsqueda* y existe interacción entre ambos factores. Una segunda prueba de ANOVA considerando los subconjuntos

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
Método de regresión	0,02339705	3	0,007799017	68,53985761	0
Algoritmo de búsqueda	2,75E-05	1	2,75E-05	0,24187026	0,623397904
Interacción	0,000897808	3	0,000299269	2,630059735	0,051292533
Error	0,02275761	200	0,000113788		
Total	0,04707999	311			

Tabla 5.20: Resultados de la prueba ANOVA de dos direcciones para ***logHIA*** considerando sólo los algoritmos de búsqueda NSGA-II y SPEA2.

de las estrategias basadas en Pareto demuestra que además, hay interacción entre ambos

esquemas Pareto (tabla 5.20). Por lo tanto, se realizó una prueba de Tukey-Kramer para cada una de las estrategias de búsqueda por separado para determinar en forma más precisa cómo se produce la diferencia entre los métodos de predicción usados.

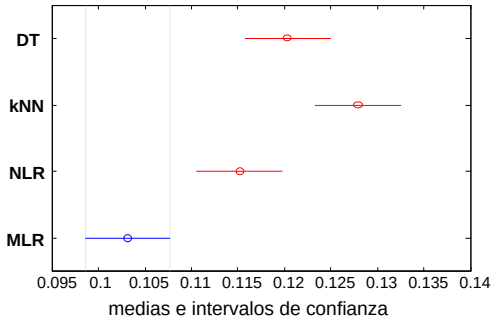

Figura 5.15: Prueba de comparación entre los métodos de regresión con respecto a la estrategia ***agregativa*** usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logHIA***.

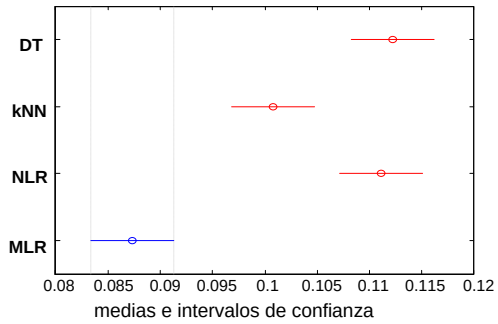

Figura 5.16: Prueba de comparación entre los métodos de regresión con respecto al algoritmo ***NSGA-II*** usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logHIA***.

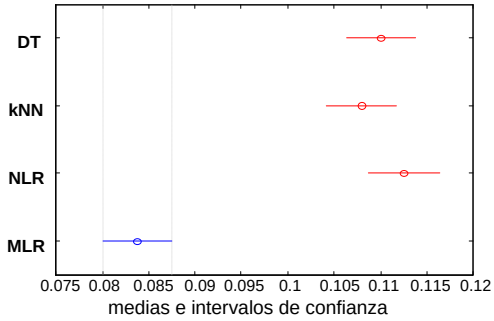

Figura 5.17: Prueba de comparación entre los métodos de regresión con respecto al algoritmo ***SPEA2*** usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logHIA***.

Las figuras 5.15, 5.16 y 5.17 muestran las medias y los intervalos de confianza para los

cuatro grupos correspondientes a los distintos métodos de regresión cuando se utiliza el algoritmo evolutivo *agregativo*, el *NSGA-II* y el *SPEA2* respectivamente. En todos los casos se puede notar que DT, kNN y NLR difieren notablemente de MLR, el cual demuestra tener el mejor desempeño para las tres estrategias de búsqueda. En particular para el algoritmo NSGA-II el método de regresión kNN logra resultados significativamente diferentes y mejores a los de DT y NLR. Como se puede observar, el método de regresión MLR demuestra ser el más prometedor para todas las estrategias de búsqueda. Para determinar si existen diferencias significativas entre las distintas estrategias de búsqueda cuando se utiliza MLR como método de regresión se llevó a cabo una nueva prueba de ANOVA de un sentido sobre estos resultados (tabla 5.21). El valor tabular de F es $F_{2,75} = 3,119$ para $\alpha = 5\%$, mientras que el valor de F obtenido por la prueba estadística es 45,27973793, lo cual confirma la existencia de valores significativamente diferentes entre las medias de cada grupo. Para saber cómo se producen tales diferencias se efectuó una prueba Tukey-Kramer considerando todas las estrategias de búsqueda con respecto al método de regresión MLR. Esta prueba demostró que, estadísticamente, las estrategias basadas en Pareto superan significativamente al algoritmo agregativo y que no hay diferencias significativas entre NSGA-II y SPEA2 (figura 5.18), al igual que lo ocurrido con el conjunto de datos ***logBBB***.

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	*p*−value
Algoritmo de búsqueda	0,0055552	2	0,0027776	45,27973793	1,27E-13
Error	0,004600733	75	6,13E-05		
Total	0,010155933	77			

Tabla 5.21: Resultados de la prueba ANOVA de un sentido para ***logHIA*** considerando los tres algoritmos de búsqueda con respecto a MLR como método de regresión.

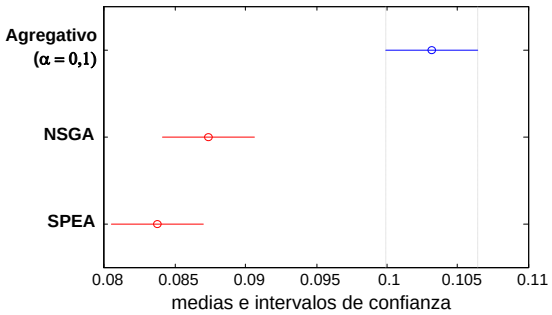

Figura 5.18: Comparación entre los tres algoritmos de búsqueda con respecto a MLR usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logHIA***.

Resultados: conjunto de datos *logP*. Se aplicó una prueba de ANOVA de dos direcciones considerando todas las posibles combinaciones para el wrapper. Los valores de los parámetros son: $k_a = 4$ (*DT, kNN, MLR* y *NLR*), $k_b = 3$ (*Agregativo, NSGA-II* y *SPEA2*), $n = 20$ (10% superior de los subconjuntos resultantes), $d.f._A = 3$, $d.f._B = 2$, $d.f._{AB} = (k_a - 1)(k_b - 1) = 3 * 2 = 6$ y finalmente $d.f._{Error} = k_a k_b (n-1) = 4 * 3 * 19 = 228$ y los tabulares de F fueron: $F_{3,228} = 2,644194$ para el factor A (*métodos de regresión*), $F_{2,228} = 3,035441$ para el factor B (*algoritmos de búsqueda*) y $F_{6,228} = 2,138491$ para el factor de interacción AB.

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	$p-$value
Método de regresión	0,081995587	3	0,027331862	109,2832503	0
Algoritmo de búsqueda	0,080584588	2	0,040292294	161,1040167	0
Interacción	0,030189168	6	0,005031528	20,11797525	0
Error	0,057023054	228	0,000250101		
Total	0,249792396	239			

Tabla 5.22: Resultados de la prueba ANOVA de dos direcciones para ***logP***.

La tabla 5.22 muestra los resultados obtenidos para $\alpha = 5\%$. A partir de esta prueba se puede concluir que las medias difieren significativamente con respecto a los *métodos de regresión* y con respecto a los *algoritmos de búsqueda*. Además, observamos que existe interacción entre ambos factores. En base a estos resultados se realizó una prueba de Tukey-Kramer para cada estrategia de búsqueda buscando cómo se producen las diferencias entre los métodos de regresión.

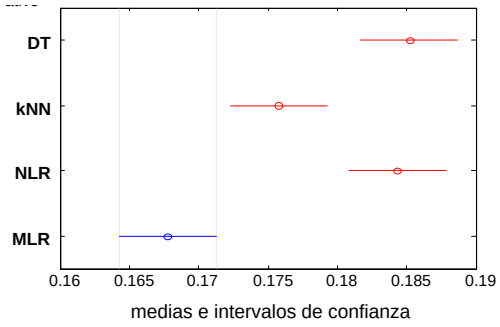

Figura 5.19: Prueba de comparación entre los métodos de regresión con respecto a la estrategia agregativa usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***.

Las figuras 5.19, 5.20 y 5.21 muestran las medias y los intervalos de confianza de los cuatro grupos (DT, kNN, NLR y MLR) con respecto a cada estrategia de búsqueda. Como se puede observar, MLR es el mejor método de regresión para la estrategia agregativa

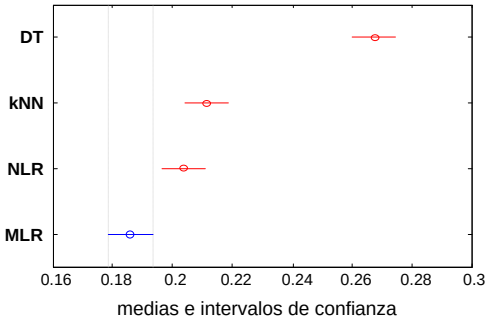

Figura 5.20: Prueba de comparación entre los métodos de regresión con respecto al algoritmo NSGA-II usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***.

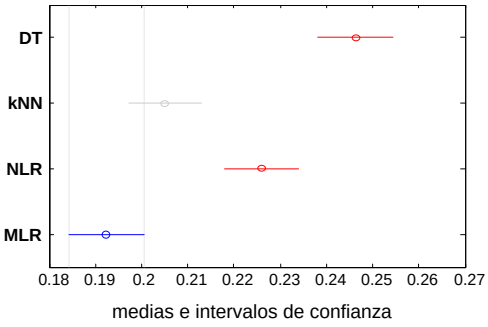

Figura 5.21: Prueba de comparación entre los métodos de regresión con respecto al algoritmo SPEA2 usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***.

y para NSGA-II, mientras que para SPEA2 hay dos mejores esquemas, MLR y kNN, que expresan diferencias estadísticamente significativas con los otros pero no entre sí (los intervalos de MLR y kNN se superponen entre sí pero no con DT ni con NLR). También se puede apreciar que el promedio de MLR es ligeramente mejor con respecto al error de predicción que el de kNN. A partir de estos resultados se hicieron dos pruebas de ANOVA de una dirección para determinar si existen diferencias significativas entre las estrategias de búsqueda, una considerando sólo el método de regresión MLR y la otra considerando sólo kNN (tablas 5.23 y 5.24). En ambos casos la prueba muestra que hay diferencias

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	*p*−value
Algoritmo de búsqueda	0,006430173	2	0,003215087	33,79667684	2,09E-10
Error	0,005422425	57	9,51E-05		
Total	0,011852598	59			

Tabla 5.23: Resultados de la prueba ANOVA de un sentido para ***logP*** considerando los tres algoritmos de búsqueda con respecto a MLR como método de regresión.

S. de *varianzas*	*S.S*	*d.f.*	*M.S.*	*F*	*p*−value
Algoritmo de búsqueda	0,014317085	2	0,007158543	41,44143045	7,73E-12
Error	0,009846111	57	0,000172739		
Total	0,024163196	59			

Tabla 5.24: Resultados de la prueba ANOVA de un sentido para ***logP*** considerando los tres algoritmos de búsqueda con respecto a kNN como método de regresión.

significativas entre las medias de cada estrategia de búsqueda, por lo que se realizaron dos pruebas de Tukey-Kramer para determinar como se producen tales diferencias. Las figuras 5.22 y 5.23 muestran las medias y los intervalos de confianza de las tres estrategias de búsqueda obtenidos con la prueba de Tukey-Kramer con un nivel confianza de 95% con respecto a MLR y a kNN respectivamente. A diferencia de lo ocurrido con ***logBBB*** y con ***logHIA*** en esta ocasión, la estrategia agregativa con $\alpha = 0.9$ parece obtener los mejores resultados, lo cual se manifiesta tanto para MLR como para kNN.

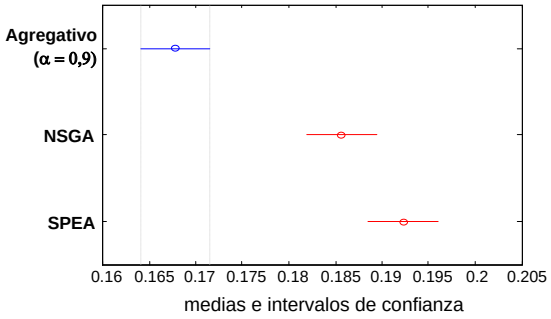

Figura 5.22: Comparación entre los tres algoritmos de búsqueda con respecto a MLR usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***.

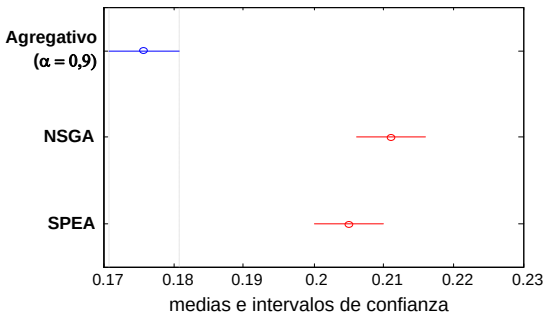

Figura 5.23: Comparación entre los tres algoritmos de búsqueda con respecto a kNN usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***.

Luego de los resultados estadísticos alcanzados, podemos concluir que la mejor opción

entre los distintos wrappers para este tercer conjunto de datos es la estrategia agregativa combinada con MLR o kNN como método de regresión.

Discusión. Se ha demostrado que en los dos primeros conjuntos de datos (***logBBB*** y ***logHIA***) las estrategias de búsqueda basadas en Pareto combinadas con el método de regresión MLR, superan al esquema agregativo. Por otro lado, en el último conjunto de datos, la estrategia agregativa con $\alpha = 0,9$ en combinación con los métodos de predicción MLR o kNN supera a los algoritmos basados en Pareto. Es muy probable que estos resultados se deban a que en ***logBBB*** y ***logHIA*** se requieren menos cantidad de descriptores que en ***logP*** para modelar el comportamiento de la propiedad de interés. Las estrategias basadas en Pareto buscan minimizar cualquiera de los objetivos en forma independiente del valor del otro objetivo. En otras palabras, un subconjunto que tiene menos descriptores que los demás individuos de la población será parte del frente de individuos no dominados, aún si la capacidad descriptiva es muy baja. Esta característica hace que estos métodos tengan cierta tendencia a encontrar subconjuntos con baja cardinalidad. Como ejemplo de este hecho, se descubrió que la cardinalidad promedio de todos los subconjuntos usando NSGA-II para el conjunto ***logP*** es de 12,71, mientras que usando un esquema agregativo con $\alpha = 0,1$ la cardinalidad promedio es 19,98 (~ 20) descriptores. Bajo estas condiciones, cuando el *número teórico óptimo* de descriptores necesarios para modelar determinado comportamiento es alto, los esquemas basados en Pareto experimentan cierta dificultad con respecto al esquema agregativo al cual se le puede especificar fácilmente la importancia de cada objetivo por medio del parámetro α.
Otro aspecto a mencionar es el buen comportamiento del método MLR en ***logBBB*** y ***logHIA***, donde se sabe que la relación entre los descriptores y la propiedad de interés es bastante lineal. Para el conjunto de datos ***logP***, el buen desempeño de MLR es interesante dado que muestra que puede ser un muy buen método de predicción aún cuando la relación entre los descriptores y la propiedad de interés es no lineal [Gha08].
En conclusión, teniendo en cuenta la diferencia entre las varianzas, se confirmó que el método de predicción aplicado afecta fuertemente la capacidad predictiva de los subconjuntos seleccionados. Por otro lado, la estrategia de búsqueda también tiene un fuerte impacto, un hecho reflejado en el análisis estadístico realizado y en las diferencias encontradas entre los primeros dos conjuntos y el tercero.
Finalmente, se pudo establecer que los resultados obtenidos son altamente competentes con los reportados en la literatura y que la mejor combinación entre métodos de regresión

y algoritmos de búsqueda para el wrapper multi-objetivo depende del conjunto de datos bajo estudio.

Análisis de los subconjuntos de descriptores obtenidos

Como puede suponerse, de acuerdo con la naturaleza paralela y estocástica de la propuesta, los subconjuntos de descriptores obtenidos al final de la segunda fase pueden ser diferentes entre sí en una misma corrida y no necesariamente los mismos a través de distintas ejecuciones. Si bien puede pensarse que una característica de robustez podría exigir que el método sugiera un único subconjunto de descriptores como el *mejor subconjunto*, la sugerencia simultánea de varios subconjuntos puede verse como una ventaja con respecto a muchos de los esquemas existentes para FS. Según Horvath este aspecto estocástico puede representar la posibilidad de ofrecer más de un subconjunto de descriptores relevantes [HBSe$^+$07]. Durante los experimentos realizados en estas investigaciones, si bien los subconjuntos seleccionados no han sido exactamente los mismos, es notable que varios descriptores que han resultado seleccionados en repetidas ocasiones, han sido reportados como relevantes en otros trabajos de selección de características o se sabe en forma teórica que son relevantes.

Como análisis breve de los descriptores seleccionados en los mejores subconjuntos, los cuales se reportan en la tabla 5.25, el descriptor *TPSA(NO)* (área de la superficie topológica usando nitrógeno y contribuciones de oxígeno polares), el cual tiene mucha importancia en la predicción de la penetración de la barrera hematoencefálica [Ert08], se encuentra presente en la mayoría de los subconjuntos para el conjunto de datos ***logBBB***. En general, los compuestos lipofílicos tienen probabilidades de cruzar la barrera, por lo que es aceptable que se encuentren con frecuencia descriptores hidrofóbicos (como *ALOGP* - coeficiente de Ghose-Crippen de partición octanol-agua) y descriptores de grupos de ácido carbóxilico (como *nRCOOH* o *Ic*).

En el caso de ***logHIA*** se encuentran presentes frecuentemente descriptores relacionados con la solubilidad en agua, como *ALOGP* o *MLOGP* (coeficiente de Moriguchi de partición octanol-agua). En el caso del conjunto de datos ***logP***, se pueden ver frecuentemente seleccionados algunos descriptores conocidos por su relación con la predicción de dicha propiedad, por ejemplo: *MW* (peso molecular), descriptores relacionados con el carbono (*nC* - número de átomos de carbono; *nCar* - suma de todos los átomos de carbono pertenecientes a cualquier estructura aromática y heteroaromática; *nCs* - número total

de átomos de carbono secundarios) y descriptores relacionados con momentos dipolares (D_H - dipolo total, hibridación; o D_S - dipolo total, hibridación + punto de carga).

Conjunto	ID	Descriptores [7]
logBBB	I	[Iv], [TPSA(NO)], [Ic], SRW09, BELv4, HATS7v / HATS8e
	II	[Iv], TPSA(NO), MLOGP2, SRW09, nROH, EEig05d, C-034, nRCOOH, O-057, nArNR2, nArCOOH, H-051, Psychotic-80, nRCOOR, HATS8u, nN(CO)2, Infective-50, HATS7u, nArCOOR, Deppresant-50
	III	[Iv], TPSA(NO), ALOGP, Mor21v, EEig12r, Ic
	IV	[Iv], TPSA(NO), SRW07, O-057, MATS2p, nOHp, R3u, RDF020p, T(N..I), nArOH, Psychotic-80, RDF050m, HATS8u, Cl-087, G3u
	V	[Iv], TPSA(NO), ALOGP, Mor16v, ISIZ, nN(CO)2, BELm4, Ic
	VI	[Iv], TPSA(NO), ALOGP, R7u+, ARR, H4p, Mor20v, TE2, BELe3
logHIA	VII	[ALOGP], LAI, Neoplastic-80, RDF045m, R5v+, DDI, N-074, IDE
	VIII	Hy, GVWAI-80, Infective-80, nP($\bar{\text{O}}$)O2R
	IX	MLOGP, TPSA(NO), RDF130m
	X	ALOGP, C-011, nPyridines
	XI	ALOGP, Mor13e, RDF045p, Vs, nArCONHR, R5uþ, PW5
	XII	ALOGP, R1v+
	XIII	MW, D_P, D_H, D_S, E2, EX, ELC, IP, PO, VMC1, VMC2, VMC4
logP	XIV	D_H, IP, PO, Mv, nH, nN, nCL, nR06, nCp, nCaR, nOHp, nOHs, nROR, nRSR, nHDon, PSA
	XV	D_S, E2, IP, Sv, Se, nAT, nBM, nDB, nAB, nH, nC, nO, nF, nCL, nI, nR03, nR06, nR11, nCp, nCs, nOHp, nOHs, nOHt, ARR
	XVI	D_H, IP, PO, nBT, nBM, nAB, nO, nF, nX, nR06, nCaR, nHAcc, Ui
	XVII	MW, E2, AMW, Mv, Mp, nBT, SCBO, nAB, nC, nBR, nCp, nCaR, nOHs, nHDon, ARR
	XVIII	MW, E2, PO, VMC2, AMW, Mv, nBM, SCBO, nDB, nH, nC, nBR, nR03, nR06, nCp, nCaR, nOHp, nOHs, nHDon, ARR

Tabla 5.25: Descriptores moleculares seleccionados por los diferentes subconjuntos reportados en las tablas 5.12, 5.13 y 5.14

Análisis de la metodología utilizando validación externa

Todos los resultados reportados hasta aquí han sido obtenidos a partir de los errores de predicción reportados por la red neuronal de la segunda fase. El principal objetivo de esta fase es el de proporcionar un método de regresión más poderoso para evaluar la capacidad descriptiva de los subconjuntos seleccionados por la primera, pero no pretende constituir una metodología de validación que estime la capacidad predictiva real de los modelos QSAR/QSPR obtenidos a partir de dichos subconjuntos de descriptores.

[7] Los corchetes denotan descriptores que fueron previamente seleccionados.

Se puede observar que la segunda fase valida internamente utilizando datos que han sido previamente utilizados para la selección de características en la etapa previa. Por este motivo esta segunda fase no pretende ser una validación estricta, sino una validación interna cuyos resultados sean altamente confiables.

El último análisis de desempeño consistió en realizar un procedimiento de validación externa. Esto significa que una porción de los datos se reservó para ser utilizada sólo por un método de evaluación aplicado después de la segunda fase. Además, dichos datos no se superponen con ninguno de los grupos de datos usados durante la infraestructura de dos fases. Por medio de esta validación se cuantificó cuán diferentes son los errores de predicción con respecto a los reportados por la segunda fase.

Para llevar a cabo la validación externa se seleccionó de manera aleatoria un 20% de los datos antes de la primera fase (S_1). Se aplicó el procedimiento convencional con la primera y segunda fase sobre el restante 80% de los datos. Por medio de la infraestructura se obtuvieron los 20 subconjuntos de descriptores más prometedores, evaluando la capacidad predictiva de estos subconjuntos por medio de la segunda fase como en los análisis previos. Posteriormente, se evaluó la capacidad predictiva de dichos subconjuntos sobre el conjunto de prueba S_1, utilizando la misma red de la segunda fase.

Para los 20 mejores subconjuntos de descriptores se analizó si los errores promedio obtenidos en S_1 no son significativamente peores que los obtenidos por la validación interna. El procedimiento estándar para comparar las medias de ambos métodos de validación se realizó utilizando la prueba de comparación t-student para muestras correlacionadas, evaluando la capacidad predictiva de cada subconjunto por medio de ambas validaciones, calculando las diferencias entre ambas evaluaciones para cada subconjunto y finalmente obteniendo la diferencia promedio como se muestra en la tabla esquemática. Cuando el 0 está fuera del intervalo de confianza podemos afirmar (con una probabilidad de error α) que hay una diferencia entre las medias de ambas observaciones. De esta forma, se estableció el intervalo de confianza para los promedios de las diferencias entre los errores promedios de ambos métodos de validación.

Una vez realizado este procedimiento pueden ocurrir tres situaciones diferentes:

1. los errores de validación externa son significativamente menores que los errores de validación interna,
2. no hay evidencia estadística de que los errores de validación externa son mayores que los errores de validación interna, o
3. algo diferente de las situaciones 1 o 2.

	validación interna	validación externa	diferencia
Subconjunto 1	Error I_1	Error E_1	$D_1 = I_1 - E_1$
...	...	...	...
Subconjunto 20	Error I_{20}	Error E_{20}	$D_{20} = I_{20} - E_{20}$
Promedio			$\bar{\theta} = \frac{1}{20} \sum_{i=1}^{20} D_i$
Desvío estándar			$\sigma(\bar{\theta})$
Nivel de confianza			α
Grados de libertad			$f = (20 - 1)$
Valor crítico de t-student			$t_{\alpha,f}$
Estimación del intervalo de confianza			$\lfloor \bar{\theta} - t_{\alpha,f}\sigma(\bar{\theta}), \bar{\theta} + t_{\alpha,f}\sigma(\bar{\theta}) \rfloor$

Tabla 5.26: Procedimiento de comparación t-student entre dos métodos con respecto al mismo conjunto de datos muestrales.

La situación 1 puede ser identificable (con $\alpha \leq 0,05$ o $\alpha \leq 0,01$) y decimos que los errores de validación externa son menores si el intervalo sólo contiene valores positivos. Sin embargo, no es posible cuantificar probabilísticamente la segunda situación. El procedimiento usual sería aceptar la hipótesis de que la validación externa no es mayor cuando no se la puede rechazar con una probabilidad $\alpha \leq 0,025$ o cuando el error de validación interna promedio es mayor que el error de validación externa y el 0 está contenido en el intervalo. En cualquier otro caso se debería asumir que el error de validación externa es mayor que el de validación interna.

	Int.Val. > Ext.Val. (1)		Int.Val. = Ext.Val. (2)	Int.Val. < Ext.Val. (3)
	$\alpha \leq 0.05$	$0.05 < \alpha \leq 0.1$		
logBBB	40%	10%	40%	10%
logHIA	10%	20%	70%	0%
logP	20%	10%	50%	20%

Tabla 5.27: Análisis de la prueba de comparación t-student para los tres conjuntos de datos para 10 replicas: (1) el error de validación externa es significativamente menor que el error de validación interna, (2) no hay diferencia estadísticamente significativa entre ambos procedimientos de validación y (3) algo distinto de (1) y (2).

Para cada conjunto de datos, se realizaron 10 réplicas del mismo experimento aplicando este procedimiento sobre el conjunto de prueba S_1, realizando diferentes separaciones de los datos cada vez. En cada ocasión se utilizó la combinación *algoritmo de búsqueda agregativo/MRL* para la primera fase. La tabla 5.27 resume los resultados obtenidos clasificados según el tipo de situación. En 11 de las 30 réplicas se produjo la situación 1, en 16 se produjo la situación 2 y en las restantes 3 se produjo la situación 3. Estos resultados

evidencian que la validación de la segunda fase es un buen estimador de la capacidad predictiva de los subconjuntos, lo cual sustenta el uso de una metodología menos rigurosa en la primera fase de la infraestructura que sea capaz de tener en cuenta la cardinalidad y la capacidad predictiva de los subconjuntos.

Complejidad computacional de la metodología

En un análisis de la complejidad computacional de esta metodología, el primer aspecto a tener en cuenta es que está limitada por la complejidad computacional del método wrapper MO, dado que es el encargado de realizar el trabajo más costoso. Particularmente, los algoritmos basados en Pareto son más costosos computacionalmente que los agregativos debido al manejo de los frentes de no dominación.
Se sabe que, para k funciones objetivo a optimizar y N individuos en la población, la complejidad computacional del NSGA-II es de $O(kN^2)$, en este caso $k = 2$ y $N = 145$. Por otro lado, en el peor caso el, tiempo computacional del SPEA2 es $O((N + \bar{N})^3)$, donde $\bar{N}$ es el tamaño de la población externa (también 145). Sin embargo, si existe una diversidad adecuada, la complejidad computacional promedio será menor $(O(log(N + \bar{N})(N + \bar{N})^2))$ [ZLT01]. Estos órdenes de ejecución están calculados con respecto a cada generación y asumiendo que no hay un costo computacional asociado al cálculo de las funciones de aptitud.
Considerar el tiempo computacional de las funciones se reduce a considerar el tiempo de evaluación de la capacidad predictiva de los subconjuntos de descriptores. A partir de observaciones empíricas, el método de predicción NLR es el más intensivo computacionalmente, teniendo un orden $O(c^3)$, donde c es el número de coeficientes a ser ajustados (en nuestro caso, $4p + 1$, donde p es el número de descriptores en el subconjunto a ser evaluado).
Para establecer la complejidad computacional en el peor caso, para cualquier combinación de método de búsqueda y método de predicción posible, definimos *O(búsqueda)* como el orden de ejecución de una generación para una estrategia de búsqueda. Similarmente, definimos como $O(F_2)$ al orden del tiempo de ejecución de una función de evaluación de la capacidad predictiva. Debido a que la función de evaluación se calcula para cada individuo en cada generación, la complejidad computacional de una generación es $O(max(N \cdot O(F_2), O(búsqueda)))$. Finalmente, la complejidad computacional de una ejecución completa del wrapper MO en el peor caso será del orden del tiempo de una generación multiplicado por el número de generaciones (siendo menor en los casos en

los que se verifique la condición de corte del algoritmo que no depende del máximo de generaciones).

Capítulo 6

Algoritmos evolutivos desarrollados para minería de texto

En este capítulo se describen los algoritmos evolutivos mono y multi-objetivo desarrollados para soporte en el área de minería de texto. Más precisamente, se aborda el problema de *recuperación de información temática*, el cual constituye una de las principales tareas requeridas en recuperación de información (IR). La metodología de trabajo comienza proponiendo una arquitectura evolutiva para recuperación temática. Como primera aproximación se propone una versión evolutiva que pretende alcanzar mucha *similitud* entre los documentos recuperados y un tema de interés. Luego evoluciona hacia una versión más compleja que contempla dos objetivos que suelen competir entre sí: *precisión* y *cobertura*. Los objetivos pragmáticos del trabajo son el diseño, implementación y evaluación de algoritmos evolutivos para lograr recuperación de material tanto relevante como novedoso, incorporando información de forma automática a medida que las generaciones avanzan, valiéndose para esta tarea de la generación inteligente de consultas.

6.1 Recuperación de información temática

La búsqueda basada en tópico es el proceso de recuperación de información basado en un contexto temático o tema de interés. Dicho tema de interés puede ser, por ejemplo, el entorno actual de un usuario utilizando una computadora, un documento inicial a partir del cual queremos obtener más información o una descripción conformada por un conjunto de términos sueltos. Este proceso puede llevarse a cabo mediante dos pasos

básicos. Primero, formulando consultas relevantes con respecto a dicho contexto temático. Segundo, presentando dichas consultas a un motor de búsqueda. A partir de estos dos pasos, es razonable que la calidad del material recuperado sea altamente dependiente de las consultas que se presentan. Este aspecto hace que la generación inteligente de consultas sea un problema de investigación importante para el área de recuperación de información. El crecimiento veloz y las cantidades enormes de información en forma electrónica han aumentado la importancia y complejidad de la recuperación de información. El recurso principal que un usuario posee para recuperar información son los motores de búsqueda. Sin embargo, los proveedores del servicio son los que determinan los mecanismos de clasificación o ranking utilizados por el servicio de búsqueda que ofrecen. Comúnmente el criterio de clasificación cambia de un servicio de búsqueda a otro. La variedad en criterios puede ser muy heterogénea, pudiendo guiarse por ejemplo por la relevancia del contenido recuperado o por la popularidad del documento encontrado (entre muchos otros ejemplos). Además, lo usual es que el criterio empleado sea opaco a la vista de quienes están utilizando la interfaz del motor de búsqueda, dejando la *formulación de consultas* como único punto de acceso a la información relevante.

Siempre que un usuario presenta una consulta a un motor de búsqueda, el motor devuelve una lista de resultados ordenados bajo alguna pauta. Esta lista de resultados, es lo que el algoritmo de clasificación interno del motor de búsqueda considera como los resultados más relevantes, siendo el primer resultado el más relevante y los demás siguiendo en orden. Dado que los criterios de clasificación de cada motor son diferentes y cada motor puede indexar distintos documentos, se sabe que una misma consulta presentada a distintos motores de búsqueda rara vez genera el mismo resultado.

A medida que el número de documentos se incrementa la tarea de retornar resultados se vuelve más compleja. Si bien se ha avanzado mucho en el desarrollo de mecanismos de clasificación que permitan obtener más resultados relevantes para el usuario, los usuarios no han modificado demasiado su habilidad. Por ejemplo, en general e históricamente los usuarios suelen mirar no más que los diez primeros resultados. Por otra parte, los usuarios tienden a formular consultas de pocas y específicas palabras o términos. Lo cual dificulta la tarea de recuperación, sobre todo cuando la palabra tiene muchos significados diferentes. En general, la expansión de consultas suele ayudar a enfocar la búsqueda en material más relevante.

La tarea de agregar términos a una consulta puede hacerse de manera manual, automática o asistida por el usuario. La efectividad de las consultas depende de lo que el usuario está

intentando obtener y de lo que el usuario quiere ver. Por ejemplo, si el objetivo es lograr una amplia cobertura, una consulta extensa será más efectiva que una específica. En este caso, el usuario intentará recuperar la mayor cantidad de información relevante, aun cuando dicha información se vea acompañada de resultados irrelevantes. Por otro lado, si la intención es lograr una buena precisión, son preferibles las consultas específicas.
El problema de la búsqueda basada en tópico puede verse como un problema de optimización si contamos con un criterio que nos permita medir el desempeño de las consultas. En este contexto, la función objetivo a ser optimizada (minimizada o maximizada) nos indicará cuán buena es una consulta de tal forma que pueda ser comparada con las demás. En las siguientes secciones veremos los diferentes esquemas desarrollados en esta tesis, cada uno de los cuales fue destinado a contestar a distintas cuestiones.

6.1.1 Antecedentes en búsqueda temática basada en contexto

Existen numerosos sistemas que demuestran la efectividad e importancia de la búsqueda temática. Algunos han sido desarrollados con propósitos de aplicación específica y otros con propósitos más generales.

- Letizia [Lie95] y el WebWatcher [AFJM95] son dos ejemplos de asistentes que colaboran con el usuario al navegar en la World Wide Web. A medida que el usuario trabaja con un navegador convencional, estos agentes llevan un registro del comportamiento del usuario e intentan anticipar que documentos le resultarán interesantes. Para ello exploran de manera concurrente y autónoma los links que el usuario va visitando. La estrategia de navegación se basa en lo que consideran más importante para el usuario, usando la forma en que el mismo ha manipulado los distintos enlaces para aprender. Usan un conjunto de heurísticas simples para modelar el posible comportamiento del usuario navegando en la web.
- El Remembrance Agent (RA) [RS96] es un programa que colabora con la memoria humana mostrando una lista de documentos que son relevantes con respecto al contexto actual del usuario. La aplicación se ejecuta sin necesitar la intervención humana de tal forma que si la persona retoma una tarea distinta el sistema también retoma las sugerencias correspondientes. La interfaz del RA trabaja dentro del editor de texto Emacs, el cual sirve además como cliente de email y navegador. La interfaz muestra las sugerencias junto con un puntaje que indica el grado de relevancia de cada una. Por otra parte, un programa de soporte en el RA es el

encargado de resolver las consultas, sugiriendo los documentos similares encontrados dentro de un pool previamente indexado. Posteriormente Rhodes creó una versión portable (en el sentido de portar en el cuerpo, vestir) del RA, el cual trabaja como un asistente electrónico que el usuario puede llevar a cualquier lado sin necesidad de una computadora [Rho97].

- HyPursuit [WVS96] es un sistema que combina la información brindada tanto por los links como por el contenido de los documentos, con el fin de dar una estructura definida al espacio de información. Organiza hypertexto en clusters dentro de una o más jerarquías, basándose en diferentes técnicas de agrupación de documentos.
- Un esquema denominado Análisis de Similitud Generalizado (GSA por Generalized Similarity Analysis), presentado por Chen [Che97], combina los enlaces, la similitud de contenido y el uso de patrones para definir relaciones de proximidad.
- El SenseMaker [BW97] es una interfaz centrada en el usuario que facilita la navegación de espacios de información dentro de librerías de información digital heterogéneas. Unifica citas y artículos de distintas fuentes y los presenta en un esquema que permite una rápida comparación de sus propiedades. El sistema permite que los usuarios examinen su contexto actual, se muevan hacia nuevos contextos o vuelvan a contextos previos. Presenta los documentos sugeridos en forma agrupada o en bundles (el término que ellos utilizan para denominar clusters), los cuales pueden ser progresivamente expandidos, facilitando una forma guiada por el usuario para búsqueda incremental. Actualmente este sistema se ha convertido en una serie de herramientas que agrupa, entre otros, un recolector de información, un navegador, una aplicación para representación de modelos y un clasificador.
- El PAW (Personal Adaptative Web) [KC98] es otro agente que trabaja de forma similar, empleando en su arquitectura varios conceptos de Machine Learning como fuzzy logic y redes neuronales.
- Miguel Andrade y su colega Alfonso Valencia presentaron un esquema para extraer información relevante referente a familias de proteínas contenida en abstracts de la base de datos MEDLINE [AV98]. En este esquema cada familia de proteínas constituye un contexto temático de interés para los autores.
- Modha y Spangler [MS00] utilizan una combinación de atributos extraídos del contenido textual de los documentos, los links entrantes y los links salientes, para definir

similitud entre documentos en un algoritmo de clustering denominado *toric k-means* basado en el algoritmo de k-medias.

- SUITOR (Simple User Interest Tracker) [MBCS00] es una arquitectura pensada como un sistema de información *atento*. Es una colección de "agentes atentos" que acumulan información de los usuarios. Estos agentes, monitorean el comportamiento del usuario y su entorno; por ejemplo, pueden determinar que está leyendo el usuario observando su mirada. También analizan las entradas por teclado, los movimientos del mouse, los URLs visitados y las aplicaciones en foco. Esta información se utiliza para recuperar material relevante con respecto al contexto, tanto desde la web como desde las bases de datos. De esta forma el usuario interactúa con la computadora de la forma usual y el sistema infiere sus intereses.
- El sistema Watson [Bud03, BHB01] busca dinámicamente, en la web y en la computadora local, información relacionada con la tarea actual del usuario. Para lograr esto, el sistema puede guiarse por un correo electrónico que el usuario está escribiendo, en cuyo caso es capaz de mostrar automáticamente una lista de emails relevantes, o puede guiarse por un documento que el usuario está redactando, en cuyo caso sabe proveer al usuario con resultados relacionados con la redacción.
- Existen Web crawlers temáticos que llevan a cabo el proceso de recuperación guiándose no sólo siguiendo los links existentes, sino también considerando el contenido encontrado para concentrarse en las páginas relevantes con respecto a cierto tema de interés [CvdBD99, MPS04].
- El sistema EXTENDER [MLR05, Mag04, MLRM04] aplica una técnica incremental para construir descripciones de tópico. Su tarea, es generar descripciones breves de nuevos tópicos relevantes con respecto a un modelo de conocimiento en construcción. Partiendo de un mapa conceptual, este sistema extrae términos que sirven como descriptores (términos que tienen capacidades para describir cierto tópico) y discriminadores (términos que sólo aparecen en determinados tópicos). Dichos términos son utilizados para guiar la búsqueda de términos correspondientes a tópicos nuevos pero relacionados con el contexto inicial. Una vez identificados los nuevos términos, el sistema los sugiere como candidatos a ser incluidos dentro del modelo de conocimiento.
- Jorg Hakerberg *et. al* [HPR$^+$08] propusieron un método basado en búsqueda contextual, para normalización de nomenclaturas de genes y extracción de interacción proteína–proteína utilizando dos tipos de modelos contextuales.

6.1.2 Adaptabilidad de consultas y búsqueda temática

La reformulación o extensión de consultas con el fin de que se adapten a las necesidades del entorno, son técnicas de recuperación de información basadas en el uso de un conjunto de documentos a partir de los cuales se pueden obtener nuevos términos [AF77]. Muchos estudios han demostrado los beneficios de contar con herramientas que brinden ayuda en la formulación, reformulación y refinamiento de consultas [Gre98, Kli01, Chu02, RKC$^+$07]. *Relevance feedback* es un mecanismo de adaptación de consultas utilizado para modificar consultas en base a la importancia de los resultados obtenidos mediante la consulta. Un escenario típico de relevance feedback involucraría 5 pasos básicos: (1) se formula una consulta, (2) el sistema devuelve un conjunto de resultados iniciales, (3) se realiza un análisis de importancia de los resultados obtenidos (relevance feedback), (4) el sistema utiliza esta evaluación para construir una representación mejor de las necesidades de información subyacentes y formular una nueva consulta y (5) el sistema devuelve un conjunto mejorado de resultados. Dependiendo del nivel de automatización del paso (3), se pueden distinguir tres formas de feedback:

- **Supervisado**: requiere realimentación explícita, lo cual usualmente se obtiene de parte de los usuarios, quienes indican la relevancia de cada documento recuperado por el sistema ([Roc71]).
- **No supervisado**: aplica un tipo de evaluación de relevancia ciega, típicamente se asumen como relevantes los k primeros documentos recuperados por el sistema ([BSM95]).
- **Semi-supervisado**: el sistema infiere la importancia de los documentos. Una forma común de realizar esta tarea es monitoreando el comportamiento de los usuarios (p. ej. ver qué documentos resultan seleccionados o cuanto tiempo invierte el usuario en cada documento). Si se sabe que la búsqueda de información se basa en determinado contexto temático, otra forma automática de inferir la importancia de un documento es calculando la similitud de los documentos con respecto al contexto actual del usuario ([JW04]).

Existen sistemas que han sido diseñados para soportar refinamiento de consultas [CD90, VWSG97, AT99, OKI$^+$01, TH04, JZS07] y otros que facilitan la exploración temática agrupando los resultados en grupos temáticos coherentes [CPKT92, HP96, AV97, KLK98, ZE99, CD00, ZHC$^+$04]. La mayoría de estos sistemas proveen una interfaz de navegación que requiere intervención explícita del usuario.

La sobrecarga debida a la necesidad de formular las consultas de forma explícita, se puede aliviar si las mismas se formulan de manera automática. Existen situaciones en las que se puede contar con información útil para guiar la reformulación. Los algoritmos desarrollados en este capítulo, son capaces de aprovechar tal información para producir consultas que al ser presentadas a motores de búsqueda o índices obtengan resultados relevantes para el tópico de interés. Teniendo en cuenta que un contexto temático puede contener una enorme cantidad de términos y que los motores de búsqueda convencionales suelen limitar la longitud de consulta, es sumamente importante realizar una selección útil de los términos.
Si bien se sabe que no siempre toda la información contenida en el contexto puede ser resumida en una consulta, se pueden implementar mecanismos capaces de extraer conjuntos de términos que sean representativos y construir consultas a partir de ellos. Los resultados obtenidos luego de presentar dichas consultas, pueden compararse con el contexto original para filtrar los que son irrelevantes. Por otra parte, los resultados relevantes pueden ser aprovechados para agregar conocimiento al mecanismo de generación de consultas. En este contexto, la propuesta de esta tesis es usar algoritmos evolutivos para implementar mecanismos capaces de aprender cuales son los mejores términos. Partiendo del contexto de interés y aumentando el conocimiento a medida que se obtiene nuevo material relevante.

6.1.3 Escenarios de alcance

El desarrollo de métodos que permitan obtener consultas de alta calidad para recolectar recursos relevantes con respecto a determinado tópico puede tener un impacto significativo en el área de IR. Estos son algunos de los escenarios en los que dichos métodos pueden ser útiles:

Búsqueda basada en la tarea del usuario

Los sistemas basados en la tarea del usuario son un caso particular de sistemas de búsqueda basados en tópico, en los cuales se aprovecha la interacción del usuario con aplicaciones informáticas, para determinar su tarea actual y formar un contexto temático que refleje las necesidades del usuario [BHB01, LBMW00]. Si nos conformamos armando las consultas simplemente a partir del contexto inicial, es altamente probable que disminuyan las posibilidades de recuperar material relevante. Sin embargo, reformulando

las consultas a través de nuevos términos formando consultas de alta calidad, un sistema de búsqueda basado en la tarea del usuario puede generar sugerencias automáticas sumamente relacionadas con los intereses del usuario.

Sistemas de recuperación para portales web

Un portal web tiene el propósito de recopilar recursos correspondientes a determinados tópicos. Los documentos recolectados se usan para suministrar búsqueda especializada y directorios de sitios. Además de poseer las características de un motor de búsqueda estándar, un portal ofrece otros servicios tales como correo electrónico, noticias, información y entretenimiento. Algunos de los portales web más conocidos son MSN, Yahoo! y AOL. Típicamente, los buscadores focalizados (focused crawlers) son los encargados de explorar la web para recopilar contenido relevante a cierto tópico y poblar los índices de los portales web [CvdBD99, Cha02, MPS04]. Como alternativa a los focused crawlers, el proceso de recuperación de recursos puede ser soportado formulando consultas enfocadas en el tópico de interés y presentando dichas consultas a un motor de búsqueda. Una vez recuperados los resultados que genera la consulta, se pueden seleccionar aquellos que estén relacionados con el tópico de interés.

Acceso a la web oculta

Una importante cantidad de información de la web se encuentra en páginas generadas en forma dinámica y constituye lo que se conoce como Web Invisible [KSS97]. Estos tipos de páginas, no existen hasta que se crean como resultado de una consulta específica presentada a formularios de búsqueda disponibles en determinados sitios (por ejemplo: las páginas que exigen loggin por medio de nombre y password, páginas comerciales como amazon.com y mercadolibre.com.ar o bases de datos temáticas como librerías científicas y librerías multimedia). Se estima que el tamaño de la Web Invisible alcanza miles de TB de información, superando en varios órdenes de magnitud a la web superficial [Ber01]. Los motores de búsqueda estándar no son capaces de alcanzar este tipo de información. En el año 2001, Sriram Raghavan y Hector Garcia Molina presentaron un modelo para explorar la Web Invisible [RGM01]. Este modelo, utiliza términos provistos por usuarios u obtenidos de una base de datos para realizar consultas a los formularios web, obteniendo de este modo información que se encuentra en bases de datos ocultas. Alexandros Ntoulas *et al.*, propusieron una arquitectura destinada a la recuperación de información de

la Web Invisible por medio de políticas de generación automática de consultas [NZC05]. Los algoritmos propuestos en este capítulo, son una alternativa para el problema de generación automática de consultas efectivas y pueden ayudar a automatizar el proceso de recuperación de recursos provenientes de la Web Invisible.

Recopilación de información persistente

Muchos usuarios están interesados en obtener determinada información en forma constante, lo que se conoce como necesidades de información persistente [SH04]. Por ejemplo, muchos ejecutivos necesitan estar al tanto de las noticias referidas a las industrias y condiciones de stock de determinados mercados para no perder oportunidades de inversión, los coleccionistas quieren saber lo último sobre sus piezas de colección, los científicos quieren estar actualizados y poder tener acceso a las últimas publicaciones en su área de investigación. Este tipo de información, con respecto a un tópico determinado, puede ser recopilada por medio de herramientas que observen la web, busquen en forma automática documentos referidos al tópico y sean capaces de presentarlos al usuario en la forma en que éste lo requiera. WebTOPIC es una herramienta semi-automática que interactúa con el usuario en un proceso de dos fases [EJ05]. En la primera etapa, el usuario define el tópico de interés y especifica determinadas características. En la segunda etapa, el usuario analiza los resultados que han sido recuperados por el sistema. De forma similar, una vez que se ha definido el perfil del usuario identificando sus intereses, dicho perfil puede ser usado para generar consultas referidas al tópico de forma automática. Con este esquema, un sistema inteligente podría recopilar material relevante disponible en forma periódica, con el objetivo de brindar un servicio para las necesidades de información persistente del usuario.

Soporte en gestión del conocimiento

El manejo efectivo del conocimiento puede requerir ir más allá de la captura de conocimiento inicial, para permitir extender el modelo de conocimiento original [LMR$^+$03, MLR05]. La web provee una fuente importante de información potencial para agregar a un modelo de conocimiento. Este material, puede ser recuperado presentando consultas contextuales a los motores de búsqueda convencionales. En este sentido, el contexto o tema de búsqueda se crea en base al modelo de conocimiento en construcción. Si bien es cierto, que una consulta probablemente no puede contener toda la información

contenida en el modelo, se pueden usar estrategias inteligentes que extraigan términos representativos y construyan una consulta efectiva. Usando la web como un gran repositorio de memoria colectiva y comenzando desde un modelo de conocimiento en progreso, las técnicas discutidas en este capítulo pueden facilitar el proceso de captura de conocimiento para lograr una extensión del modelo. Los resultados recuperados por medio de las consultas formuladas pueden filtrarse, utilizando como punto de comparación el modelo de conocimiento inicial para descartar el material irrelavante.

6.2 Uso de algoritmos evolutivos en recuperación de información

Los primeros intentos de utilizar EAs en recuperación de información datan de fines de los 80's. El objetivo principal de estos enfoques fue el uso de algoritmos evolutivos (en particular algoritmos genéticos) para la ***evolución de descripciones de conjuntos de documentos***, con el fin colaborar en tareas de indexado o clustering de documentos. Michael Gordon utiliza EAs para construir un método de indexado [Gor88]. En este esquema, cada documento se asocia con una lista de términos que lo representa. El usuario debe proveer una lista inicial de palabras, luego, se usa el EA para encontrar mejores descripciones de documentos, algo similar a lo llevado a cabo por Vijay Raghavan y Brijesh Agarwal en 1987 [RA87]. En un trabajo posterior Gordon completó el ciclo de clustering de documentos por medio de EAs [Gor91], demostrando que la evolución de descripciones de documentos permite aumentar la exactitud en el clustering de documentos relacionados. Otro trabajo que utiliza EAs, relacionado con clustering de documentos, propone un método basado en el análisis del contexto de los documentos en vez de en el contenido [LL01]. Para determinar el contexto, se examinan las estructuras de hipertexto y se siguen los URLs presentes en el documento. En esta propuesta, los EAs se utilizan para analizar dicho contexto y crear un contexto para clustering difuso (fuzzy clustering). A diferencia del clustering estricto (hard clustering), el cual asocia cada documento con un grupo o cluster, el fuzzy clustering permite que un documento pertenezca a más de un grupo.
En ***optimización de consultas***, se han utilizado EAs para aprendizaje por refuerzo del peso asociado a cada término [FS91, YK93, PBPK93]. En este sentido, cabe aclarar que a diferencia de nuestra propuesta, estos métodos evolucionan (seleccionan, recombinan y mutan) los pesos asociados a los términos y no los términos en sí mismos. Nick y Themis propusieron un sistema de agentes inteligentes que trabaja en forma similar [ZP01], el Webnaut. Este sistema utiliza EAs para recolectar y recomendar páginas web en uno de

sus agentes. El agente emplea dos EAs, los cuales crean respectivamente dos poblaciones. El primer EA, utiliza individuos compuestos por un conjunto de palabras. Estas palabras son extraídas de un diccionario creado en base a documentos sugeridos inicialmente por el usuario. El segundo EA, emplea individuos compuestos por un conjunto de operadores lógicos, tales como AND, OR o NOT. Ambas poblaciones se combinan para formar consultas. Los valores de fitness para los individuos correspondientes a ambas poblaciones se calculan en base a la similitud entre los resultados recuperados y el diccionario. Además, el sistema maneja cierto grado de realimentación por parte del usuario. Esto le permite incorporar nuevos documentos no tenidos en cuenta inicialmente y dar menos importancia a documentos en los que el usuario está menos interesado, para lo cual se modifican los pesos (frecuencias) de cada palabra dentro del diccionario.

Otros esquemas han sido más orientados hacia la ***recuperación temática***. En [CFP04] Caramia *et al.* proponen un método para mejorar los resultados al realizar búsquedas temáticas, en el cual el material presentado al usuario final se organiza en conjuntos de documentos relevantes de baja cardinalidad. Dichas páginas representan de la mejor manera posible toda la información presente en un conjunto mayor de páginas. El método utiliza un contexto/perfil del usuario para obtener páginas relevantes. Para cada una de estas páginas, se construye una vectorización vinculada de alguna manera con el contexto/perfil. Las páginas vectorizadas se analizan por medio de un algoritmo de clustering que las agrupa en subconjuntos de páginas similares. Cada subconjunto tiene la propiedad de reunir páginas altamente relevantes pero distintas entre sí. El rol del EA dentro del método es evolucionar una población de subconjuntos de páginas. Cada individuo es un pequeño subconjunto de páginas extraído de los clusters, comenzando con las páginas de mayor puntaje. La función de fitness combina tres métricas. La primera, tiene en cuenta la importancia de cada página dentro de su cluster. La segunda, procura disminuir la cantidad de páginas dentro del individuo, de manera que un individuos con pocas buenas páginas no supere a un individuo con muchas malas páginas. Finalmente la tercera, considera una medida de similitud de tal forma que las páginas dentro del subconjunto de páginas de cada individuo sean lo más distintas posible. Una vez que al EA termina, el mejor individuo es el subconjunto de páginas que se presenta al usuario. En otro trabajo relacionado con la búsqueda temática, Leroy, Lally y Chen propusieron un método para mejorar el conjunto inicial de resultados recuperados, utilizando las consultas del usuario modificadas en base al interés que demuestra el usuario sobre los documentos que se van recuperando [LLC03]. Para analizar el interés del usuario el sistema moni-

torea los vínculos seguidos por el usuario (considerados como *relevantes*) y los vínculos ignorados por el usuario (considerados como *no relevantes*). En este sistema el EA se encarga de evolucionar individuos compuestos de conjuntos de términos. Para armar dichos conjuntos se tienen en cuenta los términos propuestos por el usuario, los términos obtenidos de los ítems que el usuario consideró *relevantes* y los términos extraídos del material que el usuario consideró *no relevante*. Por otro lado, para evaluar a cada individuo el algoritmo utiliza los primeros diez resultados recuperados para cada individuo y construye tres grupos de términos: el de palabras *relevantes* asociadas a palabras existentes en los vínculos seguidos por el usuario que no son *stop word* (S_{Rx}), el de palabras *no relevantes* asociadas a los vínculos ignorados por el usuario (S_{Nx}), y un conjunto adicional que contiene *stop words* (S_S). Basándose en estos tres conjuntos de palabras, el algoritmo construye tres contextos que se actualizan en cada búsqueda realizada por el usuario y se usan para evaluar y modificar la siguiente búsqueda. El primer contexto, C_{Rx}, contiene todas las palabras del conjunto *relevante* que no están en el conjunto *no relevante* ni en el de *stop words*. El segundo contexto, C_{Nx}, es el de términos *no relevantes* (construido de forma análoga a C_{Rx} pero con el conjunto *no relevante*). Finalmente el tercer contexto, C_{Ax}, es el conjunto de términos que aparecen en el conjunto *relevante* y en el *no relevante* pero no aparecen en el de *stop words*. En base a los dos primeros se calcula la función de fitness, la cual emplea conceptos de similitud para evaluar los resultados obtenidos por cada individuo al ser presentado al motor de búsqueda, favoreciendo a aquellos que obtienen resultados más parecidos al *contexto relevante* y menos parecidos al *contexto no relevante*. Otra área de investigación relacionada es la evolución de agentes que navegan la web realizando búsqueda temática, es decir, la *evolución de focus crawlers* [MB00, MPS04, MBVL99, HYMRY98]. Menczer y Belew [MB00] mostraron que en porciones bien organizadas de la web, los agentes pueden evolucionar y aprender estrategias efectivas por medio del uso de redes neuronales y algoritmos evolutivos. En [MPS04] los autores analizan la adaptación de focus crawlers, usando un algoritmo (tipo multi-agente) en el cual los individuos pueden aprender a estimar links mediante aprendizaje por refuerzo, logrando que la población evolucione favoreciendo el proceso de exploración hacia áreas de la web que parezcan prometedoras. El objetivo final de un focus crawler es similar al de los métodos propuestos en este capítulo, es decir, recuperar recursos relevantes con respecto a determinado tópico. Sin embargo, las técnicas usadas por crawlers temáticos son diferentes a las presentadas aquí. En las alternativas presentadas en las próximas secciones, se asume que existe un índice subyacente que puede accederse a través de una

interfaz de búsqueda. Por su parte, los crawlers temáticos construyen sus propios índices recorriendo las páginas que van encontrando en el grafo de la Web.

6.3 Generación de consultas temáticas como problema de optimización

La generación de consultas temáticas de alta calidad se puede pensar de forma natural como un problema de optimización. Dependiendo de los objetivos a optimizar será posible formalizar dicho problema bajo la definición 1 (*Problema de Optimización Mono-Objetivo*), presentada en la sección 2.2 o bajo la definición 3 (*Problema de Optimización Multi-Objetivo*), presentada en la sección 2.3. En ambos casos, el espacio de búsqueda, también conocido como espacio de decisión o espacio genotípico en el contexto de EAs, puede definirse como el conjunto de posibles consultas que pueden presentarse a un motor de búsqueda. Cada consulta formulada se presenta a cierto motor de búsqueda para recuperar información. La función objetivo a optimizar debe tener en cuenta la efectividad de la consulta para recuperar material relevante. Dependiendo de los objetivos del sistema, la efectividad de una consulta puede definirse usando nociones tradicionales de IR tales como *precisión*, *cobertura*, *similitud*, u otras métricas de evaluación de desempeño personalizadas.

6.3.1 Precisión y cobertura

La recuperación efectiva de información depende, tanto de la capacidad para recuperar material relevante con respecto a las necesidades del usuario, como de filtrar la información que le sea irrelevante. Para evaluar la habilidad que posee un sistema de recuperar ítems relevantes y al mismo tiempo filtrar los ítems irrelevantes, la comunidad de recuperación de información suele utilizar las medidas de *precisión* (precision) y *cobertura* (recall). Dada una solicitud de información y su correspondiente conjunto de documentos relevantes, R, si la estrategia de recuperación genera un conjunto de resultados A, la *precisión* y *recall* se definen de la siguiente manera:

Definición 14 (Precisión [BYRN99])**.** *Es la fracción de documentos recuperados (la fracción de T) que es efectivamente relevante, es decir,*

$$Precisión = \frac{|R \cap A|}{|A|} \tag{6.1}$$

Definición 15 (Recall [BYRN99]). *Es la fracción de documentos relevantes (la fracción de R) que ha sido recuperada, es decir,*

$$Recall = \frac{|R \cap A|}{|R|} \tag{6.2}$$

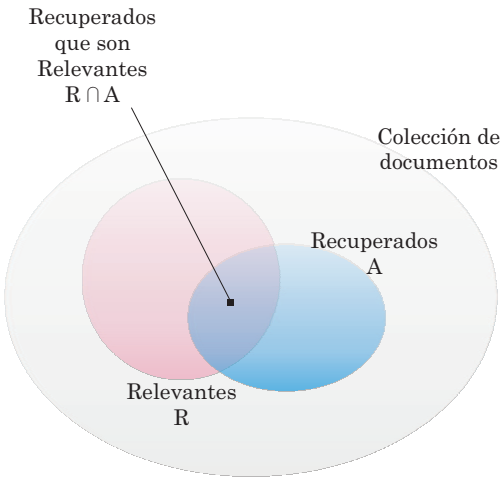

Figura 6.1: Precisión y Cobertura.

La figura 6.1 muestra el conjunto total de documentos disponibles (colección de documentos) en color gris, el conjunto de documentos recuperados, A, en color azul, el conjunto de documentos relevantes, R, en color rosa y la intersección de los mismos $R \cap A$. Podemos apreciar que la cobertura es la relación entre documentos recuperados relevantes y documentos relevantes (intersección vs. conjunto rosa), mientras que la precisión es la relación entre documentos recuperados relevantes y documentos recuperados (intersección vs. conjunto azul).

6.3.2 Modelo de ponderación TF-IDF

El modelo de ponderación TF-IDF (por sus siglas en inglés de term frequency-inverse document frequency) es una medida estadística simple usada para medir la importancia de una palabra para un documento dentro de una colección o corpus. Esta importancia aumenta en forma proporcional a la frecuencia de apariciones de la palabra en el documento y disminuye con la frecuencia de apariciones de la palabra en el corpus. De acuerdo con este modelo la relevancia está determinada por dos componentes:

- **Frecuencia del término (TF).** Dado un documento d y un término t_i la frecuencia del término es, $n(d, t_i)$, el número de veces que el término t_i ocurre en el documento d. En general esta cantidad se normaliza para evitar la tendencia a beneficiar a los documentos de mayor longitud. Por ejemplo, se lo puede dividir por la cantidad de términos en el documento en cuyo caso resulta:

$$TF(d, t_i) = \frac{n(d, t_i)}{\sum_k n(d, t_k)} \tag{6.3}$$

- **Frecuencia Inversa (IDF).** Si pensamos que cada palabra es una dimensión de un espacio vectorial, no todos los ejes de este espacio tendrán el mismo grado de importancia. Dado que existen palabras sumamente comunes que no aportan tanta información ya que pertenecen simultáneamente a muchos documentos aunque no haya una relación temática entre ellos, IDF pretende darle menos importancia a este tipo de términos. Sea t_i un término y D un corpus de documentos. Sea D_t el conjunto de los documentos que contienen al término t_i, una forma común de definir IDF [SY73] es:

$$IDF(t) = \log \frac{1 + |D|}{|D_t|} \tag{6.4}$$

El factor TF ayuda a lograr una buena cobertura. Sin embargo, la frecuencia de los términos por sí sola no es capaz de asegurar una precisión aceptable ya que puede haber términos con alta frecuencia en documentos irrelevantes. De esta forma el término IDF cumple la función de penalizar a aquellos términos que tienen poco poder discriminador. Por esta razón, TF e IDF se combinan para formar la métrica TF-IDF de la siguiente manera:

$$TF\text{-}IDF(d, t) = TF(d, t) * IDF(t) \tag{6.5}$$

En principio, se busca que un sistema logre tanto buena cobertura como buena precisión. Para lograr esto, los esquemas convencionales de IR usan factores combinados de ponderación de términos que contemplan precisión y recall simultáneamente. Sin embargo, se presentan diversas cuestiones cuando se intenta aplicar esquemas convencionales de IR para medir la importancia de los términos en presencia de corpus no convencionales [KT00, Bel00]. En particular, las medidas de *precision* y *recall* serán útiles siempre y cuando tengamos acceso a $|R|$, el número de documentos relevantes. Sin embargo, existen corpus de documentos en los cuales esta medida es imposible de determinar, el ejemplo

más común de ellos es la web. Para sobrellevar este inconveniente, distintos autores han definido otras aproximaciones para estas métricas (p. ej. Saracevic, 1995, Chu and Rosenthal, 1996, Wishard, 1998, Srinivasan *et al.*, 2005.).
En el marco de esta tesis, este capítulo presenta distintas formas para abordar el problema de ponderación de términos en presencia de dos componentes principales. Primero, buscamos la forma de identificar buenos términos que nos permitan guiar la búsqueda temática en la web. Segundo, el hecho de que cuando se formulan consultas de forma automática para búsqueda web no tenemos acceso a la colección completa de documentos.

6.3.3 Similitud

Una forma popular de medir la efectividad de un sistema al recuperar documentos, es empleando medidas de similitud. Esta estrategia es especialmente útil si existe un parámetro con el cual podemos comparar los documentos recuperados por el sistema para una determinada consulta, como suele suceder en el caso de búsquedas temáticas. Una métrica sumamente conocida que se utiliza para medir similitud es la *similitud por coseno*, la cual expresa la cercanía entre dos documentos.
Un documento d puede tener un vector de términos asociado de la forma:

$$\vec{d} = (t_1, t_2, \ldots, t_p) \tag{6.6}$$

donde t_i expresa de alguna forma la importancia que el término i posee para el documento d. Por ejemplo, t_i podría ser la cantidad de apariciones del término i en el documento d (para $i = 1, \ldots, p$), es decir, $t_i = n(i, d)$, donde $n(i, d)$ es simplemente el número de veces que i aparece en d. Una alternativa es que t_i sea la *frecuencia de aparición* del término i en el documento d, es decir, $t_i = n(d, i)/\sum_k n(d, k)$ e incluso puede usarse la medida TF-IDF. Cualquiera de estas representaciones nos permite ubicar al documento dentro del espacio vectorial. Una vez obtenida la representación para un documento, dicho documento puede compararse con la representación vectorial de otros documentos utilizando el coseno del ángulo que los separa como se ve en la figura 6.2, utilizando la definición 16.

Definición 16 (Similitud por coseno)**.** *Sean d_i y d_j dos vectores en un espacio n-*

dimensional. La similitud por coseno entre d_i y d_j se define como:

$$\sigma(d_i, d_j) = \frac{\overrightarrow{d}_i \cdot \overrightarrow{d}_j}{\| \overrightarrow{d}_i \| \cdot \| \overrightarrow{d}_j \|} \tag{6.7}$$

y es el coseno del ángulo θ entre d_i y d_j.

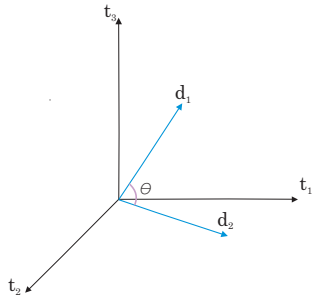

Figura 6.2: Similitud por coseno.

6.4 ¿Por qué usar algoritmos evolutivos para generación de consultas de alta calidad?

Existen varias razones por las cuales los Algoritmos Evolutivos resultan apropiados para afrontar o intentar resolver la búsqueda temática:

- **Espacio altamente multi-dimensional.** El espacio de consultas es un espacio altamente multi-dimensional, donde cada posible término concierne a una nueva dimensión. Además, la cantidad de posibles consultas es enorme, dado que existen muchísimas posibles combinaciones de palabras (cualquiera sea el lenguaje). Este tipo de problemas no puede ser resuelto de forma efectiva usando métodos analíticos convencionales y constituyen un ámbito apropiado para EAs.
- **Grandes cantidades de información.** Dado que el propósito de cada consulta es recuperar determinado conjunto de documentos o material relevante, el conjunto total de documentos también es un espacio de búsqueda en este problema. Uno de los aspectos que caracteriza a los repositorios de información más importantes es su gran dimensión. Por ejemplo, las bases de datos médicas pueden contener enormes cantidades de ítems de información. La web contiene millones de documentos. Los EAs son algoritmos de búsqueda robustos especialmente diseñados para este tipo de problemas.

- **Soluciones subóptimas.** La recuperación exitosa de información requiere que se formulen consultas de alta calidad y que se tengan en cuenta los resultados aunque no sean los óptimos. Se sabe que si bien los algoritmos evolutivos no garantizan la identificación de soluciones óptimas, si son bien implementados, son capaces de encontrar *soluciones cercanas a las óptimas*.
- **Múltiples soluciones.** Cada uno de los múltiples conjuntos de documentos recuperados puede representar un buen resultado para una búsqueda temática. Por lo tanto, podemos estar interesados en varias de las consultas de alta calidad generadas y no sólo en una de ellas. Los EAs constituyen una alternativa natural para los problemas de optimización multimodal, siendo capaces de entregar múltiples soluciones globales.
- **Exploración y Explotación.** Encontrar buenas combinaciones de términos para formular consultas, requiere la exploración del espacio temático en diferentes direcciones. Esta exploración debe ser independiente del conjunto inicial de consultas, e incluso puede requerir ir más allá del conjunto de términos inicial incorporando términos nuevos. Este aspecto puede ser efectivamente abordado aplicando los operadores genéticos apropiados en la medida adecuada. Por otra parte, los mecanismos de selección inducirán de forma natural a la explotación de las combinaciones de términos más alentadoras.
- **Entorno susceptible a cambios.** Cuando se está trabajando en búsqueda temática, el entorno puede sufrir modificaciones en el tiempo. Se puede apreciar un ejemplo común de este tipo de comportamiento en los contextos de los usuarios, ya que es de esperar que un usuario no se encuentre continuamente interesado en el mismo tópico. Los algoritmos evolutivos son ideales para este tipo de problemas, dado que son sumamente flexibles a cambios en el entorno, pudiendo adaptarse por medio del aprendizaje. Al evaluar las soluciones en el nuevo entorno, las que no se desempeñen de forma efectiva tendrán menos probabilidades de sobrevivir, aún si en generaciones pasadas fueron soluciones cercanas a las óptimas para entornos anteriores.
- **Búsqueda temática como problema de optimización.** Dado que la búsqueda temática puede pensarse naturalmente como un problema de optimización, tanto mono-objetivo, como multi-objetivo, y dada la complejidad que representa dicho problema, los esquemas mono/multi-objetivo conocidos para EAs pueden ser desarrollados para resolverlo con la esperanza (para nada utópica) de lograr resultados

tan satisfactorios como en todos los demás problemas de optimización en los que se han implementado soluciones bajo este paradigma.

6.5 Infraestructura evolutiva propuesta para generación de consultas temáticas

Considerando que no es simple generar buenas consultas basándonos solamente en un contexto temático, pero que contamos con una forma de juzgar la efectividad de una consulta a partir de los resultados que nos permite recuperar, podemos ver que un esquema evolutivo que se sustente con métricas para evaluar el desempeño de las consultas parece prometedor.

6.5.1 Cuestiones de investigación generales

Un tópico puede caracterizarse a partir de cierto conjunto de palabras. Desde un punto de vista pragmático, la selección de esas palabras debería estar orientada a cumplir con el objetivo de interés. Aún para un mismo tópico, los términos tendrán diferente grado de importancia dependiendo de si se los necesita para generar consultas, para construir la descripción de un tópico o para construir un índice, entre otros posibles. Por ejemplo, un término que sea buen descriptor puede tener pocas capacidades como discriminador. Este *buen descriptor* por sí sólo no resultaría útil para formar una consulta de alta calidad, debido a la poca precisión que consigue en los resultados. Sin embargo, si se lo combina con otros términos que sean buenos discriminadores, dichos términos pueden distinguir entre buenos y malos resultados, mejorando la calidad de la consulta.
El objetivo de la arquitectura propuesta es refinar automáticamente consultas, mejorando su calidad, de tal forma que tengan la capacidad de recuperar material relevante con respecto a determinado tema de interés. La importancia de esta tarea, radica en que se pueden facilitar muchos servicios en IR a partir de la generación de consultas de alta calidad. El material recuperado por estas consultas automáticamente mejoradas, puede ayudar en importantes tareas dentro de IR, por ejemplo, acercando información de interés a los usuarios (sin exigir su participación), brindando nuevo material a sistemas de gestión del conocimiento, recolectando conjuntos de documentos relacionados para portales temáticos o recuperando información de la Web Invisible.

En la siguiente sección se explican los componentes principales de la arquitectura propuesta.

6.5.2 Arquitectura propuesta

La figura 6.3 muestra los componentes principales de la infraestructura evolutiva y la conexión entre ellos, e incluye algunos de los diferentes servicios de información que pueden utilizar este sistema para su beneficio. En el prototipo propuesto, el sistema cuenta con:

- La representación interna del tópico de interés.
- El mecanismo de generación de consultas iniciales.
- Una población de consultas, la cual se va refinando de forma incremental a medida que el sistema evoluciona.
- Un reservorio de vocabulario nuevo.
- Los operadores genéticos, el proceso de selección y el módulo para evaluación del fitness.
- La herramienta para extracción de términos.
- El motor de búsqueda como medio de comunicación entre el sistema y la colección de documentos.

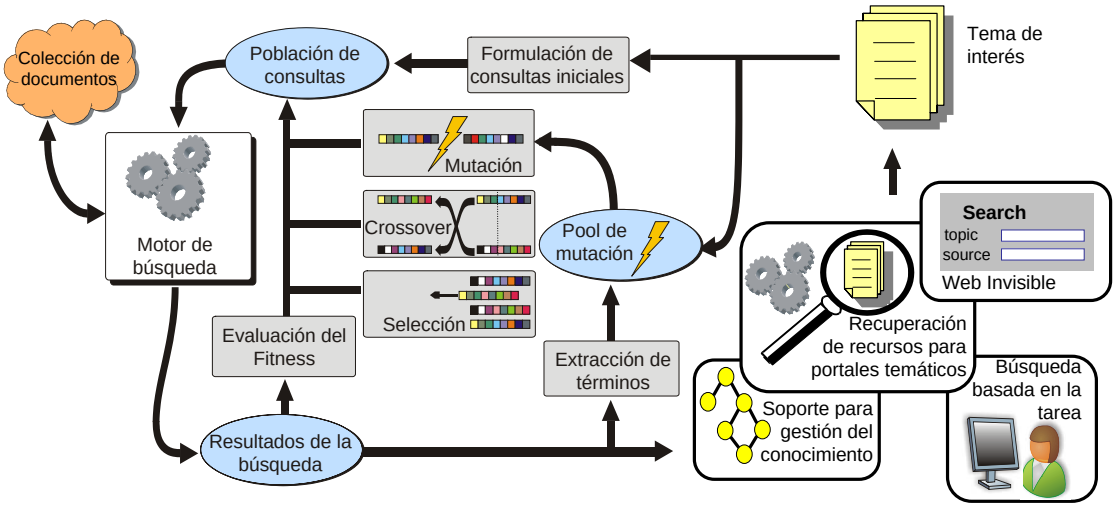

Figura 6.3: Arquitectura evolutiva propuesta para generación automática de consultas temáticas. En la esquina inferior izquierda se incluyen los dominios de aplicación.

A continuación se explican los mecanismos generales fundamentales que permiten que el sistema evolucione consultas y recupere material relevante con respecto al tema de interés.

Conformación de la población y representación de cromosomas

El espacio de búsqueda Q está constituido por todas las posibles consultas que pueden ser formuladas a una interfaz de búsqueda. Por lo tanto la población de cromosomas será un subconjunto de tales consultas. Esta población nos permitirá recorrer el corpus de documentos (ya sea en la web o en un índice local) con una tendencia orientada hacia material cada vez más relevante para el tópico de interés.
En base a lo anterior, cada cromosoma se representa como una lista de términos, donde cada término corresponde a un gen a ser manipulado por los operadores genéticos. Teniendo en cuenta estas descripciones utilizaremos las siguientes definiciones:

Definición 17 (Cromosoma Consulta)**.** *Sea T un conjunto de términos conformado por uno o más vocabularios. Un cromosoma consulta $\boldsymbol{q}$ está definido como una sucesión de términos $\{t_1, t_2, \ldots, t_p\}$, tal que $t_i \in T$, $\forall\ i = 1, \ldots, p$.*

Definición 18 (Población de Consultas)**.** *Una población de consultas P es un conjunto de cromosomas consulta $\{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n\}$.*

Podemos ver que la representación utilizada es una versión modificada de la representación de strings de bits utilizada para AGs clásicos. A diferencia de la representación clásica, cada gen es una palabra y la longitud de los individuos no necesariamente debe ser igual.

Formulación de consultas iniciales

Para generar los cromosomas consulta de la primera población inicial, $\mathbf{P}_0$, el mecanismo de formulación selecciona términos de una descripción disponible para el tópico de interés. Dicha descripción se obtiene a partir de la representación interna que el sistema posee para dicho tópico. Esta representación puede estar conformada por ejemplo, en base a las tareas que está realizando un usuario, en base a una descripción realizada manualmente para tal tópico (p. ej., extraída de una base de datos como DMOZ) o a partir de un modelo de conocimiento determinado. En cualquier caso, la selección es aleatoria. El número de términos en cada cromosoma consulta $\mathbf{q} \in \mathbf{P}_0$ es un número aleatorio entre 1 y una longitud inicial determinada. Sin embargo, esta longitud puede ser superada en el proceso de evolución por medio del operador de cruzamiento. Si bien los motores de búsqueda suelen imponer un límite al tamaño de consulta (ignorando los términos que están más allá de dicho límite), es interesante notar que el incremento más

allá de los límites en la longitud de la consulta captura, en cierto sentido, el fenómeno de herencia recesiva. En este fenómeno, algunos genes que fueron ignorados en una generación (porque estaban más allá del límite de la consulta), pueden volver a ser tenidos en cuenta en generaciones posteriores. Esto eventualmente sucede cuando, gracias al proceso de recombinación, vuelven a formar parte de un descendiente dentro de los límites de la consulta.

Recombinación

Durante este proceso algunas de las consultas de la nueva población (formada con el operador de selección) se trasladan sin modificación a la próxima generación, mientras que otras, se recombinan para generar nuevas consultas. En cada recombinación participan dos consultas padres y se generan dos nuevos individuos. Durante dicho cruzamiento, se realiza un proceso básico que consiste en copiar palabras de cada padre en los descendientes. El operador de recombinación utilizado en esta propuesta es el *operador de recombinación de un punto*, el cual resulta en nuevas consultas en las cuales un padre contribuye con los n primeros términos y el otro padre con los términos restantes. El punto de cruce, n, se selecciona en forma aleatoria.

Mutación

Al aplicar este operador se producen pequeños cambios en los individuos de la nueva población. Dichos cambios consisten en el reemplazo de un término de la consulta seleccionado al azar, t_q, por otro término, t_p, obtenido del reservorio de mutación.

Evaluación del fitness

El componente de evaluación de fitness implementa los criterios definidos para evaluar la calidad de cada consulta y, consecuentemente, para clasificarlas en algún orden de importancia. Nuestra concepción de *consulta de alta calidad* se basa en la habilidad de la consulta para recuperar material relevante con respecto al tema de interés cuando se la presenta a una interfaz de búsqueda. Existen diferentes formas de cuantificar las necesidades de información [Coo68]. Por ejemplo, un usuario puede estar interesado en obtener un único documento relevante, un número n de documentos relevantes, todos los documentos relevantes o una porción de los documentos relevantes. Es normal que el conjunto de documentos recuperados tenga una cardinalidad grande, por lo que por

propósitos prácticos, en algunos de los esquemas presentados en esta tesis evaluamos los diez primeros resultados obtenidos por cada consulta, adecuando las métricas a tal efecto.

Extracción de términos

La herramienta de extracción de términos utiliza los resultados recuperados a partir de cada consulta para extraer nuevos términos. Solo las palabras que no son *stopword* son consideradas, es decir, se descartan palabras como: a, an, in, the, etc. Los nuevos términos se agregan al reservorio de términos que serán usados por el operador de mutación.

Reservorio de mutación

Este reservorio es un paquete de términos auxiliares considerados por el EA al aplicar el operador de mutación. Este conjunto de palabras se forma inicialmente a partir de la descripción disponible para el tópico de interés. Luego, a medida que la población evoluciona, el reservorio se va enriqueciendo con nuevos términos extraídos de los documentos recuperados. La motivación para mantener este conjunto auxiliar radica en que la incorporación de nuevas palabras favorece la exploración del espacio de búsqueda. Simultáneamente, esperamos que a través de la búsqueda surjan términos que sean cada vez mejores para recuperar material relevante.

6.6 Primera implementación: versión mono-objetivo

El sistema permite optar entre dos posibles esquemas, el primero de ellos permite evaluar un objetivo en tanto que el segundo permite evaluar múltiples objetivos. Como primer paso para la implementación de la infraestructura se consideró la evaluación de un único objetivo.

6.6.1 Cuestiones de investigación

- Cuestión 1: ¿Cómo podemos evolucionar consultas cuando intentamos recuperar material similar al tópico de interés?
- Cuestión 2: ¿Cómo se pueden evaluar dichas consultas para poder ordenarlas de acuerdo a un orden de importancia?
- Cuestión 3: ¿Podemos obtener resultados satisfactorios por medio de EAs en este problema particular? ¿Los resultados obtenidos luego de la evolución son sustancialmente mejores que los iniciales, como para justificar su uso?

- Cuestión 4: ¿Cómo afectan diferentes tasas de mutación al comportamiento del sistema? ¿Es el comportamiento esperado?

6.6.2 Selección

Por medio de este operador se genera una nueva población seleccionando de forma probabilística las consultas de mayor calidad presentes en la población actual. La probabilidad de que una consulta resulte seleccionada será proporcional a su nivel de aptitud e inversamente proporcional al nivel de aptitud de las demás consultas de la población. Este método se conoce como *selección de la ruleta.*

6.6.3 Función de aptitud basada en similitud por coseno

Para determinar la efectividad de cada consulta, asociamos una función de fitness, F, al espacio de consultas Q, tal que $F : Q \rightarrow [0, 1]$. Dicha función nos permite evaluar en forma numérica cada individuo (consulta). En esta implementación, nuestra concepción de *individuo de alta calidad* se basa en la capacidad de dicho individuo, $\mathbf{q}$, para recuperar material similar al tópico de interés, c, cuando se lo presenta a un motor de búsqueda. Al realizar esta acción, es probable que el motor de búsqueda nos entregue varios documentos y no sólo uno. Este conjunto puede presentar diferentes características. Por ejemplo, podríamos obtener los siguientes tipos de resultados:

- *Conjunto con similitud promedio alta y similitud individual alta.* La mayoría de los documentos presentan una similitud elevada con respecto al tópico de interés (conjunto A en la figura 6.4).
- *Conjunto con similitud promedio media y similitud individual diversa.* Algunos de los documentos pueden ser *muy similares* al tópico, mientras que los demás presentan una similitud muy baja (conjunto B en la figura 6.4).
- *Conjunto con similitud promedio media y similitud individual media.* La mayoría de los documentos son *algo similares* al tópico sin que ninguno se destaque en forma particular (conjunto C en la figura 6.4).
- *Conjunto con similitud promedio baja y similitud individual diversa.* La mayoría de estos documentos no son similares al tópico de interés, mientras que unos pocos presentan una similitud muy alta (conjunto D en la figura 6.4).
- *Conjunto con similitud promedio baja y similitud individual baja.* Todos los documentos son *poco similares* al tópico de interés (conjunto E en la figura 6.4).

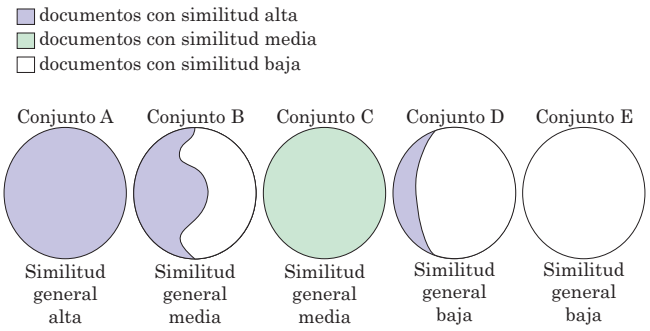

Figura 6.4: Ejemplos de los posibles resultados que podemos obtener al presentar una consulta a un motor de búsqueda.

Supongamos que estamos en presencia de dos consultas, $\mathbf{q_1}$ y $\mathbf{q_2}$, tal que $\mathbf{q_1}$ recupera muchos documentos parecidos al tópico de interés sin que ninguno de ellos se destaque por una similitud elevada, mientras que $\mathbf{q_2}$ recupera pocos documentos pero muy similares al tópico de interés ¿Cuál de estas consultas será preferible seleccionar? Debemos definir un criterio para decidir cuál de ellas preferimos que tenga más probabilidades de sobrevivir. Dado que nuestro objetivo es recuperar documentos con máxima similitud, las consultas que recuperan pocos documentos muy similares al tópico de interés, son más importantes que las consultas que recuperan muchos documentos pero con similitud baja. Es decir, en el ejemplo de la figura, serán mejores los conjuntos A, B y D, dado que los documentos finalmente seleccionados serán aquellos que más se asemejen al tópico de interés y no todos los presentes en el conjunto de documentos recuperados. En base a lo anterior, definimos la siguiente función de fitness:

Definición 19 (Función de Fitness Basada en Similitud Máxima, Fitness)**.** *Sea una consulta* $\mathbf{q}$ *y un tópico de interés o contexto temático* c *(el cual puede ser considerado como un documento) y una métrica de similitud* $\sigma : D \times D \rightarrow [0, 1]$ *definida sobre un par de documentos. Sea* $\mathbf{A_q}$ *el conjunto de documentos recuperados por un motor de búsqueda cuando* $\mathbf{q}$ *se usa como consulta. La función de fitness basada en similitud máxima,* Fitness $: Q \rightarrow [0, 1]$ *se define como:*

$$\text{Fitness}(\mathbf{q}) = \max_{d_i \in \mathbf{A_q}} (\sigma(c, d_i)) \tag{6.8}$$

En base a esta definición, podemos utilizar distintas métricas de similitud, tales como la similitud por coseno estándar (transcripta en la Definición 16), que mide la cercanía de documentos en base al ángulo de separación entre los vectores que lo representan, o

el coeficiente de similitud de Jaccard el cual permite medir la similitud de conjuntos de muestras en base a la cardinalidad de los conjuntos intersección y unión [BYRN99]. Como primer paso para la implementación mono-objetivo, elegimos como métrica la similitud por coseno. Sin embargo, dado que también estamos interesados en recuperar material novedoso, en la Sección 6.6.5 presentamos una nueva métrica de similitud que favorece a las consultas que recuperan material nuevo.
Una dificultad pragmática en nuestra definición de fitness en la ecuación 6.8 es el uso del conjunto de resultados completo $\mathbf{A_q}$. Mirar el conjunto completo de páginas recuperadas por el motor de búsqueda es extremadamente costoso en la práctica. Por lo tanto, para determinar cuan buena es una consulta se tienen en cuenta dos consideraciones. Primero, sólo se miden en similitud los diez primeros resultados del conjunto recuperado por cada consulta. Segundo, en lugar de analizar documentos completos (los cuales pueden ser sumamente extensos) el algoritmo analiza una descripción de cada documento. Estas descripciones son un servicio común que suelen ofrecer los motores de búsqueda con el propósito de brindar un pequeño resumen del documento, y normalmente se las conoce como *snippets*. Por ejemplo, Google y Yahoo permiten que los usuarios visualicen un pequeño *snippet*, en el cual entre otras características se puede ver un fragmento del documento.

6.6.4 Desempeño de la arquitectura

Criterios aplicados para la evaluación del desempeño

Para evaluar el rendimiento de la recuperación temática lograda por el algoritmo, primero debemos establecer un criterio de evaluación útil para esta tarea. Dado que el corpus utilizado en esta implementación es la web, como sucede en la mayoría de los sistemas que acceden a la web para recuperar material relevante, nuestros mecanismos de evaluación deben tener en cuenta que no se puede identificar el conjunto completo de documentos relevantes. Por lo tanto, para poder evaluar el rendimiento del EA a lo largo de las generaciones se adoptó un criterio de evaluación basado en la calidad de las consultas de toda la población. Este criterio nos permite asignar un valor de rendimiento a cada generación. Un incremento (o decremento) en este valor de calidad durante dos generaciones sucesivas significa una mejora (o deterioro) en el rendimiento para dichas generaciones. En nuestra opinión, una aproximación evolutiva para recuperación de información temática es exitosa si la calidad de las consultas de la última generación supera

a la calidad de las consultas de la generación inicial. Notemos que el rendimiento del sistema para las consultas de la generación inicial puede tomarse como una cota inferior, ya que dichas consultas son generadas utilizando términos seleccionados directamente del tópico de interés inicial, es decir, sin aplicar inteligencia alguna.
Para definir nuestro criterio de evaluación del rendimiento se utilizó la siguiente definición de σ basada en la definición de similitud por coseno:

Definición 20 (σ)**.** *Sea un tópico de interés o contexto temático c, una consulta* $\mathbf{q}$ *y sea* $\mathbf{A_q} = \{a_1, \ldots, a_2\}$ *el conjunto de recursos recuperados para* $\mathbf{q}$*. Determinamos la similitud,* σ*, entre c y uno de los resultados recuperados,* a_i*, calculando la similitud por coseno definida como:*

$$\sigma(c, a_i) = \frac{\overrightarrow{c} \cdot \overrightarrow{a}_i}{\|\overrightarrow{c}\| \cdot \|\overrightarrow{a}_i\|} \tag{6.9}$$

$\overrightarrow{c}$ *es la representación vectorial del tópico de interés obtenida a partir de los términos de c, y* $\overrightarrow{a}_i$ *es la representación vectorial de* a_i *basada en los términos de la descripción obtenida para* a_i *por medio del motor de búsqueda.*

Utilizando la definición de σ previa, definimos dos criterios para la evaluación del rendimiento:

- *Criterio basado en calidad máxima.* La calidad de una consulta $\mathbf{q}$ está dada por el máximo valor de similitud alcanzado por los documentos recuperados para esa consulta.

  Definición 21 (Calidad_Máxima)**.** *Dado un tópico de interés o contexto temático c, una consulta* $\mathbf{q}$ *y el conjunto de documentos recuperados cuando* $\mathbf{q}$ *se presenta al motor de búsqueda,* $\mathbf{A_q} = \{a_1, \ldots, a_2\}$*, la calidad máxima se define como:*

  $$\text{Calidad\_Máxima}(\mathbf{q}) = \max_{a_i \in \mathbf{A_q}} (\sigma(c, a_i)). \tag{6.10}$$

  Notemos que la función Calidad_Máxima coincide con la definición de función de fitness presentada en la Definición 19. Es de esperar que a lo largo de las generaciones este valor mejore, ya que es el que se utiliza para conducir las etapas del algoritmo evolutivo.

- *Criterio basado en calidad promedio.* La calidad de una consulta $\mathbf{q}$ está dada por el valor promedio de similitud alcanzado por los documentos recuperados para esa consulta.

Definición 22 (Calidad_Media)**.** *Dado un tópico de interés o contexto temático c, una consulta* $\mathbf{q}$ *y el conjunto de documentos recuperados cuando* $\mathbf{q}$ *se presenta al motor de búsqueda,* $\mathbf{A_q} = \{a_1, \ldots, a_2\}$*, la calidad media se define como:*

$$\text{Calidad\_Media}(\mathbf{q}) = \frac{\sum_{a_i \in \mathbf{A_q}} (\sigma(c, a_i))}{|\mathbf{A_q}|} \tag{6.11}$$

La función Calidad_Media es el promedio de los valores de similitud calculados para todos los pares (c, a_i). Al calcular el promedio para todos los pares estamos evaluando la calidad global del conjunto recuperado y no sólo al mejor de sus documentos.

Dependiendo de la meta propuesta, podemos preferir una métrica o la otra. Por ejemplo, la *Calidad Máxima* es más apropiada si estamos analizando la capacidad del sistema para recuperar un único documento muy relevante. Alternativamente, la *Calidad Media* combina la relevancia de un conjunto de resultados evaluándolos en grupo, por lo que puede ser más adecuada si estamos evaluando la capacidad del sistema para recuperar varios resultados relevantes.

Resultados

Para poder testear el rendimiento del EA los criterios de evaluación definidos en la sección anterior requieren que tengamos acceso a un tópico de interés o contexto temático c. Con este objetivo, obtuvimos seis descripciones de tópicos seleccionando tres tópicos pertenecientes al directorio DMOZ (dmoz.org). Los tópicos que resultaron seleccionados para realizar los testeos son *Business, Recreation* y *Society*. Para cada tópico se realizaron cinco corridas del EA. Cada corrida evolucionó durante 20 generaciones, con una población de 60 consultas, una probabilidad de recombinación de 0.7 (70%) y una probabilidad de mutación de 0.03 (3%). Como punto de partida para la población inicial, se decidió optar por palabras seleccionadas de la descripción propia del tópico bajo consideración. Para el caso del directorio DMOZ, esta descripción es realizada manualmente para cada tópico y subtópico por editores de la ontología, de manera que dicha descripción brinda una idea general del tópico a los usuarios que están recorriendo el directorio. A manera de ejemplo, la figura 6.6.4 muestra la descripción para el tópico *"Business/Business and Society"*.
En base a lo anterior, la población se inicializó generando 60 consultas, cuya longitud fue

Business/Business and Society

" The Business and Society category contains sites which pertain to the interaction between businesses, organizations, and governmental bodies. Most sites in this category will focus on the social, economic, or environmental impact of businesses, while many will deal with ethics and social responsibility as it pertains to business "

Figura 6.5: Ejemplo de descripción. Descripción disponible en DMOZ para el tópico *"Business/Business and Society"*.

elegida al azar entre 1 y 32 palabras. Cada término fue seleccionado en forma aleatoria de un conjunto inicial de palabras conformado por las palabras de la descripción del tópico, sacando las palabras que son muy frecuentemente usadas (tales como *a* o *the* en inglés). Para realizar un análisis estadístico del algoritmo, se tuvieron en cuenta las cinco corridas realizadas sobre cada tópico. Los resultados obtenidos para estas cinco corridas fueron agrupados de acuerdo al número de generación. Para cada generación se analizó la calidad promedio de los resultados utilizando tanto la métrica de $Calidad\_Máxima$ como la métrica de $Calidad\_Media$. Además, se calcularon las barras de error utilizando un 95% de intervalo de confianza de la media.

En el caso de $Calidad\_Máxima$, los resultados se evalúan según la Definición 21, es decir, para cada individuo de cada generación se toma el *máximo valor obtenido por los documentos recuperados* por dicho individuo. Posteriormente se calcula un promedio estadístico sobre varias corridas (cinco en este caso) para el mejor individuo de cada generación. Mientras que en el caso de la $Calidad\_Media$, la evaluación se hace según la Definición 22, es decir, para cada individuo de cada generación se toma el *valor promedio de los resultados obtenidos por los documentos recuperados* por dicho individuo. Luego

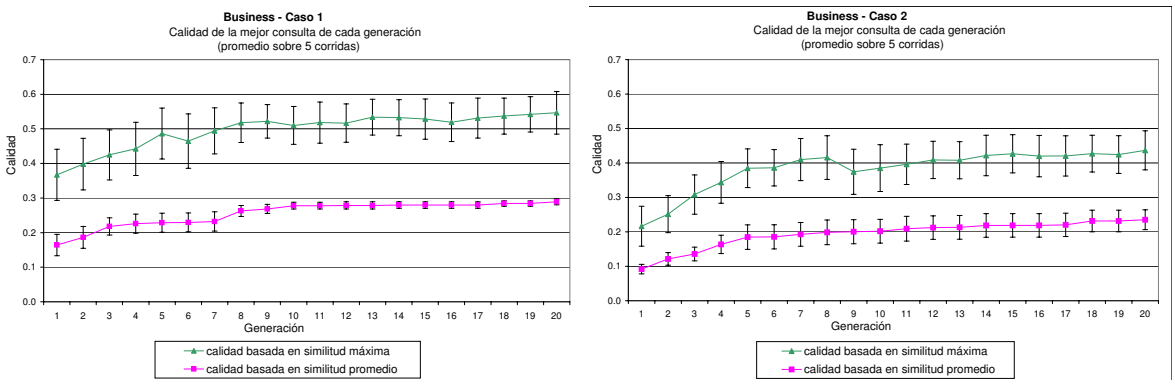

Figura 6.6: Dos testeos sobre el tópico *Business* que muestran la calidad promedio de las consultas sobre cinco corridas independientes.

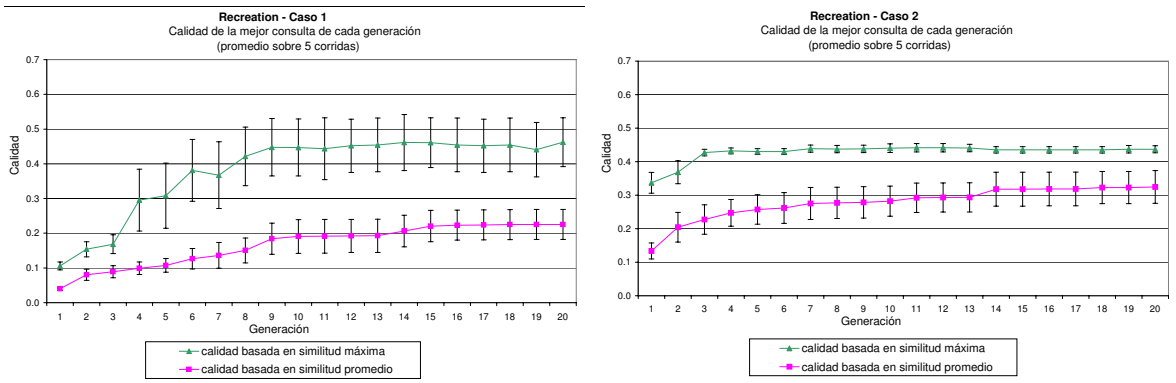

Figura 6.7: Dos testeos sobre el tópico *Recreation* que muestran la calidad promedio de las consultas sobre cinco corridas independientes.

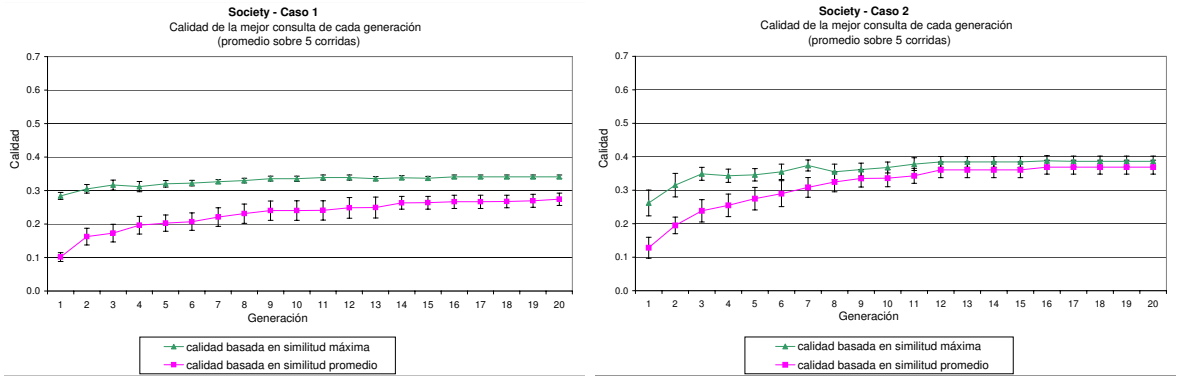

Figura 6.8: Dos testeos sobre el tópico *Society* que muestran la calidad promedio de las consultas sobre cinco corridas independientes.

se calcula un promedio estadístico sobre las cinco corridas para el mejor individuo de cada generación.

Las figuras 6.6 - 6.8 muestran el desempeño del algoritmo para los seis tópicos seleccionados. En todos los casos, la comparación de la calidad de las consultas obtenidas a través de un pequeño número de generaciones muestra que los resultados del algoritmo evolutivo logran mejoras estadísticamente significativas sobre las generaciones iniciales (podemos notar que en todos los casos las barras de error correspondientes a la primera generación no se solapan con las barras de error de la última generación). Esto significa que el EA es capaz de evolucionar consultas con una calidad considerablemente superior a la de las consultas generadas directamente a partir de la descripción de tópico.

Efectos de aplicar diferentes tasas de mutación

Una vez que el algoritmo evolutivo ha demostrado efectividad en esta primera aproximación, es interesante saber cómo afectan las diferentes tasas de mutación al desempeño del algoritmo y a la diversidad poblacional. Para evaluar este aspecto, se realizaron nuevas corridas para los mismos tópicos utilizando los mismos parámetros pero variando la tasa de mutación.

Se utilizaron tres tasas de mutación: $P_m = 0$ (mutación nula), $P_m = 0,03$ (mutación clásica) y $P_m = 0.3$ (mutación alta). Luego de aplicar las distintas probabilidades observamos que la diversidad en los valores de similitud se comportan en base a lo esperado, es decir, que la diversidad aumenta con la probabilidad de mutación. La figura 6.9 muestra la evolución de una población de 60 consultas que evolucionaron durante 20 generaciones sobre el tópico *Business* para las tres tasas de mutación. Se puede observar que la nube de puntos representando a las poblaciones presenta más variaciones en el tercer caso. Además, podemos ver que el mejor comportamiento se da para la mutación normal ($P_M = 0.03$) ya que puede observarse que tanto los valores más altos como los más bajos tienen una tendencia a mejorar.

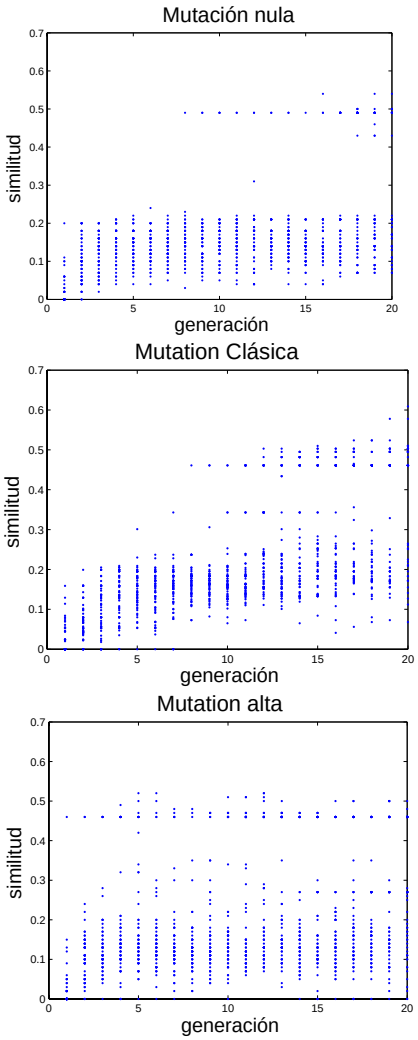

Figura 6.9: Diagramas de dispersión mostrando la distribución de los valores de similitud para los mejores resultados asociados con los individuos para cada generación con $P_m = 0$ (superior), $P_m = 0,03$ (centro) y $P_m = 0.3$ (inferior) para el tópico *Business*.

Por otra parte, se realizó un análisis comparativo del desempeño del algoritmo para los

tres casos de mutación mediante la definición de *Calidad Máxima* (definición 21) de la forma en que se explicó en la sección anterior.
Por medio de este análisis pudimos observar que si la probabilidad de mutación es muy baja (o nula en este caso) el algoritmo presenta mayor dificultad para converger a valores elevados de similitud. Mientras que, con valores muy altos de mutación se presenta mayor variación e inestabilidad. La figura 6.10 muestra los resultados obtenidos para los tópicos *Business*, *Recreation* y *Society* respectivamente.

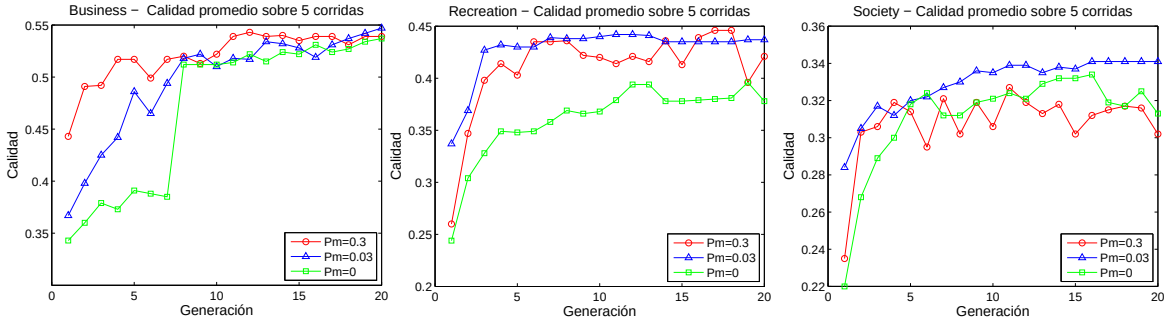

Figura 6.10: Calidad promedio de las consultas calculada sobre cinco corridas independientes para los tópicos *Business, Recreation* y *Society*; sin usar mutación ($P_m = 0$), usando la probabilidad de mutación clásica ($P_m = 0.03$) y usando una probabilidad de mutación elevada ($P_m = 0.3$).

Un análisis estadístico destinado a comparar la calidad obtenida en la primera generación con respecto a la calidad obtenida en la última generación, muestra que se obtiene una mejora sustancial al permitir que las consultas evolucionen durante 20 generaciones, y en la mayoría de los casos esta mejora resulta estadísticamente significativa dado que los intervalos de confianza obtenidos no se solapan entre sí.

	MEDIA	95% C.I.
g=1	0.343	**(0.264,0.421)**
g=20	0.537	**(0.500,0.574)**

$P_m = 0$

	MEDIA	95% C.I
g=1	0.367	**(0.305,0.429)**
g=20	0.547	**(0.530,0.564)**

$P_m = 0.03$

	MEDIA	95% C.I.
g=1	0.443	(0.375,0.511)
g=20	0.539	(0.404,0.673)

$P_m = 0.3$

Tabla 6.1: Comparación entre la primera generación y la última generación. Intervalos de confianza para calidad promedio de las consultas para el tópico *Business*.

	MEDIA	95% C.I.
g=1	0.244	**(0.225,0.264)**
g=20	0.378	**(0.336,0.420)**

Pm=0

	MEDIA	95% C.I.
g=1	0.337	**(0.289,0.385)**
g=20	0.437	**(0.395,0.479)**

Pm=0.03

	MEDIA	95% C.I.
g=1	0.260	**(0.219,0.300)**
g=20	0.421	**(0.380,0.463)**

Pm=0.3

Tabla 6.2: Comparación entre la primera generación y la última generación. Intervalos de confianza para calidad promedio de las consultas para el tópico *Recreation*.

Las tablas 6.1, 6.2 y 6.3 muestran los valores obtenidos con las tres probabilidades de mutación para los tópicos *Business, Recreation* y *Society* respectivamente. Los intervalos de confianza resaltados en negrita son los casos en los cuales no hay solapamiento alguno

	MEDIA	95% C.I.
g=1	0.220	**(0.202,0.237)**
g=20	0.313	**(0.243,0.383)**

Pm=0

	MEDIA	95% C.I.
g=1	0.284	(0.258,0.311)
g=20	0.341	(0.304,0.378)

Pm=0.03

	MEDIAS	95% C.I.
g=1	0.235	(0.204,0.267)
g=20	0.302	(0.222,0.381)

Pm=0.3

Tabla 6.3: Comparación entre la primera generación y la última generación. Intervalos de confianza para calidad promedio de las consultas para el tópico *Society*.

entre los valores de calidad obtenidos por las consultas de la primera generación y los valores obtenidos en la generación 20.

Efectos de no aplicar elitismo. Como una característica común a todos los tópicos analizados, podemos observar que se presentan altibajos en la calidad de los individuos, es decir, que las curvas no son crecientes en todo su rango. Dado que los operadores genéticos introducen cierto grado de diversidad en los individuos, es natural pensar que la calidad promedio de la población pueda decaer momentáneamente a lo largo de la evolución. Sin embargo, cuando el criterio de calidad analizado es el promedio de los valores obtenidos para Calidad_Máxima (la cual coincide con la función de fitness utilizada en la evolución), podríamos esperar que la curva sea creciente o al menos que no disminuya, ya que una disminución significa que estamos perdiendo a los individuos que han obtenido el máximo valor de Fitness. Esta pérdida se debe a que el esquema de selección utilizado no es en absoluto elitista, dando lugar a posibles pérdidas de individuos altamente calificados. Con el objetivo de no perder a los mejores individuos, las implementaciones siguientes incluyen algún nivel de elitismo. Algunos de estos resultados se reportan en las siguientes secciones.

6.6.5 Incorporación de elitismo y función de aptitud basada en similitud novedosa

En esta sección se proponen y estudian dos mejoras para el algoritmo inicial. Uno de los aspectos abordados es la influencia del elitismo como mecanismo de preservación de los mejores individuos. En los resultados obtenidos hasta aquí, podemos observar que existen altibajos en el valor promedio para la medida de Calidad_Máxima, es decir, la calidad promedio para varias corridas de la mejor de las consultas de cada generación puede bajar de una generación a la siguiente. Esto se debe a que el proceso de selección no es elitista, significando que si bien los individuos con mayor aptitud tienen grandes posibilidades de sobrevivir y pasar a la siguiente generación, no hay una certeza de que el mejor o los mejores se preserven a lo largo de la evolución. En esta sección mostramos los resultados

obtenidos por una versión que adopta un esquema elitista simple que mantiene al mejor de los individuos. Otra de las cuestiones abordadas es la forma de conseguir material relacionado temáticamente a lo que se busca, pero que resulte novedoso con respecto a lo que ya se tiene. En los resultados anteriores hemos asumido que mientras mayor es la similitud entre el tópico de interés y los documentos recuperados por una consulta, mayor es la calidad de la consulta. Sin embargo, algunos esquemas en IR optan por una postura diferente. En [BHB01], Budzik y sus colegas discuten la utilidad de analizar la forma en la que los usuarios manipulan información, estudiando no sólo la similitud del material recuperado sugerido sino también si dicho material le resulta útil a los usuarios con el fin de recomendar información adicional. El aspecto principal que plantean los autores es que no siempre un documento similar al contexto del usuario le resultará útil al usuario. En [SM01] Smyth y McClave argumentan que en algunas situaciones la diversidad puede ser tan importante como la similitud y proponen algunas estrategias para mejorar la diversidad manteniendo similitud. En [MLR05] postulan la necesidad de obtener material que sea simultáneamente nuevo y esté relacionado con el tema de interés. En ciertas circunstancias, lograr diversidad y novedad puede ser tan importante como (o más importante que) obtener máxima similitud. Esto no es algo extraño ya que, si subimos a nivel de tópico veremos que los documentos que lo componen pueden ser muy distintos entre sí (hablando a nivel de similitud de los términos que los componen) y, a pesar de esto, pertenecer al mismo tópico.
Como alternativa a la noción convencional de similitud proponemos el uso de una nueva medida de similitud, definida de la siguiente manera:

Definición 23 (σ^N)**.** *Dado un tópico de interés o contexto temático c, una consulta $\mathbf{q}$ y el conjunto de items recuperados cuando $\mathbf{q}$ se presenta a un motor de búsqueda, $\mathbf{A_q} = \{a_1, \ldots, a_2\}$, la Similitud Novedosa, $\sigma^N : Q \times D \times D \rightarrow [0, 1]$, se define como:*

$$\sigma^N(\mathbf{q}, c, a_i) = \frac{\overrightarrow{c - \mathbf{q}} \cdot \overrightarrow{a_i - \mathbf{q}}}{\|\overrightarrow{c - \mathbf{q}}\| \cdot \|\overrightarrow{a_i - \mathbf{q}}\|} \tag{6.12}$$

Donde $\overrightarrow{d - \mathbf{q}}$ es la representación vectorial del documento d en el espacio de términos con todos los valores correspondientes a las palabras de la consulta $\mathbf{q}$ forzados a cero (se aplica el mismo significado a los casos en los que $d = a_i$ y $d = c$).

Esta nueva medida persigue dos objetivos:

1. Asegurar que la mejora en la calidad de las consultas a través de las generaciones no se debe solamente al hecho de que estemos tomando cada vez más términos de la descripción del tópico de interés. Notemos que si se usa la descripción original de similitud, incorporar muchos términos de la descripción temática para formular consultas podría garantizar un valor alto de similitud entre el tópico de interés y los documentos recuperados. Sin embargo, descartando los términos de la consulta cuando computamos σ^N evitamos introducir esta tendencia.
2. Promover la obtención de documentos novedosos relacionados con el tópico de interés por medio de la introducción de nuevos términos.

A partir de la Definición 23 podemos definir una nueva función de fitness basada en similitud novedosa, Fitness^N, de la siguiente forma:

Definición 24 (Función de aptitud basada en similitud novedosa, Fitness^N)**.** *Consideremos una consulta* $\mathbf{q}$*, un tópico de interés o contexto temático* c *(el cual puede ser considerado como un documento), la métrica de similitud,* $\sigma^N : Q \times D \times D \rightarrow [0,1]$*, definida sobre una consulta y un par de documentos. Sea* $\mathbf{A_q}$ *el conjunto de documentos recuperados por un motor de búsqueda cuando se usa* $\mathbf{q}$ *como consulta, la función* Fitness^N*:* $Q \rightarrow [0,1]$ *basada en similitud novedosa se define como:*

$$Fitness^N(\mathbf{q}) = \max_{d_i \in \mathbf{A_q}} (\sigma^N(\mathbf{q}, c, d_i)) \tag{6.13}$$

Nuevamente es preciso formular criterios de evaluación para determinar el desempeño del algoritmo. Por lo tanto, en forma análoga a como se hizo anteriormente, definimos las siguientes dos medidas:

Definición 25 (Calidad_Máxima^N)**.** *Dado un tópico de interés o contexto temático* c*, una consulta* $\mathbf{q}$*, el conjunto de documentos recuperados cuando* $\mathbf{q}$ *se presenta al motor de búsqueda,* $\mathbf{A_q} = \{a_1, \ldots, a_2\}$ *y la medida de similitud novedosa* σ^N*, la calidad novedosa máxima se define como:*

$$Calidad\_Máxima^N(\mathbf{q}) = \max_{a_i \in \mathbf{A_q}} (\sigma^N(\mathbf{q}, c, a_i)). \tag{6.14}$$

Notemos que la función Calidad_Máxima^N coincide con la definición de función de fitness definida en la Definición 24. Es de esperar que a lo largo de las generaciones este valor mejore, ya que es el que se utiliza para conducir las etapas del algoritmo evolutivo.

Definición 26 (Calidad_Media^N)**.** *Dado un tópico de interés o contexto temático c, una consulta* $\mathbf{q}$*, el conjunto de documentos recuperados cuando* $\mathbf{q}$ *se presenta al motor de búsqueda,* $\mathbf{A_q} = \{a_1, \ldots, a_2\}$ *y la medida de similitud novedosa* σ^N*, la calidad novedosa media se define como:*

$$Calidad\_Media^N(\mathbf{q}) = \frac{\sum_{a_i \in \mathbf{A_q}} (\sigma^N(\mathbf{q}, c, a_i))}{|\mathbf{A_q}|} \tag{6.15}$$

Resultados

Luego de ejecutar el algoritmo cinco veces, cada una durante 20 generaciones, con los mismos parámetros que en los experimentos anteriores, pero aplicando un esquema de selección elitista pudimos comprobar que la nueva aproximación responde en forma positiva a los objetivos propuestos. La figura 6.11 muestra el desempeño del algoritmo aplicando

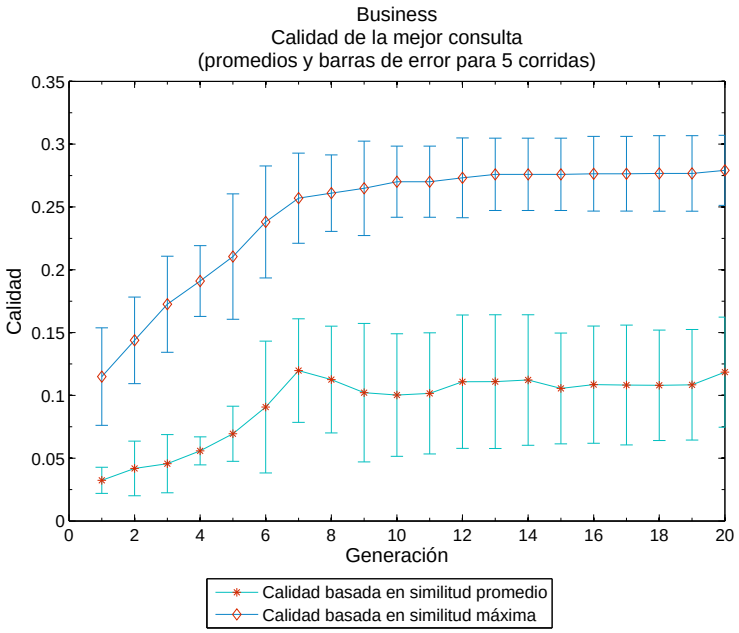

Figura 6.11: Calidad promedio de la mejor consulta calculada sobre cinco corridas independientes para el tópico *Business*, para los resultados obtenidos utilizando σ^N.

Similitud novedosa para cinco corridas del algoritmo. Se puede apreciar tanto la calidad promedio de la mejor consulta usando Calidad_Máxima^N y Calidad_Media^N, como las barras de error (con un 95 de I.C.) para cinco corridas. Una vez más la comparación entre la calidad de las consultas de la primera y última generación muestra que el algoritmo evolutivo obtiene resultados con una mejora estadísticamente significativa. También se

puede apreciar que, gracias al elitismo, el valor para calidad basada en máximo no disminuye de una generación a la siguiente.

Un experimento similar utilizando los mismos parámetros pero con 20 corridas, sobre los tópicos *Business, Recreation* y *Society*, alcanzó resultados similares. En las figuras 6.12 – 6.14 se muestran los promedios y los desvíos estándar para las 20 corridas del algoritmo para los tópicos Business, Recreation y Society respectivamente.

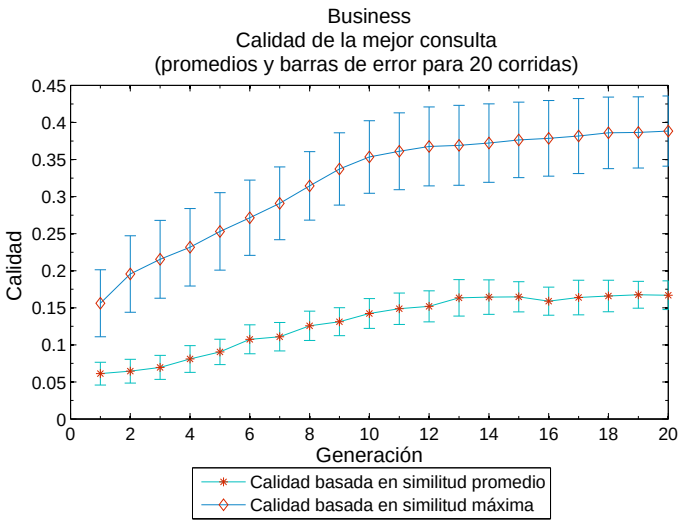

Figura 6.12: Calidad promedio de la mejor consulta calculada sobre 20 corridas independientes para el tópico *Business*, para los resultados obtenidos utilizando σ^N.

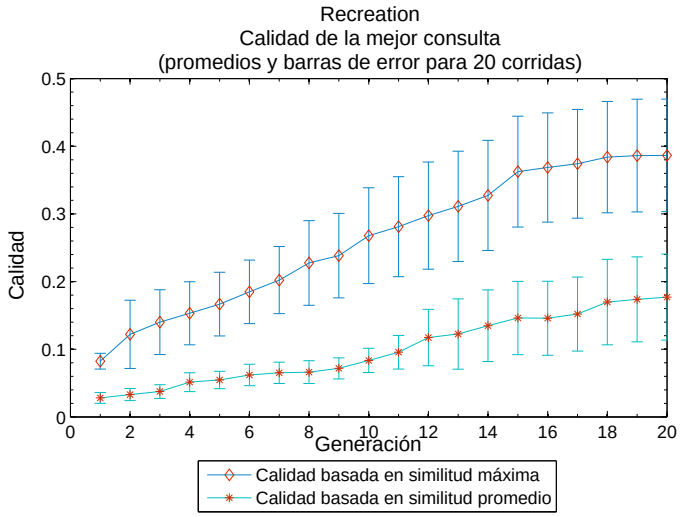

Figura 6.13: Calidad promedio de la mejor consulta calculada sobre 20 corridas independientes para el tópico *Recreation*, para los resultados obtenidos utilizando σ^N.

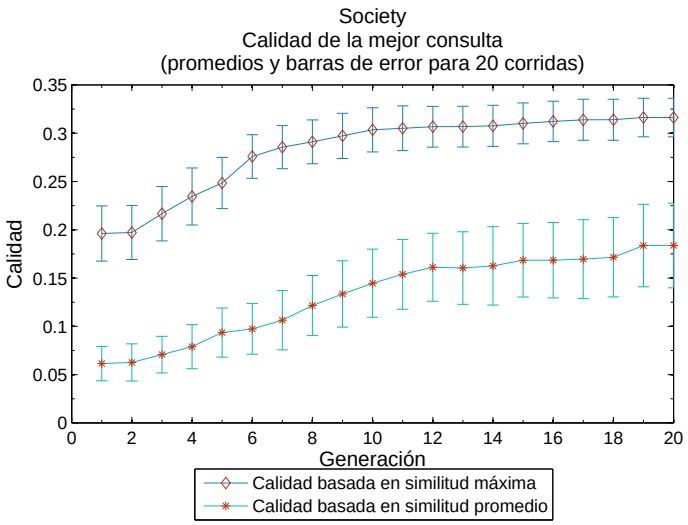

Figura 6.14: Calidad promedio de la mejor consulta calculada sobre 20 corridas independientes para el tópico *Society*, para los resultados obtenidos utilizando σ^N.

Por último, buscamos nuevos términos introducidos por el proceso de búsqueda y descubrimos que muchos de ellos resultan ser buenos descriptores para el tópico de interés, aún cuando estos no formaran parte del contexto inicial. Por ejemplo, para el tópico del gráfico anterior la descripción inicial está relacionada con *compañías de mercadeo que proveen soluciones para otras compañías planificando ingresar al mercado Norte Americano*[1], el sistema descubrió términos novedosos como: *incremento, colaboración, financiación, CRM, AAA y AOL* [2] entre muchas otras. Uno de los mejores individuos encontrados por el algoritmo evolutivo en este experimento fue la consulta: $\mathbf{q} =$ "*ABA*[3] *Strategic curricular Market educational defined vital Market improve Strategic Preparation project ABA meese*", con Calidad_Máxima$^N(\mathbf{q}) = 0.27247$.

6.7 Segunda implementación: versión multi-objetivo

A partir de la primera versión del núcleo evolutivo de la arquitectura propuesta y en base al comportamiento óptimo logrado para nuestro primer problema, nuestro siguiente paso consistió en la resolución de un problema con un grado más de complejidad, el cual

[1] Resumen en Inglés de la descripción: *marketing companies that provide solutions to other companies planning to enter the North-American marketingplace*

[2] Algunas de las palabras en Inglés encontradas que no estaban en la descripción inicial: *grow, collaboration, financing, CRM (customer relationship management), AAA (American Automobile Association), AOL (America On Line).*

[3] La Amecican Bar Association (ABA) es un grupo estadounidense de más de 400.000 abogados y estudiantes de leyes asociados de forma voluntaria.

abarcó dos objetivos contrapuestos de gran relevancia en IR. Como parte de la evolución natural de nuestra arquitectura, la segunda instancia está orientada a cumplir el objetivo general de recuperar material temáticamente relacionado con un tópico de interés al mismo tiempo que se recupera la mayor cantidad posible de material relacionado y la menor cantidad posible de material no relacionado.
Cuando nos basamos sólo en la métrica de similitud para determinar la efectividad de una consulta, no evaluamos cuantos de los *documentos relevantes* existentes estamos recuperando ni tampoco cuantos *documentos no relevantes* recuperamos. Una consulta podría recuperar una gran cantidad de documentos (relevantes y no relevantes), teniendo grandes posibilidades de obtener un ítem de mucha similitud con respecto al tópico de interés pero a costa de recuperar muchos otros documentos de muy baja similitud. En tanto que otra consulta que recupera menos cantidad de documentos (relevantes y no relevantes), podría recuperar algún documento no tan similar pero ser más precisa en la recuperación obteniendo una mejor relación entre documentos relevantes existentes, los documentos relevantes recuperados y los documentos no relevantes recuperados. Para evaluar estos aspectos utilizamos las bien conocidas métricas de *precisión* y *cobertura*, intentando maximizar ambas medidas por medio de algoritmos evolutivos multi-Objetivo.

6.7.1 Cuestiones de investigación

- Cuestión 1: ¿Cómo podemos evolucionar y establecer un rango a las consultas cuando se persiguen múltiples objetivos, tales como lograr valores elevados de precisión y cobertura?
- Cuestión 2: ¿Es mejor observar ambos objetivos pero independientemente uno de otro en un esquema Pareto, que mirarlos conjuntamente dentro de un esquema agregativo utilizando métricas conocidas de IR?
- Cuestión 3: ¿Como son los resultados obtenidos luego de la evolución con respecto a los iniciales?
- Cuestión 4: ¿Las consultas generadas en la etapa de entrenamiento por medio del EA son útiles más allá de dicha etapa? En otras palabras, ¿las mejores consultas obtenidas por el EA son efectivas cuando se las testea para el mismo tópico pero dentro de un nuevo corpus de documentos?

6.7.2 Esquema evolutivo multi-objetivo para evolución de consultas temáticas

Con el objetivo general de formular consultas de manera automática para recuperar material en base a un contexto de interés, la propuesta multi-objetivo asume que existe un índice subyacente que puede ser accedido por medio de consultas realizadas a una interfaz de búsqueda. La infraestructura utiliza una ontología para entrenar y probar distintos algoritmos evolutivos. Los documentos clasificados en la ontología se usan para crear dos índices no superpuestos: un índice de entrenamiento y un índice de testeo. La infraestructura utiliza el índice de entrenamiento para evolucionar una población de consultas temáticas. Por medio de las consultas realizadas a la interfaz de búsqueda el componente de evaluación de aptitud puede conocer qué documentos se recuperan con dicha consulta y cuáles y cuántos pertenecen al tópico de interés. Esta información es vital para el cálculo de las métricas de calidad consideradas. Los algoritmos evolutivos multi-objetivo implementados para realizar el ciclo evolutivo de la infraestructura fueron: el algoritmo NSGA-II y un algoritmo que agrega ambos objetivos en una única fórmula. Además, para realizar un análisis comparativo también se hicieron experimentos con un algoritmo evolutivo mono-objetivo que contempla sólo el objetivo *precisión* y con otro que contempla sólo el objetivo *cobertura*.

6.7.3 Selección

En los casos en los que la efectividad de la consulta puede ser codificada como un escalar (p. ej. en un esquema agregativo), se utiliza el operador de selección por torneo binario. Para el esquema basado en Pareto se utiliza el esquema de selección basado en densidad correspondiente al NSGA-II (*crowded tournament selection*).

6.7.4 Evaluación de las consultas

Para determinar la efectividad de una consulta se utilizó una representación vectorial de la consulta conjuntamente con el modelo TFIDF. En este modelo, las consultas se interpretan con una semántica disyuntiva, esto significa que la coincidencia en una palabra es suficiente para recuperar un documento. Este tipo de configuración normalmente puede conducir a un gran número de items ordenados por su similitud con el vector que representa a la consulta actual. Por lo tanto, en lugar de utilizar la medida de *precisión* original, utilizamos precisión a rango 10, *Precisión@10* [HC01], definida como la fracción de los

10 primeros documentos recuperados que corresponde a documentos relevantes. Para definir esta función asociamos con el espacio de búsqueda de todas las posibles consultas que pueden ser presentadas al motor de búsqueda Q y los tópicos T comprendidos en el índice, una función basada en la ecuación 14 definida de la siguiente manera:

Definición 27 (*Precisión@10*)**.** *Dada una consulta q y un tópico de interés t, la función Precisión@10: $Q \times T \rightarrow [0,1]$ se define como:*

$$\textit{Precisión@10}(q,t) = \frac{|D_{q10} \cap D_t|}{|D_{q10}|} \tag{6.16}$$

donde D_{q10} es el conjunto de los diez primeros documentos recuperados por el motor de búsqueda cuando se usa q como consulta y D_t es el conjunto que contiene a todos los documentos del índice asociados al tópico t.

Esta función permite evaluar a una consulta q en términos de su precisión a rango 10 con respecto a un tópico t.
La función de cobertura sobre $Q \times T$ se define en base a la ecuación 15 de la siguiente manera:

Definición 28 (*Cobertura*)**.** *Dada una consulta q y un tópico de interés t, la función Cobertura : $Q \times T \rightarrow [0,1]$ se define como:*

$$\textit{Cobertura}(q,t) = \frac{|D_q \cap D_t|}{|D_t|}. \tag{6.17}$$

donde D_q es el conjunto de documentos recuperados cuando se presenta q al motor de búsqueda y D_t es el conjunto que contiene a todos los documentos del índice asociados con el tópico t.

En base a estas definiciones se construyeron dos algoritmos evolutivos mono-objetivo, uno que intenta maximizar *Precisión@10* y otro que procura maximizar *Cobertura*, el NSGA-II que procura maximizar ambos objetivos de forma independiente uno de otro con un esquema Pareto, y un algoritmo evolutivo agregativo que intenta maximizar ambos objetivos por medio de la función $F^{\star}$.

Definición 29 ($F^\star$)**.** *Dada una consulta q y un tópico de interés t, la función $F^\star$: $Q \times T \rightarrow [0,1]$ se define como:*

$$F^\star(q,t) = \frac{2 \cdot \mathit{Precisión@10}(q,t) \cdot \mathit{Cobertura}(q,t)}{\mathit{Precisión@10}(q,t) + \mathit{Cobertura}(q,t)}. \tag{6.18}$$

Esta métrica es una adaptación de la media armónica ponderada de precisión y cobertura F_1 [Rij79].

6.7.5 Rendimiento del algoritmo

Construcción de los índices

Para construir los índices se recopilaron 448 tópicos del Open Directory Proyect (ODP - http://dmoz.org). Los tópicos fueron seleccionados del tercer nivel de la jerarquía ODP y, para asegurar la calidad del corpus, se tuvieron en cuenta las siguientes consideraciones: para cada tópico se recopilaron todos sus URLs así como los de sus subtópicos, sólo se consideraron tópicos con un mínimo de 100 URLs y se restringió el lenguaje al idioma Inglés. Teniendo en cuenta dichas consideraciones se recolectaron más de 350000 páginas. Cada uno de los 448 tópicos se dividió para utilizar 2/3 de sus páginas con el fin de construir el índice de entrenamiento y el 1/3 restante para el índice de testeo. Para realizar ambos índices se utilizó la infraestructura Terrier [OLMP07], empleando la lista de palabras vacías (stopword list) provista por el sistema y realizando Porter Stemming en todas las palabras (es decir, la reducción de cada palabra a su raíz léxica eliminando sufijos y afijos). El conjunto inicial de palabras de la infraestructura evolutiva para cada tópico se construyó a partir de la descripción del tópico provista por ODP.

Parámetros de los algoritmos evolutivos

Los parámetros evolutivos se mantuvieron constantes para los cuatro algoritmos evolutivos ejecutados durante la etapa de entrenamiento. Se configuró una población de 250 consultas que evolucionaron durante un máximo de 300 generaciones, utilizando una tasa de recombinación clásica de 0,7 y, basándonos en los estudios previos, una probabilidad de mutación de 0,03. Las consultas de la población inicial fueron formuladas con palabras de la descripción del tópico analizado. El tamaño de cada consulta fue un número de entre 1 y 32 seleccionado en forma aleatoria. Sin embargo, cabe aclarar que el operador de recombinación puede hacer que posteriormente las consultas crezcan más allá de dicho límite.

Si bien los términos que sobrepasan el límite impuesto son ignorados al momento de hacer la consulta a la interfaz de búsqueda, pueden volver a ser considerados posteriormente por nuevas consultas si luego de la recombinación quedaran dentro del límite.

Organización de los experimentos

En cada experimento se seleccionó un número de tópicos de manera aleatoria. Para cada tópico seleccionado se realizó una ejecución de la infraestructura seleccionado el algoritmo evolutivo correspondiente al experimento, tomando como contexto temático para la infraestructura la descripción ODP correspondiente al tópico. Cabe aclarar que en todos los casos el índice utilizado contiene documentos correspondientes a todos los tópicos recolectados (es decir documentos pertenecientes a 448 tópicos).
La evaluación del algoritmo se desarrolló en dos etapas destinadas a responder las cuestiones de investigación. Primero, se utilizó el conjunto de entrenamiento para evolucionar consultas y controlar el rendimiento promedio poblacional en cada generación. En base a estos resultados, se evaluó si las consultas de las primeras poblaciones son superadas por las consultas de las poblaciones subsecuentes. En este sentido, el rendimiento del sistema durante la primera generación puede ser tomado como un punto de comparación básico ya que estas consultas son las que se generan directamente con palabras de la descripción del tópico y aún no hubo intervención del algoritmo evolutivo. Este análisis permite estudiar cuán efectivas son las consultas evolucionadas por el sistema con respecto a las consultas generadas directamente de la descripción.
En la siguiente etapa de la evaluación, se utilizaron las consultas de la última generación sobre el índice de prueba con el fin de determinar si las consultas evolucionadas para un tópico particular (sobre el índice de entrenamiento) son efectivas cuando se las utiliza sobre un nuevo corpus (el índice de prueba). En este punto es importante aclarar que el índice de entrenamiento y el índice de prueba contienen los mismos tópicos (y por lo tanto corresponden las mismas descripciones) pero distintos documentos (son dos conjuntos de documentos que no se superponen entre sí). Además, también se evaluó la efectividad de las consultas generadas directamente de la descripción y consultas generadas con las técnicas de refinamiento de consultas Bo1 y Bo1*(descriptos a continuación) sobre el conjunto de prueba.

Métodos Bo1 y Bo1*. Como se dijo anteriormente, *relevance feedback* es un mecanismo de refinamiento de consultas utilizado para ajustar consultas en base a la relevancia

de los resultados obtenidos mediante la consulta. El algoritmo para relevance feedback más conocido fue propuesto por Rocchio ([Roc71]). Dado un vector inicial representando una consulta $\overrightarrow{q}$, una consulta modificada $\overrightarrow{q_m}$ se calcula de la siguiente manera:

$$\overrightarrow{q_m} = \alpha.\overrightarrow{q} + \beta.\sum_{\overrightarrow{d_j} \in D_r} \overrightarrow{d_j} - \gamma.\sum_{\overrightarrow{d_j} \in D_n} \overrightarrow{d_j}, \tag{6.19}$$

donde D_r y D_n son los conjuntos de documentos relevantes y no relevantes respectivamente, y α, β y γ son parámetros de ajuste. Una de las estrategias consiste en asignarle a α y a β un valor mayor a 0 mientras que a γ se le asigna 0, lo cuál hace que sólo se contemple el conjunto de documentos relevantes. Cuando no se dispone de un grado de relevancia por parte del usuario, se inicializa el conjunto D_r con los k primeros documentos y el conjunto D_n se inicializa vacío. Esto conduce a aplicar un feedback ciego.
El mecanismo de divergencia de aleatoriedad con estadísticas de Bose-Einstein (Bo1) [Ama03] es una generalización exitosa del método de Rocchio. Para aplicar este modelo primero se deben asignar pesos a los términos de acuerdo con su nivel de información, estimada por la divergencia entre la distribución de los términos en los primeros documentos y una distribución aleatoria de la siguiente forma:

$$w(t) = tf_x.log_2\frac{1+P_n}{P_n} + log_2(1+P_n) \tag{6.20}$$

donde tf_x es la frecuencia de los términos en los primeros documentos y P_n es la proporción de documentos de la colección que contiene a t. Una vez calculada esta medida, se expande la consulta mezclando los términos más informativos con los términos de la consulta original.
La efectividad del método de refinamiento Bo1 tiene relación con la calidad de los primeros documentos obtenidos en el primer paso de recuperación. Si se cuenta con un mecanismo de relevance feedback, es posible implementar una versión surpervisada del método Bo1, al cual referiremos como Bo1*. Este método es igual al Bo1 excepto que para realizar la asignación de pesos a los términos en lugar de considerar simplemente los primeros documentos, se consideran los primeros documentos que se sabe que son relevantes. Una vez que las consultas han sido refinadas aplicando el método Bo1* sobre el conjunto de entrenamiento, se las puede utilizar sobre un nuevo conjunto de documentos. Bajo estas condiciones, el método Bo1* puede ser considerado como una versión supervisada del método Bo1 que, al igual que los algoritmos evolutivos propuestos aquí, utiliza información

referida a la relevancia de los documentos.

Experimentos y análisis de resultados: rendimiento sobre el conjunto de entrenamiento

Dado que la representación de los individuos se mantuvo como una lista de términos, el objetivo del primer experimento consistió en analizar si es mejor unir estos términos por medio del operador lógico OR (consultas OR) o por medio del operador lógico AND (consultas AND). Para esto se realizaron ejecuciones de los cuatro algoritmos evolutivos analizados (mono-objetivo con *Precisión@10*, mono-objetivo con *Cobertura*, NSGA-II y MOEA agregativo) sobre 10 tópicos seleccionados de forma aleatoria. El segundo análisis consiste en un estudio más riguroso evaluando los algoritmos más prometedores sobre 110 tópicos seleccionados de manera aleatoria.

Algoritmos evolutivos mono-objetivo. Como primer experimento, se ejecutó un EA mono-objetivo con cada tipo de operador lógico durante 300 generaciones, con el objetivo de maximizar *Precisión@10*. Para ello se utilizó el índice de entrenamiento de 2/3 del corpus. La figura 6.15 muestra la evolución de *Precisión@10* a la izquierda (el objetivo siendo maximizado) y de *Cobertura* a la derecha para el tópico *BUSINESS/BUSINESS_SERVICES/CONSULTING* (Consulting) de ODP.

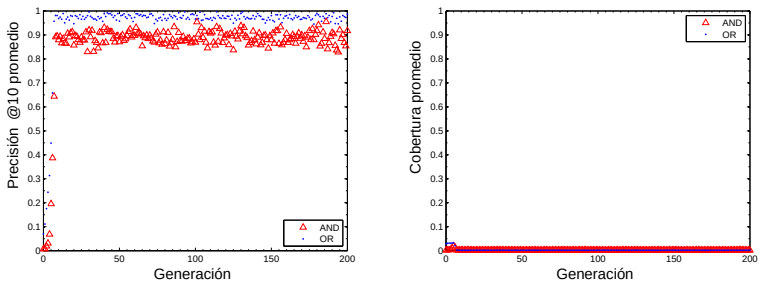

Figura 6.15: Evolución de *Precisión@10* (izquierda) y *Cobertura* (derecha) para el tópico (Consulting) cuando el objetivo maximizado es *Precisión@10*.

El gráfico muestra el rendimiento promedio poblacional de las consultas para cada generación. Como se puede observar en el gráfico de la izquierda, al maximizar *Precisión@10*, luego de un pequeño número de generaciones se pueden obtener ambos tipos de consultas (unidas por AND y unidas por OR) con un valor de *Precisión@10* cercano al óptimo. Sin embargo, esto se logra a costa de muy malos resultados en los valores de *Cobertura*. Los

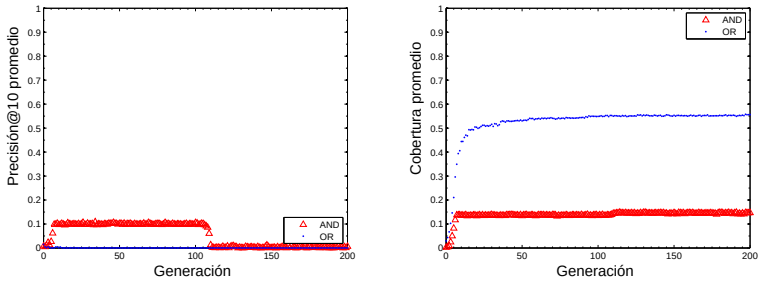

Figura 6.16: Evolución de *Precisión@10* (izquierda) y *Cobertura* (derecha) para el tópico (CONSULTING) cuando el objetivo maximizado es *Cobertura*.

cuales pueden observarse en el gráfico de la derecha.
Como era de esperar, cuando el objetivo que se maximiza es *Cobertura*, se logran bajos valores de *Precisión@10* (figura 6.16). Además, se puede notar que si bien se alcanzan muy buenos valores de *Cobertura* para las consultas OR, no ocurre lo mismo con las consultas AND. Si bien los resultados se muestran para un solo tópico, el análisis de los demás tópicos ha mostrado un comportamiento similar.

NSGA-II. El siguiente experimento consistió en realizar la ejecución del algoritmo NSGA-II implementado durante 300 generaciones, utilizando el conjunto de entrenamiento de 2/3 del corpus, con el fin de lograr consultas que alcancen altos valores tanto en *Precisión@10* como en *Cobertura*. En la figura 6.17 se ve el gráfico del rendimiento promedio alcanzado por ambos tipos de consultas (unidas por OR y unidas por AND) en cada generación para el tópico CONSULTING, examinando *Precisión@10* (izquierda), *Cobertura* (centro) y $F^\star$ (derecha). Es interesante notar que, cuando se evolucionaron consultas cuyos términos se unen con el operador lógico OR, esta estrategia logró alcanzar muy buenos niveles de *Precisión@10* sin comprometer los valores de *Cobertura*. Como resultado, la medida $F^\star$ refleja un muy buen rendimiento. Por otro lado en el caso de las consultas AND, si bien los valores alcanzados en *Precisión@10* son buenos, los valores de *Cobertura* permanecen bajos. La tendencia observada en el tópico CONSULTING también se observó en el resto de los tópicos analizados. La tabla 6.4 muestra los promedios obtenidos en *Precisión@10*, *Cobertura* y $F^\star$ por ambos tipos de consultas para los 10 tópicos analizados, para la primera y la última generación. Como se puede observar, el algoritmo NSGA-II evolucionando consultas cuyos términos se encuentran unidos por el operador lógico OR alcanza muy buenos resultados en ambos objetivos. Además, la comparación entre la primera y la última generación muestra que luego de la evolución hay

una mejora importante.

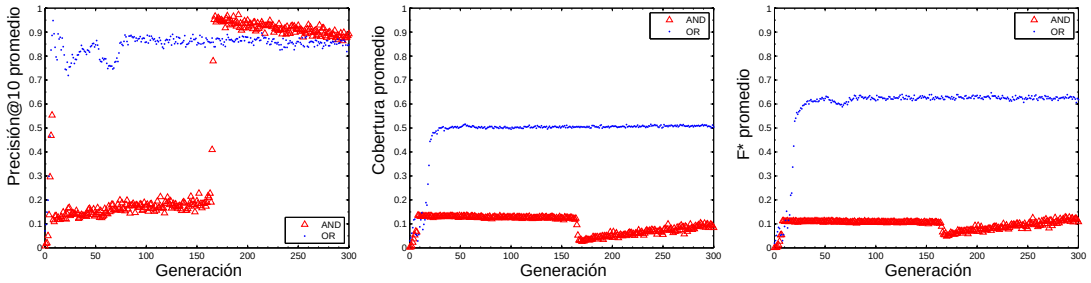

Figura 6.17: Evolución de *Precisión@10* (izquierda), *Cobertura* (centro) and $F^{\star}$ (derecha) para el tópico CONSULTING al ejecutar el algoritmo NSGA-II.

NSGA-II: consultas AND			
	Precisión@10 promedio	*Cobertura* promedio	$F^{\star}$ promedio
Primera generación	0.038	0.059	0.020
Última generación	0.689	0.196	0.176

NSGA-II: consultas OR			
	Precisión@10 promedio	*Cobertura* promedio	$F^{\star}$ promedio
Primera generación	0.055	0.049	0.022
Última generación	0.953	0.653	0.766

Tabla 6.4: Primera generación vs. última generación de consultas AND y OR evolucionadas con NSGA-II: *Precisión@10* promedio, *Cobertura* promedio y $F^{\star}$ promedio sobre 10 tópicos.

Precisión@10 promedio				
	N	media	95% C.I.	mejora
Primera generación	110	0.0611	[0.0568,0.0654]	
Última generación	110	0.9204	[0.8992,0.9415]	1407%

Cobertura promedio				
	N	media	95% C.I.	mejora
Primera generación	110	0.0459	[0.0429,0.0488]	
Última generación	110	0.5981	[0.5680,0.6283]	1204%

$F^{\star}$ promedio				
	N	media	95% C.I.	mejora
Primera generación	110	0.0219	[0.0205,0.0234]	
Última generación	110	0.7119	[0.6859,0.7378]	3144%

Tabla 6.5: Primera generación vs. última generación de consultas evolucionadas con NSGA-II: promedio, intervalo de confianza y mejora de la calidad de las consultas sobre 110 tópicos.

Con el fin de realizar una evaluación más rigurosa de este mismo algoritmo, utilizando OR como operador lógico, se realizó la ejecución sobre 110 de los 448 tópicos utilizando el índice de prueba. La tabla 6.5 muestra las medias y los intervalos de confianza para *Precisión@10*, *Cobertura* y $F^{\star}$ con respecto a los 110 tópicos para la primera y última generación. Además, se reporta la mejora alcanzada por el NSGA-II cuando se comparan las consultas evolucionadas con respecto a las de la primera generación. Esta comparación muestra que a través de las sucesivas generaciones el NSGA-II fue capaz de obtener consultas con una calidad considerablemente superior a las consultas generadas directamente desde la descripción del tópico.

MOEA agregativo. Para completar la evaluación con respecto al conjunto de entrenamiento, se monitoreó el rendimiento del MOEA agregativo para ambos tipos de consultas analizando los valores promedio de *Precisión@10*, *Cobertura* y $F^\star$ alcanzados durante 300 generaciones y utilizando el conjunto de entrenamiento de 2/3 del corpus.

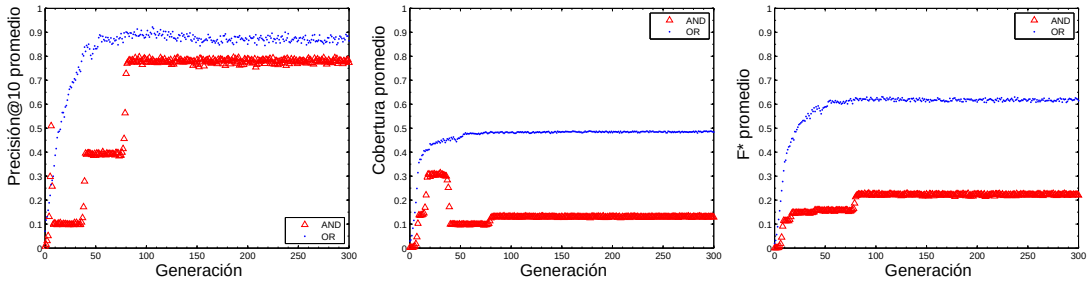

Figura 6.18: Evolución de *Precisión@10* (izquierda), *Cobertura* (centro) and $F^\star$ (derecha) para el tópico CONSULTING al ejecutar el algoritmo MOEA agregativo.

MOEA agregativo: consultas AND			
	Precisión@10 promedio	*Cobertura* promedio	$F^\star$ promedio
Primera generación	0.042	0.060	0.022
Última generación	0.570	0.358	0.372

MOEA agregativo: consultas OR			
	Precisión@10 promedio	*Cobertura* promedio	$F^\star$ promedio
Primera generación	0.054	0.049	0.022
Última generación	0.948	0.635	0.749

Tabla 6.6: Primera generación vs. última generación de consultas AND y OR evolucionadas con el MOEA agregativo: *Precisión@10* promedio, *Cobertura* promedio y $F^\star$ promedio sobre 10 tópicos.

Precisión@10 promedio				
	N	media	95% C.I.	mejora
Primera generación	110	0.0609	[0.0566,0.0652]	
Última generación	110	0.9099	[0.8833,0.9365]	1394%

Cobertura promedio				
	N	media	95% C.I.	mejora
Primera generación	110	0.0459	[0.0429,0.0488]	
Última generación	110	0.5641	[0.5331,0.5951]	1130%

$F^\star$ promedio				
	N	media	95% C.I.	mejora
Primera generación	110	0.0219	[0.0205,0.0233]	
Última generación	110	0.6804	[0.6525,0.7084]	3005%

Tabla 6.7: Primera generación vs. última generación de consultas evolucionadas con el MOEA agregativo: promedio, intervalo de confianza y mejora de la calidad de las consultas sobre 110 tópicos.

Primeramente, se ejecutó el MOEA agregativo sobre los 10 tópicos seleccionados previamente, tanto para las consultas AND como para las consultas OR. Los gráficos de la

figura 6.18 muestran el rendimiento promedio de ambos tipos de consultas (AND y OR) para cada generación para el tópico CONSULTING y la tabla 6.6 muestra los promedios obtenidos en *Precisión@10*, *Cobertura* y $F^{\star}$ sobre los 10 tópicos. Analizando la figura 6.18 y la tabla 6.6 se puede observar que el rendimiento del algoritmo multi-objetivo agregativo es similar al del NSGA-II, alcanzando muy buenos resultados en ambos objetivos en forma simultánea y logrando mejores resultados con las consultas cuyos términos están unidos por OR que en aquellas consultas en las que los términos están unidos por AND. Finalmente, se ejecutó el MOEA agregativo con consultas OR sobre los 110 tópicos mencionados previamente y se calculó el rendimiento promedio, el intervalo de confianza y la mejora sobre las tres métricas analizadas, los cuáles se muestran en la tabla 6.7. Nuevamente el rendimiento es similar al obtenido por el NSGA-II, mostrando diferencias estadísticamente significantes y una mejora sumamente importante.
Esto permite concluir preliminarmente que, para los objetivos planteados aquí (*Precisión@10* y *Cobertura* altos), los resultados de usar un esquema agregativo para ordenar las consultas en orden de importancia dentro de un ciclo evolutivo son comparables con los resultados obtenidos por un esquema no agregativo más costoso.

Experimentos y análisis de resultados: rendimiento sobre el conjunto de prueba

En el siguiente experimento, las consultas obtenidas por ambos algoritmos evolutivos multi-objetivo en cada tópico se utilizaron sobre el conjunto de prueba. Esto nos permite determinar si las consultas evolucionadas por medio de los MOEAs son efectivas cuando se las utiliza sobre un conjunto de documentos diferente al usado durante el entrenamiento. Para esto, como primera aproximación, calculamos los valores promedio de *Precisión@10*, *Cobertura* y $F^{\star}$ obtenidos sobre el conjunto de prueba de 1/3 por las consultas (AND y OR) evolucionadas para los 10 tópicos mencionados anteriormente (tabla 6.8). Esta primera comparación sobre el conjunto de prueba nos permite determinar, por un lado, que se sigue manteniendo el comportamiento observado con respecto a las consultas AND y las consultas OR, y por otro, que se sigue obteniendo una mejora significativa cuando se comparan las consultas evolucionadas vs. las consultas generadas directamente a partir del tópico de interés (Baseline). El siguiente experimento sobre el conjunto de prueba es el análogo al realizado para los 110 tópicos analizados previamente sobre el conjunto de entrenamiento. En esta ocasión, se utilizaron las consultas OR evolucionadas con ambos esquemas multi-objetivo para recuperar documentos del índice de testeo. El objetivo de

Consultas AND			
TESTING	*Precisión@10* promedio	*Cobertura* promedio	$F^\star$ promedio
Baseline	0.016	0.075	0.011
NSGA-II	0.409	0.193	0.156
MOEA agregativo	0.500	0.336	0.338

Consultas OR			
TESTING	*Precisión@10* promedio	*Cobertura* promedio	$F^\star$ promedio
Baseline	0.015	0.056	0.009
NSGA-II	0.572	0.637	0.577
MOEA agregativo	0.543	0.643	0.538

Tabla 6.8: Rendimiento de las consultas AND y OR para Baseline, NSGA-II y MOEA agregativo: *Precisión@10* promedio, *Cobertura* promedio y $F^\star$ promedio sobre 10 tópicos.

este punto es realizar un análisis más riguroso del desempeño de los algoritmos sobre el conjunto de testeo de 1/3 y comparar las consultas evolucionadas con las generadas directamente a partir de la descripción y con consultas mejoradas por medio de los métodos Bo1 y Bo1*. Es importante notar que las consultas generadas a partir de la descripción del tópico pueden ser consideradas como un buen punto de comparación. Esto se debe a que dicha descripción es un instrumento confiable que consiste en un resumen realizado manualmente por editores (personas) familiarizados con el tópico. Por otro lado, Bo1 es considerado uno de los sucesores más exitosos del método de refinamiento de consultas de Rocchio [ACR04], mientras que Bo1* es un desafío todavía mayor debido a que utiliza información sobre la importancia de los resultados obtenidos por las consultas. Paralelamente, en estos experimentos se quiso analizar el efecto de utilizar distintos tamaños para el conjunto de entrenamiento. Por esta razón se realizaron ejecuciones de los algoritmos sobre dos índices de entrenamiento de tamaño diferente: 2/3 del tamaño total del corpus y 1/2 del tamaño total del corpus (manteniendo el conjunto de prueba de 1/3 reservado para los testeos).

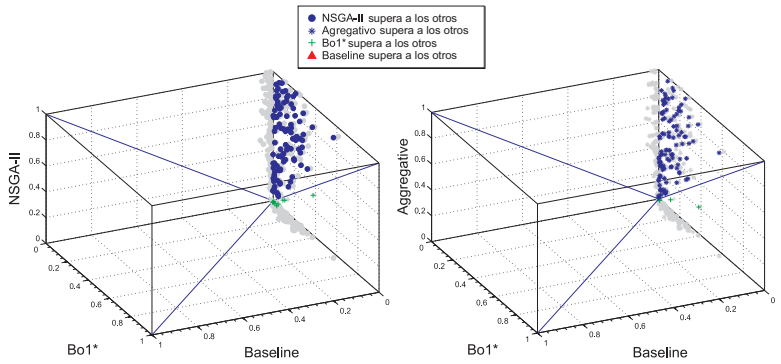

Figura 6.19: Comparación de baseline, Bo1* y NSGA-II (izquierda) y comparación de baseline, Bo1* y MOEA agregativo (derecha) para 110 tópicos con respecto a *Precisión@10*.

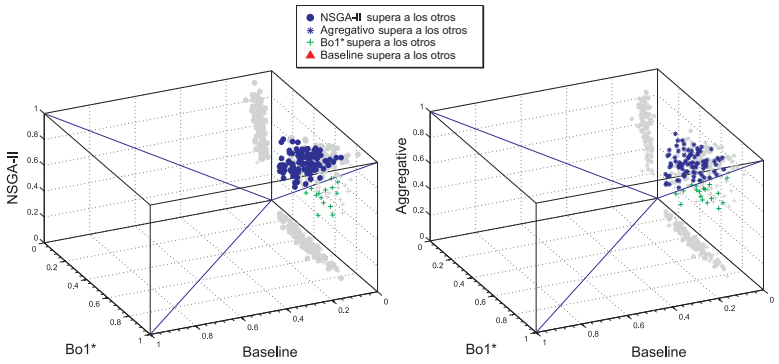

Figura 6.20: Comparación de baseline, Bo1⋆ y NSGA-II (izquierda) y comparación de baseline, Bo1⋆ y MOEA agregativo (derecha) para 110 tópicos con respecto a *Cobertura*.

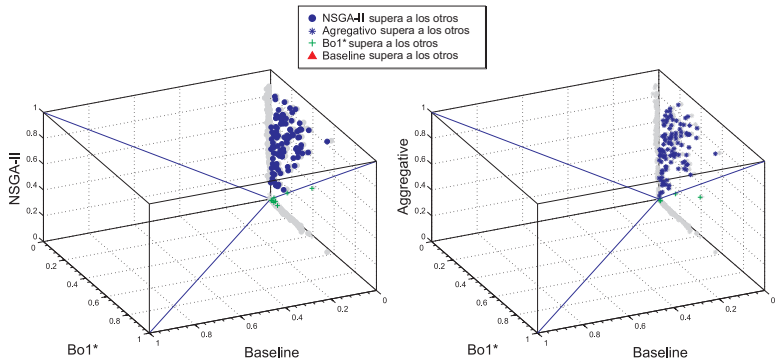

Figura 6.21: Comparación de baseline, Bo1⋆ y NSGA-II (izquierda) y comparación de baseline, Bo1⋆ y MOEA agregativo (derecha) para 110 tópicos con respecto a $F^{\star}$.

Los gráficos presentados en las figuras 6.19, 6.20 y 6.21 muestran el rendimiento promedio obtenido en el conjunto de prueba por las consultas evolucionadas para cada tópico. En estas tres figuras, cada punto representa un tópico y corresponde con una ejecución del algoritmo multi-objetivo analizado. La coordenada vertical (z) corresponde al rendimiento promedio del NSGA-II en el gráfico del lado izquierdo y al MOEA agregativo en el gráfico del lado derecho, mientras que los otros dos ejes de coordenadas (x e y) corresponden a los valores obtenidos por las consultas generadas directamente a partir de la descripción (Baseline) y por las consultas refinadas con el método Bo1⋆ respectivamente. Nótese que se utilizaron diferentes marcadores para mostrar cuál de los tres métodos supera a los otros dos. Además, se pueden observar las proyecciones de cada punto en los planos x-y, x-z e y-z. En el caso de *Precisión@10* (figura 6.19) se puede observar que el algoritmo NSGA-II supera a los métodos Baseline y Bo1⋆ para 100 de los 110 tópicos analizados, mientras que el esquema agregativo fue mejor en 105 de los tópicos. En el caso de la

métrica *Cobertura* (figura 6.20) se puede observar que el algoritmo NSGA-II supera a los métodos Baseline y Bo1* en 96 de los tópicos analizados, mientras que el esquema agregativo los supera en 91 de los tópicos. Para verificar esto gráficamente, se puede observar que para los planos x-z e y-z la mayoría de los puntos aparecen por encima de la diagonal, tanto para el algoritmo NSGA-II como para el esquema agregativo. Esto significa que el valor más alto es el obtenido por el método presentado en eje vertical (z). Finalmente, para la medida $F^\star$ (figura 6.21), el algoritmo NSGA-II supera a los métodos Baseline y Bo1* en 101 tópicos mientras que el esquema agregativo es superior en 105 de los 110 tópicos analizados.

Estos gráficos muestran que, para la mayoría de los casos, las consultas evolucionadas son más efectivas que las correspondientes a los métodos Baseline y Bo1*. Además, estos resultados muestran que las consultas evolucionadas no tienden a sobreajustar los datos de entrenamiento sino que también logran ser efectivas en un corpus totalmente diferente.

Precisión@10 promedio					
	TrS	N	media	I.C. 95%	mejora
Baseline	–	110	0.0153	[0.0131,0.0174]	–
Bo1	–	110	0.0157	[0.0134,0.0181]	3%
Bo1*	$\frac{1}{3}$	110	0.1149	[0.0983,0.1314]	651%
	$\frac{2}{3}$	110	0.1379	[0.1171,0.1587]	802%
NSGA-II	$\frac{1}{3}$	110	0.5530	[0.4970,0.6089]	3516%
	$\frac{2}{3}$	110	0.5359	[0.4791,0.5928]	3405%
MOEA Agregativo	$\frac{1}{3}$	110	0.5307	[0.4759,0.5855]	3370%
	$\frac{2}{3}$	110	0.4958	[0.4423,0.5493]	3142%
Cobertura promedio					
	TrS	N	media	I.C. 95%	mejora
Baseline	–	110	0.0512	[0.0480,0.0545]	–
Bo1	–	110	0.1283	[0.1203,0.1364]	150%
Bo1*	$\frac{1}{3}$	110	0.4295	[0.4011,0.4579]	738%
	$\frac{2}{3}$	110	0.4404	[0.4108,0.4700]	768%
NSGA-II	$\frac{1}{3}$	110	0.6007	[0.5723,0.6292]	1073%
	$\frac{2}{3}$	110	0.5860	[0.5582,0.6138]	1044%
MOEA Agregativo	$\frac{1}{3}$	110	0.5646	[0.5347,0.5945]	1002%
	$\frac{2}{3}$	110	0.5587	[0.5296,0.5877]	990%
$F^\star$ promedio					
	TrS	N	media	I.C. 95%	mejora
Baseline	–	110	0.0076	[0.0069,0.0085]	–
Bo1	–	110	0.0156	[0.0134,0.0178]	105%
Bo1*	$\frac{1}{3}$	110	0.1254	[0.1079,0.1430]	1549%
	$\frac{2}{3}$	110	0.1410	[0.1208,0.1613]	1753%
NSGA-II	$\frac{1}{3}$	110	0.5255	[0.4819,0.5690]	6805%
	$\frac{2}{3}$	110	0.5037	[0.4603,0.5471]	6519%
MOEA Agregativo	$\frac{1}{3}$	110	0.4978	[0.4553,0.5403]	6442%
	$\frac{2}{3}$	110	0.4771	[0.4360,0.5183]	6170%

Tabla 6.9: Consultas Baseline vs. consultas ajustadas con el método Bo1 y Bo1* vs. consultas evolucionadas con NSGA-II y MOEA agregativo: media, intervalos de confianza y mejora observada en la calidad promedio de las consultas sobre 110 tópicos con respecto al método Baseline. TrS refiere al tamaño del índice de entrenamiento utilizado en cada caso.

La tabla 6.9 presenta el rendimiento de las consultas obtenidas con los método Baseline, Bo1 y Bo1* y las consultas evolucionadas con el NSGA-II y el MOEA agregativo. Esta

comparación muestra que las consultas evolucionadas por el algoritmo NSGA-II y el algoritmo MOEA agregativo superan a las generadas con el método Baseline y a las consultas refinadas con las técnicas Bo1 y Bo1*. Además, las mejoras son estadísticamente significantes. Por otro lado, se puede concluir que los algoritmos NSGA-II y MOEA agregativo tienen similar rendimiento sobre el conjunto de prueba.

Como se dijo anteriormente, se realizaron experimentos con dos índices de entrenamiento de tamaño diferente. Hemos observado que cambiar el tamaño del índice de entrenamiento de 2/3 a 1/2 del corpus no afecta de manera significativa el rendimiento global de las consultas evolucionadas cuando se las utiliza sobre el conjunto de prueba. Este hecho puede deberse a que más allá de tener un tamaño significativamente menor, cada tópico preserva suficiente cantidad de información como para poder guiar la búsqueda de los MOEAs hacia buenas soluciones.

Finalmente, como característica particular de este problema de optimización, hemos notado que la distribución de los documentos parece tener cierta relación con el nivel de *Precisión@10* alcanzado. En nuestro corpus, los documentos no se encuentran distribuidos de manera uniforme entre los tópicos. Hemos notado que, en general, los tópicos más poblados permiten alcanzar mayor *Precisión@10* que los menos poblados. La intensidad de una relación lineal entre dos variables, por lo general, se mide mediante el coeficiente de correlación de Pearson (ρ), cuyos valores van desde -1, correspondiente a una correlación perfectamente negativa, hasta $+1$, correspondiente a una correlación perfectamente positiva y se calcula mediante la fórmula:

$$\rho = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2 \sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{6.21}$$

donde X_i e Y_i son los valores observados para ambas variables para la i-ésima muestra y $\bar{X}$ e $\bar{Y}$ son las medias de cada variable. El análisis de correlación se muestra en la tabla 6.10 y revela una correlación positiva entre el tamaño del tópico y la *Precisión@10* alcanzada por las consultas evolucionadas para cada tópico. Un análisis similar en las otras métricas utilizadas en estos experimentos no son informativas ya que, por definición, el tamaño del tópico (es decir la cantidad de documentos relevantes) afecta negativamente a la cobertura. De todas maneras, es importante destacar que aunque los tópicos más voluminosos tienden a conducir a altos valores de *Precisión@10*, algunos tópicos menos

numerosos también permitieron un muy buen desempeño. Esto deja dilucidar que la habilidad para aprender buenas consultas para un tópico depende de muchos factores, los cuales no son solamente cuantitativos (p. ej. el tamaño del tópico) sino también cualitativos (p. ej. el vocabulario del tópico).

	TrS	correlación	p-value
NSGA-II	$\frac{1}{2}$	0.3130	0.0009
	$\frac{2}{3}$	0.3011	0.0014
MOEA Agregativo	$\frac{1}{2}$	0.3557	0.0001
	$\frac{2}{3}$	0.2449	0.0099

Tabla 6.10: Correlación de Pearson entre tópico, tamaño y *Precisión@10*. TrS refiere al tamaño del índice de entrenamiento.

Discusión

Eficiencia. En un análisis de eficiencia del método deberíamos tener en cuenta: (1) el tiempo necesario para enviar una consulta y recuperar los resultados, (2) el tiempo necesario para el cómputo de las funciones de aptitud asociadas a cada consulta y (3) el tiempo necesario para comparar a los individuos y seleccionar a los más promisorios. Otros aspectos, como el tiempo necesario para realizar la recombinación o la mutación no tienen impacto significante. El costo de tiempo de enviar una consulta y recuperar sus resultados dependerá de si la búsqueda se hace a un corpus local o remoto. Para el estudio reportado en la versión multi-objetivo del algoritmo, se utilizaron conjuntos de documentos indexados localmente y, por lo tanto, el tiempo de acceso fue considerablemente menor que en la versión mono-objetivo, la cual accede a la web. Bajo las nuevas condiciones (un índice local) la arquitectura puede acceder a cada documento en una constante de tiempo determinada.

Las funciones de aptitud implementadas pueden calcularse identificando la porción de los diez primeros documentos que corresponde a documentos relevantes (para precisión a diez) y la porción de los documentos relevantes que han sido recuperados (para cobertura). Para estos cálculos es suficiente con tener acceso a información referida a la relevancia de cada documento y de cada tópico y no es necesario realizar el análisis del contenido de cada documento. Otras funciones de aptitud, como las basadas en similitud pueden resultar más intensas computacionalmente.

Finalmente, la complejidad de la estrategia basada en Pareto para identificar a los mejores

individuos en cada generación es $O(m * n^2)$, donde m es el número de objetivos (en estos estudios es $m = 2$) y n es el tamaño de la población [Deb01], un tiempo que puede ser significativamente reducido (a $O(m * n * log(n))$) si se aplican algoritmos más eficientes para el ordenamiento de los conjuntos de no-dominación [Jen03]. Por otra parte, la complejidad de la estrategia agregativa en cada generación es $O(n)$.
Si bien los EAs se ejecutaron durante 300 generaciones, se puede observar que, usualmente, 30 generaciones son tiempo suficiente para identificar individuos con buen rendimiento. Cada generación involucra la emisión de tantas consultas como individuos hay en la población y esto se combina con el tiempo de recuperación y el tiempo de análisis de resultados. Como consecuencia, el costo computacional del proceso completo es alto y, por lo tanto, es importante notar que el esquema propuesto no pretende brindar un mecanismo de búsqueda en tiempo real.

Favoreciendo la exploración con el reservorio de mutación. Un aspecto importante de la metodología propuesta es el reservorio de mutación que contiene nuevas posibles palabras recolectadas a lo largo de las generaciones. Este componente de la infraestructura ha demostrado sus beneficios en la versión mono-objetivo. En esta segunda versión, se analizaron nuevas palabras incorporadas por el proceso de búsqueda y se encontró que muchos de ellos son muy buenos descriptores para el tópico bajo estudio. Por ejemplo, el tópico *SHOPPING/PETS/BIRDS* contiene los siguientes términos en su descripción: "*birds, canaries, considered, doves, finches, include, kept, parrot, pets, pigeons, poultry, softbills, species, wild*". Uno de los mejores individuos evolucionados por el algoritmo NSGA-II para este tópico fue la consulta **q**=*"administr, aviari, belli,* ***birds****, cabela,* ***canaries****, chat, cleanest, cockatoo, harrison, hitag2, keepac, lamp, purpl, realtim, rib, tasti, toi, twinleath "*, con una *Precisión@10* = 1 y un *Cobertura* = 0.8968. Se puede notar que excepto por **birds** y **canaries** los términos no se encuentran en la descripción inicial del tópico. Además, una búsqueda breve en la web revela que la mayoría de ellos tienen una relación directa con el tópico (p. ej. cockatoo refiere a la especie de loro mascota *cacatua*, una de las más populares en este tipo de mascota). El hecho de encontrar nuevas palabras no presentes en la descripción de tópico en una consulta de tan buena calidad resalta la importancia de la exploración de vocabulario nuevo por medio del uso del reservorio de mutación.

Enfoque Agregativo vs. multi-objetivo. Hemos visto que el enfoque agregativo es computacionalmente menos intenso que el no agregativo. Además, distinto a lo que ocurre con los métodos de optimización basados en Pareto, las técnicas agregativas permiten definir de forma relativamente simple funciones de puntuación que favorezcan a un objetivo con respecto al otro. Por ejemplo, una generalización de la métrica F_1 puede ser definida como [Rij79]:

$$F_\beta = \frac{(1+\beta^2)\cdot(\mathit{Precisión}\cdot\mathit{Cobertura})}{(\beta^2\cdot\mathit{Precisión}+\mathit{Cobertura})} \tag{6.22}$$

Esta medida permite dar β veces más importancia tanto a la precisión como a la cobertura. Por otro lado, se sabe que la mayoría de las técnicas agregativas no son capaces de generar porciones cóncavas del frente de Pareto [CLV07] y, por lo tanto, no serían aplicables en ciertos escenarios de optimización, en cuyo caso las técnicas basadas en Pareto serán preferidas.

Capítulo 7

Conclusiones

7.1 Revisión

Debido al gran crecimiento de las bases de datos y al gran número de documentos de texto dispersos alrededor del mundo, la tarea de recuperación de información se ha vuelto una de las más importantes cuestiones de investigación en informática, dando origen a campos bien consolidados como la minería de datos y texto. La minería en general, puede verse (a grandes rasgos) como el proceso de *establecer el problema* y formular hipótesis, *recolectar datos o texto*, realizar el *preprocesamiento* necesario, llevar a cabo el proceso de *minería* propiamente dicho y *analizar e interpretar* el modelo obtenido previo a llegar a la etapa de toma de decisiones. El objetivo general de esta tesis fue el estudio, desarrollo y evaluación de herramientas evolutivas durante la etapa de *minería*, tanto para datos como para texto. Más específicamente, se plantearon distintos algoritmos evolutivos para una de las ramas generales de investigación dentro de minería de datos, el problema de *selección de características*, y para una de las tareas principales necesarias en minería de texto, el problema de *recuperación temática*.

Se ha demostrado que el problema de Selección de Descriptores (FS por sus siglas en Inglés de Feature Selection) es NP-completo [DR94], por lo que, el mecanismo computacional que se utilice debe seguir una alternativa heurística para lograr encontrar un subconjunto de variables razonablemente bueno en un período de tiempo aceptable. Nuestra propuesta consistió en incorporar una etapa evolutiva como primera fase de refinamiento dentro de una arquitectura más compleja. Esta arquitectura posee una segunda etapa, basada en el uso de redes neuronales, que mejora los resultados brindados por la primera. A través de este proceso de dos etapas se logra alcanzar resultados de alta calidad con un algoritmo de complejidad razonable. La tarea de selección de características presenta múltiples instan-

cias de aplicación en problemas reales complejos como: *análisis de niveles de expresión de genes* (usando microarrays), *extracción de genes para diagnóstico de cáncer* por medio de eliminación recursiva de características (RFE por sus siglas en inglés de Recursive Feature Elimination), *análisis de relación cualitativa y cuantitativa de propiedades para diseño de drogas*, *filtrado de texto* o *reconocimiento de caras*.

Por otro lado, el proceso de recuperación temática puede llevarse a cabo mediante dos pasos básicos. Primero, formulando consultas relevantes con respecto a un contexto temático. Segundo, presentando dichas consultas a un motor de búsqueda para recuperar documentos relevantes. En base a estos dos pasos, podemos notar que la calidad del material recuperado será sumamente dependiente de las consultas que se presentan, con lo cual, la generación inteligente de conjuntos de palabras se ha convertido en un problema de investigación importante para el área de minería de texto. En esta tesis, se propuso el uso de una arquitectura que incorpore un ciclo evolutivo que sea capaz de generar consultas de manera inteligente, con el fin de lograr la recuperación de documentos relevantes con respecto a un tópico de interés. La infraestructura está compuesta por una *representación interna del tópico* de interés, un mecanismo de *generación de consultas iniciales*, un *ciclo evolutivo* (conformado por una población de consultas, los operadores genéticos, el proceso de selección y el módulo para evaluación del fitness), un *reservorio de vocabulario nuevo*, la herramienta para extracción de términos de dicho reservorio, y un *motor de búsqueda* como medio de comunicación entre el sistema y la colección de documentos.

La versatilidad de la propuesta permite que sea útil en importantes campos de aplicación, por ejemplo: *búsqueda basada en la tarea del usuario*, *sistemas de recuperación para portales web*, *acceso a la web oculta*, *recopilación de información persistente* y *soporte en gestión del conocimiento*.

7.2 Principales contribuciones

Algoritmos desarrollados en selección de características. La selección de características presenta dos objetivos básicos: el de determinar cuál es la menor cardinalidad posible para el subconjunto de características seleccionadas, y el de determinar cuáles son las características más relevantes para ilustrar cierta relación.

7.2.1 Aproximación mono-objetivo para selección de características

Como primer paso dentro de este problema, se desarrolló un algoritmo evolutivo que asume como conocido el número de características y debe determinar cuáles son las más relevantes. El algoritmo se incorporó a la arquitectura de dos fases, formando parte del método de wrapper encargado de la búsqueda masiva de descriptores. Para evaluar a los individuos se emplearon varias técnicas de predicción: la primera basada en árboles de decisión, la segunda en regresión lineal múltiple y la tercera en regresión no lineal. La combinación de distintas técnicas de aprendizaje automatizado con frecuencia ha superado al uso de técnicas en forma individual y este es un ejemplo claro. Por ejemplo, el uso de árboles de decisión permitió una evaluación veloz de los individuos. Además, algunas de las distintas técnicas de predicción utilizadas permitieron capturar la no linealidad presente en los datos, como ocurrió con el caso del conjunto de datos correspondiente a hidrofobicidad. Dado que la arquitectura de dos fases propuesta realiza una búsqueda inicial veloz durante la primera etapa (reduciendo el número de descriptores), la segunda etapa, que es la encargada de mejorar los resultados obtenidos por el algoritmo evolutivo embebido en el wrapper, puede estar compuesta por un método más lento.
El algoritmo evolutivo incorporado al esquema de dos fases se evaluó sobre el caso de estudio de la propiedad *hidrofobicidad*, la cual es una de las propiedades fisicoquímicas más extensamente estudiadas, debido a la dificultad de determinar su valor experimentalmente y también porque está relacionada con la estimación de propiedades ADMET (Absorción, Distribución, Metabolismo, Excreción y Toxicidad). En este caso de estudio complejo la arquitectura logró la obtención de subconjuntos de descriptores de relevancia establecida por la literatura. Por ejemplo, mediante un análisis de la frecuencia de selección para los descriptores más seleccionados en todas las corridas realizadas, se pudo ver que el *número de átomos donantes o contribuyentes para los enlaces H* (nHDon), el *factor hidrófilo* (Hy) y la suma *punto de carga + hibridación* (D_S), fueron los descriptores más frecuentemente seleccionados y están considerados como descriptores de razonable influencia para la hidrofobicidad. Cabe notar, que para la evaluación de la arquitectura se tuvieron que realizar distintos experimentos con distintos tamaños de subconjuntos de descriptores seleccionados. Esta tarea fue necesaria para poder aproximar una cardinalidad razonable para los subconjuntos. Este análisis no fue absolutamente prohibitivo en cuanto a tiempo; sin embargo, no se probó de manera exhaustiva con todos los tamaños posibles e implicó que el número de experimentos se triplicara.

7.2.2 Aproximación multi-objetivo para selección de características

El segundo paso de investigación, fue el desarrollo de un esquema multi-objetivo a ser incorporado dentro del wrapper. Este algoritmo, a diferencia del primero incorpora el número de descriptores seleccionados como un segundo objetivo a minimizar. El método de evaluación del error predictivo de los subconjuntos de descriptores contempló varias técnicas de forma similar a la que se planteó para el EA mono-objetivo. Para llevar adelante la optimización simultánea del error y de la cardinalidad se utilizaron técnicas basadas en Pareto de eficacia y eficiencia conocidas y se propuso una estrategia agregativa que combina los dos objetivos. Se propusieron un total de 12 diferentes métodos de wrapper, los cuales resultaron de la combinación de tres técnicas de evolución multi-objetivo con cuatro métodos de evaluación del error predictivo. Nuevamente los resultados obtenidos por la primera fase, fueron utilizados por la segunda fase de la arquitectura para mejorar su capacidad predictiva.
La arquitectura se evaluó con tres conjuntos de datos reales distintos, dos que pueden ser modelados por una relación más bien lineal y con pocos descriptores, y el de hidrofobicidad que requiere un modelo mucho más complejo y que implica además una cardinalidad mayor en los subconjuntos de descriptores seleccionados. Los resultados obtenidos para todos los conjuntos de datos, fueron comparables y en algunos casos mejores a los reportados en la literatura. Además, se realizó un análisis comparativo entre las distintas estrategias multi-objetivo desarrolladas. Cuando se analizaron los resultados con los primeros dos conjuntos de datos, descubrimos que el método de evaluación del error de predicción basado en regresión lineal fue el que obtuvo mejores resultados, en tanto que las estrategias basadas en Pareto fueron mejores que la agregativa. Mientras que, para el conjunto de datos de hidrofobicidad, la estrategia agregativa fue mejor, sin importar cuál método de evaluación de error de predicción se utilice. Cuando el conjunto de datos no condujera a una relación sumamente compleja que no incorpore demasiados descriptores se observó que los tres esquemas evolutivos multi-objetivo alcanzan resultados exitosos, aunque los Pareto parecen realizar una mejor optimización de la cardinalidad. Por esta razón, en estos casos, el impacto más fuerte vendrá del método de evaluación de predicción utilizado. Sin embargo, cuando la relación fuera más compleja y exigiera mayor cardinalidad de los subconjuntos de descriptores seleccionados, la estrategia multi-objetivo evolutiva empleada es más influyente, siendo la estrategia agregativa la que obtuvo mejores resultados.

La arquitectura multi-objetivo propuesta para selección de características, tiende a encontrar subconjuntos de descriptores seleccionados con cardinalidad mínima y buena capacidad predictiva. Esto favorece la interpretación humana de los resultados y disminuye el número de hipótesis de aprendizaje asociadas a un subconjunto de descriptores. Dado que la arquitectura fue capaz de obtener resultados exitosos con los tres conjuntos de datos, y en particular con el de hidrofobicidad (cuya relación entre la propiedad y los descriptores se sabe que es no lineal) podemos concluir que constituye una metodología prometedora más allá de la linealidad o no presente en las relaciones entre los datos y la variable que se intenta modelar, y que su uso es conveniente para realizar selección de características en problemas complejos que involucren un gran número de características. Además puede pensarse como un tipo de método de complejidad moderada, compuesto por un mecanismo liviano de preselección de subconjuntos y seguido de un método más fuerte para evaluar la capacidad predictiva de los subconjuntos. Por último, cabe destacar que la generalidad de esta novedosa arquitectura va más allá del problema de selección en QSAR y QSPR y que puede ser pensado como un método general para selección de características en la que los EAs han demostrado su capacidad de soporte. Finalmente, el trabajo de investigación desarrollado brinda un marco de comparación y constituye una guía de los pasos básicos a seguir para evaluar cualquier proceso de selección de características que opere de forma similar.

Algoritmos desarrollados en búsqueda temática. La recuperación temática puede resolverse mediante dos pasos: la generación de consultas de alta calidad y la presentación de dichas consultas a un motor de búsqueda a fin de recuperar documentos relevantes. Sin embargo, la generación de consultas de alta calidad no es una tarea trivial. En esta tesis se propuso, desarrolló y evaluó una arquitectura que incorpora, entre sus elementos principales, un ciclo evolutivo destinado a la evolución de consultas para la recuperación de material relevante. El material recuperado por estas consultas automáticamente mejoradas puede ayudar en importantes tareas dentro de IR, por ejemplo, acercando información de interés a los usuarios (sin exigir su participación), brindando nuevo material a sistemas de gestión de conocimiento, recolectando conjuntos de documentos relacionados para portales temáticos o recuperando información de la Web Invisible.

7.2.3 Aproximación mono-objetivo para recuperación de información temática

El primer diseño contempló la utilización de un algoritmo evolutivo mono-objetivo el cual nos permitió analizar la aplicabilidad de la metodología en este problema de gran interés en IR, definir métricas de evaluación de desempeño posibles, estudiar el impacto de aplicar elitismo y distintas tasas de mutación, y proponer criterios de evaluación razonable para la arquitectura a fin de evaluar los resultados obtenidos. La primera métrica de evaluación de desempeño utilizada fue la de *similitud por coseno*, la cual expresa la cercanía entre dos documentos. En base a esta medida, se definió la calidad de una consulta, la cual no recupera un único documento, sino un *conjunto de documentos*. Gracias a la utilización del reservorio de palabras aprendidas, la arquitectura es capaz de descubrir términos nuevos y relevantes que no aparecen en la descripción original del tópico, y que permiten alcanzar documentos relevantes. El rendimiento del EA se midió sobre los criterios de evaluación definidos. Como tópico de interés se capturaron descripciones de tópicos pertenecientes al directorio DMOZ (dmoz.org). En los experimentos realizados se tuvieron en cuenta varias corridas por tópico a fin de determinar la estabilidad de la arquitectura. Los resultados demostraron que el algoritmo evolutivo logra mejoras estadísticamente significativas con respecto a las generaciones iniciales (es decir, los intervalos de confianza de la primera generación no se solapan con los intervalos de confianza de la última). Esto significa que el EA es capaz de evolucionar consultas con una calidad considerablemente superior a la de las consultas generadas directamente a partir de la descripción de tópico. Además, cabe notar que el número de generaciones (20) y la cantidad de individuos (60) son valores pequeños comparado con problemas clásicos en los que se utilizan EAs.

La aplicación de elitismo y diferentes tasas de mutación demostró que, en este problema, las poblaciones se comportan de forma similar a otros problemas de optimización. Por lo tanto, los valores clásicos para los parámetros del EA parecen ser adecuados también en este problema. Por otra parte, demostró que si fuera necesario intensificar la explotación o exploración del espacio de documentos, podemos variar los porcentajes de cruzamiento y mutación respectivamente.

Además se propuso una nueva métrica de similitud, σ^N, con la cual se pudo enfatizar la importancia de la recuperación de material similar y novedoso. Al mismo tiempo, esta métrica permite asegurar que la mejora en la calidad de las consultas a través de las generaciones no se debe solamente al hecho de que estemos tomando cada vez más

términos de la descripción del tópico de interés. Al igual que con la primera métrica, se realizaron múltiples corridas para cada tópico y los resultados demostraron que la mejora de los documentos recuperados por las consultas de la última generación son estadísticamente significativas respecto de los recuperados por las consultas de la primera generación. Además, entre los nuevos términos introducidos por el proceso de búsqueda, descubrimos muchos que resultan ser buenos descriptores para el tópico de interés, aún cuando estos no formaran parte de la descripción inicial del tópico.
Para poder evaluar el rendimiento del EA a lo largo de las generaciones se adoptó un criterio de evaluación basado en la calidad de las consultas de toda la población, el cual nos permite asignar un valor de rendimiento a cada generación. Una aproximación evolutiva para recuperación de información temática es exitosa si la calidad del material recuperado por las consultas de la última generación supera estadísticamente a la calidad del material recuperado por las consultas de la generación inicial. Esta afirmación fue cierta para todas las versiones implementadas.

7.2.4 Aproximación multi-objetivo para recuperación de información temática

Como paso siguiente dentro de la investigación en la rama de minería de texto, se diseñó, desarrolló y evaluó un ciclo evolutivo multi-objetivo para la infraestructura propuesta. Esta nueva instancia estuvo orientada a la recuperación de la mayor cantidad posible de material relacionado y la menor cantidad posible de material no relacionado. Para ello, la métrica de *similitud* utilizada en la primera versión no resulta la más adecuada. En su lugar, estos aspectos son clásicamente evaluados por medio de *precisión* y *cobertura*. Dado que las definiciones de precisión y cobertura exigen el conocimiento total del corpus de documentos, se propuso el entrenamiento del algoritmo sobre un corpus de documentos y el testeo de los resultados en un corpus diferente, asegurando de esta manera, la generalidad de las consultas evolucionadas. Para armar los índices se recolectaron tópicos del directorio DMOZ, cada uno de los cuales se dividió aleatoriamente de tal forma que una parte de sus páginas fueron destinadas a entrenamiento y la restante se reservó para el testeo. Las consultas evolucionadas sobre el corpus de entrenamiento, se utilizaron luego sobre el corpus de testeo y los resultados demostraron una mejora notable cuando se las comparó con los demás métodos y con las consultas generadas a partir de la descripción.
Para analizar las ventajas de la aplicación del EA multi-objetivo, se realizaron estudios comparativos evaluando el comportamiento del sistema cuando se persiguen los objetivos

en forma separada (esquemas mono-objetivo) y el comportamiento del sistema cuando ambos objetivos son tenidos en cuenta de manera integral. En este análisis se tuvieron en cuenta, una versión del algoritmo NSGA-II basado en técnicas de Pareto y una estrategia totalmente nueva basada en técnicas agregativas, la cual combinó y adaptó métricas ampliamente conocidas en IR.
Tanto el enfoque basado en técnicas Pareto, como el enfoque agregativo demostraron gran superioridad con respecto a los enfoques que intentaron optimizar los objetivos en forma individual. Además, los resultados obtenidos por medio de los enfoques multi-objetivo propuestos se compararon con una generalización conocida del método de Rochio (Bo1) y con una versión supervisada del mismo (Bo1*), demostrando ser notablemente mejores estadísticamente. Por otra parte, al igual que en el caso del esquema mono-objetivo, se pudieron observar mejoras significativas entre las consultas evolucionadas de la última generación y las consultas iniciales.
En base a los resultados obtenidos, podemos concluir que la infraestructura propuesta resulta sumamente útil para este tipo de problema de minería de texto. En particular, la versión multi-objetivo aborda de manera novedosa dos métricas sumamente difíciles de conciliar.

7.3 Trabajo futuro

Una de las tareas pendientes en estas investigaciones es la evaluación de ambas metodologías desde una visión más volcada al comportamiento evolutivo de las poblaciones. Este análisis podría incorporar la utilización de métricas como el indicador de hipervolumen, la distancia generacional, la cobertura entre poblaciones y el espaciado y la extensión del frente de individuos no dominados. Sin duda, otra tarea interesante es probar diferentes operadores de mutación, recombinación, selección y técnicas de nicho.
Por otro lado, con respecto a la evaluación de los individuos, esperamos llevar a cabo experimentos adicionales, en los cuales se planea utilizar nuevas funciones de aptitud basadas en otros métodos y métricas provenientes de las comunidades de aprendizaje automatizado y recuperación de información.
Con respecto a los algoritmos desarrollados para selección de características, una propuesta interesante es la aplicación de la metodología sobre otras propiedades fisicoquímicas e incluso en otras áreas de investigación (por ejemplo, para selección de genes). En el área de búsqueda temática, un trabajo prometedor consiste en la aplicación de

programación genética para lograr la evolución de consultas más complejas, las cuales podrían incorporar el uso de operadores booleanos dentro de su sintaxis. La combinación de esta metodología dentro de la arquitectura propuesta en esta tesis podría alcanzar resultados sumamente interesantes. Otras propuestas interesantes están relacionadas con la extensión o modificación de la arquitectura propuesta, por ejemplo: podría ser acoplada a sistemas de recomendación de información persistente, extenderse para aceptar sugerencias de cambio de contexto en forma dinámica, adaptarse o extenderse para ser utilizada para inferir co-ocurrencia entre términos, y colaborar en el desarrollo de herramientas para descubrir información semánticamente relacionada.

El trabajo más interesante a futuro resulta de considerar, en forma conjunta, los conocimientos y algoritmos desarrollados mediante ambas metodologías, y abordar el desarrollo de herramientas que sirvan de soporte para la reconstrucción de redes de asociación entre rutas biológicas. En particular las técnicas de minería de texto, pueden servir para el análisis automático de las grandes colecciones de documentos existentes en el área, a fin de descubrir asociaciones entre grupos de genes.

Índice de figuras

2.1 Representación del mapeo entre un espacio de decisión *3-dimensional* y un espacio objetivo *2-dimensional* llevado a cabo por $F(\mathbf{x})$. 20

2.2 Ejemplo de dominancia: $\mathbf{u} \preceq \mathbf{w}$ y $\mathbf{v} \preceq \mathbf{w}$, pero $\mathbf{u} \not\preceq \mathbf{v}$ y $\mathbf{v} \not\preceq \mathbf{u}$. 21

2.3 Ejemplo de frente de Pareto y conjunto óptimo de Pareto. Los vectores en el espacio de búsqueda cuya evaluación de la función objetivo dan como resultado un vector en el frente de Pareto constituyen el conjunto óptimo de Pareto. Ambos conjuntos se encuentran representados por cruces en cada espacio. 23

3.1 Algunas analogías entre el concepto natural de evolución y el esquema artificial en el que se basan los EAs. 28

4.1 Esquema agregativo $w_1 f_1 + w_2 f_2$ para un problema con dos objetivos . . . 39

4.2 Pasos básicos del mecanismo de selección del Vector Evaluated Genetic Algorithm (VEGA) 41

5.1 Ejemplo simple de un proceso de selección de características sobre una base de datos. 57

5.2 Pasos básicos identificados en la tarea de selección de características. El trazo discontinuo representa la posible incidencia del proceso de generación de subconjuntos en el criterio de detención. 58

5.3 Infraestructura de dos fases propuesta. 68

5.4 Estructura interna de la primera fase de la metodología. 70

5.5 Individuo que representa la selección de descriptores de la figura 5.1. 70

5.6 El 50% de los datos se reservó para entrenamiento, el 16% se utilizó para la evaluación de los subconjuntos y el 34% restante se dividió en dos conjuntos de testeo (S_1 y S_2). 75

5.7 Se realizaron 15 réplicas del EA con cada método. Para la mejor selección de obtenida con respecto a todas las réplicas se hicieron 7 réplicas de la red. Al test de ANOVA se ingresaron los valores de las 7 réplicas para cada método (7×5). . 78

5.8 Modificaciones en la primera fase de infraestructura de dos fases. 87

5.9 Relación *Número de descriptores* vs. *MSEP* para el conjunto de datos ***logBBB***. 92

5.10 Relación *Número de descriptores* vs. *MSEP* para el conjunto de datos ***logHIA***. . 93

5.11 Relación *Número de descriptores* vs. *MSEP* para el conjunto de datos ***logP***. . . 95

5.12 Prueba de comparación entre los métodos de regresión para las estrategias basadas en *Pareto* usando la prueba de Tukey-Kramer para $\alpha = 5\%$ para el conjunto ***logBBB***. 98

5.13 Prueba de comparación entre los métodos de regresión para la estrategia **agregativa** ($\alpha = 0.7$) usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logBBB***. 99

5.14 Comparación entre los tres algoritmos de búsqueda con respecto a MLR usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logBBB***. 99

5.15 Prueba de comparación entre los métodos de regresión con respecto a la estrategia ***agregativa*** usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logHIA***. 101

5.16 Prueba de comparación entre los métodos de regresión con respecto al algoritmo ***NSGA-II*** usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logHIA***. 101

5.17 Prueba de comparación entre los métodos de regresión con respecto al algoritmo ***SPEA2*** usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logHIA***. 101

5.18 Comparación entre los tres algoritmos de búsqueda con respecto a MLR usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logHIA***. 102

5.19 Prueba de comparación entre los métodos de regresión con respecto a la estrategia agregativa usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***. 103

5.20 Prueba de comparación entre los métodos de regresión con respecto al algoritmo NSGA-II usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***. 104

5.21 Prueba de comparación entre los métodos de regresión con respecto al algoritmo SPEA2 usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***. 104

5.22 Comparación entre los tres algoritmos de búsqueda con respecto a MLR usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***. 105

5.23 Comparación entre los tres algoritmos de búsqueda con respecto a kNN usando la prueba de Tukey-Kramer para un nivel de confianza de 5% para el conjunto ***logP***. 105

6.1 Precisión y Cobertura. 126

6.2 Similitud por coseno. 129

6.3 Arquitectura evolutiva propuesta para generación automática de consultas temáticas. En la esquina inferior izquierda se incluyen los dominios de aplicación. 132

6.4 Ejemplos de los posibles resultados que podemos obtener al presentar una consulta a un motor de búsqueda. 137

6.5 Ejemplo de descripción. Descripción disponible en DMOZ para el tópico *"Business/Business and Society"*. 141

6.6 Dos testeos sobre el tópico *Business* que muestran la calidad promedio de las consultas sobre cinco corridas independientes. 141

6.7 Dos testeos sobre el tópico *Recreation* que muestran la calidad promedio de las consultas sobre cinco corridas independientes. 142

6.8 Dos testeos sobre el tópico *Society* que muestran la calidad promedio de las consultas sobre cinco corridas independientes. 142

6.9 Diagramas de dispersión mostrando la distribución de los valores de similitud para los mejores resultados asociados con los individuos para cada generación con $P_m = 0$ (superior), $P_m = 0,03$ (centro) y $P_m = 0.3$ (inferior) para el tópico *Business*. 143

6.10 Calidad promedio de las consultas calculada sobre cinco corridas independientes para los tópicos *Business, Recreation* y *Society*; sin usar mutación ($P_m = 0$), usando la probabilidad de mutación clásica ($P_m = 0.03$) y usando una probabilidad de mutación elevada ($P_m = 0.3$). 144

6.11 Calidad promedio de la mejor consulta calculada sobre cinco corridas independientes para el tópico *Business*, para los resultados obtenidos utilizando σ^N. . . . 148

6.12 Calidad promedio de la mejor consulta calculada sobre 20 corridas independientes para el tópico *Business*, para los resultados obtenidos utilizando σ^N. 149

6.13 Calidad promedio de la mejor consulta calculada sobre 20 corridas independientes para el tópico *Recreation*, para los resultados obtenidos utilizando σ^N. 149

6.14 Calidad promedio de la mejor consulta calculada sobre 20 corridas independientes para el tópico *Society*, para los resultados obtenidos utilizando σ^N. 150

6.15 Evolución de *Precisión@10* (izquierda) y *Cobertura* (derecha) para el tópico (CONSULTING) cuando el objetivo maximizado es *Precisión@10*. 157

6.16 Evolución de *Precisión@10* (izquierda) y *Cobertura* (derecha) para el tópico (CONSULTING) cuando el objetivo maximizado es *Cobertura*. 158

6.17 Evolución de *Precisión@10* (izquierda), *Cobertura* (centro) and $F^\star$ (derecha) para el tópico CONSULTING al ejecutar el algoritmo NSGA-II. 159

6.18 Evolución de *Precisión@10* (izquierda), *Cobertura* (centro) and $F^\star$ (derecha) para el tópico CONSULTING al ejecutar el algoritmo MOEA agregativo. 160

6.19 Comparación de baseline, Bo1* y NSGA-II (izquierda) y comparación de baseline, Bo1* y MOEA agregativo (derecha) para 110 tópicos con respecto a *Precisión@10*. 162

6.20 Comparación de baseline, Bo1* y NSGA-II (izquierda) y comparación de baseline, Bo1* y MOEA agregativo (derecha) para 110 tópicos con respecto a *Cobertura*. . 163

6.21 Comparación de baseline, Bo1* y NSGA-II (izquierda) y comparación de baseline, Bo1* y MOEA agregativo (derecha) para 110 tópicos con respecto a $F^\star$. 163

Índice de tablas

4.1 Resumen de técnicas vistas en el presente capítulo y su ubicación dentro de las distintas clasificaciones. Los algoritmos cuyos nombres se encuentran en negrita son los que se utilizaron dentro de las arquitecturas implementadas en la presente tesis. 53

5.1 Resultados obtenidos para 15 réplicas del EA y 7 réplicas de la segunda fase de la arquitectura para cada método y para cada porción de los datos. Las columnas *Mejor* muestran el resultado promedio sobre las 7 réplicas de la red ejecutadas para el mejor de los mejores subconjuntos obtenidos en las 15 réplicas del EA. Las columnas *Prom.* muestran el resultado promedio obtenido en las $105 = 15 \times 7$ réplicas de la segunda fase de la arquitectura (7 réplicas de la red para el mejor subconjunto obtenido en cada una de las 15 réplicas del EA). 77

5.2 Resultados de ANOVA para la mejor selección de cada método sobre el conjunto de datos S_1 79

5.3 Resultados de ANOVA para la mejor selección de cada método sobre el conjunto de datos S_2 79

5.4 Resultados de la prueba de Dunnett para las 7 réplicas de la red sobre el conjunto de datos S_1 con la mejor selección de cada método. 80

5.5 Resultados de la prueba de Dunnett para las 7 réplicas de la red sobre el conjunto de datos S_2 con la mejor selección de cada método. 80

5.6 Resultados de la prueba ANOVA anidada sobre el conjunto de datos S_1 para las 7 réplicas de la red para la mejor selección de cada réplica del EA para cada método. 81

5.7 Resultados de la prueba ANOVA anidada sobre el conjunto de datos S_2 para las 7 réplicas de la red para la mejor selección de cada réplica del EA para cada método. 82

5.8 Resultados de ANOVA para el promedio de las selecciones de cada método sobre el conjunto de datos S_1 82

5.9 Resultados de ANOVA para el promedio de las selecciones de cada método sobre el conjunto de datos S_2 82

5.10 Resultados de la prueba de Dunnett para las 7 réplicas de la red sobre el conjunto de datos S_1 considerando la mejor selección de cada réplica de cada método. . . 83

5.11 Resultados de la prueba de Dunnett para las 7 réplicas de la red sobre el conjunto de datos S_2 considerando la mejor selección de cada réplica de cada método. . . 83

5.12 Comparación de resultados para ***logBBB***. Las columnas MSEP y R^2 corresponden al error cuadrado promedio y al coeficiente de determinación obtenidos sobre el conjunto de validación. El subconjunto I corresponde al mejor subconjunto reportado en [KLVHC08], mientras que el subconjunto II se obtuvo mediante el método de Figuereido [Fig03]. 92

5.13 Comparación de resultados para ***logHIA***. Las columnas MSEP y R^2 corresponden al error cuadrado promedio y al coeficiente de determinación obtenidos sobre el conjunto de validación. El subconjunto VII corresponde al mejor subconjunto reportado en [KLVHC08], mientras que el subconjunto VIII se obtuvo mediante el método de Figuereido [Fig03]. 93

5.14 Comparación de resultados para ***logP***. Las columnas MSEP y R^2 corresponden al error cuadrado promedio y al coeficiente de determinación obtenidos sobre el conjunto de validación. El subconjunto XIII corresponde al mejor subconjunto reportado en [YCE$^+$02], mientras que el subconjunto XIV se obtuvo mediante el método de Figuereido [Fig03]. 94

5.15 Rendimiento promedio de las funciones de regresión: MSEP promedio para el percentil 50 de cada combinación. 96

5.16 Resultados de la prueba ANOVA de dos direcciones para ***logBBB***. 97

5.17 Resultados de la prueba ANOVA de dos direcciones para ***logBBB*** considerando sólo los algoritmos de búsqueda NSGA-II y SPEA2. 98

5.18 Resultados de la prueba ANOVA de un sentido para ***logBBB*** considerando los tres algoritmos de búsqueda con respecto a MLR como método de regresión. . . 99

5.19 Resultados de la prueba ANOVA de dos direcciones para ***logHIA***. 100

5.20 Resultados de la prueba ANOVA de dos direcciones para ***logHIA*** considerando sólo los algoritmos de búsqueda NSGA-II y SPEA2. 100

5.21 Resultados de la prueba ANOVA de un sentido para ***logHIA*** considerando los tres algoritmos de búsqueda con respecto a MLR como método de regresión. . . 102

5.22 Resultados de la prueba ANOVA de dos direcciones para ***logP***. 103

5.23 Resultados de la prueba ANOVA de un sentido para ***logP*** considerando los tres algoritmos de búsqueda con respecto a MLR como método de regresión. 104

5.24 Resultados de la prueba ANOVA de un sentido para ***logP*** considerando los tres algoritmos de búsqueda con respecto a kNN como método de regresión. 105

5.25 Descriptores moleculares seleccionados por los diferentes subconjuntos reportados en las tablas 5.12, 5.13 y 5.14 108

5.26 Procedimiento de comparación t-student entre dos métodos con respecto al mismo conjunto de datos muestrales. 110

5.27 Análisis de la prueba de comparación t-student para los tres conjuntos de datos para 10 replicas: (1) el error de validación externa es significativamente menor que el error de validación interna, (2) no hay diferencia estadísticamente significativa entre ambos procedimientos de validación y (3) algo distinto de (1) y (2). 110

6.1 Comparación entre la primera generación y la última generación. Intervalos de confianza para calidad promedio de las consultas para el tópico *Business*. . . . 144

6.2 Comparación entre la primera generación y la última generación. Intervalos de confianza para calidad promedio de las consultas para el tópico *Recreation*. . . . 144

6.3 Comparación entre la primera generación y la última generación. Intervalos de confianza para calidad promedio de las consultas para el tópico *Society*. 145

6.4 Primera generación vs. última generación de consultas AND y OR evolucionadas con NSGA-II: *Precisión@10* promedio, *Cobertura* promedio y $F^{\star}$ promedio sobre 10 tópicos. 159

6.5 Primera generación vs. última generación de consultas evolucionadas con NSGA-II: promedio, intervalo de confianza y mejora de la calidad de las consultas sobre 110 tópicos. 159

6.6 Primera generación vs. última generación de consultas AND y OR evolucionadas con el MOEA agregativo: *Precisión@10* promedio, *Cobertura* promedio y $F^{\star}$ promedio sobre 10 tópicos. 160

6.7 Primera generación vs. última generación de consultas evolucionadas con el MOEA agregativo: promedio, intervalo de confianza y mejora de la calidad de las consultas sobre 110 tópicos. 160

6.8 Rendimiento de las consultas AND y OR para Baseline, NSGA-II y MOEA agregativo: *Precisión@10* promedio, *Cobertura* promedio y $F^{\star}$ promedio sobre 10 tópicos. 162

6.9 Consultas Baseline vs. consultas ajustadas con el método Bo1 y Bo1* vs. consultas evolucionadas con NSGA-II y MOEA agregativo: media, intervalos de confianza y mejora observada en la calidad promedio de las consultas sobre 110 tópicos con respecto al método Baseline. TrS refiere al tamaño del índice de entrenamiento utilizado en cada caso. 164

6.10 Correlación de Pearson entre tópico, tamaño y *Precisión@10*. TrS refiere al tamaño del índice de entrenamiento. 166

Lista de publicaciones derivadas

Publicaciones en revistas indexadas en ISI

- Cecchini, R. L., Lorenzetti C. M., Maguitman A. G., Brignole, N. B. "*Multi-Objective Evolutionary Algorithms for Context-Based Search*". Journal of the American Society for Information Science and Technology. Vol. 61, pp. 1258–1274 (2010). ISSN: 1532-2882. John Wiley & Sons, Inc. ***ISI Impact factor: 1,954***.
- Soto, A., Cecchini, R., Vazquez, G., Ponzoni, I. "*Multi-Objective Feature Selection in QSAR using a Machine Learning Approach*". QSAR & Combinatorial Science. Vol. 28, pp. 1509-1523 (2009). ISSN: 1868-1743. Wiley-VCH Verlag GmbH & Co. ***ISI Impact factor: 2,594***.
- Cecchini, R. L., Lorenzetti C. M., Maguitman A. G., Brignole, N. B. "*Using Genetics Algorithms to Evolve a Population of Queries*". Adaptative Information Retrieval. Information Processing and Management. Vol. 44, issue 6, pp. 1863-1878 (2008). ISSN: 0306-4573. Elsevier. ***ISI Impact factor: 1,852***

Publicaciones en otras revistas y en la serie Lecture Notes in Computer Science

- Cecchini, R. L., Lorenzetti C. M., Maguitman A. G. "*Multi-objective Query Optimization Using Topic Ontologies*". Proceedings de FQAS 2009 - Flexible Query Answering Systems. Lecture Notes in Computer Science. Vol. 5822, pp. 145-156 (2009). ISSN: 0302-9743. Springer.
- Cecchini, R. L.; Lorenzetti, C. M., Maguitman, A. G. *Evolving Disjunctive and Conjunctive Topical Queries based on Multi-objective Optimization Criteria*. Ibero-American Journal of Artificial Intelligence. Vol. 13, pp. 14-26 (2009). AEPIA. ISSN: 1137-3601. Indexada en DBLP, Scopus y Latindex.
- Soto, A.[1], Cecchini, R.[1], Vazquez, G., Ponzoni, I. "*An Evolutionary Approach for Feature Selection applied to ADMET Prediction*". Ibero-American Journal of Artificial Intelligence. Vol. 37, pp. 55-63 (2008). AEPIA. ISSN: 1137-3601. Indexada en DBLP, Scopus y Latindex.
- Soto, A., Cecchini, R., Vazquez, G., Ponzoni, I. "*A Wrapper-Based Feature Selection Method for ADMET Prediction Using Evolutionary Computing*". Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, Lecture Notes in Computer Science. Vol. 4973, pp. 188-199 (2008). ISSN: 0302-9743. Springer.

Publicaciones en conferencias internacionales

- Soto, A. J., Cecchini, R. L., Ponzoni, I., Vazquez, G. E. *"A new method for multi-objective selection of molecular descriptors for QSAR/QSPR"*. ISCB-LA 2010: International Society for Computational Biology Regional Latin American meeting. Lugar y fecha de realización: Montevideo, Uruguay, 13 al 16 de Marzo de 2010.
- Soto, A. J.[1], Cecchini, R. L.[1], Palomba, D., Vazquez, G. E., Ponzoni, I. "*An Evolutionary Approach for Multi-Objective Feature Selection in ADMET Prediction*". CLEI 2008: Conferencia Latinoamericana de Informática. Lugar y fecha de realización: Santa Fé, 8 al 12 de Septiembre de 2008. pp. 112-121. ISBN: 978-950-9770-02-7.
- Cecchini, R. L.[1], Soto, A. J.[1], Vazquez, G. E., Ponzoni, I. "*A Genetic Algorithm for Detection of Relevant Descriptors in ADMET Prediction*". BSB 2007: Brazilian Symposium on Bioinformatics 2007. Lugar y fecha de realización: Angra dos Reis, Rio de Janeiro, Brazil, 29 al 31 de Agosto de 2007. pp. 62-65. ISBN: 978-85-7669-123-5.
- Soto, A. J., Cecchini, R. L., Ponzoni, I., Vazquez, G. E. "*Computational Intelligence Methods for Physicochemical Property Prediction*". CLAFQO-9: 9th Latin American Conference on Physical Organic Chemistry. Lugar y fecha de realización: Córdoba, 30 de Septiembre - 5 de Octubre de 2007.

Publicaciones en conferencias nacionales

- Cecchini, R. L., Lorenzetti, C. M. Maguitman, A. G. "*A Multi-Objective Evolutionary Algorithm Approach to Learn Disjunctive and Conjunctive Topical Queries*". ASAI 2009: XI Argentine Symposium On Artificial Intelligence - 38 JAIIO: 38º Jornadas Argentinas de Informática e Investigación Operativa. Lugar y fecha de realización: Mar del Plata, 24 al 25 de Agosto de 2009. ISSN 1850-2776.

- Cecchini, R. L., Lorenzetti, C. M., Maguitman, A. G., Brignole, N. B. "*Genetic Algorithms for Topical Web Search: A Study of Different Mutation Rates*". VIII WASI: VIII Workshop de Agentes y Sistemas Inteligentes - CACIC 2007: XIII Congreso Argentino de Ciencias de la Computación. Lugar y fecha: Corrientes y Resistencia, 1 al 5 de Octubre de 2007. pp. 1585-1595. ISBN 978-950-656-109-3.
- Lorenzetti, C. M., Cecchini, R. L., Maguitman A. "*Intelligent Methods for Information Access in Context: The Role of Topic Descriptors and Discriminators*". VIII WASI: VIII Workshop de Agentes y Sistemas Inteligentes - CACIC 2007: XIII Congreso Argentino de Ciencias de la Computación. Lugar y fecha: Corrientes y Resistencia, 1 al 5 de Octubre de 2007. pp. 1608-1619. ISBN 978-950-656-109-3.

[1]Estos autores contribuyeron en forma equivalente en este trabajo (notado de igual forma en el trabajo original).

- Cecchini, R. L., Lorenzetti, C. M., Maguitman, A. G., Brignole, N. B. *Searching the Web in Context: Genetic Algorithms for Exploring Query Space*". SSI: 5º Simposio sobre la Sociedad de la Información - 36 JAIIO: 36ºJornadas Argentinas de Informática e Investigación Operativa. Lugar y fecha de realización: Mar del Plata, 27 al 31 de Agosto de 2007. pp. 183-196. ISSN 1850-2776.
- Soto, A. J.[1], Cecchini, R. L.[1], Vazquez, G. E., Ponzoni, I. "*Feature Selection for ADMET Prediction using Genetic Algorithms*". ASAI 2007: IX Argentine Symposium On Artificial Intelligence - 36 JAIIO: 36º Jornadas Argentinas de Informática e Investigación Operativa. Lugar y fecha de realización: Mar del Plata, 27 al 31 de Agosto de 2007. pp. 77-88. ISSN 1850-2776.
- Cecchini, R. L., Vazquez, G. E., Brignole, N. B. "*Programación Genética y Métodos Numéricos*". ASAI 2006: VIII Argentine Symposium on Artificial Intelligence - 35 JAIIO: 35º Jornadas Argentinas de Informática e Investigación Operativa. Lugar y fecha de realización: Mendoza, 4 al 8 de Septiembre de 2006. pp. 117-128. ISSN 1850-2776.
- Cecchini, R. L., Lorenzetti, C. M., Maguitman, A. "Algoritmos Genéticos Para la Búsqueda Web basada en Contextos Temáticos". WICC 2007: IX Workshop de Investigadores en Ciencias de la Computación. Lugar y fecha de realización: Trelew, 3 al 4 de Mayo de 2007. pp. 6-10. ISBN 978-950-763-075-0.

[1]Estos autores contribuyeron en forma equivalente en este trabajo (notado de igual forma en el trabajo original).

Bibliografía

[ABS00] Abiteboul, S., Buneman, P., and Suciu, D. *Data on the Web: from relations to semistructured data and XML.* Morgan Kaufmann Publishers Inc., 2000.

[ACR04] Amati, G., Carpineto, C., and Romano, G. Query Difficulty, Robustness and Selective Application of Query Expansion. In *Advances in Information Retrieval, 26th European Conference on IR research* (Berlin / Heidelberg, 2004), Springer, pp. 127–137.

[AF77] Attar, R., and Fraenkel, A. S. Local Feedback in Full-text Retrieval Systems. *Journal of the ACM (JACM) 24*, 3 (1977), 397–417.

[AFJM95] Armstrong, R., Freitag, D., Joachims, T., and Mitchell, T. WebWatcher: A Learning Apprentice for the World Wide Web. In *AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments* (Palo Alto, CA, USA, March 1995), AAAI Press, pp. 6–12.

[AH84] Anton, H., and Herr, A. *Calculus with analytic geometry*, 2nd ed. Wiley, New York, NY, USA, 1984.

[Ama03] Amati, G. *Probabilistics Models for Information Retrieval based on Divergence from Randomness.* PhD thesis, Department of Computing Science, University of Glasgow, UK, June 2003.

[Ash06] Ashlock, D. *Evolutionary Computation for Modeling and Optimization.* Interdisciplinary Applied Mathematics. Springer, 2006.

[AT99] Anick, P. G., and Tipirneni, S. The paraphrase search assistant: terminological feedback for iterative information seeking. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (1999), ACM Press, pp. 153–159.

[AV97] Anick, P. G., and Vaithyanathan, S. Exploiting clustering and phrases for context-based information retrieval. In *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval* (1997), ACM Press, pp. 314–323.

[AV98] Andrade, M., and Valencia, A. Automatic extraction of keywords from scientific text: application to the knowledge domain of protein families. *Bioinformatics 14*, 7 (1998), 600–607.

[Bäc96] Bäck, T. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms.* Oxford University Press, Oxford, UK, 1996.

[BB88] Brassard, G., and Bratley, P. *Algorithmics: theory & practice*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[Bel00] Belkin, N. J. Helping people find what they don't know. *Communications of the ACM 43*, 8 (2000), 58–61.

[Ber01] Bergman, M. K. The deep web: Surfacing hidden value. *Journal of Electronic Publishing 7*, 1 (2001).

[BEYT+03] Bekkerman, R., El-Yaniv, R., Tishby, N., Winter, Y., Guyon, I., and Elisseeff, A. Distributional word clusters vs. words for text categorization. *Journal of Machine Learning Research 3* (2003), 1183–1208.

[BFM97] Bäck, T., Fogel, D. B., and Michalewicz, Z., Eds. *Handbook of Evolutionary Computation*. IOP Publishing Ltd., Bristol, UK, UK, 1997.

[BFOS84] Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. *Classification and Regression Trees*. Chapman & Hall, New York, NY, Belmont, Calif., 1984.

[BHB01] Budzik, J., Hammond, K. J., and Birnbaum, L. Information Access in Context. *Knowledge Based Systems 14*, 1–2 (2001), 37–53.

[BHS97] Bäck, T., Hammel, U., and Schwefel, H. Evolutionary computation: comments on the history and current state. *IEEE Trans. Evolutionary Computation 1*, 1 (1997), 3–17.

[BSH+04] Bayram, E., Santago, P., Harris, R., Xiao, Y. D., Clauset, A. J., and Schmitt, J. D. Genetic algorithms and self-organizing maps: a powerful combination for modeling complex QSAR and QSPR problems. *Journal of Computer-Aided Molecular Design 18*, 7 (July 2004), 483–493.

[BSM95] Buckley, C., Singhal, A., and Mitra, M. New Retrieval Approaches Using SMART. In *Proceedings of the Fourth Text REtrieval Conference (TREC-4)* (Gaithersburg, MD, USA, 1995), D. K. Harman, Ed., vol. Special Publication 500-236, National Institute of Standards and Technology (NIST).

[Bud03] Budzik, J. L. *Information access in context: experiences with the watson system*. PhD thesis, Northwestern University, Evanston, IL, USA, 2003. Adviser: Hammond, Kristian J.

[BW97] Baldonado, M. Q. W., and Winograd, T. SenseMaker: an information-exploration interface supporting the contextual evolution of a user's interests. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (Atlanta, Georgia, March 1997), ACM Press, pp. 11–18.

[BW09] Burden, F., and Winkler, D. Optimal sparse descriptor selection for qsar using bayesian methods. *QSAR & Combinatorial Science 28*, 6-7 (2009), 645–653.

[BYRN99] Baeza-Yates, R. A., and Ribeiro-Neto, B. A. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Reading, MA, USA, May 1999.

[CD90] Chen, H., and Dhar, V. Online query refinement on information retrieval systems: a process model of searcher/system interactions. In *Proceedings of the 13th annual international ACM SIGIR conference on Research and development in information retrieval* (1990), ACM Press, pp. 115–133.

[CD00] Chen, H., and Dumais, S. Bringing order to the Web: Automatically categorizing search results. In *Proceedings of CHI'00, Human Factors in Computing Systems* (2000).

[CFP04] Caramia, M., Felici, G., and Pezzoli, A. Improving search results with data mining in a thematic search engine. *Computers & Operations Research 31*, 14 (2004), 2387–2404.

[Cha02] Chakrabarti, S. *Mining the Web. Discovering Knowledge from Hypertext Data.* Morgan Kaufmann Publishers, 2002.

[Che97] Chen, C. Structuring and visualising the WWW by generalised similarity analysis. In *Proceedings of the eighth ACM conference on Hypertext* (1997), ACM Press, pp. 177–186.

[Chu02] Chui, M. *I Still Haven't Found What I'm Looking For: Web Searching as Query Refinement.* PhD thesis, Indiana University, 2002.

[CLV07] Coello Coello, C. A., Lamont, G. B., and Van Veldhuizen, D. A. *Evolutionary Algorithms for Solving Multi-Objective Problems*, second ed. Springer, New York, September 2007. ISBN 978-0-387-33254-3.

[Coe96] Coello, C. A. C. *An empirical study of evolutionary techniques for multiobjective optimization in engineering design.* PhD thesis, Department of Computer Science, Tulane University, New Orleans, LA, USA, 1996. Chair-Alan D. Christiansen.

[Coe02] Coello Coello, C. A. Theoretical and Numerical Constraint Handling Techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering 191(11-12)* (2002), 1245–1287.

[Coo68] Cooper, W. S. Expected search length: A single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. *American Documentation 19*, 1 (January 1968), 30–41.

[CPKT92] Cutting, D. R., Pedersen, J. O., Karger, D., and Tukey, J. W. Scatter/Gather: A Cluster-based Approach to Browsing Large Document collections. In *Proceedings of the Fifteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval* (1992), pp. 318–329.

[CR96] Chu, H., and Rosenthal, M. Search engines for the World Wide Web: A comparative study and evaluation methodology. In *Annual Conference Proceedings (ASIS'96)* (October 1996), pp. 127–135.

[CS96] Cherkauer, K. J., and Shavlik, J. W. Growing simpler decision trees to facilitate knowledge discovery. In *In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (1996), AAAI Press, pp. 315–318.

[CvdBD99] Chakrabarti, S., van den Berg, M., and Dom, B. Focused crawling: a new approach to topic-specific Web resource discovery. *Computer Networks 31*, 11–16 (1999), 1623–1640.

[DAPM00] Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA–II. In *PPSN VI: Proceedings of the 6th International Conference on Parallel Problem Solving from Nature* (London, UK, 2000), Springer, Ed., vol. 1917, Springer-Verlag, pp. 849–858.

[Dar59] Darwin, C. *On the Origin of Species by Means of Natural Selection.* John Murray, 1859.

[Deb01] Deb, K. *Multi-Objective Optimization using Evolutionary Algorithms.* John Wiley & Sons, Chichester, UK, 2001.

[Deb04] Deb, K. *Multi–Objective Optimization Using Evolutionary Algorithms.* John Wiley & Sons, 2004.

[DGWC07] Dutta, D., Guha, R., Wild, D., and Chen, T. Ensemble feature selection: Consistent descriptor subsets for multiple qsar models. *Journal of Chemical Information and Modeling 47*, 3 (May 2007), 989–997.

[DL97] Dash, M., and Liu, H. Feature selection for classification. *Intell. Data Anal. 1*, 1-4 (1997), 131–156.

[DR94] Davies, S., and Russell, S. Np-completeness of searches for smallest possible feature sets. In *AAAI Symposium on Intelligent Relevance* (1994), AAAI Press, pp. 37–39.

[EJ05] Escudeiro, N. F., and Jorge, A. M. Semi-automatic creation and maintenance of web resources with webtopic. In *EWMF/KDO* (2005), M. Ackermann, B. Berendt, M. Grobelnik, A. Hotho, D. Mladenic, G. Semeraro, M. Spiliopoulou, G. Stumme, V. Svátek, and M. van Someren, Eds., vol. 4289 of *Lecture Notes in Computer Science*, Springer, pp. 82–102.

[Ert08] Ertl, P. *Polar Surface Area.* Wiley-VCH Verlag GmbH & Co. KGaA, 2008, ch. Chapter 5, pp. 111–126.

[ES03] Eiben, A. E., and Smith, J. E. *Introduction to Evolutionary Computing*, Corr. 2nd printing 2007 ed. Natural Computing Series. Springer, 2003.

[FA90] Fogel, D. B., and Atmar, J. Comparing genetic operators with gaussian mutations in simulated evolutionary processes using linear systems. *Biological Cybernetics 63* (1990), 111–114.

[FF93] Fonseca, C. M., and Fleming, P. J. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In *Proceedings of the 5th International Conference on Genetic Algorithms* (San Francisco, CA, USA, 1993), Morgan Kaufmann Publishers Inc., pp. 416–423.

[Fig03] Figueiredo, M. A. Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence 25* (2003), 1150–1159.

[Fis96] Fishman, G. *Monte Carlo : concepts, algorithms, and applications.* Springer-Verlag, Springer, 1996.

[Fog92] Fogel, D. B. *Evolving Artificial Intelligence.* PhD thesis, University of California, San Diego, 1992.

[Fog95] Fogel, D. *Evolutionary Computation.* IEEE Press, 1995.

[Fog98] Fogel, D. B. *Evolutionary Computation: The Fossil Record.* Wiley-IEEE Press, 1998.

[Fog99] Fogel, L. J. *Intelligence through simulated evolution: forty years of evolutionary programming.* John Wiley & Sons, Inc., New York, NY, USA, 1999.

[Fou85] Fourman, M. P. Compaction of symbolic layout using genetic algorithms. In *ICGA* (1985), J. J. Grefenstette, Ed., Lawrence Erlbaum Associates, pp. 141–153.

[FOW66] Fogel, L., Owens, A., and Walsh, M. *Artificial Intelligence through Simulated Evolution*. Wiley, Chichester, UK, 1966.

[FS91] Frieder, O., and Siegelmann, H. T. On the allocation of documents in multiprocessor information retrieval systems. In *Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 1991), ACM, pp. 230–239.

[FTCC05] Fernández, M., Tundidor-Camba, A., and Caballero, J. Modeling of cyclin-dependent kinase inhibition by 1-pyrazolo[3, 4-]pyrimidine derivatives using artificial neural network ensembles. *Journal of Chemical Information and Modeling 45*, 6 (2005), 1884–1895.

[GBNT04] Gilad-Bachrach, R., Navot, A., and Tishby, N. Margin based feature selection - theory and algorithms. In *ICML* (2004).

[GE03] Guyon, I., and Elisseeff, A. An introduction to variable and feature selection. *J. Mach. Learn. Res. 3* (2003), 1157–1182.

[Gha08] Gharagheizi, F. QSPR Studies for Solubility Parameter by Means of Genetic Algorithm-Based Multivariate Linear Regression and Generalized Regression Neural Network. *QSAR & Combinatorial Science 27* (2008), 165–170.

[GHGL99] Garrett, C., Huang, J., Goltz, M., and Lamont, G. Parallel real-valued genetic algorithms for bioremediation optimization of tce-contaminated groundwater. In *Proceedings of the 1999 Congress on Evolutionary Computation* (Jul 1999), IEEE, pp. 2183–2189.

[GHK+03] Gierz, G., Hofmann, K. H., Keimel, K., Lawson, J. D., Mislove, M., and Scott, D. S. *Continuous Lattices and Domains*, vol. 93 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2003.

[GOCS06] Gola, J., Obrezanova, O., Champness, E., and Segall, M. Admet property prediction: The state of the art and current challenges. *QSAR & Combinatorial Science 25* (2006), 1172–1180.

[Gol89] Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[Gor88] Gordon, M. Probabilistic and genetic algorithms in document retrieval. *Communications of the ACM 31*, 10 (1988), 1208–1218.

[Gor91] Gordon, M. D. User-based document clustering by redescribing subject descriptions with a genetic algorithm. *JASIS 42*, 5 (1991), 311–322.

[GPC08] Guerra, A., Páez, J., and Campillo, N. Artificial neural networks in admet modeling: Prediction of blood-brain barrier permeation. *QSAR & Combinatorial Science 27*, 5 (2008), 586–594.

[GR87] Goldberg, D. E., and Richardson, J. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms* (1987), J. J. Grefenstette, Ed., San Francisco, CA: Morgan Kaufmann, pp. 148–154.

[Gre98] Greenberg, J. *An Examination of the Impact of Lexical-Semantic Relationships on Retrieval Effectiveness During the Query Expansion Process.* PhD thesis, University of Pittsburgh, 1998.

[GST+99] Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E. S. Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. *Science 286*, 5439 (1999), 531–537.

[Han06] Han, M. K. J. *Data mining: concepts and techniques.* 2006.

[HBSe+07] Horvath, D., Bonachera, F., Solov' ev, V., Gaudin, C., and Varnek, A. Stochastic versus Stepwise Strategies for Quantitative Structure–Activity Relationship GenerationHow Much Effort May the Mining for Successful QSAR Models Take? *Journal of Chemical Information and Modeling 47* (2007), 927–939.

[HC01] Hawking, D., and Craswell, N. Overview of the TREC-2001 Web track. In *Proceedings of the Tenth Text REtrieval Conference TREC-2001* (Gaithersburg, MD, USA, 2001), E. M. Voorhees and D. K. Harman, Eds., vol. Special Publication 500-250, National Institute of Standards and Technology (NIST), pp. 61–67.

[HK05] Huang, T. M., and Kecman, V. Gene extraction for cancer diagnosis by support vector machines - an improvement. *Artificial Intelligence in Medicine 35*, 1-2 (2005), 185–194.

[HLT00] Huuskonen, J. J., Livingstone, D. J., and Tetko, I. V. Neural network modeling for estimation of partition coefficient based on atom-type electrotopological state indices. *Journal of Chemical Information and Computer Sciences 40*, 4 (2000), 947–955.

[HM79] Hwang, C. L., and Masud, A. S. M. *Multiple Objective Decision Making - Methods and Applications*, vol. 164 of *Lecture Notes in Economics and Mathematical Systems.* Springer-Verlag, 1979.

[HNG94] Horn, J., Nafpliotis, N., and Goldberg, D. E. A niched pareto genetic algorithm for multiobjective optimization. In *International Conference on Evolutionary Computation* (1994), pp. 82–87.

[Hol62] Holland, J. H. Outline for a logical theory of adaptive systems. *j-J-ACM 9*, 3 (jul 1962), 297–314. Early work on genetic algorithms. Reprinted in.

[Hol73] Holland, J. H. Genetic algorithms and the optimal allocation of trials. *SIAM Journal on Computing 2*, 2 (1973), 88–105.

[HP96] Hearst, M. A., and Pedersen, J. O. Reexamining the cluster hypothesis: Scatter/Gather on retrieval results. In *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval* (Zürich, CH, 1996), pp. 76–84.

[HPR+08] Hakenberg, J., Plake, C., Royer, L., Strobelt, H., Leser, U., and Schroeder, M. Gene mention normalization and interaction extraction with context models and sentence motifs. *Genome Biology 9*, Suppl 2 (2008), S14.

[HYMRY98] Hsinchun, C., Yi-Ming, C., Ramsey, M., and Yang, C. C. An intelligent personal spider (agent) for dynamic Internet/intranet searching. *Decision Support Systems 23*, 1 (1998), 41–58.

[ILES00] Inza, I., Larrañaga, P., Etxeberria, R., and Sierra, B. Feature subset selection by bayesian network-based optimization. *Artificial Intelligence 123*, 1-2 (2000), 157 – 184.

[Jen03] Jensen, M. T. Reducing the run-time complexity of multiobjective EAs: The NSGA-II and other algorithms. *IEEE Transactions on Evolutionary Computation 7*, 5 (2003), 503–515.

[JW04] Jordan, C., and Watters, C. R. Extending the Rocchio Relevance Feedback Algorithm to Provide Contextual Retrieval. In *Advances in Web Intelligence, Proceedings of the Second International Atlantic Web Intelligence Conference (AWIC)* (Berlin / Heidelberg, 2004), vol. 3034 of *Lecture Notes in Computer Science*, Springer, pp. 135–144.

[JZS07] Jansen, B. J., Zhang, M., and Spink, A. Patterns and transitions of query reformulation during web searching. *International Journal of Web Information Systems 3* (2007), 328–340.

[Kan03] Kantardzic, M. *Data Mining: Concepts, Models, Methods, and Algorithms*, 1 ed. No. ISBN: 0471228524. John Wiley & Sons, 2003.

[KC98] Khan, I., and Card, H. C. Adaptive information agents using competitive learning. *J. Netw. Comput. Appl. 21*, 2 (1998), 69–89.

[KC99] Knowles, J., and Corne, D. The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation. In *Proceedings of the Congress on Evolutionary Computation* (Mayflower Hotel, Washington D.C., USA, June-September 1999), P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, Eds., vol. 1, IEEE Press, pp. 98–105.

[KGLH03] Knarr, M. R., Goltz, M. N., Lamont, G. B., and Huang, J. *In Situ* Bioremediation of Perchlorate-Contaminated Groundwater using a Multi-Objective Parallel Evolutionary Algorithm. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)* (Canberra, Australia, December 2003), vol. 3, IEEE Press, pp. 1604–1611.

[KGV83] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983 220, 4598* (1983), 671–680.

[Kli01] Klink, S. Query reformulation with collaborative concept-based expansion. In *Proceedings of the First International Workshop on Web Document Analysis* (2001).

[KLK98] Kaski, S., Lagus, K., and Kohonen, T. Websom - Self-organizing Maps of Document Collections. *Neurocomputing 21* (1998), 101–117.

[KLVHC08] Konovalov, D. A., Llewellyn, L. E., Vander Heyden, Y., and Coomans, D. Robust cross-validation of linear regression qsar models. *Journal of Chemical Information and Modeling 48*, 10 (2008), 2081–2094.

[Kon94] Kononenko, I. Estimating attributes: Analysis and extensions of relief. In *ECML* (1994), F. Bergadano and L. D. Raedt, Eds., vol. 784 of *Lecture Notes in Computer Science*, Springer, pp. 171–182.

[Koz92] Koza, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

[KR92] Kira, K., and Rendell, L. A. The feature selection problem: Traditional methods and a new algorithm. In *AAAI* (1992), pp. 129–134.

[KSM03] Kim, Y., Street, W. N., and Menczer, F. Feature selection in data mining. 80–105.

[KSS97] Kautz, H., Selman, B., and Shah, M. The Hidden Web. *AI Magazine 18*, 2 (1997), 27–36.

[KT00] Kobayashi, M., and Takeda, K. Information retrieval on the Web. *ACM Computing Surveys 32*, 2 (2000), 144–173.

[LB08] Liu, Y.-H., and Belkin, N. J. Query reformulation, search performance, and term suggestion devices in question-answering tasks. In *IIiX '08: Proceedings of the second international symposium on Information interaction in context* (New York, NY, USA, 2008), ACM, pp. 21–26.

[LBMW00] Leake, D. B., Bauer, T., Maguitman, A., and Wilson, D. C. Capture, Storage and Reuse of Lessons about Information Resources: Supporting Task-Based Information Search. In *Proceedings of the AAAI-00 Workshop on Intelligent Lessons Learned Systems.* (Austin, Texas, 2000), AAAI Press, pp. 33–37.

[LDM02] Lavine, B. K., Davidson, C. E., and Moores, A. J. Innovative genetic algorithms for chemoinformatics. *Chemometrics and Intelligent Laboratory Systems 60*, 1-2 (2002), 161 – 171.

[LG98] Leardi, R., and Gonzalez, A. L. Genetic algorithms applied to feature selection in pls regression: how and when to use them. *Chemometrics and Intelligent Laboratory Systems 41*, 2 (July 1998), 195–207.

[Lie95] Lieberman, H. Letizia: An Agent That Assists Web Browsing. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. IJCAI-95* (Montreal, Quebec, Canada, 1995), C. S. Mellish, Ed., Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, pp. 924–929.

[LL01] Loia, V., and Luongo, P. Genetic-based fuzzy clustering for automatic web document categorization. In *SAC* (2001), ACM, pp. 477–478.

[LLC03] Leroy, G., Lally, A. M., and Chen, H. The use of dynamic contexts to improve casual internet searching. *ACM Transactions on Information Systems (TOIS) 21*, 3 (2003), 229–253.

[LLYW03] Liu, S.-S., Liu, H.-L., Yin, C.-S., and Wang, L.-S. Vsmp: A novel variable selection and modeling method based on the prediction. *Journal of Chemical Information and Computer Sciences 43*, 3 (2003), 964–969.

[LLYY08] Liu, T.-Y., Li, G.-Z., Yang, J. Y., and Yang, M. Q. Feature selection for the imbalanced qsar problems by using easyensemble. *International Journal of Computational Biology and Drug Design 1*, 4 (2008), 334–346.

[LM07] Liu, H., and Motoda, H. *Computational Methods of Feature Selection (Chapman & Hall/Crc Data Mining and Knowledge Discovery Series)*. Chapman & Hall/CRC, 2007.

[LMR+03] Leake, D., Maguitman, A., Reichherzer, T., Cañas, A., Carvalho, M., Arguedas, M., Brenes, S., and Eskridge, T. Aiding Knowledge Capture by Searching for Extensions of Knowledge Models. In *KCAP-2003 – Proceedings of the 2nd International Conference on Knowledge Capture* (Sanibel Island, FL, USA, 2003), ACM Press, pp. 44–53.

[LTC+01] L., L., T.A., D., C.R., W., A.J., L., and L.G., P. Gene assessment and sample classification for gene expression data using a genetic algorithm / k-nearest neighbor method. *Combinatorial Chemistry & High Throughput Screening 4* (December 2001), 727–739(15).

[LWDP01] Li, L., Weinberg, C. R., Darden, T. A., and Pedersen, L. G. Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the ga/knn method. *Bioinformatics 17*, 12 (2001), 1131–1142.

[Mag04] Maguitman, A. G. *Intelligent support for knowledge capture and construction.* PhD thesis, Indiana University, Indianapolis, IN, USA, 2004. Chairman-Leake,, David B.

[MB00] Menczer, F., and Belew, R. K. Adaptive retrieval agents: Internalizing local context and scaling up to the web. *Machine Learning 39*, 2/3 (2000), 203–242.

[MBCS00] Maglio, P. P., Barrett, R., Campbell, C. S., and Selker, T. SUITOR: an attentive information system. In *Proceedings of the 5th international conference on Intelligent user interfaces* (New York, NY, USA, 2000), IUI '00, ACM Press, pp. 169–176.

[MBP03] Minaei-Bidgoli, B., and Punch, W. F. Using genetic algorithms for data mining optimization in an educational web-based system. In *GECCO* (2003), E. Cantú-Paz, J. A. Foster, K. Deb, L. Davis, R. Roy, U.-M. O'Reilly, H.-G. Beyer, R. K. Standish, G. Kendall, S. W. Wilson, M. Harman, J. Wegener, D. Dasgupta, M. A. Potter, A. C. Schultz, K. A. Dowsland, N. Jonoska, and J. F. Miller, Eds., vol. 2724 of *Lecture Notes in Computer Science*, Springer, pp. 2252–2263.

[MBVL99] Martin-Bautista, M. J., Vila, M.-A., and Larsen, H. L. A fuzzy genetic algorithm approach to an adaptive information retrieval agent. *J. Am. Soc. Inf. Sci. 50*, 9 (1999), 760–771.

[MF04] Michalewicz, Z., and Fogel, D. B. *How to Solve It: Modern Heuristics.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2004.

[Mic94] Michalewicz, Z. *Genetic algorithms + data structures = evolution programs*, 2., extended ed ed. Springer-Verlag New York, Inc., Springer, 1994.

[MLR05] Maguitman, A., Leake, D., and Reichherzer, T. Suggesting novel but related topics: towards context-based support for knowledge model extension. In *IUI '05: Proceedings of the 10th International Conference on Intelligent User Interfaces* (New York, NY, USA, 2005), ACM Press, pp. 207–214.

[MLRM04] Maguitman, A., Leake, D., Reichherzer, T., and Menczer, F. Dynamic Extraction of Topic Descriptors and Discriminators: Towards automatic context-based topic search. In *Proceedings of the Thirteenth Conference on Information and Knowledge Management (CIKM)* (Washington, DC, November 2004), ACM Press, pp. 463–472.

[MM04] Mitra, P., and Majumder, D. D. Feature selection and gene clustering from gene expression data. *Pattern Recognition, International Conference on 2* (2004), 343–346.

[MNT04] Madsen, K., Nielsen, H. B., and Tingleff, O. Methods for non-linear least squares problems, 2004.

[MPS04] Menczer, F., Pant, G., and Srinivasan, P. Topical web crawlers: Evaluating adaptive algorithms. *ACM Transactions on Internet Technology (TOIT) 4*, 4 (November 2004), 378–419.

[MS00] Modha, D. S., and Spangler, W. S. Clustering hypertext with applications to Web searching. In *Proceedings of the eleventh ACM on Hypertext and hypermedia* (2000), ACM Press, pp. 143–152.

[MZ06] Meiri, R., and Zahavi, J. Using simulated annealing to optimize the feature selection problem in marketing applications. *European Journal of Operational Research 171*, 3 (2006), 842–858.

[NZC05] Ntoulas, A., Zerfos, P., and Cho, J. Downloading textual hidden web content through keyword queries. In *JCDL '05: Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries* (New York, NY, USA, 2005), ACM Press, pp. 100–109.

[OKI+01] Oyama, S., Kokubo, T., Ishida, T., Yamada, T., and Kitamura, Y. Keyword Spices: A New Method for Building Domain-specific Web Search Engines. In *IJCAI* (2001), pp. 1457–1466.

[OLMP07] Ounis, I., Lioma, C., Macdonald, C., and Plachouras, V. Research directions in Terrier: a search engine for advanced retrieval on the web. *Novatica/UPGRADE Special Issue on Web Information Access, Ricardo Baeza-Yates et al. (Eds), Invited Paper VIII*, 1 (February 2007), 49–56.

[Osy85] Osyczka, A. *Multicriteria Optimization for Engineering Design.* Academic Press, 1985.

[Par96] Pareto, V. *Cours d'Economie Politique.* Droz, Genève, 1896.

[PBPK93] Petry, F. E., Buckles, B. P., Prabhu, D., and Kraft, D. H. Fuzzy information retrieval using genetic algorithms and relevance feedback. In *Proceedings of the 56th Annual Meeting of the American Society for Information Science (ASIS)* (Medford, NJ, USA, October 1993), S. Bonzi, Ed., vol. 30, Learned Information, pp. 122–125.

[Pea84] Pearl, J. *Heuristics.* Addison-Wesley, Reading, MA, 1984.

[Phi] Phisprop. The physical properties database (physprop), syracuse research corporation (src), north syracuse, usa, http://www.syrres.com/.

[RA87] Raghavan, V., and Agarwal, B. Optimal determination of user-oriented clusters: an application for the reproductive plan. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application* (Mahwah, NJ, USA, 1987), Lawrence Erlbaum Associates, Inc., pp. 241–246.

[Rec73] Rechenberg, I. *Evolutionsstrategie: optimierung technischer systeme nach prinzipien der biologischen evolution.* Frommann-Holzboog, 1973.

[RGM01] Raghavan, S., and Garcia-Molina, H. Crawling the Hidden Web. In *Proceedings of the Twenty-seventh International Conference on Very Large Databases* (2001).

[Rho97] Rhodes, B. J. The wearable remembrance agent: a system for augmented memory. In *ISWC '97: Proceedings of the 1st IEEE International Symposium on Wearable Computers* (Washington, DC, USA, 1997), IEEE Computer Society, p. 123.

[Rij79] Rijsbergen, C. J. v. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 1979.

[RKC+07] Roulland, F., Kaplan, A. N., Castellani, S., Roux, C., Grasso, A., Pettersson, K., and O'Neill, J. Query reformulation and refinement using nlp-based sentence clustering. In *ECIR* (2007), pp. 210–221.

[Roc71] Rocchio, J. J. Relevance feedback in information retrieval. In *The Smart retrieval system - experiments in automatic document processing*, G. Salton, Ed. Prentice-Hall, Englewood Cliffs, NJ, USA, 1971, ch. 14, pp. 313–323.

[Row07] Rowley, J. The wisdom hierarchy: representations of the DIKW hierarchy. *Journal of Information Science 33*, 2 (2007), 163–180.

[RPLH89] Richardson, J. T., Palmer, M. R., Liepins, G. E., and Hilliard, M. R. Some guidelines for genetic algorithms with penalty functions. In *ICGA* (1989), J. D. Schaffer, Ed., Morgan Kaufmann, pp. 191–197.

[RS96] Rhodes, B. J., and Starner, T. Remembrance Agent: A Continuously Running Automated Information Retrieval System. In *Proceedings of the 1st International Conference on the Practical Applications of Intelligent Agents and Multi-Agent Technologies* (1996), pp. 487–495.

[RTR+01] Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C.-H., Angelo, M., Ladd, C., Reich, M., Latulippe, E., Mesirov, J. P., Poggio, T., Gerald, W., Loda, M., Lander, E. S., and Golub, T. R. Multiclass cancer diagnosis using tumor gene expression signatures. *Proceedings of the National Academy of Sciences of the United States of America 98*, 26 (2001), 15149–15154.

[Sar95] Saracevic, T. Evaluation of evaluation in information retrieval. In *Proceedings of SIGIR* (1995), pp. 138–146.

[SBT02] Selick, H. E., Beresford, A. P., and Tarbit, M. H. The emerging importance of predictive adme simulation in drug discovery. *Drug Discovery Today 7*, 2 (2002), 109 – 116.

[Sch65] Schwefel, H.-P. *Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömungstechnik*. Dipl.-Ing. Thesis, Technical University of Berlin, Hermann Föttinger–Institute for Fluid Dynamics, March 1965.

[Sch84] Schaffer, J. D. *Multiple objective optimization with vector evaluated genetic algorithms*. PhD thesis, Vanderbilt University, 1984.

[Sch85] Schaffer, J. D. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms* (Mahwah, NJ, USA, 1985), Lawrence Erlbaum Associates, Inc., pp. 93–100.

[SCVP09] Soto, A. J., Cecchini, R. L., Vazquez, G. E., and Ponzoni, I. Multi-Objective Feature Selection in QSAR using a Machine Learning Approach. *QSAR & Combinatorial Science 28*, 11-12 (2009), 1509–1523.

[SD94] Srinivas, N., and Deb, K. Multiobjective Optimization Using Nondominated Sorting in Genetic algorithms. *Evolutionary Computation 2*, 3 (1994), 221–248.

[SH04] Somlo, G., and Howe, A. E. QueryTracker: An Agent for Tracking Persistent Information Needs. In *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems* (Los Alamitos, CA, USA, 2004), IEEE Computer Society, pp. 488–495.

[Sha04] Sharma, N. The origin of the data information knowledge wisdom hierarchy, 2004.

[SIL07] Saeys, Y., Inza, I., and Larrañaga, P. A review of feature selection techniques in bioinformatics. *Bioinformatics 23*, 19 (2007), 2507–2517.

[SK96] So, S. S., and Karplus, M. Evolutionary optimization in quantitative structure-activity relationship: An application of genetic neural networks. *J. Med. Chem. 39*, 7 (March 1996), 1521–1530.

[SM01] Smyth, B., and McClave, P. Similarity vs. Diversity. In *Proceedings of the 4th International Conference on Case-Based Reasoning. Vancouver, Canada* (2001).

[SMP05] Srinivasan, P., Menczer, F., and Pant, G. A general evaluation framework for topical crawlers. *Information Retrieval 8*, 3 (2005), 417–447.

[SSM02] Singh, S., Singh, M., and Markou, M. Feature selection for face recognition based on data partitioning. In *ICPR (1)* (2002), pp. 680–683.

[SY73] Salton, G., and Yang, C. On the specification of term values in automatic indexing. *Journal of Documentation 29* (1973), 351–372.

[TAV99] Törn, A., Ali, M. M., and Viitanen, S. Stochastic global optimization: Problem classes and solution techniques. *J. of Global Optimization 14*, 4 (1999), 437–447.

[TCMP05] Todeschini, R., Consonni, V., Mauri, A., and Pavan, M. E-dragon. http://michem.disat.unimib.it/chm/help/edragon/index.html, 2005.

[TH04] Tatarinov, I., and Halevy, A. Efficient query reformulation in peer data management systems. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data* (New York, NY, USA, 2004), ACM, pp. 539–550.

[Tsy08] Tsygankova, I. G. Variable selection in qsar models for drug design. *Current Computer - Aided Drug Design 4* (2008), 132–142.

[TY03] Taskinen, J., and Yliruusi, J. Prediction of physicochemical properties based on neural network modelling. *Advanced Drug Delivery Reviews 55*, 9 (2003), 1163 – 1183. Artificial neural network modeling for pharmaceutical research.

[Vel99] Veldhuizen, D. A. V. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations.* PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.

[VWSG97] Vélez, B., Weiss, R., Sheldon, M. A., and Gifford, D. K. Fast and Effective Query Refinement. In *Proceedings of the 20th ACM Conference on Research and Development in Information Retrieval (SIGIR 97). Philadelphia, PA* (1997), pp. 6–15.

[WF05] Witten, I. H., and Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques*, 2 ed. Morgan Kaufmann, 2005.

[Whi71] Whitney, A. A direct method of nonparametric measurement selection. *Computers, IEEE Transactions on C-20*, 9 (Sept. 1971), 1100–1103.

[Whi01] Whitley, D. An overview of evolutionary algorithms: Practical issues and common pitfalls. *Information and Software Technology 43* (2001), 817–831.

[Wis98] Wishard, L. Precision Among Internet Search Engines: An Earth Sciences Case study. *Issues in Science and Technology Librarianship*, 18 (Spring 1998).

[WVS96] Weiss, R., Vélez, B., and Sheldon, M. A. HyPursuit: a hierarchical network search engine that exploits content-link hypertext clustering. In *Proceedings of the the seventh ACM conference on Hypertext* (1996), ACM Press, pp. 180–193.

[WZ03] Wegner, J. K., and Zell, A. Prediction of aqueous solubility and partition coefficient optimized by a genetic algorithm based descriptor selection method. *Journal of Chemical Information and Computer Sciences 43*, 3 (2003), 1077–1084.

[YCE+02] Yaffe, D., Cohen, Y., Espinosa, G., Arenas, A., and Giralt, F. Fuzzy artmap and back-propagation neural networks based quantitative structure-property relationships (qsprs) for octanol-water partition coefficient of organic compounds. *Journal of Chemical Information and Computer Sciences 42*, 2 (2002), 162–183.

[YK93] Yang, J.-J., and Korfhage, R. Query Optimization in Information Retrieval Using Genetic Algorithms. In *Proceedings of the 5th International Conference on Genetic Algorithms* (San Francisco, CA, USA, 1993), Morgan Kaufmann Publishers Inc., pp. 603–613.

[ZE99] Zamir, O., and Etzioni, O. Grouper: a dynamic clustering interface to Web search results. *Computer Networks (Amsterdam, Netherlands: 1999) 31*, 11–16 (1999), 1361–1374.

[ZHC+04] Zeng, H.-J., He, Q.-C., Chen, Z., Ma, W.-Y., and Ma, J. Learning to cluster web search results. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2004), ACM, pp. 210–217.

[ZK04] Zitzler, E., and Künzli, S. Indicator-based selection in multiobjective search. In *Parallel Problem Solving from Nature (PPSN VIII)* (Berlin, Germany, 2004), X. Yao et al., Eds., Springer-Verlag, pp. 832–842.

[ZLT01] Zitzler, E., Laumanns, M., and Thiele, L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Tech. Rep. 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, May 2001.

[ZP01] Zacharis, N. Z., and Panayiotopoulos, T. Web search using a genetic algorithm. *IEEE Internet Computing 5*, 2 (2001), 18–26.

[ZT98] Zitzler, E., and Thiele, L. Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. In *Conference on Parallel Problem Solving from Nature (PPSN V)* (Amsterdam, 1998), pp. 292–301.

[ZT99] Zitzler, E., and Thiele, L. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation 3*, 4 (1999), 257–271.