LRC - TME 8

Ce TME a pour but d'utiliser le logiciel LoTREC pour la logique modale minimale K, qui y est prédéfinie : il permet de mettre en œuvre des versions étendues de l'algorithme tableau pour cette logique, pour deux objectifs distincts :

- vérifier si une formule φ est satisfaite dans un monde w d'un modèle donné M (*model checking*), c'est-à-dire $M, w \models \varphi$
- tester la satisfiabilité d'une formule φ dans la logique K, c'est-à-dire examiner s'il existe $M = \langle W, R, I \rangle$ et $w \in W$ tel que $M, w \models \varphi$.

Comme dans le cas de la logique propositionnelle, l'interface comporte 4 onglets :

- l'onglet `Connectors` donne les différents connecteurs utilisables, en indiquant leur syntaxe ;
- l'onglet `Rules` donne les règles qui indiquent comment les tableaux sont construits : ré-écriture selon les connecteurs (règles α et β) et selon les modalités (règles additionnelles) ;
- l'onglet `Strategies` indique comment s'appliquent les règles, en particulier la priorité entre elles ;
- un onglet `Predefined Formulas`.

On rappelle que les formules sont écrites en notation préfixe: ainsi, la formule $\Box a \wedge \Box b$ s'écrit `and nec A nec B`.

Remarque : il est possible de zoomer sur les modèles construits en faisant clic droit + glisser.

1 Vérification de modèles dans la logique K

Cliquer sur l'onglet `Logic` puis sur `Predefined Logic` et choisir `Model-Checking-Monomodal`.

Exercices

1. Exécuter la stratégie en mode pas à pas et en examinant en détail chacune des règles :

   La toute première règle, qui s'intitule `ExampleOfModelAndFormula`, permet de définir le modèle de Kripke M avec lequel on souhaite travailler : ici elle construit 4 mondes en leur donnant les noms w, u, v et x, par la commande `createNewNode`. Elle établit ensuite les relations entre ces mondes, par la commande `link`. Le dernier argument de cette commande donne le nom de la relation, qui doit commencer par une majuscule et n'a d'importance que quand on gère plusieurs agents (cf TD et TME suivants). La commande `add` permet ensuite d'ajouter les variables propositionnelles dont l'interprétation est vraie dans un monde donné. Enfin, la règle définit la formule dont on souhaite examiner la satisfiabilité dans un monde donné en utilisant une action de la forme: `add` w `isItTrue` φ, pour demander si $M, w \models \varphi$ est une assertion vraie ou fausse.

   Après avoir exécuté cette première règle, LoTREC répond à cette question en deux phases :

   - dans la phase descendante (*top-down*), la formule initiale est décomposée, pas à pas, jusqu'à parvenir à des éléments atomiques;
   - dans la phase montante (*bottom-up*), la valeur de vérité est "remontée" jusqu'à la formule initiale.

   Pour chaque connecteur, il existe donc deux types de règles spécifiant la ré-écriture effectuée par LoTREC, que vous êtes invités à examiner dans le détail.

   Observer la façon dont la formule est itérativement décomposée par des questions posées dans le monde courant ou dans les mondes accessibles, puis la façon dont les réponses à ces questions sont agrégées pour remonter jusqu'à la formule initiale : examiner les règles, en étudiant leurs conditions et actions.

2. La règle `ExampleOfModelAndFormula` peut être éditée, pour supprimer ce qu'elle contient puis définir un nouveau modèle après avoir tout supprimé, cliquer sur `Add` dans la partie `Actions` et utiliser les trois actions suivantes, déjà présentées ci-dessus : `createNewNode` pour créer des nouveaux mondes ; `link` pour établir une relation entre deux mondes (nommer la relation R) ; et `add` pour ajouter des variables propositionnelles dont l'interprétation est vraie dans un monde donné. On peut enfin définir la formule dont on souhaite examiner la satisfiabilité dans un monde donné en utilisant une action de la forme: `add` w `isItTrue` φ, où w est le monde à partir duquel on souhaite vérifier la formule φ.

   Créer le modèle de Kripke correspondant à l'ex. 1 du TD 7 et vérifier les différentes formules en utilisant le mode pas à pas pour suivre les différentes étapes de ré-écriture.

3. Créer un modèle de Kripke M_1, en utilisant le moins de mondes possibles, où les deux contraintes suivantes sont vérifiées conjointement :

   - au moins un monde w où $M_1, w \models \neg p \wedge \Box\Diamond p$;
   - au moins un monde où $M_1, w \models q \wedge \Diamond^n \neg q$, où $\Diamond^n$ représente un nombre quelconque de modalités $\Diamond$ successives (tester pour quelques valeurs particulières de n et justifier pourquoi la réponse convient pour toute les valeurs).

4. Le connecteur d'implication n'est pas défini par défaut dans la logique prédéfinie `Model-Checking-Monomodal`.

   Créer un tel opérateur : le créer dans l'onglet `Connectors`, ajouter les règles permettant de le traiter dans l'onglet `Rules`, ajouter ces règles à la stratégie `Strategies`.

   Le tester pour la formule $p \rightarrow \Box(q \vee p)$ dans le monde w du modèle de la question 2.

2 Satisfiabilité dans la logique K

Cliquer sur l'onglet `Logic` et choisir `Predefined Logics` et `Monomodal-K`.

1. On examine d'abord la satisfiabilité de la formule $\Diamond p \wedge \Box\neg p$. Pour cela, entrer la formule dans le cadre des formules et cliquer sur `Satisfiability Check`. En partie droite de l'interface, le programme construit un graphe. Que constatez-vous? Interpréter ce graphe, en reprenant le raisonnement suivi pour traiter cette même formule en TD. Utiliser pour cela le mode pas à pas (`Step by step`).

2. Traiter ensuite les formules suivantes, d'abord sur papier, puis avec LoTREC, en interprétant les résultats obtenus et en comparant les modèles construits.

   (a) $p \wedge \Diamond(q \wedge \Box\neg p)$

   (b) $(p \wedge \neg p) \vee \Diamond\Diamond\Diamond p$

   (c) $(p \rightarrow \Diamond(q \vee \neg p)) \vee q$

3. Comment utiliser le programme pour montrer la validité de la formule $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$?

3 Bilan

Le compte-rendu de TME doit expliquer précisément les algorithmes mis en œuvre par LoTREC pour les logiques modales, pour les deux tâches, vérification de modèles et vérification de satisfiabilité, en détaillant des exemples de règles qu'il utilise, différentes de celles de l'algorithme tableau en logique propositionnelle.

Le compte-rendu doit également détailler la solution proposée pour intégrer le connecteur d'implication dans la logique `Model-Checking-Monomodal`.

LRC - TD 8

Ce TD se place sauf indication contraire, dans le cadre d'une logique épistémique dite S5, donc dotée des axiomes

- (K) : ~~si $\vdash F$, alors $\vdash KF$~~ $\vdash$ K(F -> G) -> (KF -> KG)
- (T) : $\vdash KF \to F$
- (4) : $\vdash KF \to KKF$
- (5) : $\vdash \neg KF \to K\neg KF$

Exercice 1 (Savoir que, savoir si, croire possible que...)
On rappelle que "savoir si p" équivaut à savoir p ou savoir $\neg p$, tandis que "croire possible p" équivaut à ne pas savoir que $\neg p$. Montrez formellement que les énoncés suivants sont vrais:

1. ne pas savoir si p, c'est croire possible que p soit vrai et croire possible que p ne soit pas vrai;
2. savoir que je ne sais pas p est équivalent à ne pas savoir (que) p.

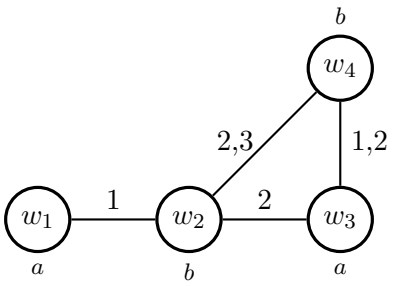
Exercice 2 (Connaître les résultats de l'élection)
Formaliser en logique épistémique les énoncés suivants:

1. Charlotte sait que Asma ou Ben connaissent le résultat de l'élection;
2. Asma croit possible que Charlotte sache que Ben connait le résultat de l'élection;
3. Ben sait qu'Asma sait qu'il ne connait pas le résultat;
4. Ben sait que si Charlotte a écouté la radio ou a appelé Asma, elle connait le résultat.

Exercice 3 (Trois femmes debout (sur un escalier))
On considère la situation suivante. Trois femmes sont sur différentes marches d'un escalier: A, sur la plus haute; B, sur la moyenne; et C, sur la plus basse; de telle façon que A voit le sommet de la tête de B et C, que B voit celui de C, et que C ne voit rien. Une quatrième femme passe devant elles et leur dit que l'une d'elles a un papillon sur la tête.

1. Modéliser cette situation par une structure de Kripke;
2. Que faut-il vérifier pour montrer que A sait si elle a le papillon sur la tête?
3. Même question pour montrer que C ne sait pas si elle a le papillon sur la tête?

Exercice 4 (Exemple sur une structure de Kripke)
On considère la structure de Kripke suivante (la relation est supposée symétrique et réflexive, la relation est donc non dirigée, et les arcs de réflexivité non spécifiés): $W = \{w_1, w_2, w_3, w_4\}$ et $R_1 = \{(w_1, w_2), (w_3, w_4)\}$, $R_2 = \{(w_2, w_4), (w_3, w_4), (w_2, w_3)\}$, et $R_3 = \{(w_2, w_4)\}$. Enfin $I(a) = \{w_1, w_3\}$, $I(b) = \{w_2, w_4\}$.



1. Les relations d'accessibilité des agents 1, 2 et 3 constituent-elles des relations d'équivalence?
2. Exprimer en logique épistémique les affirmations suivantes. Sont-elles vraies?
   (a) dans le monde w_1, l'agent 1 ne sait pas que l'agent 2 sait que a;
   (b) il existe un monde dans lequel l'agent 2 sait que l'agent 1 sait que a;
   (c) l'agent 2 sait que l'agent 3 sait b;
   (d) dans le monde w_1, l'agent 1 ne sait pas si l'agent 2 sait si a.

Exercice 5 (Le jeu des as et des huits) [tiré de "Reasoning About Knowledge", Fagin, Halpern, Moses, Vardi]

On considère un jeu de cartes dont les règles sont les suivantes: on prend les quatre as et les quatre huits d'un jeu. Trois joueurs se voient distribuer deux cartes chacun, qu'ils ne regardent pas et placent sur leur front, de manière à ce que les deux autres joueurs puissent les voir.

1. Enumérer les différents mondes possibles.
2. Représenter le modèle de Kripke correspondant à cette situation.
3. Identifier les mondes où un joueur peut directement identifier les cartes qu'il possède simplement en voyant les cartes des autres.