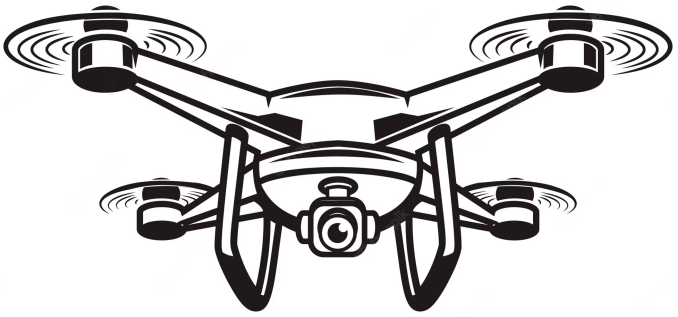
Memoria Planificación Automática



Realizado por:

-Jorge Pérez Pavón

-Adrián Espino Martínez

-Jorge Revenga Martín de Vidales

Universidad De Alcalá De Henares

26 de Marzo de 2023

Índice

Parte 1

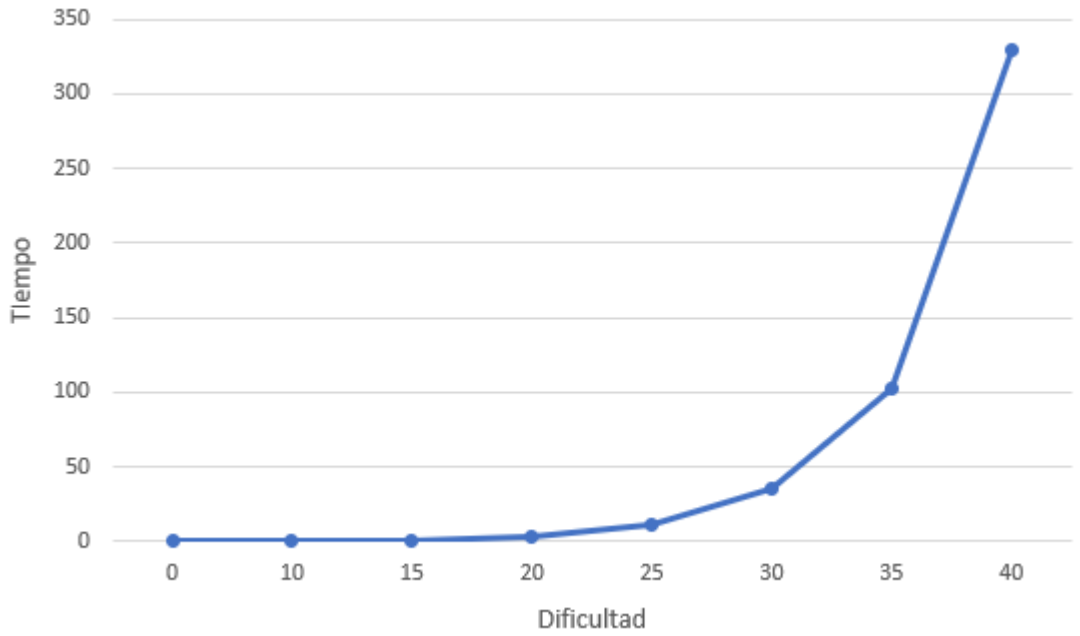
Ejercicio 1.1: Logística de servicio de emergencias, versión inicial

```
(define (domain dron)
    (:requirements :strips :typing)
    Show hierarchy
    (:types
     dron persona location contenido caja brazo - object
    )
    (:predicates
        (esta-dron ?d - dron ?l - location)
        (esta-caja ?c - caja ?l - location)
        (esta-persona ?p - persona ?l - location)
        (libre ?b - brazo)
        (lleva ?d - dron ?c - caja ?b - brazo)
        (consigue ?p - persona ?con - contenido)
        (almacena ?c - caja ?con - contenido)
    )
    (:action volar
       :parameters  (?from - location ?to - location ?d - dron)
       :precondition (esta-dron ?d ?from)
       :effect (and
           (esta-dron ?d ?to)
           (not (esta-dron ?d ?from)))
    )
    (:action coger
       :parameters  (?d - dron ?b - brazo ?c - caja ?l - location ?con - contenido)
       :precondition (and
           (esta-dron ?d ?l)
           (esta-caja ?c ?l)
           (almacena ?c ?con)
           (libre ?b))
       :effect (and
           (lleva ?d ?c ?b)
           (not (esta-caja ?c ?l))
           (not (libre ?b)))
    )
    (:action soltar
       :parameters  (?d - dron ?b - brazo ?c - caja ?l - location ?p - persona ?con - contenido)
       :precondition (and
           (esta-dron ?d ?l)
           (esta-persona ?p ?l)
           (lleva ?d ?c ?b)
           )
       :effect (and
           (consigue ?p ?con)
           (libre ?b)
           (not (lleva ?d ?c ?b))
           )
    )
)
```

Ejercicio 1.2: Generador de problemas en Python

El problema máximo que es capaz de resolver con límite de 10 segundos es el 24 (captura en el siguiente apartado) y con límite de 5 minutos es el 39, en el 40 ya se pasa.

Planificador FF									
Dificultad	**0**	**10**	**15**	**20**	**25**	**30**	**35**	**39**	**40**
Tiempo	0	0,1	0,67	3,27	11,34	35,14	102,69	220,13	329,11



Ejercicio 1.3: Comparativa rendimiento planificadores

FF

```
jorge@LenovoLEGION:~/Documentos/PA$ ./ff -o dominio.pddl -f problema_24.pddl

ff: parsing domain file
domain 'DRON' defined
 ... done.
ff: parsing problem file
problem 'PROBLEMA_24' defined
 ... done.



Cueing down from goal distance:   39 into depth [1]
                                  38            [1][2][3]
                                  37            [1]

Enforced Hill-climbing failed !
switching to Best-first Search now.
```

```
 0: COGER DRON1 BRAZO2 C1 DEPOSITO MEDICINA
 1: COGER DRON1 BRAZO1 C2 DEPOSITO MEDICINA
 2: VOLAR DEPOSITO L24 DRON1
 3: SOLTAR DRON1 BRAZO2 C1 L24 P6 MEDICINA
 4: SOLTAR DRON1 BRAZO1 C2 L24 P6 COMIDA
 5: VOLAR L24 DEPOSITO DRON1
 6: COGER DRON1 BRAZO2 C3 DEPOSITO MEDICINA
 7: COGER DRON1 BRAZO1 C4 DEPOSITO MEDICINA
 8: VOLAR DEPOSITO L24 DRON1
 9: SOLTAR DRON1 BRAZO2 C3 L24 P8 MEDICINA
10: SOLTAR DRON1 BRAZO1 C4 L24 P8 COMIDA
11: VOLAR L24 DEPOSITO DRON1
12: COGER DRON1 BRAZO2 C5 DEPOSITO MEDICINA
13: COGER DRON1 BRAZO1 C6 DEPOSITO MEDICINA
14: VOLAR DEPOSITO L23 DRON1
15: SOLTAR DRON1 BRAZO2 C5 L23 P4 COMIDA
16: VOLAR L23 L21 DRON1
17: SOLTAR DRON1 BRAZO1 C6 L21 P16 COMIDA
18: VOLAR L21 DEPOSITO DRON1
19: COGER DRON1 BRAZO2 C7 DEPOSITO MEDICINA
20: COGER DRON1 BRAZO1 C8 DEPOSITO COMIDA
21: VOLAR DEPOSITO L20 DRON1
22: SOLTAR DRON1 BRAZO2 C7 L20 P2 COMIDA
23: SOLTAR DRON1 BRAZO1 C8 L20 P22 COMIDA
24: VOLAR L20 DEPOSITO DRON1
25: COGER DRON1 BRAZO2 C9 DEPOSITO COMIDA
26: COGER DRON1 BRAZO1 C10 DEPOSITO MEDICINA
27: VOLAR DEPOSITO L18 DRON1
28: SOLTAR DRON1 BRAZO2 C9 L18 P13 COMIDA
29: VOLAR L18 L10 DRON1
30: SOLTAR DRON1 BRAZO1 C10 L10 P21 COMIDA
31: VOLAR L10 DEPOSITO DRON1
32: COGER DRON1 BRAZO2 C11 DEPOSITO MEDICINA
33: COGER DRON1 BRAZO1 C12 DEPOSITO COMIDA
34: VOLAR DEPOSITO L16 DRON1
35: SOLTAR DRON1 BRAZO2 C11 L16 P7 MEDICINA
36: SOLTAR DRON1 BRAZO1 C12 L16 P7 COMIDA
37: VOLAR L16 DEPOSITO DRON1
```

```
38: COGER DRON1 BRAZO2 C13 DEPOSITO COMIDA
39: COGER DRON1 BRAZO1 C14 DEPOSITO COMIDA
40: VOLAR DEPOSITO L15 DRON1
41: SOLTAR DRON1 BRAZO2 C13 L15 P1 COMIDA
42: SOLTAR DRON1 BRAZO1 C14 L15 P11 MEDICINA
43: VOLAR L15 DEPOSITO DRON1
44: COGER DRON1 BRAZO2 C15 DEPOSITO COMIDA
45: COGER DRON1 BRAZO1 C16 DEPOSITO COMIDA
46: VOLAR DEPOSITO L9 DRON1
47: SOLTAR DRON1 BRAZO2 C15 L9 P3 COMIDA
48: SOLTAR DRON1 BRAZO1 C16 L9 P17 MEDICINA
49: VOLAR L9 DEPOSITO DRON1
50: COGER DRON1 BRAZO2 C17 DEPOSITO COMIDA
51: COGER DRON1 BRAZO1 C18 DEPOSITO MEDICINA
52: VOLAR DEPOSITO L7 DRON1
53: SOLTAR DRON1 BRAZO2 C17 L7 P12 COMIDA
54: VOLAR L7 L6 DRON1
55: SOLTAR DRON1 BRAZO1 C18 L6 P9 COMIDA
56: VOLAR L6 DEPOSITO DRON1
57: COGER DRON1 BRAZO2 C19 DEPOSITO COMIDA
58: COGER DRON1 BRAZO1 C20 DEPOSITO COMIDA
59: VOLAR DEPOSITO L5 DRON1
60: SOLTAR DRON1 BRAZO2 C19 L5 P20 MEDICINA
61: VOLAR L5 L2 DRON1
62: SOLTAR DRON1 BRAZO1 C20 L2 P19 COMIDA
63: VOLAR L2 DEPOSITO DRON1
64: COGER DRON1 BRAZO2 C21 DEPOSITO MEDICINA
65: COGER DRON1 BRAZO1 C22 DEPOSITO COMIDA
66: VOLAR DEPOSITO L4 DRON1
67: SOLTAR DRON1 BRAZO2 C21 L4 P15 COMIDA
68: SOLTAR DRON1 BRAZO1 C22 L4 P23 COMIDA
69: VOLAR L4 DEPOSITO DRON1
70: COGER DRON1 BRAZO2 C23 DEPOSITO COMIDA
71: COGER DRON1 BRAZO1 C24 DEPOSITO COMIDA
72: VOLAR DEPOSITO L4 DRON1
73: SOLTAR DRON1 BRAZO2 C23 L4 P24 MEDICINA
74: SOLTAR DRON1 BRAZO1 C24 L4 P24 COMIDA
75: VOLAR L4 DEPOSITO DRON1
```

```
time spent:    0.00 seconds instantiating 4129 easy, 0 hard action templates
               0.00 seconds reachability analysis, yielding 147 facts and 2977 actions
               0.00 seconds creating final representation with 147 relevant facts
               0.00 seconds building connectivity graph
               9.48 seconds searching, evaluating 76624 states, to a max depth of 3
               9.48 seconds total time
```

LPG-TD

```
jorge@LenovoLEGION:~/Documentos/PA$ ./lpg-td -o dominio.pddl -f problema_81.pddl -n 1


NUMERIC_THREATS_MODE: 0

; Command line: ./lpg-td -o dominio.pddl -f problema_81.pddl -n 1


Parsing domain file:  domain 'DRON' defined ... done.
Parsing problem file:  problem 'PROBLEMA_81' defined ... done.



Modality: Incremental Planner

Number of actions            :   33130
Number of conditional actions :       0
Number of facts              :     489


Analyzing Planning Problem:
        Temporal Planning Problem: NO
        Numeric Planning Problem: NO
        Problem with Timed Initial Literals: NO
        Problem with Derived Predicates: NO

Evaluation function weights:
     Action duration 0.00; Action cost 1.00


Computing mutex... done

Preprocessing total time: 2.80 seconds

Searching ('.' = every 50 search steps):
.......... solution found:
 first_solution_cpu_time: 9.10
```

```
Plan computed:
   Time: (ACTION) [action Duration; action Cost]
 0.0000: (COGER DRON1 BRAZO2 C21 DEPOSITO COMIDA) [D:1.00; C:1.00]
 1.0000: (VOLAR DEPOSITO L30 DRON1) [D:1.00; C:1.00]
 2.0000: (VOLAR L30 L63 DRON1) [D:1.00; C:1.00]
 3.0000: (SOLTAR DRON1 BRAZO2 C21 L63 P2 MEDICINA) [D:1.00; C:1.00]
 4.0000: (VOLAR L63 DEPOSITO DRON1) [D:1.00; C:1.00]
 5.0000: (COGER DRON1 BRAZO2 C30 DEPOSITO MEDICINA) [D:1.00; C:1.00]
 6.0000: (VOLAR DEPOSITO L8 DRON1) [D:1.00; C:1.00]
 7.0000: (SOLTAR DRON1 BRAZO2 C30 L8 P3 MEDICINA) [D:1.00; C:1.00]
 8.0000: (VOLAR L8 DEPOSITO DRON1) [D:1.00; C:1.00]
 9.0000: (COGER DRON1 BRAZO2 C69 DEPOSITO MEDICINA) [D:1.00; C:1.00]
 10.0000: (VOLAR DEPOSITO L4 DRON1) [D:1.00; C:1.00]
 11.0000: (SOLTAR DRON1 BRAZO2 C69 L4 P4 MEDICINA) [D:1.00; C:1.00]
 12.0000: (VOLAR L4 DEPOSITO DRON1) [D:1.00; C:1.00]
 13.0000: (COGER DRON1 BRAZO2 C38 DEPOSITO MEDICINA) [D:1.00; C:1.00]
 13.0000: (COGER DRON1 BRAZO1 C27 DEPOSITO MEDICINA) [D:1.00; C:1.00]
 14.0000: (VOLAR DEPOSITO L54 DRON1) [D:1.00; C:1.00]
 15.0000: (VOLAR L54 L43 DRON1) [D:1.00; C:1.00]
 16.0000: (SOLTAR DRON1 BRAZO2 C38 L43 P5 MEDICINA) [D:1.00; C:1.00]
 17.0000: (VOLAR L43 L64 DRON1) [D:1.00; C:1.00]
 18.0000: (SOLTAR DRON1 BRAZO1 C27 L64 P7 MEDICINA) [D:1.00; C:1.00]
 19.0000: (VOLAR L64 DEPOSITO DRON1) [D:1.00; C:1.00]
 20.0000: (COGER DRON1 BRAZO2 C18 DEPOSITO COMIDA) [D:1.00; C:1.00]
 21.0000: (VOLAR DEPOSITO L70 DRON1) [D:1.00; C:1.00]
 22.0000: (SOLTAR DRON1 BRAZO2 C18 L70 P8 MEDICINA) [D:1.00; C:1.00]
 23.0000: (VOLAR L70 DEPOSITO DRON1) [D:1.00; C:1.00]
 24.0000: (COGER DRON1 BRAZO2 C64 DEPOSITO COMIDA) [D:1.00; C:1.00]
 25.0000: (VOLAR DEPOSITO L38 DRON1) [D:1.00; C:1.00]
 26.0000: (SOLTAR DRON1 BRAZO2 C64 L38 P9 MEDICINA) [D:1.00; C:1.00]
 27.0000: (VOLAR L38 DEPOSITO DRON1) [D:1.00; C:1.00]
```

```
Solution number: 1
Total time:      9.10
Search time:     6.30
Actions:         323
Duration:        300.000
Plan quality:    323.000
Total Num Flips: 519
     Plan file:       plan_problema_81.pddl_1.SOL
```

SGPLAN40

```
jorge@LenovoLEGION:~/Documentos/PA$ ./sgplan40 -o dominio.pddl -f problema_62.pddl
#
# Copyright (C) 2004, Board of Trustees of the University of Illinois.
#
# The program is copyrighted by the University of Illinois, and should
# not be distributed without prior approval.  Commercialization of this
# product requires prior licensing from the University of Illinois.
# Commercialization includes the integration of this code in part or
# whole into a product for resale.
#
#-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
# Author: Y. X. Chen, C. W. Hsu, and B. W. Wah, University of Illinois
#-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-


Parsing domain file
 domain 'DRON' defined
Parsing problem file
 problem 'PROBLEMA_62' defined




Starting SGPlan search ...

Solution found.
Answer file is problema_62.pddl.soln
```

```
; Time 9.54
; ParsingTime 0.00
; NrActions 282
; MakeSpan
; MetricValue

        0.010:  (COGER DRON1 BRAZO2 C62 DEPOSITO MEDICINA)[0.000]
        0.020:  (VOLAR DEPOSITO L12 DRON1)[0.000]
        0.030:  (SOLTAR DRON1 BRAZO2 C62 L12 P1 MEDICINA)[0.000]
        0.040:  (VOLAR L12 L6 DRON1)[0.000]
        0.050:  (VOLAR L6 DEPOSITO DRON1)[0.000]
        0.060:  (COGER DRON1 BRAZO2 C61 DEPOSITO MEDICINA)[0.000]
        0.070:  (VOLAR DEPOSITO L40 DRON1)[0.000]
        0.080:  (SOLTAR DRON1 BRAZO2 C61 L40 P11 MEDICINA)[0.000]
        0.090:  (VOLAR L40 L6 DRON1)[0.000]
        0.100:  (VOLAR L6 DEPOSITO DRON1)[0.000]
        0.110:  (COGER DRON1 BRAZO2 C60 DEPOSITO COMIDA)[0.000]
        0.120:  (VOLAR DEPOSITO L15 DRON1)[0.000]
        0.130:  (SOLTAR DRON1 BRAZO2 C60 L15 P12 MEDICINA)[0.000]
        0.140:  (VOLAR L15 L6 DRON1)[0.000]
        0.150:  (VOLAR L6 DEPOSITO DRON1)[0.000]
        0.160:  (COGER DRON1 BRAZO2 C59 DEPOSITO COMIDA)[0.000]
        0.170:  (VOLAR DEPOSITO L7 DRON1)[0.000]
        0.180:  (SOLTAR DRON1 BRAZO2 C59 L7 P16 MEDICINA)[0.000]
        0.190:  (VOLAR L7 L6 DRON1)[0.000]
        0.200:  (VOLAR L6 DEPOSITO DRON1)[0.000]
        0.210:  (COGER DRON1 BRAZO2 C58 DEPOSITO MEDICINA)[0.000]
        0.220:  (VOLAR DEPOSITO L50 DRON1)[0.000]
        0.230:  (SOLTAR DRON1 BRAZO2 C58 L50 P18 COMIDA)[0.000]
        0.240:  (VOLAR L50 L6 DRON1)[0.000]
        0.250:  (VOLAR L6 DEPOSITO DRON1)[0.000]
        0.260:  (COGER DRON1 BRAZO2 C57 DEPOSITO COMIDA)[0.000]
        0.270:  (VOLAR DEPOSITO L36 DRON1)[0.000]
```

SATPLAN

```
jorge@LenovoLEGION:~/Documentos/PA/satplan$ ./satplan -solver siege -domain dominio.pddl -problem problema_5.pddl
---SatPlan Version: 1.1

bb: parsing domain file
domain 'DRON' defined
 ... done.
bb: parsing problem file
problem 'PROBLEMA_5' defined
 ... done.


time:   0,     8 facts and       0 exclusive pairs
             24 ops   and     126 exclusive pairs
time:   1,    23 facts and     110 exclusive pairs

creating thin gp-based encoding...
plan graph layer 0...
plan graph layer 1...
goal constraints...
```

```
; Time 5.34
; ParsingTime    0.00
; MakeSpan 14
0: (COGER DRON1 BRAZO1 C1 DEPOSITO MEDICINA) [1]
0: (COGER DRON1 BRAZO2 C2 DEPOSITO MEDICINA) [1]
1: (VOLAR DEPOSITO L5 DRON1) [1]
2: (SOLTAR DRON1 BRAZO1 C1 L5 P1 MEDICINA) [1]
2: (SOLTAR DRON1 BRAZO2 C2 L5 P1 COMIDA) [1]
3: (VOLAR L5 DEPOSITO DRON1) [1]
4: (COGER DRON1 BRAZO2 C4 DEPOSITO COMIDA) [1]
5: (VOLAR DEPOSITO L3 DRON1) [1]
6: (SOLTAR DRON1 BRAZO2 C4 L3 P4 COMIDA) [1]
7: (VOLAR L3 DEPOSITO DRON1) [1]
8: (COGER DRON1 BRAZO2 C3 DEPOSITO MEDICINA) [1]
8: (COGER DRON1 BRAZO1 C5 DEPOSITO MEDICINA) [1]
9: (VOLAR DEPOSITO L4 DRON1) [1]
10: (SOLTAR DRON1 BRAZO1 C5 L4 P5 COMIDA) [1]
11: (VOLAR L4 L1 DRON1) [1]
12: (SOLTAR DRON1 BRAZO2 C3 L1 P3 COMIDA) [1]
13: (VOLAR L1 DEPOSITO DRON1) [1]
```

FastDownward

```
jorge@LenovoLEGION:~/Documentos/PA$ ./downward.sif --alias lama-first ./dominio.pddl ./problema_120.pddl
INFO     Running translator.
INFO     translator stdin: None
INFO     translator time limit: None
INFO     translator memory limit: None
INFO     translator command line string: /usr/bin/python3 /workspace/downward/builds/release/bin/transla
Parsing...
Parsing: [0.010s CPU, 0.004s wall-clock]
Normalizing task... [0.000s CPU, 0.000s wall-clock]
Instantiating...
Generating Datalog program... [0.000s CPU, 0.003s wall-clock]
Normalizing Datalog program...
Normalizing Datalog program: [0.010s CPU, 0.004s wall-clock]
Preparing model... [0.000s CPU, 0.007s wall-clock]
Generated 26 rules.
Computing model... [0.810s CPU, 0.812s wall-clock]
74543 relevant atoms
30849 auxiliary atoms
105392 final queue length
234873 total queue pushes
Completing instantiation... [2.440s CPU, 2.440s wall-clock]
Instantiating: [3.280s CPU, 3.284s wall-clock]
Computing fact groups...
Finding invariants...
15 initial candidates
Finding invariants: [0.040s CPU, 0.037s wall-clock]
Checking invariant weight... [0.000s CPU, 0.001s wall-clock]
Instantiating groups... [0.030s CPU, 0.034s wall-clock]
Collecting mutex groups... [0.000s CPU, 0.000s wall-clock]
Choosing groups...
602 uncovered facts
Choosing groups: [0.000s CPU, 0.001s wall-clock]
Building translation key... [0.000s CPU, 0.002s wall-clock]
Computing fact groups: [0.080s CPU, 0.080s wall-clock]
```

```
coger dron1 brazo2 c100 deposito medicina (1)
volar deposito l98 dron1 (1)
soltar dron1 brazo2 c100 l98 p96 medicina (1)
volar l98 deposito dron1 (1)
coger dron1 brazo2 c10 deposito medicina (1)
volar deposito l99 dron1 (1)
soltar dron1 brazo2 c10 l99 p42 comida (1)
volar l99 deposito dron1 (1)
coger dron1 brazo2 c1 deposito medicina (1)
volar deposito l99 dron1 (1)
soltar dron1 brazo2 c1 l99 p64 comida (1)
volar l99 deposito dron1 (1)
[t=9.3737s, 99656 KB] Plan length: 480 step(s).
[t=9.3737s, 99656 KB] Plan cost: 480
[t=9.3737s, 99656 KB] Expanded 10587 state(s).
[t=9.3737s, 99656 KB] Reopened 0 state(s).
[t=9.3737s, 99656 KB] Evaluated 10588 state(s).
[t=9.3737s, 99656 KB] Evaluations: 21176
[t=9.3737s, 99656 KB] Generated 1311902 state(s).
[t=9.3737s, 99656 KB] Dead ends: 0 state(s).
[t=9.3737s, 99656 KB] Number of registered states: 10588
[t=9.3737s, 99656 KB] Int hash set load factor: 10588/16384 = 0.64624
[t=9.3737s, 99656 KB] Int hash set resizes: 14
[t=9.3737s, 99656 KB] Search time: 5.57779s
[t=9.3737s, 99656 KB] Total time: 9.3737s
Solution found.
Peak memory: 99656 KB
Remove intermediate file output.sas
search exit code: 0
```

Problema de mayor tamaño que pueden resolver en un tiempo límite de 10 segundos entre paréntesis:

FasDownward (128) > LPG-TD (81) > SGPLAN40 (62) > FF (24) > SATPLAN (5)

Tabla de problemas de mayor tamaño que pueden resolver en un límite de 5 minutos:

Planificador	**Problema**	**Tiempo**	**Nº Acciones**	**Plan Óptimo?**
FastDownward	270	270,76	1080	No
LPG-TD	170	259,91	609	No
SGPLAN40	105	245,37	475	No
FF	39	242,78	125	No
SATPLAN	6	38,73	14	Si

El único planificador que busca un plan óptimo es el SATPLAN, los demás te devuelven el primero que encuentran.

Parte 2

Ejercicio 2.1: Servicio de emergencias, transportadores

1. Para este ejercicio hemos añadido 4 predicados nuevos y 3 acciones nuevas, relacionadas con el transportador. Los predicados son: esta-trans, trans-lleva, siguiente, trans-carga. Las acciones son: volar, meter-caja-trans y sacar-caja-trans.

```
(:predicates
    (esta-dron ?d - dron ?l - location) 5👁 1☑ 1☒
    (esta-caja ?c - caja ?l - location) 1👁 1☒
    (esta-persona ?p - persona ?l - location) 1👁
    (esta-trans ?tr - transportador ?l - location) 3👁 1☑ 1☒
    (libre-dron ?d - dron) 3👁 2☑ 2☒
    (lleva-dron ?d - dron ?c - caja) 2👁 2☑ 2☒
    (consigue ?p - persona ?con - contenido) 1☑
    (almacena ?c - caja ?con - contenido) 1👁
    (trans-lleva ?tr - transportador ?c - caja) 1👁 1☑ 1☒
    (siguiente ?n1 ?n2 - num) 2👁
    (trans-carga ?tr - transportador ?n - num) 2👁 2☑ 2☒
)
```

```
(:action volar
    :parameters (?d - dron ?desde ?hasta - location ?tr - transportador)
    :precondition (and
        (libre-dron ?d)
        (esta-dron ?d ?desde)
        (esta-trans ?tr ?desde)
    )
    :effect (and
        (not (esta-trans ?tr ?desde))
        (not (esta-dron ?d ?desde))
        (esta-trans ?tr ?hasta)
        (esta-dron ?d ?hasta)
        )
```

```
(:action meter-caja-trans
    :parameters (?d - dron ?c - caja ?tr - transportador ?l - location ?n1 ?n2 - num)
    :precondition (and
        (esta-dron ?d ?l)
        (esta-trans ?tr ?l)
        (lleva-dron ?d ?c)
        (trans-carga ?tr ?n1)
        (siguiente ?n1 ?n2))
    :effect (and
        (libre-dron ?d)
        (trans-lleva ?tr ?c)
        (not (lleva-dron ?d ?c))
        (trans-carga ?tr ?n2)
        (not (trans-carga ?tr ?n1))
        )
)

(:action sacar-caja-trans
    :parameters (?d - dron ?c - caja ?tr - transportador ?l - location ?n1 ?n2 - num)
    :precondition (and
        (esta-dron ?d ?l)
        (esta-trans ?tr ?l)
        (trans-lleva ?tr ?c)
        (libre-dron ?d)
        (trans-carga ?tr ?n2)
        (siguiente ?n1 ?n2))
    :effect (and
        (not (libre-dron ?d))
        (lleva-dron ?d ?c)
        (not (trans-lleva ?tr ?c))
        (trans-carga ?tr ?n1)
        (not (trans-carga ?tr ?n2))
```

2. En el programa para la generación de problemas hemos añadido lo siguiente:

```
    for x in carrier:
        f.write("\t" + x + " - transportador\n")

    f.write("\t" + 'n0 n1 n2 n3 n4' + " - num\n")
```

La definición de los transportadores y el espacio de almacenamiento de cajas.

```
    for x in carrier:
        f.write("\t(esta-trans " + x + " " + location[0] + ")\n")
        f.write("\t(trans-carga " + x + " " + 'n0' + ")\n")
```

```
    f.write('\t(siguiente n0 n1)\n')
    f.write('\t(siguiente n1 n2)\n')
    f.write('\t(siguiente n2 n3)\n')
    f.write('\t(siguiente n3 n4)\n')
```

La inicialización de los transportadores y de su espacio de almacenamiento.

3. Los distintos planificadores no usan el transportador, ya que al no haber costes, se busca simplemente minimizar las acciones. Sin embargo, si eliminamos la acción mover dron, estaremos forzando a los planificadores a usarlo, y todos serán capaces de hacerlo.

4.

- Con el planificador FF para un tamaño de problema de 21 obtenemos en la versión de los 4 brazos un tiempo de 244 seg (cabe destacar que el planificador solo usa 2 brazos) y en la versión del transportador un tiempo de 3,4 seg.
- Con el planificador LPG-TD para un tamaño de problema de 150 obtenemos en la versión de los 4 brazos un tiempo de 235 seg y en la versión del transportador ocurre un error: *Warning: Problem size too large. Size of the array for the actions exceeded.* Lo que nos da a pensar que con este planificador la versión del transportador es más costosa de planificar.

- Con el planificador SGPLAN40 para un tamaño de problema de 90 obtenemos en la versión de los 4 brazos un tiempo de 270 seg y en la versión del transportador un tiempo de 589 seg.

Con estos datos concluimos que encontrar el plan del transportador es más complejo con los planificadores LPG-TD y SGPLAN40 pero con el FF es mucho más sencillo que en su versión de los brazos.

Ejercicio 2.2: Servicio de emergencias, costes de acción

1. Se ha añadido el requisito :action-costs

```
(:requirements :strips :typing :action-costs)
```

2. Se han añadido las funciones total-cost y fly-cost

```
(:functions
    (total-cost) 6↗
    (fly-cost ?desde - location ?hasta - location) 2👁
)
```

3. El coste total se incrementa en las acciones meter-caja-trans, sacar-caja-contenedor, coger y soltar

```
(:action soltar
   :parameters  (?d - dron ?c - caja ?l - location ?p - persona ?con - contenido)
   :precondition (and
       (esta-dron ?d ?l)
       (esta-persona ?p ?l)
       (lleva-dron ?d ?c)
       )
   :effect (and
       (consigue ?p ?con)
       (libre-dron ?d)
       (not (lleva-dron ?d ?c))
       (increase (total-cost) 1)
       )
```

4. En la acción volar, se incrementa en función del fly-cost

```
(:action volar
    :parameters (?d - dron ?desde ?hasta - location ?tr - transportador)
    :precondition (and
        (libre-dron ?d)
        (esta-dron ?d ?desde)
        (esta-trans ?tr ?desde)
    )
    :effect (and
        (not (esta-trans ?tr ?desde))
        (not (esta-dron ?d ?desde))
        (esta-trans ?tr ?hasta)
        (esta-dron ?d ?hasta)
        )
)
```

5. Inicializamos el coste total a 0 en el generador de problemas

```
f.write('\t(= (total-cost) 0)\n')
```

6. Inicializamos los valores de la función fly-cost

```
for i in range(len(location)):
    for j in range(len(location)):
        if (i == j):
            f.write("\t(= (fly-cost " + location[i] + " " + location[j] + ") 0)\n")
        else:
            f.write("\t(= (fly-cost " + location[i] + " " + location[j] + ") " + str(flight_cost(location_coords,
```

7. Añadimos una métrica para la minimización del coste total

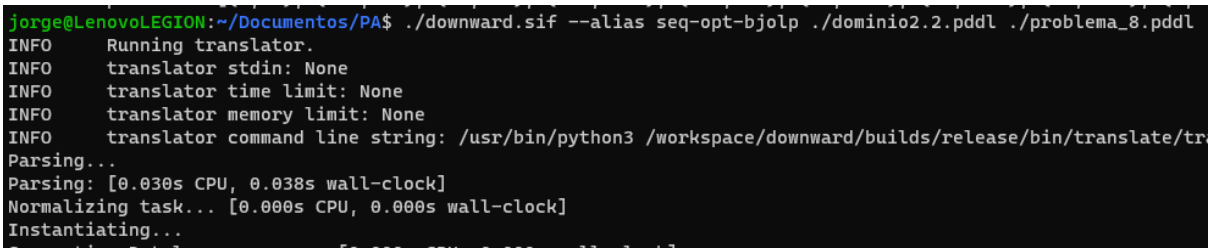
```
f.write("\t))\n")
f.write("(:metric minimize (total-cost))\n")
f.write(")\n")
```

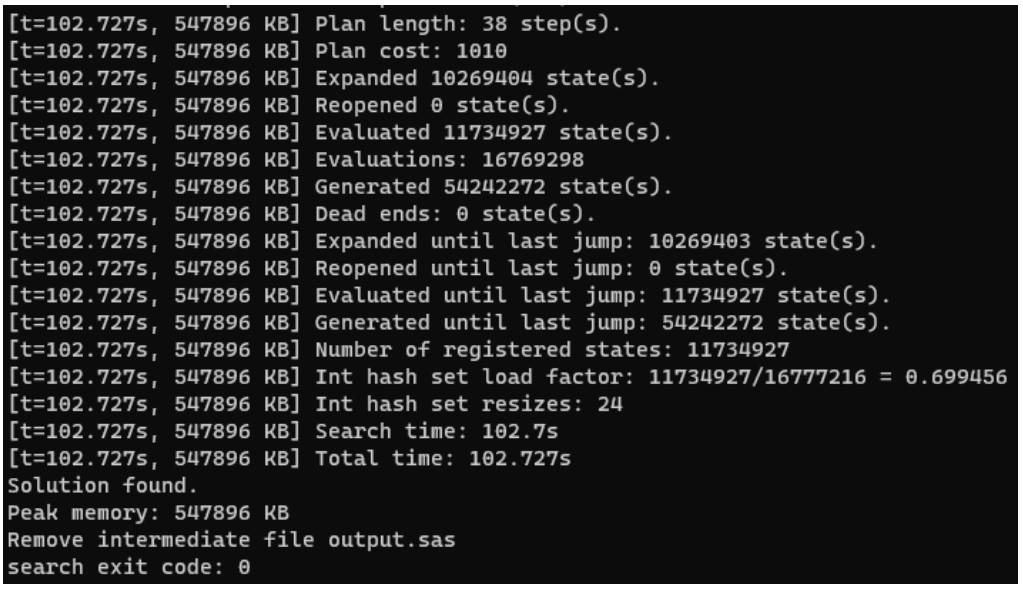
8.

a) El planificador Metric-FF utiliza el algoritmo de búsqueda A*. El parámetro -w se utiliza para introducir el coste total de la solución. Si se le da un valor de 1, el algoritmo se enfocará en la heurística y no en el coste total, lo que puede llevar a encontrar una solución que no es óptima. Si por el contrario se le da un valor muy alto, el planificador se enfocará en buscar una solución que minimice el coste, pero puede aumentar en gran medida el tiempo y el consumo de algunos recursos como la memoria.

b) El planificador LAMA, se basa en la descomposición del problema en subproblemas, y utiliza una heurística basada en landmarks para guiar la búsqueda a la solución óptima. Con la opción –alias lama-first se ejecuta una combinación de búsqueda en anchura y búsqueda heurística, esto hace que la solución se encuentre rápidamente. Con la opción –alias lama se ejecuta únicamente la búsqueda heurística, directamente con la heurística de landmarks.

c) BJOLP es un planificador de planificación lineal, que utiliza una estrategia de búsqueda bidireccional para encontrar una solución óptima. En un tiempo límite de 5 minutos, es capaz de resolver problemas con los parámetros a 8, tardando aproximadamente 102 segundos. A partir de 9, los tiempos incrementan en gran medida. La diferencia con respecto a LAMA es que este planificador si que hace uso del transportador. Por esta razón es preferible usarlo para este problema, además de que su coste no es excesivamente alto comparado con otros.





d) FDSS es un planificador de tipo portfolio que hace uso de diferentes técnicas y heurísticas para alcanzar una solución óptima, en concreto hace uso de una técnica llamada Stone Soup. Un planificador de tipo portfolio significa que hace uso de diferentes algoritmos de búsqueda, estrategias heurísticas y configuraciones de parámetros para abordar el problema de manera más efectiva. Este planificador es capaz de resolver el anterior problema en 0.14 segundos.

```
jorge@LenovoLEGION:~/Documentos/PA$ ./downward.sif --alias seq-opt-fdss-2 ./dominio2.2.pddl ./problema_8.pddl
INFO     Running translator.
INFO     translator stdin: None
INFO     translator time limit: None
INFO     translator memory limit: None
INFO     translator command line string: /usr/bin/python3 /workspace/downward/builds/release/bin/translate/tran
Parsing...
Parsing: [0.000s CPU, 0.002s wall-clock]
```

```
Translator variables: 20
Translator derived variables: 0
Translator facts: 72
Translator goal facts: 9
Translator mutex groups: 8
Translator total mutex groups size: 24
Translator operators: 784
Translator axioms: 0
Translator task size: 6077
Translator peak memory: 44992 KB
Writing output... [0.020s CPU, 0.014s wall-clock]
Done! [0.140s CPU, 0.143s wall-clock]
translate exit code: 0
```

Parte 3

Ejercicio 3.1: Acciones concurrentes en gestión de emergencias

En este apartado vamos a explicar las implicaciones del uso de la concurrencia en el dominio de gestión de emergencia que hemos desarrollado en las partes anteriores de la práctica.

Hemos decidido implementar el siguiente número de acciones que se pueden ejecutar de manera **concurrente**:

- Un transportador puede ser movido por dos drones de manera simultánea a una localización.
- Dos drones pueden volar paralelamente a una localización.
- Varias cajas pueden ser cargadas por dos drones en un solo transportador.
- Paralelamente dos drones pueden moverse, coger y soltar cajas.

Estas son las acciones que hemos decidido que **no** se puedan realizar de manera **concurrente**:

- Más de un transportador no puede ser movido a la vez por un dron.
- Mientras un dron está cargando o descargando cajas en un transportador otro dron no puede mover ese transportador.
- Un humano no puede recibir dos cajas a la vez.
- Una caja no puede ser cogida por dos drones simultáneamente.

Ejercicio 3.2: Implementación y pruebas de concurrencia

Hemos extendido el dominio de gestión de emergencias para crear planes concurrentes mediante el uso de durative actions.

Variaciones y cambios en el dominio

Para crear planes concurrentes mediante el uso de durative actions hemos eliminado los :action-cost y los hemos sustituido por :durative-actions, de esta manera podemos realizar cambios en los requisitos para conseguir nuestro fin.

Además hemos incluido dos nuevos predicados, produciéndose variaciones en las funciones pero manteniendo el fly-cost debido a que eliminamos los costes y total-cost ya no será necesario.
Vamos a tener dos tipos de acciones, al convertirse en acciones :durative-actions, tendremos distintos tipos de acciones en base a su duración.

- Duración fija al coger y soltar una caja
- Duración que variará según el fly-cost al realizar un movimiento

Eliminamos las precondiciones y todas las condiciones que teníamos se van a distribuir entre at start, over all y at end.

Investiga qué tamaño de problema es capaz de resolver el planificador OPTIC en un minuto de tiempo aproximadamente, y cómo afecta al rendimiento del planificador la variación de distintos parámetros como el número de drones o el número de cajas.

Si utilizamos todas las acciones del dominio, el dron podrá llevar cajas sin utilizar el transportador, y los resultados pueden variar a los resultados de forzar el uso del transportador.

Cuando se utilizan más de 4 drones, transportadores, localizaciones, personas y cajas, el planificador no es capaz de encontrar ningún plan. Si aumentamos el número de cajas, el planificador tardará más en encontrar un plan, ya que se tienen que considerar más costes de vuelo para minimizar la duración.

Podemos ver como usando el planificador Optic obtenemos una heurística usando el algoritmo de best-first search. Si borramos la acción de mover el dron conseguimos forzar a que se utilice el transportador, por lo tanto los resultados que obtendremos serán diferentes a los vistos antes.

En el siguiente ejemplo vamos a generar un programa con los parámetros que se muestran en la siguiente imagen, con estos parámetros son con los que más nos acercamos a que el planificador encuentre solución en 1 minuto:

```
jorge@DESKTOP-1FE3C2H:~$ python3 ./generate-problem.py -d 2 -r 2 -l 4 -p 4 -c 4 -g 4
Drones          2
Carriers        2
Locations       4
Persons         4
Crates          4
Goals           4

Types   Quantities
comida   1
medicina        3
Location positions [(0, 0), (150, 142), (149, 63), (199, 60), (70, 172)]
```

Usamos la siguiente planificación usando el planificador optic-clp:

```
jorge@DESKTOP-1FE3C2H:~$ ./optic-clp dominio3.pddl problema_4.pddl
Number of literals: 60
Constructing lookup tables: [10%] [20%] [30%] [40%] [50%] [60%] [70%] [80%] [90%] [100%]
Post filtering unreachable actions:  [10%] [20%] [30%] [40%] [50%] [60%] [70%] [80%] [90%] [100%]
No semaphore facts found, returning
No analytic limits found, not considering limit effects of goal-only operators
58% of the ground temporal actions in this problem are compression-safe
Initial heuristic = 9.000, admissible cost estimate 0.000
b (8.000 | 189.003)b (7.000 | 190.004)b (6.000 | 514.005)b (5.000 | 700.006)b (4.000 | 779.009)b (3.000 | 781.010)b (2.000 | 865.010)b (1.000 | 1026.011)(G)
; No metric specified - using makespan
```

```
; Plan found with metric 746.010
; States evaluated so far: 24367
; States pruned based on pre-heuristic cost lower bound: 4026
; Time 41.69
0.000: (coger dron1 c2 deposito comida)  [1.000]
0.000: (coger dron2 c3 deposito comida)  [1.000]
1.001: (meter-caja-trans dron1 c2 transportador1 deposito n0 n1)  [1.000]
1.001: (meter-caja-trans dron2 c3 transportador2 deposito n0 n1)  [1.000]
2.002: (coger dron2 c1 deposito comida)  [1.000]
2.002: (volar dron1 deposito l2 transportador1)  [162.000]
3.003: (meter-caja-trans dron2 c1 transportador2 deposito n1 n2)  [1.000]
4.004: (volar dron2 deposito l4 transportador2)  [186.000]
164.003: (sacar-caja-trans dron1 c2 transportador1 l2 n0 n1)  [1.000]
165.004: (soltar dron1 c2 l2 p4 medicina)  [1.000]
166.005: (volar dron1 l2 deposito transportador1)  [162.000]
190.005: (sacar-caja-trans dron2 c1 transportador2 l4 n1 n2)  [1.000]
191.006: (soltar dron2 c1 l4 p3 medicina)  [1.000]
192.007: (volar dron2 l4 l3 transportador2)  [171.000]
328.006: (coger dron1 c4 deposito comida)  [1.000]
329.007: (meter-caja-trans dron1 c4 transportador1 deposito n0 n1)  [1.000]
330.008: (volar dron1 deposito l3 transportador1)  [208.000]
538.009: (sacar-caja-trans dron1 c4 transportador1 l3 n0 n1)  [1.000]
538.010: (volar dron2 l3 deposito transportador1)  [208.000]
539.010: (soltar dron1 c4 l3 p1 medicina)  [1.000]
540.011: (volar dron1 l3 l4 transportador2)  [171.000]
711.012: (sacar-caja-trans dron1 c3 transportador2 l4 n0 n1)  [1.000]
712.013: (soltar dron1 c3 l4 p3 comida)  [1.000]

 * All goal deadlines now no later than 746.010
```