

Pràctica 3 IA

Planificació Automàtica

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona



Planificador de reservas de hotel (Extensión 4)

Autors: J.Oriol Ventura, Mario Rodrigo, Víctor Abelló

Data: 03/01/2026

Assignatura: Intel·ligència Artificial

Introducció i Objectius del Projecte.....	3
Modelatge del Domini PDDL.....	3
Gestió de l'Estat i Predicats.....	3
Estratègia d'Assignació: Divisió d'Accions.....	3
Anàlisi de la Mètrica i Justificació dels Pesos.....	4
Justificació Matemàtica.....	4
Experimentació i Resultats.....	4

Introducció i Objectius del Projecte

Aquest document detalla el desenvolupament de l'Extensió 4 per al domini de gestió hotelera. L'objectiu fonamental d'aquesta iteració és dotar el planificador de la capacitat per gestionar l'obertura d'habitacions de manera eficient, introduint criteris d'optimització avançats que van més enllà de la simple assignació.

En concret, es busca minimitzar el nombre total d'habitacions utilitzades, prioritzant la compactació de les reserves. Secundàriament, es vol reduir el desaprofitament de places (waste). Per aconseguir-ho, s'ha establert una jerarquia estricta d'objectius d'optimització: en primer lloc, evitar les cancel·lacions de reserves (n-skipped); en segon lloc, mantenir al mínim les habitacions obertes (n-used); i finalment, ajustar la capacitat de l'habitació al nombre d'hostes (waste).

Modelatge del Domini PDDL

Per implementar els requisits esmentats, s'ha modificat l'arxiu de domini `hotel-ext4-domain.pddl`, introduint nous mecanismes de control de l'estat.

Gestió de l'Estat i Predicats

S'han incorporat noves variables per monitorar l'ús dels recursos. S'ha definit el predicat (`used ?rm - room`) per marcar si una habitació ha estat oberta, juntament amb la funció numèrica (`n-used`), que comptabilitza el total d'habitacions en ús. Addicionalment, la funció (`waste`) acumula la ineficiència de cada assignació (diferència entre capacitat i ocupants).

Estratègia d'Assignació: Divisió d'Accions

Un dels reptes tècnics principals ha estat gestionar el comptador d'habitacions usades sense utilitzar logiques condicionals complexes que no gestionen bé alguns planificadors (`Metric-FF` en aquest cas). Per resoldre això, s'ha optat per una estratègia de desglossament de l'acció d'assignar en dues operacions especialitzades:

1. Assignació en Habitació Nova (`assign-room-new`): Aquesta acció només és aplicable quan l'habitació no ha estat utilitzada prèviament (precondició `not (used ?rm)`). El seu efecte inclou marcar l'habitació com a usada i incrementar el comptador `n-used` en una unitat.
2. Assignació en Habitació Existent (`assign-room-existing`): S'aplica quan l'habitació ja està marcada com a `used`. En aquest cas, es realitza la reserva sense incrementar el comptador d'habitacions, permetent així omplir els buits sense penalització addicional.

Aquesta distinció garanteix que el cost d'obrir una habitació s'apliqui exactament una vegada, simplificant l'arbre de cerca del planificador.

Anàlisi de la Mètrica i Justificació dels Pesos

La funció de cost és l'element clau que guia el comportament del planificador. S'ha dissenyat una funció lineal amb pesos ponderats per simular una optimització jeràrquica:
 $\text{Minimize} = (1.000.000 \times n\text{-skipped}) + (1.000 \times n\text{-used}) + \text{waste}$

Justificació Matemàtica

L'elecció d'aquests valors no és trivial, ja que per assegurar que el sistema respecta l'ordre de prioritats, el cost de violar una prioritat superior ha de ser sempre més gran que el cost màxim acumulat de la prioritat inferior.

Considerant el cost d'obrir una habitació (1.000) i el pitjor cas de waste possible (assignar una persona a una habitació de 4, generant un waste de 3), la condició de validesa es compleix àmpliament. Fins i tot en el problema més complex analitzat (p04-hard amb 30 reserves potencials), el waste màxim teòric seria de 90 (30 reserves \times 3 waste). Com que $1.000 > 90$, el sistema mai obrirà una habitació innecessària només per reduir el desaprofitament, complint així l'especificació.

Experimentació i Resultats

Per validar la correcció i l'escalabilitat de la solució, s'han executat diversos escenaris de prova, des del problema base fins a variants més complexes amb restriccions ajustades. Els problemes de prova (easy, medium, hard) s'han creat utilitzant un generador automàtic desenvolupat en Python expressament per a aquesta pràctica, permetent avaluar l'escalabilitat del sistema.

Els resultats obtinguts (utilitzant el planificador Metric-FF) es resumeixen a continuació:

Problema	Temps (ms)	Longitud del Pla	Cost Total	Skipped	Hab. Usades	Waste
p01-ext4	11.452	2	2002.0	0	2	2
p02-easy	15.224	8	3008.0	0	3	8
p03-medium	27.302	15	5004006.0	5	4	6
p04-hard	138.250	30	3010013.0	3	10	13

Els resultats confirmen el correcte funcionament de la funció de cost jeràrquica:

1. El cost total està dominat clarament pel terme de reserves saltades (pes 10^6), seguit pel nombre d'habitacions usades (pes 10^3) i finalment el waste.
2. El temps d'execució escala de manera raonable amb la complexitat del problema, mantenint-se per sota dels 150ms fins i tot per al cas més complex (Hard).

Nota: Els problemes de prova hard han excedit el consell de unes 200 línies pero sino ens costava veure la progressió real i pel que hem vist no hi ha hagut problema. I poden tardar lleugerament diferent si s'executen de nou.

Nota sobre Advertències del Planificador:

Durant l'execució dels experiments, s'ha observat que el planificador Metric-FF emet advertències del tipus translating negated cond for predicate.... Hem analitzat aquest avís i, atès que el planificador ha estat capaç de trobar solucions vàlides i òptimes per a tots els casos de prova sense errors d'execució, hem decidit mantenir la definició actual del domini, entenent que aquestes advertències són part del procés intern de traducció ADL a STRIPS del planificador.