Técnicas de los Sistemas Inteligentes.

Práctica 3. Problemas de satisfacción de restricciones.

UNIVERSIDAD DE GRANADA

Ignacio Yuste López.

54141533Q

TSI. A1.

Tabla de resultados

PROBLEMA	T. PLAN	TIEMPO (s)	COSTE
1	3	0.00	X
2	11	0.00	X
3	16	0.06	X
4	28	0.02	X
5	30	0.24	X
6	30	0.28	X
7	42	0.84	X
8	42	1.82	478

Los dominios propuestos en la práctica son problemas de “juguete” y eso se ve reflejado claramente en los resultados obtenidos. Únicamente la ejecución del problema 8 pasa de un segundo de tiempo de ejecución, mientras que el resto ni siquiera llega.

Con los dominios propuestos no es necesario hacer una exploración muy extensiva. Por ejemplo, en el caso del problema 8 (que es el más “difícil”), al ejecutar el planificador observamos que la búsqueda ha evaluado 13433 estados, lo cual no es número muy grande en éste ámbito.

Sin embargo, si ejecutamos el planificador para el primer problema vemos que explora 7 estados, el segundo 58 y el tercero 1797. Vemos cómo por poco que aumente la complejidad de los dominios/problemas la búsqueda necesaria para encontrar una solución aumenta rápidamente. Y recordemos que aunque MetricFF intenta devolver un plan óptimo no siempre lo consigue.

Pregunta #1

En mi experiencia, MetricFF me devolvió en la primera ejecución el plan que terminó por ser el óptimo.

Pero es necesario comentar que, mientras que la ejecución donde no busco minimizar el tamaño del plan éste se encuentra en menos de 0.3s, la ejecución que buscaba reducir el tamaño y acabó devolviendo que no existía un plan con menos pasos tardo **casi 42 minutos**. Es increíble la diferencia de tiempo. A continuación vemos lo que devuelve la llamada a MetricFF:

```
best first search space empty! problem proven unsolvable.



time spent:    0.00 seconds instantiating 169 easy, 1379 hard action templates
               0.00 seconds reachability analysis, yielding 268 facts and 1425 actions
               0.00 seconds creating final representation with 239 relevant facts, 2 relevant fluents
               0.01 seconds computing LNF
               0.00 seconds building connectivity graph
            2697.04 seconds searching, evaluating 3491885 states, to a max depth of 0
            2697.05 seconds total time
```

Ha necesitado explorar casi 350k estados antes de concluir que no existe un plan con menor tamaño para el problema propuesto. Es sencillamente increíble la cantidad de posibilidades que pueden existir para un dominio de “juguete”. Más aún si lo comparamos con que la búsqueda del plan sin intentar minimizar sólo explora 3994 estados, devolviendo el plan que resulta ser el óptimo.

Esto puede deberse a que para comprobar si hay un plan mejor, uno con menos pasos, ha de comprobar antes todos los posibles planes (o muchos de ellas, dependiendo de cómo funcione internamente MetricFF) para concluir que efectivamente no hay un plan mejor. Mientras que sin esta comprobación sólo va a devolver un plan el cual tras una exploración más breve no se ha encontrado uno mejor.

Pregunta #2

Como ya he dicho al comienzo estos problemas y dominios son de “juguete”. Ni siquiera alcanzan un mínimo de la complejidad a la que puede llegar el sistema de Starcraft (el cual no conozco en profundidad pero sé a ciencia cierta que dispone de muchos más elementos y mecánicas). Si tuviéramos que buscar un plan que nos llevase a la victoria en el juego real, dadas ciertas circunstancias, sería inviable en cuestiones de tiempo de cómputo.

Yo categorizaría este dominio como de dificultad moderada. Si bien es más complejo que lo que puede ser, por ejemplo, un planificador para horarios de profesores de una escuela, sigue sin alcanzar la complejidad que puede tener un problema del día a día que nosotros resolvemos en cuestión de milisegundos. Si bien está claro, con PDDL sabemos que los planes obtenidos van a ser correctos y eso es algo que los humanos no pueden asegurar(siempre y cuando el dominio y problema estén bien planteados)

El mayor enemigo de la planificación automática es el tiempo de cómputo necesario para resolver problemas con cierta complejidad. Sin embargo, esto poco a poco irá dejando de serlo ya que la potencia de cómputo mejora día a día y los tiempos bajan. Lo que no cambia es la correctitud de los planes obtenidos, por lo que quien sabe si en un futuro no muy lejano esto deja de ser un problema.

EXTRA: Se ha implementado un sencillo script bash con el que ejecutar los problemas de manera sencilla. Para su uso simplemente escribir en la terminar “./ejecutor (numero de problema)”. Además se incluyen dos archivos de texto plano con los resultados de las ejecuciones que comprueban que el plan obtenido es el mínimo.