TESIS DEFENDIDA POR

Everardo Gutiérrez López

Y aprobada por el siguiente comité:

Dr. Carlos Alberto Brizuela Rodríguez

Director del Comité

Dr. Hugo Homero Hidalgo Silva

Miembro del Comité

Dr. Andrei Tchernykh

Miembro del Comité

Dr. José Alberto Fernández Zepeda

Miembro del Comité

Dr. Thang Bui

Miembro del Comité

Dr. Ana Isabel Martínez García

Coordinador del programa en Ciencias de la Computación

Dr. David Hilario Covarrubias Rosales

Director de Estudios de Posgrado

10 de Septiembre del 2009.

CENTRO DE INVESTIGACIÓN CIENTÍFICA Y DE EDUCACIÓN
SUPERIOR DE ENSENADA

PROGRAMA DE POSGRADO EN CIENCIAS
EN CIENCIAS DE LA COMPUTACIÓN

Caracterización de grafos de búsqueda para problemas combinatorios

TESIS

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

DOCTOR EN CIENCIAS

Presenta:

Everardo Gutiérrez López

Ensenada, Baja California, México. Septiembre del 2009.

RESUMEN de la tesis que presenta **Everardo Gutiérrez López**, como requisito parcial para obtener el grado de DOCTOR EN CIENCIAS en CIENCIAS DE LA COMPUTACIÓN. Ensenada, B. C. Septiembre del 2009.

Caracterización de grafos de búsqueda para problemas combinatorios

Resumen aprobado por:

Dr. Carlos Alberto Brizuela Rodríguez

Director de Tesis

La existencia de problemas de optimización combinatoria para los cuales no se conocen algoritmos que puedan resolver cualquier caso en un tiempo razonable ha motivado el desarrollo de métodos alternativos que permitan obtener soluciones aceptables desde un punto de vista práctico. Una de esas alternativas son los algoritmos denominados de *búsqueda local*, los cuales se han utilizado exitósamente en varias aplicaciones reales.

La predicción del desempeño de los métodos de búsqueda local para resolver problemas de optimización combinatoria pertenecientes a la clase *NP-difícil* es una problemática aun por resolverse. Para ello se debe considerar que la definición de vecindario es un aspecto clave en los algoritmos de búsqueda local, y por cada problema Π puede existir más de un vecindario N cada uno de los cuales genera su propia estructura, es decir, su *grafo de búsqueda*. Debido a la complejidad que representa el obtener esa predicción, la mayoría de los trabajos previos se han concentrado en caracterizar, de forma teórica y experimental, la dificultad de casos específicos, o de vecindarios específicos, al ser atacados con el paradigma de búsqueda local. Los estudios experimentales han propuesto principalmente la definición de medidas de dificultad que permitan diferenciar los casos sencillos de los difíciles.

La primera parte de este trabajo presenta un análisis experimental sobre un conjunto de problemas pertenecientes a la clase *NP-difícil*, lo cual denominamos aquí como caracterización de grafos de búsqueda. Los experimentos consisten en analizar las propiedades de los grafos de búsqueda para los casos de los problemas propuestos utilizando tres medidas de dificultad: longitud de correlación (CL), estructura de *big valley* (BV) y propiedad de expansión local (LE). El poder de predicción de estas medidas se analiza para una búsqueda local voraz simple ($SGLS$) y una búsqueda local iterativa (ILS), al ser aplicadas a casos de los problemas: asignación cuadrática (QAP), localización de radio bases (BSL), agente viajero (TSP) y flujo de tareas (FSP)

Los resultados obtenidos muestran cómo ninguna de las medidas de dificultad resultó ser un buen predictor del desempeño de los algoritmos de búsqueda local utilizados sobre todos los conjuntos de casos de los problemas de prueba. A pesar de que las tres medidas predicen adecuadamente el desempeño de las búsquedas locales para los casos del QAP, no logran hacer lo mismo para los casos del TSP y BSL, mostrando que no son aplicables a cualquier

problema *NP-difícil*. Una contribución importante de este trabajo, es el establecimiento de preguntas cuyas respuestas permitirían obtener un mejor entendimiento de las capacidades de predicción de las medidas analizadas.

La segunda parte aborda la propuesta de un nuevo algoritmo de optimización multi-objetivo. Se propone adoptar el algoritmo "Go With the Winners" (GWW), el cual es utilizado para verificar la propiedad de expansión local (LE) en grafos de búsqueda, para tratar con problemas multi-objetivo. Se ha probado que el GWW es capaz de encontrar el óptimo global, con alta probabilidad, realizando un muestreo uniforme de los grafos de búsqueda que poseen la propiedad de LE. En este trabajo se propone una versión multi-objetivo del algoritmo GWW al cual se le agregan métodos básicos de búsqueda local como mecanismo de escape de los frentes no-dominados. Los resultados muestran que el algoritmo propuesto obtiene resultados competitivos tanto en su versión original como en sus versiones híbridas.

Palabras clave: Grafos de Búsqueda, Problemas Combinatorios, QAP, BSL, TSP, ILS, $MOGWW$.

ABSTRACT of the thesis presented by **Everardo Gutiérrez López**, as a partial requirement to obtain the DOCTOR degree in COMPUTER SCIENCES. Ensenada, B. C. September 2009.

Search graphs characterization for combinatorial optimization problems

Abstract approved by:

Dr. Carlos Alberto Brizuela Rodríguez

Thesis director

The existence of combinatorial optimization problems for which there are no known algorithms that can solve any instance efficiently, has motivated the development of alternative methods to obtain acceptable solutions for specific sets of instances. One of such alternatives are the *local search* (LS) algorithms, which have been successfully used in practice.

Performance prediction for local search methods dealing with optimization problems belonging to the *NP-hard* class is an open problem. When dealing with such problem we need to take into account how the neighborhood is defined since it is a key aspect in the local search performance. Moreover, for each problem Π there could be more than one neighborhood N and each one of them generates its own *search graph* on which the search is performed. Performance prediction represents a complex problem and previous attempts have focused only on characterizing, theoretically and experimentally, specific instances or specific neighborhoods difficulty when tackled with the local search paradigm. The experimental studies have proposed difficulty measures with the aim of separating hard instances from the easy ones.

The first part of this work presents an experimental analysis over a set of test problems belonging to the *NP-hard* class, we call this analysis search graph characterization. The experiments focus on analyzing the search graphs characteristics for a set of instances by using three difficulty criteria: Correlation Length (CL), Big Valley structure (BV), and Local Expansion (LE) property. The prediction power of these criteria are analyzed for a Simple Greedy Local Search ($SGLS$) and the Iterated Local Search (ILS), when they are applied to instances of the Quadratic Assignment Problem (QAP), Base Station Location (BSL), Traveling Salesman Problem (TSP), and the Flow Shop Problem (FSP).

The obtained results show that none of the analyzed difficulty measures is a good performance predictor for the local search algorithms used over the set of analyzed instances. Although the three criteria predict well the LS performance for the QAP instances, they fail to do so for the TSP and BSL instances, showing that they are not applicable to any *NP-hard* problem. An important contribution of this research, regarding these measures,

is the statement of questions whose answers would allow us to have a better understanding of their prediction capability.

The second part deal with the proposal of a new multi-objective optimization algorithm. We propose to adopt the "Go With the Winners" (GWW) algorithm, which is used to verify the LE expansion property of search graphs, to deal with multi-objective combinatorial problems. It has been shown that the GWW is able to find the global optimum with high probability by performing a uniform sampling of search graphs that have the LE property. This uniformity is a desirable property for multi-objective algorithms, therefore we propose a multi-objective version of the go with the winners ($MOGWW$) and we add local search methods to generate a hybrid version to escape from non-dominated traps and improve its performance. Experimental results show that the proposed algorithm performs competitively both in its original and hybrid versions.

Keywords: Search Graphs, Combinatorial Problems, QAP, BSL, TSP, ILS, $MOGWW$.

A mi familia

Agradecimientos

Agradezco a todas aquellas personas que me apoyaron durante esta etapa de mi vida, motivándome a no rendirme y a salir adelante con este proyecto. En el listado menciono algunos nombres, algunos títulos y algunos compromisos, en todo caso espero agradecer en persona a todos aquellos que debieron aparecer en esta lista y no están.

A mis padres, hermanos y sobrinos.

A todos los amigos que, de una forma u otra, me apoyaron durante esta larga jornada.

A mi asesor, el Dr. Carlos Alberto Brizuela Rodríguez, por brindarme de nuevo su confianza y el apoyo necesario para sortear uno a uno los problemas que se me presentaron en la realización de este trabajo.

A los miembros de mi comité, Dr. José Alberto Fernández Zepeda, Dr. Hugo Hidalgo Silva, Dr. Andrei Tchernykh y Dr. Thang Bui, por sus valiosas aportaciones y críticas.

Al Consejo Nacional de Ciencia y Tecnología.

Ensenada, México — Everardo Gutiérrez López
10 de Septiembre del 2009.

Tabla de Contenido

Capítulo Página

Tabla de Contenido (Continuación)

Capítulo Página

Lista de Figuras

Lista de Figuras (Continuación)

Lista de Tablas

Lista de Tablas (Continuación)

Capítulo I

Introducción

I.1 Búsqueda Local y Grafos de Búsqueda

La existencia de problemas de optimización combinatoria para los cuales no se conocen algoritmos que puedan resolver cualquier caso en un tiempo razonable (Garey y Johnson, 1979) ha motivado el desarrollo de métodos alternativos que permitan obtener soluciones aceptables desde un punto de vista práctico. Una de esas alternativas son los métodos denominados de *búsqueda local* (*LS* por sus siglas en inglés), los cuales se han utilizado exitósamente en varios problemas del mundo real (Aarts y Lenstra, 1997; Du y Pardalos, 1999).

Los algoritmos de búsqueda local basan su comportamiento general en la siguiente idea: tomar una solución inicial y modificarla para obtener otra que la mejore y repetir este proceso hasta que no sea posible una nueva mejora. Utilizando esta idea general, se han propuesto una gran cantidad de algoritmos y variantes que se comportan de esa forma. Estos algoritmos se han utilizado ampliamente para encontrar soluciones aproximadas para problemas de optimización combinatoria pertenecientes a la clase *NP-difícil* (Aarts y Lenstra, 1997; Du y Pardalos, 1999; Garey y Johnson, 1979). Se conoce como *problema de optimización* Π a un conjunto de casos, donde un *caso* está dado por la especificación de un par $(\mathbf{S}, f)$, siendo $\mathbf{S}$ el conjunto de todas las soluciones factibles y f la función de costo que representa un mapeo $f : \mathbf{S} \rightarrow R$ (Papadimitriou y Steiglitz, 1982).

La función N sobre la cual trabajan los algoritmos de búsqueda local se llama *vecindario*, la cual representa un mapeo $N : \mathbf{S} \rightarrow 2^{\mathbf{S}}$, de tal forma que define para cada solución $s \in \mathbf{S}$ un subconjunto $N(s) \subseteq \mathbf{S}$ de *vecinos* de s. Sobre esa estructura se realiza la búsqueda

local hasta que se cumple el criterio de parada, lo cual ocurre en la mayoría de los casos al encontrarse lo que se conoce como un óptimo local. Se puede definir a una solución como *óptimo local* con respecto a N como una solución $s \in \mathbf{S}$ que cumpla que $f(s) \leq f(t)$ para toda $t \in N(s)$, mientras que un *óptimo global* es una solución s^* tal que $f(s^*) \leq f(s)$ para toda $s \in \mathbf{S}$ (en el caso de problemas de minimización).

La definición de vecindario es un aspecto clave en los algoritmos de búsqueda local, y por cada problema Π puede existir más de un vecindario N, cada uno de los cuales genera su propia estructura sobre la cual se debe realizar la búsqueda. Estas estructuras se llaman *grafos de búsqueda* (Carson, 2001; Dimitriou y Impagliazzo, 1996) para procedimientos generales de búsqueda local, mientras que en computación evolutiva se llaman *fitness landscapes*, en donde fitness es la calidad de las soluciones y fitness landscape es el mapeo de la calidad sobre todas las soluciones (Jones, 1995; Reidys y Stadler, 2002; Weinberger, 1990).

La utilización de los métodos de búsqueda local sobre los problemas de optimización combinatoria pertenecientes a la clase *NP-difícil* plantea la problemática de predecir *a priori* el desempeño a obtener. Debido a la complejidad que representa el obtener esa predicción, la mayoría de los trabajos previos se han concentrado en caracterizar, de forma teórica (Johnson *et al.*, 1988; Papadimitriou *et al.*, 1990; Schäffer y Yannakakis, 1991) y experimental (Boese *et al.*, 1994; Carson, 2001; Jones y Forrest, 1995; Mattfeld *et al.*, 1999; Weinberger, 1990), la dificultad de casos específicos al ser enfrentados con el paradigma de búsqueda local. En el caso de los estudios experimentales, se han enfocado principalmente a la definición de medidas de dificultad que permitan diferenciar los casos sencillos de los difíciles.

Los resultados teóricos sobre la complejidad de los métodos de búsqueda local no resultan alentadores, al menos desde un punto de vista del peor caso (Johnson *et al.*, 1988; Papadimitriou *et al.*, 1990; Schäffer y Yannakakis, 1991). Sin embargo, los algoritmos de búsqueda local han mostrado ser muy eficientes en la obtención de buenos resultados para

muchos de los casos de problemas del mundo real (Aarts y Lenstra, 1997; Du y Pardalos, 1999). Esta situación motiva a realizar un análisis experimental de la dificultad de los grafos de búsqueda de los problemas de optimización combinatoria y la efectividad de las búsquedas locales sobre conjuntos de casos que se puedan caracterizar.

I.2 Descripción del Problema

Un *grafo de búsqueda* para el problema Π es un grafo $G = (\mathbf{V}, \mathbf{E})$, donde $\mathbf{V}$ es el conjunto de vértices ($|\mathbf{V}| = |\mathbf{S}|$) y $\mathbf{E}$ es el conjunto de aristas. Cada vértice se asocia a una solución $s \in \mathbf{S}$ y se etiqueta con el valor $f(s)$. Las aristas en el grafo definen las relaciones de vecindad entre los vértices, de tal forma que el vecindario de cada vértice se representa por el conjunto de vértices adyacentes (Carson, 2001; Dimitriou y Impagliazzo, 1996).

La estructura del grafo de búsqueda depende claramente de la función de vecindario, la cual se relaciona a su vez con la representación de las soluciones. Existen diversos trabajos que intentan caracterizar los grafos de búsqueda utilizados por los algoritmos de búsqueda local al tratar con problemas *NP-difíciles* (Boese *et al.*, 1994; Carson, 2001; Jones y Forrest, 1995; Mattfeld *et al.*, 1999; Weinberger, 1990). La mayoría de ellos se centran en analizar las características principales de los grafos de búsqueda y su influencia en el desempeño de los algoritmos de búsqueda local. Las principales características que se consideran en esos análisis son la relación costo-distancia (Boese *et al.*, 1994; Reeves, 1999; Jones y Forrest, 1995; Manderick *et al.*, 1991; Mattfeld *et al.*, 1999; Weinberger, 1990) y la conectividad del grafo (Bierwirth *et al.*, 2004; Carson, 2001).

La relación costo-distancia en un grafo de búsqueda se puede analizar desde varias perspectivas. En una de las más utilizadas se mide la correlación entre las soluciones inicial y final después de una caminata aleatoria a través del cálculo de lo que se conoce como "longitud de correlación" (CL por sus siglas en inglés) (Weinberger, 1990; Manderick *et al.*, 1991; Stadler y Schnabl, 1992; Stadler, 1995; Mattfeld *et al.*, 1999; Bierwirth *et al.*, 2004).

Los trabajos previos sobre esta medida han mostrado experimentalmente sobre los casos de prueba una relación inversa entre la dificultad del caso y el valor de CL que presenta su grafo de búsqueda. Otra perspectiva que se ha utilizado ampliamente, es la que realiza un análisis visual de la relación costo-distancia en una muestra uniforme del conjunto de óptimos locales (LOS por sus siglas en inglés), en aras de verificar la existencia de la estructura conocida como "Big Valley" (BV) (Boese *et al.*, 1994; Reeves, 1999; Borges y Hansen, 2000; Gutiérrez y Brizuela, 2006). Los resultados experimentales obtenidos previamente al utilizar esta perspectiva han mostrado la utilidad de que el grafo de búsqueda presente la estructura de BV para ciertos algoritmos de búsqueda local. Sin embargo, para ambas perspectivas de análisis de la relación costo-distancia, los trabajos previos dejan sin resolver cuestiones como la existencia de un umbral que permita discriminar los casos por su dificultad, así como la suficiencia de estas medidas como condiciones para garantizar un buen desempeño de las búsquedas locales.

Por otro lado, la conectividad de los grafos de búsqueda también se considera como un aspecto importante a considerar al analizar los casos de los problemas *NP-difíciles* (Aldous y Vazirani, 1994; Dimitriou y Impagliazzo, 1996; Carson, 2001). La idea es que una buena conectividad determinaría que es posible llegar al óptimo global a partir del resto de vértices, por lo que es deseable que el grafo de búsqueda posea tal propiedad. Una forma de realizar la medición de la conectividad de los grafos de búsqueda es a través de la propiedad conocida como "expansión local" (LE por sus siglas en inglés). Sin embargo, la complejidad en el cálculo directo de esa propiedad para grafos de búsqueda de tamaño exponencial, entre otros problemas, ha motivado el uso de un test experimental (Carson, 2001) para verificar su existencia (Dimitriou y Impagliazzo, 1996; Carson, 2001). Si bien algunos trabajos han mostrado la existencia de la propiedad de LE en los grafos de búsqueda de muchos de los problemas para los que las búsquedas locales han mostrado un buen desempeño, existen diversas cuestiones que siguen sin ser resueltas. Una de las preguntas que permanece sin respuesta es si existe una relación entre la dificultad de un problema que muestre la

propiedad de expansión local y el valor de longitud de de-correlación que muestra su grafo de búsqueda, mientras que otra sería la suficiencia de esta medida como condición para garantizar el buen desempeño de los algoritmos de búsqueda local.

Hasta aquí se comenta la importancia del análisis de los grafos de búsqueda así como las propuestas de medidas para caracterizar la dificultad de los problemas. Sin embargo, en este trabajo se plantea analizar las preguntas aún sin resolver sobre esas medidas, de tal forma que se pueda determinar si se pueden considerar como predictores confiables de la dificultad de casos de los problemas de prueba al intentarse resolver con algoritmos de búsqueda local.

Por otro lado se considera el que muchos de los problemas de optimización se derivan de problemas de la vida real que requieren de la optimización de más de un objetivo a la vez, los cuales sin embargo, suelen replantearse a versiones mono-objetivo para utilizar metodologías cuyo desempeño se ha estudiado ampliamente en la literatura (Aarts y Lenstra, 1997; Hoos y Stützle, 2005; Coello *et al.*, 2002; Deb, 2001). Así como el hecho de que los esfuerzos por plantear estrategias que lidien con problemas multi-objetivo se han enfocado mayormente en la extensión de métodos que tratan con casos mono-objetivo, y el que algunas de las estrategias más exitosas caen en el esquema general de las búsquedas locales y han mostrado buenos resultados desde el punto de vista práctico (Serafini, 1994; Hansen, 1997; Gandibleux *et al.*, 1997; Mariano y Morales, 1999; Gambardella *et al.*, 1999; Deb, 2001; Thompson, 2001). Por lo tanto, otro problema que se aborda en este trabajo tiene que ver con el diseño de un nuevo algoritmo de optimización multi-objetivo que utilice algoritmos básicos de búsqueda local como parte integral de su funcionamiento.

I.3 Objetivos

I.3.1 Objetivo general

El objetivo principal gira en torno a dos aspectos fundamentales: el estudio de las medidas de dificultad CL, BV y LE sobre un conjunto de casos de problemas de prueba y el posterior análisis de la capacidad de predicción que tienen sobre el desempeño de los algoritmos de búsqueda local. El segundo aspecto trata con la extensión del algoritmo GWW para resolver problemas de optimización multi-objetivo.

I.3.2 Objetivos específicos

1. Estudiar, analizar y entender las medidas de "dificultad" existentes para los grafos de búsqueda inducidos por los algoritmos de búsqueda local y verificar su alcance.

   (a) Probar las medidas propuestas en Mattfeld *et al.* (1999); Weinberger (1990); Boese *et al.* (1994) para determinar la complejidad de grafos de búsqueda de los problemas: QAP, BSL, TSP, FSP.

   (b) Comparar el grado de dificultad obtenido en cada medida contra resultados experimentales de casos representativos de esos problemas que posean los grafos de búsqueda analizados.

   (c) Analizar los resultados y determinar cuál medida caracteriza mejor la dificultad de los grafos de búsqueda analizados.

2. Analizar y entender los métodos existentes para verificar la propiedad de expansión local en los grafos de búsqueda sobre los que operan los algoritmos de búsqueda local.

   (a) Utilizar las pruebas propuestas por Carson (2001) para verificar la propiedad de expansión local en grafos de búsqueda de los problemas: QAP, BSL, TSP, FSP.

3. Determinar las características principales de los grafos de búsqueda, así como su impacto en el comportamiento de los algoritmos de búsqueda local.

   (a) Determinar cuáles son las características comunes en grafos de búsqueda para los problemas: QAP, BSL, TSP, FSP, así como la forma de medirlas.

   (b) Analizar el impacto de estas características en el desempeño de un conjunto de algoritmos de búsqueda local.

4. Modificar el algoritmo GWW para que pueda resolver problemas de optimización multi-objetivo.

I.4 Metodología de Investigación

Para alcanzar los objetivos de este trabajo, se propone realizar un análisis experimental sobre un conjunto de problemas pertenecientes a la clase *NP-difícil*. Los experimentos a realizar consisten en analizar las propiedades de los grafos de búsqueda para los casos de los problemas propuestos, seleccionando para ello las principales medidas de dificultad propuestas en la literatura. De tal forma que los resultados obtenidos permitan caracterizar los problemas a partir de los grafos de búsqueda de sus casos y predecir el desempeño de los algoritmos de búsqueda local sobre ellos. Para ello se seleccionaron tres medidas de dificultad, cuatro problemas pertenecientes a la clase *NP-difícil* y dos algoritmos de búsqueda local que se utilizan en la fase experimental.

Las características que se consideran para el análisis de los grafos de búsqueda son la relación costo-distancia (Boese *et al.*, 1994; Reeves, 1999; Jones y Forrest, 1995; Manderick *et al.*, 1991; Mattfeld *et al.*, 1999; Weinberger, 1990) y la conectividad del grafo (Bierwirth *et al.*, 2004; Carson, 2001). La primera característica se analiza desde dos perspectivas. La primera de ellas mide la correlación entre las soluciones inicial y final después de una caminata aleatoria (Weinberger, 1990; Manderick *et al.*, 1991; Stadler y Schnabl, 1992;

Stadler, 1995; Mattfeld *et al.*, 1999; Bierwirth *et al.*, 2004). Por otro lado, la segunda analiza la relación costo-distancia en un sub-conjunto de los óptimos locales para verificar la existencia o no de una estructura (Boese *et al.*, 1994; Reeves, 1999; Borges y Hansen, 2000; Gutiérrez y Brizuela, 2006). En el caso de la conectividad de los grafos de búsqueda, la existencia de una buena conectividad es deseable en el grafo de búsqueda. Por lo que se propone realizar la medición de esta característica a través de un test experimental que verifica su existencia (Dimitriou y Impagliazzo, 1996; Carson, 2001).

Dentro de los problemas a considerar, el primero de ellos es el Problema de Asignación Cuadrática (QAP por sus siglas en inglés), el cual se ha convertido en un problema de referencia para analizar el desempeño de heurísticas y otros métodos (Taillard, 1995; Burkard *et al.*, 1998; Stützle, 1999; Ahuja *et al.*, 2000; Lourenco *et al.*, 2002). Este problema muestra una gran diversidad en las características de sus casos, algunos de ellos muestran la estructura de BV y algunos no, convirtiéndolo en un buen candidato para verificar la efectividad de un conjunto de criterios de dificultad (Merz y Freisleben, 2000; Angel y Zissimopoulos, 2001, 2002). El segundo es el Problema de Localización de Radio Bases (BSL por sus siglas en inglés), proveniente del diseño de redes inalámbricas (Calégari *et al.*, 1997; Krishnamachari y Wicker, 2000; Brizuela y Gutiérrez, 2003), el cual muestra diferentes estructuras en dos vecindarios estudiados por Brizuela y Gutiérrez (2003). Se considera también el Problema del Agente Viajero (TSP por sus siglas en inglés), el cual se puede utilizar para modelar un gran número de problemas del mundo real (Gutin y Punnen, 2002; Johnson y McGeoch, 1997) y ha sido uno de los problemas combinatorios más estudiados (Gutin y Punnen, 2002). Finalmente, se seleccionó el problema de calendarización de flujo de trabajos (FSP por sus siglas en inglés) (Tamaki y Nishino, 1998; Bagchi, 1999; Brizuela y Aceves, 2003).

En cuanto a los algoritmos, se seleccionaron dos de los algoritmos de búsqueda local más sencillos para el análisis del desempeño. El primero de ellos es la búsqueda local voraz simple (Papadimitriou y Steiglitz, 1982), la cual es a su vez la base para muchos de los

algoritmos de búsqueda local más complejos (Aarts y Lenstra, 1997; Hoos, 1998). El otro algoritmo es el conocido como búsqueda local iterativa (ILS por sus siglas en inglés), el cual posee diversos mecanismos que permiten sintonizar sus componentes para buscar el balance adecuado entre intensificación y diversificación de la búsqueda, lo que lo ha llevado a ser uno de los algoritmos más eficaces para tratar de resolver problemas de la clase *NP-difícil* (Lourenco *et al.*, 2002; Gutin y Punnen, 2002).

El trabajo de investigación se plantea a través de hipótesis generales, las cuales resultan de analizar las medidas de dificultad a considerar para caracterizar los grafos de búsqueda. Posteriormente, se utilizan los resultados sobre los grafos de búsqueda de los casos analizados para establecer hipótesis específicas, derivadas de las generales, sobre los problemas de prueba. Finalmente, se realizan los experimentos requeridos para probar o rechazar esas hipótesis específicas y se utilizan los resultados para realizar las conclusiones generales de este trabajo de investigación.

I.5 Organización de la Tesis

Este trabajo consta de siete capítulos y un apéndice.

En el Capítulo 2 se realiza una revisión bibliográfica de las principales medidas de dificultad propuestas en la literatura así como los resultados obtenidos con cada una de ellas. Se propone la utilización de tres medidas de dificultad: la longitud de correlación, la estructura de los óptimos locales y la propiedad de expansión local, para el análisis de los grafos de búsqueda, y se establecen las hipótesis generales de trabajo basadas en los resultados previos.

En el Capítulo 3 se selecciona el conjunto de problemas pertenecientes a la clase *NP-difícil* que se utiliza en las pruebas experimentales. Se selecciona además un par de algoritmos de búsqueda local que sirven para verificar las hipótesis de desempeño basadas en las medidas de dificultad.

En el Capítulo 4 se define la problemática de optimización desde el punto de vista multi-objetivo. Se propone un nuevo algoritmo para tratar este tipo de problemas y se presentan dos variantes híbridas del mismo.

En el Capítulo 5 se presentan los resultados experimentales del análisis de los grafos de búsqueda de los casos de los problemas de prueba. Se plantean hipótesis específicas para los resultados obtenidos sobre esos conjuntos de casos y se verifica experimentalmente su validez utilizando los algoritmos de búsqueda local descritos en el Capítulo 3.

En el Capítulo 6 se muestran los resultados de los experimentos sobre los casos multi-objetivo utilizando el algoritmo propuesto en el Capítulo 4 al igual que sus variantes. Se muestra el desempeño del algoritmo propuesto sobre un par de problemas multi-objetivo y después se analiza la mejora obtenida al utilizar las versiones híbridas propuestas.

En el Capítulo 7 se discuten los resultados obtenidos en este trabajo de investigación, se presentan las conclusiones generales al respecto y se plantea el trabajo futuro.

Finalmente, en el Apéndice A se presenta la verificación experimental de la propiedad de expansión local utilizando la prueba de de-correlación. Se detallan los resultados de los experimentos realizados para garantizar que los valores reportados de longitud de caminata aleatoria necesarios para de-correlacionar soluciones (L) son suficientes para probar la existencia de la propiedad de LE en los grafos de búsqueda analizados.

Capítulo II

Tres criterios como medidas de dificultad

Los problemas de optimización combinatoria que se analizan en este trabajo pertenecen a la clase *NP-difícil*, para éstos no se conoce algoritmos que puedan resolver cualquier caso en un tiempo razonable (Garey y Johnson, 1979). Esto ha motivado la utilización de alternativas, como los llamados algoritmos de búsqueda local, en un intento por obtener soluciones aceptables desde un punto de vista práctico (Aarts y Lenstra, 1997; Du y Pardalos, 1999) y dejar de lado los métodos exactos. Para analizar la dificultad de los problemas y la forma en que se puede medir, se deben tomar en cuenta los resultados teóricos presentes en la literatura. El desempeño de algunos métodos sobre el citado conjunto de problemas se ha analizado desde el punto de vista teórico (Johnson *et al.*, 1988; Papadimitriou *et al.*, 1990; Schäffer y Yannakakis, 1991) y práctico (Boese *et al.*, 1994; Carson, 2001; Jones y Forrest, 1995; Mattfeld *et al.*, 1999; Weinberger, 1990), en este último caso enfocado principalmente a la definición de medidas de dificultad que permitan diferenciar los casos sencillos de los difíciles, desde un punto de vista práctico. En este capítulo se presentan los resultados obtenidos al respecto, así como las medidas que se utilizan para tratar de caracterizar la dificultad de los casos al utilizar algoritmos de búsqueda local para atacar problemas pertenecientes a la clase *NP-difícil*.

Contribución. En este capítulo se describen tres medidas de dificultad y en base a ellas se formulan hipótesis sobre el desempeño de las búsquedas locales al utilizarse en problemas de optimización combinatoria.

II.1 Antecedentes

Los resultados teóricos sobre la complejidad de los métodos de búsqueda local no resultan alentadores, al menos desde un punto de vista del peor caso (Johnson *et al.*, 1988; Papadimitriou *et al.*, 1990; Schäffer y Yannakakis, 1991). Analizando la complejidad de la búsqueda local, Johnson *et al.* (1988) introducen una nueva clase llamada PLS (polynomial-time local search). Dentro de esa clase de problemas se encuentran los *PLS-completos* que son los más difíciles. Algunos de los vecindarios pertenecientes a este grupo son el *k-intercambio* y el *Lin-Kernighan* (Lin y Kernighan, 1973) para el problema del agente viajero (TSP por sus siglas en inglés), *Swap* y *Kernighan-Lin* para el problema de Partición de Grafos, *Flip* para Max-Cut, entre otros (Krentel, 1989; Papadimitriou, 1992; Schäffer y Yannakakis, 1991). Se ha demostrado que más de la mitad de los problemas mencionados en Garey y Johnson (1979) como parte de la clase *NP-completa* tienen búsquedas locales equivalentes que pertenecen a la clase *PLS-completa* (Vaessens *et al.*, 1998). A pesar de estos resultados teóricos, los algoritmos de búsqueda local han mostrado ser eficientes en la obtención de buenos resultados en muchos casos. Esta situación motiva a realizar un análisis experimental de la dificultad de los grafos de búsqueda de los problemas de optimización combinatoria y la efectividad de las búsquedas locales sobre conjuntos de casos que se puedan caracterizar.

Para analizar la dificultad de los grafos de búsqueda se considera su estructura, la cual depende claramente de la función de vecindario, que a su vez está relacionada con la representación de las soluciones. En otras palabras, los conceptos de vecino, vecindario, óptimo local, entre otros mencionados en la introducción de este trabajo, dependen de la función **N**. Existen diversos trabajos que intentan caracterizar los grafos de búsqueda utilizados por los algoritmos de búsqueda local al tratar con problemas *NP-difícil* (Boese *et al.*, 1994; Carson, 2001; Jones y Forrest, 1995; Mattfeld *et al.*, 1999; Weinberger, 1990). La mayoría de ellos se centra en analizar las características principales de los grafos de búsqueda

y su influencia en el desempeño de los algoritmos de búsqueda local. Las principales características que se consideran en esos análisis son la relación costo-distancia (Boese *et al.*, 1994; Reeves, 1999; Jones y Forrest, 1995; Manderick *et al.*, 1991; Mattfeld *et al.*, 1999; Weinberger, 1990) y la conectividad del grafo (Bierwirth *et al.*, 2004; Carson, 2001).

A continuación se analizarán las implicaciones de las tres medidas de dificultad mencionadas en la introducción (CL, BV y LE), las condiciones bajo las cuales son válidas, y, lo que en general puede inferirse a partir de ellas.

II.2 Longitud de Correlación (CL)

A partir de la introducción de la función de auto-correlación por Weinberger (1990) se han realizado diversos intentos por utilizar esa medida, o alguna variante de ella, para predecir la dificultad de los casos de un problema al utilizar algoritmos de búsqueda local. La mayoría de estos trabajos se apoyan en uno de los principales aportes del trabajo de Weiberger que es el establecer que las caminatas aleatorias son apropiadas para estudiar los grafos de búsqueda.

Apoyados en ese resultado, en su estudio del TSP tanto simétrico como asimétrico, Stadler y Schnabl (1992) establecen como conjetura que la dificultad se decrementa a medida que la correlación del grafo de búsqueda aumenta, por lo que el proceso de optimización necesita de un mayor número de pasos si la longitud de correlación es pequeña. Por otro lado, Stadler (1995) propone que al realizar la medición de la longitud de correlación se le debe escalar con respecto al tamaño del problema, el diámetro del grafo de búsqueda, o la distancia promedio entre dos soluciones seleccionadas aleatoriamente.

Otra contribución importante es el trabajo de Angel y Zissimopoulos (1998), en el cual, después de analizar el problema de Bipartición de Grafos, afirman que para utilizar la función de correlación como medida de dificultad se debe considerar no solamente el diámetro del grafo y el tamaño del problema (Stadler, 1995), sino también el tamaño del

vecindario. Sin embargo, su mayor contribución fue el cálculo riguroso del coeficiente de correlación para ese problema.

En otro trabajo relacionado, Mattfeld *et al.* (1999) analizan tres medidas, siendo una de ellas la longitud de correlación para los grafos de búsqueda del problema *Job Shop Scheduling*. Proponen valores específicos para los parámetros utilizados para estimar la longitud de correlación utilizando la idea de Weinberger (1990) sobre las caminatas aleatorias. La conclusión a la que llegan es que los casos dificíles presentan mayores longitudes de correlación, utilizando estrategias de búsqueda adaptables, resultado contrario al obtenido cuando se utilizan búsquedas basadas en vecindarios.

En el caso del trabajo presentado por Merz y Freisleben (2000) se propone considerar tres aspectos: interacción entre los elementos que conforman la representación de las soluciones, llamada Epistasis (Kauffman, 1993), longitud de correlación, y el coeficiente de correlación aptitud-distancia (Jones y Forrest, 1995) para caracterizar la dificultad de los casos del problema de asignación cuadrática (QAP por sus siglas en inglés) al intentarse resolver con heurísticas combinadas. Merz concluye que una epistasis alta combinada con una longitud de correlación baja y óptimos locales no correlacionados son características de los casos más difíciles, mientras que valores de epistasis bajos, altas longitudes de correlación y óptimos locales correlacionados se encuentran en los casos más sencillos.

Por su parte, Angel y Zissimopoulos (2001) introducen una variante de la función de autocorrelación originalmente propuesta por Weinberger (1990), y la calculan para un conjunto de problemas combinatorios utilizando vecindarios que cumplen con las suposiciones de simetría y tamaño fijo. Sus resultados muestran que el coeficiente propuesto es una buena medida de dificultad para los casos analizados. Estos mismos autores siguen una estrategia similar al ajustar su coeficiente de correlación para obtener lo que llaman el coeficiente de rugosidad, el cual lo utilizan para estudiar el QAP. Para ello, obtienen analíticamente una cota de ese coeficiente para el vecindario 2-intercambio. Sus resultados implican que al incrementarse el coeficiente de rugosidad se obtienen peores resultados

y utilizan una mayor cantidad de tiempo de ejecución. Siguiendo estos trabajos, Angel y Zissimopoulos (2002) proponen incluir otra medida para determinar la dificultad de los casos. En este caso agregan a los conceptos previamente utilizados con respecto a la rugosidad la medida conocida como dominancia de flujo, la cual se ha utilizado frecuentemente para analizar los casos del QAP (Herroelen y Gils, 1985). La conclusión a la que llegan es que una baja dominancia de flujo, junto con una baja rugosidad, son indicadores de los casos que pueden considerarse sencillos, mientras que altos valores de dominancia y rugosidad se presentan en los casos difíciles.

Un aspecto importante a considerar es que ninguno de los trabajos previamente descritos considera las implicaciones de las medidas mencionadas cuando se emplea la misma cantidad de esfuerzo computacional para los algoritmos de búsqueda local utilizados. En esta sección se retoman los trabajos de Weinberger (1990) y Mattfeld *et al.* (1999) para analizar la longitud de correlación dada por la máxima longitud de caminata aleatoria h^* para la cual la correlación entre las soluciones inicial y final es mayor o igual a 0.5 (Manderick *et al.*, 1991).

El trabajo de Weinberger (1990) supone que los grafos de búsqueda son estadísticamente isotrópicos. Esto es, si se considera la secuencia de soluciones obtenida al realizar una caminata aleatoria, a partir de una solución seleccionada aleatoriamente, un grafo de búsqueda estadísticamente isotrópico es aquel para el cual las estadísticas de esa secuencia de soluciones son las mismas, independientemente del punto de partida (Weinberger, 1990). Retomando ese trabajo, Mattfeld *et al.* (1999) analizan los grafos de búsqueda midiendo la correlación a partir de una caminata aleatoria de tamaño l, definiendo el coeficiente de correlación para un intervalo de longitud $h < l$ de la siguiente forma:

$$\rho(h) = \frac{\frac{1}{l-h}\sum_{t=1}^{l-h}(f_t - \overline{f})(f_{t+h} - \overline{f})}{\frac{1}{l}\sum_{t=1}^{l}(f_t - \overline{f})^2} \tag{1}$$

El valor h^* para el cual sigue existiendo correlación se toma como la longitud de correlación del grafo de búsqueda. Si esta longitud no puede determinarse analíticamente

se considera un valor de h^* tal que $\rho(h^*) = 1/2$ (Manderick *et al.*, 1991) como umbral para el cual se puede decir que aun existe correlación.

Con base en lo descrito anteriormente con respecto a la medida CL (Weinberger, 1990; Stadler y Schnabl, 1992; Stadler, 1995; Bierwirth *et al.*, 2004), se puede resumir el principal resultado de la siguiente forma:

Hipótesis 1. Los valores grandes de CL describen espacios de búsqueda rugosos, los cuales favorecen el uso de búsquedas locales (Stadler y Schnabl, 1992; Stadler, 1995; Bierwirth *et al.*, 2004).

Sin embargo, existen algunos aspectos sobre la hipótesis anterior que se necesitan aclarar, siendo uno de ellos la falta de claridad en la existencia de un umbral para el valor de CL. Esto lleva a establecer algunas preguntas acerca de esta medida, en un intento por motivar un estudio más profundo que lleve a un mejor entendimiento de su significado real, y su relación con el desempeño de las búsquedas locales. De tal forma que, se define a n como el tamaño del caso, se pueden plantear las siguientes preguntas.

Los resultados reportados por Weinberger (1990), Stadler (1995), Mattfeld *et al.* (1999) y Bierwirth *et al.* (2004) solamente definen el valor de correlación que se debe obtener después de una caminata aleatoria de CL pasos para decir que las soluciones siguen correlacionadas, concluyendo que a mayor valor de CL se espera un mejor desempeño de las búsquedas locales. Sin embargo, ninguno de ellos define un valor de CL (en términos del tamaño del problema) a partir del cual ese desempeño se ve afectado de forma tal que se pueda utilizar como umbral para separar los casos en términos de su dificultad, por lo que surge la siguiente pregunta:

Pregunta 1.1. ¿Existe un umbral $\alpha(n)$ para el valor de CL tal que permita separar los casos fáciles de los difíciles? En caso de que exista ese valor $\alpha(n)$ de CL que garantice un buen desempeño de las búsquedas locales, ¿es ésta una condición necesaria y suficiente

para garantizar el éxito de las búsquedas locales?

Ese mismo conjunto de trabajos (Weinberger, 1990; Stadler, 1995; Mattfeld *et al.*, 1999; Bierwirth *et al.*, 2004), así como los estudios teóricos sobre CL realizados en (Angel y Zissimopoulos, 2001) y (Angel y Zissimopoulos, 2002) se enfocan en comparar el desempeño de las búsquedas locales sobre conjuntos de casos del problema utilizando el mismo vecindario. Ninguno de esos estudios realiza una comparación del desempeño basada en las diferencias en los valores de CL obtenidos para diferentes vecindarios, lo que lleva a formular la siguiente pregunta:

Pregunta 1.2. Partiendo del supuesto de esfuerzos computacionales similares. ¿Una búsqueda local que utilice vecindarios con valores pequeños de CL, en un caso dado, garantiza un mejor desempeño que otra que produzca valores más grandes de CL, utilizando el mismo número de evaluaciones de la función objetivo?

Pregunta 1.3. ¿Existe un conjunto de casos identificables para los cuales se cumpla la **Hipótesis 1**?

La **Hipóthesis 1** se estudia a través de la medición de la CL, para un conjunto de casos de los problemas de prueba, y la utilización de esos resultados para predecir el desempeño de un par de algoritmos de búsqueda local sobre los casos analizados.

II.3 Estructura de los Óptimos Locales (LOS)

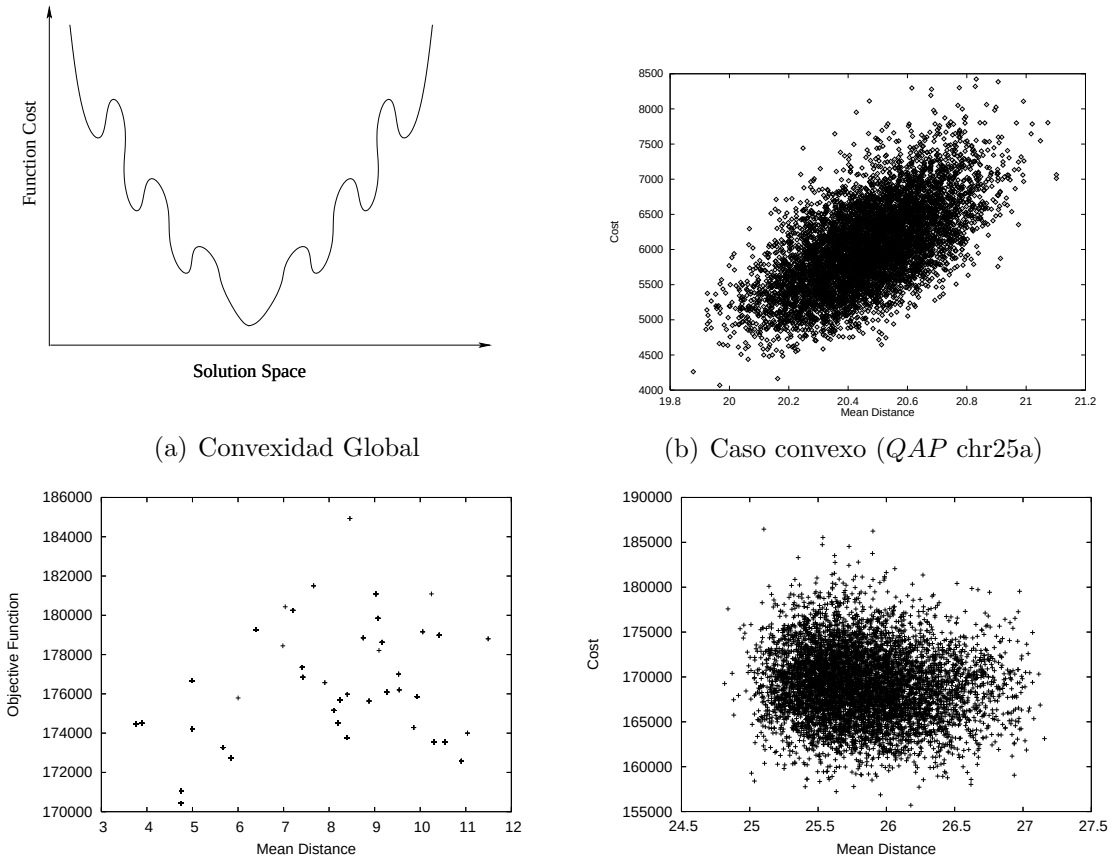
Para poder analizar esta medida se realiza un muestreo del conjunto de óptimos locales, en búsqueda de una propiedad que pudieran aprovechar las estrategias de búsqueda local. Se busca específicamente una estructura de convexidad global conocida como "Big Valley (BV)" (Boese *et al.*, 1994) a través del muestreo del conjunto de óptimos locales y la inspección visual de la gráfica resultante. En esta gráfica se muestra: en el eje x la distancia media entre cada solución y el resto de los óptimos locales, mientras que en el eje y se muestra la calidad de cada solución (*i.e.* valor de la función objetivo). Diversos trabajos

han mostrado casos con este tipo de estructura para problemas como: TSP y Bisección de Grafos ((Boese *et al.*, 1994), utilizando 2500 óptimos locales); "Flowshop" ((Reeves, 1999), 1000 óptimos locales); QAP ((Gutiérrez y Brizuela, 2006), 6400 óptimos locales); entre otros. La Figura 1 b) (Gutiérrez y Brizuela, 2006) muestra como ejemplo un caso del QAP que tiene la estructura BV en su conjunto de óptimos locales. Por otro lado, la Figura 1 a) ilustra un esquema general para el concepto de una función globalmente convexa (Gutiérrez y Brizuela, 2006). La idea es que, como puede observarse en la figura, a medida que uno se aproxima al óptimo global, los óptimos locales se encuentran, en promedio, más cerca a los demás óptimos locales. Finalmente, las figuras 1 c) y d) presentan casos que no muestran esa estructura en su conjunto de óptimos locales.

Considerando los trabajos mencionados previamente, se puede resumir el principal resultado al respecto de la estructura de BV de la siguiente manera (Boese *et al.*, 1994; Reeves, 1999):

Hipótesis 2. La existencia de la estructura de BV en los grafos de búsqueda debería ayudar a mejorar el desempeño de los algoritmos de búsqueda local que explotan esa característica (Boese *et al.*, 1994; Reeves, 1999).

Esta hipótesis ambigua parte de que la estructura de BV supone que las buenas soluciones se encuentran más cercanas entre ellas que a otras soluciones en los grafos de búsqueda (Figura 1), suponiendo que las medidas de distancia utilizadas están directamente relacionadas con las estructuras de vecindario. Por lo tanto, utilizando esta información se puede explotar esa estructura combinando buenas soluciones, *i.e.* realizando pequeños cambios, para obtener una que sea aun mejor. Por ejemplo, considerando que el procedimiento de perturbación en el algoritmo ILS intenta escapar de un óptimo local utilizando un número apropiado de cambios (pasos) (Lourenco *et al.*, 2002), este algoritmo debería ser capaz de aprovechar la existencia de la estructura de BV en los grafos de búsqueda simplemente ajustando su



(a) Convexidad Global

(b) Caso convexo (QAP chr25a)

(c) No convexo (BSL bhaskar51)

(d) No convexo (QAP tho30)

Figura 1. a) Esquema general para el concepto de función globalmente convexa. b) Ejemplo de un caso globalmente convexo. c) y d) Ejemplos de casos globalmente no convexos.

longitud de perturbación.

Sin embargo, existen varios detalles al respecto de la **Hipótesis 2** que no son claros ni se pueden inferir a partir de los trabajos en la literatura. Por lo tanto, se pueden establecer las siguientes preguntas referentes a esta medida y su relación con el desempeño de las búsquedas locales.

La principal pregunta es el cómo medir de forma precisa la existencia de la estructura BV, debido a que el simple análisis visual utilizado por Boese *et al.* (1994) y Reeves (1999) puede prestarse a inexactitudes. Y si bien se han hecho algunas propuestas para realizar

esa medición de forma cuantitativa, como la presentada por Gutiérrez y Brizuela (2006), la diferencia entre lo que es una estructura de BV y lo que no es, permanece incierta, por lo que surge la siguiente pregunta.

Pregunta 2.1. ¿Existe una medida para verificar la existencia de la estructura de BV, así como un umbral que permita separar los casos que poseen esa estructura de los que no la poseen? Esto es, ¿se puede cuantificar el grado de convexidad global?

Boese *et al.* (1994) y Gutiérrez y Brizuela (2006) identifican la estructura de BV sobre casos del TSP con el vecindario Two-Change y su utilidad como guía para la utilización de ciertas estrategias de búsquedas locales. Por otro lado, Reeves (1999) analiza seis vecindarios y cuatro medidas de distancia diferentes en búsqueda de la estructura de BV para motivar la utilización de las estrategias que exploten esa característica. Es decir, ninguno de esos trabajos utiliza la existencia de la estructura de BV como eje en una comparación del desempeño de las estrategias de búsqueda local sobre más de una estructura de vecindario, lo cual lleva a formular la siguiente pregunta.

Pregunta 2.2. ¿Un vecindario con la estructura de BV garantiza un mejor desempeño de una búsqueda local que lo utilice que otra búsqueda local que utilice un vecindario que no posea esa estructura?

Pregunta 2.3. ¿Una búsqueda local que explote la estructura de BV se desempeña mejor sobre un vecindario que posee tal estructura que sobre uno que no la posee?

Pregunta 2.4. ¿La estructura de BV es una condición necesaria y suficiente para garantizar el éxito de una búsqueda local? ¿Lo es siempre?

La **Hipóthesis 2** requiere para su estudio del muestreo de los óptimos locales, para un conjunto de casos de los problemas de prueba, y la utilización de la estructura encontrada como medida de dificultad que permita predecir el desempeño de un par de algoritmos de

búsqueda local sobre los casos analizados.

II.4 Expansión local (LE)

Aldous y Vazirani (1994) propusieron un algoritmo llamado "Go With the Winners" (GWW), el cual encuentra el óptimo global con alta probabilidad para grafos de búsqueda con una estructura de árbol. A partir de este trabajo, Dimitriou y Impagliazzo (1996) extendieron estos resultados para una clase más general de grafos de búsqueda, aquellos que poseen la propiedad denominada "expansión local" (LE). Finalmente, Dimitriou y Spirakis (2004) afirman que si el algoritmo GWW puede descubrir buenas soluciones utilizando solo unas pocas partículas (soluciones) y caminatas lo suficientemente largas, entonces ese espacio (grafo de búsqueda) es un candidato para una buena búsqueda heurística, *i.e.* existen buenos algoritmos para los grafos de búsqueda de ese problema. En vista de la relevancia de la propiedad de LE, a continuación se da su definición formal.

Un grafo $G = (V, E)$, donde V es el conjunto de vértices y E es el conjunto de aristas, tiene una expansión local de x para un polinomio $p(n)$ si para cada subconjunto $Q \subseteq S$ de al menos $|V|/p(n)$ vértices, al menos $x|Q|$ vecinos de esos vértices no están en Q, esto es, $|\mathbf{N}(\mathbf{N}(Q)) - Q| \geq x|Q|$ (Carson, 2001). Donde $x = min\{\frac{|\mathbf{N}(Q)|}{|Q|}\}$ para todo Q, tal que $|Q| \leq \frac{|S|}{2}$.

Dado esto, existen diferentes formas de verificar indirectamente si un grafo de búsqueda posee la propiedad de expansión local. Dimitriou y Impagliazzo (1996) encontraron dos condiciones suficientes para que un grafo de búsqueda posea esa propiedad. La primera de ellas es que el grafo de búsqueda esté compuesto por un número polinomial de clusters desconectados[1], y cada cluster, al ser considerado como un grafo independiente, es un expansor (Bollobas, 1985) que cumple con lo siguiente: suponga que se tiene un grafo bipartito consistente en dos capas de nodos **A** y **B**, siendo **A** la capa superior. Sea $\mathbf{S} \subseteq \mathbf{B}$,

[1]En este caso, cada cluster sería un componente conectado del grafo.

se espera que se cumpla la desigualdad $|N(N(\mathbf{S}))-\mathbf{S}| \geq v|\mathbf{S}|$ donde $|\mathbf{S}| < |\mathbf{B}|/2$ (Dimitriou y Impagliazzo, 1996; Carson, 2001). La segunda condición es que el grafo esté compuesto por un número polinomial de cluster mayormente desconectados. En este caso, la verificación se enfoca a encontrar una medida de decorrelación en la camina aleatoria (Carson, 2001).

Carson (2001) propone una serie de pruebas experimentales para verificar si un grafo de búsqueda tiene la propiedad de LE. La mayoría de esas pruebas requieren de un gran conocimiento previo sobre la distribución de los datos de entrada, siendo la prueba de decorrelación la que presenta menos limitaciones a ese respecto. En esta prueba se busca verificar la segunda condición suficiente para probar la existencia de la propiedad de LE (Carson, 2001). Tal prueba se puede describir como sigue: se seleccionan dos soluciones aleatorias (vértices del grafo de búsqueda) v y u, y se realiza una caminata aleatoria de longitud L partiendo de v, llamando a la solución final v' y comparando las distancias $d(v', v)$ y $d(v', u)$. De tal forma que, se dice que un grafo tiene la propiedad de LE si encontramos un valor de L que decorrelacione las soluciones, esto es, que al terminar la caminata las distancias $d(v', v)$ y $d(v', u)$ tengan la misma probabilidad de ser la más corta.

La existencia de la propiedad de LE indica que el grafo de búsqueda está bien conectado, lo que significa que es posible encontrar una ruta a las mejores soluciones a partir de cualquier región del mismo (Aldous y Vazirani, 1994; Dimitriou y Impagliazzo, 1996; Carson, 2001). Por lo tanto, considerando los resultados previos con respecto a la LE, se puede establecer la siguiente hipótesis:

Hipótesis 3. El éxito de los algoritmos de búsqueda local sobre los problemas de optimización combinatoria se puede explicar por la existencia de la propiedad de LE en sus grafos de búsqueda.

Sin embargo, la **Hipótesis 3** lleva a formular preguntas más detalladas acerca de la propiedad de LE y su relación con el desempeño de los algoritmos de búsqueda local.

Los resultados reportados por Carson (2001) solamente definen la forma de obtener el

valor de la longitud de la caminata aleatoria (L) que es suficiente para de-correlacionar las soluciones inicial y final, concluyendo que a la simple existencia de un valor finito de L, se espera un buen desempeño de las búsquedas locales. Sin embargo, no se define un valor de CL (en términos del tamaño del problema) a partir del cual ese desempeño se ve afectado de forma tal que se pueda utilizar como umbral para separar los casos en términos de su dificultad, por lo que surge la siguiente pregunta:

Pregunta 3.1. ¿Cuál es el umbral $\beta(n)$ para el valor de la longitud de decorrelación (L), considerando el obtenido al utilizar el test para verificar la propiedad de LE, que permita separar los casos fáciles de los difíciles?

En el conjunto de trabajos previos sobre LE (Dimitriou y Impagliazzo, 1996, 1998; Carson, 2001) no se realiza una comparación del desempeño basada en la existencia de la propiedad de LE en diferentes vecindarios, lo que lleva a formular la siguiente pregunta:

Pregunta 3.2. Para un mismo caso: ¿Cómo es el desempeño de un vecindario que genera un grafo de búsqueda con la propiedad de LE comparado con otro que produce un grafo de búsqueda sin esa propiedad? Más aun, ¿Cómo es el desempeño de un vecindario que necesite un valor mayor de L para pasar el test comparado con un vecindario que necesite un valor menor de L?

La existencia de la propiedad de LE se plantea como una propiedad suficiente por Dimitriou y Impagliazzo (1996) y Carson (2001) para el éxito del algoritmo GWW. Sin embargo, en esos trabajos no se garantiza que tal propiedad sea a su vez necesaria, por lo que puede plantearse la siguiente pregunta.

Pregunta 3.3. ¿La existencia de la propiedad LE es una condición necesaria y suficiente para garantizar el éxito de la búsqueda local?

Pregunta 3.4. ¿La **Hipótesis 3** aplica para cualquier búsqueda local o solamente para

las que posean ciertas características especiales? ¿Solamente aplica para búsquedas locales determinísticas? ¿Solamente para estocásticas? ¿Aplica para ambas?

La validez de la **Hipóthesis 3** se analiza realizando las pruebas experimentales para verificar la existencia de la propiedad de LE en los grafos de búsqueda, para un conjunto de casos de los problemas de prueba, y utilizando esos resultados para predecir el desempeño de un par de algoritmos de búsqueda local sobre los casos analizados.

Capítulo III

Problemas y Algoritmos de prueba

Con el objetivo de analizar los grafos de búsqueda y verificar la influencia de sus principales características en el desempeño de los algoritmos de búsqueda local se seleccionaron cuatro problemas pertenecientes a la clase *NP-difícil* y dos algoritmos de búsqueda local que se utilizaran en la fase experimental. Dentro de los problemas a considerar, el primero de ellos es el Problema de Asignación Cuadrática (QAP por sus siglas en inglés), el cual se ha convertido en un problema de referencia para analizar el desempeño de heurísticas y otros métodos (Taillard, 1995; Burkard *et al.*, 1998; Stützle, 1999; Ahuja *et al.*, 2000; Lourenco *et al.*, 2002). Este problema muestra una gran diversidad en las características de sus casos, algunos de ellos muestran la estructura de BV y algunos no, convirtiéndolo en un buen candidato para verificar la efectividad de un conjunto de criterios de dificultad (Merz y Freisleben, 2000; Angel y Zissimopoulos, 2001, 2002). El segundo es el Problema de Localización de Radio Bases (BSL por sus siglas en inglés), proveniente del diseño de redes inalámbricas (Calégari *et al.*, 1997; Krishnamachari y Wicker, 2000; Brizuela y Gutiérrez, 2003), el cual muestra diferentes estructuras en dos vecindarios estudiados por Brizuela y Gutiérrez (2003). Se considera también el Problema del Agente Viajero (TSP por sus siglas en inglés), el cual se puede utilizar para modelar un gran número de problemas del mundo real (Gutin y Punnen, 2002; Johnson y McGeoch, 1997) y ha sido uno de los problemas combinatorios más estudiados (Gutin y Punnen, 2002). Finalmente, se seleccionó el problema de calendarización de flujo de trabajos (FSP por sus siglas en inglés) (Tamaki y Nishino, 1998; Bagchi, 1999; Brizuela y Aceves, 2003). En cuanto a los algoritmos, se seleccionaron dos de los algoritmos de búsqueda local más simples

para el análisis del desempeño. El primero de ellos es la búsqueda local voraz simple (Papadimitriou y Steiglitz, 1982), la cual es a su vez la base para muchos de los algoritmos de búsqueda local más complejos (Aarts y Lenstra, 1997; Hoos, 1998). El otro algoritmo es el conocido como búsqueda local iterativa (ILS por sus siglas en inglés), el cual posee diversos mecanismos que permiten sintonizar sus componentes para buscar el balance adecuado entre intensificación y diversificación de la búsqueda, lo que le permite explotar la estructura de BV y lo ha llevado a ser uno de los algoritmos más eficaces en atacar problemas de la clase *NP-difícil* (Lourenco *et al.*, 2002; Gutin y Punnen, 2002). La definición de cada uno de los problemas, los conjuntos de casos a considerar, vecindarios y medidas de distancia se describen a continuación, así como los algoritmos y sus componentes principales.

Contribución. En este capítulo se establecen los conjuntos de problemas y algoritmos de prueba para el análisis experimental de las medidas de dificultad.

III.1 Problemas de prueba

III.1.1 Problema de Asignación Cuadrática

El Problema de Asignación Cuadrática consiste en asignar un conjunto de elementos a un conjunto de ubicaciones, dadas las distancias entre las ubicaciones y los flujos entre los elementos. El objetivo es encontrar una asignación de elementos en las ubicaciones tal que la suma del producto entre flujos y distancias sea mínimo. Muchos problemas prácticos se pueden formular como el QAP (Taillard, 1995; Burkard *et al.*, 1998; Stützle, 1999), el cual se define de la siguiente forma:

Problema de Asignación Cuadrática. Dados n elementos, n ubicaciones, y dos matrices D y E de $n \times xn$ para las distancias y flujos, respectivamente, el QAP se define como sigue:

$$\begin{array}{l} min \\ \phi \in \Phi \end{array} \sum_{i=1}^{n} \sum_{j=1}^{n} e_{ij} d_{\phi_i \phi_j} \tag{2}$$

donde Φ es el conjunto de todas las permutaciones de n números, ϕ_i representa la ubicación del elemento i y $e_{ij} d_{\phi_i \phi_j}$ es el costo de asignar el elemento i a la ubicación ϕ_i y el elemento j a la ubicación ϕ_j. El QAP pertenece a la clase *NP-difícil* (Garey y Johnson, 1979) y no se puede aproximar por un factor constante a menos que $P = NP$, incluso en el caso de que las distancias entre las ubicaciones satisfagan la desigualdad del triángulo (Ausiello *et al.*, 1999).

Para la versión multi-objetivo de este problema se considera la variante en la cual se tiene una matriz de distancias D y dos o más matrices de flujo E (Knowles y Corne, 2003). Esta versión se puede describir como sigue:

$$minimizar \;\; C(\pi) = \{C^1(\pi), C^2(\pi), \cdots, C^m(\pi)\} \tag{3}$$

donde

$$C^k(\pi) = \sum_{i=1}^{N} \sum_{j=1}^{N} e_{ij}^k d_{\pi_i \pi_j}, k \in \{1, \cdots, m\}, \tag{4}$$

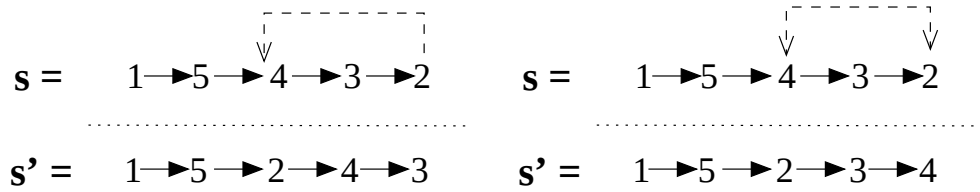
donde e_{ij}^k es el k-ésimo flujo del elemento i al elemento j, y *minimizar* significa obtener el frente Pareto.

III.1.1.1 Vecindarios del QAP

Insert. En este vecindario una solución está dada por una permutación de valores enteros representando la asignación de elementos a ubicaciones (cada ubicación está representada por una posición en la permutación). Un vecindario consiste de todas las soluciones que pueden obtenerse seleccionando un elemento y cambiando su posición en la permutación, recorriendo los elementos entre la nueva posición y la anterior para terminar con una solución factible, en total pueden generarse $(n-1)*(n-1) = n^2-2n+1$ vecinos diferentes de

una solución. Para este vecindario se utiliza como medida de distancia entre dos soluciones $s_1, s_2 \in \mathbf{S}$ el número de veces que se debe aplicar el operador **Insert** para transformar s_1 en s_2 (Stützle, 1999). La figura 2 a) muestra un ejemplo de una solución $s' \in N(s)$ generada a partir del cambio de posición de un elemento (la número 2) en la permutación.

Swap. En este vecindario (también conocido como 2-exchange (Angel y Zissimopoulos, 2002)), al igual que en **Insert**, una solución está dada por una permutación de valores enteros, en donde cada entero representa el elemento que está asignada a la ubicación dada por su posición en la permutación. El vecindario consiste en todas las soluciones que pueden obtenerse al seleccionar dos elementos e intercambiar sus posiciones en la permutación, pueden generar un total de $\binom{n}{2}$ soluciones diferentes. Para este vecindario se utiliza como medida de distancia entre dos soluciones $s_1, s_2 \in \mathbf{S}$ el mínimo número de intercambios **Swap** necesarios para transformar s_1 en s_2 (Stützle, 1999). La figura 2 b) muestra un ejemplo de una solución $s' \in N(s)$ generada a partir del intercambio de posiciones entre dos elementos (2 y 4) en la permutación.



(a) Insert (b) Swap

Figura 2. Ejemplo de los vecindarios Insert y Swap del QAP.

Se toman casos de la biblioteca de casos del QAP más utilizada (QAPLIB[1]) para realizar el análisis de sus grafos de búsqueda. Mientras que, para analizar la versión multi-objetivo de este problema se utilizan casos generados a partir del trabajo de Knowles y Corne (2003), en el cual se tiene una matriz de distancias D y dos o más matrices de flujo E.

[1]http://www.seas.upenn.edu/qaplib/

III.1.2 Localización de Radio Bases

El problema de Localización de Radio Bases (BSL por sus siglas en inglés) en las redes de comunicación celular, consiste en un conjunto de nodos de demanda $\mathbf{V}$ y un conjunto de ubicaciones candidatas de radio bases $\mathbf{B} \subseteq \mathbf{V}$. El problema radica en decidir en dónde ubicar las radio bases, tal que el máximo número de nodos de demanda sea cubierto, lo cual está sujeto a ciertas restricciones (Calégari *et al.*, 1997; Krishnamachari y Wicker, 2000; Mathar y Niessen, 2000). En aplicaciones del mundo real se necesita saber el mínimo número de radio bases que se necesitan para proveer una cobertura dada. Este problema es una variante del problema conocido como el Mínimo Conjunto Dominante (MDS por sus siglas en inglés) Mathar y Niessen (2000):

Problema del Conjunto Mínimo Dominante (Garey y Johnson, 1979). Dado un grafo no dirigido $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, donde $\mathbf{V}$ es el conjunto de vértices y $\mathbf{E}$ el conjunto de aristas, a un subconjunto $\mathbf{V}' \subseteq \mathbf{V}$ se le conoce como dominante si para cada vértice $v \in \mathbf{V} - \mathbf{V}'$ existe una arista que conecta a un vértice $u \in \mathbf{V}'$, tal que $(u, v) \in \mathbf{E}$. El problema del MDS consiste en encontrar el $\mathbf{V}$' de cardinalidad mínima ($|\mathbf{V}'^*|$).

La variante del BSL se puede definir de la siguiente manera: dado el grafo $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, donde $\mathbf{V}$ es el conjunto de nodos de demanda y cada arista $(v_i, v_j) \in \mathbf{E}$ representa que, de ser ubicada una radio base en v_i, ésta cubriría el nodo de demanda v_j; dado también un número q (el nivel de cobertura deseado), y el conjunto de ubicaciones candidatas $\mathbf{B} \subseteq \mathbf{V}$, encontrar el MDS $\mathbf{V}'$ de un subconjunto $\mathbf{C} \subseteq \mathbf{V}$ (*i.e.* $\mathbf{V}' \subseteq \mathbf{C}$) tal que $q = |\mathbf{C}|$ y el MDS resultante $\mathbf{V}' \subseteq \mathbf{B}$. Si la cardinalidad de $\mathbf{V}$ es n, entonces se necesitan resolver $\binom{n}{q}$ problemas MDS restringidos en el peor caso. El MDS pertenece a la clase *NP-difícil* (Garey y Johnson (1979), página 190).

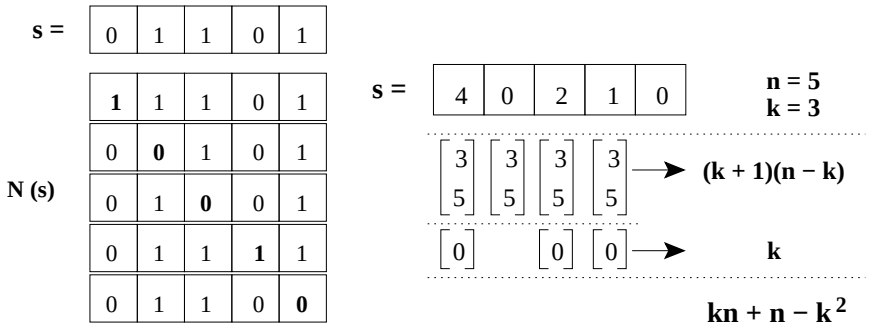
III.1.2.1 Vecindarios del BSL

Binario. En este vecindario cada solución se representa por un vector de N dígitos binarios, donde N es el número de ubicaciones candidatas de radio bases ($|\mathbf{B}|$). Un valor de 1 en la

posición i del vector significa que la ubicación candidata i está seleccionada, mientras que un valor de 0 signifca que no lo está. El vecindario consiste en todas las soluciones que se pueden obtener cambiando el valor de uno de los dígitos de la solución original, lo que da un total de n vecinos diferentes. Para este vecindario se utiliza como medida de distancia entre dos soluciones $s_1, s_2 \in \mathbf{S}$ el número de cambios necesarios para transformar s_1 en s_2, i.e. la "Distancia de Hamming". La figura 3 a) muestra un ejemplo de vecindario binario, $N(s)$, generado a partir de una solución s.

Entero. Para este vecindario, las soluciones están representadas por un vector de N dígitos enteros cuyo valor está entre 0 y N, donde N es el número de ubicaciones candidatas de radio bases ($|\mathbf{B}|$). Un valor de i en el vector (sin importar su posición) significa que la ubicación i está seleccionada para instalar ahí una radio base, el número de valores 0 en el vector representa el número de ubicaciones candidatas que no serán utilizadas. Los valores no pueden repetirse en el vector, excepto por el valor 0. Este vecindario permite tres operaciones diferentes para generar un vecino: borrar una ubicación candidata que se encuentre seleccionada, agregar una de las que no están seleccionadas, intercambiar una de las que se encuentran seleccionadas por una de las que no lo están (Brizuela y Gutiérrez, 2003). Estas operaciones generan K, $N-K$, y $K(N-K)$ vecinos diferentes, respectivamente, siendo K el número de ubicaciones candidatas seleccionadas en la solución original. Por lo tanto, un total de $(K+1)(N-K)+K = N(K+1)-K^2$ vecinos se pueden generar por este vecindario. Para este vecindario se utiliza como medida de distancia entre dos soluciones $s_1, s_2 \in \mathbf{S}$ el número de operaciones permitidas en el vecindario que se necesitan aplicar para tranformar s_1 en s_2. Nótese que el tamaño de este vecindario depende del valor de K. La figura 3 b) muestra todas las soluciones generadas a partir de s utilizando el vecindario entero $N(s)$.

Se utilizaron dos modelos del problema para generar el conjunto de casos de prueba. El primero de ellos, basado en el trabajo de Calégari (Calégari *et al.*, 1997; Calégari, 1999), es un modelo artificial en el cual se conoce la solución óptima. El segundo está basado en



(a) Binario (b) Entero

Figura 3. Ejemplo de los vecindarios Binario y Entero del BSL.)

el trabajo de Krishnamachari y Wicker (2000) y utiliza un modelo básico de propagación para generar las áreas de cobertura de las ubicaciones candidatas. Para este conjunto de casos no se conoce la solución óptima.

III.1.3 Problema del Agente Viajero

Este problema, conocido como TSP por sus siglas en inglés, encuentra una de sus aplicaciones en el ruteo en redes y ha sido uno de los problemas combinatorios más estudiados (Gutin y Punnen, 2002), y puede definirse de la siguiente forma:

Problema del Agente Viajero (Garey y Johnson, 1979). Dado un conjunto $\mathbf{C}$ de m ciudades, la distancia $d(c_i, c_j) \in Z^+$ para cada par de ciudades $c_i, c_j \in \mathbf{C}$. A una solución de este problema se le conoce como *tour*, el cual es una permutación $\langle c_{\pi(1)}, c_{\pi(2)}, ..., c_{\pi(m)} \rangle$, y su distancia total está dada por:

$$\left(\sum_{i=1}^{m-1} d(c_{\pi(i)}, c_{\pi(i+1)}) \right) + d(c_{\pi(m)}, c_{\pi(1)}), \tag{5}$$

el problema consiste en encontrar el *tour* con la distancia total más corta.

El problema del TSP puede ser simétrico ($STSP$) o asimétrico ($ATSP$). El $STSP$ tiene la característica de que las distancias son las mismas en ambas direcciones, *i.e.*, $d(c_i, c_j) = d(c_j, c_i)$ para cada par de ciudades $c_i, c_j \in \mathbf{C}$. El $ATSP$ por su parte no necesita

tener esa característica, es decir, la distancia $d(c_i, c_j)$ de la ciudad c_i a la ciudad c_j no necesita ser igual a la distancia de regreso $d(c_j, c_i)$.

El TSP pertenece a la clase *NP-difícil* (Garey y Johnson, 1979) y, en general, no se puede aproximar por un factor constante a menos que $P = NP$ (Ausiello *et al.*, 1999). Sin embargo, para casos que satisfacen la desigualdad del triángulo el mejor algoritmo de aproximación en tiempo polinomial que se conoce tiene un factor de 3/2 (Ausiello *et al.*, 1999).

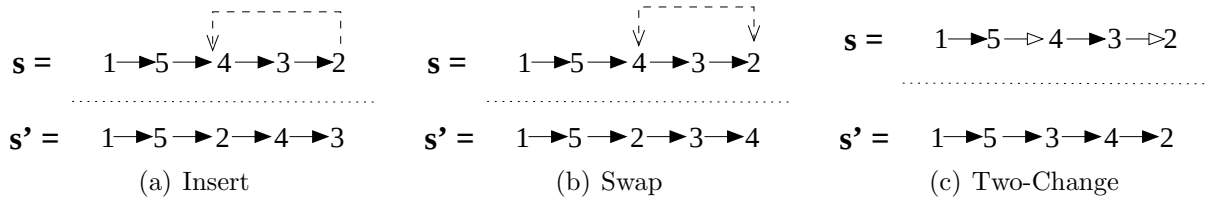
III.1.3.1 Vecindarios del TSP

Insert. En este vecindario un *tour* está representado por una permutación de valores enteros que representan a las ciudades. Un vecindario consiste en todos los *tours* diferentes que se pueden obtener seleccionando cualquier ciudad y cambiando su posición dentro del *tour*. Este movimiento provoca el borrado de tres aristas, reemplazándolas con otras tantas para crear el nuevo *tour*.

Swap. En este vecindario, al igual que en el **Insert**, un *tour* está representado por una permutación de valores enteros que representan a las ciudades. En este caso, un vecindario consiste en todos los *tours* diferentes que se pueden obtener seleccionando dos ciudades e intercambiando sus posiciones en el *tour*. Este movimiento provoca el borrado de tres aristas y su posterior reemplazo para la creación del nuevo *tour*.

Two-Change. Este vecindario consiste en todos los *tours* diferentes que se obtienen al borrar dos aristas y reemplazarlas con otras dos diferentes, de forma tal que se reconecten las dos rutas resultado del borrado inicial. Esta perturbación podría necesitar el cambio de todas las aristas en una de las rutas para obtener un *tour* válido.

Para los tres vecindarios del TSP mencionados anteriormente se utiliza como medida de distancia la llamada "Bond Distance", la cual es igual a n menos el número de aristas comunes entre las dos soluciones Boese *et al.* (1994).



(a) Insert (b) Swap (c) Two-Change

Figura 4. Ejemplo de los vecindarios Insert, Swap y Two-Change del TSP.)

Se utilizan casos de la conocida biblioteca TSPLIB[2] y de algunos generadores para $STSP$ y $ATSP$, los cuales se describen en Gutin y Punnen (2002). Para $STSP$ se utilizan casos de los siguientes generadores: Random Uniform Euclidean Instances (C1c.X), Random Clustered Euclidean Instances (E1c.X) y Random Symmetric Distance Matrices (M1c.X). Para $ATSP$ se usan casos de: Random Symmetric Matrices (smat.X), Random 2-Dimensional Rectilinear Instances (rect.X), Random Asymmetric Matrices (amat.X), Shortest-Path Closure of amat (tmat.X), Random Euclidean Stacker Crane Instances (crane.X) y Disk Drive Instances (disk.X). En el caso de la versión multi-objetivo del TSP se utilizan los casos descritos en Hansen (2000).

III.1.4 El Problema del Flujo de Trabajos

Se considera el problema del flujo de trabajos, conocido como FSP, representado a través de permutaciones, el cual consiste en un conjunto $\mathbf{J}$ de n trabajos que se deben procesar en un conjunto de máquinas $\mathbf{M}$. Cada trabajo $j \in \mathbf{J}$ tiene $m = |\mathbf{M}|$ operaciones. A su vez, cada operación O_{kj}, representando la k-ésima operación del trabajo j, tiene asociado un tiempo de procesamiento t_{kj} y cada máquina debe terminar una operación una vez que la ha iniciado (no interrupciones). Ninguna máquina puede procesar más de una operación al mismo tiempo, y ninguna operación la puede procesar más de una máquina en un mismo momento. Cada trabajo j tiene asignado un tiempo de liberación r_j así como un tiempo deseado de finalización d_j, además de que todos los trabajos deben seguir la misma ruta a

[2]http://www.iwr.uni-heidelberg.de/iwr/comopt/soft/TSPLIB95/TSPLIB.html

través de las máquinas. El objetivo es encontrar la permutación de trabajos que minimice la función objetivo (dado que el orden de las máquinas es irrelevante).

Para entender las funciones objetivo que se desean optimizar en este problema se presenta primero la siguiente notación. Sea O_{kj} el tiempo de inicio de la operación s_{kj}, y c_{kj} su tiempo de finalización. Se define $\mathbf{K}_m$ como el conjunto $\{1, 2, \cdots, m\}$. Dada esta notación, una solución factible cumple con las siguientes condiciones:

$$s_{kj} > r_j \quad \forall k \in \mathbf{K}_m, j \in \mathbf{J} \ , \tag{6}$$

$$s_{kj} + t_{kj} \leq s_{(k+1)j} \forall k \in \mathbf{K}_{m-1}, j \in \mathbf{J} \ . \tag{7}$$

Todos los pares de operaciones O_{kj} y O_{ri} procesados en la misma máquina deben cumplir con lo siguiente:

$$\begin{gathered} s_{kj} + t_{kj} \leq s_{ri} \quad \text{o} \\ s_{ri} + t_{ri} \leq s_{kj} \text{ para cada maquina en } \mathbf{M}, k \neq r \text{ o } j \neq i \quad . \end{gathered} \tag{8}$$

Considerando lo anterior se definen a continuación las funciones objetivo empezando por el tiempo máximo de finalización (conocido en inglés como makespan), el cual representa el tiempo de finalización del último trabajo, *i.e.*

$$f_1 = \max_j \{s_{mj} + t_{mj}\} \ . \tag{9}$$

La segunda función objetivo es el tiempo promedio de flujo, el cual representa el valor promedio del tiempo durante el cual los trabajos permanecen en el taller.

$$f_2 = \bar{fl} = (1/n) \sum_{j=1}^{n} fl_j \ , \tag{10}$$

donde $fl_j = s_{mj} + t_{mj} - r_j$, *i.e.* el tiempo que el trabajo j dura en el taller después de que se libera. Finalmente, el tercer objetivo es la tardanza promedio, *i.e.*

$$f_3 = \bar{T} = (1/n) \sum_{j=1}^{n} T_j \ , \tag{11}$$

donde $T_j = \max\{0, L_j\}$ y $L_j = s_{mj} + t_{mj} - d_j$. Por lo que se tiene el siguiente problema multi-objetivo:

$$\begin{aligned} &\text{optimizar } (f_1, f_2, f_3) \\ &\text{sujeto a } (6) - (8) \ . \end{aligned} \tag{12}$$

El vecindario utilizado para este problema es el Insert (Brizuela y Aceves, 2003), el cual es equivalente al presentado para el QAP (figura 2 (a)) y el TSP (figura 4 (a)). La medida de distancia entre dos soluciones equivale al número mínimo de veces que debe aplicarse este operador a una solución para convertirla en la otra, dando un total de $(n-1)(n-1) = n^2 - 2n + 1$ soluciones vecinas diferentes.

III.2 Algoritmos de Búsqueda Local

Se utilizarán dos algoritmos de búsqueda local para estudiar la influencia de las medidas de dificultad en la predicción de su desempeño sobre los problemas de prueba. La selección de estos algoritmos se debe a su sencillez y el éxito que han mostrado al aplicarse para atacar problemas *NP-difíciles* (Aarts y Lenstra, 1997; Du y Pardalos, 1999). Aunado a ello, uno de los algoritmos seleccionados, al que se conoce como Búsqueda Local Iterativa (ILS por sus siglas en inglés), posee mecanismos que permiten explotar naturalmente la existencia de una estructura de BV, la cual es una de las características analizadas en los grafos de búsqueda de los problemas de prueba. El algoritmo ILS es capaz de escapar de un óptimo local utilizando un mecanismo de perturbaciones aleatorias (Lourenco *et al.*,

2002). Además, este fenómeno de escape está relacionado a la existencia de la propiedad de LE (Carson, 2001), la cual puede ayudar a estimar la longitud de perturbación que debe utilizarse para maximizar la probabilidad de escape. A continuación se presentan la descripción de los dos algoritmos seleccionados.

III.2.1 Búsqueda Local Voraz Simple ($SGLS$)

Este algoritmo consiste simplemente en una caminata sobre la estructura de vecindario, en donde una solución vecina se acepta siempre y cuando sea mejor a la solución actual. Para realizar esa caminata se tiene que, dado un vecindario, existen diversos tipos de movimientos que pueden realizarse (Aarts y Lenstra, 1997; Hoos, 1998), los cuales pueden ser estocásticos o determinísticos. Entre ellos, los tres movimientos básicos son: i) moverse a la primer solución vecina que mejore a la actual, ii) moverse a la mejor solución del conjunto de vecinos de la solución actual, iii) alguna variante estocástica de los dos anteriores. Para este trabajo, se selecciona a la primer solución vecina que mejore a la actual para aceptarla como la nueva solución actual. El pseudo código para el algoritmo de búsqueda local voraz simple se describe a continuación (Papadimitriou y Steiglitz, 1982).

Algoritmo 1. *SGLS*

procedimiento *Búsqueda Local Voraz Simple*

$s =$ **Una_Solución_Inicial()**

Evaluar_Función_Objetivo(s**)**

mientras Mejorar(s**)** $\neq$ 'no' **do**

$s =$ **Mejorar(**s**)**;

fin

En este algoritmo la función **Mejorar(**s**)** explora en el peor caso todo el vecindario de s, $(N(s))$. La función regresa la primera solución $s' \in N(s)$ que mejore a s, en caso de no encontrarse ninguna solución que mejore a s se regresa 'no'. Cada corrida de este algoritmo

termina en un óptimo local.

III.2.2 Búsqueda Local Iterativa (ILS)

El ILS se encuentra entre los algoritmos de búsqueda local que mejor se desempeñan sobre algunos de los problemas de optimización combinatoria más conocidos (Johnson y McGeoch, 1997; Taillard, 1995; Lourenco *et al.*, 2002). También se encuentra entre los algoritmos más sencillos que explotan la propiedad de convexidad global para escapar de los óptimos locales introduciendo una caminata aleatoria hasta que se llegue a la base de atracción de un óptimo local diferente. La esencia del ILS puede resumirse como la construcción iterativa de una secuencia de soluciones generadas a través de una heurística embebida (Lourenco *et al.*, 2002). Sea f la función de costo del problema de optimización combinatoria, donde el objetivo es minimizar f. Sea **S** el conjunto de todas las soluciones factibles s. Finalmente, sea *BúsquedaLocal* el procedimiento de búsqueda local que define el mapeo del conjunto S a un conjunto más pequeño S^* de óptimos locales s^*, el procedimiento *Búsqueda Local* muestrea a S^*. Por lo tanto, dado el mapeo de S a S^*, se necesita reducir el costo sin modificar el procedimiento *BúsquedaLocal*. El ILS intenta evitar las desventajas de reiniciar la búsqueda de forma aleatoria al explorar S^* utilizando una caminata que va de una solución s^* a otra "cercana". Este algoritmo se puede describir con los siguientes pasos de alto nivel (Lourenco *et al.*, 2002):

Algoritmo 2. ILS

procedimiento *Búsqueda Local Iterativa*

$s_0 =$ **GenerarSoluciónInicial()**

$s^* =$ **BúsquedaLocal**(s_0)

repetir

$s' =$ **Perturbar**(s^*, *historial*)

$s^{*'} =$ **BúsquedaLocal**(s')

$s^* =$ **CriterioDeAceptación**(s^*, $s^{*'}$, *historial*)

hasta que la condición de paro se cumpla

fin

Considerando esta arquitectura general, el comportamiento del ILS está definido por las rutinas: **GenerarSoluciónInicial**, **BúsquedaLocal**, **Perturbar** y **CriterioDeAceptación**. A continuación se describen los principales aspectos de cada una de ellas:

GenerarSoluciónInicial. La solución inicial s_0 se puede generar utilizando diversos métodos (aleatorio, voraz, etc.) y solamente tiene influencia sobre cuál óptimo local es el primero en visitarse.

BúsquedaLocal. Esta rutina suele considerarse como una caja negra que permite muestrear el conjunto S^*. Para algunos problemas como en el caso del TSP, el desempeño del ILS depende directamente del desempeño de la búsqueda local que utilice (Lourenco *et al.*, 2002).

Perturbar. La perturbación permite al algoritmo llegar a soluciones que puedan ser el punto de partida hacia un nuevo óptimo local utilizando la rutina de búsqueda local. Por lo tanto, la perturbación debe ser suficientemente grande para "escapar" de la base de atracción del óptimo local actual, pero no debe ser tan grande que genere prácticamente una solución aleatoria. La perturbación puede depender de los óptimos locales explorados previamente, en cuyo caso se tiene una caminata con memoria sobre el conjunto de óptimos locales, en esta versión del ILS ese mecanismo está representado por *historial*. Trabajos previos aseguran que en el caso del TSP una perturbación de tamaño pequeño y fijo (double-bridge) es suficiente, mientras que para el QAP una perturbación relativamente grande parece ser necesaria (Lourenco *et al.*, 2002).

CriterioDeAceptación. Este procedimiento determina si una solución $s^{*'}$ se acepta o no como la nueva solución actual. En esta versión del ILS se muestra solamente un

componente *historial* que representa cualquier tipo de dependencia de óptimos locales explorados previamente para decidir si se acepta o no una nueva solución. Si no existe el componente de *historial*, la búsqueda induce una dinámica en la caminata aleatoria tal que la probabilidad de realizar un movimiento particular de s_1^* a s_2^* depende solamente de s_1^* y s_2^* (Lourenco *et al.*, 2002). Esta rutina y la de **Perturbar** controlan el balance entre intensificación y diversificación en la búsqueda sobre S^*.

La version del ILS utilizada en este trabajo parte de la generación de soluciones iniciales aleatorias, utiliza el algoritmo $SGLS$ como búsqueda local, utiliza una caminata aleatoria con varias longitudes ($N/8$, $N/4$, $N/2$, N, $2N$) como método de perturbación, y un criterio de aceptación que no utiliza historial y permite utilizar cualquier nuevo óptimo local.

Capítulo IV

MOGWW

Muchos de los problemas de la vida real pertenecientes a la clase *NP-difícil* requieren de la optimización de más de un objetivo a la vez, sin embargo, estos problemas se replantean en sus versiones mono-objetivo para utilizar metodologías cuyos desempeños se han estudiado ampliamente en la literatura (Aarts y Lenstra, 1997; Hoos y Stützle, 2005; Coello *et al.*, 2002; Deb, 2001). Los esfuerzos por plantear estrategias que lidien con problemas multi-objetivo se han enfocado mayormente en la extensión de métodos que tratan con casos mono-objetivo. Entre ellas se pueden mencionar como algunas de las más utilizadas a los Algoritmos Genéticos (Coello *et al.*, 2002; Deb, 2001), Recocido Simulado (Serafini, 1994; Thompson, 2001), Colonias de Hormigas (Mariano y Morales, 1999; Gambardella *et al.*, 1999), Enjambre de Partículas (Kennedy y Eberhart, 1995), Estrategias Evolutivas (Binh y Korn, 1996), Búsqueda Tabú (Hansen, 1997; Gandibleux *et al.*, 1997), entre otras.

En este trabajo se propone la extensión de uno de los métodos mono-objetivo que mejores resultados ha obtenido en el modelado de las heurísticas de búsqueda local aleatoria. Este método analiza en particular el modelo de heurísticas en el cual se da preferencia a las mejores soluciones, a ese esquema se le conoce en inglés como "Go with the Winner" (*GWW*) (Aldous y Vazirani, 1994). Siguiendo ese esquema, Aldous y Vazirani (1994) proponen el algoritmo *GWW*, restringido a grafos de búsqueda con estructura de árbol, en un intento de dar una explicación rigurosa de los algoritmos basados en el paradigma de supervivencia del más apto. Dimitriou y Impagliazzo (1996) generalizan el algoritmo *GWW* para analizar grafos de búsqueda generales, mientras que Carson (2001) propone una metodología experimental para verificar la existencia de la propiedad de expansión local en los grafos de búsqueda, necesaria para que el *GWW* pueda trabajar sobre ellos.

En este capítulo se describen los conceptos básicos de la optimización multi-objetivo y se propone una extensión del GWW para atacar problemas con más de un objetivo.

Contribución. Se propone un nuevo algoritmo multi-objetivo basado en el GWW, así como un par de versiones híbridas que intentan subsanar los puntos débiles del algoritmo propuesto.

IV.1 Optimización Multi-Objetivo

Muchos de los problemas combinatorios derivados de aplicaciones reales son multi-objetivo y para tratar con ellos se han propuesto extensiones de las principales heurísticas mono-objetivo (Coello *et al.*, 2002; Deb, 2001). Debido a ello, el análisis del desempeño de los algoritmos multi-objetivo se ha vuelto un objeto de estudio importante junto a la caracterización de la dificultad de los problemas de optimización combinatoria multi-objetivo ($MOCO$ por sus siglas en inglés). Suponiendo, sin pérdida de generalidad, un problema de minimización, los problemas de tipo $MOCO$ se pueden formular de la siguiente forma (Borges y Hansen, 2000):

$$minimizar\{z_1 = f_1(x), ..., z_J = f_J(x)\} \tag{13}$$

donde J es el número de funciones objetivo, una solución $x = [x_1, ..., x_l] \in \mathbf{D}$ es un vector de variables discretas y $\mathbf{D}$ es el conjunto de soluciones factibles. La *imagen* de una solución en el espacio de criterios es un *punto*, $z = [z_1, ..., z_J]$. La mayoría de los problemas del tipo $MOCO$ poseen más de una solución óptima, a ese conjunto se le conoce como soluciones "no-dominadas" o soluciones Pareto óptimas (Coello *et al.*, 2002; Deb, 2001). Si suponemos de nuevo problemas de minimización, la definición de dominancia se puede establecer como sigue (Borges y Hansen, 2000; Zitzler *et al.*, 2003):

Un punto $z^1 \in Z$ domina $z^2 \in Z$, $z^1 \preceq z^2$, si $\forall_j z_j^1 \leq z_j^2$, y existe al menos un j para el cual $z_j^1 < z_j^2$. La solución x^1 domina x^2, $x^1 \preceq x^2$, si la imagen de x^1 domina la imagen de x^2.

Una solución $x \in \mathbf{D}$ es *no-dominada* si no existe una $x' \in \mathbf{D}$ tal que $x' \preceq x$, al conjunto de soluciones que cumplen con esta característica se les conoce como conjunto óptimo global de Pareto. Decimos que una solución $x \in \mathbf{D}$ es óptimo local, en el sentido de Pareto, si no existe una solución vecina a ella $x' \in N(x)$ tal que $x' \preceq x$. De tal forma que, un frente no-dominado es óptimo local de Pareto, con respecto a N, si contiene solamente soluciones que son óptimos locales en el sentido de Pareto con respecto a ese mismo vecindario (Borges y Hansen, 2000; Paquete *et al.*, 2004).

Cuando se comparan dos o más conjuntos de soluciones no-dominadas, se necesita utilizar medidas de desempeño que den información sobre las relaciones de dominancia entre esos conjuntos. Se han propuesto muchas alternativas al respecto, sin embargo, la existencia de una medida única para decidir cual conjunto no-dominado es mejor sigue sin esclarecerse (Zitzler *et al.*, 2003). Con el objetivo de incrementar la confiabilidad de las comparaciones, en este trabajo se utilizan dos de las principales medidas de desempeño, las cuales se describen a continuación:

Cobertura (Zitzler y Thiele, 1999; Zitzler *et al.*, 2003). Sean $\mathbf{A}$ y $\mathbf{B}$ dos conjuntos de vectores de decisión. La función C mapea los pares ordenados de $(\mathbf{A}, \mathbf{B})$ al intervalo $[0, 1]$:

$$C(\mathbf{A}, \mathbf{B}) := \frac{|\{b \in \mathbf{B}; \exists a \in \mathbf{A} : a \preceq b\}|}{|\mathbf{B}|}. \tag{14}$$

Un valor $C(\mathbf{A}, \mathbf{B}) = 1$ significa que todos los puntos en $\mathbf{B}$ son dominados o iguales a los puntos en $\mathbf{A}$, mientras que $C(\mathbf{A}, \mathbf{B}) = 0$ significa que ninguno de los puntos en $\mathbf{B}$ es cubierto por los puntos de $\mathbf{A}$. El conjunto $\mathbf{A}$ domina a $\mathbf{B}$ cuando $C(\mathbf{A}, \mathbf{B}) = 1$ y $C(\mathbf{B}, \mathbf{A}) = 0$. El porcentaje de cobertura entre los conjuntos se define como $\%c(\mathbf{A}, \mathbf{B}) = C(\mathbf{A}, \mathbf{B}) * 100\%$.

Indicador-ϵ binario (Zitzler *et al.*, 2003). Esta medida da un factor que indica cuan peor es un conjunto de vectores de valores objetivo con respecto a otro, considerando todos los objetivos, y se calcula, para problemas bi-objetivo, de la siguiente manera:

$$I_\epsilon(\mathbf{A},\mathbf{B}) := \underset{b\in\mathbf{B}}{max}\ \underset{a\in\mathbf{A}}{min}\ max\left(\frac{a_1}{b_1},\frac{a_2}{b_2}\right). \tag{15}$$

Donde a_i y b_i son los valores de las i-ésimas funciones objetivo de las soluciones $a \in \mathbf{A}$ y $b \in \mathbf{B}$, respectivamente. Si $I_\epsilon(\mathbf{A},\mathbf{B}) > 1$ y $I_\epsilon(\mathbf{B},\mathbf{A}) \leq 1$, entonces el conjunto $\mathbf{B}$ domina completamente al conjunto $\mathbf{A}$. Esta medida indica el mínimo ϵ tal que, para cualquier solución en $\mathbf{B}$ existe al menos una solución en $\mathbf{A}$ que es peor a lo más por un factor ϵ en todos los objetivos. Al realizar comparaciones entre conjuntos se mide también el porcentaje de veces que se encuentre una dominancia completa entre ellos.

IV.2 *GWW* para problemas multi-objetivo

El algoritmo *GWW* lo propuso originalmente Aldous y Vazirani (1994), en un intento de dar una explicación rigurosa del comportamiento de los algoritmos basados en el paradigma de la supervivencia del más apto y que no utilizan operadores de cruzamiento. Este algoritmo tiene claras similitudes con la estrategia de recocido simulado (Kirkpatrick *et al.*, 1983) y está restringido a grafos de búsqueda con estructura de árbol (Aldous y Vazirani, 1994). Dimitriou y Impagliazzo (1996) generalizan el algoritmo para tratar con grafos de búsqueda generales, mientras que Carson (2001) propone una metodología experimental para verificar la existencia de la propiedad de expansión local en los grafos de búsqueda. En esta sección se presenta una extensión de la versión generalizada del *GWW*, misma que busca tratar con problemas de optimización multi-objetivo. A partir de esa extensión, a la cual se le llamará *MOGWW*, se derivan un par de versiones que utilizan búsquedas locales como estrategias de escape, generando dos nuevas versiones (híbridas) del algoritmo *MOGWW*.

IV.2.1 Algoritmo *GWW* Multi-Objetivo (*MOGWW*)

El algoritmo *MOGWW* (Brizuela y Gutiérrez, 2005) se propone como una versión multi-objetivo del *GWW* (Aldous y Vazirani, 1994; Dimitriou y Impagliazzo, 1996), el cual se

diseñó originalmente para grafos de búsqueda con una estructura de árbol y que después generalizaron Dimitriou y Impagliazzo (1996) para utilizarse en grafos de búsqueda que no son árboles. Siguiendo la misma filosofía, se propone una extensión de esa estrategia para diseñar una version multi-objetivo del algoritmo (Brizuela y Gutiérrez, 2005).

Algoritmo 3. *MOGWW*.

Entrada: Un conjunto **P** de soluciones iniciales. Un número L de pasos para la caminata aleatoria.

Salida: Un conjunto **P** de soluciones no-dominadas.

> **Paso 1**. **Inicialización**. Generar $|\mathbf{P}|$ soluciones aleatorias y ubicar una partícula en cada una de las soluciones generadas. Clasificar las soluciones de acuerdo a sus relaciones de dominancia, este paso equivale al procedimiento de ordenamiento de soluciones no-dominadas propuesto en el algoritmo $NSGA-II$ (Deb *et al.*, 2002). Inicializar i como el rango del peor frente no-dominado.
>
> **Paso 2**. Hasta que todas las partículas sean no-dominadas hacer:
>
> **Paso 3**. **Restricción**. $i = i - k$. (decrementar el umbral eliminando los k peores frentes)
>
> **Paso 4**. **Redistribución**. Sea **M** el conjunto de partículas en soluciones que son parte de los frentes no-dominados con rango $l \in \{i+1, .., i+k\}$. Si $|\mathbf{M}| = |\mathbf{P}|$, entonces terminar. En otro caso, borrar las partículas en **M** y hacer $|\mathbf{M}|$ copias de miembros de la población restante, seleccionados aleatoriamente, para reemplazarlos.
>
> **Paso 5**. **Randomización**. Para cada duplicado hecho en el paso anterior, ejecutar la siguiente caminata aleatoria de L pasos con umbral no-dominado: seleccionar un vecino aleatorio, si al vecino no lo domina ninguna solución en el frente con rango i entonces moverse hacia él, en otro caso permanecer en la

solución actual. (Note que seleccionar un vecino, aunque no se mueva hacia él, se cuenta como uno de los L pasos de la caminara aleatoria).

Paso 6. Reclasificar Las soluciones de acuerdo a sus relaciones de dominancia. Inicializar i al rango del peor frente no-dominado. Continuar con la siguiente etapa (**Paso 2**).

Este algoritmo trabaja bajo el mismo principio de su versión mono-objetivo (Dimitriou y Impagliazzo, 1996). Las principales diferencias entre las dos versiones son la introducción de una condición de umbral adecuada y la correspondiente caminata aleatoria no-dominada.

Las adecuaciones a la condición de umbral lidian con el hecho de tratarse de un problema multi-objetivo. En cada iteración se inicia clasificando las soluciones en frentes no-dominados, durante este procedimiento se asigna al peor frente el índice de mayor valor. La versión más simple para el manejo del umbral es seleccionar el peor frente y reemplazar todas las soluciones que se encuentran en él con copias de soluciones que se encuentren en cualquiera de los frentes que poseen un índice menor. Un primer paso para generalizar este procedimiento es el realizar el reemplazo de los k peores frentes con copias de soluciones del resto de los frentes, lo cual se describe en el **Algoritmo 3**. El segundo paso para la generalización es obtener las copias con las que se realiza el reemplazo solamente de j de los mejores frentes, lo cual no es parte del **Algoritmo 3** tal como está descrito.

En el caso de la caminata aleatoria no-dominada, se trata de emular el comportamiento del procedimiento contenido en el algoritmo original GWW, para lo cual se utilizan relaciones de dominancia para condicionar la aceptación del movimiento de las partículas hacia soluciones vecinas. Cada una de las copias realizadas en el **Paso 4** del **Algoritmo 3** representa el punto de inicio de una caminata aleatoria de longitud L, en cada uno de los pasos el vecino seleccionado se compara con las soluciones en el peor frente de los que no fueron borrados. Si al vecino no lo domina ninguna solución de ese frente se mueve la partícula hacia esa solución, en caso contrario la partícula se queda en la solución actual. Con esta comparación se asegura que las nuevas soluciones son al menos

tan buenas como el nuevo peor frente y mejores que cualquiera de los frentes borrados, en términos de relaciones de dominancia. La caminata aleatoria de longitud L es el mecanismo que permite al algoritmo mantener la diversidad en el conjunto de soluciones, lo cual es altamente deseable al tratar con problemas multi-objetivo.

Una de las debilidades que presenta este algoritmo está relacionada con la longitud de la caminata aleatoria (L), la cual, en caso de ser muy grande, puede volver muy costoso al algoritmo en términos de tiempo de cómputo. Sin embargo, la debilidad más importante la representa el hecho de que el algoritmo puede quedar atrapado en frentes no-dominados que no son óptimos locales. Esto se debe, principalemente, a que las comparaciones de no-dominancia se realizan solamente entre partículas, por lo que el conjunto de soluciones en el último frente no-dominado podrían tener mejores soluciones en sus vecindarios que no serán consideradas por el algoritmo. Esto nos lleva a proponer una modificación al algoritmo, de tal forma que se utilice una estrategia de búsqueda local como mecanismo de escape de los frentes no-dominados.

IV.2.2 *MOGWW* Híbrido

El algoritmo *MOGWW* itera hasta que todas las partículas se encuentran en el mismo conjunto no-dominado, sin garantizar que el frente sea un óptimo local en el sentido de Pareto (Brizuela y Gutiérrez, 2005). La idea es agregar una búsqueda local multi-objetivo como mecanismo de escape que permita al algoritmo *MOGWW* continuar la búsqueda. Ese procedimiento permite explorar los vecindarios de las soluciones no-dominadas en búsqueda de otras mejores. Esto puede llevar a una nueva composición del conjunto de soluciones e incluso a la generación de más de un frente no-dominado, esto último permite al algoritmo escapar del frente no-dominado en el que se encuentre estancado y continuar la búsqueda. Esto es, en cada ocasión que el algoritmo quede atrapado en un frente no-dominado, la búsqueda local ayuda a reiniciar el procedimiento principal del *MOGWW*.

El nuevo algoritmo se detiene cuando, después de haber aplicado la búsqueda local multi-objetivo, todas las partículas se encuentren aún en un mismo frente no-dominado. Esto hace que el algoritmo siempre regrese como resultado un frente óptimo local de Pareto. Se propone la utilización de dos estrategias de búsqueda local como mecanismo de escape del algoritmo $MOGWW$. La primera es una versión multi-objetivo de la búsqueda local voraz simple (Papadimitriou y Steiglitz, 1982), mientras que la segunda se basa en la busqueda local pareto (PLS por sus siglas en inglés) propuesta por Paquete y Stützle (2006).

IV.2.2.1 Búsqueda Local Voraz Simple Multi-Objetivo

Tomando en cuenta la búsqueda local voraz simple (Papadimitriou y Steiglitz, 1982), se puede establecer su versión multi-objetivo de la siguiente forma. A partir de una solución inicial se puede buscar una solución vecina que domine a la original y repetir este procedimiento hasta que se encuentre una solución que ninguno de sus vecinos domine. Este procedimiento se describe en el siguiente pseudo-código:

Algoritmo 4. $SGLS$.
Entrada: Una solución inicial s.
Salida: Una solución final s'.

> **Paso 1**. Explorar el vecindario de s en búsqueda de un reemplazo s' que domine a s;
>
> **Paso 2**. Si no existe tal s' entonces $s' = s$ y regresar s' como salida. De lo contrario $s = s'$ y regresar al **Paso 1**.

La estrategia de navegación que se propone seguir para explorar el vecindario es el reemplazar la solución original con la primera solución que cumpliera con la condición para realizar el reemplazo, *i.e.* la primera solución s' que domine a s. Este procedimiento se aplica a todas las partículas cada vez que el $MOGWW$ queda atrapado en un frente no-dominado. Se hará referencia a esta variante del algoritmo como $MOGWW - F$.

IV.2.2.2 Búsqueda Local Pareto

PLS es una extensión de los algoritmos básicos de búsqueda local mono-objetivo propuesta por Paquete *et al.* (2004). Las principales modificaciones para manejar problemas multi-objetivo se relacionan con los criterios de aceptación de nuevas soluciones dentro de la búsqueda local. A continuación se muestra el pseudo-código del algoritmo PLS:

Algoritmo 5. PLS.

Entrada: Una solución inicial s.

Salida: Un conjunto $\mathbf{F}$ de soluciones no-dominadas.

> **Paso 1**. Establecer $\mathbf{F} = \{s\}$ e inicializar la bandera de *visitada* de s a $falso$;
>
> **Paso 2**. Seleccionar aleatoriamente una solución s del conjunto F cuya bandera *visitada* sea igual a $falso$, hacer la selección siguiendo una distribución uniforme y evaluar todos las soluciones vecinas $s' \in N(s)$;
>
> **Paso 3**. Si a una solución s' encontrada en el vecindario de s no la domina ninguna de las soluciones del conjunto $\mathbf{F}$, se agrega a ese conjunto con su bandera *visitada* en $falso$;
>
> **Paso 4**. Una vez que se vsitan todas las soluciones vecinas de s su bandera *visitada* se iguala a *verdadero*;
>
> **Paso 5**. Si existe alguna solución en $\mathbf{F}$ con su bandera *visitada* igual a $falso$, regresar al **Paso 2**, de lo contrario regresar como salida el conjunto $\mathbf{F}$.

Se creó una versión modificada de este algoritmo para utilizarlo en el $MOGWW$. Se utilizó el conjunto de partículas $\mathbf{P}$ como entrada ($\mathbf{F} = \mathbf{P}$), *i.e.* como el conjunto de soluciones iniciales, y una vez finalizado el algoritmo también como conjunto de salida. Una vez que se encuentra un vecino s' de $s \in \mathbf{F}$ tal que sea no-dominado por $\mathbf{F}$, se busca una partícula $t \in \mathbf{F}$ que sea dominada por s' para reemplazarla. Si no existe tal partícula t, el vecino se rechaza. Cada vez que se termina de explorar el vecindario de una

partícula, su bandera *visitada* se iguala a *verdadero*, y cada vez que un nuevo vecino s' reemplaza a una partícula t, su bandera *visitada* se iguala a $falso$. Estas modificaciones permiten al algoritmo mantener un número fijo de soluciones como salida, sin embargo, esta característica puede ocasionar que las soluciones se concentren alrededor de los vecindarios de las primeras partículas exploradas. Se hace referencia a esta variante del algoritmo como $MOGWW - P$.

El **Algoritmo 6** presenta la versión híbrida del $MOGWW$ con la explicación del procedimiento de búsqueda local y su ubicación dentro de los pasos del algoritmo original.

Algoritmo 6. $MOGWW - H$.

Entrada: Un conjunto **P** de soluciones iniciales. Un número L de pasos para la caminata aleatoria.

Salida: Un conjunto **P** de soluciones no-dominadas.

> **Paso 1**. **Inicialización**. Generar $|\mathbf{P}|$ soluciones aleatorias y ubicar una partícula en cada una de las soluciones generadas. Clasificar las soluciones de acuerdo a sus relaciones de dominancia, este paso equivale al procedimiento de ordenamiento de soluciones no-dominadas propuesto en el algoritmo $NSGA-II$ (Deb *et al.*, 2002). Inicializar i como el rango del peor frente no-dominado.
>
> **Paso 2**. Hasta que todas las partículas sean no-dominadas hacer:
>
> **Paso 3**. **Restricción**. $i = i - k$. (decrementar el umbral eliminando los k peores frentes)
>
> **Paso 4**. **Redistribución**. Sea **M** el conjunto de partículas en soluciones que son parte de los frentes no-dominados con rango $l \in \{i+1, .., i+k\}$. Si $|\mathbf{M}| = |\mathbf{P}|$, entonces terminar. En otro caso, borrar las partículas en **M** y hacer $|\mathbf{M}|$ copias de miembros de la población restante, seleccionados aleatoriamente, para reemplazarlos.
>
> **Paso 5**. **Randomización**. Para cada duplicado hecho en el paso anterior,

ejecutar la siguiente caminata aleatoria de L pasos con umbral no-dominado: seleccionar un vecino aleatorio, si al vecino no lo domina ninguna solución en el frente con rango i entonces moverse hacia él, en otro caso permanecer en la solución actual. (Note que seleccionar un vecino, aunque no se mueva hacia él, se cuenta como uno de los L pasos de la caminara aleatoria).

Paso 6. Reclasificar las soluciones de acuerdo a sus relaciones de dominancia.

Paso 7. Si todas las soluciones se encuentran en el mismo frente no dominado, aplicar la búsqueda local a todas las partículas y al final reclasificarlas de nuevo de acuerdo a sus relaciones de dominancia.

Paso 8. Inicializar i al rango del peor frente no-dominado. Continuar con la siguiente etapa (**Paso 2**).

Capítulo V

Análisis de las características de los Grafos de Búsqueda y su influencia en el desempeño de las Búsquedas Locales

En este capítulo se discuten los resultados obtenidos en el análisis de los grafos de búsqueda para los casos de los problemas de prueba (BSL, QAP y TSP), utilizando las tres medidas de dificultad (CL, BV y LE) descritas en el Capítulo II. Además de ello, se analiza el impacto de esas medidas en el desempeño de un par de algoritmos de búsqueda local, como una manera experimental de verificar las hipótesis relacionadas con la capacidad de predicción de las medidas de dificultad sobre los problemas y casos de prueba.

V.1 Análisis de los grafos de búsqueda

V.1.1 Longitud de correlation (CL)

Se realizaron 100 caminatas aleatorias para cada caso utilizando la longitud propuesta por Mattfeld *et al.* (1999) para la caminata aleatoria $l = 100,000$. Se midió la longitud de correlación de los grafos de búsqueda utilizando la siguiente ecuación 1. Donde f_t es la calidad de la solución después de t pasos aleatorios, mientras que $\overline{f}$ es la calidad promedio de las soluciones a lo largo de l pasos de la caminata aleatoria. Considerando lo anterior, se define CL como:

$$CL = h^* = max\left\{h|\rho(h) \geq \frac{1}{2}\right\} \tag{16}$$

Por lo tanto, el costo de calcular CL lo da la longitud de la caminata aleatoria l y h^*.

Se encontró que la longitud de correlación (ecuación (16)) en relación a N, donde N es el tamaño del problema, depende directamente de qué tanto se modifica la calidad de la solución en cada movimiento de vecindario (ver tablas I y II). Esto es, qué tanto cambia la calidad de la solución después de un paso sobre el vecindario, considerando diferentes estructuras de vecindario. Los vecindarios que permiten cambios mayores, en cada paso, presentan valores más pequeños de h^*/N ya que modifican la calidad de la solución más rápidamente (*i.e.* en un menor número de pasos).

Las tablas I, II y II presentan la relación entre los valores promedio de CL obtenidos al utilizar el experimento propuesto por Mattfeld *et al.* (1999) y el tamaño del problema (N), así como sus correspondientes desviaciones estándar. Los resultados muestran que, los casos analizados del BSL tienen aproximadamente el mismo valor promedio de CL para ambos vecindarios (Binario y Entero). En el caso del QAP, el vecindario Insert muestra valores promedio de CL menores que los de Swap. Finalmente, para los casos del TSP se observa que el vecindario Two-Change tiene el mayor valor de CL, seguido del vecindario Insert, mientras que el vecindario Swap presenta el menor valor de los tres.

Siguiendo la **Hipótesis 1** (sección II.2), y considerando los valores de CL obtenidos para BSL, QAP y TSP, se pueden establecer las siguientes hipótesis:

Hipótesis 1.1. Para el problema BSL, ambos vecindarios, Binario y Entero, deben desempeñarse de forma similar.

Hipótesis 1.2. Para el QAP, el vecindario Swap debe mejorar al vecindario Insert en los algoritmos basados en búsqueda local.

Hipótesis 1.3. Para el TSP, los vecindarios Insert y Two-Change deben desempeñarse mejor que el Swap, mientras que el vecindario Two-Change debe mejorar los resultados del

Insert.

Esto es, si los resultados muestran el mismo valor de CL para los vecindarios del BSL y la **Hipótesis 1** se cumple, entonces la **Hipótesis 1.1** debe cumplirse también. En el caso del QAP, los resultados mostraron valores menores de CL para el vecindario Insert, por lo tanto, si la **Hipótesis 1** se cumple, entonces la **Hipótesis 1.2** debe cumplirse también. Finalmente, dado que el vecindario Two-Change obtuvo el mayor valor promedio de CL seguido del Insert y finalmente de Swap, si la **Hipóthesis 1** se cumple, entonces la **Hipótesis 1.3** debe cumplirse también.

V.1.2 Estructura del conjunto de óptimos locales

La relación entre el conjunto de óptimos locales se analiza realizando un muestreo a través de las corridas del **Algoritmo 4** (descrito en la sección III.2.1), el cual utiliza como entrada soluciones aleatorias generadas uniformemente. Para cada combinación de problema/vecindario/caso se realizaron 6400 corridas de una búsqueda local voraz simple, tomándose en cada una de ellas la solución final como salida. El procedimiento general fue una búsqueda local voraz que explora el vecindario hasta encontrar una mejor solución. Si no existe en el vecindario una solución que mejore a la original el algoritmo termina su ejecución y regresa como salida la solución actual, la cual es por definición un óptimo local. El costo de analizar esta medida es igual al costo empleado en resolver el problema, *i.e.* encontrar un número de óptimos locales tal que se pueda visualizar la estructura de ese conjunto.

Para estudiar el conjunto de óptimos locales resultantes, se graficó, para cada uno de ellos, su calidad contra su distancia promedio al resto de las soluciones en el conjunto. A través de un análisis visual se encontró que algunos de los casos presentan la estructura de BV. Esto quiere decir que, las búsquedas locales que exploten esa estructura deben ser capaces de obtener buenos resultados al tratar con esos casos (Boese *et al.*, 1994; Reeves,

1999). Las tablas I, II y III indican si una combinación problema/vecindario/caso muestra o no la estructura de BV en su conjunto de óptimos locales. Se puede notar que, el vecindario Binario del BSL genera grafos de búsqueda con esa estructura en el conjunto de óptimos locales, mientras que el vecindario Entero no. En el caso del QAP, los óptimos locales obtenidos al utilizar el vecindario Insert no mostraron la estructura de BV, y solamente algunos de los casos lo hicieron al utilizar el vecindario Swap. Por otro lado, todos los casos analizados del TSP mostraron la estructura de BV en sus conjuntos de óptimos locales.

Tomando en cuenta los resultados obtenidos en las estructuras de los óptimos locales, para BSL, QAP y TSP, y siguiendo la **Hipótesis 2** (sección II.3), se pueden proponer las siguientes variantes:

Hipótesis 2.1. Para BSL, el vecindario Binario debe mejorar al vecindario Entero al utilizar búsquedas locales que exploten la estructura de BV.

Hipótesis 2.2. Para QAP, el vecindario Swap debe mejorar al vecindario Insert al utilizar búsquedas locales que exploten la estructura de BV.

Hipótesis 2.3. Para TSP, al utilizar búsquedas locales que exploten la estructura de BV, la utilización de cualquiera de los vecindarios analizados (Insert, Swap, Two-Change) debe dar como resultado un desempeño similar.

Esto es, si los resultados mostraron que el vecindario Binario del BSL, el vecindario Swap del QAP y los tres vecindarios del TSP muestran la estructura BV, y se cumple la **Hipótesis 2**, entonces la **Hipótesis 2.1**, **Hipótesis 2.2** e **Hipótesis 2.3** debe cumplirse también.

V.1.3 Expansión Local

Se utilizó una variante del algoritmo GWW (Carson, 2001) para realizar el muestreo de los grafos de búsqueda y verificar si poseían la propiedad de expansión local. Se utilizó una prueba experimental para verificar una condición suficiente para determinar la existencia de la propiedad de expansión local (Carson, 2001). La prueba consiste en el siguiente procecedimiento: seleccionar dos partículas aleatorias (dos vértices en el grafo de búsqueda) v y u. A partir de v realizar una caminata aleatoria de longitud L (parámetro a verificar) y llamar al punto final v'. Comparar las distancias $d(v', v)$ y $d(v', u)$. Para caminatas aleatorias de longitud infinita v' termina de-correlacionada con respecto a v. Por lo tanto, a medida que la caminata aleatoria se aproxima a su comportamiento asintótico, ambas distancias son igualmente probables de ser la más corta. Debido a que en la práctica no pueden utilizarse caminatas aleatorias de longitud infinita, se necesita determinar el valor finito de L que garantice la validez de la sentencia previa. Por lo tanto, para poder afirmar que un grafo posee la propiedad de expansion local se necesita encontrar un valor de L tal que decorrelacione las soluciones.

Para realizar la prueba del GWW utilizando la idea expuesta en el párrafo anterior, se calculó la frecuencia relativa en 1000 experimentos, en cada iteración del algoritmo, como la probabilidad de tener $d(v', v) < d(v', u)$. El valor calculado debe ser aproximadamente 0.5 para indicar que el algoritmo pasó la prueba, de no ser así se debe incrementar el valor del parámetro L. El utilizar esta prueba para verificar la propiedad de LE requiere de un costo computacional igual al necesario para resolver el problema con el algoritmo GWW mas el cálculo de la probabilidad (frecuencia relativa) en cada iteración.

Las tablas I, II y III muestran para cada combinación problema/vecindario/caso si pasó o no la prueba para verificar la existencia de esta propiedad. Se puede observar que el vecindario Binario del BSL, el vecindario Swap del QAP y el vecindario Two-Change del TSP pasaron la prueba utilizando una longitud de decorrelación $L = 4N$, mientras que los vecindarios Insert y Swap lo hicieron utilizando $L = 2N$. Por otro lado, el vecindario Entero

del BSL y el Insert del QAP no pasaron la prueba. De tal manera que, si se consideran los resultados sobre la propiedad de expansión local que se muestran en las tablas I, II y III, y siguiendo la **Hipótesis 3** (sección II.4), se pueden establecer las siguientes hipótesis:

Hipótesis 3.1. Para los casos del problema BSL, el vecindario Binario debe tener un desempeño no peor que el Entero, cuando ambos se usan en algoritmos de búsqueda local.

Hipótesis 3.2. Para los casos del QAP, el vecindario Swap no debe ser peor que el Insert en algoritmos de búsqueda local.

Hipótesis 3.3. Para los casos del TSP, los tres vecindarios utilizados deben mostrar un desempeño equivalente al utilizarse en algoritmos de búsqueda local.

Tomando en cuenta que el vecindario Entero del problema BSL no pasó la prueba y es el único que posee un tamaño variable, surge otra pregunta concerniente a la **Hipótesis 3**.

Pregunta 3.5. ¿La **Hipótesis 3** se cumple para vecindarios de tamaño variable?

Los detalles experimentales de la verificación de la propiedad de expansión local utilizando la prueba de de-correlación se encuentran en el Apéndice A. Estos experimentos nos garantizan que los valores reportados de L para los cuales se cumple que la caminata aleatoria de-correlaciona a la solución final de la inicial son suficientes para probar la existencia de la propiedad de LE en los grafos de búsqueda analizados.

Tabla I. Ejemplos de casos del BSL y sus propiedades: longitud de correlación promedio ($\mu(CL)$) y su desviación estándar $\sigma(CL)$, estructura de óptimos locales (LOS: BV = Big Valley; NBV = No Big Valley), y, expansión local ($LE(L)$).

	Caso	**N**	**Binario**				**Entero**			
			$\mu(CL)/N$	$\sigma(CL)/N$	LOS	$LE(L)$	$\mu(CL)/N$	$\sigma(CL)/N$	LOS	$LE(L)$
1	b0401	32	0.242	0.014	BV	y($4N$)	0.254	0.011	NBV	-
2	b0411	48	0.233	0.008	BV	y($4N$)	0.236	0.010	NBV	-
3	b0501	50	0.253	0.010	BV	y($4N$)	0.253	0.011	NBV	-
4	b0511	75	0.235	0.010	BV	y($4N$)	0.236	0.010	NBV	-
5	b0601	72	0.266	0.011	BV	y($4N$)	0.261	0.011	NBV	-
6	b0611	108	0.243	0.011	BV	y($4N$)	0.244	0.010	NBV	-
7	c0401	32	0.260	0.015	BV	y($4N$)	0.250	0.000	NBV	-
8	c0411	48	0.250	0.006	BV	y($4N$)	0.248	0.008	NBV	-
9	c0501	50	0.280	0.009	BV	y($4N$)	0.271	0.011	NBV	-
10	c0511	75	0.261	0.009	BV	y($4N$)	0.255	0.009	NBV	-
11	c0601	72	0.292	0.012	BV	y($4N$)	0.281	0.011	NBV	-
12	c0611	108	0.259	0.011	BV	y($4N$)	0.249	0.010	NBV	-
			$\cong (1/4)N$			$4N$	$\cong (1/4)N$			

Tabla II. Ejemplos de casos del QAP y sus propiedades: longitud de correlación promedio $\mu(CL)$) y su desviación estándar $\sigma(CL)$, estructura de óptimos locales (LOS: BV = Big Valley; NBV = No Big Valley), y, expansión local ($LE(L)$).

	Caso	**N**	**Insert**				**Swap**			
			$\mu(CL)/N$	$\sigma(CL)/N$	LOS	$LE(L)$	$\mu(CL)/N$	$\sigma(CL)/N$	LOS	$LE(L)$
1	bur26a	26	0.171	0.007	NBV	-	0.273	0.010	NBV	y($4N$)
2	bur26b	26	0.000	0.000	NBV	-	0.501	0.019	NBV	y($4N$)
3	chr25a	25	0.000	0.000	NBV	-	0.100	0.004	BV	y($4N$)
4	esc32a	32	0.000	0.000	NBV	-	0.100	0.003	BV	y($4N$)
5	esc32b	32	0.000	0.000	NBV	-	0.171	0.005	BV	y($4N$)
6	kra30a	30	0.000	0.000	NBV	-	0.327	0.011	BV	y($4N$)
7	lipa60a	60	0.141	0.002	NBV	-	0.465	0.008	NBV	y($4N$)
8	sko49	49	0.409	0.008	NBV	-	0.479	0.010	BV	y($4N$)
9	tai30a	30	0.000	0.000	NBV	-	0.409	0.014	BV	y($4N$)
10	tai30b	30	0.000	0.000	NBV	-	0.368	0.012	NBV	y($4N$)
11	tai60a	60	0.394	0.007	NBV	-	0.359	0.006	NBV	y($4N$)
12	tai60b	60	0.345	0.006	NBV	-	0.414	0.007	BV	y($4N$)
										$4N$

Tabla III. Ejemplos de casos del TSP y sus propiedades: longitud de correlación promedio $\mu(CL)$) y su desviación estándar $\sigma(CL)$, estructura de óptimos locales (LOS: BV = Big Valley; NBV = No Big Valley), y, expansión local ($LE(L)$).

	Caso	**N**	**Insert**				**Swap**				**Two-Change**			
			$\mu(CL)/N$	$\sigma(CL)/N$	LOS	$LE(L)$	$\mu(CL)/N$	$\sigma(CL)/N$	LOS	$LE(L)$	$\mu(CL)/N$	$\sigma(CL)/N$	LOS	$LE(L)$
1	amat.0	100	0.226	0.009	BV	y($2N$)	0.164	0.006	BV	y($2N$)	0.330	0.013	BV	y($4N$)
2	crane.0	100	0.227	0.008	BV	y($2N$)	0.165	0.006	BV	y($2N$)	0.330	0.058	BV	y($4N$)
3	disk.0	100	0.226	0.008	BV	y($2N$)	0.165	0.006	BV	y($2N$)	0.329	0.015	BV	y($4N$)
4	rect.0	100	0.227	0.010	BV	y($2N$)	0.165	0.006	BV	y($2N$)	0.333	0.061	BV	y($4N$)
5	smat.0	100	0.226	0.008	BV	y($2N$)	0.164	0.006	BV	y($2N$)	0.331	0.013	BV	y($4N$)
6	tmat.0	100	0.227	0.008	BV	y($2N$)	0.163	0.006	BV	y($2N$)	0.332	0.015	BV	y($4N$)
7	ft70	70	0.224	0.008	BV	y($2N$)	0.160	0.006	BV	y($2N$)	0.333	0.039	BV	y($4N$)
8	kroA100	100	0.226	0.009	BV	y($2N$)	0.166	0.006	BV	y($2N$)	0.330	0.014	BV	y($4N$)
9	M1c.0	100	0.228	0.009	BV	y($2N$)	0.160	0.006	BV	y($2N$)	0.047	0.007	BV	y($4N$)
10	C1c.0	100	0.226	0.009	BV	y($2N$)	0.166	0.005	BV	y($2N$)	0.327	0.015	BV	y($4N$)
11	E1c.0	100	0.226	0.008	BV	y($2N$)	0.167	0.006	BV	y($2N$)	0.332	0.013	BV	y($4N$)
			$\cong (1/5)N$			($2N$)	$\cong (1/6)N$			($2N$)	$\cong (1/3)N$			($4N$)

V.2 Influencia sobre las búsquedas locales

En esta sección se busca analizar experimentalmente las hipótesis **1.1** a **3.2** para verificar la validez de las hipótesis generales **1, 2,** y **3**. Para ello se propone utilizar dos algoritmos básicos de búsqueda local sobre los casos de los problemas de prueba. Una vez realizados los experimentos, se comparan los resultados con las predicciones hechas utilizando las medidas de dificultad a través de las hipótesis planteadas.

V.2.1 Comparación entre vecindarios

Para este conjunto de experimentos se realizaron, para ambos algoritmos, 100 corridas de cada combinación problema/vecindario/caso. Se utilizaron soluciones iniciales aleatorias, el criterio de parada para $SGLS$ es encontrar un óptimo local, mientras que para ILS es el iterar N veces sin lograr mejorar a la solución que hasta el momento haya sido la mejor, siendo N el tamaño del problema. Debido a que para el BSL el esfuerzo computacional requerido por el vecindario Entero fue mucho mayor que el requerido por el vecindario Binario, se incrementó el número de corridas (*i.e.* soluciones iniciales) para este último en aras de obtener el mismo esfuerzo para $SGLS$ ($SGLS(E)$) e ILS ($ILS(E)$). Para todos estos experimentos se verifican las diferencias entre vecindarios con diferentes grafos de búsqueda cuyas características podrían determinar el desempeño de los algoritmos de búsqueda local. Para ello se define el *gap* entre la calidad promedio de las soluciones del algoritmo A_1 con vecindario N_1 y la calidad promedio de las soluciones del algoritmo A_2 con vecindario N_2 como sigue:

$$gap((A_1, N_1), (A_2, N_2)) = \left(\frac{mean(A_1, N_1)}{mean((A_2, N_2))} - 1 \right) * 100\%, \tag{17}$$

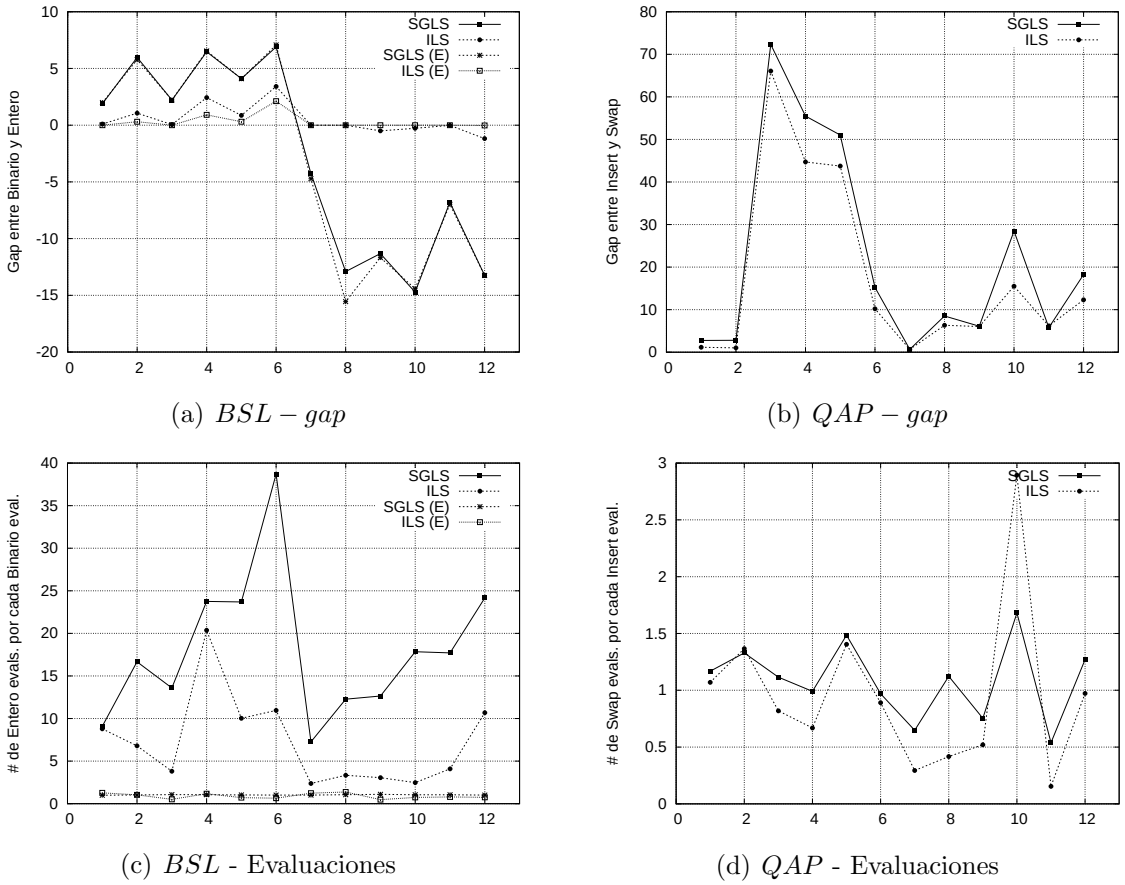
La figura 5 (superior) muestra el *gap* entre la calidad promedio de los vecindarios para casos del BSL y QAP. La figura 5 (inferior) muestra el número de evaluaciones utilizadas en uno de los vecindarios (para el BSL: Entero, para QAP: Swap) por cada evaluación

utilizada en el otro (BSL: Binario, QAP: Insert).

Se puede ver en la figura 5 (superior) que para el BSL, a pesar de que ambos vecindarios mostraron valores similares de CL, el vecindario Entero mejora al vecindario Binario al utilizar el algoritmo $SGLS$, incluso al utilizar el mismo número de evaluaciones ($SGLS(E)$). Ver tabla IV(a) para un análisis estadístico de estos resultados. En el caso del ILS, se puede ver que la diferencia entre los vecindarios es menor, e incluso llega a cero para la mayoría de los casos del $BSL-C$ al utilizar el mismo número de evaluaciones (ILS (E)). Se puede ver en la tabla IV(a) que, para ILS solamente 2 de 12 casos no muestran diferencias estadísicas entre las medias, mientras que para $ILS(E)$ fueron 8 de 12. Considerando los resultados anteriores, la **Hipótesis 1.1** y la **Hipótesis 2.1** no se cumplen para los casos analizados del BSL al utilizar el algoritmo $SGLS$, sin embargo, sí se cumplen para el algoritmo ILS. En el caso de la **Hipótesis 3.1**, tomando en cuenta los mismos resultados, no se cumple para los casos del BSL con ninguno de los algoritmos utilizados ($SGLS$ e ILS). El símbolo "-" en esta tabla implica que las medias no son estadísticamente diferentes.

Para los casos del QAP, el vecindario que mostró el valor más grande de CL (Swap) también obtuvo los mejores resultados promedio para todos los casos en ambos algoritmos $SGLS$ e ILS. Ver tabla IV(b) para un análisis estadístico de estos resultados. Se pueden ver valores mayores de *gap* entre los vecindarios incluso en el caso de que la relación entre el número de evaluaciones es cercano a uno. Aunado a ello, los dos casos (3 y 4) que mostraron las mayores diferencias entre los vecindarios, mostraron también una relación entre el número de evaluaciones cercano a uno para $SGLS$ y menor a uno para ILS. Esto quiere decir que tanto la **Hipótesis 1.2** como la **Hipótesis 2.2** y la **Hipótesis 3.2** se cumplen para los casos analizados del QAP, al utilizar los algoritmos $SGLS$ e ILS.

La parte superior de la figura 6 muestra el *gap* entre la calidad promedio de los vecindarios para casos del TSP, mientras que la parte inferior muestra el número de evaluaciones utilizadas en uno de los vecindarios por cada evaluación utilizada en el otro. Se puede ver que el vecindario Insert supera al Swap para la mayoría de los casos, excepto



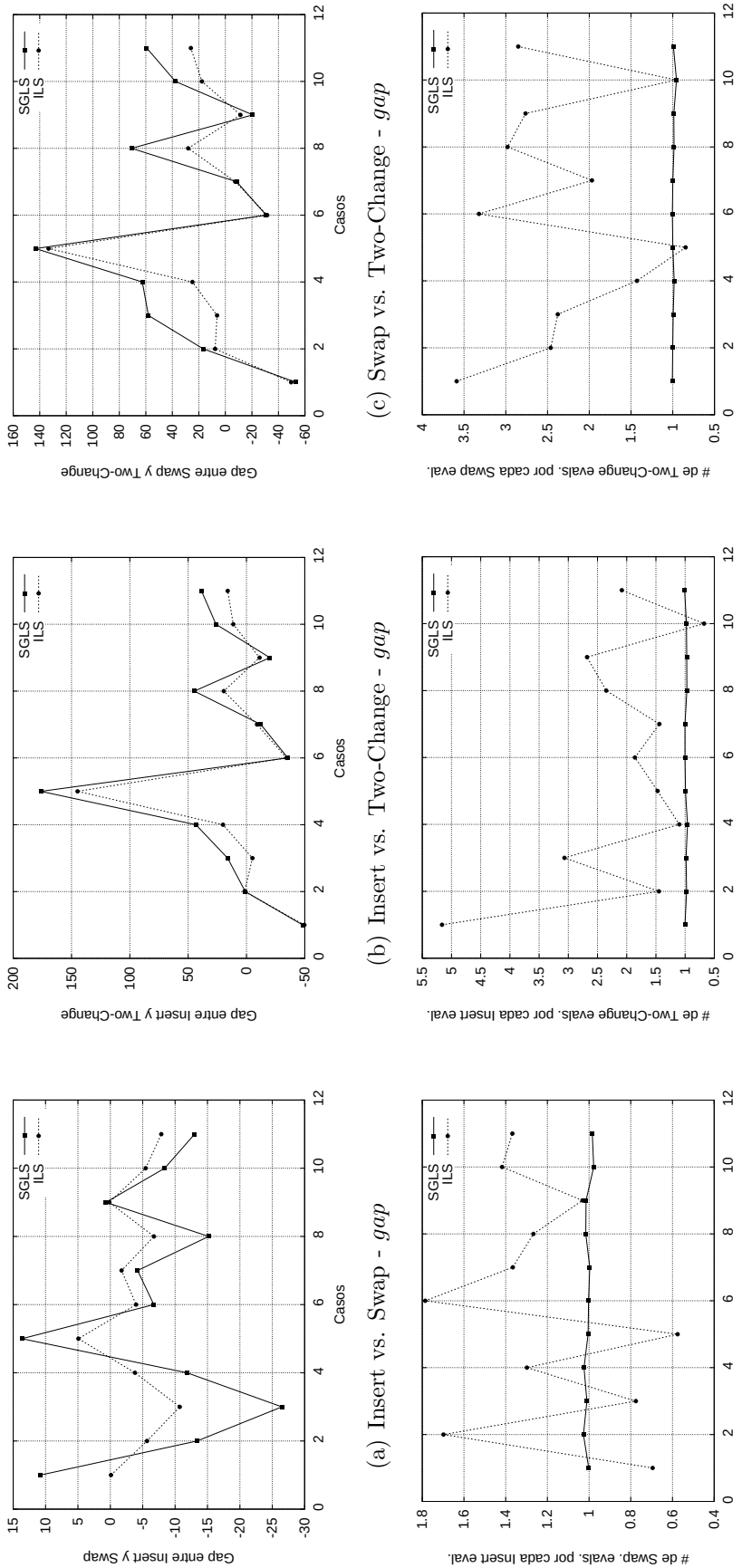
(a) $BSL - gap$ (b) $QAP - gap$

(c) BSL - Evaluaciones (d) QAP - Evaluaciones

Figura 5. Valores de *gap* entre las calidades promedio de los vecindarios y el número de evaluaciones requeridas para los casos analizados del BSL y QAP. ($SGLS$(E) y ILS(E) representan los resultados al utilizar el mismo número de evaluaciones para los vecindarios Binario y Entero del BSL.)

1, 5 y 11, al utilizar tanto $SGLS$ como ILS. Si se considera que estos dos vecindarios mostraron la estructura de BV, así como la propiedad de expansión local utilizando el mismo valor de $L = 2N$, la única medida que los distingue es la CL. Según los resultados de la tabla III, el vecindario Insert, al tener un valor de CL mayor que el vecindario Swap, debiese desempeñarse mejor, lo cual queda confirmado para ocho de los once casos. Por otro lado, al comparar el vecindario Two-Change tanto contra el Insert como contra el Swap, se puede observar que obtiene mejores resultados, excepto en los casos 1,6,7 y 9, con ambos algoritmos $SGLS$ e ILS. Para el resto de los casos, los resultados también concuerdan

con los valores obtenidos de CL. Sin embargo, para las otras dos medidas de dificultad no concuerdan los resultados, ya que se puede observar en la tabla III que los tres vecindarios poseen la estructura de BV y la propiedad de LE, por lo que se espera un desempeño equivalente. Considerando los resultados anteriores, la **Hipótesis 1.3** se cumple, mientras que la **Hipótesis 2.3** y la **Hipótesis 3.3** no se cumplen, para los casos analizados del TSP al utilizar los algoritmos $SGLS$ e ILS.



(a) Insert vs. Swap - *gap*

(b) Insert vs. Two-Change - *gap*

(c) Swap vs. Two-Change - *gap*

(d) Insert vs. Swap - Evaluaciones

(e) Insert vs. Two-Change - Evaluaciones

(f) Swap vs. Two-Change - Evaluaciones

Figura 6. Valores de *gap* entre las calidades promedio de los vecindarios y el número de evaluaciones requeridas para los casos analizados del TSP.)

V.2.2 Comparación entre Algoritmos

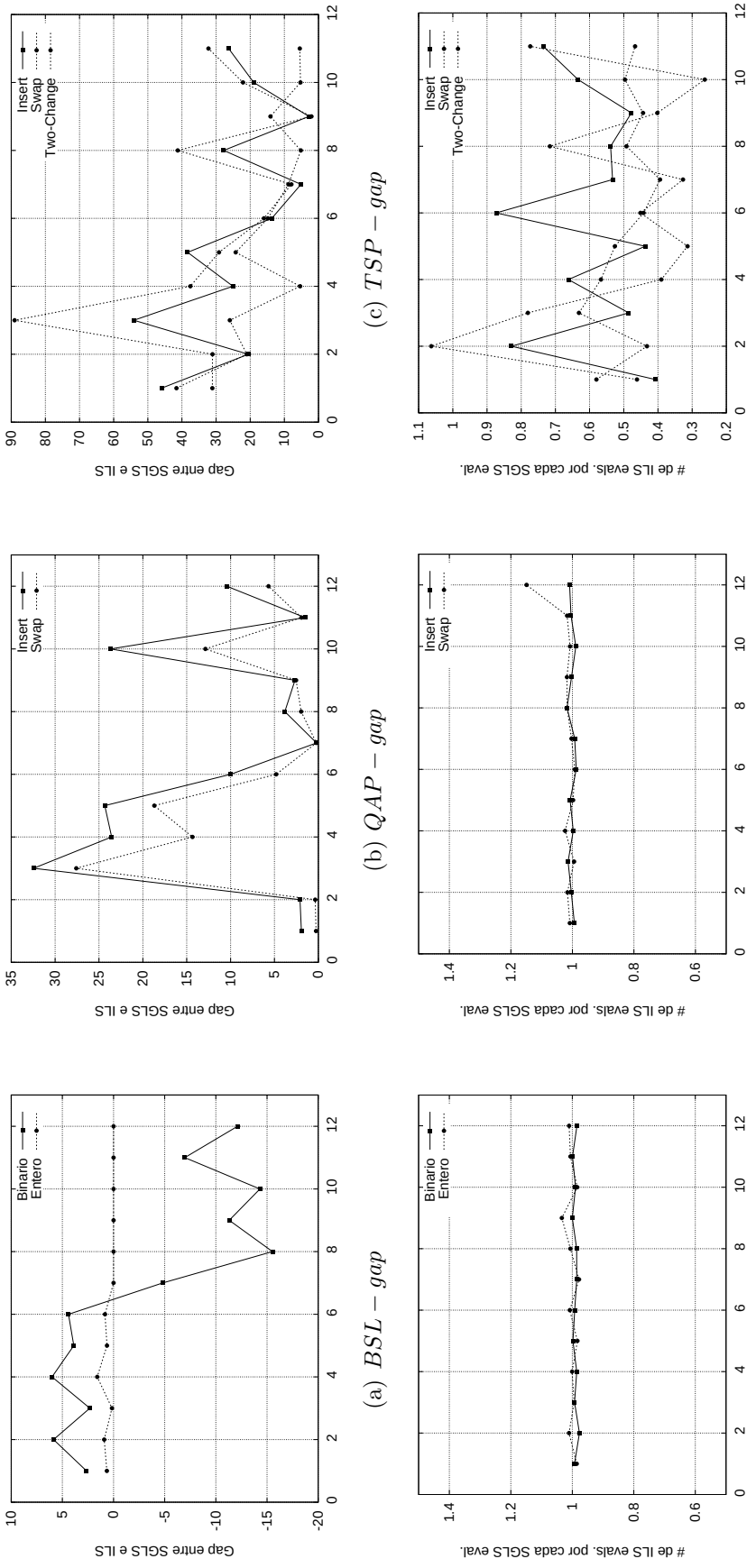
En este conjunto de experimentos se busca comparar las diferencias entre un algoritmo que explota la estructura de BV (el ILS) y otro que no tiene mecanismos para obtener ventajas de esa estructura (el $SGLS$). Se realizaron 100 corridas para cada combinación problema/vecindario/caso en ambos algoritmos, ILS y $SGLS$. A partir de esos resultados y considerando las diferencias en el esfuerzo computacional requerido por cada algoritmo, se estimó el número de corridas adicionales necesarias para que el $SGLS$ alcanzara el mismo esfuerzo computacional usado por el ILS. Se utilizaron soluciones iniciales aleatorias, el criterio de parada para el $SGLS$ fue el encontrar un óptimo local, mientras que para el ILS fue el iterar N veces sin lograr mejorar a la solución que hasta el momento haya sido la mejor, siendo N el tamaño del problema. Se busca determinar las diferencias entre el desempeño de los dos algoritmos utilizados y relacionar esas diferencias con las características de los grafos de búsqueda de los problemas analizados. Para ello se utiliza la definición de *gap* (ecuación 17) entre la calidad promedio de las soluciones del algoritmo A_1 con vecindario N_1 y la calidad promedio de las soluciones del algoritmo A_2 con vecindario N_2.

La figura 7 muestra una comparación en la cual el mismo vecindario se analiza utilizando los algoritmos $SGLS$ e ILS. La figura 7 (superior) muestra el *gap* entre la calidad promedio de los dos algoritmos para los casos analizados del BSL, QAP y TSP, respectivamente. Para los casos del $BSL - B$ (del 1 al 6 en la figura 7-a), un valor positivo de *gap* implica que la calidad media obtenida por el $SGLS$ es peor (mayor) que el valor medio obtenido por ILS en un porcentaje igual al valor de *gap*. Por otro lado, para los casos $BSL - C$ (7 al 12 de la figura 7-a), un valor negativo de *gap* implica que la calidad media obtenida por $SGLS$ es peor (menor) que la obtenida por el ILS en un porcentaje igual al valor de *gap*. La figura 7 (inferior) muestra el número de evaluaciones requeridas en el ILS por cada evaluación requerida en el $SGLS$. Este valor es la relación del número de evaluaciones de la función objetivo en el ILS y el número de evaluaciones requeridas por el $SGLS$. En las tablas V(a) y V(b) se muestra el análisis estadístico de estos resultados.

Se puede observar en la figura 7 a) que el algoritmo ILS obtuvo mejores resultados, en términos de la calidad promedio, para todos los casos del BSL utilizando el vecindario Binario, mejorando al $SGLS$ desde un 2.5% hasta un 15%. Por otro lado, la figura 7 a) muestra que al utilizar el vecindario Entero del BSL se obtienen diferencias menores entre ILS y $SGLS$. La figura 7 d) muestra el número de evaluaciones requeridas en el ILS por cada evaluación requerida en el $SGLS$.

Para los casos del QAP, el algoritmo ILS también mejoró los resultados obtenidos por el $SGLS$, en esta ocasión con porcentajes desde cercanas a 0% hasta diferencias del 32%, en este caso el ILS requirió entre 1 y 1.17 veces el número de evaluaciones requeridas por el $SGLS$. Se puede observar que cinco casos $(3, 4, 5, 6, 12)$ de los seis que mostraron los mayores valores de *gap* entre $SGLS$ e ILS, favoreciendo al ILS, mostraron también en sus grafos de búsqueda la estructura de BV, la cual, se supone, algoritmos como el ILS deben explotar.

Para los casos del TSP, el algoritmo ILS mejoró en todos los casos los resultados obtenidos por el $SGLS$, en esta ocasión con porcentajes de 5% hasta diferencias de casi 90%, en este caso el ILS requirió un menor número de evaluaciones que el $SGLS$ para casi todos los casos (excepto para el caso 2 con el vecindario Two-Change).



(a) $BSL - gap$ (b) $QAP - gap$ (c) $TSP - gap$

(d) BSL - Evaluations (e) QAP - Evaluations (f) TSP - Evaluations

Figura 7. Valores de *gap* entre las calidades promedio de los algoritmos $SGLS$ e ILS y número de evaluaciones requeridas por el ILS por cada evaluación requerida por el $SGLS$ para los casos analizados del BSL y QAP.

Tabla IV. Valores de p obtenidos con la prueba Wilcoxon al comparar los resultados de los vecindarios para $SGLS$ e ILS utilizando el mismo esfuerzo computacional. El símbolo "-" indica que no se pasó la prueba, *i.e.* las medias no son estadísticamente diferentes.

(a) BSL instances

	Instance	**SGLS**	**ILS**
1	b0401	4.650E-39	-
2	b0411	1.035E-53	1.369E-12
3	b0501	1.188E-58	-
4	b0511	1.874E-62	5.924E-18
5	b0601	2.254E-63	4.416E-19
6	b0611	1.623E-65	3.306E-20
7	c0401	2.837E-36	-
8	c0411	1.522E-53	-
9	c0501	5.769E-62	-
10	c0511	1.089E-62	-
11	c0601	8.846E-61	-
12	c0611	1.425E-64	-

(b) QAP instances

	Instance	**SGLS**	**ILS**
1	bur26a	2.977E-34	6.754E-18
2	bur26b	2.562E-34	6.966E-18
3	chr25a	2.561E-34	7.061E-18
4	esc32a	2.347E-34	5.947E-18
5	esc32b	2.296E-34	4.241E-18
6	kra30a	2.803E-34	7.022E-18
7	lipa60a	2.557E-34	7.026E-18
8	sko49	2.559E-34	7.014E-18
9	tai30a	2.562E-34	7.066E-18
10	tai30b	1.076E-34	7.049E-18
11	tai60a	2.562E-34	7.066E-18
12	tai60b	2.562E-34	7.066E-18

Las tablas V(a) y V(b) muestran que, con un nivel de confianza del 99%, las diferencias observadas en los valores de *gap* son estadísticamente significativas. Estas pruebas se realizaron utilizando las 100 corridas del ILS y el número correspondiente de corridas del $SGLS$, de tal forma que ambos algoritmos utilizaron aproximadamente el mismo número de evaluaciones de la función objetivo. Se puede observar que, para casi todos los casos, las medias generadas por los algoritmos fueron estadísticamente diferentes. Solamente para los casos $BSL - C$, al utilizar el vecindario Entero, no se encontraron diferencias estadísticamente significativas entre las medias generadas.

V.3 Discusiones

Los resultados experimentales que se presentan en este capítulo muestran el análisis de las características de los grafos de búsqueda de los casos de prueba, así como la verificación de las hipótesis utilizando dos algoritmos de búsqueda local: $SGLS$ e ILS.

En el primera sección se muestra que los problemas de prueba presentan diversidad en las características de sus casos al utilizar diferentes vecindarios. Para los casos del BSL, el

Tabla V. Valores de p obtenidos con la prueba Wilcoxon al comparar los resultados de $SGLS$ e ILS con diferentes vecindarios y esfuerzos computacionales equivalentes. El símbolo "-" indica que no se pasó la prueba, *i.e.* las medias no son estadísticamente diferentes.

(a) BSL instances

	Instance	**Binary**	**Integer**
1	b0401	7.686E-32	2.672E-36
2	b0411	1.479E-32	1.867E-35
3	b0501	3.061E-33	1.6582E-35
4	b0511	4.102E-33	1.075E-16
5	b0601	2.335E-33	2.587E-34
6	b0611	7.227E-34	2.608E-25
7	c0401	8.938E-21	-
8	c0411	2.010E-29	-
9	c0501	5.983E-33	-
10	c0511	2.975E-33	-
11	c0601	1.996E-32	-
12	c0611	1.589E-33	-

(b) QAP instances

	Instance	**Insert**	**Swap**
1	bur26a	1.410E-16	3.383E-16
2	bur26b	4.926E-17	2.474E-17
3	chr25a	2.473E-17	1.538E-17
4	esc32a	2.163E-16	1.038E-17
5	esc32b	1.161E-16	5.970E-18
6	kra30a	1.064E-16	2.331E-17
7	lipa60a	7.042E-18	9.498E-17
8	sko49	1.099E-17	7.958E-18
9	tai30a	1.212E-17	2.473E-17
10	tai30b	3.517E-17	1.733E-17
11	tai60a	7.066E-18	2.784E-17
12	tai60b	8.985E-18	9.540E-18

vecindario Binario presenta la estructura de BV y la propiedad de LE, así como el mismo valor de CL que el vecindario Entero, el cual no muestra en sus grafos de búsqueda la estructura de BV como tampoco pasa la prueba para verificar la existencia de la propiedad de LE. En los casos del QAP, utilizando el vecindario Insert, no se encuentra la estructura de BV mientras que al utilizar el vecindario Swap se encuentra solamente en algunos de los grafos de búsqueda. La propiedad de LE se encuentra solamente para el vecindario Swap, y al comparar los promedios de CL el vecindario Insert muestra valores menores. Finalmente, para los casos del TSP, todos los grafos de búsqueda muestran la estructura de BV y la propiedad de LE al utilizar los tres vecindarios: Inser, Swap y Two-Change. Sin embargo, al utilizar Two-Change se obtiene un valor promedio mayor de CL.

Estos resultados permiten comparar conjuntos de casos con valores equivalentes de CL pero con diferencias para BV y LE (casos del BSL), equivalencias en BV y LE pero con diferencias en CL (casos del TSP), y diferencias en las tres medidas analizadas (casos del QAP). Esta variedad se utilizó, combinada con los supuestos de cada medida de dificultad, para establecer hipótesis específicas de la predicción que cada medida establecería sobre el desempeño de las búsquedas locales sobre los diferentes grafos de búsqueda de los casos

de cada problema de prueba. De tal forma que en la segunda sección del capítulo, los resultados experimentales muestran si las hipótesis específicas se cumplen o no para los casos analizados.

Las hipótesis relacionadas con la medida de CL, las cuales indicaban que a valores mayores se debía esperar un mejor desempeño de las búsquedas locales, se cumplen para el QAP con ambos algoritmos, para la mayoría de los casos del TSP con ambos algoritmos, y para el BSL solamente al utilizar el algoritmo ILS. Esto es, la predicción basada en los valores de CL falla para BSL entre vecindarios con valores equivalentes y diferencias entre las otras medidas, así como en algunos casos del TSP entre vecindarios con valores diferentes de CL y equivalencias entre las otras medidas. Lo cual, considerando las otras dos medidas, plantea dos escenarios diferentes en dos problemas de prueba diferentes para los cuales la predicción del desempeño de las búsquedas locales utilizando CL falla.

Por otro lado las hipótesis que están relacionadas a la estructura de BV indicaban que los casos que presentaran esta estructura debía ayudar a los algoritmos que la explotaran a obtener un mejor desempeño, lo cual se cumple para los casos del QAP con ambos algoritmos, para los casos del BSL solamente con el algoritmo ILS, y no se cumple para los casos del TSP. La predicción del desempeño de las búsquedas locales utilizando la existencia de la estructura de BV falla para BSL entre vecindarios con diferencias claras en sus estructuras de BV y en la propiedad de LE así como equivalencia en CL. También falla en los casos del TSP entre vecindarios con equivalencias en las estructuras de BV y en la propiedad de LE así como diferencias en CL. Por lo tanto, la predicción del desempeño de las búsquedas locales utilizando la estructura de BV falla para dos escenarios diferentes en dos problemas de prueba diferentes.

Finalmente, las hipótesis derivadas de la propiedad de LE solamente se cumplen para los casos del QAP con ambos algoritmos. Esto es, las hipótesis basadas en la utilización de la propiedad de LE falla para los casos del BSL entre vecindarios con diferencias en LE y BV pero equivalencias en CL, y los casos del TSP entre vecindarios con equivalencias en

LE y BV pero diferencias en CL. Lo cual plantea, al igual que con las medidas anteriores, dos escenarios diferentes en dos problemas de prueba diferentes para los cuales la predicción del desempeño de las búsquedas locales utilizando LE falla.

Todo lo anterior lleva a concluir en general que ninguna de las medidas de dificultad es capaz de predecir el desempeño de los algoritmos de búsqueda local para todos los casos de los problemas de prueba.

Capítulo VI

Resultados Multi-Objetivo

VI.1 Resultados del *MOGWW*

En esta sección se presentan los resultados obtenidos con el algoritmo *MOGWW* para dos problemas de optimización combinatoria multi-objetivo: calendarización de flujo de trabajos (*FSP* por sus siglas en inglés), y el problema de asignación cuadrática bi-objetivo (*bQAP* por sus siglas en inglés). Los experimentos se concentran en comparar los resultados obtenidos con algunos de los mejores resultados presentados en trabajos previos para el conjunto de casos analizados.

VI.1.1 Problema del Flujo de Tareas

Los experimentos realizados para el conjunto de casos de este problema tienen dos objetivos principales. El primero de ellos es el estudio del desempeño del algoritmo *MOGWW* al utilizar diferentes longitudes de caminata aleatoria. El segundo es el análisis del desempeño relativo al comparar el *MOGWW* con un conocido algoritmo genético multi-objetivo (*MOGA* por sus siglas en inglés).

El vecindario para este problema es el definido por el operador Insert descrito por Brizuela y Aceves (2003). La medida de distancia utilizada en la prueba de de-correlación es el mínimo número de veces que se necesita aplicar el operador Insert a una solución para llegar a la otra.

En el primer conjunto de experimentos, se utilizaron dos valores diferentes como longitud de la caminata aleatoria para la prueba de de-correlación: $L = 35$ y $L = 75$, las

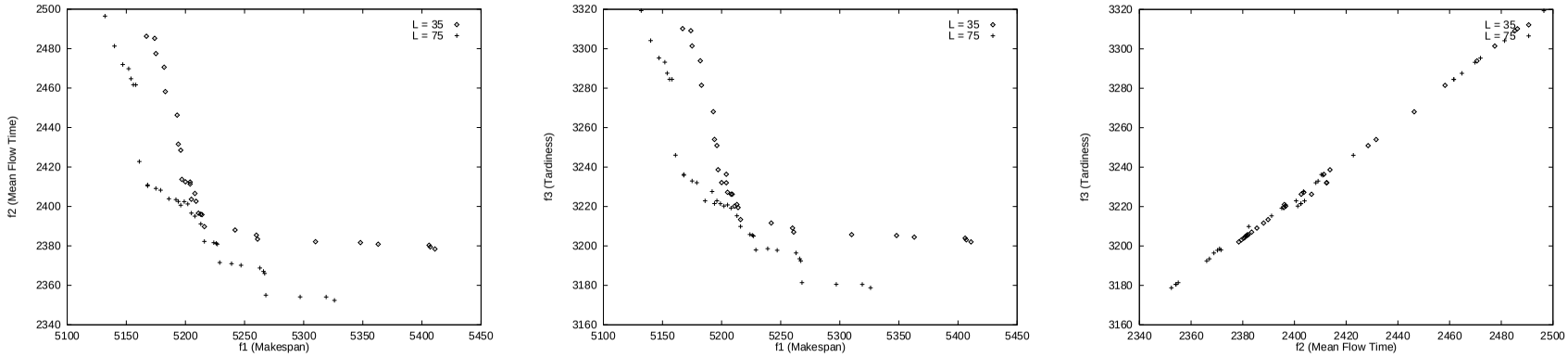
probabilidades obtenidas fueron 0.678 (con desviación de 2.26%) y 0.522 (desviación de 3.08%), respectivamente. Ya que el valor de referencia es 0.5, al utilizar $L = 35$ no se pasa la prueba, mientras que se considera que pasa al utilizar $L = 75$. Una vez que se ha decidido utilizar $L = 75$, los experimentos se enfocan en estudiar el desempeño del algoritmo $MOGWW$ comparando las soluciones que obtiene con las obtenidas por el $MOGA$.

VI.1.1.1 Longitud de Caminata Aleatoria

El conjunto de casos utilizado fue el que se describe en Brizuela y Aceves (2003), mismo que se encuentra publicamente disponible [1]. Diez casos de 75 trabajos y 20 máquinas fueron utilizados para los experimentos. Para cada caso el número de partículas se fijó en $P = 100$ y la longitud de de-correlación a $L = 75$, esta última se fijó tomando en cuenta la prueba de de-correlación propuesta por Carson (2001).

Para cada caso se realizaron 50 corridas, tomando la unión de todas las soluciones no-dominadas resultantes. La figura 8 muestra las proyecciones de las soluciones obtenidas utilizando dos diferentes longitudes de de-correlación, una que no pasa la prueba ($L = 35$), y otra que si lo hace ($L = 75$). Los resultados obtenidos con $L = 75$ muestran una cobertura del 20% sobre las soluciones obtenidas con $L = 35$, mientras que la cobertura inversa fue de 0.0%. Resultados equivalentes, no mostrados aquí se obtuvieron para el resto de los casos. Los resultados muestran la importancia de utilizar la longitud apropiada para obtener una buena uniformidad y, por lo tanto, un buen desempeño.

[1] *http://www.cicese.mx/∼cbrizuel/MOGWW/instances.html*



(a) Makespan - Mean Flow Time (b) Makespan - Tardiness (c) Mean Flow Time - Tardiness

(d) Makespan - Mean Flow Time (e) Makespan - Tardiness (f) Mean Flow Time - Tardiness

Figura 8. Proyecciones, por pares de funciones objetivo, de la unión de todas las soluciones no-dominadas de 50 corridas para el caso 6. Las gráficas (a), (b) y (c) muestran resultados del $MOGWW$, $L = 35$ vs. $L = 75$ ($P = 100$). Las gráficas (d), (e) y (f) muestran resultados de $MOGA$ vs. $MOGWW$ con $L = 75$ y $P = 100$.

VI.1.1.2 Comparación de Desempeño ($MOGA$ vs. $MOGWW$)

Para verificar la efectividad del algoritmo $MOGWW$ sobre el conjunto de casos, se comparan las soluciones obtenidas con $L = 75$ con los resultados obtenidos por el $MOGA$ (Brizuela y Aceves, 2003). La figura 8 muestra las proyecciones de los frentes no-dominados obtenidos por el $MOGWW$ y los reportados en Brizuela y Aceves (2003). Los parámetros utilizados por el $MOGA$ son: tamaño de población 100, número de generaciones 2000, probabilidad de cruzamiento 1.0 y probabilidad de mutación 0.1. Las soluciones obtenidas por el algoritmo $MOGWW$ muestran una cobertura de 36.57% sobre las soluciones obtenidas por $MOGA$, mientras que la cobertura inversa es 0.0%.

La tabla VI muestra las relaciones de cobertura entre la unión de todas las soluciones no-dominadas obtenidas en 50 corridas del $MOGWW$ y la unión de todas las soluciones no-dominadas obtenidas en 50 corridas del $MOGA$ (Brizuela y Aceves, 2003). Los resultados para todos los casos, muestran una clara superioridad del algoritmo $MOGWW$ sobre el $MOGA$. Aun cuando el algoritmo $MOGA$ utilizado en Brizuela y Aceves (2003) no es el mejor conocido para el problema, sus resultados son competitivos. Algunos resultados similares, no mostrados aquí, se obtuvieron al comparar el $MOGWW$ con el algoritmo $NSGA - II$ (Deb *et al.*, 2002) (uno de los más utilizados para problemas multi-objetivo) para el mismo conjunto de casos.

Al comparar el número de iteraciones utilizadas por los algoritmos, se tiene que el $MOGA$ itera un número fijo de veces (2000), mientras que el $MOGWW$ itera un promedio de 1028.26 con una desviación estándar de 10.19%. En cuanto al número de evaluaciones, todas las corridas del $MOGA$ utilizan 200000 evaluaciones, mientras que las corridas del $MOGWW$ utilizan un promedio de 715776.00 evaluaciones, con una desviación estándar de 10.83%. Por lo tanto el $MOGWW$ requiere más tiempo de cómputo que el $MOGA$, lo cual indica que si se incrementara el tamaño de la población del $MOGA$ se podrían obtener mejores resultados. Sin embargo, el principal punto a destacar es el mostrar que el algoritmo propuesto ($MOGWW$) se comporta de forma similar a su versión mono-objetivo. Por otro

Tabla VI. Relaciones de cobertura entre la unión de todas las soluciones no-dominadas de 50 corridas del $MOGA$ y el $MOGWW$. $\%c_1$ representa la cobertura del $MOGA$ sobre $MOGWW$, mientras que $\%c_2$ representa la cobertura del $MOGWW$ sobre $MOGA$.

	Algorithm	Iguales	$\%c_1$	$\%c_2$	No Comparables
Caso 0	$MOGA$	0.00%	0.00%	19.23%	80.77%
	$MOGWW$	0.00%	19.23%	0.00%	80.77%
Caso 1	$MOGA$	0.00%	0.00%	43.69%	56.31%
	$MOGWW$	0.00%	43.69%	0.00%	56.31%
Caso 2	$MOGA$	0.00%	0.00%	30.62%	69.38%
	$MOGWW$	0.00%	30.62%	0.00%	69.38%
Caso 3	$MOGA$	0.00%	0.00%	16.74%	83.26%
	$MOGWW$	0.00%	16.74%	0.00%	83.26%
Caso 4	$MOGA$	0.00%	0.00%	23.12%	76.88%
	$MOGWW$	0.00%	23.12%	0.00%	76.88%
Caso 5	$MOGA$	0.00%	0.00%	32.27%	67.73%
	$MOGWW$	0.00%	32.27%	0.00%	67.73%
Caso 7	$MOGA$	0.00%	0.00%	32.80%	67.20%
	$MOGWW$	0.00%	32.80%	0.00%	67.20%
Caso 8	$MOGA$	0.00%	0.00%	24.34%	75.66%
	$MOGWW$	0.00%	24.34%	0.00%	75.66%
Caso 9	$MOGA$	0.00%	0.00%	59.10%	40.90%
	$MOGWW$	0.00%	59.10%	0.00%	40.90%

lado, una característica importante es que el algoritmo necesita solamente dos parámetros, uno de los cuales se puede fijar fácilmente antes de la corrida del algoritmo.

VI.1.2 QAP Bi-Objetivo

Para cada caso se utilizó el algoritmo $MOGWW$ con tres valores diferentes $N, 2N$, y $4N$ para sus dos parámetros: P y L, por lo que se utilizaron nueve combinaciones de valores diferentes para los parámetros. Para cada combinación se realizaron 50 corridas y se tomó el conjunto de soluciones no-dominadas de todas las corridas para realizar las comparaciones. Se sintonizó el $MOGWW$ para que conservara solamente el mejor frente en cada iteración y reemplazara con copias perturbadas de ese frente a las soluciones del resto. El vecindario utilizado lo define el operador Swap descrito en Paquete y Stützle (2006), el cual considera como vecino a todas las soluciones dadas por el intercambio de la asignación de la ubicación de dos elementos.

VI.1.2.1 Frentes Pareto: formas y calidad

Se compararon los resultados del $MOGWW$ utilizando cada combinación de los valores de P y L, con los mejores obtenidos por Paquete y Stützle (2006) utilizando su algoritmo $W - RoTS$. La tabla VII muestra la cobertura de $W - RoTS$ sobre $MOGWW$, así como los valores para los indicadores binarios, para todos los casos. Por ejemplo, en el primer renglón y primera columna se muestra la relación de cobertura entre $W - RoTS$ y $MOGWW$ ($\%dom = 13.0$), en la segunda columna muestra el valor del indicador épsilon entre $MOGWW$ y $W - RoTS$ ($\epsilon_1 = 1.125$), la tercera columna muestra el valor inverso para la misma medida ($\epsilon_2 = 0.982$), y, finalmente, la cuarta columna da el tiempo de ejecución promedio de 50 corridas en segundos ($t = 0.04$), todos estos resultados para el caso con valor de correlación -0.75, tamaño del caso 25, y valores de los parámetros $P = L = N$. La cobertura del $MOGWW$ sobre el $W - RoTS$ ($\%c_2$) es 0.0 para todos los casos, por lo que no se muestra en la tabla. Los valores en negritas son los mejores resultados para cada

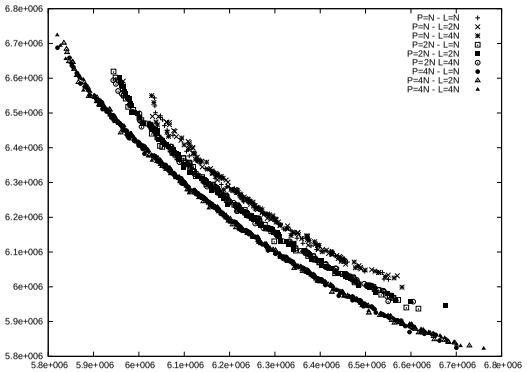
combinación de tamaño de problema y correlación en términos de los valores de $\%dom$ y ϵ_1. Para una correlación de -0.75, tamaños 25 y 75, se marcan dos mejores resultados debido a que uno de ellos es mejor en términos de $\%dom$ (es menos cubierto por $W - RoTS$) y el otro es mejor en términos de ϵ_1 (se encuentra más cercano al frente de $W - RoTS$).

Para los casos con correlación -0.75 se puede notar que el $MOGWW$ parece desempeñarse mejor al incrementarse el valor de P, ya que los mejores valores se obtienen utilizando siempre un valor de $P = 4N$ y diferentes valores de L. Este comportamiento cambia a medida que se incrementa el valor de correlación, de tal forma que al llegar a un valor de 0.75, el valor de L parece ser ahora el que mayor influencia tiene sobre el desempeño del $MOGWW$ ya que los mejores resultados se obtuvieron al utilizar $L = 4N$ y diferentes valores de P. Nótese que para un valor de correlación de 0.0 se necesita incrementar ambos parámetros para obtener mejores resultados, *i.e.* no existe un sesgo que se pueda aprovechar al incrementar solamente uno de los parámetros del $MOGWW$.

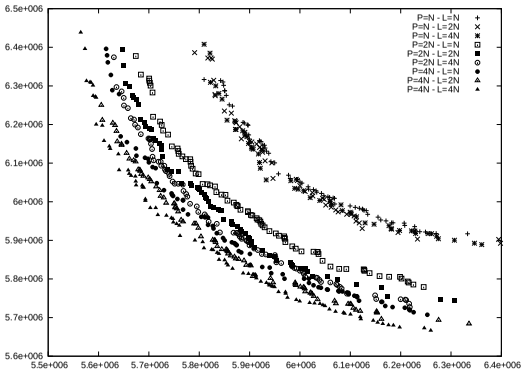
La figura 9 muestra los frentes obtenidos por el $MOGWW$ para los casos de tamaño 50 y correlaciones -0.75, 0.0 y 0.75. Se puede observar que la forma de los frentes cambia de acuerdo a los valores de correlación. Observamos en esa figura que, para un valor de correlación -0.75, los frentes mejoran solamente al incrementar el valor de P. Este comportamiento cambia a medida que se incrementa el valor de la correlación, llegando a que el valor de L tenga la mayor influencia para un valor de correlación 0.75. Se observa el mismo comportamiento para los casos de tamaño 25 y 75, no mostrados aquí.

Tabla VII. Cobertura de $W - RoTS$ sobre $MOGWW$ ($\%c_1$) e indicadores binarios: $\epsilon_1 = I_\epsilon(MOGWW, W - RoTS)$; $\epsilon_2 = I_\epsilon(W - RoTS, MOGWW)$, para casos con correlación (ρ) $-0.75, 0.0$ y 0.75.

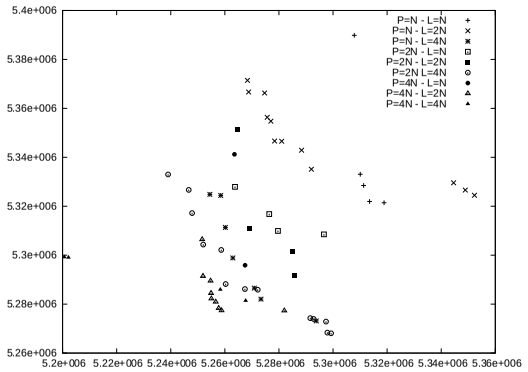
$\rho = -0.75$	N	$L = N$				$L = 2N$				$L = 4N$			
		$\%c_1$	ϵ_1	ϵ_2	$t(s)$	$\%c_1$	ϵ_1	ϵ_2	$t(s)$	$\%c_1$	ϵ_1	ϵ_2	$t(s)$
$P = N$	25	13.0	1.125	0.982	0.04	13.5	1.129	0.981	0.04	14.2	1.132	0.981	0.06
	50	11.8	1.114	0.981	0.34	11.7	1.115	0.981	0.54	11.8	1.113	0.981	0.94
	75	10.4	1.101	0.984	1.72	10.4	1.101	0.984	2.66	10.5	1.099	0.984	4.54
$P = 2N$	25	11.6	1.111	0.984	0.1	11.4	1.111	0.986	0.16	12.0	1.107	0.987	0.3
	50	10.4	1.098	0.984	1.62	10.4	1.100	0.985	2.42	10.4	1.102	0.984	4.16
	75	9.6	1.085	0.988	7.76	9.6	1.087	0.987	12.04	10.4	1.086	0.999	21.7
$P = 4N$	25	**9.2**	**1.084**	0.991	0.46	**8.7**	**1.085**	0.992	0.76	9.2	1.085	0.992	1.36
	50	**8.5**	**1.077**	0.989	6.9	8.8	1.078	0.989	11.36	8.7	1.077	0.989	19.96
	75	8.1	1.072	0.991	34.76	**8.2**	**1.070**	0.990	55.9	**8.0**	**1.071**	0.991	128.52
$\rho = 0.0$													
$P = N$	25	47.4	1.083	0.956	0.08	45.9	1.091	0.960	0.12	46.8	1.086	0.957	0.22
	50	59.1	1.070	0.959	1.36	57.9	1.066	0.961	2.32	57.4	1.068	0.963	3.84
	75	63.1	1.061	0.966	7.3	62.1	1.058	0.966	12.04	65.7	1.061	0.964	31.52
$P = 2N$	25	32.1	1.063	0.973	0.22	27.4	1.054	0.982	0.44	26.0	1.050	0.982	0.8
	50	39.9	1.045	0.976	4.04	35.1	1.039	0.981	8.0	33.0	1.036	0.982	15.22
	75	41.5	1.040	0.980	22.54	39.5	1.035	0.981	42.04	36.3	1.033	0.983	96.7
$P = 4N$	25	20.0	1.050	0.987	0.46	17.3	1.039	0.990	0.88	**16.1**	**1.033**	0.991	1.68
	50	29.4	1.033	0.985	8.4	25.0	1.031	0.988	15.58	**21.5**	**1.025**	0.990	31.76
	75	34.0	1.030	0.984	43.56	26.6	1.026	0.990	86.44	**22.3**	**1.022**	0.992	173.88
$\rho = 0.75$													
$P = N$	25	97.1	1.031	0.975	0.08	83.3	1.019	0.987	0.14	85.7	1.020	0.985	0.28
	50	100.0	1.030	0.978	0.68	100.0	1.026	0.978	2.34	100.0	1.021	0.986	4.8
	75	100.0	1.020	0.983	7.62	100.0	1.020	0.983	12.72	**100.0**	**1.013**	0.990	25.68
$P = 2N$	25	88.1	1.027	0.980	0.14	68.6	1.021	0.996	0.26	**28.6**	**1.012**	0.995	0.5
	50	100.0	1.023	0.981	4.24	100.0	1.022	0.983	4.14	100.0	1.019	0.987	8.2
	75	100.0	1.022	0.984	11.82	100.0	1.018	0.986	21.9	100.0	1.014	0.990	48.06
$P = 4N$	25	85.7	1.024	0.980	0.26	71.4	1.016	0.990	0.44	85.7	1.019	0.988	0.84
	50	100.0	1.022	0.983	4.12	100.0	1.019	0.987	7.34	**81.3**	**1.017**	0.991	15.92
	75	100.0	1.019	0.984	21.44	100.0	1.019	0.984	38.04	100.0	1.014	0.989	76.44



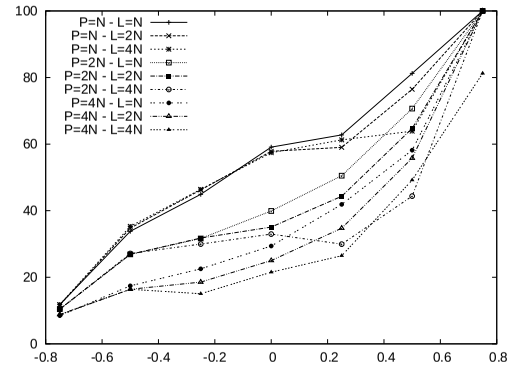
(a) Frentes para $\rho = -0.75$



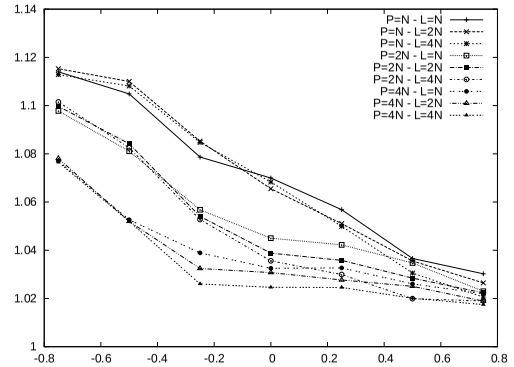
(b) Frentes para $\rho = 0.0$



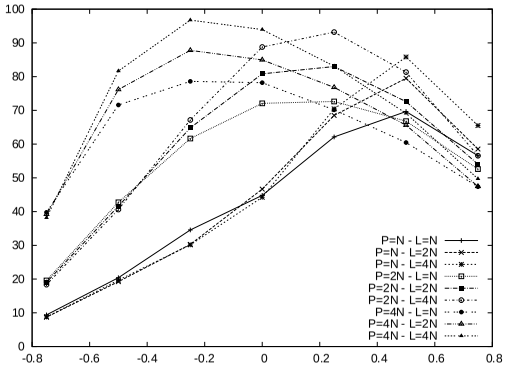
(c) Frentes para $\rho = 0.75$



(d) ρ vs. $\%c(W - RoTS, MOGWW)$



(e) ρ vs. $I_\epsilon(MOGWW, W - RoTS)$



(f) ρ vs. (RW/P)

Figura 9. Resultados para los casos de tamaño 50. Las gráficas (a), (b) y (c) muestran los frentes no-dominados obtenidos por el $MOGWW$, las gráficas (d) y (e) muestran la comparación entre $W - RoTS$ y $MOGWW$, finalmente, la gráfica (f) muestra la relación RW/P.

La figura 9(d) muestra el porcentaje de cobertura del $W - RoTS$ sobre el $MOGWW$ para los casos de tamaño 50. Note que el $MOGWW$ se desempeña mejor, en términos de cobertura, para casos con correlación negativa y peor para los que tienen correlación positiva. Por otro lado, en la figura 9 (e) se muestran los indicadores binarios $I_\epsilon(MOGWW, W - RoTS)$ entre los frentes generados por $W - RoTS$ y $MOGWW$, para los casos de tamaño 50. Considerando esta medida el $MOGWW$ se desempeña mejor para correlciones positivas. Se observan los mismos resultados, en términos de cobertura e indicadores binarios, para los casos de tamaño 25 y 75, no mostrados aquí.

Los resultados obtenidos para los indicadores binarios parecen contradecir los obtenidos para las relaciones de cobertura. Esto se debe a que las relaciones de cobertura solamente indican si un conjunto de soluciones está cubriendo a otro conjunto de soluciones, y no informan sobre la distancia entre ellos en términos de los valores de función objetivo. Por otro lado, los indicadores binarios solamente informan sobre la cercanía entre las soluciones de ambos conjuntos, en términos de los valores de función objetivo. Considerando que los casos con correlación positiva muestran frentes Pareto más pequeños resulta más difícil cubrirlos, incluso con un frente que se encuentre cercano en términos de los valores de las funciones objetivo.

Todos estos resultados muestran que para los casos con valores de correlación positiva se necesita incrementar el valor de L para mejorar el desempeño del algoritmo $MOGWW$, mientras que para casos con correlaciones negativas esa mejora se obtiene al incrementar el valor de P. En general, tanto $MOGWW$ como $W - RoTS$ se desempeñan mejor, en términos de relaciones de cobertura, para casos con correlación negativa. Mientras que, para los casos con correlación positiva el $MOGWW$ se desempeña peor en términos de relaciones de cobertura, pero mejor en términos de los indicadores binarios.

VI.1.2.2 Caminatas Aleatorias: Convergencia y Calidad

Tomando en cuenta que el algoritmo $MOGWW$ utiliza caminatas aleatorias (RW por sus siglas en inglés) como el único mecanismo para mantener la diversidad entre las partículas, el número promedio de RW por partícula podría ayudar a entender los resultados obtenidos.

La figura 9(f) muestra la relación entre el número promedio de RW y el número de partículas para los casos de tamaño 50. El valor de RW representa el total de caminatas aleatorias de longitud L, contabilizadas a partir del primer paso del algoritmo hasta el punto en que se llega a un frente óptimo local no-dominado. Se puede observar que esa relación crece a medida que se incrementa el valor de P para casos con correlaciones negativas. Sin embargo, este comportamiento cambia a medida que se consideran los casos con correlaciones positivas, y para los casos con correlación 0.75, la relación RW/P crece a medida que se incrementan los valores de L en lugar de los de P. Además, para cada valor de correlación se obtienen los mejores resultados, en términos de cobertura e indicadores binarios, en aquellas combinaciones de P y L con un mayor valor de RW/P como se puede observar en la tabla VII. Esto es, parece existir una relación directa entre la relación RW/P y la calidad de las soluciones dadas por el $MOGWW$. Esto puede deberse a que la relación RW/P da una idea de qué tan rápido se pierde la diversidad entre las partículas del $MOGWW$, quedando atrapado en un óptimo local. Resultados equivalentes se observan para los casos de tamaño 25 y 75, no mostrados aquí.

VI.2 $MOGWW$ Híbrido

En esta sección se presentan los resultados obtenidos por las versiones híbridas del $MOGWW$. En particular se prueban esas versiones sobre el problema de $bQAP$, considerando que, según los resultados de la sección anterior, fue en el que obtiene los peores resultados al utilizar la versión sin hibridizar.

El primer conjunto de experimentos consiste en verificar cuál de las dos versiones

híbridas propuestas ($MOGWW-F$ y $MOGWW-P$) se desempeña mejor en el problema de prueba ($bQAP$), para lo cual se parte de la configuración de los experimentos utilizada para la versión original. Una vez que se ha determinado la versión con mejor desempeño, se procede a comparar sus resultados contra los mejores conocidos para el conjunto de casos de prueba (obtenidos con $W-RoTS$ Paquete y Stützle (2006)).

VI.2.1 MOGWW vs. Versiones Híbridas

Se necesita verificar de inicio si las versiones híbridas $MOGWW-F$ y $MOGWW-P$ son mejores que el algoritmo original $MOGWW$. Para ello se calculó la cobertura y los indicadores binarios entre los frentes no-dominados generados en cada corrida por $MOGWW$ y sus equivalentes generados por $MOGWW-F$ y $MOGWW-P$. Esas medidas se calcularon para los 50 pares de corridas, tomando los valores promedio de la cobertura y los indicadores binarios, y calculando la medida de cobertura como el porcentaje de veces que las soluciones de un algoritmo dominaron completamente a las soluciones del otro.

La Tabla VIII muestra los resultados obtenidos al comparar el algoritmo $MOGWW$ con $MOGWW-F$ y $MOGWW-P$, utilizando $P=L=N$. El desempeño de las versiones híbridas sobre el $MOGWW$, en términos de cobertura, es mejor para $MOGWW-P$ en todos los casos. Para los que tienen correlación 0.75, las diferencias son de $2.1\%, 2.9\%$ y 8.6% para los tamaños de problema $25, 50$ y 75, respectivamente, mientras que, para el resto de los casos la diferencia es de 20% a 31.9%. Si se consideran las relaciones de dominancia, ambas versiones híbridas muestran resultados similares para casos con correlación 0.75, mientras que para el resto de los casos el $MOGWW-F$ muestra resultados iguales o mejores que $MOGWW-P$. Ambas versiones híbridas muestran una fuerte dominancia y cobertura sobre el algoritmo original; sin embargo, el $MOGWW$ utiliza solamente de un 2% a un 41% del tiempo de ejecución requerido por $MOGWW-F$ y $MOGWW-P$.

Tabla VIII. Comparación de resultados del algoritmo $MOGWW$ y sus versiones híbridas $MOGWW-F$ y $MOGWW-P$ utilizando $P = L = N$: cobertura sobre $MOGWW$ $\%c_1$, cobertura del $MOGWW$ sobre las versiones híbridas $\%c_2$, dominancia sobre $MOGWW$ $\%d_1$, dominancia de $MOGWW$ sobre las versiones híbridas $\%d_2$, indicadores binarios: $\epsilon_1 = I_\epsilon(MOGWW, HybridMOGWW)$; $\epsilon_2 = I_\epsilon(HybridMOGWW, MOGWW)$, y la relación (r) entre el tiempos promedio de ejecución del $MOGWW$ y las versiones híbridas.

ρ	N	MOGWW vs. MOGWW-F ($P = L = N$)							MOGWW vs. MOGWW-P ($P = L = N$)						
		$\%c_1$	$\%c_2$	$\%d_1$	$\%d_2$	ϵ_1	ϵ_2	r	$\%c_1$	$\%c_2$	$\%d_1$	$\%d_2$	ϵ_1	ϵ_2	r
0.75	25	86.4	3.0	88.0	4.0	1.021	0.989	0.24	88.5	5.9	84.0	6.0	1.021	0.988	0.29
	50	92.0	2.1	94.0	2.0	1.014	0.992	0.09	94.9	1.2	94.0	2.0	1.016	0.989	0.10
	75	91.0	2.4	90.0	2.0	1.009	0.995	0.14	99.6	0.0	96.0	0.0	1.013	0.991	0.17
0.50	25	62.5	1.2	80.0	0.0	1.030	0.989	0.28	82.5	0.0	92.0	0.0	1.033	0.984	0.36
	50	68.5	0.2	96.0	0.0	1.018	0.992	0.19	93.3	0.0	94.0	0.0	1.024	0.988	0.27
	75	68.9	0.0	100.0	0.0	1.015	0.992	0.28	96.9	0.0	96.0	0.0	1.019	0.990	0.41
0.25	25	57.8	0.0	100.0	0.0	1.041	0.982	0.22	86.6	0.0	96.0	0.0	1.043	0.981	0.33
	50	65.2	0.0	100.0	0.0	1.028	0.983	0.18	93.8	0.0	100.0	0.0	1.034	0.984	0.32
	75	68.1	0.0	100.0	0.0	1.021	0.986	0.19	94.0	0.0	100.0	0.0	1.027	0.989	0.34
0.00	25	54.3	0.0	98.0	0.0	1.048	0.979	0.21	84.0	0.0	94.0	0.0	1.054	0.979	0.40
	50	62.8	0.0	100.0	0.0	1.035	0.977	0.14	93.4	0.0	100.0	0.0	1.040	0.982	0.29
	75	69.1	0.0	100.0	0.0	1.030	0.979	0.11	96.0	0.0	100.0	0.0	1.035	0.986	0.26
−0.25	25	49.9	0.0	100.0	0.0	1.050	0.977	0.16	78.6	0.0	94.0	0.0	1.050	0.982	0.30
	50	61.2	0.0	100.0	0.0	1.038	0.976	0.11	90.6	0.0	98.0	0.0	1.041	0.983	0.26
	75	65.7	0.0	100.0	0.0	1.034	0.977	0.07	93.9	0.0	100.0	0.0	1.036	0.985	0.23
−0.50	25	44.6	0.0	100.0	0.0	1.051	0.979	0.11	73.7	0.0	94.0	0.0	1.049	0.982	0.18
	50	53.3	0.0	100.0	0.0	1.043	0.977	0.07	84.6	0.0	100.0	0.0	1.039	0.985	0.17
	75	56.9	0.0	100.0	0.0	1.038	0.979	0.04	88.8	0.0	100.0	0.0	1.033	0.987	0.14
−0.75	25	26.5	0.0	94.0	0.0	1.037	0.991	0.11	52.0	0.0	78.0	0.0	1.030	0.992	0.20
	50	34.6	0.0	100.0	0.0	1.034	0.988	0.03	65.4	0.0	96.0	0.0	1.024	0.991	0.07
	75	37.7	0.0	100.0	0.0	1.031	0.988	0.02	68.9	0.0	100.0	0.0	1.020	0.992	0.05

A continuación se comparan las dos versiones híbridas entre ellas para determinar cuál de ellas utilizar en el resto de los experimentos. La tabla IX muestra los resultados obtenidos al comparar $MOGWW-F$ y $MOGWW-P$ utilizando $P = L = N$. Esta tabla contiene la cobertura y relaciones de dominancia, los indicadores binarios y la relación entre los tiempos de ejecución para diferentes valores de correlación entre las matrices de flujo. Ambos algoritmos muestran casi los mismos valores de cobertura y dominancia entre ellos para una correlación de 0.75 y $N = 25$, para el resto de los casos $MOGWW - P$ mejora claramente a $MOGWW - F$ en términos de cobertura, mientras que en términos de dominancia esa mejora solo se obtiene para los casos con correlación positiva. Adicionalmente, $MOGWW - F$ utiliza de 121% a 324% más tiempo de ejecución del requerido por $MOGWW - P$.

Una vez que se ha determinado la mejor versión híbrida del $MOGWW$, y dado que la versión original requiere un menor tiempo de ejecución, se incrementan los valores de los parámetros P y L para realizar una comparación equitativa con la versión híbrida seleccionada. La tabla X muestra los resultados obtenidos al comparar $MOGWW - P$ utilizando $P = L = N$ y $MOGWW$ con $P = L = 2N$. Se puede observar que $MOGWW - P$ mejora claramente a $MOGWW$ en términos de valores de cobertura, dominancia e indicadores binarios. Más aún, al utilizar esos valores de los parámetros, el tiempo requerido por $MOGWW-P$ es menor en 12 de los 21 casos, es mayor para 8 casos y aproximadamente el mismo que $MOGWW$ en el restante. Esto es, al utilizar $P = L = N$ para $MOGWW-P$ y $P = L = 2N$ para $MOGWW$, el primero mejora claramente al segundo utilizando una menor cantidad de tiempo de cómputo para más de la mitad de los casos.

VI.2.2 MOGWW-P vs. W-RoTs

Una vez verificado que el $MOGWW - P$ mejora al original $MOGWW$, se comparó la versión híbrida con los mejores resultados conocidos para el conjunto de casos. Para ello, y tomando en cuenta que solo se cuenta con un frente de referencia proveniente de $W - RoTs$ para cada caso, se calcularon los valores de las medidas comparando el resultado de cada

Tabla IX. Comparación de resultados entre $MOGWW-P$ y $MOGWW-F$ utilizando $P=L=N$: cobertura de $MOGWW-P$ sobre $MOGWW-F$ $\%c_1$, cobertura de $MOGWW-F$ sobre $MOGWW-P$ $\%c_2$, dominancia de $MOGWW-P$ sobre $MOGWW-F$ $\%d_1$, dominancia de $MOGWW-F$ sobre $MOGWW-P$ $\%d_2$, indicadores binarios: $\epsilon_1 = I_\epsilon(MOGWW-F, MOGWW-P)$; $\epsilon_2 = I_\epsilon(MOGWW-P, MOGWW-F)$, y relación (r) entre tiempos promedio de ejecución de $MOGWW-F$ y $MOGWW-P$.

ρ	N	MOGWW-F vs. MOGWW-P ($P=L=N$)						
		$\%c_1$	$\%c_2$	$\%d_1$	$\%d_2$	ϵ_1	ϵ_2	r
0.75	25	35.2	30.3	34.0	34.0	1.005	1.004	1.21
	50	52.9	17.5	52.0	18.0	1.005	1.000	1.15
	75	67.6	7.4	66.0	10.0	1.005	0.998	1.15
0.50	25	30.5	7.8	22.0	4.0	1.013	1.006	1.29
	50	54.8	3.8	50.0	4.0	1.010	1.000	1.41
	75	48.4	2.2	38.0	2.0	1.007	1.001	1.47
0.25	25	23.1	2.5	10.0	0.0	1.015	1.012	1.50
	50	41.0	0.1	18.0	0.0	1.012	1.005	1.73
	75	40.7	0.1	12.0	0.0	1.009	1.005	1.85
0.00	25	25.4	0.6	12.0	0.0	1.020	1.015	1.90
	50	24.9	0.1	0.0	0.0	1.011	1.011	2.11
	75	32.7	0.0	2.0	0.0	1.010	1.011	2.42
−0.25	25	19.2	0.1	0.0	0.0	1.018	1.021	1.90
	50	24.5	0.0	0.0	0.0	1.012	1.016	2.46
	75	24.8	0.0	0.0	0.0	1.010	1.015	3.04
−0.50	25	18.6	0.0	0.0	0.0	1.017	1.023	1.73
	50	21.6	0.0	0.0	0.0	1.012	1.022	2.56
	75	20.3	0.0	0.0	0.0	1.009	1.021	3.24
−0.75	25	16.6	0.0	0.0	0.0	1.015	1.021	1.80
	50	16.3	0.0	0.0	0.0	1.008	1.021	2.12
	75	16.2	0.0	0.0	0.0	1.006	1.020	2.32

Tabla X. Comparación de resultados entre $MOGWW - P$ utilizando $P = L = N$ y $MOGWW$ utilizando $P = L = 2N$: cobertura de $MOGWW - P$ sobre $MOGWW$ $\%c_1$, sobertura de $MOGWW$ sobre $MOGWW - P$ $\%c_2$, dominancia de $MOGWW - P$ sobre $MOGWW$ $\%d_1$, dominancia de $MOGWW$ sobre $MOGWW - P$ $\%d_2$, indicadores binarios: $\epsilon_1 = I_\epsilon(MOGWW, MOGWW - P)$; $\epsilon_2 = I_\epsilon(MOGWW - P, MOGWW)$, y relación (r) entre tiempos promedio de ejecución de $MOGWW$ y $MOGWW - P$.

ρ	N	MOGWW ($P = L = 2N$) vs. MOGWW-P ($P = L = N$)						
		$\%c_1$	$\%c_2$	$\%d_1$	$\%d_2$	ϵ_1	ϵ_2	r
0.75	25	63.3	11.2	66.0	14.0	1.012	0.997	0.93
	50	76.9	6.4	80.0	8.0	1.009	0.996	0.60
	75	88.0	2.2	86.0	4.0	1.008	0.996	0.48
0.50	25	42.8	6.7	32.0	8.0	1.014	1.004	1.29
	50	64.7	0.5	52.0	2.0	1.012	1.000	0.94
	75	65.0	0.0	50.0	0.0	1.010	1.001	0.82
0.25	25	39.0	0.0	12.0	0.0	1.019	1.009	1.67
	50	46.3	0.0	18.0	0.0	1.016	1.006	1.32
	75	51.2	0.0	12.0	0.0	1.013	1.005	1.19
0.00	25	37.1	0.0	6.0	0.0	1.025	1.014	2.20
	50	42.3	0.0	10.0	0.0	1.019	1.009	1.71
	75	49.8	0.0	14.0	0.0	1.017	1.009	1.51
−0.25	25	41.5	0.0	12.0	0.0	1.029	1.014	1.70
	50	51.3	0.0	20.0	0.0	1.024	1.008	1.61
	75	58.5	0.0	36.0	0.0	1.022	1.006	1.48
−0.50	25	40.4	0.0	10.0	0.0	1.031	1.012	1.45
	50	54.9	0.0	28.0	0.0	1.027	1.006	1.18
	75	60.3	0.0	32.0	0.0	1.024	1.004	1.00
−0.75	25	36.6	0.0	20.0	0.0	1.025	1.007	0.80
	50	45.8	0.0	26.0	0.0	1.019	1.004	0.48
	75	50.6	0.0	22.0	0.0	1.017	1.003	0.37

corrida del $MOGWW - P$ con ese frente y se tomaron los valores promedio, siguiendo el enfoque propuesto por Paquete y Stützle (2006). En el caso de la medida de dominancia, ésta representa el porcentaje de veces en que el frente de $W - RoTs$ domina completamente a los frentes generados por $MOGWW - P$.

La tabla XI muestra los resultados obtenidos al comparar $MOGWW - P$, utilizando $P = L = \{N, 2N, 4N\}$, y $W - RoTs$. Debido a que los frentes del $MOGWW - P$ nunca cubren ($\%c_2$) ni dominan ($\%d_2$) completamente al frente de referencia, dando como resultado un valor de 0% para todos los casos, no se presentan esas columnas en esta tabla. Se puede observar que la cobertura sobre $MOGWW - P$ se decrementa a medida que se utilizan mayores valores para P y L, alcanzando un máximo de 97.9% y un mínimo de 1.0% para $P = L = 4N$, con valores menores a 9% para todos los casos con correlación negativa. Sin embargo, en términos de dominancia la mayoría de los valores se mantienen sobre 90% para los mismos valores de P y L, solamente para los casos con correlación -0.75 se nota un decremento significativo llegando incluso a 0% para un par de casos. Aunado a ello, para 7 de los 21 casos la dominancia es 100%, y para 14 de ellos es mayor a 95%. En términos de tiempos de ejecución, $MOGWW - P$ sigue utilizando solamente una fracción del tiempo requerido por $W - RoTs$, utilizando desde 3.9% hasta 28.4% del tiempo usado por $W - RoTs$ (Paquete y Stützle, 2006) para obtener el frente de referencia.

Tabla XI. Comparación de resultados entre $W-RoTs$ y $MOGWW-P$: dominancia de $W-RoTs$ sobre $MOGWW-P$ $\%c_1$, dominancia de $W-RoTs$ sobre $MOGWW-P$ $\%d_1$, indicadores binarios: $\epsilon_1 = I_\epsilon(MOGWW-P, W-RoTs)$; $\epsilon_2 = I_\epsilon(W-RoTs, MOGWW-P)$, y relación (r) entre tiempos promedio de ejecución de $MOGWW-P$ y $W-RoTs$.

ρ	N	MOGWW-P ($P=L=N$) vs. W-RoTs					MOGWW-P ($P=L=2N$) vs. W-RoTs					MOGWW-P ($P=L=4N$) vs. W-RoTs				
		$\%c_1$	$\%d_1$	ϵ_1	ϵ_2	r	$\%c_1$	$\%d_1$	ϵ_1	ϵ_2	r	$\%c_1$	$\%d_1$	ϵ_1	ϵ_2	r
0.75	25	93.7	100.0	1.034	0.974	0.003	88.9	100.0	1.027	0.981	0.013	78.6	100.0	1.020	0.988	0.059
	50	100.0	100.0	1.026	0.980	0.010	96.7	100.0	1.021	0.984	0.039	92.5	98.0	1.017	0.988	0.158
	75	100.0	100.0	1.019	0.986	0.018	99.0	100.0	1.016	0.988	0.071	97.9	100.0	1.012	0.992	0.284
0.50	25	49.4	100.0	1.043	0.981	0.003	38.9	100.0	1.033	0.988	0.013	36.1	98.0	1.025	0.991	0.058
	50	58.4	100.0	1.036	0.983	0.009	47.1	98.0	1.028	0.988	0.036	45.0	100.0	1.022	0.990	0.151
	75	44.1	100.0	1.027	0.989	0.016	36.1	100.0	1.021	0.992	0.065	33.8	100.0	1.016	0.993	0.263
0.25	25	28.8	100.0	1.058	0.985	0.003	23.1	100.0	1.039	0.990	0.012	20.8	98.0	1.032	0.992	0.053
	50	33.2	100.0	1.047	0.988	0.007	26.1	100.0	1.034	0.991	0.034	21.3	100.0	1.026	0.994	0.147
	75	24.5	100.0	1.035	0.991	0.013	18.7	98.0	1.024	0.994	0.060	17.0	100.0	1.018	0.996	0.255
0.00	25	17.2	100.0	1.084	0.987	0.002	16.7	100.0	1.060	0.990	0.011	15.0	98.0	1.044	0.993	0.054
	50	18.3	100.0	1.056	0.989	0.006	17.1	100.0	1.040	0.993	0.031	14.5	98.0	1.026	0.995	0.146
	75	14.8	100.0	1.049	0.993	0.011	15.6	100.0	1.032	0.994	0.054	13.7	100.0	1.021	0.997	0.254
−0.25	25	12.8	100.0	1.093	0.990	0.002	9.7	98.0	1.064	0.994	0.010	7.6	88.0	1.040	0.997	0.054
	50	12.3	100.0	1.069	0.993	0.006	9.8	100.0	1.049	0.994	0.027	8.3	96.0	1.030	0.997	0.144
	75	11.1	100.0	1.059	0.995	0.009	10.6	100.0	1.041	0.995	0.045	7.1	88.0	1.024	0.998	0.247
−0.50	25	6.9	98.0	1.101	0.993	0.003	6.0	98.0	1.077	0.995	0.009	5.7	90.0	1.047	0.997	0.051
	50	6.8	100.0	1.093	0.995	0.006	5.8	100.0	1.072	0.996	0.023	4.9	96.0	1.047	0.998	0.131
	75	4.7	100.0	1.078	0.997	0.009	4.6	100.0	1.062	0.997	0.036	3.9	90.0	1.040	0.999	0.216
−0.75	25	2.7	40.0	1.132	1.001	0.002	2.3	2.0	1.110	1.002	0.008	2.3	0.0	1.082	1.004	0.039
	50	1.7	66.0	1.111	1.000	0.007	1.4	26.0	1.094	1.000	0.022	1.3	2.0	1.075	1.003	0.095
	75	1.1	14.0	1.094	1.001	0.013	1.2	2.0	1.083	1.001	0.035	1.0	0.0	1.066	1.002	0.166

La figura 10 muestra la cobertura de $W - RoTS$ sobre $MOGWW - P$ y los correspondientes valores de los indicadores binarios entre sus resultados. La figura muestra los resultados para $P = L = \{N, 2N, 4N\}$ y todos los casos con diferentes valores de correlación entre sus matrices de flujo. Se puede notar que para casi todos los casos se puede mejorar los resultados al incrementar los valores de P y L. Las mejoras obtenidas con estos cambios son más significativas para los casos con correlación negativa si se consideran los indicadores binarios, mientras que en términos de los valores de cobertura las mayores mejoras se obtienen para los casos más pequeños. Se pueden observar diferentes variaciones en los valores de cobertura, desde menores a 1% para casos con correlación negativa hasta casi 15% para casos con correlación positiva. En el caso de los indicadores binarios, la figura muestra mejoras para todos los casos, mismas que van desde 1% hasta 5%.

Los resultados obtenidos para los indicadores binarios parecen contradecir los obtenidos para las relaciones de cobertura. Esto se debe a que las relaciones de cobertura solamente indican si un conjunto de soluciones cubre a otro conjunto de soluciones, y no informa sobre la distancia entre ellos en términos de los valores de función objetivo. Por otro lado, los indicadores binarios solamente informan sobre la cercanía entre las soluciones de ambos conjuntos, en términos de los valores de función objetivo. Considerando que los casos con correlación positiva muestran frentes Pareto más pequeños resulta más difícil cubrirlos, incluso con un frente que se encuentre cercano en términos de los valores de las funciones objetivo.

La mejora obtenida al incrementar los valores de P y L redundan en mayores tiempos de ejecución, como se puede notar en la figura 11. El esfuerzo computacional crece aproximadamente 10 veces al pasar de $P = L = N$ a $P = L = 4N$ para los tres tamaños de casos. Los tiempos computacionales reportados para $W - RoTS$ son 88, 726 y 2550 segundos para los tamaños de casos 25, 50 y 75, respectivamente. Esto es, el algoritmo $MOGWW - P$ obtiene resultados comparables a los obtenidos por $W - RoTS$, en términos de cobertura e indicadores binarios para la mayoría de los casos utilizando un

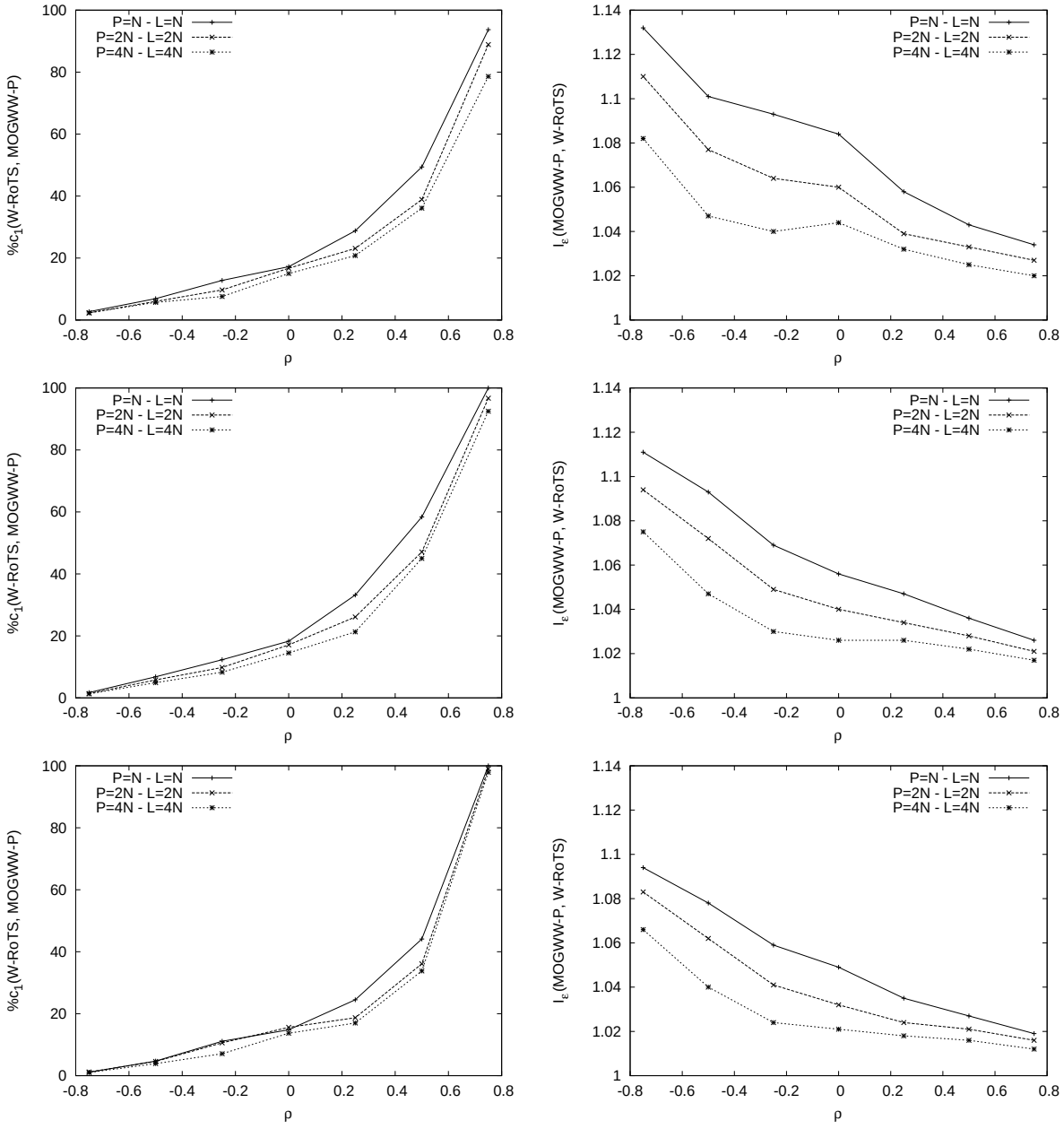


Figura 10. Porcentaje de cobertura de $W - RoTS$ sobre $MOGWW - P$ (izquierda) e Indicadores Binarios $I_\epsilon(MOGWW - P, W - RoTS)$ (derecha), casos de tamaño 25 (superior), 50 (intermedio), and 75 (inferior).

menor esfuerzo computacional. Más aun, el algoritmo $MOGWW - P$ puede mejorar sus resultados incrementando los valores de sus parámetros, tal como puede verse en la figura 10.

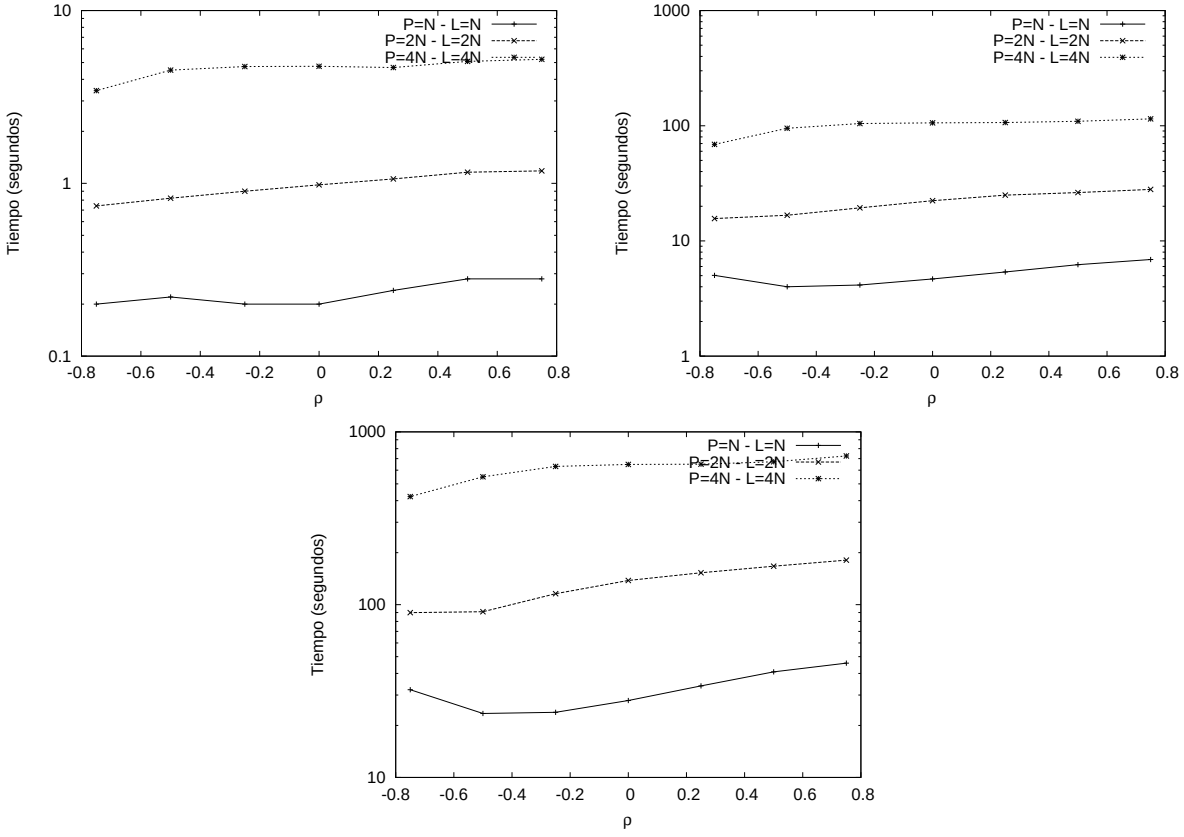


Figura 11. Tiempos promedio de ejecución para $MOGWW - P$, tamaños de casos 25, 50 y 75.

Después de analizar todos los resultados anteriores se puede entender mejor el comportamiento de las versiones híbridas del $MOGWW$. En el caso de $MOGWW - F$, y tomando en cuenta que se optó por forzar al algoritmo a aceptar un movimiendo hacia una nueva solución solamente si ésta es dominante (**Paso 1**, **Algoritmo 4**), podría detener su búsqueda después de pocos pasos, obteniendo apenas pequeñas mejoras. Podría ser más conveniente en algunos casos el permitir al algoritmo realizar movimientos a soluciones no-dominadas en espera de encontrar caminos alternos hacia mejores soluciones. En el caso de $MOGWW - P$ el problema parece derivarse de que la principal característica del algoritmo

original $MOGWW$ es el mantener a cada paso la diversidad en el conjunto de soluciones. Considerando esto, incluso en el caso de que se mejore cada solución podrían existir casos en los que las soluciones mejoradas se encuentren muy cercanas a las originales, quedando todas al final de nuevo en un frente no-dominado. Sería interesante verificar que ocurre si se aplica una búsqueda local solamente a algunas de las P partículas.

Capítulo VII

Conclusiones

VII.1 Sumario

En este trabajo se presenta un análisis experimental de tres medidas de dificultad usadas para caracterizar grafos de búsqueda sobre los que trabajan los algoritmos de búsqueda local. Para ello se abordan casos de problemas pertenecientes a la clase *NP-difícil*. Se diseña un conjunto de experimentos para estudiar la utilidad de esas medidas en la predicción del desempeño de un par de búsquedas locales. Finalmente, se propone la extensión del algoritmo GWW para problemas de optimización multi-objetivo y se realiza un análisis experimental de la influencia de agregar una búsqueda local como mecanismo de escape.

Las tres medidas de dificultad consideradas son: CL, BV y LE, y se aplican sobre casos de los problemas QAP, BSL y TSP. Para los dos primeros se analizan dos vecindarios diferentes mientras que para el TSP se analizan tres. El análisis se realiza a través de la verificación de cada una de las medidas de dificultad en los casos de prueba utilizando cada uno de los vecindarios propuestos. De esta forma se obtiene un listado de los casos y sus respectivas características que se utiliza como punto de partida en el análisis experimental de la capacidad de predicción de cada una de las medidas sobre el desempeño de las búsquedas locales. Esto lleva a la formulación de hipótesis específicas para cada problema de prueba, con miras a realizar una verificación experimental de las mismas.

Para verificar las predicciones (e hipótesis) derivadas del análisis de los grafos de búsqueda, se realizan experimentos sobre todos los casos, y sus respectivos vecindarios, con una búsqueda local voraz simple ($SGLS$) y con la conocida como búsqueda local iterativa (ILS). Esos resultados se utilizan para realizar comparaciones de desempeño entre

vecindarios y entre búsquedas locales. El resultado de tales comparaciones se utiliza para respaldar o refutar de forma experimental las hipótesis establecidas así como los trabajos previos realizados en este mismo sentido.

Para probar el algoritmo propuesto se utilizan casos de los problemas QAP y FSP en sus versiones multi-objetivo. Se aplica el algoritmo propuesto $MOGWW$ y se comparan sus resultados con los mejores conocidos a la fecha, para verificar su efectividad sobre los problemas de prueba. En el caso de las versiones híbridas se utiliza la búsqueda $SGLS$ y una adaptación de PLS como mecanismo de escape en el algoritmo $MOGWW$. Estas versiones se prueban experimentalmente sobre casos del QAP multi-objetivo y se comparan con los obtenidos por el $MOGWW$ original y con los mejores conocidos.

VII.2 Conclusiones

Analizando los resultados obtenidos a lo largo de este trabajo de investigación se llega a las siguientes conclusiones:

- Considerando que dos de las hipótesis derivadas de la medida CL no se cumplen: para los casos del BSL al utilizar $SGLS$ (**Hipótesis 1.1**) y para varios de los casos del TSP al utilizar tanto $SGLS$ como ILS (**Hipótesis 1.3**); se puede concluir que CL no es una medida de dificultad que permita la predicción del desempeño de las búsquedas locales para los conjuntos de casos analizados de los problemas BSL y TSP. Estos resultados nos muestran que las conjeturas hechas por Stadler y Schnabl (1992), Stadler (1995) y Bierwirth *et al.* (2004), sobre la mayor efectividad de las búsquedas locales sobre problemas que posean un mayor valor de CL, deben considerarse solamente dentro del ámbito de los problemas y casos específicos analizados en sus estudios.

- Dado que dos de las hipótesis relacionadas a la medida BV no se cumplen: para los casos del BSL (**Hipótesis 2.1**) y para los casos del TSP (**Hipótesis 2.3**); se

concluye que la medida de dificultad basada en la estructura de BV no permite realizar predicciones confiables del desempeño de las búsquedas locales que explotan esa estructura (ILS) para los conjuntos de casos analizados de los problemas BSL y TSP. Esta conclusión no está en contradicción con las dadas por Boese *et al.* (1994), Reeves (1999), Gutiérrez y Brizuela (2006) sobre la mejora en los resultados obtenidos al utilizar sobre casos que muestran la estructura de BV algoritmos de búsqueda local que la exploten, solamente nos indican que no es una condición necesaria para el éxito de esas búsquedas locales.

- Tomando en cuenta que dos de las hipótesis establecidas sobre la medida LE no se cumplen: para los casos del BSL (**Hipótesis 3.1**) y para los casos del TSP (**Hipótesis 3.3**); se concluye que los resultados obtenidos con las pruebas para verificar la existencia de la LE tampoco permiten la predicción del desempeño de las búsquedas locales para los conjuntos de casos analizados de los problemas BSL y TSP al comparar diferentes vecindarios. Esto no implica que la propiedad de LE no permita la predicción del buen desempeño de los algoritmos de búsqueda local, tal como lo especifica Carson (2001), debido a que la prueba experimental utilizada para analizar esa propiedad verifica una condición suficiente pero no necesaria para su existencia.

- La longitud de caminata aleatoria (L) necesaria para de-correlacionar dos soluciones puede ser utilizada para comparar el desempeño de las búsquedas locales sobre grafos de búsqueda que poseen la propiedad de LE, al menos para los casos analizados del TSP, a mayor valor de L se obtiene un mejor desempeño en las dos búsquedas locales analizadas $SGLS$ e ILS.

- Existe una relación directa entre los valores de CL y L, para valores mayores de CL se necesitan valores mayores de L para alcanzar la distribución estacionaria. Para BSL se obtuvo en ambos vecindarios el mismo valor de CL y de L, para QAP se

obtuvo un mayor valor tanto de CL como de L para el vecindario Swap, finalmente, para el TSP se obtuvo el valor mayor de CL y de L para Two-Change.

- El algoritmo propuesto para problemas de optimización multi-objetivo ($MOGWW$) es una alternativa competitiva ya que obtuvo resultados comparables a los mejores conocidos.

- La utilización de búsquedas locales como mecanismo de escape de los frentes no dominados en el algoritmo $MOGWW$ permite una mejora en el desempeño del mismo.

VII.3 Trabajo futuro

Tomando en cuenta las conclusiones obtenidas de este trabajo de investigación, así como aquellos aspectos que quedaron fuera de su alcance, se plantean las siguientes líneas de investigación como trabajo futuro:

- Proponer una medida cuantitativa que permita analizar la estructura de los óptimos locales en términos de su relación costo-distancia, tal que no necesite del conocimiento a priori del total de las soluciones o del total de óptimos locales o del óptimo global para su cálculo y, sin embargo, sea capaz de predecir el desempeño de los algoritmos de búsqueda local al utilizar diferentes vecindarios.

- Profundizar en el análisis del uso de la longitud de caminata aleatoria (L) como discriminante entre vecindarios que presenten la propiedad de LE. Utilizar un mayor número de problemas de prueba que muestren más de un vecindario que posean esa propiedad para verificar la relación entre el valor de L y el desempeño de las búsquedas locales sobre sus grafos de búsqueda.

- Ampliar el estudio del algoritmo $MOGWW$ como opción para tratar con problemas multi-objetivo utilizando un mayor conjunto de casos de prueba, así como realizando

pruebas para verificar cómo se ve afectado el desempeño del algoritmo al incrementar el número de objetivos a optimizar.

- Utilizar una mayor variedad de búsquedas locales como mecanismo de escape, en búsqueda de aquella que complemente de la mejor manera el comportamiento del algoritmo $MOGWW$ para la obtención de mejores resultados.

Bibliografía

Aarts, E. H. L. y J. K. Lenstra, editores 1997. "Local search in combinatorial optimization". Wiley, Chichester. 536 pp.

Ahuja, R., J. Orlin, y A. Tivari 2000. "A greedy genetic algorithm for the quadratic assignment problem". Comput. Oper. Res., 27(10):917-934 p.

Aldous, D. y U. Vazirani 1994. "Go with the winners algorithms". En: "35th IEEE Symposium on Foundations of Computer Science", Los Alamitos, CA. IEEE Computer Society Press. 492-501 p.

Angel, E. y V. Zissimopoulos 1998. "Autocorrelation coefficient for the graph bipartition problem". Theoretical Computer Science, 191:229-243 p.

Angel, E. y V. Zissimopoulos 2001. "On the landscape ruggednedd of the quadratic assignment problem". Theoretical Computer Science, 263:159-172 p.

Angel, E. y V. Zissimopoulos 2002. "On the hardness of the quadratic assignment problem with metaheuristics". J. Heuristics, 8(4):399-414 p.

Ausiello, G., P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, y M. Protasi 1999. "Complexity and approximation: Combinatorial optimization problems and their approximability properties". Springer-Verlag, Berlin. 524 pp.

Bagchi, T. P. 1999. "Multiobjective scheduling by genetic algorithms". Kluwer Academic Publishers. 376 pp.

Bierwirth, C., D. C. Mattfeld, y J. P. Watson 2004. "Landscape regularity and random walks for the job-shop scheduling problem". En: "EvoCOP 2004, LNCS 3004", Berlin. Springer-Verlag. 21-30 p.

Binh, T. T. y U. Korn 1996. "An evolution strategy for the multiobjective optimization". En: "Second International Conference on Genetic Algorithms (Mendel 96)". 23-28 p.

Boese, K. D., A. B. Kahng, y S. Muddu 1994. "A new adaptive multi-start technique for combinatorial global optimization". Operations Research Letters, 16:101-113 p.

Bollobas, B. 1985. "Random graphs". Academic Press. 500 pp.

Borges, P. C. y M. P. Hansen 2000. "A study of global convexity for a multiple objective traveling salesman problem". En: Ribeiro, C. C. y M. P. Hansen, editores, "Essays and Surveys in Metaheuristics". Kluwer, 129-150 p.

Brizuela, C. y R. Aceves 2003. "Experimental genetic operators analysis for the multi-objective permutation flowshop". En: Fonseca, C., P. Fleming, E. Zitzler, K. Deb, y L. Thiele, editores, "EMO03, LNCS 2632", Berlin. Springer-Verlag. 578-592 p.

Brizuela, C. y E. Gutiérrez 2003. "An experimental comparison of two different encoding schemes for the location of base stations in cellular networks". En: "EvoCOP 2003, LNCS 2611", Berlin. Springer-Verlag. 176-186 p.

Brizuela, C. y E. Gutiérrez 2005. "Multi-objective go with the winners algorithm: A preliminary study". En: Coello, C. A. C., A. Hernández-Aguirre, y E. Zitzler, editores, "EMO05 2005, LNCS 3410", Berlin. Springer-Verlag. 206-220 p.

Burkard, R. E., E. C. Cela, P. M. Pardalos, y L. S. Pitsoulis 1998. "The quadratic assignment problem". En: Pardalos, P. M. y D. Z. Du, editores, "Handbook of Combinatorial Optimization". Springer, 241–338 pp.

Calégari, P. R. 1999. "Parallelization of population-based evolutionary algorithms for combinatorial optimization problems". Tesis de Doctorado, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland.

Calégari, P. R., F. Guidec, P. Kuonen, y D. Kobler 1997. "Parallel island-based genetic algorithm for radio network design". Journal of Parallel and Distributed Computing (JPDC): Special Issue on Parallel Evolutionary Computing, 47:86-90 p.

Carson, T. 2001. "Empirical and analytic approaches to understanding local search heuristics". Tesis de Doctorado, University of California at San Diego, San Diego, CA. 152 pp.

Coello, C. A. C., G. B. Lamont, y D. A. Van-Veldhuizen, editores 2002. "Evolutionary algorithms for solving multi-objective problems". Kluwer, New York. 800 pp.

Deb, K., editor 2001. "Multi-objective optimization using evolutionary algorithms". Wiley, Chichester.

Deb, K., A. Pratap, S. Agarwal, y T. Meyarivan 2002. "A fast and elitist multiobjective genetic algorithm: Nsga-ii". IEEE Transactions on Evolutionary Computation, 6(2):182-197 p.

Dimitriou, T. y R. Impagliazzo 1996. "Towards an analysis of local optimization algorithms". En: "28th ACM Symposium on the Theory of Computing, STOC 1996", Pennsylvania. ACM. 304-313 p.

Dimitriou, T. y R. Impagliazzo 1998. "Go with the winners for graph bisection". En: "9th Annual SIAM Symposium on Discrete Algorithms, SODA 1998", San Francisco. SIAM. 510-520 p.

Dimitriou, T. y P. Spirakis 2004. "How to tell a good neighborhood from a bad one: Satisfiability of boolean formulas". En: "3rd International Workshop on Experimental Algorithms, WEA 2004", Berlin. Springer. 199-212 p.

Du, D. y P. M. Pardalos, editores 1999. "Handbook of combinatorial optimization". Kluwer Academic Publishers, London. 2410 pp.

Gambardella, L. M., E. Taillard, y G. Agazzi 1999. "Macs-vrptw: A multiple ant colony system for vehicle routing problems with time windows". En: Corne, D., M. Dorigo, y F. Glover, editores, "New Ideas in Optimization". McGraw Hill, 63-76 p.

Gandibleux, X., N. Mezdaoui, y A. Fréville 1997. "A tabu search procedure to solve combinatorial optimisation problems". En: Caballero, R., F. Ruiz, y R. E. Steuer, editores, "Advances in Multiple Objectives and Goal Programming". 291-300 p.

Garey, M. R. y D. S. Johnson 1979. "Computers and intractability: A guide to the theory of np-completeness". W. H. Freeman, San Francisco. 340 pp.

Gutiérrez, E. y C. A. Brizuela 2006. "Ils-perturbation based on local optima structure for the qap problem". En: Gelbukh, A. F. y C. A. R. García, editores, "MICAI 2006: Advances in Artificial Intelligence, 5th Mexican International Conference on Artificial Intelligence, Apizaco, Mexico, November 13-17", volume 4293 of Lecture Notes in Computer Science. Springer. 404-414 p.

Gutin, G. y A. P. Punnen 2002. "The traveling salesman problem and its variations". Kluwer Academic Publishers, London. 848 pp.

Hansen, M. P. 1997. "Tabu search in multiobjective optimisation: Mots". En: "Proceedings of the *13th International Conference on Multiple Criteria Decision Making (MCDM'97)*", Cape Town, South Africa.

Hansen, M. P. 2000. "Use of substitute scalarizing functions to guide a local search base heuristics: the case of moTSP". Journal of Heuristics, 6:419-431 p.

Herroelen, W. y A. V. Gils 1985. "On the use of flow dominance in complexity measures dor facility layout problems". International Journal of Production Research, 23(1):97-108 p.

Hoel, P. G., editor 1962. "Introduction to mathematical statistics". John Wiley & Sons, New York. 427 pp.

Hoos, H. H. 1998. "Stochastic local search: Methods, models, applications". Tesis de Doctorado, Technischen Universität Darmstadt, Darmstadt, Germany. 220 pp.

Hoos, H. H. y T. Stützle 2005. "Stochastic local search foundations and applications". Morgan Kaufmann, San Francisco, CA. 658 pp.

Johnson, D. S. y L. A. McGeoch 1997. "The travelling salesman problem: A case study in local optimization". En: Aarts, E. H. L. y J. K. Lenstra, editores, "Local Search in Combinatorial Optimization". Wiley, 215-310 p.

Johnson, D. S., C. H. Papadimitriou, y M. Yannakakis 1988. "How easy is local search?". Journal of Computer and System Science, 37:79-100 p.

Jones, T. 1995. "Evolutionary algorithms, fitness landscape and search". Tesis de Doctorado, The University of New Mexico, Albuquerque, New Mexico. 224 pp.

Jones, T. y S. Forrest 1995. "Fitness distance correlation as a mesure of problem difficulty for genetic algorithms". En: Eshelman, L., editor, "Fifth International Conference on Genetic Algorithms", San Francisco, CA. Morgan Kaufmann. 184-192 p.

Kauffman, S. A. 1993. "The origins of order; self-organization and selection in evolution". Oxford University Press, New York. 734 pp.

Kennedy, J. y R. C. Eberhart 1995. "Particle swarm optimization". En: "Proceedings of the 1995 *IEEE International Conference on Neural Networks* IEEE International Conference on Neural Networks", Perth, Australia. 1942-1948 p.

Kirkpatrick, S., C. D. Gelatt, y M. P. Vecchi 1983. "Optimization by simulated annealing". Science, 220(4598):671-680 p.

Knowles, J. D. y D. Corne 2003. "Instance generators and test suites for the multiobjective quadratic assignment problem". En: Fonseca, C., P. Fleming, E. Zitzler, K. Deb, y L. Thiele, editores, "Evolutionary Multi-criterion Optimization (EMO 2003), LNCS 2632", Faro, Portugal. Springer-Verlag. 295-310 p.

Krentel, M. W. 1989. "Structure in locally optimal solutions". En: "30th Annual Symposium on Foundations of Computer Science", Los Alamitos, CA. IEEE Computer Society Press. 216-222 p.

Krishnamachari, B. y S. B. Wicker 2000. "Experimental analysis of local search algorithms for optimal base station location". En: "Proc. International Conference on Evolutionary Computing for Computer, Communication, Control and Power", Chennai, India. IEEE Computer Society Press.

Lin, S. y B. Kernighan 1973. "An effective heuristic algorithm for the traveling salesman problem". Operations Research, 21:498-516 p.

Lourenco, H. R., O. Martin, y T. Stutzle 2002. "Iterated local search". En: Glover, F. y G. Kochenberger, editores, "Handbook of Metaheuristics". Kluwer, 321-353 p.

Manderick, B., M. D. Weger, y P. Spiessens 1991. "The genetic algorithm and the structure of the fitness landscape". En: Belew, R. K., editor, "Fourth International Conference on Genetic Algorithms", San Mateo, CA. Morgan Kaufmann. 143-150 p.

Mariano, C. E. y E. Morales 1999. "Moaq an ant-q algorithm for multiple objective optimization problems". En: Banzhaf, W., J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, y R. E. Smith, editores, "Genetic and Evolutionary Computing Conference (GECCO 99)". 894-901 p.

Mathar, R. y T. Niessen 2000. "Optimum positioning of base stations for cellular radio networks". Wireless Networks, 6:421-428 p.

Mattfeld, D. C., C. Bierwirth, y H. Kopfer 1999. "A search space analysis of the job shop scheduling". Annals of Operational Research, 86:441-453 p.

Merz, P. y B. Freisleben 2000. “Fitness landscape analysis and memetic algorithms for the quadratic assignment problem”. Evolutionary Computation, 4(4):337-352 p.

Papadimitriou, C. H. 1992. “The complexity of the lin-kernighan heuristic for the traveling salesman problem”. SIAM Journal on Computing, 21:450-465 p.

Papadimitriou, C. H., A. A. Schaffer, y M. Yannakakis 1990. “On the complexity of local search”. En: “STOC ’90: Proceedings of the twenty-second annual ACM symposium on Theory of computing”, Maryland, USA. 438-445 p.

Papadimitriou, C. H. y K. Steiglitz 1982. “Combinatorial optimization: Algorithms and complexity”. Prentice Hall, Englewood Cliffs, NJ. 496 pp.

Paquete, L., M. Chiarandini, y T. Stützle 2004. “Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study”. En: Gandibleux, X., M. Sevaux, K. Sörensen, y V. T’kindt, editores, “Metaheuristics for Multiobjective Optimisation, LNCS 535”, Berlin. Springer-Verlag. 177-200 p.

Paquete, L. y T. Stützle 2006. “A study of stochastic local search algorithms for the biobjective qap with correlated matrices”. European Journal of Operational Research, 169:943-959 p.

Reeves, C. R. 1999. “Landscapes, operators and heuristic search”. Annals of Operational Research, 86:473-490 p.

Reidys, C. M. y P. F. Stadler 2002. “Combinatorial landscapes”. SIAM Review, 44(1):3-54 p.

Schäffer, A. A. y M. Yannakakis 1991. “Simple local search problems that are hard to solve”. SIAM Journal on Computing, 20(1):56-87 p.

Serafini, P. 1994. “Simulated annealing for multiple objective optimization problems”. En: Tzeng, G., H. Wang, U. Wen, y P. Yu, editores, “Proceedings of the Tenth International Conference on Multiple Criteria Decision Making: Expand and Enrich the Domains of Thinking and Application”. 283-292 p.

Stadler, P. F. 1995. “Towards a theory of landscapes”. En: na, R. L. P., R. Capovilla, R. G. Pelayo, H. Waelbroeck, y F. Zertuche, editores, “Complex Systems and Binary Methods”. Springer Verlag, 77-163 p.

Stadler, P. F. y W. Schnabl 1992. “The landscape of the travelling salesman problem”. Phys. Lett., 161:337-344 p.

Stützle, T. 1999. “Iterated local search for the quadratic assignment problem”. Reporte Técnico AIDA-99-03, TU Darmstadt.

Taillard, E. D. 1995. “Comparison of iterative searches for the quadratic assignment problem”. Location Science, 3:87-105 p.

Tamaki, H. y E. A. Nishino 1998. "Genetic algorithm approach to multi-objective scheduling problems with regular and non-regular objective functions". En: "Proceedings of *IFAC LSS'98*". 289-294 p.

Thompson, M. 2001. "Application of multi objective evolutionary algorithms to analogue filter tuning". En: Zitzler, E., K. Deb, L. Thiele, C. A. C. Coello, y D. Corne, editores, "First International Conference on Evolutionary Multi-Criterion Optimization. LNCS No. 1993". Springer. 546-559 p.

Vaessens, R. J., E. H. L. Aarts, y J. K. Lenstra 1998. "A local search template". Computers Ops Res., 25(11):969-979 p.

Weinberger, E. 1990. "Correlated and uncorrelated fitness landscapes and how to tell the difference". Biological Cybernetics, 63:325-336 p.

Zitzler, E. y L. Thiele 1999. "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach". IEEE Transactions on Evolutionary Computation, 4(3):257-271 p.

Zitzler, E., L. Thiele, M. Laumanns, C. Fonseca, y V. Fonseca 2003. "Performance assessment of multiobjective optimizers: An analysis and review". IEEE Transactions on Evolutionary Computation, 7(2):117-132 p.

Apéndice A

Análisis experimental de la propiedad de expansión local

A.1 Prueba de de-correlación

Carson (2001) propone una prueba a la que llama "forgetting the starting place", y que en este trabajo llamamos "prueba de de-correlación", para verificar la segunda condición suficiente para verificar la existencia de la propiedad de expansión local en un grafo de búsqueda. Esta prueba, así como la forma de medirla experimentalmente, se describe en la sección V.1.3, presentándose solamente los resultados finales. En esta sección se muestran a detalle las pruebas experimentales realizadas para llegar a tales resultados para cada problema analizado.

A.1.1 Localización de Radio Bases (BSL)

La figura 12 muestra en la esquina superior izquierda la probabilidad de de-correlación para $P = 64$ y diferentes valores de L, utilizando el vecindario Binario para un caso del BSL. Se puede observar que utilizar un valor de $L = 2N$ genera una probabilidad cercana a 0.5. Esto significa que $L = 2N$ es un valor suficiente para que el algoritmo GWW de-correlacione las soluciones y muestree el grafo de búsqueda. Tomando en cuenta la **Hipótesis 1** y estos resultados, se puede observar que este caso posee la propiedad de expansión local en su grafo de búsqueda. La figura 12 muestra también en la esquina superior derecha el análisis del mismo caso utilizando el vecindario Entero. En este caso los valores de la probabilidad de de-correlación cambian significativamente a lo largo de las

iteraciones del algoritmo, terminando en un valor mayor a 0.5 cuando se utiliza $L = N$ y aproximadamente 0.3 cuando se utiliza $L = 2N$ y $L = 4N$. Estos resultados no permiten concluir si este vecindario posee la propiedad de expansión local. Los resultados presentan muchas iteraciones para las cuales el valor de la probabilidad de de-correlación estuvo muy alejado de 0.5, lo cual no permite concluir si en esas iteraciones se pudieron de-correlacionar las soluciones de acuerdo a la prueba. Estas variaciones en la probabilidad pueden deberse al tamaño variable del vecindario Entero. Finalmente, la parte inferior de la figura 12 muestra resultados equivalentes para otro caso del BSL utilizando los mismos vecindarios.

De acuerdo a Carson (2001), estos resultados dirían que el vecindario Binario del BSL genera grafos de búsqueda que deberían tratarse exitosamente con algoritmos de búsqueda local, debido a que poseen la propiedad de expansión local. Por otro lado, en el caso del vecindario Entero del mismo problema, no se puede concluir si los grafos de búsqueda que genera tienen o no la propiedad de expansión local, ya que la prueba realizada verifica una propiedad suficiente pero no necesaria para que eso se cumpla. Esto es, los resultados obtenidos no permiten saber si el vecindario Entero genera grafos de búsqueda con la propiedad de expansión local ni si algoritmos de búsqueda local se desempeñarían bien en esos casos debido a esta u otra propiedad de sus grafos de búsqueda.

A.1.2 Problema del Agente Viajero (TSP)

La figura 13 muestra en la parte superior la probabilidad de de-correlación para $P = 64$ y diferentes valores de L utilizando el vecindario Two-Change para dos casos del TSP. Los resultados muestran que el valor de 0.5 se alcanza al utilizar $L = 2N$ para el caso ft70, mientras que ese valor no es suficiente para el caso kroA100. La figura 13 muestra también la probabilidad de de-correlación para los casos C1c.0 y amat.0 utilizando el mismo vecindario. Se puede observar en las gráficas que la probabilidad es de casi 1 para $L = N$, aproximadamente 0.58 para $L = 2N$ y alrededor de 0.38 para $L = 4N$. Estos resultados indican que, incluso en los casos para los que se encontró un valor de L tal que

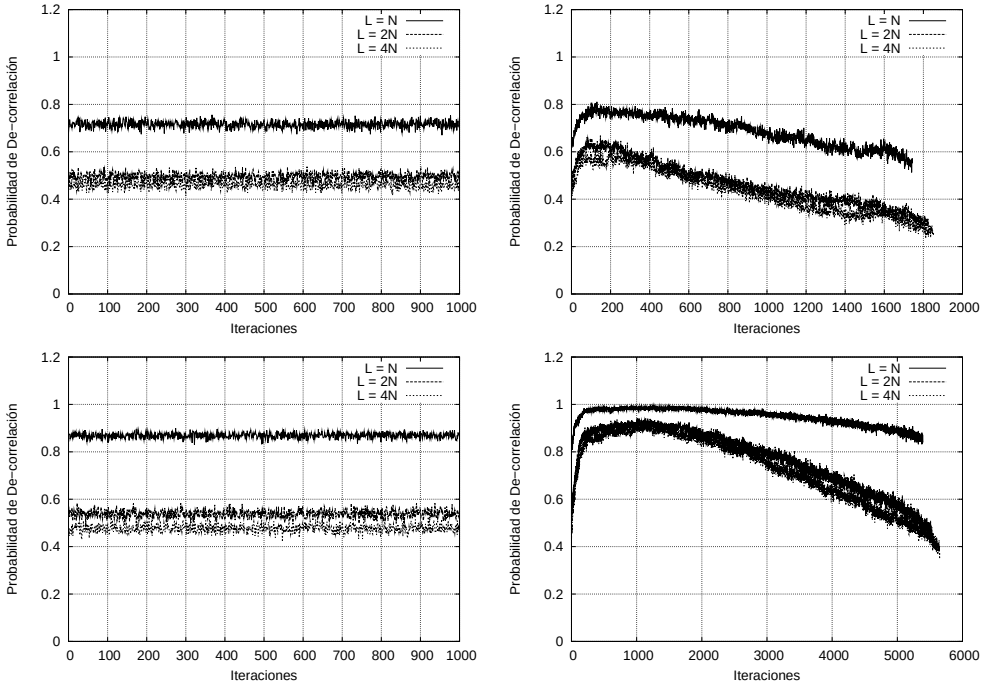


Figura 12. Probabilidad de de-correlación para los casos del BSL. De izquierda a derecha y de arriba a abajo: bhaskar51 (Binaria y Entera); cale149 (Binaria y Entera).

la probabilidad de de-correlación es aproximadamente 0.5, ese valor sigue decrementándose al utilizar valores mayores de L. Esto implica que no se ha alcanzado el punto estacionario y que existe un sesgo experimental que lleva a ese comportamiento. Esto último puede deberse al número finito, y relativamente pequeño, de muestras, lo que lleva a obtener probabilidades de de-correlación de 0.5 con valores de L que no son suficientes para alcanzar el punto estacionario. Este problema se analiza en la sección A.2. Sin embargo, se sabe que es posible de-correlacionar suficientemente la solución final de tal forma que escape de la inicial utilizando un valor de L tal que la probabilidad de de-correlación se mantenga menor a 0.5. Para sobrepasar las limitaciones del experimento, descritas anteriormente, se necesita encontrar un valor de L suficientemente grande para alcanzar el punto estacionario. Todos los casos del TSP, simétricos y asimétricos, muestran aproximadamente los mismos valores de de-correlación. Más aún, para el vecindario Swap también se encontró un valor de L que permite concluir que los grafos de búsqueda de los casos analizados tienen la propiedad de

expansión local. En el caso del vecinario Insert, se necesita un estudio más a fondo ya que no se encontró un valor adecuado de L, esta situación se discute en la sección A.2.

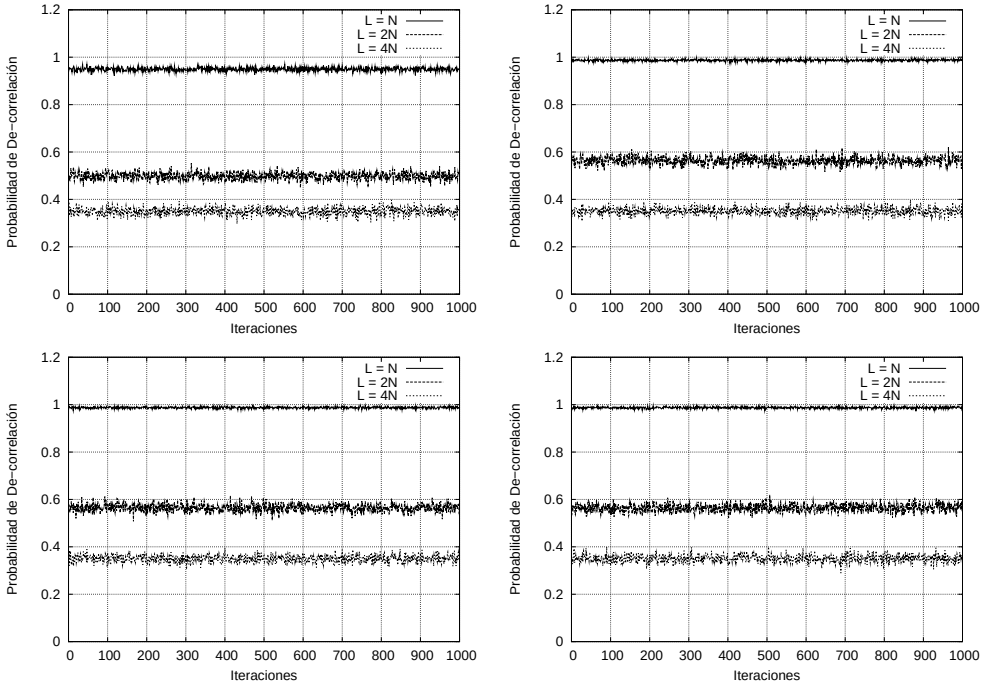


Figura 13. Probabilidad de de-correlación para casos del TSP utilizando el vecindario Two-Change. De izquierda a derecha y de arriba a abajo: ft70, kroA100, C1c.0 y amat.0.

A.1.3 Problema de Asignación Cuadrática QAP

La figura 14 muestra en la parte superior izquieda la probabilidad de de-correlación para el caso tai30b del QAP utilizando el vecindario Swap y diferentes longitudes en la caminata aleatoria. La probabilidad obtenida es cercana a 1 para $L = N$, aproximadamente 0.5 para $L = 2N$ y cercana a 0.4 para $L = 4N$. La figura 14 muestra también resultados similares para los casos sko42, chr25a y tho30. Estos casos muestran una probabilidad de de-correlación cercana a 0.5 utilizando $L = 2N$, sin embargo, ese valor continua decrementándose al incrementar el valor de L. Este problema, descrito en los resultados del TSP, se analiza en la sección A.2. El resto de los casos del QAP muestran aproximadamente

los mismos valores de de-correlación. En el caso del vecindario Insert no se encontró un valor de L suficientemente grande para pasar la prueba de de-correlación. Esto es, después de incrementar repetidamente el valor de L, la probabilidad de de-correlación nunca cruza el umbral de 0.5. De acuerdo a estos resultados, se puede concluir que el vecindario Swap del QAP genera grafos de búsqueda con la porpiedad de expansión local, lo cual no se cumple para el vecindario Insert del mismo problema.

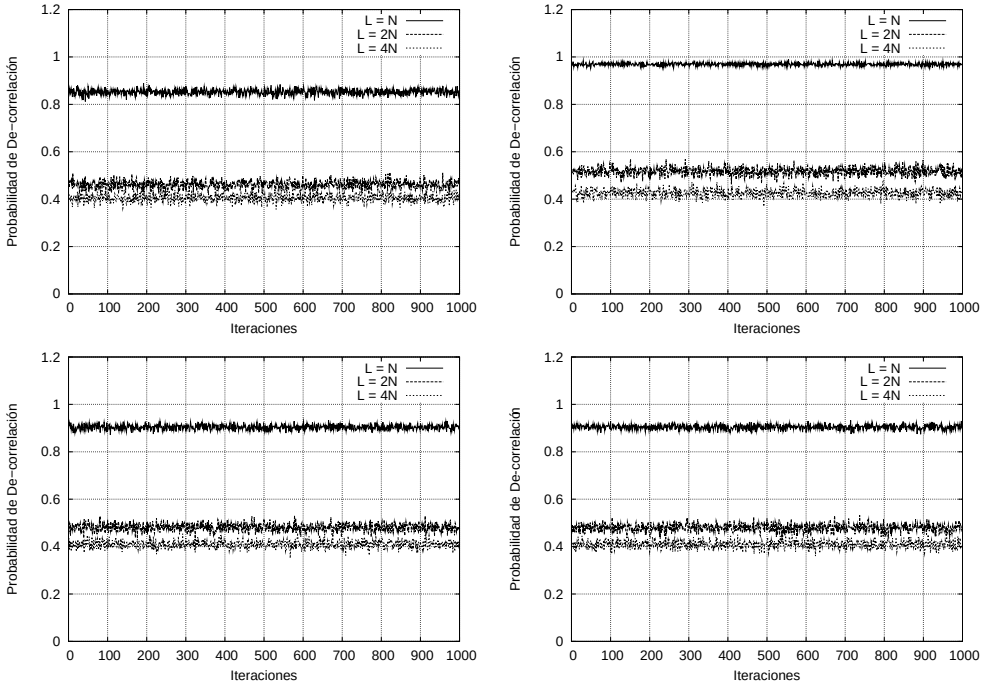


Figura 14. Probabilidad de de-correlación para casos del QAP utilizando vecindario Swap. De izquierda a derecha y de arriba a abajo: tai30b, sko42, chr25a, and tho30.

Los resultados muestran para la mayoría de las combinaciones problema/vecindario/caso valores de L suficientes para pasar la prueba de de-correlación, lo cual permite concluir que esos grafos de búsqueda tienen la propiedad de expansión local. Sin embargo, para algunos casos, a medida que se incrementaba el valor de L, la probabilidad de de-correlación sigue decrementándose. En la siguiente sección se verifican los valores dados por la prueba de de-correlación y se busca el punto estacionario en el cual, sin importar que se siga incrementando el valor de L, la probabilidad de de-correlación se mantiene sin cambio.

A.2 Verificación de la prueba de de-correlación

Para verificar los valores de L encontrados para los casos de prueba, se analiza la distribución de las distancias entre los puntos inicial y final de la caminata aleatoria. Se generaron $100,000$ soluciones aleatorias y para cada una de ellas se aplicaron caminatas aleatorias de diferentes longitudes. Utilizando estos resultados, se compararon las distribuciones de las distancias graficando el histograma de las distancias encontradas entre los puntos inicial y final de las caminatas aleatorias realizadas. Se necesita encontrar la longitud de caminata aleatoria que sea lo suficientemente grande para alcanzar la distribución estacionaria. En un escenario ideal, este valor debiese ser el mismo valor de L dado como resultado de la prueba de de-correlación. Sin embargo, debido a las limitaciones experimentales ya discutidas en la sección anterior, estos valores podrían no ser los mismos. En caso de encontrar que los valores de L dados no son suficientemente grandes, se necesitaría ajustar ese valor, incrementándolo hasta encontrar uno que alcance la distribución estacionaria, lo cual aseguraría la de-correlación de las soluciones al ser utilizado como longitud de la caminata aleatoria.

A.2.1 Casos del BSL

La figura 15 muestra los valores medios de la probabilidad de de-correlación para los casos del BSL al utilizar el vecindario Binario con diferentes longitudes de caminata aleatoria. Se puede observar que el valor de $L = 4N$ es suficiente para alcanzar el punto estacionario. La figura 15 también muestra los valores de probabilidad de de-correlación al utilizar el vecindario Entero. Para este último vecindario el punto estacionario se alcanza cuando aun se tiene una probabilidad de de-correlación mayor a 0.5, además de poseer una desviación estándar muy grande. Esto querría decir que una caminata de longitud L no asegura la de-correlación de la solución final con respecto a la inicial. Sin embargo, debido a la variabilidad mostrada en la figura 16, se necesita realizar un análisis más profundo para

llegar a una conclusión sobre este caso.

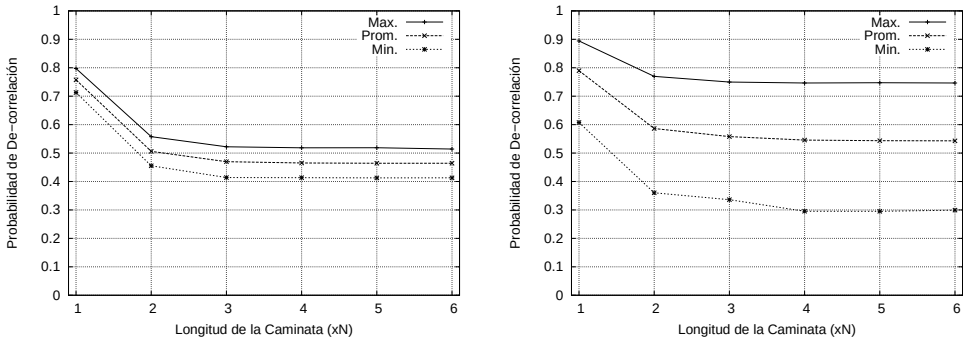


Figura 15. Probabilidad promedio de de-correlation con diferentes valores de L para casos del BSL: Binario (izquierdo) Entero (derecho).

La figura 16 muestra la distribución de las distancias obtenida para los casos del BSL utilizando los dos vecindarios analizados. La prueba de de-correlación para estos casos, mostrada en la sección A.1.1, indicaba que un valor de $L = 2N$ es suficiente para de-correlacionar las soluciones final e inicial. Considerando ese resultado, las distribuciones de las distancias se calcularon utilizando $L = 2N, 4N$ y $8N$. Se puede notar claramente un cambio en la distribución al pasar de $L = 2N$ a $L = 4N$, mientras que se mantiene prácticamente igual de $L = 4N$ a $L = 8N$. Esto indica que aunque la prueba de de-correlación indicaba que el valor de $L = 2N$ daba como resultado una probabilidad de de-correlación cercana a 0.5, la distribución estacionaria se alcanza al utilizar un valor de $L = 4N$. La tabla XII confirma este resultado al mostrar que las distribuciones continúan cambiando hasta que L toma el valor de $4N$. Se puede notar que la correlación entre la distribución de las distancias al utilizar $L = 4N$ y $L = 8N$ es casi 1, esto es, son practicamente la misma distribución.

Las distribuciones de las distancias obtenidas para los vecindarios Binario y Entero muestran diferentes formas, tal como se muestra en la figura 16. Para el vecindario Entero, se tiene una forma tipo Gaussiana, mientras que para el vecindario Binario se tiene una forma en la cual se muestran valores solo para las distancias pares. Esto es, las distribuciones

del vecindario Binario tienen también forma tipo Gaussiana considerando solamente las distancias pares, mientras que las distancias impares tienen frecuencia de cero. Esto último se debe a que se está utilizando la distancia de Hamming entre las soluciones final e inicial después de L cambios de bit (flips), lo cual puede establecerse de la siguiente forma:

Proposición 1. Dadas dos cadenas de bits del mismo tamaño s_0 y s_L, si s_L resulta de aplicar L cambios de bit (flip) a s_0, entonces, la distancia de Hamming entre s_0 y s_L, $d(s_0, s_L)$, es impar si L es impar, y es par si L es par.

Debido a que se están utilizando solamente valores pares de L como longitud de la caminata aleatoria, las distancias entre las soluciones inicial y final son siempre pares. Eso explica la forma de la distribución de las distancias mostradas en la figura 16 para el vecindario Binario.

A.2.2 Casos del TSP

La figura 17 muestra los valores promedio de de-correlación para los casos del TSP utilizando los tres vecindarios analizados. Se puede observar que para los vecindarios Insert y Swap un valor de $L = 2N$ parece suficiente para alcanzar el punto estacionario, mientras que para el vecindario Two-Change se necesita un valor de $L = 4N$.

La figura 18 muestra la distribución de las distancias obtenidas para los casos del TSP utilizando los tres vecindarios analizados. La prueba de de-correlación para estos casos muestra en la sección A.1.2 que utilizar valores de $L = N$ (para Insert y Swap) y $L = 2N$ (para Two-Change) eran suficientes para de-correlacionar las soluciones inicial y final. Sin embargo, la figura 17 muestra que, para los casos analizados, se alcanza el punto estacionario al utilizar un valor de $L = 2N$ para Insert y Swap, mientras que para Two-Change se necesita un valor de $L = 4N$. Considerando estos resultados, se calculó la distribución de

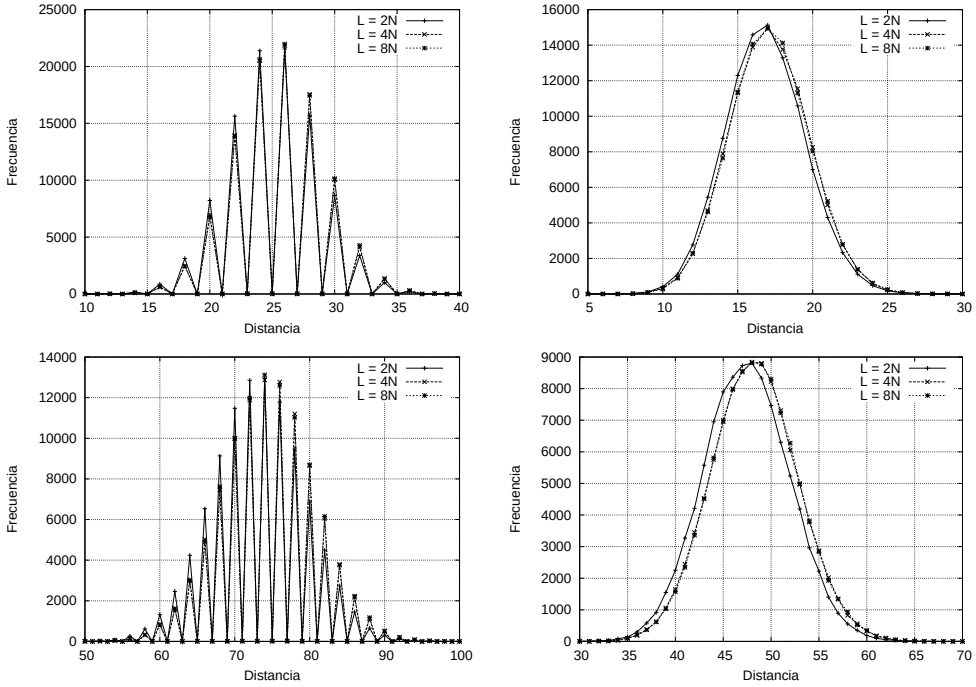


Figura 16. Distribución de las distancias entre los puntos inicial y final de una caminata aleatoria para los casos del BSL. De izquierda a derecha y de arriba a abajo: bhaskar51 (Binaria y Entera); cale149 (Binaria y Entera).

las distancias utilizando $L = 2N, 4N$ and $8N$. Se puede ver que la distribución cambió al pasar de $L = 2N$ a $L = 4N$ para Two-Change, mientras que para Insert y Swap permaneció casi igual. Para todos los vecindarios, la distribución parece mantenerse sin cambio al pasar de $L = 4N$ a $L = 8N$. Esto significa que, por ejemplo, en el caso del vecindario Two-Change, a pesar de que un valor de $L = 2N$ da como resultado una probabilidad de de-correlación cercana a 0.5, la distribución estacionaria se alcanza hasta que se utiliza un valor de $L = 4N$. Por lo tanto, ese es el valor de L necesario para de-correlacionar las soluciones inicial y final utilizando una caminata aleatoria.

A.2.3 Casos del QAP

La figura 19 muestra los valores promedio de probabilidad de de-correlación para los casos del QAP utilizando los dos vecindarios analizados. Se puede observar que un valor de

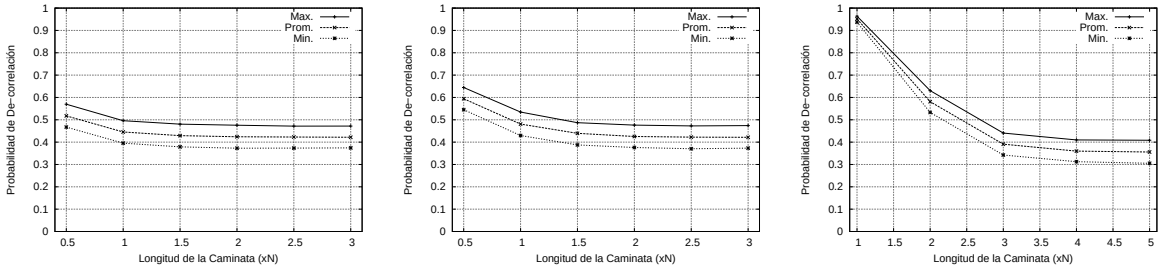


Figura 17. Probabilidad promedio de de-correlation con diferentes valores de L para casos del TSP. Insert (izquierda), Swap (centro) y Two-Change (derecha).

$L = 2N$ no fue suficiente para alcanzar el punto estacionario para el vecindario Insert. Sin embargo, el promedio de la probabilidad de de-correlación fue mayor a 0.5. Esto permite concluir lo mismo que se concluye para el vecindario Entero del BSL, lo que sifnifica que no se puede garantizar que los grafos de búsqueda analizados tienen la propiedad de expansión local, incluso considerando que este vecindario no tiene una desviación tan grande como el Entero. En el caso del vecindario Swap, un valor de $L = 4N$ es suficiente para alcanzar el punto estacionario, como puede verse en la figura 19. Este se encuentra en una probabilidad de de-correlación cercana a 0.4, lo cual permite concluir que estos grafos de búsqueda tienen la propiedad de expansión local.

La figura 20 muestra los resultados obtenidos para los casos del QAP utilizando los dos vecindarios analizados. La prueba de de-correlación para estos casos, mostrada en la sección A.1.1, muestra que valores de $L = N$ para el vecindario Insert y de $L = 2N$ para el vecindario Swap, son suficientes para de-correlacionar las soluciones inicial y final. Por lo tanto, la distribución de las distancias se calcularon utilizando $L = N, 2N$ y $4N$ para Insert, y $L = 2N, 4N$ y $8N$ para Swap. Se puede notar que, para el vecindario Insert, la distribución de las distancias cambia al pasar de $L = N$ a $L = 2N$, y parece permanecer igual al pasar de $L = 2N$ a $L = 4N$. Sin embargo, como ya se ha visto para este vecindario, el punto estacionario se alcanza con un valor de probabilidad de de-correlación mayor a 0.5. Lo cual quiere decir que no importa cuan grande sea la longitud de la caminata aleatoria,

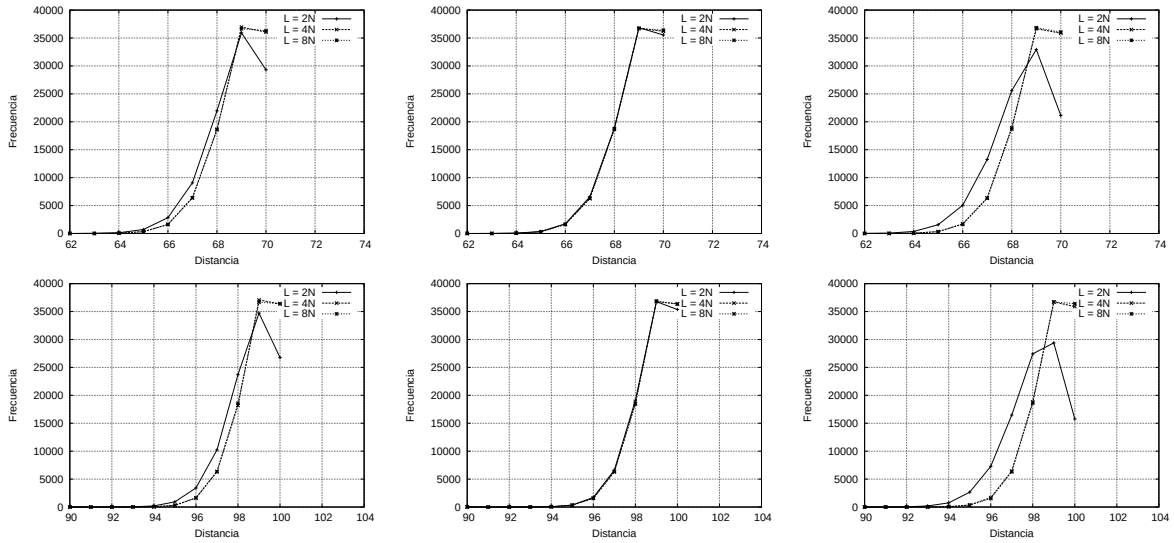


Figura 18. Distribución de las distancias entre los puntos inicial y final de una caminata aleatoria para casos del TSP utilizando tres vecindarios diferentes. De izquierda a derecha y de arriba a abajo: ft70 (Insert, Swap, Two-Change); kroA100 (Insert, Swap, Two-Change).

ésta no es capaz de de-correlacionar las soluciones. Por lo tanto, si se toma un óptimo local y se le aplica una caminata aleatoria, la solución resultante sigue estando correlacionada al óptimo local original, lo cual podría llevar a caer de nueva cuenta en su base de atracción. Debido a esto, no se puede concluir que estos grafos de búsqueda tienen la propiedad de expansión local. Por otro lado, para el vecindario Swap la distribución de las distancias cambia al pasar de $L = 2N$ a $L = 4N$, y permanece igual al pasar de $L = 4N$ a $L = 8N$. Esto significa que aunque un valor de $L = 2N$ da una probabilidad de de-correlación cercana a 0.5, la distribución estacionaria se alcanza hasta que se utiliza un valor de $L = 4N$. Por lo que, considerando estos resultados, los grafos de búsqueda generados por este vecindario tienen la propiedad de expansión local.

La distribución de las distancias obtenidas para el vecindario Swap tienen una forma con valores grandes para las distancias pares y valores cercanos a cero para las distancias impares. Esto se debe a que se están midiendo distancias entre las soluciones inicial y final después de L cambios Swap, lo cual puede establecerce como sigue:

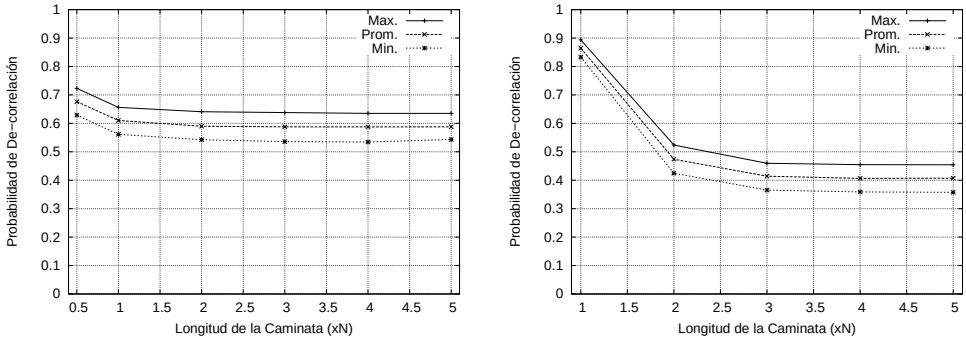


Figura 19. Probabilidad promedio de de-correlación a diferentes valores de L para los casos del QAP: vecindario Insert (izquierda) y vecindario Swap (derecha).

Proposición 2. Dadas dos cadenas de permutaciones de N valores diferentes s_0 y s_L, si s_L resulta de aplicar L cambios Swap a s_0, entonces, la distancia Swap entre s_0 y s_L, $d(s_0, s_L)$, es impar si L es impar, y es par si L es par.

Debido a que se están aplicando solamente valores pares como longitudes de la caminata aleatoria, las distancias entre las soluciones inicial y final serán pares con alta probabilidad. Esto explicaría la forma de la distribución de las distancias mostrada en la figura 20 para el vecindario Swap.

La tabla XII muestra la correlación de Pearson (Hoel, 1962) entre pares de distribuciones de distancias utilizando diferentes longitudes de caminata aleatoria. Se puede notar que los valores de correlación se acercan a 1 a medida que incrementamos el valor de L, lo que quiere decir que se acerca a la distribución estacionaria. Para todos los casos, una longitud de caminata aleatoria de N parece estar lejana a la distribución estacionaria ya que está pobremente correlacionada con la distribución de las distancias dada por $2N$. Por otro lado, al pasar de $2N$ a $4N$ se muestra una buena correlación entre las distribuciones para los problemas BSL y QAP. Finalmente, para todos los casos analizados, una longitud de caminata aleatoria de $4N$ parece adecuada ya que muestra una correlación cercana a 1 con la distribución de las distancias al utilizar $8N$.

Tabla XII. Correlación entre distribuciones de las distancias utilizando diferentes longitudes de caminata aleatoria.

Problema	**Caso**	**Vecindario**	**Correlación**		
			N - $2N$	$2N$ - $4N$	$4N$ - $8N$
BSL	bhaskar51	Binary	-0.06465	0.99599	0.99996
		Integer	0.86405	0.99662	0.99980
	cale149	Binary	-0.03623	0.98747	0.99989
		Integer	0.65498	0.99106	0.99990
TSP	ft70	Insert	0.24553	0.98960	0.99997
		Swap	0.82402	0.99991	0.99998
		Two-Change	0.24121	0.94235	0.99997
	kroA100	Insert	0.08649	0.97688	0.99994
		Swap	0.68435	0.99978	0.99999
		Two-Change	0.08402	0.87916	0.99994
	C1c.0	Insert	0.08562	0.97795	0.99998
		Swap	0.68542	0.99981	0.99999
		Two-Change	0.08553	0.88485	0.99985
	amat.0	Insert	0.08635	0.97888	0.99997
		Swap	0.68834	0.99973	0.99998
		Two-Change	0.08583	0.88157	0.99994
QAP	tai30b	Insert	0.99826	0.99990	0.99979
		Swap	0.38970	0.98470	0.99999
	sko42	Insert	0.99838	0.99994	0.99993
		Swap	0.17850	0.96958	0.99999
	chr25a	Insert	0.99820	0.99995	0.99997
		Swap	-0.06302	0.98913	0.99997
	tho30	Insert	0.99852	0.99992	0.99995
		Swap	0.39010	0.98530	0.99998

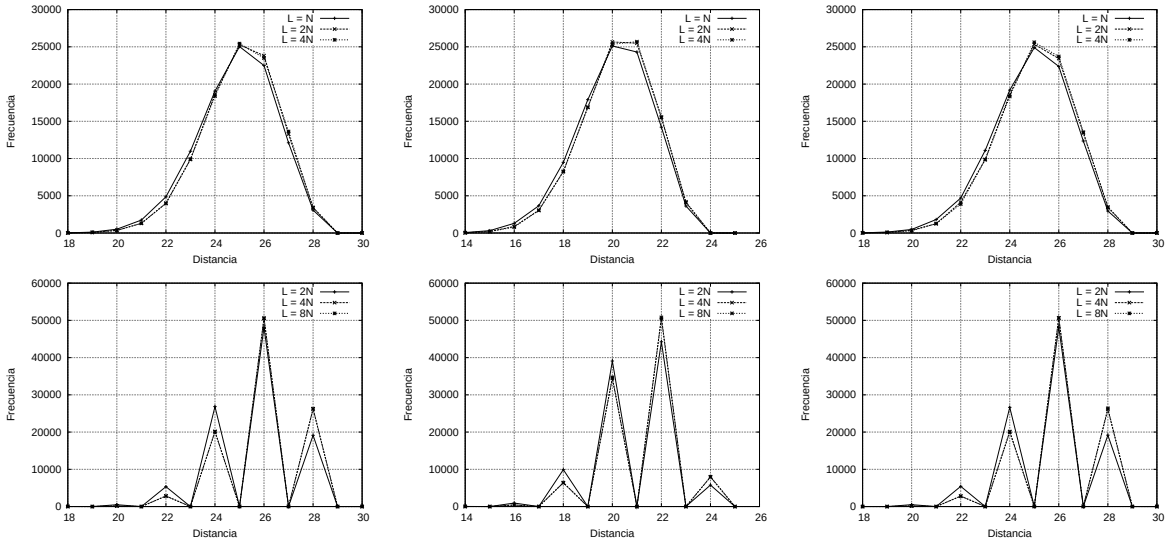


Figura 20. Distribución de las distancias entre los puntos inicial y final de una caminata aleatoria para los casos del QAP utilizando dos vecindarios diferentes. De izquierda a derecha y de arriba a abajo: Insert (tai30b, chr25a, tho30); Swap (tai30b, chr25a, tho30).