Vérification formelle du respect de valeurs morales dans les SMA

Grégory Bonnet, Bruno Mermet, Gaële Simon

Laboratoire GREYC - UMR 6072
Université du Havre, France
Bruno.Mermet@unicaen.fr

RÉSUMÉ. *L'utilisation croissante d'agents autonomes artificiels dans des secteurs comme le milieu hospitalier ou les transports amène à réfléchir au respect de règles morales partagées par tous. Le problème est d'autant plus crucial que les règles morales informellement validées par tous sont souvent incompatibles les unes avec les autres, et que ce sont souvent des règles éthiques qui amènent l'humain, suivant les circonstances, à privilégier telle ou telle règle morale. Utiliser la preuve pour vérifier qu'un agent respecte bien des règles morales et éthiques pourraient aider à accroître la confiance que nous pouvons avoir en de telles unités logicielles. Dans cet article, nous montrons, en nous appuyant sur une étude de cas, comment, à partir de règles morales a priori contradictoires mais ordonnées par des règles éthiques exprimant un système de valeurs, nous parvenons à définir un ensemble de propriétés formelles qui, lorsqu'elles sont établies par un agent, permettent d'assurer que l'agent en question respecte bien la règle éthique souhaitée.*

ABSTRACT. *The increasing use of autonomous artificial agents in hospitals or in transport control systems leads to consider whether moral rules shared by many of us are followed by these agents. This a particularly hard problem because most of these moral rules are often not compatible. In such cases, humans usually follow ethical rules to promote one moral rule or another. Using formal verification to ensure that an agent follows a given ethical rule could help in increasing the confidence in artificial agents. In this article, we show how a set of formal properties can be obtained from an ethical rule ordering conflicting moral rules with respect to a value system. If the behaviour of an agent verifies these properties (which can be proven using our existing proof framework), it means that this agent follows this ethical rule.*

MOTS-CLÉS : *spécification formelle, systèmes multi-agents, éthique computationelle.*

KEYWORDS: *formal specification, multi-agent systems, ethic, computational ethics.*

DOI:10.3166/RIA.31.449-470

1. Introduction

L'introduction d'agents autonomes artificiels dans des domaines comme le milieu hospitalier, le trading haute fréquence ou les transports pourrait soulever de nombreux

problèmes si ces agents ne sont pas en mesure de comprendre et suivre certaines règles morales. Par exemple, des agents capables de comprendre et d'utiliser le code de déontologie médicale pourraient s'appuyer sur des motivations éthiques afin de choisir quelles informations diffuser, à qui et sous quelles conditions, conformément au principe du secret médical. C'est pour cela que la communauté (Numérique d'Allistene, 2014 ; Life Institute, 2015) semble récemment faire preuve d'intérêt pour l'éthique des agents autonomes, comme en témoignent de nombreux articles (Beavers, 2011 ; McDermott, 2008 ; McLaren, 2006 ; Moor, 2006) et conférences [1].

Cet article se place dans le cadre du projet ETHICAA [2] dont l'objectif est de traiter de la gestion des conflits moraux et éthiques entre agents autonomes. Si les travaux dans le domaine de l'intelligence artificielle s'intéressent essentiellement aux questions de raisonnement et de décision éthique (Berreby *et al.*, 2015a ; Cointe *et al.*, 2016 ; Saptawijaya, Pereira, 2014 ; Tufis, Ganascia, 2012 ; Arkin, 2009 ; Coelho, Rocha Costa, 2009 ; Ganascia, 2007), il n'y a que peu de travaux s'intéressant à la vérification formelle de tels comportements. Or, la spécificité des codes moraux et éthiques est qu'ils peuvent ne pas être respectés en fonction des circonstances sans pour autant que cela pose problème. Par exemple dans un contexte humain, si voler est immoral, celui qui vole parce qu'il a faim n'est que rarement taxé d'immoralité.

Remarquons que dans le domaine du génie logiciel, les questions d'éthique ou de valeurs morales sont toujours présentes, que cela soit conscient ou non de la part des concepteurs (Nissenbaum, 2005 ; Shilton *et al.*, 2014). C'est pourquoi la *conception sensible aux valeurs* ou *Value Sensitive Design* se propose d'identifier clairement durant la phase de conception des valeurs sociales ou morales désirées et de s'assurer que les logiciels ou systèmes conçus les respectent bel et bien (Friedman, 1996 ; Friedman *et al.*, 2013). Cependant, là encore, il n'y a pas de travaux s'intéressant à la vérification formelle *a posteriori* de ces exigences.

Aussi, cet article propose un environnement de spécification et de vérification formelle du comportement éthique d'un agent autonome. En reprenant les travaux d'Abramson et Pike, une règle morale est représentée par une propriété formelle qu'un agent doit respecter (Abramson, Pike, 2011) et un agent a un comportement éthique s'il respecte toutes les règles morales attendues selon les circonstances. Cet aspect circonstanciel des règles à vérifier est spécifique à la vérification de comportement éthique, et c'est le principal verrou scientifique que nous tâchons de lever dans cet article. Ainsi, nous envisageons deux contextes d'utilisation de notre travail :

– fournir des outils pour développer des agents ayant un comportement se conformant à une certaine éthique faisant intervenir certaines valeurs morales, et donc certaines règles morales ;

1. Symposium on Roboethics, International Conference on Computer Ethics and Philosophical Enquiry, Workshop on AI and Ethics, International Conference on AI and Ethics.

2. http://ethicaa.org/

– fournir des outils pour évaluer si le comportement d'un agent développé hors de tout contexte éthique est bien conforme à une certaine éthique.

Ceci soulève des problèmes théoriques. En effet, considérant qu'une règle morale peut être représentée par une formule $\mathcal{F}$ en logique du premier ordre suffisamment expressive pour bien des cas pratiques, nous cherchons alors à établir qu'un agent a un comportement éthique si son comportement vérifie $\mathcal{F}$ ou qu'il n'a pas un comportement éthique si son comportement ne vérifie pas $\mathcal{F}$. Toutefois, un tel système logique est simplement semi-décidable : il n'est pas possible de prouver dans le cas général qu'un système ne vérifie pas une formule $\mathcal{F}$. En effet, si un prouveur automatique ne réussit pas à prouver qu'un agent vérifie bien une formule $\mathcal{F}$, il n'est pas possible d'identifier automatiquement si cela est dû au fait que l'agent ne vérifie pas $\mathcal{F}$ ou si c'est parce que le prouveur n'a pas réussi à faire la preuve du contraire. Aussi, nous proposons d'utiliser un cadre formel qui réduise au maximum le nombre de formules correctes qui ne sont pas prouvées automatiquement.

Cet article est structuré comme suit. Nous présentons en section 2 des systèmes formels de vérification en nous attardant plus spécifiquement sur ceux consacrés aux SMA avant de définir ce que nous appelons règles morales et éthiques. Dans la section 3, nous détaillons plus précisément le système formel, puis nous présentons son utilisation dans un contexte éthique en section 4.

2. État de l'art

Depuis les débuts de l'informatique, la nécessité de garantir la correction d'un logiciel est apparue comme étant un problème majeur pour les développeurs. Cette nécessité est devenue vitale pour les systèmes critiques, c'est-à-dire les applications dédiées à des domaines où la sûreté de fonctionnement est vitale (comme les transports par exemple). Cependant, vérifier formellement un logiciel est un processus long et difficile qui va à l'encontre des critères de rentabilité et d'efficacité de certaines entreprises. Il existe principalement deux grands types de procédés de validation : le test et la preuve. Ici, nous ne nous intéresserons qu'à cette dernière. Les preuves peuvent être effectuées soit par des *model-checkers*, soit par des prouveurs de théorèmes qui peuvent ponctuellement faire appel à des *model-checkers*. Les *model-checkers* reposent sur un principe de test exhaustif, tandis que les prouveurs de théorème utilisent le calcul des séquents pour essayer, de manière heuristique, de générer des démonstrations.

Le processus de preuve peut être long et difficile, mais il permet de prouver des spécifications très préliminaires et de les raffiner progressivement jusqu'à l'obtention d'un code exécutable, avec des preuves progressives. Les erreurs sont ainsi détectées plus tôt, réduisant leur coût. Les formules à prouver sont également simplifiées, facilitant leur démonstration automatique. Ces preuves reposent alors sur une spécification formelle écrite grâce à un langage, un modèle ou une méthode formelle (ces termes sont souvent employés indifféremment pour exprimer la même chose).

2.1. Modèles et méthodes dédiés aux SMA

Le but premier des modèles dédiés aux SMA était d'aider les développeurs à concevoir des SMA. Bien qu'il en existe de nombreux modèles (Sabas *et al*., 2002), le plus célèbre est certainement le modèle BDI (Rao, Georgeff, 1995) devenu aujourd'hui un standard aux nombreuses extensions, comme l'architecture BOID qui ajoute la notion d'*obligation* aux notions de croyance, de désir et d'intention du modèle BDI (Broersen *et al*., 2001).

Deux des premières méthodes formelles dédiées aux SMA furent MetateM (Fisher, 1994) et Desire (Brazier *et al*., 1997). Cependant, aucun des deux ne permet de spécifier les propriétés que le système doit garantir. Au contraire, les méthodes reposant sur la notion de rôle introduisent une abstraction qui aide à effectuer la phase d'analyse des besoins dans le travail de développement. Dans Gaia, un SMA est spécifié deux fois : par son comportement (via des propriétés de vivacité) et par des propriétés d'invariance. Aussi, cette méthode introduit les bases nécessaires pour effectuer des preuves de comportement. Cependant, effectuer de telles preuves n'est malheureusement pas possible avec Gaia car les propriétés ne sont pas associées aux agents mais aux rôles et, lorsqu'un agent implante plusieurs rôles, aucune sémantique formelle ne vient déterminer comment ces rôles se combinent.

Une autre famille de méthodes est celle des méthodes orientées buts. La plupart se situent au niveau des agents et non au niveau du SMA, ce qui implique que la phase d'agentification doit avoir été effectuée préalablement. On peut cependant trouver deux exceptions : Moise (Hubner *et al*., 2002) et PASSI (Cossentino, Potts, 2002). Par exemple dans PASSI, les types d'agents sont produits en regroupant des *cas d'utilisation* identifiés lors de la phase d'analyse. Il n'y a cependant pas de guide pour grouper ces cas d'utilisation.

Enfin, plus récemment, Dastani *et al.* ont proposé le langage 2APL (Dastani, 2008). Malheureusement, ce langage formel n'inclut pas de système de preuve. De plus, 2APL n'est pas compositonnel, ce qui rend le système trop monolithique s'il devait être utilisé dans un contexte de preuve.

2.2. Travaux centrés sur la preuve

Comme cela a été évoqué précédemment, il y a principalement deux façons de vérifier la correction d'une spécification : le *model-checking* et la preuve de théorèmes.

La plupart des travaux menés sur les agents utilisent le *model-checking* (Bordini *et al*., 2003 ; Alechina *et al*., 2004 ; Raimondi, Lomuscio, 2004 ; Kacprzak *et al*., 2004). Cependant, tous ces travaux partagent la même limite : l'explosion combinatoire des trajectoires possible du système rend la preuve de SMA complexes difficile, voire impossible. Ces systèmes se réduisent d'ailleurs la plupart du temps à de la preuve sur des formules propositionnelles et non sur des prédicats.

Les travaux sur l'utilisation de la preuve de théorèmes pour la vérification de SMA sont beaucoup plus rares. La principale raison est certainement due au fait que, la logique du premier ordre étant semi-décidable, les tentatives de preuves sont faites en utilisant des heuristiques et la preuve d'une propriété juste peut échouer. Cependant, de nombreux prouveurs de théorèmes peuvent maintenant réussir à prouver des propriétés très complexes automatiquement. C'est notamment le cas de PVS (Owre *et al.*, 1992) et de KRT, le prouveur de l'atelier B (Abrial, 1996).

Une autre approche intéressante utilise le raisonnement abductif (Russo *et al.*, 2001). Cependant, son modèle, basé sur les événements, la rend peu facile à utiliser pour les SMA.

D'autres modèles existent, reposant sur la programmation logique. C'est notamment le cas de CaseLP (Martelli *et al.*, s. d.) et DCaseLP (Baldoni *et al.*, 2005), plutôt adaptés à la preuve de théorèmes. Cependant, les seules preuves qui semblent applicables sur ces modèles semblent être des preuves de protocoles d'interaction. Congolog (de Giacomo *et al.*, 2000) et CASL (Shapiro *et al.*, 2002) sont aussi deux langages intéressants, reposant sur le calcul de situation. De plus, ils permettent tous les deux d'effectuer des preuves. Cependant, ces preuves ne concernent que l'enchaînement des actions, pas leur sémantique.

2.3. Règles morales et éthiques

Dans la littérature, que cela soit en philosophie ou plus récemment en neurologie et sciences cognitives (Damasio, 2008 ; Greene, Haidt, 2002), les concepts de morale et d'éthique ont été discutés. Bien que ces termes désignent initialement une même idée, certains auteurs les nuancent (Comte-Sponville, 2012 ; Ricoeur, 1990 ; Timmons, 2012). En effet, la morale établit des règles permettant d'évaluer comme bien ou mal des situations ou des actions et l'éthique permet de concilier les règles morales lorsqu'elles entrent en conflit ou posent des problèmes dans leur applicabilité, en s'appuyant parfois sur un système de valeurs. C'est sur ces distinctions que nous fondons nos définitions.

2.3.1. Règles morales

Ainsi, nous définissons une règle morale comme une règle qui doit préciser, dans un contexte donné, quels sont les états du système qui sont bien ou mal. Nous pouvons alors représenter une règle morale sous la forme $contexte \rightarrow P_{var}$, où P_{var} est un prédicat défini sur l'ensemble des variables connues. Une règle morale peut donc être vue comme une règle d'invariance conditionnelle particulière, dans le sens où sa vérification n'est pas nécessaire pour garantir un fonctionnement correct du système, mais qu'il est nécessaire de la vérifier si l'agent doit être utilisé dans un système où la règle morale en question s'applique. Par exemple, dans le cadre d'une voiture autonome, on peut considérer qu'une propriété de sûreté serait $voie = Autoroute \rightarrow vitesse \leq 130$. La voiture **doit** vérifier cette propriété. Cependant, afin de ne pas mettre de vie en danger, une règle morale de prudence r_p précise qu'en cas de ver-

glas, la voiture ne doit pas dépasser 30 km/h. Nous la notons formellement ainsi : $temps = verglas \rightarrow vitesse \leq 30$. Cette propriété n'est pas indispensable pour que la voiture ait un comportement valide, mais elle prend du sens dans un système où les utilisateurs considèrent la préservation des vies humaines.

2.3.2. Valeurs morales

Comme nous l'avons vu précédemment, les règles morales sont couramment soutenues et justifiées par des valeurs morales (liberté, bienveillance, sagesse, conformisme, etc.). Psychologues, sociologues et anthropologues admettent pour la plupart que les valeurs morales sont l'élément central dans l'évaluation de la justesse d'une action, d'une personne ou d'un évènement (Rokeach, 1973 ; Swchartz, Bilsky, 1990 ; Schwartz, 2012). Par exemple, la règle morale présentée précédemment repose sur la valeur morale de *prudence*. Bien sûr, différentes règles morales peuvent reposer sur la même valeur morale. De plus, les valeurs semblent être universelles et exister en nombre fini. Elles sont ordonnées dynamiquement au sein d'un *système de valeurs* par ordre d'importance relative au regard d'un contexte particulier (van Marrewijk, Werre, 2003 ; Wiener, 1988).

2.3.3. Règles éthiques

Lorsqu'un individu ou un agent se place dans un cadre avec plusieurs règles morales, il arrive que deux de ces règles, ou plus, soient en conflit. Une règle éthique permet notamment, dans un tel cas, de préciser comment agir. Si certaines règles – comme la doctrine du double effet (McIntyre, 2014) – peuvent être des inférences complexes, nous considérons ici qu'une règle éthique doit alors permettre d'exprimer un système de valeur, c'est-à-dire comment ordonner des règles morales en fonction d'un ordre de préférence dynamique sur les valeurs morales.

Nous considérons également que, suivant les circonstances, une règle éthique peut donner des résultats différents. Si nous reprenons l'exemple du véhicule autonome, une règle morale r_r, sous-tendue par le concept de respect des autres conducteurs, consiste également à préciser que sur autoroute, il ne faut pas rouler à moins de 80 km/h, ce qui s'exprime formellement par $voie = Autoroute \rightarrow vitesse \geq 80$. Nous avons alors un conflit entre la règle de prudence r_p évoquée précédemment et cette nouvelle règle r_r : s'il y a du verglas et que la voiture est sur l'autoroute, sa vitesse doit être inférieure à 30km/h d'après r_p, mais supérieure à 80 d'après r_r. Une règle éthique peut alors préciser que, dans tous les cas, la prudence (exprimée par r_p) est préférable au respect (exprimé par r_r). Mais une règle éthique pourrait aussi préciser que si on est en route pour les urgences, alors l'ordre de priorité des valeurs morales de prudence et de respect est inversé.

Dans les travaux présentés ici, nous supposerons que l'ordre exprimé par la règle éthique est un ordre total. Par ailleurs, nous prenons cet ordre comme un donnée d'entrée de nos travaux. Sa construction sort du cadre de cet article.

2.4. Peu de travaux pour vérifier l'éthique

Traiter de la modélisation et de la prise de décision éthique est abordé dans la littérature sous deux angles principaux.

– Le raisonnement moral est tout d'abord souvent modélisé par des logiques déontiques (Bringsjord *et al.*, 2016 ; Chisholm, 1963 ; Horty, 1994) ou des logiques non-monotones (Berreby *et al.*, 2015b ; Ganascia, 2007). Ces travaux s'attachent en particulier à exprimer des théories morales spécifiques (raisonnement aristotélicien, raisonnement kantien, etc.). On peut notamment citer (Calardo *et al.*, 2015) qui décrit une logique permettant de raisonner sur les préférences, ou encore (Brewka *et al.*, 2004), qui permet d'exprimer des choix prioritaires. Une des approches abstraites les plus abouties s'appuie sur les *systèmes d'argumentation valués* décrivant des valeurs associées à des arguments logiques et des préférences sur ces valeurs (Bench-Capon, Atkinson, 2009) en tenant parfois compte des autres agents du système (Atkinson, Bench-Capon, 2016).

– D'autres travaux s'appuient sur la théorie de la décision individuelle ou collective. Dans ces approches, le caractère éthique d'une action est généralement représenté par des poids qui sont aggrégés (Anderson, Anderson, 2014) puis comparé à un critère de décision (généralement une maximisation de l'utilité). Dans un contexte collectif, ces critères expriment une notion d'équité. Par exemple, alors que les classiques valeur de Shapley ou index de Banzhaf expriment une équité pure, d'autres mesures comme les index de Gini ou les valeurs de solidarité (Nowak, Radzik, 1994) expriment des comportements tendant à aider les agents les moins bien lotis par la décision. Plus récemment, (Abel *et al.*, 2016) ont proposé une approche d'apprentissage par renforcement, s'appuyant donc sur des algorithmes de décision séquentielle, pour apprendre le caractère éthique d'une action. Ces travaux combinent des fonctions d'utilité données *a priori* et des requêtes auprès d'un utilisateur humain permettant à l'agent de mettre à jour son modèle d'utilité.

Dans tous les cas, ces travaux s'intéressent à la modélisation *a priori* de l'éthique et à la prise de décision. En revanche, ils ne s'intéressent pas à la vérification formelle du comportement d'un agent. De plus, ces propositions ne permettent pas de raisonner sur la sémantique de ce qu'expriment l'éthique sur laquelle les choix sont faits. Cela ne permet donc pas de vérifier formellement qu'un état du système vérifie bien les valeurs morales préférées dans la situation courante.

Traiter les problèmes de vérification formelle de l'éthique est notamment considéré dans (Abramson, Pike, 2011). Dans cet article, les auteurs expliquent pourquoi utiliser des méthodes formelles pourrait être intéressant pour garantir que des agents respectent des règles éthiques. Cependant, il ne s'agit que d'une prise de position : aucune méthode concrète n'est proposée. Dans (Dennis *et al.*, 2015), les auteurs proposent de formaliser et de vérifier la procédure de décision éthique suivante, décrite dans (Winfield *et al.*, 2014) : quand un choix doit être effectué entre plusieurs actions, une valeur est affectée à chaque action potentielle selon le niveau de sûreté garanti par l'action. Si une action est plus sûre qu'une autre, c'est elle qui est exécutée. Il y a

cependant un inconvénient majeur au système : l'aspect éthique n'est pris en compte que lorsqu'un choix entre des actions doit être effectué en utilisant la procédure de sélection d'action décrite précédemment. Ainsi, ce travail n'est pas assez général pour fournir une réponse satisfaisante. En effet, cela ne peut être valable que pour des règles portant sur des choix ponctuels d'action, et non sur des règles correspondant à des aspects plus généraux du comportement des agents.

3. GDT4MAS

Pour prendre en considération les problèmes éthiques dans la vérification formelle, nous avons choisi d'utiliser la méthode GDT4MAS (Mermet, Simon, 2011 ; 2013). En effet, cette méthode, qui intègre également un modèle, présente plusieurs caractéristiques intéressantes pour la prise en compte de problèmes éthiques :

– Cette méthode propose un langage formel pour exprimer non seulement les propriétés qu'un agent ou qu'un SMA doit respecter, mais aussi le comportement des agents;

– Les propriétés sont spécifiées en logique du premier ordre, une notation formelle bien connue et expressive;

– Le processus de preuve peut être effectué automatiquement.

Nous présentons rapidement dans la suite la méthode GDT4MAS. Plus de détails peuvent être trouvés dans les articles cités ci-dessus.

3.1. Concepts principaux

La méthode GDT4MAS nécessite de spécifier trois notions : l'environnement, les types d'agents et les agents eux-mêmes, qui sont des instances des types d'agents. Dans la suite de cette section, nous décrivons brièvement chacune de ces parties.

3.1.1. Environnement

L'environnement est spécifié par un ensemble de variables typées et une propriété d'invariance $i_{\mathcal{E}}$.

Exemple : *luminosite* et *eclipse* pourraient être des variables d'environnement d'un agent avec les typages : $luminosite \in \{jour, nuit\} \wedge eclipse \in Bool$ et la propriété d'invariance $eclipse \rightarrow luminosite = nuit$.

3.1.2. Types d'agents

Les types d'agents sont chacun spécifiés par un ensemble de variables typées, un invariant et un comportement. Le comportement d'un agent est principalement défini par un *arbre de décomposition des buts* (GDT [3]). Un GDT est un arbre de buts, dont

3. Goal Decomposition Tree.

la racine correspond au but principal de l'agent. Un plan est associé à chaque but : lorsque ce plan est exécuté avec succès il établit le but auquel il est associé. Un plan peut être constitué soit d'une simple action, soit d'un ensemble de buts reliés par un *opérateur de décomposition*.

Exemple : Un but "rentrer dans un pièce" pourrait être décomposé en 1 plan constitué de 2 sous-buts : "ouvrir la porte" puis "entrer". Quant au but "entrer", sa résolution pourrait se ramener à un plan se réduisant à la seule action "avancer".

Un but B est principalement décrit par un nom n_B, une condition de satisfaction sc_B et une propriété garantie en cas d'échec gpf_B. La condition de satisfaction (SC [4]) d'un but est spécifiée formellement par une formule qui doit être satisfaite lorsque l'exécution du but réussit. Dans le cas contraire, la propriété garantie en cas d'échec (GPF [5]) d'un but spécifie ce qui est malgré tout garanti lorsque l'exécution d'un plan associé à un but échoue (ceci n'a pas de sens dans le cas d'un but dont le plan réussit toujours, ce qui est appelé un but NS [6]).

Exemple : Le but "ouvrir la porte" la porte, quand il est résolu avec succès à pour conséquence que la porte est ouverte, mais que l'agent se trouve toujours devant la porte. Sa condition de satisfaction pourrait être alors : $SC \equiv statutPorte = ouverte \wedge positionAgent = devantPorte$. Mais la résolution de ce but peut échouer (porte verrouillée ou coincée, agent maladroit, etc.). Dans ce cas, on ne sait pas grand chose du statut de la porte, mais on sait que l'agent, lui, n'a toujours pas bougé. La propriété garantie en cas d'échec est donc $GPF \equiv positionAgent = devantPorte$.

Les SC et les GPF sont des *formules de transition d'état* (STF [7]) car elles expriment la relation entre deux états, appelés état initial et état final. Par la suite, nous utiliserons le terme *STF non déterministe* quand, pour un état initial donné, plusieurs états finaux peuvent satisfaire la STF. Par exemple, la formule $x' > x$ est une STF non déterministe car, pour un état initial donné (comme $x = 0$), plusieurs états finaux peuvent la satisfaire (comme $x' = 2$ ou $x' = 10$).

Exemple : Dans le cas du but "ouvrir la porte", la GPF pourrait être représentée par : $GPF = statutPorte' \neq ouverte \wedge positionAgent' = positionAgent$. En effet, la porte n'est pas ouverte, mais on ne sais pas son statut (entrebaillée, ouverte, par exemple) exacte. Quant à la position de l'agent, on sait qu'il n'a pas bougé, mais il n'était pas forcément devant la porte quand il a effectué l'action (il pouvait être sur le côté par exemple, ou bien plus loin mais il avait une télécommande permettant normalement d'ouvrir la porte, etc.).

Un type d'agent peut être paramétré, c'est-à-dire utiliser des variables dont les valeurs d'initialisation seront spécifiques à chaque agent instance du type en question.

---

4. Satisfaction Condition
5. Guaranted Property in case of Failure.
6. Necessarily Satisfiable.
7. State Transition Formula.

3.1.3. Agents

Les agents sont spécifiés comme des instances de types d'agents. Si le type est paramétré, alors on spécifie pour chaque agent des valeurs effectives pour chacun des paramètres des types d'agents.

3.2. Exemple de GDT

La figure 1 montre un exemple de GDT. Sur ce schéma, les ellipses représentent des buts et les flèches des actions. Le but de ce comportement est d'allumer la lampe d'une pièce n (n est un paramètre du GDT). Pour procéder ainsi, l'agent essaie de rentrer dans la pièce. En effet, une cellule photo-électrique est censée détecter quand quelqu'un entre dans la pièce et allume alors la lampe, ce qui semble donc être un plan valable. Pourtant, la cellule photo-électrique ne marche pas systématiquement (c'est pourquoi le but *Entering into the room* est NNS, c'est-à-dire non NS), et l'agent peut donc devoir utiliser l'interrupteur. Plus de détails peuvent être trouvés dans (Mermet, Simon, 2013).

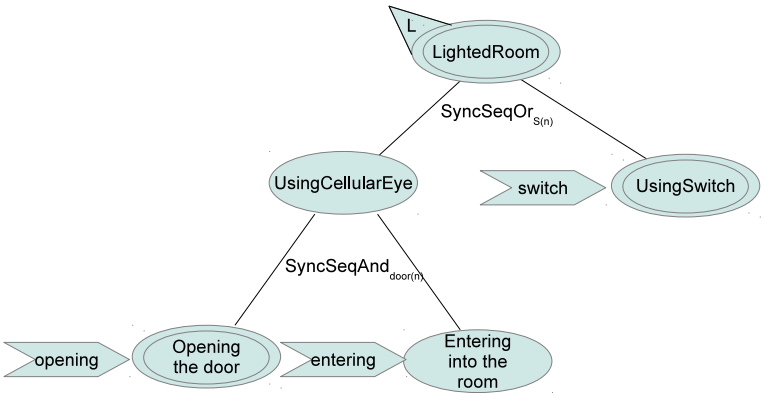

Figure 1. Exemple de GDT

3.3. Principes de preuve

Le mécanisme de preuve proposé par GDT4MAS a pour but de prouver les propriétés suivantes :

– les agents préservent leurs propriétés d'invariance (Mermet, Simon, 2013) qui sont des propriétés précisant que l'agent reste dans des états valides ;

– le comportement des agents est consistant (autrement dit, les plans associés aux buts sont corrects) ;

– les agents établissent leurs propriétés de vivacité, qui formalisent certaines caractéristiques dynamiques attendues de l'agent.

Le mécanisme de preuve repose sur quelques principes essentiels : les *obligations de preuve* (propriétés à prouver pour garantir la correction du système) peuvent être générées automatiquement à partir d'une spécification GDT4MAS. Elles sont exprimées en logique du premier ordre et peuvent être vérifiées par tout prouveur adéquat. Enfin, le système de preuve est compositionnel : la preuve de la correction d'un agent est décomposée en plusieurs petites obligations de preuves indépendantes.

4. Prouver une éthique

Un agent suivant une règle éthique est un agent qui doit vérifier certaines règles morales, mais pas tout le temps, puisque cela est *a priori* impossible (sinon, il n'y aurait pas de dilemme éthique). Cette situation n'est pas une situation prévue par les systèmes de vérification formelle, qui cherchent à vérifier qu'une propriété est toujours vraie. C'est pourquoi, dans ce travail, nous montrons comment transformer les règles morales en propriétés "toujours vraies" qu'un système de vérification formelle pourra vérifer.

4.1. Caractérisation du problème

Considérons un agent ag dont le comportement a été formellement spécifié et vérifié, par rapport aux propriétés qu'il doit garantir. Supposons maintenant que cet agent doit être utilisé dans un univers donné avec une règle éthique er reposant sur un ensemble de règles morales. La question qui nous intéresse est la suivante : est-ce que ag vérifie la règle éthique er ?

Comme GDT4MAS permet notamment de prouver les propriétés d'invariance, nous proposons d'exprimer les règles morales et les règles éthiques sous forme de telles propriétés. Il s'avère que la plupart des règles morales peuvent être facilement traduites en propriétés d'invariance. Nous proposons de structurer chaque règle morale ainsi :

$$(val, \{(when_i, \{(var_i, set_i)\})\})$$

Cela signifie que chaque règle morale est spécifiée par la valeur morale qui la sous-tend (val) et contraint, dans différents contextes ($when_i$), les valeurs (set_i) qui peuvent être assignées à différentes variables (var_i). Ainsi, si nous reprenons l'exemple de la règle de prudence r_p donnée dans la section 2.3, nous pourrions exprimer cette règle ainsi :

$$(prudence, \{(temps = Verglas, \{(vitesse, \{0 \dots 30\})\})\})$$

Cependant, exprimer une règle éthique sous la forme de propriétés d'invariance n'est pas si évident. En effet, Une telle règle ne caractérise pas des états du système mais fournit des ordres de priorité sur les règles morales suivant différents contextes.

Soit MV l'ensemble des valeurs morales qui interviennent dans le cadre éthique auquel l'agent doit se conformer (dans notre exemple, $MV = \{prudence, respect\}$)

et soit $\mathcal{P}$ l'ensemble des prédicats définissables sur les variables qui peuvent être vues par un agent donné a. Une règle éthique er est définie ainsi :

$$er \in \mathcal{P} \mathrel{+\!\!>} (1 \ldots card(MV) \mathrel{>\!\!\!>\!\!>} MV)$$

Ici, $X \mathrel{+\!\!>} Y$ représente l'ensemble des fonctions partielles de X vers Y et $X \mathrel{>\!\!\!>\!\!>} Y$ représente l'ensemble des bijections de X sur Y. Ainsi, informellement, cette définition signifie que, dans certains cas caractérisés par un prédicat donné $p \in \mathcal{P}$, les valeurs morales MV sont ordonnées. Par exemple, si $p \in \mathcal{P}$ est un prédicat, $er(p)(1)$ définit la valeur morale la plus prioritaire lorsque p est vrai, $er(p)(2)$ définit la seconde la plus prioritaire, et ainsi de suite.

Considérons l'exemple suivant : soit un agent A_1 qui doit décider la couleur d'un feu de circulation $tl1$ affecté à une route $r1$ à un croisement avec une route $r2$. Dans le système dans lequel cet agent agit, deux règles morales s'appliquent : la première, $mr1$, associée à la valeur morale de prudence, exprime que, pour éviter les accidents, lorsque la couleur du feu de la route $r2$ est $vert$ ou $orange$ alors la couleur de $tl1$ ne peut pas être $vert$. La règle $mr1$ peut alors être formalisée ainsi :

$$(prudence, \{(tl2 \in \{green, orange\}, \{tl1, \{orange, red\}\})\})$$

La seconde règle morale, $mr2$, associée à la valeur morale d'urgence, exprime que la route $r1$ est une route très prioritaire et qu'ainsi, la couleur de $tl1$ doit être toujours $vert$. Cela peut être formalisé ainsi :

$$(urgence, \{(true, \{tl1, \{green\}\})\})$$

Bien évidemment, ces deux règles ne peuvent pas toujours être toutes les deux vérifiées. C'est notamment le cas lorsque le deuxième feu est $vert$: d'après $mr1$, la couleur de $tl1$ doit être $orange$ ou $rouge$ alors que selon $mr2$, la couleur de $tl1$ doit être $vert$.

Supposons maintenant que dans le système considéré, une règle éthique er donne des priorités aux valeurs morales. Par exemple, supposons que er précise que la valeur de prudence est plus importante que la valeur d'urgence. Formellement, cela se traduirait par la règle éthique suivante :

$$\{(true, \{(1, prudence), (2, urgence)\})\}$$

4.2. Solution proposée

Dans le cadre de nos travaux, nous souhaitons vérifier qu'un agent vérifie bien une règle éthique définie sur un certain nombre de règles morales. Or un tel agent ne peut pas vérifier l'ensemble des règles morales qui le concernent puisque, comme

nous l'avons expliqué précédemment, il arrive que ces règles soient contradictoires et qu'un état du système accessible par l'agent ne puisse vérifier toutes les règles morales. Aussi, afin de garantir qu'un agent respecte une règle éthique donnée, nous proposons un *système de transformation de prédicats* qui transforme les prédicats associés aux règles morales en d'autres prédicats, prouvables de façon à tenir compte de la règle éthique qui s'applique. Dans l'étude présentée ici, nous ne considérons qu'une situation avec deux règles morales mais le principe proposé pourrait être appliqué à un système avec plus de règles morales. Les règles morales et la règle éthique vont être transformées en un ensemble de propriétés d'invariance. Dans la suite, la transformation est montrée dans le cas où une seule variable est affectée par les règles morales. Dans le cas général, il faut générer les mêmes formules pour chaque variable apparaissant dans l'ensemble des règles morales (si une variable n'apparaît que dans certaines règles morales, elle est ajoutée dans les autres avec comme unique "contrainte" l'appartenance à son domaine de définition).

Considérons une telle variable V. Supposons également que la règle morale $mr1$ fournit les contraintes suivantes sur V :

$$mr1 = \left(val_1, \left\{ \begin{array}{l} (when_{mr1_1}, (V, set_{mr1_1})) \\ (when_{mr1_2}, (V, set_{mr1_2})) \end{array} \right\} \right)$$

Supposons qu'une seconde règle morale $mr2$ fournit les contraintes suivantes sur V :

$$mr2 = \left(val_2, \left\{ \begin{array}{l} (when_{mr2_1}, (V, set_{mr2_1})) \\ (when_{mr2_2}, (V, set_{mr2_2})) \\ (when_{mr2_3}, (V, set_{mr2_3})) \end{array} \right\} \right)$$

Par la suite, nous dirons qu'une règle morale mr s'appliquent (et nous noterons $app(mr)$) si et seulement si l'une des clauses $when$ associées à la règle eset vraie. Ainsi, nous avons :

$$app(mr1) \equiv when_{mr1_1} \vee when_{mr1_2}$$
$$app(mr2) \equiv when_{mr2_1} \vee when_{mr2_2} \vee when_{mr2_3}$$

Enfin, considérons une règle éthique spécifiant que sous la condition $cond_1$, la valeur morale val_1 est prioritaire par rapport à val_2, tandis que c'est le contraire sous la condition $cond_2$. Cette règle éthique er est donc définie ainsi :

$$er = \left\{ \begin{array}{l} (cond_1, \{(1, val_1), (2, val_2)\}) \\ (cond_2, \{(1, val_2), (2, val_1)\}) \end{array} \right\}$$

De la règle éthique et des valeurs morales associées aux règles morales, nous déduisons une *règle éthique modifiée* mer établissant un ordonnancement des règles morales :

$$mer = \left\{ \begin{array}{l} (cond_1, \{(1, mr1), (2, mr2)\}) \\ (cond_2, \{(1, mr2), (2, mr1)\}) \end{array} \right\}$$

Dans le cas où plusieurs règles morales sont associées à la même valeur morale, nous supposons qu'elles ne sont *a priori* pas contradictoires. Du coup, l'ordre dans lequel nous les classons l'une par rapport à l'autre est sans importance.

Nous générons alors un ensemble de propriétés d'invariance prouvables. Pour l'exemple, nous obtenons tout d'abord deux propriétés qui résultent du fait que sous $cond_1$, $mr1$ est une règle prioritaire :

$$\begin{array}{l} cond_1 \rightarrow (when_{mr1_1} \rightarrow V \in set_{mr1_1}) \\ cond_1 \rightarrow (when_{mr1_2} \rightarrow V \in set_{mr1_2}) \end{array}$$

Puis il faut prendre en compte que, sous $cond_1$, quand $mr1$ ne s'applique pas, $mr2$ s'applique :

$$\begin{array}{l} cond_1 \rightarrow \left(\begin{array}{l} (\neg when_{mr1_1} \wedge \neg when_{mr1_2}) \\ \rightarrow \\ (when_{mr2_1} \rightarrow V \in set_{mr2_1}) \end{array} \right) \\ cond_1 \rightarrow \left(\begin{array}{l} (\neg when_{mr1_1} \wedge \neg when_{mr1_2}) \\ \rightarrow \\ (when_{mr2_2} \rightarrow V \in set_{mr2_2}) \end{array} \right) \\ cond_1 \rightarrow \left(\begin{array}{l} (\neg when_{mr1_1} \wedge \neg when_{mr1_2}) \\ \rightarrow \\ (when_{mr2_3} \rightarrow V \in set_{mr2_3}) \end{array} \right) \end{array}$$

Enfin, il reste à définir les propriétés qui expriment que lorsque $cond_1$ est vraie, quand mr_1 et mr_2 s'appliquent, si c'est possible, une valeur satisfaisant les 2 règles morales doit être choisie :

$$\left(\begin{array}{l} (cond_1 \wedge when_{mr1_1} \wedge when_{mr2_1}) \\ \rightarrow \\ (set_{mr1_1} \cap set_{mr2_1} \neq \emptyset \rightarrow V \in set_{mr1_1} \cap set_{mr2_1}) \end{array} \right)$$

$$\left(\begin{array}{l} (cond_1 \wedge when_{mr1_1} \wedge when_{mr2_2}) \\ \rightarrow \\ (set_{mr1_1} \cap set_{mr2_2} \neq \emptyset \rightarrow V \in set_{mr1_1} \cap set_{mr2_2}) \end{array} \right)$$

$$\left(\begin{array}{l} (cond_1 \wedge when_{mr1_1} \wedge when_{mr2_3}) \\ \rightarrow \\ (set_{mr1_1} \cap set_{mr2_3} \neq \emptyset \rightarrow V \in set_{mr1_1} \cap set_{mr2_3}) \end{array} \right)$$

$$\left(\begin{array}{l}(cond_1 \wedge when_{mr1_2} \wedge when_{mr2_1}) \\ \rightarrow \\ (set_{mr1_2} \cap set_{mr2_1} \neq \emptyset \rightarrow V \in set_{mr1_2} \cap set_{mr2_1})\end{array}\right)$$

$$\left(\begin{array}{l}(cond_1 \wedge when_{mr1_2} \wedge when_{mr2_2}) \\ \rightarrow \\ (set_{mr1_2} \cap set_{mr2_2} \neq \emptyset \rightarrow V \in set_{mr1_2} \cap set_{mr2_2})\end{array}\right)$$

$$\left(\begin{array}{l}(cond_1 \wedge when_{mr1_2} \wedge when_{mr2_3}) \\ \rightarrow \\ (set_{mr1_2} \cap set_{mr2_3} \neq \emptyset \rightarrow V \in set_{mr1_2} \cap set_{mr2_3})\end{array}\right)$$

Des propriétés d'invariance similaires doivent aussi être générées lorsque $cond_2$ est vraie, mais cette fois-ci avec $mr2$ comme règle prioritaire.

Appliquons ce mécanisme sur l'exemple présenté précédemment. La prudence étant toujours prioritaire par rapport à l'urgence, la règle $mr1$ est toujours prioritaire par rapport à la règle $mr2$. De plus, comme $cond_1$ vaut $vrai$, les formules peuvent être simplifiées. De plus, comme il n'y a qu'un cas (un seul $when$) pour $mr1$ et $mr2$, les formules précédentes peuvent être simplifiées comme suit. Sous $cond_1$, $mr1$ est une règle prioritaire :

$$when_{mr1_1} \rightarrow V \in set_{mr1_1}$$

Sous $cond_1$, quand $mr1$ ne s'applique pas, $mr2$ s'applique :

$$(\neg when_{mr1_1}) \rightarrow (when_{mr2_1} \rightarrow V \in set_{mr2_1})$$

Sous $cond_1$, quand mr_1 et mr_2 s'appliquent, si possible, une valeur satisfaisant les deux règles morales doit être choisie :

$$(when_{mr1_1} \wedge when_{mr2_1}) \rightarrow \left(\begin{array}{l} set_{mr1_1} \cap set_{mr2_1} \neq \emptyset \\ \rightarrow \\ V \in set_{mr1_1} \cap set_{mr2_1}\end{array}\right)$$

De plus, nous avons :

$$\left\{\begin{array}{l} V \equiv tl1 \\ when_{mr1_1} \equiv (tl2 \in \{green, orange\}) \\ set_{mr1_1} \equiv (\{orange, red\}) \\ when_{mr2_1} \equiv (true) \\ set_{mr2_1} \equiv (\{green\})\end{array}\right.$$

Nous obtenons ainsi l'invariant suivant, qui doit être vérifié pour garantir que l'agent $a1$ respecte la règle éthique qui spécifie que la prudence est prioritaire à l'urgence :

$$
\begin{array}{l}
tl2 \in \{green, orange\} \rightarrow tl1 \in \{orange, red\} \\
tl2 \notin \{green, orange\}) \rightarrow tl1 \in \{green\} \\
(tl2 \in \{green, orange\}) \rightarrow \left(\begin{array}{l} \{orange, red\} \cap \{green\} \neq \emptyset \\ \rightarrow \\ tl1 \in \{orange, red\} \cap \{green\} \end{array} \right)
\end{array}
$$

Comme $\{orange, red\} \cap \{green\} = \emptyset$, cette propriété d'invariance peut être simplifiée en :

$$
\begin{array}{l}
tl2 \in \{green, orange\} \rightarrow tl1 \in \{orange, red\} \\
tl2 \notin \{green, orange\}) \rightarrow tl1 \in \{green\}
\end{array}
$$

Ainsi, grâce au système de transformation de prédicat proposé, nous obtenons un nouvel invariant qui sera respecté par un agent qui respecte les règles morales données en suivant la règle éthique du système. Par la même occasion, la spécification formelle d'un tel agent pourra être prouvée.

5. Étude de cas

Dans cette section, nous montrons l'application des propositions présentées précédemment sur un cas dans lequel une question éthique plus traditionnelle se pose. On se place dans le cadre de 3 agents A, B et C qui doivent trouver une date de réunion. Un agent peut proposer une date et les deux autres agents doivent indiquer à tous si la date leur convient ou pas. Par exemple, l'agent A peut proposer une date de réunion à B et C. Si B ou C n'acceptent pas la date, ils doivent donner la raison de leur refus aux autres agents. Soit d une date proposée par A. L'agent C doit respecter les deux règles morales suivantes :

– $mr1$, règle rattachée à la valeur morale de *respect* : C ne souhaite pas faire de mal ;

– $mr2$, règle rattachée à la valeur morale d'*honnêteté* : C doit dire à A et B pourquoi la date d ne lui convient pas.

Cependant, si la vraie raison qui amène C à refuser la date d fait du mal à A (par exemple avoir un rendez-vous galant avec la femme de A), il y a un conflit entre les deux règles morales $mr1$ et $mr2$. Pour résoudre ce conflit, l'agent C se fonde sur une règle éthique er stipulant que dans tous les cas, respecter les autres et plus important qu'être honnête.

Afin de formaliser le problème, nous introduisons quelques notations. Appelons $E_{RP} = \{r1, r2, r3, r4\}$ l'ensemble des réponses que C peut donner à A ou B et V_{RC} la variable contenant la vraie raison du refus de C. Pour concrétiser le problème, nous pouvons donner le sens suivant aux différentes réponses possibles :

– $r1$: J'ai un rendez-vous galant avec la femme de A ;

– $r2$: Je suis malade ;

– $r3$: A organise mal les réunions ;

– $r4$: J'ai eu un accident avec la voiture que B m'a prêtée.

Nous représentons également l'ensemble des réponses douloureuses pour chacun des agents par une fonction $F_{RD} \in agents \rightarrow \mathcal{P}(E_{RP})$. Pour l'exemple, nous prendrons $F_{RD} = \{(A, \{r1, r3\}), (B, \{r4\})\}$, ce qui signifie que $r1$ et $r3$ sont des réponses douloureuses pour l'agent A, tandis que $r4$ est une réponse douloureuse pour l'agent B. Nous appelons également V_{RFA} la variable contenant la réponse faite à l'agent A et V_{RFB} la variable contenant la réponse faite à l'agent B.

Nous avons donc identifié deux règles morales :

– $mr1$: C ne souhaite pas faire de mal à A et à B, donc ses réponses doivent être des réponses possibles non douloureuses ;

– $mr2$: C ne veut pas mentir, donc la réponse doit être la vraie raison.

Ces règles peuvent être formalisées ainsi :

$$mr1 : \left(respect, \left\{true, \left\{ \begin{array}{l} (V_{RFA}, E_{RP} - F_{RD}(A)) \\ (v - RFB, E_{RP} - F_{RD}(B)) \end{array} \right\}\right\}\right)$$

$$mr2 : (honnetete, \{true, \{(V_{RFA}, \{V_{RC}\}), (V_{RFB}, \{V_{RC}\})\}\})$$

Enfin, nous avons une règle éthique qui s'applique, er, qui précise que dans tous les cas, le respect est plus important que l'honnêté. Autrement dit, nous avons :

$$er = \{(true, \{(1, respect), (2, honnetete)\})\}$$

D'après ce que nous avons vu dans la section précédente, nous pouvons déduire la règle éthique modifiée suivante :

$$mer = \{(true, \{(1, mr1), (2, mr2)\})\}$$

En appliquant les résultats de la section précédente, nous rajoutons à l'invariant de l'agent C les formules suivantes (nous ne montrons ici que les formules générées pour V_{RFA} ; il faut normalement également rajouter des formules similaires pour V_{RFB}.

Sous $cond_1$, $mr1$ est une règle prioritaire :

$$true \rightarrow (true \rightarrow V_{RFA} \in E_{RP} - F_{RD}(A)) \tag{1}$$

Sous $cond_1$, quand $mr1$ ne s'applique pas, $mr2$ s'applique :

$$true \rightarrow ((\neg true \wedge \neg true) \rightarrow (true \rightarrow V_{RFA} \in \{V_{RC}\})) \tag{2}$$

Sous $cond_1$, quand $mr1$ et $mr2$ s'appliquent, si possible, une valeur satisfiant les deux règles morales doit être choisie :

$$\begin{array}{l} (true \wedge true \wedge true \rightarrow \\ \quad ((E_{RP} - F_R D(A)) \cap \{V_{RC}\} \neq \emptyset \rightarrow \\ \quad\quad V_{RFA} \in (E_{RP} - F_{RD}(A)) \cap \{V_{RC}\})) \end{array} \tag{3}$$

La règle 2 est trivialement vraie. Ces trois règles peuvent donc se simplifier en les 2 suivantes :

$$V_{RFA} \in E_{RP} - F_{RD}(A) \tag{4}$$

$$\begin{array}{c}((E_{RP} - F_{RD}(A)) \cap \{V_{RC}\} \neq \emptyset \rightarrow \\ V_{RFA} \in (E_{RP} - F_{RD}(A)) \cap \{V_{RC}\}))\end{array} \tag{5}$$

Ces propriétés, ajoutées à l'invariant de l'agent, permettent, si elles sont prouvées pour l'agent C de garantir que le comportement de cet agent respecte la règle éthique évoquée. Ainsi, dans le cas présenté, et pour le cas où la vraie raison du refus de C serait $r1$, un agent dont le comportement respecte la règle éthique présentée précédemment devrait simplement vérifier :

$$V_{RFA} \in \{r1, r2, r3, r4\} - \{r1, r3\}$$

Soit :

$$V_{RFA} \in \{r2, r4\}$$

Par contre, si la vraie raison est $r2$, le comportement de l'agent C devrait vérifier les deux propriétés suivantes :

$$\begin{array}{l}V_{RFA} \in \{r1, r2, r3, r4\} - \{r1, r3\} \\ V_{RFA} \in (\{r1, r2, r3, r4\} - \{r1, r3\}) \cap \{r2\}\end{array}$$

Ce qui revient à n'autoriser que l'unique solution : $V_{RFA} = r2$. En poursuivant sur chaque raison, nous arrivons au tableau suivant :

V_{RC}	$r1$	$r2$	$r3$	$r4$
V_{RFA}	$r2, r4$	$r2$	$r2, r4$	$r4$

Nous avons ainsi des propriétés qui seront vérifiées par un agent qui préfère mentir à blesser, mais qui, dès qu'il le peut, dit la vérité. En effet, lorsque la vraie raison ne blesse pas A ($r2$ ou $r4$), un agent dont le comportement a été prouvé devra donner à A cette vraie raison. Par contre, lorque la vraie raison blesse A ($r1$ ou $r3$), un agent correct mentira en donnant un autre justificatif ne blessant pas A (ici, $r2$ ou $r4$).

6. Conclusion et perspectives

Dans cet article, nous avons montré qu'il est possible de prouver formellement qu'un agent peut respecter des règles morales potentiellement contradictoires à partir du moment où une règle éthique permet d'établir, dans les cas où des contradictions apparaissent, quelles sont les priorités qui doivent exister entre les valeurs morales sur lesquelles reposent les différentes règles morales. Pour ce faire, nous avons introduit des transformeurs de prédicats permettant de générer un ensemble de prédicats cohérents à partir de règles morales pourtant contradictoires. Après un exemple simple

ayant servi à introduire les concepts, nous avons montré sur un cas concret que le système proposé pouvait être appliqué sur des cas réels.

Pour limiter la taille des formules, nous nous sommes restreints, dans cet article, à ne montrer ce qui se passe qu'avec 2 règles morales. Bien sûr, le système est automatisable et généralisable à un nombre quelconque de règles morales. Par contre, les formules théoriques deviennent alors très grosses, et peu lisibles. Cependant, cela ne rend pas pour autant le travail de preuve plus compliqué car une fois appliquées, ces formules se simplifient énormément (c'est d'ailleurs ce que l'on constate déjà dans notre étude de cas).

D'autres études de cas seront cependant nécessaires pour valider la portée du système proposé. Notamment, nous nous sommes restreints à des règles morales qui pouvaient être exprimées sous forme de contraintes disjointes d'affectation de valeurs à des variables. Il nous apparaît important de vérifier la portée de cette restriction. Pour les cas où cette restriction rendrait invalide notre méthode, il faudra alors étudier comment l'étendre à des affectations de valeurs liées. Par exemple, on pourrait imaginer que la règle de prudence caractérisant la conduite en cas de verglas pourrait relier la vitesse maximum à l'angle que fait la direction de la voiture avec la ligne droite ainsi : $temps = Verglas \rightarrow vitesse + angle/2 \leq 40$. En effet, plus la voiture prend un virage serré, plus la vitesse doit être réduite pour éviter que la voiture ne dérape.

Dans ce travail, nous n'avons pas envisagé le cas où différentes règles morales associées à une même valeur morale pouvaient être contradictoires car cela nous semble peu pertinent. Cependant, il reste à s'en assurer. Si tel n'est pas le cas, il faudrait alors pouvoir rajouter une notion de priorité entre règles rattachées à une même valeur morale.

Bibliographie

Abel D., MacGlashan J., Littman M. (2016). Reinforcement learning as a framework for ethical decision making. In *AAAI workshops: AI, ethics and society.*

Abramson D., Pike L. (2011). When formal Systems Kill: Computer Ethics and Formal Methods. *APA Newsletter on Philosophy and Computers*, vol. 11, n° 1.

Abrial J.-R. (1996). *The B-Book*. Cambridge Univ. Press.

Alechina N., Logan B., Whitsey M. (2004). A complete and decidable logic for resource-bounded agents. In *3rd aamas.*

Anderson M., Anderson S. (2014). Toward ensuring ethical behavior from autonomous systems: a case-supported principle-based paradigm. *Industrial Robot*, vol. 42, n° 4, p. 324-331.

Arkin R. (2009). *Governing lethal behavior in autonomous robots*. Chapman and Hall.

Atkinson K., Bench-Capon T. (2016). Value based reasoning and the actions of others. In *22nd european conference on artificial intelligence*, p. 680-688.

Baldoni M., Baroglio C., Gungui I., Martelli A., Martelli M., V. Patti V. M. nd *et al*. (2005). Reasoning About Agents' Interaction Protocols Inside DCaseLP. In *Lncs*, vol. 3476, p. 112-131.

Beavers A. (2011). Moral machines and the threat of ethical nihilism. In *Robot ethics: the ethical and social implication of robotics*, p. 333-386. MIT Press.

Bench-Capon T., Atkinson K. (2009). Argumentation in Artificial Intelligence. In G. Simari, I. Rahwan (Eds.), *Abstract argumentation and values*, p. 45-64. Springer.

Berreby F., Bourgne G., Ganascia J.-G. (2015a). Modelling moral reasoning and ethical responsibility with logic programming. In *20th lpar*, p. 532-548.

Berreby F., Bourgne G., Ganascia J.-G. (2015b). Modelling moral reasoning and ethical responsibility with logic programming. In *20th lpar*, p. 532–548.

Bordini R., Fisher M., Visser W., Wooldridge M. (2003). Verifiable multi-agent programs. In *1st promas.*

Brazier F., Eck P. van, Treur J. (1997). Simulating social phenomena. In, vol. 456, p. 103-109. LNEMS.

Brewka G., Benferhat S., Berre D. L. (2004). Qualitative choice logic. *Artif. Intell.*, vol. 157, n° 1-2, p. 203–237.

Bringsjord S., Ghosh R., Pane-Joyce J. (2016). Deontic counteridenticals and the design of ethically correct intelligent agents: First steps. In *1st edia*, p. 38–43.

Broersen J., Dastani M., Huang Z., Hulstijn J., Torre L. V. der. (2001). The BOID architecture: conflicts between beliefs, obligations, intentions and desires. In *5th aa*, p. 9-16.

Calardo E., Governatori G., Rotolo A. (2015). Semantics for modelling reason-based preferences. In *PRIMA 2015: Principles and practice of multi-agent systems - 18th international conference, bertinoro, italy, october 26-30, 2015, proceedings*, p. 101–117.

Chisholm R. M. (1963). Contrary-to-duty imperatives and deontic logic. *Analysis*, vol. 24, n° 2, p. 33–36.

Coelho H., Rocha Costa A. da. (2009). *On the intelligence of moral agency.*

Cointe N., Bonnet G., Boissier O. (2016). Ethical judgment of agents' behaviors in multi-agent systems. In *15th international conference on autonomous agents & multiagent systems*, p. 1106-1114.

Comte-Sponville A. (2012). *La philosophie*. PUF.

Cossentino M., Potts C. (2002). A CASE tool supported methodology for the design of multi-agent systems. In *SERP.*

Damasio A. (2008). *Descartes' error: Emotion, reason and the human brain*. Random House.

Dastani M. (2008). 2APL: a practical agent programming language. *JAAMAS*, vol. 16, p. 214–248.

de Giacomo G., Lesperance Y., Levesque H. J. (2000). Congolog, a concurrent programming language based on the situation calculus. *Artificial Intelligence*, vol. 121, n° 1-2, p. 109-169.

Dennis L., Fisher M., Winfield A. (2015). Towards Verifiably Ethical Robot Behaviour. In *Artifial intelligence and ethics aaai workshop.*

Fisher M. (1994). A survey of concurrent METATEM – the language and its applications. In *1st ictl*, p. 480–505.

Friedman B. (1996). Value-sensitive design. *Interactions*, vol. 3, n° 6, p. 16-23.

Friedman B., Kahn P., Borning A., Huldtgren A. (2013). Value sensitive design and information systems. In *Early engagement and new technologies: Opening up the laboratory*, p. 55-95. Springer Netherlands.

Ganascia J. (2007). Modeling ethical rules of lying with answer set programming. *Ethics and Information Technology*, vol. 9, p. 39-47.

Greene J., Haidt J. (2002). How (and where) does moral judgment work? *Trends in Cognitive Sciences*, vol. 6, n° 12, p. 517-523.

Horty J. F. (1994). Moral dilemmas and nonmonotonic logic. *Journal of philosophical logic*, vol. 23, n° 1, p. 35–65.

Hubner J., Sichman J., Boissier O. (2002). Spécification structurelle, fonctionnelle et déontique d'organisations dans les SMA. In *Jfiadsm.* Hermes.

Kacprzak M., Lomuscio A., Penczek W. (2004). Verification of multiagent systems via unbounded model checking. In *3rd aamas.*

Life Institute F. of. (2015). *Research priorities for robust and beneficial artificial intelligence.*

Martelli M., Mascardi V., Zini F. (s. d.). CaseLP: a Complex Application Specification Environment base on Logic Programming. In *8th iclp.*

McDermott D. (2008). Why ethics is a high hurdle for ai. In *North american conference on computers and philosophy.*

McIntyre A. (2014). Doctrine of double effect. In E. N. Zalta (Ed.), *The stanford encyclopedia of philosophy*, Winter éd..

McLaren B. (2006). Computational models of ethical reasoning: challenges, initial steps, and future directions. *IEEE Intelligent Systems*, vol. 21, n° 4, p. 29-37.

Mermet B., Simon G. (2011). Specifying recursive agents with GDTs. *JAAMAS*, vol. 23, n° 2, p. 273-301.

Mermet B., Simon G. (2013). A new proof system to verify GDT agents. In *Idc*, vol. 511, p. 181-187. Springer.

Moor J. (2006). The nature, importance, and difficulty of machine ethics. *IEEE Intelligent Systems*, vol. 21, n° 4, p. 29-37.

Nissenbaum H. (2005). *Values in technical design*. Rapport technique. Encyclopedia of Science, Technology and Ethics.

Nowak A., Radzik T. (1994). A solidarity value for n-person transferable utility games. *International Journal of Game Theory*, vol. 23, p. 43-48.

Numérique d'Allistene C. de réflexion sur l'Éthique de la Recherche en science et technologies du. (2014). *éthique de la recherche en robotique*. Rapport technique. CERNA.

Owre S., Shankar N., Rushby J. (1992). Pvs: A prototype verification system. In *11th cade.*

Raimondi F., Lomuscio A. (2004). Verification of multiagent systems via orderd binary decision diagrams: an algorithm and its implementation. In *3rd aamas.*

Rao A., Georgeff M. (1995). BDI agents from theory to practice. In *Technical note 56.* AAII.

Ricoeur P. (1990). *Soi-même comme un autre*. Points Essais.

Rokeach M. (1973). *The nature of human values*. New York Free Press.

Russo A., Miller R., Nuiseibeh B., Kramer J. (2001). *An abductive approach for analysing event-based requirements specifications*. Rapport technique. Department of Computing, Imperial College.

Sabas A., Delisle S., Badri M. (2002). A comparative analysis of multiagent system development methodologies: Towards a unified approach. In *Cybernetics and systems*, p. 599-604. Austrian Society for Cybernetics Studies.

Saptawijaya A., Pereira L. (2014). Towards modeling morality computationally with logic programming. In *16th ispadl*, p. 104-119.

Schwartz S. (2012). An overview of schwartz theory of basic values. *Online Readings of Psychology and Culture*, vol. 2, n° 1.

Shapiro S., Lespérance Y., Levesque H. J. (2002). The Cognitive Agents Specification Language and Verification Environment for Multiagent Systems. In *2nd aamas*, p. 19-26.

Shilton K., Koepfler J., Fleischmann K. (2014). How to see values in social computing: methods for studying values dimensions. In *17th acm conference on computer supported cooperative work & social computing*, p. 426-435.

Swchartz S., Bilsky W. (1990). Toward a theory of the universal content and structure of values: Extensions and cross cultural replications. *Journal of Personality and Social Psychology*, vol. 58, p. 878-891.

Timmons M. (2012). *Moral theory: An introduction*. Rowman and Littlefiled.

Tufis M., Ganascia J.-G. (2012). Normative rational agents: A BDI approach. In *1st workshop on rights and duties of autonomous agents*, p. 37-43. CEUR Proceedings Vol. 885.

van Marrewijk M., Werre M. (2003). Multiple levels of corporate sustainability. *Journal of Business Ethics*, vol. 4, n° 2-3, p. 107-119.

Wiener Y. (1988). Forms of value systems: A focus on organisational effectiveness and cultural change and maintenance. *Academy of Management Review*, vol. 13, n° 4, p. 534-545.

Winfield A., Blum C., Liu W. (2014). Towards and Ethical Robot: Internal Models, Consequences and Ethical Action Selection. In *Lncs*, vol. 8717, p. 85-96.