DATA 61

CPCES: conformant planner with counter-example based sampling

▷ Alban Grastien, Enrico Scala

Australian National University

October 2, 2018

www.csiro.au

CSIRO

Idée générale

CPCES:

- CEGAR (raffinement d'abstraction pilotée par des contre exemples) pour la planification conformante

Pourquoi c'est intéressant :

- Comment l'implémenter ?
- Comment l'améliorer ?
- Qu'est-ce que ca nous apprend sur le problème de planification conformante ?

Outline

Australian National University

Le problème : planification conformante

Notre planificateur : CPCES

Parlons de tags

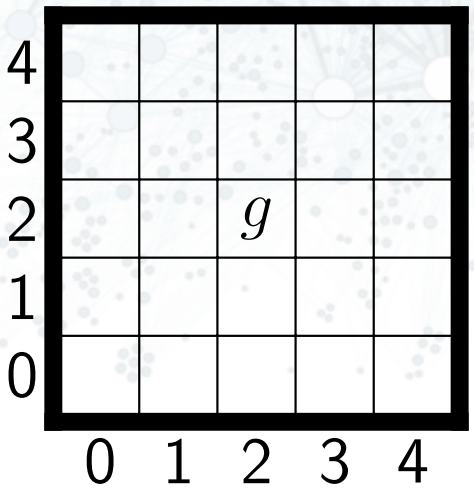
Conclusions

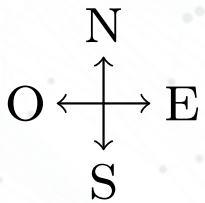
Définition du problème de planification conformante

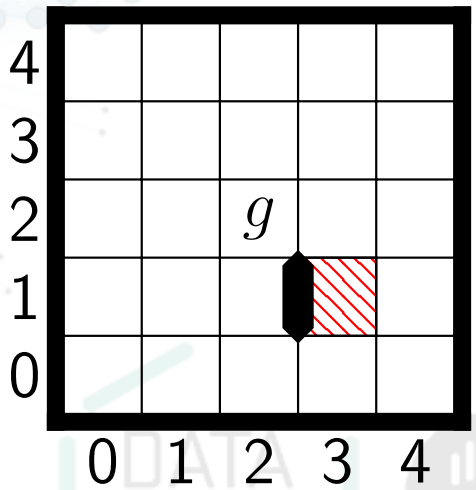
- Planification à base de modèle
- Incertitude sur l'état initial
- Pas d'observation
- Note : les représentations du domaine et de l'ensemble d'états initiaux en SAS+ / STRIPS sont compactes, mais le nombre d'états initiaux est exponentiel

Exemple

- Quatre mouvements possible: N, S, E, O ; le robot ne bouge pas s'il atteint le mur
- Le robot ne peut sortir des marais
- La position initiale est inconnue (hors marais) — le but est g





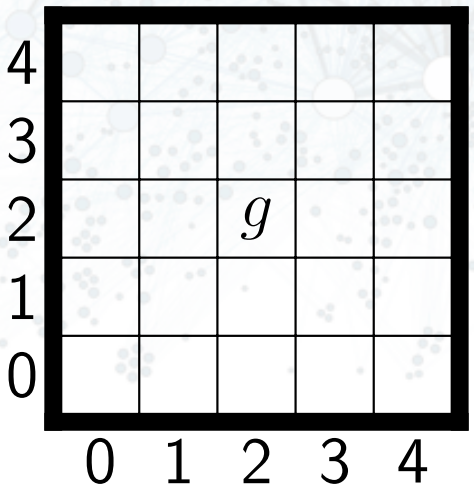


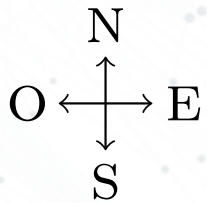
Solution:

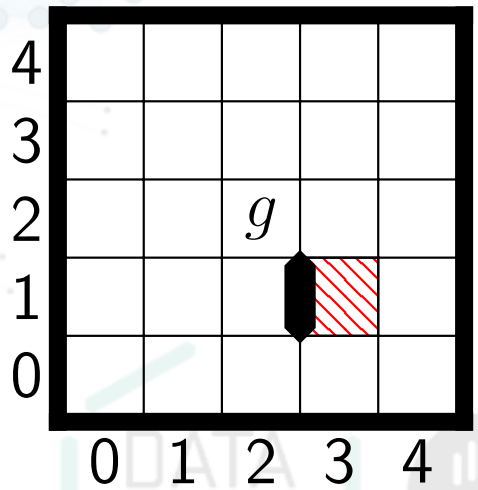
N4 O4 S2 E2

Exemple

- Quatre mouvements possible: N, S, E, O ; le robot ne bouge pas s'il atteint le mur
- Le robot ne peut sortir des marais
- La position initiale est inconnue (hors marais) — le but est g







Solution:
N4 O4 S2 E2

Solution:
E4 N4 E2 O2 S2

Quand utilise-t-on ce problème ?

Pas de ressource (temps, puissance de calcul, capteurs, etc.) pour analyser les observations

- rover sur Mars
- Vehicule sous-marin autonome
- Systèmes nécessitant une réponse rapide (réseaux électriques, web services, etc.)

Mais c'est surtout une plateforme pour des problèmes plus intéressants

Comment résoudre des problèmes de planification conformantes ?

- Recherche dans l'espace de croyance (*belief states*)
- Utilisation de représentation compacte de l'état de croyance (`bdd`, `cnf`)
- Planification classique puis replanification
- Manipulation des connaissances

Outline

Australian National University

Le problème : planification conformante

Notre planificateur : CPCES

Parlons de tags

Conclusions

gCPCES : l'algorithme

- gCPCES maintient un ensemble (à l'origine vide) d'états initiaux possibles *intéressants*, que l'on appele l'échantillon (*sample*)

- gCPCES calcule un **plan putatif**, qui est valide pour l'échantillon
  - S'il n'y en a pas, le problème n'admet pas de solution

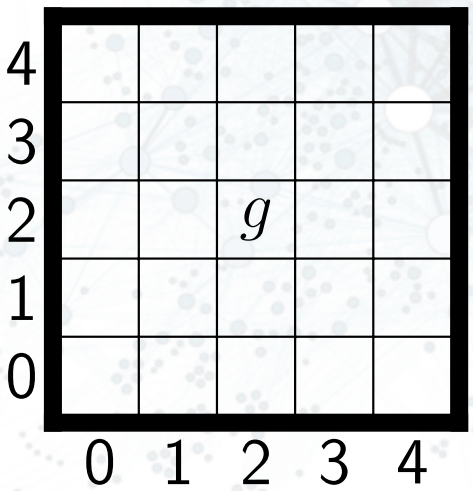
- gCPCES cherche un **contre-exemple**, un état initial pour lequel le plan est invalide
  - S'il n'y en a pas, alors le plan putatif est une solution

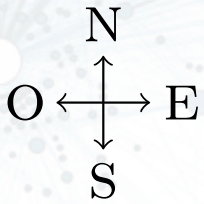
- Le countre-exemple est ajouté à l'échantillon

- L'on continue jusqu'à ce qu'une des procédures échoue

Exemple

Exécution de gCPCES





- Échantillon initial : $\{\ \}$
- Plan putatif : ε
- Contre-exemple : $\langle 0, 0 \rangle$
- Plan putatif : E2 N2
- Contre-example : $\langle 3, 1 \rangle$
- Plan putatif : S N2 O3 E2
- Contre-exemple : $\langle 4, 4 \rangle$
- Plan putatif : O4 S4 E2 N2
- Contre-exemple : Aucun
- Solution : O4 S4 E2 N2

Calculer un plan putatif pour un petit échantillon

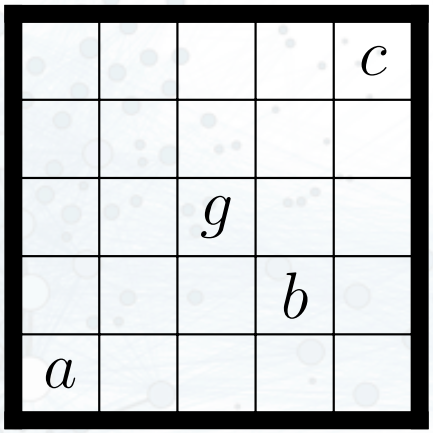
- (Remarque ; c'est un problème de planification conformante.)

- On a un échantillon $\{q_1, \ldots, q_k\}$ de k états initiaux
- On crée une copie du problème de planification par état initial (i.e., k copies)
- On synchronise ces copies, de telle manière qu'une action affecte toutes les copies en même temps $\rightarrow$ problème de planification classique
- On utilise un planificateur classique (efficace car les heuristiques sont bonnes)

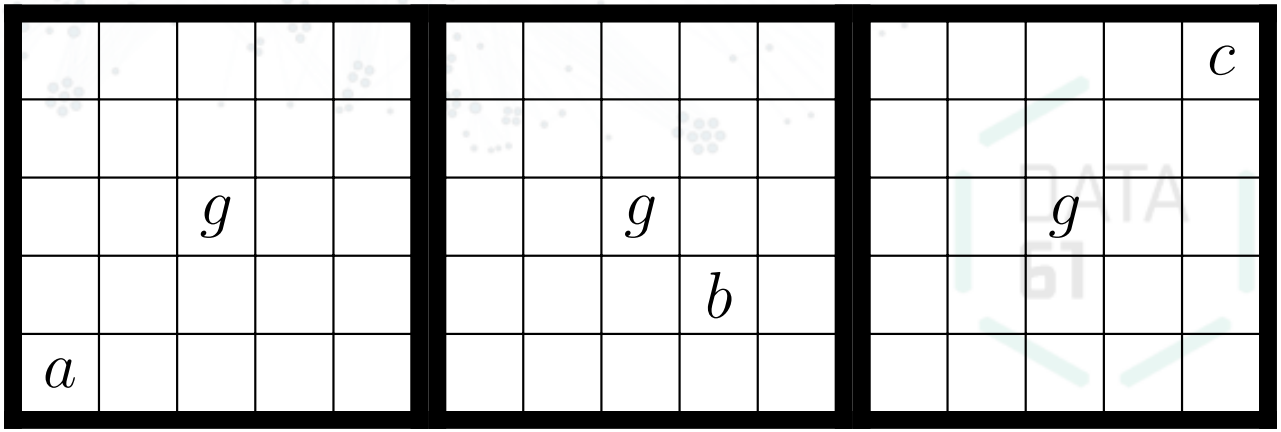
- Possible uniquement si k est faible

Calculer un plan putatif pour un petit échantillon

Exemple



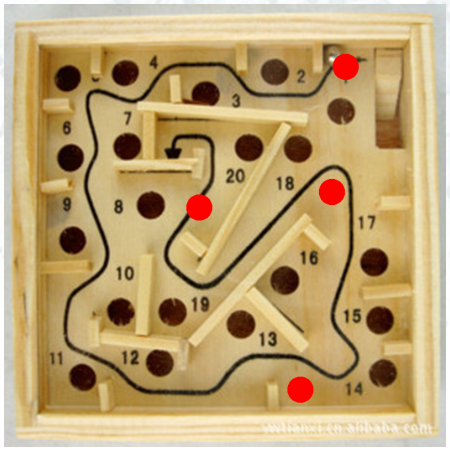
↓



Calculer un plan putatif pour un petit échantillon

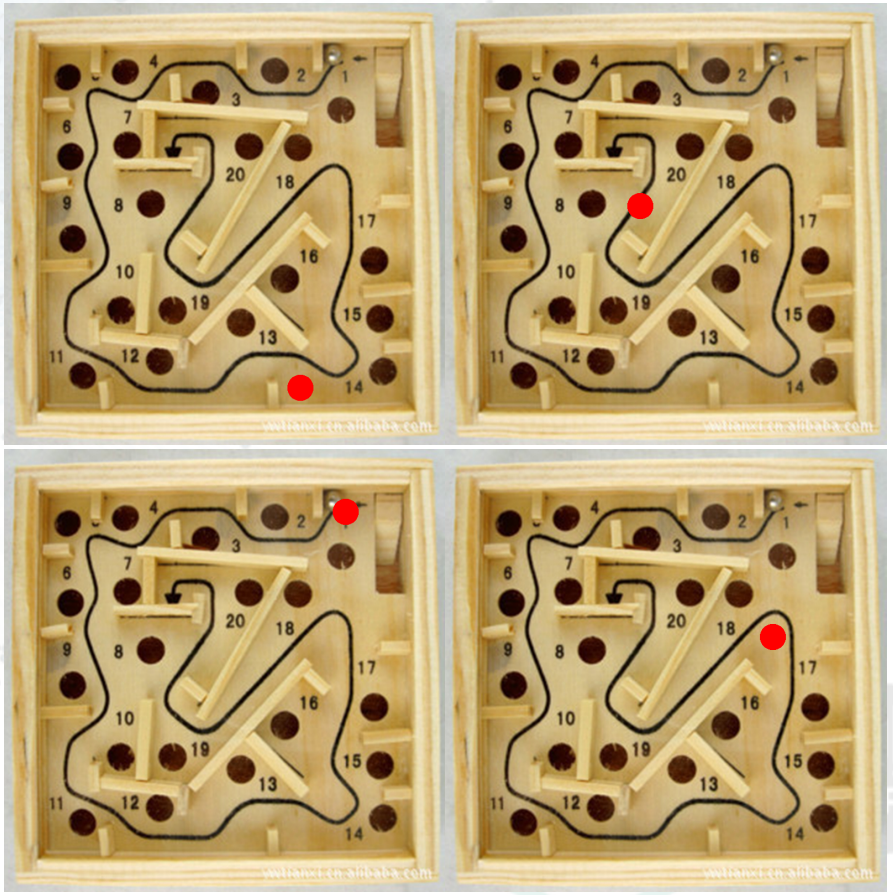
Intuition

Labyrinthe avec état initial inconnu :



$\rightarrow$

k labyrinthes collés ensemble :



Calculer un plan putatif pour un petit échantillon

En pratique (avec PDDL)

- Pour chaque variable d'état v et chaque état q de l'échantillon, on crée une variable $v_{/q}$
- $v_{/q}$ représente la valeur (connue) de v si l'état initial était q

```
(:action drop
:parameters (?o - obj ?p - pos )
:precondition
(and (located ?p) (trash_at ?p))
:effect (and
(when (holding ?o)
 (and (not (holding ?o))
  (disposed ?o)))))
```

$\rightarrow$

```
(:action drop
:parameters (?o - obj ?p - pos )
:precondition (forall (?i - inter)
(and (located ?p ?i) (trash_at ?p ?i)))
:effect (and (forall (?i - inter)
(when (holding ?o ?i)
 (and (not (holding ?o ?i))
  (disposed ?o ?i))))))
```

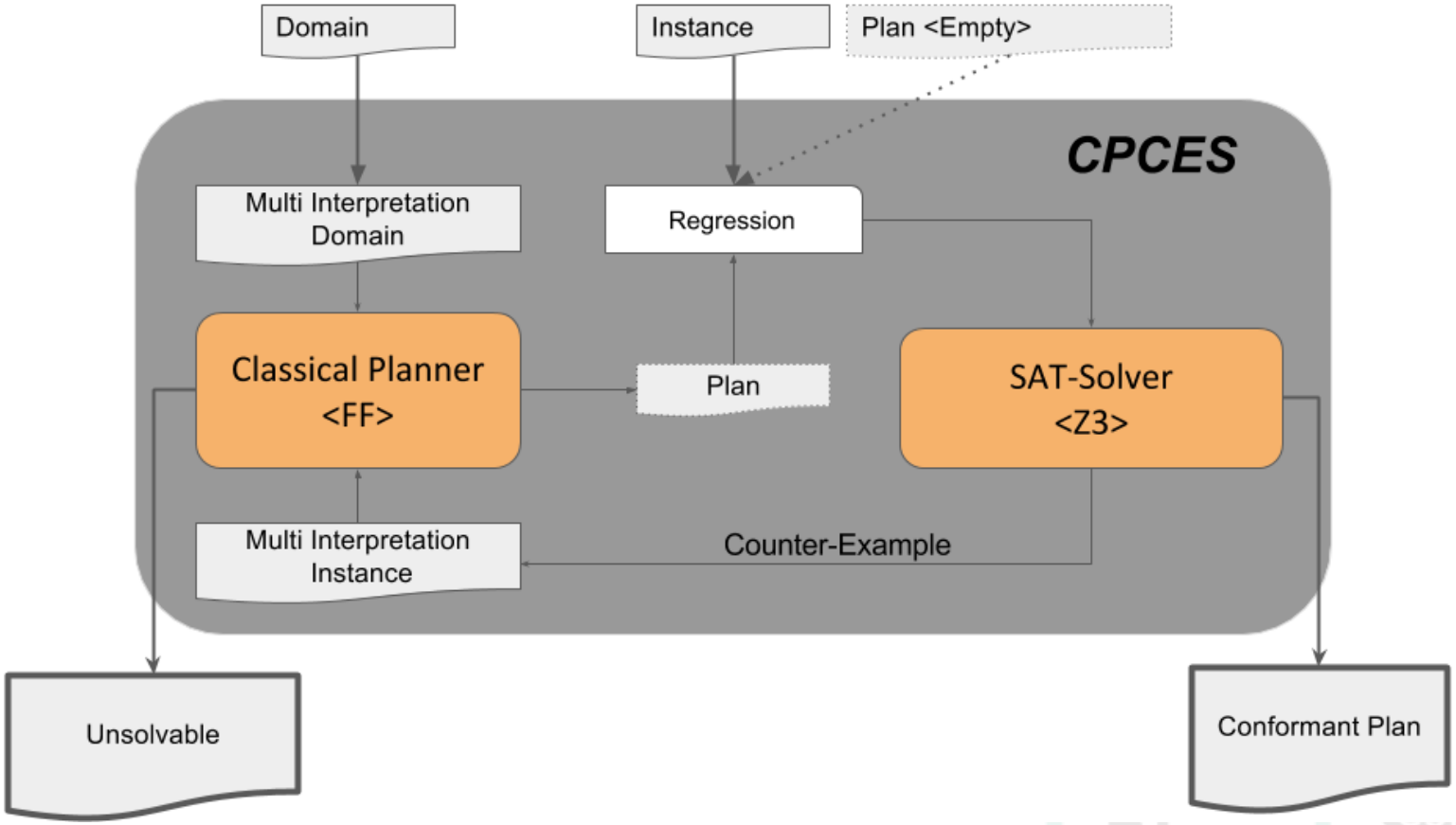
Calculer un contre-exemple

- Étant donné un candidat putatif $\pi = a_1 \dots a_k$
- Un contre-exemple est un état initial q_0 tel que π n'est pas applicable depuis q_0 ou $\pi(q_0)$ n'est pas un état final
- NP-Complet

- Soit $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \dots \xrightarrow{a_k} q_k$ la séquence d'états obtenus par l'application du plan (sans tenir compte de l'applicabilité)
- Cette séquence implique $n \times (k+1)$ variables, où n est le nombre de variables d'état
- On crée un problème de satisfaction propositionelle (SAT), satisfiable si et seulement si π n'est pas valide :

$$Init(q_0) \wedge Appli(q_0, a_1, q_1) \wedge \dots \wedge Appli(q_{k-1}, a_k, q_k) \wedge \big(\neg Apble(q_0, a_1) \vee \dots \vee \neg Apble(q_{k-1}, a_k) \vee \neg Final(q_k)\big)$$

Architecture



Évaluation empirique

Comparaison avec T1 pour des problème de *largeur* un ou plus

	Couv.		Longueur		Temps (s)	
Domain	CC	T1	CC	T1	CC	T1
Bomb(9)	8	**9**	136.0	**130.5**	102.7	< 1
Coins(9)	8	**9**	**81.3**	140.2	8.8	< 1
Dispose(11)	5	**8**	**210.4**	246.2	321.2	10.1
Uts(15)	14	14	**53.9**	63.6	65.6	4.6
Blocks(4)	**4**	2	**12.5**	13.0	< 1	< 1
Raos(3)	**3**	1	**16.0**	21.0	< 1	< 1
OneDispose(10)	**6**	4	**67.5**	79.3	13.3	202.2
LookGrab(18)	**18**	15	50.9	**33.7**	28.5	129.8
Total	**66**	62				

Outline

Australian National University

Le problème : planification conformante

Notre planificateur : CPCES

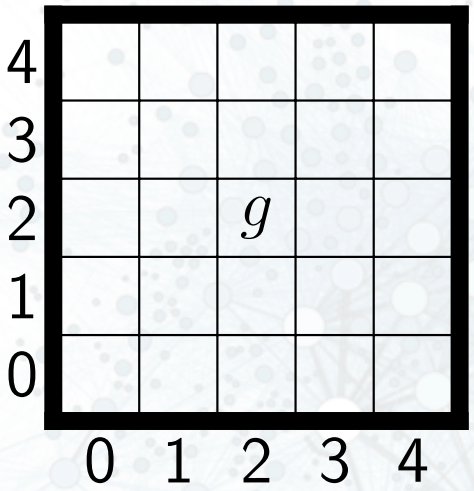
Parlons de tags

Conclusions

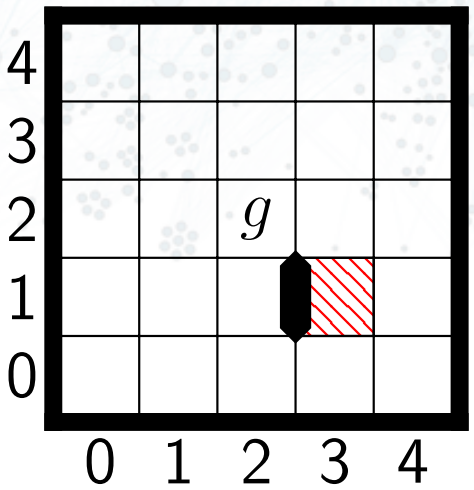
Définition d'un tag

- Une variable d'état c **dépend** d'une autre variable a, $a \preceq c$, si
  - $a = c$
  - a apparait dans la condition d'un effet (d'action) qui modifie la valeur de c
  - $a \preceq b$ et $b \preceq c$ pour une autre variable b
- Le **contexte** de c est la liste $ctx(c)$ de variables dont dépend c
- Le nombre de variables dans le contexte de c dont la valeur initiale est inconnue est la **largeur** de c
- La **largeur** du problème est la plus grande largeur de variable
- Un **tag** est une affection des variables de $ctx(c)$ possible dans l'état initial est un tag
- Il y a au plus $n \times D^w$ tags
  - n est le nombre de variables
  - D est la taille du plus grand domaine des variables
  - w est la largeur du problème

Exemple



- La position verticale n'affecte jamais la position horizontale
- $ctx(x) = \{x\}$, $ctx(y) = \{y\}$
- Tags : $(x = 0)$, $(x = 1)$, etc.



- La position verticale peut affecter la position horizontale
- $ctx(x) = ctx(y) = \{x, y\}$
- Tags : $(x = 0) \wedge (y = 0)$, $(x = 1) \wedge (y = 0)$, etc.

Tags et la valeur des variables incertaines

- Théorème :
  - Si l'on connait la valeur initial de $ctx(v)$, alors on connait la valeur de v à tout moment

- Dans notre réduction on peut remplacer $v_{/q}$ par $v_{/t}$
- Possible algorithme :
  - Calcule tous les tags
  - Génère toutes les variables $v_{/t}$ et redéfinit les actions
  - Use un planificateur classique
- Cette réduction requiert $n \times D^w$ variables

Tags et l'ensemble des plans valides

- Théorème :
  - Étant donnés deux problèmes P et P' de planification conformante identiques sauf pour leur ensemble d'états initiaux
  - $Tags(P') \subseteq Tags(P) \Rightarrow Solutions(P') \supseteq Solutions(P)$

- En particulier
  $Tags(P') = Tags(P) \Rightarrow Solutions(P') = Solutions(P)$
- Conséquence : du point de vue du planificateur, la liste des tags est ce que le diagnostiqueur doit déterminer

Tags et CPCES

- À chaque itération de gCPCES, l'ensemble des tags de l'échantillon s'accroit

  $\rightarrow$ le nombre d'itérations de gCPCES est au plus $n \times D^w$

- Amélioration de CPCES:

  $\rightarrow$ Il est souhaitable de générer des contre-exemples qui comprennent le plus de nouveaux tags possible

Outline

Australian National University

Le problème : planification conformante

Notre planificateur : CPCES

Parlons de tags

Conclusions

Conclusions

- Nous résolvons le problème de planification conformante en trouvant des contre-exemples intéressants
- Nous avons travaillé sur le problème de planification conformante, mais l'idée s'applique à beaucoup de problèmes similaires :
  - planification avec des variables numériques ou hybriques
  - observabilité partielle
  - probabilités
  - actions non déterministes
- Améliorations prévues :
  - apprentissage de macro-actions
  - amélioration du choix des contre-exemples
  - réduction à base de tag