Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Aplicación de técnicas de planning para la generación automática de estrategias de administración de recursos en escenarios de inundación

Tesis presentada para optar al título de
Licenciado/a en Ciencias de la Computación

Lorena Soledad Batlle - Norberto Andrés Furlan

Director: Daniel Ciolek

Codirector: Guido Chari

Buenos Aires, 2017

APLICACIÓN DE TÉCNICAS DE PLANNING PARA LA GENERACIÓN AUTOMÁTICA DE ESTRATEGIAS DE ADMINISTRACIÓN DE RECURSOS EN ESCENARIOS DE INUNDACIÓN

Las situaciones de emergencia, tales como inundaciones o incendios, derivan en complejas operaciones de defensa civil, en las cuales se debe evacuar a la población que se encuentra en el área afectada. Cada uno de estos operativos requieren una rápida y eficiente utilización de los recursos disponibles.

El objetivo de este trabajo es evaluar cuán adecuada es la técnica de planning para generar automáticamente estrategias que permitan asistir en la toma de decisiones durante ejercicios de entrenamiento y puesta a prueba de planes de contingencia para la evacuación ante inundaciones.

Las contribuciones del trabajo son, por un lado, utilizando el lenguaje lógico standard PDDL, el diseño de un modelo flexible que representa el entorno de las situaciones de evacuación. Por otro lado, la evaluación de la técnica respecto al modelo desarrollado, para lo cual se implementó un sistema informático con interfaz gráfica que facilita su accesibilidad.

Los resultados demuestran que la técnica de planning logra generar planes en el tiempo de cómputo requerido en la práctica. Esto se debe principalmente a la utilización de algoritmos de búsqueda basados en heurísticas independientes del dominio.

Palabras claves: planificación - inundaciones - evacuación - asistencia - dominio - técnicas de planning - plan de evacuación.

APPLICATION OF PLANNING TECHNIQUES FOR THE AUTOMATIC GENERATION OF RESOURCE MANAGEMENT STRATEGIES IN FLOOD SCENARIOS.

Emergency situations, particularly floods, derive in complex civil defense operations, where people in the affected area must be evacuated. Each of these operations require a fast and efficient use of the available resources.

The goal of this work is to evaluate how appropriate are planning techniques to generate automated plans to assist in decision making during training exercises and to evaluate contingency plans for flood evacuations.

The contributions of this work are, on one side, the design of a flexible model that represents the environment of evacuation situations using the standard logical language PDDL. On the other side, the evaluation of planning techniques for the developed model that also includes the development of a graphic interface in order to improve the accessibility to the system.

The testing results show that planning techniques can generate plans in the required computational time in practice. This is mainly due to the algorithms used which are based on domain independent heuristics.

Keywords: planning- floods - evacuation - assistance - domain - planning techniques - evacuation plan.

AGRADECIMIENTOS

Agradecimientos de Andrés

A mis papás Norberto y Mónica por darme todas las fuerzas y ayudarme en este gran camino. En especial a mi mamá, que aunque no este físicamente en este momento, uno de sus grandes deseos era verme presentar este trabajo y recibir el diploma. Gracias por su contención y por brindarme lo necesario para llegar hasta aquí.

A mis queridas hermanas por ayudarme a relajarme en momentos difíciles.

A Fede por darme las fuerzas necesarias para estudiar, cursar y terminar. Por su apoyo y compañia incondicional, y por relegar planes para que siga estudiando y pueda recibirme.

A mi mejor amiga Lore por estar siempre a mi lado, ser una gran guía y compañera. Agradezco por su contención y ayuda en momentos en dificiles y, en especial, por su apoyo y empuje diario para que terminemos este trabajo.

A mis amigos de la facu: Ale, Ivan, Pablo, Fran, y Flor que hicieron que este difícil camino sea más sencillo y divertido.

A mi madrina, por haberme guiado en la elección de la carrera y por transmitirme su pasión por el estudio y la tecnología.

A toda mi gran familia y mis amigos, por bancarme en todo momento y comprender el esfuerzo realizado.

Agradecimientos de Lorena

A mis viejos Noemi y Jorge por mostrarme el camino del crecimiento y darme las herramientas para seguirlo. Por apoyarme en cada una de mis etapas de estudio, sin ellos nada de esto sería posible.

A mi hermano por mostrarme (refregarme) que en los fin de semana había una vida afuera de exactas.

A mi novio Ale por bancarme todos los lamentos en el último y más pesado tramo de la carrera. Por darme fuerzas para finalmente terminar.

A mi amigazo Andrés por estar, desde el comienzo y hasta el final de mi carrera, siempre codo a codo estudiando, por tenerme tanta paciencia.

A mis amigos de la facu: Ivan, Pablo y Fran que son unos de los pocos que pueden entenderte lo que es estudiar en Exactas, por todos los momentos compartidos.

A mis amigos de la vida y familia por bancarme todo el tiempo desaparecida y todavía estar ahí.

Agradecimientos compartidos

A Daniel Ciolek por guiarnos y respondernos siempre con buena predisposición y rapidez y por resolvernos los problemas a lo largo de toda la tesis. Muy agradecidos.

A Diana Otero por el empuje desde el primer momento para que este proyecto salga adelante y por la confianza depositada en nosotros.

A todos los docentes y no docentes del Departamento de Computación de la Facultad de Ciencias Exactas y Naturales de la Universidad de Buenos Aires, por brindarnos una educación pública de excelencia, formarnos como profesionales, por la buena predisposición y paciencia.

¡Gracias a todos!

Índice general

1. INTRODUCCIÓN

En los últimos años se han registrado, a nivel mundial, diversas situaciones de emergencia que derivan en complejas operaciones de defensa civil. Las causas que derivan en situaciones de emergencia abarcan tanto incidencias de tipo natural como consecuencias del accionar humano.

En particular en las inundaciones se conjugan causas naturales y humanas. Entre los factores naturales se encuentran los geográficos y los meteorológicos. Mientras que, por causas de acción humana, se reconocen las inundaciones provocadas por intervenir los sistemas naturales.

Muchas situaciones de emergencia generan la necesidad de organizar operativos que permitan la evacuación de la población que se encuentra en el área afectada. Cada uno de estos operativos requieren una rápida y eficiente utilización de los recursos disponibles conforme a las características particulares de cada escenario. Los múltiples organismos responsables de la asistencia deben gestionar sus recursos para resolver, en el menor tiempo posible, las necesidades que se generan a partir de la emergencia en cuestión.

Se formuló como hipótesis que la técnica de planning es adecuada para generar automáticamente estrategias que permitan asistir en la toma de decisiones durante ejercicios de entrenamiento y puesta a prueba de planes de contingencia para la evacuación ante inundaciones. Se considera que la técnica es adecuada si, por un lado, el tiempo de cómputo es razonable a efectos prácticos. Por otro lado, genera estrategias que mitigan el riesgo de las víctimas, lo cual se supone, sucede al minimizar el tiempo total que lleva implementar la estrategia.

El objetivo de este trabajo es evaluar cuán adecuadas son las técnicas de planning conocidas hasta el momento, para generar automáticamente estrategias que logren ubicar, en el menor tiempo posible, en las áreas de evacuación, a todas las personas que se encuentren en áreas afectadas y no puedan evacuarse por sus propios medios. Es decir, se debe enviar los medios de transporte a las áreas afectadas donde se encuentran los damnificados y transportarlos a las áreas de evacuación.

Una alternativa a este problema sería diseñar un algoritmo ad-hoc, basado en metaheurísticas o programación lineal, que generen estrategias de evacuación considerando características particulares de un problema, como por ejemplo las inundaciones. Sin embargo, se debería rediseñar el algoritmo si se quiere extenderlo a otras situaciones de evacuación de personas como por ejemplo, incendios, alertas ante terremotos, maremotos o erupciones volcánicas.

En contrapartida, la técnica de planning (véase [1] [2] [3] para revisiones de la literatura sobre el tema) se basa en una descripción formal del entorno a partir de la cual se pueden generar soluciones que satisfacen un conjunto de objetivos dados. Dicha descripción es independiente del dominio, por lo tanto, es esperable que su extensión sea menos costosa que la de un algoritmo ad-hoc. Si bien utilizando planning es posible buscar soluciones óptimas, esto suele requerir un costo computacional prohibitivo, tanto en tiempo de procesamiento como de recursos físicos del sistema. Por ende, en planning usualmente

se emplean heurísticas que suelen generar resultados de alta calidad sin ofrecer garantías de optimalidad.

Existen trabajos previos que resuelven problemas asociados a evacuaciones usando algoritmos ad-hoc (véase [4] [5] [6] para revisiones de la literatura sobre el tema). Sin embargo, ninguno de esos procedimientos utiliza planning para generar las estrategias automáticamente.

Se utilizó el lenguaje lógico standard PDDL [7] para diseñar un modelo que represente el entorno de las situaciones de evacuación para el caso particular de las inundaciones, teniendo en cuenta su extensibilidad a otros entornos similares. Por otro lado, se experimentó con la técnica respecto al modelo desarrollado, obteniendo estrategias que satisfagan criterios de calidad. Para validar la hipótesis se implementó un sistema informático con interfaz gráfica que facilita la accesibilidad de usuarios sin expertise en planning.

El proyecto se llevó a cabo en colaboración con las Fuerzas Armadas Argentinas, quedando la herramienta desarrollada a su disposición y a disposición de otras agencias gubernamentales o no gubernamentales que lo soliciten.

En lo mejor de nuestro conocimiento, se puede afirmar que no existen otros sistemas basados en técnicas de planning que generen automáticamente estrategias de administración de recursos en escenarios de emergencia. Se pueden encontrar trabajos previos relacionados con sistemas informáticos de emergencias, y por otro lado, sistemas basados en técnicas de planning aplicados a otras áreas.

Actualmente, las Fuerzas Armadas Argentinas poseen un sistema desarrollado por el Departamento de Modelado y Manejo de Crisis del Ministerio de Defensa de la República Argentina (CITEDEF). Dicho sistema está diseñado para sostener otras redes y fomentar el intercambio de información cuando una situación de emergencia lo requiera. Cabe destacar que no utiliza técnicas de simulación basadas en inteligencia artificial, pues su desarrollo se basa en algoritmos informáticos específicos que facilitan la planificación logística, el uso eficiente de recursos y en análisis de planes de respuesta.

La *National Aeronautics and Space Administration (NASA)* tiene un grupo dedicado a la investigación e implementación de técnicas de planning. Dichas técnicas se aplican experimentalmente en diversas misiones de exploración espacial y administración de vehículos aéreos. Dentro de los resultados que están a nuestro alcance, no se encontraron proyectos aplicados a situaciones de emergencias causadas por factores naturales.

En los siguientes capítulos se detalla el trabajo realizado. En el capítulo 2. *Preliminares* se exponen todos los conceptos necesarios para poder comprender la técnica de planning y se describen las características de los recursos utilizados para el desarrollo de la evaluación. En el capítulo 3. *Desarrollo* se explican los modelos del entorno que se evaluaron y el desarrollo de una modificación a los algoritmos de planning que permite mejorar su desempeño en entornos con ciertas características. En el capítulo 4. *Evaluación*, se llevaron a cabo comparaciones entre los distintos modelos y el desempeño del algoritmo de planning al aplicar la modificación desarrollada. Finalmente, en el capítulo 5. *Conclusión* se expone, en relación a la hipótesis propuesta, cuán adecuada es la técnica de planning para asistir en el tipo de situaciones planteadas.

2. PRELIMINARES

2.1. Planning

Planning [1] es una rama de la inteligencia artificial que busca construir un plan a partir de un modelo formal del dominio y una especificación de los objetivos que deben ser alcanzados por un agente autónomo.

Un algoritmo basado en técnicas de planning es denominado **planner**. Tal como se observa en la Figura 2.1, un planner tiene como entrada un **dominio** que describe el entorno, un **estado inicial** que describe cómo se encuentra el entorno al momento de iniciar la estrategia y un conjunto de **objetivos** que deben ser cumplidos por dicha estrategia. Como salida del algoritmo se obtiene una estrategia o un **plan**.

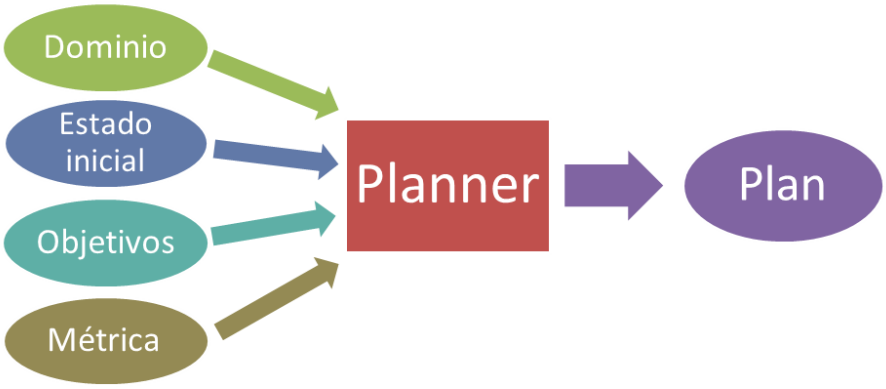

Fig. 2.1: Esquema de entrada y salida del planner.

Un **plan** es una secuencia de acciones pertenecientes al dominio cuya ejecución garantiza poder alcanzar, desde el estado inicial, un estado final donde se satisfacen los objetivos. La secuencia de ejecución está completamente determinada por las acciones contenidas en el plan, es decir, no es necesario obtener información adicional durante la ejecución del mismo.

El **planning clásico** se basa en las siguientes presunciones:

- **Dominio finito:** el dominio tiene una cantidad finita de variables binarias que describen los estados del entorno.
- **Acciones instantáneas:** las acciones son transiciones instantáneas entre los diferentes estados.
- **Determinismo:** el estado inicial está totalmente especificado y cada acción aplicada a un estado genera un único nuevo estado con probabilidad 1.
- **Objetivos:** existe al menos un estado que debe ser alcanzado.

- **Off-line planning:** los cambios del entorno que se produzcan durante la elaboración de un plan no serán contemplados en el mismo.
- **Observación total:** el estado del planner es totalmente conocido en todo momento.

Sin embargo, las presunciones anteriores pueden ser demasiado restrictivas en escenarios reales. Por un lado, no es posible la reducción a dominios determinísticos si existe incertidumbre en el estado inicial, eventos externos o acciones que al ejecutarse tengan un efecto impredecible o no determinado. Al utilizar modelos no determinísticos se logra un modelado más consistente con la realidad, dado que permiten considerar más de un estado inicial y una acción puede transicionar a diferentes estados. Cabe destacar que estos modelos son más complejos y su costo en la práctica, en general, resulta prohibitivo.

Por otro lado, además de los recursos, el modelado de algunos dominios requiere la representación explícita del tiempo para contemplar acciones con duración, acciones concurrentes y/o restricciones temporales. El tiempo continuo no puede ser modelado directamente con máquinas de estados finitos básicas, por esta razón, existen extensiones de planning clásico con la capacidad de administrarlo por medio del uso de autómatas temporizados.

2.2. Representación de problemas - Planning Domain Definition Language (PDDL)

Los algoritmos de planning requieren de un lenguaje lógico para modelar los escenarios de planificación, es decir, modelar los estados, las acciones y los objetivos. La clave es contar con un lenguaje que sea lo suficientemente expresivo como para poder describir una gran variedad de problemas.

Planning Domain Definition Language (PDDL) [7] es el estándar de facto utilizado para representar la estructura lógica de un problema de planning. PDDL está basado en otros lenguajes bien conocidos de formalización de problemas de planning, como ser STRIPS [8] y Action Description Language (ADL) [9]. Sin embargo, a diferencia de los anteriores, PDDL incorpora la posibilidad de expresar recursos como el tiempo. La sintaxis de este lenguaje continúa la tradición de lenguajes LISP.

PDDL está ideado para expresar las características del entorno utilizando los siguientes componentes:

- **Objetos:** las entidades del entorno relevantes para el problema.
- **Predicados:** las propiedades binarias de cada objeto.
- **Funciones:** las propiedades cuantificables de cada objeto, sobre las cuales se pueden efectuar operaciones aritméticas.
- **Estado inicial:** es una conjunción de predicados verdaderos instanciados que representan las condiciones lógicas del entorno que se cumplen al inicio del problema.
- **Acciones:** esquemas que modelan las condiciones y efectos necesarios para transicionar de un estado a otro.

El esquema de una acción contiene:

- Un nombre
- Un conjunto de parámetros con especificación de tipos.
- Una duración: es el tiempo que demora en ejecutarse la acción. Puede ser el resultado de una operación.
- Una precondición: es un conjunto de predicados verdaderos que se deben satisfacer para que la acción pueda ejecutarse.
- Un efecto: es un conjunto de predicados que se satisfacen luego de ser ejecutada la acción.

- **Objetivos:** un objetivo es una conjunción de predicados verdaderos instanciados. Un estado S satisface un objetivo G si S contiene todos los predicados de G.
- **Métrica:** una métrica es una función que determina la optimización de un recurso del plan generado.

A partir del standard 2.1 de PDDL [7], con el objetivo de facilitar la representación de las condiciones temporales del dominio, se incorporaron al lenguaje condiciones explícitas de tiempo, aplicables a las precondiciones y a los efectos de las acciones.

Las condiciones aplicadas sobre cada predicado de la precondición o el efecto, permiten establecer cuándo el predicado se debe cumplir, es decir, si debe ser válido al comienzo, al final o invariante durante toda la acción.

Cabe destacar que la elección de un lenguaje estándar para especificar el entorno del dominio nos permitió contribuir con un modelo independiente al algoritmo de planning utilizado, es decir, permitiría ser evaluado con distintos planners.

2.2.1. Ejemplo

El siguiente ejemplo modela el viaje de un avión Boeing 777 desde el Aeropuerto Internacional Ministro Pistarini de Buenos Aires (EZE) al Aeropuerto Internacional John F. Kennedy de New York (JFK). Se determina la duración del viaje entre aeropuertos en función de la distancia entre ellos y la velocidad crucero del avión. El plan generado tiene como objetivo que el avión Boeing 777 se encuentre en el aeropuerto JFK y a su vez, debe intentar minimizar la métrica tiempo de viaje.

Dominio

Tipos de los objetos: se identifican dos objetos, avión y aeropuerto.

```
(:types
          Plane Airport - Object)
```

Predicados: describen las propiedades binarias de cada objeto. En este caso, si un avión se encuentra o no en un aeropuerto.

```
(:predicates
          (at ?aPlane - Plane ?anAirport - Airport))
```

Funciones: describen las propiedades cuantificables de los objetos. En este caso, de los aeropuertos, determinando la distancia entre aeropuertos; y de los aviones, determinando su velocidad.

```
(:functions
          (distance ?from - Airport ?to - Airport)
          (cruiseSpeed ?aPlane - Plane))
```

Acciones: esquemas que modelan las condiciones y efectos necesarios para transicionar de un estado a otro. En este caso, se modelan las condiciones para que un avión deje de estar en un aeropuerto y pase a estar en otro, es decir, el vuelo de un avión desde un aeropuerto a otro.

Nombre

```
(:durative -
          action Fly
```

Parámetros: avión, aeropuerto origen y aeropuerto destino.

```
:parameters
          (?aPlane - Plane ?from - Airport ?to - Airport)
```

Duración: duración del vuelo como la distancia entre aeropuertos sobre la velocidad crucero del avión.

```
:duration
          (= ?duration (/ (distance ?from ?to)
                          (cruiseSpeed ?aPlane)))
```

Condición: avión ubicado en el aeropuerto de origen.

```
:condition
          (at start (at ?aPlane ?from))
```

Efecto: al inicio de la acción el avión deja de estar ubicado en el aeropuerto de origen y al final de la misma el avión está ubicado en el aeropuerto destino.

```
:effect
          (and (at start (not (at ?aPlane ?from)))
               (at end (at ?aPlane ?to)))))
```

Estado inicial

Objetos: se inicializan los objetos propios del ejemplo. En este caso, un Boeing 777 de tipo avión y tres aeropuertos EZE, JFK y MIA.

```
(:objects
          Boeing777 - Plane
          EZE JFK MIA - Airport)
```

Inicialización: se inicializa el predicado de la ubicación del avión Boeing 777 en EZE.

```
(:init
          (at Boeing777 EZE)
```

Se inicializan las funciones de velocidad del avión Boeing 777.

```
          (= (cruiseSpeed Boeing777) 900.0)
```

Se inicializan las distancias entre los aeropuertos.

```
          (= (distance EZE JFK) 8546.0)
          (= (distance JFK EZE) 8546.0)
          (= (distance MIA JFK) 1760.0)
          (= (distance JFK MIA) 1760.0)
          (= (distance EZE MIA) 7128.0)
          (= (distance MIA EZE) 7128.0)
          (= (distance EZE EZE) 0)
          (= (distance JFK JFK) 0)
          (= (distance MIA MIA) 0))
```

Objetivo: el objetivo de este plan sería que el avión esté ubicado en el aeropuerto JFK.

```
(:goal
          (at Boeing777 JFK))
```

Métrica a minimizar: se quiere que el plan se ejecute en el menor tiempo posible.

```
(:metric
          minimize (total-time)))
```

Plan

La solución al problema será un plan que contenga la/s acción/es que permiten realizar el vuelo del avión desde EZE a JFK intentando optimizar el tiempo de viaje.

Los posibles planes solución, tal como se observa en la Figura 2.2, son todos aquellos que comienzan en el estado inicial y finalizan en el estado objetivo, por ejemplo:

- EZE - MIA - JFK
- EZE - MIA - EZE - JFK
- EZE - MIA - EZE - MIA - JFK
- EZE - JFK
- ...

Sin embargo, el plan que se espera obtener es aquel que permita volar desde EZE a JFK en el mínimo tiempo posible tal como se especificó en la métrica a minimizar en el estado inicial, es decir, el plan EZE - JFK.

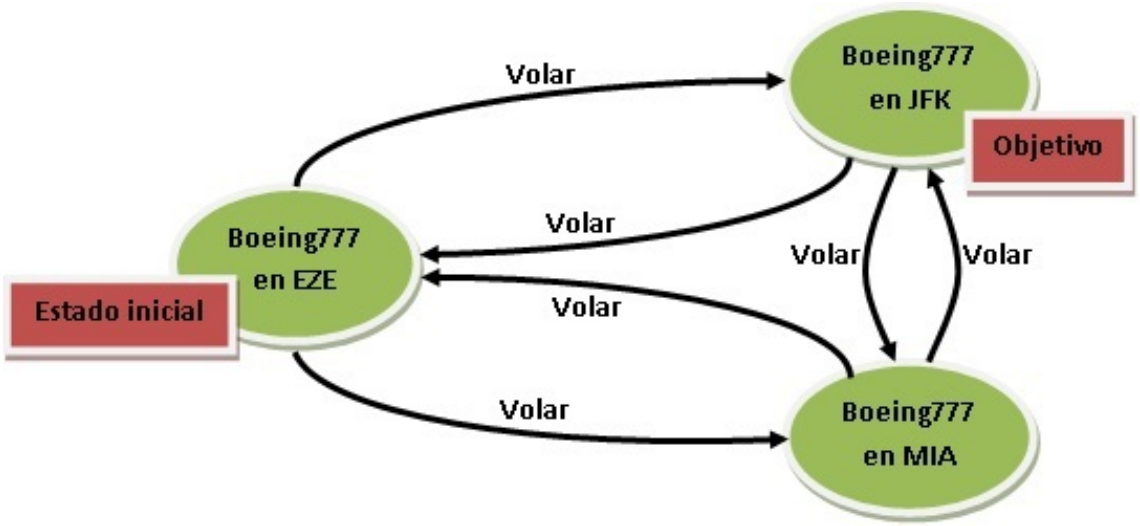

Fig. 2.2: Esquema de los estados y las transacciones entre estados del ejemplo.

Plan: se especifica el tiempo de inicio de la acción, la acción a desarrollar y el tiempo de ejecución de la acción.

```
0.0 (Fly Boeing777 EZE JFK) [9.495555]
```

Hasta aquí se ha descrito la sintaxis para representar los problemas de planning. Respecto a la semántica de este lenguaje, es necesario definir la aplicabilidad de una acción, los efectos de su ejecución y la solución a un problema.

2.2.2. Aplicación de una acción

Una acción es aplicable en un estado si este satisface la precondición de la acción. En un esquema de acción establecer su aplicabilidad requiere realizar una sustitución a la precondición para instanciarla.

El resultado de ejecutar una acción aplicable en un estado S, es un nuevo estado S' igual a S en donde los valores de los predicados se modifican por los efectos de la acción. Los predicados instanciados verdaderos se agregan al estado, mientras que los falsos se eliminan.

Notar que si un efecto positivo ya está en S, el mismo no es agregado dos veces y si un efecto negativo no está en S se ignora. El lenguaje asume que los literales no mencionados en los efectos permanecen invariables, esto es conocido en planning como el problema del encuadre (*Frame-Problem* [1]).

2.2.3. Solución

Se puede definir, de forma simple, la solución a un problema de planning como una secuencia de aplicaciones que permiten alcanzar, desde el estado inicial, un estado que satisface los objetivos.

2.3. Algoritmos de planning - Planner

Los algoritmos de planning (planners) se basan en la especificación de las precondiciones y los efectos de las acciones de un problema para realizar la búsqueda del estado objetivo. Uno de los enfoques más sencillos es **Forward Search** que comienza su búsqueda desde el estado inicial hasta alcanzar el estado que contiene todos los objetivos. Pero también, existen algoritmos en dirección inversa, como **Backward Search**, que parten su búsqueda desde el estado objetivo hasta el estado inicial.

2.3.1. Forward Search

Los planners que utilizan **Forward Search** comienzan su búsqueda en el estado inicial del problema. Consideran distintas secuencias de acciones hasta encontrar una que alcance un estado que contenga todos los objetivos.

Los problemas de planificación con algoritmos de búsqueda Forward Search contienen:

- Un estado inicial de la búsqueda, que es el estado inicial del problema.
- Las acciones aplicables en un estado.
- La evaluación del objetivo que verifica si el estado satisface el objetivo del problema.
- El costo de ejecución de cada acción.

En la práctica las búsquedas Forward Search por sí solas son ineficientes dado que, por un lado, no abordan el problema de la acción irrelevante, es decir, en cada estado se consideran todas las acciones aplicables. Por otro lado, una de las mayores dificultades que enfrentan las técnicas de planning es el gran tamaño de los espacios de búsqueda que se generan en situaciones reales, a pesar de las simplificaciones en las presunciones del dominio. Este factor de ramificación genera un impacto en la eficiencia de los algoritmos. En problemas donde el espacio de estados es infinito, realizar la búsqueda es computacionalmente inaccesible. Por estas razones, se requieren heurísticas que hagan eficientes dichas búsquedas.

2.3.2. Heurística para espacios de búsqueda

Tanto Forward Search como Backward Search requieren, para mejorar su eficiencia, una función heurística que estime la distancia desde un estado A al estado que contiene los objetivos. La idea es considerar los efectos de las acciones y los objetivos que deben ser logrados para estimar cuántas acciones son necesarias para lograr todos los objetivos. Encontrar el valor exacto es un problema exponencial, sin embargo, en general, es posible encontrar estimaciones razonables con un costo accesible.

Un posible enfoque es generar un **Relaxed Planning Graph (RPG)** a partir de la especificación del problema en cuestión, modificando las precondiciones y los efectos de las acciones. Esta heurística permite reducir el espacio de búsqueda del problema original y aproximar su costo, dado que el problema relajado es más fácil de resolver.

La idea más simple para generar un plan relajado consiste en eliminar las precondiciones de las acciones, de este modo, cada acción siempre será aplicable y cualquier predicado se puede lograr en un paso (siempre que haya una acción aplicable, si no, el objetivo es inalcanzable). No obstante, esta heurística presenta gran variabilidad, por lo que debe combinarse con otras que mejoran su aproximación.

Otra idea para generar problemas relajados es eliminar los efectos negativos de las acciones, sin eliminar las precondiciones. Es decir, si una acción tiene el efecto $A \wedge \neg B$ en el problema original, en el problema relajado tendrá sólo el efecto A. Esto significa que no habrá interacciones negativas entre acciones, porque ninguna acción puede eliminar los predicados alcanzados. Todas las soluciones del problema original son solución del problema relajado, pero no todas las soluciones del problema relajado son solución del problema original. De este modo, se puede estimar la distancia al objetivo en el problema original, utilizando la cantidad de acciones necesarias para que se satisfaga el objetivo en el problema relajado.

Al momento, los planners más utilizados son los que combinan algoritmos Forward Search con la última heurística descrita. Estos resultados, probablemente cambien a medida que se exploran nuevas heurísticas y nuevas técnicas de búsqueda. Dado que la planificación es exponencialmente difícil, ningún algoritmo será eficiente para todos los problemas, pero muchos problemas prácticos pueden ser resueltos por planners con dichas características.

2.4. Planner seleccionado

2.4.1. Motivación

Los problemas de planning que interesan resolver en este trabajo involucran acciones temporales que se ejecutan de forma concurrente, así como también, acciones que consumen recursos y planes que deben alcanzar los objetivos en un tiempo determinado. Dado esos requerimientos, el planning clásico con acciones instantáneas es insuficiente. Por esa razón, existen esfuerzos por construir planners que administren métricas temporales.

La adaptación del planning clásico a las métricas temporales plantea los siguientes desafíos. Por un lado, los planners con métrica temporal suelen tener significativamente mayores espacios de búsqueda que los planners clásicos, afectando su escalabilidad, dado que se debe considerar una mayor cantidad de factores. Por otro lado, el propósito de la planificación no puede ser limitado solamente a la satisfacción del objetivo, sino también se debe considerar la optimización de los tiempos asociados, como la latencia, el tiempo promedio de finalización, el consumo de recursos, etcétera.

2.4.2. SAPA

SAPA [10] es la implementación de un planner temporal, que a diferencia de uno clásico, permite administrar recursos, acciones con duración, restricciones temporales y objetivos finales.

Su principal innovación es la implementación de un conjunto de heurísticas basadas en la distancia, logrando un alto desempeño en problemas de este tipo. A pesar del desarrollo de otras herramientas similares, el desempeño de SAPA se encuentra vigente desde la AIPS 2002 Planning Competition [11], donde fue el planner que consiguió planes de mayor calidad en dos de los dominios más complejos considerando tiempo, acciones con duración y recursos.

Cabe destacar que, si bien existen otros planners escalables, estos administran por separado requerimientos temporales, recursos o un subconjunto de ellos, mientras que SAPA no establece dicha separación. Por otro lado, es un planner **independiente del dominio**, es decir, no requiere información de control específica del problema a resolver.

Además, a diferencia de otros planners con similares características, como TLPlan [11], SAPA es más accesible dado que, no solo es de código abierto sino que también está implementado en lenguaje JAVA, por lo tanto, es multiplataforma.

Implementación SAPA

SAPA utiliza las estrategias que se detallan a continuación, con el objetivo de hacer frente a los desafíos que se presentan al extender las técnicas de planning clásico (véase 2.4.1. *Motivación*). Implementa una heurística **Forward Chaining Search** en conjunto con una heurística que estima la distancia al objetivo con el fin de guiar la búsqueda.

Ambas heurísticas se aplican sobre una estructura de grafo llamada **Relaxed Temporal Planning Graph (RTPG)**, basada en la relajación del problema (RPG) (véase 2.3.2. *Heurística para espacios de búsqueda*), que constituye una abstracción reducida del espacio de búsqueda.

La construcción del RTPG consiste en un grafo de estados alcanzables, basado en el problema original pero sin considerar las precondiciones negativas. A diferencia del grafo original, RTPG es una estructura simplificada que **se puede computar en tiempo polinomial**.

A partir del RTPG se puede, por un lado, reducir el espacio de búsqueda al descartar todos los estados que alcanzan el objetivo en el tiempo límite y, por otro lado, guiar la búsqueda, utilizando como estimación la distancia al nodo que contiene el objetivo.

Gracias a esta técnica, en general se obtienen soluciones en un tiempo razonable para problemas de planning complejos y se logra una buena escalabilidad conservando la calidad de los planes resultantes.

La heurística **Forward Chaining Search** comienza la búsqueda del plan en el estado inicial del problema de planning. Las acciones aplicables a un estado son aquellas que cumplen con las precondiciones del estado actual. SAPA utiliza un conjunto de heurísticas, que selecciona dinámicamente, para determinar la acción a ejecutar entre todas las acciones aplicables.

Luego de la ejecución de una acción aplicable seleccionada, el estado resultante es el estado original agregando los efectos positivos y eliminando los negativos. Este estado resultante será el nuevo estado actual.

Al ejecutarse una acción, se valida si el estado actual satisface el objetivo del problema de planning, es decir, si el estado contiene al objetivo. Mientras esto no suceda, el algoritmo continuará con la selección de otra acción aplicable hasta que encuentre una secuencia que alcance un estado objetivo o no haya más acciones aplicables. En el segundo caso, procederá a realizar *backtracking* sobre la última acción ejecutada para seleccionar otra acción aplicable. El algoritmo finaliza sin encontrar solución cuando no existen, a partir del estado inicial, acciones aplicables que aún no han sido seleccionadas.

Algoritmo de Forward Search implementado por SAPA

```
estado = estadoInicial
plan = ∅
while(estado no satisface objetivo)
     accionesAplicables =
      acción | acción que satisface precondiciones del estado
     if(accionesAplicables == ∅)
          estado = backtracking(plan)
     else
          acción = eleccionHeuristica(accionesAplicables)
          estado = ejecutar(estado, accion)
          plan = plan + accion
return plan
```

3. DESARROLLO

En este capítulo se detallan las características relevadas sobre las inundaciones. Luego, se detalla cómo, utilizando el lenguaje standard PDDL y a partir de dicho relevamiento, se modeló el problema. Por último, se explica cómo se desarrolló una modificación al algoritmo de planning seleccionado, con el objetivo de mejorar su desempeño para el caso particular de las situaciones de emergencia.

3.1. Relevamiento del entorno

Ante una situación de emergencia las Fuerzas Armadas[1] pueden ser convocadas a formar parte, junto con otras organizaciones gubernamentales o no gubernamentales, del comité de emergencia creado para coordinar la asistencia. En caso de tener que asistir en la evacuación, las Fuerzas Armadas elaboran manualmente un plan basado en el conocimiento y la experiencia de la persona responsable de tal tarea. Esta metodología es sensible al error humano, en consecuencia, no es posible asegurar la calidad de los planes desarrollados.

Para la elaboración de un plan de evacuación se pone a disposición personal militar organizado jerárquicamente, así como también, una flota de camiones, botes y vehículos anfibios. En ciertas situaciones, incluso, se destinan espacios militares para ser utilizados como centros de evacuación.

Se relevó lo siguiente:

- El territorio nacional se encuentra dividido en distintas jurisdicciones para una mejor administración.
- Cada jurisdicción tiene asignada una cierta cantidad de personal y vehículos, dependiendo de su densidad poblacional.
- La colaboración entre jurisdicciones requiere de autorizaciones que pueden demorar la ejecución del plan de evacuación, por lo tanto, es recomendable priorizar la utilización de los recursos propios.
- Hay disponible una base de datos de los vehículos existentes que se actualiza periódicamente tras un relevamiento del estado de los mismos. De cada vehículo se conoce su ubicación, capacidad de carga, capacidad de combustible, consumo, velocidad máxima, transitabilidad[2], jurisdicción y personal responsable.
- Los vehículos asignados para transportar otros vehículos son utilizados únicamente para dicho propósito. Como por ejemplo, varios botes pueden ser transportados en un mismo camión de carga, no utilizable para transportar víctimas.

[1] Se relevó la información necesaria, en colaboración con las Fuerzas Armadas, para comprender qué sucede ante una situación de emergencia, en particular, en una evacuación ante inundaciones

[2] Transitabilidad: lugar por donde puede transitar un vehículo.

- Frente a una evacuación:
  - Personal con conocimiento en el área afectada realiza una evaluación de los terrenos para estimar la cantidad de personas a asistir.
  - Se verifica la disponibilidad de los vehículos al momento de armar una estrategia.
  - Personal capacitado en inundaciones realiza una evaluación del posible desarrollo de la emergencia.
  - Se determinan los centros de evacuación disponibles. Por un lado, se recibe información externa y, por otro lado, se evalúa la disposición de espacios militares destinados a utilizarse como centros de evacuación. Luego, se establece una prioridad de uso entre todos los centros de evacuación disponibles, la cual está relacionada con la actividad que estos realizan.

3.2. Modelado

A partir de la información relevada se realizó un modelo del dominio, es decir, del entorno de las inundaciones.

3.2.1. Presunciones del dominio

Con el propósito de garantizar la existencia de un plan que cumpla los objetivos y de simplificar el modelo para que resulte computacionalmente accesible, se presume lo siguiente sobre el dominio:

P1 *Los vehículos pueden llegar a un área por la que pueden transitar, sin importar el área del que partieron.* No se modeló la forma en que los vehículos llegan al área donde pueden transitar dado que aumenta la complejidad del modelo. Por ejemplo, no se ha modelado la carga, el transporte y la descarga de los botes en camiones.

P2 *Los vehículos solo se trasladan a las áreas de evacuación si se encuentran completos o si no hay más víctimas a evacuar en el área.* De este modo, se reduce la generación de planes en los cuales haya viajes de vehículos incompletos cuando podrían no estarlo, logrando así, favorecer la optimización de los recursos.

P3 *Los vehículos no asisten víctimas mientras transitan entre áreas o en áreas que no pueden transitar.* Por ejemplo, un camión no puede asistir víctimas que se encuentran en un área inundada, dado que su transitabilidad no le permite acceder. Por simplicidad no se modeló la asistencia en áreas intermedias, pero podría agregarse posteriormente.

P4 *El combustible de los vehículos será suficiente para realizar cualquier recorrido.* Con el propósito de no aumentar la cantidad de variables del entorno, se consideró que el conductor del vehículo es responsable de la administración del combustible de forma satisfactoria.

P5 *No existen áreas inaccesibles que contengan víctimas.* El planner no genera planes que cumplen parcialmente los objetivos, por lo tanto, sin dicha presunción no se podría garantizar la existencia de un plan que cumpla con todos los objetivos. Es decir, no sería posible generar un plan que cumple con el objetivo de asistir a todas las víctimas si, al menos una de ellas, se encuentra en un área inaccesible.

P6 *La cantidad de víctimas no debe superar la capacidad total de los centros de evacuación.* Nuevamente, no se podría garantizar la existencia de un plan que cumpla con todos los objetivos. Es decir, no sería posible generar un plan que cumple con el objetivo de asistir a todas las víctimas si, al menos una de ellas, no puede ser descargada en área de evacuación con capacidad disponible.

P7 *El tamaño de un área debe ser determinado de modo que sea despreciable el tiempo que demanda recorrerla.*

3.2.2. Especificación del dominio

Se modeló el dominio utilizando el lenguaje estándar PDDL, especificando los siguientes objetos, predicados, funciones y acciones.

Objetos

Se determinó un tipo de objeto abstracto denominado *Locatable*, que modela aquellos objetos que pueden ser ubicados en un espacio. Dicho espacio fue modelado por otro tipo de objeto denominado *Area*.

A partir de esos dos tipos de objetos, se establece la siguiente jerarquía haciendo que el modelo sea lo suficientemente flexible para atender a otras situaciones de emergencia o simulaciones que requieran la evacuación de víctimas ubicadas en un área.

Locatable (Véase Figura 3.1):

- *Victim*: ser vivo afectado por una emergencia.
- *Vehicle*: vehículo utilizado para asistir en la emergencia. Existen distintos tipos de vehículos diferenciados por el área que pueden transitar:
  - *Landcraft*: vehículo terrestre.
  - *Aircraft*: vehículo aéreo.
  - *Watercraft*: vehículo acuático.
  - *Ambphibious*: vehículo anfibio.

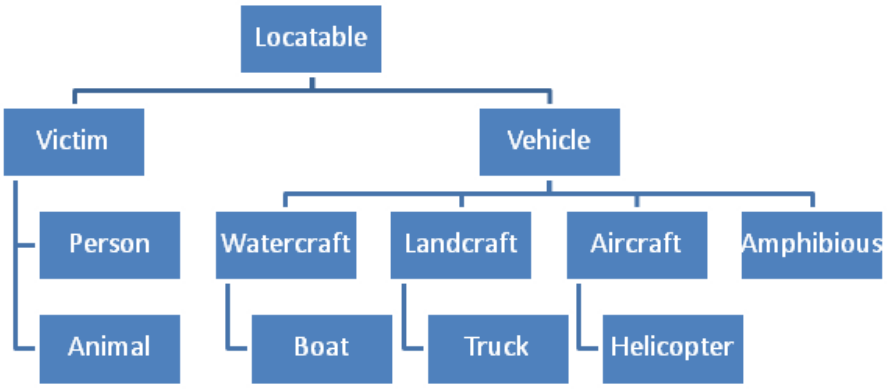

Fig. 3.1: Esquema de herencias del objeto *Locatable*.

Area (Véase Figura 3.2):

- *Emergency area*: área que debe ser evacuada.
- *Evacuation area*: área segura para evacuación.
- *Parking area*: área de partida de los vehículos.

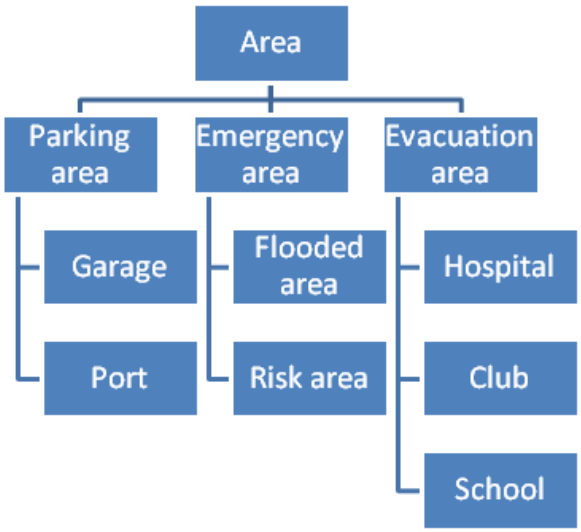

Fig. 3.2: Esquema de herencias del objeto *Area*.

Cada área se diferencia a su vez por su transitabilidad, por ejemplo, en áreas inundadas pueden transitar vehículos de tipo acuático, pero no de tipo terrestre.

En la versión implementada para este trabajo, dentro del tipo *Area* (véase Figura 3.3), se definieron dos tipos de *Emergency Areas*: *Flooded Area*, que modela un área inundada

no transitable por vehículos terrestres, y *Risk Area*, que modela un área con riesgo de inundación no transitable por vehículos acuáticos.

Evacuation Area modela un área de evacuación que se asume intransitable por vehículos acuáticos, pues se entiende que dicha área no debe estar inundada, ni en riesgo de inundarse. Por último, el tipo *Parking Area* modela el depósito de los vehículos que se presume como lugar de origen y siempre es transitable y accesible por cualquiera de ellos.

La accesibilidad entre las *Flooded Areas* y *Evacuation Areas* puede verse imposibilitada si no se cuenta con áreas intermedias y vehículos que puedan transitarlas. Con el propósito de garantizar la presunción *P6 (no existen áreas inaccesibles)*, se considera que existe al menos una *Risk Area*, un *Truck* y un *Boat* que permite el acceso entre *Flooded Areas* y *Evacuation Areas*. En la sección 3.3. *Modificación de la especificación del dominio* se evalúa un modelo en el cual se vincula explícitamente cada *Flooded Area* a una o más *Risk Area*.

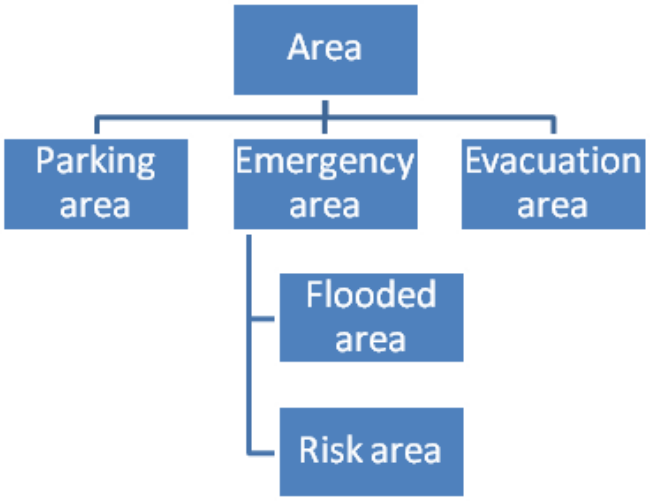

Fig. 3.3: Esquema de herencias implementadas del objeto *Area*.

Dentro del tipo *Locatable* (véase Figura 3.4), se modeló el tipo *Victim* para hacer referencia a todo tipo de víctima que deba ser rescatada sin diferenciarlas entre sí. Una alternativa consiste en modelar a las víctimas como variables numéricas dado que, en principio, sólo es relevante conocer la cantidad de víctimas evacuadas o no. Sin embargo, la mayoría de los planners temporales disponibles no manejan adecuadamente las restricciones numéricas dado que se utilizan únicamente en pos de perseguir un criterio a minimizar y no para describir las restricciones lógicas del problema. En el capítulo 6. *Anexo A: Dominio con variables numéricas* se explican los resultados no satisfactorios que se obtuvieron al utilizar un modelo PDDL basado en una representación numérica.

Finalmente, *Vehicle* abarca los tipos *Boat* y *Truck* que modelan un vehículo acuático y uno terrestre, respectivamente.

Cabe destacar que gracias a la forma jerárquica en que se establecieron los tipos de objetos es simple extender el modelo, es decir, incluir objetos con otras características. En el caso del tipo *Vehicle* se pueden agregar vehículos con diferentes características de transitabilidad, como por ejemplo, aéreos o anfibios.

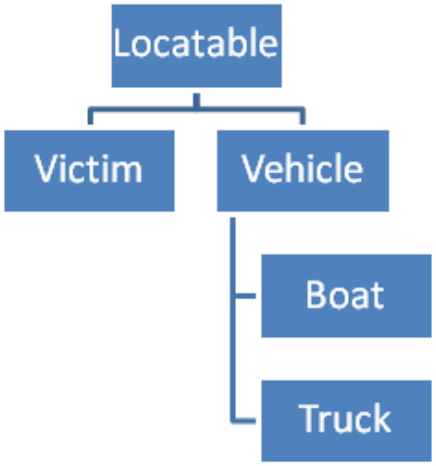

Fig. 3.4: Esquema de herencias implementadas del objeto *Locatable*.

Del mismo modo, se pueden agregar otros tipos de áreas dentro de los tres grandes grupos de áreas definidas. En el caso de los objetos de tipo *Emergency Areas* se podrían modelar áreas con diferentes riesgos. Mientras que dentro de *Evacuation Area* se podrían definir áreas según su funcionalidad o sus instalaciones, como ser escuelas, clubes, hospitales, entre otros. Finalmente, los objetos de tipo *Parking Area* se podrían diferenciar según su transitabilidad, como ser, embarcaderos, depósitos vehiculares, estaciones de carga de combustible, etcétera.

Predicado

Se modela la ubicación para un objeto *Locatable*, la disponibilidad de un vehículo y el estado de la víctima: evacuada o no.

- Indica si un objeto ubicable *unLocatable* está en el área *unArea*.

```
(at ?unLocatabe - Locatable ?unArea - Area)
```

- Indica si una víctima *unaVictima* está en el vehículo *unVehiculo*.

```
(in ?unaVictima - Victim ?unVehiculo - Vehicle)
```

- Indica si un vehículo *unVehiculo* se encuentra disponible.

```
(available ?unVehiculo - Vehicle)
```

- Indica si una víctima *unaVictima* se encuentra asistida.

```
(aided ?unaVictima - Victim)
```

Se podrían incorporar otros predicados que especifiquen distintas prioridades sobre los objetos del modelo, como por ejemplo, modelar una asistencia diferenciada de *Victim*, impidiendo que una víctima con menor prioridad ascienda a un vehículo si en el área existen otras con mayor prioridad.

Funciones

Se modela la distancia entre áreas; la velocidad, capacidad en cada estado y capacidad máxima de víctimas de los vehículos; la cantidad de víctimas en cada área y la capacidad máxima de víctimas de las áreas de evacuación.

- Indica la distancia desde el área *from* al área *to*.

```
(distance ?from - Area ?to - Area)
```

- Indica la velocidad del vehículo *unVehiculo*.

```
(speed ?unVehiculo - Vehicle)
```

- Indica la cantidad de víctimas que hay dentro del vehículo *unVehiculo*.

```
(victimsIn ?unVehiculo - Vehicle)
```

- Indica la capacidad del vehículo *unVehiculo*.

```
(maxCapacity ?unVehiculo - Vehicle)
```

- Indica la cantidad de víctimas en el área de emergencia *unaEmArea*.

```
(victims ?unaEmArea - EmergencyAreas)
```

- Indica la capacidad del área de evacuación *unaEvArea*.

```
(maxOccupancy ?unaEvArea - EvacuationArea)
```

Se podrían agregar funciones que modelen otras características de los objetos, por ejemplo, considerar el consumo de combustible de los vehículos.

Acciones

Las acciones generales necesarias para llevar a cabo el plan son:

- *Go*: modela el viaje de un vehículo desde un área hacia otra área transitable por el mismo, verificando que el vehículo esté disponible y con capacidad.

```
(:durative-action    Go
 :parameters (
               ?aVehicle - Vehicle
               ?origin - Area
               ?destination - Area)
 :duration
               (= ?duration
                  (* (/ (distance ?origin ?destination)
                        (speed ?aVehicle)) 60))
 :condition
               (and  (at start (at ?aVehicle ?origin))
                     (at start (> (maxCapacity ?aVehicle) (0)))
                     (at start (available ?aVehicle)))
 :effect
               (and  (at start (not (at ?aVehicle ?origin)))
                     (at end   (at ?aVehicle ?destination))))
```

- *Come Back*: modela el viaje de un vehículo desde un área transitable por el mismo hacia otra área. Se requiere como precondición que el vehículo vuelva completo o con todas las víctimas del área.

```
(:durative-action    ComeBack
 :parameters (
               ?aVehicle - Vehicle
               ?origin - Area
               ?destination - Area)
 :duration
               (= ?duration
                  (* (/ (distance ?origin ?destination)
                        (speed ?aVehicle)) 60))
 :condition
               (and  (at start (at ?aVehicle ?origin))
 :effect
               (and  (at start (not (at ?aVehicle ?origin)))
                     (at end   (at ?aVehicle ?destination))))
```

- *Load*: modela el ascenso de una víctima a un vehículo dentro de un área de emergencia, verificando la capacidad del vehículo. Se actualiza la función de cantidad de víctimas en el vehículo y la función de cantidad de víctimas en el área.

```
(:durative-action Load
 :parameters (
               ?aVehicle - Vehicule
               ?anArea - Area
               ?aVictim - Victim)
 :duration
               (= ?duration 1)
 :condition
               (and  (over all (at ?aVehicle ?anArea))
                     (at start (at ?aVictim ?anArea))
                     (at start (< (victimsIn ?aVehicle)
                                  (maxCapacity ?aVehicle))))
 :effect
               (and  (at end (not (at ?aVictim ?anArea)))
                     (at end (in ?aVictim ?aVehicle))
                     (at end (increase (victimsIn ?aVehicle) 1))
                     (at end (decrease (victims ?anArea) (1)))))
```

- *Unload*: modela el descenso de una víctima desde un vehículo a un área. Si el descenso es en una *Evacuation Area*, se verifica la capacidad del área de evacuación y se identifica como rescatada a la víctima. Se actualiza la función cantidad de víctimas del vehículo y del área.

```
(:durative-action Unload
 :parameters (
               ?aVehicle - Vehicle
               ?anArea - Area
               ?aVictim - Victim)
 :duration
               (= ?duration 1)
 :condition
               (and  (over all (at ?aVehicle ?anArea))
                     (at start (in ?aVictim ?aVehicle))
 :effect
               (and  (at end (not (in ?aVictim ?aVehicle)))
                     (at end (at ?aVictim ?anArea))
                     (at end (decrease (victimsIn ?aVehicle) (1)))
                     (at end (decrease (maxOccupancy ?anArea) 1))))
```

3.2.3. Especificación del objetivo del plan

El objetivo del plan es asistir a todas las víctimas. Para especificar dicho objetivo dentro del modelo descrito anteriormente, se debe indicar que todos los objetos *Victim* tengan positivo su respectivo predicado *aided*, lo que implica que están ubicados en algún *Evacuation Area*.

3.2.4. Especificación de la métrica a minimizar

La métrica a minimizar es el tiempo de implementación del plan. En el modelo descrito anteriormente será el tiempo que demanda que todos los objetos *Victim* alcancen su respectivo predicado *aided* en positivo.

3.2.5. Especificación del estado inicial

El estado inicial depende de cada inundación y cambia en el transcurso de la misma, por esa razón, se especifica al momento de querer generar un plan, para lo cual, se deben indicar las zonas de riesgo y de evacuación; las víctimas y su ubicación; y los recursos disponibles para la evacuación.

Ejemplo

En el siguiente ejemplo se observa un estado inicial:

- Se especifican los objetos existentes, en este caso, una víctima, un vehículo, un área de emergencia y un área de evacuación.

```
(:objects
    aVictim  - Victim           ; La víctima.
    aVehicle - Vechicle         ; El vehículo.
    anEmArea - EmergencyArea    ; El área de emergencia.
    anEvArea - EvacuationArea)  ; El área de evacuación.
```

- Se inicializan los predicados y las funciones de los objetos.

```
(:init
    ;; VEHICLE INIT
    (available aVehicle )           ; Vehículo disponible.
    (= (speed aVehicle ) 50.0)      ; Velocidad el vehículo 50 Km/h.
    (= (maxCapacity aVehicle ) 25)  ; Capacidad del vehículo 25 víctimas.
    (= (victimsIn aVehicle ) 0)     ; Ninguna víctima dentro del vehículo.
```

```
;; EMERGENCY AREAS INIT
(= (victims anEmArea) 1)        ; Una víctima en el área de emergencia.

;; EVACUATION AREAS INIT
(= (maxOccupancy anEvArea) 150) ; Capacidad del área de evacuación
                                  150 víctimas.

;; LOCATIONS INIT
(at aVictim aEmArea)    ; La víctima se encuentra en el área de emergencia.
(at aVehicle aEvArea)   ; El vehículo se encuentra en el área de
                          evacuación.

;; DISTANCES INIT
(= (distance anEvArea anEvArea)0.0) ; Distancia entre áreas.
(= (distance anEvArea anEmArea)7.0)
(= (distance anEmArea anEvArea)7.0)
(= (distance anEmArea anEmArea)0.0))
```

- Se establece el objetivo del plan para este estado inicial.

```
(:goal
     (and   (aided aVictim)))   ; La víctima con predicado rescatada.
```

- Se establece la métrica a minimizar.

```
(:metric
     minimize (total-time)))    ; Se minimizará el tiempo de
                                    implementación del plan.
```

3.3. Modificación de la especificación del dominio

Dado que los tipos de vehículos *Boat* solo pueden transitar por *Flooded Areas* y las *Evacuation Areas* solo pueden ser accedidas con vehículos de tipo *Truck* entonces, el planner debe seleccionar alguna *Risk Area* intermedia que le permita cambiar a las víctimas de un vehículo a otro.

En el contexto del problema, las áreas inundadas deberían tener en su periferia un área en riesgo de inundación que, por su características, no puede ser utilizada como área de evacuación.

El dominio sin asociación manual de áreas detallado en la sección 3.2.2. *Especificación del dominio*, será llamado de aquí en adelante **dominio automático**. En dicho dominio, el planner construye un espacio de búsqueda a partir de todas las *Risk Areas*. Por lo tanto, el planner seleccionará heurísticamente una *Risk Area* entre todas las posibles. En el caso particular de la Figura 3.5, el planner seleccionaría la *Risk Area 4*, dado que dentro de las posibles, es la que resulta con menor distancia total a la *Evacuation Area*, es decir, menor tiempo de recorrido. Nótese que la distancia sería igual a 11.

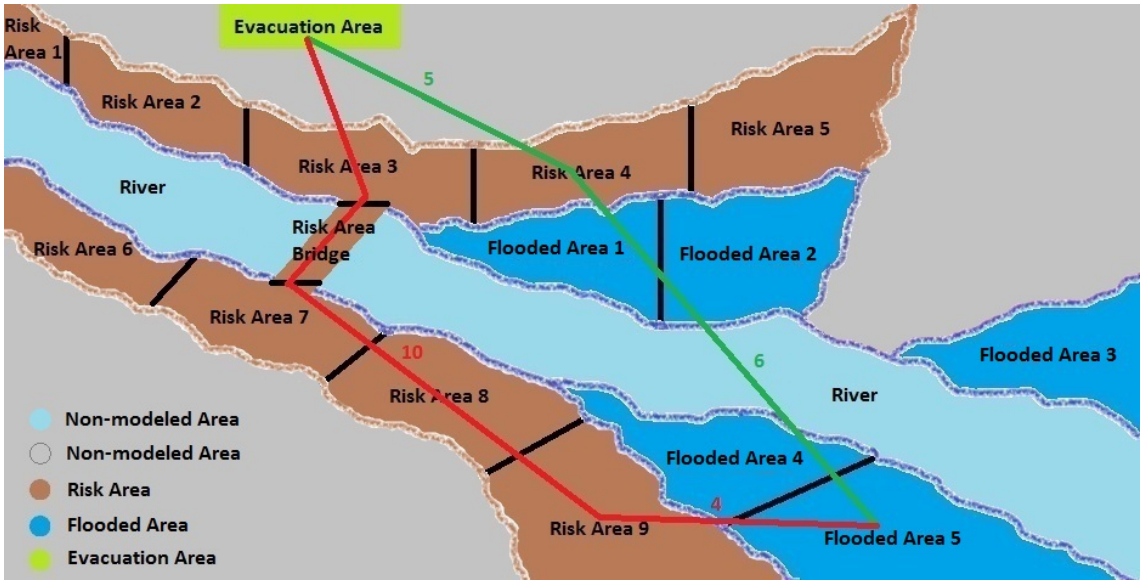

Fig. 3.5: Escenario de un posible problema.

En base al conocimiento de la existencia de áreas intermedias, se decidió modificar el modelo obteniendo una segunda versión de dominio, **dominio manual**. En dicho dominio, toda *Flooded Area* debe estar asociada con al menos una *Risk Area*, las cuales se presumen a mínima distancia entre sí. De esta forma, se delega al operador la tarea de definir el subconjunto de áreas asociadas, limitando al planner la elección de la *Risk Area* adecuada dentro de dicho subconjunto.

En el ejemplo de la Figura 3.5, suponiendo que el operador, bajo el criterio de asociar áreas contiguas, decide asociar la *Flooded Area 5* solamente con la *Risk Area 9*. Entonces, el espacio de búsqueda de áreas a las que puede acceder el planner desde la *Flooded Area 5* se ve reducido únicamente a la *Risk Area 9*, por lo tanto, será la seleccionada. Cabe destacar que el tiempo de selección se reduce, pero la distancia total resultante es igual a 14, es decir, es mayor a la obtenida con el dominio automático.

Con el dominio manual se pretende reducir el espacio de búsqueda construido para seleccionar la *Risk Area*, ya que estaría compuesto solamente por el subconjunto de *Risk Areas* asociadas. Como ya se mencionó, esta reducción impactaría en los tiempos de procesamiento del planner para generar un plan. Sin embargo, se debe tener en cuenta que dicha modificación restringe y complejiza el dominio, lo que podría resultar contraproducente para el desempeño de la heurística.

En el capítulo 4. *Evaluación* se realizará la comparación entre las dos versiones de dominios desarrolladas, el automático y el manual.

3.4. Modificación del planner

Se realizó un análisis preliminar del comportamiento del planner SAPA (véase 2.4.2. *SAPA*) utilizando el modelo desarrollado. Se observó que el desempeño se degrada exponencialmente a medida que aumenta la cantidad de objetos involucrados. Es decir, la

construcción de un plan en un tiempo razonable, se ve afectada por el crecimiento exponencial del espacio de búsqueda, cuya dimensión depende directamente de la cantidad de objetos.

Frente a esta problemática, se implementó una modificación que permite opcionalmente introducir información específica de control conocida por el operador. El propósito de esta modificiación es reducir el espacio de búsqueda sin la necesidad de aumentar el poder de cómputo, ni limitar la cantidad de objetos involucrados. De esta forma, se obtendría una reducción en el tiempo y la memoria requeridos para la creación de un plan. En el capítulo 4. *Evaluación* se presenta la comparación entre el desempeño del planner original y el modificado.

Se pudo comprobar, como era esperable, la existencia de acciones que se repiten consecutivamente. Tal es el caso de la acción *Load* que, según las presunciones del dominio, se debe ejecutar hasta que el vehículo en cuestión se encuentre completo o no haya más víctimas para evacuar en el área. Lo mismo sucede con *Unload*, donde el vehículo debe vaciarse para continuar con la asistencia.

En el marco de este trabajo, se denominaron **acciones repetibles** a las acciones que, por la naturaleza del dominio, es sabido que se deben ejecutar consecutivamente variando uno solo de sus parámetros.

Se modificó la implementación del planner SAPA para que acepte como parámetro de entrada un conjunto de acciones repetibles R. Dicho conjunto es utilizado para reconocer un estado S que tiene como próxima acción a ejecutar alguna acción repetible $r \in R$.

La implementación consiste en intervenir la ejecución del planner capturando el estado S y su conjunto de acciones aplicables $A(S)$. Comenzando con el proceso, se genera un nuevo conjunto $AR(S) \subseteq A(S)$ que contiene las acciones que coinciden con r en nombre y en todos sus parámetros excepto el último. Luego, se ejecuta la acción $r \in AR(S)$ alcanzando un nuevo estado temporal S_k. Partiendo de S_k se reitera el proceso hasta que el conjunto $AR(S_k)$ no contenga más acciones repetibles r.

Finalizada la intervención se continúa con la ejecución normal del planner, es decir, buscando la solución a partir del nuevo estado S_k. De esta manera se obliga al planner a ejecutar las acciones repetibles sin explorar el espacio de búsqueda, es decir, no invierte tiempo intentando ejecutar acciones que, por la naturaleza del dominio, no permiten alcanzar una solución.

Además, el planner debe realizar *backtracking* para desestimar estados que no le permiten alcanzar una solución. Al encontrarse con un estado generado por la modificación se aplica *backtracking* directamente al primer estado capturado, a diferencia del planner original que aplica *backtracking* a cada estado intermedio. De esta forma, el planner modificado reduce, respecto al planner original, el tiempo empleado por dicha técnica.

No es correcto marcar cualquier acción como repetible pues eso cambiaría la semántica del problema y podrían perderse soluciones. No obstante, es posible ver que esta modificación es correcta si el modelador tiene información específica del dominio que garantiza la existencia de acciones repetibles. Es decir, conoce que no es conveniente detener la repetición de las mismas cuando aún se cumple su precondición.

Se puede asegurar que la modificación no descarta soluciones de calidad, dado que sólo

elimina aquellas que ignoran la acción repetible cuando aún se cumple su precondición. Por el dominio del problema, dichas soluciones descartadas seguro serán de menor o igual calidad que las soluciones que repiten la acción.

A continuación, en el capítulo 4. *Evaluación*, se realizará una comparación entre el planner original y el modificado.

4. EVALUACIÓN

En este capítulo se detalla cómo fueron especificados los estados iniciales y los resultados obtenidos al evaluar la factibilidad de la aplicación de técnicas de planning con el dominio de evacuaciones ante inundaciones desarrollado.

En primer lugar, se evalúa el desempeño del planner modificado en comparación con el planner original. En segundo lugar, se compara el desempeño de los dominios desarrollados, es decir, el dominio manual en comparación con el dominio automático. Por último, se evalúa el desempeño del planner modificado con el dominio automático y la calidad de los planes resultantes.

4.1. Transcripción de estados iniciales a lenguaje PDDL

Dada la complejidad que implica especificar estados iniciales en PDDL, se decidió implementar una herramienta (véase 7. *Anexo B: Manual*) que construye un estado inicial a partir de datos ingresados por la interfaz gráfica.

Todos los estados iniciales relacionados con un dominio responden a un mismo esquema. Para realizar la transcripción a PDDL, se debe especializar dicho esquema con los datos ingresados en la interfaz. Por lo tanto, al tratarse de un procedimiento directo, se puede garantizar la correcta transcripción.

Los datos que se ingresan a través de la interfaz gráfica permiten especificar las características principales de los objetos modelados en el dominio:

- *Áreas:*
  - Tipo.
  - Cantidad de víctimas y/o capacidad según corresponda.
  - Distancias.
- *Vehículos:*
  - Tipo.
  - Velocidad.
  - Capacidad máxima.
  - Ubicación.
  - Cantidad de víctimas dentro.

Cabe destacar que, sin pérdida de generalidad, no es necesario identificar a las víctimas dentro de cada área, pues para lograr el objetivo del plan, es suficiente conocer la cantidad de víctimas en cada una. Por lo tanto, en la evaluación, la ubicación inicial de las víctimas es determinada de forma aleatoria en algún *Emergency Area*, conservando la cantidad de víctimas asignadas al área.

4.2. Entorno de ejecución para la evaluación

Los valores de entorno fueron seleccionados priorizando la aplicabilidad de la herramienta, considerando un equipo de escritorio accesible en la actualidad.

Todas las evaluaciones fueron ejecutadas en un mismo equipo con las siguientes características:

- Procesador:
  - Modelo: Intel Core i7.
  - Núcleos: 4.
  - Velocidad: 2.6 - 3.5 GHz.
  - Caché: 6 MB.
- RAM: 16 GB.
- Disco de estado sólido: 128 GB.
- Sistema operativo: Windows 10 - 64bits.

Se determinaron los siguientes características/parámetros sobre la Java Virtual Machine (JVM), salvo que se especifique lo contrario:

- Versión Java: 1.8.0_101.
- Java(TM) SE Runtime Environment (build 1.8.0_101-b13).
- Java HotSpot(TM) 64-Bit Server VM (build 25.101-b13, mixed mode).
- Garbage collector: First Collector (G1) (véase [12]).
- Límite máximo de memoria asignado (Xmx): 8 GB[1].
- Cada estado inicial se ejecutó en una JVM independiente.

4.3. Datos de evaluación

En una entrevista con personal militar experimentado en inundaciones se relevaron los requerimientos para que el sistema sea adecuado en la práctica:

- Se estimó la cantidad de víctimas para la cual una jurisdicción de las fuerzas militares debería estar preparada para brindar asistencia, 300 víctimas por estado inicial.

[1] Se estimó un valor de Xmx que, en la actualidad, fuera accesible en equipos de escritorio, sabiendo que a medida que se disminuyen los valores de Xmx se reduce la cantidad de casos para los que se obtiene solución.

- Se estableció 1 hora como el límite de tiempo máximo para la búsqueda de un plan adecuado.
- Se relevó la cantidad y características de los recursos que ponen a disposición ante una evacuación de tal magnitud.

En todos los estados iniciales, salvo que se especifique los contrario, se utilizaron los siguientes datos:

Tiempo límite de corte: 1 hora.

Víctimas: 300 víctimas a ser asistidas.

Vehículos:

Tipo	Cantidad	Velocidad	Capacidad
Boat	10	20	12
Truck	10	50	25

Áreas:

- 1 *Parking Area* con 20 vehículos.
- 10 *Evacuation Area* con capacidad para 50 víctimas cada una.
- 1 *Risk Area*.
- 1 *Flooded Area*.

4.4. Resultados

A continuación se presentan los resultados de las evaluaciones realizadas.

Para evaluar y comparar el desempeño de los planners se consideraron tres métricas: tiempo, memoria y estados explorados. Se consideró la métrica tiempo dado que es indispensable conocer cuánto tiempo demora el planner en encontrar una solución, pues respondiendo a la hipótesis del trabajo, se quiere evaluar si la técnica de planning es adecuada para generar planes en un tiempo razonable.

Dada la importancia del tiempo, se tuvieron en cuenta otras dos métricas que se supone influyen directamente sobre el mismo, la memoria y los estados explorados. Por un lado, la memoria requerida por el planner puede afectar la búsqueda de la solución e imposibilitar su uso en equipos domésticos. Por otro lado, la cantidad de estados explorados permite observar si existe una relación entre el tiempo y la exploración que realiza el planner para encontrar una solución.

Para evaluar y comparar la calidad de los planes se consideró el tiempo que demanda la ejecución de los planes resultantes, respondiendo a la hipótesis del trabajo, se quieren generar planes que asistan a todas las víctimas en el menor tiempo posible.

Todos los gráficos se muestran en escala logarítmica dado que los datos abarcan un amplio rango de valores.

4.4.1. Desempeño del planner original vs planner modificado

En un análisis preliminar, el desempeño del planner original no permite alcanzar los valores necesarios para cumplir con los requerimientos relevados, es decir, generar planes en menos de una hora que asistan almenos 300 víctimas. Por el contrario, se espera que el planner modificado supere el desempeño del anterior y alcance los valores esperados. Por esta razón, se llevaron a cabo comparaciones de desempeño entre ambos planners teniendo en cuenta su respuesta ante el incremento de víctimas o áreas afectadas.

Caso de análisis 1: comparación de los planners con el dominio automático variando la cantidad de víctimas a evacuar.

Dados los requerimientos, se necesita un planner con un desempeño que permita asistir, en un tiempo razonable, a una cantidad de víctimas igual o superior a 300. Por esta razón, se decidió observar el desempeño de los planners en función de la cantidad de víctimas.

Se consideró una cantidad total de víctimas entre 20 y 500, con una tasa de incremento:

- 10 entre 20 y 150 víctimas.
- 50 entre 150 y 500 víctimas.

La tasa de incremento fue elegida con el propósito de observar con mayor detalle los casos en que ambos planners encuentran solución. Luego, se aumentó el incremento a partir de las 150 víctimas, con el propósito de verificar si el planner modificado lograba superar el objetivo requerido.

Resultados

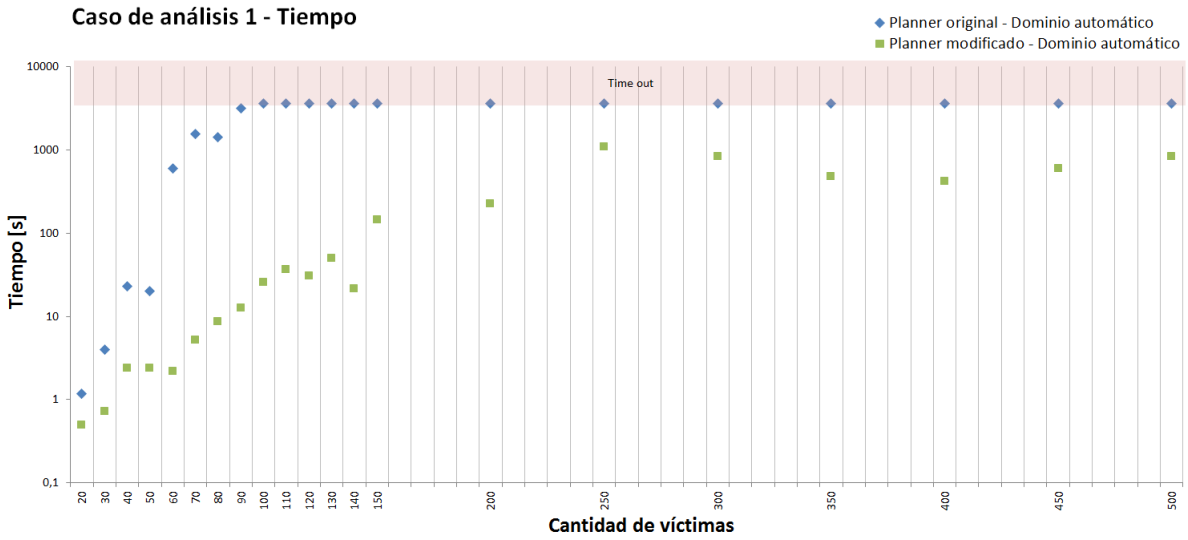

Fig. 4.1: Comparación del tiempo requerido por ambos planners con el dominio automático.

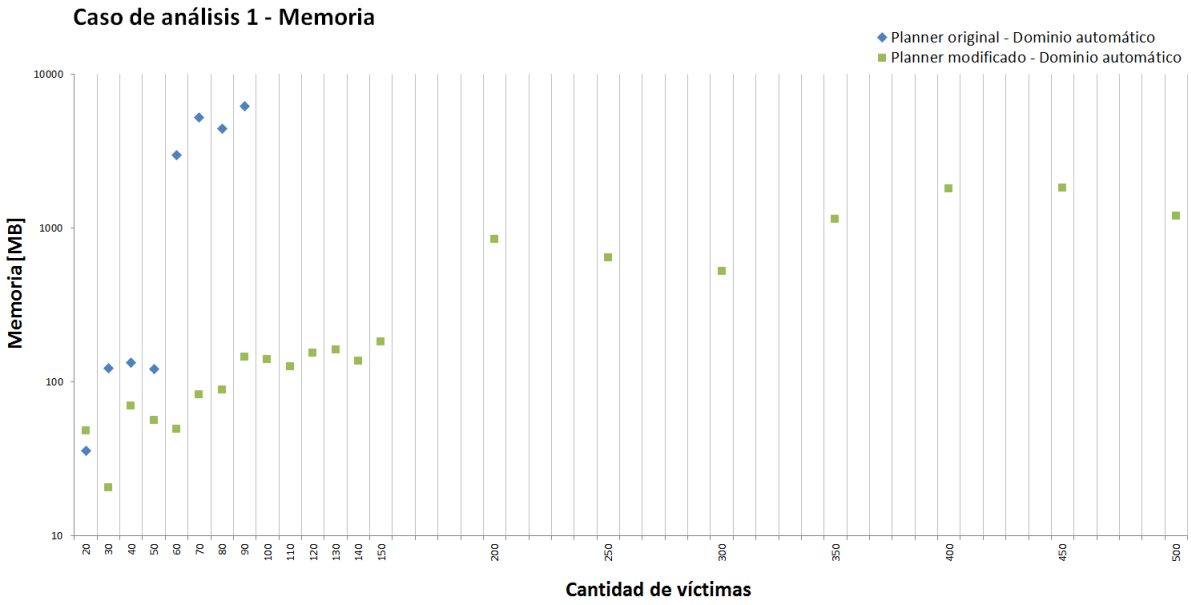

Fig. 4.2: Comparación de la memoria requerida por ambos planners con el dominio automático.

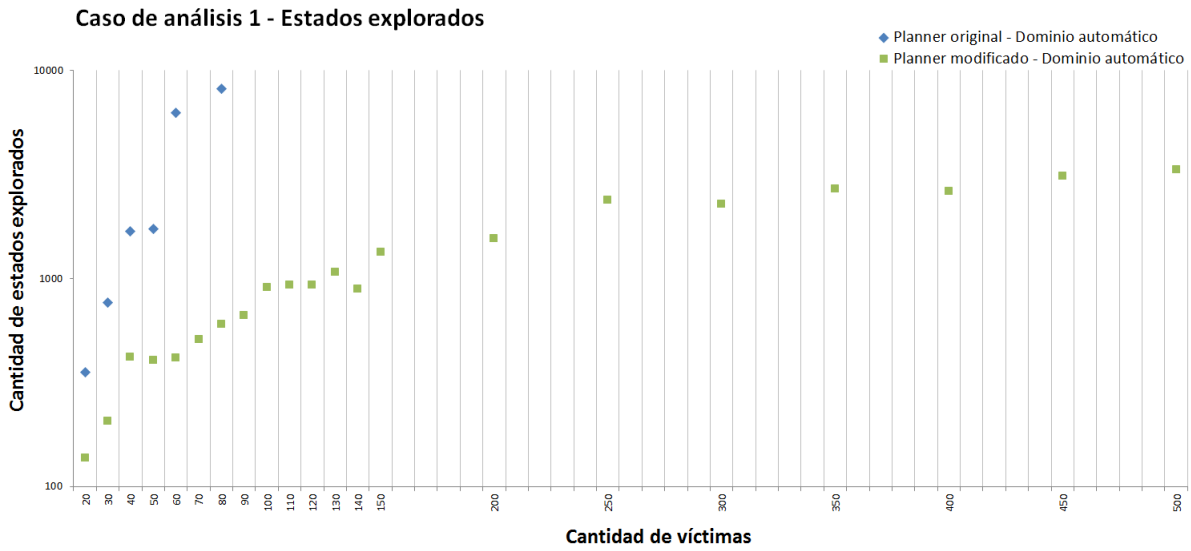

Fig. 4.3: Comparación de la cantidad de estados explorados por ambos planners con el dominio automático.

En el gráfico Fig. 4.1 se puede observar que el planner original no encuentra solución, dentro del tiempo límite máximo 1 hora, en casos donde se superan las 90 víctimas, mientras que el planner modificado, alcanza y supera las 300 víctimas requeridas a fines prácticos.

En los casos en que ambos planners encuentran solución, se observa, por un lado, que el planner modificado reduce considerablemente las métricas de tiempo (Fig. 4.1), memoria (Fig. 4.2) y estados explorados (Fig. 4.3) con respecto al planner original, por lo tanto,

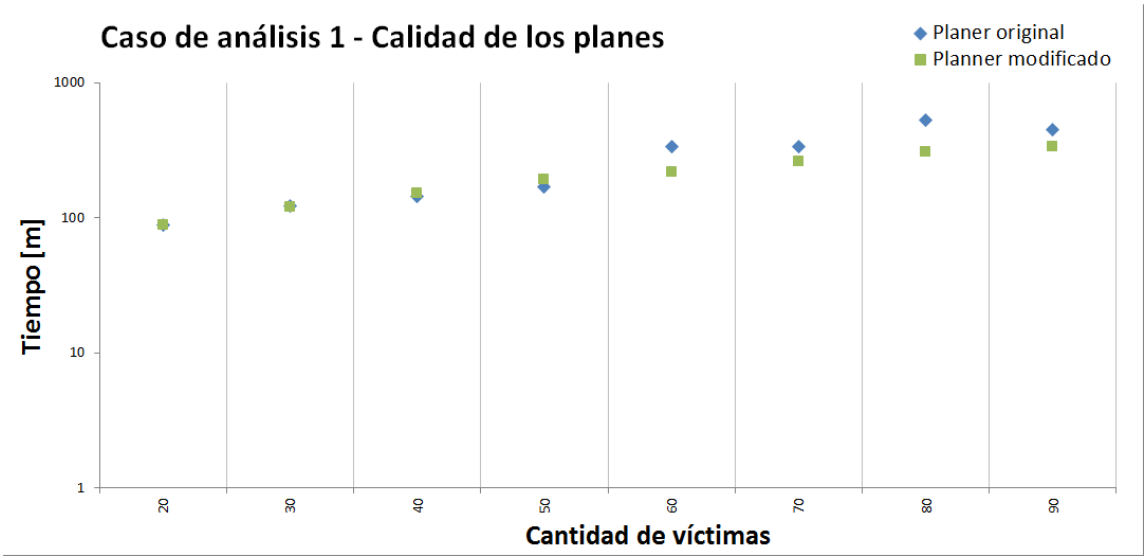

Fig. 4.4: Comparación de la calidad de planes generados por ambos planners con el dominio automático.

mejora su desempeño. Por otro lado, analizando la calidad de los planes generados en el gráfico Fig. 4.4, se puede observar que los obtenidos por el planner modificado tienden, en comparación con el planner original, a requerir menor tiempo para alcanzar el objetivo.

Se puede observar una relación entre el tiempo, la memoria y los estados explorados, es decir, las métricas aumentan exponencialmente a medida que se incrementa la cantidad de víctimas involucradas en el problema.

Existen casos anómalos en los gráficos Fig. 4.1 y Fig. 4.3 donde, respectivamente, al aumentar la cantidad de víctimas disminuye el tiempo y los estados explorados. Dicha anomalía es consecuencia de las diferentes topologías que existen para un estado inicial, las cuales pueden provocar una variación en el proceso de búsqueda de una solución. Es factible que un pequeño incremento de víctimas no constituya realmente un problema más complejo para el planner, sin embargo, se destaca la tendencia exponencial que se puede observar en el gráfico.

Caso de análisis 2: comparación de los planners con el dominio automático variando la cantidad de áreas de emergencia.

En una inundación, el área afectada puede ser lo suficientemente grande como para que el tiempo que demande recorrerla no sea despreciable. Por la presunción *P7 (debe ser despreciable el tiempo que demanda recorrer un área)* dicha área no puede ser considerada como una sola área de emergencia y debe ser dividida en subáreas más pequeñas. Considerar mayor cantidad de áreas, puede complejizar la búsqueda de una solución, y por lo tanto, afectar el desempeño de los planners.

Para llevar a cabo esta evaluación, se incrementó la cantidad de áreas de emergencia con una tasa de incremento 2 entre 4 y 24; donde la mitad corresponden a *Flooded Area* y la otra mitad a *Risk Area*.

Con el propósito de poder llevar a cabo la comparación entre los planners, dado que el planner original no supera las 90 víctimas, se consideraron 60 víctimas a ser asistidas.

Resultados

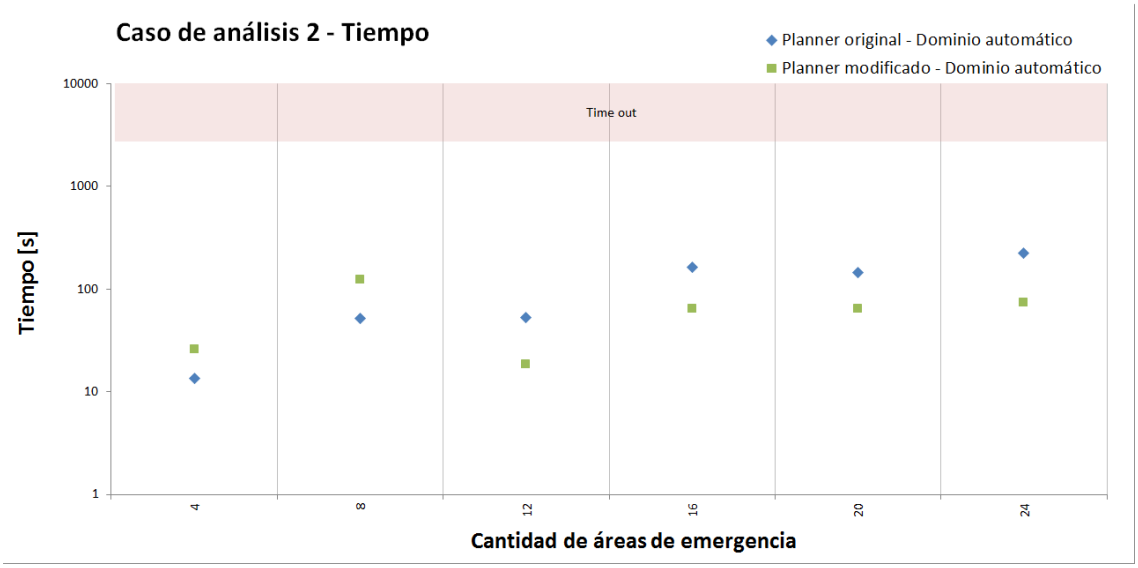

Fig. 4.5: Comparación del tiempo requerido por ambos planners con el dominio automático.

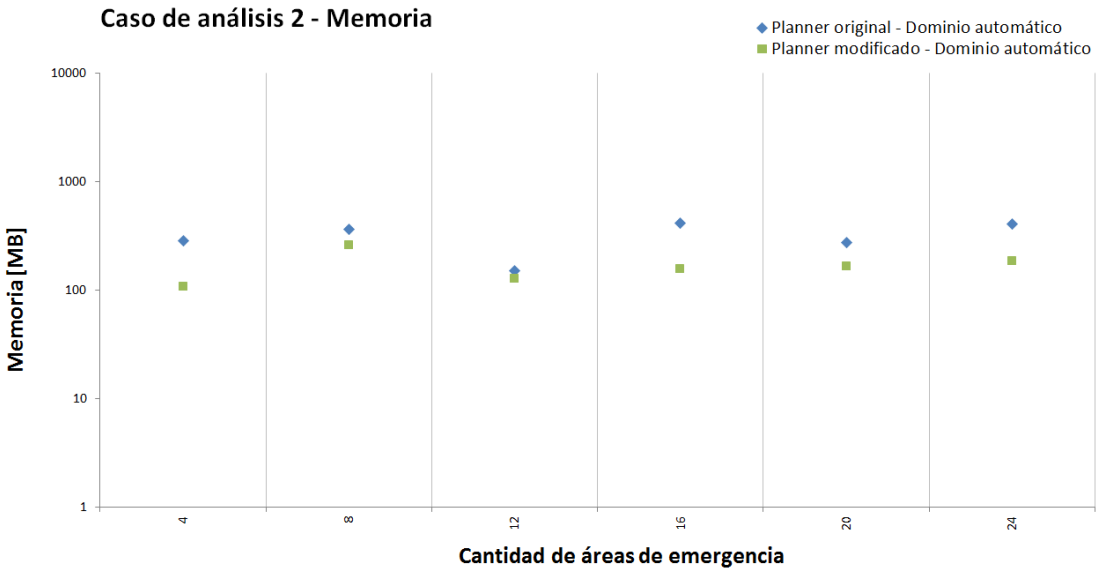

Fig. 4.6: Comparación de la memoria requerida por ambos planners con el dominio automático.

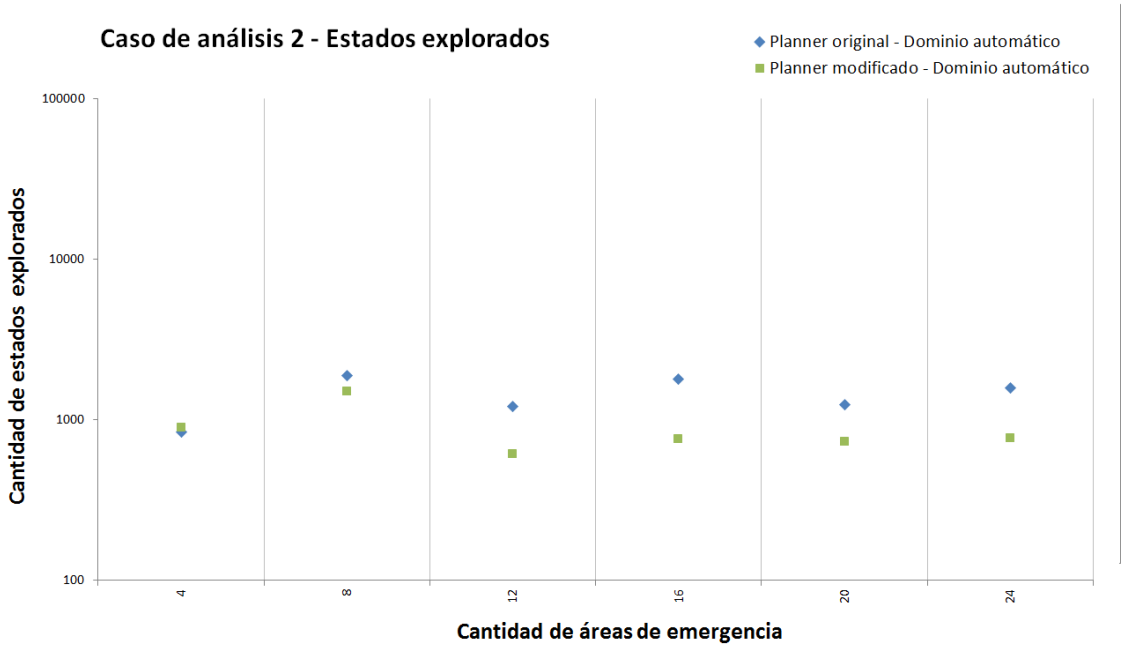

Fig. 4.7: Comparación de la cantidad de estados explorados por ambos planners con el dominio automático.

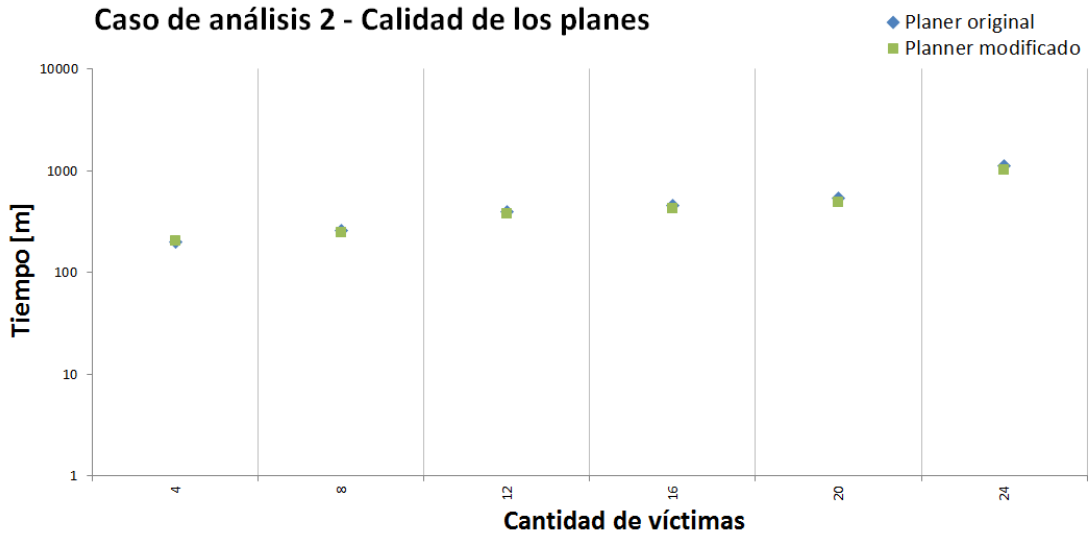

Fig. 4.8: Comparación de la calidad de planes generados por ambos planners con el dominio automático.

En los gráficos se puede observar que el planner modificado tiende a reducir las métricas de tiempo (Fig. 4.5), memoria (Fig. 4.6) y estados explorados (Fig. 4.7) con respecto al planner original, lo cual evidencia que el planner modificado presenta un mejor desempeño que el original. Más aún, al aumentar la cantidad de áreas, el tiempo se conserva en forma casi constante, mientras que el planer original lo aumenta considerablemente.

Sin embargo, las métricas de memoria y estados explorados no mantienen una relación

significativa con la cantidad de áreas. La cantidad de estados explorados y, sobretodo, la memoria, se mantienen prácticamente constantes en función de la cantidad de áreas de emergencias.

Además, existen casos anómalos donde las métricas se reducen o aumentan significativamente respecto al aumento de la cantidad de áreas, lo cual, como se mencionó anteriormente, se debe a las distintas topologías de los estados iniciales, que provocan variaciones en el tiempo que requiere encontrar una solución, pues es factible que aumentar la cantidad de áreas no necesariamente implique un problema más complejo para el planner.

Analizando la calidad de los planes generados (Fig. 4.8), nuevamente, se observa la tendencia del planner modificado a reducir el tiempo de los planes, respecto al planner original.

A partir de los casos *Caso de análisis 1: comparación de los planners con el dominio automático variando la cantidad de víctimas a evacuar* (véase 4.4.1) y *Caso de análisis 2: comparación de los planners con el dominio automático variando la cantidad de áreas de emergencia* (véase 4.4.1), se puede afirmar que el planner modificado supera el desempeño del planner original, e incluso logra solucionar estados iniciales con cantidad de víctimas que superan los requerimientos prácticos, es decir, la modificación permitió aumentar el límite de los problemas computables.

4.4.2. Desempeño del planner modificado con el dominio manual

Con el propósito de evaluar el desempeño del planner modificado con el dominio manual, se consideró su desempeño ante el incremento de víctimas y áreas de emergencia y la calidad de los planes generados. Además se lo comparó con el desempeño alcanzado por el mismo planner con el dominio automático. En dicha comparación se puede observar, si el desempeño del dominio manual supera o no, el alcanzado con el dominio automático.

Caso de análisis 3: comparación de los dominios con el planner modificado variando la cantidad de víctimas a evacuar.

Se realizó una comparación entre los dos dominios, manual y automático, utilizando el planner modificado en ambos casos, considerando el incremento de víctimas a asistir.

Para evaluar ambos dominios en igualdad de condiciones, se utilizó el mismo conjunto de estados iniciales mencionado en el *Caso de análisis 1: comparación de los planners con el dominio automático variando la cantidad de víctimas a evacuar* (véase 4.4.1. *Desempeño del planner original vs planner modificado*), adaptándolo a la especificación del dominio manual, es decir, se asignó a cada *Flooded Area* una *Risk Area*.

Al considerar estados iniciales con dos áreas de emergencia, una *Risk Area* y una *Flooded Area*, las cuales se encuentran asociadas en el contexto del dominio manual, se puede asegurar que no debería existir diferencia sustancial entre ambos dominios. Puesto que esto equivale a considerar todas las áreas como seleccionables, tal como hace el dominio automático.

Resultados

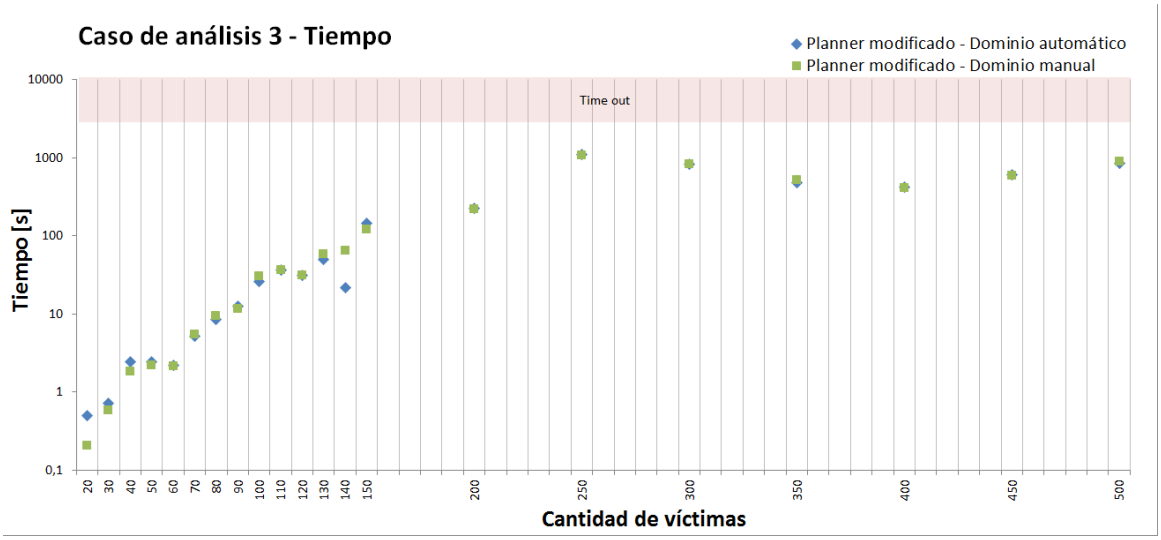

Fig. 4.9: Comparación del tiempo requerido por el planner modificado con ambos dominios.

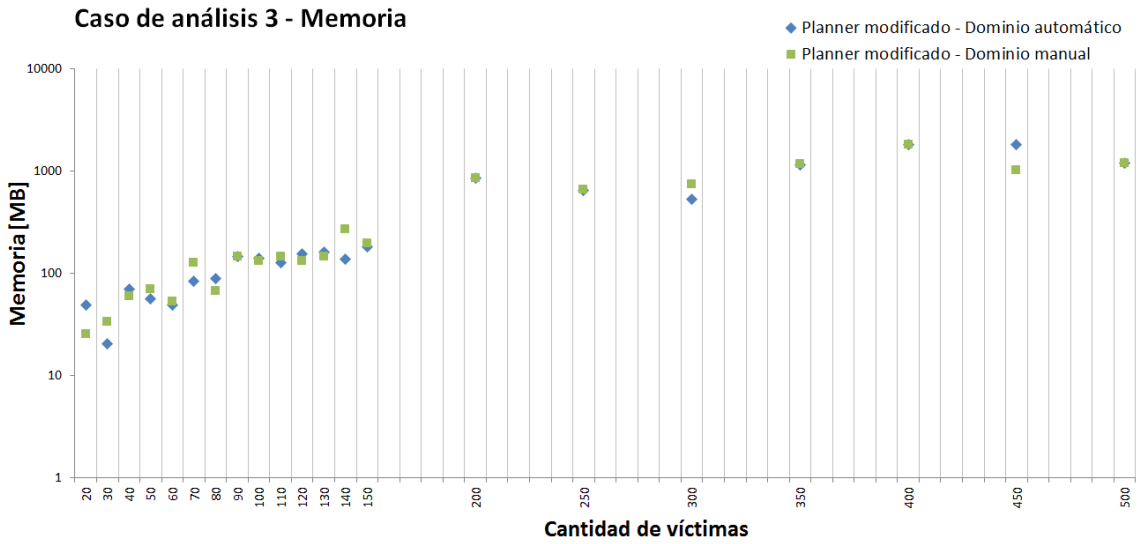

Fig. 4.10: Comparación de la memoria requerida por el planner modificado con ambos dominios.

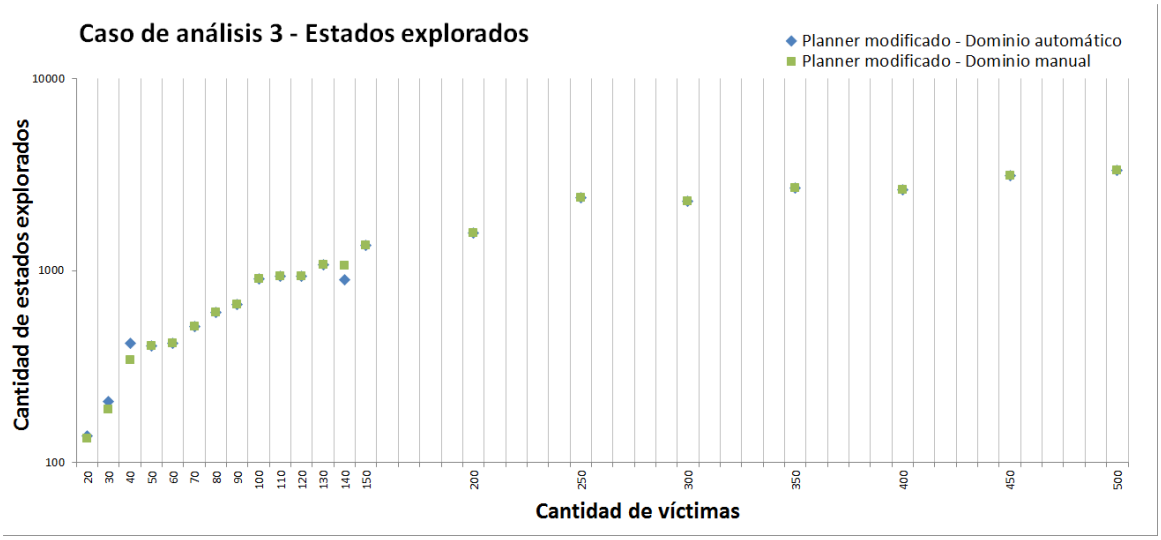

Fig. 4.11: Comparación de la cantidad de estados explorados por el planner modificado con ambos dominios.

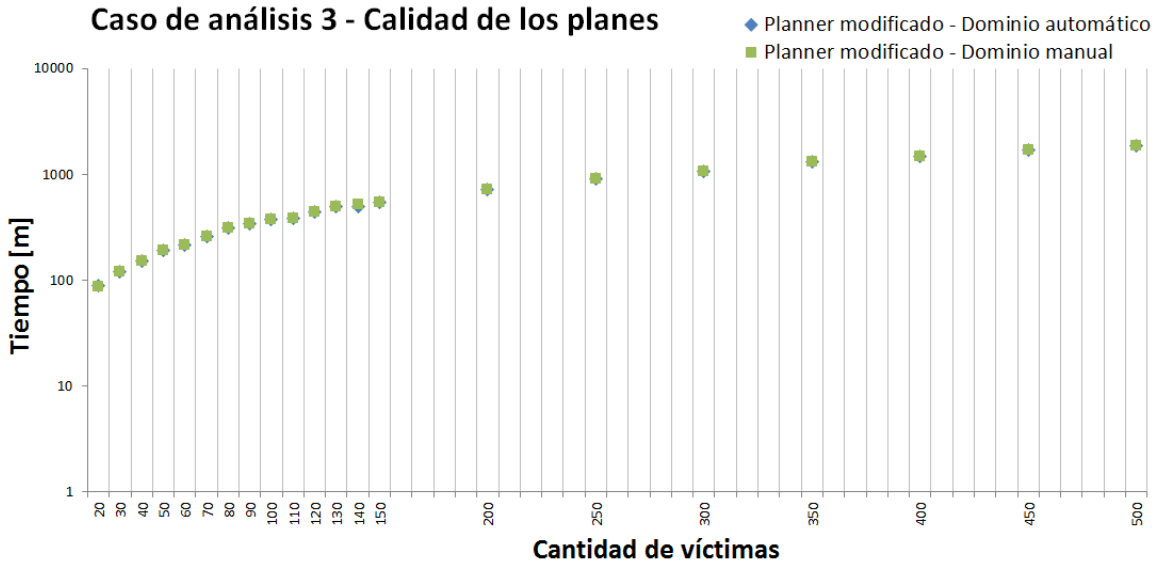

Fig. 4.12: Comparación de la calidad de planes generados por el planner modificado con ambos dominios.

En los gráficos de tiempo (Fig. 4.9), memoria (Fig. 4.10) y estados explorados (Fig. 4.11) se puede observar que con ambos dominios, manual y automático, al aumentar la cantidad de víctimas, el desempeño del planner es similar. Es decir, bajo iguales condiciones de estados iniciales, el domino manual no mejora ni empeora considerablemente el desempeño del planner respecto al desempeño logrado con el dominio automático.

En el gráfico Fig. 4.12, donde se evalúa la calidad de los planes generados por el planner modificado con ambos dominios, se observa que más del 85 % de los planes demandan el mismo tiempo de ejecución. Mientras que, los planes restantes, se diferencian en menos de un 5 % uno respecto al otro. Sin embargo, existe una leve tendencia del planner modificado

con el dominio automático a generar planes con mejor calidad que con el dominio manual.

Se puede deducir que, al aumentar la cantidad de víctimas, la heurística utilizada por el planner para reducir el espacio de búsqueda, genera resultados similares a los que podría aportar un operador con buen criterio de asignación.

Caso de análisis 4: comparación de los dominios con el planner modificado variando la cantidad de áreas de emergencia.

Dado que el dominio manual consiste en asociar áreas de emergencia, se supuso que modificar la cantidad de áreas sería un posible factor que afecta el desempeño del planner con este dominio. Por esta razón, se decidió evaluar el desempeño del planner modificado con ambos dominios en relación a la cantidad de áreas de emergencia.

Para evaluar ambos dominios en igualdad de condiciones, se utilizó el mismo conjunto de estados iniciales mencionado anteriormente en el *Caso de análisis 2: comparación de los planners con el dominio automático variando la cantidad de áreas de emergencia* (véase 4.4.1. *Desempeño del planner original vs planner modificado*), adaptándolo a la nueva especificación, es decir, se asignó a cada *Flooded Area* una *Risk Area*. Para evidenciar aún más la diferencia, se sumaron los casos con cantidad de áreas de emergencia: 32 y 40; donde la mitad corresponden a *Flooded Area* y la otra mitad a *Risk Area*.

Resultados

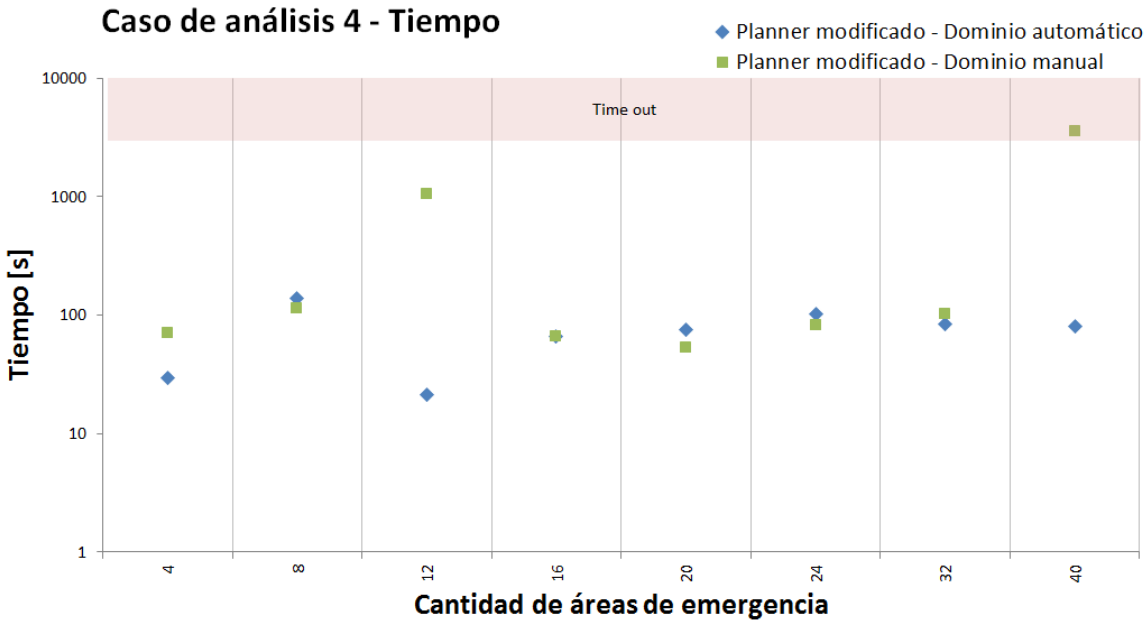

Fig. 4.13: Comparación del tiempo requerido por el planner modificado con ambos dominios.

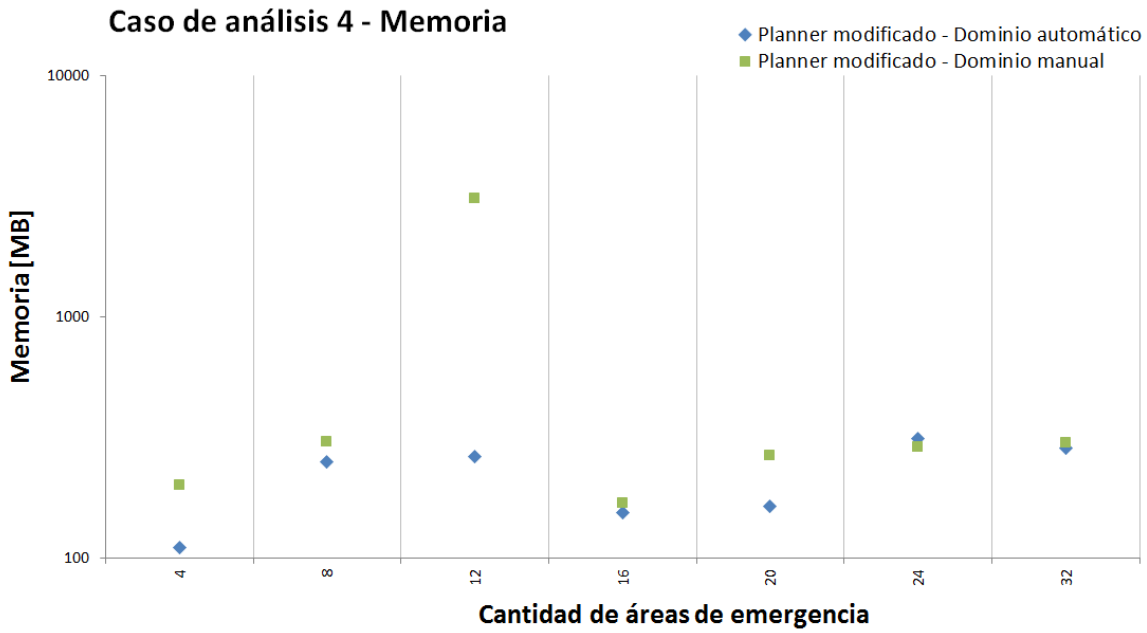

Fig. 4.14: Comparación de la memoria requerida por el planner modificado con ambos dominios.

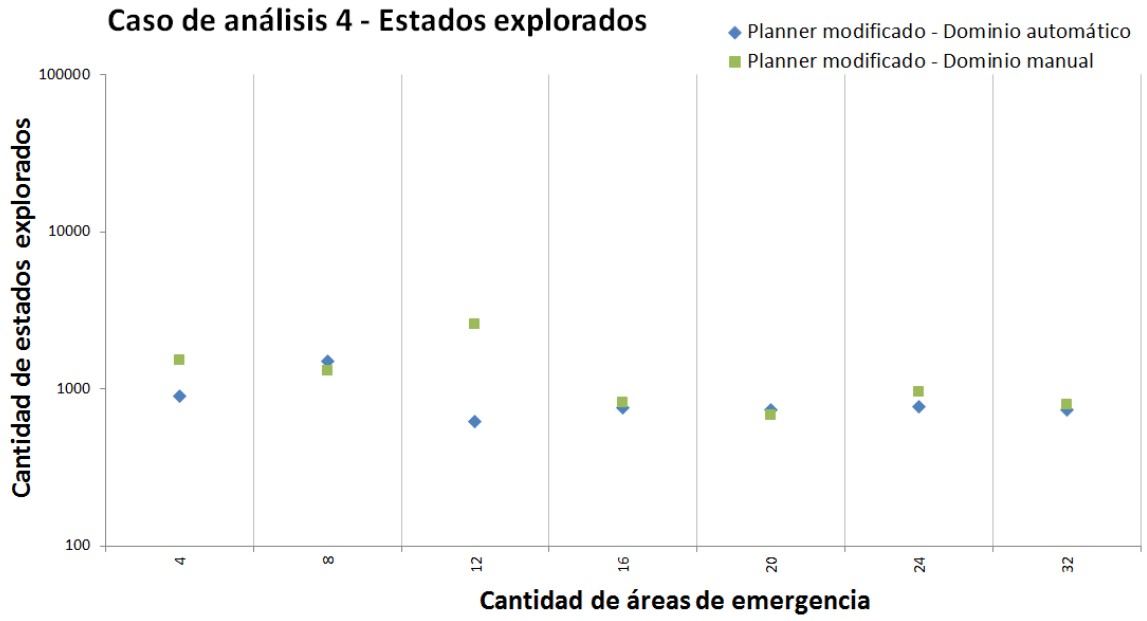

Fig. 4.15: Comparación de la cantidad de estados explorados por el planner modificado con ambos dominios.

En los gráficos de tiempo (Fig. 4.13), memoria (Fig. 4.14) y estados explorados (Fig. 4.15) se puede observar que el desempeño del planner con el dominio manual no mejora respecto al dominio automático, por el contrario, tiende a superarlo en todas las métricas. Incluso, en el gráfico Fig. 4.13 se evidencia que el planner con el dominio manual, en el caso con 40 áreas, no encuentra solución dentro del tiempo límite considerado.

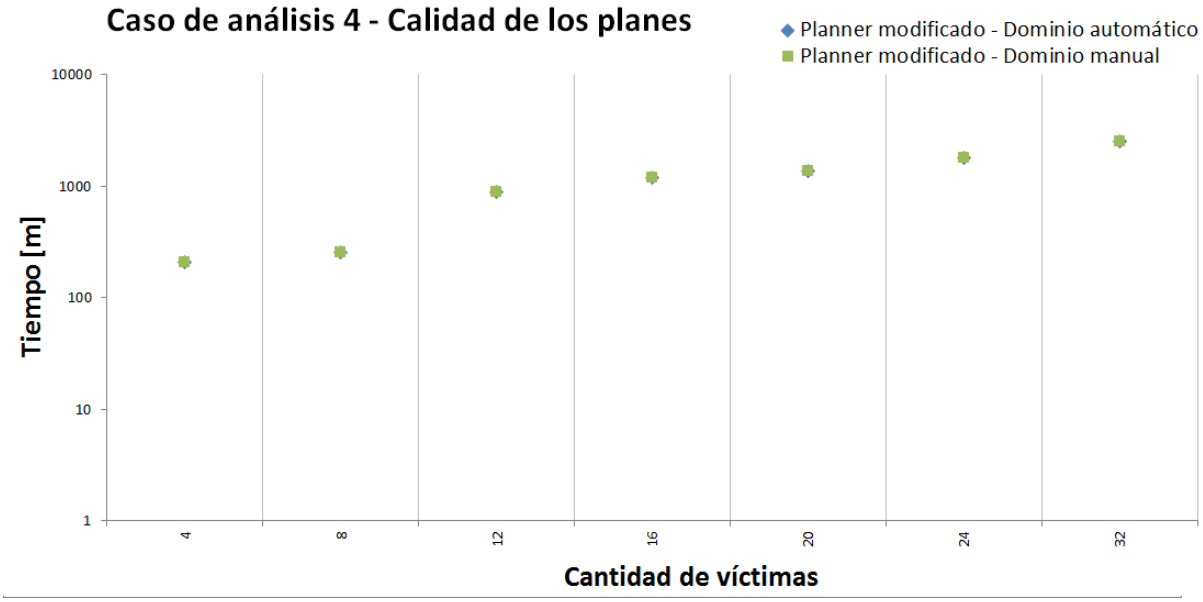

Fig. 4.16: Comparación de la calidad de planes generados por el planner modificado con ambos dominios.

Al igual que en los casos de análisis anteriores, se observan casos anómalos donde la variabilidad refleja las diversas topologías de los estados iniciales que afectan la búsqueda realizada por el planner, pues aumentar la cantidad de áreas no necesariamente implica un problema más complejo para el planner. Lo más destacable es observar la tendencia que el planner sigue con ambos dominios.

Respecto a la calidad de los planes generados (Fig. 4.16), se puede observar una marcada tendencia del planner modificado a generar planes que demandan menor tiempo de ejecución que los planes generados por el planner original.

En base a los análisis de esta sección, se puede observar que al incrementar la cantidad víctimas o de áreas de emergencias, el desempeño del planner con el dominio automático es igual o más robusto que el dominio manual.

4.4.3. Desempeño general del planner modificado con el dominio automático

A partir de los análisis anteriores, se observó un mejor desempeño del planner modificado con el dominio automático, por esta razón, se decidió profundizar en la evaluación del mismo.

Caso de análisis 5: planner modificado con el dominio automático variando la cantidad de vehículos.

Con el propósito de analizar la eficiencia del planner respecto a la cantidad de vehículos disponibles se analizó el desempeño del planner modificado con el dominio automático va-

riando la cantidad de vehículos. El análisis se hizo bajo la siguiente suposición: el planner debería mejorar su desempeño hasta alcanzar un punto donde enviar más vehículos no es necesario para minimizar el tiempo.

Se varió la cantidad de vehículos con una tasa de incremento 5 entre 10 y 45, donde la mitad corresponden a *Boat* y la otra mitad a *Truck*. El límite superior fue determinado por la imposibilidad del planner de generar soluciones para casos mayores. Se consideraron las mismas características de velocidad, capacidad y ubicación de los vehículos que en los casos anteriores.

Resultados

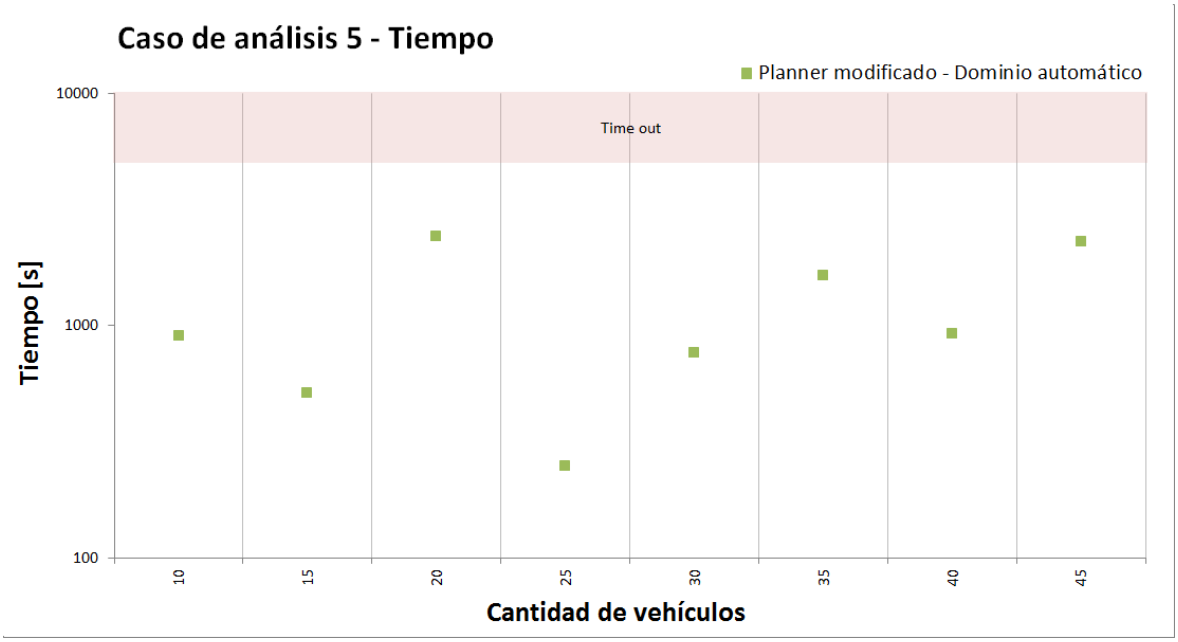

Fig. 4.17: Desempeño del planner modificado con el dominio automático en relación al tiempo requerido.

En los gráficos Fig. 4.18 y Fig. 4.19 se observa respectivamente y contrario a lo esperable, que la memoria y los estados explorados no se ven considerablemente afectados por el aumento de la cantidad de vehículos.

Al evaluar la calidad de los planes se encontró que todos los planes generados en este caso de análisis utilizan igual cantidad de vehículos, uno de cada tipo, entonces, todos los planes resultantes son similares, pero no iguales, producto de las diferencias en la topología de cada estado inicial. Razón por la cual, no se observa una pronunciada variabilidad en los gráficos de memoria y estados explorados.

Se observó que los planes generados no envían más de un vehículo del mismo tipo, lo cual permitiría que varios vehículos asistan en paralelo una misma área. Por lo tanto, al contrario de lo esperado, el aumento en la cantidad de vehículos no implica planes que se ejecutan en menor tiempo. En el *Caso de análisis 9: calidad de los planes obtenidos por el planner modificado con el dominio automático aumentando la cantidad de vehículos.*

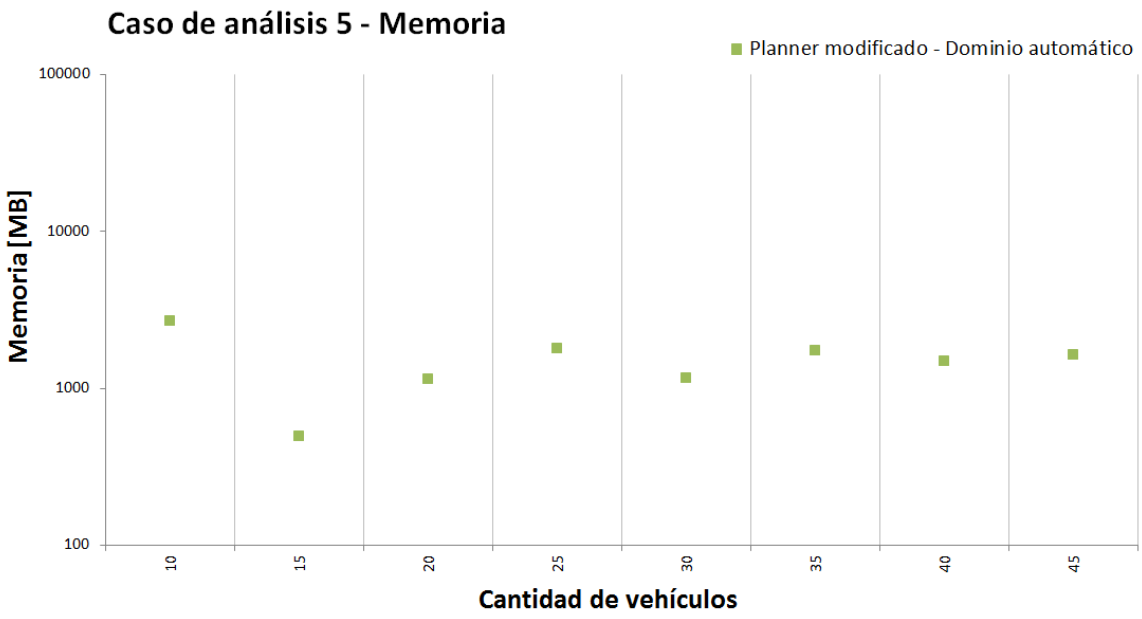

Fig. 4.18: Desempeño del planner modificado con el dominio automático en relación a la memoria requerida.

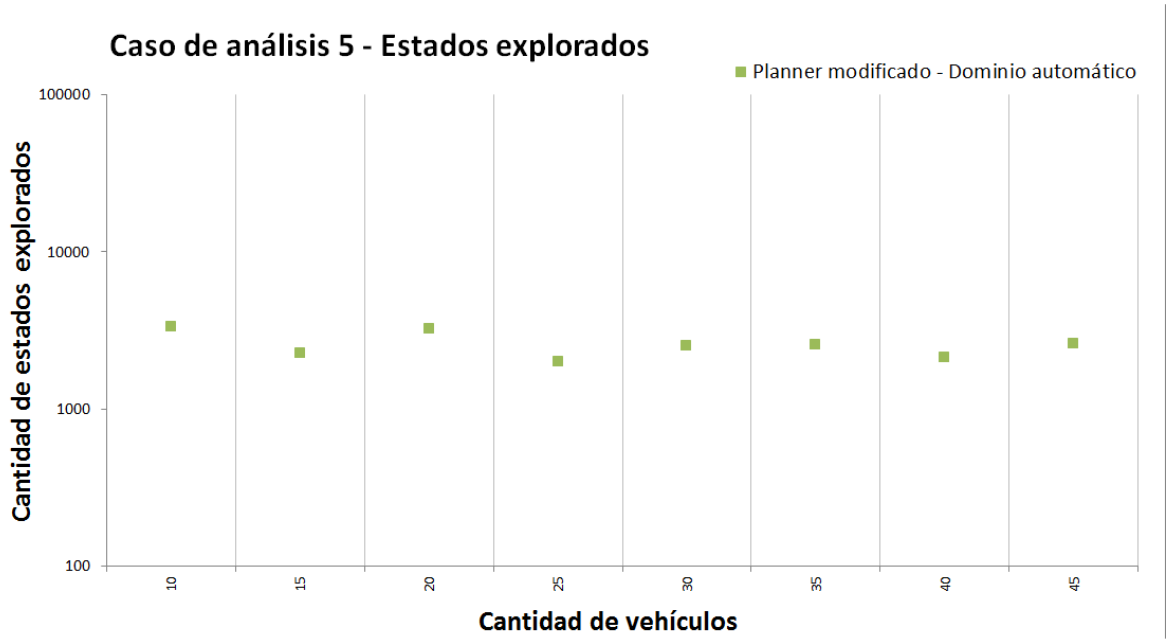

Fig. 4.19: Desempeño del planner modificado con el dominio automático en relación a la cantidad de estados explorados.

(véase 4.4.4. *Calidad de los planes*), se desarrolla este comportamiento que responde a una limitación inherente al planner, dado que depende de la selección de acciones aplicables que realiza, por lo tanto, es independiente del dominio y la modificación que se desarrollaron.

En el gráfico Fig. 4.17 se observa una tendencia de aumento del tiempo a medida que

aumenta la cantidad de vehículos. Se entiende que la mayor cantidad de recursos para administrar tiene como consecuencia, el aumento del tiempo que le insume al planner generar un plan. Sin embargo, se observa una gran variabilidad entre el caso 20 y 25, al igual que en los casos de análisis anteriores, esto se corresponde con casos donde la topología del estado inicial afecta a la búsqueda del planner.

Caso de análisis 6: planner modificado con el dominio automático aumentando la cantidad de víctimas a evacuar.

Se realizó un análisis de estrés del planner modificado en función de la cantidad de víctimas, con el propósito de obtener una cota superior que permita conocer los límites alcanzados por el planner modificado con el dominio automático.

Por otro lado, se decidió observar la relación que existe entre los estados explorados, la memoria requerida y el tiempo requerido. Se espera observar una relación lineal entre las métricas.

La tasa de incremento es 100, entre 100 y 1000 víctimas. Como consecuencia del aumento de víctimas, se aumentó la capacidad de las *Evacuation Area* para conservar la presunción *P6 (La cantidad de víctimas no debe superar la capacidad total de los centros de evacuación).*

Resultados

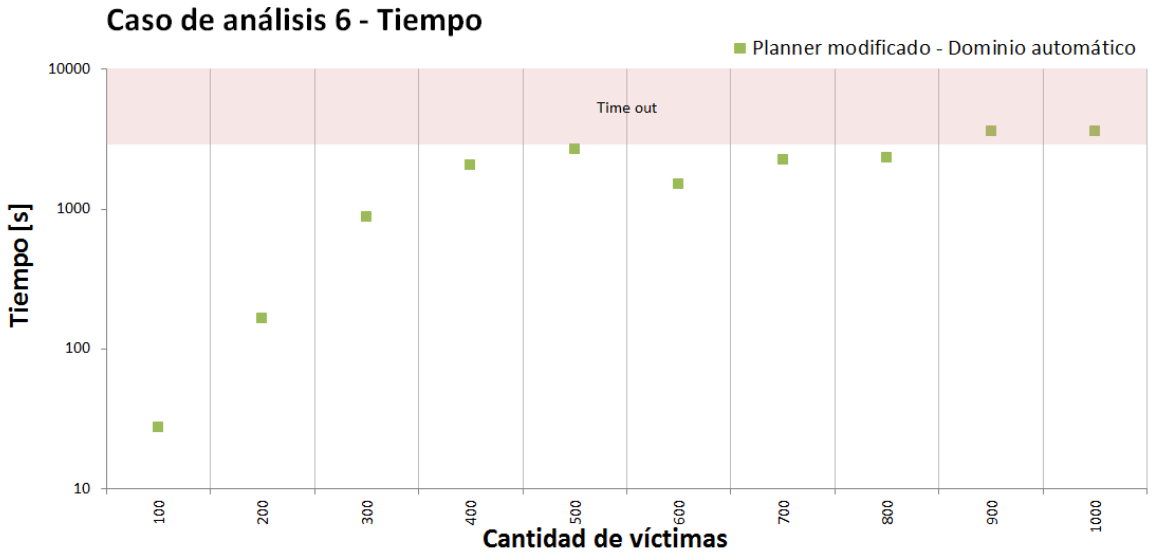

Fig. 4.20: Desempeño del planner modificado con el dominio automático en relación al tiempo requerido.

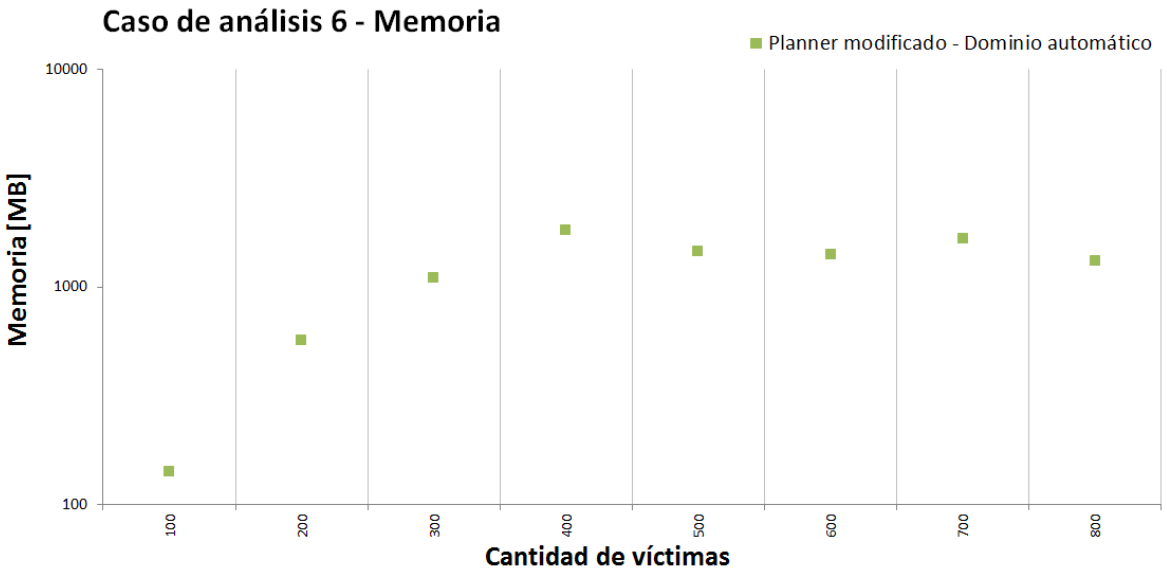

Fig. 4.21: Desempeño del planner modificado con el dominio automático en relación a la memoria requerida.

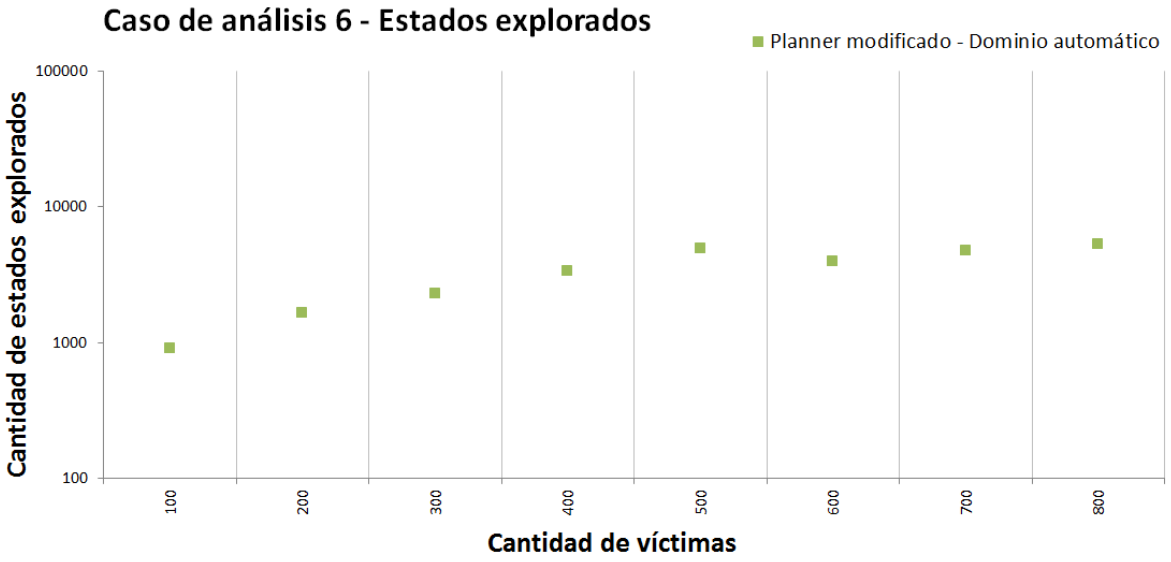

Fig. 4.22: Desempeño del planner modificado con el dominio automático en relación a la cantidad de estados explorados.

En el análisis de estrés del planner modificado con el dominio automático se puede observar que la cantidad máxima de víctimas para la cual encuentra solución es 800.

La cota superior alcanzada supera ampliamente la cota requerida para su uso en la práctica, la cual era 300 víctimas. Además, bajo las mismas condiciones, queda evidenciado nuevamente, que el planner modificado mejora el desempeño del planner original, superando su cota superior de 90 víctimas obtenida en el *Caso de análisis 1: comparación de los planners con el dominio automático variando la cantidad de víctimas a evacuar*

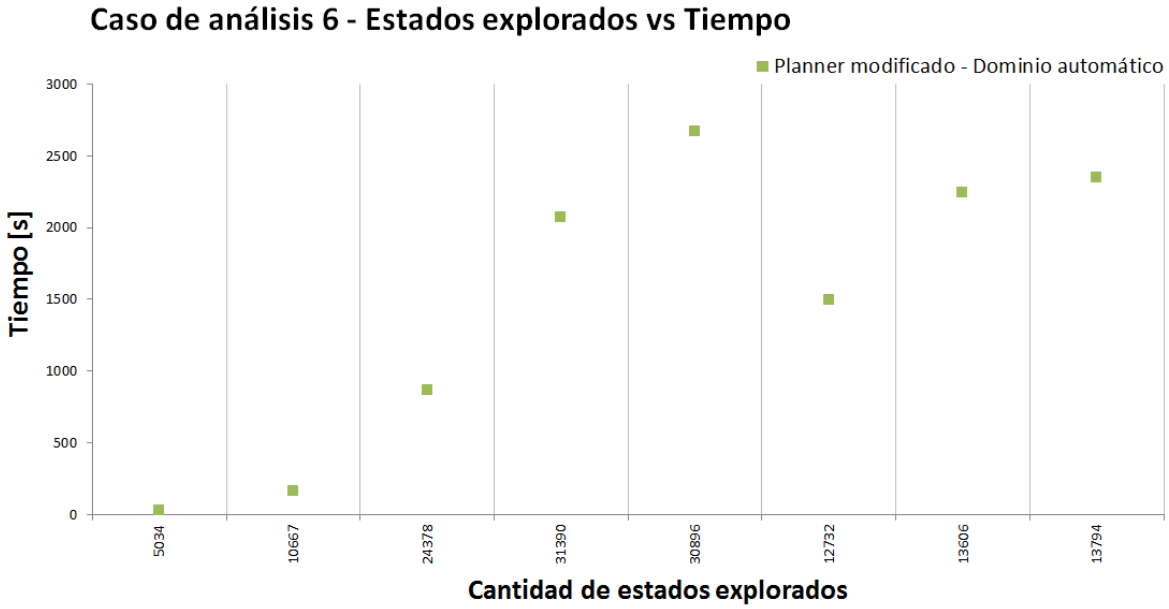

Fig. 4.23: Desempeño del planner modificado con el dominio automático en relación a la cantidad de estados explorados y el tiempo requerido.

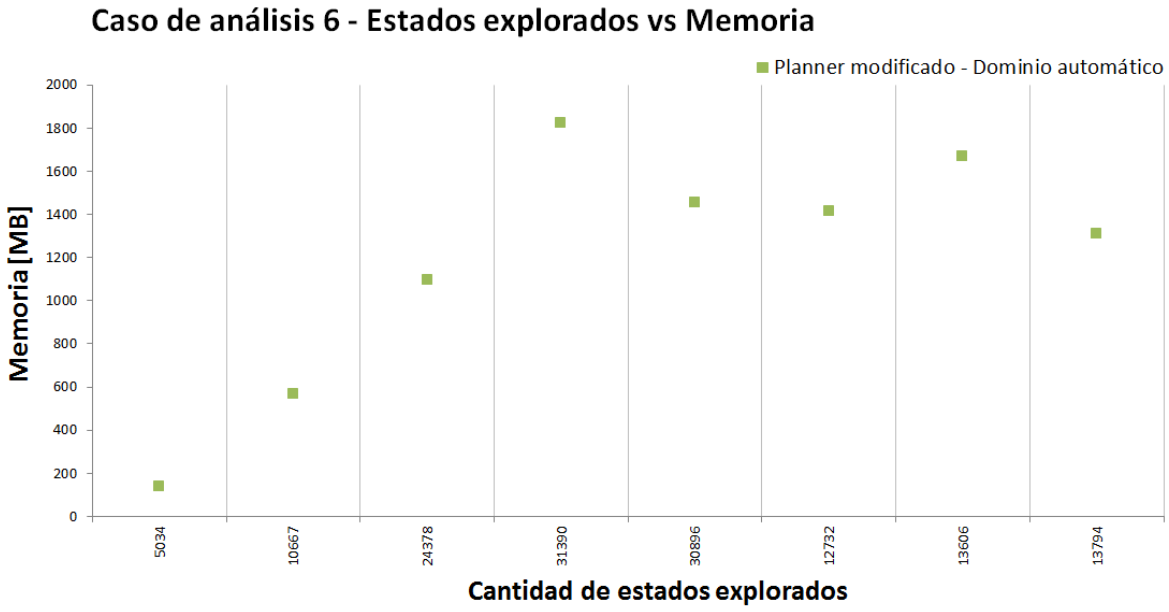

Fig. 4.24: Desempeño del planner modificado con el dominio automático en relación a la cantidad de estados explorados y la memoria requerida.

(véase 4.4.1. *Desempeño del planner original vs planner modificado*).

Como es esperable, el aumento en la cantidad de víctimas determina una tendencia de aumento exponencial en las métricas tiempo (Fig. 4.25), memoria (Fig. 4.26) y estados explorados (Fig. 4.27). Por otro lado, se evidencia una relación directa entre las tres

métricas.

Por último, tal como era esperable, existe una relación lineal entre la memoria en función de los estados explorados y el tiempo requerido en función de los estados explorados. El aumento de la cantidad de estados explorados, requiere el almacenamiento de más objetos, por lo tanto el uso de más memoria. Mientras que, el aumento de estados a explorar, demanda más tiempo.

Caso de análisis 7: planner modificado con el dominio automático aumentando la cantidad de víctimas a evacuar comparando distintos *garbage collector*.

Se compararon distintos *garbage collector* provistos por la Java Virtual Machine (JVM) (véase [13] para revisiones de la literatura sobre el tema) al ejecutar el planner modificado con el dominio automático. Es conocido que la Java Virtual Machine provee distintos parámetros que permiten ajustar los algoritmos de administración del heap de memoria. Dichos ajustes podrían influir en el tiempo de generación del plan. El propósito de este análisis es conocer cómo influye cada política de administración de memoria en el desempeño del planner.

Se compararon los siguientes *garbage collector*:

- First (G1)
- Parallel
- Serial
- Concurrent Mark Sweep (CMS)

Se utilizó el mismo conjunto de estados iniciales mencionado en el *Caso de análisis 6: planner modificado con el dominio automático aumentando la cantidad de víctimas a evacuar* (véase 4.4.3).

Resultados

En el análisis de los *garbage collectors* sobre el planner modificado con el dominio automático se puede observar que las distintas políticas de administración de memoria no evidencian una diferencia considerable en el desempeño del planner, conservando la cantidad máxima de víctimas para la cual se encuentra solución: 800.

No se han evaluado casos intermedios entre 800 y 900 víctimas dado que, encontrar valores más precisos, no sería representativo a los fines de cumplir con los requerimientos prácticos, considerando que la cota superior alcanzada supera ampliamente la cota requerida, 300 víctimas.

El tiempo que demanda la generación de los planes no se ve afectado considerablemente por la utilización de los distintos *garbage collector*. Mientras que se observa una leve disminución de la memoria requerida al utilizar el *Concurrent Mark Sweep Garbage Collector (CMS)* respecto a los demás *garbage collectors*.

La cantidad de estados explorados no se ve afectada dado que, la política de administración de memoria no influye sobre la búsqueda realizada por el planner.

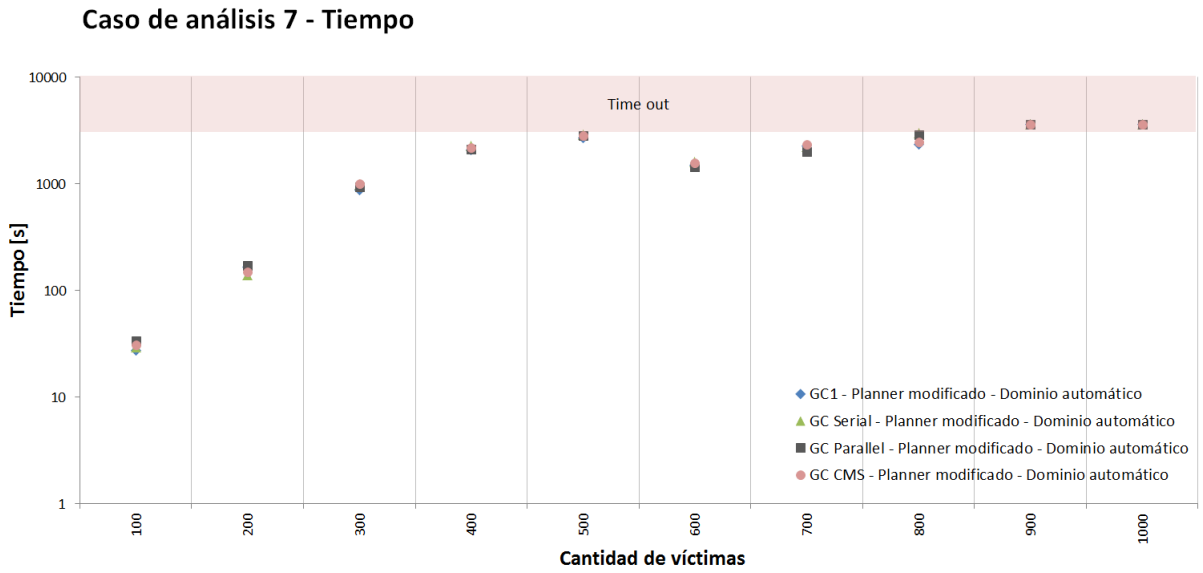

Fig. 4.25: Desempeño del planner modificado con el dominio automático en relación al tiempo requerido.

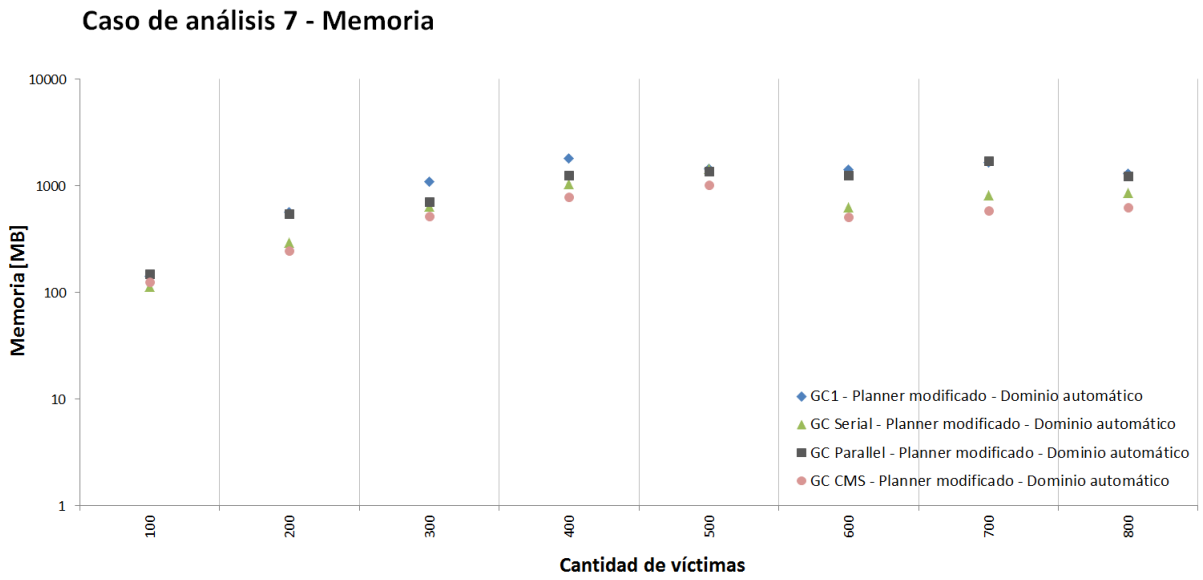

Fig. 4.26: Desempeño del planner modificado con el dominio automático en relación a la memoria requerida.

Se propone a futuro, más allá de los resultados obtenidos en este caso de análisis, realizar una investigación que determine cómo influyen, sobre el desempeño del planner, los diferentes parámetros que permite ajustar la JVM.

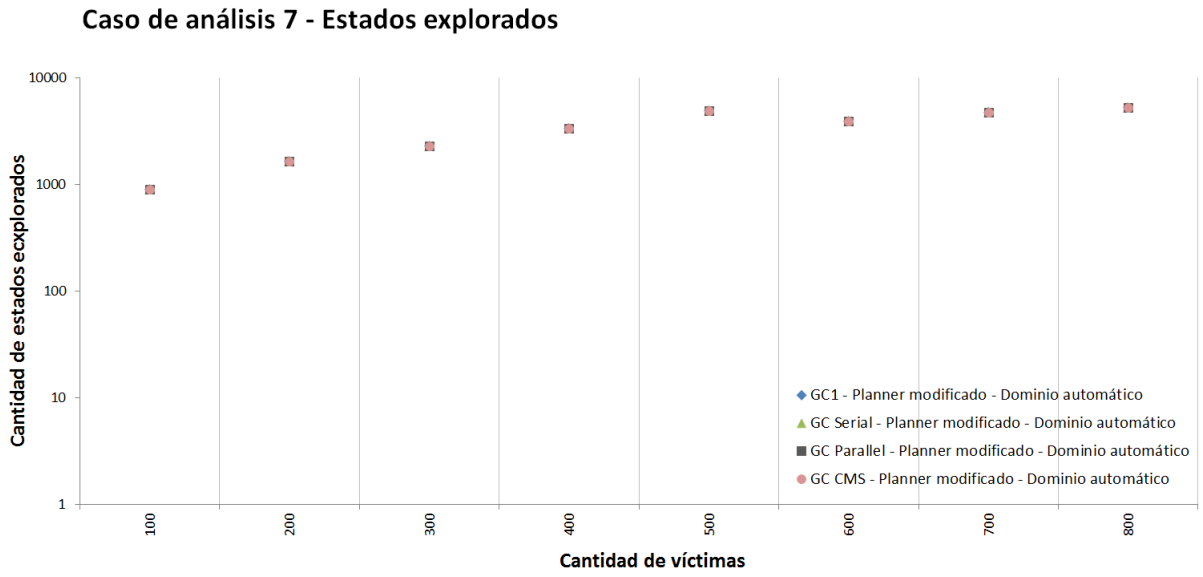

Fig. 4.27: Desempeño del planner modificado con el dominio automático en relación a la cantidad de estados explorados.

4.4.4. Calidad de los planes

Con el propósito de evaluar la calidad de los planes resultantes de la ejecución del planner modificado con el dominio automático, se compararon los resultados obtenidos por el planner con las soluciones confeccionadas manualmente.

Se consideraron estados iniciales con pocos objetos involucrados para volver factible el desarrollo de los planes de forma manual. Por inferencia se supone que las conclusiones obtenidas también se cumplen en casos con mayor cantidad de objetos involucrados.

El propósito de este trabajo es generar planes que asistan a todas las víctimas en el menor tiempo posible, por esta razón, se consideró como métrica de calidad el tiempo que demanda la ejecución de un plan.

Caso de análisis 8: calidad de los planes obtenidos por el planner modificado con el dominio automático aumentando la cantidad de víctimas a evacuar.

Se pusieron a disposición dos vehículos, un *Boat* y un *Truck*, con igual velocidad (20 [Km/h]) para facilitar los cálculos de tiempo en función de las distancias en el cálculo de la solución manual.

Se consideraron dos *Evacuation Area* a distinta distancia, con el propósito de evaluar el comportamiento del planner con respecto a la selección de áreas.

La cantidad de víctimas se aumentó con una tasa de incremento 2, entre 2 y 6 víctimas.

Resultados

En la tabla Tab. 4.1 se detalla para cada estado inicial evaluado, el tiempo que demanda la ejecución de cada plan generado en forma manual, con el planner original y con el planner modificado.

Cantidad de víctimas	**Manual [minutos]**	**Planner original [minutos]**	**Planner modificado [minutos]**
2	14	14	14
4	18	20	20
6	22	24	24

Tab. 4.1: Comparación de la calidad de planes generados manualmente, por el planner modificado y por el planner original.

En la tabla Tab. 4.1 se observa que los planes generados por el planner modificado con el dominio automático son de igual o menor calidad que los desarrollados manualmente.

Los planes generados por el planner modificado con el dominio automático tienen un tiempo de ejecución que no supera el 15 % del tiempo que demora la ejecución de un plan generado manualmente para el mismo estado inicial. Dichos valores son aceptables para su uso en la práctica.

Respecto al planner original, para este caso de análisis no se observan diferencias respecto al modificado. Sin embargo, cabe destacar, que en casos de análisis anteriores, por ejemplo en el Caso de análisis 1 (véase 4.4.1. *Desempeño del planner original vs planner modificado*), se ha observado que el planner modificado tiende a mejorar la calidad de los planes respecto al planner original.

Caso de análisis 9: calidad de los planes obtenidos por el planner modificado con el dominio automático aumentando la cantidad de vehículos.

En este caso, la cantidad de vehículos considerada fue 2 y 4, donde la mitad corresponden a *Boat* y la otra mitad a *Truck*. Como en el caso anterior se consideró una velocidad de 20 [Km/h]. La capacidad de los mismos fue determinada en 1 víctima para evaluar el comportamiento del planner en la selección de vehículos.

Solamente se consideró una *Evacuation Area* y una *Flooded Area* con 2 víctimas, cumpliendo con la simplificación de la cantidad de objetos involucrados.

Resultados

En la tabla Tab. 4.2 se detalla para cada estado inicial evaluado, el tiempo que demanda la ejecución de cada plan generado en forma manual, con el planner original y con el planner modificado.

Cantidad de vehículos	Manual [minutos]	Planner original [minutos]	Planner modificado [minutos]
2	21	29	29
4	11	29	29

Tab. 4.2: Comparación de la calidad de planes generados manualmente, por el planner modificado y por el planner original.

En este caso de análisis, así como en varios casos anteriores, tanto el planner modificado como el original, no envían más de un vehículo de cada tipo para asistir una misma área de emergencia. En los pocos planes generados por el planner original, en los cuales envía más de un vehículo a una misma área, no lo hace en simultáneo.

Sin embargo, los planes realizan otras acciones en paralelo. Lo cual se evidencia, en el *Caso de análisis 2: comparación entre planner original y planner modificado* (véase 4.4.1. *Desempeño del planner original vs planner modificado*) y en el *Caso de análisis 4: comparación entre dominio manual y dominio automático* (véase 4.4.2. *Desempeño del planner modificado con el dominio manual*), donde todos los planes generados ejecutan acciones en paralelo que implican la utilización de más de un vehículo al mismo tiempo, pero a diferentes áreas de emergencia.

En resumen, el planner, tanto el original como el modificado, con cualquiera de los dominios, manual o automático, no selecciona de manera eficiente las acciones que implican la utilización de vehículos para asistir un área. Entonces, ambos planners tienen dificultades con la paralelización de la asistencia.

Este comportamiento responde a una limitación inherente al planner, dado que depende de la selección de acciones aplicables que realiza. Además, como se ha observado, por un lado, es independiente de la modificación implementada, dado que el comportamiento también se observa en el planner original y la implementación no interfiere con la heurística de selección de acciones aplicables utilizada por el planner. Por otro lado, es independiente del dominio ya que en el mismo no se modela ninguna acción o predicado que modifique la selección de recursos.

Dicha dificultad explica los resultados obtenidos en la tabla Tab. 4.2 al aumentar la cantidad de vehículos. La calidad de los planes se degrada considerablemente respecto al plan manual que, a diferencia, considera la evacuación de una misma área con vehículos en simultáneo. No se observa diferencia en los resultados obtenidos con el planner original respecto al modificado dado que, como ya se mencionó, tiene un comportamiento similar respecto a la paralelización. Por lo tanto, la modificación del planner no degrada la calidad de los planes respecto a la calidad de los generados por el planner original.

Se considera que el comportamiento descrito anteriormente también se manifestará en problemas más complejos, por esta razón, se propone como trabajo a futuro, indagar si es posible mejorar la calidad de los planes sin incurrir en un costo computacional prohibitivo.

5. CONCLUSIÓN

En este trabajo se ha demostrado empíricamente que el uso de las técnicas de planning permite construir de forma eficiente planes que tengan como objetivo ubicar en las áreas de evacuación a todas las víctimas que se encuentren en áreas inundadas.

Se puede observar que la técnica utilizada permite independizar el modelo del dominio respecto de la implementación encargada de construir el plan, facilitando la extensibilidad y flexibilidad a otros dominios sin tener que modificar la implementación del planner.

El dominio automático (véase 3.2.2. *Especificación del dominio*) desarrollado permite ser extendido para modelar otros factores dentro del mismo problema o incluso, extenderlo a otros dominios similares de evacuación en situaciones de emergencia.

Se modificó el planner seleccionado (véase 2.4. *Planner seleccionado*) para que administre las acciones repetibles pertenecientes al dominio, lo que permite reducir la cantidad de estados explorados en el espacio de búsqueda. Tal como se observa en los resultados, la disminución de la cantidad de estados explorados, reduce considerablemente el consumo de memoria y tiempo que insume la construcción del plan. Por lo tanto, la modificación realizada en el planner impactó directamente en el desempeño del planner sin afectar su generalidad, permitiendo generar planes para casos más complejos utilizando menos recursos computacionales.

Se desarrolló un segundo dominio, llamado dominio manual (véase 3.3. *Modificación de la especificación del dominio*), en el cual el operador del sistema le debe asignar a cada *Flooded Area* al menos una *Risk Area*. El objetivo de dicha modificación fue reducir aún más el espacio de búsqueda, lo cual impactaría en el desempeño del planner.

Los resultados obtenidos al comparar el desempeño del planner con ambos dominios desarrollados, concluyeron que el dominio automático tiene ventajas por encima del dominio manual, dado que obtiene similar o mejor desempeño y no depende de un operador para la asignación entre áreas.

Se puede asegurar que los planes generados por el planner modificado tienden a demandar menos tiempo de ejecución que los planes generados por el planner original. Es decir, el planner modificado tiende a generar planes de mayor calidad. Sin embargo, a medida que se aumenta la cantidad de recursos, la calidad de los planes se degrada en relación a los planes construidos manualmente. Dicho comportamiento se debe a una limitación técnica encontrada en el planner, por la cual no es capaz de paralelizar la asistencia de forma eficiente.

Como trabajo a futuro se propone, por un lado, indagar si es posible mejorar la calidad de los planes sin incurrir en un costo computacional prohibitivo. En particular, se propone explorar la integración de técnicas de simulación que generen escenarios de emergencia, que permitan evaluar y mejorar la calidad de los planes generados.

Las simulaciones tienen como objetivo, entre otros, ayudar a investigadores y desarrolladores a evaluar el desempeño de modelos en escenarios más complejos que aquellos considerados al momento del diseño de dichos modelos. Esto usualmente permite observar

y comprender fenómenos novedosos, emergentes, muy raramente previsibles al momento del diseño exploratorio de un modelo y su entorno. Cuando los modelos son expresables mediante formalismos, como en este trabajo, se puede tomar ventaja de las técnicas de simulación basadas en modelos formales. Es decir, se puede trabajar sobre traducciones rigurosas entre el modelo formal de partida, como ser un modelo de planning, y un modelo formal para simulación, lo que ofrecería ventajas por ejemplo desde el punto de vista de la robustez, reusabilidad y generalidad de la estrategia.

Por otro lado, se propone a futuro trabajar en la extensión del dominio desarrollado y en la adaptación a otros problemas similares de evacuación, como ser, incendios, terremotos, etcétera. Además, indagar si es posible mejorar el desempeño del planner al modificar los parámetros de la JVM.

Respecto a la extensión del dominio se planea incorporar al modelo otros vehículos como por ejemplo, los anfibios o aéreos. Para lo cual, gracias a la flexibilidad que presenta el modelo desarrollado, se requiere agregar solamente un nuevo objeto de tipo *Vehicle* (véase 3.2.2) y especificar las acciones correspondientes. Además incorporar la administración del combustible, que requiere agregar una función que represente el combustible y una acción que modele la recarga del mismo, considerando el consumo promedio del vehículo.

Por último, sería deseable asegurar y determinar los caminos que deben seguir los vehículos, extendiendo el sistema con un módulo que calcule la posible ruta y el tiempo promedio sobre la geografía real.

6. ANEXO A: DOMINIO CON VARIABLES NUMÉRICAS

A los efectos de construir un plan que cumpla con el objetivo de evacuar a todas las víctimas, es relevante conocer cuántas víctimas deben ser rescatadas y no conocer qué víctimas. Es decir, no es necesario modelar la entidad *Víctima* con atributos que las distingan entre sí, por lo tanto, se podría utilizar una variable numérica que simplifique y modele la realidad.

Considerando dicho análisis, se modeló un dominio que utiliza variables numéricas para representar la cantidad de víctimas en los distintos estados: en cada área, en cada vehículo y que aún no han sido asistidas. El objetivo se cumple cuando la variable que representa la cantidad de víctimas que deben ser asistidas es igual a cero.

Recordemos que la acción *Load* modela el ascenso de víctimas a un vehículo. Con éste modelo, una sola ejecución de *Load*, representa el ascenso de la mayor cantidad de víctimas posible, es decir, se logra la transición inmediata entre el estado, donde las víctimas se encuentran en un área de emergencia, y el estado donde la mayor cantidad de víctimas posible se encuentran dentro de un vehículo. Lo mismo aplica a la acción *Unload* logrando la transición inversa entre estados. Como consecuencia, se reduce la cantidad de acciones del plan, lo cual impactaría en el tiempo y memoria requerido para la creación del mismo.

Los resultados de la implementación evidenciaron que el planner presenta dificultades para manejar precondiciones con variables numéricas. Al realizar un seguimiento del comportamiento de la implementación del planner con este dominio, se observó que la imposibilidad radica cuando el planner relaja el plan (véase 2.4.2. *SAPA*), dado que, cuando selecciona las precondiciones positivas se eliminan aquellas que dependen de variables numéricas. Históricamente, este comportamiento es un obstáculo en la mayoría de los planners porque en las competencias [11] para las cuales se desarrollan, las variables numéricas se utilizan únicamente como criterio de optimización y no, como parte de la lógica del dominio.

En conclusión, la implementación del planner junto con el dominio descripto anteriormente no logra los objetivos esperados.

7. ANEXO B: MANUAL

La herramienta desarrollada permite construir un estado inicial por medio de la interfaz gráfica y generar el plan correspondiente utilizando el planner modificado junto con el dominio automático.

7.0.1. Recursos

Para ejecutar la herramienta se debe tener una carpeta con los siguientes archivos:

- **PlanGenerator.jar** - Ejecutable de la herramienta desarrollada
- **InitialState** - Carpeta donde se guardarán los estados iniciales a resolver.
- **Solution** - Carpeta donde se guardarán las soluciones.
- **AutomaticDomain.pddl** - Dominio que no considera asociación de áreas manual.
- **PlannerModified.jar** - Ejecutable Java del planner.
- **RepeatableActionsList.txt** - Lista de acciones repetibles.

7.0.2. Entorno de ejecución

Se requiere un equipo con las siguientes características:

- RAM $> 4\ GB$
- Java SE versión > 1.7
- Sistema operativo: Windows - Linux - MacOS.
- Se recomienda asignarle a la Java Virtual Machine un límite máximo de memoria mayor a $2\ GB$.

7.0.3. Modo de uso

Pantalla *Start*

Elegir *New initial state* para generar un nuevo estado inicial.

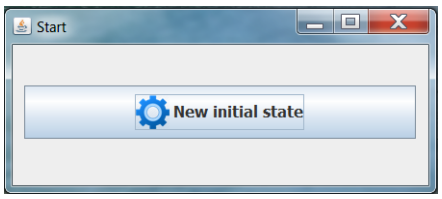

Fig. 7.1: Pantalla *Start*.

Pantalla *Evacuation Areas*

Completar:

- *Number of evacuation areas:* cantidad de áreas de evacuación.
- *Max. occupancy:* capacidad de cada una de esas áreas.
- *Actual occupancy:* cantidad de víctimas que ya se encuentran en cada una de esas áreas.

Luego, presionar *Add areas* para agregar todas las áreas a la grilla. Repetir hasta agregar todas las áreas deseadas.

Para editar un área agregada, modificar el valor del parámetro sobre la grilla. Para eliminar un área agregada, presionar sobre *Delete* en la fila de la grilla correspondiente.

Para continuar, presionar *Next*. Para volver a la pantalla *Start*, presionar *Restart*

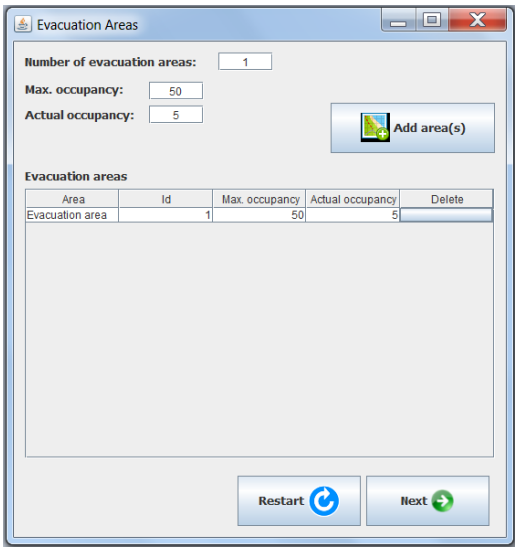

Fig. 7.2: Pantalla *Evacuation Areas*.

Pantalla *Emergency Areas - Flooded Areas*

Completar:

- *Number of flooded areas:* cantidad de áreas inundadas.
- *Victims:* cantidad de víctimas en cada área.

Luego, presionar *Add areas* para agregar todas las áreas a la grilla. Repetir hasta agregar todas las áreas deseadas.

Para editar un área agregada, modificar el valor del parámetro sobre la grilla. Para eliminar un área agregada, presionar sobre *Delete* en la fila de la grilla correspondiente.

Para continuar, presionar *Next*. Para volver a la pantalla anterior, presionar *Back*.

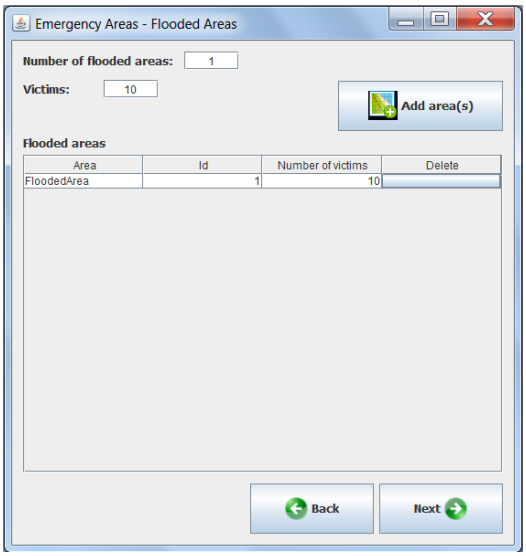

Fig. 7.3: Pantalla *Emergency Areas - Flooded Areas*.

Pantalla *Emergency Areas - Risk Areas*

Completar:

- *Number of risk areas:* cantidad de áreas en riesgo.
- *Victims:* cantidad de víctimas en cada área.

Luego, presionar *Add areas* para agregar todas las áreas a la grilla. Repetir hasta agregar todas las áreas deseadas.

Para editar un área agregada, modificar el valor del parámetro sobre la grilla. Para eliminar un área agregada, presionar sobre *Delete* en la fila de la grilla correspondiente.

Para continuar, presionar *Next*. Para volver a la pantalla anterior, presionar *Back*.

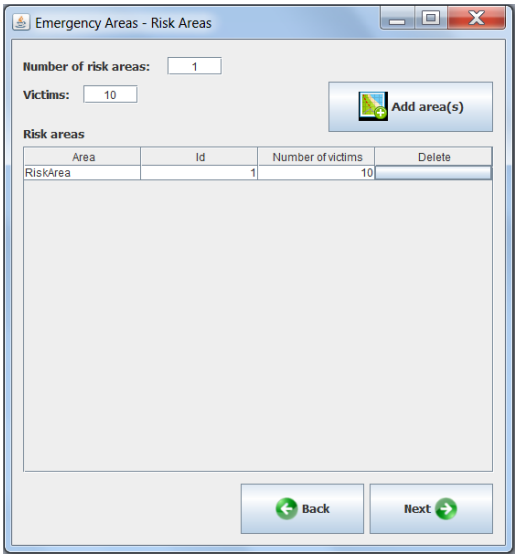

Fig. 7.4: Pantalla *Emergency Areas - Risk Areas*.

Pantalla *Parking Areas*

Completar:

- *Number of parking areas:* cantidad de áreas de estacionamiento.

Luego, presionar *Add areas* para agregar todas las áreas a la grilla. Repetir hasta agregar todas las áreas deseadas.

Para eliminar un área agregada, presionar sobre *Delete* en la fila de la grilla correspondiente.

Para continuar, presionar *Next*. Para volver a la pantalla anterior, presionar *Back*.

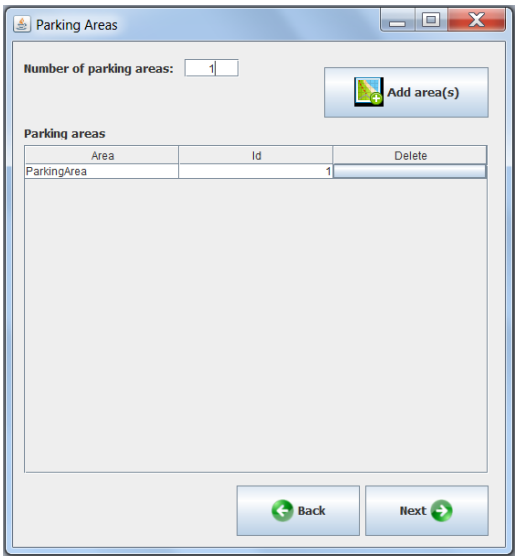

Fig. 7.5: Pantalla *Parking Areas*.

Pantalla *Vehicles*

Completar:

- *Number of vehicles:* cantidad de vehículos.
- *Type:* tipo de vehículo.
- *Location:* área en la que está ubicado el vehículo.
- *Speed:* velocidad del vehículo [Km/h].
- *Max. capacity:* capacidad máxima del vehículo.
- *Victims in:* cantidad de víctimas dentro del vehículo.

Luego, presionar *Add vehicle(s)* para agregar todos los vehículos a la grilla. Repetir hasta agregar todos los vehículos deseados.

Para eliminar un vehículo agregado, presionar sobre *Delete* en la fila de la grilla correspondiente. Para editar un vehículo agregado, presionar sobre *Edit* en la fila de la grilla correspondiente, modificar los parámetros y volver a agregarlo.

Para continuar, presionar *Next*. Para volver a la pantalla anterior, presionar *Back*.

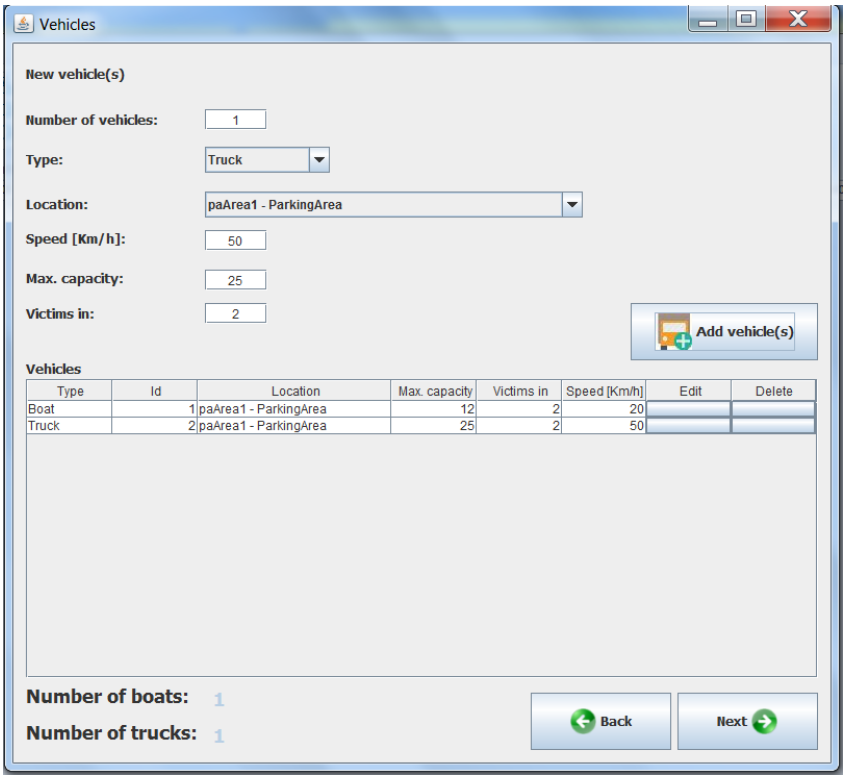

Fig. 7.6: Pantalla *Vehicles*.

Pantalla *Distances between areas*

Completar con las distancias $[Km]$ que hay entre las distintas áreas ingresadas.

Para generar el estado inicial, presionar *Generate Initial State*. Luego, el sistema informará que el estado inicial se ha generado con éxito. Y le preguntará al usuario: ¿Desea generar el plan de los estados iniciales guardados: *.../InitialState?*

Para volver a la pantalla anterior, presionar *Back*.

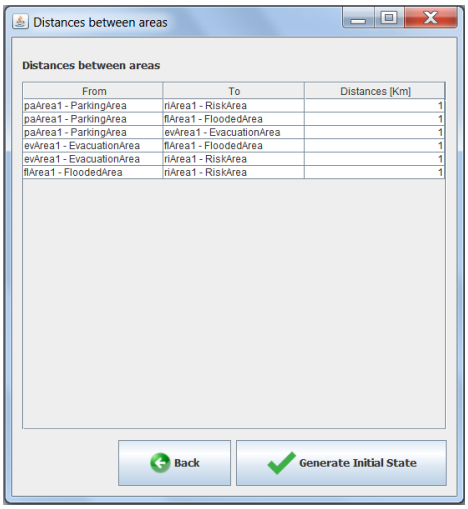

Fig. 7.7: Pantalla *Distances between areas*.

Pantalla *Results*

Se podrá observar la solución de todos los estados iniciales que se encuentren guardados en la carpeta *InitialState*.

Para modificar el estado inicial y replanificar, presionar *Replan* .

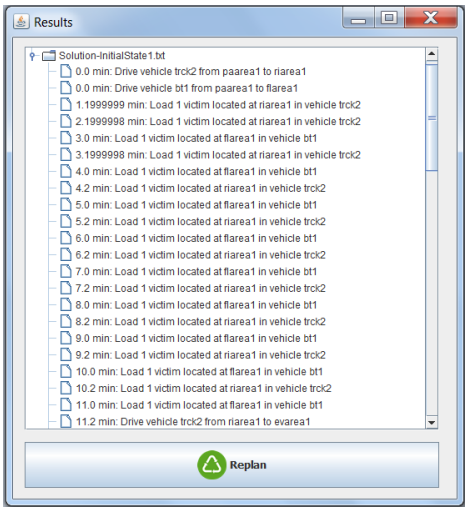

Fig. 7.8: Pantalla *Results*.

8. ANEXO C: EJECUCIÓN DE LOS CASOS DE ANÁLISIS

Para la ejecución de los casos de análisis se debe tener una carpeta con los siguientes archivos:

- **InitialState** - Carpeta donde se guardarán los estados iniciales a resolver.
- **Solution** - Carpeta donde se guardarán las soluciones.
- **ManualDomain.pddl** - Dominio que considera asociación manual entre áreas.
- **AutomaticDomain.pddl** - Dominio que no considera asociación manual entre áreas.
- **PlannerModified.jar** - Ejecutable Java del planner.
- **RepeatableActionsList.txt** - Lista de acciones repetibles.

Desde una terminal ubicada en la carpeta anterior se debe ejecutar alguno de los siguientes comandos para cada caso.

```
>java -Xmx8192m -jar "PlannerModified" "DomainFileName"
        "InitialState/InitialStateFileName"
                -repeatableActions "RepeatableActionsListFileName"
```

donde:

- **-Xmx 8192m**: especifica la máxima memoria asignada a la Java Virtual Machine (JVM), en este caso $8GB$.
- **DomainFileName**: es el dominio que se quiere utilizar:
  - AutomaticDomain.pddl
  - ManualDomain.pddl
- **InitialStateFileName**: es el nombre del archivo del estado inicial que se quiere utilizar. Debe estar guardado en la carpeta *InitialState*.
- **-repeatableActions**: es el *flag* que permite considerar acciones repetibles. Las mismas deben especificarse en el archivo *RepeatableActionsListFileName*.
- **RepeatableActionsListFileName**: es el archivo donde se encuentra la lista de acciones repetibles que se quieren considerar al estar el *flag -repeatableActions* activo.

BIBLIOGRAFíA

[1] P. Russell S. y Norvig. *Artificial intelligence: a modern approach. Second edition.* New Jersey, Estados Unidos: Prentice Hall, 2003.

[2] M. Bacchus F. y Ady. "Planning with resources and concurrency a forward chaining approach". En: *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-2001).* 2001, págs. 417-424.

[3] F. Garrido A. y Barber. "Integrating planning and scheduling". En: *Applied Artificial Intelligence.* 2001, págs. 471-491.

[4] V. Bayram. "Optimization models for large scale network evacuation planning and management: A literature review". En: *Surveys in Operations Research and Management Science.* 2016.

[5] S. Gwynne; E. Galea; M. Owen; P. Lawrence; L. Filippidis. "A review of the methodologies used in the computer simulation of evacuation from the built environment". En: *Building and Environment.* Vol. 34. 1999, págs. 741-749.

[6] Z. Xiongfei; S. Qixin; H. Rachel; R. Bin. "Network Emergency Evacuation Modeling: A Literature Review". En: *Proceedings of the 2010 International Conference on Optoelectronics and Image Processing, ICOIP.* 2010, págs. 30-34.

[7] D. Fox M. y Long. "PDDL2.1: An extension to PDDL for expressing temporal planning domains". En: *Journal of Artificial Intelligence Research.* 2003, págs. 61-124.

[8] Richard E. Fikes y Nils J. Nilsson. "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving". En: *Proceedings of the 2nd International Joint Conference on Artificial Intelligence.* IJCAI'71. 1971, págs. 608-620.

[9] Edwin P. D. Pednault. "ADL: Exploring the Middle Ground Between STRIPS and the Situation Calculus". En: *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning.* 1989, págs. 324-332.

[10] S. Do M. B. y Kambhampati. "Sapa: A Domain-Independent Heuristic Metric Temporal Planner". En: *Proceedings of the Sixth European Conference on Planning.* Vol. September 12–14, 2001. 2001, págs. 57-68.

[11] D. Fox M. y Long. "The 3rd International Planning Competition: Results and Analysis". En: *Journal of Artificial Intelligence Research.* Vol. September 12–14, 2001. 2003, págs. 1-59.

[12] Oracle. *The Garbage-First Garbage Collector.* 2009. URL: `http://www.oracle.com/technetwork/java/javase/tech/g1-intro-jsp-135488.html`.

[13] Oracle. *Java Platform, Standard Edition HotSpot Virtual Machine Garbage Collection Tuning Guide, Release 8.* 2015. URL: `https://docs.oracle.com/javase/8/docs/technotes/guides/vm/gctuning/`.