Jeux pour l'informatique et complexité des stratégies

Pierre Vandenhove[1,2]

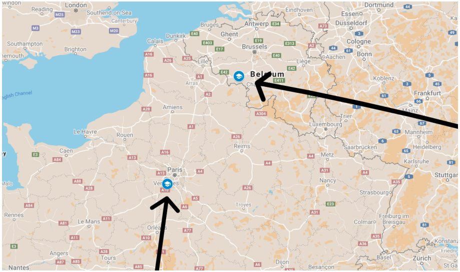
[1]F.R.S.-FNRS & UMONS – Université de Mons, Belgium

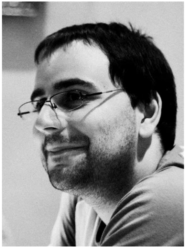
[2]Université Paris-Saclay, CNRS, ENS Paris-Saclay, LMF, France

21 avril 2022 – Séminaire Jeunes

Introduction

- Objectif : présenter mon **sujet de recherche**.
- Études de maths (option informatique) à l'UMONS.
- Doctorat commencé en octobre 2019.
- Cosupervisé par. . .





Mickael Randour,
UMONS

Patricia Bouyer,
Université Paris-Saclay, LMF,
France

Plan

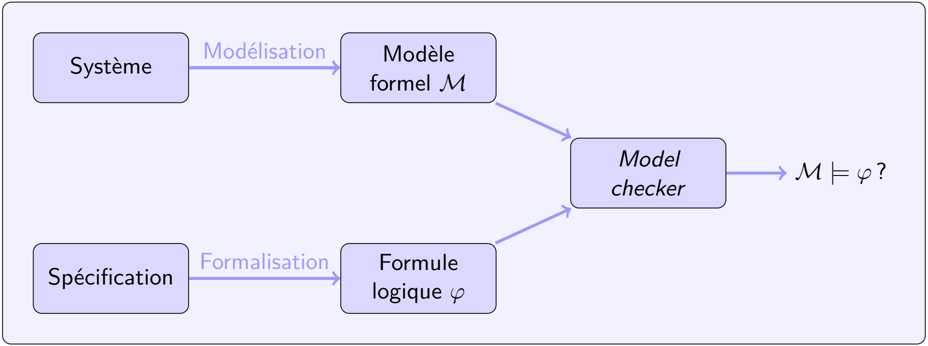
- **Vérification formelle** : vérifier le comportement de systèmes informatiques.
- Modélisation via la **théorie des jeux**.
- Question de recherche : comprendre la **complexité des stratégies** dans ces jeux.

Motivation : systèmes réactifs

- **Systèmes réactifs** = systèmes qui interagissent continuellement avec leur environnement (serveur web, aspirateur automatique, avion...).
- **Réagir** aux événements de l'environnement tout en accomplissant un **objectif**.
- Sujets aux **erreurs**, parfois graves (pertes financières, morts).
- Solution 1 : **tests** ? Pas exhaustif.
- Solution 2 : **vérification formelle** et **synthèse**.

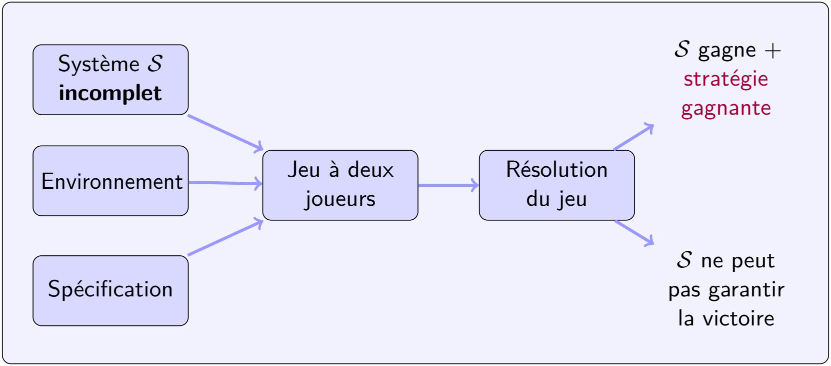
Vérification formelle

- On souhaite une **preuve** formelle du bon comportement d'un système.
- On travaille avec des **modèles**/abstractions de systèmes.
- **Spécification** : description des comportements acceptables du système.



Synthétiser un contrôleur

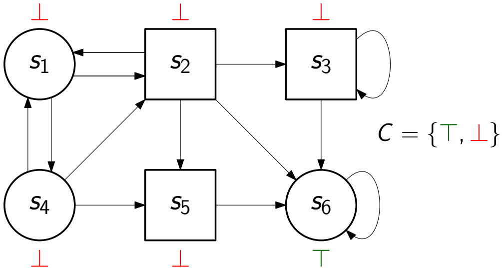
- Plus ambitieux : générer un **contrôleur** qui garantit la spécification.
- Définition lacunaire du système.
- Environnement vu comme un adversaire **antagoniste**.



- Modélisation via la **théorie des jeux**.

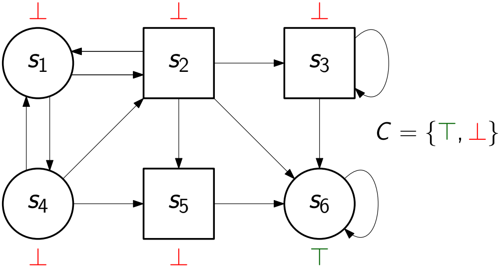
Jeu sur graphe

- **Jeu sur graphe** à deux joueurs reprenant les états du système.



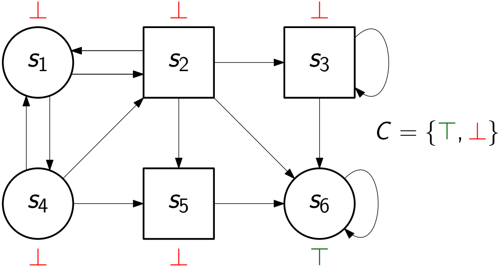
- Certains sommets ◯ contrôlés par **le système** (le joueur 1, $\mathcal{P}_1$), d'autres □ par **l'environnement** (le joueur 2, $\mathcal{P}_2$).
- Interaction de durée **infinie** entre les deux joueurs.
- On définit un **objectif** tel que $\mathcal{P}_1$ gagne ssi le système accomplit **sa spécification**.

Formellement



- **Arène** $\mathcal{A}$ à deux joueurs : S_1 ($\bigcirc$, pour $\mathcal{P}_1$) et S_2 ($\square$, pour $\mathcal{P}_2$), arêtes E.
- Ensemble C de **couleurs**. Les **états** sont colorés.
- Exemple : si les états visités sont $s_1 s_4 s_5 s_6 s_6 \ldots$, la séquence de couleurs générée est $\bot\bot\bot\top\top\ldots \in C^\omega$.
- **Objectif** : ensemble $W \subseteq C^\omega$. Jeu à **somme nulle**.

Exemple : jeu d'**accessibilité**

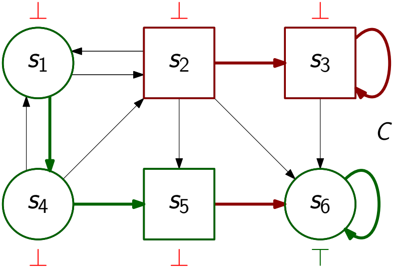


- Objectif : **atteindre** $\top$:

$$W = \{c_1 c_2 \ldots \in C^\omega \mid \exists i \geq 1, c_i = \top\}.$$

- Qui peut garantir la victoire quand le jeu commence en s_1 ?

Exemple : jeu d'**accessibilité**



$C = \{\top, \bot\}$

- Une stratégie qui suffit pour gagner regarde uniquement l'état courant :

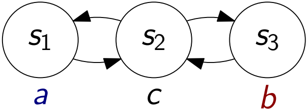
$$\sigma\colon S_i \to S.$$

  Une telle stratégie est dite **sans mémoire**.
- Les stratégies sans mémoire suffisent pour gagner dans **toutes** les arènes pour cet objectif, quand cela est possible !
  $\rightsquigarrow$ Jamais nécessaire de changer d'avis dans un état.

Les stratégies sans mémoire ne suffisent pas toujours

- $C = \{a, b, c\}$.
- Objectif : voir infiniment souvent a **et** infiniment souvent b :

$$W = \{c_1 c_2 \ldots \in C^\omega \mid \exists^\infty i \geq 1, c_i = a \land \exists^\infty i \geq 1, c_i = b\}.$$



- $\mathcal{P}_1$ peut gagner en jouant $acbcacbc\ldots$, mais pas sans mémoire.

Définition générale d'une **stratégie**

Une **histoire** est une séquence $s_1 \dots s_n \in S^*$ "cohérente" d'états de l'arène.

Pour $i \in \{1, 2\}$, on note $\mathsf{Hists}_i(\mathcal{A})$ les histoires $s_1 \dots s_n$ telles que $s_n \in S_i$.

Définition

Une **stratégie** de $\mathcal{P}_i$ est une fonction $\sigma\colon \mathsf{Hists}_i(\mathcal{A}) \to S$ telle que si $\sigma(s_1 \dots s_n) = s_{n+1}$, alors (s_n, s_{n+1}) est une arête de $\mathcal{A}$.

Problèmes pour implémenter un contrôleur :

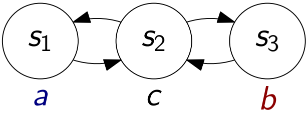
- il y a une infinité de stratégies,
- $\mathsf{Hists}_i(\mathcal{A})$ est infini.

Un ordinateur n'a pas une mémoire infinie !

Retour à l'exemple précédent

- Les stratégies **sans mémoire** ne suffisent pas pour l'exemple précédent.
- $C = \{a, b, c\}$, voir infiniment souvent a et infiniment souvent b :

$$W = \{c_1 c_2 \ldots \in C^\omega \mid \exists^\infty i \geq 1, c_i = a \wedge \exists^\infty i \geq 1, c_i = b\}.$$



- Compromis : utiliser de la **mémoire finie**. Ici, il suffit de retenir si on vient de voir a ou b !

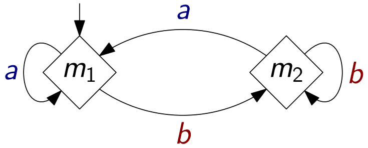
Stratégie à mémoire finie

- On condense l'information des histoires $\mathrm{Hists}_i(\mathcal{A})$ dans un objet fini $\rightsquigarrow$ perte d'information, mais potentiellement suffisant !
- Un modèle de calcul adapté dérive des **automates**.

Définition

Structure de mémoire $(M, m_{\mathsf{init}}, \alpha_{\mathsf{upd}})$: ensemble fini d'états M, état initial $m_{\mathsf{init}} \in M$, fonction *update* $\alpha_{\mathsf{upd}} : M \times C \to M$.

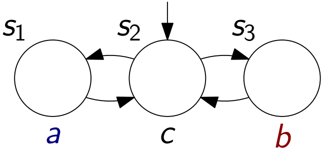
- Exemple pour se souvenir si a ou b a été vu en dernier :

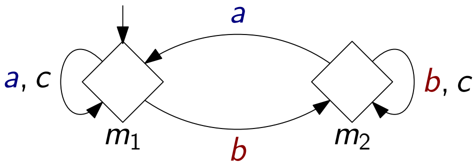


- Pour jouer, on se base sur l'état courant de $\mathcal{A}$ **et** sur l'état courant de la mémoire (ici, m_1 ou m_2).

Illustration

On définit une stratégie gagnante $\sigma : S_i \times M \to S$.

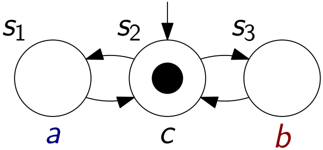


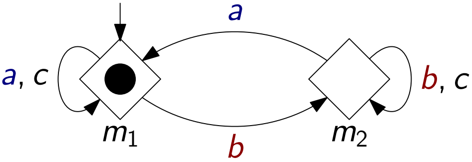


- Cette structure de mémoire suffit pour gagner dans cette arène.
- Pour **toute** arène, si gagner est possible, alors cette structure suffit !
- Plus petite structure de mémoire qui suffise pour $\mathcal{P}_1$ pour cet objectif.

Illustration

On définit une stratégie gagnante $\sigma \colon S_i \times M \to S$.

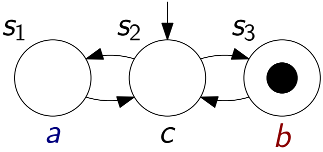




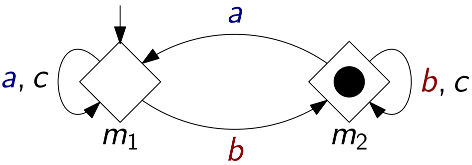
- Cette structure de mémoire suffit pour gagner dans cette arène.
- Pour **toute** arène, si gagner est possible, alors cette structure suffit !
- Plus petite structure de mémoire qui suffise pour $\mathcal{P}_1$ pour cet objectif.

Illustration

On définit une stratégie gagnante $\sigma\colon S_i \times M \to S$.



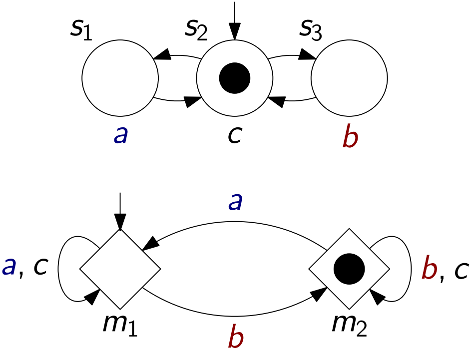
$$\sigma(s_2, m_1) = s_3$$



- Cette structure de mémoire suffit pour gagner dans cette arène.
- Pour **toute** arène, si gagner est possible, alors cette structure suffit !
- Plus petite structure de mémoire qui suffise pour $\mathcal{P}_1$ pour cet objectif.

Illustration

On définit une stratégie gagnante $\sigma\colon S_i \times M \to S$.

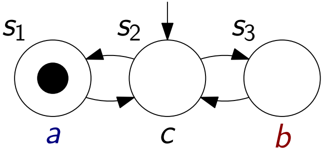


$$\sigma(s_2, m_1) = s_3$$
$$\sigma(s_3, \_) = s_2$$

- Cette structure de mémoire suffit pour gagner dans cette arène.
- Pour **toute** arène, si gagner est possible, alors cette structure suffit !
- Plus petite structure de mémoire qui suffise pour $\mathcal{P}_1$ pour cet objectif.

Illustration

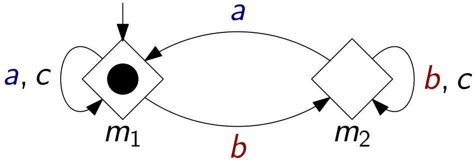
On définit une stratégie gagnante $\sigma\colon S_i \times M \to S$.



$$\sigma(s_2, m_1) = s_3$$

$$\sigma(s_3, \_) = s_2$$

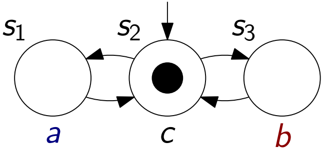
$$\sigma(s_2, m_2) = s_1$$



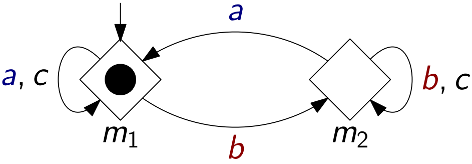
- Cette structure de mémoire suffit pour gagner dans cette arène.
- Pour **toute** arène, si gagner est possible, alors cette structure suffit !
- Plus petite structure de mémoire qui suffise pour $\mathcal{P}_1$ pour cet objectif.

Illustration

On définit une stratégie gagnante $\sigma\colon S_i \times M \to S$.



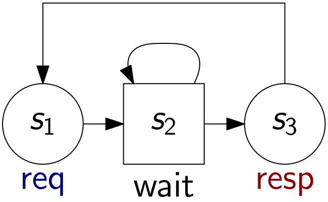
$$\sigma(s_2, m_1) = s_3$$
$$\sigma(s_3, \_) = s_2$$
$$\sigma(s_2, m_2) = s_1$$
$$\sigma(s_1, \_) = s_2$$



- Cette structure de mémoire suffit pour gagner dans cette arène.
- Pour **toute** arène, si gagner est possible, alors cette structure suffit !
- Plus petite structure de mémoire qui suffise pour $\mathcal{P}_1$ pour cet objectif.

Exemple nécessitant de la mémoire infinie

- $C = \{\text{req}, \text{resp}, \text{wait}\}$.
- Objectif W : nombre fini de req, **ou** $\exists N \in \mathbb{N}$ tel que chaque req est suivi par resp en moins de N étapes.



- $\mathcal{P}_2$ gagne en augmentant de plus en plus le temps entre deux requêtes.
- Ne peut pas être implémenté avec un automate fini
$\rightsquigarrow$ $\mathcal{P}_2$ a **besoin** de **mémoire infinie**.

Problèmes de recherche

- Hiérarchie entre les stratégies

$$\begin{aligned}\text{Sans mémoire } (S_i \to S) &\subsetneq \text{Mémoire finie } (S_i \times M \to S) \\ &\subsetneq \text{Générales } (\mathrm{Hists}_i(\mathcal{A}) \to S).\end{aligned}$$

- Comprendre (conditions suffisantes ou caractérisations) dans quels **contextes** les stratégies **simples** suffisent.
  - Classes d'arènes (finies, dénombrables, infinies, *stochastiques*...).
  - Classes d'objectifs ($W \subseteq C^\omega$, maximiser une fonction $f\colon C^\omega \to \mathbb{R}$, maximiser la probabilité d'un événement, ω-*réguliers*...).
  - Classes de stratégies "simples". Les stratégies **sans mémoire** sont bien comprises, les stratégies **à mémoire finie** moins.
- Complexité de calculer la quantité de mémoire pour un objectif donné.

Conclusion : exemple de résultat

Généralisation de résultats sur les stratégies sans mémoire[1, 2]

Lift de 1 à 2 joueurs[3, 4]

Soit W un objectif. Si

- dans tous les jeux de $\mathcal{P}_1$ à 1 joueur, une mémoire $\mathcal{M}_1$ suffit pour $\mathcal{P}_1$,
- dans tous les jeux de $\mathcal{P}_2$ à 1 joueur, une mémoire $\mathcal{M}_2$ suffit pour $\mathcal{P}_2$,

alors une mémoire $\mathcal{M}_1 \otimes \mathcal{M}_2$ suffit pour $\mathcal{P}_1$ et $\mathcal{P}_2$ dans les jeux à 2 joueurs.

Réduit un raisonnement sur les **jeux** à un raisonnement sur les **graphes**.

Merci !

1. Gimbert et Zielonka, « Games Where You Can Play Optimally Without Any Memory », 2005.
2. Colcombet et Niwiński, « On the positional determinacy of edge-labeled games », 2006.
3. Bouyer, Le Roux et al., « Games Where You Can Play Optimally with Arena-Independent Finite Memory », 2020.
4. Bouyer, Randour et Vandenhove, « Characterizing Omega-Regularity Through Finite-Memory Determinacy of Games on Infinite Graphs », 2022.