

# Résolution de jeux de sûreté joués sur graphes

Huylenbroeck Florent

UMONS

25 novembre 2022

# Sommaire

## Jeux joués sur graphes

### Cas fini

- Attracteurs

- Stratégie gagnante

- Exemple

### Cas infini

- Représentation rationnelle du problème

- Apprentissage

- Représentation booléenne du problème

- Stratégie gagnante

- Exemple

### Questions

# Jeux joués sur graphes

Objectif : modéliser les interactions d'un système et de son environnement.

**Sûreté** : éviter certaines configurations.

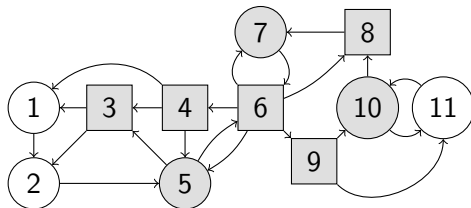
*Eviter un deadlock, qu'une valeur soit nulle, ...*


**Atteignabilité** : atteindre certaines configurations.


*Au moins une requête reçoit une réponse, une variable est initialisée ...*

# Jeux joués sur graphes

Une jeu joué sur graphe est défini par les ensembles  $V_s$ ,  $V_e$ ,  $E$  formant une *arène*, ainsi que  $F$  et  $I$ .



 Sommets du système ( $V_s$ ).

 Sommets de l'environnement ( $V_e$ ).

  Sommets correspondant à l'objectif ( $F$ ).

Les arcs reliant ces sommets représentent l'ensemble  $E$ .

# Jeux joués sur graphes

Au début de la partie, un *pion* est placé sur un sommet de  $I$  dans l'arène.

Le joueur possédant le sommet sur lequel est le pion décide du prochain coup.

→ Il faut trouver une stratégie gagnante selon l'objectif.

# Cas fini

Les ensembles  $V_s$  et  $V_e$  sont finis.

On va déterminer les sommets à partir desquels le système peut toujours remplir son objectif. Si tous ces sommets sont dans  $I$ , alors le système pourra toujours respecter la spécification.

Trouver cet ensemble de sommet, appelé *ensemble gagnant*, est possible, par exemple en construisant un *attracteur*.

# Attracteur : définition

## $i^e$ -attracteur

Soit un sous-ensemble de sommets  $F$  d'une arène. Le  $i^e$ -attracteur pour un joueur vers  $F$  est l'ensemble des sommets tel que, depuis tous les sommets de cet attracteur, le joueur peut forcer une visite d'un sommet de  $F$  en au plus  $i$  coups.

## attracteur

Soit un sous-ensemble de sommets  $F$  d'une arène. L'attracteur vers ces sommets pour un joueur est l'ensemble de sommets tels que le joueur peut forcer une visite d'un sommet de  $F$  depuis n'importe lequel de ces sommets.

# Attracteur : calcul

L'idée est de calculer la série des  $i^e$ -attracteurs vers  $F$  jusqu'à ce qu'augmenter  $i$  n'ajoute plus de sommets à l'attracteur.

$$\begin{aligned}Attr_s^0(F) &= F \\Attr_s^{i+1}(F) &= Attr_s^i(F) \\&\quad \cup \{v' \in V_s \mid \exists (v, v') \in E : v \in Attr_s^i(F)\} \\&\quad \cup \{v' \in V_e \mid \forall (v, v') \in E : v \in Attr_s^i(F)\}\end{aligned}\tag{1}$$

On obtiendra ainsi  $Attr_s^0(F) \subseteq Attr_s^1(F) \subseteq \dots = Attr_s(F)$

La complexité de cette construction est en  $O(m + n)$  pour une arène composée de  $n$  sommets et  $m$  arcs.



# Dualité jeux d'atteignabilité et sûreté

## Jeux d'atteignabilité

Construire l'attracteur vers l'ensemble objectif  $F$  pour le système donne immédiatement une stratégie gagnante pour celui-ci : à chaque coup, déplacer le pion d'un sommet de  $Attr_s^{i+1}(F)$  vers un sommet dans  $Attr_s^i(F)$

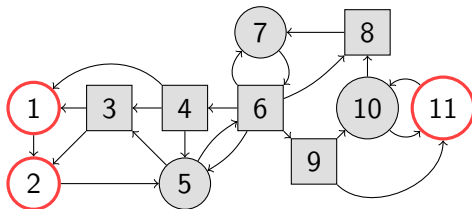
→ La construction opposée nous donne une stratégie gagnante pour un jeu de sûreté.

## Jeux de sûreté

Construire l'attracteur pour l'environnement vers les sommets hors de l'objectif  $F$  indique au système quels sommets éviter. Il obtient ainsi une stratégie gagnante en restant hors de cet attracteur à chaque coup.

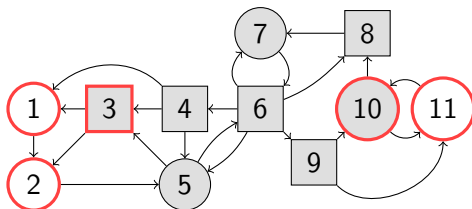
## Exemple

Calculons l'attracteur de l'environnement vers les sommets hors de l'objectif pour l'arène précédente. Les sommets du 0<sup>e</sup>-attracteur sont entourés en rouge.

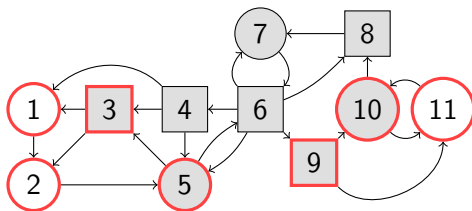


# Exemple

1<sup>er</sup>-attracteur

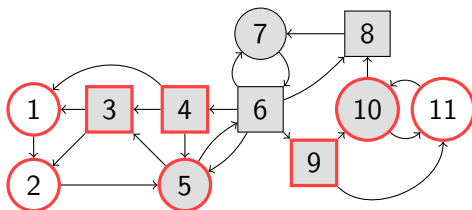


2<sup>e</sup>-attracteur



# Exemple

3<sup>e</sup>-attracteur, 4<sup>e</sup>-attracteur, ...



Au terme de la quatrième itération, aucun sommet n'est ajouté à l'attracteur.

→ La stratégie gagnante pour le système est de jouer ses coups vers les sommets 6, 7 et 8. Si le pion est initialement placé sur un de ces sommets, le système gagne.

# Cas infini

On autorise  $V_s$  et  $V_e$  à être infinis.

Un algorithme linéaire en terme d'arcs et de sommets aurait une complexité **infinie**.

→ Il faut passer par une étape d'abstraction.

# Alphabet, mots et langage

## Alphabet, mot et langage

Un alphabet  $\Sigma$  est un ensemble de caractères. Un mot sur cet alphabet est une suite de caractères de cet alphabet. Le mot vide est noté  $\epsilon$ . L'ensemble des mots sur un alphabet est noté  $\Sigma^*$ .

Un langage  $L$  sur  $\Sigma$  est un sous-ensemble de  $\Sigma^*$

## Exemple

Le langage  $L = \{u \in \Sigma^* \mid u = ab^n, n \in \mathbb{N}_0\}$  est le langage sur  $\Sigma = \{a, b\}$  des mots commençant par  $a$ , suivi d'une répétition de  $b$ .

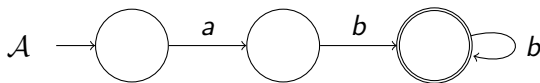
# Automates

## Automate fini

Un *automate* fini est un 5-uple  $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$  où

- ▶  $Q$  est un ensemble fini, non vide, d'états.
- ▶  $\Sigma$  est l'alphabet lu par l'automate.
- ▶  $\delta$  est une fonction  $Q \times \Sigma \rightarrow Q$  appelée *fonction de transition* de l'automate. L'expression  $\delta(q_1, u) = q_2$  indique que l'automate transitionne de l'état  $q_1$  à l'état  $q_2$  quand le mot  $u \in \Sigma^*$  est lu depuis l'état  $q_1$ .
- ▶  $q_0 \subseteq Q$  est l'ensemble des états initiaux.
- ▶  $F \subseteq Q$  est l'ensemble des états finaux.

## Exemple



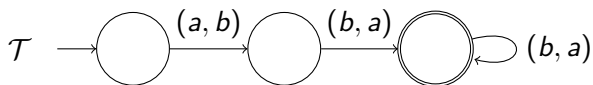
$\mathcal{A}$  est l'automate qui accepte le langage de l'exemple précédent.

# Transducteurs

## Transducteur

Un transducteur est un automate fini dont les transitions se font par des paires de caractères, dont les composantes appartiennent chacune à un alphabet respectif. Il permet de définir une relations entre deux langages.

## Exemple

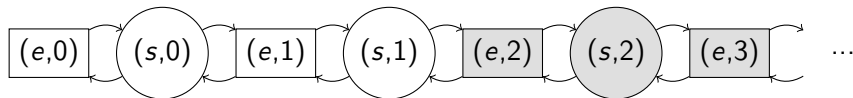


$\mathcal{T}$  est le transducteur qui permet d'établir la relation "d'inversion" du langage de  $\mathcal{A}$  de l'exemple précédent.



## Exemple de jeu de sûreté infini

Introduisons maintenant un exemple de jeu de sûreté joué sur une arène infinie



Arène infinie, fermée à gauche. Le système ne peut déplacer le pion que vers la droite ou ne pas le déplacer. L'environnement ne peut le déplacer que vers la gauche ou pas du tout.

L'ensemble objectif  $F$  correspond aux sommets dont le numéro est supérieur à 2.

Initialement, le pion est placé sur un sommet du système dans  $F$ .

# Représentation rationnelle

Afin de représenter notre jeu de sûreté sous forme d'automates, il faut trouver un encodage, afin de définir l'alphabet utilisé par les automates.

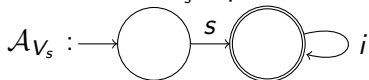
Pour cet exemple, l'alphabet choisit est  $\Sigma = \{s, e, i\}$ . Les sommets seront encodés par les mots  $u \in \{si^n, ei^n, n \in \mathbb{N}\}$

## Exemple

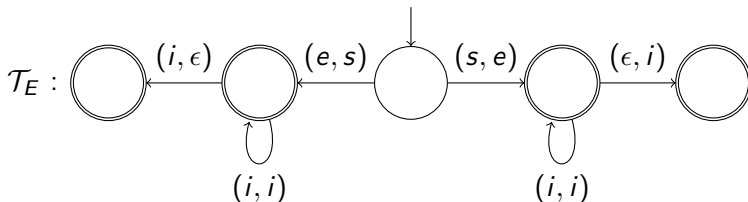
Le premier sommet de l'ensemble  $F$  sera encodé par  $eii$ , le sommet à sa droite  $sii$ , etc.

# Représentation rationnelle

L'automate  $\mathcal{A}_{V_s}$  représente les sommets du système.



Le transducteur  $\mathcal{T}_E$  représente les transitions.



En construisant de tels automates pour les ensembles  $V_e$ ,  $I$  et  $F$ , on a réduit notre problème infini à un problème fini.

# Représentation rationnelle

L'objectif est maintenant de construire un automate  $\mathcal{A}_{W_s}$  acceptant le langage des mots représentant les sommets de l'ensemble gagnant du système.

→ Synthèse inductive guidée par le contre-exemple.

# Apprentissage : ensemble gagnant

Afin de construire l'automate  $\mathcal{A}_{W_s}$ , nous allons définir rigoureusement ce qu'est un ensemble gagnant.

## Ensemble gagnant

Soit un jeu de sûreté défini par les ensembles  $V_s$ ,  $V_e$ ,  $I$  et  $F$ , l'*ensemble gagnant* pour le système est l'ensemble  $W_s$  tel que :

- ▶ (1)  $I \subseteq W_s$
- ▶ (2)  $W_s \subseteq F$
- ▶ (3)  $\forall v \in W \cap V_s, Succ(v) \cap W \neq \emptyset$
- ▶ (4)  $\forall v \in W \cap V_e, Succ(v) \subseteq W$

Où  $Succ(v)$  est la fonction retournant les successeurs du sommet  $v$ .

Cet ensemble n'est pas garanti d'exister pour un jeu de sûreté.

# Apprentissage : SIGCE

La *synthèse inductive guidée par le contre-exemple* est une manière d'apprendre un automate sur base d'un échange entre un enseignant (qui connaît le jeu) et un élève.

Cet apprentissage se déroule à l'intérieur d'une boucle telle qu'à chaque étape,

- ▶ L'élève conjecture un automate voulant décrire l'ensemble gagnant et le soumet à l'enseignant.
- ▶ L'enseignant vérifie la définition d'ensemble gagnant pour cet automate.
  - ▶ Si une règle est enfreinte, il retourne un *contre-exemple*.
  - ▶ Sinon, l'apprentissage s'arrête et l'automate correspond à l'ensemble gagnant.

# Apprentissage : enseignant

Afin de vérifier les conditions de la définition d'ensemble gagnant, l'enseignant utilise des opérations diverses sur les automates afin de construire de nouveaux automates et vérifie le langage de ceux-ci.

Selon ces vérifications, l'enseignant retourne deux types de contre-exemples :

- ▶ Les contre-exemples *positifs* (resp. *négatifs*) : un mot  $u$  tel que  $u$  doit obligatoirement être accepté (resp. rejeté) par  $\mathcal{A}_{W_s}$ .
- ▶ Les contre-exemples d'*implication universelle* (resp. *existentielle*) : une paire  $(u, \mathcal{A})$  telle que si le mot  $u$  est accepté par  $\mathcal{A}_{W_s}$ , alors tous les mots de  $\mathcal{L}(\mathcal{A})$  (resp. au moins un) doivent aussi être acceptés.

# Apprentissage : élève

Afin de conjecturer un automate, l'élève va construire des problèmes de satisfiabilité de taille incrémentale, basés sur les contre-exemples reçus. La taille du problème va dépendre du nombre de sommets dans l'automate qu'il va essayer de conjecturer.

Si un de ces problèmes est satisfiable, alors un *modèle* pour ce problème permettra de construire un automate à soumettre à l'enseignant.



# Formules booléennes

Afin d'assurer la cohérence de l'automate avec les contre-exemples, l'élève va utiliser plusieurs variables booléennes.

- ▶  $d_{p,a,q}$  : si cette variable est vraie, alors une transition du sommet  $p$  vers le sommet  $q$  par  $a$  existera dans l'automate.
- ▶  $f_q$  : si elle est vraie, le sommet  $q$  sera un sommet acceptant.
- ▶  $x_{u,q}$  : si elle est vraie, la lecture du mot  $u$  par l'automate amènera sur l'état  $q$ .

Ainsi, la formule

$$\bigwedge_{u \in P} \bigwedge_{q \in Q} x_{u,q} \rightarrow f_q \quad (2)$$

où  $P$  contient les contre-exemples positifs, forcera les mots de ces contre-exemples à être accepté par l'automate.

# Traduction d'un modèle en un ensemble gagnant

$\mathcal{A}_{W_s} = \{Q, \Sigma, q_0, \delta, F\}$  tel que :

- ▶  $Q = \{0, \dots, n\}$  est fixé à l'avance, où  $n$  est la valeur incrémentale du problème de satisfiabilité.
- ▶  $\Sigma = \{s, e, i\}$
- ▶  $q_0 = 0$
- ▶  $\delta$  contient les transitions de  $p$  vers  $q$  par  $a$  pour toute variable  $d_{p,a,q}$  vraie dans le modèle.
- ▶  $f_q$  contient chaque sommet  $q$  tels que  $f_q$  est vraie dans le modèle.

# Stratégie gagnante

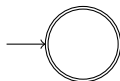
Si l'apprentissage s'arrête, alors l'ensemble gagnant retourné donne immédiatement une stratégie gagnante pour le système.

Cependant, si un tel ensemble n'existe pas, alors l'apprentissage ne s'arrêtera jamais.

→ L'algorithme n'est que partiel (et, s'il s'arrête, est de l'ordre de  $O(n2^{n^2})$  pour un ensemble gagnant décrit par un automate à  $n$  sommets).

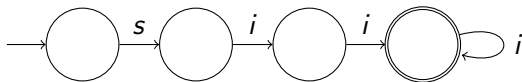
## Exemple

Premier automate conjecturé par l'élève, en l'absence de contre-exemple :



L'enseignant retourne un contre-exemple positif : *sii*.

Sur base de ce contre-exemple, l'élève soumet l'automate

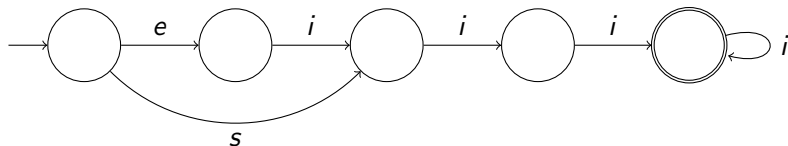


à l'enseignant.

## Exemple

L'enseignant retourne un contre-exemple d'implication existentielle  $(sii, \mathcal{A})$  avec  $\mathcal{L}(\mathcal{A}) = \{eii, eiii\}$ .

L'élève retourne ensuite l'automate



L'enseignant ne trouve pas de contre-exemple à retourner car la définition d'ensemble gagnant est respectée. L'apprentissage s'arrête.

# Questions

Avez-vous des questions ?

# Remerciements

Merci à Gaëtan Staquet, Clément Tamines,  
et plus particulièrement à Véronique Bruyère,  
pour leur attention et leur suivi au cours de ce mémoire.

Florent Huylenbroeck.