Logique modale, logique épistémique et model checking

Théo Losekoot

SUPERVISEUR: Sophie Pinchinat, logicA - INRIA

Relecteur attentif : Mathieu Poirier

Cet article est inspiré du livre [1]

12 mars 2019

Résumé

Pour raisonner sur un certain nombre de situations, il est essentiel de les modéliser de façon formelle. Une modélisation permet de raisonner sur une abstraction de la situation, dans laquelle ne sont représentés que les points utiles au raisonnement que l'on souhaite avoir. Une modélisation formelle permet de rendre plus rigoureuse la méthode de raisonnement et donc parfois possible son automatisation.

La logique permet cette modélisation formelle dans beaucoup de situations. Il existe plusieurs types de logiques qui peuvent être utilisées dans des situations différentes, en fonction du type d'énoncé ou de système qu'il faut modéliser. Nous verrons ici comment modéliser des MAS (Système Multi-Agents) grâce à la logique épistémique, et comment vérifier et raisonner sur certaines propriétés concernant la connaissance des agents de ces systèmes.

1 Introduction

Il existe de nombreux problèmes qui, à priori, ne montrent aucune similitude. Parmi eux, on peut citer les problèmes de routage des réseaux, des voitures autonomes, de l'étude des essaims insectes, de l'architecture des systèmes, etc. Pourtant, tous ces problèmes présentent un aspect commun. Il s'agit d'entités qui évoluent et communiquent entre elles dans un même environnement. C'est ce que l'on appelle des *Systèmes Multi-Agents* (MAS).

Les MAS sont une nouvelle manière de voir le monde, qui peuvent être vus et utilisés, en fonction du problème à résoudre, comme un nouveau paradigme de calcul ou de design, ou comme un nouveau paradigme pour penser et parler à propos du monde.

Ce papier a pour but de présenter une introduction à certains concepts permettant de raisonner sur des MAS. À la section 2 sera présenté le concept d'un système multi-agents. La section 3 portera sur la logique modale et le modèle de Kripke, qui sont un langage logique et une modélisation formelle permettant d'exprimer des propriétés dans certains systèmes multi-agents. La section 4 portera sur une instanciation de la logique modale qui permet de raisonner à propos des connaissances des agents : la logique épistémique. Enfin, en section 5, nous introduirons le concept de model checking, qui est une manière de vérifier des propriétés logiques au sein d'un modèle.

2 Système multi-agents

Dans un MAS, un agent est une entité qui peut avoir certaines propriétés, dont les plus courantes sont :

— autonomie - Agit sans intervention directe, a du contrôle sur ses actions et possède un état interne

— réactivité : Réagit aux changements dans l'environnement ;

— pro-activité : Prend des initiatives ;

— ayant un but : Agit dans l'objectif de remplir son but ;

— sociabilité : Interagit avec les autres agents ;

— matérialisation : Est capable d'observer et de modifier son environnement ;

— intelligence : Définition souvent floue ;

— rationalité : Fait toujours la bonne action par rapport aux informations qu'il possède.

Ces propriétés que peuvent posséder les agents sont souvent discutées, mais il reste sûr qu'un agent agit. Un agent est donc généralement défini par un ensemble de perceptions, un ensemble d'actions et d'une fonction

$$\text{agir} : \text{perceptions} \mapsto \text{actions}$$

Pour mieux comprendre la notion de MAS, la figure 1 est un exemple.

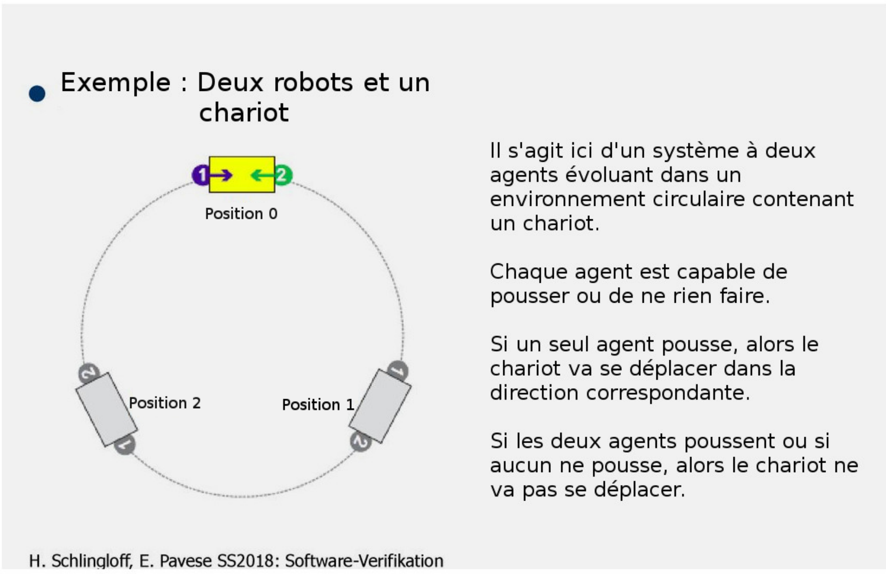

FIGURE 1 – Explications : les boules 1 et 2, respectivement en violet et vert, sont les agents. Ils sont toujours "collés" au rectangle jaune, le chariot.

La prochaine étape est de spécifier une propriété dans un langage logique. Cette propriété peut être soit désirable, dans ce cas l'objectif est de chercher à la satisfaire, soit indésirable, et dans ce cas l'objectif est de vérifier qu'une telle propriété soit toujours fausse.

À partir d'une propriété, on peut chercher à vérifier plusieurs choses :

— *validité* : qu'elle soit vraie dans tous les *modèles* (notion introduite à la section suivante) de tous les systèmes ;

— *satisfiabilité* : qu'elle soit vraie dans au moins un modèle d'au moins un système ;

— *model checking* : qu'elle soit vraie dans un modèle précis d'un système précis.

Des exemples de propriétés dans ce système sont : "Les agents se trouvent en position 2", "L'agent 1 *sait* qu'il se trouve en position 0", ou encore "L'agent 1 est en position 0 mais peut se déplacer en position 1".

Pour énoncer correctement ces propriétés, il va falloir définir une logique et un modèle de ce système.

3 Logique modale

La logique classique, à laquelle nous pensons généralement, est le calcul propositionnel. Le calcul propositionnel permet de représenter des propositions telles que "il pleut", "cet article est fabuleux", ou encore "il pleut ou il ne pleut pas". Les formules qu'il est possible de former avec cette logique sont soit des variables propositionnelles, que l'on note souvent p, q, ... , soit des formules de la forme $\neg\phi$, $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \rightarrow \psi$, avec ϕ et ψ des formules. Grâce à cette logique, il est possible de raisonner sur certaines propositions, mais il est impossible de représenter des phrases telles que "*Il est nécessaire qu*'il pleuve", "*Demain,* il pleut", "*L'agent 1 sait* qu'il se trouve en position 0". La logique modale permet de traiter ces propositions.

3.1 Logique modale et Modèle de Kripke

La logique modale est une extension du calcul des propositions. Ses nouveautés sont une vision différente des valuations ainsi que l'ajout de deux nouveaux opérateurs.

Les deux nouveaux opérateurs sont $\Box\phi$, qui indique que ϕ est vraie dans toutes les situations possibles atteignables, et $\Diamond\phi$, qui indique que ϕ est vraie dans au moins une situation possible atteignable. Ce qui est entendu par "situation possible" et "atteignable" réfère à la nouvelle vision de la valuation et est expliqué en section 3.3.

3.2 Syntaxe

La syntaxe d'une formule de la logique modale, sur un ensemble de propositions atomiques $Prop$, est donc, pour $p \in Prop$ et ϕ, ψ des formules, de l'une des formes suivantes : $p \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \rightarrow \psi \mid \Box\phi \mid \Diamond\psi$.

3.3 Sémantique

Définition 1 (Kripke) *Soit $Prop = \{p, q, r, ...\}$ un ensemble de propositions atomiques. Un **modèle de Kripke** est un triplet $M = (St, R, V)$. St (**St**ates) est l'ensemble des mondes, ou états, possibles. R (**R**elation) est une relation binaire entre les états, dite relation d'accessibilité ($R \subseteq St \times St$). V (**V**aluation) est la fonction d'interprétation des propositions ($V : Prop \mapsto \mathcal{P}(St)$) qui renvoie l'ensemble de tous les mondes dans lesquels la proposition est vraie.*

Pour illustration, dans notre exemple précédent, un monde possible serait le cas représenté à l'image 1, c'est à dire un monde dans lequel $Prop = \{pos_0, pos_1, pos_2\}$ et $pos_0 = \top, pos_1 = \bot, pos_2 = \bot$.

Soit $M = (St, R, V)$ un modèle de Kripke et $s \in St$ un monde possible de ce modèle. La valeur de vérité d'une formule ϕ dans (M, s) est définie par induction comme suit, grâce à la relation sémantique $(M, s) \models \phi$ qui signifie "ϕ est vrai dans le monde s du modèle M".

$(M, s) \models p$	lorsque	$s \in V(p)$
$(M, s) \models \neg\phi$	lorsque	$(M, s) \not\models \phi$
$(M, s) \models \phi \wedge \psi$	lorsque	$(M, s) \models \phi$ et $(M, s) \models \psi$
$(M, s) \models \phi \vee \psi$	lorsque	$(M, s) \models \phi$ ou $(M, s) \models \psi$
$(M, s) \models \phi \rightarrow \psi$	lorsque	$(M, s) \models \psi$ ou $(M, s) \not\models \phi$
$(M, s) \models \Box\phi$	lorsque	$\forall s' \in St$, si sRs' alors $(M, s') \models \phi$
$(M, s) \models \Diamond\phi$	lorsque	$\exists s' \in St$ tq sRs' et $(M, s') \models \phi$

3.4 Extensions

La logique modale peut être étendue à la logique multi-modale, qui est la même chose mais avec plusieurs opérateurs $\Box_i$ et $\Diamond_i$, qui sont chacun interprétés avec la relation d'accessibilité correspondante R_i.

La logique modale est une logique générique qui peut représenter plusieurs aspects de la réalité suivant le sens donné à la/aux relations d'accessibilité. Les plus répandues sont :

— la relation d'accessibilité représente le temps : logique temporelle ;
— la relation d'accessibilité représente les actions : logique dynamique ;
— la relation d'accessibilité représente les connaissances : logique épistémique ;
— la relation d'accessibilité représente la causalité : logique contrefactuelle ;
— la relation d'accessibilité représente les obligations : logique déontique.

4 Logique épistémique

La logique épistémique repose sur la logique (multi-)modale et a pour but l'étude de la connaissance des agents. Chaque agent possède son propre opérateur $\Box_i$ et donc sa propre relation d'accessibilité R_i. Dans R_i, deux mondes sont reliés si l'agent i ne les distingue pas. Cette relation représente donc la notion d'indistinguabilité pour l'agent i.

Pour des raisons historiques, une autre notation est utilisée dans cette logique, $K_i\phi$, qui signifie "l'agent i sait que ϕ". K_i correspond donc à l'opérateur $\Box_i$. Cet opérateur est défini formellement comme :

$$(M, s) \models K_i\phi \qquad \text{lorsque} \qquad \forall s' \in St, \text{ si } sR_is' \text{ alors } (M, s') \models \phi.$$

Il est bon de savoir que le modèle d'un système dans la logique épistémique est souvent construit du point du vue d'un observateur externe omniscient qui a une vue globale sur le système.

4.1 Extensions de l'exemple des robots

Reprenons notre exemple des deux robots et du chariot, améliorons-le et modélisons-le, cf figure 2.

Le *système* peut être décrit comme indiqué dans la figure 1, avec les compléments ci-dessous.

Les deux robots et le chariot (qui se déplacent toujours ensemble) peuvent être dans une et une seule des trois positions (0, 1, ou 2), et jamais entre deux positions. Un décor a été ajouté. Une lumière verte éclaire les positions 0 et 2, et un chemin de pierre passe par les positions 0 et 1, cf la figure 2. Pour la suite, on suppose que le robot 1 n'a comme capteur que des yeux, il peut donc observer la lumière mais pas la différence de texture au sol. Aussi, le robot 2 n'a que des capteurs de toucher, il distingue donc les textures au sol mais n'est pas sensible à la lumière.

4.2 Modélisation de l'exemple

Pour *modéliser* un tel système, il faut se demander quels sont les états St du modèle $M_{ex} = (St, R_1, ..., R_n, V)$. Ici, la meilleure solution semble être de faire correspondre chaque position du système à un monde possible du modèle, appelons-les s_0, s_1 et s_2 (pour pos_0, pos_1, pos_2). Il faut maintenant modéliser les relations d'accessibilité. Comme les relations représentent l'indistinguabilité, elles sont nécessairement réflexives (chaque état est indistinguable de lui-même), symétriques (si s_1 est indistinguable de s_2, alors s_2 est indistinguable de s_1), et transitives ($s_1R_1s_2 \wedge s_2R_1s_3$ implique $s_1R_1s_3$).

Comme le robot (agent) 1 ne distingue pas les deux positions éclairées, on a alors $s_0R_1s_2$. En revanche, il sait distinguer entre éclairé et non-éclairé, donc il n'y a ni $s_0R_1s_1$, ni $s_2R_1s_1$. Le robot 2 distingue les positions avec chemin ou sans chemin de manière similaire. Le modèle de Kripke M_{ex} obtenu est en figure 2.

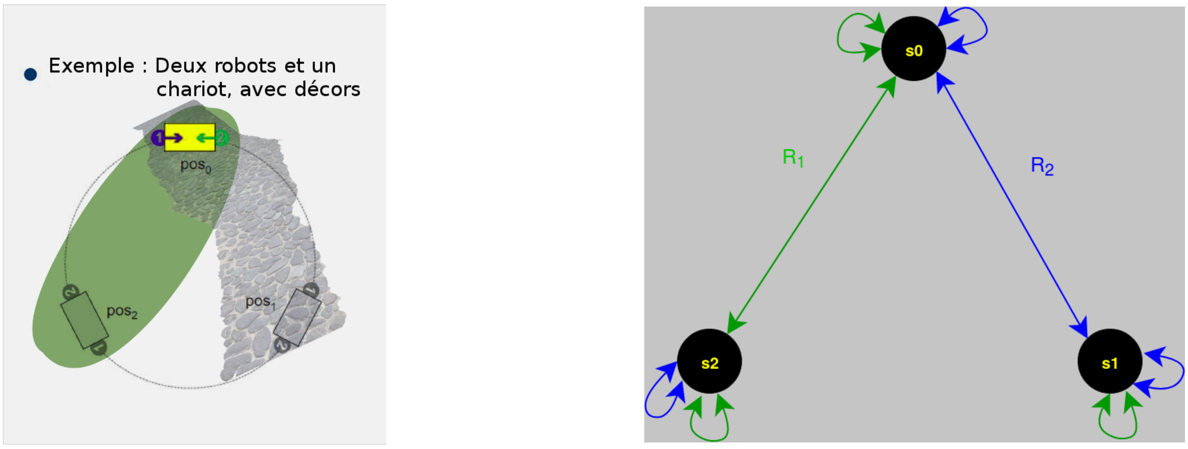

FIGURE 2 – Le schéma de gauche représente le *système* de notre exemple. Le schéma de droite représente le *modèle* (de Kripke) de l'exemple, M_{ex}

4.3 Utilisation du modèle

Maintenant que le système est modélisé, il devient possible de raisonner sur son modèle. Voici une liste de certaines propriétés qui pourraient être intéressantes à exprimer et leur écriture en langage logique. Les formules choisies ici sont vraies.

— "Quand les agents sont en position 1, alors l'agent 1 le sait.", qui s'écrit $(M_{ex}, s_1) \models K_1(pos_1)$.

— "Quand les agents sont en position 1, alors l'agent 2 ne sait pas s'il est en position 1 ou en position 0, mais il sait qu'il n'est pas en position 2", qui s'écrit $(M_{ex}, s_1) \models K_2(pos_1 \vee pos_0) \wedge K_2(\neg pos_2)$.

— "Quand les agents sont en position 2, alors l'agent 2 le sait et sait qu'il n'est ni en position 0, ni en position 1", qui s'écrit $(M_{ex}, s_2) \models K_2(pos_2) \wedge K_2(\neg pos_0 \wedge \neg pos_1)$.

Attention à la modélisation. Lors de la représentation en modèle de Kripke, il faut également faire attention aux implicites que l'on ajoute, notamment en ne représentant pas certains mondes et certaines relations.

Par exemple, le modèle M_{ex} construit en figure 2 exprime aussi $(M_{ex}, s_0) \models K_1(pos_2 \rightarrow K_2(pos_2) \wedge \neg pos_2 \rightarrow K_2(pos_1 \vee pos_0))$. Effectivement, à partir de l'état s_0, le robot 1 ne voit aucun monde possible dans lequel le robot 2 pourrait penser être dans la position 2. Si cette propriété est indésirable (ce qui n'est pas le cas ici), il aurait fallu modéliser le système comme en figure 3. Notons que l'implicite symétrique existe toujours dans cette figure.

4.4 Extensions courantes

Trois nouveaux opérateurs sont définis afin de parler de la connaissance d'un groupe d'agents. Soit A un ensemble d'agents.

Le premier opérateur, $E_A\phi$, signifie "tout le monde dans le groupe A sait que ϕ" et se nomme "connaissance mutuelle/collective".

Le deuxième opérateur, $C_A\phi$, signifie "tout le monde dans le groupe A sait que ϕ, et tout le monde sait que tout le monde le sait" et se nomme "connaissance commune".

Le troisième opérateur, $D_A\phi$, signifie "Si les agents de A partageaient les informations qu'ils possèdent, ils pourraient savoir que ϕ" et se nomme "connaissance distribuée".

Le premier opérateur est surtout là pour des raisons historiques, mais les deux suivants sont très utilisés, notamment dans le cas où les agents ont de la connaissance partagée qui n'est pas connaissance commune. Cela indiquerait un besoin de communiquer.

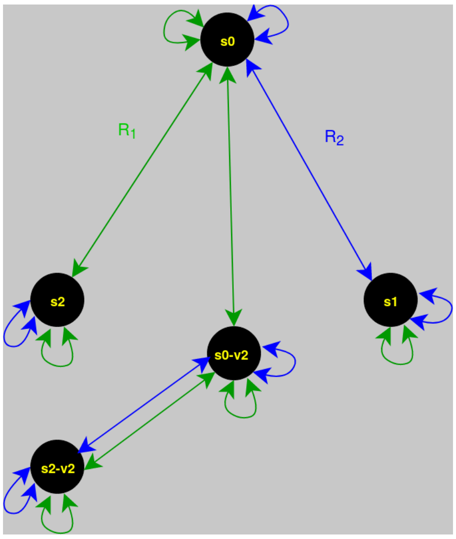

FIGURE 3 – Une variante du modèle M_{ex} dans lequel le robot 1 ne sait pas que le robot 2 est capable de savoir s'ils se trouvent en position 2 ou non. Les nouveaux états $s_0 - v2$ et $s_2 - v2$ représentent les mêmes positions que, respectivement, s_0 et s_2.

Une fois un certain modèle établi, il est intéressant de pouvoir vérifier des propriétés logiques de notre système, que l'on exprimera en logique modale.

La section suivante présente une méthode de vérification de formules de la logique modale. Cette méthode est également applicable à la logique épistémique.

5 Model checking de la logique modale

Le model checking est un problème de décision qui consiste en la vérification d'une propriété logique en considérant un modèle de Kripke donné. Cela diffère de la vérification et de la satifaisabilité d'une formule, qui eux sont des problèmes considérant tous les modèles de tout système.

Le model checking se décline en deux versions, le model checking local et le model checking global.

Définition 2 (model checking local) *Le* ***model checking local*** *est le problème de décision qui, étant donné un modèle, un état de ce modèle et une formule, décide si la formule est vraie dans cet état de ce modèle.*

Le MCL a pour spécification

$INPUT$: M, s, ϕ (Modèle, état, formule)

$OUTPUT$: Vrai si $(M, s) \models \phi$, faux sinon.

Définition 3 (model checking global) *Le* ***model checking global*** *n'est plus un problème de décision, mais une fonction qui rend un ensemble d'états du modèle. Étant donné un modèle et une formule, le model checking global renvoie tous les états de ce modèle tels que la formule est vraie.*

Le MCG a pour spécification

$INPUT$: M, ϕ (Modèle, formule)

$OUTPUT$: $\{s \in St \mid (M, s) \models \phi\}$

En logique modale, le model checking est décidable. Cela est dû au fait que le modèle de Kripke est assez simple. En effet, dans d'autres logiques qui utilisent des modèles plus complexes, au mieux la complexité augmente, et au pire cela devient indécidable.

Algorithme de model checking local Cet algorithme est assez rapide à écrire car il est très similaire à la sémantique du langage logique.

$MCL : Model \times State \times formula \mapsto Bool$

$MCL(M, s, \phi) =$
if $\phi \equiv p$, return $s \in V(\phi)$
if $\phi \equiv \neg\psi$, return $not(MCL(M, s, \psi))$
if $\phi \equiv (\psi \wedge \theta)$, return $MCL(M, s, \psi)$ and $MCL(M, s, \theta)$
if $\phi \equiv (\psi \vee \theta)$, return $MCL(M, s, \psi)$ or $MCL(M, s, \theta)$
if $\phi \equiv (\psi \rightarrow \theta)$, return $MCL(M, s, \theta)$ or not $(MCL(M, s, \psi))$
if $\phi \equiv \Box_i\psi$, return $(\forall s' \in St,$ if (sR_is') then $MCL(M, s', \phi))$
if $\phi \equiv \Diamond_i\psi$, return $(\exists s' \in St \mid sR_is'$ and $MCL(M, s', \phi))$

Algorithme de model checking global À partir de cet algorithme de model checking local, il est aisé d'obtenir l'algorithme pour le model checking global. En effet, il suffit de remplacer chaque opérateur de la logique par son correspondant en théorie des ensembles. Les opérateurs peuvent donc être remplacés comme suit : $\neg\phi$ devient $St \backslash MLC(\phi)$, $\wedge$ devient $\cap$, $\vee$ devient $\cup$, et la modalité de nécessité, $\Box_i\phi$, est transformée en l'ensemble des états s tels que tout état en relation (via R_i) avec s vérifie ϕ. La modalité de possibilité, $\Diamond_i\phi$, est transformée en l'ensemble des états s tels qu'il existe un état en relation avec s vérifiant ϕ. Plus de précisions peuvent être trouvées dans le livre [1].

Complexité Dans la logique modale, la complexité du model checking local est équivalente à la complexité du model checking global. Elle est en $\mathcal{O}(|\phi| \times |M|)$, où $|\phi|$ est le nombre d'opérateurs dans la formule et $|M|$ le nombre d'arêtes dans le modèle, c'est à dire le nombre de couples dans la relation R du modèle.

On peut voir dans l'algorithme que, à chaque appel de la fonction MCL, la taille cumulée des formules dans les appels récursifs est $|\phi| - 1$. Les seuls cas dérogeant à cette règle sont les cas où la formule est de la forme $\phi \equiv \Box_i\psi$ ou $\phi \equiv \Diamond_i\psi$. Dans ces cas, la taille des formules dans les appels récursifs est $|\phi| - 1$ et il y a maximum $|M|$ appels. La complexité du model checking est donc bien en $\mathcal{O}(|\phi| \times |M|)$.

6 Conclusion

Les notions discutées dans ce rapport constituent principalement une introduction à la logique modale ainsi qu'à une de ses instanciations, la logique épistémique. Le modèle présenté ici est assez rudimentaire, ce qui lui permet d'avoir des algorithmes (de model checking, de satisfiabilité, etc) de complexité relativement faible. Il existe de nombreux enrichissements de modèle qui n'ont pas été traités ici, comme prendre en compte des aspects quantitatifs, probabilistes, temporels, etc, ou encore plusieurs de ces traits simultanément (comme la logique épistémique dynamique). Bien sûr, enrichir le modèle signifie enrichir le langage pour pouvoir prendre en compte des nouveaux traits. Cela conduit à une augmentation de la complexité des algorithmes du problème de model checking et le rend même parfois indécidable. Il s'agit d'un domaine de recherche très actif car il est possible de modéliser de nombreux systèmes complexes, qui se révèlent être omniprésents.

Références

[1] Wojciech, Jamroga , Logical methods for specification and verification on multi-agent systems.