Estado del arte planificadores

José Miguel Chacón, Víctor Gómez, Germán Portilla

Universidad Internacional de la Rioja, Logroño (España)

3 de Marzo de 2022

RESUMEN

Los planificadores constituyen una parte fundamental de la generación de soluciones a partir de condiciones estructuradas a los problemas, la diferencia entre los planificadores radica en el alcance, la capacidad del planificador el lenguaje donde se estructure su dominio, y además el tiempo de ejecución de las tareas definidas para cada situación, en este documento se relacionan 4 planificadores dispuestos en la IPC 2018 que mide los resultados de cada uno de estos.

PALABRAS CLAVE

icaps, planificadores

I. INTRODUCCIÓN

En los tiempos actuales es poco común ver la aplicación de algoritmos basados en planificadores, pero son muy conocidos en diferentes áreas, si no que por el enfoque y la complejidad de implementación las tareas se hacen mas complejas y se buscan estrategias de solución a problemas mucho mas sencillas, que garanticen respuestas rápidas pero no eficientes y es en este punto que los planificadores ya sea de tipo uní agente o multiagente toman su papel fundamental, puesto que además de buscar una solución a problemas de logística de adecuación o de comportamiento, también lo hacen en tiempos reducidos que dependen directamente de la complejidad asignada y la función costo que garantiza la solución esperada. Para el caso de los problemas de costo son muy comunes en los planificadores puesto que a partir del conjunto de criterios se puede establecer una matriz de correlaciones que evidencie una respuesta adecuada a la solución y que aparte de encontrarla satisfaga de una u otra manera el problema planteado, y de este modo generar un uso de las herramientas presentadas en la asignatura a los problemas que se dispongan vinculando sentencias de IA que mejoren el comportamiento de las variables.

II. ESTADO DEL ARTE

A. DecStar – STAR-topology DECoupled Search at its best

Este planificador presentado en el IPC 2018 hace parte del grupo de los planificadores mas prácticos para solución a las tareas de búsqueda de rutas y soluciones a problemas , es relativamente nuevo, hacia el año 2015 y lo compone una estructura de búsqueda secuencial, con el manejo de funciones heurísticas de mapeo, y con un camino a la separación de los estados del espacio , con el fin de separar las tareas, y forzando en topología estrella la estructura del mismo, La modificación del planificador STD contiene múltiples factores que se basan en el estado de las variables , estas preguntas de parcialidad son reconocida por Hoffman y Gnad que contemplan la topología estrella o star en la solución de la ruta logística de menor costo, aunque la función objetivo no se integra como se debe es un criterio fundamental en la planificación de factores singulares. La búsqueda introducida por Gnad es una integración de mapeo rápido con un sistema de planificación asistido, y con la separación o particionamiento de tareas que lo hacen un porcentaje mas ágil, así mismo se presenta la simetría de la ruptura

que parte desde el año 1991, como el ajuste de la extensión permisible del entorno, y que al final representa una orbita de área generada.

En la implementación de operaciones del método se tienen diversas estrategias practicas basadas en algoritmos desde el mapeo estructurado como la inteligencia artificial en el ajuste de parámetros de componentes y planificación que garantizan el conjunto de heurísticas aplicables al proceso, las limitantes del planificador DevStar radican principalmente en el tiempo de búsqueda de soluciones basadas en las heurísticas seleccionadas, y con DA* hasta el A*. se establece en 100 s a 180s que podrían ir en aumento dependiendo el conjunto de condiciones que se monten sobre el problema en cuestión.

El portafolio de configuración del método es extenso y el tiempo junto a la respuesta radica principalmente el la selección de la búsqueda puesto que al variar este criterio tanto la heurística se ve afectada como la frecuencia máxima en respuesta y el criterio de Prunning suministrado por tabla. En la búsqueda o mapeo ágil se presenta disminución significativa en el impacto temporal, pero puede estar enfocado directamente al factor usado, al criterio de búsqueda y la limitación, el planificador DevStar hace parte de los métodos de variables limitadas pero el conjunto de variaciones permisibles son elevadas pero puede presentar cambios en la estructura y generar cambios significativos en el recorte de tiempo, la búsqueda de solución y la eficiencia en la respuesta eseprada. (Gnad et al., n.d.)

B. SYMPLE: Symbolic Planning based on EVMDDs

En la búsqueda de un planificador optimo surge el simple, el cual aprovecha de manera optima los recursos y estructura relacional. De condiciones binarias puesto que son las estrategias mas populares en la separación de respuestas a los problemas que se planteen, varios autores contribuyeron de manera significativa al trabajo desarrollado entre os que se destacan kissman, Edelkamp y Hoffman que buscaron una alternativa netamente simbólica que diera solución a los problemas de planificación, el concepto acuñado por Geiber en el 2018, y se basa directamente en una base simbólica para la planificación optima mediante una relación directa de sets y relaciones de prioridad o transición entre el anterior y el posterior, se presenta al igual que los anteriores en el IPC del 2018 como uno de los planificadores mas versátiles, cuya búsqueda no es dependiente al costo de las acciones, no se relaciona la función objetivo pero debe considerarse puesto que la carga del planeador radica sobre este criterio de alta relevancia, de este planificador se tienen 2 versiones la simple-1 y la simple-2. Y las variaciones radican en la formulación de búsqueda y enrutador de las respuestas optimas a los problemas planteados.

La implementación de el simple radica desde el pro procesamiento del juego, planteados por kissman y simba donde los procesar son los siguientes, Gamer , h2 Gamer y simba puede relacionarse a comportamiento variado de heurísticas pero son una combinación de múltiples acciones en una relación de pesos por extensión nodal, la búsqueda del algoritmo o planificador Simple, presenta una iteración inferior con dirección a la búsqueda automática. Se integra una función Medly el cual consiste en un conjunto de operaciones necesarias que garantizan la planificación simbólica en los diferentes escenarios de proceso, en los cuales se enmarca el intermedio, optimista y pesimista, este ultimo es la condición de aumento en árbol y ramificación contemplada.

C. Delfi: Online Planner Selection for Cost-Optimal Planning

Es un planificador propuesto por Katz, Sohrabi y Hortz. Se basa directamente en e=un planificador de costo optimo mediante funciones objetivo, en un dominio aprovechable, según el autor la planificación optima relaciona el Excel en un conjunto de ciclos y sentencias que afinan su comportamiento, este dominio de trabajo tiene una particular forma de trabajar puesto que se instauran técnicas populares de la planificación y la normalización de búsqueda de rutas de manera rápida y eficiente, el agente

diferenciador de este planificador radica en el uso online puesto que usa sentencias definidas que se retroalimentan con cada una de las pruebas lo que garantiza una eficiencia y búsqueda del menor costo en la función objetivo, además integra un entorno grafico que se soporta sobre técnicas de inteligencia artificial o exploits de Deep learning o aprendizaje profundo que mejoran el aprendizaje de los modelos además de afinar los criterios de búsqueda en la ruta y ejecución de las tareas.

Dentro del análisis presentado en el documento esta la inclusión de múltiples planificadores óptimos que se consideran como una colección y de esto se determinan criterios relevantes de ajuste en la mejor modificación de resultados, se presenta el planificador de Hermert del año 2006 el cual entregaba respuestas rápidas pero sin un ajuste en el resultado idóneo, al mismo tiempo el portafolio propuesto es de tipo singular o simple muy diferente a otros métodos múltiples o de agente múltiple, en el documento se presentan al menos 17 trabajos o planificadores que reposan en las versiones de IPC 2014 hasta la fecha, cada uno con variantes relevantes en el proceso de búsqueda mediante las heurísticas definidas.

El planificador tiene diferentes ítems relevantes entre los cuales se integra o se instaura el uso de grupos definidos o configuraciones asignadas mediante efectos condicionales, que extienden la implementación en términos de eficiencia, de al menos 1000 expansiones con un 10 % de estados no percibidles que se consideran dentro de los 16 planificadores. Además el algoritmo integra manejos de heurísticas como la LM , heurística tipo blind , patrón de base de datos, particionamiento de cruce a cero con particionamiento de patrones de la base de datos, variantes heurísticas propuestas por drager y Podelski en el 2009 con el criterio de bisimulación, se adiciona del mismo modo criterios de mema basados en estrategia de múltiples sentencias enlazadas, computación previa de respuestas automáticas y respuesta optima de convergencia con una variable simple.

El planificador Delfi presenta múltiples capas en su estructura, parte desde la convolucion en 2D principal en la aplicación de el DL, con unas etiquetas o codificación de entrada en 128 a 128 con 1 canal, la salida es simétrica o complementaria a la entrada, luego presenta una estructura similar a la entrada pero con 2 dimensiones de entrada lo que genera un aumento dimensional en su salida, luego en el orden del DL se tiene el Maxpooling en 2D con una salida singular en unos canales pero con el tope en otros, se da de la manera 64,64,128, luego es necesario un flatten son una salida múltiple con la entrada previa, en una cascada, luego una etapa dropuot con salida similar a la entrada, y finalmente una de densidad con un dato de longitud pero una salida representada en canal. Luego la creación del modelo y su entrenamiento se basa en el algoritmo de Keras presentado en el Chollet con tensor Flow, esta generación de modelo se basa es una estructura de red convolucion de capas sobrepuestas. Los tiempos de demora dependen de la capacidad del algoritmo y el conjunto de sentencias que se tengan en la búsqueda de una solución optima basada en una heurística practica y rápida normalmente optimizada con el uso del gradiente estocástico, momentum y ancho de traza. (Ma et al., 2018)

Al final de la aplicación del planificador se ejecuta un análisis que suministra estadística del grado de aceptación e impacto del mismo, se tiene una evaluación sobre planificador optimo en el IPC 2018 con una estructura de cíclica junto a la mejora automatizada del CNN que favorece el comportamiento del método o algoritmo propuesto, tanto el Delfi como el Oracle tienen el mayor grado de base instanciada persistente que garantiza que los resultados suministrados sean óptimos con el menor costo, de una función objetivo estructurada al inicio de su aplicación. (Katz et al., n.d.)

D. Scorpion

Este planificador se basa en una estructura de optimizacion secuencial integrada en el IPC 2018 como el anterior, tiene aspectos similares como el de la rapidez en la busqueda de las soluciones según como lo prsenta Helmert en el 2006, con la incluion de una heurustica admisible, esta heuristica admisible depen-

de de un componente de admision combinada con una particion de costo como s eprresentara mas adelante. La heurustica de adbstraccion debende de un conjutno de tareas que contienen efectos de condicion qddue son usados en las diferentes aplicaciones, Las tareas incluyen la caretesiana, con decomposiciones segmentadas que un palno equilibrado de transisiciones, icluye una base de patrones con decaimiento que generan un espacio de patrones dentro de un espacio que facilita las tareas en el enrutado. Ademas de un sistema de patrones

Las tareas con los efectos condicionales generan la capacidad de computo de cada una de la stareas designadas para la busqueda de la plificacion optima y capaz según sea el problema estructurado, este tipo de generacion de heuristica es sistematico lo que quierre decir que esta framentado en fases de perfil de union, pero con un mejor alcace. Aplicar una heuristica CART como lo relaciona el autor suministra resultados favorables, enfocados en la eficiencia del metodo en factores relevantes, y las mezclas ajustadas según el criterio de heuristica pirncipal, cada una de las pruebas realizadas incluyen lo agricola, redes y datos, Snake y Spider del mismo modo que los temas.

Usando una heuristca tipo h2 , como lo presenta torralba en el 2015 es un operador generado al interiror de un metodo que puede invocar de manera independiente según sea su componente ey rapidez y en la busqueda.En los experimentos realizados sobre el planificador que al igual que el anterior estan integrados dentro del IPC, se tienen al menos 3 versiones de heuristicas CART, con diferentes tareas y cargas limitantes sobre las sentencias, el uso de la memoria es elevado casi todo el tiempom y aumentos conseuentes de almenos 7Gb , esto con almenos 80 % aproximadamentes con la heuristica relacionada anteriormente, ya si se presentan mezclas dentro del escorpion es solo determinar con parametros los resultados que va otorgando. (Francès et al., n.d.)

Este planificador fue probado en campos como el agriculo y otros en los cuales se determino la capacidad de dar solucion rapida, a los problemas que se van presentadno según corresponda, agregando planos cartesianos abstratos en el decaimiento de los PDBs que soportan la heuristica. En las redes de datos las abstracciones cartesianas resuleven 14 tareas conrra 12 tareas sin abstracciones cartesianas, dependiendo del control heuristco que se tenga, que garantice la solucion, aunque la produndidad del metodo no es la esperada se presenta los rasgos preliminares de aplicación de la tecnica o del planificador enfocado en la solucion sistema de operación o problemas con el menor tiempo y el menor costo.(Seipp, n.d.)

III. INSTALACIÓN Y PRUEBA

Para la prueba de los planeadores en mención, se creó una máquina virtual en Google Cloud Platform (GCP por sus siglas en inglés), con el objetivo de usar un sistema operativo linux, que funcione correctamente con las dependencias necesarias para generar las imágenes de los mismos. A continuación se detallan los pasos realizados. La máquina tiene las siguientes características:

- Memoria RAM: 7.5 GB
- Disco duro: 20 GB
- CPU: Intel haswell
- SO: Ubuntu 18.4 LTS

Basic information

Name	ubuntu
Instance Id	6478875598435584873
Description	None
Type	Instance
Status	Running
Creation time	Mar 1, 2022, 7:08:39 PM UTC-05:00
Zone	us-central1-a
Instance template	None
In use by	None
Reservations	Automatically choose
Labels	None
Deletion protection	Disabled
Confidential VM service	Disabled
Preserved state size	0 GB

Figura 1: Información de la máquina en GCP

Para compilar el plan, usamos el software *Singularity*. Singularity es una plataforma de

contenedores que permite administrar contenedores que empaquetan piezas de software de forma portátil y reproducible. En la VM creada, se compila el código fuente del release 3.9.5.

Figura 2: Versión de singularity utilizada

Para cada uno de los planificadores elegidos, se descarga su contenedor correspondiente usando wget.

```
wget urlDeLaImagenenBitbucket
```

Para generar las imágenes, se ejecuta el siguiente comando en la terminal:

```
sudo singularity build
    plannerNombre.img Singularity
```

Esto genera las imágenes para cada planeador, que al final guardamos en un folder llamado planners. Posterior a esto, para probar el desempeño de los planeadores, elegimos un problema sencillo del mundo de los bloques:

Listing 1: Dominio de los bloques

```
; Mundo de los bloques
(define (domain blocksworld)

(:requirements :typing
:negative-preconditions)

(:types block) ;we do not need a
; table type as we use a special
; ontable predicate

(:predicates
  (on ?a ?b - block)
  (clear ?a - block)
  (holding ?a - block)
  (handempty)
  (ontable ?x - block)
)

(:action pickup
; Levantar un bloque
:parameters (?x - block)
:precondition (and (ontable ?x)
  (handempty)
  (clear ?x))
:effect (and (holding ?x)
  (not (handempty))
  (not (clear ?x))
  (not (ontable ?x))
 )
)
(:action unstack
; Desapilar un bloque
:parameters (?x ?y - block)
:precondition (
  and (on ?x ?y)
        (handempty)
        (clear ?x)
)
:effect (and (holding ?x)
  (not (handempty))
  (not (clear ?x))
  (clear ?y)
  (not (on ?x ?y))
 )
)

(:action putdown
:parameters (?x - block)
:precondition (and
  (holding ?x)
)
:effect (and
(ontable ?x)
 (not (holding ?x))
 (handempty)
 (clear ?x)
 )
)

(:action stack
:parameters
(?x ?y - block)
:precondition
(and (holding ?x)
 (clear ?y)
 )
:effect (
 and (on ?x ?y)
        (not (holding ?x))
        (handempty)
        (not (clear ?y))
        (clear ?x)
  )
 )
)
```

Listing 2: Problema.

```
(define (problem blocks)
(:domain blocksworld)
```

Figura 3: Objetos en el dominio elegido

```
(:objects
  red green blue
  brown pink gold - block
)

(:init
  ;tower
  (ontable red) ; Block red
  (on green red) ; Block green
  (on blue green)
  (clear blue) ; Block blue
  ;tower
  (ontable brown) ; Block brown
  (on pink brown) ; Block pink
  (on gold pink)
  (clear gold) ; Block gold
  (handempty)
)

(:goal (and
  (on red brown)
  (on green red)
  (holding gold)
))
)
```

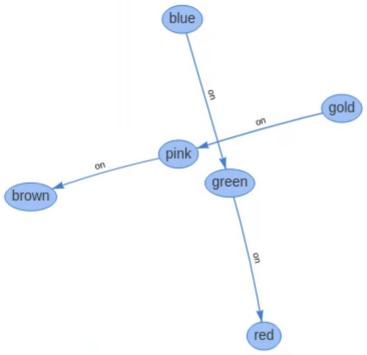

Figura 4: Objetos en el problema

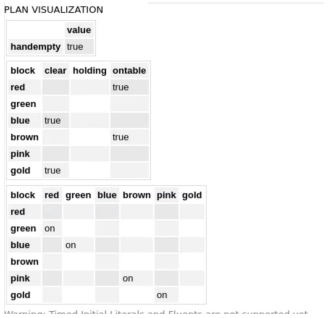

Figura 5: Visualización del plan

Usando VSCode, podemos ver el plan de forma gráfica en la figura 6.

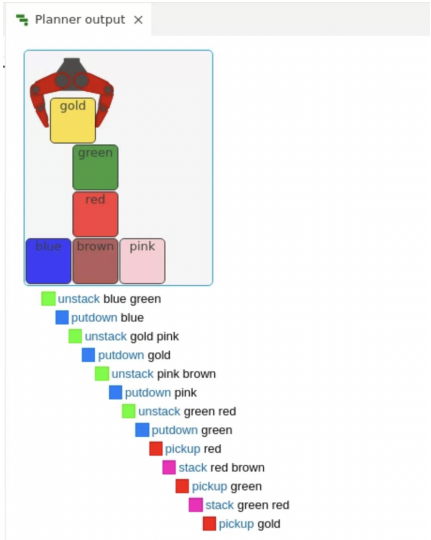

Figura 6: Plan encontrado usando el planner por defecto

Para probar cada uno de los planeadores, de acuerdo a la documentación, copiamos los archivos pddl a un folder, nombrándolos domain y problem respectivamente, definimos los parámetros COSTBOUND, los límites de tiempo, el nombre de la imagen generada y ejecutamos para cada caso:

```
RUNDIR="$(pwd)/rundir"
DOMAIN="$RUNDIR/domain.pddl"
PROBLEM="$RUNDIR/problem.pddl"
PLANFILE="$RUNDIR/nombredelOutput"
COSTBOUND=42 # only in cost-
bounded track
ulimit -t 1800
ulimit -v 8388608
```

A. DecStar – STAR-topology DECoupled Search at its best

```
singularity run -C -H $RUNDIR
  planners/plannerDecstar.img
  $DOMAIN $PROBLEM $PLANFILE $COSTBOUND
```

Plan encontrado:

```
(unstack blue green)
(putdown blue)
(unstack gold pink)
(stack gold blue)
(unstack pink brown)
(putdown pink)
(unstack green red)
(stack green gold)
(pickup red)
(stack red brown)
(unstack green gold)
(stack green red)
(unstack gold blue)
```

Métrica	Resultado
Longitud	13
Coste	13
Estados expandidos	134
Estados evaluados	263
Estados generados	453
Tiempo de busqueda	0.01s
Tiempo total	0.01s
Memoria pico	4660 KB

Cuadro 1: Resultados de DecStar

B. SYMPLE: Symbolic Planning based on EVMDDs

```
singularity run -C -H $RUNDIR
  planners/plannerSymple.img
  $DOMAIN $PROBLEM $PLANFILE $COSTBOUND
```

Plan encontrado:

```
(unstack gold pink)
(putdown gold)
(unstack pink brown)
(putdown pink)
(unstack blue green)
(stack blue pink)
(unstack green red)
(putdown green)
(pickup red)
(stack red brown)
(pickup green)
(stack green red)
(pickup gold)
```

Métrica	Resultado
Longitud	13
Coste	13
Tiempo de busqueda	1.62s
Tiempo total	1.62s
Memoria pico	32272 KB

Cuadro 2: Resultados de Symple

C. Scorpion

```
singularity run -C -H $RUNDIR
  planners/plannerScorpio.img
  $DOMAIN $PROBLEM $PLANFILE $COSTBOUND
```

Plan encontrado

```
unstack blue green (1)
putdown blue (1)
unstack gold pink (1)
stack gold blue (1)
unstack pink brown (1)
putdown pink (1)
unstack green red (1)
stack green pink (1)
pickup red (1)
stack red brown (1)
unstack green pink (1)
stack green red (1)
unstack gold blue (1)
Plan length: 13 step(s).
```

D. Delfi: Online Planner Selection for Cost-Optimal Planning

```
singularity run -C -H $RUNDIR
  planners/plannerDelfi.img
  $DOMAIN $PROBLEM $PLANFILE $COSTBOUND
```

Plan encontrado

Métrica	Resultado
Longitud	13
Coste	13
Estados expandidos	46
Estados evaluados	117
Estados generados	169
Tiempo de busqueda	0s
Tiempo total	204.182s
Memoria pico	21680 KB

Cuadro 3: Resultados de Scorpio

```
unstack gold pink (1)
putdown gold (1)
unstack blue green (1)
putdown blue (1)
unstack pink brown (1)
putdown pink (1)
unstack green red (1)
stack green blue (1)
pickup red (1)
stack red brown (1)
unstack green blue (1)
stack green red (1)
pickup gold (1)
```

Métrica	Resultado
Longitud	13
Coste	13
Estados expandidos	122
Estados evaluados	210
Estados generados	391
Tiempo de busqueda	0.00315251s
Tiempo total	6.06935s
Memoria pico	218460 KB

Cuadro 4: Resultados de Delfi

IV. CONCLUSIONES

Se ha probado el proceso de descarga, instalación y pruebas de 4 algoritmos de planificación participantes de la competencia IPC 2018, elegidos algunos por ser conocidos y un par elegidos al azar. Todos ellos presentaron, afortunadamente, un proceso similar de descarga e instalación lo que permitió volcar la experiencia en el proceso de prueba de funcionamiento y aplicación sobre un problema particular de planificación. Si bien el problema utilizado para probar los algoritmos no era de gran complejidad, permite de buena manera cumplir los objetivos de la experiencia que son: poder instalar un planificador, ejecutarlo y obtener resultados. No está en los objetivos el determinar cual planificador es mejor, ya que eso significaría realizar muchas mas pruebas y con diferentes escenarios de planificación. De todas formas, la aplicación de los 4 algoritmos en un problema particular nos presentó algunos resultados que vale la pena indicar. Los 4 algoritmos lograron el objetivo de encontrar un plan y todos con una ruta con costo de plan en valor 13. Sin embargo, cada uno lo resolvió de su manera y con diferencias en los caminos tomados. De lo que podemos considerar como parámetros y resultados relevantes se pueden comentar los siguientes comportamientos: Decstar expandió 134 estados y lgeneró estados expandidos hasta el ultimo salto en valor 119 , Symple no reporta detalles en sus logs, Scorpion expandió 43 estados y genero estados expandidos hasta el ultimo salto en un número de 40. Por su parte Delfi presenta 122 estados expandidos y expandidos hasta el ultimo salto 84. Según esto el algoritmo Scorpion resolvió el problema en menos acciones y por lo tanto con menos costo computacional.

Lo mas importante de recalcar de esta experiencia es que es posible tener acceso y utilizar, sin costo, distintos planificadores de buen nivel y con distintas lógicas algorítmicas, se ve que esta técnica va en evolución y que cada competencia que se vaya generando traerá nuevas formas de resolver los problemas cada vez mas complejos y que todo ese nuevo conocimiento queda accesible para su uso. No es raro por lo tanto, ante una implementación compleja, aplicar mas de un planificador en la solución e ir comparando según se vayan viendo los resultados.

A. Referencias

Francès, G., Francès, F., Geißer, F., Gnad, D., Haslum, P., Pommerening, F., Ramirez, M., Seipp, J., Sievers, S. (n.d.). Heuristics and Search for Domain-independent Planning (HSDIP).

Gnad, D., Shleyfman, A., Hoffmann, J. (n.d.). DecStar-STAR-topology DECoupled Search at its best.

Katz, M., Sohrabi, S., Samulowitz, H., Sievers, S. (n.d.). Delfi: Online Planner Selection for Cost-Optimal Planning. http://issues.fastdownward. Ma, T., Ferber, P., Huo, S., Chen, J., Katz, M. (2018). Online Planner Selection with Graph Neural Networks and Adaptive Scheduling. http://arxiv.org/abs/1811.00210

Seipp, J. (n.d.). Scorpion. Speck, D., Geißer, F., Mattmüller, R., Mattmüller, M. (n.d.). SYMPLE: Symbolic Planning based on EVMDDs.