Un dibujo de un personaje de caricatura

El contenido generado por IA puede ser incorrecto.

PRÁCTICA 1

PARTE 3

Marcos Fuster Peña

Daniel de la Fuente Sánchez

Pablo Tirado Barragán

**Memoria de la Práctica 1 – Planificación Clásica con PDDL**

**1. Introducción**

En esta parte del trabajo, avanzaremos un poco en el concepto de planificación llevado hasta ahora, e implementaremos sistemas de planificaciones con acciones concurrentes mediante el planificador OPTIC.

**Los objetivos específicos de la práctica incluyen:**

* Crear un dominio con soporte de acciones concurrentes y acciones durativas, descartando la aplicación de costes de la parte inmediatamente anterior. Todo mediante la comprensión y soporte de OPTIC
* Analizar el estudio de la concurrencia de la resolución de problemas de planificación por el planificador OPTIC

**1.2 Estructura de la Documentación**

**Esta memoria está estructurada en las siguientes secciones:**

* **Introducción**: Presenta los objetivos de la práctica y una visión general de la documentación.
* **Desarrollo de la Práctica**: Explica el modelado del dominio en PDDL, la generación de problemas en Python y la evaluación de los planificadores.
* **Resultados y Análisis**: Muestra los experimentos realizados y una comparación del rendimiento de distintos planificadores.
* **Conclusiones**: Resume los hallazgos clave y propone mejoras futuras.

**1.3 Contenido del Proyecto**

**El proyecto incluye los siguientes archivos y directorios:**

**PDDL**

* Parte3/pddl/dominio-drones-parte3.pddl – Modelo del dominio de planificación.
* Parte3/pddl/problema-parte3.pddl – Instancia de problema generado como prueba con solo un dron
* Parte3/pddl/problema-varios-drones.pddl – Instancia de problema con varios drones

**Generador de Problemas**

* Parte3/src/generate\_problem.py – Script en Python para la generación de problemas.
* Parte3/src/test\_planiffier.py – Script en Python para generar gráficos de eficiencia de los planificadores dados.

**Planificadores**

* planificadores/optic-clp – planificador con soporte de acciones paralelas sobre el que se desarrolla esta parte de la práctica.

**ProblemasGenerados**

* Almacenamiento provisional de problemas generados por el generador de problemas

**Resultados**

* Almacenamiento de los resultados obtenidos en los ejecutables de generación y testeo

**2. Desarrollo de la Práctica**

**2.1 Modelado del Problema en PDDL**

* **Modificaciones del dominio de la parte 2**:

El dominio de esta tercera parte introduce acciones durativas y fluentes numéricos, lo que permite una representación más realista del comportamiento de los drones en el tiempo, manteniendo la funcionalidad de transporte simultáneo de cajas mediante contenedores introducida en la parte 2.

En primer lugar, destacamos que los tipos definidos no han cambiado respecto a la anterior parte.

Los requisitos del dominio han sido ampliados para incluir:

* **:durative-actions**: Permite definir acciones con duración en el tiempo.
* **:fluents**: Introduce predicados que pueden cambiar de valor.
* **:numeric-fluents**: Permite definir y manipular valores numéricos.

Hemos mantenido los mismos predicados que en la parte 2, pero su utilización dentro de las acciones ha cambiado para adaptarse al modelo temporal, sin embargo mantenerlo significa que problemas de la parte 2 son compatibles con esta parte. De esta forma somos capaces de reutilizar código.

* **Funciones:**

Se mantienen las funciones existentes:

* **(fly-cost ?l1 - localizacion ?l2 - localizacion)**: Representa el costo en términos de recursos para volar entre localizaciones.
* **(total-cost)**: Acumulador que mantiene el costo total del plan generado.

A diferencia de la parte 2, no se han implementado explícitamente las funciones de combustible, ya que esta parte se centra en el aspecto temporal de las acciones.

* **Principales diferencias con el dominio anterior:**

La principal diferencia respecto a la parte 2 es la introducción de **acciones durativas** en lugar de acciones instantáneas. Esto implica:

1. Cada acción tiene ahora una **duración** específica.
2. Las condiciones de las acciones se dividen en tres momentos:
   * **at start**: Condiciones que deben cumplirse al inicio de la acción.
   * **over all**: Condiciones que deben mantenerse durante toda la ejecución de la acción.
   * **at end**: Condiciones que deben cumplirse al final de la acción.
3. Los efectos también se pueden aplicar en diferentes momentos:
   * **at start**: Efectos que se aplican inmediatamente al inicio de la acción.
   * **at end**: Efectos que se aplican una vez la acción ha finalizado.
   * **Acciones durativas implementadas:**

**Acción: coger**

**Descripción**: Un dron recoge un contenedor vacío del depósito.

**Duración**: 1 unidad de tiempo.

**Condiciones**:

* Al inicio (at start):
  + El dron debe estar libre.
  + El contenedor debe estar libre.
  + El contenedor debe estar en la localización.
* Durante toda la acción (over all):
  + El dron debe permanecer en la localización.
  + La localización debe ser un depósito.
  + El número de cajas debe ser cero.

**Efectos**:

* Al inicio (at start):
  + El contenedor deja de estar libre.
  + El dron deja de estar libre.
  + Se establece el contador de cajas en el contenedor a cero.
* Al final (at end):
  + El dron tiene el contenedor.
  + El contenedor ya no está en la localización.

**Acción: dejar**

**Descripción**: Un dron deja un contenedor vacío en el depósito.

**Duración**: 1 unidad de tiempo.

**Condiciones**:

* Al inicio (at start):
  + El dron debe tener el contenedor.
  + El número de cajas debe ser cero.
  + El contenedor debe tener cero cajas.
* Durante toda la acción (over all):
  + El dron debe permanecer en la localización.
  + La localización debe ser un depósito.

**Efectos**:

* Al inicio (at start):
  + El dron deja de tener el contenedor.
* Al final (at end):
  + El dron vuelve a estar libre.
  + El contenedor vuelve a estar libre.
  + El contenedor está en la localización.

**Acción: coger-caja**

**Descripción**: Un dron recoge una caja de una localización.

**Duración**: 1 unidad de tiempo.

**Condiciones**:

* Al inicio (at start):
  + La caja debe estar en la localización.
  + El dron no debe tener caja.
  + La caja debe estar libre.
* Durante toda la acción (over all):
  + El dron debe permanecer en la localización.

**Efectos**:

* Al inicio (at start):
  + La caja ya no está en la localización.
  + La caja ya no está libre.
  + El dron ya no está sin caja.
* Al final (at end):
  + El dron tiene la caja.

**Acción: meter**

**Descripción**: Un dron mete una caja que lleva en su contenedor.

**Duración**: 1 unidad de tiempo.

**Condiciones**:

* Al inicio (at start):
  + El dron debe tener la caja.
  + El contenedor debe tener n1 cajas.
* Durante toda la acción (over all):
  + El dron debe tener el contenedor.
  + El número n1 debe ser el anterior a n2.

**Efectos**:

* Al inicio (at start):
  + La caja entra en el contenedor.
  + El contenedor ya no tiene n1 cajas.
* Al final (at end):
  + El dron ya no tiene la caja.
  + El dron vuelve a estar sin caja.
  + El contenedor ahora tiene n2 cajas.

**Acción: sacar**

**Descripción**: Un dron saca una caja del contenedor.

**Duración**: 1 unidad de tiempo.

**Condiciones**:

* Al inicio (at start):
  + La caja debe estar en el contenedor.
  + El dron no debe tener caja.
  + El contenedor debe tener n2 cajas.
* Durante toda la acción (over all):
  + El dron debe tener el contenedor.
  + El número n1 debe ser el anterior a n2.

**Efectos**:

* Al inicio (at start):
  + La caja ya no está en el contenedor.
  + El contenedor ya no tiene n2 cajas.
* Al final (at end):
  + El dron tiene la caja.
  + El dron ya no está sin caja.
  + El contenedor ahora tiene n1 cajas.

**Acción: volar**

**Descripción**: El dron se desplaza de una localización a otra.

**Duración**: Calculada como 0.001 + el costo de vuelo entre las localizaciones.

**Condiciones**:

* Al inicio (at start):
  + El dron debe estar en la localización de origen.
* Durante toda la acción (over all):
  + El dron no debe llevar caja directamente.

**Efectos**:

* Al inicio (at start):
  + El dron ya no está en la localización de origen.
* Al final (at end):
  + El dron está en la nueva localización.
  + Se incrementa el costo total del plan en función del vuelo.

**Acción: entregar**

**Descripción**: Un dron entrega una caja a una persona que necesita su contenido. **Duración**: 1 unidad de tiempo. **Condiciones**:

* Al inicio (at start):
  + El dron debe tener la caja.
  + La persona debe necesitar el contenido.
* Durante toda la acción (over all):
  + El dron debe estar en la misma localización que la persona.
  + La persona debe estar en la localización.
  + La caja debe contener el contenido que necesita la persona.

**Efectos**:

* Al inicio (at start):
  + La persona obtiene el contenido.
  + La persona ya no necesita el contenido.
* Al final (at end):
  + El dron ya no tiene la caja.
  + El dron vuelve a estar sin caja.

**2.2 Generador de Problemas en Python**

Hemos reutilizado el generador de problemas de la parte anterior, ya que, como hemos explicado anteriormente, los problemas de la parte 2 son compatibles con esta.

* 1. **Ejemplo de problema:**

En este ejempl.o utilizaremos el fichero: problema-varios-drones.pddl, que consiste de 2 drones, 4 personas, 4 localizaciones y 7 cajas en total.

Vamos a observar la planificación que propone optic. Aunque el planificaodr es capaz de generar varias soluciones optimizando las anteriores, como el tiempo de ejecución es muy elevado, nos quedaremos con la primera de todas.

Observamos que hay un paralelismo significativo en la ejecución de tareas entre los dos drones (d1 y d2), realizando diferentes tareas en distintos momentos:

Texto

El contenido generado por IA puede ser incorrecto.**Inicio simultáneo (tiempo 0.000)**:

* d1 realiza "coger-caja" y "coger"
* d2 comienza a volar hacia l4

**Operaciones en paralelo**:

* Mientras d1 completa su primera misión de entrega (0.000-23.003), d2 está realizando vuelos (0.000-35.004)
* Cuando d1 inicia su segunda misión (43.004), d2 está volando hacia l1
* El sistema maximiza la eficiencia haciendo que ambos drones trabajen constantemente

**Independencia operativa**:

* Cada drone tiene su propia secuencia de tareas
* No necesitan esperar a que el otro termine para continuar sus operaciones
* Los recursos compartidos (como el depósito) son utilizados por ambos en diferentes momentos

**Optimización de rutas**:

* El planificador asigna tareas evitando conflictos de recursos
* Se prioriza mantener ambos drones en movimiento constante

Finalmente podemos observar que, aunque logra obtener todas las goals, hay parte ineficiente en este programa, puesto que el dron 2 hace vuelos inútiles, como al principio, desplazándose a l2 para posteriormente volver al depósito. Así pues, observamos que d1 entrega 3 cajas y d2 solo entrega 1. Por lo que, concluimos que no es una solución óptima pero válida.

**3.** **Evaluación del Planificador**

Para evaluar el planificador se abordará un estudio de escalabilidad y rendimiento, centrado en determinar el tamaño máximo de problema que OPTIC es capaz de resolver en un tiempo máximo de 1 minuto. Sobre el problema mayor resuelto dentro de ese límite temporal, se investigará el impacto en el rendimiento del planificador al variar individualmente diferentes parámetros del problema, tales como el número de drones, la cantidad de cajas y el número de personas. Se realizarán pruebas manteniendo constante el número de drones e incrementando la magnitud del problema, así como pruebas en las que se mantenga la magnitud del problema y se incremente la cantidad de drones. Finalmente, se analizarán tanto el tiempo de ejecución como el coste asociado a los planes generados por el planificador, proporcionando así una visión completa del comportamiento del sistema bajo distintas configuraciones.

* **Aumento de número de drones**

Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.En el análisis del efecto del aumento en el número de drones se ha observado lo siguiente: con una configuración fija de 10 personas, 10 ubicaciones y 12 cajas, el planificador OPTIC logra generar planes en tiempos razonables para uno y dos drones. Sin embargo, al incrementar a tres drones, el tiempo de ejecución se incrementa significativamente, superando el límite de un minuto.

Aunque disponer de más drones reduce el coste de los planes, este beneficio se ve contrarrestado por el notable aumento en el tiempo de resolución. En consecuencia, concluimos que, en el escenario evaluado, el planificador OPTIC no es capaz de resolver problemas con tres drones en menos de un minuto.

* **Aumento del tamaño del problema**

En los experimentos realizados manteniendo un número fijo de drones y aumentando la magnitud del problema se han observado tendencias interesantes tanto en el coste como en el tiempo de ejecución:

* **Caso de 1 dron:**  
  Se ha podido observar que el tamaño máximo del problema alcanzable es de 36, con costes elevados que oscilan entre 1500 y 2500. El tiempo de ejecución crece de manera exponencial a partir de una magnitud de 20. Por otro lado, el coste presenta un comportamiento en forma de "dientes de sierra", lo que indica que, en ocasiones, a pesar de aumentar la magnitud del problema, es posible encontrar soluciones con un coste inferior.

Gráfico, Gráfico de dispersión

El contenido generado por IA puede ser incorrecto.Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.

* **Caso de 2 drones:**  
  En este escenario, el tamaño máximo del problema se reduce a menos de 14. Los costes asociados son muy similares a los del caso de 1 dron, mientras que el tiempo de ejecución presenta un crecimiento más lineal. Sin embargo, se aprecia que la ejecución con 2 drones es más demandante, evidenciando que el problema alcanza su límite en una magnitud inferior en comparación con el caso de 1 dron.

Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.Gráfico, Gráfico de líneas

El contenido generado por IA puede ser incorrecto.

En conclusión, para este planificador OPTIC, al mantener un número fijo de drones, el aumento en la magnitud del problema tiene un impacto significativo en el tiempo de ejecución y, en menor medida, en el coste de la solución. Con un solo dron se puede gestionar un problema de mayor tamaño, aunque a costa de un incremento exponencial en el tiempo de ejecución y variaciones en el coste, mientras que con dos drones, aunque el coste se mantiene similar, la capacidad de escalabilidad se reduce considerablemente, limitando el tamaño del problema resoluble en menos de 1 minuto.

**4. Conclusiones**

La extensión del dominio de gestión de emergencias mediante acciones durativas ha permitido la generación de planes concurrentes en el que la duración de las acciones, determinada por la función “fly-cost”, sustituye el coste clásico de las mismas. Esta transformación facilitó la integración de múltiples drones y transportadores en un mismo escenario, validándose la capacidad del planificador OPTIC para generar soluciones que cumplen los requisitos definidos, aunque resaltando ciertos límites en cuanto a la escalabilidad en problemas complejos.

Los experimentos realizados muestran que, con un escenario fijo de 10 personas, 10 ubicaciones y 12 cajas, OPTIC genera planes en tiempos razonables para uno o dos drones, pero al incluir un tercer dron el tiempo de ejecución se incrementa de forma significativa, superando el minuto. Además, se ha observado que al mantener un número fijo de drones y aumentar la magnitud del problema, los casos con un solo dron alcanzan un tamaño mayor (hasta 36), aunque con un crecimiento exponencial en el tiempo a partir de magnitudes intermedias, mientras que con dos drones el tamaño máximo resoluble se reduce drásticamente (menos de 14). En ambos casos, el coste de las soluciones sigue un patrón irregular, evidenciando que, a veces, problemas de mayor magnitud pueden producir costes relativamente menores.