Uma Proposta de Resolução de Conflitos entre Normas e Valores

Jean Carlos Oliveira Santos[1], Karen da Silva Figueiredo[2]

Instituto de Computação, Universidade Federal do Mato Grosso, Cuiabá- MT, Brasil

jcoliveira93@gmail.com[1], karen@ic.ufmt.br[2]

Abstract. *Agents are autonomous entities and they have different interests. Therefore, in an open multi-agent system, conflicts can arise between the norms of the system and personal values of the agents. In this paper, we propose an algorithm to help the agent to resolve those conflict cases based on the value's influence of the norms.*

Resumo. *Agentes são entidades autônomas e têm interesses diferentes. Portanto, em um sistema multiagente aberto, podem surgir conflitos entre as normas do sistema e os valores pessoais dos agentes. Neste trabalho, propomos um algoritmo para ajudar o agente a resolver os casos de conflito entre normas e valores com base na influência dos valores nas normas.*

1. Introdução

Cada vez mais, pesquisas de sistemas multiagentes (SMAs) estão se concentrando nos aspectos culturais das organizações de agentes, tais como normas e valores, e.g. [Dechesne et al. 2013, van der Weide et al. 2010]. A cultura pode ser vista como uma coleção social dos padrões de comportamento aprendidos, compartilhados e transmitidos com o objetivo de adaptar os indivíduos de um grupo [Keesing 1974].

As normas são padrões de comportamento compartilhados por um grupo e usados para regular ações que devem ser executadas pelos agentes (obrigações) e as ações que não devem ser executadas pelos agentes (proibições), a fim de lidar com a heterogeneidade dos agentes.

Os valores são outros padrões de comportamento cultural, ajudando os agentes a avaliar e decidir o que fazer com base no que eles acreditam que é certo para si e para seu grupo. Segundo Schwartz e Bilsky (1987), os valores são princípios ordenados por uma importância relativa que guiam a seleção ou avaliação de comportamentos ou eventos. Quando um agente executa ações, estas ações irão promover ou rebaixar seus valores, de acordo com a importância dos mesmos.

Como as normas são utilizadas para regular as ações de um agente e os valores do agente são promovidos ou rebaixados pelas ações, conflitos podem surgir entre as normas do sistema e os valores do agente. Por exemplo, uma norma pode obrigar um agente a executar uma ação que rebaixa um ou mais de seus valores [Figueiredo e da Silva 2013].

Neste trabalho, apontamos os casos de conflitos que podem surgir entre normas e valores e propomos um algoritmo para ajudar o agente a resolver esses casos. Este

trabalho é uma extensão do algoritmo preliminar apresentado por Figueiredo e da Silva (2013) que é capaz somente de identificar os casos de conflito, sem propor uma forma de resolução. Até onde sabemos, este é o primeiro trabalho em SMAs que implementa uma solução para este tipo de conflito.

2. Casos de Conflito entre Normas e Valores

As *normas* regulam a execução de *ações* ao descrever **obrigações** e **proibições** para os agentes. As *ações*, por sua vez, podem **promover** ou **rebaixar** os *valores* do agente. Desse modo, as ações são um elemento central na ligação entre normas e valores. Assim, é necessário analisar a ação de uma norma para avaliar se existe ou não conflito entre a norma e os valores do agente. Também é necessário considerar os casos em que uma ação é a especialização de outra (relacionamento de generalização) e quando uma ação é parte de outra ação (relacionamento de composição), pois o relacionamento entre as ações afetam a análise dos valores que estão sendo promovidos e rebaixados.

Considerando o exposto, os casos em que uma norma estará em conflito com os valores de um agente são[1]:

1. *A norma define a obrigação de executar uma ação que rebaixa um ou mais valores importantes para o agente.*

2. *A norma define uma proibição de executar uma ação que promove um ou mais valores importantes para o agente.*

3. *A norma define uma obrigação de executar uma superação e todas as suas subações rebaixam pelo menos um valor importante para o agente.*

4. *A norma define uma proibição de executar uma superação e pelo menos uma de suas subações promove um ou mais valores importante para o agente.*

5. *A norma define uma obrigação de executar uma ação composta e a ação composta ou pelo menos uma das ações da composição rebaixam um ou mais valores importantes para o agente.*

6. *A norma define uma proibição de executar uma ação composta e a ação composta ou pelo menos uma das ações da composição promovem um ou mais valores importantes para o agente.*

Em todos os casos acima, a norma define um comportamento que vai contra os interesses pessoais do agente, constituindo uma situação conflitante para o agente.

3. Resolvendo os Casos de Conflito

Nesta seção, propomos um algoritmo para a resolução dos casos de conflitos apontados na seção anterior. Utilizamos a Linguagem Z [Spivey 1988] para a apresentação do algoritmo por ser uma linguagem intuitiva, de fácil entendimento, e já utilizada para descrição de algoritmos arquiteturais para agentes [D'Inverno et al. 1998, dos Santos Neto et al. 2012, Figueiredo e da Silva 2013].

Quando um dos casos de conflito da Seção 2 é identificado, o agente precisará decidir o que fazer: seguir o sistema e cumprir a norma apesar de seus valores, ou violar a norma e se manter com seus valores. O algoritmo que propomos para esta solução se

1 Devido ao pouco espaço os casos de conflitos foram apenas enunciados. Mais informações estão disponíveis no artigo [Figueiredo e da Silva 2013].

baseia no conceito da influência dos valores (*value's influence*, ou *VI*) da norma, que por sua vez é calculado de acordo com a importância que o valor tem para o agente (segundo a Teoria Universal dos Valores [Schwartz e Bilsky 1987]). Para cada caso de conflito, a influência dos valores é calculada recursivamente da seguinte maneira:

VI para o caso de conflito 1. Se a norma é uma *obrigação*, o VI será igual a soma da importância de todos os valores promovidos pela ação menos a soma das importâncias dos valores rebaixados pela ação.

VI para o caso de conflito 2. Se a norma for uma *proibição*, o VI será igual a soma da importância dos valores rebaixados menos a soma da importância dos valores promovidos pela ação.

VI para os casos de conflito 3 e 4. Se a ação regulada pela norma for uma *superação*, o VI da norma será igual ao VI do maior VI encontrado entre cada *subação* da ação regulada pela norma. A subação com o maior VI indica o melhor caso de execução da ação, i.e. o caso em que mais valores são promovidos e/ou menos valores são rebaixados.

VI para os casos de conflito 5 e 6. Se a ação regulada pela norma for uma *ação composta*, o VI da norma será igual a soma do VI de todas as normas derivadas de cada ação que a compõem e do VI da ação regulada pela norma. Como a ação composta exige que todas as ações que a compõem sejam executadas, a soma do VI de todas as ações é necessária.

Para realizar o cálculo do VI da norma descrito acima, o algoritmo proposto utiliza três funções: VP, VD e VI. As funções VP e VD são utilizadas pela função VI, elas recebem uma ação e retornam o somatório da importância dos valores promovidos e rebaixados pela ação, respectivamente.

$VP(Action) \rightarrow \mathbb{N}$
$sum : \mathbb{N}$

---

01. $\forall\, a : Action \bullet$
02. $\quad sum = 0$
03. $\quad \{\forall\, v : a.promotes \bullet sum = sum + v.importance\}$
04. $\quad VP(a) = sum$

$VD(Action) \rightarrow \mathbb{N}$
$sum : \mathbb{N}$

---

01. $\forall\, a : Action \bullet$
02. $\quad sum = 0$
03. $\quad \{\forall\, v : a.demotes \bullet sum = sum + v.importance\}$
04. $\quad VD(a) = sum$

$VI(Norm \times Action) \rightarrow \mathbb{Z}$
$influence : \mathbb{Z}$
$greatest : Action$

---

01. $\forall\, n : Norm, a : Action \bullet$
02. $\quad$ **if** $(a.subactions \neq 0)$ **then**
03. $\quad\quad greatest = \emptyset$
04. $\quad\quad \{\forall\, sub : a.subactions \bullet$
05. $\quad\quad\quad$ **if** $(VI(n, sub) > VI(n, greatest))$ **then**
06. $\quad\quad\quad\quad greatest = sub$
07. $\quad\quad \}$
08. $\quad\quad influence = VI(n, greatest)$
09. $\quad$ **else if** $(a.compositeactions \neq 0)$ **then**
10. $\quad\quad$ **if** $(n.deonticConcept = OBLIGATION)$ **then**
11. $\quad\quad\quad influence = VP(a) - VD(a)$
12. $\quad\quad$ **else if** $(n.deonticConcept = PROHIBITION)$ **then**
13. $\quad\quad\quad influence = VD(a) - VP(a)$
14. $\quad\quad \{\forall\, sub : a.compositeactions \bullet$
15. $\quad\quad\quad influence = influence + VI(n, sub)$
16. $\quad\quad \}$
17. $\quad$ **else**
18. $\quad\quad$ **if** $(n.deonticConcept = OBLIGATION)$ **then**
19. $\quad\quad\quad influence = VP(a) - VD(a)$
20. $\quad\quad$ **else if** $(n.deonticConcept = PROHIBITION)$ **then**
21. $\quad\quad\quad influence = VD(a) - VP(a)$
22. $\quad VI(n, a) = influence$

A função VI mapeia uma norma e sua ação para um número inteiro que indica o VI conforme os casos previamente apresentados. Primeiramente, a função verifica se a o tipo da ação pela norma. Se a ação é uma *superação*, a função guarda o maior VI encontrado recursivamente cada subação (casos de conflito 3 e 4 – linhas 2 a 8). Se a ação é uma *ação composta*, a função guarda recursivamente o somatório do VIs de cada ação da composição mais o VI da própria ação composta (casos de conflito 5 e 6 – linhas 9 até 16). Finalmente, se a ação for simples, a função guarda o VI calculado a partir do tipo da norma (obrigação ou proibição) utilizando as funções VP e VD (casos de conflito 1 e 2 – linhas 17 a 21). Ao final das iterações, a variável *influence* é retornada pela função com o VI calculado armazenado (linha 22).

A função *ComplyWithNorm* é a função principal do algoritmo de resolução proposto. Ela é utilizada pelo agente para checar se existem conflitos entra uma norma e seus valores e então resolvê-los. A função *ComplyWithNorm* mapeia uma norma e sua ação para um booleano, que indica se o agente deve cumprir ou não a norma.

$$ComplyWithNorm(Norm \times Action) \rightarrow Boolean$$

```
01. ∀ n : Norm, a : Action •
02.     if (checkNormValueConflicts(n, a)) then
03.         ComplyWithNorm(n, a) = VI(n, a) ≥ 0
04.     else
05.         ComplyWithNorm(n, a) = TRUE
```

Primeiramente, a função verifica a existência de algum dos casos de conflito da Seção 2 através da função *checkNormValueConflict*[2] (linha 2). Caso haja conflito, a função chama a função VI: se o resultado for maior ou igual a zero, a função *ComplyWithNorm* retorna ***verdadeiro***; caso contrário, retorna ***falso*** (linhas 2 a 3). Ou seja, se o valor do VI da norma for maior ou igual a zero, considerando todos os tipos de ações, o ganho do valores promovidos compensa a perda dos valores rebaixados no caso de obrigações, ou o contrário no caso de proibições. Assim, o agente preferirá cumprir a norma. Mas, se o VI da norma for menor que zero, é melhor para o agente seguir seus valores, mesmo que isso signifique violar a norma.

Caso não haja conflito, *ComplyWithNorm* também retorna verdadeiro (linhas 4 e 5), indicando que o agente não tem razão para violar a norma, de acordo com seus valores.

4. Exemplificando a Proposta

Nesta seção, apresentamos um exemplo para demonstração do algoritmo proposto. Para este exemplo, considere um grupo de agentes robôs de resgate (compostos por módulos como lanterna, sensores de calor, comunicação, etc.) que realizam buscas de pessoas feridas em escombros e destroços, por exemplo, em desabamentos. Considere que estes robôs possuem a seguinte norma regulando o seu comportamento:

Norma 1: Todos os robôs são proibidos de desligar um módulo durante uma missão.

2 A função *checkNormValueConflicts* que indica se há ou não um conflito entre as normas e os valores é apresentada em nossos trabalho anterior [Figueiredo e da Silva 2013] e omitida neste trabalho por falta de espaço.

Considere ainda que a ação da norma acima (desligar módulo - *turnOffModule*) possui a seguinte estrutura:

turnOffModule ({subactions: turnOffLights, turnOffSensors, turnOffComunication}, {promotes: power})
turnOffLights ({demotes: security})
turnOffSensors ({demotes: efficiency})
turnOffComunication ({demotes: security, cooperation})

A ação *turnOffModule* pode ser benéfica em algumas situações, como no caso de um módulo desnecessário para a situação estar gastando energia que poderia ser usada para estender o tempo de funcionamento do robô em uma missão, entretanto ela também pode comprometer a segurança do próprio robô, por exemplo. Por se tratar da proibição de uma *superação* que promove um valor do agente, esta norma está em conflito com seus valores (caso de conflito 4).

Dado o presente cenário, imagine que existem dois robôs de resgate (Robô 01 e Robô 02) com os conjuntos de valores pessoais distintos descritos abaixo que precisam avaliar se irão cumprir ou não a Norma 1 através do nosso algoritmo.

Conjunto de Valores do Robô 01:
power (importance: 2)
security (importance: 3)
efficiency (importance: 4)
cooperation (importance: 2)

Conjunto de Valores do Robô 02:
power (importance: 3)
security (importance: 2)
efficiency (importance: 2)
cooperation (importance: 0)

Quando a função *ComplyWithNorm* é chamada, ela identifica que existe um caso de conflito entre a Norma 1 e os valores do Robôs 01 e 02 (função *checkNormValueConflicts*) e então calcula o VI da Norma 1 através da função *VI*. Por se tratar de uma *superação*, a função *VI* irá calcular recursivamente o VI para cada uma das *subações* e irá retornar o maior VI para o caso.

VI (Norm1, turnOffModule) = greatest(VI(turnOffLights), VI(turnOffSensors), VI(Norm1,turnOffComunication))
VI (Norm1, turnOffLights) = importance(security) – importance(power)
VI (Norm1, turnOffSensors) = importance(efficiency) – importance(power)
VI (Norm1, turnOffComunication) = (importance(security) + importance (cooperation)) – importance(power)

Se o VI retornado pela função for maior ou igual a zero os robôs irão seguir a norma. Caso contrário, os robôs irão violar a norma. Como cada robô possui um conjunto de valores diferentes, vamos analisar a saída do algoritmo para cada caso.

Cálculo do VI para o Robô 01:
VI (Norm1, turnOffLights) = 3 – 2 = 1
VI (Norm1, turnOffSensors) = 4 – 2 = 2
VI (Norm1, turnOffComunication) = (3 + 2) – 2 = 3
VI (Norm1, turnOffModule) = 3

Cálculo do VI para o Robô 01:
VI (Norm1, turnOffLights) = 2 – 3 = -1
VI (Norm1, turnOffSensors) = 2 – 3 = -1
VI (Norm1, turnOffComunication) = (2 + 0) – 3 = -1
VI (Norm1, turnOffModule) = -1

Deve o Robô 01 cumprir com a norma? Neste caso, o VI da proibição de *turnOffModule* será maior que zero, portanto a função *ComplyWithNorm* entende que os valores rebaixados ao executar a ação que está proibida pela norma não compensam os

valores que seriam promovidos. Assim, o Robô 01 irá cumprir com a norma e manter seus módulos sempre ligados.

Deve o Robô 02 cumprir com a norma? Neste caso, o VI da proibição será um resultado negativo (-1), portanto a função *ComplyWithNorm* entende que os valores promovidos ao executar a ação da norma compensam a violação da proibição. Assim, o Robô 02 escolherá desligar um módulo, diminuindo suas funções e sua segurança, mas funcionando por mais tempo e atendendo a missão por um período maior.

5. Conclusão e Trabalhos Futuros

A resolução de conflitos entre normas e valores em tempo de execução é importante pelo fato que novos conflitos podem surgir a qualquer momento e o agente precisa saber lidar com essas situações. Primeiramente, as normas são constantemente modificadas devido a dinâmica do SMAs, e segundo, os valores dos agentes são alterados devido a convivência e influências culturais [Schwartz e Bilsky 1987].

Este trabalho propôs um algoritmo para a ajudar o agente na resolução dos conflitos entre normas e valores baseado na importância dos valores pessoais e sua influência nas normas. Nesta resolução consideramos não só ações simples, mas também as ações mais complexas com especializações e composições. Nosso próximo passo é estender o algoritmo para que este inclua no processo decisório a análise das sanções (recompensas e punições) aplicadas ao cumprir ou violar uma norma.

Referências

Dechesne, F., Di Tosto, G., Dignum, V. and Dignum, F. (2013) "No smoking here: values, norms and culture in multi-agent systems", In: Artificial Intelligence and Law, v. 21, n. 1, 79 – 107.

D'Inverno, M., Kinny, D., Luck, M. and Wooldridge, M. (1998) "A Formal Specification of dMARS", In: Proceedings of the 4th International Workshop on Intelligent Agents IV, London, UK, Springer-Verlag, 155 – 176.

dos Santos Neto, B., da Silva, V. and de Lucena, C. (2012) "An architectural model for autonomous normative agentes", In: Advances in Artificial Intelligence-SBIA 2012, Springer Berlin Heidelberg, 152 – 161.

Figueiredo, K. S., da Silva, V. T. (2013) "Identifying Conflicts between Norms and Values", In: International Workshop on Coordination, Organizations, Institutions and Norms in Agent Systems (COIN@AAMAS 2013), Saint Paul, MN, USA.

Keesing, R. M. (1974) "Theories of culture", In: Annual Review of Anthropology 3, p.73 – 97.

Schwartz, S. and Bilsky, W. (1987) "Toward a universal psychological structure of human values", In: Journal of Personality and Social Psychology 53(3).

Spivey, J. M. (1988) "Understanding Z: a specification language and its formal semantics", In: Cambridge University Press, New York, NY, USA.

van der Weide, T., Dignum, F., Meyer, J., Prakken, H. and Vreeswijk, G. (2010) "Practical reasoning using values", In: Argumentation in Multi-Agent Systems, Springer Berlin Heidelberg, 79 – 93.