INSTITUTO TECNOLÓGICO DE CIUDAD MADERO

División de Estudios de Posgrado e Investigación

Opción I

Tesis Profesional

Tema:

"Estudio del Balance entre Intensificación y Diversificación en la Búsqueda Dispersa Aplicada al Problema de la Triangulación de Insumos"

Que para obtener el grado de:

Maestro en Ciencias de la Computación

Presenta:

Cindy Guadalupe Hernández Morales

G13073002

Directora:

Dra. Laura Cruz Reyes

Codirectora:

Dra. Claudia G. Gómez Santillán

Cd. Madero, Tamaulipas Mayo, 2014

"2014, Año de Octavio Paz"

Cd. Madero, Tamps; a 08 de Mayo de 2014.

OFICIO No.: U5.295/14
AREA: DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN
ASUNTO: AUTORIZACIÓN DE IMPRESIÓN DE TESIS

ING. CINDY GUADALUPE HERNÁNDEZ MORALES
NO. DE CONTROL G13073002
P R E S E N T E

Me es grato comunicarle que después de la revisión realizada por el Jurado designado para su examen de grado de Maestría en Ciencias de la Computación, el cual está integrado por los siguientes catedráticos:

PRESIDENTE :	DR. HÉCTOR JOAQUÍN FRAIRE HUACUJA
SECRETARIO :	DRA. GUADALUPE CASTILLA VALDÉZ
VOCAL :	DRA. LAURA CRUZ REYES
SUPLENTE	DRA. CLAUDIA GUADALUPE GÓMEZ SANTILLÁN
DIRECTORA DE TESIS :	DRA. LAURA CRUZ REYES
CO-DIRECTORA:	DRA. CLAUDIA GUADALUPE GÓMEZ SANTILLÁN

Se acordó autorizar la impresión de su tesis titulada:

"ESTUDIO DEL BALANCE ENTRE INTENSIFICACIÓN Y DIVERSIFICACIÓN EN LA BÚSQUEDA DISPERSA APLICADA AL PROBLEMA DE LA TRIANGULACIÓN DE INSUMOS"

Es muy satisfactorio para la División de Estudios de Posgrado e Investigación compartir con Usted el logro de esta meta.

Espero que continúe con éxito su desarrollo profesional y dedique su experiencia e inteligencia en beneficio de México.

A T E N T A M E N T E
"Por mi patria y por mi bien"®

M. P. MARIA YOLANDA CHÁVEZ CINCO
JEFA DE LA DIVISIÓN

ESTADOS UNIDOS MEXICANOS
S.E.P.
DIVISION DE ESTUDIOS
DE POSGRADO E
INVESTIGACION
I T C M

c.c.p.- Minuta
Archivo

MYCHC 'NLCO 'jar

Dedico esta tesis a mis padres.

Agradecimientos

Agradezco al Instituto Tecnológico de Cd. Madero por su apoyo incondicional; en especial a las doctoras Laura Cruz y Guadalupe Castilla que con paciencia amor maternal contribulleron en mi formación científica.

Índice General

Índice de Figuras

Índice de Tablas

Índice de Algoritmos

Publicaciones

Estudio del Balance de Intensificación-Diversificación en la Búsqueda Dispersa Aplicada al Problema de Ordenamiento Lineal

Resumen

En este trabajo se realizá un estudiado del balance de la intensidicación y diversificación en una búsqueda dispersa en el contexto del problema de la triangulación de tablas de $entrada$-$salida$. Este problema surge en el contexto de la economía, donde el modelo de Leontief representa las interacciones entre los sectores económicos mediante una tabla cuyas entradas representan la cantidad monetaria que determinado sector de la economía entrega a los diferentes sectores. En las ciencias computacionales este problema es equivalente al problema de ordenamiento lineal (LOP) que ha sido estudiado extensamente debido a las diversas e importantes aplicaciones que tiene en campos como la electrónica, las ciencias sociales ó las telecomunicaciones. Se ha demostrado que LOP pertenece a la clase de problemas NP-duros, que no son viables de resolver mediante métodos exactos y donde los métodos aproximados representan la elección a seguir para obtener una solución eficiente.

El desempeño de los enfoques metaheurísticos está fuertemente vinculado con encontrar un balance adecuado entre diversas estrategias que aprovechan la experiencia acumulada de búsqueda (intensificación) y de exploración del espacio de búsqueda (diversificación) para el método de solución. Existen estudios realizados del balance entre intensificación y diversificación en algunas metaheurísticas, pero hasta el momento no hay una metodología para la búsqueda dispersa que permita obtener un buen desempeño.

La principal aportación de este trabajo es el diseño de una metodología para representar el balance de intensificación y diversificación en la metaheurística de búsqueda dispersa. Adicionalmente se aportan cinco indicadores para representar intensificación y diversificación en la búsqueda dispersa, así como un conjunto de estrategias de construcción y actualización del conjunto de referencia que proveen diferentes niveles de diversificación en dicho conjunto.

Para evaluar la metodología propuesta se realizó un estudio experimental utilizando un conjunto de 30 instancias estándar. Se calculó la suma de los indicadores de intensificación por un lado y la suma de los indicadores de diversificación. Los resultados muestran que la estrategia que presentó el mejor balance entre la suma de indicadores de intensificación y

la suma de indicadores de diversificación es la estrategia de construcción del conjunto de referencia que reporta el mejor desempeño de la metaheurística de búsqueda dispersa para la solución LOP.

1

Introducción

En este capítulo se describen conceptos fundamentales para introducir los problemas de optimización combinatoria y una breve introducción sobre los métodos de solución comúnmente utilizados para este tipo de problemas, con especial énfasis en los métodos heurísticos. Posteriormente se plantea de manera formal el problema de Ordenamiento Lineal, la motivación que impulsó el desarrollo de esta tesis y los objetivos establecidos para el proyecto de investigación. Finalmente se muestra la estructura de la tesis con una breve descripción del contenido de cada capítulo.

1.1 Marco conceptual

Esta sección inicia con definiciones relevantes en el contexto de problemas de optimización combinatoria, posteriormente se describe el problema de la triangulación que surge en el contexto de la economía el cual es equivalente al problema de ordenamiento lineal. La sección finaliza con la descripción de algunos algoritmos de solución aproximada considerados para abordar el problema de estudio.

1.1.1 Definiciones relevantes

El conjunto de definiciones que se presenta son necesarios para introducir el problema de optimización combinatoria de Ordenamiento Lineal.

Problemas de optimización.- En las ciencias matemáticas, un problema de optimización es equivalente a la solución de la siguiente expresión:

$$f(\tau); \tau \in S \subseteq \mathbb{R}^n, \tag{1.1}$$

donde las variables de desición, $\tau = (\tau_1, \tau_2, ..., \tau_n)$ son los componentes de un vector τ cuya dimensionalidad depende del problema particular a optimizar, y pueden ser directamente relacionadas con el modelo que es utilizado para formular el problema [1].

La función objetivo $f(\tau)$ a optimizar (ya sea maximizar o minimizar), es la expresión que se utiliza para medir cuantitativamente la calidad de las decisiones; consiste de un valor obtenido mediante la aplicación de los criterios de selección en cualquier expresión matemática que modela el comportamiento de las variables del problema. El espacio de solución S, se expresa como la solución de un sistema de restriciones [27].

Las técnicas para resolver tales problemas son casi siempre de naturaleza iterativa y su convergencia es estudiada usando la matemática de análisis real [29].

Los problemas de optimización pueden ser divididos en dos categorías en función de la naturaleza de los valores de codificación (discretos o continuos). Optimización continua es la categoría que se ocupa de los problemas cuya solución se codifica utilizando valores reales, por otro lado, los problemas de optimización discreta incluyen aquellos cuyas soluciones son codificadas por los valores que pertenecen al conjunto de los enteros. En la segunda categoría hay una clase de problemas de gran interés dentro de la comunidad científica, los llamados problemas de optimización combinatoria. Estos problemas consisten en la búsqueda de un objetivo dentro de un conjunto finito o infinito numerable de posibilidades [29].

Optimización combinatoria.- Los problemas más comunes que surgen en diferentes campos de la ciencia e ingeniería, consisten en problemas que requieren la búsqueda de la

mejor configuración de un conjunto de parámetros del problema y están sujetos a un conjunto de restricciones con el fin de alcanzar un objetivo ya sea de minimización o maximización; los cuales pueden ser exactos y aproximados. Muchos de estos problemas aún no han sido resueltos satisfactoriamente, por lo que representa un reto para los investigadores el diseñar técnicas de solución eficientes y novedosas.

Un problema de optimización combinatoria $P = (S, f)$ puede ser definido como sigue:

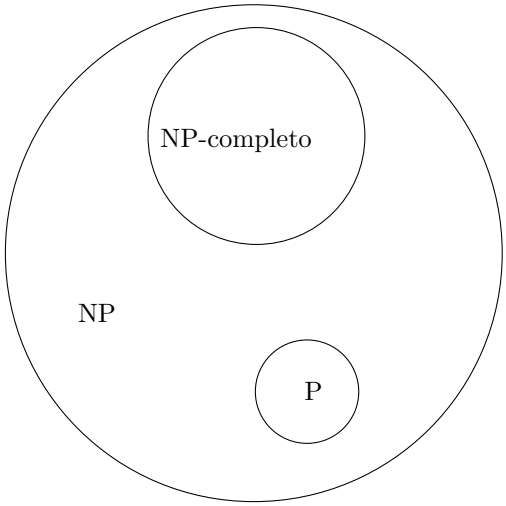
- Un conjunto de variables $\tau_1, \tau_2, ..., \tau_n$;
- Dominio de variables $D_1, D_2, ..., D_n$;
- Una función objetivo f, donde $f : D_1 \text{ x } .... \text{ x } D_n \rightarrow \mathbb{R}^+$; $S = \{s = \{(\tau_1, v_1), ..., (\tau_n, v_n)\} | v_i \in D_i, \}$, y s satisface todas las restricciones.

S es llamado un espacio de búsqueda (o espacio de soluciones), ya que cada elemento del conjunto puede ser visto como una solución candidata. Para resolver un problema de optimización se encuentra una solución $\tau^* \in S$ con un valor de función objetivo mínimo, es decir, $f(\tau^*) \leq f(\tau) \forall \tau \in S$. τ^* es llamado solución global de (S, f) y $S^* \subseteq S$ se denomina el óptimo global de todo el cojunto de soluciones [2].

Complejidad Computacional.- La teoría de la NP-completitud muestra que muchos problemas para los cuales no se conoce algoritmo polinomial están relacionados (computacionalmente).

Un problema de clase NP-completo tiene la siguiente propiedad: Si algún problema NP-completo tiene solución con un algoritmo determinista en tiempo polinomial, entonces P $=$ NP. Si algún problema de la clase NP no tiene solución determinista en tiempo polinomial, entonces P $\neq$ NP y todos los problemas NP-completos no tienen solución determinista en tiempo polinomial [37]. Se puede ver gráficamente en la Figura 1.1, que un problema de la clase NP-completo sí está en NP, pero NP-completo $\neq$ NP.

Para demostrar que un problema es NP-completo, se debe probar que el problema está en NP y que todos los problemas en NP-completo se puedan transformar polinomialmente a dicho problema [8].



$NP - completo \neq NP$

Figura 1.1: Representación de problemas NP

Un problema de optimización es llamado NP-duro, si y solo sí su versión de decisión es NP-completo (NPC). Para probar que el problema de decisión $\Pi \in NPC$ es suficiente y necesario probar que $\Pi \in NP$ y que existe $\Pi^* \in NPC$ tal que $\Pi^* \leq_P \Pi$.

Una importante propiedad de los problemas NP-completos es que si $\Pi \in NPC$, entonces $\Pi \in P$ si y solo sí $P = NP$.

1.1.2 Problema de la Triangulación

El problema de la triangulación de matrices tiene una gran aplicación en economía, donde la globalización y la aceleración en el mundo de la economía incorpora mayor complejidad en la interacción entre los diferentes sectores económicos de un país. El modelo de entrada y salida desarrollado por Vassili Leontief [19] ha sido de gran importancia en la interpretación de la interacción económica. Este modelo construye una matriz E denominada matriz de $entrada$-$salida$ donde el tamaño es el número de sectores económicos en una región y donde, cada entrada e_{ij} en E denota la cantidad de insumos (en valor monetario) que el sector i provee al sector j en un periodo de tiempo.

El problema de la triangulación consiste en encontrar una permutación simultanea de columnas y filas de la matriz E que maximice la suma de los valores de la diagonal superior;

una matriz resultante del problema de la triangulación se llama matriz triangulada. Asimismo, el cociente es obtenido dividiendo la suma de los elementos por encima de la diagonal principal de E entre la suma de todos los elementos excepto los elementos de la diagonal, representa en economía un indicador llamado $linealidad$; el cual debe obtenerse a partir de matrices de $entrada\text{-}salida$ trianguladas. Este indicador se define formalmente como sigue:

$$\lambda = \frac{\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} e_{ij}}{\sum_{i=1}^{m} \sum_{j=1, i \neq j}^{m} e_{ij}} \tag{1.2}$$

Donde m es el número de sectores y $E = \{e_{ij}\}$ es la matriz cuadrada de tamaño m x m que representa la matriz de $entrada\text{-}salida$.

La propiedad $de\ linealidad$ establece una jerarquía entre los sectores económicos de un país, posicionando a los sectores que son preferentemente productores arriba de los que son consumidores sector de consumo. En consecuencia, los valores de $linealidad$ cerca de uno representan una economía que tiene una fuerte dependencia de un sector económico específico, por lo que será más susceptible a los cambios en la demanda. Economías fuertes tienden a obtener valores bajos (cercanos a cero) en este indicador, ya que no exhiben notable dependencia de un sector en particular.

Por ejemplo para representar el grado en el que una economía se apega a una jerarquía entre los sectores ganadero, industrial y de servicios se muestra la Tabla 1.2, de insumos−productos (Tabla de $entrada\text{-}salida$), la candidad que entrega el sector ganadero al sector industrial es de 90, el industrial al de servicios es de 77, y así sucesivamente; el indicador de $linealidad$ para este ejemplo es calculado con la Ecuación 1.2, al sustituir los valores de la Tabla 1.2, da como resultado $\lambda = 0.330$ (ver desarrollo en la Ecuación 1.3); este valor indica que la economía no depende de un sector.

$$\lambda = \frac{90 + 150 + 77}{70 + 90 + 180 + 90 + 150 + 77} = 0.330 \tag{1.3}$$

La economía mundial es la globalización de: la producción, los mercados, las finanzas, las comunicaciones, y la mano de obra. La expansión de la producción de empresas transnacionales

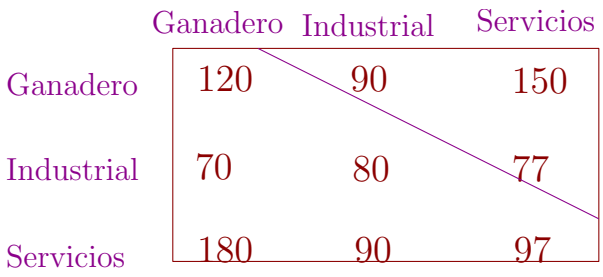


Figura 1.2: Tabla de Insumos-Productos

a muchos países alrededor del mundo ha generado un incremento en el número de sectores que suelen participar en la economía de un país. Por ejemplo, en los años 80, hubo países cuyas economías ya incluían entre los 400 y 500 sectores [19]. Por lo anterior, el tamaño de las tablas de $entrada\text{-}salida$ generadas por las economías de hoy en día son muy grandes, por lo tanto es imposible abordar el problema subyacente de triangulación usando los métodos clásicos de programación lineal. En estos casos, los métodos de solución basados en metaheurísticas representan una alternativa viable para obtener soluciones de buena calidad con una inversión de tiempo adecuado para las aplicaciones en la economía y la toma de decisiones.

1.2 Problema Ordenamiento Lineal (LOP)

El problema del ordenamiento lineal es equivalente al problema de la triangulación de matrices de $entrada\text{-}salida$ [3] y puede ser modelado como un problema de optimización combinatoría, el cual tiene como meta maximizar la función objetivo encontrando una permutación adecuada de los n elementos de un conjunto finito N. LOP considera una matriz m x m de valores $E = e_{ij}$, el problema LOP consiste en encontrar una permutación τ^* de columnas y filas de E, de tal manera que la suma de los costos en la matriz triangular superior sea maximizada. La expresión matemática para resolver este problema es la siguiente:

$$LOP(\tau^*) = max\left(\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} e_{p_i p_j}\right) \tag{1.4}$$

Donde p_i es el índice de la columna y fila en la posición i en la permutación τ^*.

Enseguida se describe el cálculo de la función objetivo mediante un ejemplo. Dada una

permutación $\tau = \{1, 2, 3, 4, 5, 6\}$ y su correspondiente matriz de costos E (ver Figura 1.3), el cáculo del valor de la función objetivo para la matriz E de tamaño 6 x 6 es el siguiente (ver Ecuación 1.5).

$$E = \begin{array}{cccccc|c} 1 & 2 & 3 & 4 & 5 & 6 & \\ \hline 20 & 120 & 80 & 110 & 30 & 90 & 1 \\ 30 & 70 & 15 & 55 & 10 & 50 & 2 \\ 50 & 80 & 40 & 60 & 20 & 67 & 3 \\ 120 & 50 & 30 & 70 & 80 & 45 & 4 \\ 20 & 10 & 50 & 80 & 95 & 160 & 5 \\ \underline{40} & 150 & 120 & 90 & 25 & \underline{120} & 6 \end{array}$$

Figura 1.3: Matriz cuadrada de *entrada-salida*

$$\begin{array}{l} LOP(E_\tau)=e_{1,2}+e_{1,3}+e_{1,4}+e_{1,5}+e_{1,6}+e_{2,3}+e_{2,4}+e_{2,5}+e_{2,6}+e_{3,4}+e_{3,5}+e_{3,6}+e_{4,5}+e_{4,6}+e_{5,6} \\ LOP(E_\tau)=120+80+110+30+90+15+55+10+50+60+20+67+80+45+160=992 \end{array} \quad (1.5)$$

Después de aplicar una permutación de filas y columnas en τ se obtiene $\tau' = \{4, 2, 3, 1, 5, 6\}$. La Figura 1.4, muestra la permutación de las filas y columnas 4 y 1 producidas por τ; también se muestra el cálculo de la función objetivo para τ'.

$$E = \begin{array}{cccccc|c} 4 & 2 & 3 & 1 & 5 & 6 & \\ \hline 70 & 50 & 30 & 120 & 80 & 45 & 4 \\ 55 & 70 & 15 & 30 & 10 & 50 & 2 \\ 60 & 80 & 40 & 50 & 20 & 67 & 3 \\ 110 & 120 & 80 & 20 & 30 & 90 & 1 \\ 80 & 10 & 50 & 20 & 95 & 160 & 5 \\ \underline{90} & 150 & 120 & 40 & 25 & \underline{120} & 6 \end{array}$$

$$\begin{array}{l} LOP(P') = 50 + 30 + 120 + 80 + 45 + 15 + 30 + 10 \\ + \ 50 \ + \ 50 \ + \ 20 \ + \ 67 \ + \ 30 \ + \ 90 \ + 160 \ = \ 847 \end{array}$$

Figura 1.4: Intercambio de las filas y columnas 4 y 1 de la matriz cuadrada de *entrada-salida*

Si se realiza otro intercambio de filas y columnas 6 y 2 en la permutación se obtiene $\tau'' = \{4, 6, 3, 1, 5, 2\}$. En la Figura 1.5, muestra el intercambio de los elementos, el cálculo producido en τ'' y el valor obtenido $LOP(\tau'')$ es igual a 970.

Lo anterior muestra que la mejor solución entre τ, τ' y τ'' es τ porque obtuvo el mayor valor $LOP(\tau)$ con 992 ya que este problema es de maximización, es decir, se pretende obtener el mayor valor de $LOP(E)$.

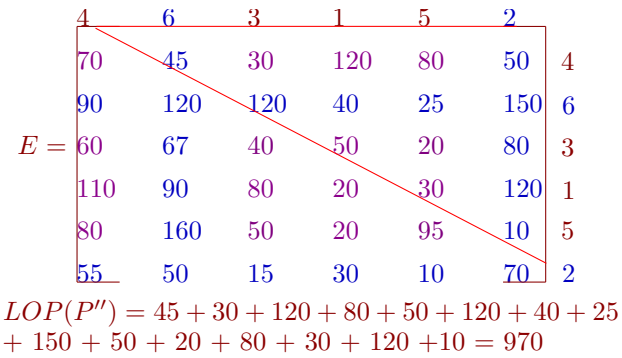


Figura 1.5: Intercambio de las filas y columnas 6 y 2 de la matriz cuadrada de $entrada\text{-}salida$

La solución del LOP puede ser representada con una permutación [4], por lo tanto, para garantizar la solución exacta, habría que explorar las $\frac{n!}{(n-2)!}$ [17] (donde n es el tamaño de la permutación) soluciones posibles, obtenidas al permutar las columnas y filas en la matriz de $entrada\text{-}salida$. Tal operación requiere mucho poder de cómputo por lo cual los algoritmos aproximados son una opción prometedora para encontrar soluciones competitivas en un tiempo reducido.

1.2.1 Metaehurísticas

Para resolver instancias de problemas NP-duro se pueden utilizar algoritmos exactos ó aproximados. Los primeros siempre encuentran la solución óptima pero el tiempo de ejecución crece factorialmente con el tamaño del problema.

En contraposición a los algoritmos exactos, los aproximados no garantizan la obtención de soluciones óptimas. Sin embargo, suelen obtener soluciones satisfactorias en tiempos de procesamiento generalmente polinomiales. En esta sección se describen los algoritmos de solución aproximada que tienen relación con el trabajo que se desarrollará en esta tesis.

1.2.1.1. Algoritmo recocido simulado (SA).

El algoritmo recocido simulado (Simulated Annealing, SA) fue propuesto por Kirkpatrick *et al.* [16], y su uso se ha extendido en problemas de optimización. Es un algoritmo de búsqueda capaz de escapar de óptimos locales, permitiendo que bajo cierta probabilidad se admitan

movimientos que empeoren la mejor solución encontrada hasta el momento. La probabilidad de aceptar soluciones malas depende del parámetro de temperatura t_i, cuando la temperatura disminuye, la probabilidad de aceptar peores soluciones decrementa en cada iteración del algoritmo (ver Algoritmo 1).

SA se basa en la analogía de un proceso termodinámico llamado templado de metales. En este proceso se requiere someter el metal a altas temperaturas, para despues enfriarlo lentamente, con el fin de obtener una estructura muy estable en su composición. [7, 21]. La estabilidad de la estructura depende de la velocidad de enfriamiento. Si la temperatura inicial no es lo suficientemente alta o si se aplica un enfriamiento rápido, se llega a un estado llamado metaestable, en el cual se obtiene un metal imperfecto. Por lo tanto el enfriamiento debe de ser cuidadosamente lento. Desde este punto de vista, el algoritmo SA simula los cambios de energía en un sistema sometido a un proceso de enfriamiento hasta que converge en un estado de equilibrio [10].

La analogía entre el sistema físico del SA y un problema de optimización corresponde a lo siguiente: la función objetivo del problema es análoga a la función de energía, una solución del problema de optimización corresponde a un estado del sistema, las variables de decisión asociadas a una solución del problema equivalen a las posiciones moleculares, el óptimo global corresponde con la estructura de mayor estabilidad, encontrar un mínimo local implica que se ha llegado a un estado metaestable y finalmente el control T, corresponde a la temperatura [8].

1.2.1.2. Búsqueda en vecindad variable (VNS).

La búsqueda en vecindad variable (Variable Neighborhood Search, VNS) fue propuesta por Mladenović y Hansen [25] como una efectiva metaheurística para problemas de optimización combinatoria. La idea básica consiste en un cambio sistemático de vecindario con un algoritmo de búsqueda local [25]. Este proceso de búsqueda explora los vecindarios ($N_k(\tau)$) cada vez más distantes de la solución actual (τ) y de allí salta a uno nuevo, sí y solo sí, una mejor solución ha sido encontrada ($\tau^{'} \in N_k(\tau)$), como lo muestra la Figura 1.6.

Algoritmo 1 Pseudocódigo del algoritmo SA

entrada: Esquema de enfriamiento
salida : τ

```
1  inicio
2      τ ← τ0;                                   // Generación de solución inicial
3      T ← Tmax;                                 // Temperatura inicial
4      repetir
5          repetir
6              Genera una solución vecina τ';
7              △E ← f(τ') − f(τ);
8              si △E ≤ 0 entonces
9                  τ ← τ';                       // Solución vecina aceptada
10             fin
11             si no acepta τ' con una probabilidad e^(−△E)/T);
12         hasta que se cumple la condición de equilibrio;
13         T ← g(T);                             // Temperatura actualizada
14     hasta que la condición de paro se cumple;
15     regresa τ;
16 fin
```

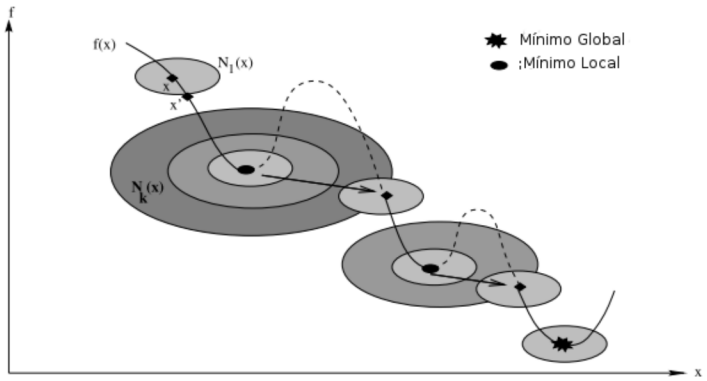


Figura 1.6: Esquema básico del funcionamiento del algoritmo VNS de descenso [14].

El algoritmo básico VNS [14] está presentado en el Algoritmo 2, el cual empieza con alguna solución inicial τ, después se genera $\tau^{'}$ que se obtiene aleatoriamente por medio de la función *Shaking* con el fin de evitar los ciclos (Línea 5); esta función puede consistir en una operación tan sencilla como los intercambios de atributos en la solución [14]. La Línea 6, es el proceso de mejora de la solución que se puede implementar con cualquier método de busqueda local; para obtener τ. En la Línea 7, se compara el nuevo valor de la solución $f(\tau^{''})$ con la aptitud de la solución $f(\tau)$ obtenida en el vecindario k. Si se obtuvo una mejora, k regresa a su valor

inicial (Línea 8). En caso contrario, el siguiente vecindario es considerado (Línea 10) [13].

Algoritmo 2 Pseudocódigo del algoritmo básico VNS [14]

entrada: seleccionar un conjunto de estructura de vecindario N_k, $k = 1,...,k_{max}$; elegir una condición de paro;
salida : τ;

```
1  inicio
2      τ ← encontrar una solución inicial;
3      repetir
4          para k ← 1 hasta k = k_max hacer
5              τ' ← Shaking(τ, k); // Genera τ' aleatoriamente del k^th vecindario de
                   τ(τ' ∈ N_k(τ)).
6              τ'' ← búsqueda local(τ'); // Aplicar algún método de búsqueda local a
                   la solución τ' para obtener un óptimo local τ''.
7              si (f(τ'') > f(τ) entonces
8                  τ ← τ''; k ← 1;            // Continua la búsqueda con N_1(k ← 1).
9              fin
10             si no k ← k + 1;
11         fin
12     hasta que la condición de paro se cumple;
13     regresa τ;
14 fin
```

1.2.1.3. Algoritmo memético.

La idea esencial de los algoritmos meméticos (Memetic Algorithm, MA) es hacer una hibridación de un algoritmo genético (Genetic Algoritmo, GA) con métodos de búsqueda local y aprovechar al máximo la capacidad operativa de los métodos basados en vecindarios y la capacidad de exploración de los GAs [26]. En diferentes contextos y situaciones, los MAs también se conocen como la búsqueda local genética o algoritmos genéticos híbridos.

1.2.1.4. Algoritmos híbridos.

En los últimos años, ha aumentado considerablemente el interés en algoritmos híbridos en el campo de la optimización. Esto ha sido motivado por el éxito que han obtenido estos Algoritmos al resolver gran cantidad de problemas de optimización de la vida real. En los algoritmos híbridos de bajo nivel, los procedimientos metaheurísticos están embebidos unos

dentro de otros. La principal idea es una combinación de algoritmos que, dada la función de una metaheurística, se sustituye por otra metaheurística quedando un único método de optimización [7].

A continuación se presentan algunos trabajos de la literatura especializada que resuelven problemas de optimización con algoritmos híbridos de SA con VNS.

Roshanaei *et al.* [31] propusieron un algoritmo híbrido de búsqueda en múltiples vecindarios (Multi-Neighborhood Search, MNS) con SA (llamado MNSSA) para dar solución al problema abierto de calendarización de tareas (open shop scheduling problem, OSSP). Ellos usaron SA para intensificar la solución al utilizar vecindades de tamaño pequeño y de MNS, el cual toma el principal concepto de VNS, que consiste en diversificar durante la búsqueda, mediante el principio de la exploración sistemática. Es decir, ejecuta consecutivamente todas las funciones de vecindad sin restricción; utililizando múltiples estructuras de vecindad.

El algoritmo MNSSA empieza con alguna solución inicial generada por un operador llamado *LTRMPT*. En cada temperatura genera un número pequeño de soluciones vecinas por medio de un operador llamado *SHIFT*, las cuales son aceptadas o rechazadas por el mecanismo de Metropolis. El operador *SHIFT* selecciona aleatoriamente la posición, checa la salida, y se intercambia en la permutación. Después emplea el operador llamado *DM*, el cual separa la solución del espacio de búsqueda actual, manteniendo una probabilidad de encontrar mejores soluciones. *DM* selecciona tres operaciones aleatoriamente, las cuales son recalculadas en tres diferentes posiciones aleatorias; de esta forma obtiene un vecino. *DM* produce un número considerable de vecinos y la mejor solución vecina es aceptada incluso si es peor que la solución actual. MNSSA obtuvo mejores resultados que los reportados previamente en el estado del arte, probó ser robusto y el más eficiente.

Rodriguez-Cristerna y Torres-Jimenez [30] propusieron un algoritmo híbrido basado en SA con VNS (llamado SA-VNS) para la construcción de arreglos de cobertura mixta. SA-VNS empieza con una solución inicial que es construida aleatoriamente, maximizando la distancia de Hamming. Para mejorar una solución M utilizan dos funciones de vecindad llamadas NF_1 y NF_2, las cuales son combinadas durante el proceso de búsqueda, de manera que, NF_1 se

va a ejecutar con una probabilidad p y NF_2 con una probabilidad 1-p. Usan el esquema de enfriamiento geométrico para decrementar la temperatura actual. La función *Shake* establece arbitrariamente los símbolos de $k^{'}$ celdas seleccionadas de forma aleatoria. La implementación del VNS fue realizada en el ciclo de cadena de Markov; donde el máximo número de estructura de vecindario es determinado por $k^{'}_{max}$. El valor de $k^{'}_{max}$ está estrechamente relacionado con la fuerza t del arreglo de cobertura a analizar, es decir, $k^{'}_{max} = t + 1$. La sintonización de los parámetros de su algoritmo lo realizaron con arreglos de cobertura.

El desempeño de SA-VNS fue probado con 19 arreglos de cobertura mixta, comparando los resultados con los mejores resultados del estado del arte. Este Algoritmo pudo igualar 12 soluciones y mejorar 6 cotas.

1.2.1.5. Algoritmos de búsqueda dispersa (SS).

La búsqueda dispersa fue propuesta por Fred Glover en 1977 en [11], SS es una estrategia determinística que se ha aplicado con éxito a problemas de optimización combinatoria y continua. SS es una metaheurística evolutiva poblacional que recombina soluciones seleccionadas de un conjunto de referencia para construir nuevas soluciones [7]. El método comienza con la generación de una población inicial con individuos diversos. El conjunto de referencia (generalmente de tamaño pequeño) se construye al seleccionar de la población soluciones representativas de buena calidad. Las soluciones del conjunto de referencia se agrupan sistemáticamente en subconjuntos de dos o más individuos. Posteriormente, las soluciones de cada subconjunto son combinadas con alguna técnica para producir nuevos individuos. El método de mejora se aplica a cada nueva solución generada. El método se puede implementar mediante un algoritmo de búsqueda local. El conjunto de referencia se actualiza para incorporar tanto soluciones de alta calidad como soluciones diversas con respecto a los elementos que se encuentran en el mismo conjunto de referencia. El proceso itera hasta que un criterio de paro es alcanzado. El diseño del algoritmo de búsqueda dispersa generalmente se basa en las fases mostradas en la Figura 1.7.

El pseudocódigo básico del algoritmo de búsqueda dispersa se muestra en el Algoritmo

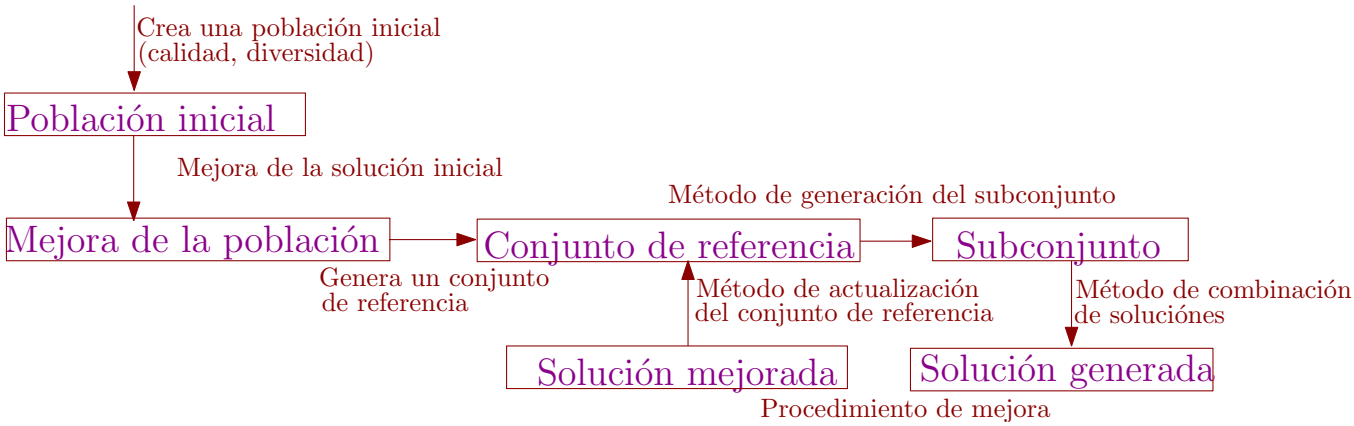


Figura 1.7: Componentes del algoritmo de búsqueda dispersa

Algoritmo 3 Pseudocódigo del algoritmo básico SS [7]

entrada: elegir una condición de paro;
salida : La mejor solución;

```
1  inicio
2      Inicializar la población Pop usando un método de diversificación;
3      Aplicar un método de mejora a la población;
4      Método de actualización del conjunto de referencia;
5      repetir
6          Método de generación de subconjuntos;
7          repetir
8              Método de combinación de solución;
9              Método de mejora;                          // búsqueda local
10         hasta que la condición de paro se cumple;
11         Método de actualización del conjunto de referencia;
12     hasta que la condición de paro se cumple;
13     regresa La mejor solución encontrada;
14 fin
```

3. SS tiene cinco componentes básicos de búsqueda: método de generación de diversidad, de mejora y de combinación, los que pueden verse como procedimientos específicos; mientras que los otros dos componentes de búsqueda (método de generación y actualización del conjunto de referencia) son mas bien genéricos. Los métodos que componen el algoritmo SS son descritos a continuación:

Método de diversificación

El método genera un conjunto de soluciones iniciales diversas. En general se aplican procedimientos basados en aleatoriedad ($randomprocedures$) para generar soluciones diversas.

Método de mejora

Este método usa un algoritmo de búsqueda local para mejorar las nuevas soluciones obtenidas mediante el método de generación de soluciones diversas, así como también para mejorar las soluciones obtenidas por el método de combinación de soluciones.

Método de generación y actualización del conjunto de referencia

El conjunto de referencia contiene tanto soluciones de alta calidad como soluciones diversas que son usadas para generar otras soluciones. Por lo anterior, el conjunto de referencia $RefSet$ está compuesto por soluciones con alto valor en la función objetivo ($RefSet1$) y soluciones con mayor diversidad ($RefSet2$), es decir $Refset = RefSet1 + RefSet2$.

Método de generación de subconjuntos

Este método opera sobre el conjunto de referencia $RefSet$ para generar subconjuntos de soluciones para crear soluciones combinadas. Este método usualmente genera todos los subconjuntos de tamaño r (en general $r = 2$) [7]. Este procedimiento es similar a los algoritmos evolutivos en el mecanismo de selección, solo que en los evolutivos se aplica sobre el tamaño de la población y en SS sobre el conjunto de referencia y difiere principalmente en que el conjunto $RefSet$ es mucho más pequeño.

Método de combinación de solucines

A partir del conjunto de referencia se construyen sistemáticamente subconjuntos de soluciones las cuales se combinan para obtener nuevas soluciones. Este método puede ser visto como el operador de cruza en los algoritmos evolutivos donde más de dos individuos son

recombinados.

1.2.1.6. Intensificación y diversificación.

La idea original de la intensificación y diversificación nació en la búsqueda tabu. Hoy en día no sólo la búsqueda tabu cuenta con estas estrategias, ya que se han adoptado a otras metaheurísticas por su gran potencial [6]. Dado una metaheurística, la intensificación puede ser definida como el proceso de búsqueda exhaustiva que realiza dentro de un vecindario. Por lo general, una metaheurística no intensifica en cualquier momento, sino que tiene en cuenta varios factores como verificar la calidad de la solución que se obtiene dentro de la vecindad, por ejemplo, si una mala solución ha sido encontrada entonces no es recomendable intensificar la búsqueda en esa región, por lo que no se lleva a cabo la búsqueda exhaustiva del espacio completo de soluciones vecinas.

Por otro lado, la diversificación es el proceso por el cual una metaheurística puede ser capaz de visitar diferentes vecindarios. Por lo general, una metaheurística no diversifica constantemente sin la aplicación de un criterio, ya que el algoritmo podría convertirse en una búsqueda totalmente aleatoria. Una metaheurística realiza el proceso de diversificación tomando en cuenta varios factores uno sería, comprobar que determinada región del espacio de búsqueda no ha sido visitada.

1.3 Justificación

Desde el punto de vista práctico el problema LOP es importante debido a que existen diversos problemas reales que pueden ser modelados mediante el uso de la matriz de $entrada$-$salida$ y cuya solución implica encontrar la máxima suma de los elementos sobre la diagonal superior. Tal es el caso del problema de la triangulación derivado del ámbito económico donde se pretende obtener el $indicador\ de\ linealidad$ para representar el grado en el que en una economía los sectores económicos se apegan a una jerarquía.

Otras aplicaciones prácticas se encuentran en los problemas del campo de la electrónica

donde cuenta con aplicaciones para el diseño de circuitos integrados y dibujo automático de grafos [32], si bien en la computación móvil se utiliza para modelar el problema de la indexación y la asignación de los datos en los canales de transmisión [20].

Dentro de las ciencias sociales surge LOP en el problema de agregación de preferencias, donde se utiliza para la clasificación de las preferencias. Por otro lado, en el campo de la arqueología se utiliza en el problema de asignación a la ordenación cronológica de un conjunto de artefactos recuperados de diferentes lugares de estudio.

Desde el punto de vista científico, el problema de estudio es importante, debido a que LOP es NP-duro [15], [9] y [12]. Esta clase de problemas se caracterizan porque el tamaño del espacio de búsqueda involucrado crece de forma factorial en función del tamaño del problema, lo cual conlleva limitantes para la aplicación de métodos exactos para su solución debido al tamaño del espacio de búsqueda. Para ello es necesario desarrollar algoritmos aproximados que sean altamente eficientes y que reduzcan considerablemente los tiempos de cómputo registrados en el estado del arte.

Se ha encontrado **??** que el buen desempeño de los algoritmos metaheurísticos depende del balance entre diversificación e intensificación, sin embargo aún existe muy poco estudio en sobre técnicas que permitan que permitan realizar este balance.

Por lo expuesto anteriormente, se concluye que el tema que se abordará en esta tesis es relevante y complejo.

1.4 Hipótesis

Es posible mejorar el desempeño de una metaheurística mejorando el balance entre intensificación y diversificación.

1.5 Alcances y limitaciones

Este proyecto de investigación sólo se enfocará en desarrollar una metodología para estudiar el balance de intensificación y diversificación. Dicha metodología se probará en el algoritmo de búsqueda dispersa para el problema de ordenamiento lineal utilizando un subconjunto de instancias de prueba representativo del estado del arte.

1.6 Objetivos generales y específicos del proyecto

1.6.1 Objetivo general

Contribuir con una metodología para estudiar el balance de intensificación y diversificación para el algoritmo de búsqueda dispersa aplicado al problema de ordenamiento lineal.

1.6.2 Objetivos específicos

Los objetivos particulares para este trabajo de investigación son:

- Identificar los componentes clave que participan en el balance de intensificación y diversificación.
- Analizar los componentes clave en el algortimo de búsqueda dispersa para la solución de LOP con el fin de encontrar el posible balance óptimo entre estos componentes clave.
- Diseñar una representación para el balance de intensificación y diversificación.
- Evaluar experimentalmente la representación propuesta.

1.7 Organización de la tesis

A continuación, se presenta una breve descripción del contenido de los siguientes capítulos:

- **Capítulo 1:** Se presentan algunos conceptos básicos para introducir de manera formal el problema LOP y otros conceptos necesarios para el mejor entendimiento del documento de tesis.
- **Capítulo 2:** Se describen los métodos propuestos en la literatura especializada mediante los cuales se ha abordado el problema LOP.
- **Capítulo 3:** El enfoque propuesto en esta tesis para resolver el problema LOP es descrito en este capítulo. Se detalla cada componente utilizado en el algoritmo propuesto para este problema.
- **Capítulo 4:** Se describen los experimentos realizados para la evaluación de las metodologías propuestas.
- **Capítulo 5:** Se presenta un análisis comparativo, entre los resultados obtenidos por nuestro algoritmo y aquellos reportados en la literatura.
- **Capítulo 6:** Se presenta el trabajo futuro y las conclusiones a las que se llegan en este trabajo de investigación.

2

Estado del arte

En este capítulo se describe brevemente algunos métodos metaheurísticos que se han propuesto en la literatura para resolver el problema que da origen a este trabajo de tesis.

2.1 Algoritmos metaheurísticos de solución

La primer metaheurística para resolver LOP fue desarrollada por Laguna *et al.* [18]. Ellos diseñaron una solución eficiente basada en búsqueda tabú con estrategias de intensificación y diversificación. Para la fase de intensificación usaron una función de vecindad con inserción llamada $best$, la memoria de corto plazo se basa en la experiencia reciente mientras que la memoria a largo plazo en la fase de diversificación. Una estrategia de $path\text{-}relinking$ fue incluida para mayor intensificación; basada en un conjunto de soluciones elite. La memoria a largo plazo registró la frecuencia de las soluciones visitadas con el fin de seleccionar las soluciones menos exploradas. Al final de un ciclo de cada estrategia usan una búsqueda local que mejora la solución obtenida. Un indicador de $influencia$ fue usado para obtener un sesgo en la selección de sectores, favoreciendo con mayor probabilidad a los sectores con mayor valor del indicador de influencia. El rendimiento del algoritmo fue evaluado usando 49 instancias para

LOP, estas fueron obtenidas de la biblioteca estándar LOLIB, 75 instancias Stanford Graph Base y 75 instancias generadas aleatoriamente. El algoritmo propuesto fue comparado con dos heurísticas; los resultados de tabú muestran que la memoria de largo plazo es un importante componente ya que obtiene mejoras en todos los casos con respecto al tabú básico. Laguna *et al.* hicieron hincapie en la importancia del equilibrio entre la intensificación y diversificación ya que la combinación entre ellos produce buenos resultados [18].

Schiavinotto *et al.* [34] realizaron un estudio detallado y análisis del espacio de búsqueda de LOP así como el diseño de dos algoritmos: búsqueda local iterada y un algoritmo memético. El estudio del espacio de búsqueda de las instancias LOP muestra que los casos de prueba suelen tener grandes correlaciones de distancia de aptitud. Por tanto, estos métodos son prometedores para este problema ya que son capaces de explotar tanto el buen rendimiento de la búsqueda local y la alta correlación de distancia de aptitud.

Puesto que los métodos de solución pueden tener ventaja sobre otros métodos, desarrollaron un algoritmo memético el cual es una hibridación de un algoritmo genético con búsqueda local. El método de búsqueda local utiliza una función de vecindad mediante inserción y un criterio de selección de vecinos llamado $best$ y $first$. La configuración final incluye una población de 25 individuos, un operador de cruza OB (Order Based) y una estrategia de diversificación que restablece la población después de 30 iteraciones sin mejora. En este algoritmo memético no es aplicado el operador de mutación.

Además, desarrollaron una búsqueda iterada local que utiliza una búsqueda local entre la $best$ y la $first$, también un proceso de perturbación basado en el intercambio de movimientos. Este algoritmo memético obtuvo excelentes resultados posicionandose como el método de solución para LOP con los mejores resultados del estado del arte.

Un estudio exhaustivo de los métodos de solución para LOP fue desarrollado por Martí [22] en el que se incluyen los resultados de una evaluación estandarizada de 7 heurísticas y 10 metaheurísticas. Los resultados en este conjunto de pruebas muestran que el algoritmo memético obtiene el mejor rendimiento del estado del arte LOP. El proceso de evalución lo ejecutaron aplicando 10 y 600 segundos sobre el conjunto de instancias $OPTI$ y UBI; el cual incluye

255 instancias estandar.

Richard Congram [5] propuso una metodología para explorar vecindarios de tamaño exponencial en tiempo polinomial. El introduce un nuevo método que aplica una estrategia basada en programación dinámica la cual fue llamada $Dynasearch$, trata de combinar una serie de movimientos individuales independientes del vecindario tal como el intercambio, inserción, 3-opt y Lin-Kernighan, en una sola iteración para construir un sólo movimiento $dynasearch$. Los resultados muestran que el algoritmo iterativo $dynasearch$ supera a todas las versiones $multi$-$boot$ descendente y para cualquier criterio de aceptación supera los algoritmos $first$-$improve$ y $best$-$improve$ descendentes.

Por otra parte, la búsqueda tabú obtuvo mayor número de óptimos junto con un alto promedio del valor de la función objetivo en un tiempo reducido; el algoritmo $dynasearch$ mostró ser 5 veces más rápido que la búsqueda tabú [18]. También fue evaluado un enfoque híbrido con criterio de aceptación, el cual aplica el criterio de aceptar cualquier solución que mejora en algunas iteraciones, después continua con el criterio de aceptar solo la mejor solución. Resultados muestran que el enfoque híbrido supera en calidad de solución y tiempo de cómputo al enfoque $dynasearch$ que usa un criterio simple.

Campos *et al.* [3] implementaron un algoritmo de búsqueda dispersa para LOP, su principal contribución se encuentra en mecanismos novedosos para combinar soluciones con el fin de crear en el conjunto de soluciones un balance entre divesidad y calidad de solución. También utiliza un proceso de rastreo que genera soluciones estadísticamente diferentes a la naturaleza de las combinaciones y los rangos de soluciones que produjeron las mejores soluciones finales. Desarrollaron 10 métodos para generar soluciones diversas, 6 de estos basados en $GRASP$ con una función voraz. Por ejemplo, el método que aportó mayor diversidad y calidad a las soluciones empleó un contador de frecuencias para recordar el número de veces que el elemento i aparecen en la posición j. Los contadores de frecuencia se utilizan para penalizar la atractividad de un elemento en una posición dada durante construcciones subsecuentes, por lo tanto inducen la diversidad con respecto a las soluciones en P. También usan la ecuación para la atracividad de asignar el elemento i a la posición j la cual es una función $greedy$

adaptativa.

El desempeño del algoritmo fue probado con más de 300 instancias y comparado con los mejores algoritmos identificados previamente, demostrando ser eficaz y competitivo con respecto a los mejores métodos de la literatura.

Campos *et al.* [4] desarrollaron dos metaheurísticas basadas en búsqueda dispersa y búsqueda tabú donde la evaluación de la función objetivo es como una caja negra haciendo de los algoritmos de búsqueda independientes del contexto. Su principal contribución es un procedimiento que no utiliza conocimiento del contexto del problema para su solución, así como su evaluación en cuatro problemas de permutación búsqueda dispersa se proponen un generador de diversidad y 10 métodos de combinación de soluciones, también dos métodos de mejora basados en $hill\text{-}climbing$ y busqueda tabú. Las soluciones obtenidas con los métodos propuestos han sido comparadas con las mejores soluciones de cada problema. El procedimento mostró ser competitivo para LOP comparado con otros diseños pero para los otros problemas los resultados no fueron muy favorables. Los experimentos muestran que el procedimiento podría ser efectivo para problemas cuyas soluciones puedan ser representadas con permutaciones mientras la función objetivo sea capaz de discriminar entre soluciones vecinas en el espacio de búsqueda.

Rafael Martí *et al.* [22] proponen un algoritmo de búsqueda dispersa para LOP, enfatizan sus aportaciones en tres elementos del algoritmo: la creación, actualización y reconstrucción del conjunto de referencia estos se describen a detalle a continuación.

Creación del conjunto de Referencia.- Este conjunto está compuesto por $b = b_1 + b_2$, b_1 está formado con los elementos de mejor calidad de solución del conjunto P (donde P es un conjunto de aproximadamente 100 soluciones) y estos son eliminados de P, para cada elemento en P que no está en el conjunto de referencia $Refset$ se calcula la mínima distancia a los elementos de $Refset$, después se selecciona el elemento que tenga la máxima mínima distancia; esta solución es añadida a $RefSet$, posteriormente es eliminada en P y las mínimas distancias son actualizadas. Este proceso es repetido b_2 veces para obtener un conjunto de referencia b_1 soluciones de buena calidad y b_2 soluciones diversas. El método para construir

soluciones diversas es llamado FQ, el cual toma con un nivel de importancia equivalente la información de frecuencia, el mejor desempeño, la calidad y diversidad. El método de mejora se basa en la metodología de $GRASP$ usando inserción con movimientos; para incrementar el valor de la función objetivo es aplicado este método de mejora.

Método de actualización del conjunto de referencia.- El cual aplica el método de generación de subconjuntos que involucra la generación de todos los posibles pares del conjunto de referencia generando una solución de prueba por cada par, posteriormente se actualiza el conjunto de referencia con las mejores soluciones de acuerdo con el valor de la función objetivo entre el conjunto de referencia actual y las soluciones de prueba, este proceso termina hasta que ninguna solución del conjunto de prueba pasa a formar parte del nuevo conjunto de referencia.

El método para reconstruir el conjunto de referencia.- El cual primero asume que el conjunto está ordenado de la mejor solución (τ_1) a la peor (τ_b). Cuando una nueva solución de prueba τ es generada su valor de la función objetivo se incorpora al conjunto $RefSet$ y se actualiza de tal forma que si es una solución peor que τ_b entonces el conjunto de referencia no es actualizado, por tanto se realiza la reconstrucción del conjunto de referencia de la siguiente manera: primero se eliminan los elementos de $RefSet$, posteriormente se construye P con un conjunto diverso de soluciones nuevas. Después secuencialmente se seleccionan b_2 soluciones de P usando el mismo criterio de la máxima mínima distancia a los elementos de $RefSet$. Ellos reportan una serie de experimentos el primero es diseñado para encontrar la mejor configuración entre parámetros del algoritmo propuesto. El segundo es para conocer el rango de las soluciones del conjunto de referencia que generan las mejores soluciones encontradas en el proceso de búsqueda. El tercer experimento consiste en comparar algunos algoritmos del estado del arte para LOP sobre matrices de $entrada - salida$, instancias $OPT - I$ y problemas aleatorios. Los resultados muestran que el algoritmo propuesto es más robusto por el número de óptimo obtenidos, mientras que la diferencia de la desviación estándar con respecto al óptimo entre los algoritmos del estado del arte es muy pequeña.

2.2 Resumen

Los enfoques metaheurísticos han demostrado que son técnicas que permiten obtener buena calidad de solución en tiempo competitivo para LOP. Los algoritmos híbridos por su naturaleza llevan inmersos búsqueda local y estrategias de diversificación, lo cual ayuda en obtener buenas soluciones y un buen rendimiento en la metaheurística, teniendo como resultado soluciones competitivas. También los resultados muestran la gran importancia de mantener un apropiado balance de intensificación y diversificación en las metaheurísticas para obtener alta calidad de solución; ya que si existe mayor diversificación (no significa que sea aleatorio) en la búsqueda se corre el riesgo de visitar varios puntos dentro del espacio de búsqueda y no explotar un vecindario promotedor. Sin embargo, si el algoritmo realiza mucha intensificación puede converger de forma prematura.

3

Metodología propuesta

Se ha encontrado que el desempeño de los enfoques metaheurísticos depende fuertemente de establecer una interacción estrategica entre los métodos de intensificación y diversificación. Sin embargo, no existe una regla sobre como balancear estos importantes componentes y los artículos de investigación que abordan este problema son escasos hasta ahora. La metodología propuesta en este trabajo se fundamenta en buscar interrelaciones entre los métodos de construcción del conjunto de referencia, el desempeño del algoritmo, y el balance de intensificación y diversificación.

En particular el algoritmo de búsqueda dispersa tiene elementos o métodos bien definidos en donde se puede apreciar que son intensificadores, diversificadores o neutros. El enfoque de estudio de esta investigación es utilizar a los elementos del algoritmo SS que son claramente intensificadores y diversificadores para poder modificar el balance entre ellos.

Los principales pasos propuestos de la metodología desarrollada para el estudio del balance en SS son descritos en los siguientes puntos.

1. Análizar los procesos involucrados en SS con el fin de tipificar cuales de ellos son procesos que explicitamente tienen tarea intensificadora y cuales tienen tarea diversificadora.

Tabla 3.1: Análisis de los procesos de SS para identificar su principal tarea

Procesos	Intensificación	Diversificación	Ambos
Generar P		✓	
Generar $RefSet$	✓	✓	
Combinar		✓	
Actualizar y Reiniciar $RefSet$	✓	✓	
Mejorar	✓		
Generar subconjuntos			✓

$P =$ Población y $RefSet =$ Conjunto de referencia.

La Tabla 3.1, muestra los procesos del algoritmo de SS indicando con una marca (✓) la tarea que interviene en cada uno de ellos.

2. El análisis realizado en el paso 1 permite enfocar el estudio en los procesos de SS que tienen tareas tanto intensifacadoras como diversificadoras. En la Tabla 3.1, se puede ver claramente que la naturaleza de los procesos para generar y actualizar el conjunto de referencia tienen tareas duales, por lo tanto ahí es donde se va a centrar la investigación.

    - Considerando que los procesos que se relacionan con la construcción del conjunto de referencia cumplen con la dualidad en tareas, es importante buscar un balance adecuado para manter el conjunto de referencia con soluciones de diversidad y de alta calidad.

3. Diseñar e implementar estrategias encaminadas a producir diferentes balances de intensificación y diversificación en $RefSet$, con el fin de asegurar de alguna manera que existan soluciones distantes.

4. Realizar un análisis detallado del comportamiento de la metaheurística a lo largo de la ejecución con especial enfasis en el conjunto de referencia; de igual manera realizar el seguimiento de la mejor solución con intención de observar la aportación de cada proceso en la búsqueda global.

    - Paro esto se realizá un experimento utilizando cinco diferentes métodos de construcción para el algoritmo SS; específicamente para los procedimientos de cons-

trucción inicial y actualización del conjunto de referencia.

3.1 Algoritmo búsqueda dispersa

3.1.1 Descripción general

Los principales procesos del algoritmo SS (ver Figura 3.1) son: un *método de diversificación*, responsable de producir P, un conjunto grande de soluciones diversas, de las que se extraen las soluciones para construir el conjunto *RefSet*. Un *método de mejora*, que por lo regular se implementa mediante una Búsqueda Local (*LS* por sus siglas en inglés) o una heurística. Un método para generar y actualizar el conjunto de referencia *RefSet*, se aplica el mismo enfoque para inicializar y actualizar el conjunto *RefSet*, el cual consiste en dos pasos: en el primer paso se selecciona las mejores soluciones de acuerdo a su valor objetivo, posteriormente son incorporadas en el conjunto *RefSet* y después eliminadas del conjunto *P*; en el segundo paso se selecciona de *P* las soluciones más diversas con respecto al conjunto *RefSet* y se eliminan del conjunto *P*. La medida de diversidad se calcula en base a alguna métrica de distancia (ver Subsección 3.1.5). Un *método generador de subconjuntos*, el cual construye subconjuntos de soluciones tomadas del conjunto *RefSet*. Un *proceso de combinación de soluciones*, para combinar los subconjuntos previamente formados, el cual se basa en una técnica inspirada en el operador llamado Order Based crossover (*OB*) [40].

Se proponen cinco estrategias para generar y actualizar el conjunto *RefSet*. Estas estrategias utilizan cuatro métricas de distancia entre permutaciones y dos indicadores de disimilaridad, las cuales son descritas en la Subsección 3.1.2. Las métricas utilizadas son: distancia Hamming [38], distancia de desviación [33], distancia de desviación modificada [28], distancia entre una permutación y un conjunto de permutaciones. Para medir la disimilaridad entre dos conjuntos de permutaciones, los índices utilizados son: *IDIV* e *IDISIM* [39].

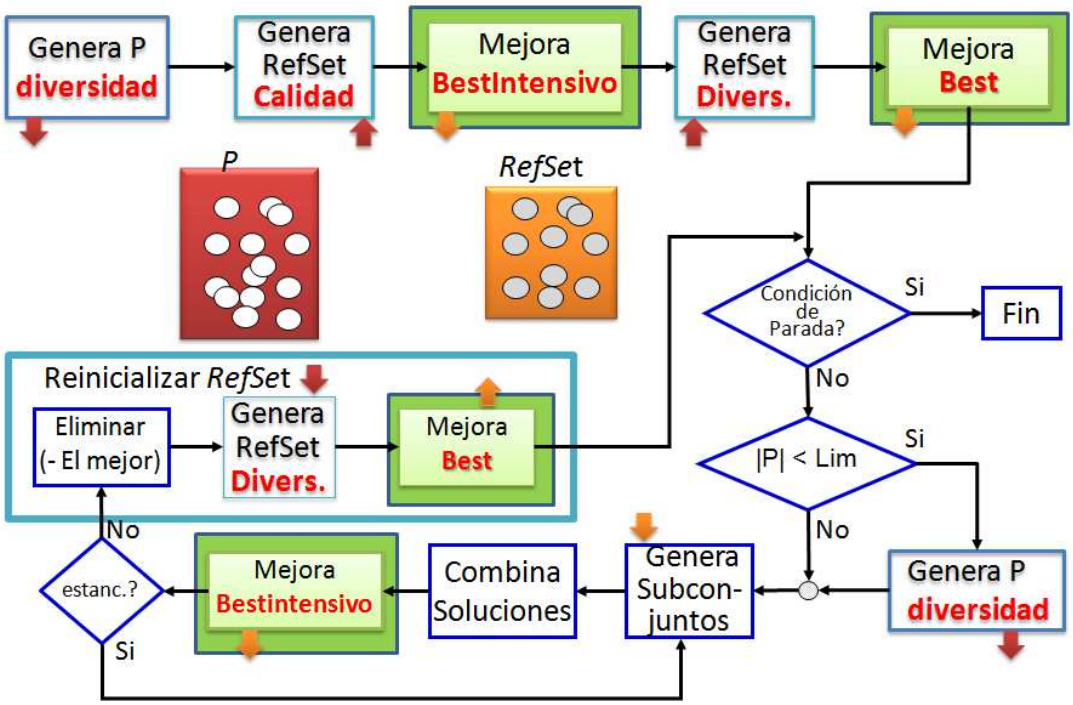


Figura 3.1: Esquema general del algoritmo de búsqueda dispérsa [39]

3.1.2 Componentes

A continuación se describen detalladamente los procesos principales del algoritmo SS para el problema LOP. El Algoritmo 6, muestra el pseudocódigo de SS.

El método de diversidad crea de forma aleatoria un conjunto de $|P|$ soluciones, el cual proverá soluciones para la construcción del conjunto de referencia, con cardinalidad dada por $|P| = 10 \times |RefSet| = 10 \times 10 = 100$. Las soluciones en P son generadas aleatoriamente y ordenadas de acuerdo a su valor objetivo.

El método de mejora utiliza movimientos de inserción en los elementos de una permutación. El proceso de mejora en la SS se realiza mediante dos algoritmos de búsqueda local: LS_1 explora todos los vecinos de la solución actual y elige el mejor encontrado (ver Algoritmo 4). LS_2 implementa una búsqueda más intensiva ya que vuelve a revisar para cada elemento de la permutación todos sus vecinos siempre que alguna mejora ocurra en una iteración (ver Algoritmo 5).

El proceso de actualización del conjunto de referencia. El proceso de creación de

Algoritmo 4 Pseudocódigo de LS_1

entrada: Alguna solución τ;
salida : La mejor solución;

inicio
  $costo \leftarrow$ Calcula Costo(τ);
  para $i \leftarrow 1$ **hasta** $i = |\tau|$ **hacer**
    si $i > 2$ **entonces**
      $costo_{max} \leftarrow$ Calcula Costo(Movimiento Inserción($\tau, i-1$));
    fin
    si no
      $costo_{max} \leftarrow$ Calcula Costo(Movimiento Inserción($\tau, i+1$));
    fin
    $j \leftarrow i-1$;
    $mejora \leftarrow 0$;
    mientras *no* $mejora$ *y* $j \geq 1$ **hacer**
      si *Calcula Costo*(*Movimiento Inserción*(τ, j)) $> costo_{max}$ **entonces**
        $costo_{max} \leftarrow$ Calcula Costo(Movimiento Inserción(τ, j));
        $j_{max} \leftarrow j$;
        $mejora \leftarrow 1$;
        break;
      fin
      $j \leftarrow j-1$;
    fin
    $j \leftarrow i+1$;
    mientras *no* $mejora$ *y* $j \geq n$ **hacer**
      si *Calcula Costo*(*Movimiento Inserción*(τ, j)) $> costo_{max}$ **entonces**
        $costo_{max} \leftarrow$ Calcula Costo(Movimiento Inserción(τ, j));
        $j_{max} \leftarrow j$;
        $mejora \leftarrow 1$;
        break;
      fin
      $j \leftarrow j+1$;
    fin
    si $mejora > 0$ *y* $costo_{max} > 0$ **entonces**
      $\tau^{'} \leftarrow$ Movimiento Inserción(τ, i, j_{max});
      $costo \leftarrow costo + costo_{max}$;
      $\tau^{*} \leftarrow \tau^{'}$;
    fin
  fin
  regresa *La mejor solución encontrada;*
fin

Algoritmo 5 Pseudocódigo de LS_2

entrada: Alguna solución τ;
salida : La mejor solución;

```
1  inicio
2      costo ← Calcula Costo(τ);
3      repetir
4          para i ← 1 hasta i = |τ| hacer
5              si i > 2 entonces
6                  costo_max ← Calcula Costo(Movimiento Inserción(τ, i − 1));
7              fin
8              si no
9                  costo_max ← Calcula Costo(Movimiento Inserción(τ, i + 1));
10             fin
11             para j ← i − 1 hasta j = 1 hacer
12                 si Calcula Costo(Movimiento Inserción(τ, j)) > costo_max entonces
13                     costo_max ← Calcula Costo(Movimiento Inserción(τ, j));
14                     j_max ← j;
15                 fin
16             fin
17             para j ← i + 1 hasta j = |τ| hacer
18                 si Calcula Costo(Movimiento Inserción(τ, j)) > costo_max entonces
19                     costo_max ← Calcula Costo(Movimiento Inserción(τ, j));
20                     j_max ← j;
21                     mejora ← j;
22                 fin
23             fin
24             si costo_max > 0 entonces
25                 τ' ← Movimiento Inserción(τ, i, j_max);
26                 costo ← costo + costo_max;
27                 τ* ← τ';
28                 mejora ← 1;
29             fin
30         fin
31     hasta que mejora = 1;
32     regresa La mejor solución encontrada;
33 fin
```

$RefSet$ inicia con la construcción del conjunto, el cual se compone de b_1 soluciones, de las cuales, la mitad son diversas ($b_1 = b/2$) y la otra mitad son de calidad ($b_2 = b/2$); inicia con las b_2 mejores soluciones de calidad tomadas de P, después es aplicado el método de mejora (LS_1). Las b_1 soluciones más diversas con respecto al conjunto de referencia son tomandas de P y mejoradas con LS_2. La métrica de distancia propuesta por Pantrigo [28] ($d_{mdev}(R_i, Q_j)$, dadas dos soluciones R_i, Q_j donde $R_i \in P$ y $Q_j \in RefSet$) es utilizada para seleccionar las soluciones más diversas de P. La diversidad es calculada con el índice $IDIV(R_i)$ para cada solución candidata $R_i \in P$; este índice utiliza la mètrica de distancia d_{mdev}. Las soluciones en P con mayor valor $IDIV$ son incluidas en el conjunto de referencia.

La actualización del conjunto de referencia se realiza mediante la incorporación de la solución resultante del proceso de combinación, si sólo si, es mejorada la peor solución de $RefSet$.

Adicionalmente este proceso incluye una reinicialización del conjunto $RefSet$ cuando se detecta convergencia en las soluciones del conjunto de referencia. El proceso de reinicialización mantiene la mejor solución de $RefSet$ y sustituye el resto por soluciones tomadas de P utilizando las estrategias de diversificación descritas en la Subsección 3.1.5. Finalmente el método de mejora LS_2 es aplicado a todo el conjunto de referencia.

El método de generación de subconjuntos: Este método opera en el conjunto de referencia, para producir un subconjunto de soluciones como una base para la creación de soluciones combinadas. Este método selecciona todos los subconjuntos de tamaño.

El método de combinación combina dos soluciones del conjunto de referencia utilizando el operador OB. Este operador ha sido aplicado satisfactoriamente para problemas de calendarización, donde el objetivo es obtener el orden (o secuencia) en el cual los trabajos son asignados a la calendarización [24]. La característica básica del operador OB es que permite mantener el orden relativo de los alelos en los cromosomas. Un número de elementos son seleccionados de uno de los padres y copiado al hijo. Los alelos perdidos son tomados del otro padre en orden [40] (ver Figura 3.2).

El método de combinación utilizado elige aleatoriamente $0.4 \times |\tau|$ posiciones de la primera

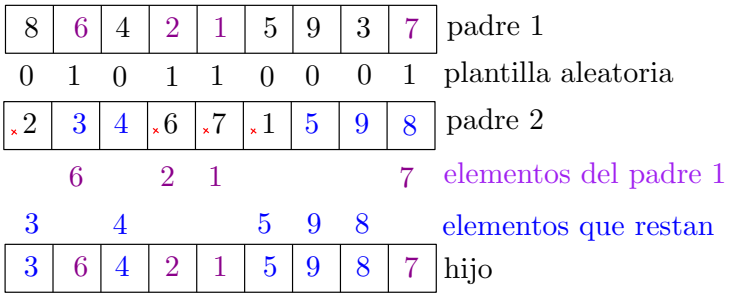


Figura 3.2: Un ejemplo del operador OB [24]

permutación, después los valores que están en las posiciones no seleccionadas se copian directamente en las correspondientes posiciones a la nueva permutación y después ordena los valores en las posiciones seleccionadas de acuerdo al orden de aparición en la segunda permutación y finalmente copia estos valores en las posiciones faltantes en la nueva permutación.

3.1.3 Métricas de distancia

Un conjunto de estrategias enfocadas a mejorar la diversidad en el algoritmo de búsqueda dispersa son aplicadas en el proceso de generación y actualización del conjunto de referencia *RefSet*. Estas estrategias fueron basadas en el uso de un conjunto de indicadores de distancia y diversidad para evaluar las soluciones en el conjunto P con el fin de realizar un filtrado a las soluciones que se añadirán en el conjunto de referencia.

La **distancia de Hamming**, es definida como el número de veces que los caracteres en la misma posición en dos permutaciones son diferentes. Esta distancia puede ser normalizada dividiéndola por la máxima distancia. La distancia es máxima y además igual a n si ambas permutaciones son diferentes en cada una de las posiciones de n [36]. Dada las permutaciones S y T la distancia Hamming es dada por la Ecuación 3.1.

$$d_{em}(S,T) = \sum_{i=1}^{n} x_i, \text{donde } x_i = \begin{cases} 0 & \text{si } S(i) = T(i) \\ 1 & \text{en otro caso} \end{cases} \tag{3.1}$$

Martí *et al.* [33] propusieron una métrica de distancia entre dos permutaciones, que en este trabajo es llamada **distancia de desviación**. Esta distancia es obtenida como la suma

Algoritmo 6 Pseudocódigo del algoritmo SS para LOP [7]

entrada: Elegir una condición de paro;
salida : La mejor solución;

```
1  inicio
2      P ← Método de Diversidad();
3      RefSet ← LS1(Generación de Calidad RefSet(P));
4      RefSet ← {RefSet, LS2(Generación de Diversidad RefSet(P))};
5      repetir
6          si |P| < Tamaño Mínimo entonces
7              P ← {P, Método de Diversidad()};
8          fin
9          repetir
10             actualización RefSet ← 0;
11             alea1 ← Número Aleatorio();
12             alea2 ← Número Aleatorio Basado en Ruleta(RefSet);
13             sol ← LS1(Método de Combinación(RefSet(alea1), RefSet(alea2)));
14             si Calidad(sol) > Calidad(Peor Solución (RefSet)) entonces
15                 RefSet ← {sol, {RefSet−Peor Solución(RefSet)}};
16                 actualización RefSet = 1;
17             fin
18         hasta que actualización RefSet = 1;
19         RefSet ← Eliminar Excepto el Mejor(RefSet);
20         RefSet ← {RefSet, LS2(Generación de Diversidad RefSet(P))};
21     hasta que la condición de paro se cumple;
22     regresa La mejor solución encontrada;
23 fin
```

de las diferencias absolutas de las posiciones de sus elementos. Dadas las permutaciones S y T, tal que $|S| = |T|$, la distancia entre S y T es dada por la Ecuación 3.2.

$$d_{dev}(S,T) = \sum_{i=1}^{n} |S_i - T_i| \tag{3.2}$$

Pantrigo *et al.* [28] propusieron una métrica de distancia para el problema Cutwidth, en este trabajo es nombrada como **distancia de desviación modificada**. Esta métrica calcula la distancia entre dos permutaciones dadas de la misma dimensión R y S (ver Ecuación 3.3), midiendo la distancia absoluta entre las posiciones de los elementos en ambas permutaciones y la diferencia absoluta entre posiciones de los elementos en R y los elementos en la permutación S en orden inverso.

$$d_{mdev}(R,S) = \text{mín} \left\{ \sum_{i=1}^{n} |r_i - s_i|, \sum_{i=1}^{n} |r_i - s_{n-i+1}| \right\} \tag{3.3}$$

Distancia entre una permutación y un conjunto de permutaciones. Sea Q_i una permutación y $R = \{R_1, R_2, R_3, ...R_n\}$ un conjunto de n permutaciones. La mínima distancia entre Q_i y R, $d_{min}(Q_i, R)$ es la mínima distancia entre Q_i y todas las permutaciones en R, lo anterior se expresa con la Ecuación 3.4.

$$d_{min}(Q_i, R) = \text{ mín } (d(Q_i, R_j), j = 1, ..., n) \tag{3.4}$$

3.1.4 Métricas de diversidad

Las **métricas IDIV e IDISIM** permiten medir la diversidad en los conjuntos relacionados de permutaciones; IDIV considera la diversidad que una permutación Q_i provee al conjunto de permutaciones en R e IDISIM evalua la disimilitud entre dos conjuntos de permutaciones [39].

La **métrica IDIV** calcula la diversidad que la permutación Q_i contribuya al conjunto R de permutaciones; este es obtenido como el promedio de la distancia entre Q_i y las permutaciones

en R, este cálculo es representado por la Ecuación 3.5.

$$IDIV(Q_i) = \frac{\left(\sum_{j=1}^{n} d(Q_i, R_j)\right)}{n} \tag{3.5}$$

La **métrica IDISIM** evalua la disimilaridad entre Q y R, siendo estos dos conjuntos de permutaciones dados (ver Ecuación 3.6).

$$IDISIM(Q, R) = \text{ máx}\{d_{mdev}(Q_i, R); \text{para } i = 1, 2, \ldots, n\} \tag{3.6}$$

3.1.5 Estrategias de diversificación en el conjunto de referencia

El conjunto de referencia $RefSet$ es responsable de mantener la diversidad durante la búsqueda en el algoritmo de búsqueda dispersa. Para mantener diversidad en este conjunto, es necesario utilizar métricas $descriptoras$ o de diversidad para medir de alguna forma el nivel de cercanía, o en su defecto la distancia entre las soluciones. Por lo anterior, se evaluaron diferentes estrategias para formar parte de los elementos diversificados que se integrarán en el conjunto de referencia.

Estrategia DivA aplica la métrica de desviación modificada (d_{mdev}). A continuación se describe el procedimiento para implementar esta estrategia.

1. Calcula la suma de las distancias entre cada elemento en P y los elementos ya incluidos en el conjunto $RefSet$, como se experesa en la Ecuación 3.7.

$$d_{sum}(P_i) = \sum_{j=1}^{|RefSet|} d_{mdev(P_i,R_j)}, i = 1, 2, \ldots, |P| \tag{3.7}$$

2. Seleccionar de P el elemento P_i que obtiene la mayor distancia d_{sum}.

3. Incorpora P_i a R y lo elimina de P, $R = R \cup P_i$ y $P = P/P_i$.

4. Repetir los paso 2 y 3 para seleccionar las $b/2$ soluciones que serán incorporadas en el conjunto $RefSet$.

Estrategia Div1 utiliza una selección por lotes, la métrica de distancia de desviación modificada y la métrica de diversidad ($IDIV$). Se lleva a cabo el siguiente procedimiento para eligir $b/2$ soluciones de P.

1. Para cada $P_i \in P$: calcular la distancia d_{dev} de P_i a los elementos en R. Posteriormente, calcular la métrica de diversificación $IDIV$ entre P_i y los elementos en R, esto se almacena en un vector llamado $VIDIV$ y crea un índice $IndexIDIV$.

2. Ordena los elementos de $VIDIV$ junto con el índice $IndexIDIV$.

3. Elige los primeros $b/2$ elementos del vector $VIDIV$, los incorporar al conjunto $RefSet$ y los elimina del conjunto P.

Estrategia Div es una estrategia de selección de disimilitud. Esta estrategia utiliza la distancia de desviación modificada d_{dev}, la mínima distancia d_{min} y la estrategia de disimilitud $IDISIM$; la estrategia de selección contiene los siguientes pasos:

1. Para cada $P_i \in P$. Calcula la métrica de disimilitud $IDISIM$ de $\{Pi \cup RefSet\}$. Posteriormente, selecciona el elemento P_i con mayor $IDISIM$.

2. Incorpora el elemento P_i elegido en $RefSet$, depués lo elimina de P.

3. Repite los pasos 1 y 2 hasta seleccionar de P el resto de los $(b/2) - 1$ elementos para ser incorporados en el conjunto $RefSet$.

Estrategia DIV2 utiliza la distancia de desviación modificada y la métrica de diversidad $IDIV$. Se lleva a cabo el siguiente procedimiento para implementar esta estrategia.

1. Para cada $P_i \in P$. Calcula la métrica de diversidad $IDIV$ de $\{P_i \cup RefSet\}$, después selecciona el elemento P_i de P con mayor distacia $IDIV$.

2. Incorpora el elemento P_i en el conjunto $RefSet$.

3. Repite el paso 1 y 2 hasta elegir de P los $(b/2) - 1$ elementos restantes para agregarlos al conjunto $RefSet$.

3.2 Resumen

Se incorporaron en el algoritmo de búsqueda dispersa estrategias de diversificación para la generación del conjunto de referencia, dos métodos de busqueda local con diferentes niveles de intensificación y un proceso de combinación. El centro de este enfoque consiste en mantener un balance entre intensificación y diversificación en el conjunto de referencia para lograr un mejor rendiemiento en el algoritmo de búsqueda dispera.

4

Diseño experimental

En este capítulo se describen las instancias de prueba que fueron tomadas de la literatura y usadas para la fase de experimentación. También se detallan las condiciones experimentales utilizadas para el estudio del balance entre intensificación y diversificación en el conjunto de referencia. Se presentan tres estudios experimentales enfocados a evaluar la metodología para medir el balance de intensificación y diversificación que se propone en este trabajo, así como a medir el impacto que el balance de intensificación y diversificación tiene sobre el desempeño del algoritmo búsqueda dispersa para LOP.

4.1 Instancias de prueba

En la literatura especializada se encuentran instancias de dominio público con las que se han probado propuestas de solución para LOP, como son: instancias XLOLIB, RandomA1, RandomA2, RandB y Special. En este trabajo de investigación se utilizó un subconjunto de prueba que involucra las instancias mencionadas con anterioridad. A continuación se describen los conjuntos de instancias utilizados.

- El conjunto de prueba propuesto por Schiavinotto y Stützle [35], fueron generadas de tablas de $entrada$-$salida$ del mundo real mediante la replica de problemas grandes. La distribución de los números en estas instancias refleja de alguna manera las tablas de $entrada$-$salida$ reales. Estas instancias han sido llamadas *XLOLIB* y contiene instancias de dos tamaños $n = 150$ y $n = 250$. El conjunto estándar contiene 98 instancias, 49 son de tamaño 150 y 49 de tamaño 250. Posteriormente fueron eliminadas 20 instancias porque la suma de sus entradas no es representada como entero de 4-byte. Para esta investigación se utilizó un subconjunto de 10 instancias del conjunto en cuestión: N-t59n11xx_150, N-tiw56r67_150, N-t70f11xx_150, N-t75e11xx_150, N-be75eec_150, N-t59b11xx_250, N-t70d11xx_250, N-t75e11xx_250, N-tiw56r54_250 y N-be75eec_250.

- Las instancias Random de tipo A fueron propuestas por Martí [18], las cuales son compuestas de 100 problemas aleatorios que han sido extensamente utilizados. Los problemas de tipo 1 (llamados *RandomA1*) son generados con una distribución uniforme. El tamaño de las instancias varia entre $n = 100$, 150, 200 y 500, existen 25 instancias en cada $n \in \{100, 150, 200, 500\}$, por lo cual suman en total 100 instancias. De este conjunto se consideraron 10 instancias, las cuales son las siguientes: N-t1d100.04, N-t1d100.12, N-t1d100.20, N-t1d150.07, N-t1d150.22, N-t1d200.07, N-t1d200.20, N-t1d200.25, N-t1d500.08 y N-t1d500.25.

- Las instancias de tipo 2 fueron propuestos por Martí [18], son generadas aleatoriamente y las llamaron *RandomA2*. Los problemas se generan al contar el número de veces que aparece un sector en una posición más alta con respecto a otro en un conjunto de permutaciones generadas aleatoriamente. El conjunto está compuesto por 75 instancias, 25 de tamaño 100, 150 y 200, respectivamente. De este conjunto las instancias utilizadas para la experimentación fueron: N-t2d150.10, N-t2d150.20, N-t2d200.03 y N-t2d200.21.

- El conjunto de instancias *RandomB* fue generado con matrices enteras donde la superdiagonal se dibuja de manera uniforme en el intervalo $[0, U_1]$ y las entradas de la subdiagonal en $[0, U_2]$, donde $U_1 \geq U_2$. La dificultad es afectada por la diferencia de

U_1-U_2 [23]. El tamaño del conjunto es de 20 instancias, de las cuales fueron utilizadas sólo 3; N-p50-01, N-p50-09 y N-p50-18.

- El conjunto de prueba *Special* está compuesto por 6 instancias que fueron utilizadas en experimentos publicados en el estado del arte, un subconjunto de tres instancias fueron elegidas para la experimentación [23]: N-atp111, N-atp163 y N-pal31.

4.2 Condiciones experimentales

El algoritmo SS fue escrito en el lenguaje de programación C++ y compilado en Visual Studio 6. El algoritmo fue ejecutado una vez, utilizando un tiempo límite 10 segundos en una computadora personal Toshiba con procesador Centrino, a 1.8 GHz, 80 GB DD y 1 GB en memoria RAM con sistema operativo Windows, versión XP Profesional.

4.3 Estudio del balance en el conjunto de referencia

Se realizaron tres experimentos que ayudaron a evaluar la metodología para medir el balance entre intensificación y diversificación y su aportación al desempeño del algoritmo de búsqueda dispersa para LOP.

Se aplican cinco estrategias para construir el conjunto de referencia. Estas estrategias tienen como finalidad incorporar diferentes niveles de diversificación en el conjunto de referencia. La nomenclatura utilizada para nombrar la estrategia en cada experimento está compuesta de la siguiente forma: la primera es el nombre de la estrategia utilizada para generar el conjunto de referencia y la segunda es el nombre de la estrategia utilizada para actualizar el conjunto de referencia, ambas separadas por un guión (-). Las estrategias utilizadas son descritas en la Subsección 3.1.5. Por ejemplo, la nomenclatura Div-Div utilizada en los experimentos, indica que se utilizó la estrategia Div para generar el conjunto de referencia y la estrategia Div para actualizar el conjunto de referencia.

4.3.1 Experimento 1: Balance entre diversificación e intensificación

Evaluación de la metodología propuesta para medir el balance entre diversificación e intensificación en el primer experimento se estudia el balance que se logra con los diferentes métodos de construcción del conjunto $RefSet$ (descritos en la Subsección 3.1.5), utilizando la metodología propuesta. En este experimento se registró el valor de la mejor solución encontrada y el promedio de las soluciones obtenidas con cada estrategia propuesta a lo largo de las iteraciones del algoritmo obteniéndose las correspondientes gráficas de estos dos indicadores a lo largo de los 10 segundos de ejecución del algoritmo SS.

Las Figuras 4.1 y 4.2, muestran ejemplos de las gráficas de resultados para la instancia N-tiw56r67_150 (del conjunto XLOLIB). Se puede ver claramente que las estrategias de construcción del conjunto de referencia producen diferencias significativas en las características de desempeño. A partir de estas diferencias en las características se obtienen los indicadores propuestos con el fin de comparar el tipo de comportamiento de las estrategias en el algoritmo de SS; ya sea intensificador ó diversificador.

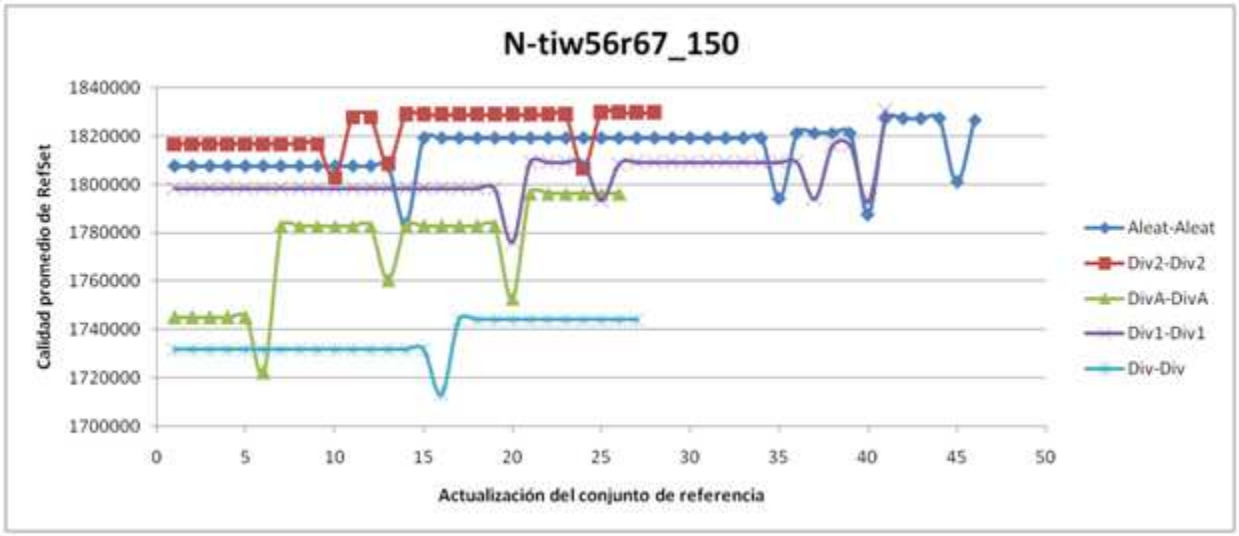


Figura 4.1: Promedio de la calidad de solución del conjunto $RefSet$ a lo largo de las iteraciones.

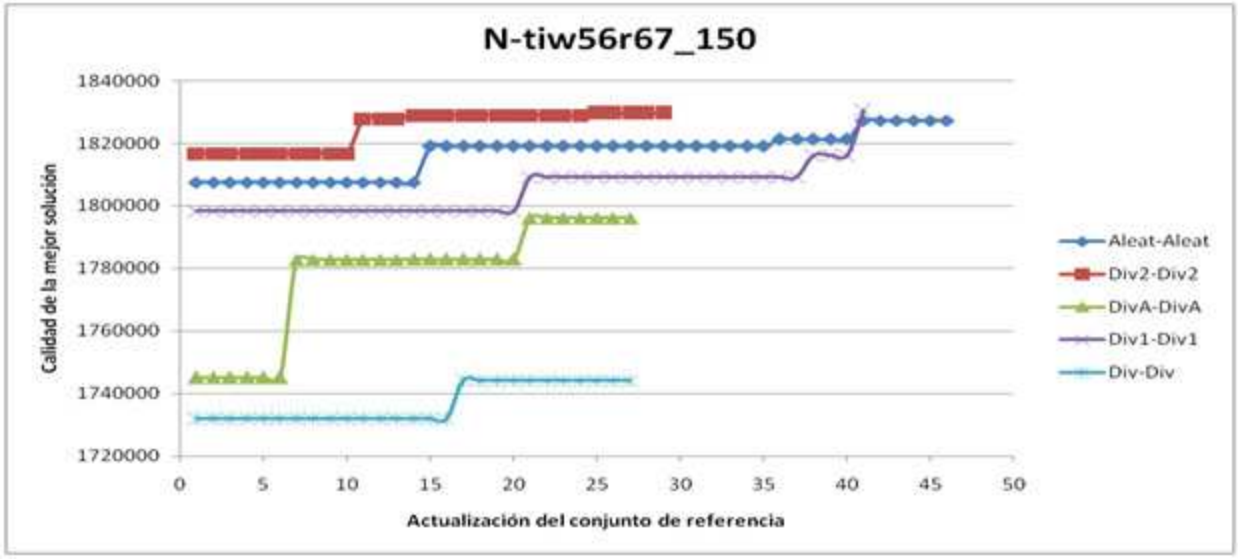


Figura 4.2: La mejor calidad de solución a lo largo de las iteraciones usando los diferentes indicadores.

En la primera columna de la Tabla 4.1 se describen los indicadores propuestos, la segunda columna tipifica el tipo de tarea que evalúa el indicador (diversificación o intensificación) y las siguientes columnas muestran los valores normalizados de los indicadores para las diferentes estrategias utilizadas en el conjunto $RefSet$. Estos valores representan de alguna manera el grado de intensificación y diversificación que promueve cada estrategia utilizada para construir el conjunto. Los indicadores 2 y 3 que miden la capacidad diversificadora se extraen de la gráfica que se muestra en la Figura 4.1, el resto de los indicadores se obtienen de la gráfica que se muestra en la Figura 4.2, estos indicadores se enfocan a medir la capacidad intensificadora de cada estrategia.

Tabla 4.1: Muestra los valores normalizados para los indicadores utilizados.

N-tiw56r67_150						
Indicadores	Capacidad	Aleat-Aleat	Div2-Div2	DivA-DivA	Div1-Div1	Div-Div
1) Desempeño de cada estrategia con respecto a la aleatoria.	Intensificación	2	3	1	1	1
2) Número de actualizaciones realizadas al $RefSet$.	Diversificación	3	1	1	3	1
3) Número de puntos de reducción en la calidad promedio de $RefSet$.	Diversificación	3	3	3	3	1
4) Calidad de solución relativa de la mejor solución encontrada con respecto a las demás estrategias.	Intensificación	2	3	1	3	1
5) Número de veces que la mejor solución es mejorada.	Intensificación	3	3	2	3	1

La forma de interpretar las graficas de resultados que se muestran en las Figuras 4.1 y 4.2 para los indicadores es descrita con mayor detalle a continuación: para obtener el valor del indicador 1 se compara la calidad de cada estrategia con respecto a la estrategia aleatoria, si su calidad siempre es superior se le asigna el valor de 4, 2 si es igual, 3 si es variable y 1 si el desempeño es peor. Cabe señalar que como la estrategia aleatoria se compara consigo misma se le asigna el valor de 2. El indicador 2 se obtiene considerando el número de actualizaciones al conjunto de referencia, es decir, el número de veces en el que hubo una actualización en el conjunto de referencia a lo largo de la ejecución del algoritmo. El indicador 3 es el número de puntos donde se obtuvo una reducción en la calidad promedio del conjunto de referencia; se puede apreciar en la gráficamente como el número de picos inferiores que se muestran en la gráfica de la calidad promedio del conjunto de referencia. Para obtener el indicador 4 las estrategias se ordenan con respecto a su calidad de solución de mayor a menor, asignandoles

un valor de forma decremental donde el valor 5 es para la estrategia con mayor desempeño y a la de menor desempeño se asigna el valor 1; el indicador 5 se relaciona con el estancamiento que puede tener el algoritmo, es decir, el número de veces que la mejor solución es mejorada, es fácil de identificar porque graficamente este comportamiento se observa como un escalón.

Como por ejemplo, utilizando la Figura 4.2, se determina el valor para el indicador 1 que representa el desempeño de cada estrategia con respecto a la aleatoria. Por lo tanto para la estrategia $Div2$-$Div2$ este indicador tiene un valor de cuatro, para $Aleat$-$Aleat$ es de dos, para $Div1$-$Div1$, $DivA$-$DivA$ y Div-Div es de uno.

El indicador 5 representa el número de veces que la mejor solución es superada se determina utilizando la Figura 4.2, en la cual se contabilizan los "escalones" para cada estrategia (se presenta cuando hay un incremento en la mejor solución). Por ejemplo la estrategia Div-Div obtiene un valor de uno para este indicador, las estrategias $Div1$-$Div1$ y $DivA$-$DivA$ tiene dos "escalones" por tanto su valor es de dos y así sucesivamente. Los valores de los indicadores fueron normalizados tomando un rango de uno a tres, la Tabla 4.1, muestra un ejemplo de los resultados normalizados para la instancia N-tiw56r67_150.

El balance entre diversificación e intensificación para cada estrategia se calcula sumando para todas las instancias los indicadores de diversificación por un lado y los indicadores de intensificación por otro lado. El balance entre la intencificasion y diversificación se obtiene mediante la división de sus correspondientes sumatorias.

La Tabla 4.2, muestra los resultados de los cálculos antes mencionados, donde la primer columna contiene el nombre de la estrategia, la segunda es la suma de los indicadores de diversificación, la tercera es la suma de los indicadores de intensificación y la última presenta el resultado de la división de la segunda columna entre la tercera. Con base en los resultados obtenidos, las estrategias $Div2$-$Div2$ y $DivA$-$DivA$ tendrían un mejor balance ya que obtienen un valor más cercano a uno, por otro lado las estrategias $Aleat$-$Aleat$ y $Div1$-$Div1$ son las que presentan un menor balance tendiente a diversificación.

La Figura 4.3, contiene gráficas que muestran la relación entre intensificación y diversificación para las cinco estrategias evaluadas para las 30 instancias; en el eje x se encuentran

Tabla 4.2: Medida del balance entre intensificación y diversificación.

Estrategias	Sumas		Resultado
	Diversificación	Intensificación	
Aleat-Aleat	226	129	1.7519
Div2-Div2	**138**	**114**	**1.2105**
DivA-DivA	**143**	**119**	**1.2017**
Div1-Div1	229	127	1.8031
Div-Div	143	105	1.3619

representadas por un número las instancias. El indicador total de intensificación se muestra en color rojo y representa la sumatoria de los indicadores de intensificación mencionados, mientras que el indicador total de diversificación en azul corresponde a la sumatoria de los indicadores de diversificación (ver Tabla 4.1). El mejor balance entre estos dos indicadores (diversificación e intensificación) se representaría en un comportamiento igual entre ellos, es decir, que se empalman en todas las instancias.

Las Figuras 4.3b y 4.3c corresponden a las estrategias $Div2$-$Div2$ y $DivA$-$DivA$ y son las que presentan el mejor balance de intensificación/diversificación, ya que ambas gráficas tienen un comportamiento muy similar casi empalmados. La estrategia aleatoria (ver Figura 4.3a), presenta el peor balance porque presenta una mayor distancia entre las gráficas correspondientes a estos dos indicadores totales (ver Figura 4.3b), mismo tipo de comportamiento también se presenta en la estrategia $Div1$-$Div1$ (ver Figura 4.3d, donde se ve claramente que el predominio es mayor por la diversificación que por la intensificación. En cambio Div-Div obtiene un buen balance solo para algunas instancias como muestra la Figura 4.3e.

4.3.2 Experimento 2: Medición de la diversificación del conjunto de referencia

Este experimento se realizó para evaluar los diferentes niveles de diversificación producidos por las distintas estrategias de construcción de $RefSet$ de una forma mas directa. Para esto, se calcula la distancia entre todos las parejas de soluciones que se pueden obtener. Se aplica la métrica de distancia entre permutaciones propuesta en [33] y se obtiene su promedio para cada una de las estrategias de construcción.

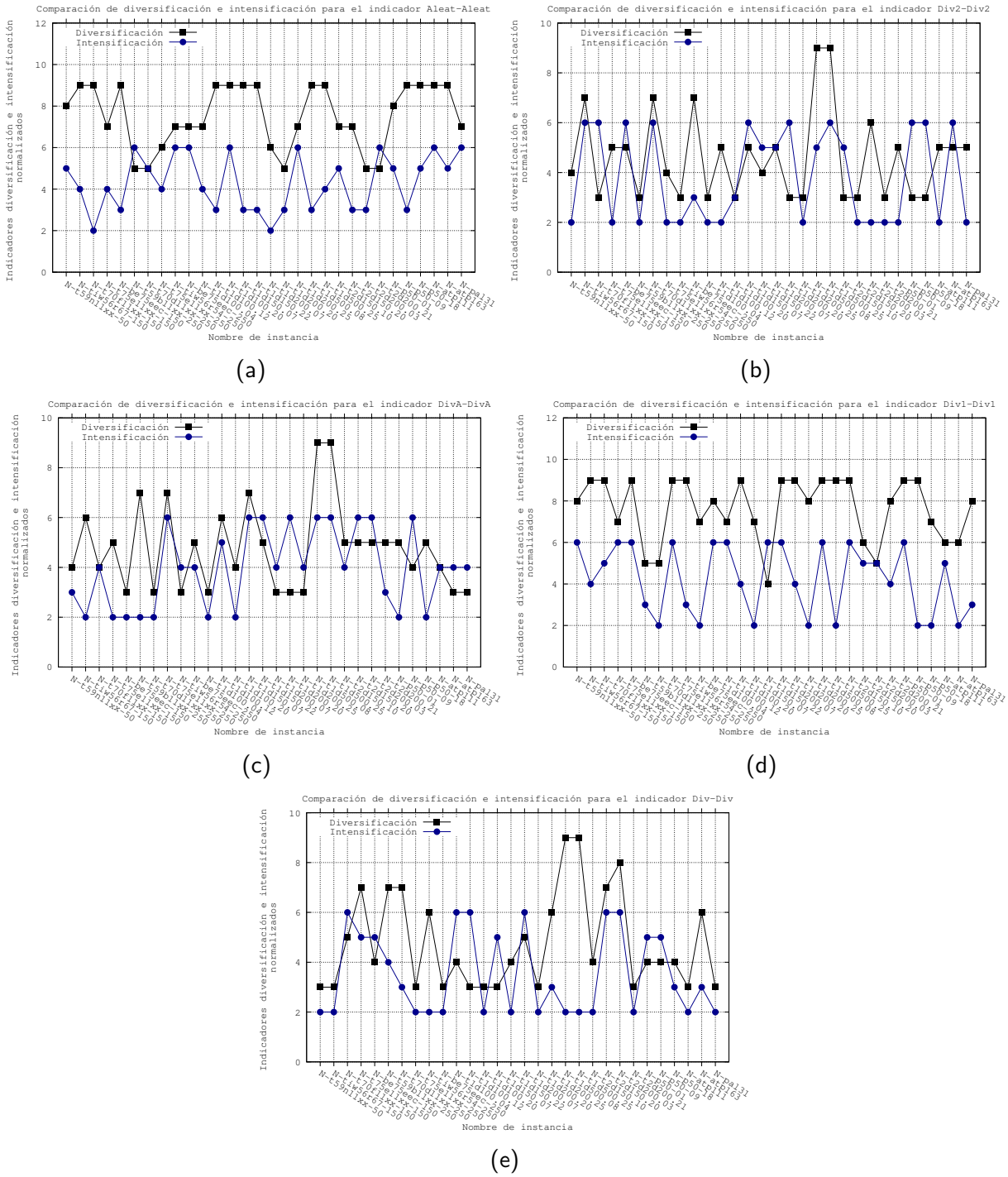


Figura 4.3: Comparación entre intensificación y diversificación para un conjunto de 30 instancias. a) estrategia aleatoria, b) estrategia $Div2$-$Div2$, c) estrategia $DivA$-$DivA$, d) estrategia $Div1$-$Div1$ y e) estrategia Div-Div.

La Figura 4.4 muestra para cada instancia, el valor promedio de la instancia en el conjunto de referencia para las cinco estrategias de diversidad; se ve claramente en dicha figura que las distancias promedio para las estrategias $Aleat$-$Aleat$, $DivA$-$DivA$ y $Div1$-$Div1$ son las mayores, presentándose este comportamiento de forma similar para todo el conjunto de instancias de prueba; $Div2$-$Div2$ y Div-Div presentan menor valor de la distancia promedio con respecto a las demás estrategias. También se puede ver en la figura que hay tres niveles de diversidad en todas las instancias, en el nivel intermedio se encuentra la estrategia $Div2$-$Div2$, lo cual indica que el promedio de las distancias en el conjunto de referencia no es muy grande ni muy pequeño en comparación con las demás estrategias, por lo tanto esta estrategia produce no promueve tanta diversidad en las soluciones de $RefSet$ como $Aleat$-$Aleat$ pero tampoco produce tan poca diversidad como Div-Div.

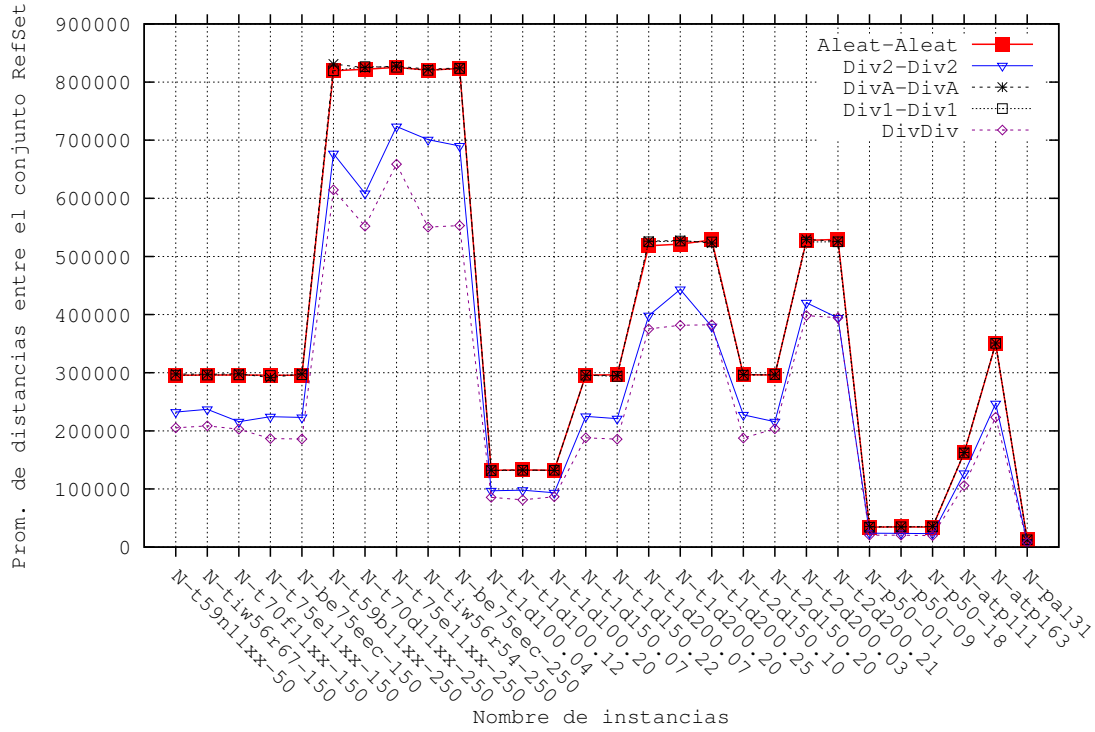


Figura 4.4: Distancias entre pares de soluciones en el conjunto de referencia

4.3.3 Experimento 3: Desempeño del algoritmo SS

Se realizó un experimento con el principal propósito de evaluar el desempeño del algoritmo de búsqueda dispersa con las estrategias de diversificación. El experimento consistió de una sola corrida con un tiempo límite de 10 segundos para las instancias RandomA1 para cada combinación de estrategias para la generación y actualización del conjunto de referencia.

La Tabla 4.3, describe el resultado experimental para el conjunto de instancias RandomA1; la primer columna contiene la combinación de estrategias de diversificación usada en cada corrida del algoritmo; la primer estrategia es usada para la generación del conjunto de referencia y la segunda para la actualización del conjunto $RefSet$ ambas separadas por un guión (-). La segunda columna reporta el porcentaje promedio de error con respecto a la mejor solución conocida y la última indica el número de veces que fue obtenida la mejor solución conocida.

Tabla 4.3: Resultados experimentales para el conjunto de instancias RandomA1 con un tiempo de ejecución de 10 segundos.

Estrategia de diversificación	Prom. del error del mejor conocido ($\%$)	$\#$ la mejor solución
DivA-DivA	**0.17803879**	**20**
Div-Div	0.17221362	16
Div1-Div1	0.16602106	17
Div1-Div	**0.15919057**	**18**
Div-Div1	0.16191157	12
Div1-Div2	0.16923719	15
Div2-Div1	0.16473085	16
Div1-DivA	0.16926965	13
DivA-Div1	**0.16104983**	**18**
Div2-Div2	**0.13150481**	**22**

En la Tabla 4.3, se observa que las estrategias que obtuvieron el mejor compromiso entre un menor porcentaje de error y un mayor número de veces que se obtuvo la mejor solución conocida, fueron: $Div1$-Div, $DivA$-$DivA$, $DivA$-$Div1$ y $Div2$-$Div2$. Sin embargo, de todas

ellas la que obtuvo el mejor compromiso entre capacidad intensificadora; representada por el porcentaje de error promedio (0.13150481) y capacidad diversificadora; representada por el numero de veces en que se encuentra encuentra la mejor solución conocida (22) es la combinación ($Div2$-$Div2$).

De acuerdo a los resultados obtenidos en los tres experimentos realizados se puede aseverar que la estrategia $Div2$-$Div2$ presenta un buen balance de intensificación y diversificación.

4.4 Resumen

En este capítulo se presentaron los diferentes conjuntos de instancias utilizados para probar el desempeño del algoritmos SS, también se describió a detalle los tres experimentos que fueron realizados para estudiar el balance entre intensificación y diversificación del conjunto de referencia utilizando la metodología propuesta; el estudio experimental ayudó a identificar la mejor estrategia que obtiene el mejor balance en el conjunto $RefSet$.

5

Conclusiones y trabajo futuro

En este capítulo se presentan las conclusiones obtenidas a partir del análisis de la implementación y los resultados de la experimentación realizada. Además, se describe el trabajo futuro que dará continuidad a este trabajo de investigación.

5.1 Conclusiones

Esta investigación se realiza un estudio del balance de la intensificación y diversificación en una búsqueda dispersa. El caso de estudio se integra con cinco estrategias para la construcción y actualización del conjunto de referencia de SS y se propone una metodología para medir el balance entre intensificación y diversificación. Por otro lado, se observó que la estrategia que obtiene mayor balance de acuerdo a esta metodología produce el mejor desempeño en cuanto a calidad de solución utilizando una forma de evaluación clásica del desempeño de la metaheurística.

La metodología propuesta calcula el balance de diversificación e intensificación como el cociente entre la sumatoria de los indicadores de diversificación y la sumatoria de los indicadores de intensificación. De manera que los valores cercanos a uno indican un mejor balance, valores

mayores a uno representan un predominio de diversificación y valores menores a uno indican predominio de intensificación.

Las sumatorias de los indicadores de intensificación y diversificación para cada estrategia es comparada, de manera que las estrategias que obtienen valores similares entre ambas sumatorias tienen un mejor balance, mientras que valores diferentes indican que existe un predominio de algún comportamiento y por lo tanto un bajo balance.

Por otro lado, se mide la distancia del conjunto de referencia (*RefSet*, ver Subsección 3.1.1) para cada una de las estrategias de construcción calculando el promedio de las distancias entre todas las soluciones contenidas en el conjunto (se utilizó la distancia de desviación propuesta en [33]).

Se observó que la estrategia de construcción del conjunto de referencia *RefSet* que obtiene mayor distancia entre las soluciones es $Aleat$ (ver Subsección 3.1.5 y 4.3), y se observa que las estrategias obtienen diferentes valores de distancia promedio entre las soluciones del conjunto de referencia. Se identifican tres diferentes niveles; $Aleat$, $Div1$ y Div (ver Subsección 3.1.5 y 4.3) obtienen mayor distancia entre las soluciones mientras que, $Div2$ y Div obtienen menor distancia.

Finalmente se realizó un experimento con la combinación de pares de estrategias la primera para la generación del conjunto de referencia y la segunda para la actualización del mismo, con el fin de encontrar la combinación que produzca un mayor desempeño en el algoritmo SS en cuanto al error promedio de la calidad de la solución y al número de veces que la mejor solución es encuentrada. En este experimento la estrategia $Div2$ obtiene el mejor compromiso entre intensificación y diverisificación ya que obtiene tanto un bajo porcentaje de error promedio respecto a la mejor solución conocida como un alto número de mejores soluciones conocidas. Esto resultados indican que la estrategia $Div2$-$Div2$ presenta el balance más adecuado de intensificación y diversificación entre todas las combinaciones evaluadas.

Considerado los resultados de los tres experimentos en los cuales se obtiene, que la estrategia $Div2 - Div2$ presenta un balance equilibrado de intensificación y diversificación de acuerdo a la metodología propuesta, y por otro lado que esta misma estrategia obtiene un nivel

medio en la distancia promedio que guardan entre si las soluciones del conjunto de referencia y además esta estrategia muestra el balance más adecuado ya que obtiene el mayor compromiso entre el porcentaje de error promedio y el número de mejores soluciones obtenidas de acuerdo al experimento tres. Considerado estos resultados es posible inferir que un comportamiento equilibrado de intensificación y diversificación representa el balance más adecuado para la solución de LOP utilizando SS con base en la metodología para la medición del balance propuesta en este trabajo.

5.2 Trabajo futuro

Si bien la metodología planteada en este trabajo de investigación permitió identificar la estrategia que provee el balance más adecuado entre intensificación y diversificación para obtener el mejor desempeño de la metaheurística, y por otro lado permitió establecer que el balance más adecuado está relacionado con el equilibrio de intensificación y diversificación que presenta la metaheurística, aún es posible enriquecer algunos aspectos y resultados.

1. Diseñar nuevas estrategias para producir diferentes niveles de diversificación en otros procesos de SS tales como: el proceso de combinación, ó que produzcan diferentes niveles de intensificación en procesos como el de mejora, con el fin de medir el balance resultante aplicando la metodología propuesta en este trabajo.

2. Automitizar mediante una herramienta la obtención de los indicadores propuestos en este trabajo.

3. Evaluar la metodología propuesta aplicandola en otras metaheurísticas.

4. Obtener los indicadores en tiempo de ejecución con el fin de ajustar parámetros de la metaheurística de acuerdo al balance ontenido.

5. Aplicar la metodología propuesta a otros problemas con el fin de obtener una referencia del balance más adecuado para cada problema.

Bibliografía

[1] EH Emile HL Aarts and Jan K Lenstra. *Local search in combinatorial optimization*. Princeton University Press, 1997.

[2] Christian Blum and Andrea Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys (CSUR)*, 35(3):268–308, 2003.

[3] Vicente Campos, Fred Glover, Manuel Laguna, and Rafael Martí. An experimental evaluation of a scatter search for the linear ordering problem. *Journal of Global Optimization*, 21(4):397–414, 2001.

[4] Vicente Campos, Manuel Laguna, and Rafael Martí. Context-independent scatter and tabu search for permutation problems. *INFORMS Journal on Computing*, 17(1):111–122, 2005.

[5] Richard K Congram. *Polynomially searchable exponential neighbourhoods for sequencing problems in combinatorial optimisation*. PhD thesis, University of Southampton, 2000.

[6] A Duarte, JJ Pantrigo, and M Gallego. Metaheurísticas. *Madrid, España, Dykinson*, 2007.

[7] T. El-Ghazali. Metaheuristics: from design to implementation. *Jonh Wiley and Sons Inc., Chichester*, 2009.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[9] Michael R Gary and David S Johnson. Computers and intractability: A guide to the theory of np-completeness, 1979.

[10] Michel Gendreau. *Handbook of metaheuristics*, volume 146. Springer, 2010.

[11] Fred Glover. Heuristics for integer programming using surrogate constraints. *Decision Sciences*, 8(1):156–166, 1977.

[12] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. A cutting plane algorithm for the linear ordering problem. *Operations research*, 32(6):1195–1220, 1984.

[13] P. Hansen, N. Mladenović, and J.A. Moreno Pérez. Variable neighbourhood search: methods and applications. *Annals of Operations Research*, 175(1):367–407, 2010.

[14] Pierre Hansen and Nenad Mladenović. Variable neighborhood search: Principles and applications. *European journal of operational research*, 130(3):449–467, 2001.

[15] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 1972.

[16] S. Kirkpatrick, MP Vecchi, et al. Optimization by simmulated annealing. *science*, 220(4598):671–680, 1983.

[17] Bernard Kolman, Robert C. Busby, and Sharon Cutler Ross. *Discrete Mathematical Structures*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 4th edition, 1999.

[18] Manuel Laguna, Rafael Marti, and Vicente Campos. Intensification and diversification with elite tabu search solutions for the linear ordering problem. *Computers & Operations Research*, 26(12):1217–1230, 1999.

[19] Wassily Leontiev. *Input output economics*. Oxford University Press, 1986.

[20] Shou-Chih Lo and Arbee LP Chen. Efficient index and data allocation for wireless broadcast services. *Data & Knowledge Engineering*, 60(1):235–255, 2007.

[21] S. Luke. *Essentials of Metaheuristics*. Lulu, 2009. Available for free at http://cs.gmu.edu/∼sean/book/metaheuristics/.

[22] Rafael Martí and Gerhard Reinelt. *The linear ordering problem: exact and heuristic methods in combinatorial optimization*, volume 175. Springer, 2011.

[23] Rafael Martí, Gerhard Reinelt, and Abraham Duarte. A benchmark library and a comparison of heuristic methods for the linear ordering problem. *Computational optimization and applications*, 51(3):1297–1317, 2012.

[24] Alfonsas Misevičius and Bronislovas Kilda. Comparison of crossover operators for the quadratic assignment problem. *Information Technology and Control*, 34(2):109œ119, 2005.

[25] N. Mladenović and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100, 1997.

[26] Pablo Moscato and Carlos Cotta. A gentle introduction to memetic algorithms. *Handbook of metaheuristics*, pages 105–144, 2003.

[27] Ibrahim H Osman and James P Kelly. *Meta-heuristics: theory and applications*. Springer, 1996.

[28] Juan J Pantrigo, Rafael Martí, Abraham Duarte, and Eduardo G Pardo. Scatter search for the cutwidth minimization problem. *Annals of Operations Research*, 199(1):285–304, 2012.

[29] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Dover Publications, 1998.

[30] A. Rodriguez-Cristerna and J. Torres-Jimenez. A simulated annealing with variable neighborhood search approach to construct mixed covering arrays. *Electronic Notes in Discrete Mathematics*, 39:249–256, 2012.

[31] V Roshanaei, MM Seyyed Esfehani, and M Zandieh. Integrating non-preemptive open shops scheduling with sequence-dependent setup times using advanced metaheuristics. *Expert systems with applications*, 37(1):259–266, 2010.

[32] Ilya Safro, Dorit Ron, and Achi Brandt. Multilevel algorithms for linear ordering problems. *Journal of Experimental Algorithmics (JEA)*, 13:4, 2009.

[33] Rafael Martí Sánchez and Manuel Laguna. Scatter search: Diseño básico y estrategias avanzadas. *Inteligencia artificial: Revista Iberoamericana de Inteligencia Artificial*, 7(19):123–130, 2003.

[34] Tommaso Schiavinotto and Thomas Stützle. Search space analysis of the linear ordering problem. In *Applications of Evolutionary Computing*, pages 322–333. Springer, 2003.

[35] Tommaso Schiavinotto and Thomas Stützle. The linear ordering problem: Instances, search space analysis and algorithms. *Journal of Mathematical Modelling and Algorithms*, 3(4):367–402, 2004.

[36] Marc Sevaux, Kenneth Sörensen, et al. Permutation distance measures for memetic algorithms with population management. In *Proceedings of 6th Metaheuristics International Conference (MIC'05)*. Citeseer, 2005.

[37] M. Sipser. *Introduction to the Theory of Computation*. International Thomson Publishing, 1996.

[38] Kenneth Sörensen and Marc Sevaux. Ma|pm: memetic algorithms with population management. *Computers & Operations Research*, 33(5):1214–1225, 2006.

[39] Castilla G. Valdez. *Metaheurísticas Aplicadas a la Solución del Problema de Ordenamiento Lineal*. PhD thesis, Instituto Tecnológico de Ciudad Madero, 2013.

[40] Manuel Vazquez and L Darrell Whitley. A hybrid genetic algorithm for the quadratic assignment problem. In *GECCO*, pages 135–142. Citeseer, 2000.