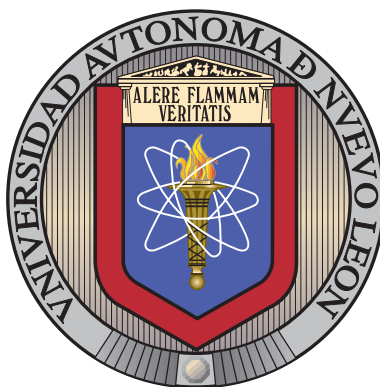


UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

DIVISIÓN DE ESTUDIOS DE POSGRADO



METAHEURÍSTICAS APLICADAS A LA  
PLANIFICACIÓN DE ORDEN PARCIAL

POR

ROSA LILIANA GONZÁLEZ ARREDONDO

EN OPCIÓN AL GRADO DE

MAESTRÍA EN CIENCIAS

EN INGENIERÍA DE SISTEMAS

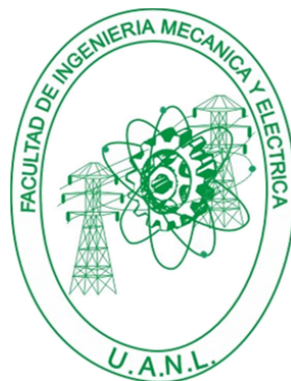
SAN NICOLÁS DE LOS GARZA, NUEVO LEÓN

MAYO 2014

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

DIVISIÓN DE ESTUDIOS DE POSGRADO



METAHEURÍSTICAS APLICADAS A LA  
PLANIFICACIÓN DE ORDEN PARCIAL

POR

ROSA LILIANA GONZÁLEZ ARREDONDO

EN OPCIÓN AL GRADO DE

MAESTRÍA EN CIENCIAS

EN INGENIERÍA DE SISTEMAS

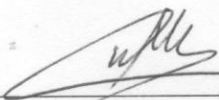
SAN NICOLÁS DE LOS GARZA, NUEVO LEÓN

MAYO 2014

**Universidad Autónoma de Nuevo León**  
**Facultad de Ingeniería Mecánica y Eléctrica**  
**División de Estudios de Posgrado**

Los miembros del Comité de Tesis recomendamos que la Tesis «Metaheurísticas aplicadas a la planificación de orden parcial», realizada por el alumno Rosa Liliana González Arredondo, con número de matrícula 1200123, sea aceptada para su defensa como opción al grado de Maestría en Ciencias en Ingeniería de Sistemas.

El Comité de Tesis



---

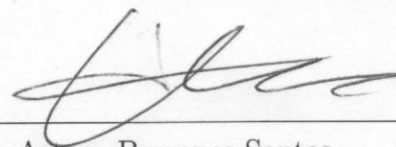
Dr. Romeo Sánchez Nigenda

Asesor

---

Dra. Lluvia Morales

Revisor



---

Dr. Arturo Berrones Santos

Revisor

Vo. Bo.

---

Dr. Simón Martínez Martínez

Subdirector de Estudios de Posgrado

*A mi esposo, padres y hermana.*

# ÍNDICE GENERAL

---

<b>Agradecimientos</b>	<b>x</b>
<b>Resumen</b>	<b>xi</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Descripción del problema . . . . .	1
1.2. Justificación . . . . .	3
1.3. Objetivos . . . . .	4
1.3.1. Objetivos Específicos . . . . .	4
1.4. Hipótesis . . . . .	5
1.5. Aportaciones científicas . . . . .	5
1.6. Metodología . . . . .	6
1.7. Estructura de la Tesis . . . . .	7
<b>2. Marco teórico</b>	<b>9</b>
2.1. Planificación . . . . .	9
2.1.1. Lenguaje de planificación PDDL . . . . .	10
2.1.2. Planificadores . . . . .	12

---

2.1.3. Dominio y problema . . . . .	13
2.2. Algoritmos de planificación . . . . .	17
2.2.1. Espacio de estados . . . . .	18
2.2.2. Espacio de planes . . . . .	20
<b>3. Formulación del problema y metodología de solución</b>	<b>34</b>
3.1. Introducción . . . . .	34
3.2. Planteamiento del problema . . . . .	35
3.3. Recocido simulado . . . . .	35
3.3.1. Movimientos de aceptación . . . . .	38
3.3.2. Esquema de enfriamiento . . . . .	39
3.4. Metodología de solución . . . . .	40
3.4.1. Planificadores desarrollados . . . . .	44
<b>4. Experimentos y Resultados</b>	<b>50</b>
4.1. Experimentación . . . . .	50
4.2. Resultados obtenidos . . . . .	53
<b>5. Conclusiones y trabajo futuro</b>	<b>59</b>
5.1. Conclusiones . . . . .	59
5.2. Trabajo futuro . . . . .	60
<b>A. Resultados: planificador y dominio</b>	<b>62</b>

# ÍNDICE DE FIGURAS

---

2.1. Problema . . . . .	12
2.2. Mundo de los bloques . . . . .	14
2.3. Espacio de estados . . . . .	19
2.4. Espacio de planes . . . . .	20
2.5. Total y parcial . . . . .	22
2.6. Mundo de los bloques parte1 . . . . .	26
2.7. Mundo de los bloques parte2 . . . . .	26
2.8. Mundo de los bloques parte3 . . . . .	27
2.9. Mundo de los bloques parte4 . . . . .	28
2.10. Mundo de los bloques parte5 . . . . .	29
2.11. Mundo de los bloques parte6 . . . . .	30
2.12. Mundo de los bloques parte7 . . . . .	31
2.13. Mundo de los bloques parte8 . . . . .	32
3.1. Recocido simulado . . . . .	38
3.2. Planes parciales . . . . .	48

---

3.3. Costo de planes . . . . .	48
4.1. Resultados POPSA1 por dominio . . . . .	54
4.2. Resultados POPSA2 por dominio . . . . .	55
4.3. Resultados POP SAM dominio . . . . .	55
4.4. Resultados planificadores . . . . .	56



# ÍNDICE DE TABLAS

---

4.1. Problemas y dominios . . . . .	57
4.2. Tabla de promedios temperatura y enfriamiento . . . . .	58
A.1. Resultados VHPOP . . . . .	63
A.2. Resultados POP-SA1 . . . . .	64
A.3. Resultados POP-SA2 . . . . .	65
A.4. Resultados POP-SAM . . . . .	66

# AGRADECIMIENTOS

---

Agradezco a mi esposo Eric Silva por el apoyo y motivación en todo momento desde el inicio de la maestría, por su comprensión, entusiasmo, confianza y por creer que soy capaz de terminar este proyecto.

A mis padres Carlos González y Rosa Arredondo por su apoyo incondicional, quienes me impulsan y me dan la motivación necesaria para seguir acumulando logros. A mi hermana por estar conmigo siempre.

A mi asesor de tesis Dr. Romeo Sánchez por su paciencia, apoyo, orientación y palabras de motivación que siempre me ofreció durante la trayectoria de la maestría.

Le agradezco al Dr. Arturo Berrones y la Dra. Lluvia Morales por haber formado parte de mi comité de tesis, por su apoyo y retroalimentación.

A mis compañeros de generación y de PISIS.

Agradezco a la Facultad de Ingeniería Mecánica y Eléctrica de la Universidad Autónoma de Nuevo León y por supuesto al Posgrado en Ingeniería de Sistemas.

Gracias al Consejo Nacional de Ciencia y Tecnología (CONACyT) por la beca otorgada para la realización de este trabajo de tesis.

# RESUMEN

---

Rosa Liliana González Arredondo.

Candidato para el grado de Maestro en Ciencias  
con especialidad en Ingeniería de Sistemas.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Título del estudio:

## METAHEURÍSTICAS APLICADAS A LA PLANIFICACIÓN DE ORDEN PARCIAL

Número de páginas: 71.

**OBJETIVOS Y MÉTODO DE ESTUDIO:** EL objetivo de la tesis consiste en incrementar la cobertura del espacio de soluciones del algoritmo de planificación de orden parcial, a través del uso de metaheurísticas. Se propone diseñar y adaptar el recorrido simulado al algoritmo de planificación de orden parcial. Se realiza en paralelo a este objetivo un análisis exhaustivo de los planificadores que se han desarrollado hoy en día bajo la metodología de orden parcial, así como la factibilidad de la implementación de metaheurísticas dentro del algoritmo, buscando como resultado orientar al planificador a la toma de decisiones para generar planes de solución a los problemas propuestos.

### Objetivos Específicos

1. Estudio de las características y funcionamiento general de los planificadores.
2. Modelación de problemas de planificación en PDDL.
3. Análisis de la metaheurística recocido simulado (simulated annealing), características con las que cuenta, determinando así su adaptabilidad para los fines requeridos del problema a trabajar.
4. Diseño y desarrollo de un algoritmo unificado que incluya las características de orden parcial y del recocido simulado.
5. Implementación en el planificador.
6. Generación de pruebas de comportamiento y funcionalidad.
7. Solución de problemas de planificación.
8. Analisis de resultados obtenidos.

CONTRIBUCIONES Y CONCLUSIONES: La implementación del recocido simulado en el planificador de orden parcial ha demostrado tener resultados satisfactorios ya que permite explorar a profundidad el espacio de soluciones presentado y explorar los distintos caminos que se generan para llevar a cabo la generación de un plan, logrando con ello dar solución a problemas que, de otra manera no sería posible resolver. La metodología de solución que lleva a cabo el algoritmo de orden parcial permite la descomposición del problema en subproblemas, lo que da lugar a la factibilidad de implementar técnicas, en este caso el recocido simulado para orientar al planificador a la toma de decisiones y solución de problemas en diversos puntos del desarrollo de la solución.

Firma del asesor: \_\_\_\_\_

Dr. Romeo Sánchez Nigenda

## CAPÍTULO 1

# INTRODUCCIÓN

---

### 1.1 DESCRIPCIÓN DEL PROBLEMA

Una de las áreas más importantes dentro de la inteligencia artificial es la planificación. Planificación es el proceso de búsqueda y generación de una secuencia de acciones, que a partir de un estado inicial, permiten alcanzar un conjunto de objetivos [31] [12].

Los problemas de planificación abundan, algunos ejemplos y aplicaciones se pueden observar en diversas áreas, desde la secuencia para acomodar una serie de bloques, hasta exploraciones espaciales (mars exploration rover). Incluso, en la industria manufacturera se emplean técnicas de planificación para planificar la secuencia de máquinas dobladoras.

Para dar solución a los problemas de planificación se utilizan los planificadores, actualmente existe una diversidad de ellos, los cuales se distinguen por la metodología utilizada para la solución de problemas. Los algoritmos más importantes de planificación se basan en estados [26], grafos [25] y planes de orden parcial [11].

Al crear algoritmos de planificación se pueden resolver problemas que representan mucha dificultad. Representa un gran reto el diseño de algoritmos eficientes y desarrollo de implementaciones robustas. Los algoritmos de planificación han alcanzado un gran éxito tanto en la industria como en ciertas disciplinas académicas, incluyendo la robótica, manufactura, aplicaciones aeroespaciales, entre otros.

En los últimos años se ha tenido un considerable crecimiento en el área lo que indica que las aplicaciones aun pueden ser extendidas [20].

Algunos ejemplos de aplicaciones de algoritmos de planificación son los siguientes:

- Vuelos a través del aire o espacio: algoritmos de planificación pueden ayudar a navegar helicópteros de manera autónoma a través de obstáculos.
- Planificación de misiones de naves espaciales pueden ser realizadas por medio de algoritmos de planificación.
- Diseño de medicinas: se ha tenido impacto incluso en campos que van lejos de la robótica tales como la biología computacional.
- Tareas de ensamble en automóviles.

Así como los ejemplos anteriores existen muchos más usos de los algoritmos de planificación [20].

El algoritmo de planificación que se está abordando en este problema es el POP( Planificación de orden parcial), el cual basa su representación y búsqueda en el espacio de planes, trabajando como una secuencia parcialmente ordenada, en el cual sólo las decisiones esenciales se registran, haciendo elecciones no deterministas hasta que todas las acciones sean soportadas y protegidas por alguna posible interferencia. La problemática principal con planificación es el tamaño del espacio de búsqueda. Esto impacta directamente a POP porque plan parcial representa en sí aun conjunto de planes, lo que hace que la búsqueda y toma de decisiones en espacios grandes se complique aún más para POP.

A este algoritmo se incorporan metaheurísticas basadas en la técnica del recocido simulado, la cual se aborda a detalle en el capítulo 2. Con esta unificación de técnicas se busca abarcar amplio un espacio de soluciones posibles, las cuales actualmente no son consideradas en a las metodologías que utilizan los planificadores basados en orden parcial de la actualidad.

## 1.2 JUSTIFICACIÓN

La motivación principal para abordar el algoritmo de orden parcial radica en que éste es muy flexible ya que resuelve los problemas de manera gradual, es decir toma sólo las decisiones esenciales que se van presentando conforme se avanza en la solución del problema, esto permite el ajuste de parámetros de decisión y la implementación de técnicas diversas en cada uno de los pasos en el algoritmo.

Estudios recientes han confirmado que la planificación de orden parcial ha sido la mejor manera para manejar los problemas de planificación con subproblemas independientes, asimismo, brinda facilidad de ejecución ya que representa explícitamente las partes de un plan. Además de los puntos mencionados, permite la representación de modelos de planificación complejos con recursos y tiempo. Por todo esto, La planificación de orden parcial permanece como una parte importante del campo para tareas específicas como operaciones de calendarización.

Aún con las ventajas que introduce el algoritmo, los planificadores que han sido desarrollados con esta técnica todavía presentan áreas de oportunidad. Algunos de ellos compiten a nivel mundial en la Conferencia Internacional de Planificación Automatizada y Scheduling (ICAPS). En estas conferencias, el último planificador basado en Planificación de Orden Parcial obtuvo el premio al "Best Newcomer", pero solo pudo resolver el 54% de los problemas establecidos [18]. Esto es un indicador de que estos tipos de algoritmos, aunque competitivos, todavía tienen una área de oportunidad bastante amplia para mejorar la cobertura de soluciones.

Variaciones de POP han sido desarrollados anteriormente, (heurística aditiva de VHPOP y planes relajados de REPOP), los cuales basan su estrategia de solución en la selección de fallas y heurísticas basadas en la distancia y análisis de factibilidad [11] así como en grafos de planificación, respectivamente [25]. Sin embargo aún se cuenta con áreas de oportunidad, las cuales se pretenden atacar mediante mejoras en los puntos de decisión al elegir un nuevo plan.

## 1.3 OBJETIVOS

EL objetivo principal de la tesis consiste en incrementar la cobertura del espacio de soluciones del algoritmo de planificación de orden parcial. La idea es modificar el algoritmo base de orden parcial considerando el algoritmo de recocido simulado (simulated annealing) con la finalidad de diversificar el espacio de soluciones que algoritmo base de POP explora. A la par de este objetivo se estudia la factibilidad de la implementación del recocido simulado dentro del algoritmo, buscando como resultado orientar al planificador a la toma de decisiones para generar planes de solución a los problemas propuestos.

### 1.3.1 OBJETIVOS ESPECÍFICOS

1. Estudio de las características y funcionamiento general de los planificadores.
2. Modelación de problemas de planificación en PDDL.
3. Análisis de las metaheurísticas existentes y las características con las que cuentan, eligiendo aquella que presente una buena adaptabilidad para los fines requeridos del problema a trabajar.
4. Detección de puntos de decisión en el algoritmo de orden parcial como prospectos para implementación de técnicas de mejora de decisiones para la generación de soluciones.
5. Diseño y desarrollo de un algoritmo unificado que incluya las características de orden parcial y del recocido simulado.
6. Implementación de desarrollo de algoritmo en el planificador.
7. Generación de pruebas de comportamiento y funcionalidad.
8. Solución de problemas de planificación.
9. Analisis de resultados preliminares y depuración de parámetros.



## 1.4 HIPÓTESIS

El desarrollo de un planificador con características de POP y recocido simulado permitirá la exploración de distintas áreas en el espacio de soluciones, logrando con ello encontrar diversos caminos para dar solución a los problemas propuestos. Como parte de la investigación es necesario determinar si es posible realizar ésta combinación de técnicas y si el comportamiento resultante es aplicable a distintos problemas de planificación.

## 1.5 APORTACIONES CIENTÍFICAS

La generación de planificadores es un tema que se ha estado tratando desde hace tiempo. Actualmente ha tenido gran auge debido a las aplicaciones que éstos tienen en diversos ámbitos para ayudar a la solución de distintos problemas, algunos de ellos mencionados en la primera sección de este capítulo.

Mediante las implementaciones que se proponen en este trabajo de tesis se busca potencializar el funcionamiento de los planificadores y dar soluciones alternas y a mayor cantidad de problemas establecidos. Diversos campos pueden verse beneficiados con la propuesta. Con este desarrollo se estaría contribuyendo en los siguientes puntos:

- Realizar una búsqueda más completa dentro del espacio de soluciones, abarcando distintas áreas de la misma.
- El hacer saltos en el espacio de soluciones lleva al planificador a generar distintos caminos para crear una solución por lo que se pueden tener distintas soluciones a un mismo problema.
- Debido al grado de aleatoriedad que tiene el planificador, es posible encontrar una solución aún y cuando un algoritmo voraz no la localiza, ya que el algoritmo voraz consiste en elegir siempre la mejor solución en cada paso de la búsqueda,

nuestro planificador permite en ciertas ocasiones elegir una solución no tan buena, pretendiendo con este punto encontrar una solución en un área no explorada.

## 1.6 METODOLOGÍA

El presente trabajo de investigación inicia con un análisis exhaustivo de los algoritmos de Planificación de Orden Parcial con el propósito de reconocer las áreas de oportunidad y la viabilidad de extenderlos con funciones meta-heurísticas.

Para lograr el conocimiento profundo del algoritmo POP se procederá a una fase de selección de problemas de planificación, y los problemas resultantes se resolverán con dichos algoritmos. El propósito es entender mejor el comportamiento de los algoritmos en problemas complejos de planificación.

Se desarrollará el algoritmo extendido de POP con recocido simulado lo cual permitiría incrementar la cobertura del espacio de soluciones. Se estudiarán diversas modificaciones al algoritmo base de POP-SA propuesto con el fin de mejorar aún más la cobertura de problemas resueltos por nuestro algoritmo. Se espera que un análisis exhaustivo de los parámetros más importantes del algoritmo propuesto proporcione la información necesaria para una adaptación completa de recocido simulado en POP.

Se realiza una selección de problemas los cuales varían desde problemas juguete como el mundo de los bloques, problemas de transporte y problemas en los cuales un robot se mueve a través de una malla a distintas locaciones de la misma, algunos de ellos son utilizados en las competencias internacionales de planificación [18]. Un número significativo de experimentos son realizados para probar la funcionalidad de las modificaciones realizadas en el planificador, en base a los resultados obtenidos se procede con las conclusiones del proyecto.

## 1.7 ESTRUCTURA DE LA TESIS

El trabajo de investigación comienza con la descripción del problema a abordar y su importancia, se dan a conocer las características del tema y cómo se pretende atacarlo mediante las estrategias de solución seleccionadas. Además se incluye la motivación que llevó a desarrollar este proyecto a través de un estudio a profundidad de los temas relacionados a la planificación que se encuentran en la literatura y cómo aportar soluciones aplicables en el campo de la inteligencia artificial, justificando así la relevancia del problema.

En la hipótesis mostramos cuáles son los resultados que se esperan obtener mediante la aplicación de las modificaciones realizadas en las técnicas utilizadas tanto en el planificador como en las metaheurísticas elegidas. Se incluyen además una serie de objetivos generales y específicos a alcanzar durante el desarrollo de la investigación.

En el primer capítulo también se hace referencia a la metodología a seguir para alcanzar los objetivos establecidos.

En el capítulo dos se introduce el marco teórico, presentando a profundidad el estado del arte de la planificación incluyendo características, componentes, descripción de funcionalidad y ejemplos prácticos de aplicaciones. A su vez, este capítulo explica las técnicas utilizadas para abordar el problema de planificación y su funcionamiento.

El tercer capítulo corresponde a la metodología de solución utilizada, explicando a detalle el funcionamiento de la metaheurística del recocido simulado y sus características generales. Además se incluye la formulación del problema, en donde se establece la descripción del tema a abordar y cómo se pretende atacarlo, presentando el funcionamiento y desglose de las estrategias y métodos aplicados.

La experimentación y resultados obtenidos son abordados en el capítulo cuatro,

---

en éste se muestra el comportamiento del algoritmo desarrollado en los problemas utilizados, los cuales son descritos en esta misma sección. Se evalúa la funcionalidad de los planificadores, además del análisis exhaustivo de los resultados obtenidos con el fin de evaluar la calidad de las soluciones e identificar áreas de oportunidad.

Por último, en el capítulo cinco se describen las conclusiones generadas en base a los resultados obtenidos previamente en la experimentación, presentando también el trabajo futuro, buscando con ello potencializar el funcionamiento del planificador desarrollado.

## CAPÍTULO 2

# MARCO TEÓRICO

---

## 2.1 PLANIFICACIÓN

Planificación en inteligencia artificial es el proceso de búsqueda y articulación de una secuencia de acciones que permitan alcanzar un objetivo a partir de una situación inicial [31] [28] [15].

La planificación estudia las acciones y los cambios que éstas producen en los estados de modelos que representan problemas reales.

A continuación se definen los elementos utilizados para generar un problema de planificación: descripción del estado inicial, descripción del objetivo y la descripción de las posibles acciones disponibles, condiciones para aplicarlas y efecto de las mismas [34] [41] [31].

- Estado Inicial: incluye la descripción del estado actual, objetos que lo componen, características y relaciones entre ellos
- Objetivo: es la descripción de la meta a alcanzar, especifica aquellos objetos y relaciones que tienen que ser verdaderos al final del plan.
- Acciones: describen las condiciones que deben existir para que se lleve a cabo una transición entre los estados, y los efectos de las mismas. Es decir, las acciones describen la función de transición entre todos los estados de nuestro problema.

La principal característica de la planificación es la representación de estados, objetivos y acciones, éstas últimas son representadas por descripciones lógicas de precondiciones y efectos, permitiendo al planificador realizar conexiones entre acciones y estados. Otra de las características en planificación es que tiene la libertad de agregar acciones al plan siempre que sea necesario, en vez de hacerlo de manera gradual empezando por el estado inicial.

Por último, es importante mencionar que la mayoría de las partes del mundo son independientes entre si, lo que permite elegir una meta a la vez e ir las resolviendo de manera paulatina (estrategia divide y vencerás). Este tipo de metodologías son eficientes debido a que casi siempre es más fácil resolver muchos subproblemas pequeños que uno solo muy grande. Sin embargo, no es útil en casos donde el costo de combinar las soluciones de los sub problemas es muy alto.

### 2.1.1 LENGUAJE DE PLANIFICACIÓN PDDL

Los problemas son representados a través de lenguajes estándar de planificación con sintaxis definida, uno de ellos es PDDL(Planning Domain Definition Language). PDDL fué desarrollado por Drew Mc-Dermott y dado a conocer en la International Planning Competition de 1998 [22][1]. El lenguaje de planificación PDDL desde entonces se ha convertido en un estándar para la presentación de modelos de planificación[2].

Una importante característica que se ha desarrollado en PDDL es que el lenguaje ha permitido tener un impacto significativo en la investigación de la planificación debido a la facilidad en que los sistemas comparten el estándar y el incremento en la disponibilidad de compartir recursos de planificación. La introducción de PDDL ha facilitado el desarrollo científico de la planificación. Desde 1998 se ha tenido fuerte auge en la investigación para la aplicación de tecnologías de planificación en problemas reales [19].

Además de los puntos mencionados anteriormente, PDDL cuenta con las características siguientes:

- Lenguaje basado en acciones formulado para la solución de problemas de planificación.
- Utiliza precondiciones para describir que tan aplicables son las acciones y sus efectos, éste punto es explicado a detalle en la siguiente sección.
- Capacidad de proveer funciones objetivo que pueden ser utilizadas para la toma de decisiones en cuanto a la evaluación de un plan. Un ejemplo de métrica que puede ser representada es que el costo de un problema sea minimizado.
- La finalidad del lenguaje es expresar en términos definidos el comportamiento del problema, incluyendo sus acciones, condiciones necesarias para ejecutarlas y cuáles son sus efectos al aplicarlas.
- Utiliza expresiones numéricas y operadores aritméticos con sintaxis definida.
- Los modelos están orientados a objetos, de manera que las acciones son vistas como métodos que aplican a un objeto dado .

Para la generación de soluciones se utiliza además de la representación del problema, el dominio y un planificador.

- El problema representa las condiciones iniciales y objetivos a alcanzar.
- El dominio representa las acciones posibles que se pueden tomar en cuenta para generar un plan, aquí se describen las características y condiciones necesarias para ejecutar las acciones.
- El planificador es un sistema que a partir de la representación o modelos de planificación encuentra una solución (plan) a un problema dado.

En la figura 2.1 se representan los elementos de un problema de planificación.



Figura 2.1: Problema

### 2.1.2 PLANIFICADORES

El dominio y problema, una vez modelados adecuadamente en su lenguaje, son introducidos en un planificador, el cual es el encargado de dar solución al problema mediante algoritmos de búsqueda. El planificador es una herramienta computacional que permite obtener de manera automática los planes adecuados para alcanzar una meta. Los planificadores pueden utilizar distintas metodologías para ejecutar un problema, tales como grafos, búsqueda de planes parciales, estados, etc. Estos algoritmos serán explicados a detalle más adelante en este capítulo.

Como resultado de un planificador se obtiene una secuencia ordenada de acciones, las cuales llevandolas a cabo sobre el entorno descrito en el estado inicial, permiten cumplir el objetivo, a esta secuencia de acciones se le denomina plan.

Para poder implementar un problema es importante desarrollar un modelo de representación adecuado. Una vez que se tiene el modelo éste es aplicado a un algoritmo de solución el cual debe ser capaz de comprender y resolver el problema.



### 2.1.3 DOMINIO Y PROBLEMA

Para dar solución a problemas de planificación se requiere utilizar los elementos dominio y problema. En el dominio se representan las posibles operaciones a realizar, reglas de aplicación y funciones de transición de estados. En el problema se encuentran representadas las condiciones iniciales y objetivos que hay que alcanzar a lo largo de la generación del plan.

Para describir mejor tanto el archivo dominio como el archivo problema mostraremos como ejemplo el conocido problema del mundo de los bloques, el cual consiste en un conjunto de bloques sobre una mesa, los bloques pueden ser colocados uno sobre otro, pero solamente uno puede ser apilado encima de otro, se levanta un bloque a la vez y se direcciona en otra posición, ya sea encima de la mesa o en otro bloque. El objetivo es encontrar los pasos a seguir (plan) para llegar de una configuración inicial a un estado final deseado.

Para describir mejor la representación de un problema en PDDL podríamos tener como objetivo colocar un bloque A sobre bloque B, y el bloque B sobre el bloque C, tal como se observa en la figura 2.2, la escritura en PDDL quedaría de la siguiente manera:

```
(define (problem mcd-sussman-anomaly)
(:domain mcd-blocksworld)
(:objects a b c)
(:init (block a) (block b) (block c) (block table)
(on c a) (on b table) (on a table))
(:goal (and (on b c) (on a b))))
```

El problema contiene los siguientes elementos:

1. Objetos: a, b c, son los bloques a utilizar

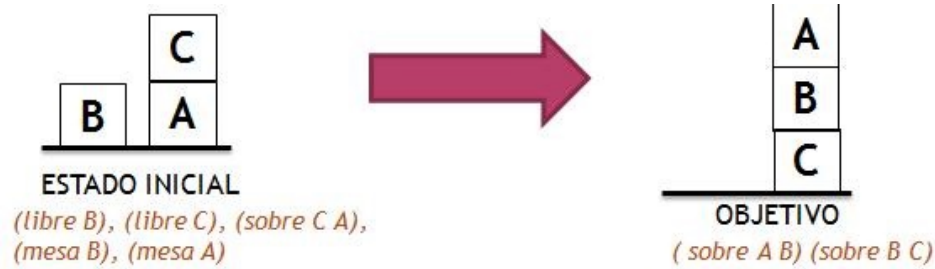


Figura 2.2: Mundo de los bloques

2. Condiciones iniciales: es el estado de las variables que componen el mundo actual init (block a) (block b) (block c) (block table) (on c a) (on b table) (on a table)), indica que al inicio del ejercicio se tiene el bloque a, b y c, se cuenta con una mesa y la descripción de cada uno de los objetos es que el bloque C se encuentra sobre el bloque A, el bloque B se encuentra sobre la mesa y A se encuentra sobre la mesa (ver primera parte de la Figura 2.2).
3. Objetivos a alcanzar: es el estado final que se desea alcanzar, en el ejemplo citado se desea la configuración B sobre C y A sobre B.
4. Predicados: se definen como las proposiciones utilizadas para representar la relación entre objetos, como parte de la representación del problema se incluyen en condiciones iniciales y objetivos .

El dominio es la función de transición del problema, dicho de otra manera, la teoría del dominio es una descripción de todas las posibles acciones que se pueden realizar, como afecta cada una de ellas a los objetos del mundo, a sus características y a sus relaciones, a partir de un lenguaje formal.

#### Ejemplo de Dominio

```
(define (domain blocks-world-domain)
  (:constants table)
  (:predicates (on ?x ?y) (clear ?x) (block ?x))
  (:action put on
```

```

:parameters (?x ?y ?z)
:precondition (and (on ?x ?z) (clear ?x) (clear ?y)
(not (= ?y ?z)) (not (= ?x ?z))
(not (= ?x ?y)) (not (= ?x table)))
:effect (and (on ?x ?y) (not (on ?x ?z))
(when (not (= ?z table)) (clear ?z))
(when (not (= ?y table)) (not (clear ?y))))))

```

Uno de los elementos del dominio son los predicados, los cuales, como se mencionó en la descripción del problema, indican relación entre objetos. Se pueden representar de la siguiente manera:

```
(:predicates (on ?x ?y) (clear ?x) (block ?x))
```

En donde ?x y ?y tomaran valores de A,B o C, (on ?x ?y) indica la posición en la que estaran los bloques uno sobre otro, (clear ?x) indica que el bloque no tiene ningún objeto encima de él, (block ?x) se refiere a un bloque en particular.

EL factor principal del dominio son las acciones, las cuales serán descritas a fondo en la siguiente sección.

## ACCIONES

Las acciones son funciones necesarias para resolver un problema, permiten la transición de un estado a otro. Se especifican en términos de las precondiciones que deben cumplirse antes de ser ejecutadas y las consecuencias que se siguen cuando se ejecutan. Constituyen una parte esencial del dominio del problema y sólo pueden ser ejecutadas cuando su precondición es verdadera. Cuando una acción es ejecutada cambia la descripción del mundo.

La acción put on (poner sobre) se describe como sigue.

```
(:action put on
  :parameters (?x ?y ?z)
  :precondition (and (on ?x ?z) (clear ?x) (clear ?y)
    (not (= ?y ?z)) (not (= ?x ?z))
    (not (= ?x ?y)) (not (= ?x table)))
  :effect ( and (on ?x ?y) (not (on ?x ?z))
    (when (not (= ?z table)) (clear ?z))
    (when (not (= ?y table)) (not (clear ?y))))
)
```

La acción mencionada contiene los siguientes elementos:

- **Parámetros** : son los posibles valores que pueden tomar las literales de la acción, es decir, las variables de entrada, incluye los objetos relacionados al dominio.

```
:parameters (?x ?y ?z)
```

Los parámetros u objetos que se estarán utilizando , son A, B, C.

Las literales tienen la sintaxis ?x, donde ?x se refiere a las variables de entrada de la acción y están definidos por los objetos establecidos en el problema.

- **Precondiciones**: contiene un conjunto de literales positivas o negativas, las cuales se deben cumplir para poder ejecutar una acción. Por ejemplo, la precondición aquí mostrada para la acción put on, indica que para poder aplicar la acción el objeto ?x debe estar sobre el objeto ?z, el objeto ?x debe estar libre y el objeto ?y debe estar libre. A su vez, ?y no debe ser igual al valor de ?z, ?x no debe ser igual a ?z, ?x no debe ser igual a ?y y ?x no debe ser mesa.

```
:precondition (and (on ?x ?z) (clear ?x) (clear ?y)
  (not (= ?y ?z)) (not (= ?x ?z))
  (not (= ?x ?y)) (not (= ?x table)))
```

- Efectos: al igual que las precondiciones, contiene literales, en donde se define el resultado de ejecutar cierta acción. Un efecto de la acción es el conjunto que puede incluir tanto literales positivas como negativas. En el ejemplo de aplicar la acción `put on`, el efecto obtenido indica que el objeto `?x` está sobre `?y`, el objeto `?x` no está sobre `?z`, cuando `?z` no es igual a mesa `?z` ningún objeto está sobre `z` y cuando `?y` no es igual a mesa ningún objeto está sobre `?y`.

```
:effect (and (on ?x ?y) (not (on ?x ?z))
(when (not (= ?z table)) (clear ?z))
(when (not (= ?y table)) (not (clear ?y))))))
```

PDDL permite expresar la estructura de objetos en un dominio, mediante los parámetros que aparecen en las acciones y los argumentos de los predicados, acciones con precondiciones negativas y efectos condicionales así como el uso de pre y post condiciones.

Más adelante se explica el desarrollo del problema y cómo se van interrelacionado las acciones entre sí para lograr el objetivo deseado.

## 2.2 ALGORITMOS DE PLANIFICACIÓN

Los algoritmos de planificación han sido aplicados en una variedad de problemas, además de los mencionados en las secciones anteriores. El trabajo en el desarrollo de algoritmos ha progresado desde la modelación hasta la implementación en software utilizado en la industria. En algunos casos la investigación sigue siendo aplicada para incrementar el potencial de métodos de planificación [20]. Una de las metodologías utilizadas en la generación de algoritmos es la estrategia *divide y vencerás*, en la cual se resuelven problemas de manera recursiva, aplicando los siguientes pasos en cada nivel de la recursión [4].

1. Dividir: consiste en dividir el problema en subproblemas, los cuales son instancias más pequeñas del mismo problema.
2. Conquistar: resolver los problemas recursivamente.
3. Combinar: las soluciones de los subproblemas se combinan para obtener la solución del problema original

La tarea de un algoritmo de planificación es encontrar una secuencia finita de acciones que, cuando son aplicadas transforma el estado inicial en un estado final distinto que cubra los objetivos planteados en el problema.

Los algoritmos de planificación se engloban en tres paradigmas generales: planificación por estados, grafos y planes parciales, los cuales serán descritos a detalle en las secciones siguientes.

### 2.2.1 ESPACIO DE ESTADOS

El espacio de estados consiste en un conjunto finito de estados  $V$  y un conjunto finito de acciones  $A$ . En el espacio de estados, las acciones disponibles producen transiciones entre los estados al modificar sus contenidos, esto origina que el espacio de estados se vea representado como un grafo  $G=(V, A)$ , donde los nodos corresponden a los estados del problema ( $V$ ) y los arcos entre ellos corresponden a las transiciones o acciones ( $A$ ) entre los mismos.

En el espacio de estados, cada nodo representa una descripción del mundo y cada arco representa la aplicación de una acción válida para un determinado estado. En este formalismo un plan se encuentra definido como una secuencia de arcos (un camino) que permiten cruzar el espacio de estados de un nodo inicial a un nodo objetivo. En la figura 2.3 podemos visualizar el espacio de estados en un grafo.

Las descripciones de las acciones de un problema de planificación especifican tanto precondiciones como efectos, por lo tanto, son posibles las búsquedas en ambas

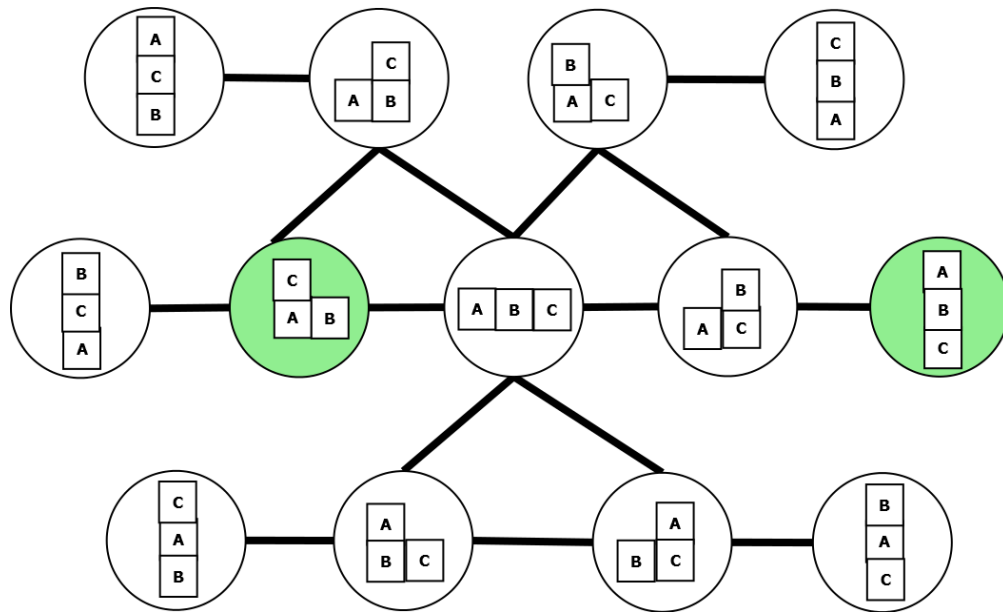


Figura 2.3: Espacio de estados

direcciones, búsquedas hacia adelante desde el estado inicial o búsquedas hacia atrás desde el estado final, estos también son llamados progresivos y regresivos.

### REGRESIÓN

La búsqueda regresiva consiste en encontrar un camino a partir del objetivo, esto es, empieza por las metas que hay que satisfacer y va trabajando hacia atrás en busca de operadores que ofrezcan la solución deseada, esto se realiza a través de los nodos intermedios y de esta manera se busca llegar al estado inicial, logrando así la solución. La principal ventaja de este tipo de búsqueda es que permite considerar solamente acciones relevantes. La restricción a acciones relevantes se traduce en búsquedas hacia atrás que tienen un factor de ramificación mucho menor que en búsquedas hacia adelante.

### PROGRESIÓN

Toma su nombre debido a que mantiene una dirección de avance. En la búsqueda progresiva o hacia adelante se comienza en el estado inicial, considerando secuen-

cias de acciones hasta encontrar una que alcance el estado objetivo [36] [32]. Se ha demostrado que la búsqueda hacia adelante en un espacio de estados es ineficaz si no utiliza buenas heurísticas, es por ello que es de suma relevancia utilizar una buena heurística, ya que corre el riesgo de quedar incompleto en la búsqueda de solución. Además utiliza acciones irrelevantes, esto se da porque todas las posibles acciones son consideradas desde el estado en que nos encontramos [31].

## PLANIFICACIÓN DE ORDEN TOTAL

Encuentra un camino desde el estado inicial al estado final, la búsqueda se desarrolla en un espacio de estados y puede ser progresiva o regresiva. El espacio de búsqueda esta conformado por secuencias totalmente ordenadas de acciones.

### 2.2.2 ESPACIO DE PLANES

En la búsqueda a través del espacio de planes los nodos representan planes parciales y los arcos son las operaciones de refinamientos al plan. Los refinamientos consisten en el ordenamiento de acciones en un plan. Más adelante en éste capítulo se explica el detalle sobre refinamientos. En la figura 2.4 se muestra la representación del espacio de planes.

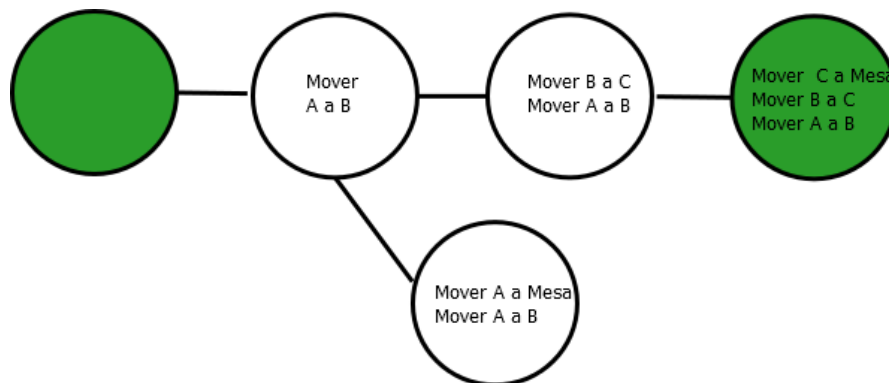


Figura 2.4: Espacio de planes



## PLANIFICACIÓN DE ORDEN PARCIAL

En el espacio de planes, una solución se representa como una secuencia de acciones parcialmente ordenada, por lo que los algoritmos que utilizan dicho espacio caen dentro del formalismo de Planificación de Orden Parcial. Originalmente, la planificación de orden parcial fué introducida en el año 1975 por Sacerdoti como una manera de mejorar la eficiencia de la planificación evitando compromisos de un orden en particular con los subobjetivos a alcanzar [23].

Es un algoritmo utilizado para la solución de problemas en el cual, el espacio de planes se representa como una secuencia parcialmente ordenada. Su principal ventaja es la flexibilidad en el orden de construcción del plan, debido a que toma las decisiones más importantes al principio en lugar de seguirlas en cierto orden establecido

En la planificación de orden parcial se trabaja sobre cada subproblema separadamente, tomando decisiones sobre el orden en que suceden las acciones desde todos los subproblemas. La planificación de orden parcial utiliza un enfoque en el cual se trabaja en varios subobjetivos de manera independiente y éstos subobjetivos son solucionados por medio de subplanes. El esquema de orden parcial tiene la ventaja de ser flexible en cuanto al orden en que se construye el plan , es decir puede trabajar sobre importantes acciones primero antes de elegir las acciones en el orden establecido.

La estrategia de aplazar una opción durante la búsqueda se conoce como mínimo compromiso [31] en la cual solo las decisiones esenciales se registran haciendo elecciones hasta que todas las precondiciones sean satisfechas. Ésta metodología reduce el factor de ramificación en el espacio de búsqueda [43].

EL planificador POP se implementa como una búsqueda en el espacio de los planes, comenzando con un plan vacío y posteriormente se consideran formas de refinar el plan hasta que se tenga un plan completo que resuelva el problema.

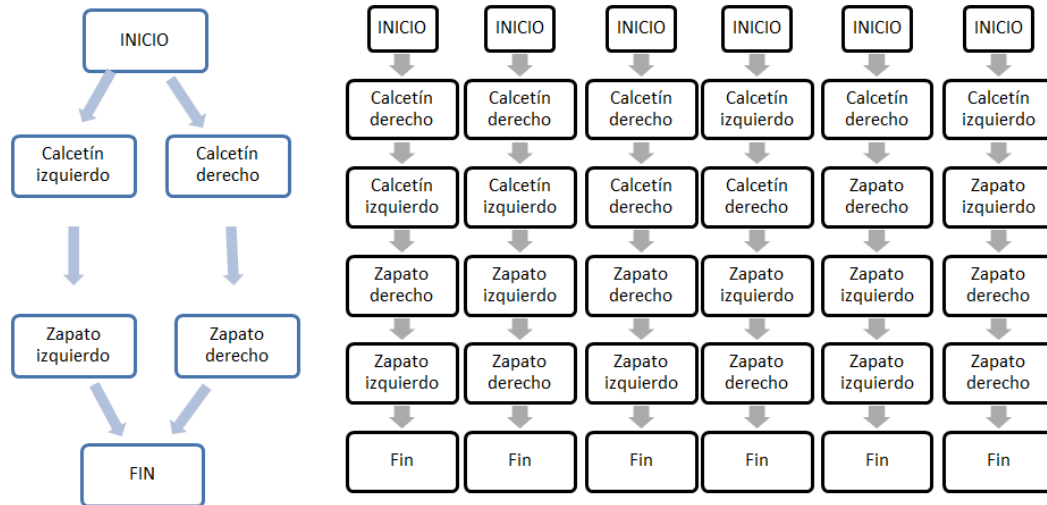


Figura 2.5: Total y parcial

En la figura 2.5 podemos observar un plan de orden parcial que constituye la solución para un problema de zapatos y calcetines. La solución es representada como un grafo de acciones y no como una secuencia. La solución de orden parcial corresponde a seis posibles planes de orden total, cada uno de ellos conocidos como linealización del plan de orden parcial [31]. La principal diferencia entre la planificación de orden parcial y de orden total es que una solución de orden parcial no restringe el ordenamiento entre acciones cuando no es necesario, en el ejemplo 2.5 no importa si empezamos primero con el izquierdo o derecho, pero ambas acciones tienen que preceder al zapato. En la planificación total se tiene que especificar un ordenamiento total entre las acciones, aunque éste no sea necesario.

Un plan de orden parcial se representa con los siguientes elementos:  $P < A, O, L >$  en el cual  $A$  es el conjunto de acciones,  $O$  son las restricciones y  $L$  son los enlaces causales.

1. Conjunto de acciones: Un plan vacío contiene las acciones inicio y fin. Inicio no posee precondiciones y tiene como efectos los literales en el estado inicial del problema de planificación, el estado final no tiene efectos y tiene como precondiciones los literales del objetivo del problema de planificación

2. Conjunto de restricciones ordenadas. Cada limitación ordenada es de la forma  $A < B$ , la cual se lee A antes de B y significa que la acción debe ser ejecutada en algún momento antes de la acción B, pero no necesariamente en el estado inmediatamente anterior. Las restricciones ordenadas deben describir un orden parcial apropiado. Cualquier ciclo ( $A < B$  y  $B < A$ ) representa una contradicción. Las restricciones ordenadas ayudan a prevenir inconsistencias en el plan.
  
3. Relaciones causales: Cada enlace causal se encarga de unir dos acciones, las cuales se dividen en dos categorías: acción productora (A) y acción consumidora (B), de esta manera el enlace causal creará una relación entre los efectos de la acción productora con las precondiciones de la acción consumidora [21] [34]. Representándolo de la siguiente manera:  $A \xrightarrow{p} B$  y se lee como A alcanza B a través de p [9].

Indica que p es un efecto de la acción A y una precondición para B. Una acción C entraría en conflicto si C tiene el efecto  $\neg p$  y si C pudiera traer A antes que B.

El mecanismo de búsqueda de orden parcial consta de dos componentes de toma de decisión relevantes, la selección de nodo o plan parcial y la selección de la falla a resolver, las fallas son descritas más adelante en esta misma sección. La primera, selección de nodo consiste en elegir a cuál plan parcial se le va a aplicar un refinamiento en el siguiente paso, una vez que el plan ha sido seleccionado para su refinamiento se realiza una serie de evaluaciones para asegurar que las acciones sean consistentes, esto es que no tengan ningún tipo de falla, para lograrlo, el planificador deberá entonces seleccionar una falla a atacar, la cual implica elegir entre una condición abierta y un enlace no seguro [29].

Una falla se presenta cuando una acción introducida interfiere con decisiones anteriores. Cuando el plan contiene una falla existe el riesgo de que el plan no funcione correctamente. Para prevenir este tipo de situaciones el algoritmo de planificación

debe revisar si hay fallas y tomar las medidas necesarias para solucionarlas. Las fallas que se pueden presentar son: condiciones abiertas y enlaces no seguros.

- Condiciones abiertas: uno de los efectos de cierta acción no se encuentra conectada con alguna precondition de otra acción. Se soluciona agregando un enlace causal de otra acción, ya sea una nueva o una existente [42].
- Enlace no seguro: la precondition de una acción entra en conflicto con el efecto de otra acción [27]. Para corregir este tipo de situaciones se aplican ciertos refinamientos, los cuales consisten en el ordenamiento de acciones. Los refinamientos se dividen en las siguientes categorías:
  1. Degradación: asegura que la acción 1 sea ejecutada antes de la acción 2
  2. Promoción: realiza un movimiento contrario, asegurando que la acción 2 sea ejecutada antes que la acción 1 [34].

Un planificador con mínimo compromiso debe llevar registro de las constantes, ordenamiento, enlaces y hacer cumplir la consistencia entre ellos, lo cual agrega complejidad a los algoritmos de planificación [43]. Se define un plan consistente como un plan en el cual no hay ciclos en las restricciones ordenadas y no existen conflictos con los enlaces causales. Un plan consistente con preconditiones no abiertas es una solución.

En el algoritmo 2.1 podemos observar el comportamiento de la búsqueda de una solución a un problema de planificación, dentro del espacio de planes parciales, en el cual, dado un problema, regresa un plan de solución o ningún plan en caso de que el problema carezca de solución. El problema de planificación es un conjunto de condiciones iniciales  $I$  y un conjunto de objetivos  $G$  y es representado por un plan inicial con acciones *dummy*  $a_0 \prec a_\infty$ , donde los efectos de  $a_0$  representan las condiciones iniciales del problema y las preconditiones de  $a_\infty$  representan los objetivos del problema. El procedimiento de la línea 2 `CrearPlanInicial` regresa un plan con éstas características. Se genera un conjunto de planes parciales  $P$  que aún no son

**Algoritmo 2.1:** Algoritmo de orden parcial

---

```

1 Encontrar plan (I,G) dadas las condiciones iniciales y objetivos
2  $P \leftarrow \text{CrearPlanInicial}(I,G)$ 
3 while  $P \neq O$  mientras existan planes pendientes do
4    $\pi \leftarrow$ seleccionar un plan parcial de P, y eliminarlo de los planes
   pendientes
5   if  $(F(\pi) == 0)$  si el plan elegido no tiene fallas
6     then
7       return  $\pi$  se tiene una solución
8   else
9      $f \leftarrow$ seleccionar una falla del conjunto de fallas en un plan  $\pi$ 
10    aplicar refinamientos al plan  $\pi$  e incluir dichos refinamientos en la
    cola de planes pendientes P
11    return failure, el problema no tiene solución

```

---

visitados. En cada etapa del proceso un plan  $\phi$  es seleccionado (línea 4) y se elimina de los planes pendientes P, después se valida si el plan seleccionado tiene fallas que reparar. Se aplican los refinamientos correspondientes para resolver la falla (línea 10) y el proceso continúa hasta que P está vacío, en cuyo caso el problema no tiene solución (línea 11) o hasta que un plan sin fallas sea encontrado, de ser así, se tiene la solución al problema (línea 7) [11].

A continuación se muestra el ejemplo de manera gráfica del mundo de los bloques solucionado mediante el algoritmo de orden parcial.

1. Empieza con un estado inicial cuyas precondiciones son nulas y sus efectos consisten en: (sobre A) (libre B) (libre C) (sobre B mesa). También cuenta con un estado Final el cual tiene las siguientes condiciones abiertas: (Sobre B C) (Sobre A B) que corresponden a los objetivos a alcanzar y a su vez Final tiene efectos nulos. Figura 2.6.

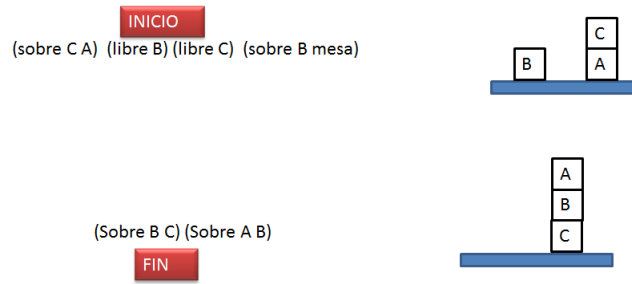


Figura 2.6: Mundo de los bloques parte1

2. Debido a que el objetivo (sobre A B) no se puede ligar a ninguno de los efectos de inicio, se agrega una nueva acción mover A de mesa a B y se hace el link causal sobre A B, tal como se observa en la figura 2.7
3. Se agregan las acciones Mover B de mesa a C y mover C de A a mesa. Figura 2.8.

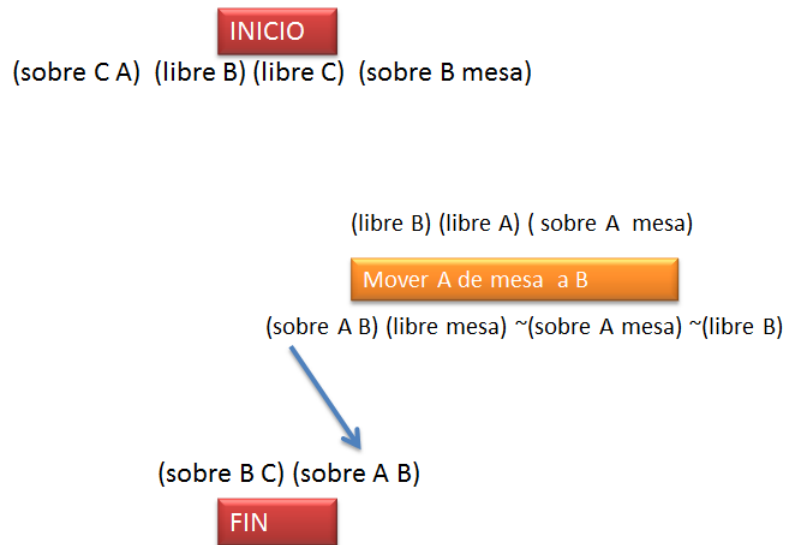


Figura 2.7: Mundo de los bloques parte2

4. Se toma la precondition (libre A) de la acción Mover A de mesa a B y se satisface con la acción mover C de mesa a A, y el resto de las precondiciones se satisfacen con el estado inicial. Figura 2.9.

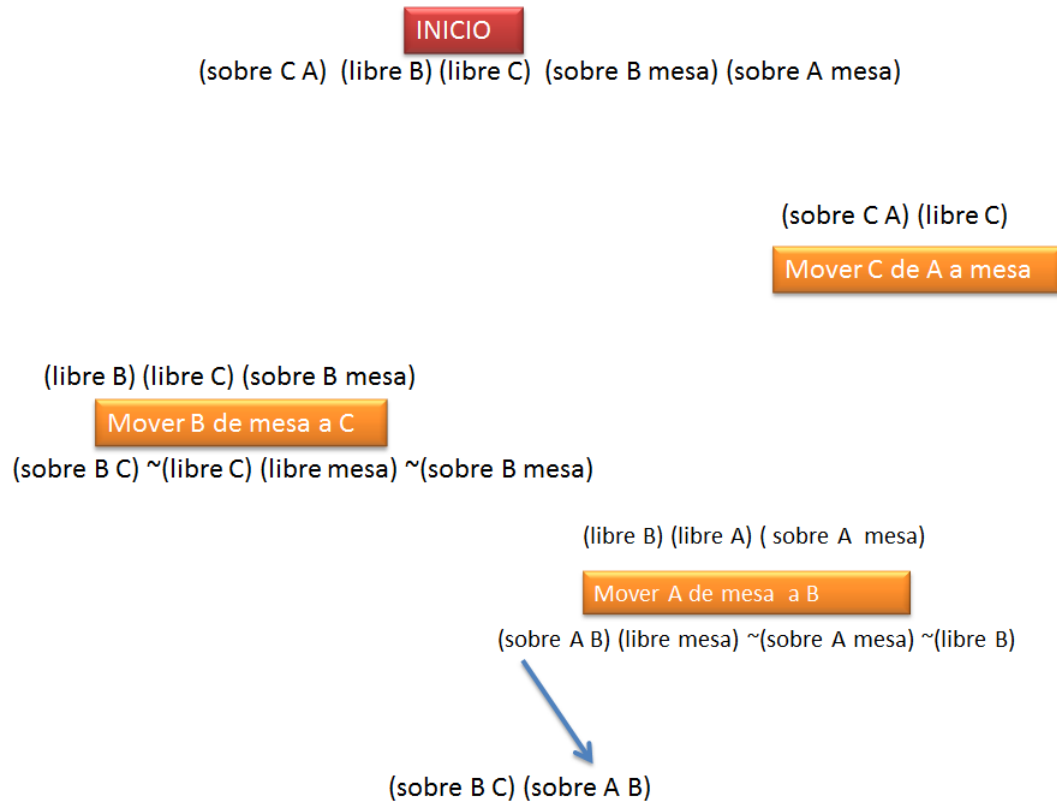


Figura 2.8: Mundo de los bloques parte3

5. Las precondiciones de la acción mover C de mesa a A se pueden satisfacer con los efectos de inicio, se agregan los enlaces. Figura 2.10.
6. Existe una amenaza en la acción mover B de mesa a C con el efecto (libre C) (marcado en rojo), condición requerida para la acción mover C de A a mesa, esta amenaza se resuelve ordenando la acción Mover B de mesa a C después de la acción mover C de A a mesa . Figura 2.11.
7. Las precondiciones de mover B de mesa a C se satisfacen con el estado inicial y el objetivo de fin (sobre B c) se satisface con el efecto de mover B de mesa a C. de esta manera se tiene un plan completo donde se satisfacen todas las precondiciones. Figura 2.12.

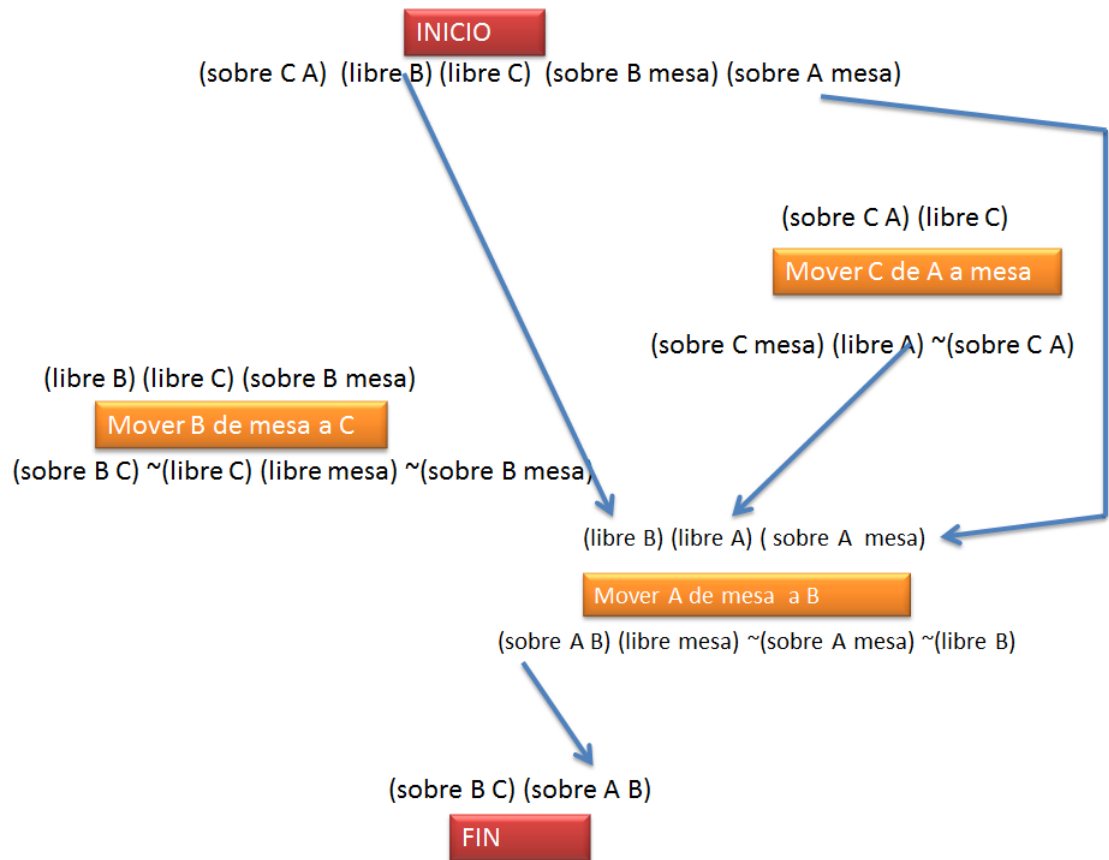


Figura 2.9: Mundo de los bloques parte4

8. Quedando el plan con la siguiente secuencia de acciones, tal como se visualiza en la figura 2.13.

Mover C de A a mesa

Mover B de mesa a C

Mover A de mesa a B

La planificación de orden parcial posee una ventaja por su capacidad para descomponer problemas en subproblemas. Su principal motivación es la eficiencia evitando el compromiso prematuro en las decisiones y por consiguiente evita retrocesos innecesarios, mejorando así el rendimiento del planificador [38]. Sin embargo no representa los estados directamente, de modo que es más difícil estimar cuánto de alejado está un plan de orden parcial de alcanzar un objetivo [37].



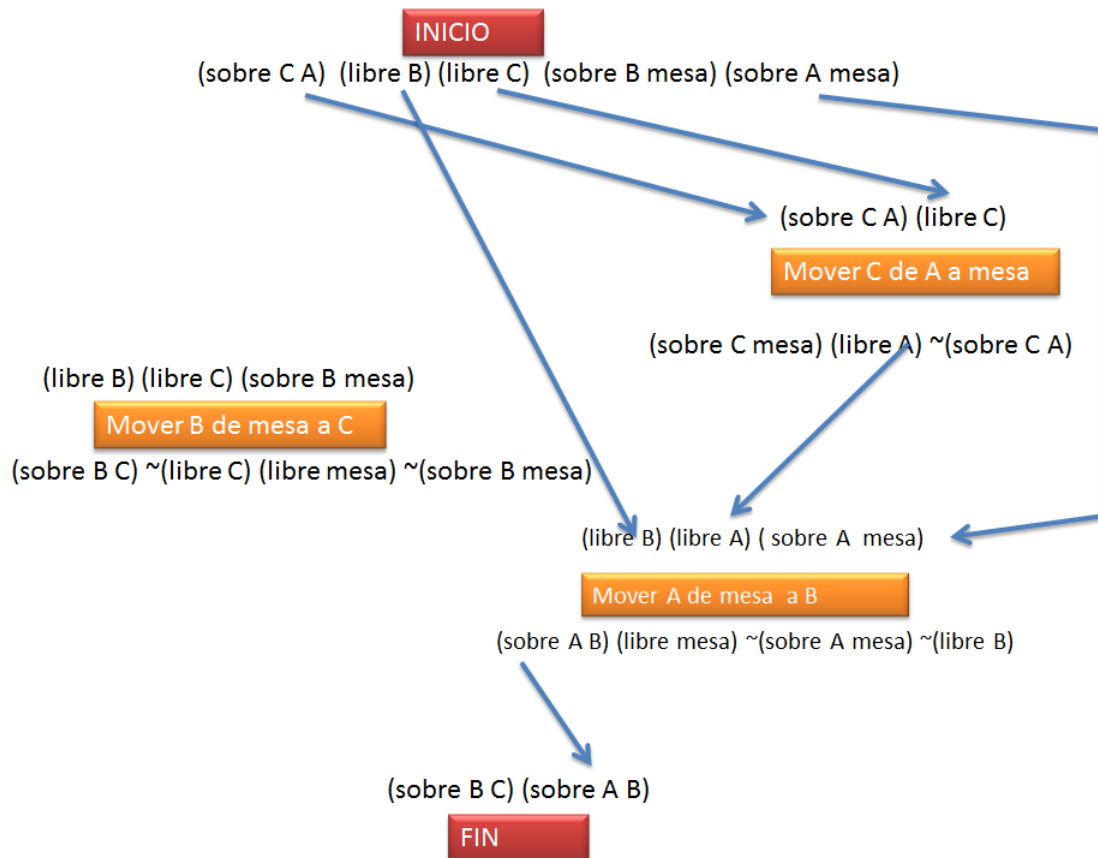


Figura 2.10: Mundo de los bloques parte5

Se ha desarrollado una variedad de planificadores, sin embargo para finalidades de este trabajo nos centraremos en aquéllos basados en el modelo de orden parcial. A continuación veremos el funcionamiento de algunos de los planificadores que se han desarrollado bajo este esquema.

- UCPOP: opera con acciones que tienen efectos condicionales, precondiciones, efectos y objetivos universalmente cuantificados. Su funcionamiento comienza con un plan inicial, cuyos efectos son las condiciones iniciales y un plan final cuyas precondiciones son los objetivos a alcanzar. UCPOP intenta completar el plan inicial agregando pasos nuevos y restricciones hasta que todas las precondiciones son satisfechas, los dos principales pasos son soportar las

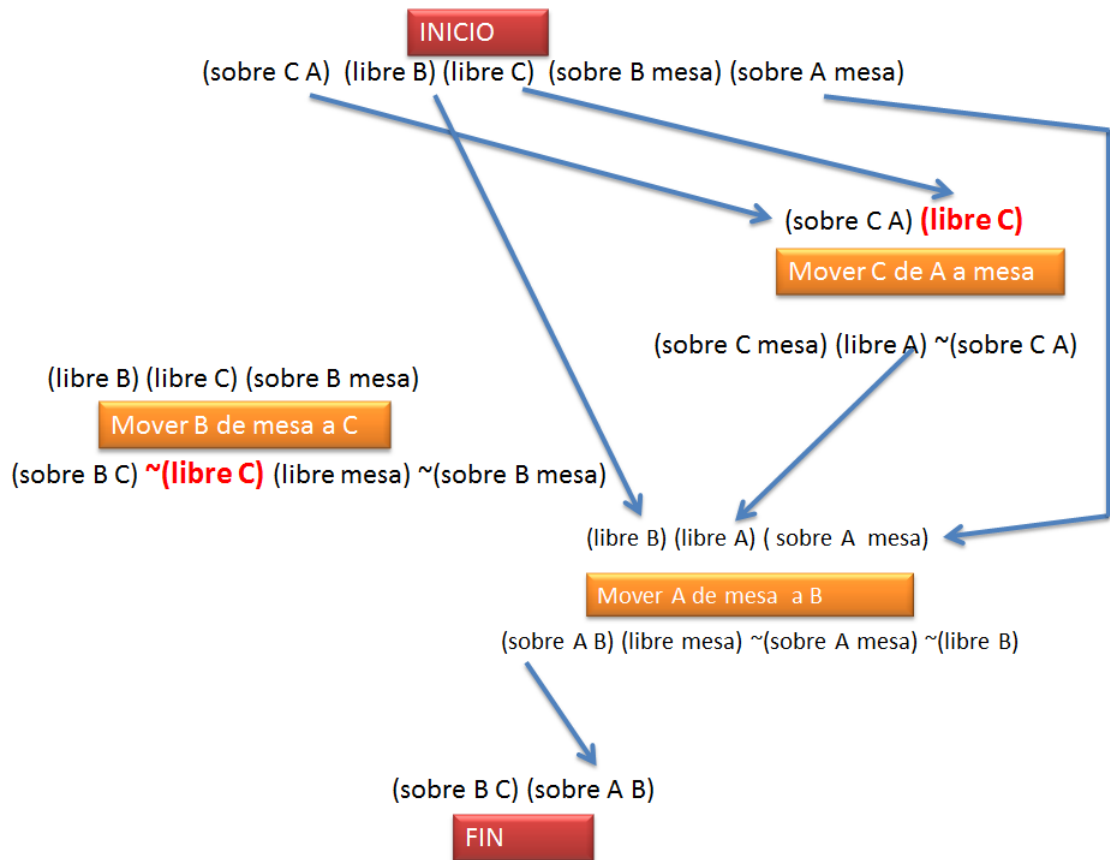


Figura 2.11: Mundo de los bloques parte6

condiciones abiertas y resolver las fallas. Para soportar las condiciones abiertas UCPOP elige de manera no determinista y agrega un enlace causal. Para resolver las fallas elige de igual manera de forma no determinista un método para resolverlo, ya sea reordenando los pasos del plan o agregando constantes nuevas de igualdad [26] [35] [8].

- VHPOP (Versatile Heuristic Partial Order Planning), desarrollado en 2003 por L.S. Younes y Reid Simmons, el cual está basado en UCPOP, es un planificador de orden parcial de enlaces causales el cual incluye la posibilidad de utilizar estrategias en base al costo de cada decisión respecto a la cercanía para llegar a la solución final. Utiliza estrategias basadas en las fallas del problema y heurísticas basadas en el análisis de alcanzabilidad.

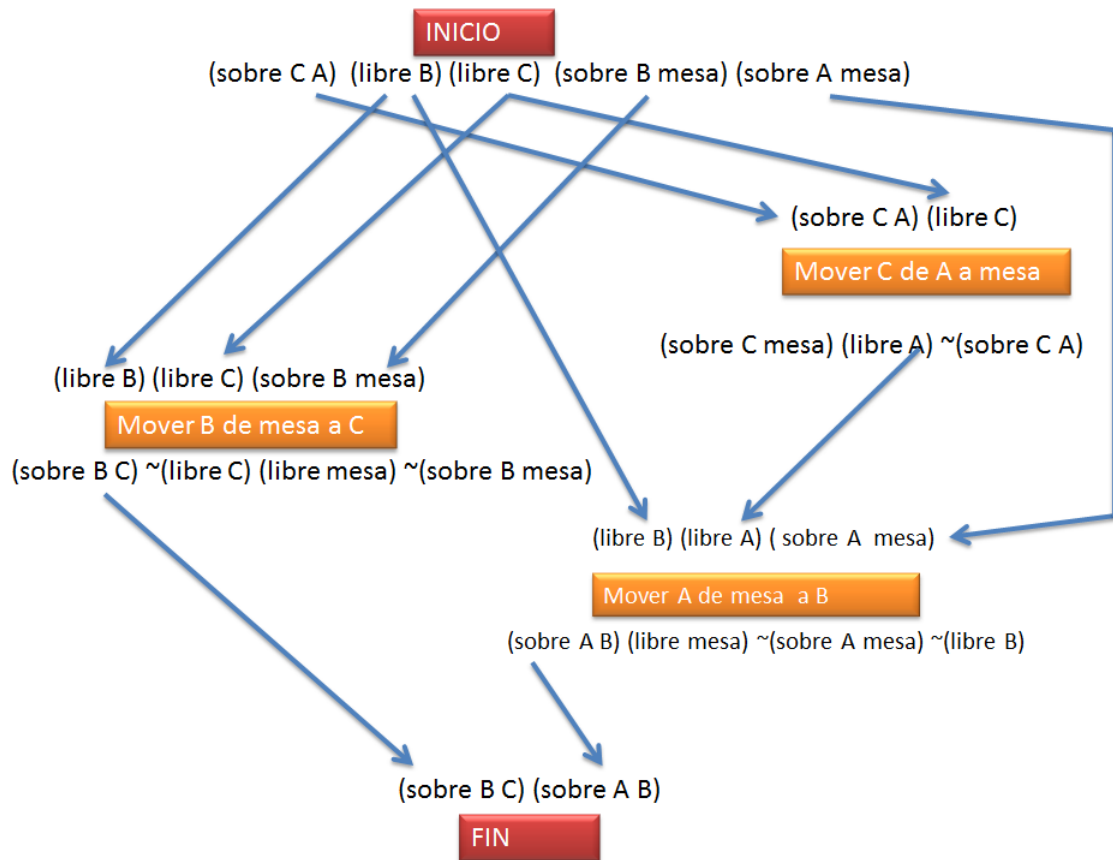


Figura 2.12: Mundo de los bloques parte7

VHPOP tuvo participación en la tercera competencia Internacional de Planificación (IPC3) 2003 logrando resolver el 54% de los problemas establecidos [18].

- REPOP: su metodología de uso se basa en el espacio de estados y búsqueda CSP, adaptándolos al algoritmo POP, utiliza heurísticas basadas en distancia y análisis de factibilidad y técnicas de procesamiento de restricciones. Las heurísticas basadas en distancia son utilizadas como base para establecer el costo de planes parciales y como métricas para la selección de fallas. Las últimas dos técnicas son utilizadas para garantizar la consistencia de los planes parciales detectando conflictos implícitos y resolverlos [24].

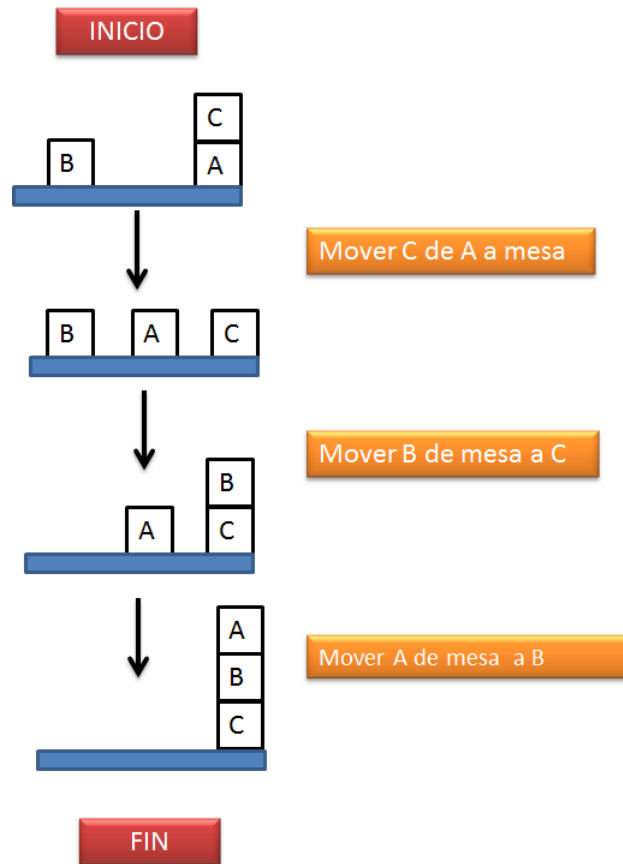


Figura 2.13: Mundo de los bloques parte8

VHPOP, el planificador seleccionado para esta investigación, ha competido en la competencia internacional de planificación organizador por la Conferencia Internacional sobre Planificación y Programación Automatizada (ICAPS). La Conferencia Internacional sobre Planificación y Programación Automática (ICAPS) es un foro de investigadores y profesionales en planificación y programación (scheduling), tecnologías críticas en áreas como la manufactura, sistemas espaciales, software de ingeniería, robótica, educación y entretenimiento.

ICAPS fué creado como resultado de la fusión entre entre las conferencias AIPS, por sus siglas Conferencia Internacional de Inteligencia Artificial, Planificación y Programación y y ECP, Conferencia Europea de Planificación. El objetivo principal de ICAPS es promover el área de la planificación y programación automatizada, a través de la organización de reuniones técnicas, incluyendo la conferencia anual

---

ICAPS, además de escuelas de verano, tutoriales y actividades de entrenamiento en varios eventos por medio de la organización de competencias de planificación y programación , así como la evaluación comparativa y otros medios de avance para promover el estado del arte en el campo, logrando así que jóvenes científicos se involucren mediante la difusión de publicaciones, sistemas de planificación, dominios, simuladores, herramientas de software y material técnico. La última conferencia realizada tuvo lugar en 2013 en Roma, Italia.[13]

## CAPÍTULO 3

# FORMULACIÓN DEL PROBLEMA Y METODOLOGÍA DE SOLUCIÓN

---

### 3.1 INTRODUCCIÓN

Conseguir soluciones óptimas en ciertos problemas de importancia industrial y científica es intratable. En la práctica se utilizan comúnmente soluciones buenas, las cuales son obtenidas mediante algoritmos heurísticos o metaheurísticos.

Las heurísticas en los planificadores son utilizadas como una función de evaluación, estimando el costo de alcanzar un objetivo desde un punto dado. Existe un variedad de algoritmos que utilizan funciones heurísticas de evaluación para reducir la búsqueda, algunas de ellas incluyen algoritmo de ascenso de colinas (hill climbing), A\* o IDA [16]. Un método de búsqueda metaheurístico se puede definir como metodologías generales de nivel superior que pueden ser utilizadas como estrategias de guía en el diseño de heurísticas subyacentes para resolver un problema específico [39].

Las metaheurísticas tienen la capacidad de proporcionar soluciones aceptables en tiempos razonables para resolver problemas difíciles y complejos aplicados en ciencia e ingeniería. A diferencia de los métodos exactos, sin embargo, las metaheurísticas no garantizan la optimalidad de las soluciones obtenidas.

## 3.2 PLANTEAMIENTO DEL PROBLEMA

Dentro de las áreas de oportunidad existentes en el algoritmo POP encontramos la toma de decisiones al momento de seleccionar los planes parciales a trabajar durante la búsqueda y la selección del refinamiento a aplicar una vez que el plan ha sido seleccionado. Estos dos puntos de decisión han sido considerados por la comunidad científica desarrollando funciones heurísticas para evaluar la calidad de planes parciales y refinamientos. Sin embargo, no existen registros en la literatura de la aplicación directa de metaheurísticas automatizadas a POP, hasta ahora.

Nuestra solución considera la modificación del motor de búsqueda de POP usando técnicas metaheurísticas de recocido simulado, con la finalidad de diversificar las soluciones consideradas por POP. Nuestro trabajo se explica a detalle en la siguiente sección.

## 3.3 RECOCIDO SIMULADO

Se aplica la técnica del recocido simulado (simulated annealing), propuesta en 1982 por S. Kirkpatrick[39], en el algoritmo POP para la selección de planes.

El recocido simulado es una metaheurística de búsqueda local que ha sido aplicada a problemas de optimización combinatoria[39], los cuales están relacionados con teoría de algoritmos, complejidad computacional e inteligencia artificial. Este tipo de problemas resuelven instancias explorando el campo de soluciones (usualmente grande).

El recocido simulado ha probado su efectividad en varios campos tales como el diseño de circuitos electrónicos, procesamiento de imágenes, organización de la red de procesamiento de datos, etc. Establece la conexión entre este tipo de comportamiento de termodinámica y la búsqueda del mínimo global para un problema de optimización discreta [6].

Es llamado así debido a la analogía con el proceso físico de enfriamiento de sólidos mediante el cual se aplica calentamiento sobre un material y después se somete a un enfriamiento lentamente y de manera controlada hasta obtener una estructura fuerte y cristalina en la que se alcanza un estado de energía mínimo [17][39][7]. Si el sistema de enfriamiento es suficientemente lento la configuración final resulta en un sólido en el cual se tiene una integridad estructural superior. El algoritmo simula los cambios de energía en el sistema sujeto a un proceso de enfriamiento hasta que converge en un estado de equilibrio.

La principal aportación del algoritmo es que provee un mecanismo para escapar del óptimo local permitiendo movimientos malos (ascendentes), de esta manera es posible saltar fuera del óptimo local y potencialmente caer en una solución más prometedora [33] [17]. Este comportamiento se puede visualizar en la figura 3.1. La idea es hacer suficiente exploración en el espacio de soluciones, de esta manera se disminuye la probabilidad de quedar atrapado en un mínimo local. La característica principal con la que cuenta esta técnica es que examina los planes vecinos de manera aleatoria, ya sea que el nuevo vecino sea mejor, o en caso de que no, pase una prueba de aleatorización para ser elegido [10].

El rango bajo el cual el sistema es enfriado se le conoce como calendario de enfriamiento (annealing schedule). Si el enfriamiento ocurre muy rápido se puede alcanzar una solución local, si de otra manera se realiza un enfriamiento lento se puede alcanzar la solución mínima global. A medida que la temperatura alcanza el valor de cero los movimientos a una solución peor (hill climbing) son menos frecuentes.

Para implementar el algoritmo, es necesario seleccionar un esquema de enfriamiento, el cual define la temperatura actual para cada paso del algoritmo. El enfriamiento tiene un impacto considerablemente fuerte en el éxito del algoritmo [39]. Los parámetros a considerar son:

- El valor inicial que será usado como temperatura.
- Criterio para decidir cuándo se reduce la temperatura.



- Método utilizado para realizar la disminución de la temperatura.
- Temperatura final, la cual define el criterio de parada.

Se empieza con una temperatura alta, lo que permite realizar saltos grandes dentro del espacio de soluciones y gradualmente estos saltos van disminuyendo conforme disminuye la intensidad de la temperatura permitiendo escapar de un óptimo local.

En el algoritmo 3.1 se explica a detalle la plantilla del algoritmo correspondiente al recocido simulado [33], el cual se desarrolla en varias iteraciones. En cada iteración se genera un vecino de manera aleatoria (línea 4), si el movimiento es mejor que el actual se acepta siempre (línea 7), en caso de ser peor el vecino es seleccionado con cierta probabilidad la cual depende de la temperatura actual y de la disminución de energía  $\Delta E$  de la función objetivo (línea 9) .  $\Delta E$  representa la diferencia de el valor objetivo (energía) entre la solución actual y la solución vecina. Conforme avanza el algoritmo la probabilidad de tomar soluciones malas decrementa, es por eso que los movimientos malos son mas frecuentes al inicio del algoritmo, ver figura 3.1.

La probabilidad sigue una distribución de Boltzman [39] [3].

$$P = e^{-\Delta/T}$$

El parámetro de control temperatura determina la probabilidad de aceptar soluciones no tan buenas. Dado un valor particular de temperatura son exploradas diversas soluciones. Una vez que el estado de equilibrio es alcanzado la temperatura se decrementa gradualmente de acuerdo al sistema de enfriamiento definido de tal manera que pocas soluciones malas vayan siendo aceptadas al final de la búsqueda.

Si la temperatura disminuye despacio, se realizan saltos mas grandes en el espacio de soluciones lo que nos lleva a encontrar un óptimo global[30]. En la figura 3.1 se muestra como el recocido simulado permite escapar de un óptimo local. Mientras mayor sea la temperatura existe mayor probabilidad de que un movimiento malo sea aceptado.

## Recocido simulado

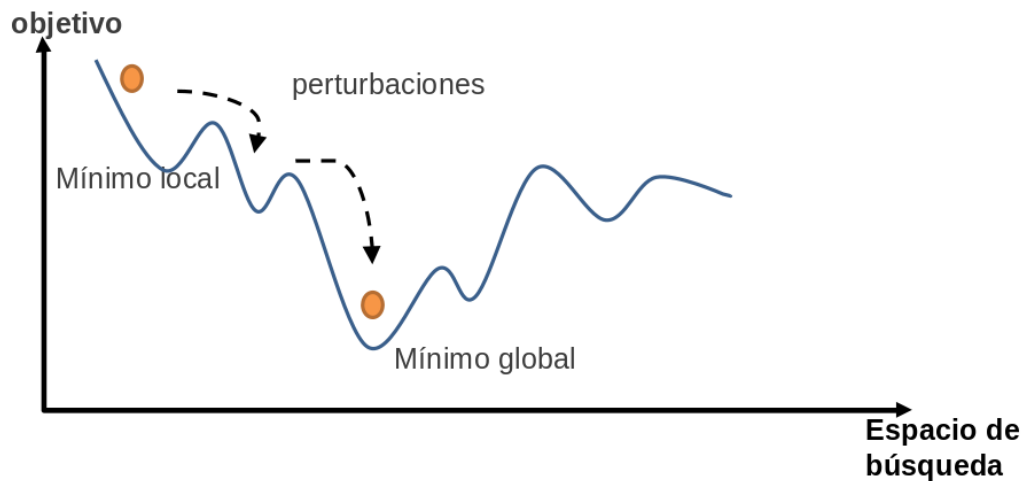


Figura 3.1: Recocido simulado

Para realizar un diseño eficiente de la metaheurística es importante la definición del vecindario y la generación de la solución inicial, además del diseño de los siguientes elementos:

- Función de aceptación de probabilidad: es el principal elemento del recocido simulado que habilita a los vecinos (soluciones potenciales) no mejores a ser aceptados.
- Esquema de enfriamiento: define la temperatura en cada paso del algoritmo. Tiene un papel esencial en la eficiencia y efectividad del algoritmo.

### 3.3.1 MOVIMIENTOS DE ACEPTACIÓN

Los movimientos de aceptación permiten al sistema escapar del óptimo local debido a la aceptación probabilística de un vecino con solución peor. La probabilidad de aceptar a este vecino es proporcional a la temperatura  $T$  e inversamente proporcional al cambio de la función objetivo  $\Delta E$ . A altas temperaturas la probabilidad de aceptación aun movimiento peor es alta. Si  $T = \infty$ , todos los movimientos son aceptados. A bajas temperaturas la probabilidad de aceptar un movimiento malo

---

**Algoritmo 3.1:** Algoritmo de Recocido Simulado

---

```

1 M= numero de movimientos a permitir
2 T= temperatura actual
3 for  $m= 1$  to  $M$  do
4   | Generar movimiento aleatorio de la solucion actual.
5   | Evaluar el cambio de energía  $\Delta E$ 
6   | if  $\Delta E < 0$  then
7   |   | se acepta el movimiento a una solucion mejor
8   | else
9   |   | se acepta el movimiento a una solucion mala con
   |   | probabilidad= $e^{-\Delta/T}$ 

```

---

decrementa. Si  $T=0$ , ningún movimiento malo es aceptado y la búsqueda es equivalente a una búsqueda local (hill climbing).

### 3.3.2 ESQUEMA DE ENFRIAMIENTO

El esquema de enfriamiento define para cada paso del algoritmo la temperatura actual. Tiene un gran impacto en el éxito del algoritmo de optimización del recocido simulado. La funcionalidad del recocido simulado es muy sensible a las elecciones del esquema de enfriamiento.

#### TEMPERATURA INICIAL

Si la temperatura inicial es muy alta, la búsqueda sería como una búsqueda local aleatoria, por otro lado, si la temperatura es muy baja, la búsqueda se convierte en una búsqueda local, tomando solamente los planes mejores. Por esto es importante mantener un balance entre estos dos extremos. La temperatura inicial no debe ser muy alta para conducir a una búsqueda aleatoria por un periodo de tiempo pero lo suficientemente alta para permitir movimientos peores. Si se tiene

una aceptación de todos los movimientos malos durante la fase inicial del algoritmo el tiempo computacional de ejecución se vuelve muy alto, de ahí la importancia de determinar este valor de manera adecuada.

#### ENFRIAMIENTO

En el recocido simulado, la temperatura es decrementada de manera gradual, hasta llegar al valor cero, o el determinado previamente. Existe una estrecha relación entre la calidad de las soluciones obtenidas y la rapidez del esquema de enfriamiento. Si la temperatura se decrementa lentamente, se pueden obtener mejores soluciones pero con un tiempo computacional más significativo. Existen diversos métodos para realizar esta actualización.

La temperatura puede ser actualizada de distintas maneras, uno de los métodos utilizados es el esquema geométrico, en el cual se utiliza la fórmula:

$$T = \alpha T$$

Donde  $\alpha$  toma un valor entre 0 y 1. Es la función de enfriamiento más utilizada, trabajos anteriores han demostrado que los valores de  $\alpha$  dan mejor resultado cuando se encuentran en el rango de .5 y .99. [39].

#### CRITERIO DE PARADA

Respecto al criterio de parada, la teoría sugiere que la temperatura final sea igual a cero. Sin embargo en la práctica es común parar la búsqueda cuando la probabilidad de aceptar un movimiento es casi cero.

### 3.4 METODOLOGÍA DE SOLUCIÓN

El planificador desarrollado, el cual llamaremos POP-SA está basado en la combinación de la funcionalidad del algoritmo de orden parcial y en el comportamiento

de la metaheurística conocida como recocido simulado. Se realiza un análisis exhaustivo de ambos métodos y se aplican modificaciones orientadas a la solución de problemas de planificación.

A lo largo de esta sección se presenta la adaptación de las características del recocido simulado en el planificador y cómo éstas ayudan a resolver más problemas y con distintos resultados que algunos de los planificadores de orden parcial.

La idea de aplicar recocido simulado al algoritmo de planificación de orden parcial surge en relación a la libertad que otorga POP para seleccionar planes gracias a la estrategia que utiliza de mínimo compromiso, además de la factibilidad que presenta al adaptar el recocido simulado en la toma de decisiones de POP. Debido a que el algoritmo de orden parcial cuenta con dos componentes para el control de búsqueda de soluciones, uno de ellos, la selección de nodos o planes parciales en el cual se decide cual plan se estará trabajando como siguiente paso de la solución y el otro punto es la selección de fallas a tratar [14]. El atacar estos puntos de decisión ha demostrado ser útil en cuanto al funcionamiento de los planificadores. Es por ello que en esta sección se muestra como orientar al planificador a tomar decisiones para determinar la elección del plan siguiente. En trabajos anteriores se han implementado estrategias para la selección de planes, en su mayoría utilizando clasificación de planes en base al número de pasos para llegar a la solución, número de condiciones abiertas ó número de fallas en el plan parcial [29].

El propósito de aplicar recocido simulado a POP es el de permitirle al algoritmo de orden parcial explorar otras regiones del espacio de búsqueda. Al permitirle considerar soluciones malas, estamos siguiendo la intuición del recocido simulado de considerar soluciones no antes vistas y debido a que el espacio de búsqueda de POP es grande (PSPACE en general) creemos que su bajo porcentaje de solución de problemas (54 %) se debe en parte a que no explora lo suficiente y en el lugar adecuado. Al implementar recocido simulado buscamos abarcar un espacio de soluciones que con el ascenso de colinas original no se explora.

Nuestra implementación utiliza un reinicio en la temperatura del recocido simulado, esta técnica ha probado ser útil ya que la aplicación de recalentamientos permite al recocido simulado moverse en distintas direcciones y encontrar una mejor solución [40]. La temperatura inicial del algoritmo es asignada en base a experimentaciones previas e información en la literatura donde se determinó que el tamaño del grafo del problema es un dato que proporciona buenos resultados, en este caso se utilizan los valores de 10, 100 y 1000. También se realizó investigación y experimentación con los valores de  $\lambda$  para definir el decremento de temperatura, los valores utilizados son .5, .9 y .99, ya que éstos han demostrado dar resultados satisfactorios en el esquema geométrico donde la temperatura  $T = \lambda T$ .

Tal como se observa en el algoritmo de orden parcial, los dos puntos de decisión más importantes son la selección de un nuevo plan parcial a trabajar y la selección de una falla del conjunto de fallas correspondiente al plan seleccionado previamente. En el presente trabajo se realizan modificaciones al primer punto, la selección de un plan nuevo a trabajar, el cual se elige mediante la técnica del recocido simulado como se presenta a continuación. El primer paso para realizar la adaptación del algoritmo es la selección de parámetros a utilizar. Tal como se describió en el capítulo 2, el recocido simulado debe contar con un vecindario, costo, función probabilística y sistemas de enfriamiento, cada uno de estos puntos se define como sigue.

Selección de parámetros:

- Función de vecindario: cada uno de los nodos vecindario está dado por los refinamientos que son aplicables al plan con el que se está trabajando actualmente. A lo largo del desarrollo del algoritmo se implementaron diversas metodologías para definir el vecindario a trabajar.
  1. Total de planes pendientes (refinamientos a los planes que se han generado hasta el momento actual del problema).
  2. Hijos del nodo actual (refinamientos del plan en el que se encuentra posicionado el problema).

### 3. Clasificación de planes del vecindario de acuerdo al costo de los mismos.

Las estrategias 1, 2 y 3 están definiendo el espacio de búsqueda. La opción 1 es la menos conservadora ya que incluye todo el espacio de búsqueda, pero también podría ser la más costosa. La opción 2 es la más conservadora, puesto que solamente incluye a los hijos del nodo actual e intuitivamente explora el espacio de una manera más restringida, mientras que la tercera opción es intermedia puesto que, aunque considera todos los planes, trata de balancear la probabilidad de seleccionar un plan bueno al considerar los costos de cada plan para asignarles a éstos costos, en vez de los planes mismos, una probabilidad de seleccionar a un elemento de la clase.

- Definición del costo para la comparación de estados: el costo es asignado en base a heurísticas que nos indican las características de los planes parciales generados, pueden incluir desde el número de acciones en el plan, condiciones abiertas con las que cuenta el plan y costo estimado desde el plan de inicio hasta el plan final.
- Diseño del esquema de enfriamiento: define para cada paso del algoritmo la temperatura actual, los elementos a considerar son la temperatura inicial, la función de enfriamiento y la temperatura final, la cual define el criterio de parada del algoritmo. Tiene un papel esencial en la eficiencia y efectividad del algoritmo.
- Función probabilística para la aceptación de un estado: está dada por la distribución  $P=e^{-\Delta/T}$ . Es el elemento principal del recocido simulado, habilita a los estados no tan buenos para ser seleccionados [39].

La aceptación de los movimientos vecinos es el medio a través del cual al sistema se le permite escapar de quedar atrapado en un óptimo local mediante la función probabilística de aceptación a un vecino no tan bueno. La probabilidad de aceptación es proporcional a la temperatura  $T$  e inversamente proporcional al cambio de energía en la función objetivo  $\Delta E$ . A altas temperaturas la probabilidad de aceptar peores

movimientos vecinos es alta. Si la temperatura  $T = \infty$  todos los movimientos serán aceptados. A bajas temperaturas la probabilidad de aceptar peores movimientos decreciente. Si  $T=0$ , ningún movimiento malo es aceptado y la búsqueda es equivalente a una búsqueda local (hill climbing), lo que equivale a ejecutar el algoritmo POP original

Se utiliza la estrategia de reinicio de temperatura o recalentamiento para salir de los óptimos locales. El recalentamiento implica un incremento en la temperatura del recocido simulado después de la convergencia a una temperatura inicial [5]. La finalidad de el reinicio de la temperatura es que el recalentamiento podría ubicar la solución en una parte de la búsqueda del espacio de soluciones que está muy lejos del óptimo local y el recalentamiento continuo no permite moverse en la dirección correcta. El reinicio permite al algoritmo moverse en diferentes direcciones y encontrar una mejor solución.

### 3.4.1 PLANIFICADORES DESARROLLADOS

Durante el diseño del algoritmo unificado se desarrollaron distintas versiones de planificadores POP-SA. Los planificadores utilizan el funcionamiento principal de orden parcial, algoritmo 2.1 y recocido simulado, algoritmo 3.1. Sin embargo, conforme el avance de la experimentación se agregaron funcionalidades, teniendo la intuición de que con ellas se pueden obtener mejores resultados. A continuación se muestra el detalle de cada uno de los planificadores obtenidos. Los planificadores desarrollados cuentan con los siguientes elementos:

- Temperatura inicial: es el valor asignado a la temperatura al inicio del problema.
- Plan aleatorio: Plan seleccionado de manera aleatoria del vecindario correspondiente al plan actual.
- Vecindario: son los planes correspondientes a los refinamientos del plan actual.



- Plan actual: es el nodo o plan en el que esta posicionado en el punto específico del problema.
- Costo: valor asignado para determinar que tan bueno es un plan, éste dato está dado por la metodología LIFO ( last in first out) en donde los planes creados al final son los que tienen un menor costo, y por lo tanto son los primeros en ser elegidos.
- $\lambda$ : es el factor para realizar el decremento de la temperatura.
- MejorPlanVecindario: selecciona el plan del vecindario que contenga el menor costo.

#### SELECCIÓN DE PLAN BASADO EN LOS PLANES PENDIENTES

Se desarrolla posteriormente una segunda versión del planificador, la cual llamaremos POP-SA-1. Al igual que la versión anterior, basa su funcionamiento en la selección del plan a trabajar, sin embargo su característica principal es que el vecindario se compone del total de planes pendientes generados hasta el punto actual ,lo que ofrece un rango más amplio de aleatoriedad en la selección de plan a trabajar. Su funcionamiento se explica en el algoritmo 3.3

#### SELECCIÓN DE PLAN BASADO EN EL NODO ACTUAL

En la primera versión creada, la cual llamaremos POP-SA-2, como se ha mencionado en capítulos anteriores, se desarrolla la unificación de las metodologías de planificación de orden parcial y recocido simulado, las cuales interactúan en un solo planificador. Su funcionamiento se basa en la selección del plan parcial, el cuál es elegido de manera aleatoria del vecindario correspondiente a los refinamientos aplicables al plan actual ó hijos del nodo actual. El detalle de su funcionamiento se explica en el algoritmo 3.1. El plan del vecindario se toma bajo las siguientes circunstancias: Si la temperatura es mayor a 1 (línea 1), se valida el costo del plan aleatorio contra

**Algoritmo 3.2:** Algoritmo POP-SA-1

---

```

1 while (plan actualizado=true) do
2   if Temperatura > 1 then
3     Seleccionar PlanAleatorio del vecindario (Total de planes
4     pendientes generados hasta el momento).
5     if (Costo_PlanAleatorio < Costo_PlanActual) then
6       Se acepta el plan aleatorio como plan actual
7     else
8       Se acepta el plan aleatorio con probabilidad  $P=e^{-\Delta/T}$ 
9       Temperatura= Temperatura *  $\lambda$ 
10    if Temperatura < 1 then
11      Temperatura = Temperatura_Inicial
12      PlanActual = MejorPlanVecindario

```

---

el plan actual (línea 4), si el plan seleccionado aleatoriamente es mejor que el plan actual actual (tiene un costo menor), se acepta (línea 5) , si es peor, se acepta en base a la probabilidad dada (línea 7), en caso de que la temperatura se encuentre en un valor menor a 1, se hace un reinicio de temperatura volviendo a su valor inicial y se tomará siempre el plan con el menor costo de los correspondientes al vecindario (líneas 9 , 10 y 11).

## SELECCIÓN DE PLAN CON CLASIFICACIÓN DE COSTOS

Esta versión toma el nombre de POP-SA-MAP. En base a los resultados obtenidos por los planificadores anteriores, se realizan modificaciones en cuanto a la clasificación de planes del vecindario, para orientar al algoritmo a la toma de decisiones, ésta clasificación se realiza de acuerdo al costo de cada uno de los planes, tratando de nivelar la probabilidad de ser seleccionados identificando el costo para cada plan, y utilizándolos para clasificarlos por una misma probabilidad.

**Algoritmo 3.3:** Algoritmo POP-SA-2

---

```

1 while (plan actualizado=true) do
2   if Temperatura > 1 then
3     Seleccionar PlanAleatorio del vecindario correspondiente a los
4       refinamientos del plan actual.
5     if (Costo_PlanAleatorio < Costo_PlanActual) then
6       | Se acepta el plan aleatorio como plan actual
7     else
8       | Se acepta el plan aleatorio con probabilidad  $P=e^{-\Delta/T}$ 
9       | Temperatura= Temperatura *  $\lambda$ 
10    if Temperatura < 1 then
11      | Temperatura = Temperatura_Inicial
12      | PlanActual = MejorPlanVecindario

```

---

En el algoritmo 3.4 se muestra el detalle de la tercera versión POP-SA-MAP.

En la figura 3.2 podemos observar un ejemplo de un grafo, en el cual estamos situados en PlanActual, y el vecindario corresponde a los planes parciales: Plan1, Plan2, Plan3, Plan4 y Plan5, los cuales cuentan con un costo asignado, en donde un mejor plan contiene el costo con menor valor.

Para elegir el siguiente plan a trabajar, se realiza una clasificación de costos, como se observa en la figura 3.3, los cuáles son elegidos aleatoriamente, los costos funcionan como un índice para la selección del plan, de esta manera a cada plan se le da la misma probabilidad de ser seleccionado, independientemente del costo.

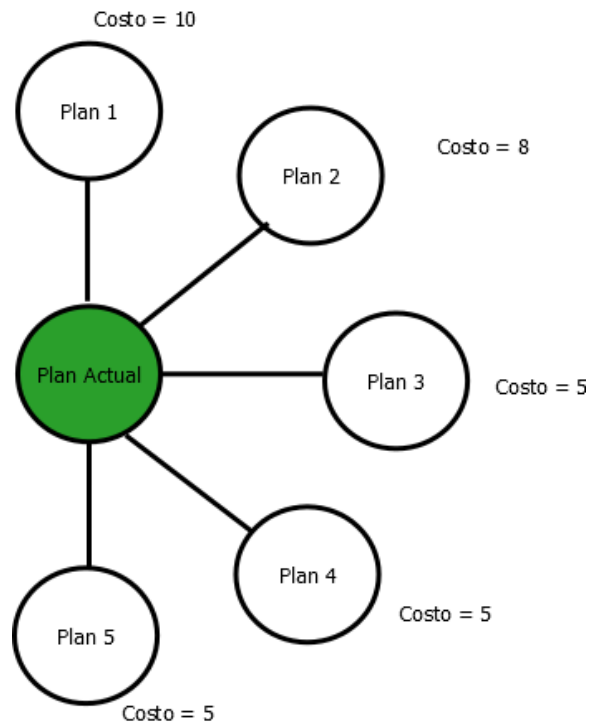


Figura 3.2: Planes parciales

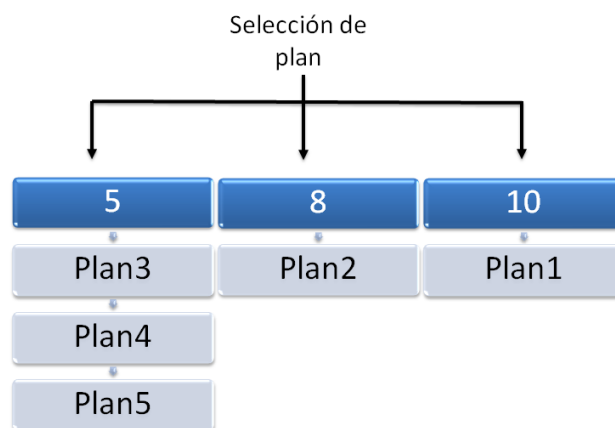


Figura 3.3: Costo de planes

---

**Algoritmo 3.4:** Algoritmo POP-SA-MAP

---

```
1 PlanesPendientesCosto = planes pendientes clasificados por costo, en
   donde el costo es el índice de los planes con éste valor.
2 while (plan actualizado=true) do
3   if Temperatura > 1 then
4     Seleccionar aleatoriamente un costo (índice) y tomar el primer plan
       generado con éste costo.
5     if (Costo_PlanAleatorio < Costo_PlanActual) then
6       | Se acepta el plan aleatorio como plan actual
7     else
8       | Se acepta el plan aleatorio con probabilidad  $P=e^{-\Delta/T}$ 
9       |  $Temperatura= Temperatura * \lambda$ 
10  if Temperatura < 1 then
11    | Temperatura = Temperatura_Inicial
12    | Seleccionar aleatoriamente un costo (índice) y tomar el primer plan
       generado con éste costo
13    | PlanActual= PlanAleatorio; se acepta siempre el plan aleatorio.
```

---

## CAPÍTULO 4

# EXPERIMENTOS Y RESULTADOS

---

En esta fase del proyecto se identifican los problemas y dominios utilizados para la experimentación y son ejecutados por los planificadores generados para su solución. Se desarrollan tres versiones de planificadores, los cuales basan su funcionamiento en la elección de planes del vecindario, el cual se determina de la siguiente manera:

1. POP-SA-1 : Total de planes pendientes (refinamientos a los planes que se han generado hasta el momento actual del problema). Algoritmo 3.2.
2. POP-SA-2: Hijos del nodo actual (refinamientos del plan en el que se encuentra posicionado el problema). Algoritmo 3.3.
3. POP-SA-MAP: Clasificación de planes del vecindario de acuerdo al costo de los mismos. Algoritmo 3.4.

### 4.1 EXPERIMENTACIÓN

Los planificadores son desarrollados en lenguaje de programación C++. Las características del equipo utilizado para la experimentación son las siguientes:

- Procesador: Intel Core i7-2640M CPU 2.80GHz
- Disco duro : 49,9 GB
- Memoria RAM 5,8 GiB

- Sistema operativo 32-bit

Se llevaron a cabo experimentos con 51 problemas y 18 dominios ,realizando 4 corridas para cada uno. Los dominios y modelos utilizados corresponden al catálogo de problemas de VHPOP (en el capítulo 2 se mencionan las características de este planificador)[11].

A continuación se muestra la descripción de los problemas seleccionados para la evaluación:

- briefcase-world-domain: consiste en encontrar un plan para abrir un maletín con un número de cerrojos en donde cada cerrojo puede ser configurado en abierto o cerrado. Si un seguro de apertura se cambia se convierte en cerrado y si un seguro de cerrado se cambia se convierte en abierto. El problema indica que el maletín solo está abierto cuando todos los cerrojos están abiertos.
- bulldozer-domain: una persona tiene que entrar a un vehículo, manejar a algún lugar, salir y regresar a otro lugar. Se presenta un problema de recursión en este dominio debido a que los caminos y puentes van en dos direcciones.
- ferry-domain: transporta un número de autos desde su inicio hasta una ciudad objetivo utilizando un barco. Cada ciudad es accesible desde cualquier otra ciudad, los autos pueden ser desembarcados y el barco solamente puede transportar un auto a la vez.
- flat-tire-domain: reemplazar una llanta ponchada con una de repuesto. Incluye utilización de herramientas y las acciones para llevar a cabo el proceso.
- fridge-domain: consta de un refrigerador y tornillos, los cuales son manipulados de tal manera que el refrigerador se apague cuando éstos son removidos de la placa posterior.
- grid-domain : problema en el que un solo robot se mueve entre ubicaciones en una malla en forma de mapa, las ubicaciones pueden ser bloqueadas y

existen llaves que pueden ser recolectadas para permitir acceso. El objetivo del problema implica el transporte de la llave para determinados lugares.

- gripper-domain: colección de pelotas que tienen que ser transportadas por un robot con dos pinzas de un cuarto al cuarto adyacente.
- hanoi-domain: conocido problema de las torres de hanoi que consiste en apilar discos en tres estacas distintas en un tablero.
- ho-world-domain: consiste en reparar problemas de plomería.
- logistics-domain: problema de transporte que involucra aviones y camiones, los camiones tienen restringido el movimiento en las ciudades y los aviones en los aeropuertos.
- mcd-blocks-world-domain: problema clásico del mundo de los blocks, el cual consiste en acomodar los cubos en ciertas configuraciones.
- monkey-domain: implica monos moviéndose desde distintos lugares para conseguir plátanos.
- robot-domain: robot transporta un objeto de un lugar a otro.
- rocket-domain: un cohete tiene que transportar objetos a distintos lugares, verifica si cuenta o no con el combustible necesario.
- simple-blocks-domain: serie de bloques en una mesa, configurados de distintas maneras.
- trains-domain: transportación de carga entre ciudades en un tren.
- uni-briefcase-world-domain: transportación de objetos en un maletín a distintos lugares.

En la tabla 4.1 se observa la clasificación de problemas correspondientes a cada uno de los dominios utilizados.



## 4.2 RESULTADOS OBTENIDOS

Se realiza la experimentación ejecutando 4 corridas por cada bloque de problemas, utilizando el planificador POP-SA-1, POP-SA-2, POP-SAM y VHPOP. Los parámetros a utilizar son:

- Temperatura : 10, 100, 1000
- Enfriamiento  $\lambda$  : .5, .9, .99

En la tabla 4.2 se encuentran los resultados obtenidos al ejecutar los planificadores correspondientes al recocido simulado, POP-SA-1, POP-SA-2 y POP-SAM. Se muestran también los valores de temperatura y rangos de enfriamiento utilizados, obteniendo con ello el porcentaje de planes resueltos de los 51 problemas totales establecidos.

El apartado correspondiente al tiempo, columna 5, está especificado en milisegundos. Es el tiempo promedio para los 51 problemas y sus 4 corridas. Se fija un tiempo límite de solución para cada uno de los problemas de 2 minutos, el cuál fué establecido como criterio de parada para la búsqueda de una solución. En caso de que un problema no tenga solución se asigna un tiempo de cero para ese problema en específico.

En la columna 6, correspondiente a las acciones, se observa el promedio de acciones utilizadas para la generación de un plan. Es importante mencionar que en caso de que un problema no encuentre solución se le asignó un valor de cero como acciones generadas.

Dado que el dominio es una parte esencial en la planificación es importante incluir el comportamiento obtenido por cada uno de los planificadores.

En la figura 4.1 podemos observar resultado de problemas resueltos por do-

minio. La línea roja corresponde al planificador POP-SA1, en color verde se muestra el acumulado de POP-SA-1. En los resultados acumulados se toman los mejores valores para cada una de las configuraciones de temperatura y enfriamiento utilizados en POP-SA-1 para las 4 corridas. La línea azul corresponde al planificador VHPOP, cuyo funcionamiento se explica a detalle en el capítulo 2. Tanto POP-SA1 como su acumulado se encuentran por debajo de VHPOP en dominios como flat-tire, gripper, hanoi-1, monkey, simple-blocks y uni-briefcase-world. Sin embargo muestra un ligero incremento en briefcase-world

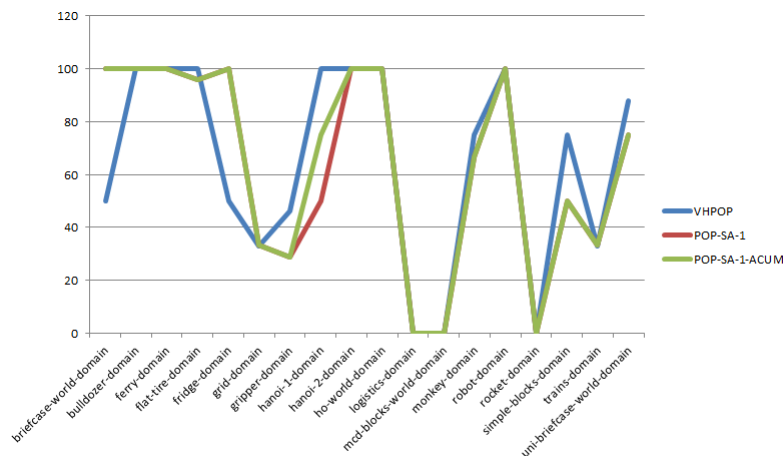


Figura 4.1: Resultados POPSA1 por dominio

En la gráfica 4.2 se visualiza el dominio y problemas resueltos con VHPOP, POP-SA-2 y se agrega también el acumulado de resultados obtenidos en POP-SA, en color verde. Como se muestra en la gráfica se tiene un importante incremento en el dominio briefcase-world, en donde vhhpop logra resolver el 50 % de los problemas, POP-SA-2 el 94 % y el acumulado POP-SA-2 el 100 %, fridge-domain donde VHPOP obtiene el 50 % y tanto POP-SA como su acumulado logran el 100 %, además de simple-blocks y trains, por mencionar algunos.

En la gráfica 4.3 se da a conocer el promedio de problemas resueltos por dominio para el planificador POP-SA-MAP, cuya característica es la clasificación de planes de acuerdo al costo de los mismos.

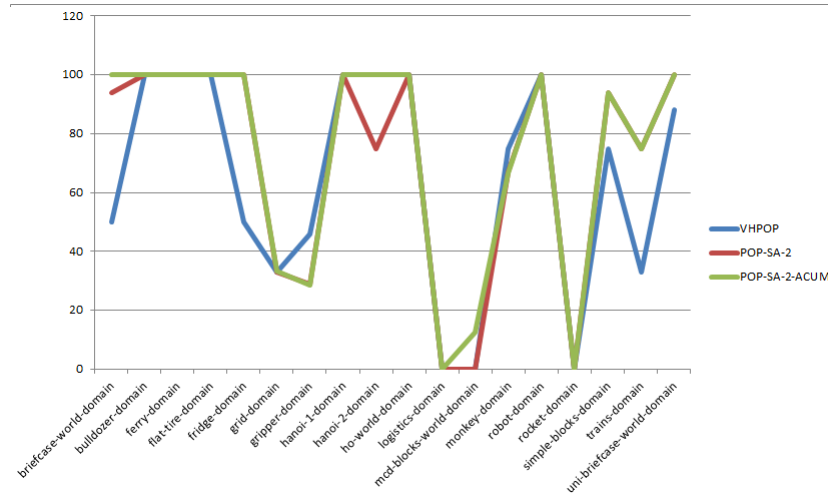


Figura 4.2: Resultados POPSA2 por dominio

Se tienen resultados positivos en el dominio briefcase, fridge, grid, y simple blocks, con una mejora de 40, 50, 17, 25 y 12% respectivamente en comparación con VHPOP.

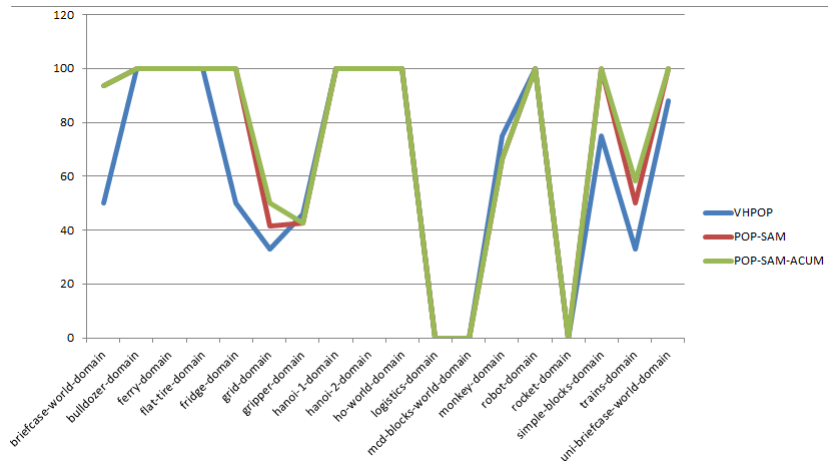


Figura 4.3: Resultados POP SAM dominio

Por último en la figura 5.4 se observa en una gráfica de barras el comportamiento para cada uno de los planificadores utilizados y el número de problemas resueltos en promedio para las 4 corridas.

Los resultados son: POP-SA-1 y su acumulado con 57 y 58 % respectivamente, VHPOP obtiene un 59% de problemas resueltos, POP-SA-2 y su acumulado con 66 y 67% y por último POP-SAM con un 68% y su acumulado sube al 69%. Éste último logra superar a VHPOP con un 10 %.

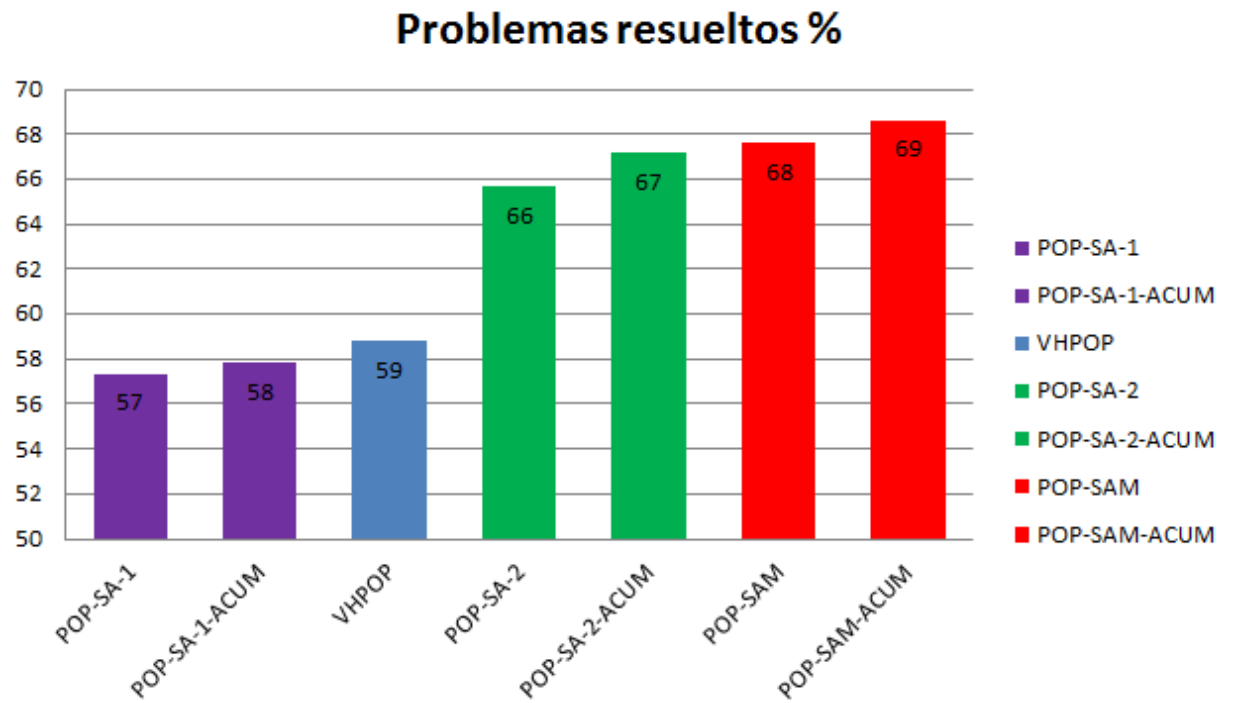


Figura 4.4: Resultados planificadores

<b>Dominio</b>	<b>Problemas</b>
briefcase-world-domain	get-paid, get-paid2, get-paid3, get-paid4
bulldozer-domain	get-back-jack
ferry-domain	test-ferry
flat-tire-domain	fix1, fix2, fix3, fix4, fix5, fixit
fridge-domain	fixa, fixb
grid-domain	sg1, sg2, sg3
gripper-domain	gripper10, gripper12, gripper2, gripper20, gripper4, gripper6, gripper8
hanoi-1-domain	hanoi-1
hanoi-2-domain 3	hanoi-2
ho-world-domain	ho-demo
logistics-domain	log-a, log-b, log-c, log-d
mcd-blocks-world-domain	mcd-sussman-anomaly, mcd-tower-invert
monkey-domain	monkey-test1, monkey-test2, monkey-test3
robot-domain	r-test1, r-test2
rocket-domain	rocket-ext-a, rocket-ext-b
simple-blocks-domain	bw-large-as, bw-large-bs, bw-large-cs, bw-large-ds
trains-domain	trains1, trains2, trains3
uni-briefcase-world-domain	uget-paid, uget-paid2, uget-paid3, uget-paid4

Tabla 4.1: Problemas y dominios

Planificador	Temp.	Enfriamiento	Planes %	Tiempo	Acciones
<b>POP-SA-1</b>	<b>10</b>	<b>50</b>	<b>57</b>	<b>52272.69</b>	<b>3.10</b>
POP-SA-1	10	90	54	56495.75	2.81
POP-SA-1	10	99	54	58184.64	2.86
POP-SA-1	100	50	56	56209.59	2.94
POP-SA-1	100	90	50	63015.87	2.48
POP-SA-1	100	99	45	68003.56	2.28
POP-SA-1	1000	50	55	56564.72	2.91
POP-SA-1	1000	90	51	62887.64	2.61
POP-SA-1	1000	99	42	71978.07	2.08
POP-SA-2	10	.50	57	54382.41	3.33
POP-SA-2	10	.90	64	46174.87	3.86
POP-SA-2	10	.99	63	47778.77	4.32
POP-SA-2	100	.50	62	47911.73	3.71
POP-SA-2	100	.90	62	49269.56	3.92
POP-SA-2	100	.99	53	59958.98	3.69
<b>POP-SA-2</b>	<b>1000</b>	<b>.50</b>	<b>66</b>	<b>45099.49</b>	<b>4.05</b>
POP-SA-2	1000	.90	62	51073.09	3.91
POP-SA-2	1000	.99	49	63131.91	3.53
POP-SAM	10	.50	62	48031.61	3.90
POP-SAM	10	.90	67	42467.41	4.34
<b>POP-SAM</b>	<b>10</b>	<b>.99</b>	<b>68</b>	<b>41391.59</b>	<b>4.49</b>
POP-SAM	100	.50	60	50735.48	3.71
POP-SAM	100	.90	58	58087.77	3.51
POP-SAM	100	.99	53	59781.52	2.80
POP-SAM	1000	.50	56	54613.06	3.50
POP-SAM	1000	.90	53	58034.03	3
POP-SAM	1000	.99	50	61486.51	2.68

Tabla 4.2: Tabla de promedios temperatura y enfriamiento

## CAPÍTULO 5

# CONCLUSIONES Y TRABAJO FUTURO

---

### 5.1 CONCLUSIONES

En base a los resultados obtenidos en la experimentación previa se llega a la conclusión de que al aplicar el recocido simulado en el algoritmo de orden parcial tiene un impacto importante en el comportamiento del mismo, ya que permite realizar una búsqueda mas exhaustiva en el total de las posibles soluciones logrando así tomar buenas decisiones al momento de generar una solución al problema.

La configuración de temperatura inicial y esquema de enfriamiento son un elemento importante en el comportamiento del planificador, es por ello que se seleccionaron rangos específicos que mostraron dar los mejores resultados en base a las experimentaciones previas en los planificadores desarrollados. Los valores utilizados fueron:

- Temperatura inicial de 1000 en POP-SA y 10 en POP-SAM.
- Enfriamiento geométrico dado por la fórmula  $T = \lambda T$ , donde  $\lambda$  toma un valor de .5 en POP-SA y .99 en POP-SAM
- Reinicio de temperatura a su valor inicial cuando ésta llega a un valor de cero. Este apartado es implementado debido a que al bajar la temperatura, la probabilidad de elegir un plan malo es casi cero, haciendo el reinicio, damos oportunidad a que se continúe con la búsqueda de soluciones potenciales.

El planificador demuestra dar buenos resultados cuando se utiliza la función de asignación de costo de planes correspondiente a LIFO (last in first out), en donde tienen un mejor costo aquellos planes que son generados al final.

Se demuestra además que el tiempo límite asignado de 2 minutos, es suficiente para hacer una exploración en un espacio bastante amplio de posibles soluciones (en algunos casos se expande a más de 10 mil nodos (planes parciales). Así como también el tiempo total utilizado para dar solución a un problema es menor en el planificador POP-SAM.

Al realizar un comparativo con otro planificador de orden parcial se observa un incremento por parte del planificador POP-SA del 7% del total de problemas resueltos para cada uno de ellos, cada uno con las características mencionadas anteriormente. Logrando resolver el 66% de los problemas establecidos, sin embargo con la segunda versión del recocido simulado en el planificador POP-SAM se logra incrementar un 9% con respecto a VHPOP. Se lleva a cabo también un análisis de resultados del planificador POP-SAM en donde acumula el total de problemas resueltos para cada una de las configuraciones de temperatura inicial y esquema de enfriamiento utilizados en las pruebas. Dando como resultado un mejor rendimiento logrando incrementar un 10% con respecto a VHPOP, es decir 69% de problemas resueltos del total asignado.

## 5.2 TRABAJO FUTURO

Como trabajo futuro se proponen los siguientes puntos:

- Realizar experimentación con diferentes niveles de temperatura inicial y enfriamiento para el recocido simulado para obtener la configuración óptima que permita la solución a todos los problemas propuestos.
- Implementar técnicas para asignar costo a cada uno de los planes y hacerlos elegibles en base a la factibilidad de los mismos para conducir a una solución.



- 
- Incorporar un conjunto de problemas distintos a los utilizados actualmente y observar el comportamiento obtenido de acuerdo al tipo de dominio y modelación del problema utilizados.
  - Atacar los distintos puntos de decisión en el algoritmo de orden parcial aplicando técnicas metaheurísticas.
  - Comparación de resultados con planificadores on metodologías de solución distintas al algoritmo de orden parcial.
  - Implementación de técnicas para la selección de fallas a reparar.
  - Clasificación de planes en base a costos, dando mayor probabilidad de ser elegidos a aquellos planes que tengan un menor costo, es decir, un mejor plan.

## APÉNDICE A

# RESULTADOS: PLANIFICADOR Y DOMINIO

---

Información referente a los resultados promedio por dominio obtenidos para cada uno de los planificadores utilizados, de un total de 4 corridas.

- Porcentaje de problemas resueltos por dominio (columna 3)
- El tiempo está asignado en milisegundos.
- El límite establecido para cada uno de los problemas es de 2 minutos por corrida. Es el criterio de parada en caso de no encontrar una solución en este lapso de tiempo.
- Los problemas que no pudieron ser solucionados se les asigna cero como número de acciones generadas.

Los datos obtenidos se encuentran en las tablas A.1, A.2, A.3 y A.4.

<b>Planificador</b>	<b>Dominio</b>	<b>Problemas %</b>	<b>Tiempo</b>	<b>Acciones</b>
VHPOP	briefcase-world-domain	50	60004.25	1.5
VHPOP	bulldozer-domain	100	7976.25	9
VHPOP	ferry-domain	100	9	7
VHPOP	flat-tire-domain	100	872.17	4
VHPOP	fridge-domain	50	60002	1.5
VHPOP	grid-domain	33.33	20038.5	2
VHPOP	gripper-domain	46.43	63254.37	3
VHPOP	hanoi-1-domain	100	4926	7
VHPOP	hanoi-2-domain	100	812	7
VHPOP	ho-world-domain	100	0	4
VHPOP	logistics-domain	0	120003	0
VHPOP	mcd-blocks-world-domain	0	120003	0
VHPOP	monkey-domain	75	30009.75	4.92
VHPOP	robot-domain	100	10	5
VHPOP	rocket-domain	0	120003.5	0
VHPOP	simple-blocks-domain	75	24054.0625	7.75
VHPOP	trains-domain	33.33	50683.92	2.33
VHPOP	uni-briefcase-world-domain	87.5	30021.5	5.625

Tabla A.1: Resultados VHPOP

<b>Planificador</b>	<b>Dominio</b>	<b>Problemas</b>	<b>Tiempo</b>	<b>Acciones</b>
POP-SA1	briefcase-world-domain	100	7.75	4.5
POP-SA1	bulldozer-domain	100	1140	9
POP-SA1	ferry-domain	100	4	7
POP-SA1	flat-tire-domain	95.83	9491.42	3.5
POP-SA1	fridge-domain	100	406	4.5
POP-SA1	grid-domain	33.33	80051	2
POP-SA1	gripper-domain	28.57	85723.85	1.43
POP-SA1	hanoi-1-domain	50	69900	3.5
POP-SA1	hanoi-2-domain	100	12094.5	7
POP-SA1	ho-world-domain	100	0	4
POP-SA1	logistics-domain	0	120003.75	0
POP-SA1	mcd-blocks-world-domain	0	120003	0
POP-SA1	monkey-domain	66.67	40010.33	4.67
POP-SA1	robot-domain	100	20	5
POP-SA1	rocket-domain	0	120003.5	0
POP-SA1	simple-blocks-domain	50	60691.75	3.8125
POP-SA1	trains-domain	33.33	80100.67	1.677
POP-SA1	uni-briefcase-world-domain	75	30397.25	4.8125

Tabla A.2: Resultados POP-SA1

<b>Planificador</b>	<b>Dominio</b>	<b>Problemas</b>	<b>Tiempo</b>	<b>Acciones</b>
POP-SA2	briefcase-world-domain	93.75	7500.9375	4.1875
POP-SA2	bulldozer-domain	100	439	9.25
POP-SA2	ferry-domain	100	19	7
POP-SA2	flat-tire-domain	100	4131.92	4.46
POP-SA2	fridge-domain	100	85	5
POP-SA2	grid-domain	33.34	80594	2
POP-SA2	gripper-domain	28.57	85722	1.43
POP-SA2	hanoi-1-domain	100	22095.25	7
POP-SA2	hanoi-2-domain	75	57628.25	5.25
POP-SA2	ho-world-domain	100	0	4
POP-SA2	logistics-domain	0	120005.75	0
POP-SA2	mcd-blocks-world-domain	0	120004.5	0
POP-SA2	monkey-domain	66.67	40004.67	5.42
POP-SA2	robot-domain	100	11.5	5
POP-SA2	rocket-domain	0	120006.5	0
POP-SA2	simple-blocks-domain	93.75	25044.25	9.75
POP-SA2	trains-domain	75	44747	4.58
POP-SA2	uni-briefcase-world-domain	100	2147.8125	6.4375

Tabla A.3: Resultados POP-SA2

<b>Planificador</b>	<b>Dominio</b>	<b>Problemas</b>	<b>Tiempo</b>	<b>Acciones</b>
POP-SAM	briefcase-world-domain	93.75	7508.4375	4.375
POP-SAM	bulldozer-domain	100	222	10
POP-SAM	ferry-domain	100	13	7
POP-SAM	flat-tire-domain	100	514.83	4.46
POP-SAM	fridge-domain	100	89.5	5
POP-SAM	grid-domain	41.67	71629.5	2.83
POP-SAM	gripper-domain	42.86	69968.75	3
POP-SAM	hanoi-1-domain	100	16849.75	7.5
POP-SAM	hanoi-2-domain	100	5688.25	7
POP-SAM	ho-world-domain	100	0	4.25
POP-SAM	logistics-domain	0	120003	0
POP-SAM	mcd-blocks-world-domain	0	120003	0
POP-SAM	monkey-domain	66.66	40010.33	4.667
POP-SAM	robot-domain	100	55	5.25
POP-SAM	rocket-domain	0	120003	0
POP-SAM	simple-blocks-domain	100	7986.375	12
POP-SAM	trains-domain	50	74494.42	3.25
POP-SAM	uni-briefcase-world-domain	100	3658.19	6.75

Tabla A.4: Resultados POP-SAM

# BIBLIOGRAFÍA

---

- [1] AIPS-98 PLANNING, C. C., «PDDL The Planning Domain Definition Language version 1.2», recurso libre, disponible en <http://homepages.inf.ed.ac.uk/mfourman/tools/propplan/pddl.pdf>, 1998.
- [2] BORNSTEIN, N. y S. SVANE, *Heuristics in generic planning using PDDL*, 2011.
- [3] BRANDÃO DE OLIVEIRA, H. C., G. CRISPIM VASCONCELOS y G. BATOS ALVARENGA, «A Multi-Start Simulated Annealing Algorithm for the Vehicle Routing Problem with Time Windows», *IEEE Computer Society Washington*, 2006.
- [4] CORMEN, T. H., C. E. LEISERSON, R. L. RIVERST y C. STEIN, *Introduction to algorithms*, 2009.
- [5] CRUZ CHAVEZ, M. A. y J. FRAUSTO-SOLIS, «Simulated Annealing with Restart to Job Shop Scheduling Problem Using Upper Bounds», *Artificial Intelligence and Soft Computing - ICAISC 2004*, **3070**, 2004.
- [6] DEKKERS, A. y E. AARTS, «Global optimization and simulated annealing», *Mathematical Programming*, **50**, 1991.
- [7] DRÉO, J., A. PÉROWSKI, P. SIARRY y E. TAILLARD, *Metaheuristics for hard optimization*, 2003.
- [8] ESTLIN, T. A. y R. J. MOONEY, «Hybrid Learning of Search Control for Partial-Order Planning», recurso libre, disponible en <http://www.cs.utexas.edu/ml/papers/dolphin-ewsp-95.pdf>, 1996.

- 
- [9] GOLDEN, K., O. ETZIONI y D. WELD, «Planning with Execution and Incomplete Information», recurso libre, disponible en <http://homes.cs.washington.edu/weld/papers/tr96-01-09.pdf>, 1996.
- [10] GROSAN, C., A. AJITH y H. ISHIBUCHI, *Hybrid evolutionary algorithms*, 2007.
- [11] H İAKAN L. S., Y. y R. G. SIMMONS, «VHPOP: Versatile Heuristic Partial Order Planner», *Journal of Artificial Intelligence Research*, **20**, 2003.
- [12] HENRY WINSTON, P., *Artificial Intelligence*, 1992.
- [13] ICAPS, «Home Page ICAPS», recurso libre, disponible en <http://www.icaps-conference.org/>, 2013.
- [14] JOSLIN, D. y M. E. POLLACK, «Least-Cost Flaw Repair: A Plan Refinement Strategy for Partial-Order Planning», *AAAI-94 Proceedings*, 1994.
- [15] KAMBHAMPATI, S., «A Comparative analysis of Partial Order Planning and Task Reduction Planning», recurso libre, disponible en <http://phanxipan.eas.asu.edu/htn-po-sigart.pdf>, 1995.
- [16] KORF, R. E., «Planning as a search: a quantitative approach», *Artificial Intelligence*, **33**, 1987.
- [17] L GENDREAU, M. y J.-Y. POTVIN, *Handbook of Metaheuristics*, 2010.
- [18] LONG, D. y M. FOX, «The 3rd International Planning Competition: Results and Analysis», *Journal of Artificial Intelligence*, 2003.
- [19] LONG, D. y M. FOX, «pddl2.1 : An Extension to pddl for Expressing Temporal Planning Domains», *Journal of Artificial Intelligence Research*, 2003.
- [20] M. LAVALLE, S., *Planning Algorithms*, 999<sup>a</sup> edición, Cambridge University Press, recurso libre disponible en <http://planning.cs.uiuc.edu/>, 2006.
- [21] MCALLESTER, D. y D. ROSENBLITT, «Systematic Nonlinear Planning», *In Proceedings of the Ninth National Conference on Artificial Intelligence*, 1991.



- [22] McDERMOTT, D., «The 1998 AI Planning Systems Competition», *AI Magazine*, **21**(2), págs. 35–55, 2000.
- [23] MINTON, S., J. BRESINA y M. DRUMMOND, «Total-Order and Partial-Order Planning: A Comparative Analysis», *Journal of Artificial Intelligence Research*, **2**, 1994.
- [24] NGUYEN, X. y S. KAMBHAMPATI, «Reviving Partial Order Planning», recurso libre, disponible en <http://rakaposhi.eas.asu.edu/ucpop-revive.pdf>, 2001.
- [25] NGUYEN, X., S. KAMBHAMPATI y R. S. NIGENDA, «Planning graph as the basis for deriving heuristics for plan synthesis by state space and CSP search», *Artificial Intelligence*, **135**, 2002.
- [26] PENBERTHY, J. S. y D. S. WELD, «UCPOP, A Sound, Complete, Partial Order for ADL», recurso libre, disponible en <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.6.9198rep=rep1type=pdf>, 1992.
- [27] PEOT, M. A. y D. E. SMITH, «Threat-Removal Strategies for Partial-Order Planning», *AAAI*, 1993.
- [28] POLLACK, M. E., «The uses of plans», *Artificial Intelligence*, **57**(1), págs. 43–68, 1992.
- [29] POLLACK, M. E., D. JOSLIN y M. PAOLUCCI, «Flaw Selection Strategies For Partial-Order Planning», *Journal of Artificial Intelligence Research*, **6**, 1997.
- [30] RICH, E. y K. KNIGHT, *Handbook of Metaheuristics*, 1991.
- [31] RUSSEL, S. J. y P. NORVIG, *Inteligencia artificial, un enfoque moderno*, 2004.
- [32] RUSSEL, S. J. y P. NORVIG, *Artificial Intelligence: A Modern Approach*, 2009.
- [33] RUTENBAR, R. A., «Simulated annealing algorithms, an overview», *IEEE circuits and devices magazine*, 1989.

- 
- [34] S. WELD, D., «An Introduction to Least Commitment Planning», *AI MAGAZINE*, 1994.
- [35] SCHUBERT, L. y A. GEREVINI, «Accelerating partial order planners by improving plan and goal choices», *Journal of Artificial Intelligence Research*, **5**, 1996.
- [36] SMITH, D. E., J. FRANK y A. K. JÓNSSON, «Bridging the Gap Between Planning and Scheduling», *Knowledge Engineering Review*, **15**, 2000.
- [37] SRINIVASAN, R. y A. E. HOWE, *IJCAI*, 1995.
- [38] SUBBARAO, K., «Planning as Refinement Search: A unified framework for comparative analysis of Search Space Size and Performance», *Artificial Intelligence*, **76**, 1993.
- [39] TALBI, E.-G., *Metaheuristics from design to implementation*, 2009.
- [40] TIERNEY, K., *Optimizing Liner Shipping Fleet Repositioning Plans*, 2013.
- [41] WELD, D. S., «Recent advances in planning», *AI Magazine*, **20**(2), págs. 93–123, 1999.
- [42] WILLIAMSON, M. y S. HANKS, «Flaw Selection Strategies for Value-Directed Planning», *AIPS 1996 Proceedings*, 1996.
- [43] YOUNES, H. L. y R. SIMMONS, «On the role of ground actions in refinement planning», *In Proceedings of the Sixth International Conference on Artificial Intelligence Planning and Scheduling Systems*, 2002.

# FICHA AUTOBIOGRÁFICA

---

Rosa Liliana González Arredondo

Candidato para el grado de Maestro en Ingeniería  
con especialidad en Ingeniería de Sistemas

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Tesis:

METAHEURÍSTICAS APLICADAS A LA  
PLANIFICACIÓN DE ORDEN PARCIAL

Nacida en San Nicolás de los Garza Nuevo León el 7 de Junio de 1986. Egresada como Ingeniero Administrador de sistemas en el año 2004 de la Facultad de Ingeniería Mecánica y Eléctrica de la Universidad Autónoma de Nuevo León.