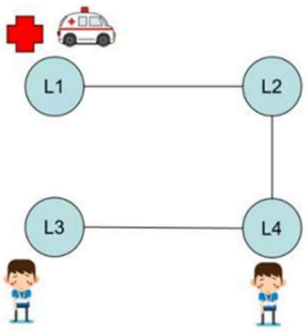
Actividad 3: Planificación en STRIPS / PDDL

Razonamiento y Planificación Automática

Aldo Alberto Bernal Castillo
15/05/2022

Abstract

El presente trabajo muestra la ejecución y entendimiento de 4 planificadores que fueron mostrados en la conferencia ICAPS 2018, dichos planificadores fueron ejecutados con un Domain y un Problem que resuelve lo siguiente:



Una vez comprobada la ejecución de forma correcta, se procedió a realizar modificaciones en el problema con la finalidad de resolver uno nuevo y analizar los resultados.

Introducción

La planificación y programación automatizadas, a veces denominada simplemente planificación de IA, es una rama de la inteligencia artificial que se ocupa de la realización de estrategias o secuencias de acción, normalmente para que las ejecuten agentes inteligentes, robots autónomos y vehículos no tripulados. A diferencia de los problemas clásicos de control y clasificación, las soluciones son complejas y deben descubrirse y optimizarse en un espacio multidimensional. La planificación también está relacionada con la teoría de la decisión.

En entornos conocidos con modelos disponibles, la planificación se puede realizar sin conexión. Las soluciones se pueden encontrar y evaluar antes de la ejecución.

En entornos dinámicamente desconocidos, la estrategia a menudo debe revisarse en línea. Los modelos y las políticas deben adaptarse. Las soluciones suelen recurrir a procesos iterativos de prueba y error comúnmente vistos en inteligencia artificial. Estos incluyen programación dinámica, aprendizaje por refuerzo y optimización combinatoria. Los lenguajes utilizados para describir la planificación y la programación a menudo se denominan lenguajes de acción.

Estos procesos iterativos de prueba y error comúnmente vistos en inteligencia artificial. Estos incluyen programación dinámica, aprendizaje por refuerzo y optimización combinatoria. Los lenguajes utilizados para describir la planificación y la programación a menudo se denominan lenguajes de acción.

En la planificación de IA, los planificadores suelen introducir un modelo de dominio (una descripción de un conjunto de acciones posibles que modelan el dominio), así como el problema específico que se resolverá especificado por el estado inicial y el objetivo, en contraste con aquellos en los que no hay dominio de entrada especificado.

Dichos planificadores se denominan "independientes del dominio" para enfatizar el

hecho de que pueden resolver problemas de planificación de una amplia gama de dominios.

Los ejemplos típicos de dominios son el apilamiento de bloques, la logística, la gestión del flujo de trabajo y la planificación de tareas de robots. Por lo tanto, se puede utilizar un solo planificador independiente del dominio para resolver problemas de planificación en todos estos diversos dominios. Por otro lado, un planificador de rutas es típico de un planificador de dominio específico.

Desarrollo

DecStar

DecStar es una técnica recientemente introducida en la planificación de IA, la cual explota la independencia entre los componentes de una tarea de planificación para reducir el tamaño del espacio-estado de la representación; dividiendo las variables de estado en componentes, de tal manera que la interacción entre estos toma la forma de una **topología estrella**, la búsqueda desacoplada solo se busca sobre secuencias de acción afectando el componente central de la topología, y enumera asignaciones accesibles para cada componente por separado.

Esto puede conducir a una reducción exponencial del tamaño de la búsqueda-espacio. No siempre es fácil encontrar una partición para la tarea planificada dada, aun así, amplía el algoritmo STD (topología estrella) por una opción, que ejecuta una búsqueda estándar siempre que haya fallado en encontrar una partición (buena).

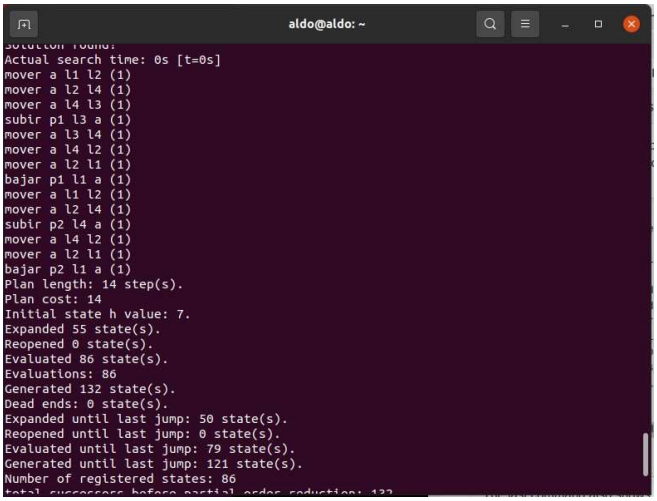


Figura 1.1

En la figura 1.1 podemos comprobar que se corrió el planificador de manera correcta, en 0.0 seg. (posiblemente en milésimas de segundos).

Planificador Delfi1

El planificador Delfi es un planificador de **portafolio online**. Delfi explota técnicas de **Deep learning** (en contraste con otras técnicas existentes) para aprender un modelo que predice cuál de los planificadores en el portafolio puede resolver una tarea planificada determinada, dentro de los limites de tiempo y memoria dada. Delfi usa representaciones gráficas de una tarea planificada, la cual permite explotar herramientas existentes para la convolución de la imagen.

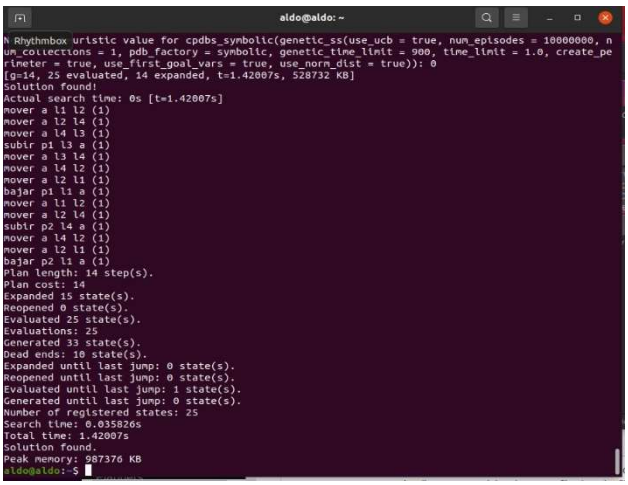


Figura 2.1

En la figura 2.1 podemos comprobar que se corrió el planificador de manera correcta con 14 pasos, en 1.42 seg.

MSP

(Meta-Search-Planner) es un sistema de meta-razonamiento que busca, a través del espacio de planificadores, representaciones y heurísticas problema por problema.

Dado un problema de planificación, MSP se realiza en dos fases: fase de meta-búsqueda y fase de resolución de problemas. La fase de meta-búsqueda es un proceso de búsqueda que genera una combinación adecuada para resolver de manera óptima el problema en específico. La fase de resolución de problema, es básicamente una llamada al planificador seleccionado con otros elementos de la combinación seleccionada por el algoritmo.

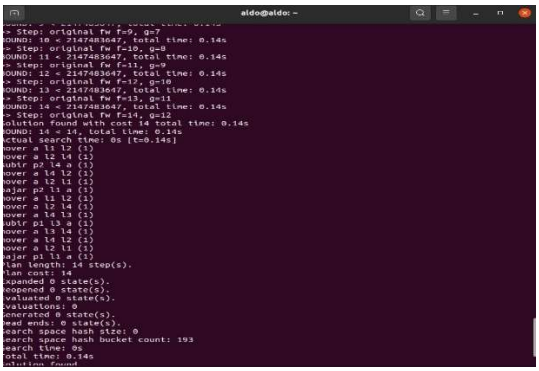


Figura 3.1

En la figura 3.1 podemos comprobar que se corrió el planificador de manera correcta con 14 pasos, en 1.4 seg.

Scorpion

El planificador utiliza A* con la heurística admisible para crear planes óptimos. La heurística en general esta basada en componentes abstractos heurísticos que son combinados mediante la partición de costos saturados

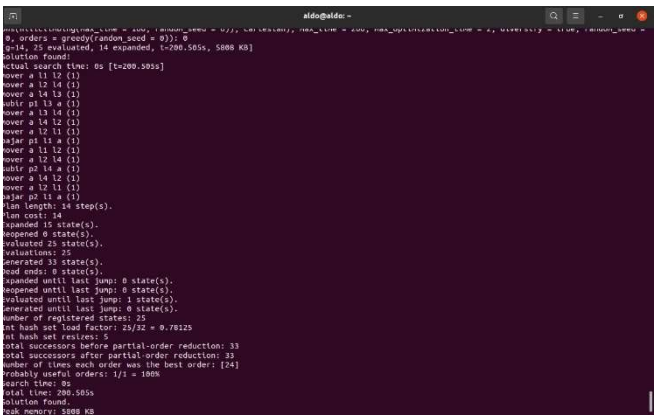
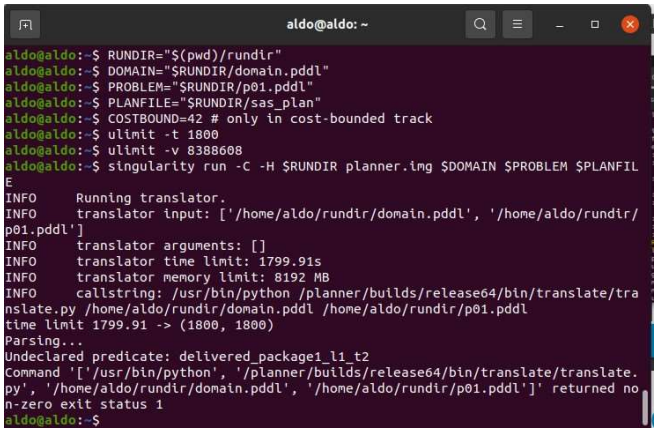


Figura 4.1

En la figura 2.1 podemos comprobar que se corrió el planificador de manera correcta con 14 pasos, en 200.505 seg (el que mayor tiempo ha tardado).



Modificación del problema

Se decidió hacer 2 variantes ligeros del problema para no influenciar su tiempo de ejecución. Analizando la ejecución podremos ver que los 2 mejores planificadores que obtuvieron mejores resultados son **DecStar** y **MSP**.

Aunque Delfi1 tuvo un tiempo de ejecución muy corto (milisegundos), analizando los resultados en consola podremos observar que la utilización de memoria fue demasiada (987376 KB) a comparación de los demás.

Para el nuevo problema se crearon 2 localizaciones extras, además de movimiento en diagonal, para aumentar la dificultad del problema.

Se movieron de localización a los pacientes de L4 y L3 a L5 y L6, mientras que la ambulancia y el hospital se dejaron como estaban.

Modificación 1

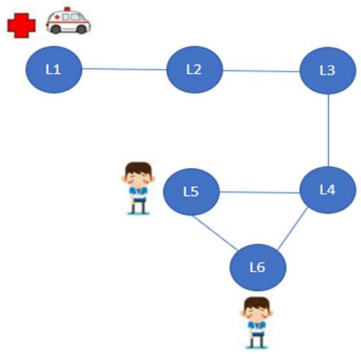


Figura 5.1

En la figura 5.1 podemos ver el problema a resolver con la primera modificación, con 2 nuevas localizaciones (L5 y L6), además de la reubicación de los pacientes (L6 y L5, ahora).

A continuación, **se mostrarán los resultados de los planificadores DecStar y MSP con el archivo Problem.pddl, cambiado acorde a la primera modificacion.**

DecStar

Fue más rápido (0.05 seg) que MSP, la utilización de su **memoria** (5204 KB) para ejecutar el problema en el planificador fue una gran sorpresa, haciéndolo a este el gran ganador. **20 pasos** fueron suficientes para resolver el problema.

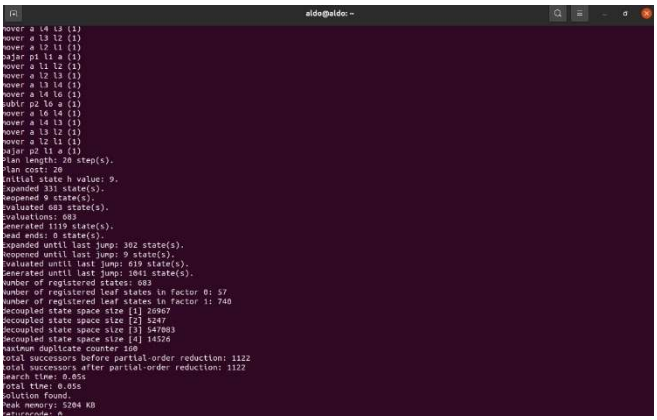


Figura 5.2

En la figura 6.1 se muestra el planificador DecStar ejecutando el problema modificado.

MSP

Este planificador **no fue el más rápido** (0.16 seg) y además fue el que **más memoria usó** (238928 KB), al igual que DecStar, utilizó 20 pasos para la ejecución del problema en el planificador.

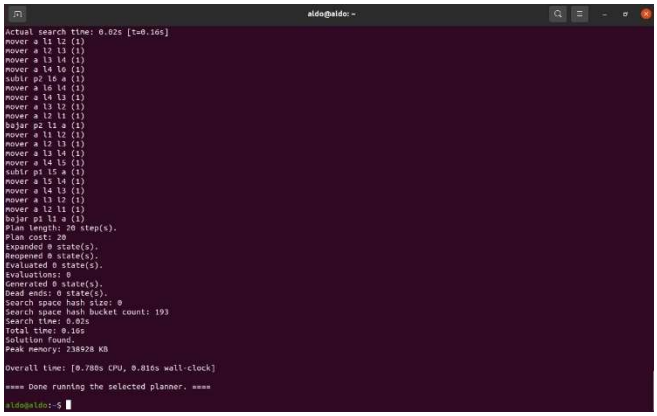


Figura 5.3

En la figura 6.1 se muestra el planificador MSP ejecutando el problema modificado.

Modificación 2

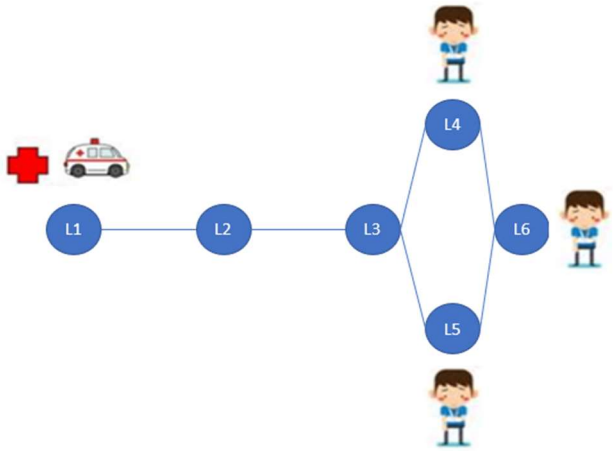


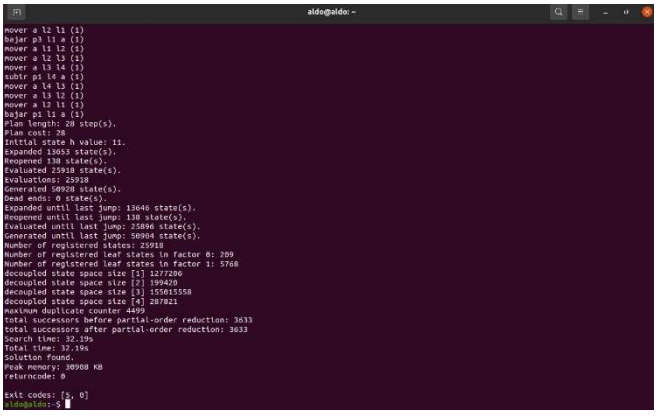
Figura 6.1

En la figura 6.1 podemos ver el problema a resolver con la segunda modificación, con 2 nuevas localizaciones (L5 y L6), se agrega un nuevo paciente (L4) y se reubican los 2 restantes (L6 y L5).

A continuación, **se mostrarán los resultados de los planificadores DecStar y MSP con el archivo Problem.pddl, cambiado acorde a la segunda modificacion.**

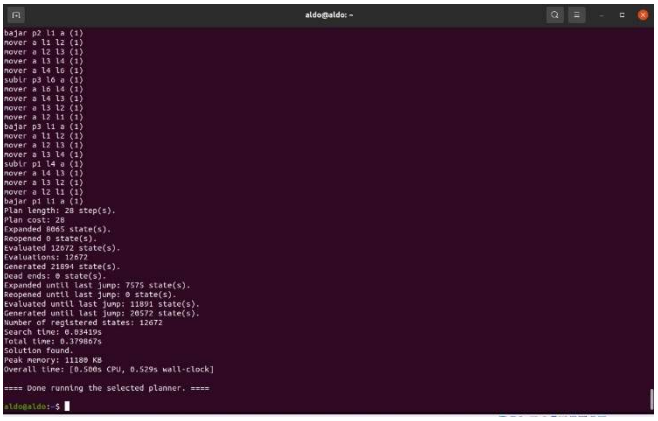
DecStar

A diferencia de la primera modificación, DecStar resulto ser el menos viable. Con un uso de **memoria de 30908 KB,** un tiempo de ejecución de 32.19 seg y un costo de **28 pasos**, queda muy por debajo de MSP.



MSP

Este algoritmo resulto la mejor opción para la segunda modificación, ya que solamente tardo **.500 seg** en ejecutarse, sorprendentemente el uso de memoria de **11180 KB** es mucho más bajo que DecStar , y tardo **28 pasos** el resolver el problema.



Conclusión

Se ejecutaron 2 planificadores con 2 modificaciones en la problemática (2 localizaciones extra , cambio de ubicación en los pacientes y se agregó un nuevo paciente) de manera rápida en Ubuntu 20.04.4 LTS y VirtualBox 6.1.

Cabe resaltar que, aunque el resultado es el mismo (20 pasos y 28 pasos, dependiendo el problema), el algoritmo que usa cada planificador es diferente, DecStar usa topología estrella mientras que MSP usa un proceso de búsqueda que genera una combinación adecuada para resolver de manera óptima el problema en específico.

Como se dijo en el trabajo presentado, **DecStar fue el que mejor obtuvo resultados para la modificación 1 , mientras que MSP fue el mejor que obtuvo resultados para la modificación 2** ,estos resultados fueron determinados por el uso de memoria y el tiempo de ejecución de los 2 algoritmos por separado

Referencias

Tema: Planificación en Inteligencia Artificial. Agentes Planificadores. (s. f.). Facultad:Ingeniería Escuela:Computación Asignatura: Sistemas Expertos e Inteligencia Artificial. Recuperado 13 de marzo de 2022, de https://docplayer.es/91429912-Tema-planificacion-en-inteligencia-artificial-agentes-planificadores.html

https://planning.wiki/ref

http://editor.planning.domains