Planification basée sur un problème SAT

Dinia Adil

Maryam Cherradi

Loïc Bichon

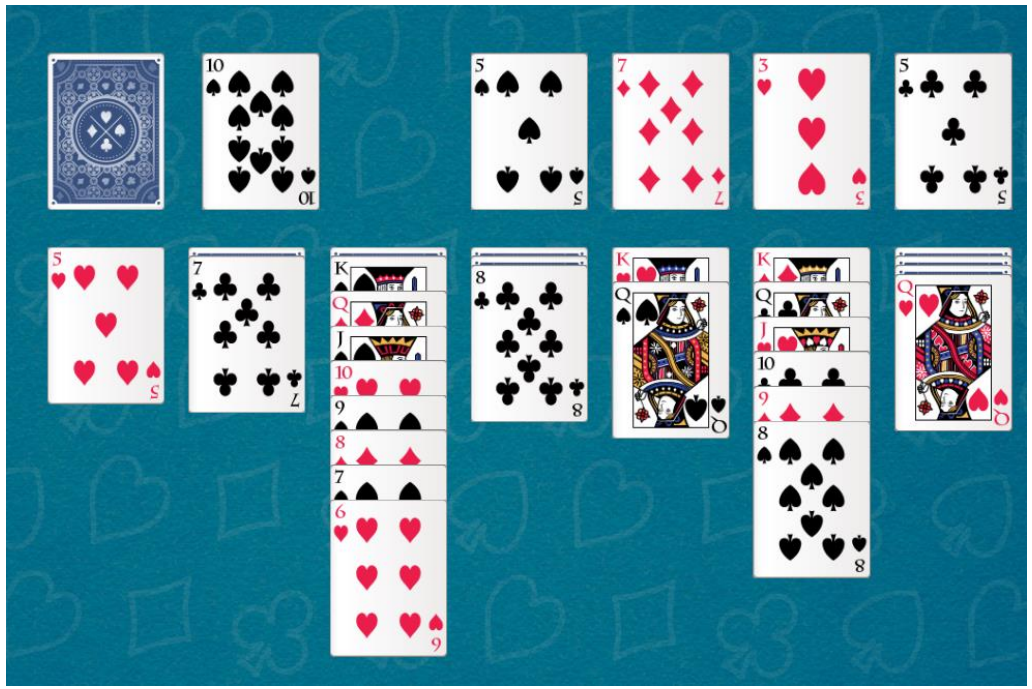
Loïc Maurin

Domaine et problèmes proposés

Jeu de Solitaire

Domaine du Solitaire

- Solitaire avec environnement observable (cartes de la pioche et cartes empilées connues)



- Solitaire avec environnement observable (cartes de la pioche et cartes empilées connues)
- Objets
  - Card : carte
  - Colonne : colonne
- Prédicats :
  - $On(c, x)$: vrai si une carte c est sur une carte ou pile x
  - $Free(c)$: vrai si une carte c est libre
  - $OnDeck(x)$: vrai si une carte c est dans la pioche
  - $Hidden(x)$: vrai si une carte x est dans une pile
  - $Placed(c)$: vrai si une carte c ou une colonne c est placée
  - $canMoveOnTop(c, x)$: contraintes d'empilement des cartes sur les rangées
  - $canPlaceOnTop(c, x)$: contraintes de placement des cartes dans l'ordre sur les piles finales
- Actions :
  - $fromDeckToBase$: piocher une carte
  - $fromBaseToBase$: déplacer une carte dans les rangées
  - $placeFromBase$: empiler une carte depuis la pioche
  - $placeFromDeck$: empiler une carte depuis les rangées

Problèmes de tailles variables

- Nombre de cartes :
  - 3, 4, 5, 7 cartes par couleur
- Nombre de couleurs :
  - 1 couleur
  - 2 couleurs (rouge/ noir)
- Règles sur les couleurs :
  - Empilement rouge noir uniquement
  - Empilement noir/noir rouge/rouge : génère des déplacements autorisés supplémentaires mais relaxe le problème
- Pistes d'améliorations :
  - On pourrait alors introduire des prédicats de couleurs (au sens des jeux de cartes) et des prédicats de succession dans l'ordre des cartes pour obtenir une représentation plus compacte (en particulier dans le cas de la transcription en problème SAT)

Encodage et résolution SAT

Recherche de plan

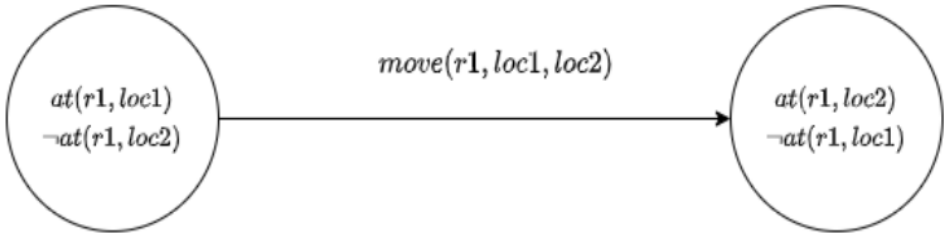
Principe

1 Instance PDDL

Etat initial (objets et prédicats)

Préconditions -> Action -> Effet

Etat but



2 Conversion au format CNF

Une clause : prédicat ou action sur objets (positive ou négative, possibilité de mettre un OR)

Toutes les clauses (ET) : tout ce que l'on sait ou que l'on peut déduire

Ajout d'une variable temps t

$$at(r1, loc1, 0) \wedge \neg at(r1, loc2, 0) \wedge move(r1, loc1, loc2, 0) \wedge at(r1, loc2, 1) \wedge \neg at(r1, loc1, 1)$$

3 Résolution SAT

4 Conversion en plan

Quelle action mise à *True* au pas de temps *t* ?

(une seule grâce à l'axiome d'exclusion mutluelle)

But atteint avec le plan trouvé ?

Non

Oui

Etape 2 avec

t ← t+1

On cherche un plan de longueur supérieure

☺

PDDL Parser

```
Domain name: simple
action: move
  parameters: [['?r', 'robot'], ['?from', 'location'], ['?to', 'location']]
  positive_preconditions: [['atl', '?r', '?from'], ['adjacent', '?from', '?to']]
  negative_preconditions: []
  add_effects: [['atl', '?r', '?to']]
  del_effects: [['atl', '?r', '?from']]

---------------------------
Problem name: simple-problem
Objects: {'robot': ['rob'], 'location': ['loc1', 'loc2']}
State: frozenset({('atl', 'rob', 'loc1'), ('adjacent', 'loc2', 'loc1'), ('adjacent', 'loc1', 'loc2')})
Positive goals: frozenset({('atl', 'rob', 'loc2')})
Negative goals: frozenset()
```

https://github.com/pucrs-automated-planning/pddl-parser

Encodage au format SAT

1

$$\bigwedge_{f \in s_0} f_0 \wedge \bigwedge_{f \notin s_0} \neg f_0$$

Building Initial States (Image by Author)

2

$$\bigwedge_{f \in g^+} f_n \wedge \bigwedge_{f \in g^-} f_n$$

Building Goal States (Image by Author)

3

$$a_i \Rightarrow \left(\bigwedge_{p \in precond(a)} p_i \wedge \bigwedge_{e \in effects(a)} e_{i+1} \right)$$

Building Actions (Image by Author)

4

$$\neg f_i \wedge f_{i+1} \Rightarrow \left(\bigvee_{a \in A | f_i \in effects^+(a)} a_i \right) \wedge$$

$$f_i \wedge \neg f_{i+1} \Rightarrow \left(\bigvee_{a \in A | f_i \in effects^-(a)} a_i \right)$$

Building Explanatory Frame Axioms (Image by Author)

5

$$\neg a_i \vee \neg b_i$$

Building Complete Exclusion Axiom (Image by Author)

```
proposition_formulas = init_state_clauses + goal_state_clauses + actions_clauses + explanatory_frame_axioms + complete_exclusion_axiom
```

Author = Debby Nirwan
https://towardsdatascience.com/ai-planning-as-satisfiability-with-davis-putnam-algorithm-53f05d2f2825https://github.com/debbynirwan/planning_sat/blob/main/domain/simple-problem.pddl

Optimisations mises en place

- Afin de réduire le nombre de variables dans les formules CNF à satisfaire, nous avons décidé éliminé les répétitions dans les paramètres des ground actions et des prédicats

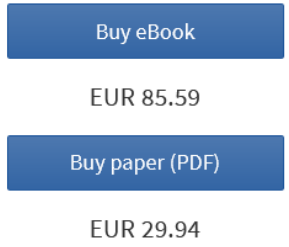
- Rajout optionnel de l'utilisateur de prédicats inchangeables par des actions: cela nous permet d'ignorer ces prédicats dans les axiomes du cadre, et rajouter des clauses spécifiants que ces prédicats restent dans le même état que l'état initial

Résolution SAT

Solveur **Cryptominisat**

Principe :

SAT solveur adapté pour les problème de cryptographie



⇒ Mystère

Fonctionnement :

```
solver = pycryptosat.Solver()
solver.add_clause([1, -5, 4])
solver.add_clause([-1, 5, 3, 4])
solver.add_clause([-3, -4])
solver.solve()
(True, (None, True, False, False, True, True))
```

a, b, c, d, e

a ou ~e ou d

~a ou e ou c ou d

~c ou ~d

[a ou ~e ou d] et [a ou ~e ou d] et [~a ou e ou c ou d] et [~c ou ~d]

a et ~b et ~c et d et e

Problème satisfiable ☺

https://github.com/msoos/cryptominisat

Extending SAT Solvers to Cryptographic Problems 2009; Soos, Mate; Nohl, Karsten; Castelluccia, Claude

Décodage du plan

Incrément de t jusqu'à ce que le but soit atteint i.e. problème satisfiable

↓

Extraction des clauses correspondant à des actions parmi les clauses `True` de l'output du solver SAT

↓

Ordonner par temps

↓

Plan

$$at(r1, loc1, 0) \wedge \neg at(r1, loc2, 0) \wedge move(r1, loc1, loc2, 0) \wedge at(r1, loc2, 1) \wedge \neg at(r1, loc1, 1)$$

Plan :

- move r1 de loc1 à loc 2 au pas de temps 0
- Fin car but atteint

Performances du solveur

Instance	Robot	4 cartes 1 couleur	5 cartes 1 couleur	4 cartes 1 couleur + cartes cachée	3 cartes 2 couleurs	52 cartes 4 couleurs
Temps d'exécution (s)	0.72	5.17	17.03	153.7	151.9	crash
Taille du plan	1	5	5	8	6	💀
Optimal ?	☺	☺	☺	☺	☺	💀

Conclusion:
Limites et améliorations possibles

$$\neg a_i \vee \neg b_i$$

Building Complete Exclusion Axiom (Image by Author)

Explosion combinatoire =

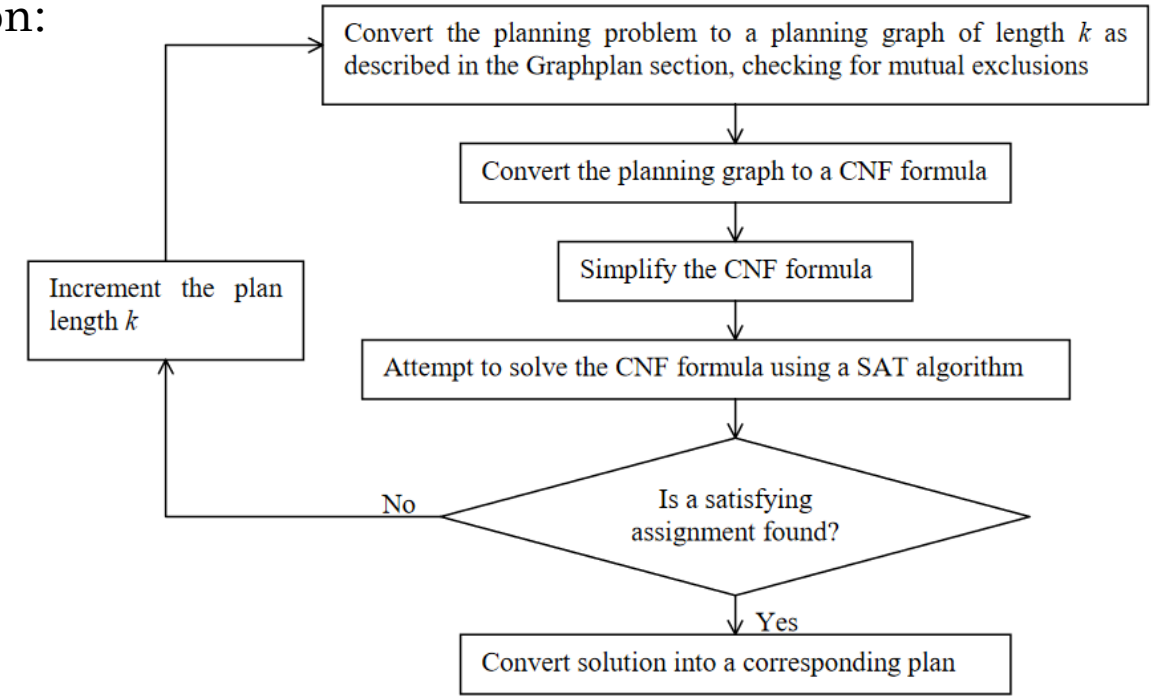
Problème avec les grandes instances !

$$\neg f_i \wedge f_{i+1} \Rightarrow \left(\bigvee_{a \in A | f_i \in effects^+(a)} a_i \right) \wedge$$

$$f_i \wedge \neg f_{i+1} \Rightarrow \left(\bigvee_{a \in A | f_i \in effects^-(a)} a_i \right)$$

Building Explanatory Frame Axioms (Image by Author)

Solution:



Unifying SAT-based and Graph-based Planning Henry Kautz & Bart Selman 2006
https://www.ijcai.org/Proceedings/99-1/Papers/047.pdf

https://www-users.cs.york.ac.uk/~frisch/NB/slack-body.pdf

Autre amélioration possible :
Recherche dichotomique sur la longueur du plan

Merci