Tesis defendida por

Marco Antonio Barbosa Santoyo

y aprobada por el siguiente Comité

Dr. Israel Marck Martínez Pérez
Director del Comité

M.C. José Luis Briseño Cervantes
Miembro del Comité

Dr. Miguel Ángel Alonso Arevalo
Miembro del Comité

Dr. José Antonio García Macías
Coordinador del programa de
posgrado en Ciencias de la Computación

Dr. Jesús Favela Vara
Director de la
Dirección de Estudios de Posgrado

Marzo de 2014

CENTRO DE INVESTIGACIÓN CIENTÍFICA Y DE EDUCACIÓN SUPERIOR DE ENSENADA, BAJA CALIFORNIA

Programa de Posgrado en Ciencias en Ciencias de la Computación

---

Aplicaciones de algoritmos evolutivos a finanzas

Tesis

que para cubrir parcialmente los requisitos necesarios para obtener el grado de

Maestro en Ciencias

Presenta:

Marco Antonio Barbosa Santoyo

Ensenada, Baja California, México

2014

Resumen de la tesis de Marco Antonio Barbosa Santoyo, presentada como requisito parcial para la obtención del grado de Maestro en Ciencias en Ciencias de la Computación.

Aplicaciones de algoritmos evolutivos a finanzas

Resumen aprobado por:

Dr. Israel Marck Martínez Pérez

Director de Tesis

Los mercados financieros, compuestos tradicionalmente por los mercados de valores y recientemente por los mercados de divisas, presentan un gran atractivo para el área de investigación, principalmente por el potencial de remuneración monetaria. Anteriormente se han usado métodos de cómputo natural para construir modelos que ayuden en las predicciones de cambios de precios y también para optimizar estrategias de inversión en estos mercados. Inicialmente se emplearon Redes Neuronales, principalmente para realizar modelado de series de tiempo como herramienta para realizar inversiones basadas en predicciones de las variaciones de los mercados. Recientemente se han tratado de aplicar diferentes enfoques de Cómputo Evolutivo, como son Programación Genética para modelado y Algoritmos Genéticos para evolución y optimización de estrategias de compra-venta. En este proyecto se propone desarrollar la implementación de un Algoritmo Genético para evolucionar los parámetros de una estrategia de inversión para el mercado de divisas basada en estrategias de ruptura, que son usadas frecuentemente por los inversionistas de este mercado.

Palabras Clave: **Algoritmos Genéticos, NSGA-II, SPEA2, Finanzas, Mercado de Divisas**

Abstract of the thesis presented by Marco Antonio Barbosa Santoyo, in partial fulfillment of the requirements of the degree of Master in Sciences in Computer Sciences.

Applications of evolutive algorithms to finance

Abstract approved by:

Dr. Israel Marck Martínez Pérez

Thesis Director

Financial markets, including stock and foreign exchange markets, are of high interest for research purposes mainly because of their potential for monetary reward. Natural computing methods have been previosly used to build models that aim to predict market price changes as well as to optimize investment strategies in such markets. Initiallly, neural networks were used as a tool for time series modelling in order to make investments based on predictions of market variations. Similarly, evolutionary computing techniques, such as genetic programming and genetic algorithms, have recently been used for modeling and optimization of trading strategies. In this thesis, a genetic algorithm for the optimization of a trading rule in the foreign exchange market is presented. The evolved parameters belong to a class of technical rules called "breakout strategies", which are frequently used by investors in the FOREX market.

Keywords: **Genetic Algorithms, NSGA-II, SPEA2, Finance, Foreign Exchange**

A mi madre, que me enseñó que con suficiente paciencia se puede hacer o deshacer cualquier cosa.

A mi padre, de quien aprendí que lo que más importa en la vida es lo que uno quiere hacer.

"...
I have lost the will to live
Simply nothing more to give
There is nothing more for me
Need the end to set me free

Things not what they used to be
Missing one inside of me
Deathly lost, this can't be real
Cannot stand this hell I feel

Emptiness is filling me
To the point of agony
Growing darkness taking dawn
I was me, but now he's gone
..."

Metallica – Fade to black

Agradecimientos

A todos los investigadores y personal de CICESE que siempre tienen buena disposición para orientar a los estudiantes.

A los miembros de mi comité de tesis: Dr. Israel Marck Martínez Pérez, M.C. José Luis Briseño Cervantes y el Dr. Miguel Ángel Alonso Arévalos, por su paciencia infinita durante el desarrollo de mi proyecto.

A los investigadores del Departamento de Ciencias de la Computación, particularmente los aquí mencionados que me dieron clase (en orden cronológico):

Dr. Pedro Gilberto López Mariscal , Dr. Vitali Kober, Dr. Hugo Homero Hidalgo Silva, Dr. Andrey Chernykh, Dr. José Alberto Fernández Zepeda, Dr. Carlos Alberto Brizuela Rodríguez, Dr. Jesús Favela Vara, Dr. Luis Adrián Castro Quiroa y Dra. María Isabel Ramírez Aguilar (ésta última del Departamento de Oceanografía Física).

A Rosa Carolina "Caro" Amador Tavarez y Lydia Salazar Ochoa, las "secres" por su apoyo moral y logístico a todos los estudiantes (e investigadores) del Departamento de Ciencias de la Computación.

Al M.C. Raúl Alberto Romero Wells y a Jorge Antonio Soria Solorio por el sus consejos técnicos, apoyo logístico en clases, exposiciones, etc.

A María Citlali Romero Manzano, Norma Alicia Fuentes Domínguez, Dolores Sarracino Ramírez y Lina Ivonne Best Guzmán que siempre tienen una sonrisa para atender a todos los estudiantes.

A la Lic. Lil Judith Bidart Escobar por su enorme paciencia durante las revisiones del formato de este documento y por su pronta respuesta a mis correos.

Al M.C. Héctor Zatarain Aceves por su **invaluable** ayuda para traducir mi programa de C# a Java, así como para depurar errores.

Agradezco a todos mis compañeras(os) y amigas(os) que no nombré en esta página, por sus diversos tipos de apoyo durante el desarrollo de este trabajo pero que menciono de manera no exhaustiva y cronológica en el apéndice D.

Al CONACyT por su apoyo económico durante el desarrollo de este proyecto.

Contenido

Página

Lista de Figuras

Figura Página

Figura Página

Lista de tablas

Tabla Página

Capítulo 1 Introducción

1.1 Antecedentes y motivación

La predicción de fenómenos dependientes del tiempo es importante en varios tipos de problemas del mundo real. Uno de ellos es el de los mercados financieros, donde existen recompensas monetarias altas para quien realice predicciones acertadas (Kaboudan, 2000; Santini y Tettamanzi, 2001). En particular, aunque la misma información está disponible para todos (típicamente precios anteriores), no todos pueden analizarla de manera útil, siendo posible obtener ganacias mientras el mercado ajusta sus precios, pues una fuente de información sin utilizar puede dar una ventaja a inversionistas (Larkin y Ryan, 2008).

La ingeniería financiera nos dice que los cambios de precios son intrínsicamente aleatorios en una situación ideal, lo cual nos lleva a creer en la imparcialidad de los mercados. Sin embargo, análisis numérico reciente, apoyado por el rápido avance de la tecnología de cómputo, ha propiciado un nuevo nivel de ingeniería financiera. Esencialmente, aplicaciones de técnicas computacionales nos dicen de la posiblidad de obtener una ganancia con un cierto grado de certidumbre (Tanaka-Yamawaki, 2004).

Los métodos de Cómputo Bioinspirados, como los Algoritmos Evolutivos (AE), pueden generar modelos para realizar predicciones que se ajustan bien a las series de tiempo (datos históricos), pero la calidad de las predicciones obtenidas de las muestras se degradan con el tiempo (Brabazon *et al.*, 2012), por lo que se pueden hacer predicciones a corto plazo para mercados financieros con estas heurísticas, pero no predicciones a largo plazo (Mishra *et al.*, 2010).

Los Algoritmos Genéticos (AG), un tipo de AE, se usan principalmente para resolver problemas de optimización (Eiben y Smith, 2003). En el área de finanzas, se han usado para tratar de hacer predicciones de precios en los mercados de valores (Drake y Marks, 1998) y para determinar reglas técnicas de compra-venta en dichos mercados (Allen y Karjalainen, 1999). Más recientemente, se han usado AGs para tratar de generar estrategias de inversión en el mercado de divisas y optimizar estrategias ya existentes, así como también explotar tendencias en estos mercados (Hryshko y Downs, 2003, 2004; Mendes *et al.*, 2012; Fernández-Pérez *et al.*, 2012). El objetivo de este proyecto es obtener una regla o estrategia técnica de compra-venta para el mercado de divisas usando AGs, que maximice rentabilidad minimizando el riesgo de inversión. En la siguiente sección se describirá el marco teórico necesario para desarrollar este proyecto.

1.2 Objetivos de la investigación

1.2.1 Objetivo general

Implementación de un algoritmo genético que optimice el desempeño de estrategias de rompimiento diario en el mercado de divisas, maximizando la rentabilidad y minimizando la pérdida mayor durante un periodo de tiempo dado para el tipo de cambio entre las de divisas euro y dólar americano (EURUSD).

1.2.2 Objetivos específicos

1. Implementar la estrategia de rompimiento para poder evaluar el rendimiento de una inversión que la utilice.

2. Hacer una implementación de los algoritmos NSGA-II y SPEA2 para optimizar estrategias de inversión considerando aspectos prácticos como costos de transacciones, e intereses diarios.

3. Maximizar la rentabilidad de la estrategia de compra-venta y minimizar el monto de mayor pérdida durante el periodo de inversión.

4. Implementar el algoritmo considerando la posibilidad de tener un rompimiento diario para la estrategia de compra-venta.

1.2.3 Definición del problema y pregunta de investigación

En el problema de optimización de parámetros para una estrategia de inversión se busca, partiendo de un capital inicial, realizar una serie de inversiones durante un periodo de tiempo finito para maximizar la ganancia acumulada y al mismo tiempo minimizar la pérdida más grande obtenida, de tal manera que no se ponga en riesgo dicho capital.

De aquí se deriva la siguiente pregunta que se trata de investigar con este proyecto:

¿Qué tan eficiente puede ser una estrategia de compra-venta basada en rompimiento generada por algoritmos evolutivos?

1.2.4 Planteamiento del problema.

La idea detrás de estos experimentos es tener una guía de inversión que permita tener ganancias minimizando las pérdidas que pongan en riesgo un capital inicial. Para esto se necesitan

los parámetros de inversión que se intentan encontrar con dichas simulaciones y ver que tan bien se hubieran comportado en una situación real.

Dado que el espacio de búsqueda para los mejores parámetros es extenso y su búsqueda requiere de una gran cantidad de tiempo de cómputo, se espera poder encontrar una combinación de parámetros que sean lo suficientemente buenos y tengan un desempeño aceptable dentro de un tiempo razonable, para poder utilizar la estrategia de inversión después de un periodo de entrenamiento, repetir el entrenamiento y volver a utilizar la estrategia de inversión con nuevos datos.

1.3 Metodología

Para cumplir con los objetivos y obtener una respuesta a la pregunta de investigación, se realizó la siguiente metodología de 3 fases: implementar la función objetivo, utilizar la función objetivo para realizar un entrenamiento con una base de datos que representa un periodo de tiempo y finalmente utilizar la misma función objetivo para hacer una evaluación de los mejores resultados obtenidos en un periodo de tiempo que siga inmediatamente al usado como entrenamiento.

Primero se plantean los objetivos a optimizar, que son la ganancia y la pérdida mayor. Esto nos indica la manera de hacer la implementación de la estrategia de inversión como una función objetivo para obtener las medidas de evaluación de sus parámetros.

Posteriormente se implementó dicha función objetivo para evaluar el desempeño de la estrategia de inversión con un conjunto de parámetros en un periodo de tiempo dado por una base de datos de precios del tipo de cambio EURUSD.

Después se busca optimizar los parámetros de la función objetivo mediante el uso de algoritmos genéticos. Para esto se realizó dicha implementación para evaluar el desempeño de la estrategia de inversión mediante un "entrenamiento" usando la función objetivo para variar los parámetros y tratar de identificar los mejores con una base de datos de un periodo de tiempo dado para un tipo de cambio de una divisa, en este caso EURUSD del año 2011, con una granularidad de datos de 5 minutos. Esta variación se llevó acabo usando los Algoritmos Genéticos NSGA-II y SPEA2. Cada evaluación dio como resultado la ganancia acumulada y la pérdida mayor en dicho periodo para los parámetros usados.

Los mejores individuos de este entrenamiento, según los dos criteros a optimizar, que constituyen frentes de pareto de las diferentes modalidades de los experimentos, se mezclaron

para obtener los mejores individuos de todos estos en un frente de pareto constituído por elementos de todos los frentes anteriores. Se seleccionó una muestra representativa de este frente y sus individuos fueron evaluados con una base de datos para diversos periodos del año 2012, todos inmediatamente posteriores al periodo de entrenamiento, para el tipo de cambio EURUSD con una granularidad de 5 minutos. Para esta evaluación se obtuvo tambien una ganancia y una pérdida más grande para el periodo.

En base a los resultados obtenidos en la evaluación se trata de determinar que tipo de algoritmo genético puede ser mejor para el entrenamiento y con cuales parámetros, para tratar de optimizar el uso de recursos de cómputo utilizados para esta estrategia de inversión.

1.4 Organización de la tesis

A continuación se describe brevemente el contenido de los siguientes capítulos: En el Capítulo 2 se describen de manera general los conceptos de Algoritmos Genéticos y Finanzas necesarios para entender el contenido de la tesis, así como un resumen de los trabajos de investigación más relevantes relacionados con esta área.

En el Capítulo 3 se describen los requerimientos técnicos necesarios para llevar a cabo los experimentos. En el Capítulo 4 se mencionan algunas de las consideraciones teóricas y prácticas para el diseño y la implementación de los experimentos y posteriormentse se hace una descripción de los experimentos y los resultados obtenidos, empezando por la fase de entrenamiento y finalmente la fase de evaluación de los “mejores resultados”. Finalmente, en el último capítulo se plantea una discusión de los resultados y las conclusiones obtenidas. También se mencionan algunas mejoras técnicas que podrían influir positivamente sobre una futura repetición de los experimentos.

Capítulo 2 Marco teórico

En este capítulo se hará una introducción a los diferentes temas necesarios para el desarrollo de este proyecto. Se empezará por el tema de Cómputo Natural para ubicar el lugar que tienen los Algoritmos Genéticos dentro de esta área y se irá profundizando en el tema hasta abordar los algoritmos NSGA-II y SPEA2 que serán usados en este trabajo. Posteriormente se abordará el tema de Finanzas para ubicar el área del Mercado de Divisas y las consideraciones necesarias para desarrollar estrategias de inversión, así como algunas herramientas que se usan con este fin, para llegar por último a la estrategia de ruptura, que es la herramienta que se implementará como parte de este proyecto.

2.1 Cómputo bioinspirado

En esta sección se mencionan los antecedentes necesarios para ubicar a los algoritmos genéticos dentro de la rama de Ciencias de la Computación conocida como Algoritmos Evolutivos. Primero se describe el área de Cómputo Evolutivo. Posteriormente se hace una distinción entre algoritmos para optimización mono-objetivo y algoritmos para optimización multi-objetivo. Finalmente se mencionan los algoritmos SPEA2 y NSGA-II, que son algoritmos genéticos de optimización multiobjetivo.

2.1.1 Cómputo evolutivo

El Cómputo Evolutivo (CE) es una área de investigación de las Ciencias de la Computación que se caracteriza por estar inspirado en la evolución natural, y cuyo proceso simplificado se describe a continuación: en un ambiente existe una población de individuos que luchan por sobrevivir y reproducirse. Los individuos son evaluados en su ambiente para determinar una aptitud, la cual determina si sobreviven, y si se reproducen para obtener así una nueva generación de individuos. La reproducción emplea una operación de cruzamiento entre individuos de la población, tomando partes de cada uno para producir individuos nuevos que heredan así características de los padres. Posteriormente se puede utilizar una operación de mutación sobre los hijos para introducir cambios que no serían heredados (Eiben y Smith, 2003).

En la Figura 1 se muestra el diagrama general de un algoritmo evolutivo. Observe que dependiendo del tipo de algoritmo las operaciones de cruzamiento o mutación pueden ser opcionales, y los criterios de terminación pueden variar, así como los criterios para la selección

de padres y sobrevivientes, como se explicaá más adelante.

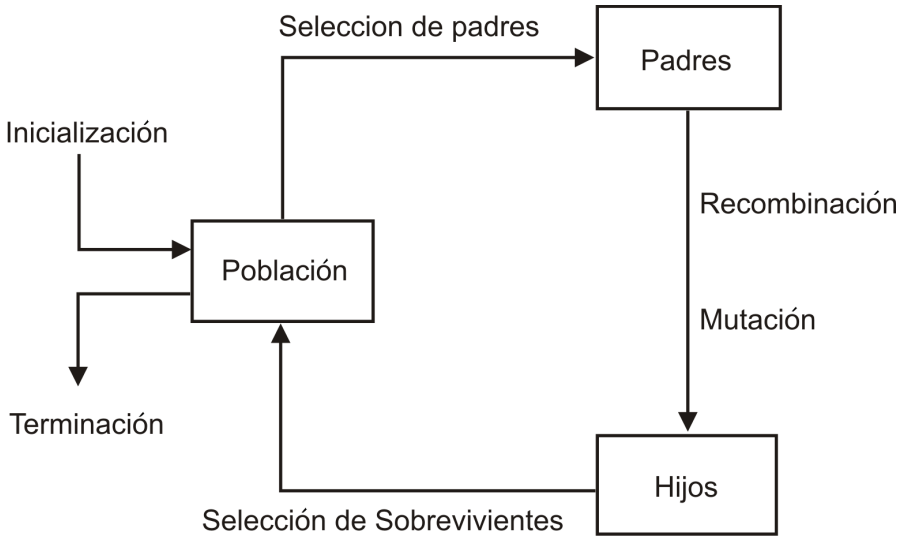


Figura 1: Esquema general de un algoritmo evolutivo.

Se dice que un algoritmo es elitista si los mejores individuos de una generación se conservan para la siguiente. Cada población se transformará de manera estocástica, generación por generación, para obtener nuevas e idealmente mejores poblaciones. Como en la naturaleza, este es un proceso aleatorio que no garantiza resultados. Pero por su misma aleatoriedad puede hacer que evada trampas (en forma de máximos o mínimos locales) que pueden atrapar a métodos determinísticos (Poli *et al.*, 2008).

En el escenario anterior, el ambiente representa el problema a resolver, un individuo de la población representa una posible solución y la aptitud representa la calidad de la solución (Eiben y Smith, 2003). El problema por lo general es la búsqueda de soluciones, que puede ser demasiado difícil (tardada) cuando el espacio de soluciones es grande y una búsqueda exhaustiva es prácticamente imposible (Chen y Yeh, 1997). El CE se aplica con éxito cuando el tamaño del espacio de soluciones dificulta la búsqueda para otros métodos. Un inconveniente es que no hay un resultado general que garantice la convergencia de CE hacia un óptimo global, pero puede encontrar una solución aceptable relativamente rápida para muchos problemas (Allen y Karjalainen, 1999).

Las diferentes subáreas del CE se caracterizan por las maneras en que implementan la estrategia de evolución, representan a sus poblaciones de inviduos y los tipos de problemas que atacan. Estas son: Programación Genética, Estrategias Evolutivas y Algoritmos Genéticos (Eiben y Smith, 2003).

La Programación Genética (PG) se usa por lo general para resolver problemas de mod-

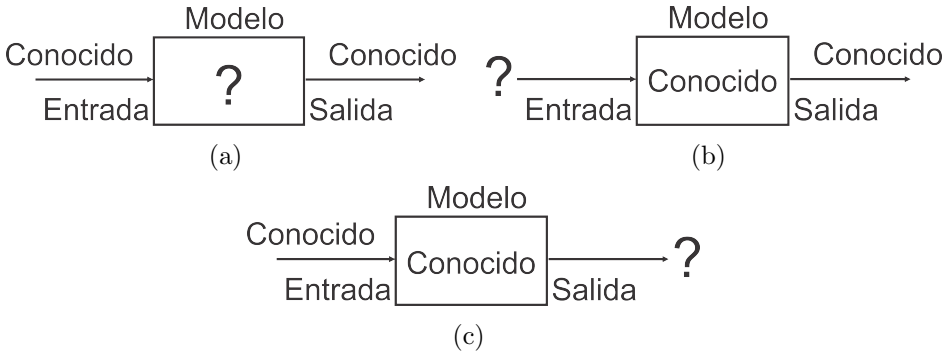


Figura 2: Diferentes tipos de problemas. a) Problema de modelado. b) Problema de optimización. c) Problema de simulación.

elado usando principalmente cruzamiento sobre sus poblaciones. Las Estrategias Evolutivas usan principalmente mutación sobre sus poblaciones para resolver problemas de optimización continua. Los Algoritmos Genéticos por lo general usan cruzamiento como herramienta principal y mutación como herramienta secundaria en su estrategia de evolución. Usan cadenas basadas en un alfabeto finito como individuos de su población y son más utilizados para resolver poblemas combinatorios de optimización (Eiben y Smith, 2003).

Brevemente se mencionará la diferencia entre un problema de optimización y uno de modelado. Suponga que se tiene un sistema con una entrada de datos, una operación que se aplica a los datos, y una salida como resultado de dicha operación. Dependiendo del componente que se desconozca en el sistema será el tipo de problema, como se muestra en la Figura 2. Si se desconoce la entrada se tiene un problema de optimización (Figura 2(b)). Si se desconoce la operación, se tiene un problema de modelado (Figura 2(a)). Y finalmente, si se desconoce la salida, se tiene un problema de simulación (Figura 2(c))(Eiben y Smith, 2003).

2.1.2 Algoritmos genéticos

El algoritmo genético fue originalmente concebido por Holland como un medio para estudiar comportamiento adaptativo. Sin embargo, ha sido considerado principalmente como método de optimización de funciones. El primer algoritmo genético diseñado tiene las siguientes características: representación binaria, selección proporcional a su aptitud, una baja probabilidad de mutación y un énfasis en recombinación genéticamente inspirada como medio de generar nuevas soluciones posibles. Esto se conoce como *Algoritmo Genético Simple* o *Algoritmo Genético Canónico* (Eiben y Smith, 2003).

El primer paso para construir un Algoritmo Evolutivo es decidir una representación para la posible solución al problema que se quiere resolver. Esto involucra definir un genotipo y un mapeo de genotipo a fenotipo. Escoger el tipo de repesentación que se usará es una de las partes más difíciles del diseño de un algoritmo evolutivo (Eiben y Smith, 2003).

Existen una variedad de representaciones posibles. La primera de ellas es la representación binaria. Consiste en hacer que una cadena de 0's y 1's sea la representación de cada posible respuesta. Otro tipo de representación es la entera, en la que se usan cadenas de números enteros para representar cada una de nuestras posibles respuestas. De la misma manera, se pueden utilizar cadenas de valores reales para representar las respuestas. Esto es más conveniente cuando los valores que queremos representar como genes vienen de distribuciones continuas en vez de distribuciones discretas. Existen también representaciones de permutaciones, donde una posible respuesta sería la manera en que una serie de eventos podría ocurrir. En esta situación no se pueden tener elementos repetidos en la cadena. En la Figura 3 se muestra un ejemplo de un genotipo homogéneo, todos los elemenos son de tipo flotante.

Figura 3: Ejemplo de un genotipo homogéneo. Todos los elementos son de tipo flotante.

Una vez que se ha decidido sobre el tipo de representación para las posibles respuestas, se puede proceder a desarrollar operaciones de mutación y/o cruzamiento que sean adecuadas para esta representación. La operación de mutación se refiere al tipo de operaciones con las que se genera un nuevo elemento usando solo un elemento padre. Generalmente requiere perturbar un solo elemento de la cadena para realizar su objetivo. Un ejemplo sería cambiar un bit de 0 a 1 o de 1 a 0 en el caso de una representación binaria, o incrementar o decrementar un elemento en caso de una representación entera. En la Figura 4 se ilustra como 3 genes sombreados con gris serían mutados por esta operación.

Las operaciones de cruzamiento involucran crear elementos nuevos partiendo de dos o más elementos padres. La estrategia general implica tomar una subcadena de un genotipo padre y el material faltante de otro padre diferente para generar un elemento hijo que tenga características de los dos padres. Existen varias estrategias para realizar la selección de los elementos que se copian de un padre a un hijo. En el cruzamiento de 1-punto se selecciona un punto al azar y la cadena desde el inicio hasta ese punto se copia de un padre, y otra cadena, partiendo de ese punto hasta el final, se copia de otro padre, para generar un nuevo hijo. En

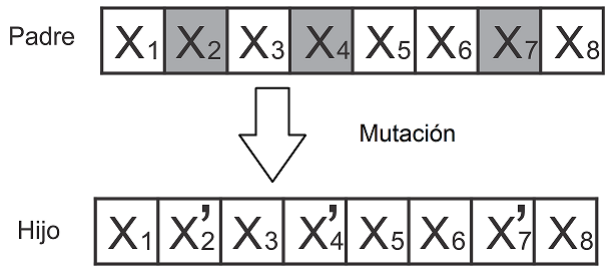


Figura 4: Ejemplo de mutación en cromosomas. Los elementos sombreados son los que serán sometidos a la operación de mutación.

el cruzamiento uniforme cada elemento del genotipo de un padre tiene la misma probabilidad de ser copiado o no hacia un hijo. En caso de que no sea copiado, entonces ese elemento se obtiene del otro padre. En la Figura 5 se ilustra como se obtendrían dos hijos a partir de dos padres, definiendo primero puntos de corte y luego usando los elementos acotados de esa manera para generar nuevos elementos. Los elementos sombreados pertenecen a un padre y los claros a otro.

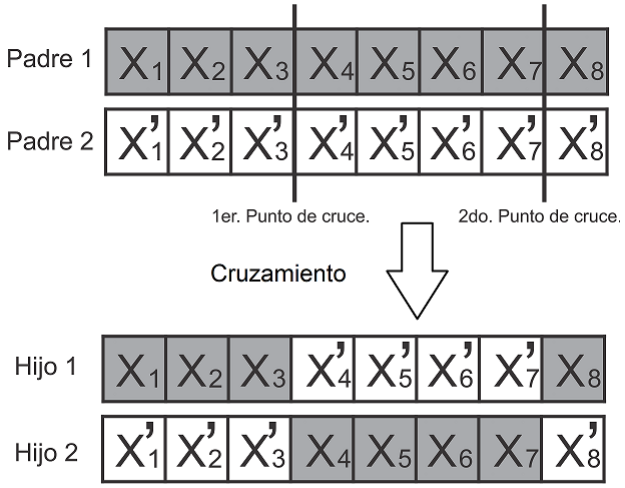


Figura 5: Ejemplo de cruzamiento de 2 puntos en cromosomas. Los elementos sombreados proceden de un padre. Los elementos claros proceden de otro padre. Las líneas punteadas representan los puntos de cruce.

La población de posibles respuestas puede ser manejada de varias maneras. En la literatura se manejan dos principales que son: El reemplazo generacional y el modelo de estado estacionario. En el modelo generacional cada generación de hijos nuevos reemplaza completamente a la generación de padres. En el modelo de estado estacionario, solo algunos elementos son reemplazados.

Para seleccionar los padres existen varias estrategias. Todas ellas involucran asignar

diferentes probabilidades de selección. En la estrategia de selección proporcional se le asigna una probabilidad de selección a un individuo relativa a su aptitud comparada con la aptitud de toda la población. En este caso, los individuos que sobresalgan de la población tienen la tendencia a dominar la población rápidamente.

Una estrategia para combatir este problema se denomina Selección por Clasificación, donde los individuos se ordenan de acuerdo a su aptitud y se les asigna probabilidad de selección dependiendo del rango en vez de usar solamente su valor de aptitud.

Una manera de implementar estas probabilidades de selección es por medio de una *ruleta*, donde el resultado sería el mismo que girar una ruleta de un brazo y el tamaño de los agujeros sería relativo a las probabilidades de selección. Otra manera sería hacer un torneo de selección, donde un inviduo compite contra otro u otros para ser seleccionado. De esta manera no es necesario evaluar la aptitud de todos los individuos de antemano y solo se evalúan aquellos que formen parte del torneo.

Una vez que se tienen los padres y los hijos, se seleccionan aquellos sobrevivientes que serán parte de la siguiente generación. Existen varias estrategias para realizar dicha selección. El reemplazo basado en edad se basa en definir un periodo máximo en el que un individuo pueda permanecer como parte de la población. Al final de este periodo deberá ser reemplazado por otro individuo. En el algoritmo genético simple se sigue esta estrategia donde los padres duran una sola generación como parte de la población y luego son reemplazados completamente por los hijos.

Tambien se pueden reemplazar basados en la aptitud. Dos estrategias principales existen en este enfoque: Reemplazar los peores, donde se toman los peores elementos de la población y se reemplazan por otros elementos con mejor aptitud. La otra estrategia es *Reemplazo por elitismo*, donde el mejor individuo siempre se mantiene como parte de la población y en caso de que un hijo no sea igual o mejor, se descarta el hijo.

De manera general esta ha sido la descripción del funcionamiento de los algoritmos genéticos. Se puede consultar la referencia de Eiben y Smith (2003) para más detalles.

2.1.3 Algoritmos evolutivos de optimización multiobjetivo (AEOM)

La meta de un Algoritmo Evolutivo (AE) en un problema de optimización es encontrar una solución que maximice una función de aptitud directamente relacionada a una medida de calidad. Un problema donde la calidad de una solución se define en relación a varios objetivos, generalmente en conflicto, se dice que es un *problema multiobjetivo*. Un algoritmo

de optimización multiobjetivo trata de presentar un conjunto diverso de soluciones posibles, con diferentes compromisos entre los objetivos (Eiben y Smith, 2003).

Dadas dos soluciones, las cuales se califican de acuerdo a un conjunto de valores objetivo, una de ellas se dice que domina a la otra si su calificación es al menos igual de alta para todos los objetivos y es estrictamente mejor para al menos uno. Para objetivos en conflicto, no existe una solución única que domine a las demás. Se dice que solución es *no dominada* si no es dominada por ninguna otra. El conjunto de soluciones no dominadas se llama "Frente de Pareto" o "Conjunto de Pareto" (Eiben y Smith, 2003).

En la Figura 6 se muestra un frente de Pareto para el caso en que se intenten minimizar dos objetivos. Los elementos unidos con la línea roja representan el conjunto de Pareto.

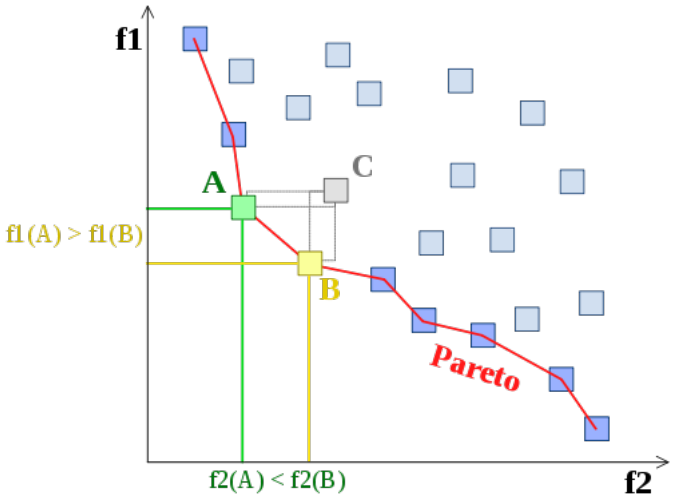


Figura 6: Frente de Pareto obtenido por la minimización de 2 objetivos: f1 y f2. El frente de Pareto es el conjunto de soluciones que domina a las demás. Obtenido de http://en.wikipedia.org/wiki/File:Front_pareto.svg.

El algoritmo NSGA es un enfoque no elitista de un AEOM. Asigna aptitud basada en dividir a la población en frentes de igual dominancia. El algoritmo busca todos los puntos no dominados en la población que no estén etiquetados como pertenecientes a un frente previo. Luego etiqueta el nuevo conjunto como perteneciente al nuevo frente actual e incrementa el contador de frentes, repitiendo hasta que todas las soluciones han sido etiquetadas. Cada punto de un frente dado obtiene como aptitud el número de todas las soluciones en frentes inferiores (Eiben y Smith, 2003).

El algoritmo NSGA-II es un enfoque elitista de AEOM, el cual se describe con más detalle en la siguiente sección. Tiene un criterio de selección basado en el operador de comparación de multitudes. Todos los individuos de un frente se ordenan de acuerdo a un rango basado

en medida de multitud que es igual a la suma de las distancias de los dos individuos más cercanos de cada objetivo (Mishra *et al.*, 2010).

2.1.4 NSGA-II

El algoritmo NSGA fue propuesto por Srinivas y Deb en 1995. La complejidad computacional del NSGA simple es O(MN^3) donde M es el número de objetivos y N es el tamaño del conjunto de datos. Posteriormente Deb y sus estudiantes (Deb *et al.*, 2000) desarrollaron una variante del NSGA, que llamaron NSGA-II con complejidad O(MN^2).

A grandes rasgos el algoritmo NSGA-II funciona de la siguiente manera: De una población N de individuos, donde cada individuo representa una posible respuesta, se generan otros N individuos hijos. Para esto se usan los operadores usuales: cruzamiento de 1-punto y mutación. De la mezcla de estas dos poblaciones, ahora de tamaño $2N$ se toman los mejores N individuos de acuerdo con la función de aptitud y una ordenación para dividir la población en frentes no dominados. Se toman primero los individuos de los mejores frentes no dominados, uno por uno, hasta que se completan N individuos. En caso de que no haya espacio para un frente completo en la nueva población, se toma en cuenta un operador de distancia de amontonamiento para escoger individuos de las regiones del frente con menos individuos (D'Souza *et al.*, 2010).

2.1.5 SPEA2

El algoritmo SPEA (*Strength Pareto Evolutionary Algorithm*) fue desarrollado por Zietzler y Thiele en 1998. SPEA usa una clasificación basada en un archivo externo de soluciones no-dominadas y usa técnicas de agrupación (*clustering*) para truncar la población externa. SPEA2 trata de mejorar SPEA tomando en cuenta cuantos individuos domina cada uno y por cuantos individuos es dominado. Además, usa una medida de densidad para discriminar entre soluciones con el mismo rango, donde la densidad de una solución se define como el inverso de la distancia a su k-ésimo vecino más cercano en el espacio de la función objetivo, y un método nuevo para truncar el archivo garantiza que se preserven soluciones en los extremos (Zitzler *et al.*, 2001).

2.2 Introducción a finanzas

En esta sección se dará una breve introducción a los antecedentes en el área de finanzas necesarios para entender la parte relacionada con inversiones en el mercado de divisas. Se abordarán temas como mercados financieros y sus funciones, así como las técnicas de análisis de dichos mercados que son necesarias para construir estrategias de inversión.

2.2.1 Finanzas

Finanzas es la ciencia del capital. Como tal, finanzas es un híbrido entre las áreas de economía, estadística y contabilidad. De la Economía se toma la forma de pensar en costos y beneficios. De la estadística se toman los métodos para cuantificar incertidumbre. Y de la contabilidad se toman los métodos para extraer información de los estados financieros de una compañía. Las finanzas se dividen en personales y empresariales (Welch, 2004).

En este documento nos referiremos al área "personal" de las finanzas, donde una persona interactúa con un mercado financiero para comprar o vender un producto (Pilbeam, 2005).

La *hipótesis del mercado eficiente* dice que los mercados financieros son "eficientes en cuanto a información", esto es, los precios de los valores reflejan inmediatamente toda la información pertinente conocida, tal que no es posible ganarle al mercado usando información que ya es conocida por este. Sin embargo, la predicción de precios de mercados financieros ha sido por mucho tiempo una área atractiva para realizar investigación (Larkin y Ryan, 2008).

2.2.2 Mercados financieros

Un mercado financiero es un mecanismo que permite comprar y vender instrumentos financieros, productos y otros artículos con bajos costos de transacción y a precios que reflejan la *hipótesis del mercado eficiente*. Existen diferentes tipos de mercados financieros, además de los mercados de valores y de divisas (Pilbeam, 2005).

El mercado financiero más conocido es el *mercado de valores*. Ahí se venden acciones de compañías públicas. Una acción es una hoja de papel que promete a su dueño una fracción de las ganancias de dicha compañía (Welch, 2004). El *mercado de divisas* es el mercado con mayor volumen de transacciones en el mundo. Como su nombre lo indica, en este mercado se realiza la compra y venta de divisas (Galant y Dolan, 2007). Existen otros mercados como el mercado de dinero y el de derivados (Pilbeam, 2005).

En este documento se hará distinción entre el *mercado de divisas* y los demás tipos

de mercados, los cuales se mencionarán como *mercados de valores*. El término *mercados financieros* se usará para englobar a estos dos tipos de mercados.

2.2.3 Predicción de mercados

Hay dos tipos de enfoques para la predicción financiera: análisis técnico y análisis fundamental. En el análisis técnico se usan precios pasados en forma de histogramas (series de tiempo) como el de la Figura 7, para predecir precios (Hirabayashi *et al.*, 2009). En el análisis fundamental se usa información global de la compañía tales como estados financieros y otros factores relacionados a su ambiente para juzgar el valor de su acciones (Tokuoka y Tanaka-Yamawaki, 2008).

Figura 7: Comportamiento típico del mercado de divisas. Se muestra el comportamiento del tipo de cambio USDEUR del año 2007 al 2010. Tomado de http://www.oanda.com/currency/historical-rates/.

Ambos enfoques tratan de resolver el mismo problema: determinar en que dirección se van a mover los precios. La diferencia entre el análisis fundamental y el técnico es que el primero explora las causas de los movimientos del mercado y el segundo explora el efecto de estos movimientos (Hryshko y Downs, 2004).

Analistas que están involucrados profesionalmente en predicción de valores usan datos de análisis fundamental y técnico. El primero considera razones objetivas (fundamentales) para el cambio de precios, como son indicadores económicos o la situación política a nivel país y compañía. El segundo se basa en transformaciones específicas e interpretación visual de los datos anteriores del mercado, especialmente precios y volumen (Mańdziuk y Jaruszewicz, 2011).

Los inversionistas pueden usar cualquiera o ambos de estos métodos diferentes pero de cierta manera complementarios para escoger valores. Por ejemplo, muchos inversionistas fundamentales usan indicadores técnicos para decidir puntos de entrada y salida. Muchos inversionistas técnicos usan análisis fundamental para limitar su universo de posibles opciones de valores (Contreras *et al.*, 2012).

2.2.4 Análisis fundamental

El *análisis fundamental* se concentra sobre las fuerzas económicas de oferta y demanda para determinar si los precios van a subir, bajar, o permanecer casi iguales. Examina todos los factores relevantes que afectan el precio para determinar el valor intrínseco del mercado. Valor intríseco significa valor basado en la ley de oferta y demanda (Hryshko y Downs, 2004).

El análisis fundamental mantiene que los mercados pueden valorar de manera errónea algún bien a corto plazo, pero que el precio "correcto" eventualmente se va a alcanzar. Se pueden obtener ganancias haciendo operaciones de compra-venta sobre el bien valuado de manera incorrecta y esperando a que el mercado reconozca su "error" y re-valúe dicho bien (Contreras *et al.*, 2012). El análisis fundamental usa información y noticias sobre la situación económica y política del entorno interno y externo que afecta el valor de lo que se intenta comprar o vender. Pueden ser reportes económicos, tasas de interés, políticas monetarias, flujos de comercio internacional y flujos de inversión (Galant y Dolan, 2007). El análisis fundamental de un negocio o empresa requiere analizar sus estados y salud financieros, sus ventajas administrativas y competitivas, así como sus mercados y competidores. Cuando se aplica a bonos y divisas, se concentra en el estado general de la economía (tasas de interés, producción, etc.) de un país. Cuando se trata de acciones de compañías se concentra sobre la información contable de ellas (Contreras *et al.*, 2012).

El análisis fundamental generalmente se preocupa por tendencias a mayor largo plazo que el análisis técnico, así que dado que la mayor parte de los especuladores en FOREX realizan operaciones intradiarias, éstas solamente usan análisis técnico y no toman en consideración el método fundamental (Hryshko y Downs, 2004).Algunos indicadores fundamentales son: el flujo de precio en efectivo, crecimiento de ventas, valor del precio de libros, y la ganancia sobre bienes.

2.2.5 Análisis técnico

El análisis técnico dice que toda la información ya está reflejada en el precio de un bien. La tendencia es tu "amiga" y los cambios en los sentimientos preceden y predicen cambios en las tendencias. Las respuestas emocionales de los inversionistas a los cambios de precios llevan a patrones reconocibles en los diagramas de precios. Al análisis técnico no le importa cual es el "valor" de un bien. Sus predicciones de precios son solo extrapolaciones de patrones de precios históricos (Contreras *et al.*, 2012). Prácticamente el análisis técnico intenta predecir precios del mercado examinando patrones de compra-venta y comparando las tendencias de los diagramas actuales con los del pasado (Hryshko y Downs, 2004).

El análisis técnico es una colección amplia de métodos y estrategias que intenta explotar fluctuaciones en los mercados de valores. En este enfoque se generan reglas de compra-venta a partir de datos históricos, para determinar el momento apropiado para comprar y vender los activos. Los especuladores usan estas reglas de compra-venta para obtener ganancias de operaciones frecuentes en el mercado. A este enfoque se le llama estrategia de compra-venta. Los especuladores creen que de esta manera pueden obtener más ganancias que los "inversionistas", que hacen compras y las mantienen a plazo más largo (Esfahanipour y Mousavi, 2011).

Los operadores que basan sus decisiones en análisis técnico son conocidos como *técnicos*. Los *técnicos* se dividen en dos grupos: "cartistas" tradicionales y técnicos estadísticos. Para un cartista, el análisis de gráficas es su herramienta principal de trabajo. El análisis de gráficas (*charting*) sigue siendo mayormente subjetivo y el éxito de este enfoque depende de la habilidad de cada cartista. En contraste, el técnico estadístico aborda el mercado en términos de desarrollar un sistema de compra-venta mecanizado. Dichos sistemas luego son programados en computadoras que generan los consejos de compra-venta mecánicos para "comprar" o "vender" (Hryshko y Downs, 2004).

El análisis técnico es importante por varias razones. El análisis fundamental puede proveer una estimación de las situaciones de oferta/demanda, razones de precio/ganancia, estadísticas económicas y más, pero no tiene manera de medir la parte "emocional" de los mercados. La *caminata aleatoria* dice que los precios de un mercado de un día no tienen que ver con los precios del siguiente día. Pero este punto de vista académico no toma en cuenta un componente importante – la gente. La gente recuerda precios de un día anterior y actuan en base a ello. Además, el comportamiento de los precios es la manera más sencilla y directa

de ver las relaciones de oferta/demanda. Y en base a esto se puede establecer una estrategia disciplinada para realizar compras o ventas. Por estas razones, los indicadores técnicos no se deben ignorar aunque no se crea en su utilidad (Nison, 1991).

Los técnicos usan diagramas para buscar patrones típicos, como son el bien conocido patrón de cabeza y hombros, o inversión de doble tapa/fondo; estudian indicadores técnicos, medias móviles y buscan figuras como líneas y soportes, resistencias, canales y formaciones más obscuras como banderas, estandartes y patrones de tazas y asas. También usan indicadores de mercados de muchos tipos, algunos de los cuales son transformaciones matemáticas de precios, incluyendo muchas veces volumen a la alza y baja, así como otros datos de entrada. Estos indicadores son usados para ayudar a determinar si un bien va tener una tendencia, y si es así, la probabilidad de su dirección y continuación. También buscan relaciones entre indíces de precio y volumen e indicadores de mercado (Contreras *et al.*, 2012). Algunos de ellos son el índice de fuerza relativa, promedios móviles, niveles de soporte y resistencia, divergencia del volumen de precio, etc.

2.2.6 Velas japonesas

Las gráficas de velas tienen su origen en los mercados de arroz japoneses. Originalmente se intercambiaban y vendían pacas de arroz, pero después se vendía recibos o cupones de pacas de arroz que equivalían a un producto ya existente, o la promesa de un producto que se cosecharía en algún momento futuro. Un mercader muy exitoso llamado Munehisa Homma, nacido en el siglo XVIII en una familia adinerada, amasó gran fortuna en estos mercados de arroz. Sus principios de compra-venta aplicados a estos mercados, evolucionaron a lo que es la metodología de *velas japonesas* actualmente usada en Japón (Nison, 1991).

En la Figura 8(a) se muestra una gráfica de precios para el periodo de las 20 horas a las 14 horas para un tipo de cambio, usando velas japonesas y en la Figura 8(b) se muestra la misma gráfica usando barras.

Las velas japonesas son nuevas para la mayoría de los técnicos occidentales quiénes por lo general usan la gráfica de barras. Se utilizan los mismos datos para construir ambas gráficas: precios de apertura, máximo, mínimo, y cierre. En una gráfica de barras la línea vertical define el máximo y el mínimo. La línea horizontal a la izquierda de la línea vertical es el precio de apertura. La línea horizontal a la derecha de la línea vertical es el precio de cierre (Nison, 1991).

Aunque las gráficas de barras y las de velas usen los mismos datos se dibujan diferente,

(a) (b)

Figura 8: a) Gráfica de velas japonesas. b) Gráfica de barras. Tomado de https://fxtrade.oanda.co.uk/demo-account/trade.

como se puede ilustrar en la Figura 9(a). En la Figura 9 se muestran los mismos componentes para las gráficas de barras, pero observe que al comparar las gráficas de las Figuras 8(a) y 8(b) se puede apreciar con mayor facilidad los movimientos de precios por el color de las velas.

La parte ancha de la vela es llamada el *cuerpo real*. Representa el rango entre la apertura y cierre de la sesión. Cuando el cuerpo real es negro (relleno) significa que el cierre de la sesión fue más bajo que la apertura. Si el cuerpo real es blanco (vacío), significa que el cierre fue más alto que la apertura (Nison, 1991).

(a) (b)

Figura 9: a) Partes de las velas japonesas. b) Partes de las barras.

Las líneas delgadas encima y debajo del cuerpo real son las *sombras*. Estas sombras representan los precios extremos de la sesión. La sombra encima del cuerpo real es llamada la *sombra superior* y la sombra bajo el cuerpo real es conocida como la *sombra inferior*. De manera similar, la cima de la sombra superior es el máximo de la sesión, y el fondo de la

sombra inferior es el mínimo de la sesión. Es fácil ver porque a esto se la llama gráfica de velas pues las líneas individuales con frecuencia parecen velas con mechas. Una vela sin sombra superior se dice que tiene una *cabeza rasurada*. Una vela sin sombra inferior se dice que tiene una *cola rasurada*. Para los japoneses el cuerpo real es el movimiento esencial del precio. Las sombras son consideradas generalmente como fluctuaciones de precio sin relevancia (Nison, 1991).

2.2.7 Indicadores técnicos

Hay dos tipos de indicadores de mercados: los indicadores técnicos y los indicadores fundamentales. Los indicadores fundamentales se basan en análisis fundamental y hacen uso de diferentes factores relacionados con un activo. Un ejemplo de estos indicadores fundamentales es el *Crecimiento de ventas* que se enfoca en la salud financiera de una compañía y se calcula con la siguiente fórmula por sus siglas en inglés ("Sales Growth"): $SG_t = \frac{REVTQt-1}{REVTQ_{t-2}} - 1$ y que se interpreta como: el crecimiento de un periodo t es igual a dividir las ganancias del periodo, por sus siglas en inglés ("REVenue Total of Quarter") $t-1$, entre las ganancias del periodo $t-2$ y a ese resultado restarle 1.

Los indicadores técnicos en su mayoría son transformaciones matemáticas de los precios históricos de algún activo disponible en un mercado. Estos indicadores incluyen frecuentemente volumen de subida o bajada, datos de avance o retroceso y otras entradas. Dichos indicadores se usan para determinar si un activo tiene alguna tendencia, y si la tiene, que probabilidad tiene de tomar alguna dirección y de que continúe en ella. Los inversionistas técnicos tambien buscan relaciones entre índices de precio y volumen e indicadores de mercado.

Un par de ejemplos de estos indicadores técnicos son la *media móvil* y el *índice de fuerza relativo*. La media móvil es un promedio de los n precios anteriores de un activo disponible en un mercado y se expresa por la siguiente fórmula a partir de sus siglas en inglés (" Moving Average"): $MA_n = \frac{PRCCM_t+PRCCM_{t-1}+...+PRCCM_{t-(n-2)}+PRCCMt-(n-1)}{n}$. El índice de fuerza relativo, por sus siglas en inglés "RSI", intenta graficar la fuerza o debilidad de un activo en un mercado basándose en los precios de cierre de un periodo de compra-venta reciente. Se calcula con la siguiente fórmula: $RSI = 100 - \frac{100}{1+RS}$. La fuerza relativa (RS), se calcula en base a la razón de dos medias móviles : una para los precios a la alza, y otra para los precios a la baja con la siguiente fórmula: $RS = \frac{\sum_{i=1}^{m} UPCP_i}{\sum_{i=1}^{m} DWCP_i}$.

Para mayor información respecto a los indicadores de mercado consultar Contreras *et al.*

(2012).

2.2.8 Estrategias de compra-venta

Una estrategia de compra-venta (ECV) es un conjunto de reglas objetivas que definen las condiciones que se deben cumplir para que se ejecuten entradas y salidas a inversiones (Mehta *et al.*, 2011). Una regla de compra-venta (RCV) es una función que regresa una señal de "comprar" o "vender" a partir del historial de un precio. La ECV especifica la posición a tomar, dada la posición actual y la señal de la RCV (Allen y Karjalainen, 1999). Una RCV puede incluir indicadores fundamentales o técnicos para generar las señales de compra o venta. Por ejemplo, el indicador fundamental mencionado anteriormente de crecimiento de ventas puede influir positivamente o negativamente en una ECV cuando forma parte de una RCV específica (Contreras *et al.*, 2012).

Existen gran variedad de estrategias de compra-venta, entre ellas, la compra-venta técnica o estrategia técnica, la cual usa gráficas y diagramas para identificar señales de convergencia o divergencia que pueden indicar cuando comprar o vender. Dichas gráficas se basan en indicadores técnicos que al tener un comportamiento en particular generan "señales" para comprar o vender. Algunas estrategias técnicas de compra-venta manejan el concepto de rompimiento de canal. Un "canal" se delimita por al menos 2 máximos y 2 mínimos de precios. El rompimiento se lleva a cabo cuando la variación de precios se sale de este canal, como se muestra en la Figura 10.

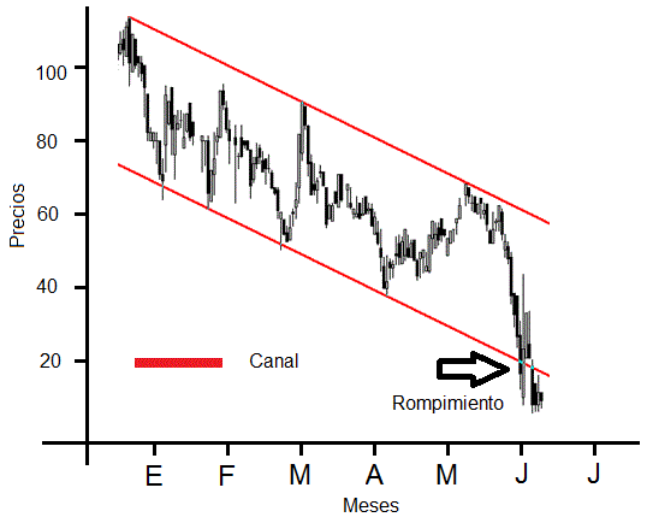


Figura 10: Representación gráfica de un rompimiento de canal.

El uso de este tipo de sistemas permite manejar grandes cantidades de datos relacionados

con los factores que afectan el desempeño de una inversión (variables macroeconómicas, información de la compañía, indicadores de la industria, variables de mercado, etc.), al mismo tiempo que evita las reacciones psicológicas que los operadores sufren al invertir en mercados financieros (Contreras *et al.*, 2012).

Una de las dificultades que los autores manifiestan es el tiempo computacional necesario para entrenar el sistema de compra-venta con datos diarios de precios de valores. Esta restricción es aún más crítica cuando tomamos en cuenta que la mayoría de los operadores invierten de manera intra-diaria (lo que significa que las posiciones de inversión son canceladas durante el mismo día). Por lo tanto, el operador en mercados financieros parece recomendar el uso de datos con frecuencia más corta para entrenar y aplicar sistemas de compra-venta mecánicos (Contreras *et al.*, 2012).

Sin embargo, el enfoque sobre datos intra-diarios (en vez de datos diarios) causa algunas dificultades en el diseño y aplicación de los sistemas de compra-venta por el tiempo más corto de respuesta que se requiere, dado que no es posible esperar horas para obtener una decisión de inversión generada por el sistema de compra-venta mecánico cuando las operaciones tienen que hacerse continuamente durante el día (Contreras *et al.*, 2012).

Consecuentemente, la dificultad de resolver el problema de inversión depende del número de indicadores incluídos en el sistema de compra-venta y el rango permitido para los parámetros que serán usados como valores de dichos indicadores. Observe que enfrentamos un problema duro de optimización combinatoria que crece exponencialmente con indicadores y rango de parámetros (Contreras *et al.*, 2012).

2.2.9 Estrategia de "compra y mantiene"

La estrategia "*Buy and Hold*" o "compra y mantiene", es una estrategia de inversión pasiva. Un inversionista compra un activo y lo mantiene durante un periodo prolongado de tiempo, sin importar pequeños cambios en su valor (Esfahanipour y Mousavi, 2011).

La mayoría de los estudios anteriores han usado la estrategia de compra-y-venta como plataforma de comparación (Chen *et al.*, 2008). Por ejemplo, How *et al.* (2010) usan una función objetivo (*fitness*) basada en la diferencia entre la ganancia que obtiene su estrategia y la estrategia de compra y mantiene, tomando en cuenta costos de transacciones y precios sin normalizar. Fernández-Pérez *et al.* (2010) usan esta estrategia como comparación en su artículo de investigación donde utilizan un algoritmo genético para estimar los mejores parámetros para un modelo de tendencias de precios en el mercado de divisas.

2.2.10 Posiciones de mercado

Los mercados financieros tienen términos específicos para expresar acciones de inversión. Como se mencionó con anterioridad, el hecho de hacer una compra de un activo esperando que suba de precio para luego venderlo y obtener una ganancia se le denomina “tomar una posición *larga*” . También existe lo que que denomina “tomar una posición *corta*” , que consiste en vender algo que nunca fue de su propiedad, es necesario tomar prestado el activo (y pagar una pequeña cuota a la entidad que lo presta) para poder venderlo. Posteriormente se vuelve a comprar el activo para obtener una ganancia y se regresa a su dueño original. En este último caso la idea es que el activo baje de precio después de su venta para luego comprarlo a un precio más bajo del que se vendió y obtener así una ganancia.

En caso de que no se tenga una posición en el mercado se dice que se está ”tablas” o ”plano” . Si se tiene una posición abierta y se quiere cerrarla, se dice que ”quiere quedar tablas” . Si se tiene una posición corta, se tiene que comprar para quedar tablas. Si se tiene una posición larga, se tiene que vender para quedar plano. El único momento en que no se está expuesto en el mercado o se tiene riesgo financiero, es cuando se está tablas (Galant y Dolan, 2007).

2.2.11 Estrategia de ruptura

El análisis técnico puede dar indicios a lo largo de una fluctuación grande en los precios de los mercados, permitiendo que los operadores puedan predecir con mayor exactitud la dirección y amplitud de futuras variaciones en los precios. Lo más importante, es que el análisis técnico es la clave para construir una estrategia de compra-venta bien definida. Por ejemplo, criterios de análisis fundamental, expectativas de datos, o el instinto pueden sugerir que el precio de una divisa va a bajar. Pero, exactamente ¿en qué punto se hace la entrada corta? ¿Dónde tomar las ganancias y dónde cortar las pérdidas? Se puede usar análisis técnico para refinar puntos de entrada y de salida, y para decidir si se van a añadir posiciones o reducirlas y donde (Galant y Dolan, 2007).

La estrategia de ruptura es una estrategia de compra-venta técnica que establece un punto de entrada, basado en un canal, y posteriormente define puntos para tomar ganancias y cortar pérdidas relativos al punto de entrada, como se ilustra en la Figura 11. Para establecer el canal se toma un periodo de tiempo, de donde se extrae un punto de *resistencia* y un punto de *soporte*. Generalmente la resistencia son máximos que se repiten, y el soporte son mínimos

que se repiten. Una vez que se tiene el punto de resistencia se calcula el punto de entrada agregando una variación a este valor. A partir de ahí se calculan los valores de riesgo y salida, usando el punto de entrada como base. Lo mismo sucede con el punto de soporte. Los valores generados a partir de la resistencia se usarán para tomar posiciones largas, y los valores generados a partir del soporte se usarán para tomar posiciones cortas (Cekirdekci y Iliev, 2010; Holmberg *et al.*, 2013).

La idea detrás de esto es que una vez que el precio suba por encima de la resistencia, dicha resistencia se convertirá en un nuevo soporte por lo que es el momento de hacer una compra. Para el soporte sucede algo similar, y se convierte en una nueva resistencia cuando el precio se decrementa por debajo de este, por lo que es el momento de hacer una venta.

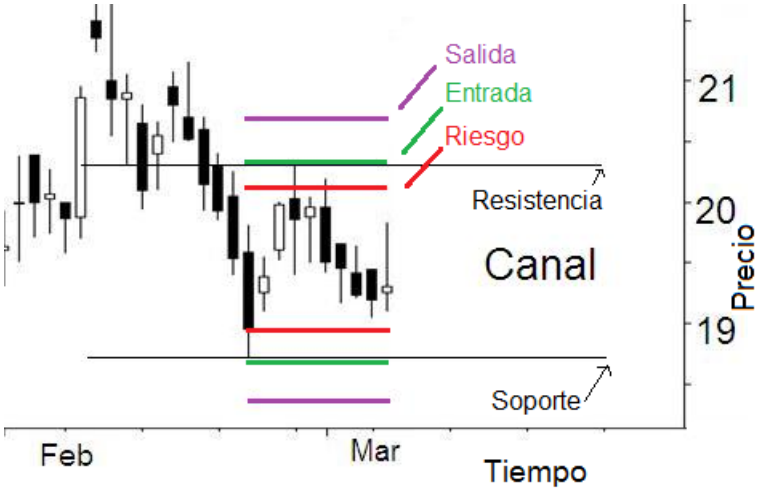


Figura 11: Componentes de la estrategia de ruptura. La entrada, riesgo y salida son calculados a partir de la resistencia y roporte que componen el canal.

2.2.12 Mercado de divisas

La mayor parte de la investigación que se ha realizado sobre análisis técnico, originalmente fue dirigida al mercado de valores (MV). Sin embargo, desde los años 70 este enfoque de compra-venta ha sido adoptado por inversionistas del mercado de divisas (MD) (Dempster *et al.*, 2001). Un MD es un mercado donde una divisa es intercambiada por otra(s) divisa(s). En dicho intercambio una divisa es considerada *doméstica* y la otra *extranjera*. Además, existe un tipo de cambio donde una divisa tiene un precio en términos de otra (Wang, 2005). Este precio sube o baja (Wilson y Banzhaf, 2010).

El MD difiere del MV en varias maneras. Se pueden hacer transacciones las 24 horas del día en todo el mundo, de manera casi instantánea (no se necesita tiempo para ponerse de

acuerdo para la transacción) (Wilson y Banzhaf, 2010). Además los precios de transacciones en el MD son relativamente baratos, por lo que influyen menos que en los MV (Hirabayashi *et al.*, 2009). Cabe destacar que los MD son los más activos de todos los mercados financieros. Por estas características no hay cambios abruptos entre diferentes días (entre el cierre y la apertura del mercado) o rezagos en cambios de precios por falta de entidades disponibles para realizar una transacción (Wilson y Banzhaf, 2010).

2.2.13 Compra-venta de divisas

La compra-venta de divisas consiste en una sola transacción que es de manera simultánea una compra y una venta. En el MV si se compra una acción de una compañía, se tiene esa acción y se espera que suba de precio. Cuando se quiere cerrar esa posición simplemente se vende la acción.

Pero con divisas, la compra de una divisa involucra la venta simultánea de otra divisa. Esto es la parte de *intercambio* de "exchange" en "foreign exchange" por nombre en inglés. Lo que es lo mismo, si se busca que el dólar suba de precio, la pregunta es "¿suba con respecto a qué?". La respuesta es otra divisa. En términos relativos, si el dólar sube comparado con otra divisa, esa otra divisa también baja con respecto al dólar. Pensando en términos del MV: cuando se compra una acción se vende efectivo y cuando se vende una acción se compra efectivo.

Generalmente las divisas vienen en pares, con nombres que combinan a las dos divisas que se van a intercambiar, una por otra. Las claves de las divisas siguen los códigos ISO (International Stardardization Organization) para cada divisa, como lo son GBP para libra esterlina, USD para dólar americano, JPY para yen japonés, etc. Por ejemplo un nombre para un par de divisas es: GBPJPY. Esta notación ha evolucionado a través de los años y refleja lo que tradicionalmente es una divisa *fuerte* contra una divisa *débil*, con la divisa fuerte nombrada primero. También refleja la convención de mercado en la que la primera divisa de el par es la divisa *base*.

La divisa base es lo que se compra o se vende cuando se compra o se vende un par de divisas. También es el valor nominal de la transacción. Por ejemplo, comprar 100,000 EURJPY significa comprar 100,000 euros y vender lo equivalente en yen japonés. Si se venden 100,000 GBPCHF, se venden 100,000 libras esterlinas y se compra el equivalente en francos suizos. La segunda divisa del par es llamada la divisa complementaria ("counter currency") o divisa secundaria.

En el MD una posición larga se refiere a haber comprado un par de divisas, que significa que se compró la divisa base y se vendió la divisa secundaria. Por lo tanto, la operación anterior de compra de 100,000 EURJPY puede ser denominada *larga*. De manera equivalente, tomar una posición corta significa vender la divisa base y comprar la divisa secundaria. Similarmente, la operación anterior de venta de 100,000 GBPCHF puede ser denominada *corta*.

Es en la divisa secundaria en la que ocurren las fluctuaciones en los precios y será la denominación para las pérdidas y ganancias. Por ejemplo si se compra GBPJPY y el precio sube, y se toma una ganancia, esta ganancia es en yen y no en libras.

2.2.14 Pips

Las fluctuaciones de precios para los pares de divisas se expresan en pips. Un pip es la unidad mínima de movimiento para una divisa en el mercado de divisas (Hryshko y Downs, 2004). Pilbeam (2005) define un pip como 0.0001 de un par de divisas, lo cual es mayormente cierto, excepto para los pares que involucran el yen japonés. En estos casos un pip es 0.01 del valor del par de divisas, pues dicho valor se expresa con dos dígitos después del punto decimal. Para todos los demás pares de divisas se tienen cuatro dígitos después de punto decimal. En todos los casos, un pip es el dígito menos significativo. Un pip también es llamado *punto* (Galant y Dolan, 2007).

Los pips brindan una manera sencilla de calcular ganancias y pérdidas. Por ejemplo, supongamos que el par de divisas EURUSD se cotiza a 1.2853 y luego cambia a 1.2873, un aumento de 20 pips. Si baja de 1.2853 a 1.2792 se dice que bajó 61 pips. Para convertir esos movimientos de pips a cálculos de ganancias y pérdidas solo se tiene que saber el tamaño de la posición. Para una posición de 100,000 EURUSD el movimiento de 20 pips equivale a $ 200 (EUR 100,000 $\times$ 0.0020 = $200). Para una posición de 50,000 EURUSD, el movimiento de 61 puntos se traduce en $305 (EUR 50,000 $\times$ 0.0061 = $305).

Hasta el venerable pip se encuentra en proceso de actualización conforme avanza la tecnología. Los mercados electrónicos se mueven rápidamente hacia fraccionalizar los pips, para realizar transacciones en décimas de pip ($\frac{1}{10}$ de pip) y precios en mitades de pip han sido usados de manera cotidiana para ciertos pares de divisas en los mercados interbancarios durante muchos años (Galant y Dolan, 2007).

En este proyecto se propone implementar una estrategia técnica de inversión para el mercado de divisas, utilizando los algoritmos genéticos de optimización multiobjetivo NSGA-

II y SPEA2. El objetivo es evolucionar una regla técnica de compra-venta, basada en las estrategias de rompimiento de canal.

2.3 Investigación previa relevante

Se han encontrado los siguientes trabajos relevantes que hacen uso de algoritmos genéticos (AG) aplicados al mercado de divisas.

Dunis *et al.* (1999) utilizaron AGs para indentificar tendencias relativamente fuertes cuando existen, y señalar el momento de comprar o vender, en el mercado de divisas. Posteriormente Lawrenz y Westerhoff (2003) usan un AG para modelar el comportamiento de los tipos de cambio basándose en las acciones de participantes en el mercado de divisas, a la hora de evaluar y actualizar sus reglas de inversión. Por otro lado, Hryshko y Downs usaron un AG en un conjunto de reglas de compra-venta para optimizar una estrategia para hacer operaciones en el mercado de divisas (Hryshko y Downs, 2003, 2004, 2005). Alvarez-Diaz y Alvarez (2003) usaron un AG para tratar de encontrar una función analítica que mejor aproximara la variabilidad de los tipos de cambio estudiados. En contraste, Tanaka-Yamawaki (2004) realizó un sistema para predecir fluctuaciones de precios caracterizando la función de densidad de probabilidad, la profundidad de memoria de la serie de tiempo y la estabilidad de las condiciones de probabilidad para las direcciones de movimientos en precios por tick en el mercado de divisas para el tipo cambiario dólar/yen y utilizó un AG para mejorar predicciones de tendencias.

Hirabayashi *et al.* (2009) usan un sistema para el mercado de divisas basado en AGs para generar reglas de compra-venta de manera automática, basadas en indicadores técnicos. Se enfocan en tratar de predecir el mejor momento para realizar una inversión, y no en predecir los precios futuros como lo hacen la mayoría de otros trabajos. En cambio, Zhang y Ren (2010) usaron AG para optimizar una estrategia de compra-venta para el mercado de divisas basadas en indicadores técnicos y determinar si tener una posición neutral en dichas estrategias es o no una ventaja. Por otro lado, Fernández-Pérez *et al.* (2010, 2012) usaron un AG para probar la hipótesis de la caminata aleatoria contra la existencia de tendencias en los precios históricos de 95 divisas contra USD. Aplicando un enfoque diferente, Budík y Doskočil (2011) usaron la plataforma "Adaptrade", que usa AG's para realizar análisis estadístico sobre algunos instrumentos financieros tomados como indicadores técnicos y así crear una estrategia para generar un portafolio de inversión del mercado de divisas, basado

en USD, EUR y GBP. Rasekhi *et al.* (2011) utilizaron un algoritmo genético para tratar de modelar el comportamiento del tipo de cambio EUR/USD usando modelos fundamentales que incluyen una variedad de factores.

Mańdziuk y Jaruszewicz (2011) usan un AG para preseleccionar las variables de entrada basados en los datos de fuentes tal vez correlacionadas, que serán usadas por un predictor basado en redes neuronales, para tratar de estimar el valor de cierre del DAX . Por otro lado, Budík y Doskočil (2011) usaron un AG para diseñar un portafolio de inversiones para el mercado de divisas con los pares GBPUSD y USDJPY con aplicación a operaciones de compra-venta intradiaria. Además, Mendes *et al.* (2012) usaron un AG para optimizar un conjunto de reglas que constituyen un sistema de compra-venta para el mercado de divisas con 2 pares de monedas. En un enfoque diferente, Mayo (2012) utilizó un software llamado *Evolutionary Data Selection* para quitar porciones de los datos que se alimentan a un sistema de predicción de precios del mercado de divisas intradiarios para tratar de obtener mayor precisión.

A continuación se mencionan los trabajos encontrados relacionados con el mercado de divisas que usan otros algoritmos evolutivos que no son AG's.

Dempster *et al.* (2001) usaron PG para tratar de generar estrategias rentables usando indicadores técnicos populares y realizar compra-venta intradiaria en el mercado de divisas. Por otro lado, Brabazon y O'neill (2004) usan Evolución Gramatical para generar reglas de compra-venta técnicas para el mercado de divisas de Londres y las compara con una estrategia *buy-and-hold*. Huang y Lu (2009) usaron un algoritmo evolutivo de bi-rácimos para tratar de encontrar relaciones entre divisas y así ayudar a explicar relaciones económicas y financieras entre países que usan el sistema de los mercados de divisas. En otro trabajo, Myszkowski y Bicz (2010) usan PG para investigar la usabilidad de un Algoritmo Evolutivo para generar reglas para invertir en el mercado de divisas usando el par EURUSD. Wilson y Banzhaf (2010) usaron PG Líneal para realizar compra-venta automática de cuatro pares de divisas, donde compra para obtener ganancias y vende para prevenir pérdidas, con niveles moderados de compra-venta. Usando un enfoque diferente, Mehta *et al.* (2011) usaron la plataforma de compra-venta TradeStation para desarrollar una estrategia automatizada para realizar operaciones de inversión en el mercado de divisas y desarrollaron indicadores y estrategias en el lenguaje de programación propietario de dicha plataforma para probar y evaluar cuales eran más exitosos.

Finalmente se mencionan algunos trabajos que usan alguna forma de cómputo evolutivo

y tienen que ver con algún mercado financiero.

Allen y Karjalainen (1999) usaron un AG para encontrar reglas técnicas de compra-venta para el mercado de valores. Mencionan que el método se puede aplicar al mercado de divisas. En contraste, Kaboudan (2000) diseñó un algoritmo de PG para predecir series de tiempo del mercado de valores y desarrolló estrategias para compra-venta en las que obtuvo ganancias relativamente altas en inversiones durante cincuenta días con seis acciones. Similarmente, Santini y Tettamanzi (2001) describen una aplicación de PG para predecir tendencias en mercados financieros. De la misma manera, Schwaerzel y Bylander (2006) usaron Programación Genética con funciones trigonómetricas y funciones estadísticas de orden alto para predecir series de tiempo en el mercado de divisas, principalmente con libra esterlina y yen japonés. Usando otro enfoque, Zhou *et al.* (2006) usaron indicadores financieros en AG para seleccionar y optimizar portafolios de acciones en el mercado de valores. En otro trabajo, Chen *et al.* (2008) usaron PG para ver que tan bien podían simular el descubrimiento de reglas de inversión basadas en análisis técnico y las probaron con una variedad de mercados financieros.

Tokuoka y Tanaka-Yamawaki (2008) usaron un AG para seleccionar la mejor combinación de indicadores para cada acción (del mercado de valores) de los indicadores usados en análisis técnico para predecir precios del mercado de valores. En contraste, García-Almanza *et al.* (2008) usaron PG para descubrir patrones en series de datos financieros y detectar oportunidades de inversión. Con un enfoque parecido, Larkin y Ryan (2008) introdujeron un nuevo modelo usando PG para la predicción financiera donde se clasifican noticias como positivas, negativas o neutrales en relación con un mercado particular o sector de interés. Becker *et al.* (2008) usaron PG en 3 algoritmos multiobjetivo para tratar de construir un portafolio con las acciones con mayor desempeño en el mercado de valores.

Mehrara *et al.* (2010) usan redes neuronales con reglas de análisis técnico para determinar que es posible obtener altas ganancias en el mercado de oro explotando sútiles patrones en los cambios de precios, y por lo tanto, dicho mercado no es eficiente. En otro tipo de mercado, How *et al.* (2010) usaron PG para investigar si el hecho de que una acción (*stock*) sea pequeña representa una condición necesaria para que estuviera valuada erróneamente y una condición suficiente para operaciones de compra-venta técnica redituable. Esfahanipour y Mousavi (2011) usaron PG para generar reglas de compra-venta técnicas en el mercado de valores de Therán. Finalmente, Contreras *et al.* (2012) paralelizan un AG para mejorar el desempeño de un sistema de compra-venta aplicado al mercado de valores, tomando en cuenta varios

indicadores fundamentales y técnicos para varias compañías.

2.4 Resumen

En este capítulo se describió el lugar que ocupan los algoritmos genéticos NSGA-II y SPEA2 dentro del área de cómputo evolutivo. Estos algoritmos destacan porque utilizan estrategias para tratar de mantener una diversidad de sus elementos. También se ubicaron a los mercados financieros dentro del área general de finanzas y se describieron de manera general aspectos de la teoría y práctica necesarios para entender una estrategia de inversión orientada al mercado de divisas. Y finalmente se presentó un resumen de los trabajos relacionados con esta área de investigación. El objetivo de este capítulo es ofrecer una guía para entender como es que se aplicó la estrategia de ruptura con los algoritmos NSGA-II y SPEA2 en la fase experimental de esta tesis.

Capítulo 3 Requerimientos técnicos

En este capítulo se presenta el análisis realizado para el desarrollo de las herramientas de apoyo y el programa para realizar los experimentos. Se consideran los detalles necesarios para obtener y preparar las bases de datos pertinentes para su uso en los experimentos, así como las características formales de las funciones objetivo a optimizar. También se mencionan las herramientas en las que se basó el desarrollo del programa para realizar los experimentos y las plataformas utilizadas para tal fin.

3.1 Preparación de los datos

3.1.1 Obtención de base de datos

Para realizar los experimentos se obtuvo una base de datos del sitio www.forexrate.co.uk. Al iniciar el desarrollo de la tesis el sitio web permitía bajar secuencias de 2000 datos a partir de una fecha dada, por lo que fue necesario realizar un programa que bajara las bases de datos por partes y luego procesarlas para unificarlas. En dicho sitio web, era posible especificar la fecha de inicio para obtener los datos, pero no se podía especificar la hora, por lo que se tenían datos duplicados al final de un archivo y al inicio del siguiente. Además de que por la gran cantidad de archivos que se obtuvieron al descargar los datos con granularidad por minuto, era un poco difícil su manejo.

Posteriormente, durante la fase de pruebas, el sitio web presentó una modificación en su interface web, que facilitó la obtención de la base de datos en archivos con más de 2000 datos, donde se podía ya especificar la fecha de inicio y la fecha final que fuera de interés.

Cada archivo tenía una serie de columnas con los campos fecha, hora, volumen de transacciones en lotes de 100,000 unidades, precio de apertura, precio de cierre, mínimo y máximo para cada par de monedas. Se bajaron los archivos correspondientes a las variaciones de precios por minuto y por días para los tipos de cambio EURUSD, GBPUSD, EURCHF, EURGBP y USDJPY para los años 2008, 2009, 2010, 2011 y 2012. En la Tabla 1 se muestra un extracto del contenido de un archivo obtenido para el año 2011 con el tipo de cambio EURUSD.

Tabla 1: Muestra de base de datos de precios por minuto del año 2011 para el tipo de cambio EURUSD.

FECHA	HORA	VOLUMEN	APERTURA	CIERRE	MIN	MAX
12/31/2010	15:41:01	250	1.3396	1.3396	1.3392	1.3399
12/31/2010	15:42:01	427	1.3399	1.3399	1.3394	1.3403
12/31/2010	15:43:01	248	1.3399	1.3401	1.3396	1.3403
12/31/2010	15:44:01	247	1.3401	1.3394	1.3392	1.3402
12/31/2010	15:45:01	332	1.3395	1.3391	1.3391	1.3398

3.1.2 Preprocesamiento

Las bases de datos obtenidas contienen caracteres especiales que son terminadores de línea o de archivo del sistema operativo Ms-dos, o Windows, que en algunas ocasiones producen problemas al ser cargados en Java. Para esto fue necesario realizar un preprocesamiento para quitar estos caracteres especiales de todos los archivos.

También se realizó un programa para buscar líneas duplicadas de datos, que pudieron haberse introducido al obtener la base de datos a partir de fechas dadas que estuvieran traslapadas con fechas anteriores. Esto no fue necesario en la fase de evaluación de pruebas, pues como se mencionó, la interface web cambió completamente, evitando los datos duplicados.

Una vez que se quitaron caracteres especiales y se verificó que no hubiera líneas duplicadas, se procedió a pegar los datos para generar bases de datos de precios por los periodos necesarios para los experimentos. Es posible que en estas bases de datos existan "huecos" donde no haya información para ciertas fechas, como es el caso normalmente de los fines de semanas o días festivos, pero también puede ser por alguna falla al momento de captura de datos por parte del proveedor de la base de datos. En este caso no se puede hacer nada, excepto ajustar los periodos en el momento de que se procesa la estrategia de inversión y calcular los parámetros de inversión en base a los datos que se tienen.

Se desarrolló además un programa para convertir los precios por minuto a precios cada 5 minutos, lo que involucró tomar un dato dado y los datos siguientes que estuvieran disponibles en un periodo menor a 5 minutos, suponiendo que tal vez habría datos faltantes, es decir, que en un periodo de 5 minutos, teniendo datos cada minuto, tal vez solo haya información para los minutos 1, 2 y 4, pero no para los minutos 3 y 5. Dicho programa debió ser capaz de anticipar esta situación y hacer los cálculos en base a los datos disponibles, sin usar los datos

correspondientes al siguiente periodo suponiendo que serían parte de los datos faltantes.

Posteriormente, teniendo estos datos por periodos de 5 minutos, se realizó una búsqueda de los precios mínimo y máximo para cada periodo, tomando la fecha, hora y el precio de apertura del primer dato, y el precio de cierre del último dato, además de sumar los volúmenes de transacciones de todos los datos para generar una nueva entrada en una base de datos con los nuevos valores obtenidos. Dicho programa puede generar también precios cada 10 minutos, 20 minutos, 30 minutos, o por hora. Los precios por días fueron obtenidos en una consulta aparte de la página web, por lo que no fueron generados.

A partir del cambio en la interface del consulta del sitio web www.forexrate.co.uk, es posible obtener los datos con la granularidad deseada directamente, sin hacer ningún tipo de procesamiento, excepto por quitar los caracteres especiales.

3.1.3 Base de datos de entrenamiento y base de datos de pruebas

Para realizar los experimentos se hizo una distinción entre los datos necesarios para entrenar y los datos necesarios para probar los resultados obtenidos. Con la finalidad de determinar que datos se usarían, se realizaron algunas pruebas con bases de datos para los tipos de cambio EURUSD, GBPUSD, EURCHF, EURGBP y USDJPY con granularidad de precios diarios. Los resultados preliminares fueron alentadores, pero al momento de realizar pruebas con mayor cantidad de datos el tiempo de procesamiento incrementaba considerablemente, pasando de unos cuantos segundos con precios diarios a varias horas con granularidad de precios por minuto, por lo que finalmente se seleccionaron las bases de datos de entrenamiento correspondientes al tipo de cambio EURUSD para el año 2011 con granularidad de 5 minutos donde una corrida de entrenamiento duraba en promedio 30 horas.

La base de datos seleccionada para probar los parámetros obtenidos con la fase de entrenamiento de la estrategia de inversión fue la correspondiente al periodo inmediatamente posterior al periodo de entrenamiento, es decir, la del año 2012, usando la misma granularidad de 5 minutos. Se construyeron 5 periodos de prueba diferentes para determinar el mejor periodo de rendimiento de la estrategia. Los diferentes periodos fueron: enero, enero-marzo, enero-junio, enero-septiembre y enero-diciembre. Las pruebas realizadas, con esta granularidad, duraban desde varios segundos hasta unos cuantos minutos, por lo que comparadas con el entrenamiento, eran bastante rápidas.

3.2 Herramienta de inversión

Este trabajo se basó en un programa llamado "Ninja Trader" (www.ninjatrader.com). "Ninjatrader" es una solución completa para corredores de valores, acciones y divisas. Este programa también es gratis para producir gráficas, análisis de mercados, desarrollo de sistemas de inversión y simulación de inversiones. Para más información consultar http://www.ninjatrader.com/about.php.

Se hicieron pruebas preliminares para ver como funcionaba el programa al momento de implementar una estrategia de inversión y se analizó la manera en que NinjaTrader procesaba los datos. Con base en esto se desarrolló una implementación preliminar de la estrategia de inversión usando el lenguaje de programación C# que utiliza por omisión dicha herramienta. El funcionamiento del procesamiento de datos de NinjaTrader se describe a continuación:

Dado un conjunto de N datos ordenados de $N-1$ a 0, donde $N-1$ es el dato más antiguo que se tiene y 0 el dato más reciente, se evalua la función objetivo para determinar si realiza alguna acción tal como 'abrir una inversión'. Una vez que se recibe un dato nuevo, más reciente que el dato 0, el conjunto anterior $\{d_{N-1},...,d_0\}$ será reenumerado como $\{d_N,...,d_1\}$ y el dato nuevo será el dato d_0, teniendo ahora un conjunto $\{d_N,...,d_0\}$. Se evalúa otra vez la función objetivo sobre el nuevo conjunto de datos. Y así sucesivamente mientras sigan llegando datos nuevos. En el apéndice A se puede ver de manera general el algoritmo usado para realizar la evaluación durante el periodo de inversión.

El programa NinjaTrader puede obtener los datos de final de día de Kinetick, uno de sus proveedores de datos. El programa viene preconfigurado para realizar esta conexión y esta operación es transparente para el usuario. Estos datos son gratuitos, pero solo contienen los datos de precios por cada día lo cual no era suficiente para el desarrollo de este trabajo. Una vez realizada la implementación del programa en C# para el programa NinjaTrader, se tomó este código como guía para realizar una implementación en Java. Los requerimientos principales fueron: 1) Que fuera posible evaluar un solo conjunto de parámetros para la estrategia de inversión en un periodo dado de tiempo. 2) Que pudiera funcionar con múltiples elementos para poder evaluar poblaciones enteras de posibles respuestas durante la ejecución del algoritmo genético. Era indispensable que el programa pudiera ser ejecutado manualmente para poder hacer evaluaciones a manera de verificación, y además para correr la fase de pruebas, así como para disminuir la posibilidad de error al tener 2 programas diferentes que se tuvieran que modificar por separado. Dicha implementación se tomó como función

evaluadora de aptitud.

Esta función se usó en un programa denominado Opt4j (Lukasiewycz *et al.*, 2011). Para dicho programa se desarrollaron varios módulos que se encargan de definir el problema, crear una población de individuos a evaluar, realizar la evaluación de aptitud para cada individuo, y correr la simulación del algoritmo evolutivo durante las generaciones especificadas. La estructura de dichos módulos ya está previamente establecida y se describe con más detalle en el Capítulo 4.

3.3 Definición de la función objetivo

Suponga que se adquiere una cantidad dada de bienes a un precio de entrada Pe, y luego se venden esos mismos bienes a un precio de salida Ps. Si la diferencia (Ps - Pe) es mayor que 0, se dice que se tiene una ganancia G. De lo contrario G representa una pérdida. A esta transacción se le agrega una pequeña cantidad que representa una comisión aplicada por la empresa que realiza dicha transacción en el mercado y generalmente es un porcentaje del valor de dicha operación. Además que se tiene que considerar que en caso de que se realice una entrada larga o una entrada corta, los precios de entrada y de salida pueden invertirse, pues es posible primero realizar una venta y luego una compra, por lo que la fórmula final quedará: $G = T(e)(Ps - Pe) - C$, donde $T(e)$ representa un signo para el cálculo, que es positivo en caso de que sea entrada larga, y negativo en caso de que sea entrada corta, y C es la comisión de operación.

Durante un periodo dado se puede realizar este proceso de compra-venta n veces, teniendo como resultado el siguiente conjunto: $\{G_1,...,G_n\}$ que representa las n posibles ganancias o pérdidas para esas n operaciones de inversión.

Las metas de la estrategia de inversión durante un periodo son sencillas: tener la mayor ganancia total o ganancia neta posible, y que la pérdida más grande sea lo más pequeña posible.

De aquí que para un conjunto de parámetros que constituyen una estrategia de inversión particular, es decir, un solo individuo de la población, se calcula la ganancia total sumando todas las ganancias del periodo completo representado por una base de datos, con la siguiente fórmula: $Gt = \sum_{i=1}^{n} G_i$. Y para calcular la pérdida mayor se toman todas las pérdidas que hubo en el periodo, seleccionándose la más negativa: $Pm = Min(G_i)$, donde $G_i < 0$, y $1 \leq i \leq n$. En otras palabras: se maximiza $|G_i|$ cuando $G_i < 0$, como se muestra en la

Figura 12. Supongamos que a, b, c, y d fueron las pérdidas para un periodo de inversión. La pérdida mayor es la que esté más a la izquierda, es decir el elemento mínimo de todos, en este caso la literal d.

Figura 12: Pérdida más grande. Las literales representan pérdidas en un periodo de inversión. El valor d representa la mayor pérdida.

Como se contemplan varios conjuntos de parámetros a evaluar, donde cada conjunto representa un individuo de una población, digamos k individuos, se tiene que escoger la ganancia total Gt más grande de todas las ganancias para los individuos de la población, lo cual se representa como $Max(Gt_i), 1 \leq i \leq k$; además se tiene que escoger la pérdida más chica de todas las pérdidas mayores, de la siguiente manera: $Max(Pm_i), 1 \leq i \leq k$, o lo que es lo mismo: minimizar $|Pm_i|$.

En la Figura 13 muestra como escoger la pérdida más chica. Supongamos que a, b, c y d fueron las pérdidas mayores para 4 diferentes individuos de una población, el valor a representa la pérdida menor, pues está más a la derecha de todos, es decir, el elemento máximo.

Figura 13: Pérdida más chica. Las literales representan pérdidas para 4 diferentes individuos.El valor a representa al individuo con la menor pérdida.

Por lo tanto, las ecuaciones (1) y (2) representan los resultados de aplicar la función objetivo a una base de datos de precios para un par de divisas completo, cuya ganacia G_i puede ser calculada con las ecuaciones (3) y (4).

Esto es

$$Gt = \sum_{i=1}^{n} G_i \tag{1}$$

y

$$Pm = Min(G_i), G_i < 0. \tag{2}$$

G_i puede ser calculada como

$$G_i = T(e_i)(Ps_i - Pe_i) - C_i, \tag{3}$$

donde

$$T(e_i) = \begin{cases} +1, si\ e_i = EntradaLarga \\ -1, si\ e_i = EntradaCorta. \end{cases} \tag{4}$$

De esta manera, para obtener los mejores individuos de toda la población con k individuos, Opt4J selecciona los elementos con mayor ganancia total Gt y menor pérdida mayor Pm, formalizando las ecuaciones (1) y (2) al siguiente problema de optimización como se muestra a continuación en (5):

$$\begin{aligned} &\text{Maximizar } Gt_j, \\ &\text{maximizar } Pm_j, \\ &\text{con } 1 \leq j \leq k. \end{aligned} \tag{5}$$

3.4 Sistema de optimización

Se utilizó Opt4j como herramienta para realizar los experimentos. Opt4J es un sistema modular de optimización de tareas desarrollado en el lenguaje Java, el cual se basa en dividir dichas tareas en subtareas correlacionadas que se deben optimizar de manera concurrente (Lukasiewycz *et al.*, 2011). Una de las ventajas de este programa es que era posible programar la ejecución por lotes de los experimentos lo cual no era posible en NinjaTrader. Este programa es de código abierto y corre en varias plataformas como lo son Windows y Linux. Otra de las ventajas de este sistema es que cuenta con una implementación de los algoritmos NSGA-II y SPEA 2, entre otros. Para poder usarlo solo se necesita desarrollar algunas funciones que permiten integrar la función objetivo a su lista de funciones y definir como se van a definir las poblaciones de posibles respuestas.

3.5 Herramientas de apoyo

Para agilizar algunas tareas relacionadas con la ejecución de los experimentos se desarrollaron algunas herramientas de apoyo. Entre dichas tareas se pueden mencionar además de preprocesar las bases de datos de tipos de cambio para detectar y corregir errores en el formato, procesar los datos resultantes de las corridas de los experimentos para agilizar su análisis e interpretación, automatizar la generación de gráficas usando GNUPlot, así como el monitoreo del sistema para modificar la prioridad de los procesos relacionados con los experimentos. En el capítulo siguiente se mencionan algunos detalles específicos del funcionamiento de dichas herramientas y se muestra el código de implementación de ellas en el apéndice C.

3.6 Resumen

En este capítulo se presentaron los elementos a tomar en cuenta para desarrollar el sistema con el cual se realizaron los experimentos. Esto es, las bases de datos utilizadas, herramientas de programación, las consideraciones teóricas que debería cumplir la función objetivo, los aspectos técnicos de la plataforma de inversión que se utilizó como modelo, y los requerimientos para que la función objetivo pudiera ser usada en el sistema de optimización Opt4j durante el entrenamiento, y posteriormente de manera independiente durante la fase de pruebas.

Capítulo 4 Diseño conceptual, implementación y resultados del algoritmo

En este capítulo se expone la metodología seguida para la realización de los experimentos que incluyen la fase de entrenamiento y la fase de pruebas. Dichos experimentos se realizaron bajo la plataforma Opt4J después de la implementación de la función objetivo y los diferentes módulos requeridos, también se describen los detalles que fueron considerados en el desarrollo y ejecución de la fase de entrenamiento para obtener los "mejores parámetros" para la estrategia de inversión, así como su posterior evaluación en la fase de pruebas y los resultados obtenidos.

Tabla 2: Lista de parámetros configurables para NSGA-II y SPEA2 en la herramienta Opt4J

Módulo	Parámetro	Posibles valores
BasicCrossover	integerType	XPOINT, RATE
	integerXPoints	Valor Entero
	integerRate	Flotante entre 0 y 1
BasicMutate	mutationRateType	CONSTANT
	mutationRate	Flotante entre 0 y 1
EvolutionaryAlgorithm	generations	Entero
	alpha	Entero
	mu	Entero
	lambda	Entero
	crossoverRate	Flotante entre 0 y 1
SPEA 2	tournament	Entero
NSGA-II	tournament	Entero

4.1 Diseño de los algoritmos

Para la realización de los experimentos se utilizaron las implementaciones de los algoritmos NSGA-II y SPEA2 incluídas en la plataforma Opt4J, las cuales hacen uso de la función objetivo desarrollada. Cada algoritmo hace uso de parámetros en común en un archivo de configuración xml. Los parámetros configurados para ambos fueron: generaciones, tamaño de población, cantidad de padres, número de hijos, probabilidad de cruzamiento, tipo de cruzamiento, parámetro para cruzamiento, tipo de mutación (constante), probabilidad de mutación y tamaño de torneo (Lukasiewycz *et al.*, 2012). En la Tabla 2 se muestran los módulos con parámetros configurables y sus posibles opciones. Además la función objetivo

utiliza sus propios parámetros de evaluación, los que son generados por Opt4J según la configuración definida en uno de los módulos que se mencionarán más adelante.

4.1.1 Representación del cromosoma

Seleccionar la representación de los cromosomas es un paso muy importante que puede complicar o simplificar el procesamiento del algoritmo genético (Eiben y Smith, 2003).

Seis de los parámetros a evaluar de la estrategia de ruptura se representan como valores flotantes con 4 dígitos después del punto decimal, a menos que se maneje el yen japonés, en cuyo caso la representación tiene 2 dígitos después del punto decimal. Estos parámetros son los valores para Entrada, Riesgo y Salida para una entrada larga, y además valores de Entrada, Riesgo y Salida para una entrada corta. También se usan valores enteros para el tamaño del periodo y la hora de inicio. Como el tamaño más pequeño de una posible variación en el precio de la mayoría de los pares de monedas es un pip, resulta más conveniente representar los parámetros para la estrategia de ruptura con valores enteros que representen dichos pips para obtener un cromosoma homogéneo con todos los parámetros representados por enteros. De esta manera se pueden aprovechar las operaciones definidas sobre números enteros sin tener que definir operaciones especiales para un cromosoma heterogéneo.

Por estas razones, para representar el cromosoma se utilizó una cadena de números enteros, donde cada uno representa un parámetro de la estrategia, además de un elemento para el periodo y la hora de inicio. Los elementos F_1, F_2, F_3, representan los parámetros de Entrada, Riesgo y Salida, respectivamente, para una entrada larga. De igual manera, los elementos F_4, F_5, F_6 representan los mismos parámetros pero para una entrada corta. Finalmente, el elemento F_7 representa el periodo y el elemento F_8 representa la hora de inicio para realizar una inversión, como se muestra en la Figura 14.

Figura 14: Representación del cromosoma para la estrategia de rompimiento. Los elementos F_1, F_2, F_3, representan los parámetros de Entrada, Riesgo y Salida para una entrada larga. Los elementos F_4, F_5, F_6 representan los mismos parámetros para una entrada corta. El elemento F_7 representa el periodo y el elemento F_8 representa la hora de inicio para realizar una inversión.

4.1.2 Operadores de cruce y mutación

De la población de posibles soluciones se realiza un torneo entre 2 individuos seleccionados al azar para obtener un elemento padre. A estos elementos padres se les aplican los operadores de cruce y mutación para generar los elementos hijos.

Los operadores de cruce aplicados con una probabilidad $p_c = 0.8$ utilizados para la etapa experimental fueron: *1 punto* y *uniforme*. El cruzamiento de un punto fue el operador de recombianción original propuesto en Holland (1975) y examinado en DeJong (1975). Funciona seleccionando un número aleatorio r en el rango [1, l - 1] (siendo l la longitud del cromosoma), y después partiendo ambos padres en este punto y creando los dos hijos al intercambiar las colas.

A diferencia del cruzamiento de 1-punto, que trabaja dividiendo los padres en secciones de genes contiguos y luego reensamblándolas para producir a los hijos, el cruzamiento uniforme trabaja manejando cada gen de manera independiente y tomando una decisión aleatoria respecto al padre del cual será heredado. Esto se implementa generando una cadena de variables aleatorias con una distribución uniforme sobre [0,1]. En cada posición, si el valor es mayor que un parámetro p (usualmente 0.5), el gen se hereda del primer padre; de otra forma se hereda del segundo padre. Un segundo hijo se crea usando el mapeo inverso. El operador de mutación aplicado fue el denominado *variación (o redefinición) aleatoria* y también es llamado *uniforme* cuando se usa para representaciones de punto flotante. La variación aleatoria funciona escogiendo con probabilidad p_c para cada gen un nuevo valor de manera aleatoria de un conjunto de valores permitidos. En este caso se definió p_c=0.01 y el nuevo valor de cada elemento se genera de manera aleatoria tal que su valor quede entre las cotas superiores e inferiores establecidas para cada elemento del genotipo. Estas cotas fueron definidas en el módulo EstrategiaRompimientoCreator que genera la población inicial de soluciones en Opt4J, quedando 0 como la cota mínima y 100 como la cota máxima de los parámetros para la estrategia de ruptura, 0 y 24 como las respectivas cotas para la hora de inicio y 100 y 2000 para el tamaño del periodo.

4.1.3 Función objetivo

La función objetivo se desarrolló a partir de la definición formal que se presentó en el capítulo anterior para las ecuaciones (1) y (2). Se implementó en la forma de un módulo que recibe una secuencia de parámetros a evaluar y regresara 2 valores: la ganancia total durante el

periodo de evaluación de la base de datos disponible para un individuo, así como su pérdida más grande.

4.1.4 Estrategia de ruptura

La estrategia de ruptura se implementó dentro de la función objetivo. Al analizar la base de datos de precios se realizan cálculos que toman en cuenta los parámetros a optimizar para dicha estrategia, que son: Periodo, Hora de Inicio, Entrada Larga, Riesgo Largo, Salida Larga, Entrada Corta, Riesgo Corto, Salida Corta. La función objetivo calcula a partir de la fecha inicial, que cumpla con el requisito de la Hora de Inicio, un canal en base a un máximo y un mínimo de un periodo. Estos valores serán tomados como Resistencia y Soporte respectivamente. Al agregarles los valores para las entradas, riesgos y salidas se tienen los umbrales para que funcione la estrategia. En el Algoritmo 1 se muestra este proceso.

Algoritmo 1 Algoritmo `obtener_canal()`

Entrada: Tipo Entero: $EntradaLarga$, $RiesgoLargo$, $SalidaLarga$, $EntradaCorta$, $RiesgoCorto$, $SalidaCorta$, $HoraInicio$, $Periodo$.
Salida: Tipo Entero: $ValorEntradaLarga$, $ValorRiesgoLargo$, $ValorSalidaLarga$, $ValorEntradaCorta$, $ValorRiesgoCorto$, $ValorSalidaCorta$.

1: Establecer periodo de tamaño $Periodo$ con primer elemento cuya hora $\geq HoraInicio$.
2: Calcular valores *máximo* y *mínimo* del $Periodo$.
3: $ValorEntradaLarga \leftarrow máximo + EntradaLarga$.
4: $ValorRiesgoLargo \leftarrow ValorEntradaLarga - RiesgoLargo$.
5: $ValorSalidaLarga \leftarrow ValorEntradaLarga + SalidaLarga$.
6: $ValorEntradaCorta \leftarrow mínimo - EntradaCorta$.
7: $ValorRiesgoCorto \leftarrow ValorEntradaCorta + RiesgoCorto$.
8: $ValorSalidaCorta \leftarrow ValorEntradaCorta - SalidaCorta$.
9: **return** $ValorEntradaLarga$, $ValorRiesgoLargo$, $ValorSalidaLarga$, $ValorEntradaCorta$, $ValorRiesgoCorto$, $ValorSalidaCorta$.

En caso de que los valores de Entrada se rebasen se tomará una posición en el mercado (larga o corta), y si se rebasan los valores de salida se cancelará la posición y se tomará una ganancia. En caso contrario, si se rebasa el valor de riesgo se cancelará la posición para evitar una pérdida. Esto se repetirá para cada periodo mientras se tengan datos disponibles en la base de datos. En el Algoritmo 2 se muestra el funcionamiento general de la estrategia de ruptura.

El apéndice A contiene un pseudocódigo de este algoritmo con mayor detalle y más apegado a la implementación.

Algoritmo 2 Algoritmo de Estrategia de Ruptura

Entrada: Tipos Entero: $EntradaLarga$, $RiesgoLargo$, $SalidaLarga$, $EntradaCorta$, $RiesgoCorto$, $SalidaCorta$, $HoraInicio$, $Periodo$
Salida: Tipos Flotantes: $Ganancia$, $PerdidaMayor$

1: $GananciaTotal \leftarrow 0$
2: $Perdida \leftarrow 0$
3: **for all** Periodos de tamaño $Periodo$ de la base de datos **do**
4: Aplicar a cada periodo el algoritmo `obtener_canal(` $EntradaLarga$, $RiesgoLargo$, $SalidaLarga$, $EntradaCorta$, $RiesgoCorto$, $SalidaCorta$, $HoraInicio$, $Periodo$ `)` y obtener los valores $ValorEntradaLarga$, $ValorRiesgoLargo$, $ValorSalidaLarga$, $ValorEntradaCorta$, $ValorRiesgoCorto$, $ValorSalidaCorta$ que componen el canal, los cuales se usan para evaluar el siguiente periodo.
5: **end for**
6: **if** Existe entrada activa **then**
7: Esperar a que se cumplan las condiciones de salida de la inversión ($Precio \geq ValorSalidaLarga$ o $Precio \leq ValorSalidaCorta$) para cancelar la inversión activa y tomar ganancia o ($Precio < ValorRiesgoLargo$ o $Precio > ValorRiesgoCorto$) para cancelar la inversión y evitar pérdida.
8: $GananciaTotal \leftarrow GananciaTotal + Ganancia$
9: **if** $PérdidaMayor < Pérdida$ **then**
10: $PérdidaMayor \leftarrow Pérdida$
11: **end if**
12: **else if** Se rompió el canal ($Precio \geq ValorEntradaLarga$ o $Precio \leq ValorEntradaCorta$) **then**
13: Realizar una entrada larga o corta, según sea el caso.
14: **end if**
15: **return** $GananciaTotal$, $PérdidaMayor$.

4.1.5 Criterio de finalización

El único criterio de finalización disponible para Opt4J es la ejecución de todas las generaciones definidas en el archivo de configuración para la simulación. En este caso fueron 30 generaciones para experimentos con 20 individuos y 50 generaciones para experimentos con 10 individuos, para que el tiempo de evaluación de los experimentos fuera aproximadamente igual, es decir, 600 evaluaciones y 500 evaluaciones, por lo que se tardaban 30 y 25 horas en correr, respectivamente.

4.1.6 Pseudocódigo NSGA-II

El Algoritmo 3 muestra el pseudocódigo general para el ciclo principal del algoritmo NSGA-II según Deb *et al.* (2000).

Algoritmo 3 Ciclo principal del Algoritmo NSGA-II

1: Generar población inicial de padres P_0 de tamaño N
2: Aplicar operadores de selección por torneo binario, cruce y mutacion para generar la población inicial de hijos Q_0 de tamaño N
3: t = 1
4: $R_t = P_t \cup Q_t$ combinar padres e hijos en un solo conjunto
5: F = `ordenamiento-nodominado(`R_t`)` obtener frentes no dominados del conjunto de padres e hijos
6: **while** $|P_{t+1}| < N$ población de padres no llena **do**
7: `asignar-distancia-amontonamiento(`F_i`)` calcular distancia de amontonamiento en el frente F_i
8: $P_{t+1} = P_{t+1} \cup F_i$ incluir el frente no dominado en la población de padres
9: **end while**
10: `Sort(`P_{t+1}`,`$\geq_n$`)` ordenar P_{t+1} en orden descendente usando $\geq_n$
11: $P_{t+1} = P_{t+1}[0 : N]$ seleccionar los primeros N elementos de P_{t+1}
12: Q_{t+1} = `crear-nueva-poblacion(`P_{t+1}`)` usar selección, cruzamiento y mutación para crear una nueva población Q_{t+1}
13: $t = t + 1$

4.1.7 Pseudocódigo SPEA2

El Algoritmo 4 muestra el pseudocódigo para el ciclo principal del algoritmo SPEA2 publicado en Zitzler *et al.* (2001).

4.2 Diseño e implementación de los algoritmos

El diseño del sistema y de la función objetivo es similar desde el punto de vista funcional. El sistema no tiene ningún tipo de interacción con otro sistema. El único flujo de datos ocurre entre el usuario y el sistema como se muestra en la Figura 15. El usuario configura una serie de parámetros para la ejecución y al finalizar la tarea, el sistema regresa los resultados pertinentes. Los parámetros configurados por el usuario son: número de generaciones, tamaño de población, tipo de operadores, entre otros. También existen parámetros que están codificados de manera estática dentro del sistema, como lo son: cotas superiores e inferiores para los genotipos y la manera de generar las poblaciones, entre otros. Para mayor información consultar Lukasiewycz *et al.* (2011) o el código implementado en Opt4J.

De una manera similar, como se muestra en la Figura 16, el sistema interactúa con la función objetivo haciendo una llamada con una serie de parámetros: Entrada Larga, Salida Larga, Riesgo Largo, Entrada Corta, Salida Corta, Riesgo Corto, Periodo, Hora de Inicio. A su vez, la función objetivo regresa el resultado de la evaluación de la base de datos de

Algoritmo 4 Ciclo principal del Algoritmo SPEA2

Entrada: Tipos Entero: N tamaño población, $\overline{N}$ tamaño archivo, T maximo numero de generaciones
Salida: Lista: A conjunto no dominado

1: Inicialización: Generar población inicial P_0y archivo vacío (conjunto externo) $\overline{P_0} = \oslash$. Definir $t = 0$.
2: Asignación de aptitud: Calcular aptitud de individuos en la población P_t y el archivo $\overline{P}_t$.
3: Selección de entorno: Copiar todos los individuos no dominados en P_t y $\overline{P}_t$ a $\overline{P}_{t+1}$. Si el tamaño de $\overline{P}_{t+1}$ excede a $\overline{N}$, reducir $\overline{P}_{t+1}$ por medio del operador de truncamiento, de otra manera si el tamaño de $\overline{P}_{t+1}$ es menor que $\overline{N}$ entonces llenar $\overline{P}_{t+1}$ con individuos dominados de P_t y $\overline{P}_t$.
4: Finalización: Si $t \geq T$ u otro criterio de terminación se satisface entonces definir A como el conjunto de vectores de decisión representados por los individuos no dominados en $\overline{P}_{t+1}$.
5: Selección de padres: Ejecutar selección de torneo binario con reemplazo en $\overline{P}_{t+1}$ para llenar el conjunto de apareamiento.
6: Variación: Aplicar operadores de recombinación y mutación al conjunto de apareamiento y definir P_{t+1} como la población resultante. Incrementar el contador de generaciones $t = t + 1$ e ir al paso 2.
7: **return** A.

precios previamente definida en un archivo de configuración, para los parámetros definidos. Dicho resultado está compuesto por la *Ganancia total* y *Pérdida más grande* para el periodo evaluado. Este proceso se repite de manera automática por el sistema, para evaluar todos los miembros de la población de cada generación definida en la configuración dada por el usuario.

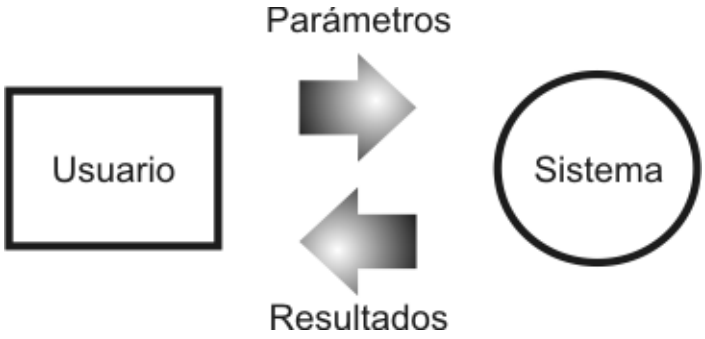


Figura 15: Flujo de datos en el sistema.

El núcleo del sistema es la función objetivo. Esta se aplica a todo el periodo de tiempo definido por la base de datos del tipo de cambio para una divisa y realiza las inversiones pertinentes en caso de que los parámetros definidos lo permitan. Durante el periodo de desarrollo dicha función objetivo se implementó de manera que pudiera ser ejecutada de manera manual, para poder así evaluar casos de prueba durante el desarrollo y además

para que pudiera ser usada en la evaluación de la fase de prueba, posterior a la fase de entrenamiento con Opt4J.

4.3 Módulos implementados

El sistema comprende varias partes que se explicarán a continuación: Función objetivo, Emulador de herramienta de inversión, Módulos para Opt4J. La función objetivo se encarga de realizar el procesamiento de la base de datos para obtener una pérdida o ganancia al finalizar la evaluación de los parámetros de inversión que recibe. El Emulador de la herramienta de inversión es una subsección del programa que se encarga de procesar los datos de la misma manera que lo haría NinjaTrader. Finalmente, los módulos para Opt4J son programas que tienen la estructura requerida para que nuestro problema sea ejecutado por el sistema de optimización Opt4J.

4.3.1 Módulo de función objetivo: `EmulaNinjaTrader.java`

La función objetivo tiene un diseño sencillo que se muestra en la Figura 16. Los parámetros que se proveen son parte de la estrategia de inversión y la base de datos se define con anterioridad a la ejecución del programa. En esta base de datos se define el periodo de evaluación para la función y la granularidad de los datos. Cabe mencionar que la función objetivo trabaja exactamente de la misma manera que lo haría dentro de la herramienta de inversión NinjaTrader: Evalúa desde el primer dato disponible hasta el dato más reciente, justo como se describió en el capítulo anterior.

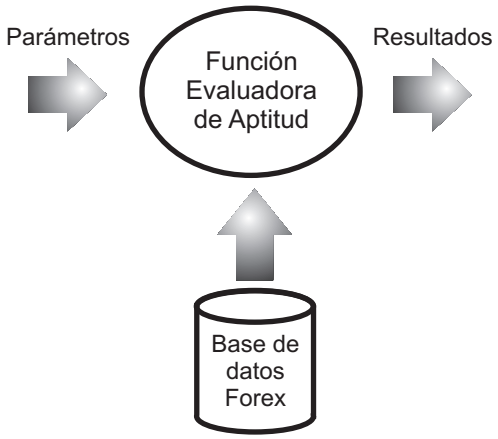


Figura 16: Diseño de la función objetivo.

La primera implementación de la función objetivo se realizó dentro del programa *Nin-*

jaTrader. Dicho programa cuenta con un ambiente de desarrollo integrado (*IDE* por sus siglas en inglés) simplificado que brinda suficientes facilidades para desarrollar código, realizar depuración y compilar en el lenguaje de programación C#.

En el Algoritmo 2 se describe el funcionamiento de la función objetivo de manera general. La idea es dividir la base de datos en periodos y de cada periodo obtener un canal el cual servirá para determinar si se hará o no una inversión durante el periodo siguiente. En caso de que se haga inversión se esperará hasta que se cancele para obtener una ganancia o pérdida. Al final se regresará la Ganancia total calculada con esa base de datos, así como su pérdida más grande.

Para la segunda implementación de la función objetivo se tomó el código en C# y se convirtió a Java para desarrollarse e integrarse junto con los módulos de Opt4J. La clase que implementa la función objetivo es EmulaNinjaTrader, a través del método onBarUpdate. El método EmulaNinjaTrader funciona como emulador de NinjaTrader como se describirá en una sección más adelante. También se desarrolló una clase auxiliar para que la función objetivo pudiera ejecutarse de manera manual, como se describe en la siguiente sección.

La versión para Opt4j fue desarrollada y ejecutada en una plataforma Linux. Una ventaja de esta conversión fue la posibilidad de programar la ejecución por lotes de los experimentos sin necesidad de ningún tipo de intervención y/o monitoreo, lo cual no era posible con *NinjaTrader*. De esta manera se minimizó el tiempo muerto entre ejecuciones de los experimentos.

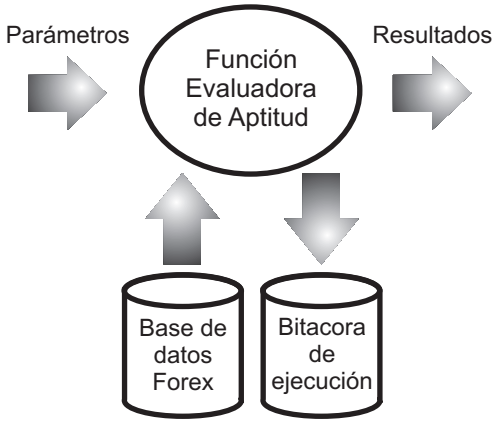


Figura 17: Función objetivo independiente con bitácora.

4.3.2 Módulo para ejecución independiente de función objetivo

Originalmente se implementó la función objetivo como un programa independiente para poder realizar evaluaciones preliminares de un solo conjunto de parámetros y obtener los valores

de resultado para ganancia y pérdida mayor. Esta funcionalidad se conservó durante la adaptación del programa para que funcionara desde el sistema Opt4J con una diferencia: En la ejecución independiente tiene activada una bitácora de transacciones para ayudar en la depuración del sistema como se muestra en la Figura 17.

Dicha bitácora de transacciones está desactivada en el sistema cuando se ejecuta desde Opt4J para evitar cargar el sistema con múltiples escrituras al disco duro por cada ejecución. Potencialmente se podrían realizar 2 escrituras por periodo evaluado. Y cada base de datos tendría docenas o centenas de periodos. Esto sería por cada individuo evaluado.

Este módulo se implementa a través de la clase: ArchivoPrincipal. Se puede llamar con la siguiente instrucción dentro de una ambiente de "shell" en Linux: java estrategia.Archivoprincipal <trayectoria-a-base-de-datos>P HI EL RL SL EC RC SC, donde los parámetros P, HI, EL, RL, SL, EC, RC y SC son números enteros y significan: Periodo, Hora de Inicio, Entrada Larga, Riesgo Largo, Salida Larga, Entrada Corta, Riesgo Corto y Salida Corta, respectivamente. En la Figura 18 se muestra un ejemplo de ejecución con parámetros reales y un ejemplo de su salida.

```
mabs@ubuntu:~/cicese/Tesis Files/PT/codigo$ !1034
java estrategia.ArchivoPrincipal /home/mbarbosa/Desktop/forex\
data/pruebaavance4/eur-usd-2010-1day.csv 6 8 11 206 94 851 11 973
ArchPrin:Tmp:/home/mbarbosa/Desktop/forex data/pruebaavance4/
eur-usd-2010-1day.csv
/home/mbarbosa/PT/codigo
EmulaNT: MiDinero: 100000.0MiTamanoLote: 10000
EmulaNT: Parametros: 6 8 0.0011 0.0206 0.0094
                        0.0851 0.0011 0.0973
EmulaNT: Finales: 6 8 0.0011 0.0206 0.0094
                       0.0851 0.0011 0.0973
primera fecha: 01/01/2010 primera hora: 00:00:01
inicio: 0 ultimo: 260
Emula: %Barras: 0 9 19 29 39 49 59 69 79 89 99
ArchPrin: MiGananciaAcumulada: 545.9999999999985
          PeorPerdida: -220.0000000000002
```

Figura 18: Ejemplo de ejecución manual de la función objetivo desde la clase `Archivo Principal`.

4.3.3 Emulador de herramienta de inversión

Para hacer que la ejecución del código en Java correspondiera con la ejecución del código en C# dentro del sistema *NinjaTrader* se desarrolló una sección de código que hiciera una conversión en la secuencia de los datos que se cargaron de la base de datos histórica de tipo de cambio.

NinjaTrader hace sus cálculos tomando en cuenta que el valor más reciente es el valor "0". Dado que se conecta a un proveedor de servicios que le alimenta datos en tiempo real, dicho valor "0" siempre es el más reciente. Al desarrollar la primera versión del prototipo de la función objetivo, se hizo tomando en cuenta este detalle técnico, por lo que al cambiar el código a Java se tuvo que hacer un procesamiento similar para la base de datos sobre la cual se correrían los experimentos.

Este procesamiento se hace dentro de la clase EmulaNinjaTrader, dentro del método del mismo nombre, que a su vez, una vez que ha hecho este procesamiento dato por dato, manda llamar a la función objetivo a través del método onBarUpdate para que evalúe todos los datos disponibles cada momento que se agrega un dato nuevo a la base de datos.

Por la manera en que funciona el programa, conforme se procesa la información de la base de datos dato por dato para ir aplicando la estrategia de inversión a cada periodo, se va reenumerando la secuencia de datos siempre que "llega" un dato nuevo, según se describió con anterioridad en el capítulo de análisis, lo cual introduce un orden $O(n^2)$ en el procesamiento de la base de datos.

4.3.4 Módulos adicionales para Opt4j

Opt4J tiene una arquitectura modular para implementar problemas de optimización a evaluar. Los módulos desarrollados son los siguientes, en orden alfabético:

- `EstrategiaRompimientoCanal`: Se encarga de implementar la interface "fenotipo" de Opt4J. En este caso particular es una lista de enteros. No requiere de ningún procesamiento especial ya que los genotipos implementados se usan tal cual como fenotipos.

- `EstrategiaRompimientoCreator`: Se encarga de generar "genotipos" que serán los individuos de la población. En este módulo se definen las cotas inferiores y superiores para cada gen, y luego se generan los genes del genotipo de manera aleatoria dentro las cotas definidas.

- `EstrategiaRompimientoDecoder`: Se encarga de traducir los "genotipos" a "fenotipos". En este caso solo se agregan los elementos del genotipo usando el módulo `EstrategiaRompimientoCanal`.

- `EstrategiaRompimientoEvaluator`: Manda llamar la función objetivo con un individuo de la población de "fenotipos" para que sea evaluado y obtener su ganancia y

pérdida mayor.

- `EstrategiaRompimientoModule`: Encadena las clases `Creator`, `Decoder` y `Evaluator` para que puedan ser usadas por Opt4J.

- `EstrategiaRompimientoProblem`: Inicializa el problema y permite que sea listado en la interface de Opt4J para su ejecución en el proceso de optimización.

Para mayor información se puede consultar (Lukasiewycz *et al.*, 2011) y (Lukasiewycz *et al.*, 2012).

4.4 Equipo de cómputo utilizado

Para la ejecución de los experimentos se utilizó una computadora con las siguientes características físicas:

Computadora Quadcore Intel (4 procesadores 3GHz)

8 Gb RAM

1Tb Disco Duro

4.5 Programas y herramientas utilizadas

Algunas de las herramientas tal vez sean dependientes de la versión de los programas utlizados, por ejemplo el archivo Makefile para realizar la compilación del programa y la construcción del archivo Jar. También el archivo renice-java.bash que se encarga de monitorear los procesos relacionados con java en el sistema para que tengan mayor prioridad de ejecución sobre los demás procesos del sistema. Otro archivo desarrollado fue demonio.csh que permite ejecutar programas en el ambiente gráfico local estando conectado de manera remota. También se generó un programa para calcular un frente de pareto a partir del archivo de las poblaciones finales de individuos obtenidas. Estos programas se pueden consultar en el Apéndice C.

Hay algunos programas que no se incluyen, que se usan para procesar los archivos de texto resultantes de las corridas, que realizan acciones sencillas como por ejemplo: cambiar separadores de espacios por comas, concatenar archivos de datos, separar archivos con varias columnas en un archivo por cada columna, juntar archivos que contengan una columna con otros para generar un archivo con columnas reordenadas, separar datos por cada 10

generaciones para graficar en gnuplot. Dichos programas son “scripts” de 1 línea de código que usa herramientas propias de Linux o algún sistema tipo “unix” como lo son: cat, grep, sort, cut, paste.

A continuación se muestra una lista no exhaustiva de los programas y herramientas utilizadas para el desarrollo y ejecución del programa utilizado en los experimentos, así como de las herramientas auxiliares mencionadas con anterioridad:

- Ubuntu 12.10
- Java 1.7
- Opt4J 2.7
- make 3.82
- bash 4.2.39
- tcsh 6.17.00
- bc 1.06.95

4.6 Diseño de experimentos

Los algoritmos NSGA-II y SPEA2 fueron utilizados para diseñar una estrategia de ruptura. Para esto, se utilizó una base de datos de entrenamiento para el año 2011 del tipo de cambio EURUSD, y se realizó una evaluación posterior con una base de datos para el año 2012. A continuación se describe con más detalle todo el procedimiento.

4.6.1 Calibración de los Algoritmos NSGA-II y SPEA2

Inicialmente se corrieron una serie de experimentos preliminares para tratar de determinar el número de generaciones necesario para un buen entrenamiento en ambos algoritmos. Primero se configuró el algoritmo NSGA-II en Opt4j con 800 generaciones, una población de 100 individuos, de los cuales 100 serían seleccionados como padres, producirían 100 hijos con una probabilidad de cruzamiendo de 0.9 y a los cuales se les aplicaría una mutación con probabilidad de 0.01. El canal se calcularía usando un periodo de 10 barras de precios. Se usaron las bases de datos de precios diarios para los pares de monedas EURCHF, EURGBP, EURUSD, GBPUSD, USDCHF y USDJPY. Los experimentos se corrieron 5 veces y se tomaron los

resultados que produjeron la mejor ganancia por cada corrida. En la Figura 19 se muestran las gráficas de convergencia de la ganancia mayor total para cada corrida y se puede observar que para la mayoría de los pares de monedas se obtiene la mejor ganancia aproximadamente a las 30 generaciones, con la excepción de los pares USDJPY y EURGBP que convergieron a las 120 y 500 generaciones, respectivamente.

Después se realizaron pruebas para tratar de estimar el tiempo de ejecución total de los experimentos, con la finalidad de mantener dicho tiempo dentro de un límite de aproximadamente una semana. Estas pruebas consistieron en correr evaluaciones de 1 solo individuo con 3 diferentes bases de datos, cada una con granularidad de precios como sigue: 1 día, 5 minutos y 1 minuto. En la Tabla 3 se muestra la cantidad de datos aproximada para cada base de datos según su granularidad.

En la bibliografía se encontró que Mendes *et al.* (2012) encontraron de manera experimental que una población de 10 individuos les permitía mantener el tiempo computacional razonable para correr sus simulaciones. Mencionan también desventajas que esta cantidad de inviduos puede provocar en el aprendizaje del algoritmo y define lineamientos para contrarrestar sus efectos. Dichos lineamientos se consideraron para el desarrollo de este trabajo, pero su implementación se pospuso para realizarse en trabajo futuro.

Tabla 3: Cantidad de datos y tiempo de entrenamiento por cada tipo de base en datos de precios diarios.

Período de Entrenamiento	Granularidad de Precios	Cantidad de Datos (Aproximada)	Tiempo de Ejecución
1 año	1 día	200	<1 segundo
1 año	5 minutos	75,000	3.5 minutos
1 año	1 minuto	375,000	1.75 horas

Considerando la situación anterior y tomando en cuenta que se necesitarían aproximadamente 40 generaciones para que convergiera nuestra ganancia por cada experimento, se decidió hacer corridas con 10 y 20 individuos para tratar de determinar si existía alguna ventaja al usar más individuos aunque se tuviera que reducir el número de generaciones de los experimentos. Por lo tanto, para las corridas de 20 individuos se definieron 30 generaciones para obtener un resultado, lo que llevó a tener 600 evaluaciones por experimento que tardarían en ejecutarse aproximadamente 35 horas, necesitando un total de 15 días para correr 10 experimentos. Para las corridas de 10 individuos se seleccionaron 50 generaciones, para que

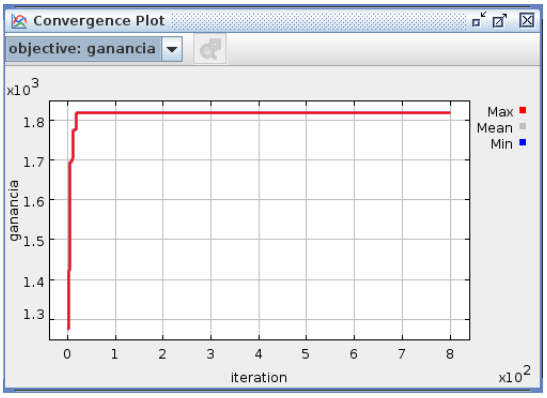
el número de evaluaciones fuera similar al de 20 generaciones, y un poco por encima de la cantidad aproximada de las 40 generaciones necesarias. Con 500 evaluaciones se necesitaron aproximadamente 30 horas, lo que fue un total de 12 días para correr 10 experimentos. Los parámetros seleccionados para el entrenamiento de los algoritmos se muestran en las Tablas 4 y 5.

Posteriormente se realizaron 9 corridas experimentales variando la mutación de 0.01 a 0.015 y 0.005, realizando 3 experimentos por cada mutación, pero no se obtuvieron mejoras notables en los resultados, teniendo aproximadamente las mismas ganancias y las mismas pérdidas mayores. Usando una probabilidad de mutación de 0.01 como parámetro base no se pudo determinar si era mejor o peor que los otros parámetros. También se realizaron 9 corridas experimentales, variando la probabilidad de cruzamiento de 0.8, 0.75 y 0.85, obteniendo igualmente resultados inconclusos. Por estas razones se optó por mantener 0.01 como parámetro de probabilidad de mutación y 0.8 como probabilidad de cruzamiento, manteniendo el apego a los valores mayormente citados en la bibliografia: (Allen y Karjalainen, 1999; Brabazon y O'neill, 2004; Hryshko y Downs, 2004; Tokuoka y Tanaka-Yamawaki, 2008; Hirabayashi *et al.*, 2009; Mishra *et al.*, 2010; Zhang y Ren, 2010; Rasekhi *et al.*, 2011).

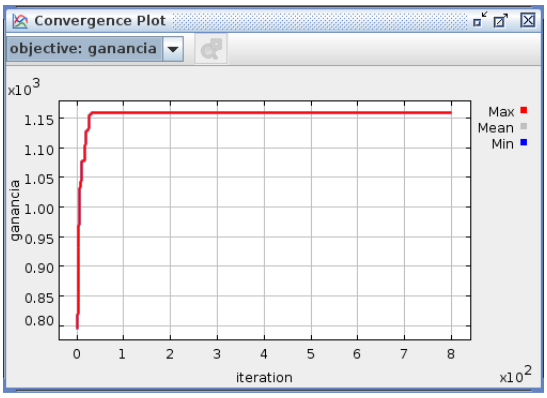
Una vez implementada la función objetivo, se corrieron una serie de experimentos para observar el tiempo de ejecución durante la evaluación de un conjunto de parámetros los cuales correspondían a un individuo de nuestra población de posibles soluciones. Con esto se estimó el tiempo aproximado que tardaría en ser evaluada una población durante cada generación de la simulación. Por otro lado se observó que, para los experimentos de 10 individuos, la población convergía aproximadamente en 40 generaciones, mientras que para poblaciones de 20 individuos se observó que convergían en aproximadamente 30 generaciones.

Tabla 4: Experimentos y parámetros utilizados en el algoritmo NSGA-II.

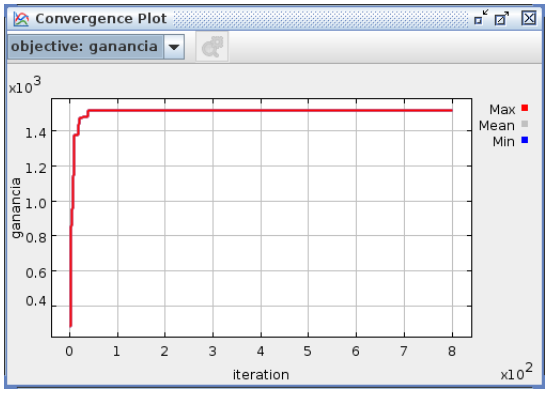
	NSGA-II			
Experimento	1	2	3	4
Clave	NSGAIIX1	NSGAIIX2	NSGAIIX3	NSGAII4
Individuos	10	10	20	20
Generaciones	50	50	30	30
Prob. Mutacion	0.01	0.01	0.01	0.01
Prob.Cruce	0.8	0.8	0.8	0.8
Tipo Cruce	1 punto	Uniforme	1 punto	Uniforme



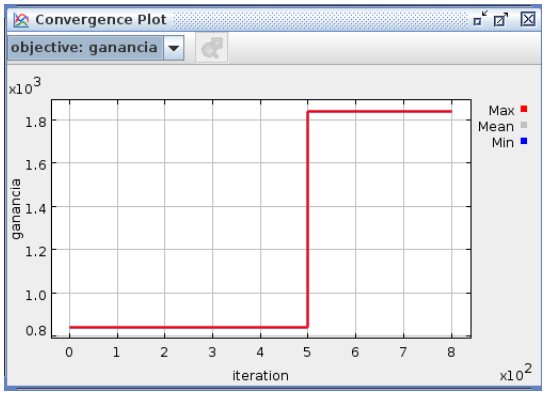
(a) EURUSD



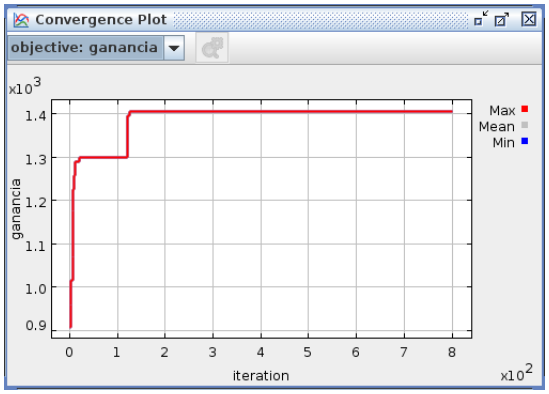
(b) GBPUSD



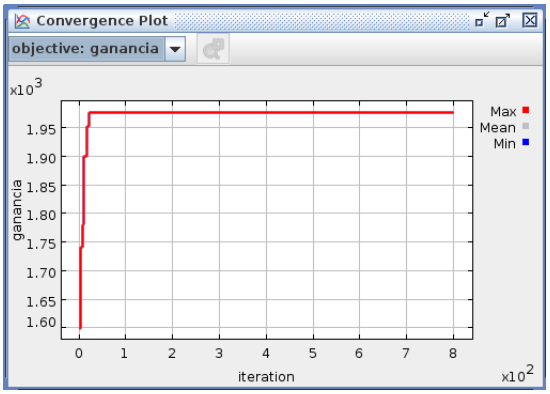
(c) USDCHF



(d) USDJPY



(e) EURGBP



(f) EURCHF

Figura 19: Resultados de experimentos preliminares para determinar el número de generaciones necesarias para ejecutar los algoritmos de optimización.

Tabla 5: Experimentos y parámetros utilizados en el algoritmo SPEA2.

Experimento	1	2	3	4
Clave	SPEA2X1	SPEA2X2	SPEA2X3	SPEA2X4
Individuos	10	10	20	20
Generaciones	50	50	30	30
Prob. Mutacion	0.01	0.01	0.01	0.01
Prob.Cruce	0.8	0.8	0.8	0.8
Tipo Cruce	1 punto	Uniforme	1 punto	Uniforme

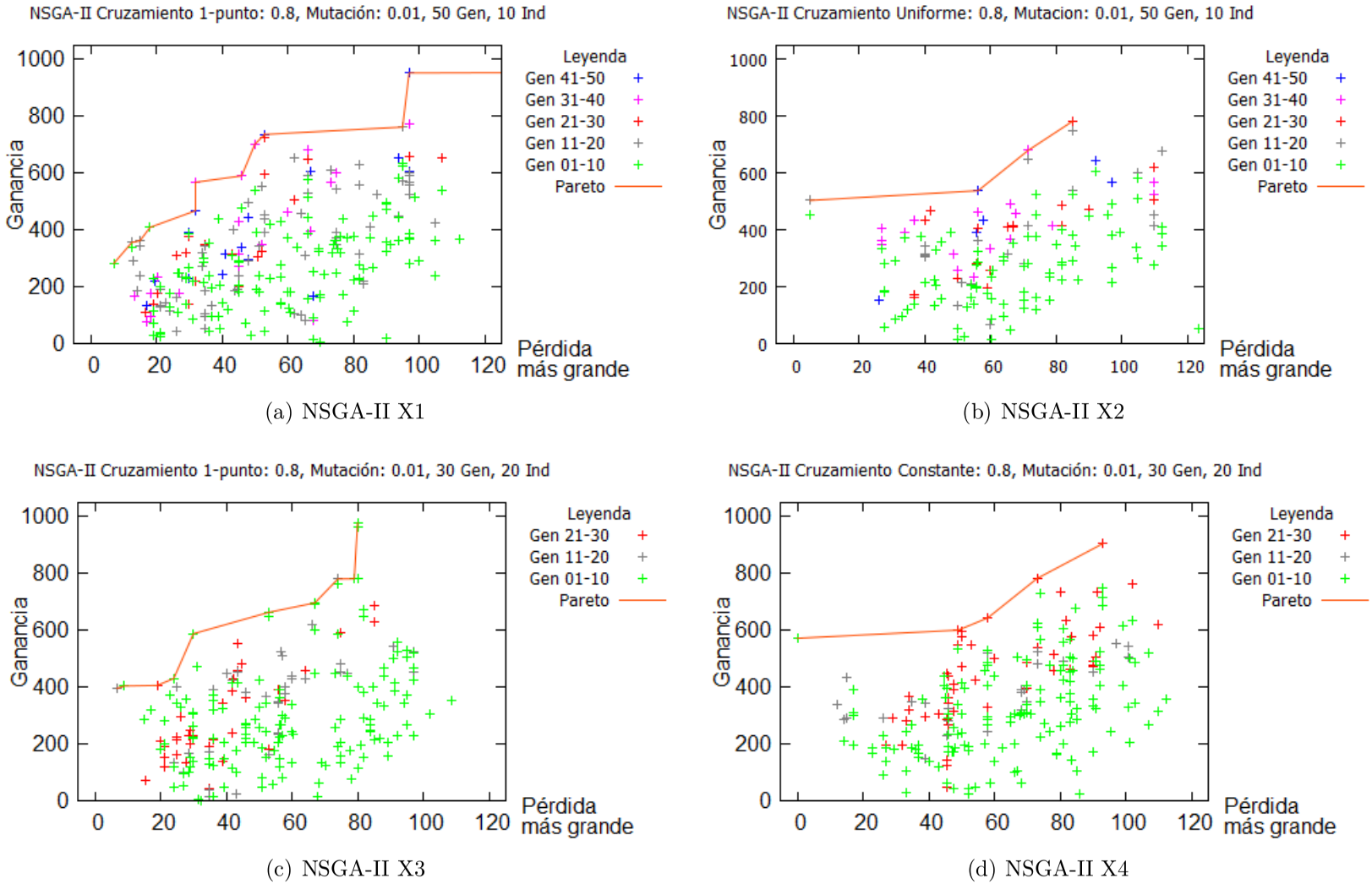
En base a estos resultados experimentales se decidió restringir cada experimento a aproximadamente 500 evaluaciones, para que en total una ejecución tardara alrededor de 30 horas. Cada experimento se repitió 10 veces. Además de variar la cantidad de inviduos se utilizaron 2 tipos diferentes de operaciones de cruzamiento. En las Tablas 4 y 5 se muestran los parámetros utilizados en los experimentos realizados con los algoritmos NSGA-II y SPEA2, respectivamente.

4.7 Resultados

A continuación se presentan los resultados de los experimentos que comprenden la fase de entrenamiento, así como la evaluación de los mejores parámetros obtenidos a partir de estos resultados.

4.7.1 Resultados de la fase de entrenamiento

En las Figuras 20 y 21 se muestran los individuos obtenidos de los experimentos con los algoritmos NSGA-II y SPEA2. Cada eje x representa la pérdida mayor para los experimentos, y cada eje y representa la ganancia acumulada. Los individuos en las figuras se muestran de diferente color dependiendo de la generación en la que fueron obtenidos para tratar de determinar cuantas iteraciones del programa fueron necesarias para obtener los individuos no dominados. Para tratar de obtener los mejores individuos de todos los experimentos se calcularon los frentes de Pareto para cada grupo.Una vez calculados estos frentes, se generó un conjunto nuevo con estos elementos y se volvió a calcular un frente de Pareto para este nuevo conjunto, como se muestra en la Figura 22.



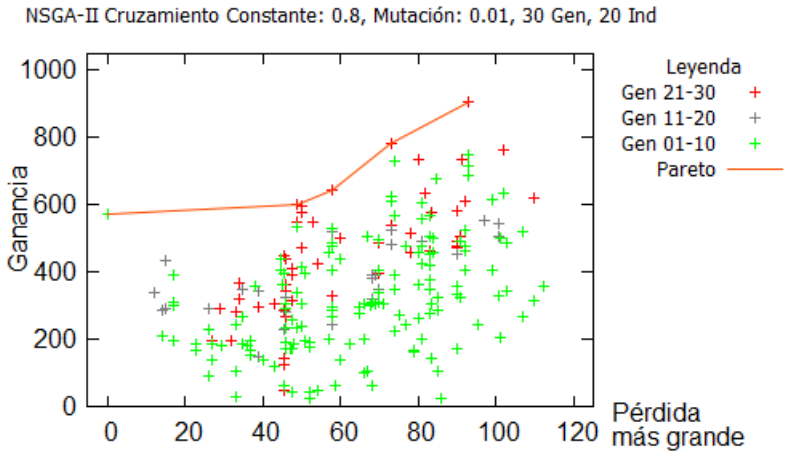
(a) NSGA-II X1

(b) NSGA-II X2

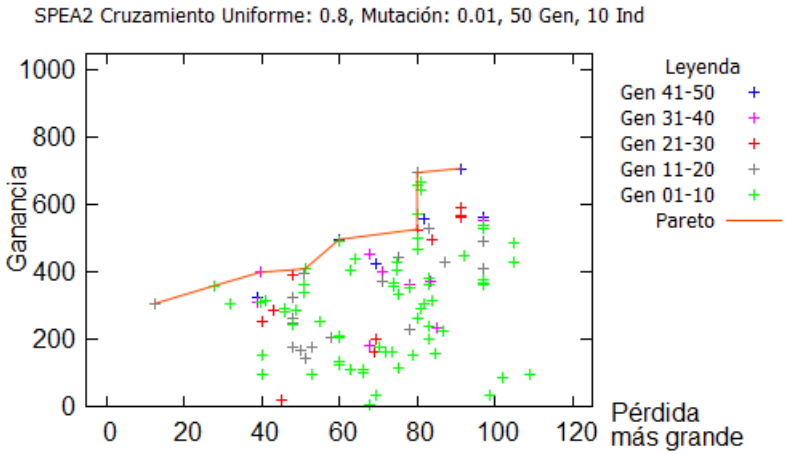
(c) NSGA-II X3

(d) NSGA-II X4

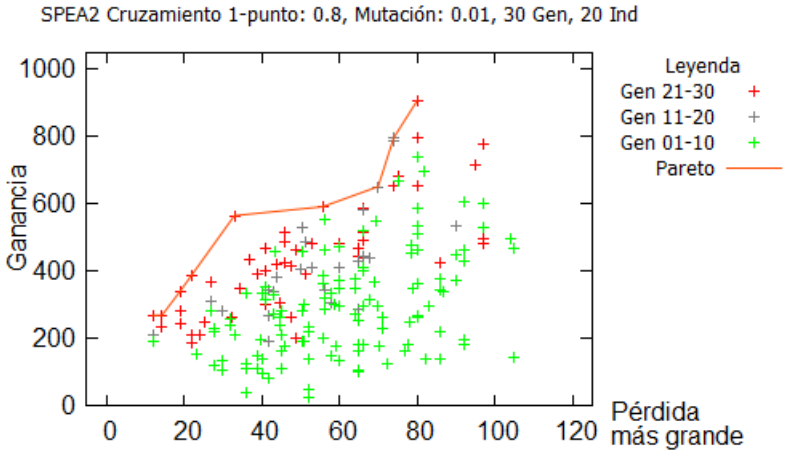
Figura 20: Resultados experimentales con el algoritmo NSGA-II (fase de entrenamiento). (a) Configuración NSGAII-X1, (b) configuración NSGAII-X2, (c) configuración NGSAII-X3 y (d) configuración NSGAII-X4. Las características de cada configuración se muestran en la parte superior de la figura.



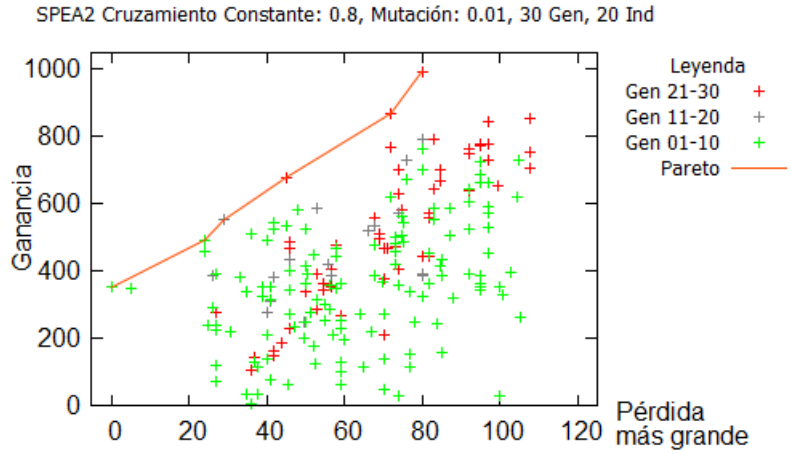
(a) SPEA2 X1



(b) SPEA2 X2



(c) SPEA2 X3



(d) SPEA2 X4

Figura 21: Resultados experimentales con el algoritmo SPEA2 (fase de entrenamiento). (a) Configuración SPEA2-X1, (b) configuración SPEA2-X2, (c) configuración SPEA2-X3 y (d) configuración SPEA2-X4. Las características de cada configuración se muestran en la parte superior de la figura.

Se puede observar que este frente de Pareto consta de 7 individuos, mencionando de izquierda a derecha a que conjunto de Pareto pertenece cada uno de ellos: fp-nx4, fp-nx3, fp-sx4, fp-nx1, fp-nx1, fp-sx4, fp-sx4. A partir de estos frentes se puede obtener la siguiente información: Los elementos que terminan en dígito 3 o 4 provienen de configuraciones con 20 individuos y 30 generaciones de ejecución. Los elementos con terminación 1 o 2 provienen de configuraciones con 10 individuos y 50 generaciones de ejecución.Los individuos que tienen "nx" en su etiqueta provienen de la ejecución de experimentos con el algoritmo NSGA-II, y los que tienen "sx" provienen de experimentos con el algoritmo SPEA2.

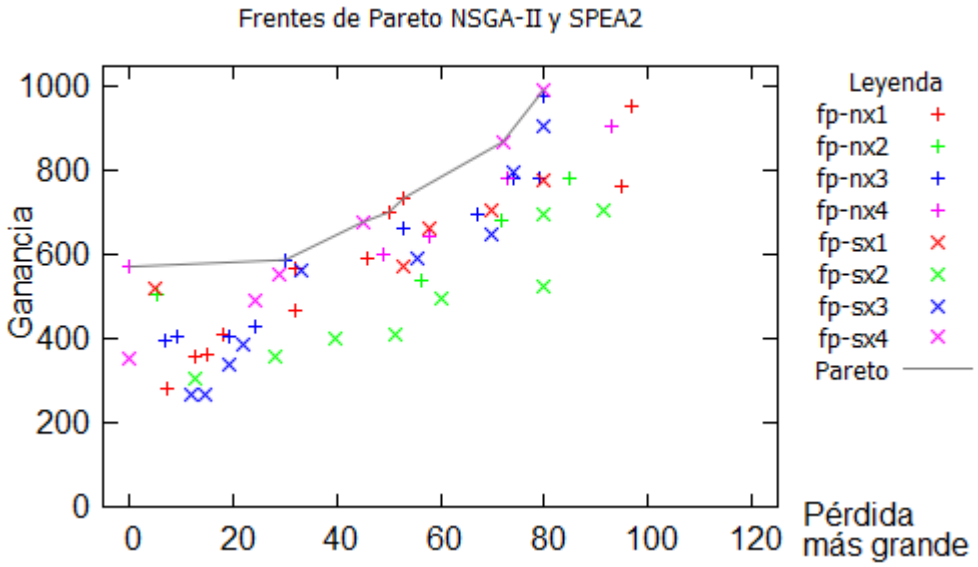


Figura 22: Frente de todos los frentes. De los 7 elementos del frente se tienen 4 generados por el algoritmo NSGA-II y 3 por SPEA2.

Por lo tanto tenemos 5 elementos generados a partir de experimentos con 20 individuos y 30 generaciones: fp-nx4, fp-nx3, fp-sx4, fp-sx4, fp-sx4, mientras que tenemos 2 elementos generados de experimentos con 10 individuos y 50 generaciones: fp-nx1, fp-nx1. También se observa que 4 individuos provienen de la ejecución del algoritmo NSGA-II, y 3 provienen del algoritmo SPEA2.

Por otro lado se puede apreciar que de las 8 configuraciones solamente 3 de ellas se encuentran representadas en este frente de frentes, siendo por ejemplo el frente fp-sx2 dominado casi en su totalidad por todos los demás frentes.

4.7.2 Resultados de la fase de pruebas

Para evaluar el desempeño de los individuos que forman parte del mejor frente de Pareto que se obtuvo de combinar todos los frentes, se ejecutó la función objetivo con una base de datos de tipo de cambio del año 2012. Esta función se ejecutó en 4 diferentes periodos, para tratar de determinar en que periodo de tiempo se tenía una mejor ganancia antes de tener que volver a ejecutar un entrenamiento. Los periodos que se eligieron fueron: enero 2012, enero-marzo 2012, enero-junio 2012, enero-septiembre 2012, y enero-diciembre 2012.

Los individuos seleccionados fueron 5 de un total de 7 individuos. Se descartaron dos elementos, uno perteneciente al frente fp-nx1 y otro perteneciente al frente fp-sx4, por su proximidad al tercer elemento del frente fp-nx1 que se tomó para representar a este conjunto de valores pues sus valores estaban relativamente próximos, comparados con los otros elementos del frente.

Tabla 6: Individuos seleccionados del frente de frentes.

Individuo	Experimento	clave
1	NSGA-II X4	I1-nx4
2	NSGA-II X3	I2-nx3
3	NSGA-II X1	I3-nx1
4	SPEA2 X4	I4-sx4
5	SPEA2 X4	I5-sx4

Uno de los criterios para seleccionar estos 5 individuos fueron su distribución a lo largo de todo el frente, tratando de incluir elementos que representaran al frente incluyendo sus extremos y elementos del centro para no perder información sesgando la muestra y teniendo solamente elementos de un extremo, por ejemplo. Otra consideración fue tratar de incluir diversidad en el tipo de algoritmos utilizados para generar dichos elementos, es decir, mantener elementos provenientes de diferentes configuraciones de experimentos realizados, para de esta manera, tratar de determinar de manera intuitiva cual fue la mejor configuración.

Los individuos seleccionados fueron numerados de izquierda a derecha como se describe en la Tabla 6. Dichos individuos fueron evaluados en los periodos descritos con anterioridad y nos arrojaron las ganancias que se muestran en la siguiente sección. En la Tabla 7 se muetran los resultados de la evaluación de prueba en el periodo de Enero 2012.

Tabla 7: Ganancias y pérdidas para entrenamientos y pruebas de los mejores individuos.

	Entrenamiento		Pruebas 1 mes	
Individuo	Ganancia	Pérdida	Ganancia	Pérdida mayor
I1-nx4	572.2	0	-25.4	49.8
I2-nx3	588.1	30	-87.7	76.4
I3-nx1	701.55	50	-31.1	47.4
I4-sx4	868.85	72	157.4	66
I5-sx4	994.25	80	150.4	58.8

En la Figura 23 se muestra la ganancia acumulada para los diferentes individuos evaluados en la fase de pruebas. Observe que los individuos I4-sx4 y I5-sx4 tienen la ganancia inicial más alta y su ganancia sigue creciendo a lo largo de los primeros 6 meses del periodo de evaluación. Para cuestión de comparación se muestra la estrategia "Buy and Hold" para el mismo periodo. En este caso, dicha estrategia tuvo un desempeño similar durante los primeros 3 meses. Los demás individuos tienen pérdidas durante los primeros 6 meses de la evaluación.

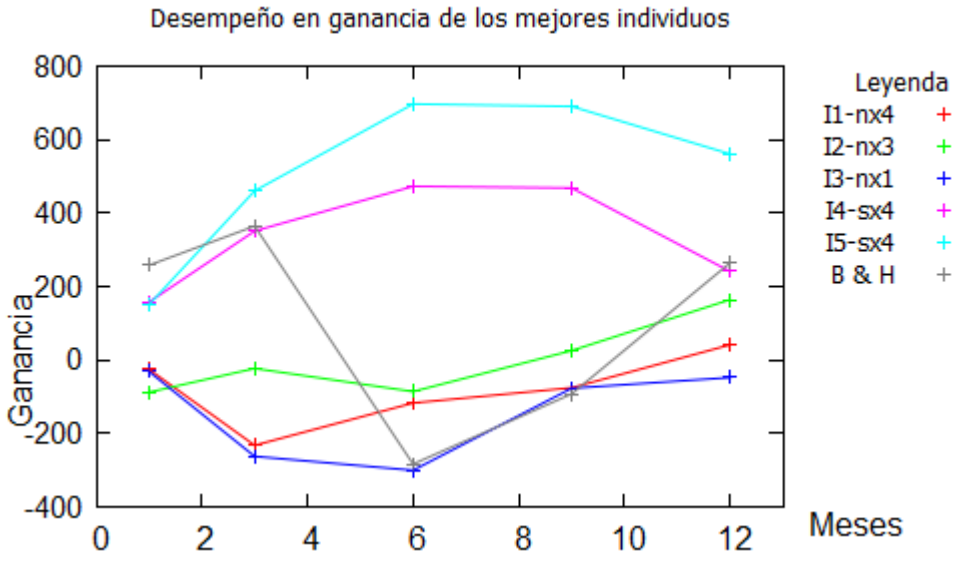
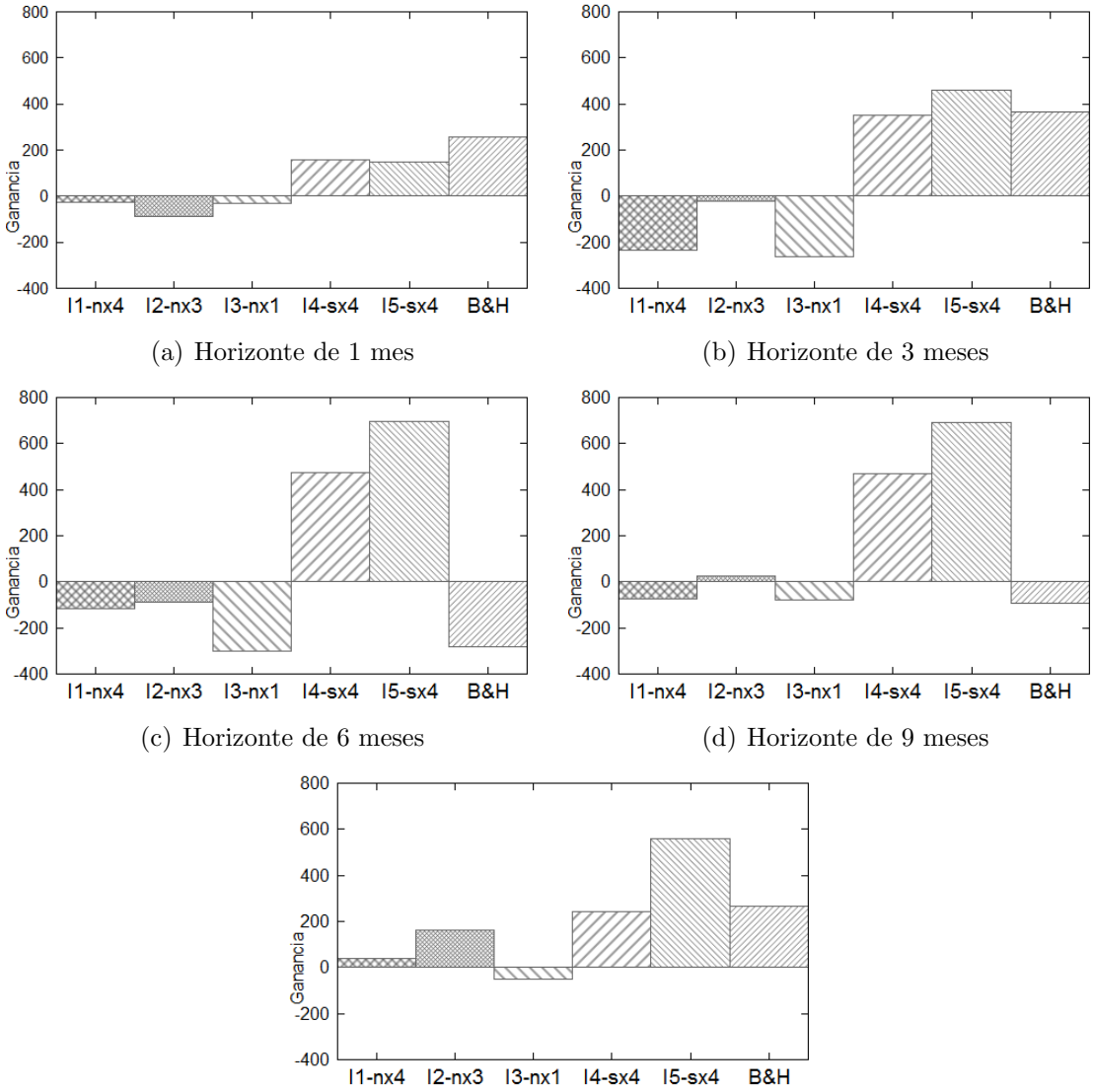


Figura 23: Ganancias de los mejores individuos en el periodo 2012.

En la Figura 24 se muestran las ganancias acumuladas para el final de los diferentes periodos de inversión. En este caso se observa que los individuos I4-sx4 e I5-sx4 tienen ganancia mayor que todos los demás individuos en la mayor parte de los periodos.



(a) Horizonte de 1 mes

(b) Horizonte de 3 meses

(c) Horizonte de 6 meses

(d) Horizonte de 9 meses

(e) Horizonte de 12 meses

Figura 24: Resultados en ganancia (en dólares) de la fase de pruebas para los individuos seleccionados. A manera de comparación se muestra la ganancia obtenida por la estrategia "Buy & Hold" en cada horizonte de inversión. Los periodos de prueba para la figuras son: (a) Enero de 2012, (b) Enero-Marzo de 2012, (c) Enero-Junio 2012, (d) Enero-Septiembre 2012 y para la (e) Enero-Diciembre 2012.

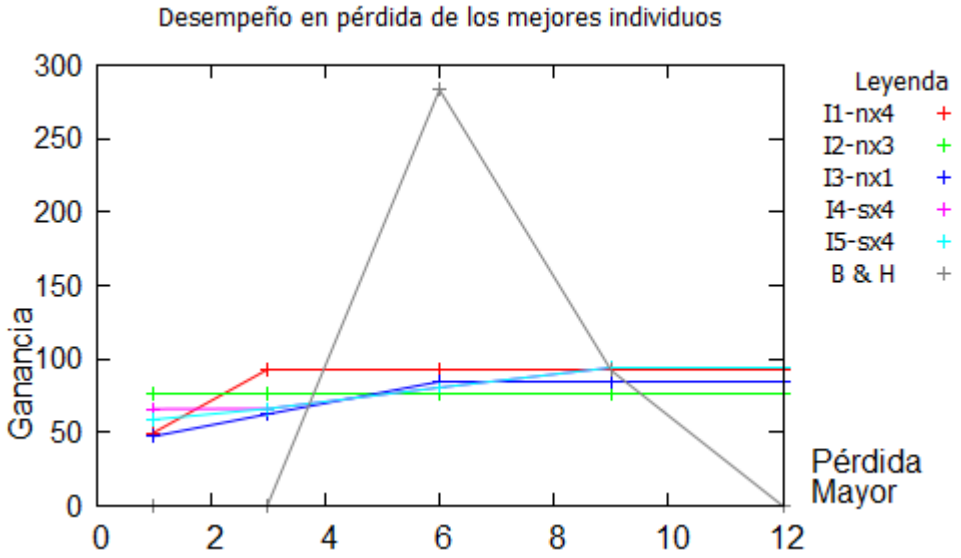


Figura 25: Pérdidas mayores de los mejores individuos en el periodo 2012.

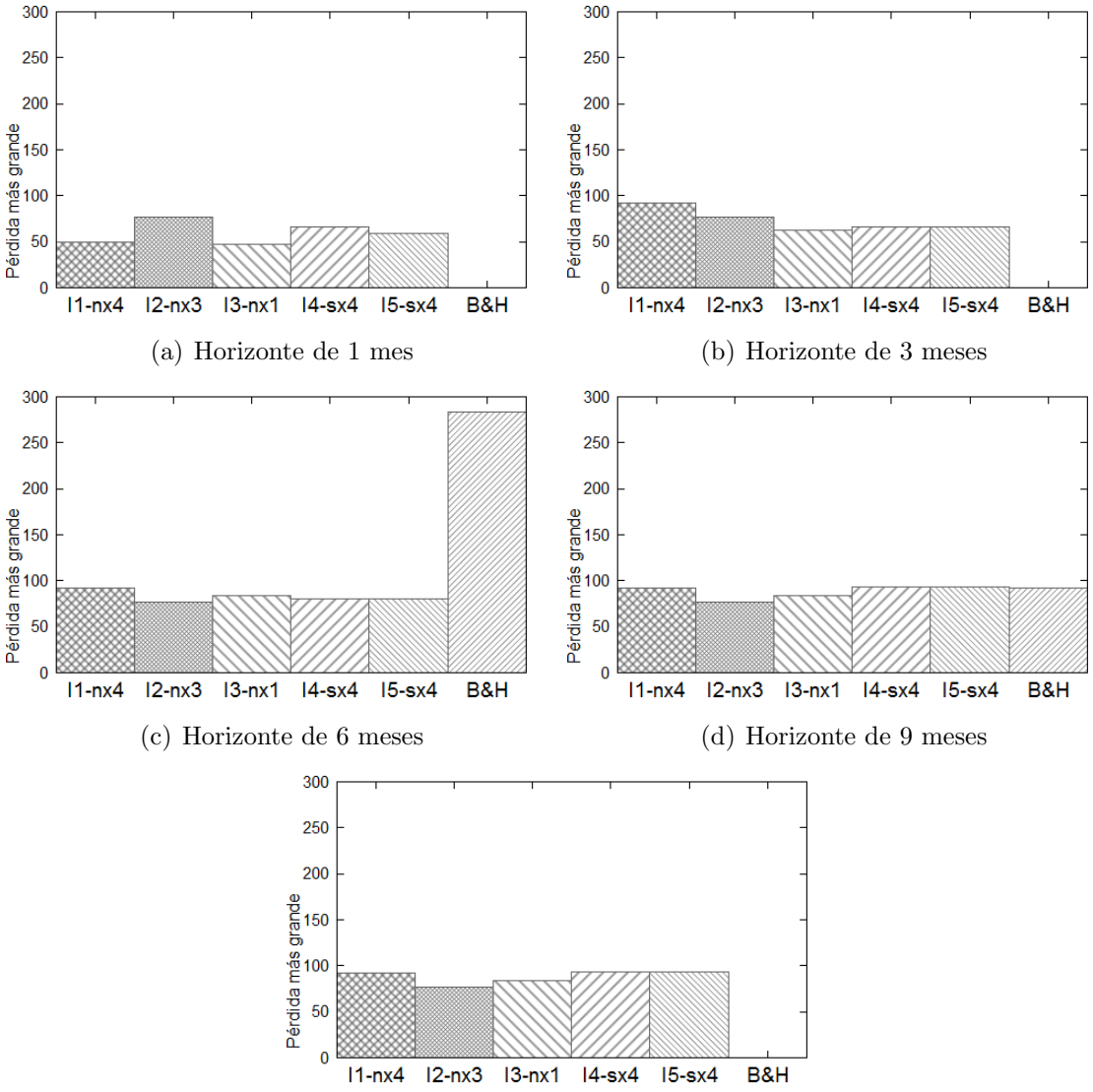
En la Figura 25 se muestran las pérdidas que ocurrieron en los periodos evaluados. Observe que la estrategia "Buy and Hold" tuvo la mayor pérdida acumulada en un periodo de 6 meses y después fue disminuyendo hasta llegar a 0 en 12 meses. En este caso, no existe una pérdida mayor pues la estrategia "Buy and Hold" no realiza diferentes "entradas" de inversiones, por lo que pérdida mayor es la "ganancia negativa" que se tiene para el periodo.

En la Figura 26 se muestran las pérdidas más grandes para cada individuo en cada periodo de evaluación. En esta gráfica se aprecia que todos los individuos sufren una "peor pérdida" aproximadamente similar en cada periodo. También se muestra la pérdida para la estrategia "Buy and hold". Por la naturaleza de dicha estrategia, solo se muestra la pérdida total para cada periodo.

4.7.3 Análisis de resultados de fase de entrenamiento

Para tratar de determinar de manera intuitiva la mejor configuración de un algoritmo, se compararon los frentes de entrenamiento obtenidos durante esta fase. Por ejemplo, la Figura 27(a) muestra la ganancia para las diferentes configuraciones del algoritmo NSGA-II.

Como se puede observar, no existe una dominancia constante de ningún frente. Excepto, en el caso de que el frente producido por x2 es dominado por casi todos los demás frentes. Quitando los elementos de los extremos de los frentes, el algoritmo NSGA-II no arroja diferencias notables en sus diferentes configuraciones.



(a) Horizonte de 1 mes

(b) Horizonte de 3 meses

(c) Horizonte de 6 meses

(d) Horizonte de 9 meses

(e) Horizonte de 12 meses

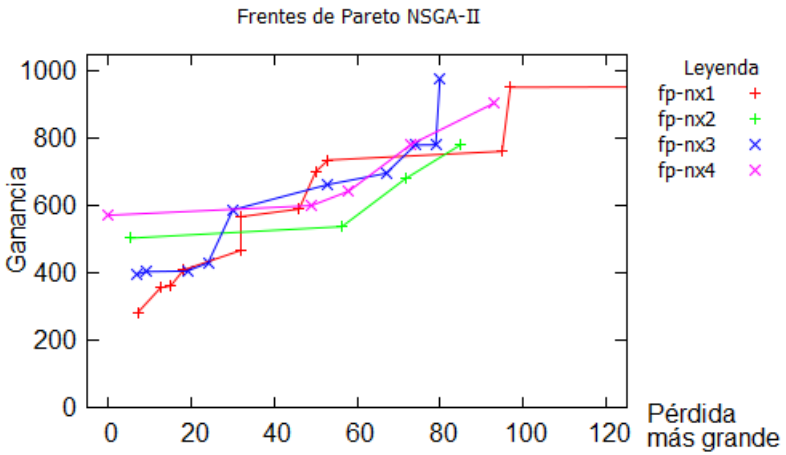
Figura 26: Resultados en pérdida (en dólares) de la fase de pruebas para los individuos seleccionados, incluyendo la estrategia "Buy and Hold", excepto que por la naturaleza de esta última, se muestra la pérdida total para cada periodo. Los periodos de prueba para la figuras son: (a) Enero de 2012, (b) Enero-Marzo de 2012, (c) Enero-Junio 2012, (d) Enero-Septiembre 2012 y para la (e) Enero-Diciembre 2012.

En la Figura 27(b) se puede apreciar que el frente producido por la misma configuración, x2, es dominado por todos los demás frentes, excepto por un único punto. También se aprecia como el frente generado por la configuración x4 es un poco más constante en dominar a todos los otros frentes.

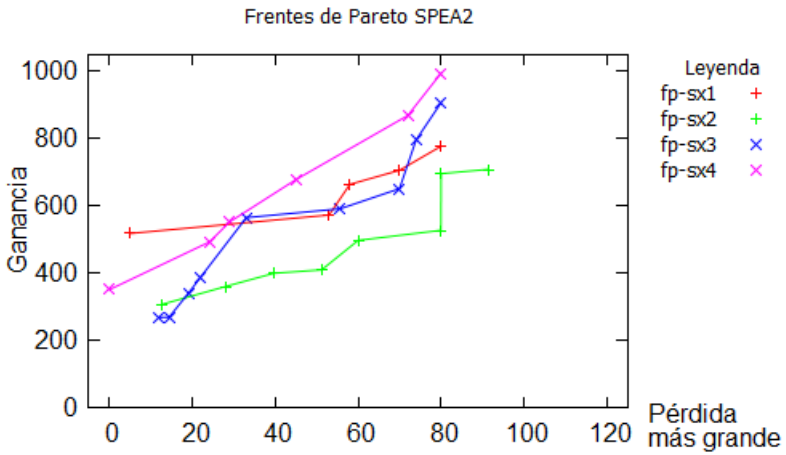
Comparando el desempeño de los frentes generados por cantidad de individuos, empezando por la Figura 27(c) que muestra los frentes generados con experimentos que usaron poblaciones de tamaño 10, se puede observar que en este caso el frente sx2 fue dominado completamente por todos los demás frentes. Dicho frente fue generado por el experimento 2 de SPEA2. También se aprecia una tendencia a dominar la parte derecha de la gráfica del frente nx1, excepto por un solo punto aproximadamente en donde se tiene una pérdida de 100USD, con una ganancia de 700USD, es decir el punto 100,700. Los frentes generados por los experimentos nx2 y sx1 son muy similares.

Para los frentes generados por configuraciones compuestas de 20 individuos, en la Figura 27(d) se puede apreciar que son muy similares, siendo la única que resalta la configuración sx4, generada por el algoritmo SPEA2. Y aunque domina en la parte derecha de la gráfica, es dominado en la parte izquierda por el frente nx4, aunque sigue dominando a los frentes nx3 y sx3. Tambien hay que observar que el frente sx3 es dominado casi en su totalidad por los demás frentes, excepto en la parte derecha de la gráfica donde domina parcialmente a nx3 y nx4.

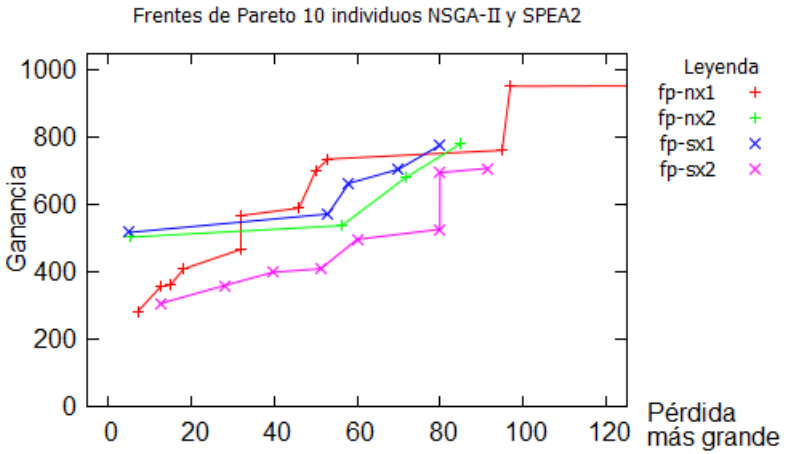
Por otro lado, con respecto a la composición del frente de frentes en la Figura 22, el cual incluye elementos de los experimentos nx1, nx3, nx4 y sx4, se puede observar como los individuos del frente sx2 son dominados casi por todos los demás frentes. Además, todos los demás elementos tienden a aglomerarse cerca unos de otros por lo que tienen aproximadamente la misma aptitud unos con respecto a otros.



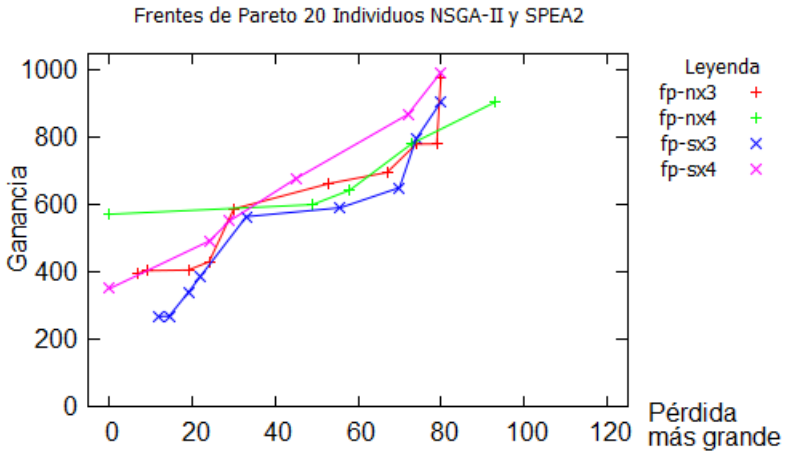
(a) Frentes de pareto NSGA-II



(b) Frentes de pareto SPEA2



(c) Frentes de pareto generados con experimentos de 10 Individuos



(d) Frentes de pareto generados con experimentos de 20 Individuos

Figura 27: Comparación de los diferentes frentes de Pareto.

4.7.4 Análisis de resultados de fase de pruebas

En la Figura 23 se muestran los resultados de la evaluación de los 5 individuos obtenidos del super frente de Pareto. Se puede apreciar que las mejores ganancias acumuladas se obtienen con los individuos 4 y 5 obtenidos con el experimento 4 de SPEA2. Observe que la estrategia Compra-y-mantiene tiene una buena ganancia inicial, pero conforme transcurren los meses se degenera hasta obtener pérdidas. Los individuos 1, 2 y 3 empiezan teniendo pérdidas a partir del primer mes y luego se empiezan a recuperar a partir de los 6 meses.

Cabe mencionar que las ganancias que se muestran en la Figura 23 pueden no representar de manera adecuada la eficiencia de los individuos, pues aunque muestra una ganancia acumulada que se incrementa mientras pasan los meses, en realidad se tiene que la ganancia mensual promedio se decrementa, disminuyendo con cada mes que pasa. Teniendo, que para el mejor individuo, que podría ser el individuo I5 por ejemplo, la ganancia del primer mes es mayor a 100 USD, aunque para el segundo mes, apenas alcanza los 200 USD acumulados, es decir, apenas 100 USD por mes.

En la Figura 25 se aprecia como las pérdidas mayores son relativamente similares para todos los elementos en el primer mes de su evaluación, manteniéndose relativamente estables durantes los diferentes periodos, excepto por la del individuo sx4 que parece ir creciendo en el transcurso del año, aunque no de manera significativa pues solo crece de 60 a 95 USD. Observe sin embargo que no queda mucho peor que las pérdidas a final de año de los demás individuos. Aunque si se considera como crecen aproximadamente 40% su valor inicial, si empeora de manera significativa al compararse con otros individuos como I2, que aunque tiene la pérdida inicial mayor de todas, ésta se mantiene hasta el final del año. Pero si se toma en cuenta que las pérdidas varían, yendo de un rango inicial de 50 a 80 USD el primer mes, y finalizan en un rango de 75 a 95 USD al final del año, este cambio no es tan alarmante.

La estrategia “compra-y-mantiene” tiene un desempeño bastante bueno al inicio, teniendo una pérdida de 0 los primeros 3 meses. A partir de ahí empieza a crecer hasta 3 veces la de los demás individuos. En este caso particular esta estrategia tuvo el mejor desempeño durante estos primeros meses. Y si se toma en cuenta la ganancia para el primer mes de esta estrategia, en este caso, fue mucho mejor que cualquiera de los individuos obtenidos del super-frente de Pareto.

4.8 Resumen

En este capítulo se mencionaron algunos detalles teóricos y técnicos que se tomaron en cuenta para desarrollar la función de aptitud basada en el sistema de inversión "NinjaTrader" y su posterior integración al sistema de optimización Opt4J que se ejecutó en una plataforma Linux. También se presentaron los resultados de las simulaciones experimentales para obtener los mejores individuos posibles en un tiempo finito. Se corrieron 8 experimentos diferentes, cada uno repetido 10 veces durante el periodo de entrenamiento, de donde se seleccionaron los 8 frentes de Pareto respectivos y se combinaron para luego formar un frente de Pareto compuesto de los elementos de todos estos frentes. Posteriormente se evaluaron elementos seleccionados de dicho frente en un periodo de prueba para observar su desempeño. Finalmente se presentó un análisis de los resultados obtenidos, tanto de la fase de entrenamiento como de la fase de pruebas.

Capítulo 5 Conclusiones

En este trabajo se utilizaron los algoritmos NSGA-II y SPEA2 para optimizar los parámetros de una estrategia de inversión para el mercado de divisas. Para lograr tal objetivo se implementó una función para evaluar una base de datos de precios intradiarios para el par de divisas EURUSD a manera de entrenamiento. El entrenamiento emuló el comportamiento de una plataforma de inversión comercial denominada *NinjaTrader*, conforme a la cual se obtuvo un frente de Pareto con los mejores parámetros encontrados, optimizando dos objetivos en el periodo: la ganancia acumulada y la pérdida más grande. Dicho entrenamiento se llevó a cabo utilizando una plataforma de optimización multiobjetivo para el lenguaje de programación Java en donde se integró la función objetivo a través de una serie de módulos. De los resultados ahí obtenidos se escogieron los mejores de todos para ser evaluados en un periodo inmediatamente posterior al periodo de entrenamiento y se comparó la ganancia y la pérdida obtenidas con la estrategia "Buy and hold" que es citada en la literatura relacionada como plataforma de comparación.

5.1 Discusión

En base a los resultados obtenidoss se puede sugerir que la operación de cruzamiento uniforme cuando se aplica a poblaciones de 10 individuos, la cual corresponde a los experimentos nx2 y sx2, arroja resultados peores que los obtenidos con cualquier otra configuración. Por lo que se puede ahorrar tiempo usando otras configuraciones que dan mejores resultados a la hora de hacer entrenamientos. También se tiene que los resultados de cruzamiento de 1 punto, con 20 individuos para SPEA2 fueron un poco peor que los demás, pero no tan consistentemente dominados con los resultados de nx2 y sx2.

Los resultados también sugieren que la configuración de 20 individuos con cruzamiento uniforme para SPEA2 arroja mejores individuos que todos los demás experimentos, dominaron consistentemente a casi todos los otros frentes en todas las comparaciones. Por lo que si se tuviera que decidir en usar una sola configuración con un solo algoritmo, sx4 sería la mejor opción en este caso.

Dado que las ganancias promedio se van degradando con cada mes, se recomienda que los entrenamiento se hagan cuando menos cada mes, para tratar de mantener el nivel de ganancia. En este caso, los dos mejores individuos fueron producidos por SPEA2 con la configuración x4, y fueron los que tenían la mayor ganancia durante el entrenamiento, a

pesar de que su pérdida más grande también fue la mayor, por lo que este factor, en este caso, no parece ser tan importante.

Comparando con nuestra estrategia de referencia, estos resultados parecen no ser bastante buenos, y parece que usar la estrategia compra-y-mantiene es mejor, pero hay que tomar en cuenta que en este caso, el comportamiento para el tipo de cambio EURUSD era creciente los primeros meses del año lo que propició este mejor desempeño.

5.2 Trabajo futuro

- Rediseñar función objetivo para disminuir el tiempo de ejecución. Esto permitirá realizar un mayor número de experimentos.
- Cambiar el genotipo para evaluar su efecto en el cruzamiento de 1 punto. Para esto se considera reordenar el genotipo para que quede simétrico de manera que al cruzarse la mitad sean los parámetros para una entrada larga y la otra mitad sean para una entrada corta.
- Automatizar los procesos de evaluación experimental y evaluación de desempeño. Esto permitirá disminuir el tiempo de análisis necesario para mezclar los resultados experimentales de una secuencia de corridas, la extracción de frentes de pareto, su mezcla en un superfrente, y la evaluación de sus elementos en un periodo de prueba.
- Realizar más experimentos para determinar si el algoritmo SPEA2 con la configuración del experimento 4 siempre tiene mejor desempeño que los demás experimentos.
- Realizar más experimentos para determinar si con los individuos con más ganancia del frente de frentes siempre se tendrá la mejor ganancia en la fase de evaluación, o si conviene más tomar los individuos con las pérdidas menores.
- Realizar más experimentos para tratar de determinar si vale la pena alargar el periodo de entrenamiento para obtener mayores ganancias en el periodo de evaluación.
- Realizar más experimentos para tratar de determinar si acortando el periodo de entrenamiento disminuye la aptitud de las respuestas encontradas relativas a las que se encontraron en estos experimentos.
- Realizar más experimentos para determinar con mayor exactitud el periodo de vida útil de nuestros parámetros de inversion.

- Otra ventaja de realizar una mayor cantidad de experimentos es la posibilidad de poder calcular métricas especializadas para optimización multiobjetivo con la idea de encontrar las configuraciones con mejor desempeño.

- También se podría investigar la posiblidad de comparar los resultados de diferentes tipos de cambio para determinar si existe alguna solución de algún tipo de cambio que tenga ganancia aceptable en otros tipos de cambio.

- Implementar la sustitución de una parte de la población por elementos generados de manera aleatoria, denominados "migrantes" para evitar las desventajas que puede producir una población reducida de individuos (Mendes *et al.*, 2012).

Cabe mencionar que una modificación que se le hizo a la función de aptitud permite guardar cada uno de los individuos generados para evitar la evaluación redundante de invidiuos ya existentes. Si a este archivo se le incluyera el número de generación al que pertenece cada individuo evaluado se tendría un registro de cuantos individuos nuevos habría por generación para tratar de manipular con mayor certidumbre los parámetros de probabilidad de cruzamiento, probabilidad de mutación, y de esta manera obtener estadísticas relacionadas.

Referencias bibliográficas

Allen, F. y Karjalainen, R. (1999). Using genetic algorithms to find technical trading rules. *Journal of Financial Economics*, **51**(2): 245–271.

Alvarez-Diaz, M. y Alvarez, A. (2003). Forecasting exchange rates using genetic algorithms. *Applied Economics Letters*, **10**(6): 319–322.

Becker, Y. L., Fox, A., y Fei, P. (2008). An empirical study of multi-objective algorithms for stock ranking. En R. Riolo, T. Soule, y B. Worzel, editores, *Genetic Programming Theory and Practice V*, capítulo 14, páginas 239–259. Springer, New York, NY.

Brabazon, A. y O'neill, M. (2004). Evolving technical trading rules for spot foreign-exchange markets using grammatical evolution. *Computational Management Science*, **1**(3-4): 311—327.

Brabazon, A., Dang, J., Dempsey, I., O'Neill, M., y Edelman, D. (2012). Natural computing in finance: A review. En G. Rozenberg, T. Bäck, y J. N. Kok, editores, *Handbook of Natural Computing*, Vol. 4, capítulo 51, páginas 1707–1735. Springer, Berlin, Heidelberg.

Budík, J. y Doskočil, R. (2011). Investment portfolio optimization based on genetic algorithm. En E. Freitakas y M. Grigelionis, editores, *Proceedings of the 1st International Scientific Conference Practice and Research in Private and Public Sector - 2011*, páginas 134–141, Vilnius, Lithuania. Mykolas Romeris University. Recuperado de https://www.mruni.eu/en/university/faculties/ekonomikos_fakultetas/conferences/practice_and_research/2011/.

Cekirdekci, M. E. y Iliev, V. (2010). Trading system development: Trading the opening range breakouts. Thesis, Worcester Polytechnic Institute, England. Requirement for the Degree of Bachelor of Science.

Chen, S.-H. y Yeh, C.-H. (1997). Toward a computable approach to the efficient market hypothesis: An application of genetic programming. *Journal of Economic Dynamics and Control*, **21**(6): 1043–1063.

Chen, S. H., Kuo, T. W., y Hoi, K. M. (2008). Genetic programming and financial trading: How much about "what we know". En C. Zopoundis, M. Doumpos, P. M. Pardalos, y D. Z. Du, editores, *Handbook of Financial Engineering*, páginas 99–154. Springer, New York, NY.

Contreras, I., Hidalgo, J. I., Nuñez Letamendía, L., y Y., J. (2012). Parallel architectures for improving the performance of a ga based trading system. En F. Fernádez de Vega, J. I. Hidalgo Pérez, J. Lanchares, y J. Kacprzyk, editores, *Parallel Architectures & Bioinspired Algorithms*, páginas 189–218. Springer, Berlin, Heidelberg.

Deb, K., S., A., Pratap, A., y Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. En M. Schoenauer, K. Deb, R. Günther, X. Yao, E. Lutton, J. J. Merelo, y H. P. Schwefel, editores, *Parallel Problem Solving from Nature PPSN VI, 6th International Conference 2000. Proceedings*, páginas 849–858, Paris, France. Springer. Recuperado de http://repository.ias.ac.in/83498/.

DeJong, K. A. (1975). *An Analysis of the Behaviour of a Class of Genetic Adaptive Systems*. Tesis de doctorado, The University of Michigan, USA.

Dempster, M. A. H., Payne, T. W., Romahi, Y., y Thompson, W. P. (2001). Computational learning techniques for intraday fx trading using popular technical indicators. *Neural Networks, IEEE Transactions on*, **12**(4): 744–754.

Drake, A. E. y Marks, R. E. (1998). Genetic algorithms in economics and finance: Forescasting stock market prices and foreign exchange – a review. Australian Graduate School of Management, University of New South Wales. 28 pp. Recuperado de http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.7.3047&rep=rep1&type=pdf.

D'Souza, R. G. L., Chandra Sekaran, K., y A., K. (2010). Improved nsga-ii based on a novel ranking scheme. *Journal of Computing*, **2**(2): 91–95. Recuperado de: http://arxiv.org/abs/1002.4005.

Dunis, C., Harris, A., Leong, S., y Nacaskul, P. (1999). Optimising intraday trading models with genetic algorithms. *Neural Network World*, **9**: 193–224. 22 pp. Recuperado de http://www.ljmu.ac.uk/AFE/AFE_docs/cibef0499.pdf.

Eiben, A. E. y Smith, J. E. (2003). *Introduction to Evolutionary Computing*. Springer, New York, NY. 530 pp.

Esfahanipour, A. y Mousavi, S. (2011). A genetic programming model to generate risk-adjusted technical trading rules in stock markets. *Expert Systems with Applications*, **38**(7): 8438–8445.

Fernández-Pérez, A., Fernández-Rodríguez, F., y Sosvilla-Rivero, S. (2010). Detecting and exploiting trends in the foreign exchange markets. En *Proceedings of the European Financial Management Association 2010 Annual Meetings - 2010*, Aarhus, Denmark. European Financial Management Association. 41 pp. Recuperado de http://www.efmaefm.org/0EFMAMEETINGS/EFMA%20ANNUAL%20MEETINGS/2010-Aarhus/EFMA2010_0218_fullpaper.pdf.

Fernández-Pérez, A., Fernández-Rodríguez, F., y Sosvilla-Rivero, S. (2012). Exploiting trends in the foreign exchange markets. *Applied Economics Letters*, **19**(6): 591–597.

Galant, M. y Dolan, B. (2007). *Currency Trading For Dummies, Getting Started Edition*. Wiley Publishing, Inc., Hoboken, NJ. 46 pp.

García-Almanza, A., Tsang, E. P. K., y Galvan-Lopez, E. (2008). Evolving decision rules to discover patterns in financial data sets. En E. J. Kontoghiorghes, B. Rustem, y P. Winker, editores, *Computational Methods in Financial Engineering*, páginas 239–255. Springer, Berlin.

Hirabayashi, A., Aranha, C., y Iba, H. (2009). Optimization of the trading rule in foreign exchange using genetic algorithm. En G. Raidl, editor, *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, páginas 1529–1536, Montréal, Canada. ACM. Recuperado de http://www.iba.t.u-tokyo.ac.jp/papers/2009/caranha1GECCO2009.pdf.

Holland, J. H. (1975). *Adaption in Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor, MI. 183 pp.

Holmberg, U., Lönnbark, C., y Lundström, C. (2013). Assessing the profitability of intraday opening range breakout strategies. *Financial Research Letters*, **10**(1): 27–33.

How, J., Ling, M., y Verhoeven, P. (2010). Does size matter? a genetic programming approach to technical trading. *Quantitative Finance*, **10**(2): 131–140.

Hryshko, A. y Downs, T. (2003). An implementation of genetic algorithms as a basis for a trading system on the foreign exchange market. En *Evolutionary Computation, 2003. CEC 03. The 2003 Congress on*, Vol. 3, páginas 1695–1701, Canberra, Australia. IEEE.

Hryshko, A. y Downs, T. (2004). System for foreign exchange trading using genetic algorithms and reinforcemente learning. *International Journal of Systems Science*, **35**(13-14): 763–774.

Hryshko, A. y Downs, T. (2005). A machine learning approach to intraday trading on foreign exchange markets. En M. Gallagher, J. P. Hogan, y F. Maire, editores, *Intelligent Data Engineering and Automated Learning - IDEAL 2005, 6th International Conference, Proceedings*, páginas 588–595, Brisbane, Queensland, Australia. Springer.

Huang, Q. y Lu, M. (2009). Evolutionary discovery of co-movement patterns among foreign currencies. En P. Qiu, C. Yiu, H. Zhang, y X. Wen, editores, *Image and Signal Processing, 2009. CISP '09. 2nd International Congress on*, páginas 1–5, Tianjin, China. IEEE.

Kaboudan, M. A. (2000). Genetic programming prediction of stock prices. *Computational Economics*, **16**(3): 207–236.

Larkin, F. y Ryan, C. (2008). Good news: Using news feeds with genetic programming to predict stock prices. En M. O'Neill, L. Vanneschi, S. Gustafson, A. I. Alcázar, I. De Falco, A. Della Cioppa, y E. Tarantino, editores, *Genetic Programming. 11th European Conference, EuroGP 2008. Proceedings*, páginas 49–60, Naples, Italy. Springer.

Lawrenz, C. y Westerhoff, F. (2003). Modeling exchange rate behavior with a genetic algorithm. *Computational Economics*, **21**(3): 209–229.

Lukasiewycz, M., Glaß, M., Reimann, F., y Teich, J. (2011). Opt4j - a modular framework for meta-heuristic optimization. En N. Krasgonor, editor, *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, páginas 1723–1730, Dublin, Ireland. ACM.

Lukasiewycz, M., Glas, M., y Reimann, F. (2012). *Opt4J Documentation: The Optimization Framework for Java*. 25 pp. Manual técnico incluido en el programa Opt4J. Recuperado de http://opt4j.sourceforge.net/download.html.

Mańdziuk, J. y Jaruszewicz, M. (2011). Neuro-genetic system for stock index prediction. *Journal of Intelligent and Fuzzy Systems*, **22**(2-3): 93–123.

Mayo, M. (2012). Evolutionary data selection for enhancing models of intraday forex time series. En C. Di Chio, A. Agapitos, S. Cagnoni, y C. Cotta, editores, *Applications of Evolutionary Computation. EvoApplications 2012. Proceedings*, páginas 184–193, Málaga, Spain. Springer.

Mehrara, M., Moeini, A., Ahrari, M., y Varahrami, V. (2010). Inefficiency in gold market. *International Research Journal of Finance and Economics*, (43): 42–52. Recuperado de https://www.yumpu.com/en/document/view/5510683/inefficiency-in-gold-market-eurojournals.

Mehta, J. R., Menghini, M. D., y Sarafconn, D. A. (2011). Automated foreign exchange trading system. Thesis, Worcester Polytechnic Institute, England. Requirement for the Degree of Bachelor of Science.

Mendes, L., Godinho, P., y Dias, J. (2012). A forex trading system based on a genetic algorithm. *Journal of Heuristics*, **18**(4): 627–656.

Mishra, S. K., Panda, G., Meher, S., y Majhi, R. (2010). Application of multi-objective evolutionary algorithms in computational finance. En *National Conference on Convergence of Management Practices, National Institute of Technology Warangal (NITW)*, Pune, India. National Institute of Technology, Rourkela. Recuperado de http://hdl.handle.net/2080/1333.

Myszkowski, P. B. y Bicz, A. (2010). Proceedings of the international multiconference on computer science and information technology. En M. Ganzha y M. Paprzycki, editores, *Proceedings of the International Multiconference on Computer Science and Information Technology*, Vol. 5, páginas 81–88, Wisla, Poland. IEEE.

Nison, S. (1991). *Japanese Candlestick Charting Techniques*. New York, NY. 315 pp.

Pilbeam, K. (2005). *Finance and Financial Markets*. Palgrave Macmillan, New York, NY, segunda edición. 479 pp.

Poli, R., Langdon, W. B., McPhee, N. F., y Loza, J. R. (2008). *A Field Guide to Genetic Programming*. New York, NY. 250 pp. Recuperado de http://www.gp-field-guide.org.uk.

Rasekhi, S., Rostamzadehl, M., y Branch, S. (2011). Fundamental modeling exchange rate using genetic algorithm: A case study of european countries. *Journal of Economics and Behavioral Studies*, **3**(6): 352–359.

Santini, M. y Tettamanzi, A. (2001). Genetic programming for financial time series prediction. En J. Miller, M. Tomassini, P. L. Lanzi, C. Ryan, A. G. B. Tettamanzi, y L. W. B., editores, *Genetic Programming. 4th European Conference, EuroGP 2001. Proceedings*, páginas 361–370, Lake Como, Italy. Springer.

Schwaerzel, R. y Bylander, T. (2006). Predicting financial time series by genetic programming with trigonometric functions and high-order statistics. 10 pp. Recuperado de http://www.cs.utsa.edu/~bylander/pubs/pubs.html.

Tanaka-Yamawaki, M. (2004). Tick-wise predictions of foreign exchange rates. En M. G. Negoita, R. J. Howlett, y L. C. Jain, editores, *Knowledge-Based Intelligent Information and Engineering Systems, 8th International Conference, KES 2004, Proceedings, Part I*, páginas 449–454, Wellington, New Zealand. Springer.

Tokuoka, S. y Tanaka-Yamawaki, M. (2008). Trend predictions of tick-wise stock prices by means of technical indicators selected by genetic algorithm. *Artificial Life and Robotics*, **12**(1-2): 180–183.

Wang, P. (2005). Foreign exchange markets and foreign exchange rates. En *The Economics of Foreign Exchange and Global Finance*, capítulo 1, páginas 1–15. Springer, Berlin, Heidelberg.

Welch, I. (2004). *A First Course in Finance*. 609 pp. Recuperado de http://www.freeinfosociety.com/pdfs/misc/introtofinance.pdf.

Wilson, G. y Banzhaf, W. (2010). Proceedings of the 12th annual conference on genetic and evolutionary computation. En J. Branke, editor, *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, páginas 1139–1146, Portland, Oregon, USA. ACM.

Zhang, H. y Ren, R. (2010). High frequency foreign exchange trading strategies based on genetic algorithms. En Z. Li y Z. Hu, editores, *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*, Vol. 2, páginas 426–429, Wuhan, Hubei, China. IEEE.

Zhou, C., Yu, L., Huang, T., Wang, S., y Lai, K. K. (2006). Selecting valuable stock using genetic algorithm. En T. Wang, X. Li, S. Chen, X. Wang, H. Abbass, H. Iba, G. Chen, y X. Yao, editores, *Simulated Evolution and Learning, 6th International Conference, SEAL 2006. Proceedings*, páginas 688–694, Hefei, China. Springer.

Zitzler, E., Laumanns, M., y Thiele, L. (2001). Spea2: Improving the strength pareto evolutionary algorithm. Tik-report 103, Swiss Federal Institute of Technology (ETH) Zurich, Zurich, Switzerland. 21 pp. Recuperado de http://www.tik.ee.ethz.ch/db/public/tik/?db=publications&form=report_single_publication&publication_id=1287.

Apéndice A Pseudocódigo para el algoritmo general de inversión

El siguiente algoritmo se divide en secciones para mejorar su legibilidad en este documento.

Algoritmo 5 Ciclo principal del algoritmo general de inversion

```
    Entrada: Tipos Entero: EntradaLarga, RiesgoLargo, SalidaLarga, EntradaCorta, Ries-
    goCorto, SalidaCorta, HoraInicio, Periodo
    Salida: GananciaTotal, PerdidaMayor
 1: i = 0, GananciaTotal = 0, PerdidaMayor = 0
 2: while (Existe otra Barra[i] en la base de datos) do
 3:    PrecioActual = Barra[i].precio
 4:    if (Existen solicitudes de entradas cortas o largas pendientes) then
 5:       ejecutar entradas cortas o largas pendientes
 6:    end if
 7:    if (Existen solicitudes de salidas cortas o largas pendientes) then
 8:       ejecutar-salidas-pendientes(GananciaTotal, PerdidaMayor)
 9:    end if
10:    if (Barra[i] es la primera primera barra de un periodo) then
11:       definir valores de Entrada, Riesgo y Salida para este periodo en base a los valores
          obtenidos en el canal del periodo anterior
12:    end if
13:    if (Barra[i] es la ultima barra de un periodo) then
14:       obtener-canal(EntradaLarga, RiesgoLargo, SalidaLarga, EntradaCorta, Riesgo-
          Corto, SalidaCorta, HoraInicio, Periodo)
15:    end if
16:    if (posicion = Plana) /*No ha habido entradas*/ then
17:       posicion-plana(PrecioActual, EntradaLarga, SalidaLarga, EntradaCorta, Salida-
          Corta)
18:    end if
19:    if (posicion = Larga) /*ya hubo compras*/ then
20:       posicion-larga(PrecioActual, EntradaLarga, RiesgoLargo, SalidaLarga)
21:    end if
22:    if (posicion = Corta) /*ya hubo ventas*/ then
23:       posicion-corta(PrecioActual, EntradaCorta, RiesgoCorto, SalidaCorta)
24:    end if
25:    if (Barra[i] es la ultima barra de un periodo) then
26:       Hacer calculos finales: alarmas, estadisticas, etc.
27:    end if
28:    i++
29: end while
```

Algoritmo 6 procedimiento para ejecutar solicitudes pendientes

```
    Entrada: Tipos Flotante: GananciaTotal, PerdidaMayor
    Salida: GananciaTotal, PerdidaMayor
1:  ejecutar salidas cortas o largas pendientes y obtener Ganancia
2:  GananciaTotal = GananciaTotal + Ganancia
3:  if (Ganancia < 0) then
4:     Perdida = - Ganancia
5:     if (PerdidaMayor < Perdida ) then
6:        PerdidaMayor = Perdida
7:     end if
8:  end if
9:  return  GananciaTotal, PerdidaMayor
```

Algoritmo 7 procedimiento para posicion plana

```
    Entrada: Tipos Flotante: PrecioActual,EntradaCorta,SalidaCorta,EntradaLarga, SalidaLarga
1:  if (DiadelaSemana ≠ Viernes) //Restriccion de fin de semana then
       // Condicion 1 solicitud de entrada larga: compra
2:     if (PrecioActual ≥ EntradaLarga) and (PrecioActual < SalidaLarga) then
3:        Solicitar entrada larga
4:     end if
       // Condicion 5 solicitud de entrada corta: venta
5:     if (PrecioActual ≤ EntradaCorta) y (PrecioActual > SalidaCorta) then
6:        Solicitar entrada corta
7:     end if
8:  end if
    Fin de Restriccion de fin de semana
```

Algoritmo 8 procedimiento para posicion larga

```
    Entrada: Tipos Flotante: PrecioActual, EntradaLarga, RiesgoLargo, SalidaLarga
    // Condicion 2 tomar ganancias: venta
 1: if PrecioActual ≥ SalidaLarga then
 2:    Solicitar salida larga correspondiente a entrada larga previa
 3: end if
    // Condicion 3 poner advertencia de posible perdida
 4: if PrecioActual < EntradaLarga y PrecioActual > RiesgoLargo then
 5:    Poner advertencia
 6: end if
    // Condicion 4 evitar perdida: venta
 7: if PrecioActual ≤ RiesgoLargo then
 8:    Solicitar salida larga correspondiente a entrada larga previa
 9: end if
    (Evitar variacion de fin de semana: venta)
10: if DiadelaSemana ≠ Viernes y HoradelDia ≥ 1500 then
11:    Solicitar salida larga correspondiente a entrada larga previa
12: end if
```

Algoritmo 9 procedimiento para posicion corta

```
    Entrada: Tipos Flotantes: PrecioActual, EntradaCorta, RiesgoCorto, SalidaCorta
    // Condicion 6 tomar ganancias: compra
 1: if PrecioActual ≤ SalidaCorta then
 2:    Solicitar salida corta correspondiente a entrada corta previa
 3: end if
    // Condicion 7 poner advertencia de posible perdida
 4: if PrecioActual > EntradaCorta y PrecioActual < RiesgoCorto then
 5:    Poner advertencia
 6: end if
    // Condicion 8 evitar perdida: compra
 7: if PrecioActual ≥ RiesgoCorto then
 8:    Solicitar salida corta correspondiente a entrada corta previa
 9: end if
    (Evitar variacion de fin de semana: compra)
10: if DiadelaSemana ≠ Viernes y HoradelDia ≥ 1500 then
11:    Solicitar salida corta correspondiente a entrada corta previa
12: end if
```

Apéndice B Makefile

Archivo para compilar y crea un archivo de tipo .jar a partir de una lista de archivos .java que contienen las diferentes clases necesarias. Se ejecuta con el comando "make". Opcionalmente se puede ejecutar con "make clean" para borrar todas las clases ya generadas.

```
JAVAC=/usr/bin/javac
OPT4J27JAR=.:/home/mbarbosa/opt4j-2.7/opt4j-2.7.jar
PROBLEM=EstrategiaRompimientoProblem
CREATOR=EstrategiaRompimientoCreator
CANAL=EstrategiaRompimientoCanal
DECODER=EstrategiaRompimientoDecoder
EVALUATOR=EstrategiaRompimientoEvaluator
MODULE=EstrategiaRompimientoModule
CARGACONF=CargaArchivoConf
EMULANT=EmulaNinjaTrader
EMULATABLA=EmulaTablaIndi
PRINCIPAL=ArchivoPrincipal
DESTINO=estrategia
JAR=/usr/bin/jar
ARCHIVOJAR=estra
PRUEBA=./prueba-clases.bash



estrategia: $(DESTINO)/$(PROBLEM).class $(DESTINO)/$(CREATOR).class \
                $(DESTINO)/$(CANAL).class $(DESTINO)/$(DECODER).class \
                $(DESTINO)/$(EVALUATOR).class  $(DESTINO)/$(MODULE).class \
                $(DESTINO)/$(CARGACONF).class  $(DESTINO)/$(EMULANT).class \
                $(DESTINO)/$(EMULATABLA).class \
                $(DESTINO)/$(PRINCIPAL).class $(ARCHIVOJAR).jar
```

```
$(DESTINO)/$(PROBLEM).class: $(DESTINO)/$(PROBLEM).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(PROBLEM).java

$(DESTINO)/$(CREATOR).class: $(DESTINO)/$(CREATOR).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(CREATOR).java

$(DESTINO)/$(CANAL).class: $(DESTINO)/$(CANAL).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(CANAL).java

$(DESTINO)/$(DECODER).class: $(DESTINO)/$(DECODER).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(DECODER).java

$(DESTINO)/$(EVALUATOR).class: $(DESTINO)/$(EVALUATOR).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(EVALUATOR).java

$(DESTINO)/$(MODULE).class: $(DESTINO)/$(MODULE).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(MODULE).java

$(DESTINO)/$(CARGACONF).class: $(DESTINO)/$(CARGACONF).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(CARGACONF).java

$(DESTINO)/$(EMULANT).class: $(DESTINO)/$(EMULANT).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(EMULANT).java

$(DESTINO)/$(EMULATABLA).class: $(DESTINO)/$(EMULATABLA).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(EMULATABLA).java

$(DESTINO)/$(PRINCIPAL).class: $(DESTINO)/$(PRINCIPAL).java
	$(JAVAC) -cp $(OPT4J27JAR) $(DESTINO)/$(PRINCIPAL).java

$(ARCHIVOJAR).jar: $(DESTINO)/$(PROBLEM).class $(DESTINO)/$(CREATOR).class \
		$(DESTINO)/$(CANAL).class $(DESTINO)/$(DECODER).class \
		$(DESTINO)/$(EVALUATOR).class $(DESTINO)/$(MODULE).class \
```

```
                $(DESTINO)/$(CARGACONF).class $(DESTINO)/$(EMULANT).class \
                $(DESTINO)/$(EMULATABLA).class \
                $(DESTINO)/$(PRINCIPAL).class
        $(JAR) cvf $(ARCHIVOJAR).jar $(DESTINO)/*.class
        $(PRUEBA)

clean:
        rm -rf estra.jar $(DESTINO)/*.class 2> /dev/null
```

Apéndice C Herramientas desarrolladas

A continuación se presentan algunas de las herramientas desarrolladas como apoyo para procesamiento de datos y ejecución de los experimentos.

C.1 renice-java.bash

Este archivo se ejecuta como administrador (root) y se deja corriendo en el trasfondo del sistema para monitorear los procesos de java y darles más prioridad sobre otros procesos.

```
#!/usr/bin/bash
CORRE="true"

while [ $CORRE == "true" ] ; do
        KK2=‘ps uax | grep java | grep home‘
        #echo $KK2
        PROCESO=‘echo -n $KK2 | cut -f 2 -d ’ ’ ‘
        #PROCESO=‘echo $PROCESO | tr -cd ’[:digit:]’ ‘
        echo $PROCESO
        #echo $PRIO
        /bin/renice -n -17 -p $PROCESO > /dev/null  2>&1
        sleep 1800
        #CORRE="false"
done
```

C.2 demonio.csh

Por restricciones de acceso remoto se creo este programa para poder ejecutar procesos en un ambiente gráfrico local sin estar físicamente en ese lugar. Se necesita acceder remotamente al sistema por una terminal en modo texto y las instrucciones programas serán ejecutadas posteriormente por este programa que deberá estarse ejecutando de manera continua sobre una terminal del ambiente gráfico.

```
#!/bin/csh

set ARCHIVO = "comandos.bash"
set CONFIG = "./demonio.conf"
set RESULTADO = "resultado.txt"

if ( -e $ARCHIVO ) rm $ARCHIVO

set SEGUNDOS = ‘grep -v "^#" $CONFIG| grep espera | cut -f 2 -d ’=’‘
#echo $SEGUNDOS

while ( 1 )
        sleep 1
        if ( -e $ARCHIVO ) then
```

```
                chmod 755 $ARCHIVO
                sleep 1
                date >> $RESULTADO
                ./$ARCHIVO >>& $RESULTADO
                date >> $RESULTADO
                sleep 1
        rm $ARCHIVO
        endif
        sleep $SEGUNDOS
end
if ( -e $ARCHIVO ) rm $ARCHIVO
```

C.3 frentepareto.csh

Este programa obtiene un frente de pareto a partir de un archivo que contenga datos de todos los individuos de las poblaciones finales generadas por los experimentos.

Un ejemplo de contenido de dicho archivo:

```
281.9999999999957,6.9999999999992,nd
356.4000000000078,12.5500000000001,nd
362.149999999999,15.0000000000006,nd
409.3500000000017,18.1000000000009,nd
```

Donde el primer elemento de cada renglón es la ganancia, el segundo elemento es la pérdida mayor, y el tercer elemento una etiqueta para expresar dominancia o no dominancia.

```
#!/bin/csh
# how to add to an array example
#http://www-cs.canisius.edu/ONLINESTUFF/UNIX/shellprogramming.html
#  echo $#name  for size of array
#  set name = (mark sally kathy tony)
#  set name = ($name doran)

# dominated (at first none are dominated)
#set d = ( 0 0 0 0 0 0 0 0 0 0 )
#set g = ( 1 1 1 1 2 2 2 3 3 4 )
#set p = ( 1 2 3 4 2 3 4 3 4 4 )
#echo $p[1] = 1 empiezan en 1

if ( ! -e datossinnega.csv ) then
        printf "File datossinnega.csv does not exist."
        exit
endif

cat datossinnega.csv | cut -f 4,5 -d ',' | sort -n | uniq > datossortuniq.csv
```

```
#ganancia
cat datossortuniq.csv | cut -f 1 -d ',' > dbcol1fp.txt
#perdida
cat datossortuniq.csv | cut -f 2 -d ',' > dbcol2fp.txt

@ t = `wc -l dbcol1fp.txt | cut -f 1 -d ' ' `
@ i = 0

# the contents of these arrays is 0
@ d = ( )
#@ g = ( )
#@ p = ( )

if ( -e dbcol3fp.txt ) rm dbcol3fp.txt

printf "set g = (\n" > dbcol0fp.txt
printf ")\n" > dbcol11fp.txt
printf "set p = (\n" > dbcol12fp.txt
printf ")\n" > dbcol21fp.txt
@ i++
while ( $i < $t )
        #dominated 1
        printf "set g = ( %s\n" \$\g >> dbcol0fp.txt
        printf ")\n" >> dbcol11fp.txt
        printf "set p = ( %s\n" \$\p >> dbcol12fp.txt
        printf ")\n" >> dbcol21fp.txt
        set d = ( $d 0 )
        @ i++
end
paste -d ' ' dbcol0fp.txt dbcol1fp.txt dbcol11fp.txt > datoscol1vars.csh
paste -d ' ' dbcol12fp.txt dbcol2fp.txt dbcol21fp.txt > datoscol2vars.csh

source ./datoscol1vars.csh
source ./datoscol2vars.csh

@ i = 1
@ j = 1

#echo $#d
#set d = ( $d 1 )

while ( $i < $t )
        echo $i $#d
        @ j = $i + 1
        while ( $j <= $t )
                #if ( $i == 1 )
                #echo "$g[$i] ge $g[$j] $p[$i] lt $p[$j]"
```

```
                if ( (‘echo " $g[$i] >= $g[$j]" | bc‘) &&
  (‘echo "$p[$i] < $p[$j]" | bc‘) )  then
                        @ d[$j] = 1
                endif
                if ( (‘echo " $g[$j] >= $g[$i]" | bc‘) &&
  (‘echo "$p[$j] < $p[$i]" | bc‘) )  then
                        @ d[$i] = 1
                endif
                #if ( $i == 1 )
                #echo "$g[$i] gt $g[$j] $p[$i] le $p[$j]"
                if ( (‘echo "$g[$i] > $g[$j]" | bc‘) &&
  (‘echo "$p[$i] <= $p[$j]" | bc‘) )  then
                        @ d[$j] = 1
                endif
                if ( (‘echo "$g[$j] > $g[$i]" | bc‘) &&
  (‘echo "$p[$j] <= $p[$i]" | bc‘) )  then
                        @ d[$i] = 1
                endif
                @ j++
        end
        @ i++
end

#echo $d
#echo $g
#echo $p

@ i = 1
while ( $i <= $t )
        if ( $d[$i] == 0 ) then
                printf "nd\n" >> dbcol3fp.txt
        else
                printf "d\n" >> dbcol3fp.txt
        endif
        @ i++
end

paste -d ’,’ dbcol1fp.txt dbcol2fp.txt dbcol3fp.txt > datosdominancia.csv
cat datosdominancia.csv | grep "nd" > datospareto.csv

rm datoscol?vars.csh datossortuniq.csv dbcol*fp.txt
```

Apéndice D Agradecimientos

Primero que nada, agradezco la ayuda brindada durante la fase inicial de escritura de este documento por el M.C. Gabriel "El chato" Mejia Ruiz para revisar el contenido y depurar mi programa, y por los M.C. Valeria Soto Mendoza y Víctor Manuel Cervantes Salido para definir la estructura la tesis. También a la M.C. Vanessa Miranda López y a Miguel Francisco "El mike" Navarro García por su ayuda para mejorar varias de las figuras en la etapa de revisión y correcciones.

Compañeros

A mis compañeros y compañeras,

Cada generación fue diferente: unos tremendos, otros calmados, en general buena onda todos. Quiero pensar que de todos aprendí algo. Les agradezco su apoyo, amistad, paciencia, llamadas de atención, ayuda, terapiadas, desahogos y todo lo demás.

Aquí les dedico unas palabras tergiversadas:

"...
Desde este día hasta el final del mundo,
siempre nos recordaremos entre nosotros,
los pocos, nosotros pocos, la bola de hermanos;
porque quien haciendo tarea conmigo se haya desvelado,
será mi hermano, aunque sea insoportable,
por esto, lo será un poco menos:
Y los que estén en su lecho de muerte
se arrepentirán de no haber estado aquí,
y se sentirán menos, mientras habla alguien
que se haya desvelado con nosotros en esos días.
... "

– William Shakespeare, Henry V (más o menos).

Generación 2007

José S. Acosta R., Jessica Beltrán M., José L. Beltrán M., Yair Castro G., Raymundo García G., Ricardo Garibay M., Liz I. Gaxiola O., María G. Lugo I., Gustavo F. Mendoza S., Alma L. Nuñez R., Nancy D. Pacheco V., Luis D. Rodríguez A., Ismael Villanueva M.

Generación 2008

Jorge Álvarez L., Fermín A. Armenta C., Orlando Barrera J., Inés Beltrán F., César A. Carranza de la C., Rosario I. Corona de la F., Jorge M. Cortés M., Daniel E. Herández M., Raúl Loredo M., Fernando Luque S., Martín Mancilla G., Vanessa Miranda L., Rene F. Navarro H., Thelma V. Ocegueda M., Marco A. Sánchez D., Javier E. Velarde S., Amado R. Velázquez M.

Generación 2009

Limberg de J. Ake B., Miguel A. Alonso A., Gerardo E. Altamirano G., Jessica Arballo G., Daniel Brubeck S., Karina Caro C., Eddie H. Clemente T., José D. Colbes S., León F.

Dozal G., Paul E. Estrada M., Ulises Gutiérrez O., Martín G. Marmolejo V., Gabriel Mejía R., Francisco I. Pedraza C., Moisés Pérez G., Eduardo Quintana C., Ramón I. Zaragoza C., Héctor Zatarain A.

Generación 2010

Alonso M. Aragón A., Aritz Barrondo C., Víctor M. Cervantes S., Ignacio Cruz P., Mashenka García, Antonio García G., Susana Garduño M., Anuar Lezama B., Cesar C. López P., Eduardo Martínez H., Roberto A. Martínez V., Daniel Miramontes J., César A. Payán R., Euler J. Ponce C., Héctor I. Ramírez C., Valeria Soto M., Gioconda S. Tarqui T., Iván Ulloa E., Rossio C. Uscamayta Q., Héctor Yanajara I., Maythé R. Zuñiga R.

Generación 2011

Ubaldo I. Blanco C., Victoria E. Camacho O., Gustavo Flores B., Luis A. Guerra G., Hugo A. Guillén R., Oscar I. Hernández G., Andersen Herrera H., Oslwaldo Jaime S., Paúl Llamas V., Luz M. Lozano G., Najash Marrón G., Antonio I. Morán M., Rene Olea L., Nelson E. Ordoñez G., Fernando G. Razo M., Efraín Rincón S., David O. Rodríguez U., Neftalí D. Watkinson M., Carlos R. Zalapa C.

Generación 2012

Jesús A. Beltrán V., Franceli L. Cibrian R., Julia Diaz E., Netzahualcoyotl Herández C., Ema L. Lozano G., Rosa A. Ortega del C., Lino A. Rodríguez C., Ana K. Velázquez S., Miguel A. Ylizaliturri S.

Generación 2013

A ellos no los ubico. Sé que existen y están en clase.

Otros Departamentos

Mis compañeros de Herramientas de aprendizaje: Aidee, Andrea, Leopoldo, Sara E. Hernández A., Ekaterina, Ana Alma. Perdón pero no me acuerdo de sus nombres completos.

Estudiantes de Doctorado

Pablo M. Aguilar G., Carlos F. Caloca de la P., Ismael E. Espinoza C. Daniel Fajardo D., Adán Hirales C., David Madrigal A. Violeta Medina R., Ariel A. Quezada P.,

Y perdón si me faltó alguien de las otras generaciones, ya no supe si me faltaba alguna cara conocida, o si recuerdo la cara no recuerdo el nombre. Aquí menciono principalmente a compaẽros de clases o conocidos.

Profesores

A todos los profesores del posgrado que me dieron clase y mencioné al principio: Gracias por la oportunidad. Creo que mordí más de lo que podía tragar.

Amigos

Y por último a mis amigos y amigas, que siempre me animaban y seguían contando conmigo aunque no estuviera presente:

Leticia C. Yescas T., Renee Torres G., Vanessa López N., Claudia L. Chavarín L., Claudia Preciado, April J. Niemela, Elizabeth A. Poulson, Leticia Sotelo D., Eduardo Morales E., Eric O. Sarmiento, Joel Z. García R., Laura Cárdenas, Carlos L. López, Berenice A. Castañeda T., Ángel Barajas, Jeanette Barajas, Bárbara L. Carro L., Leslie L. Alba A., Salvador Mora, Yesenia Trujillo, Sergio Morales, Carlos Yee R., Mauricio Munguía R., Zenaida Paredes T., Juan M. López, Ivan Monay D., Jorge A. Milanez S., Miguel F. Navarro G., Alejandra Chee C., Nora Diaz, Víctor Munguía V., Jorge Márquez, Jerónimo Cambero, Alejandro Sotelo F., Diana Miranda R., Selene Camarena, Carlos Yee, Elvia González, Cathy Ibarra, Cristhian Colbes, Karina Lugo I., Michelle Fernández. Y a la familia Buchannan's, Glenlivet, McAllan, Glenfiddich, Jameson, Smirnoff , Daniels y otras, que junto con mi música, que nunca para de tocar, me acompañaron en largos días con sus noches y en algunos momentos de descanso.

A mis alumnos de seguridad en cómputo que me inspiraron a intentar este viaje: Berenice, Josué, Lorenzo, Izmari, Judith, Roberto, Antonio, Ángel, Selene.

Muchas gracias por sus buenos deseos y palabras de aliento a M.C. Judith I. Luna S., Dra. Constanza Ricaurte V., Josipa Filipovič, Dora Massieu, Ricardo Garduño, Lic. B. Alejandra Fuentes M.

Y finalmente, agradezco a M.C. Miriam Lecuanda O. (de Acuicultura 2011), M.C. Luz M. Lozano G., M.C. Daniel Miramontes J., M.C. Karina Caro C., M.C. Gabriel Mejía R., M.C. Susana Garduño M. su apoyo y compañía durante los momentos nefastos.

...
"Welcome to where time stands still
No one leaves and no one will
Moon is full, never seems to change
Just labeled mentally deranged
Dream the same thing every night
I see our freedom in my sight
No locked doors, No windows barred
No things to make my brain seem scarred

Sleep my friend and you will see
That dream is my reality
They keep me locked up in this cage
Can't they see it's why my brain says Rage"
...

Metallica – Welcome Home (Sanitarium)

Gracias a todos,

Mabs