Trabajo Práctico:

Dominio Puerto

Planificación Inteligente

Máster en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

2021/22

Daniel Parres Montoya

Alejandro Granados Bañuls

Índice:

1. Introducción

Esta es la memoria del trabajo práctico de la asignatura de Planificación Inteligente donde se trabaja con el dominio Puerto.

En este dominio se presenta el puerto de una ciudad, donde se tienen varios muelles de descarga. En cada muelle existe una o varias grúas y diferentes pilas donde se van poniendo contenedores. También existen varias cintas que comunican ambos muelles.

El objetivo del problema es dejar **disponibles** unos contenedores llamados **objetivo** en el muelle 1 para ser cogidos por una grúa cuando llegue el barco. Los contenedores objetivo son representados de color verde y los no objetivo de color blanco. Por lo que los contenedores verdes deben estar en el muelle 1 sin tener ningún contenedor no objetivo encima. En esta práctica se proponen 5 ejercicios con diferentes configuraciones para el dominio.

Restricciones del problema original a tener en cuenta:

- Las pilas tienen una altura máxima global.
- Únicamente existe una grúa por muelle.
- Existe una cinta con dos carriles (uno para cada dirección) entre el muelle 1 y 2.
- Para que un contenedor esté disponible debe de estar en el top de la pila o ser verde y estar debajo de otro verde.

En el Ejercicio 1 se debe de desarrollar el **dominio proposicional**, resolver el problema que se propone en el enunciado de la práctica, ejecutar los planificadores FF, LPG y LPG con la opción -*timesteps* y OPTIC, y finalmente se propone que se especifiquen otras instancias de problema cambiando la situación inicial, final y modificando ciertas restricciones.

Por otro lado, en el Ejercicio 2 se propone desarrollar el **dominio temporal**, definiendo peso a los contenedores, establecimiento tiempo de apilar/desapilar, especificar distancia de la cinta transportadora y otras modificaciones.

En el Ejercicio 3 se diseña el dominio con **recursos numéricos**, y se proponen dos versiones del dominio, en la primera el recurso no es renovable, y en la segunda si es renovable.

En el Ejercicio 4 se hace un desarrollo parcial de un **árbol POP**, sin tiempo ni recursos, de un problema específico.

Finalmente en el Ejercicio 5, partiendo del mismo problema del Ejercicio 4, se construye el **grafo de planificación relajado** y se contestan a unas preguntas.

2. Dominio proposicional

En este apartado se presenta la solución propuesta para el Ejercicio 1 del dominio.

2.1. Tipos

Antes de presentar los predicados y acciones definidas en el dominio es menester comentar los tipos de objeto que se utilizan para resolver el problema:

- Cinta
- Muelle
- Grúa
- Pila
- Altura
- Contenedor

Las cintas tienen dos carriles, pero para resolver el problema se ha decidido definir cada carril como una cinta en sí misma.

2.2. Predicados

- ***(verde ?c - contenedor)***: Indica que un contenedor es objetivo.
- ***(no-verde ?x - (either pila contenedor))***: Indica que un contenedor no es objetivo. Se permite el tipo pila en este predicado porque de esta forma las acciones para dejar un contenedor en pila se simplifican a 2 (ya que de otra forma habría que definir 3 acciones: dejar contenedor en pila vacía, dejar contenedor sobre contenedor blanco, dejar contenedor sobre contenedor verde).
- ***(libre ?x - (either cinta grua))***: Indica si una grúa o cinta están libres, es decir, no hay ningún contenedor ocupando la grúa o cinta.
- ***(encima ?c - contenedor ?x - (either pila contenedor))***: Indica el orden de los contenedores en una pila.
- ***(ubicado-en ?x1 - (either grua pila contenedor) ?x2 - (either cinta muelle grua pila))***: Indica la ubicación de las grúas, pilas y contenedores en el problema.
- ***(top ?x - (either pila contenedor) ?p2 - pila)***: Indica cuál es el contenedor *top* de la pila, ya que si el primer parámetro es un contenedor quiere decir que dicho contenedor es el top de una pila, mientras que si el top es la propia pila indica que está vacía.

- ***(disponible ?x - (either pila contenedor))***: Indica cuando un contenedor se encuentra disponible para ser cogido por la grúa. Cuando se coge un contenedor de una pila, el contenedor de abajo debe pasar a estar disponible. Pero cuando en una pila únicamente hay un contenedor, se pone la pila como disponible, marcando así que la pila está vacía y permitiendo que exista una única acción para coger contenedor. Por esa razón también se aplica a pilas.
- ***(conecta-a ?ct - cinta ?m_o - muelle ?m_d - muelle)***: Indica la dirección de una cinta, es decir, el muelle origen y el muelle destino hacia donde se mueve la cinta.
- ***(next ?t1 - altura ?t2 - altura)***: Indica el orden de las alturas. Como en este primer dominio no es posible usar recursos numéricos se deben definir las alturas de manera booleana, por lo que esta es la forma de definir el orden de las alturas. Por ejemplo, (next n0 n1) significa que la altura n0 es menor a la altura n1.
- ***(altura ?p - pila ?t - altura)***: Indica la altura actual de la pila.

2.3. Acciones

- ***(action coger-pila)***: Permite a la grúa coger un contenedor de una pila, haciendo que el contenedor que se encontraba debajo quede disponible. En caso de que una pila contenga únicamente un contenedor, deja disponible la pila, de esta forma se informa de que la pila está vacía.
- ***(action dejar-blanco-pila)***: Permite a la grúa dejar cualquier contenedor sobre un contenedor blanco o una pila vacía.
- ***(action dejar-verde-verde)***: Permite a la grúa dejar un contenedor verde sobre otro verde. La diferencia con la acción *dejar-blanco-pila* es que cuando se deja un verde sobre otro verde, el contenedor que queda abajo se queda disponible.
- ***(action dejar-cinta)***: Permite a la grúa dejar un contenedor en una cinta para llevar el contenedor del muelle origen al muelle destino. Dentro de esta acción se realiza también la acción de mover hacia el muelle destino.
- ***(action coger-cinta)***: esta acción permite a la grúa coger un contenedor de una cinta que ha llegado al muelle destino.

2.4. Problema propuesto en el enunciado

A continuación se presenta el problema propuesto por el enunciado donde se tiene 3 contenedores verdes y se tiene como objetivo ubicar los contenedores verdes en el muelle 1 y que estén disponibles.

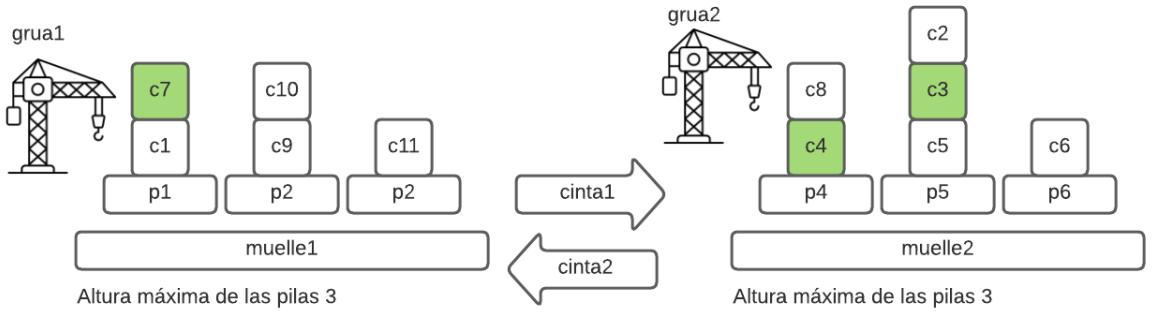

Figura 1. Problema propuesto Ejercicio 1.

La instancia del problema se encuentra en *src/ex1/problem1.pddl* y se usará el dominio definido en *src/ex1/domain.pddl*. Se podrá ejecutar usando el archivo Makefile con el comando *make [planificador] EX=1 PROBLEM=1*.

Se han definido 2 grúas, 6 pilas, 2 muelles, 2 cintas, 11 contenedores y 4 tipos de alturas. Se han inicializado los predicados, ubicando cada elemento en su respectivo muelle y cada contenedor en su correspondiente pila. También se ha instanciado el orden de cada contenedor en su pila y se han declarado los contenedores top de cada pila. Las grúas y las cintas están libres y también se debe definir la relación de alturas de las pilas. Además se debe informar de qué contenedores son verdes, cuáles son blancos y cuales están disponibles.

El objetivo del problema propuesto es que los contenedores 3, 4 y 7 se encuentren ubicados en el muelle 1 y además estén disponibles.

2.5. Ejecución planificadores en el problema propuesto

En este apartado se ejecutan los planificadores FF, LPG, LPG con la opción *-timesteps* y OPTIC sobre el problema propuesto de la Figura 1, que en nuestra entrega corresponde con *src/ex1/problem1.pddl*.

Después de varias ejecuciones, los mejores resultados provienen del planificador FF, pues proporcionaba planes con pocas acciones a diferencia de OPTIC y LPG. En tiempo FF también superaba a los otros planificadores, pues tardaba pocos segundos en resolver el problema. En cambio, OPTIC y LPG habían momentos que tardaban mucho en acabar una ejecución. Es menester recalcar que OPTIC tardaba

en acabar una ejecución porque invertía mucho tiempo en buscar soluciones mejores a la que había encontrado, es decir, que aunque encontraba una solución medianamente rápida no acababa la ejecución.

2.6. Otras instancias de problema

En este apartado se especifican otras instancias del problema tratando de cumplir todos los puntos que se proponen en el enunciado de la práctica de forma incremental:

- Especificar alturas distintas de pilas para cada muelle: Problema 2.
- Variar el número de pilas e incluir nuevos contenedores: Problema 3.
- Añadir un tercer muelle: Problema 4.
- Añadir una segunda grúa en un mismo muelle: Problema 5
- Problema complicado propuesto por la profesora Eva Onaindia: Problema Eva.

Cabe destacar que los predicados y acciones no se han modificado respecto a los introducidos en los apartados Predicados y Acciones. Esto es debido a que se ha tratado de hacer un diseño lo más general posible tratando de dotar flexibilidad al dominio.

En todos los problemas de este apartado el objetivo es conseguir que los contenedores verdes estén disponibles y en el muelle 1.

2.6.1. Problema 2

En este problema se modifica la altura máxima de las pilas de los muelles, siendo la altura máxima de las pilas 1 en el muelle 1 y 3 en el muelle 2. Este problema se encuentra en *src/ex1/problem2.pddl*.

Para esto se han definido alturas lógicas por muelle (*muelle1_n0*, *muelle1_n1*, *muelle2_n0…*) con su correspondiente orden.

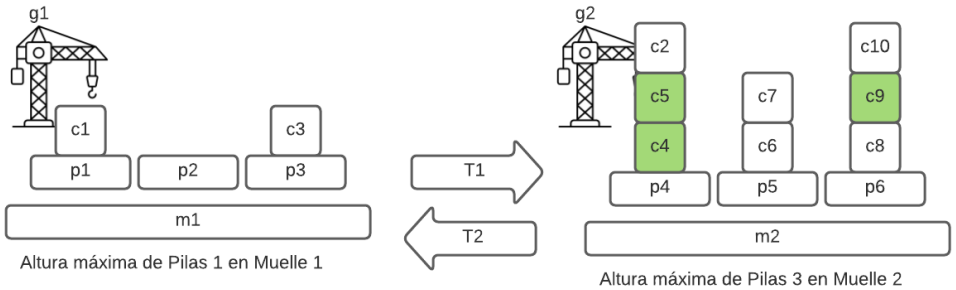

Figura 2. Problema 2 Ejercicio 1.

2.6.2. Problema 3

En este problema se varía el número de pilas y se incluyen nuevos contenedores. La ubicación del problema 3 es *src/ex1/problem3pddl*.

Para definir dicho problema simplemente se han tenido que definir más contenedores y más pilas, además se ha añadido una altura máxima de 2 en las pilas del muelle 1 y de 3 en el muelle 2.

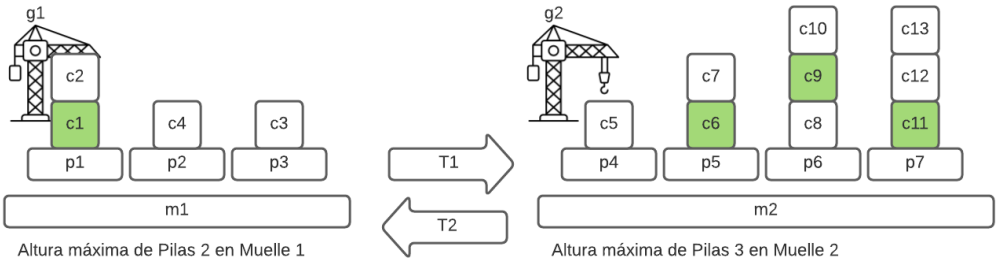

Figura 3. Problema 3 Ejercicio 1

2.6.3. Problema 4

El Problema 4 añade un tercer muelle y se encuentra en *src/ex1/problem4.pddl*.

No se ha tenido que definir ningún predicado nuevo, simplemente se ha añadido el tercer muelle, se han declarado los contenedores, pilas y la grúa del tercer muelle y la cinta que permite la comunicación con el muelle 1.

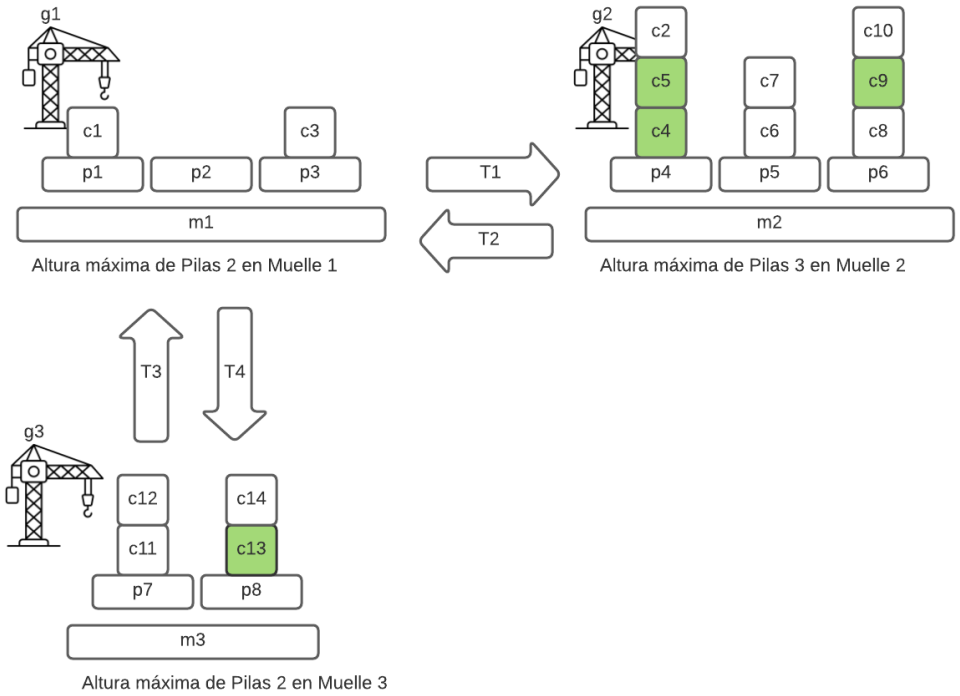

Figura 4. Problema 4 Ejercicio 1

2.6.4. Problema 5

El Problema 5 es igual que el anterior pero con dos grúas en el muelle 1, se encuentra en *src/ex1/problem5.pddl*.

Se considera interesante probar si puede existir alguna diferencia entre usar 1 o 2 grúas en el Muelle 1. Después de ejecutar el problema tanto con *LPG* como con *OPTIC* los resultados son iguales, tanto en acciones como en duración. El punto es que en el problema 5 se usan todas las grúas que se definen.

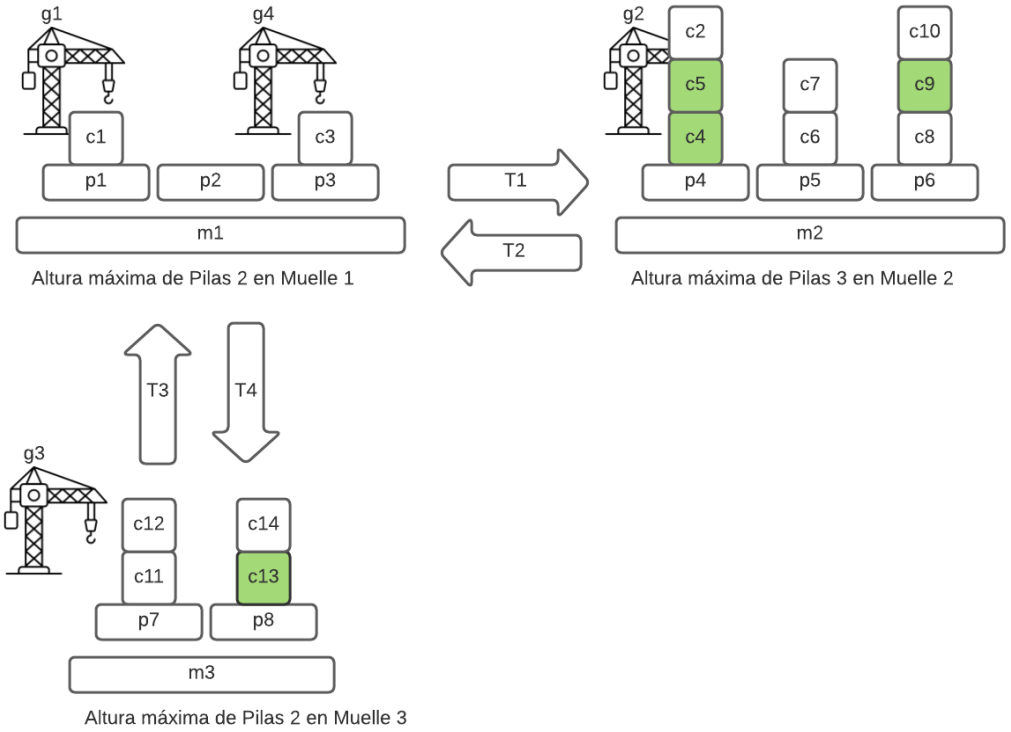

Figura 5. Problema 5 Ejercicio 1

2.6.5. Problema Eva

Finalmente, como reto la profesora Eva Onaindia nos propuso un problema complicado que hemos decidido incorporar a la memoria, este se encuentra en *src/ex1/problemeva.pddl*.

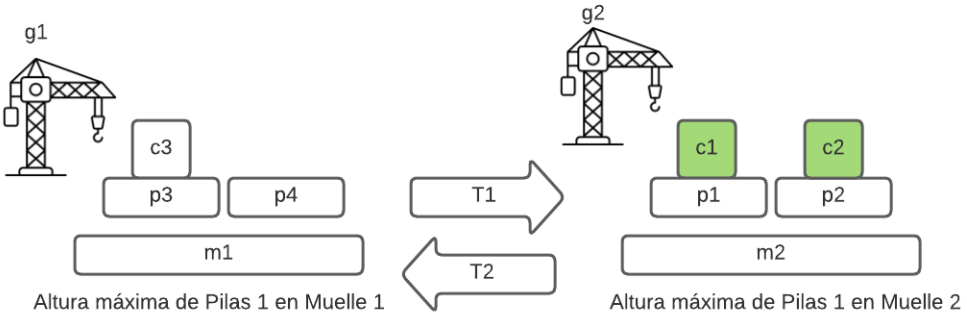

Figura 6. Problema Eva Ejercicio 1

El dominio definido es capaz de resolver tanto el problema propuesto en el enunciado del trabajo como los diferentes problemas que se han definido en el apartado 2.4. Otras instancias de problema, por lo que se considera que se ha definido un dominio general y flexible.

2.7. Resultados

Antes de continuar es menester recalcar que para medir el tiempo de ejecución de los planificadores se ha tenido en cuenta el tiempo que tarda el planificador desde que se lanza el comando hasta que se acaba el proceso.

Una vez ejecutados todos los problemas con los planificadores lpg, lpg con timesteps, optic y ff se han podido obtener una serie de observaciones. El planificador ff es el que obtiene resultados más rápidos y bastante optimizados en comparación con los demás. Esto se puede deber al hecho de que los problemas definidos son casi prácticamente secuenciales.

Respecto a lpg, en el primer caso los primeros planes encontrados no eran de muy buena calidad (en algunos casos la duración de los planes era 100 veces mayor que en el caso de ff) pero si se aumentaba la opción *-n* a 3 los planes encontrados eran de muy buena calidad, igualando varias veces a ff. En el caso de optic las soluciones eran siempre de muy buena calidad, igualando a ff.

Tanto en lpg como en optic los tiempos de ejecución eran muy largos, aunque ambos encontraban soluciones relativamente rápido, pero el planificador se mantenía ejecutando mientras buscaba planes mejores.

Finalmente, hablando de paralelización, optic y lpg devuelven acciones que se podían realizar en paralelo, así que aunque el número de acciones generadas por estos eran mayores o iguales a ff, en muchos casos la duración del plan era menor.

3. Dominio temporal

Para el dominio temporal se ha decidido añadir un recurso numérico llamado *peso* que como su nombre indica es el peso del contenedor y otro llamado *altura*, que indica la altura en formato entero. Hay varios elementos del dominio que están relacionados con esta nueva altura numérica:

- Pila: Medir la altura actual de la pila
- Grúa: Medir la altura de la grúa
- Cinta: Medir la altura a la que está la cinta.

Esto se ha realizado así porque las duraciones de las acciones dependían de la altura de la pila y del peso de las cajas. Con la altura para cada elemento se puede definir una fórmula para calcular la duración de las acciones relacionadas con el movimiento de la grúa:

$$duracion = peso \cdot (alturaGrua - altura)$$

Figura 7: Fórmula duración acciones de la grúa

Se puede ver en la figura 7 que la fórmula sirve tanto para coger como dejar un contenedor de la pila o de la cinta, ya que las dos tienen una altura definida.

También se añadió una longitud para la cinta, así como una velocidad. Esto se hizo para poder medir la duración de la acción de mover un contenedor por la cinta. Con lo anterior se destaca que se tuvo que añadir una nueva acción con la que se moviera un contenedor por la cinta de un muelle origen a uno destino y eliminar esta funcionalidad de la acción *dejar-cinta*. La nueva acción sería la siguiente:

- ***(action mover-cinta)***: Permite a una cinta mover un contenedor de un muelle origen a uno destino.

La duración de la acción de mover cinta se calcula dividiendo la longitud de la cinta por la velocidad de la cinta.

3.1. Redefinición dominio y problema 1

Para la nueva definición del dominio, especificada en *src/ex2/domain.pddl* se ha tenido en cuenta las funcionalidades que aportan los planificadores respecto a los dominios temporales, así que se a la hora de definir las declaraciones *at start, over all* y *at end* se pensó permitir lo máximo posible la paralelización de las acciones, aunque el problema es muy secuencial.

Es menester recalcar que hay acciones que tienes varias fases, como la acción de coger un contenedor de una pila, en el que se puede destacar que hay tres fases:

1. Bajar gancho grúa
2. Enganchar contenedor
3. Subir gancho grúa con el contenedor

Hay efectos y precondiciones que ocurren en fases intermedias de las acciones, así que se decidió que todo lo que ocurra a mitad de una acción (es decir, que ocurra en una fase intermedia) será añadido con la declaración *at start*. Los efectos que sean objetivos o precondiciones que ocurran al final tendrán la declaración *at end*. El resto tendrán la declaración *over-all*.

Con lo anterior especificado, la definición en el dominio temporal del problema 1, ubicado en *src/ex2/problem1.pddl* no cambiaba mucho, simplemente se tuvieron que añadir los recursos definidos (altura, longitud, velocidad y peso).

3.2. Ejecución problema 1

Una vez ejecutado el problema tanto con *OPTIC* como con *LPG* con *n=3* se aprecia que hay acciones que se realizan simultáneamente (*mover-cinta* y *coger-pila* en ambos casos). Esos resultados muestran que se puede optimizar los recursos usando los dominios temporales y minimizar, por ejemplo, el tiempo invertido en el plan.

Respecto a los resultados de los planificadores, si se usa la opción *n=1* de *LPG*, ambos planificadores proporcionan un resultado muy parecido, pero si aumentamos en el número de *n* a 3, *lpg* proporciona un plan mejor respecto a tiempo que *OPTIC*, pero con un coste temporal en encontrarlo mayor.

3.3. Ejecución otras instancias

Después de ejecutar los problemas definidos en el punto 2.6 se puede apreciar que las acciones con las que más se paraleliza son con las de *mover-cinta*, ya que no interviene ninguna grúa, por lo que se pueden realizar en paralelo.

También se aprecia que las ejecuciones en *LPG* son más lentas, aunque acabe encontrando planes buenos en comparación con *OPTIC* (siempre que el parámetro *n* sea mayor o igual a 3).

Hay un problema en el que *LPG* tarda mucho en resolver, y es el problema *eva*. Esto puede deberse a que las restricciones del problema son muy fuertes y el planificador se *estresa* buscando la solución, pues es difícil de encontrar.

4. Dominio con recursos numéricos

En este problema se ha definido el recurso consumo de gasolina de la cinta, que tendrá que ser minimizado. Para ello se ha modificado la acción *mover-cinta*, al que se va a añadir el efecto en el que se aumenta el total de gasolina consumida.

A parte de eso también se ha añadido una acción llamada *mover-cinta-rapido*, que realiza la misma acción de mover un contenedor de un muelle origen a uno destino por una cinta pero al doble de velocidad, por lo que la duración se reduce. El problema de esto es que el consumo de gasolina pasa a ser el doble.

Esto se ha hecho así para poder comprobar la diferencia entre minimizar el recurso y minimizar el tiempo invertido en el plan. Se estima que en el momento de minimizar el consumo del recurso los planificadores tendrán predisposición por usar la acción *mover-cinta* y en el caso de minimizar el tiempo *mover-cinta-rapido*.

Finalmente, se ha implementado un dominio en el que el consumo de gasolina en una cinta tiene un límite. Para ello se definió un recurso *gasolina* asignado a una cinta, que representa el total de gasolina que posee una cinta. La acción *mover-cinta* y la acción *mover-cinta-rapido* sólo podrán realizarse siempre que quede el suficiente combustible. Para recargar la gasolina de una cinta se definió la acción *recargar-cinta*, con el que se recargaría la gasolina con una cantidad definida.

Al ejecutar los problemas con los cambios en los dominios anteriormente mencionados se observa como en los casos en los que se quiere minimizar el tiempo siempre se usa la acción de mover cinta rápido. Pero en los problemas en los que es necesario recargar, sigue usando mover rápido en la cinta aunque en algunos casos usa mover cinta normal. Esto puede deberse a que es más rentable mover la cinta a velocidad normal para no gastar recurso y así no tener que recargar, pues se perdería más tiempo recargando que sin recargar.

También se observa que en varias ocasiones en el caso de minimizar el recurso hay acciones que se repetían en bucle. Por ejemplo, coger y dejar un contenedor en una pila. Estas dos acciones se repetían constantemente realizando un bucle. Esto puede deberse a que esas acciones no consumían recurso, y como el objetivo era minimizar el consumo de recurso no tenían un impacto en esa métrica, por lo que podía darse ese bucle sin impacto en la función objetivo.

5. Árbol POP

En este apartado se van a realizar 3 iteraciones de un árbol POP sobre el problema propuesto en la Figura 8.

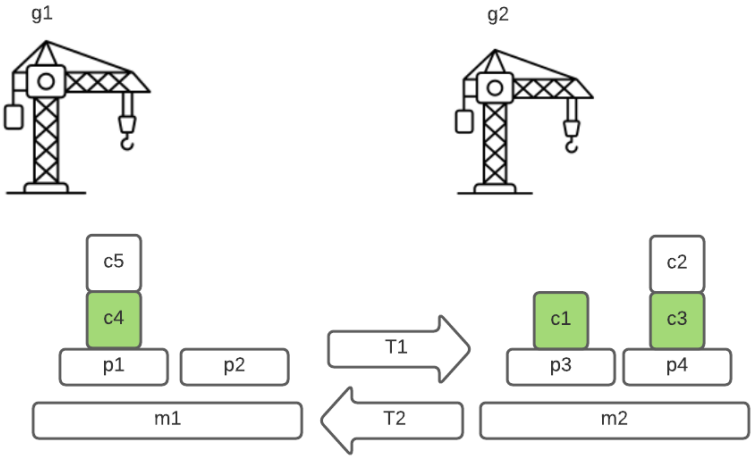

Figura 8: Estado inicial para árbol POP y Graphplan

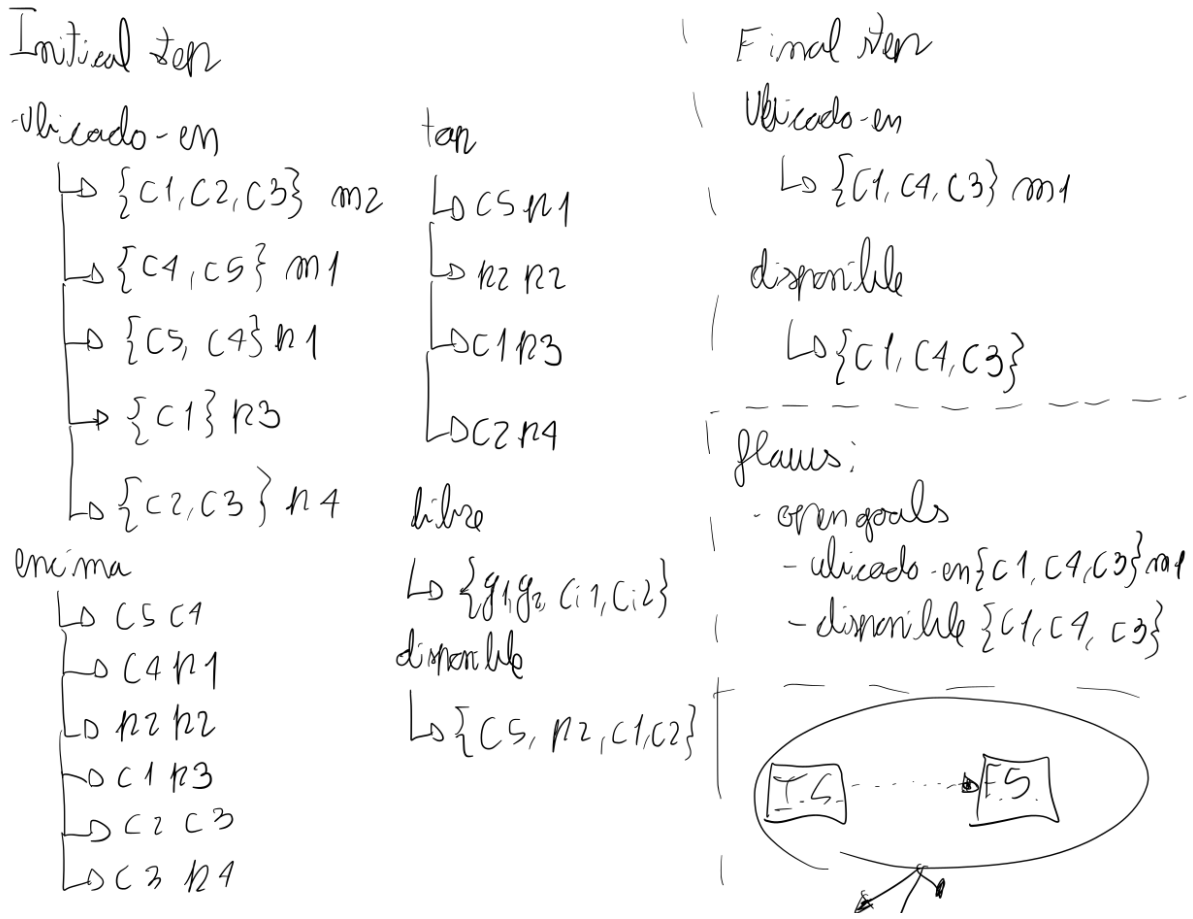

Figura 9: Estado inicial árbol POP

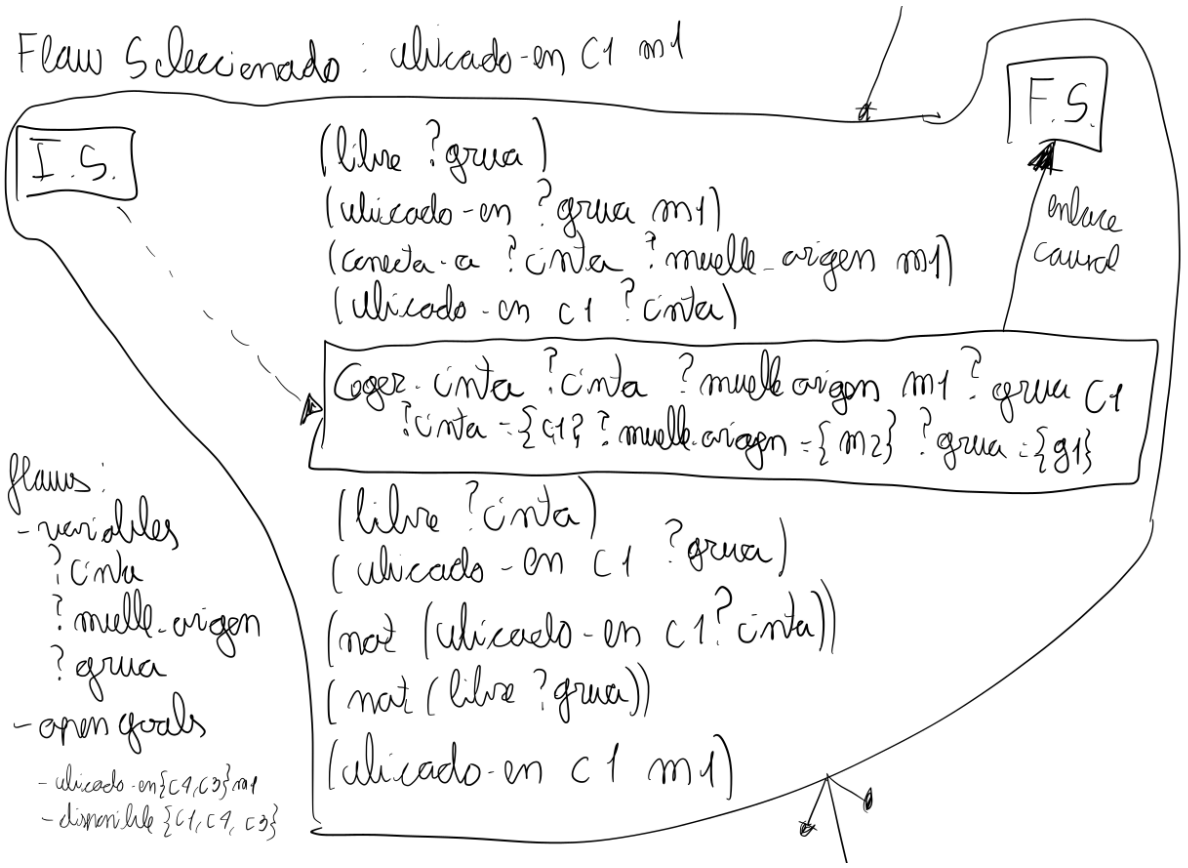

Figura 10: Primera iteración árbol POP

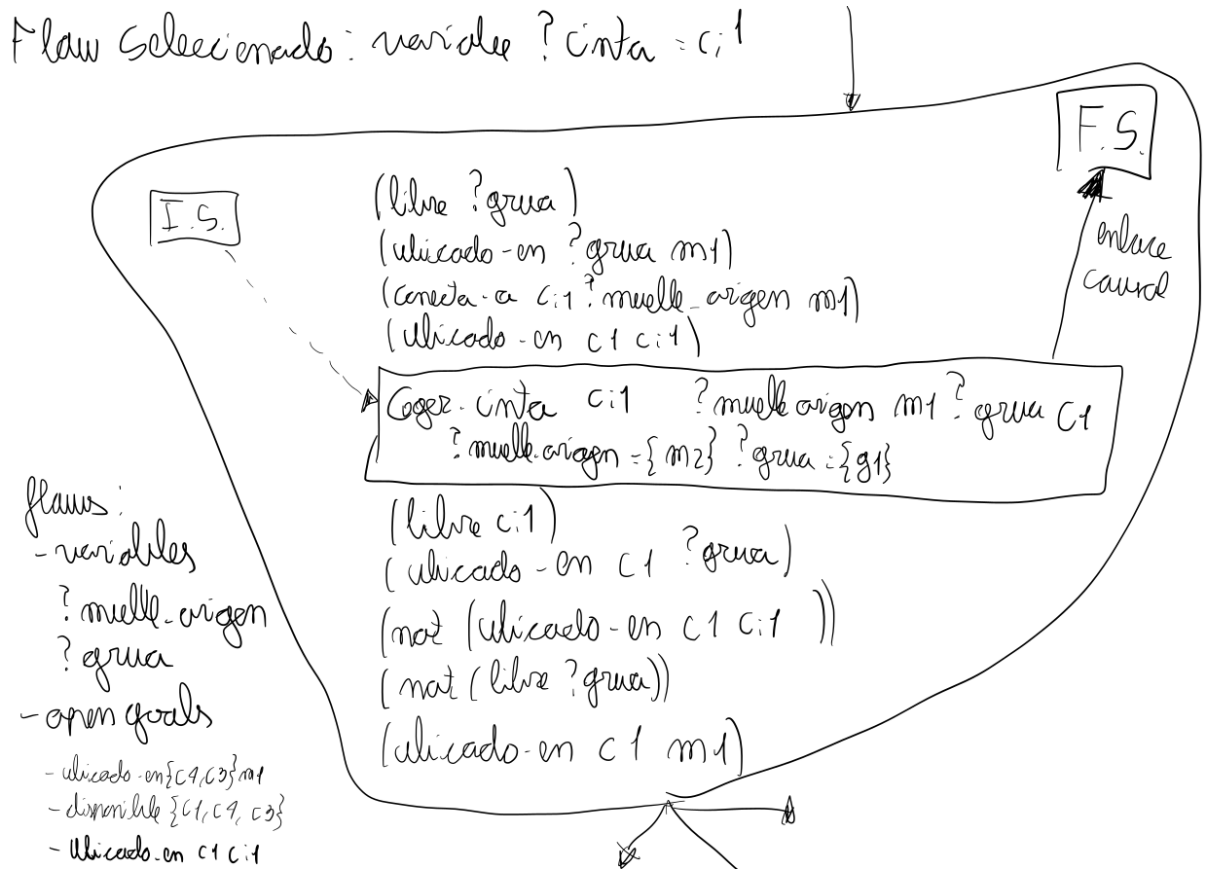

Figura 11: Segunda iteración árbol POP

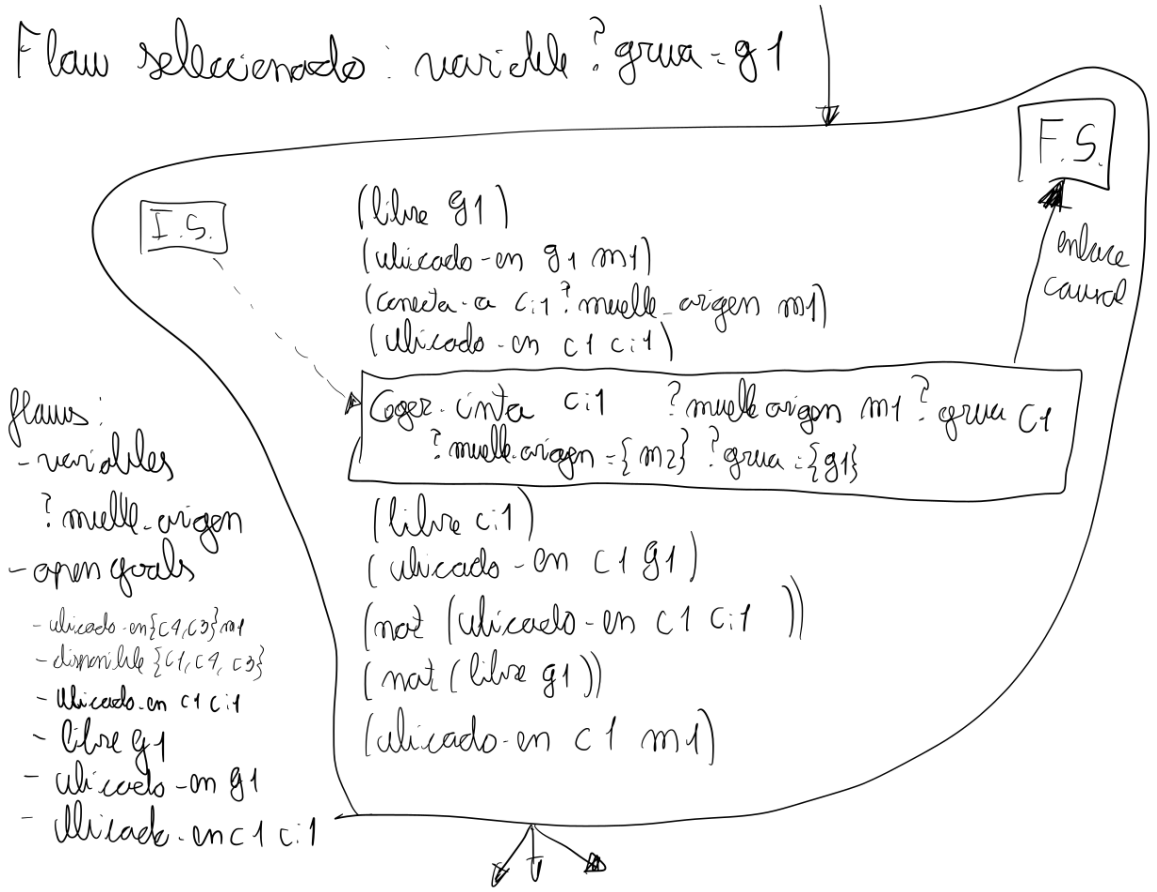

Figura 12: Tercera iteración árbol pop

En la Figura 12 se puede ver el resultado de realizar la tercera iteración de un árbol POP. Este problema podría tener problemas en la resolución de conflictos respecto a los hechos de *libre*. Esto es así porque en muchos casos se depende de que las grúas o las cintas estén libres.

Si se generan dos coger-cinta (instanciando las acciones de coger-cinta en la cinta 2) habría problemas, ya que ambas dependen de que la grúa 1 esté libre y tienen como efecto que la grúa 1 deje de estar libre, estas amenazas tendrían que ser resueltas con relaciones de orden. Esto pasaría también con las acciones de coger elementos de una pila.

En el estado inicial del ejercicio se han obviado varios hechos que no iban a cambiar en las iteraciones ni en el proceso de buscar una solución.

6. Graphplan

Como último ejercicio se propone realizar el grafo de planificación relajado del problema de la Figura 8, por lo que no se van a tener en cuenta los efectos negativos de las acciones ni tampoco las relaciones de exclusión mutua.

Los objetivos en forma de predicados es:

```
ubicado-en contenedor4 muelle1
    disponible contenedor4

ubicado-en contenedor1 muelle1
    disponible contenedor1

ubicado-en contenedor3 muelle1
    disponible contenedor3
```

Debido al gran número de predicados necesarios para definir el problema y resolverlo, los predicados que no cambian, como por ejemplo `ubicado-en pila1 muelle1` o las alturas no se tienen en cuenta. También, cuando el número de acciones irrelevantes para conseguir el objetivo es grande, se ha decidido poner "...", de esta forma se marca que se sabe que existen más acciones, pero ninguna de estas es relevante para conseguir un objetivo deseado.

El Graphplan se encuentra en un fichero Excel en la carpeta de la entrega.

6.1. Valor de las Heurísticas *h_sum* y *h_max*

Se tienen 6 objetivos, tal y como se ha presentado, cada uno se cumple en un paso concreto del grafo, pero cabe destacar que por ejemplo en P[0] ya se tienen dos objetivos cumplidos:

```
ubicado-en contenedor4 muelle1
    disponible contenedor1
```

Ya que, en la Figura 8 se puede observar que el contenedor4 se encuentra en el muelle1 y el contenedor1 no tiene contenedores encima, por lo que se encuentra disponible.

Se realizan las posibles acciones sobre P[0], y al coger con la grúa el contenedor5 queda disponible el contenedor4 cumpliendose otro objetivo, lo mismo ocurre al coger el contenedor2, que deja disponible el contenedor 3. De esta forma ya se tienen 4 objetivos cumplidos en P[1].

No se consigue ningún otro objetivo hasta llegar a P[3], donde se produce el efecto de A[3] donde la grua1 coge el contenedor1 de la cinta, provocando que el contenedor1 esté ubicado en el muelle1. En el mismo paso se deja el contenedor3 en la cinta2, dando lugar a que en P[4] ocurra que la grua1 coja el contenedor3 de la cinta, por lo que se cumple el último objetivo de que el contenedor3 esté ubicado en el muelle1.

Debido a que los objetivos de disponible se cumplen antes de que los contenedores lleguen a los muelles, cuando un contenedor que viene del muelle2 en la cinta2 es cogido por la grua1, automáticamente pasa a estar en el muelle1 y dicho contenedor tiene cumplidos sus dos objetivos `disponible contenedorX` y `ubicado-en contenedorX muelle1`.

La heurística *h_max* en el problema propuesto de la Figura 8 tiene un valor de 4, ya que el último objetivo que se cumple es `ubicado-en contenedor3 muelle1` en P[4], tal y como se muestra en la Tabla 1.

Por otro lado, la heurística *h_sum* se obtiene de la suma de los costes de nivel de los objetivos, por lo que a partir de los resultados de la Tabla 1 se tiene que *h_sum* = 0 + 0 + 1 + 1 + 3 + 4 = 9.

Objetivo	Nivel
`ubicado-en contenedor4 muelle1`	0
`disponible contenedor4`	1
`ubicado-en contenedor1 muelle1`	3
`disponible contenedor1`	0
`ubicado-en contenedor3 muelle1`	4
`disponible contenedor3`	1

Tabla 1: Nivel en el que se consigue cada objetivo.

6.2. Plan relajado

Una vez calculado del Graphplan del problema de la Figura 8 se puede extraer el plan relajado, para esto, se parte de cada objetivo alcanzado y se apunta a las acciones que han permitido lograr dicho objetivo, estas acciones se marcan en rojo en el Excel. Entonces se van estudiando las acciones que han permitido alcanzar los diferentes objetivos hasta llegar a P[0]. En la Tabla 2 se presentan las acciones

que se han marcado en rojo en el Excel, es decir, aquellas que forman el plan relajado. Por lo que el número de acciones obtenidas para el *plan_relajado* es 8.

A[1]	A[2]	A[3]	A[4]
`coger(grua1, contenedor5)`			
`coger(grua2, contenedor1)`			
`coger(grua2, contenedor2)`			
	`dejar-en-cinta( cinta2, contenedor1)`		
	`coger(grua1, contenedor3)`		
		`coger-de-cinta( grua1, contenedor1)`	
		`dejar-en-cinta(c inta2, contenedor3)`	
			`coger-de-cinta( grua1, contenedor3)`

Tabla 2: Plan Relajado.

Se puede apreciar que el plan se puede extraer en tiempo polinómico y además, no es necesario el uso de operaciones de *backtracking* debido a que no se tiene en cuenta la exclusión mútua.

6.3. Heurística más informada

Los valores de cada heurística se presentan en la Tabla 3. Si se calcula cual es el menor número de pasos para resolver el problema de la Figura 8 con el dominio proposicional, se obtiene que es 12, por lo que la heurística más informada es *h_sum* con un valor de 9, debido a que es la que más se acerca a este valor.

Cabe destacar que la heurística *h_max* tiene un valor demasiado optimista, por lo que no se considera de interés para este problema. Por otro lado, la heurística

plan_relajado es bastante informada ya que tiene un valor de 8, muy cercano al valor de *h_sum*.

Heurística	**Valor**
h_max	4
h_sum	9
plan_relajado	8

Tabla 3: Valores de las heurísticas.

7. Conclusiones y desarrollos futuros

Con este trabajo nos hemos introducido a la programación con PDDL, donde se han desarrollado 4 dominios diferentes:

- Proposicional
- Temporal
- Recursos no renovables
- Recursos renovables

Además se han propuesto diferentes problemas con diferentes tallas con el fin de testear la robustez de los dominios desarrollados. Los diferentes problemas han sido ejecutados con los planificadores FF, LPG y OPTIC. Y se ha conseguido desarrollar dominios flexibles para gran variedad de problemas relacionados con el puerto.

Personalmente ha sido una práctica en la que se han abarcado diferentes objetivos, dichos objetivos condensan casi todo el temario impartido en las clases de teoría, por lo que ha sido un trabajo entretenido y eficiente para asentar los conocimientos.

En conclusión, en esta práctica hemos aprendido a modelar un problema con diferentes dominios y a analizar su complejidad.