

Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

Planificación en STRIPS/PDDL

1. Introducción

En trabajos anteriores se ha explorado el estado del arte en el área de planificadores automáticos para la Inteligencia Artificial (IA). En este texto se retoma el uso de los planificadores para resolver problemas expresados en STRIPS/PDDL. PDDL es una familia de lenguajes que permiten definir un problema de planificación. Esta familia ha evolucionado junto con el desarrollo de la planificación, por lo que hay numerosas versiones disponibles con diferentes niveles de expresividad. (Green, 2021b)

Es importante señalar que los lenguajes PDDL comparten elementos entre sí, hay otros que son ampliamente utilizados pero no universales y algunos más son exclusivos de alguna versión de PDDL. En este texto se exploran *grosso modo* los fundamentos de PDDL a través de tres problemas de planificación automática.

2. Marco teórico

En IA la planificación explora por medio de métodos lógicos y de programación un proceso que pretende resolverse gracias a técnicas autónomas. En un problema de planificación existe un estado inicial que debe transformarse en un estado meta mediante la realización de un conjunto de acciones. (Ghallab et al., 2004; Green, 2021a)

En PDDL, se deben elaborar dos programas: dominio y problema. El dominio describe las determinaciones que forman el contexto, mientras que el problema detalla el estado inicial, final así como los participantes particulares de la situación.

Ambas descripciones, dominio y problema, serán los datos de entrada para que un planificador automático busque cual es la ruta de acción a seguir para pasar del estado inicial al estado final. Generalmente los planificadores reducen el problema a un contexto de búsqueda dentro de un grafo, de esta forma se expresa la solución como una ruta a seguir.

3. Marco referencial

En este texto se utilizarán los siguientes planificadores: FF, Metis (Sievers and Katz, 2018), LPG-td y el Solver Planning Domains (SPD). También se consideraron otros planificadores como IBaCoP (Cenamor et al., 2018) y Scorpion (Seipp, 2018), sin embargo, fueron descartados por su bajo desempeño en tiempo de ejecución.

En el dominio del problema a resolver se presenta un mapa con una serie de localizaciones, algunas de ellas son hospitales, otras tienen pacientes. Se asume que un

Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

robot conduce una ambulancia, originalmente localizada en un hospital, y debe llegar a la ubicación de los pacientes, subirlos a la ambulancia y llevarlos al hospital. La ambulancia solo puede atender un paciente a la vez.

Los problemas variarán según el número de ubicaciones, pacientes y caminos disponibles en el mapa. La figura 1 ilustra los tres problemas considerados, uno de ellos es el original u obligatorio para este texto (1c), los otros dos son variaciones del primero (1a y 1b).

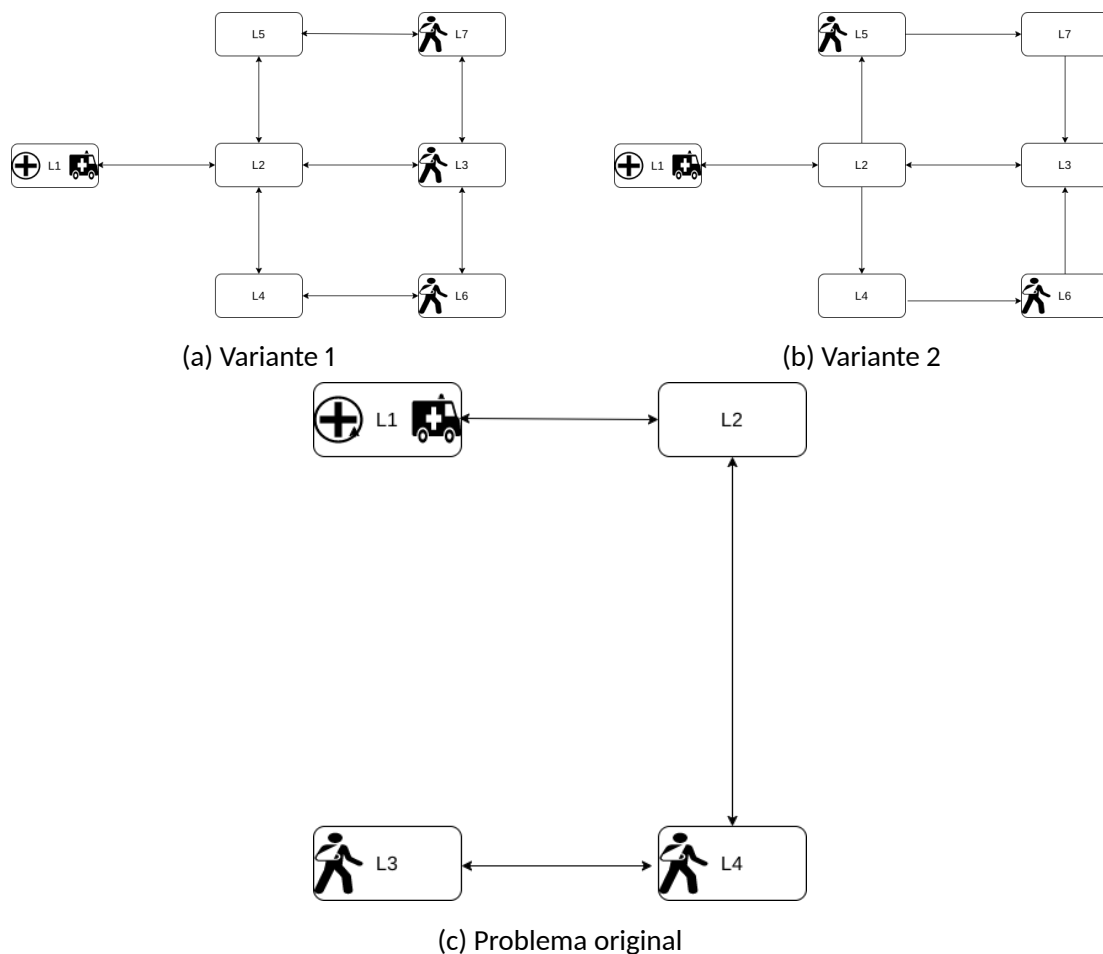


Figura 1: Problemas resueltos

4. Dominio y problemas

Tanto STRIPS como PDDL son lenguajes declarativos (Domínguez Espinoza, 2017). En el dominio (Ver listing 1) se distinguirá primero el nombre del mismo: *emergency* en este caso. Posteriormente, habrá listas de propiedades: el primer elemento de la lista es un parámetro (*:nombre del parámetro*) que indica el tipo de predicados que tiene la lista, Estos parámetros pueden repetirse según el nombre del parámetro, si la lista corresponde a acciones (*:action*), habrá tres parámetros adicionales: *:parameters*,

Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

:*precondition* y :*effect*. La primera de estas listas serán las variables que intervienen en la acción; la segunda serán los predicados que deben cumplirse para ejecutar la acción; la última los cambios consecuentes a realizar la acción.

Listing 1: domain.pddl

```

1  (define (domain emergency)
2    (:predicates
3      (ambulance ?a) ; Ambulancia
4      (at ?a ?l)      ; Si la ambulancia/paciente está en un
                        lugar
5      (carry ?p ?a)   ; Si el paciente está en la ambulancia
6      (current ?l)    ; La ubicación current de la ambulancia
7      (free ?a)       ; Si está vacía la ambulancia
8      (location ?l)   ; Los lugares
9      (path ?f ?t)    ; Camino entre dos lugares
10     (patient ?p))   ; Paciente
11  (:action move
12    :parameters (?from ?to)
13    :precondition (and
14                  (current ?from)
15                  (location ?from)
16                  (location ?to)
17                  (path ?from ?to))
18    :effect (and
19            (current ?to)
20            (not (current ?from))))
21  (:action getinto
22    :parameters (?p ?l ?a)
23    :precondition (and
24                  (ambulance ?a)
25                  (at ?p ?l)
26                  (current ?l)
27                  (free ?a)
28                  (location ?l)
29                  (patient ?p))
30    :effect (and
31            (carry ?p ?a)
32            (not (at ?p ?l))
33            (not (free ?a))))

```

Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

```

34  (:action getout
35      :parameters (?p ?l ?a)
36      :precondition (and
37          (ambulance ?a)
38          (carry ?p ?a)
39          (current ?l)
40          (location ?l)
41          (patient ?p))
42      :effect (and
43          (at ?p ?l)
44          (free ?a)
45          (not (carry ?p ?a))))

```

En el problema (Ver listing 2) se encuentra el nombre del mismo. Después se leen cuatro listas de propiedades: *:domain*, *:objects*, *:init* y *:goal*. La lista *:objects* consiste en un conjunto de átomos que nombran los participantes del problema, mientras *:init* y *:goal* listan predicados que describen la situación inicial y final del problema.

Listing 2: problem01.pddl

```

1  (define (problem original)
2      (:domain emergency)
3      (:objects ambulance hospital L2 L3 L4 patient1 patient2)
4      (:init
5          (ambulance ambulance)
6          (at patient1 L3)
7          (at patient2 L4)
8          (current hospital)
9          (free ambulance)
10         (location L2)
11         (location L3)
12         (location L4)
13         (location hospital)
14         (path L2 L4)
15         (path L2 hospital)
16         (path L3 L4)
17         (path L4 L2)
18         (path L4 L3)
19         (path hospital L2)
20         (patient patient1)
21         (patient patient2))

```

Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

```

22      (:goal (and
23              (at patient1 hospital)
24              (at patient2 hospital))))

```

En general se ha descrito un programa y un ejemplo del mismo. Adicionalmente, hay dos dificultades principales al crear estos programas: 1) La documentación sobre el lenguaje; 2) La verificación del programa escrito.

Sobre la documentación del lenguaje, no fue posible encontrar un estándar o documentación oficial sobre cualquiera de las versiones de PDDL. Cada versión tiene diferentes opciones de parámetros posibles, aún así tan solo con el uso del parámetro *:typing*, el más común entre versiones, se presentaron problemas. También se dice que la sintaxis es la misma de Lisp, mas al usar una expresión *S* del conector lógico OR, fue igualmente imposible hacer funcionar el programa. Los programas presentados tienen la sintaxis más básica de STRIPS, el antecesor de PDDL y con mayor correspondencia a Lisp, una familia de lenguajes conocida y bien documentada.

Agregado a lo anterior se encuentra la falta de un verificador léxico/semántico para el programa. La única forma de verificación se encuentra en la ejecución sobre el planificador, el cual no da detalles adecuados para la depuración del programa ya que no es su propósito. Afortunadamente, existe el editor en línea de *planning.domains*, que tiene las mismas carencias que el modo PDDL de GNU/Emacs, no corrige la sintaxis y no permite escoger con antelación la versión de PDDL que se desea usar, por lo que no es posible saber si los errores son ocasionados por el programador o por limitaciones del planificador que se usa para probar el programa.

5. Resultados y comparación

A continuación se encontrarán los resultados del problema original (Ver figura 1c) de cada uno de los planificadores probados. Los planificadores hacen el mismo número de pasos, catorce, salvo LPG-td que registró 16 pasos. Si se observa con atención entre los pasos 1 y 3 el planificador regresa innecesariamente entre los lugares L2 y L4 (Ver líneas 13 a 15 en listing 4), por lo que en realidad también ofrece 14 pasos efectivos, quizá exista un error en el planificador.

Dentro del plan entregado, los planificadores FF y Metis coinciden en el resultado final, en ambos se recoge primero al paciente de la ubicación L3 (Ver listing 3 y 7); los planificadores LPG-td y SPD por su parte decidieron recoger primero al paciente de la ubicación L4, dicha ubicación es la más cercana al hospital, mismo que define el estado inicial del robot (Ver listing 5 y 4). Sin embargo, esta diferencia no hace más costoso ninguno de los dos planes posibles.

Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

Respecto al tiempo de ejecución, FF y LPG-td marcaron en cero el contador de dos decimales. Si se recurre al comando `time`, se puede ver que FF tarda $0.008s$ y LPG-td en promedio tarda $0.040s$. Metis es el segundo más rápido, incluso cuando requiere un contenedor Singularity, con un tiempo de $0.001s$, aunque el más rápido fue SPD, que se corrió desde un servidor externo mediante un API ??, requirió tan solo $1.8999 \times 10^{-10}s$ (Ver listing 6).

Listing 3: Resultado de FF

```

1  ff: parsing domain file
2  domain 'EMERGENCY' defined
3  ... done.
4  ff: parsing problem file
5  problem 'ORIGINAL' defined
6  ... done.
7
8  Cueing down from goal distance:      7 into depth [1] [2] [3] [4] [5]
9                                         6          [1]
10                                        5          [1] [2]
11                                       4          [1] [2] [3]
12                                      3          [1]
13                                     2          [1]
14                                    1          [1]
15                                   0
16
17  ff: found legal plan as follows
18
19  step      0: MOVE HOSPITAL L2
20            1: MOVE L2 L4
21            2: MOVE L4 L3
22            3: GETINTO PATIENT1 L3 AMBULANCE
23            4: MOVE L3 L4
24            5: MOVE L4 L2
25            6: MOVE L2 HOSPITAL
26            7: GETOUT PATIENT1 HOSPITAL AMBULANCE
27            8: MOVE HOSPITAL L2
28            9: MOVE L2 L4
29           10: GETINTO PATIENT2 L4 AMBULANCE
30           11: MOVE L4 L2
31           12: MOVE L2 HOSPITAL
32           13: GETOUT PATIENT2 HOSPITAL AMBULANCE
33
34  time spent:    0.00 seconds instantiating 22 easy, 0 hard action templates
35                0.00 seconds reachability analysis, yielding 15 facts and 22
36                  actions
37                0.00 seconds creating final representation with 15 relevant facts
38                0.00 seconds building connectivity graph
39                0.00 seconds searching, evaluating 21 states, to a max depth of 5
40                0.00 seconds total time

```

Listing 4: Resultado de LPG-td

```

1  ; Version LPG-td-1.4

```

Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

```

2 ; Seed 33654337
3 ; Command line: ./planners/LPG-td-1.4/lpg-td -o domain.pddl -f problem01.pddl -n 1
  -out out/pl-lpg-td
4 ; Problem problem01.pddl
5 ; Actions having STRIPS duration
6 ; Time 0.00
7 ; Search time 0.00
8 ; Parsing time 0.00
9 ; Mutex time 0.00
10 ; NrActions 16
11
12 0: (MOVE HOSPITAL L2) [1]
13 1: (MOVE L2 L4) [1]
14 2: (MOVE L4 L2) [1]
15 3: (MOVE L2 L4) [1]
16 4: (GETINTO PATIENT2 L4 AMBULANCE) [1]
17 5: (MOVE L4 L2) [1]
18 6: (MOVE L2 HOSPITAL) [1]
19 7: (GETOUT PATIENT2 HOSPITAL AMBULANCE) [1]
20 8: (MOVE HOSPITAL L2) [1]
21 9: (MOVE L2 L4) [1]
22 10: (MOVE L4 L3) [1]
23 11: (GETINTO PATIENT1 L3 AMBULANCE) [1]
24 12: (MOVE L3 L4) [1]
25 13: (MOVE L4 L2) [1]
26 14: (MOVE L2 HOSPITAL) [1]
27 15: (GETOUT PATIENT1 HOSPITAL AMBULANCE) [1]

```

Listing 5: Plan de Solver Planning Domain

```

1 (move hospital 12)
2 (move 12 14)
3 (getinto patient2 14 ambulance)
4 (move 14 12)
5 (move 12 hospital)
6 (getout patient2 hospital ambulance)
7 (move hospital 12)
8 (move 12 14)
9 (move 14 13)
10 (getinto patient1 13 ambulance)
11 (move 13 14)
12 (move 14 12)
13 (move 12 hospital)
14 (getout patient1 hospital ambulance)

```

Listing 6: Resultado de Solver Planning Domain

```

1 --- OK.
2 Match tree built with 22 nodes.
3
4 PDDL problem description loaded:
5   Domain: EMERGENCY
6   Problem: ORIGINAL
7   #Actions: 22

```

Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

```

8      #Fluents: 15
9      Landmarks found: 2
10     Starting search with IW (time budget is 60 secs)...
11     rel_plan size: 7
12     #RP_fluents 8
13     Caption
14     {#goals, #UNnachieved, #Achieved} -> IW(max_w)
15
16     {2/2/0}:IW(1) -> [2][3][4][5][6][7]rel_plan size: 5
17     #RP_fluents 6
18     {2/1/1}:IW(1) -> [2][3][4][5][6][7][8][9]rel_plan size: 0
19     #RP_fluents 0Plan found with cost: 14
20     Total time: -1.89999e-10
21     Nodes generated during search: 58
22     Nodes expanded during search: 51
23     IW search completed

```

Listing 7: Resultado de Metis

```

1      Solution found!
2      Actual search time: 0s [t=0.00119607s]
3      move hospital 12 (1)
4      move 12 14 (1)
5      move 14 13 (1)
6      getinto patient1 13 ambulance (1)
7      move 13 14 (1)
8      move 14 12 (1)
9      move 12 hospital (1)
10     getout patient1 hospital ambulance (1)
11     move hospital 12 (1)
12     move 12 14 (1)
13     getinto patient2 14 ambulance (1)
14     move 14 12 (1)
15     move 12 hospital (1)
16     getout patient2 hospital ambulance (1)
17     Plan length: 14 step(s).
18     Plan cost: 14
19     Expanded 30 state(s).
20     Reopened 0 state(s).
21     Evaluated 40 state(s).
22     Evaluations: 80
23     Generated 69 state(s).
24     Dead ends: 0 state(s).
25     Expanded until last jump: 25 state(s).
26     Reopened until last jump: 0 state(s).
27     Evaluated until last jump: 33 state(s).
28     Generated until last jump: 58 state(s).
29     Number of registered states: 40
30     total successors before partial-order reduction: 69
31     total successors after partial-order reduction: 69
32     Search time: 0s
33     Total time: 0.00119607s
34     Solution found.
35     Peak memory: 4960 KB

```


Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

Al describir los problemas variantes se observó que para Metis los tiempos de ejecución aumentaron a $0.0046s$ en la primera variante del problema, mientras que para los otros tres planificadores los tiempos se mantuvieron sin cambios, incluso SPD mantuvo el tiempo de ejecución completamente igual. La diferencia negativa la marcó LPG-td que en la primera variante del problema volvió a registrar un número mayor de pasos: 26 pasos; mientras que los otros planificadores necesitaron solo 22 pasos en la solución. En la segunda variante, al tener caminos en una sola dirección, LPG-td por fin logró obtener 16 pasos, mismos que los otros planificadores.

Los listings 8 y 9 muestran los planes obtenidos para resolver los dos problemas adicionales planteados.

Listing 8: Plan de la primera variante con Solver Planning Domain

```

1  (move hospital 12)
2  (move 12 13)
3  (getinto patient1 13 ambulance)
4  (move 13 12)
5  (move 12 hospital)
6  (getout patient1 hospital ambulance)
7  (move hospital 12)
8  (move 12 13)
9  (move 13 16)
10 (getinto patient2 16 ambulance)
11 (move 16 13)
12 (move 13 12)
13 (move 12 hospital)
14 (getout patient2 hospital ambulance)
15 (move hospital 12)
16 (move 12 13)
17 (move 13 17)
18 (getinto patient3 17 ambulance)
19 (move 17 13)
20 (move 13 12)
21 (move 12 hospital)
22 (getout patient3 hospital ambulance)

```

Listing 9: Plan de la segunda variante con Solver Planning Domain

```

1  (move hospital 12)
2  (move 12 14)
3  (move 14 16)
4  (getinto patient2 16 ambulance)
5  (move 16 13)
6  (move 13 12)
7  (move 12 hospital)
8  (getout patient2 hospital ambulance)
9  (move hospital 12)
10 (move 12 15)
11 (getinto patient1 15 ambulance)
12 (move 15 17)
13 (move 17 13)

```

Asignatura	Datos del alumno	Fecha
Razonamiento y planificación automática	Apellidos: Domínguez Espinoza Nombre: Edgar Uriel	23 de abril de 2022

```

14 (move 13 12)
15 (move 12 hospital)
16 (getout patient1 hospital ambulance)

```

6. Conclusión

Durante este texto se ha explorado la escritura de dominios y problemas de planificación automática en lenguajes STRIPS/PDDL, y su verificación por medio de diversos planificadores de código abierto. Se han descrito los elementos básicos obligatorios para la escritura de código y los resultados obtenidos de los planificadores.

Se ha visto que los planificadores tienen rendimiento constante aunque existe variación entre ellos. Si bien los resultados son muy similares, un tiempo de ejecución mejorado puede ser importante al resolver problemas de planificación más grandes.

Se ha hecho énfasis en las dificultades provocadas por la falta de documentación clara y herramientas de desarrollo avanzadas. Quizá el trabajo a futuro más interesante se encuentra en estos dos puntos, pues permitiría una mejor colaboración entre aquellos interesados en la planificación automática.

Los detalles adicionales con resultados más detallados que no pudieron ser incluidos en este documento se encuentran en el repositorio de la actividad: <https://gitlab.com/genomorro/unir/-/tree/RPA-03>.

Referencias

- Cenamor, I., de la Rosa, T., and Fernández, F. (2018). IBaCoP-2018 and IBaCoP2-2018. *Ninth International Planning Competition (IPC 2018)*.
- Domínguez Espinoza, E. U. (2017). *Procesador de lenguas naturales*. Escuela Nacional de Antropología e Historia.
- Ghallab, M., Nau, D., and Traverso, P. (2004). *Automated Planning: Theory and Practice*. Elsevier.
- Green, A. (2021a). What is ai planning?
- Green, A. (2021b). What is planning domain definition language (pddl)?
- Seipp, J. (2018). Scorpion (planner abstract). *Ninth International Planning Competition (IPC 2018)*.
- Sievers, S. and Katz, M. (2018). Metis. *Ninth International Planning Competition (IPC 2018)*.