

Planificadores del estado del arte

Erick Wilfredo Díaz Saborio

January 4, 2021

1 Planificación

La planificación es una componente muy importante de la inteligencia artificial, en un problema de planificación tenemos un estado inicial y deseamos cumplir una meta por medio de acciones en un plan.

1.1 Competición Internacional de Planificación

El objetivo de la competición internacional de planificación o IPC por sus siglas en inglés, es promover la investigación en planificación auto y evaluar el estado del arte de los planificadores con un determinado número de problemas. La competencia cuenta con cuatro vías clásicas que son: *optimal*, *bounded-cost*, *satisficing*, y *agile*. En este trabajo se seleccionaron los primeros lugares del track *optimal* y *satisficing*.

1.2 Proceso de instalación y ejecución

El proceso de instalación es el mismo para los planificadores que participaron en la competencia IPC 2018. Los pasos descritos a continuación son para un sistema operativo Ubuntu 18.04. Debe-

mos instalar Singularity, para poder ejecutar los planificadores, este software permite crear contenedores donde se empaquetan flujos de software científico. Después de instalar Singularity en nuestra máquina hay que descargar el repositorio del planificador que queremos ejecutar y movernos a la *branch* **ipc-2018-seq-opt** o **ipc-2018-seq-sat**, dependiendo de el track que estamos evaluando. El siguiente paso es crear la imagen con Singularity, utilizando el comando *build* y le enviamos como parámetros el nombre de salida para la imagen que se va a construir y el archivo de receta Singularity. En la imagen se instalarán todas las dependencias del planificador, con la siguiente línea:

```
> sudo singularity build  
    planner.img Singularity
```

Una vez se logra crear la imagen con Singularity, podemos ejecutar el planificador, primero debemos crear una carpeta de ejecución en donde copiaremos los archivos de dominio y el problema, estos archivos se encuentran en otro repositorio (<https://bitbucket.org/ipc2018-classical/domains/src/master/>). En la carpeta el planificador también creará un archivo que contendrá los pasos de el plan.

```
> mkdir rundir
> cp path/to/domain.pddl rundir
> cp path/to/problem.pddl rundir
```

Una vez esta lista la carpeta de ejecución, utilizamos los siguientes comandos para ejecutar el planificador:

```
> RUNDIR="$(pwd)/rundir"
> DOMAIN="$RUNDIR/domain.pddl"
> PROBLEM="$RUNDIR/problem.pddl"
> PLANFILE="$RUNDIR/sas_plan"
> COSTBOUND=42
> ulimit -t 1800
> ulimit -v 8388608

> singularity run -C -H $RUNDIR
  planner.img $DOMAIN $PROBLEM
  $PLANFILE $COSTBOUND
```

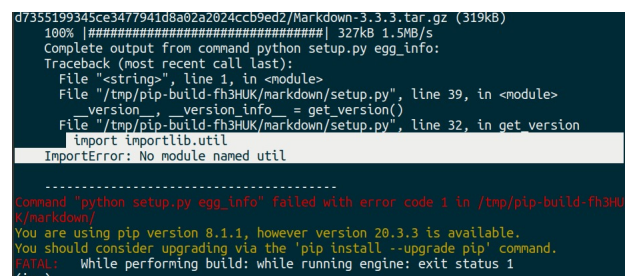
2 Planificador Delfi

Delfi utiliza deep learning para predecir cual de los planificadores dentro de su portafolio tendrá mejor rendimiento en términos de tiempo y memoria para un determinado problema. La idea principal de Delfi se basa en que un solo planificador no va a tener un buen rendimiento en todos los dominios de planificación, por lo que la mejor solución es tener un conjunto o portafolio de planificadores. Inicialmente Delfi se pensó como un problema de regresión, es decir predecir el tiempo que le iba a tomar a determinado planificador completar la tarea, pero debido a ciertos problemas de rendimiento, dada la limitada cantidad de data, se prefirió enfocarlo como un problema de clasificación, para el entrenamiento del modelo se generaron un con-

junto de tareas y todos los planificadores las intentan resolver, de esta forma pueden registrar que planificadores resuelven o no las tareas, esta data es la utilizaron para entrenar el modelo de redes neuronales. La arquitectura de red neuronal seleccionada es muy interesante ya que en su mayoría las redes neuronales convoluciones (CNN) son utilizadas para imágenes. Debido a la limitada cantidad de data con la que contaban se escogió una arquitectura de CNN bastante simple con el objetivo de evitar *overfit*, la arquitectura cuenta con 2 capas convoluciones una de maxpooling luego se aplanan la salida y se aplica dropout como forma de regularización.

2.1 Proceso de instalación y ejecución

Con el proceso de instalación no se pudo completar debido a un error, Figura 1, al momento de generar la imagen con Singularity. Se intento agregar la instalación de la librería en el archivo Singularity pero el error persistió.



```
d7355199345ce3477941d8a02a2024ccb9ed2/Markdown-3.3.3.tar.gz (319kB)
100% |#####| 327kB 1.5MB/s
Complete output from command python setup.py egg_info:
Traceback (most recent call last):
  File "<string>", line 1, in <module>
  File "/tmp/pip-build-fh3HUK/markdown/setup.py", line 39, in <module>
    version, __version_info__ = get_version()
  File "/tmp/pip-build-fh3HUK/markdown/setup.py", line 32, in get_version
    import importlib.util
ImportError: No module named util

-----
Command "python setup.py egg_info" failed with error code 1 in /tmp/pip-build-fh3HUK/markdown/
You are using pip version 8.1.1, however version 20.3.3 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
FATAL: While performing build: while running engine: exit status 1
(fire)
```

Figure 1: Error de instalación con Singularity

3 Planificador Complementary2

El planificador Complementary2 utiliza una base de datos de patrones (PDB), es decir una tabla que contiene las soluciones óptimas de una versión simplificada de la tarea, y utiliza estos datos como heurísticas. El planificador Complementary2 es una implementación de la heurística CPC presentada en (Franco et al., 2017) Complementary2 almacena las distancias óptimas entre las acciones y las metas. La idea de Complementary2 es encontrar un grupo de patrones que minimizan el tiempo de ejecución del algoritmo A*, durante la ejecución se estima también el tiempo de ejecución. En la búsqueda de patrones se utiliza muestreo estratificado o SS por sus siglas en ingles, el muestreo estratificado estima propiedades numéricas en los arboles de búsqueda.

3.1 Proceso de instalación y ejecución

El proceso de instalación y ejecución es el mismo que esta descrito en el punto 1.2.

En la Figura 3 podemos ver el plan resultante de la ejecución, en el dominio agrícola para el problema 1, en el track Optimal, definidos en la competencia.

4 Planificador Fast Downward Stone Soup 2018

Fast Downward Stone Soup es un portafolio de planificadores, basado en el sistema

```
Plan length: 65 step(s).
Plan cost: 786
Expanded 66 state(s).
Reopened 0 state(s).
Evaluated 313 state(s).
Evaluations: 313
Generated 392 state(s).
Dead ends: 247 state(s).
Expanded until last jump: 0 state(s).
Reopened until last jump: 0 state(s).
Evaluated until last jump: 1 state(s).
Generated until last jump: 0 state(s).
Number of registered states: 313
Search time: 0.00171411s
Total time: 32.0772s
Solution found.
Peak memory: 1209232 KB
```

Figure 2: Resultado de ejecución

```
1 (collect_resource worker2 worker1 worker2 round1 act_reed reed)
2 (collect_resource worker1 noworker worker2 round1 act_wood wood)
3 (ag_finish_round_backhome round1 worker2)
4 (ag_finish_round_renew round1 noworker)
5 (ag_advance_round_normal round1 round2 act_sheep)
6 (build_room worker2 worker1 worker2 worker3 round2 room3)
7 (family_growth worker1 noworker worker2 worker3 round2 clay_room3)
8 (ag_finish_round_backhome_withchild round2 worker2 worker3)
9 (ag_finish_round_renew round2 noworker)
10 (ag_advance_round_normal round2 round3 act_sow)
11 (collect_resource worker3 worker2 worker3 round3 act_clay clay)
12 (collect_resource worker2 worker1 worker3 round3 act_reed reed)
13 (plow_field worker1 noworker worker3 round3)
14 (ag_finish_round_backhome round3 worker3)
15 (ag_finish_round_renew round3 noworker)
16 (ag_advance_round_normal round3 round4 act_fences)
17 (take_food worker3 worker2 worker3 round4 num2 num3)
18 (take_grain worker2 worker1 worker3 round4 carrot)
19 (sow worker1 noworker worker3 round4 carrot)
20 (ag_finish_round_backhome round4 worker3)
21 (ag_finish_round_renew round4 noworker)
22 (ag_harvest_collecting_veg round4 stage1 carrot num5 num6)
23 (ag_harvest_collect_end round4 stage1)
24 (ag_harvest_feed round4 stage1 worker3 num6 num6 num0)
25 (ag_harvest_breed_end round4 stage1)
```

Figure 3: Plan

de planificación Fast Downward. Para este portafolio de planificadores utilizaron 66 algoritmos de planificación todos ellos forman parte del portafolio Fast Downward, los cuales participaron en la competencia IPC del año 2014. Por cada uno de los 66 planificadores se agrego uno con la diferencia que usan una lista adicional basada en tipos, por ultimo se agregaron 12 algoritmos mas, siendo estos una variante de configuración, en total el portafolio esta compuesto por 144 configuraciones de planificación. El

portafolio de planificadores se ejecuta en forma secuencial, pero no se cuanta con el tiempo completo de ejecución ya se necesita hacer un preprocesamiento de los datos antes de ejecutar los planificadores

4.1 Proceso de instalación y ejecución

El proceso de instalación y ejecución es el mismo que esta descrito en el punto 1.2.

```
[g=20, 187 evaluated, 44 expanded, t=1.88975s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 22
[g=21, 210 evaluated, 75 expanded, t=1.89014s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 21
[g=22, 211 evaluated, 76 expanded, t=1.89018s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 20
[g=23, 212 evaluated, 77 expanded, t=1.89022s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 19
[g=26, 312 evaluated, 92 expanded, t=1.89158s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 18
[g=27, 313 evaluated, 93 expanded, t=1.89163s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 17
[g=28, 314 evaluated, 94 expanded, t=1.89168s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 16
[g=31, 430 evaluated, 110 expanded, t=1.89413s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 15
[g=32, 431 evaluated, 111 expanded, t=1.8942s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 14
[g=33, 432 evaluated, 112 expanded, t=1.89426s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 13
[g=36, 550 evaluated, 128 expanded, t=1.89675s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 12
[g=37, 551 evaluated, 129 expanded, t=1.89678s, 95808 KB]
New best heuristic value for lmcoun(lmg, admissible = true): 11
[g=38, 552 evaluated, 130 expanded, t=1.89682s, 95808 KB]
Peak memory: 95808 KB
caught signal 24 -- exiting
exitcode: -24

remaining time: -0.23
config 0: relative time 88, remaining 429
```

Figure 4: Resultado de ejecución

En la Figura 5 podemos ver el plan resultante de la ejecución, en el dominio agrícola para el problema 1, en el track satisficing definidos en la competencia. En el track satisficing este planificador utiliza el valor obtenido de el costo asociado a un plan para acortar la búsqueda de los siguientes planificadores a evaluar.

```
25 2
26 Atom open_action(act_sow)
27 NegatedAtom open_action(act_sow)
28 end_variable
29 begin_variable
30 var2
31 -1
32 2
33 Atom open_action(act_fences)
34 NegatedAtom open_action(act_fences)
35 end_variable
36 begin_variable
37 var5
38 -1
39 2
40 Atom harvest_phase(stagel, harvest_init)
41 NegatedAtom harvest_phase(stagel, harvest_init)
42 end_variable
43 begin_variable
44 var1
45 -1
46 2
47 Atom open_action(act_cattle)
48 NegatedAtom open_action(act_cattle)
49 end_variable
50 begin_variable
```

Figure 5: Plan resultante

5 Planificador DecStar

Este planificador utiliza una técnica nueva, llamada Star-Topology Decoupling (STD). Esta técnica explota la interdependencia que existe entre los planificadores de un portafolio para reducir el tamaño de la representación de estados. Utiliza una topología estrella donde separa en cada hoja de la estrella secuencias de acciones, esta descomposición reduce de forma exponencial el espacio de búsqueda. La principal ventaja que tiene la forma de búsqueda desacoplada en comparación a una búsqueda normal, es que un estado desacoplado corresponde a un estado final de la secuencia de estados a partir de la acción central, un estado desacoplado puede ser capaz de representar exponencialmente muchos estados explícitos. DecStar también introduce un método de optimización para evitar duplicidad en la topología estrella, este método se llama *frointier pruning* a diferencia de otros métodos para eliminar duplicados que comparan si dos estados de-

sacoplados tienen el mismo centro y los estados alcanzables son los mismos generando un lato costo computacional, *frointier pruning* compara solo una muestra de los estados alcanzados.

5.1 Proceso de instalación y ejecución

El proceso de instalación y ejecución es el mismo que esta descrito en el punto 1.2.

```
remaining time: 188.86
config 6: relative time 180, remaining 180
args: ['/planner/src/search/downward-release', '--search', 'astar(blind)', '--inte
nal-plan-file', '/home/erick/Documents/EBDIE_razonamiento_planificacion_auto/acti
vidad_1/team2/rundir/sas_plan']
timeout: 188.86 -> (189, 190)
reading input... [t=0s]
simplifying transitions... done!
done reading input! [t=0.12s]
building causal graph...done! [t=0.12s]
Variables: 90
Bytes per state: 16
done! [t=0.12s]
done initializing global data [t=0.12s]
conducting best first search with reopening closed nodes, (real) bound = 214748364
7
initializing blind search heuristic...
F = 1 [1 evaluated, 0 expanded, t=0.12s, 16824 KB]
Best heuristic value: 1 [g=0, 1 evaluated, 0 expanded, t=0.12s, 16824 KB]
F = 61 [9 evaluated, 1 expanded, t=0.12s, 16824 KB]
F = 121 [36 evaluated, 9 expanded, t=0.12s, 16824 KB]
F = 122 [63 evaluated, 36 expanded, t=0.12s, 16824 KB]
F = 123 [90 evaluated, 63 expanded, t=0.12s, 16824 KB]
F = 124 [117 evaluated, 90 expanded, t=0.12s, 16824 KB]
F = 184 [322 evaluated, 117 expanded, t=0.12s, 16824 KB]
F = 244 [937 evaluated, 322 expanded, t=0.12s, 16824 KB]
F = 245 [1552 evaluated, 937 expanded, t=0.13s, 16824 KB]
F = 246 [1737 evaluated, 1552 expanded, t=0.13s, 16824 KB]
F = 247 [1922 evaluated, 1737 expanded, t=0.13s, 16824 KB]
F = 277 [3098 evaluated, 1922 expanded, t=0.13s, 16824 KB]
F = 307 [3152 evaluated, 1938 expanded, t=0.13s, 16824 KB]
F = 337 [6773 evaluated, 3152 expanded, t=0.14s, 16824 KB]
F = 338 [6879 evaluated, 3258 expanded, t=0.14s, 16824 KB]
F = 339 [6985 evaluated, 3364 expanded, t=0.14s, 16824 KB]
F = 340 [7091 evaluated, 3470 expanded, t=0.14s, 16824 KB]
F = 367 [8217 evaluated, 3576 expanded, t=0.14s, 16824 KB]
F = 368 [11732 evaluated, 7091 expanded, t=0.15s, 16824 KB]
F = 369 [12525 evaluated, 10606 expanded, t=0.16s, 17116 KB]
```

Figure 6: Resultado de ejecución

En la Figura 7 podemos ver el plan resultante de la ejecución, en el dominio agrícola para el problema 1, en el track Optimal definidos en la competencia.

6 Conclusión

A pesar de los diferentes algoritmos utilizados en los planificadores las arquitecturas de los planificadores en las primeras

```
1 begin_version
2 3
3 end_version
4 begin_metric
5 1
6 end_metric
7 90
8 begin_variable
9 var81
10 -1
11 2
12 Atom open_action(act_cattle)
13 NegatedAtom open_action(act_cattle)
14 end_variable
15 begin_variable
16 var83
17 -1
18 2
19 Atom open_action(act_improve)
20 NegatedAtom open_action(act_improve)
21 end_variable
22 begin_variable
23 var40
24 -1
25 2
```

Figure 7: Plan resultante

posiciones de la competencia tienen componentes en común. La mayoría esta compuesto por un portafolio de planificadores y tienen un componente que optimiza la forma de seleccionar el planificador a ejecutar, el que mas interesante me pareció es el caso de Delfi que utiliza una CNN para predecir cual de los planificadores en su portafolio va a completar la tarea en el tiempo esperado, los otros planificadores utilizan diferentes técnicas para realizar esta selección como bases de datos de patrones en común con búsquedas especializadas como es el caso de Complementary2, FastDoward tiene su propio portafolio pero este realiza una ejecución secuencial con los planificadores en su portafolio. Todos los métodos seleccionados tienen en común implementaciones diferentes de *pruning* adaptadas a la arquitectura específica, pero con el mismo objetivo de optimizar las búsquedas.

References

- S. Franco, L. H. S. Lelis, and M. Barley. The complementary2 planner in the ipc 2018. URL www.aaai.org.
- S. Franco, A. Torralba, L. H. S. Lelis, and M. B.- ley. *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. 2017.
- M. Katz, S. Sohrabi, H. Samulowitz, and S. Sievers. Delfi: Online planner selection for cost-optimal planning. URL <https://ipc2018-classical.bitbucket.io/planner-abstracts/teams2324.pdf>.
- J. Seipp and G. Röger. Fast downward stone soup 2018. URL <http://fai.cs.uni-saarland.de/hoffmann/>.