5
O Modelo em Lógica Modal

5.1
Introdução

Usa-se lógica para formalizar um sistema multi-agentes aberto com confiança porque com ela é possível se fixar em uma linguagem artificial bem-definida e estruturada, expressando propriedades de uma maneira rigorosa e matemática. Outra vantagem é que a ambigüidade pode ser removida com o uso de lógica. Tem-se ainda que ao se expressar as propriedades de agentes e de sistemas multi-agentes como axiomas lógicos e teoremas em uma linguagem com uma semântica clara, os pontos principais da teoria são explicitados. Finalmente, a teoria torna-se transparente: as propriedades, os relacionamentos e as inferências são abertas a examinação. Usando-se uma modelagem em linguagem natural, por exemplo, que em geral é mal-estruturada e mal-definida, não se consegue tal clareza (Dal97, End02, Men01, Woo00).

Seria possível usar lógica clássica de primeira ordem para formalizar o sistema, já que a mesma é expressiva o suficiente para ser usada para codificar praticamente qualquer forma de conhecimento. Com certa engenhosidade, é possível se codificar em lógica clássica de primeira ordem quase tudo o que for codificável em lógica modal, por exemplo. Portanto, pode-se afirmar que a lógica clássica de primeira ordem é de fato um formalismo genérico. Entretanto, as sentenças ficam muito mais extensas e menos intuitivas quando abordam modalidades ou idéias temporais, algo presente ao se modelar sistemas multi-agentes.

Mais ainda, existem diversos casos de traduções que não se cumprem em lógica clássica de primeira ordem para conceitos definidos de forma computável (leia-se decidível) em lógica modal. Outros exemplos são as classes de *frames* não-elementares, que não podem ser definidos em lógica clássica de primeira ordem, mas que são modalmente especificáveis. Esses são definidos por axiomas que não possuem a classe de *frames* com modelos canônicos que os validam (a saber *frames* com o reverso da relação de acessibilidade bem-fundados), não podendo ser especficados por nenhum conjunto de sentenças em lógica clássica

de primeira ordem.

Mais especificamente, as lógicas modais podem ser entendidas como linguagens especializadas para representar propriedades de estruturas relacionais. Elas são úteis por sua capacidade de formalizar aspectos relacionais (do *frame*) de forma efetiva, em comparação com a versão dos mesmos em primeira ordem. No caso de propriedades temporais, por exemplo, a estrutura relacional em questão é a estrutura temporal do mundo. Se conectivos modais não forem utilizados para representar propriedades da estrutura relacional, parece ser necessário introduzir a estrutura relacional na linguagem de representação explicitamente, dificultando a leitura das fórmulas. Ou seja, com lógica modal, o formalismo fica mais próximo da linguagem natural. Assim, o entendimento e a busca por propriedades no mesmo fica bastante facilitada.

Outra vantagem da lógica modal é que, enquanto a maioria das lógicas modais é decidível, a lógica clássica de primeira ordem é semi-decidível, sendo a maior parte dos sistemas de lógica modal PSPACE-completo. Apesar da alta complexidade, na prática, usando-se técnicas de model-checking (Cla99), consegue-se provar a satisfabilidade de uma fórmula em um dado modelo eficientemente no tamanho do mesmo.

A seguir, modela-se o funcionamento de um sistema multi-agentes. Posteriormente, modela-se o funcionamento de um agente nesse sistema.

5.2 Modelagem para o Sistema

Primeiramente é necessário formalizar o sistema multi-agentes. Como o sistema é aberto, em cada instante de tempo, agentes podem entrar, sair ou permanecer no sistema. Há basicamente duas maneiras diferentes para se formalizar o sistema usando a semântica de mundos possíveis:

- Só há mudança de estado do sistema quando um agente sai ou entra no sistema. Ou seja, se o conjunto de agentes permanecer o mesmo, não há transição de um mundo para outro.
- A cada instante de tempo há uma transição de um mundo para outro, que pode ser o mesmo se o conjunto de agentes permanecer o mesmo.

O segundo modelo foi o escolhido por permitir a representação de tempo explicitamente. Adicionalmente, mesmo que o conjunto de agentes não mude, eles estão executando ações e, possivelmente, interagindo entre si, o que não é instantâneo. Mais ainda, eles podem alterar os seus próprios estados mentais, o que também não é instantâneo. Portanto, enquanto um agente está sem tomar nenhuma ação, apenas tomando decisões e analisando o ambiente,

atualizando seus estados mentais, o tempo está passando. Mais ainda, o tempo é considerado como sendo global. Ou seja, supõe-se que o tempo passa da mesma forma e é o mesmo para todos os agentes do sistema. Portanto, o modelo temporal é síncrono. Para isso, pode-se supor que os agentes podem consultar seus parceiros, sua organização ou o sistema quanto ao tempo. Isso pode ser necessário quando um agente entra no sistema, pois o sistema anterior do agente poderia vir a se comportar de uma maneira diferente quanto ao tempo. Afinal, cada sistema tem as suas convenções, podendo variar quanto à definição de tempo.

O modelo temporal possui as seguintes características (Cla99, Woo00):

- discreto;
- síncrono;
- limitado no passado (possui um estado inicial);
- não-limitado no futuro (não possui um estado final);
- linear no passado (só há um passado), e
- ramificado no futuro (o futuro é não-determinístico; os eventos futuros ainda não são determinados).

O modelo temporal é formalizado usando-se uma estrutura temporal ramificada, que é total, e um grafo direcionado e linear no passado sobre um conjunto de instantes T. Note-se que uma relação binária R_T é total se cada nó em R_T tiver ao menos um sucessor. Ou seja, cada elemento na relação sempre vai ter um par. Deve ficar claro que dizer que a relação é total não significa que ela é uma ordem total, ou seja, dizer que a relação é linear. Logo, a relação binária de tempo R_T sobre T é total se satisfizer a seguinte condição:

$$\forall t \in T \Rightarrow \exists t'[t' \in T \text{ e } (t, t') \in R_T]$$

Como não há um estado final, essa propriedade é satisfeita. A relação $R_T \subseteq T \times T$ representa a estrutura temporal ramificada que codifica todas as maneiras nas quais o sistema pode evoluir.

O sistema é formalizado através de uma estrutura de mundos possíveis $< W, R_S >$, onde $W \neq \emptyset$, cada mundo $w \in W$ representa o conjunto de agentes presentes em um dado instante e R_S representa a relação de transição de um mundo para outro, representando também a passagem de um instante de tempo. Note-se que R_S é diferente do da relação de passagem de tempo definido no parágrafo anterior. R_T é apenas uma relação temporal. Já R_S é a relação que define a transição do sistema de um estado para outro. Note-se que sempre há uma transição R_T, ocorre uma transição R_S e vice-versa, já que

toda a vez que o tempo evolui, o sistema evolui e vice-versa. Se o conjunto de agentes continuar sendo o mesmo no instante seguinte, tem-se que $R_S(w, w)$. Já se o conjunto de agentes de $w \in W$ se modificar para ser o conjunto de agentes de $w' \in W$, tem-se que $R_S(w, w')$ (Hug96).

Já a presença dos agentes é representada por variáveis proposicionais a_0, a_1, ..., correspondendo aos agentes 0, 1, ... Ou seja, a variável proposicional a_i representa o agente i. Quando um agente estiver presente no sistema, o valor da variável proposicional correspondente será verdadeiro; caso contrário, falso. As variáveis de indivíduo a_i têm designação rígida (possuem o mesmo significado em todos os mundos). Ou seja, o significado de a_i é o mesmo em todos os mundos, que é a presença do agente i em um instante de tempo. Então, a_i não vai se referir a um agente j em algum mundo ou a qualquer outra coisa que não seja em relação a presença ou não do agente i no sistema.

Então, tem-se que um modelo para a estrutura do sistema multi-agentes é da forma $< W, R_S, V >$, onde V é uma atribuição de valores a variáveis proposicionais em um mundo $w \in W$ (Hug96). Nesse caso, a atribuição de valores para as variáveis vai depender do fato de o agente estar ou não presente no sistema em um dado mundo. Logo, tem-se que: $V(\alpha_i, w) = 1$ se e somente se o agente i estiver presente no mundo $w \in W$. Caso contrário, $V(\alpha_i, w) = 0$.

Quanto às sentenças lógicas, será utilizada a lógica modal proposicional, com $\Box$ representando a modalidade de "necessidade" e $\Diamond$, a de "possibilidade". A noção de tempo está presente implicitamente, passando a todo instante, mude o conjunto de agentes ou não, haja ações de agentes ou não.

Para um sistema multi-agentes genérico, o modelo é reflexivo e um grafo contendo ciclos, onde cada mundo (nó) está relacionado com todos os outros. Isso acontece porque a cada instante qualquer agente pode entrar ou sair do sistema, ou o conjunto de agentes presentes permanecer o mesmo. O sistema genérico também é, obviamente, não-determinístico, pois não se sabe quais agentes podem entrar ou sair em um dado instante. Mas na prática isso pode não necessariamente acontecer. Por exemplo, se novos agentes puderem entrar no sistema mas não puderem sair, os mundos futuros não podem ter transição para mundos com o mesmo conjunto de agentes de mundos passados. Portanto, dependendo do comportamento do sistema, determinados axiomas da lógica modal podem valer ou não e, com isso, é possível a existência de sistemas com diferentes propriedades. Alguns exemplos são dados a seguir (Che80, Hug96), onde nas fórmulas abaixo a_i são meta-variáveis que podem ser instanciadas. É possível também se trocar a_i por fórmulas que representam quais agentes estão presentes ou ausentes, tal como $a_i \wedge a_j$ e $\neg a_j$. Nesses casos, a leitura da fórmula é modificada, mas não a propriedade que vale no sistema. É importante dizer

que as fórmulas abaixo valem em todos os mundos de todos os *frames* que possuem as propriedades a elas relacionadas.

$$K : \Box(a_i \rightarrow a_j) \rightarrow (\Box a_i \rightarrow \Box a_j)$$

Se em todo estado futuro, se o agente j estiver presente quando o agente i estiver presente, então, em todo estado futuro, se o agente i estiver presente, então, em todo estado futuro o agente j estará presente. Esse axioma vale para qualquer sistema multi-agente.

$$T : \Box a_i \rightarrow a_i$$

Se em todo mundo futuro o agente i estiver presente, então o agente i está presente no mundo atual. Esse axioma é válido se o modelo for reflexivo, isto é, se a cada instante for sempre possível que o conjunto de agentes presentes não seja modificado, ou seja, que nenhum agente saia ou entre no sistema. Então, o conjunto de agentes pode ser o mesmo (ou repetido) em um instante futuro. O axioma não será válido se necessariamente um agente entrar ou sair do sistema em um dado instante de tempo. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w \in W((w, w) \in R_S).$$

$$D : \Box a_i \rightarrow \Diamond a_i$$

Esse axioma diz que o sistema multi-agentes sempre continuará funcionando, nunca tendo um estado final. Na prática, a maioria dos sistemas multi-agentes possui essa propriedade, que representa a serialidade. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w \in W \, \exists w' \in W((w, w') \in R_S).$$

$$4 : \Box a_i \rightarrow \Box\Box a_i$$

Esse axioma diz que se um agente estiver presente em todo instante futuro, ele continuará presente no instante a seguir. Em um sistema onde esse axioma valha, se necessariamente houver agentes após o estado inicial do sistema, eles não poderão mais sair. Ou seja, se novos agentes tiverem que entrar, eles não poderão deixar o sistema. Esse axioma representa a propriedade de transitividade. Escrevendo em FOL a propriedade para os *frames* onde

a fórmula acima sempre vale, tem-se que:

$$\forall w, w', w'' \in W((w, w') \in R_S \wedge (w', w'') \in R_S \rightarrow (w, w'') \in R_S).$$

Um exemplo de sistema onde isso ocorre é em comércio eletrônico e em sites de leilões virtuais. Uma vez cadastrado no sistema, um agente não pode se descadastrar, ou seja, sair do sistema. Como exemplos, tem-se a Amazon.com e o eBay.

$$5 : \Diamond a_i \rightarrow \Box\Diamond a_i$$

Esse axioma diz que, se o agente i estiver presente em um mundo futuro, então ele estará presente em todo mundo posterior a esse mundo futuro. Outra forma de descrevê-lo é dizer que, dada a passagem de um instante de tempo, se o conjunto de agentes puder ser modificado de duas maneiras distintas, sendo que em uma delas pode permanecer igual, então, a partir dessas duas configurações futuras pode-se chegar na outra e vice-versa no próximo instante de tempo. Por exemplo, se em um dado instante puderem entrar um agente i ou um agente j, mas não os dois ao mesmo tempo, no instante seguinte deve ser possível o outro agente entrar. Isso significa que o modelo que representa o sistema é euclidiano. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w, w', w'' \in W((w, w') \in R_S \wedge (w, w'') \in R_S \rightarrow (w', w'') \in R_S).$$

$$B : a_i \rightarrow \Box\Diamond a_i$$

Esse axioma vale em sistemas nos quais, se for possível mudar o conjunto de agentes presentes no sistema em um dado instante, é possível voltar a configuração anterior, ou seja, ao mesmo conjunto de agentes que estava presente antes. Ou seja, se passou um instante de tempo e determinados agentes entraram, deve ser possível que no instante seguinte aos agentes que entraram saírem. O mesmo vale para agentes que saíram em um dado instante. Esse axioma representa a propriedade de simetria. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w, w' \in W((w, w') \in R_S \rightarrow (w', w) \in R_S).$$

$$Triv : a_i \leftrightarrow \Box a_i$$

Nesse caso, o conjunto de agentes nunca muda. Ou seja, não é um sistema multi-agentes aberto. Porém, ele está funcionando. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w, w' \in W((w, w) \in R_S \wedge w = w').$$

$$Ver : \Box a_i$$

Nesse caso, o sistema está parado. Também não é uma propriedade de interesse. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w, w' \in W((w, w) \notin R_S \wedge w = w').$$

$$T_C : a_i \rightarrow \Box a_i$$

Nesse caso, o conjunto de agentes nunca muda. O sistema pode estar ou não parado. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w, w' \in W(w = w').$$

$$D_C : \Diamond a_i \rightarrow \Box a_i$$

Nesse caso, se em um mundo futuro um agente estiver presente, então ele vai estar presente em todos os mundos futuros. O sistema é determinístico: ou o sistema muda o conjunto de agentes de uma única forma ou permanece para sempre com o mesmo conjunto de agentes, ou então está parado. Pode-se também dizer que a relação de acessibilidade é parcialmente funcional. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w, w', w'' \in W((w, w') \in R_S \wedge (w, w'') \in R_S \rightarrow w' = w'').$$

$$G : \Box(\Box a_i \rightarrow a_i) \rightarrow \Box a_i$$

$G + K = GL$, que é a lógica da provabilidade. Pode-se tomar com sendo a que tem por classe de frames os transitivos, finitos e irreflexivos (finito-transitivo quer dizer que a cadeia de transitividades é finita). A modalidade em GL é mais

associada a conhecimento como demonstração, que pode ser descrita como: se for possível provar que o que quer que seja que for provável for verdadeiro ($\Box(\Box a_i \rightarrow a_i)$), então o que quer que seja é provável ($\Box a_i$). G é um exemplo de axioma que não possui a classe de frames com modelo canônico que o valida.

O axioma G vai ser válido em um sistema onde nunca vai haver um mundo onde o conjunto de agentes vai continuar sendo o mesmo após o instante seguinte, por ser irreflexivo. Ou seja, a cada instante de tempo sempre vai haver mudança no conjunto de agentes. Mais ainda, por ser transitivo, os agentes que entrarem não vão poder mais sair e vice-versa. Adicionalmente, a execução do sistema é finita. A irreflexividade até poderia ocorrer em um sistema muito grande, com um número enorme de agentes querendo entrar a todo momento, por exemplo. Mas, na prática, não é uma propriedade muito razoável. Tem-se ainda a finitude, que não é uma propriedade desejável. A primeira propriedade já foi escrita em FOL quando da explicação dos sistemas transitivos onde o axioma 4 ($\Box a_i \rightarrow \Box\Box a_i$) sempre vale. A finitude e a irreflexividade são escritas em FOL a seguir, respectivamente:

$$\forall w \in W \exists w' \in W(R_S^*(w, w') \wedge \forall w'' \in W \neg R_S(w', w'')) \quad \text{e} \quad \forall w \in W \neg R_S(w, w),$$

onde R_S^* é a relação R_S iterada descrita a seguir:

$$\forall w, w' \in W(((R_S^*(w, w') \leftrightarrow R_S(w, w')) \vee$$
$$\exists w'' \in W(R_S^*(w, w') \leftrightarrow R_S(w, w'') \wedge R_S^*(w', w''))).$$

$$\Box(a_i \wedge \Box a_i \rightarrow a_j) \vee \Box(a_j \wedge \Box a_j \rightarrow a_i)$$

Nesse caso, o conjunto de mundos possíveis é fracamente conectado, ou seja, a partir de um conjunto de agentes presentes no sistema, é possível se chegar a qualquer outro conjunto possível de agentes no sistema através da entrada, saída ou permanência dos agentes. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w \forall w' \forall w''((w, w') \in R_S \wedge (w, w'') \in R_S$$
$$\rightarrow (w', w'') \in R_S \vee w' = w'' \vee (w'', w') \in R_S)$$

É importante enfatizar que o axioma K vale em todos os tipos de sistema citados acima. Mais ainda, em sistemas onde valem mais de um axioma além de K, é possível se conseguir mais propriedades. Por exemplo, o sistema multi-agentes genérico, onde a qualquer instante agentes podem entrar ou sair satisfaz T e 5. Um sistema multi-agentes que satisfaz tais axiomas têm como lógica modal correspondente o sistema $S5$. Exemplos de sistemas onde agentes podem entrar e sair a qualquer momento são aqueles que envolvem computação móvel.

Já um sistema onde valha T ($\Box a_i \rightarrow a_i$) e o axioma abaixo

$$(a_i \wedge a_j \wedge \Diamond(a_i \wedge \neg a_j)) \rightarrow \Box a_i$$

diz que se dois agentes estiverem presentes e em algum instante futuro um deles sair, em todo instante futuro apenas o que não saiu vai estar presente. Isso significa que de um estado para outro, só há uma maneira de o conjunto de agentes se modificar. Isto é, o conjunto de agentes pode se manter igual ou se modificar de apenas uma maneira no máximo. Na semântica de mundos possíveis, isso significa que só há transição de um mundo para ele mesmo (reflexibilidade) e de, no máximo, para apenas um outro mundo. Escrevendo em FOL a propriedade para os *frames* onde a fórmula acima sempre vale, tem-se que:

$$\forall w, w', w'' \in W((w, w) \in R_S \wedge ((w, w') \in R_S \wedge (w, w'') \in R_S \rightarrow w' = w'')).$$

Note-se que a linguagem e a semântica para a modelagem do sistema são bem simples. Agora, é necessário modelar o comportamento de um agente. Para modelar os agentes do sistema, a lógica e a semântica são bem mais complexas, como será explicado nas sessões a seguir.

5.3 A linguagem Lógica para Lidar com o Agente

A linguagem lógica utilizada neste texto, uma extensão da linguagem multi-modal de primeira ordem LORA (Woo00), é poli-sortida, permitindo quantificação sobre vários tipos de objetos: agentes, (seqüências de) ações, conjuntos de agentes (grupos) e outros indivíduos do mundo (ambiente). Todos esses tipos (*sorts*) têm um conjunto correspondente de termos (variáveis e constantes de indivíduo) no alfabeto da linguagem. Adicionalmente, a linguagem contém modalidades para as crenças, desejos e intenções, um predicado de primeira-ordem para capturar o modelo de confiança, conectivos temporais (que também são modalidades), os conectivos e quantificadores da lógica clássica de primeira-ordem e, finalmente, operadores sobre a pertinência em grupos e em agentes de uma ação, no caso de uma ação tomada em conjunto por um grupo de agentes. Logo, o alfabeto da linguagem contém os seguintes símbolos:

1. Um conjunto enumerável $Pred$ de predicados;

2. Um conjunto enumerável $Const$ de constantes, que é a união dos seguintes conjuntos mutuamente disjuntos:

- $Const_{Ag}$ – constantes para agentes;
- $Const_{Ac}$ – constantes para seqüências de ações;
- $Const_{Gr}$ – constantes para conjuntos de agentes (grupos), e
- $Const_{U}$ – constantes para os demais indivíduos.

3. Um conjunto enumerável Var de variáveis, que é a união dos seguintes conjuntos mutuamente disjuntos:

   - Var_{Ag} – variáveis que denotam agentes;
   - Var_{Ac} – variáveis que denotam seqüências de ações;
   - Var_{Gr} – variáveis que denotam conjuntos de agentes (grupos), e
   - Var_{U} – variáveis que denotam os demais indivíduos.

4. Operadores modais:

   - Bel – a modalidade de crença;
   - Des – a modalidade de desejo;
   - Int – a modalidade de intenção;
   - $said$ – a modalidade disse;
   - $\Box$ – a modalidade de necessidade;
   - $\Diamond$ – a modalidade de possibilidade;
   - A – o quantificador universal de caminho;
   - E – o quantificador existencial de caminho;
   - $\mathcal{U}$ – o conectivo temporal binário "até" (*until*);
   - $\mathcal{W}$ – o conectivo temporal binário "a não ser que" (*unless*), e
   - $\bigcirc$ – o conectivo temporal unário "próximo" (*next*).

5. Predicados de primeira-ordem para falar de confiança, que são definidos na seção 5.5:

   - $TrustB$ – confiança binária de um agente em outro (binário);
   - $TrustF$ – confiança de um agente em outro com níveis fixos de confiança (quaternário);
   - $TrustV$ – confiança de um agente em outro com níveis variáveis de confiança (ternário);
   - $TrustPB$ – confiança binária parametrizada de um agente em outro (ternário);
   - $TrustPF$ – confiança parametrizada de um agente em outro com níveis fixos de confiança (cinco parâmetros), e
   - $TrustPV$ – confiança parametrizada de um agente em outro com níveis variáveis de confiança (quaternário).

6. Ações de agentes:

- *says* – um agente diz algo;
- *do* – um agente delega uma tarefa a outro;
- *in* – um agente entra no sistema, e
- *out* – um agente sai do sistema.

7. Função que retorna o conjunto das capacidades necessárias que um agente precisa ter para fazer ou falar sobre algo: $capacidades()$;

8. Operadores adicionais:

   - $\in$ – pertinência de um agente a um grupo de agentes ou pertinência de uma capacidade a uma ação ou ao conteúdo de uma mensagem, e
   - $Agts$ – os agentes que executam uma ação.

9. $\top$ (verdadeiro), $\bot$ (falso);

10. Os construtores de ações:

    - ";" – composição seqüencial;
    - "—" – escolha não-determinística;
    - "*" – iteração, e
    - "?" – ações de teste;

11. Os conectivos clássicos:

    - $\vee$ – disjunção;
    - $\wedge$ – conjunção;
    - $\neg$ – negação;
    - $\rightarrow$ – implicação, e
    - $\leftrightarrow$ – equivalência;

12. Os quantificadores universal ($\forall$) e existencial ($\exists$), e

13. Símbolos de pontuação: "(" e ")".

Associado a cada predicado há um número natural que é a sua aridade, dada pela função $aridade()$:

$$aridade: \ Pred \rightarrow Nat.$$

Predicados de aridade 0 são os símbolos proposicionais.

Um tipo pode ser Ag, Ac, Gr ou U. Se σ for um tipo, então o conjunto $Term_\sigma$, de termos do tipo σ é definido como a seguir:

$$Term_\sigma = Var_\sigma \cup Const_\sigma$$

O conjunto $Term$ de todos os termos é definido por:

$$Term = Term_{Ag} \cup Term_{Ac} \cup Term_{Gr} \cup Term_{U}$$

Usa-se τ (com decorações τ', τ_0, ...) para membros de $Term$. Para indicar que um termo particular τ é do tipo σ, adota-se o subscrito τ_σ.

Note-se que no que se refere aos termos, está se falando de constantes e variáveis, mas nada se diz sobre funcionais, porque não está se considerando os funcionais neste trabalho para não haver problemas de designação. Esse problema pode ocorrer quando se coloca aparato de primeira ordem em lógica modal em geral. Afinal, quando se tem uma função $f(x_1, ..., x_n)$, por ser um símbolo funcional, ele tem que ter um valor. Porém, para certas combinações de valores de x_i, pode ser que a função f seja indefinida. Por exemplo, seja uma função $r(x)$ que diz quem é o atual rei da França. Não existe atual rei da França, mas um símbolo funcional tem que ter valor. Tendo-se um predicado $Rei(x, y)$ que diz que y é o rei de x, não há problemas de designação vácua. Então, $f(x_1, ..., x_n) = y$ vira o predicado $F(x_1, ..., x_n, y)$. Se os funcionais estivessem presentes, existiriam opções com a semântica deles. Eles poderiam ter designação rígida (ter a mesma semântica em todos os mundos) ou não. Deve-se notar que com lógica de primeira-ordem, pode-se fazer a opção de deixá-los como definíveis a partir de fórmulas de primeira-ordem. Em qualquer dos casos (ter a mesma semântica sempre ou não) surgiriam problemas de designação (Kri06).

É importante dizer que, na seção anterior, ao se modelar apenas o sistema, usa-se variáveis proposicionais da forma a_i para se representar a presença do agente i no sistema. Nesta seção e nas seguintes, usa-se apenas i, pois não se está interessado em ter nenhum valor de verdade com relação a uma constante ou variável que representa um agente e sim dizer que determinada propriedade está associada a um certo agente específico.

Tem-se também $UCap$, que representa o universo de todas as capacidades possíveis que um agente pode ter.

A confiança, como já dito, foi definida como um predicado de primeira ordem, um tipo de predicado para cada uma das abordagens citadas na seção 4.6.

A sintaxe da lógica para lidar com os agentes é definida a seguir. A primeira parte, exibida na figura 5.1, mostra os elementos simples da linguagem: seus termos, predicados, variáveis e os elementos que descrevem as capacidades.

<ag-term>	:==	qualquer elemento de $Term_{Ag}$
<ac-term>	:==	qualquer elemento de $Term_{Ac}$
<gr-term>	:==	qualquer elemento de $Term_{Gr}$
<u-term>	:==	qualquer elemento de $Term_{U}$
<term>	:==	qualquer elemento de $Term$
<pred>	:==	qualquer elemento de $Pred$
<var>	:==	qualquer elemento de Var
<cap>	:==	qualquer elemento de $UCap$

Figura 5.1: Os elementos da linguagem

Já na figura 5.2, encontra-se a sintaxe das ações, que descreve o formato das ações simples e compostas que aparecem na linguagem. Note-se que as mensagens para outros agentes parecem ser *broadcasts*, já que não dizem quem é o receptor. Porém, essa informação pode ser passada em uma camada mais acima no protocolo de comunicação, não precisando ficar presente nesse nível. A gramática das ações é semelhante à PDL (Gab84). Note-se que LORA, ao contrário da linguagem definida aqui, não possui ações específicas, como *in*, *out*, etc. Apenas ações genéricas denominadas de α e composições de ações. Essa linguagem também não possui formas de se expressar novas ações específicas, que têm como primeiro parâmetro o agente que a executa, que, para um sistema de comércio eletrônico, poderiam ser, por exemplo:

- $vende(i, \varphi)$ – agente i vende φ;
- $compra(i, \varphi)$ – agente i compra φ;
- $vende(i, j, \varphi)$ – agente i vende φ ao agente j;
- $compra(i, j\varphi)$ – agente i compra φ do agente j;
- $paga(i, j)$ – agente i para o agente j,
- etc.

<ac-exp>	:==	<ac-exp>;<ac-exp> | <ac-exp> "|" <ac-exp>
	|	<ac-term> | <st-form>? | <ac-exp>*
	|	*says*(<ag-term>, <st-form>) | *in*(<ag-term>)
	|	*do*(<ag-term>, <ac-term>) | *out*(<ag-term>)
	|	<acao> (<ag-term>, <term>, ..., <term>)

Figura 5.2: Sintaxe das ações

Quanto às figuras 5.3 e 5.4, "st" significa estado (*state*) e "pt", caminho (*path*). Na figura 5.3, a modalidade $(said\, i\, \varphi)$ diz que o agente i disse φ. Nela, supõe-se que aos predicados sejam aplicados o número correto de argumentos. A sintaxe usada para lidar com fórmulas de estado e de caminho é a de

CTL^* (Cla99). Foi escolhida a lógica CTL^* e não simplesmente CTL (um subconjunto de CTL^*) porque, além de CTL^* ser usada em LORA, linguagem estendida neste texto, CTL não é expressiva o suficiente para capturar muitas das propriedades interessantes de estruturas temporais ramificadas (Woo00). Porém, o grau de complexidade do processo de *model checking* para CTL é polinomial, enquanto que para CTL^* é PSPACE-Completo (Azi94).

<st-form> :== $\top$ | $\bot$ | <pred> (<term>, <term>, ..., <term>)
| (*Bel* <ag-term><st-form>) | (<term> = <term>)
| (*Des* <ag-term><st-form>) | (*A* | *E*) <pt-form>
| (*Int* <ag-term><st-form>) | $\neg$ <st-form>
| (*said* <ag-term><st-form>) | <trust-pred>
| <st-form> ($\wedge$ | $\vee$ | $\rightarrow$ | $\leftrightarrow$) <st-form>
| (<ag-term> $\in$ <gr-term>)
| (*Agts* <ac-exp><gr-term>)
| ($\forall$ | $\exists$) <st-form>
| <cap> $\in$ *capacidades*(<term>)

Figura 5.3: Sintaxe das fórmulas de estado

<pt-form> :== (*Happens* <ac-exp>) | <st-form> | $\bigcirc$ <pt-form>
| <pt-form> ($\mathcal{U}$ | $\mathcal{W}$) <pt-form> | $\neg$ <pt-form>
| <pt-form> ($\wedge$ | $\vee$ | $\rightarrow$ | $\leftrightarrow$) <pt-form>
| ($\forall$ | $\exists$) <pt-form> | ($\Box$ | $\Diamond$) <pt-form>

Figura 5.4: Sintaxe das fórmulas de caminho

Por fim, os predicados de confiança binários serão verdadeiros quando a confiança for 1, e falsos, caso contrário. Nos demais casos, é preciso dar um valor v. Se o número de níveis de confiança for fixo, há um parâmetro adicional n que indica o número de níveis de confiança. Se a confiança for parametrizada em diferentes aspectos, o parâmetro c indica qual a capacidade do agente que está sendo avaliada em questão. Os predicados de confiança estão definidos na figura 5.5.

<trust-pred> :== $TrustB$(<ag-term>, <ag-term>)
| $TrustF$(<ag-term>, <ag-term>, Nat, Rac)
| $TrustV$(<ag-term>, <ag-term>, $Real$)
| $TrustPB$(<ag-term>, <ag-term>, <cap>)
| $TrustPF$(<ag-term>, <ag-term>, <cap>, Nat, Rac)
| $TrustPV$(<ag-term>, <ag-term>, <cap>, $Real$)

Figura 5.5: A sintaxe dos predicados de confiança

Note-se o uso dos parênteses nos operadores modais, tal como em Bel, de modo a tornar explícita a ligação entre a modalidade e os seus argumentos. E,

principalmente, para diferenciar dos predicados de primeira ordem, tal como os predicados para confiança.

5.4 Modelagem para o Agente

Para o agente, um mundo w é um par $< T', R'_T >$ tal que $T' \subseteq T$ é um conjunto não-vazio de instantes de tempo e $R'_T \subseteq R_T$ é uma estrutura de tempo ramificada em T', onde T e R_T foram definidos na seção 5.2.

Formalmente, W é definido como:

$$W = \{< T', R'_T > \mid T' \subseteq T, R'_T \subseteq R_T \text{ e } T' = (dom(R'_T) \cup range(R'_T)\}.$$

Se $w \in W$ for um mundo, então T_w representa o conjunto de instantes de tempo em w, e R_w, a relação de tempo ramificada em w. Mais ainda, um par $< w, t >$, onde $t \in T_w$ é chamado de situação, sendo que o conjunto de todas as situações em w é denotado por S_w:

$$S_w = \{< w, t > \mid w \in W \text{ e } t \in T_w\}.$$

A figura 5.6 ilustra os conceitos de mundos, passagem de tempo e situações. Cada mundo possui vários instantes de tempo e a relação R_T mostra quais instantes de tempo estão relacionados entre si. Já uma situação é representada por um retângulo na figura. Há mais de uma transição possível por causa do não-determinismo.

Note-se que os mundos não podem voltar ao passado (o passado é linear) e a cada instante sempre há passagem de tempo:

$$\forall t_1, t_2 \in T'(< t_1, t_2 > \in R'_T \rightarrow t_2 > t_1).$$

Adicionalmente, não pode haver convergência nos instantes de tempo futuros também por causa da linearidade do passado:

$$\forall t_1, t_2, t_3 \in T'(< t_1, t_3 > \in R'_T \wedge < t_2, t_3 > \in R'_T \rightarrow t_1 = t_2).$$

Com essa modelagem, é possível determinar se o histórico de transições de dois mundos são iguais, se um mundo é sub-mundo de outro, se dois mundos são equivalentes ou se são bi-similares (Cla99, Woo00).

Para usar a lógica modal como uma lógica epistêmica, a fórmula $\Box p$ é lida como "sabe-se p". Os mundos no modelo são interpretados como alternativas epistêmicas e a relação de acessibilidade define quais são as alternativas a partir de um dado mundo. A lógica lida com o conhecimento de um único agente. Para lidar com o conhecimento de vários agentes, é necessário modificar a estrutura

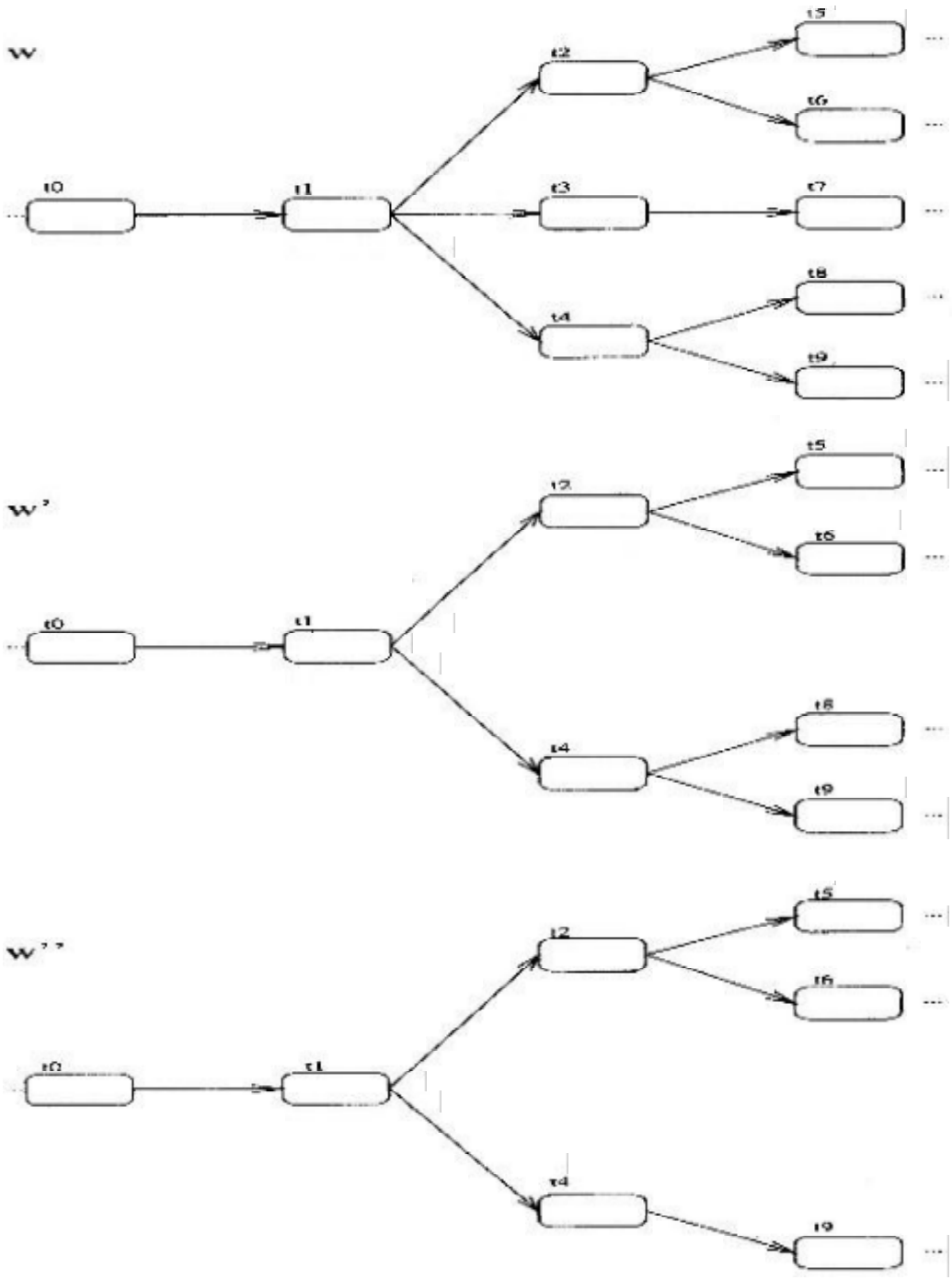

Figura 5.6: Exemplo de um modelo temporal ramificado

$< W, R >$, onde $W \neq \emptyset$ e $R \subseteq W \times W$, para que tenha um conjunto indexado de relações de acessibilidade, uma para cada agente, ficando com o formato $< W, R_0, ..., R_{n-1} >$, onde n é o número de agentes do sistema multi-agentes, R_i é a relação de acessibilidade do agente i e $i \in \{0, ..., n-1\}$. Estende-se agora o operador modal $\Box$ por um conjunto indexado de operadores modais $\{K_i\}$. A fórmula $K_i p$ deve ser lida como "o agente i sabe p", Como o operador modal $\Diamond$ é derivado do $\Box$, para se expressar o conceito de possibilidade será

usado $\neg K_i \neg p$, que expressa que "o agente i não sabe não-p". Resumindo, a lógica em questão é multi-modal, tendo como modalidades $K_0, K_1, ..., K_{n-1}$.

O estado de um agente é definido por suas crenças, desejos, intenções e confiança em outros agentes. Portanto, as crenças de um agente em uma dada situação, que é parcialmente indexada pelo tempo, são caracterizadas por um conjunto de situações, que são consistentes com as crenças do agente. Um agente acredita em p se p for verdade em todas as situações possíveis. Isso é chamado de "alternativas de crença" ou "situações acessíveis na crença". O mesmo pode-se dizer para os desejos e para as intenções.

É importante notar que a semântica de mundos possíveis apresentada nesta seção não é convencional, ou seja, os mundos não são instantâneos e sim estruturas de tempo ramificado. Isso é uma dificuldade usual na apresentação dos modelos em LORA (Woo00), linguagem que esse texto estende. A intuição é a de que tais estruturas representam a incerteza de um agente não só com relação a como o mundo é, mas também como ele vai ser no futuro. Os arcos na relação R, quando corresponderem à execução de ações atômicas executados pelos agentes dentro do sistema, terão como rótulo as ações aos quais estão relacionados via a função $Act()$, onde D_{Ac} é um conjunto não vazio de ações:

$$Act : R_T \rightarrow D_{Ac}.$$

Essa função é parcial, pois uma transição pode ocorrer sem que alguma ação de um agente ocorra. Nesse caso, a transição será rotulada como ϵ, havendo apenas passagem de tempo. Nessa passagem de tempo, pode haver modificação de algum estado mental de algum agente. A entrada ou saída de agentes também são consideradas ações, realizadas pelos agentes que entram ou saem, respectivamente.

Cada ação está associada a um único agente, dada pela função $Agt()$, onde $D_{Ag}(t)$ é um conjunto não vazio de agentes que está presente em um instante de tempo $t \in T$. Isso é necessário porque o sistema é aberto, onde o conjunto de agentes varia:

$$Agt : D_{Ac} \rightarrow D_{Ag}(t).$$

Caso uma ação seja executada por um grupo de agentes e não por somente um agente, apenas um agente desse grupo será relacionado à ação.

De modo a caracterizar as crenças de cada agente, usa-se a função $B()$, que atribui a cada agente uma relação sobre situações:

$$B : D_{Ag}(t) \rightarrow \mathcal{P}(W \times T \times W).$$

Funções $B()$ serão chamadas de relações de acessibilidade de crenças, embora, na verdade, elas sejam funções que atribuem relações de acessibilidade de crenças a agentes. De modo a simplificar, diz-se $B_t^w(i)$ para denotar o conjunto de mundos acessíveis ao agente i na situação $< w, t >$. Formalmente, $B_t^w(i)$ é definida como a seguir:

$$B_t^w(i) = \{w' \mid < w, \iota, w' > \in B(i)\}$$

As relações de acessibilidade de crenças devem satisfazer várias propriedades. A primeira é que, se um mundo w' for acessível a um agente na situação $< w, t >$, então t deve ser um instante de tempo tanto em w como em w'. Formalmente, esse requerimento (chamado de compatibilidade mundo/instante de tempo) é capturado como o seguinte: Se $w' \in B_t^w(i)$, então $\iota \in w$ e $\iota \in w'$.

Requer-se também que a relação que a função $B()$ atribui a todo agente seja serial, transitiva e euclidiana. $B()$ é dita:

- serial se para todas as situações $< w, t >$, houver algum mundo w' tal que $w' \in B_t^w(i)$;
- transitiva se $w' \in B_t^w(i)$ e $w'' \in B_t^{w'}(i)$ implicar em $w'' \in B_t^w(i)$, e
- euclidiana se $w' \in B_t^w(i)$ e $w'' \in B_t^w(i)$ implicar em $w'' \in B_t^{w'}(i)$.

Esses requisitos garantem que a lógica de crenças corresponde ao sistema lógico bem conhecido $KD45$ ($S5 - fraca$, que é o $S5$ sem a reflexividade)(Che80, Hug96, Woo00).

Para dar uma semântica aos desejos e às intenções, procede-se de maneira similar, definindo-se a função $D()$ para desejos e $I()$ para intenções. Assume-se que tanto $D()$ como $I()$ atribuem aos agentes relações seriais. Isso garante que as modalidades de desejos e intenções tenham uma lógica KD (Che80, Hug96, Woo00). Também é necessário que tanto $D()$ quanto $I()$ satisfaçam a propriedade de compatibilidade mundo/instante de tempo, análogo ao que se tem para $B()$. Assim como as relações de acessibilidade de crenças, escreve-se $D_t^w(i)$ e $I_t^w(i)$ para denotar os mundos acessíveis dos desejos e das intenções para um agente i na situação $< w, t >$.

Precisa-se também apresentar alguns aparatos técnicos para a manipulação de estruturas de tempo ramificado. Seja $w \in W$ um mundo. Então, um caminho através de w é uma seqüência $(t_0, t_1, ...)$ de instantes de tempo, tal que para todo $u \in Nat$, tem-se que $(t_u, t_{u+1}) \in R_w$. Seja $caminhos(w)$ a denotação dos caminhos que passam por w. Se p for um caminho e $u \in Nat$, então $p(u)$ denota o $u + 1$-ésimo elemento de p. Portanto, $p(0)$ é o primeiro ponto de p, $p(1)$ é o segundo e assim sucessivamente. Se p for um caminho e

$u \in Nat$, então o caminho obtido de p através da remoção dos primeiros u instantes de tempo é denotado por $p^{(u)}$.

É necessário também identificar os objetos aos quais se pode referir. A linguagem contém termos que identificam esses objetos e, em particular, esses são os objetos sobre os quais se pode quantificar. Esses objetos representam o domínio, o universo de discurso.

O domínio da linguagem apresentada neste texto contém os agentes que estão presentes em um dado instante de tempo, dado pelo conjunto $D_{Ag}(t)$. $D_{Ag}(t) \subseteq D_{Ag}$, onde D_{Ag} representa o conjunto de todos os agentes que podem vir a estar dentro do sistema. O domínio também contém as ações, representadas pelo conjunto D_{Ac}. Também permite-se quantificações sobre seqüências de ações, representadas pelo conjunto D^*_{Ac}; grupos de agentes presentes em determinado instante de tempo, representados pelo conjunto $D_{Gr}(t)$ (onde $D_{Gr}(t) \subseteq \mathcal{P}(D_{Ag}(t))$ e $D_{Gr} \subseteq \mathcal{P}(D_{Ag})$), e outros termos, representados pelo conjunto D_U.

Colocando esses componentes juntos, um domínio (de interpretação), D, é uma estrutura:

$$D(t) =< D_{Ag}(t), D_{Ac}, D_{Gr}(t), D_U >$$

onde

- $D_{Ag}(t) \subseteq D_{Ag} = \{0, 1, ..., n\}$, é um conjunto não-vazio de agentes presentes no instante t;
- $D_{Ac} = \{\alpha, \alpha', ...\}$ é um conjunto não-vazio de ações;
- $D_{Gr}(t) = \mathcal{P}(D_{Ag}(t))\backslash\{\emptyset\}$ é um conjunto não-vazio de subconjuntos de $D_{Ag}(t)$, ou seja, o conjunto de grupos sobre $D_{Ag}(t)$, e
- D_U é um conjunto não-vazio de outros indivíduos.

Se $D(t) =< D_{Ag}(t), D_{Ac}, D_{Gr}(t), D_U >$ for um domínio, denota-se por $\overline{D}$ pelo conjunto:

$$\overline{D(t)} = D_{Ag}(t) \cup D^*_{Ac} \cup D_{Gr}(t) \cup D_U$$

Se $u \in Nat$, então, por $\overline{D}^u$, tem-se o conjunto de u-tuplas sobre $\overline{D}$:

$$\overline{D}^u = \underbrace{\overline{D} \times ... \times \overline{D}}_{u}$$

Introduz-se também a função derivada $[[...]]_{V,C}$, a qual denota um termo arbitrário. Se V for uma atribuição de variável e C for uma interpretação para

constantes, então $[[...]]_{V,C}$ representa a função:

$$[[...]]_{V,C} : Term \times T \rightarrow \overline{D}$$

que interpreta termos arbitrários relativos a V e C, como a seguir:

$$[[\tau, t]]_{V,C} \doteq \begin{cases} C(\tau, t) & \text{se } \tau \in Const \\ V(\tau) & \text{caso contrário} \end{cases}$$

Como V, C e o instante de tempo no qual se está avaliando a fórmula sempre será claro pelo contexto, a referência a eles em geral será suprimida, escrevendo-se apenas $[[\tau]]$. Note-se que, assim como em LORA, as constantes não precisam necessariamente ter uma interpretação fixa, embora neste trabalho as constantes tenham uma interpretação fixa a não ser que o contrário seja explicitado. Já a interpretação de variáveis é fixa em todos os instantes de tempo.

Defini-se agora um modelo para a linguagem, que é uma estrutura como a seguir:

$$M = < T, R_T, W, D, Act, Agt, B, D, I, C, \Phi >$$

onde

- T é o conjunto de todos os instantes de tempo;
- $R_T \subseteq T \times T$ é uma relação total de tempo ramificada, linear no passado, sobre T;
- W é um conjunto de mundos sobre T;
- $D(t) = < D_{Ag}(t), D_{Ac}, D_{Gr}, D_U >$ é um domínio;
- $Act : R_T \rightarrow D_{Ac}$ é uma função parcial que associa uma ação com um arco em R_T;
- $Agt : D_{Ac} \rightarrow D_{Ag}(t)$ associa um agente presente em um instante $t \in T$ a cada ação;
- $B : D_{Ag}(t) \rightarrow \mathcal{P}(W \times T \times W)$ é uma relação de acessibilidade de crenças;
- $D : D_{Ag}(t) \rightarrow \mathcal{P}(W \times T \times W)$ é uma relação de acessibilidade de desejos;
- $I : D_{Ag}(t) \rightarrow \mathcal{P}(W \times T \times W)$ é uma relação de acessibilidade de intenções;
- $C : Const \times T \rightarrow \overline{D}$ interpreta constantes, e
- $\Phi : Pred \times W \times T \rightarrow \mathcal{P}(\bigcup_{u \in Nat} \overline{D}^u)$ interpreta predicados.

A semântica da linguagem é definida em duas partes, para fórmulas de caminho e para fórmulas de estado, respectivamente. A semântica das fórmulas de caminho é dada via a relação de satisfabilidade de fórmula de caminho $\models_p$, que é usada em interpretações de fórmulas de caminho e em fórmulas

de caminho. Uma interpretação de fórmula de caminho é uma estrutura $< M, V, w, p >$, onde M é um modelo, V é uma atribuição de variáveis, w é um mundo em M e p é um caminho a partir de w. As regras definindo a relação de satisfabilidade para fórmulas de caminho são dadas na figura 5.7. Se $< M, V, w, p > \models_p \varphi$, então diz-se que $< M, V, w, p >$ satisfaz φ, ou, equivalentemente, que φ é verdadeiro em $< M, V, w, p >$.

As regras semânticas da figura 5.7 para fórmulas de caminho fazem uso de uma importante definição auxiliar: *ocorre*(). Essa definição é usada para definir o operador *Happens*. Escreve-se $ocorre(\alpha, p, u, v)$ para indicar que uma ação α ocorre entre os instantes de tempo $u, v \in Nat$ no caminho p. Note-se que *ocorre*() é um predicado de meta-linguagem.

Formalmente, o predicado *ocorre*() é definido indutivamente por nove regras, uma para cada construtor de expressão de ação e uma para a execução de cada ação primitiva. A primeira regra representa um dos casos base, onde uma ação primitiva é executada:

$$ocorre(\alpha, p, u, v) \text{ sss } v = u + k \text{ e } [[\alpha, p(u)]] = \alpha_1, ..., \alpha_k \text{ e}$$
$$Act(p(u), p(u{+}1)) = \alpha_1, ..., Act(p(u{+}k{-}1), p(u{+}k)) = \alpha_k \text{ (onde } \alpha \in Term_{Ac})$$

Portanto, uma ação primitiva α ocorre no caminho p entre u e $u + 1$ se o arco de u até $u + 1$ for rotulado com α. Para uma ação específica construída especialmente para uma aplicação, não é definida uma semântica com o meta-predicado *ocorre*() porque as conseqüências dela variam de acordo com a ação executada.

Já a segunda regra captura a semântica de quando um agente i envia uma mensagem ao agente j. Esse agente deve dizer a mensagem em um determinado instante dentro do intervalo onde ela ocorreu. Logo, o "ou" abaixo é exclusivo:

$$ocorre(says(i, \varphi), p, u, v) \text{ sss } v = u + k \text{ e}$$
$$Act(p(u), p(u{+}1)) = says(i, \varphi) \text{ ou ... ou } Act(p(u{+}k{-}1), p(u{+}k)) = says(i, \varphi)$$
$$\text{e } (Bel\, j\, (said\, i\, \varphi))$$

A terceira regra para quando um agente i delega uma ação a um agente j é semelhante. É importante dizer que neste texto assume-se que o fato de um agente delegar uma tarefa a outro significa que o outro se comprometeu a fazer o que lhe foi pedido previamente. Porém, isso não quer dizer que j de fato o fará.

$$ocorre(do(j, \alpha), p, u, v) \text{ sss } v = u + k \text{ e}$$
$$Act(p(u), p(u+1)) = do(j, \alpha) \text{ ou ... ou } Act(p(u+k-1), p(u+k)) = do(j, \alpha)$$
$$\text{e } (Bel\, i\, do(j, \alpha))$$

Quando um novo agente entra, ele também entra apenas em um determinado instante de tempo. Mais ainda, esse agente passa a pertencer ao conjunto de agentes presentes no sistema no instante seguinte à sua entrada:

$$ocorre(in(i), p, u, v) \text{ sss } v = u + k \text{ e}$$
$$\text{se } Act(p(u), p(u+1)) = in(i) \text{ então } i \in (D_{Ag}(u+1) - D_{Ag}(u))$$
$$\text{ou ... ou}$$
$$\text{se } Act(p(u+k-1), p(u+k)) = in(i) \text{ então } i \in (D_{Ag}(u+k) - D_{Ag}(u+k-1))$$

A regra para a saída de um agente é semelhante à da entrada, sendo que agora o agente deixa de pertencer ao conjunto de agentes presentes no sistema no instante seguinte à sua saída:

$$ocorre(out(i), p, u, v) \text{ sss } v = u + k \text{ e}$$
$$\text{se } Act(p(u), p(u+1)) = out(i) \text{ então } i \in (D_{Ag}(u) - D_{Ag}(u+1))$$
$$\text{ou ... ou}$$
$$\text{se } Act(p(u+k-1), p(u+k)) = out(i) \text{ então } i \in (D_{Ag}(u+k-1) - D_{Ag}(u+k))$$

A sexta regra captura a semântica da composição seqüencial. A expressão de ação $\alpha; \alpha'$ ocorrerá entre os instantes de tempo u e v sss há algum instante de tempo n entre u e v tal que α é executada entre os instantes u e n e α, entre n e v:

$$ocorre(\alpha; \alpha', p, u, v) \text{ sss}$$
$$\exists n \in u, ..., v \text{ tal que } ocorre(\alpha, p, u, n) \text{ e } ocorre(\alpha', p, n, v)$$

A semântica da escolha não-determinística é ainda mais simples. A expressão de ação $\alpha|\alpha'$ será executada entre os instantes de tempo u e v sss α ou α' for executada entre esses instantes:

$$ocorre(\alpha|\alpha', p, u, v) \text{ sss } ocorre(\alpha, p, u, v) \text{ ou } ocorre(\alpha', p, u, v)$$

Já a semântica da interação se baseia no fato que executar α^* é o mesmo que:

- não fazer nada, ou
- executar α uma vez e depois executar α^*.

Isso leva à seguinte regra de ponto fixo:

$$ocorre(\alpha^*, p, u, v) \text{ sss } u = v \text{ ou } ocorre(\alpha; (\alpha^*), p, u, v)$$

Finalmente, tem-se a regra que define a semântica para as ações de teste. A idéia é a de que a expressão de ação $\varphi?$ ocorre entre os instantes de tempo u e

v no caminho p se φ for satisfeita em p no instante u. Essa regra recursivamente faz uso da definição de quando uma fórmula é satisfeita ou não em um estado, a ser definida a seguir. Na prática, a interpretação usada para avaliar a fórmula sempre será clara através do contexto:

$$ocorre(\varphi?, p, u, v) \text{ sss } < M, V, w, p(u) > \models_s \varphi$$

$< M, V, w, p > \models_p \varphi$	sss $< M, V, w, p(0) > \models_s \varphi$ (onde φ é uma fórmula de estado)
$< M, V, w, p > \models_p \neg\varphi$	sss $< M, V, w, p) > \not\models_p \varphi$
$< M, V, w, p > \models_p \varphi \vee \psi$	sss $< M, V, w, p > \models_p \varphi$ ou $< M, V, w, p > \models_p \psi$
$< M, V, w, p > \models_p \varphi \wedge \psi$	sss $< M, V, w, p > \models_p \varphi$ e $< M, V, w, p > \models_p \psi$
$< M, V, w, p > \models_p \varphi \rightarrow \psi$	sss $< M, V, w, p > \not\models_p \varphi$ ou $< M, V, w, p > \models_p \psi$
$< M, V, w, p > \models_p \varphi \leftrightarrow \psi$	sss $< M, V, w, p > \models_p \varphi$ sss $< M, V, w, p > \models_p \psi$
$< M, V, w, p > \models_p \forall x\varphi$	sss $< \overline{M, V \dagger \{x \mapsto d\}}, w, p) > \models_p \varphi$ $\forall u \in Nat$ e $\forall d \in \overline{D(p(u))}$ onde x e d são do mesmo tipo
$< M, V, w, p > \models_p \exists x\varphi$	sss $< \overline{M, V \dagger \{x \mapsto d\}}, w, p) > \models_p \varphi$ $\forall u \in Nat$ $\exists d \in \overline{D(p(u))}$ onde x e d são do mesmo tipo
$< M, V, w, p > \models_p \varphi \mathcal{U} \psi$	sss $\exists u \in Nat$ tal que $< M, V, w, p(u) > \models_p \psi$ e $\forall v \in Nat, (0 \leq v < u), < M, V, w, p(v) > \models_p \varphi$
$< M, V, w, p > \models_p \varphi \mathcal{W} \psi$	sss $\exists u \in Nat$ tal que $< M, V, w, p(u) > \models_p \psi$ e $\forall v \in Nat, (0 \leq v < u), < M, V, w, p(v) > \models_p \varphi$, ou $\forall v \in Nat, < M, V, w, p(v) > \models_p \varphi$
$< M, V, w, p > \models_p \bigcirc\varphi$	sss $< M, V, w, p(1) > \models_p \varphi$
$< M, V, w, p > \models_p \Box\varphi$	sss $\forall u \in Nat, < M, V, w, p(u) > \models_p \varphi$
$< M, V, w, p > \models_p \Diamond\varphi$	sss $\exists u \in Nat$ tal que $< M, V, w, p(u) > \models_p \varphi$
$< M, V, w, p > \models_p (Happens\, \alpha)$	sss $\exists u \in Nat$ tal que $ocorre(\alpha, p, 0, u)$

Figura 5.7: Regras definindo a semântica de fórmulas de caminho

A fórmula de caminho $\Diamond\varphi$ será satisfeita em algum caminho se φ for satisfeita em algum ponto ao longo do caminho. Já $\Box\varphi$ será satisfeita em algum caminho se φ for satisfeita em todos os pontos ao longo do caminho. Tem-se também uma versão fraca do conectivo $\mathcal{U}$: $\varphi \mathcal{W} \psi$ é lida como φ a não ser que ψ. Portanto, $\varphi \mathcal{W} \psi$ significa que:

- φ é satisfeita até que ψ seja satisfeita, ou
- φ é sempre satisfeita.

Diz-se que esse conectivo é fraco porque ele não requer que ψ seja eventualmente verdadeira. Logo, note-se que as seguintes equivalências são verdadeiras:

$$\Diamond\varphi \equiv true\, \mathcal{U}\, \varphi \qquad\qquad \varphi \mathcal{W} \psi \equiv (\varphi \mathcal{U} \psi) \vee \Box\varphi$$

A semântica para fórmulas de estado é dada via a relação de satisfabilidade de fórmula de estado $\models_s$. Essa relação é válida em interpretações de

fórmulas de estado e em fórmulas de estado. Interpretações de fórmulas de estado são estruturas da forma $< M, V, w, t >$, onde M é um modelo, V é uma atribuição de variáveis, w é um mundo em M e $t \in T_w$ é um instante de tempo em w. As regras definindo essa relação são dadas na figura 5.8. A semântica para os predicados de confiança é definida implicitamente ao se definir a semântica de predicados de primeira ordem, pois são apenas predicados especiais. O mesmo vale para uma fórmula que diz que determinada ação ou mensagem está relacionada à determinada capacidade, tal como para as funções que retornam uma sentença que representa um determinado estado mental do agente. Assim como com as fórmulas de caminho, se $< M, V, w, t > \models_s \varphi$, então diz-se que $< M, V, w, t >$ satisfaz φ, ou, equivalentemente, que φ é verdadeiro em $< M, V, w, t >$.

$< M, V, w, t > \models_s \top$	
$< M, V, w, t > \not\models_s \bot$	
$< M, V, w, t > \models_s P(\tau_1, ..., \tau_n)$	sss $([[\tau_1]], ..., [[\tau_n]]) \in \Phi(P, w, t)$
$< M, V, w, t > \models_s \neg\varphi$	sss $< M, V, w, t) > \not\models_s \varphi$
$< M, V, w, t > \models_s \varphi \vee \psi$	sss $< M, V, w, t > \models_s \varphi$ ou $< M, V, w, t > \models_s \psi$
$< M, V, w, t > \models_s \varphi \wedge \psi$	sss $< M, V, w, t > \models_s \varphi$ e $< M, V, w, t > \models_s \psi$
$< M, V, w, t > \models_s \varphi \rightarrow \psi$	sss $< M, V, w, t > \not\models_s \varphi$ ou $< M, V, w, t > \models_s \psi$
$< M, V, w, t > \models_s \varphi \leftrightarrow \psi$	sss $< M, V, w, s > \models_s \varphi$ sss $< M, V, w, t > \models_s \psi$
$< M, V, w, t > \models_s \forall x\varphi$	sss $< M, V \dagger \{x \mapsto d\}, w, t) > \models_s \varphi$ $\forall d \in \overline{D(t)}$ onde x e d são do mesmo tipo
$< M, V, w, t > \models_s \exists x\varphi$	sss $< M, V \dagger \{x \mapsto d\}, w, t) > \models_s \varphi$ $\exists d \in \overline{D(t)}$ onde x e d são do mesmo tipo
$< M, V, w, t > \models_s (Bel\, i\, \varphi)$	sss $\forall w' \in w$, se $w' \in B_t^w([[i]])$, então $< M, V, w', t > \models_s \varphi$
$< M, V, w, t > \models_s (Des\, i\, \varphi)$	sss $\forall w' \in w$, se $w' \in D_t^w([[i]])$, então $< M, V, w', t > \models_s \varphi$
$< M, V, w, t > \models_s (Int\, i\, \varphi)$	sss $\forall w' \in w$, se $w' \in I_t^w([[i]])$, então $< M, V, w', t > \models_s \varphi$
$< M, V, w, t > \models_s (said\, i\, \varphi)$	sss $< M, V, w, p > \models_p (Happens\, says(i, \varphi))$ e $\exists! p \in caminhos(w)$ e $\exists u \in Nat$ tal que $p(u) = t$
$< M, V, w, t > \models_s (Agts\, \alpha\, g)$	sss $Agts(\alpha, t) = [[g]]$
$< M, V, w, t > \models_s (\tau = \tau')$	sss $[[\tau]] = [[\tau']]$
$< M, V, w, t > \models_s (i \in g)$	sss $[[i]] \in [[g]]$
$< M, V, w, t > \models_s A\varphi$	sss $\forall p \in caminhos(w)$ se $p(0) = t$, então $< M, V, w, t > \models_p \varphi$
$< M, V, w, t > \models_s E\varphi$	sss $\exists p \in caminhos(w)$ se $p(0) = t$, então $< M, V, w, t > \models_p \varphi$

Figura 5.8: Regras definindo a semântica de fórmulas de estado

A regra para o operador $Agts$ faz uso de uma função de mesmo nome, o

qual estende a função Agt para expressões de ações arbitrárias:

$$Agts(\alpha, t) \doteq \begin{cases} \{Agt(\alpha_1), ..., Agt(\alpha_k)\} & \text{onde } [[\alpha, t]] = \alpha_1, ..., \alpha_k \\ Agts(\alpha_1, t) \cup Agts(\alpha_2, t) & \text{onde } \alpha = \alpha_1; \alpha_2 \text{ ou } \alpha = \alpha_1 | \alpha_2 \\ Agts(\alpha_1, t) & \text{onde } \alpha = \alpha_1^* \\ \emptyset & \text{onde } \alpha \text{ é da forma } \varphi? \end{cases}$$

O primeiro caso na definição indutiva lida com seqüências de ações. O segundo caso lida com a composição seqüencial e a escolha não-determinística. O terceiro caso lida com interação e o caso final, com ações teste.

Os predicados de confiança são tratados como predicados comuns. Embora apenas um tipo desses predicados possa ser usado para modelar a confiança em um determinado tipo de situação em uma aplicação, semanticamente eles podem ser equivalentes em alguns casos. Por exemplo, usar a confiança binária é equivalente a usar a confiança com apenas dois valores fixos, tanto no caso global como no parametrizado, onde MIN e MAX são constantes lógicas representando os graus mínimos e máximos de confiança, que em geral são 0 e 1, respectivamente:

$$\forall i \forall j (TrustB(i,j) \leftrightarrow TrustV(i,j,2,MAX))$$
$$\wedge (\neg TrustB(i,j) \leftrightarrow TrustV(i,j,2,MIN))$$

$$\forall i \forall j \forall c (TrustPB(i,j,c) \leftrightarrow TrustPV(i,j,c,2,MAX))$$
$$\wedge (\neg TrustPB(i,j,c) \leftrightarrow TrustPV(i,j,c,2,MIN))$$

Já quando um agente confia em todas as capacidades de um parceiro, tem-se que isso é equivalente à confiança global total. O mesmo ocorre no caso da desconfiança total:

$$\forall i \forall j \forall c (TrustB(i,j) \leftrightarrow TrustV(i,j,MAX) \leftrightarrow TrustF(i,j,n,MAX)$$
$$\leftrightarrow (TrustPB(i,j,c) \leftrightarrow TrustPV(i,j,c,MAX) \leftrightarrow TrustPF(i,j,c,n,MAX))$$

$$\forall i \forall j \forall c (\neg TrustB(i,j) \leftrightarrow TrustV(i,j,MIN) \leftrightarrow TrustF(i,j,n,MIN)$$
$$\leftrightarrow (\neg TrustPB(i,j,c) \leftrightarrow TrustPV(i,j,c,MIN)$$
$$\leftrightarrow TrustPF(i,j,c,n,MIN))$$

Contudo, quando se tem o mesmo grau de confiança para cada capacidade não necessariamente se tem esse valor para a confiança global, pois cada capacidade do agente pode ter maior ou menor importância dependendo da aplicação.

Finalizando, com a semântica da lógica, é possível se caminhar na estrutura e saber informações do estado mental de um agente partindo de um

estado mental de um outro agente. Dessa maneira, pode-se calcular o valor de verdade de uma fórmula onde um agente tem uma fórmula num estado mental que depende do estado mental de outro agente. Porém, para um agente saber do estado mental de um outro agente, os agentes precisam trocar informações a esse respeito. Por exemplo, para o agente i acreditar que o agente j acredita em φ ($(Bel\, i\, (Bel\, j\, \varphi))$, j tem que dizer a i que ele acredita em φ. Entretanto, i não pode ter a certeza de que j está falando a verdade. Se o grau de confiança que i tem em j for alto, ele pode até considerar o que seu parceiro disse como se fosse verdadeiro, mas ele não tem como saber se o que lhe foi dito é a verdade.

5.5 Os Estados Mentais de um Agente

Como já foi dito, cada um dos componentes de um agente BDI é modelado como um operador modal, onde cada agente tem as suas próprias modalidades referentes a crenças, desejos e intenções:

$(Bel\, i\, \varphi)$	O agente i acredita em φ.
$(Des\, i\, \varphi)$	O agente i tem o desejo φ.
$(Int\, i\, \varphi)$	O agente i tem a intenção φ.

A confiança não é definida como modalidade porque apenas diz se um agente confia ou não em outro, podendo haver graus de confiança e/ou parametrização. Portanto, não se pode confiar numa fórmula da lógica aqui definida, que exprimem fatos, e não outros agentes.

A confiança é um predicado de primeira ordem. Foram definidos vários predicados, um para cada tipo de confiança. Os predicados a seguir serão verdadeiros se e somente se:

$TrustB(i, j)$	O agente i confia no agente j.
$TrustF(i, j, n, v)$	O agente i tem um grau de confiança v (v de valor) em j, onde v é um dos n possíveis valores para o grau de confiança.
$TrustV(i, j, v)$	O agente i tem um grau de confiança v em j, onde v é um valor entre 0 e 1.
$TrustPB(i, j, c)$	O agente i confia no agente j quanto à capacidade c.
$TrustPF(i, j, c, n, v)$	O agente i tem um grau de confiança v em j quanto à capacidade c, onde v é um dos n possíveis valores para o grau de confiança da capacidade c.
$TrustPV(i, j, c, v)$	O agente i tem um grau de confiança v em j quanto à capacidade c, onde v é um valor entre 0 e 1.

K, D e o axioma da necessidade são válidos para todas as modalidades citadas acima. Para as crenças, valem também os axiomas 4 e 5. Logo, tem-se os seguintes teoremas para os estados mentais, cujas provas formais podem ser encontradas logo a seguir (Woo00):

1. $\models_s (Bel\, i\, (\varphi \rightarrow \psi)) \rightarrow ((Bel\, i\, \varphi) \rightarrow (Bel\, i\, \psi))$ (K)
2. $\models_s (Bel\, i\, \varphi) \rightarrow \neg(Bel\, i\, \neg\varphi)$ (D)
3. $\models_s (Bel\, i\, \varphi) \rightarrow (Bel\, i\, (Bel\, i\, \varphi))$ (4)
4. $\models_s \neg(Bel\, i\, \varphi) \rightarrow (Bel\, i\, \neg(Bel\, i\, \varphi))$ (5)
5. se $\models_s \varphi$ então $\models_s (Bel\, i\, \varphi)$ (necessidade)
6. $\models_s (Des\, i\, (\varphi \rightarrow \psi)) \rightarrow ((Des\, i\, \varphi) \rightarrow (Des\, i\, \psi))$ (K)
7. $\models_s (Des\, i\, \varphi) \rightarrow \neg(Des\, i\, \neg\varphi)$ (D)
8. se $\models_s \varphi$ então $\models_s (Des\, i\, \varphi)$ (necessidade)
9. $\models_s (Int\, i\, (\varphi \rightarrow \psi)) \rightarrow ((Int\, i\, \varphi) \rightarrow (Int\, i\, \psi))$ (K)
10. $\models_s (Int\, i\, \varphi) \rightarrow \neg(Int\, i\, \neg\varphi)$ (D)
11. se $\models_s \varphi$ então $\models_s (Int\, i\, \varphi)$ (necessidade)

1. *Prova.* Suponha-se que $< M, V, w, t > \models_s (Bel\, i\, (\varphi \rightarrow \psi))$ e $< M, V, w, t > \models_s (Bel\, i\, \varphi)$ para um $< M, V, w, t >$ arbitrário. Usando-se a semântica do Bel, sabe-se que $< M, V, w', t > \models_s \varphi \rightarrow \psi$ e $< M, V, w', t > \models_s \varphi$ para todo w' tal que $w' \in B_t^w([[i]])$. Então $< M, V, w', t > \models_s \psi$ e, portanto, $< M, V, w, t > \models_s (Bel\, i\, \psi)$. ■

2. *Prova.* Suponha-se o contrário. Então, para algum $< M, V, w, t >$, tem-se tanto $< M, V, w', t > \models_s (Bel\, i\, \varphi)$ quanto $< M, V, w', t > \models_s (Bel\, i\, \neg\varphi)$. Como a relação de acessibilidade de crença é serial, sabe-se que há pelo menos um $w' \in B_t^w([[i]])$, e, a partir da semântica de Bel, sabe-se que $< M, V, w', t > \models_s \varphi$ e $< M, V, w', t > \models_s \neg\varphi$. Logo, $< M, V, w', t > \models_s \varphi$ e $< M, V, w', t > \nvdash_S \varphi$, o que é uma contradição. ■

3. *Prova.* Suponha-se que $< M, V, w, t > \models_s (Bel\, i\, \varphi)$ para um $< M, V, w, t >$ arbitrário. Como a relação de acessibilidade de crença é transitiva, para todo w' e w'', se $w' \in B_t^w([[i]])$ e $w'' \in B_t^{w'}([[i]])$, então $w'' \in B_t^w([[i]])$. Usando-se a semântica de Bel, tem-se que $< M, V, w', t > \models_s \varphi$ e $< M, V, w'', t > \models_s \varphi$ pois $w' \in B_t^w([[i]])$ e $w'' \in B_t^w([[i]])$, respectivamente. Como tem-se que $w'' \in B_t^{w'}([[i]])$ e

$< M, V, w'', t > \models_s \varphi$, tem-se que $< M, V, w', t > \models_s (Bel\, i\, \varphi)$. Por fim, como tem-se que $< M, V, w', t > \models_s (Bel\, i\, \varphi)$ e $w' \in B_t^w([[i]])$, tem-se que $< M, V, w, t > \models_s (Bel\, i\, (Bel\, i\, \varphi))$. ■

4. *Prova.* Suponha-se que $< M, V, w, t > \models_s \neg(Bel\, i\, \varphi)$. Então, para algum $w' \in B_t^w([[i]])$, tem-se que $< M, V, w', t > \models_s \neg\varphi$. Agora, considera-se qualquer mundo w'' tal que $w'' \in B_t^w([[i]])$. Como a relação de acessibilidade de crença é euclidiana, e tem-se que $w' \in B_t^w([[i]])$ e $w'' \in B_t^w([[i]])$, tem-se também que $w' \in B_t^{w''}([[i]])$. Como $w'' \in B_t^w([[i]])$, tem-se também que $< M, V, w'', t > \models_s \neg\varphi$. Como w'' é arbitrário, $w' \in B_t^{w''}([[i]])$ e $< M, V, w'', t > \models_s \neg\varphi$, tem-se que $< M, V, w', t > \models_s \neg(Bel\, i\, \varphi)$. Por fim, como $w' \in B_t^w([[i]])$ e $< M, V, w', t > \models_s \neg(Bel\, i\, \varphi)$, tem-se que $< M, V, w, t > \models_s (Bel\, i \neg(Bel\, i\, \varphi))$. ■

5. *Prova.* Suponha-se que $\models_s \varphi$. Como $\models_s \varphi$, pela definição φ é satisfeita em todas as estruturas de interpretação e, em particular, em $< M, V, w', t > \models_s \varphi$ para todo $w' \in B_t^w([[i]])$. Logo, $< M, V, w, t > \models_s (Bel\, i\, \varphi)$. ■

6. *Prova.* Semelhante a prova de 1. ■

7. *Prova.* Semelhante a prova de 2. ■

8. *Prova.* Semelhante a prova de 5. ■

9. *Prova.* Semelhante a prova de 1. ■

10. *Prova.* Semelhante a prova de 2. ■

11. *Prova.* Semelhante a prova de 5. ■

O axioma da necessidade pode parecer estranho à primeira vista, pois torna o agente onisciente; ele acredita em tudo o que é verdade. Em se tratando de seres humanos, por exemplo, tal axioma não faria sentido. Contudo, esses axiomas sempre dizem respeito a fórmulas lógicas. Portanto, um agente capaz de fazer inferências lógicas é obrigado a acreditar nas tautologias como sendo verdadeiras. O axioma da necessidade também é considerado válido para os desejos e para as intenções mais para tornar esses operadores normais. Também pode-se dizer que um agente deseja e intenciona tudo o que é válido implicitamente.

Se essas modalidades fossem reflexivas, os seguintes axiomas valeriam:

$$\models_s (Bel\, i\, \varphi) \rightarrow \varphi \qquad \models_s (Des\, i\, \varphi) \rightarrow \varphi \qquad \models_s (Int\, i\, \varphi) \rightarrow \varphi$$

Porém, tais axiomas não fazem sentido no caso geral; dizer que se o agente acredita, deseja ou intenciona algo, é dizer que esse algo é verdadeiro. Por exemplo, o agente pode acreditar em algo que não é verdade por estar percebendo mal o seu ambiente. Já se o agente deseja ou intenciona algo, se seu desejo ou a sua intenção ainda não tiver sido alcançado, ele não será verdade. Logo, nenhuma dessas modalidades é reflexiva.

Os axiomas abaixo também não fariam sentido para os desejos e as intenções, porque não é muito razoável dizer que se deseja que deseja algo ou deseja que não deseja algo. A mesma intuição vale para a modalidade de intenção.

$$\models_s (Des\, i\, \varphi) \rightarrow (Des\, i\, (Des\, i\, \varphi)) \quad \models_s \neg(Des\, i\, \varphi) \rightarrow (Des\, i\, \neg(Des\, i\, \varphi))$$
$$\models_s (Int\, i\, \varphi) \rightarrow (Int\, i\, (Int\, i\, \varphi)) \quad \models_s \neg(Int\, i\, \varphi) \rightarrow (Int\, i\, \neg(Int\, i\, \varphi))$$

É sempre possível se acreditar em algo que está em um dos estados mentais do agente. Ou seja, os seguintes axiomas valem:

$$\models_s (Des\, i\, \varphi) \rightarrow (Bel\, i\, (Des\, i\, \varphi))$$
$$\models_s (Int\, i\, \varphi) \rightarrow (Bel\, i\, (Int\, i\, \varphi))$$
$$\models_s \forall i \forall j (TrustB(i,j) \rightarrow (Bel\, i\, TrustB(i,j)))$$
$$\models_s \forall i \forall j (TrustPB(i,j,c) \rightarrow (Bel\, i\ TrustPB(i,j,c)))$$
$$\models_s \forall i \forall j (TrustF(i,j,n,v) \rightarrow (Bel\, i\ TrustF(i,j,n,v)))$$
$$\models_s \forall i \forall j (TrustPF(i,j,c,n,v) \rightarrow (Bel\, i\ TrustPF(i,j,c,n,v)))$$
$$\models_s \forall i \forall j (TrustV(i,j,v) \rightarrow (Bel\, i\ TrustV(i,j,v)))$$
$$\models_s \forall i \forall j (TrustPV(i,j,c,v) \rightarrow (Bel\, i\ TrustPV(i,j,c,v)))$$

Os contrários dos axiomas acima também valem. Ou seja, se o agente acredita em algo que está em um dos seus estados mentais, então esse algo está no estado mental. Por exemplo, se um agente acredita que deseja algo, então ele deseja esse algo. Se um agente acredita que intenciona algo, então ele intenciona esse algo. E assim por diante. O mesmo também se pode dizer com relação às crenças. Se um agente acredita que acredita em algo, então ele acredita nesse algo.

$$\models_s (Bel\, i\, (Bel\, i\, \varphi)) \rightarrow (Bel\, i\, \varphi)$$
$$\models_s (Bel\, i\, (Des\, i\, \varphi)) \rightarrow (Des\, i\, \varphi)$$
$$\models_s (Bel\, i\, (Int\, i\, \varphi)) \rightarrow (Int\, i\, \varphi)$$
$$\models_s \forall i \forall j ((Bel\, i\, TrustB(i,j)) \rightarrow TrustB(i,j))$$
$$\models_s \forall i \forall j ((Bel\, i\ TrustPB(i,j,c)) \rightarrow TrustPB(i,j,c))$$
$$\models_s \forall i \forall j ((Bel\, i\ TrustF(i,j,n,v)) \rightarrow TrustF(i,j,n,v))$$
$$\models_s \forall i \forall j ((Bel\, i\ TrustPF(i,j,c,n,v)) \rightarrow TrustPF(i,j,c,n,v))$$
$$\models_s \forall i \forall j ((Bel\, i\ TrustV(i,j,v)) \rightarrow TrustV(i,j,v))$$
$$\models_s \forall i \forall j ((Bel\, i\ TrustPV(i,j,c,v)) \rightarrow TrustPV(i,j,c,v))$$

Deve-se notar que não se usa quantificadores em relação às "variáveis" de agentes ligadas às modalidades. Isso acontece porque, na verdade, a "variável" de identificador de agente não é um parâmetro da modalidade e sim um índice, havendo uma modalidade de cada tipo para cada agente.

Por fim, foram apresentadas algumas relações entre as modalidades que representam os estados mentais do agente. Como o foco deste trabalho é a confiança em um agente BDI, não serão explicadas as demais relações entre essas modalidades e entre os fatos que ocorrem no sistema, que são muito importantes, porém já estão bem desenvolvidas na literatura, como se pode ver em (Woo00).

5.6 Outras Modalidades

Além das crenças, dos desejos e das intenções, há outras modalidades importantes, onde algumas serão discutidas nesta seção.

Primeiramente, há o quantificador universal de caminho A que se comporta como na lógica modal normal $S5$. E é o seu dual, ou seja $E\varphi \equiv \neg A \neg \varphi$. A prova dos teoremas abaixo encontram-se a seguir (Woo00):

1. $\models_s A\,(\varphi \rightarrow \psi) \rightarrow (A\,\varphi \rightarrow A\,\psi)$ (K)
2. $\models_s A\,\varphi \rightarrow \neg A\,\neg \varphi$ (D)
3. $\models_s A\,\varphi \rightarrow \varphi$ (T, para fórmulas de estado φ)
4. $\models_s A\,\varphi \rightarrow A\,A\,\varphi$ (4)
5. $\models_s \neg A\,\varphi \rightarrow A\,\neg A\,\varphi$ (5)
6. se $\models_s \varphi$ então $\models_s A\,\varphi$ (necessidade)

1. *Prova.* Suponha-se que $< M, V, w, t >\models_s A\,(\varphi \rightarrow \psi)$ e $< M, V, w, t >\models_s A\,\varphi$ para $< M, V, w, t >$ arbitrário. Então, $< M, V, w, p >\models_p \varphi \rightarrow \psi$ e $< M, V, w, p >\models_p \varphi$ para todo $p \in caminhos(w)$ tal que $p(0) = t$. Portanto, $< M, V, w, p >\models_p \psi$ para todo $p \in caminhos(w)$ tal que $p(0) = t$. Logo, tem-se que $< M, V, w, t >\models_s A\,\psi$. ■

2. *Prova.* Suponha-se que $< M, V, w, t >\models_s A\,\varphi$ e $< M, V, w, t >\models_s A\,\neg\varphi$ para $< M, V, w, t >$ arbitrário. Então, pela semântica de A, tem-se que $< M, V, w, p >\models_p \varphi$ e $< M, V, w, p >\models_p \neg\varphi$ para todo $p \in caminhos(w)$ tal que $p(0) = t$. Como a relação de tempo ramificada é total, tem-se que há pelo menos um caminho com tal propriedade. Logo, $< M, V, w, p >\models_p \varphi$ e $< M, V, w, p >\not\models_p \varphi$, o que é uma contradição. ■

3. *Prova.* Suponha-se que $< M, V, w, t >\models_s A\,\varphi$ para $< M, V, w, t >$ arbitrário. Então, pela semântica de A, sabe-se que $< M, V, w, p >\models_p \varphi$ para todo $p \in caminhos(w)$ tal que $p(0) = t$. Como a relação de tempo ramificada é total, tem-se que há pelo menos um caminho com tal propriedade. Portanto, tem-se que $< M, V, w, t >\models_s \varphi$. ■

4. *Prova.* Suponha-se que $< M, V, w, t >\models_s A\,\varphi$ para $< M, V, w, t >$ arbitrário. Como $A\,\varphi$ é uma fórmula de estado, tem-se que $< M, V, w, p >\models_p A\,\varphi$ para todo $p \in caminhos(w)$ tal que $p(0) = t$. Logo, tem-se que $< M, V, w, t >\models_s A\,A\,\varphi$. ■

5. *Prova.* Suponha-se que $< M, V, w, t >\models_s \neg A\,\varphi$ para $< M, V, w, t >$ arbitrário. Como $\neg A\,\varphi$ é uma fórmula de estado, tem-se que $< M, V, w, p >\models_p \neg A\,\varphi$ para todo $p \in caminhos(w)$ tal que $p(0) = t$. Logo, tem-se que $< M, V, w, t >\models_s A\,\neg A\,\varphi$. ■

6. *Prova.* Suponha-se que $\models_s \varphi$. Como $\models_s \varphi$, pela definição φ é satisfeita em todas as estruturas de interpretação e, em particular, em $< M, V, w, t >\models_s \varphi$ para todo w. Como φ é uma fórmula de estado, tem-se que $< M, V, w, p >\models_p \varphi$ para todo $p \in caminhos(w)$ tal que $p(0) = t$. Logo, tem-se que $< M, V, w, t >\models_s A\,\varphi$. Como w é arbitrário, tem-se que $\models_s A\,\varphi$. ■

Tem-se também a modalidade *said*, que diz que um determinado agente disse algo:

$$(said\,i\,\varphi) \qquad \text{O agente } i \text{ disse } \varphi.$$

Para essa modalidade, valem K e o axioma da necessidade. Quanto ao último, se algo é válido e como os agentes são racionais, sempre pode-se dizer que eles disseram esse algo. Então

$$\models_s (said\,i\,(\varphi \rightarrow \psi)) \rightarrow ((said\,i\,\varphi) \rightarrow (said\,i\,\psi))\ (K)$$
$$\text{se } \models_s \varphi \text{ então } \models_s (said\,i\,\varphi)\ \text{(necessidade)}$$

Já os axiomas 4, 5 e T obviamente não valem. Ou seja, tais axiomas não são válidos:

$\models_s (said\,i\,\varphi) \rightarrow (said\,i\,(said\,i\,\varphi))$ $\qquad$ $\models_s \neg(said\,i\,\varphi) \rightarrow (said\,i\,\neg(said\,i\,\varphi))$

$\models_s (said\,i\,\varphi) \rightarrow \varphi$

Quanto ao axioma D, ele pode valer ou não, dependendo do modelo adotado. Se os agentes nunca se contradizerem, ele vale. Caso contrário, ele não vale. Se os agentes fossem seres humanos, obviamente o axioma não valeria. Mais ainda, se o agente estiver dizendo algo sobre o que ele vê de seu ambiente

e esse ambiente se modificar, fazendo com que a propriedade descrita deixe de valer, D também não será verdadeiro. Eis o axioma:

$$\models_s (said\, i\, \varphi) \rightarrow \neg(said\, i\, \neg\varphi)(D))$$

Por fim, assim como com os desejos e as intenções, se algo foi dito por um agente, o mesmo vai acreditar que disse esse algo:

$$\models_s (said\, i\, \varphi) \rightarrow (Bel\, i\, (said\, i\, \varphi))$$

Ou seja, um agente sempre acredita no que está em seus estados mentais e que emitiu determinada mensagem.

Quando um agente percebe algo no ambiente, como já dito, ele armazena uma representação disso em suas crenças. Logo, se um agente j diz algo, i deve armazenar essa informação se puder percebê-la. Deste ponto em diante supõe-se que o receptor da mensagem sempre consegue percebê-la corretamente para efeitos de não deixar as sentenças ao longo do resto do texto muito longas. Também assume-se que o agente possa perceber uma ação executada no ambiente instantaneamente. Logo, tem-se as seguintes sentenças, que possuem o mesmo significado. Note-se que é feita a suposição de que a atualização das crenças e do modelo de confiança são instantâneas, algo que não necessariamente ocorre em uma implementação real de agentes. Nesse caso, $\bigcirc$ deveria ser trocado por $A\Diamond$, significando que, qualquer que seja a execução futura do agente, a propriedade será verdadeira em algum instante futuro. Um exemplo onde isso poderia ocorrer é se o agente precisar fazer uma análise detalhada de sua percepção para poder atualizar as suas crenças.

$$(said\, j\, \varphi) \rightarrow (Bel\, i\, (said\, j\, \varphi)) \qquad (Happens\, says(j, \varphi)) \rightarrow \bigcirc(Bel\, i\, (said\, j\, \varphi))$$

Portanto, a modalidade *said* também pode ser vista como uma camada adicional nas crenças de um agente para a comunicação entre agentes.

5.7 Relacionando Confiança no Modelo

Nesta seção relaciona-se o modelo de confiança de um agente com os seus demais estados mentais, com o comportamento dos demais agentes e com a entrada de um agente do sistema, formulando-se, assim, propriedades que possivelmente são desejáveis na relação entre agentes BDI em um MAS aberto. Nas sub-seções a seguir tenta-se enumerar vários tipos de propriedades, porém, não se pretende esgotá-las. Deve-se dizer que elas podem ser usadas

para a validação de um sistema multi-agentes aberto, para saber se ele se comporta como o desejado. Ou então, como políticas e/ou recomendações para o funcionamento adequado de um MAS com confiança aberto.

Nas fórmulas das subseções a seguir, usa-se combinações dos operadores de estado e caminho, como em CTL. Então, tem-se que:

- $A\Box P$ significa que a propriedade P vale para todo mundo em todo caminho;
- $A\Diamond P$ significa que a propriedade P vale para algum mundo em todo caminho;
- $E\Box P$ significa que a propriedade P vale para todo mundo em algum caminho, e
- $E\Diamond P$ significa que a propriedade P vale para algum mundo em algum caminho.

Confiança de um agente em si mesmo

Nesta subseção, aborda-se a confiança que um agente pode ter em si mesmo e a relação disso com a confiança nos demais agentes. No caso puramente binário, o fato de um agente sempre confiar em si próprio pode ser um axioma. Afinal, um agente que não confia em si não estará sendo racional, o que não é de interesse deste trabalho. Portanto:

$$\models_s \forall i\, TrustB(i,i)$$

Em casos reais, pode-se ter agentes patológicos que não confiam em si, tal como ocorre com pessoas em profunda depressão.

Quanto aos casos onde a confiança pode ter mais de dois valores, um agente não necessariamente precisa confiar 100% em si mesmo, mas assume-se que precisa confiar mais em si do que nos outros. Afinal, assim como com os seres humanos, sempre se conhece mais de si mesmo do que dos outros. Novamente, um agente que confia mais nos outros do que em si mesmo não está sendo racional, algo que não é de interesse deste texto. Logo:

$$\models_s \forall i \forall j\, (TrustF(i,j,n,v_1) \wedge TrustF(i,i,n,v_2) \rightarrow v_2 \geq v_1)$$
$$\models_s \forall i \forall j\, (TrustV(i,j,v_1) \wedge TrustV(i,i,v_2) \rightarrow v_2 \geq v_1)$$

Por fim, quanto à confiança parametrizada, a confiança que o agente tem de suas próprias capacidades varia de acordo com o que ele sabe do que é capaz. Nesse caso, ele pode confiar mais em um outro agente do que em si mesmo com relação a determinadas capacidades. Sendo assim, se um outro

agente for mais capaz do que ele para realizar determinada tarefa, é possível que ele prefira delegá-la ao outro. Isso não implica que ele delegará a tarefa: pode ser que o agente ao qual se quer delegar a tarefa não possa executá-la por estar muito ocupado. Ou então, como a realização de uma tarefa em geral exige mais de uma capacidade, pode ser que o outro agente seja muito capaz em algumas e em outras não. Também tem-se capacidades com relação às mensagens que um agente envia. Afinal, um agente, ao enviar uma mensagem, diz algo sobre determinado assunto, podendo ele falar sobre algo que entende ou não. As capacidades necessárias para realizar uma ação ou dizer algo formam um conjunto que está dentro de um universo de todas as capacidades possíveis.

$$capacidades : D_{Ac} \bigcup D_U \rightarrow \mathcal{P}(UCap)$$

Então, $c \in capacidades(\varphi)$ se e somente se c for uma capacidade necessária para a execução adequada da ação φ ou para falar com propriedade sobre φ.

Com relação à delegação de tarefas, pode-se formalizar isso da seguinte maneira: se um agente i precisar realizar uma ação α e há um outro agente j com alguma capacidade para realizar essa ação maior do que a do agente i, é possível que em algum estado futuro essa tarefa seja delegada ao outro agente. Desse modo, tem-se as seguintes sentenças para cada tipo de confiança parametrizada:

$$\forall i \exists j \exists c((c \in capacidades(\alpha) \wedge TrustPB(i,j,c) \wedge \neg TrustPB(i,i,c) \rightarrow E\Diamond(Happens\, do(j,\alpha)))$$

$$\forall i \exists j \exists c((c \in capacidades(\alpha) \wedge TrustPF(i,j,c,n,v_1) \wedge TrustPF(i,i,c,n,v_2) \wedge v_1 > v_2 \rightarrow E\Diamond(Happens\, do(j,\alpha)))$$

$$\forall i \exists j \exists c((c \in capacidades(\alpha) \wedge TrustPV(i,j,c,v_1) \wedge TrustPV(i,i,c,v_2) \wedge v_1 > v_2 \rightarrow E\Diamond(Happens\, do(j,\alpha)))$$

É importante notar que o fato de o agente i encontrar um agente j que ele acha que tem a maior capacidade entre todos os agentes que ele conhece para executar uma tarefa não quer dizer que a tarefa será delegada a j. Aqui, ter a maior capacidade significa uma avaliação global de todas as capacidades para realizar uma ação. Mais uma vez, há fatores que podem fazer com que i prefira delegá-la a outro agente e até mesmo fazê-la ele mesmo. Por exemplo, se o agente j for desonesto ou não puder cumprir a tarefa solicitada por algum motivo (como estar muito atarefado no momento).

O agente i também pode vir a delegar uma tarefa a um agente j em quem ele não confia ou possui pouca confiança. Isso pode acontecer quando i não tiver como fazer determinada tarefa e não tiver nenhum parceiro minimamente confiável com quem interagir.

Modificação do Grau de Confiança

Nesta subseção, descreve-se sentenças mostrando propriedades que descrevem quando e como a confiança que um agente tem em outro deve ser modificada.

Abaixo, relaciona-se o predicado de confiança com as crenças de um agente ou ações tomadas pelo agente sobre o que um outro agente disse ou se o outro agente fez o que lhe fora pedido.

No caso da confiança binária, um agente i não confia no agente j caso o último tenha dito algo que entre em contradição com as crenças de i.

$$\forall i \forall j((Bel\, i\, (said\, j\, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi) \rightarrow \neg TrustB(i,j))$$

É importante notar que o fato de a informação enviada entrar em contradição com o que o agente acredita não significa que j tenha lhe mandado uma informação inadequada com relação ao ambiente ou agido maliciosamente (mentido propositalmente). i pode simplesmente estar com uma percepção inacurada do seu ambiente e j pode ter lhe mostrado qual era a percepção correta. Porém, assim como seres humanos podem deixar de confiar em outros por lhe dizerem fatos que vão de encontro com as suas crenças, mesmo que incorretas, o mesmo pode acontecer na interação entre agentes. Ainda, tem-se que um agente pode não deixar de confiar rm outro agente que disse algo que não está correto vindo de outro agente por isso não contradizer as suas crenças, algo que também acontece com seres humanos.

Assim como a função booleana $concis()$ do *loop* de controle dos agentes, a detecção de uma contradição é semi-indecidível no caso geral. E, no caso das lógicas cujos processos de decisão são decidíveis, tem-se uma alta complexidade computacional. Portanto, novamente vê-se a necessidade de se buscar uma lógica para representar os estados mentais do agente que seja decidível e que torne o processo o mais eficiente possível, para que essa abordagem de agentes seja factível.

Se a confiança for binária parametrizada, o agente i deixará de confiar no agente j quanto às capacidades de φ:

$$\begin{aligned}\forall i \forall j \forall c((Bel\, i\, (said\, j\, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi) \\ \wedge c \in capacidades(\varphi) \rightarrow \neg TrustPB(i,j,c))\end{aligned}$$

Um agente i também não vai confiar em um agente j caso esse último não faça uma tarefa que lhe fora solicitada por i. Algo que pode ser assumido é que, quando se diz $(Happens\, \alpha)$, a ação α foi realizada da maneira adequada. Logo, se o agente que ficou incubido de realizá-la não a fizer direito, pode-se dizer que ele realizou uma ação $\alpha' \neq \alpha$.

Nas fórmulas a seguir, se o agente j nunca fizer o que lhe fora solicitado, o agente i não vai confiar nele em um estado futuro. É importante ressaltar que essa relação é muito forte ao dizer que o agente i só vai deixar de confiar no agente j se o mesmo nunca fizer a tarefa solicitada. Não há nada explícito na lógica para expressar um prazo para que a tarefa seja cumprida. Afinal, não é razoável que o agente i espere indefinidamente para que a sua solicitação seja cumprida. Claro que a ação pode incluir um prazo e fazê-la fora do prazo pode significar realizar uma tarefa diferente.

$$\forall i \forall j((Bel\, i\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \rightarrow A\Diamond\neg TrustB(i, j))$$

$$\forall i \forall j \forall c((Bel\, i\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge c \in capacidades(\alpha) \rightarrow A\Diamond\neg TrustPB(i, j, c))$$

Já se o agente j executar uma ação α' ao invés de α, tem-se as seguintes sentenças, onde o predicado $inadequada()$ verifica que uma ação α' é uma ação α executada de maneira não-adequada:

$$\forall i \forall j((Bel\, i\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge \exists\Diamond(Happens\, \alpha') \wedge inadequada(\alpha', \alpha) \rightarrow A\Diamond\neg TrustB(i, j))$$

$$\forall i \forall j \forall c((Bel\, i\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge c \in capacidades(\alpha) \wedge \exists\Diamond(Happens\, \alpha') \wedge inadequada(\alpha', \alpha) \rightarrow A\Diamond\neg TrustPB(i, j, c))$$

É importante dizer que um agente i, assim como armazena mensagens em suas crenças, ele também armazena as solicitações que seus parceiros lhe fazem e as solicitações que ele faz aos outros:

$$(Happens\, do(j, \alpha)) \rightarrow \bigcirc(Bel\, i\, do(j, \alpha))$$

$$(Happens\, do(i, \alpha)) \rightarrow \bigcirc(Bel\, i\, do(i, \alpha))$$

Pode-se também relacionar a confiança com as ações que aconteceram, ou seja, com uma ação de um agente j dizer algo e com a solicitação de uma tarefa a j:

$$\forall i \forall j((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi) \rightarrow \bigcirc\neg TrustB(i, j))$$

$$\forall i \forall j((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge \exists\Diamond(Happens\, \alpha') \wedge inadequada(\alpha', \alpha) \rightarrow A\Diamond\neg TrustB(i, j))$$

$$\forall i \forall j \forall c((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi) \\ \wedge c \in capacidades(\varphi) \rightarrow \bigcirc \neg TrustPB(i, j, c))$$

$$\forall i \forall j \forall c((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge c \in capacidades(\alpha) \\ \wedge \exists\Diamond(Happens\, \alpha') \wedge inadequada(\alpha', \alpha) \rightarrow A\Diamond\neg TrustPB(i, j, c))$$

Faz-se o mesmo agora para o caso onde há graus de confiança. Note-se que sempre o caso onde o grau de confiança é um valor dentro de um intervalo é bastante similar ao caso onde se tem níveis fixos de confiança. Isso já foi mostrado acima ao se colocar as sentenças para a relação da confiança em si mesmo e a confiança em outro agente.

Quando o agente j diz algo que é contraditório com o que i acredita, i diminui seu grau de confiança em j se ela não for a menor possível (em geral, 0, mas pode ser também -1 dependendo da implementação). Já se o agente j disser algo que não é contraditório com as crenças de i, o último aumentará seu grau de confiança no primeiro se esse não for o maior possível (em geral, 1). Nas sentenças abaixo, MIN e MAX são constantes lógicas.

$$\forall i \forall j((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi) \\ \wedge TrustV(i, j, v_1) \wedge\ v_2 < v_1 \wedge v_1 > MIN \rightarrow \bigcirc TrustV(i, j, v_2))$$

$$\forall i \forall j((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi) \\ \wedge TrustF(i, j, n, v_1) \wedge\ v_2 < v_1 \wedge v_1 > MIN \rightarrow \bigcirc TrustF(i, j, n, v_2))$$

$$\forall i \forall j \forall c((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi) \\ \wedge TrustPV(i, j, c, v_1) \wedge\ v_2 < v_1 \wedge v_1 > MIN \wedge c \in capacidades(\varphi) \\ \rightarrow \bigcirc TrustPV(i, j, c, v_2))$$

$$\forall i \forall j \forall c((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi) \\ \wedge TrustPF(i, j, c, n, v_1) \wedge\ v_2 < v_1 \wedge v_1 > MIN \wedge c \in capacidades(\varphi) \\ \rightarrow \bigcirc TrustPF(i, j, c, n, v_2))$$

$$\forall i \forall j((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \\ \wedge \neg(\Gamma \wedge \varphi \rightarrow \neg\psi) \wedge TrustV(i, j, v_1) \wedge\ v_1 < v_2 \wedge v_1 < MAX \\ \rightarrow \bigcirc TrustV(i, j, v_2))$$

$$\forall i \forall j((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \\ \wedge \neg(\Gamma \wedge \varphi \rightarrow \neg\psi) \wedge TrustF(i, j, n, v_1) \wedge\ v_1 < v_2 \wedge v_1 < MAX \\ \rightarrow \bigcirc TrustF(i, j, n, v_2))$$

$$\forall i \forall j \forall c((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge \neg(\Gamma \wedge \varphi \rightarrow \neg\psi) \\ \wedge TrustPV(i, j, c, v_1) \wedge\ v_1 < v_2 \wedge v_1 < MAX \wedge c \in capacidades(\varphi) \\ \rightarrow \bigcirc TrustPV(i, j, c, v_2))$$

$$\forall i \forall j \forall c((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge \neg(\Gamma \wedge \varphi \rightarrow \neg\psi) \\ \wedge TrustPF(i, j, c, n, v_1) \wedge\ v_1 < v_2 \wedge v_1 < MAX \wedge c \in capacidades(\varphi) \\ \rightarrow \bigcirc TrustPF(i, j, c, n, v_2))$$

As fórmulas a seguir tratam da situação onde o agente i delega uma tarefa ao agente j. No primeiro caso, o agente j não faz o que lhe foi pedido, ao contrário do que acontece no segundo caso. Novamente, são propriedades muito fortes, pois no primeiro caso o agente só diminui o grau de confiança se o agente j nunca fizer a tarefa solicitada. Já no segundo, o grau de confiança aumentará, mesmo que j leve muito mais tempo do que o razoável para fazer o que lhe foi pedido:

$$\forall i \forall j((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge TrustV(i, j, v_1) \\ \wedge v_2 < v_1 \wedge v_1 > MIN \rightarrow A\Diamond TrustV(i, j, v_2))$$

$$\forall i \forall j((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge TrustF(i, j, n, v_1) \\ \wedge v_2 < v_1 \wedge v_1 > MIN \rightarrow A\Diamond TrustF(i, j, n, v_2))$$

$$\forall i \forall j \forall c((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge TrustPV(i, j, c, v_1) \\ \wedge v_2 < v_1 \wedge v_1 > MIN \wedge c \in capacidades(\alpha) \rightarrow A\Diamond TrustPV(i, j, c, v_2))$$

$$\forall i \forall j \forall c((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge TrustPF(i, j, c, n, v_1) \\ \wedge v_2 < v_1 \wedge v_1 > MIN \wedge c \in capacidades(\alpha) \rightarrow A\Diamond TrustPF(i, j, c, n, v_2))$$

$$\forall i \forall j((Happens\, do(j, \alpha)) \wedge A\Diamond(Happens\, \alpha) \wedge TrustV(i, j, v_1) \\ \wedge v_1 < v_2 \wedge v_1 < MAX \rightarrow A\Diamond TrustV(i, j, v_2))$$

$$\forall i \forall j((Happens\, do(j, \alpha)) \wedge A\Diamond(Happens\, \alpha) \wedge TrustF(i, j, n, v_1) \\ \wedge v_1 < v_2 \wedge v_1 < MAX \rightarrow A\Diamond TrustF(i, j, n, v_2))$$

$$\forall i \forall j \forall c((Happens\, do(j, \alpha)) \wedge A\Diamond(Happens\, \alpha) \wedge TrustPV(i, j, c, v_1) \\ \wedge v_1 < v_2 \wedge v_1 < MAX \wedge c \in capacidades(\alpha) \rightarrow A\Diamond TrustPV(i, j, c, v_2))$$

$$\forall i \forall j \forall c((Happens\, do(j, \alpha)) \wedge A\Diamond(Happens\, \alpha) \wedge TrustPF(i, j, c, n, v_1) \\ \wedge v_1 < v_2 \wedge v_1 < MAX \wedge c \in capacidades(\alpha) \rightarrow A\Diamond TrustPF(i, j, c, n, v_2))$$

Já se j realizar a ação de maneira inadequada, tem-se que:

$$\forall i \forall j((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge \exists\Diamond(Happens\, \alpha') \\ \wedge inadequada(\alpha', \alpha) \wedge TrustV(i, j, v_1) \wedge\ v_2 < v_1 \wedge v_1 > MIN \\ \rightarrow A\Diamond TrustV(i, j, v_2))$$

$$\forall i \forall j((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge \exists\Diamond(Happens\, \alpha') \\ \wedge inadequada(\alpha', \alpha) \wedge TrustF(i, j, n, v_1) \wedge\ v_2 < v_1 \wedge v_1 > MIN \\ \rightarrow A\Diamond TrustF(i, j, n, v_2))$$

$$\forall i \forall j \forall c((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge \exists\Diamond(Happens\, \alpha') \\ \wedge inadequada(\alpha', \alpha) \wedge TrustPV(i, j, c, v_1) \wedge\ v_2 < v_1 \wedge v_1 > MIN \\ \wedge c \in capacidades(\alpha) \rightarrow A\Diamond TrustPV(i, j, c, v_2))$$

$$\forall i \forall j \forall c((Happens\, do(j, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge \exists\Diamond(Happens\, \alpha') \\ \wedge inadequada(\alpha', \alpha) \wedge TrustPF(i, j, c, n, v_1) \wedge\ v_2 < v_1 \wedge v_1 > MIN \\ \wedge c \in capacidades(\alpha) \rightarrow A\Diamond TrustPF(i, j, c, n, v_2))$$

No caso da confiança binária, diz-se quando o agente i não deve confiar no agente j, mas nada foi dito de quando ele deve confiar em seu parceiro. Portanto, tem-se como propriedade desejável que eventualmente, em algum ramo de execução do sistema, o agente i vai confiar no agente j. Logo:

$$\forall i \forall j(\neg TrustB(i, j) \rightarrow E\Diamond TrustB(i, j))$$

$$\forall i \forall j \forall c(\neg TrustPB(i, j, c) \rightarrow E\Diamond TrustPB(i, j, c))$$

Contudo, essas não são propriedades que necessariamente devem estar presentes. Afinal, pode ser o caso de o agente j sempre agir de modo a ser considerado não confiável. Nesses casos, tem-se que:

$$\forall i \forall j(A\Box((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \Gamma) \wedge (Bel\, i\, \psi) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi)) \\ \rightarrow \neg E\Diamond TrustB(i, j))$$

$$\forall i \forall j \exists c(A\Box((Happens\, says(j, \varphi)) \wedge (Bel\, i\, \Gamma) \wedge (Bel\, i\, \psi) \wedge (\Gamma \wedge \varphi \rightarrow \neg\psi) \\ \wedge c \in capacidades(\varphi)) \rightarrow \neg E\Diamond TrustPB(i, j, c))$$

$$\forall i \forall j(A\Box((Happens\, do(j, \alpha)) \wedge \neg E\Diamond(Happens\, \alpha)) \rightarrow \neg E\Diamond TrustB(i, j))$$

$$\forall i \forall j \forall c(A\Box((Happens\, do(j, \alpha)) \wedge \neg E\Diamond(Happens\, \alpha) \wedge c \in capacidades(\alpha)) \\ \rightarrow \neg E\Diamond TrustB(i, j))$$

$$\forall i \forall j(A\Box((Happens\, do(j, \alpha)) \wedge \neg E\Diamond(Happens\, \alpha) \wedge E\Diamond(Happens\, \alpha') \\ \wedge inadequada(\alpha, \alpha')) \rightarrow \neg E\Diamond TrustB(i, j))$$

$$\forall i \forall j \forall c(A\Box((Happens\, do(j, \alpha)) \wedge \neg E\Diamond(Happens\, \alpha) \\ \wedge c \in capacidades(\alpha) \wedge E\Diamond(Happens\, \alpha') \wedge inadequada(\alpha, \alpha')) \\ \rightarrow \neg E\Diamond TrustB(i, j))$$

Também pode-se dizer que um agente que sempre envia mensagens coerentes e que faz o que lhe é pedido é eventualmente confiável em todo o ramo de execução do sistema. Não se diz que ele será necessariamente confiável porque i pode querer ficar "testando" para ver se j se comporta como ele deseja durante um determinado período de tempo até concluir que o seu parceiro é mesmo de confiança.

$$\forall i \forall j (A\Box((Happens\ says(j,\varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge \neg(\Gamma \wedge \varphi \rightarrow \neg\psi)) \rightarrow A\Diamond TrustB(i,j))$$

$$\forall i \forall j \forall c (A\Box((Happens\ says(j,\varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma) \wedge \neg(\Gamma \wedge \varphi \rightarrow \neg\psi) \wedge c \in capacidades(\varphi)) \rightarrow A\Diamond TrustPB(i,j,c))$$

$$\forall i \forall j (A\Box((Happens\ do(j,\alpha)) \rightarrow A\Diamond(Happens\, \alpha)) \rightarrow A\Diamond TrustB(i,j))$$

$$\forall i \forall j \forall c (A\Box((Happens\ do(j,\alpha)) \rightarrow A\Diamond(Happens\, \alpha)) \wedge c \in capacidades(\alpha)) \rightarrow A\Diamond TrustPB(i,j,c))$$

Confiança em Novos Agentes

Deve-se também dizer como será determinada a confiança em um agente que acabou de entrar no sistema. No caso do modelo binário, quem implementa o sistema pode escolher entre não confiar ou confiar no agente ou então confiar ou não nas capacidades do mesmo. No caso das capacidades, não faz sentido o agente i começar confiando em determinadas capacidades e em outras não. Afinal, sobre um novo agente no sistema nada se sabe. Por exemplo, não é possível se dizer se ele vai ser honesto ou se vai fazer o que for pedido no prazo.

$$\forall i \forall j ((Happens\ in(j)) \rightarrow \bigcirc TrustB(i,j)) \text{ ou}$$
$$\forall i \forall j ((Happens\ in(j)) \rightarrow \bigcirc \neg TrustB(i,j))$$

$$\forall i \forall j \forall c ((Happens\ in(j)) \rightarrow \bigcirc TrustPB(i,j,c)) \text{ ou}$$
$$\forall i \forall j \forall c ((Happens\ in(j)) \rightarrow \bigcirc \neg TrustPB(i,j,c))$$

Já quando um novo agente entra em um sistema onde o grau de confiança possui mais de dois valores, o valor inicial pode ser um valor arbitrário entre o menor e o maior valor possível. Quando for possível se avaliar cada capacidade individualmente, também coloca-se o mesmo valor para cada uma. Afinal, se nada se conhece do novo agente, não se pode arbitrar valores diferentes para as suas capacidades.

$$\forall i \forall j ((Happens\ in(j)) \rightarrow \bigcirc TrustF(i,j,n,v) \wedge MIN \leq v \leq MAX)$$

$$\forall i \forall j ((Happens\ in(j)) \rightarrow \bigcirc TrustV(i,j,v) \wedge MIN \leq v \leq MAX)$$

$$\forall i \forall j \forall c ((Happens\ in(j)) \rightarrow \bigcirc TrustPF(i,j,c,n,v) \wedge MIN \leq v \leq MAX)$$

$$\forall i \forall j \forall c ((Happens\ in(j)) \rightarrow \bigcirc TrustPV(i,j,c,v) \wedge MIN \leq v \leq MAX)$$

Interação com Agentes Confiáveis

Um agente só vai desejar interagir com outro se ele confiar no seu parceiro ou se o grau de confiança for maior do que um mínimo desejado. Porém há casos em que ele pode interagir mesmo que não confie se não tiver outra alternativa. Para as sentenças a seguir, esses casos são descartados. No caso de confiança parametrizada em capacidades, deve-se ter um grau mínimo para cada capacidade. É razoável que esses graus mínimos sejam diferentes para cada capacidade. Por exemplo, faz sentido que a honestidade tenha um grau mínimo mais alto se valores financeiros estiverem envolvidos. Já se a interação for em um sistema onde o tempo é um recurso crítico, é desejável que o agente com quem se vai interagir seja eficiente.

$$\forall i \forall j (TrustB(i,j) \rightarrow E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j \forall c (TrustPB(i,j,c) \wedge c \in capacidades(\varphi) \rightarrow E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j (TrustV(i,j,v) \wedge v > VAL \rightarrow E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j \forall c (TrustPV(i,j,c,v) \wedge v > val(c) \wedge c \in capacidades(\varphi) \rightarrow E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j (TrustF(i,j,n,v) \wedge v > VAL \rightarrow E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j \forall c (TrustPF(i,j,c,n,v) \wedge v > val(c) \wedge c \in capacidades(\varphi) \rightarrow E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j (TrustB(i,j) \rightarrow E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j \forall c (TrustPB(i,j,c) \wedge c \in capacidades(\alpha) \rightarrow E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j (TrustV(i,j,v) \wedge v > VAL \wedge c \in capacidades(\alpha) \rightarrow E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j \forall c (TrustPV(i,j,c,v) \wedge v > val(c) \wedge c \in capacidades(\alpha) \rightarrow E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j (TrustF(i,j,n,v) \wedge v > VAL \rightarrow E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j \forall c (TrustPF(i,j,c,n,v) \wedge v > val(c) \wedge c \in capacidades(\alpha) \rightarrow E\Diamond(Happens\ do(j,\alpha)))$$

Ainda há o caso contrário, quando um agente não deseja interagir com um outro se não confiar no mesmo ou se o grau de confiança for menor do que um determinado valor mínimo. No caso parametrizado, ele vai deixar de

interagir com um parceiro se ele confiar menos do que um mínimo para alguma capacidade.

$$\forall i \forall j(\neg TrustB(i,j) \rightarrow \neg E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j \exists c(\neg TrustPB(i,j,c) \wedge c \in capacidades(\varphi) \rightarrow \neg E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j(TrustV(i,j,v) \wedge v < VAL \rightarrow \neg E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j \exists c(TrustPV(i,j,c,v) \wedge v < val(c) \wedge c \in capacidades(\varphi) \rightarrow \neg E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j(TrustF(i,j,n,v) \wedge v < VAL \rightarrow \neg E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j \exists c(TrustPF(i,j,c,n,v) \wedge v < val(c) \wedge c \in capacidades(\varphi) \rightarrow \neg E\Diamond(Happens\ says(i,\varphi)))$$

$$\forall i \forall j(\neg TrustB(i,j) \rightarrow \neg E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j \exists c(\neg TrustPB(i,j,c) \wedge c \in capacidades(\alpha) \rightarrow \neg E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j(TrustV(i,j,v) \wedge v < VAL \rightarrow \neg E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j \exists c(TrustPV(i,j,c,v) \wedge v < val(c) \wedge c \in capacidades(\alpha) \rightarrow \neg E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j(TrustF(i,j,n,v) \wedge v < VAL \rightarrow \neg E\Diamond(Happens\ do(j,\alpha)))$$

$$\forall i \forall j \exists c(TrustPF(i,j,c,n,v) \wedge v < val(c) \wedge c \in capacidades(\alpha) \rightarrow \neg E\Diamond(Happens\ do(j,\alpha)))$$

Atualização das Crenças com o Conteúdo das Mensagens

Quando um agente recebe uma mensagem vinda de outro, ele sempre atualiza suas crenças com o fato de que seu parceiro disse algo. Já para atualizar com o seu conteúdo, é necessário que o grau de confiança de i em j seja maior do que determinado valor, embora ele esteja correndo o risco de atualizar suas crenças com uma informação errônea. Cada par de sentenças abaixo é equivalente, sendo que a primeira está escrita em termos de uma crença do agente i de algo que já aconteceu, de que j disse algo. Já a segunda sentença do par tem o mesmo significado da primeira, só que é definida em função da ação de envio de mensagem.

$$\forall i \forall j((Bel\, i\, (said\, j\, \varphi)) \wedge TrustB(i,j) \rightarrow (Bel\, i\, \varphi))$$

$$\forall i \forall j((Happens\, says(j,\varphi)) \wedge TrustB(i,j) \rightarrow \bigcirc(Bel\, i\, \varphi))$$

$$\forall i \forall j \forall c((Bel\, i\, (said\, j\, \varphi)) \wedge TrustPB(i,j,c) \wedge c \in capacidades(\varphi) \rightarrow (Bel\, i\, \varphi))$$

$$\forall i \forall j \forall c((Happens\, says(j,\varphi)) \wedge TrustPB(i,j,c) \wedge c \in capacidades(\varphi) \rightarrow \bigcirc(Bel\, i\, \varphi))$$

$$\forall i \forall j((Bel\, i\, (said\, j\, \varphi)) \wedge TrustV(i,j,v) \wedge v > VAL \rightarrow (Bel\, i\, \varphi))$$

$$\forall i \forall j((Happens\, says(j,\varphi)) \wedge TrustV(i,j,v) \wedge v > VAL \rightarrow \bigcirc(Bel\, i\, \varphi))$$

$$\forall i \forall j \forall c((Bel\, i\, (said\, j\, \varphi)) \wedge TrustPV(i,j,c,v) \wedge c \in capacidades(\varphi) \wedge v > val(c) \rightarrow (Bel\, i\, \varphi))$$

$$\forall i \forall j \forall c((Happens\, says(j,\varphi)) \wedge TrustPV(i,j,c,v) \wedge c \in capacidades(\varphi) \wedge v > val(c) \rightarrow \bigcirc(Bel\, i\, \varphi))$$

$$\forall i \forall j((Bel\, i\, (said\, j\, \varphi)) \wedge TrustF(i,j,n,v) \wedge v > VAL \rightarrow (Bel\, i\, \varphi))$$

$$\forall i \forall j((Happens\, says(j,\varphi)) \wedge TrustF(i,j,n,v) \wedge v > VAL \rightarrow \bigcirc(Bel\, i\, \varphi))$$

$$\forall i \forall j \forall c((Bel\, i\, (said\, j\, \varphi)) \wedge TrustPF(i,j,c,n,v) \wedge c \in capacidades(\varphi) \wedge v > val(c) \rightarrow (Bel\, i\, \varphi))$$

$$\forall i \forall j \forall c((Happens\, says(j,\varphi)) \wedge TrustPF(i,j,c,n,v) \wedge c \in capacidades(\varphi) \wedge v > val(c) \rightarrow \bigcirc(Bel\, i\, \varphi))$$

As sentenças acima podem parecer contraditórias com as sentenças que definem a atualização do modelo de confiança quando um agente diz algo. Naquele caso, se o agente j enviar uma informação contraditória com relação às crenças de i, i vai confiar menos ou simplesmente deixar de confiar em j, não importando se antes j era um agente muito confiável ou simplesmente confiável. Isso porque um agente que era confiável pode passar a se comportar maliciosamente ou dizer informações incorretas por ter crenças inadequadas ou não ter capacidades necessárias para falar sobre determinado assunto, por exemplo. Nesses casos, é bastante razoável que i deixe de confiar ou confie menos em j. Já nas sentenças acima, i vai atualizar as suas crenças com o conteúdo da mensagem levando em conta apenas a confiança em seu parceiro, mesmo que o conteúdo da mensagem recebida seja contraditório com as crenças do agente. Afinal, i pode estar desatualizado ou ter uma visão inadequada do ambiente e, se um parceiro confiável lhe avisar disso, ele pode corrigir as suas crenças. Ambas as abordagens são úteis dependendo da aplicação e da situação em que ocorrem.

Informações sobre o Comportamento de Outros Agentes

Por fim, um agente i pode pedir informações a um agente j com relação à confiança que ele tem em um terceiro agente k para saber como esse se comporta. Para que o agente i acredite no que j disse com relação a k, ele precisa ter um grau de confiança em j maior do que um determinado valor. Nesse caso, quanto à confiança parametrizada, só é necessário levar em conta a honestidade de j, chamada de h. Nas sentenças abaixo, nos casos não-parametrizados, considera-se que o agente i não conhece o agente k, onde o fato de um agente conhecer o outro é definido no predicado binário $conhece()$. Já quando a confiança for dividida em capacidades, tem-se que um agente i pode conhecer a capacidade c_1 de k, mas não conhecer a c_2, por exemplo. Então, se i não conhecer uma capacidade c de k, ele pode questionar um parceiro j a esse respeito. Nessa situação, $conhece()$ é ternário, onde o terceiro parâmetro é a capacidade conhecida. Note-se que, quando um agente conhece todas as capacidades de um outro agente, semanticamente tem-se a seguinte equivalência:

$$\forall i \forall j \forall c(conhece(i,j) \leftrightarrow conhece(i,j,c))$$

A seguir, supõe-se que o mesmo modelo de confiança seja usado em todo o sistema. É importante enfatizar que o fato de um agente não conhecer um outro ou a capacidade de um outro não quer dizer que ele não tenha um valor padrão de confiança, algo que acontece quando um novo agente entra no sistema. Considera-se que ele vai passar a conhecer o outro agente ou alguma de suas capacidades após se informar sobre o mesmo.

$$\forall i \forall j \forall k((Happens\, says(j, TrustB(j,k)) \wedge TrustB(i,j) \wedge \neg conhece(i,k) \rightarrow \bigcirc TrustB(i,k) \wedge conhece(i,k))$$

$$\forall i \forall j \forall k((Happens\, says(j, \neg TrustB(j,k)) \wedge TrustB(i,j) \wedge \neg conhece(i,k) \rightarrow \bigcirc \neg TrustB(i,k) \wedge conhece(i,k))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPB(j,k,c)) \wedge TrustPB(i,j,h) \wedge \neg conhece(i,k,c) \rightarrow \bigcirc TrustPB(i,k,c) \wedge conhece(i,k,c))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, \neg TrustPB(j,k,c)) \wedge TrustPB(i,j,h) \wedge \neg conhece(i,k,c)) \rightarrow \bigcirc \neg TrustPB(i,k,c) \wedge conhece(i,k))$$

$$\forall i \forall j \forall k((Happens\, says(j, TrustV(j,k,v_1)) \wedge TrustV(i,j,v_2) \wedge v_2 > VAL \wedge \neg conhece(i,k) \rightarrow \bigcirc TrustV(i,k,v_1) \wedge conhece(i,k))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPV(j,k,c,v_1)) \wedge TrustPV(i,j,h,v_2) \wedge v_2 > VAL \wedge \neg conhece(i,k,c) \rightarrow \bigcirc TrustPV(i,k,c,v_1) \wedge conhece(i,k,c))$$

$$\forall i \forall j \forall k((Happens\, says(j, TrustF(j,k,n,v_1)) \wedge TrustF(i,j,n,v_2) \wedge v_2 > VAL \\ \wedge \neg conhece(i,k) \rightarrow \bigcirc TrustF(i,k,n,v_1) \wedge conhece(i,k))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPF(j,k,c,n,v_1)) \wedge TrustPF(i,j,h,n,v_2) \\ \wedge \neg conhece(i,k,c) \wedge v_2 > VAL \rightarrow \bigcirc TrustPF(i,k,c,,n,v_1) \wedge conhece(i,k,c))$$

Já se o agente i conhecer o agente k, ele já terá um valor de confiança para o último. Logo, ao interagir com j, ele quer atualizar seu valor de confiança em k ou em relação a uma capacidade de k. Assim, ele terá que fazer algum cálculo a partir dos valores antigo e novo. Nos casos não-binários, essa conta pode ser uma simples média aritmética ou dar um maior peso ao valor mais recente. Nesse caso, tal como em (Gue06) o comportamento mais recente tem maior peso na reputação de um agente. Abaixo, *calculo*() é um predicado que tem como os dois primeiros parâmetros os valores de confiança que ele tinha e o novo, respectivamente e o novo nível de confiança de i em k. No caso binário, i pode escolher entre um valor e outro, respectivamente.

$$\forall i \forall j \forall k((Happens\, says(j, TrustB(j,k)) \wedge TrustB(i,k) \wedge TrustB(i,j) \\ \wedge conhece(i,k) \rightarrow \bigcirc TrustB(i,k))$$

$$\forall i \forall j \forall k((Happens\, says(j, \neg TrustB(j,k)) \wedge \neg TrustB(i,k) \wedge TrustB(i,j) \\ \wedge conhece(i,k) \rightarrow \bigcirc \neg TrustB(i,k))$$

$$\forall i \forall j \forall k((Happens\, says(j, TrustB(j,k)) \wedge \neg TrustB(i,k) \wedge TrustB(i,j) \\ \wedge calculo(0,1,v) \wedge conhece(i,k) \\ \rightarrow (v = 0 \rightarrow \bigcirc \neg TrustB(i,k)) \wedge (v = 1 \rightarrow \bigcirc TrustB(i,k)))$$

$$\forall i \forall j \forall k((Happens\, says(j, \neg TrustB(j,k)) \wedge TrustB(i,k) \wedge TrustB(i,j) \\ \wedge calculo(1,0,v) \wedge conhece(i,k) \\ \rightarrow (v = 0 \rightarrow \bigcirc \neg TrustB(i,k)) \wedge (v = 1 \rightarrow \bigcirc TrustB(i,k)))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPB(j,k,c)) \wedge TrustPB(i,k,c) \\ \wedge TrustPB(i,j,h) \wedge conhece(i,k,c) \rightarrow \bigcirc TrustPB(i,k,c))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, \neg TrustPB(j,k,c)) \wedge \neg TrustPB(i,k,c) \\ \wedge TrustPB(i,j,h) \wedge conhece(i,k,c) \rightarrow \bigcirc \neg TrustPB(i,k,c))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPB(j,k,c)) \wedge \neg TrustPB(i,k,c) \\ \wedge TrustPB(i,j,h) \wedge calculo(0,1,v) \wedge conhece(i,k,c) \\ \rightarrow (v = 0 \rightarrow \bigcirc \neg TrustPB(i,k,c)) \wedge (v = 1 \rightarrow \bigcirc TrustPB(i,k,c)))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, \neg TrustPB(j,k,c)) \wedge TrustPB(i,k,c) \\ \wedge TrustPB(i,j,h) \wedge calculo(1,0,v) \wedge conhece(i,k,c) \\ \rightarrow (v = 0 \rightarrow \bigcirc \neg TrustPB(i,k,c)) \wedge (v = 1 \rightarrow \bigcirc TrustPB(i,k,c)))$$

$$\forall i \forall j \forall k((Happens\, says(j, TrustV(j,k,v_1)) \wedge TrustV(i,j,v_2) \wedge v_2 > VAL \\ \wedge TrustV(i,j,v_3) \wedge calculo(v_3, v_1, v) \wedge conhece(i,k) \rightarrow \bigcirc TrustV(i,k,v))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPV(j,k,c,v_1)) \wedge TrustPV(i,j,h,v_2) \\ \wedge v_2 > VAL \wedge TrustPV(i,j,c,v_3) \wedge conhece(i,k,c) \wedge calculo(v_3, v_1, v) \\ \rightarrow \bigcirc TrustPV(i,k,c,v))$$

$$\forall i \forall j \forall k((Happens\, says(j, TrustF(j,k,n,v_1)) \wedge TrustF(i,j,n,v_2) \wedge v_2 > VAL \\ \wedge TrustF(i,j,n,v_3) \wedge calculo(v_3, v_1, v) \wedge conhece(i,k) \rightarrow \bigcirc TrustF(i,k,n,v))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPF(j,k,c,n,v_1)) \wedge TrustPF(i,j,h,n,v_2) \\ \wedge v_2 > VAL \wedge TrustPF(i,j,c,n,v_3) \wedge calculo(v_3, v_1, v) \\ \wedge conhece(i,k,c) \rightarrow \bigcirc TrustPF(i,k,c,,n,v))$$

Nos dois grupos de sentenças acima, o agente i confiava no parceiro j que lhe passava as informações sobre k. Se ele não confiar em j, o grau de confiança em k não vai mudar, sendo o que ele já tinha.

$$\forall i \forall j \forall k((Happens\, says(j, TrustB(j,k)) \wedge TrustB(i,k) \wedge \neg TrustB(i,j) \\ \rightarrow \bigcirc TrustB(i,k))$$

$$\forall i \forall j \forall k((Happens\, says(j, \neg TrustB(j,k)) \wedge \neg TrustB(i,k) \wedge \neg TrustB(i,j) \\ \rightarrow \bigcirc \neg TrustB(i,k))$$

$$\forall i \forall j \forall k((Happens\, says(j, TrustB(j,k)) \wedge \neg TrustB(i,k) \wedge \neg TrustB(i,j) \\ \rightarrow \bigcirc \neg TrustB(i,k))$$

$$\forall i \forall j \forall k((Happens\, says(j, \neg TrustB(j,k)) \wedge TrustB(i,k) \wedge \neg TrustB(i,j) \\ \rightarrow \bigcirc TrustB(i,k))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPB(j,k,c)) \wedge TrustPB(i,k,c) \\ \wedge \neg TrustPB(i,j,h) \rightarrow \bigcirc TrustPB(i,k,c))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, \neg TrustPB(j,k,c)) \wedge \neg TrustPB(i,k,c) \\ \wedge \neg TrustPB(i,j,h) \rightarrow \bigcirc \neg TrustPB(i,k,c))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPB(j,k,c)) \wedge \neg TrustPB(i,k,c) \\ \wedge \neg TrustPB(i,j,h) \rightarrow \bigcirc \neg TrustPB(i,k,c))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, \neg TrustPB(j,k,c)) \wedge TrustPB(i,k,c) \\ \wedge \neg TrustPB(i,j,h) \rightarrow \bigcirc TrustPB(i,k,c))$$

$$\forall i \forall j \forall k((Happens\, says(j, TrustV(j,k,v_1)) \wedge TrustV(i,j,v_2) \wedge v_2 \leq VAL \\ \wedge TrustV(i,j,v_3) \rightarrow \bigcirc TrustV(i,k,v_3))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPV(j,k,c,v_1)) \wedge TrustPV(i,j,h,v_2) \\ \wedge v_2 \leq VAL \wedge TrustPV(i,j,c,v_3) \rightarrow \bigcirc TrustPV(i,k,c,v_3))$$

$$\forall i \forall j \forall k((Happens\, says(j, TrustF(j,k,n,v_1)) \wedge TrustF(i,j,n,v_2) \wedge v_2 \leq VAL \\ \wedge TrustF(i,j,n,v_3) \rightarrow \bigcirc TrustF(i,k,n,v_3))$$

$$\forall i \forall j \forall k \forall c((Happens\, says(j, TrustPF(j,k,c,n,v_1)) \wedge TrustPF(i,j,h,n,v_2) \\ \wedge v_2 \leq VAL \wedge TrustPF(i,j,c,n,v_3) \rightarrow \bigcirc TrustPF(i,k,c,,n,v_3))$$

5.8 Sucesso de Intenções e Planos

Nas figuras das várias versões dos *loops* de controle de um agente presentes no capítulo 4, aparecem os predicados *sucesso*(), *impossivel*() e *concis*(). Nesta seção eles são formalizados e é explicado como a confiança pode afetá-los. Todavia, a função *motivado*() não será formalizada porque motivação é uma propriedade muito difícil de ser escrita formalmente por ser muito subjetiva. Por exemplo, ele pode se desmotivar ao atingir uma intenção por considerá-la muito difícil de ser alcançada. Contudo, como se formalizar o que é entendido por difícil? Outra situação que pode acontecer é a de o agente encontrar uma nova intenção a ser alcançada mais interessante do que a atual para ele. Como formalizar tal propriedade considerando que uma intenção que é muito interessante para um agente pode ser totalmente irrelevante para outro?

sucesso()

Sem levar em conta a confiança, o sucesso de uma ou mais intenções pode ser definido como o fato de elas agora fazerem parte das crenças.

$$\forall i(sucesso(I,B) \equiv (Bel\, i\, B) \wedge (Int\, i\, I) \wedge (B \rightarrow I)$$

Ou seja, um conjunto de fórmulas (I) que o agente queria atingir se tornaram realidade, suas intenções foram um sucesso.

Na maioria dos casos a confiança não é necessária para saber se uma intenção foi atingida ou não, já que basta que o que se deseja alcançar na intenção seja verdade (e ele possa perceber isso de alguma forma) para que o agente a dê por atingida. Contudo, se o agente tiver como intenção testar a confiança em outro agente e tiver como condição de sucesso que ele pode confiar nesse agente, tem-se as seguintes sentenças para cada tipo de confiança:

$$\forall i \forall j(sucesso(I,B,T) \equiv \\ sucesso(I,B) \vee ((Int\, i\, I) \wedge (I \rightarrow TrustB(i,j)) \wedge TrusB(i,j)))$$

$$\forall i \forall j \forall c(sucesso(I, B, T)$$
$$\equiv sucesso(I, B) \vee ((Int\, i\, I) \wedge (I \rightarrow TrustPB(i, j, c)) \wedge TrustPB(i, j, c)))$$

$$\forall i \forall j(sucesso(I, B, T) \equiv sucesso(I, B)$$
$$\vee((Int\, i\, I) \wedge (I \rightarrow TrustV(i, j, v_2)) \wedge TrustV(i, j, v_1) \wedge v_1 \geq v_2))$$

$$\forall i \forall j \forall c(sucesso(I, B, T) \equiv sucesso(I, B)$$
$$\vee((Int\, i\, I) \wedge (I \rightarrow TrustPV(i, j, c, v_2)) \wedge TrustPV(i, j, c, v_1) \wedge v_1 \geq v_2))$$

$$\forall i \forall j(sucesso(I, B, T) \equiv sucesso(I, B)$$
$$\vee((Int\, i\, I) \wedge (I \rightarrow TrustF(i, j, n, v_2)) \wedge TrustF(i, j, n, v_1) \wedge v_1 \geq v_2))$$

$$\forall i \forall j \forall c(sucesso(I, B, T) \equiv sucesso(I, B)$$
$$\vee((Int\, i\, I) \wedge (I \rightarrow TrustPF(i, j, c, n, v_2)) \wedge TrustPF(i, j, c, n, v_1) \wedge v_1 \geq v_2))$$

impossivel()

Sem levar a noção de confiança em conta, pode-se dizer que uma ou mais intenções são impossíveis de serem alcançadas se elas no momento não já tiverem sido atingidas (não forem realidade) e, de acordo com as crenças atuais, nunca puderem ser atingidas:

$$\forall i(impossivel(I, B) \equiv (Bel\, i \neg I) \wedge (Int\, i\, I) \wedge (B \rightarrow A\Box\neg I))$$

Quando a confiança é inserida, uma ou mais intenções passarão a ser impossíveis se o agente precisar interagir com um outro (trocar mensagens ou delegar uma tarefa) para realizar uma ação e não encontrar um parceiro em quem ele possa confiar. A necessidade de interação é dada pelo predicado *interagir*(). Já $plano(\alpha, I)$ indica que a ação α está necessariamente presente em um plano para atingir as intenções I presentes no conjunto de intenções do agente. Ou seja, a ação α é imprescindível para atingir as intenções I.

$$\forall i(impossivel(I, B, T) \equiv impossivel(I, B)$$
$$\vee\exists\alpha(interagir(\alpha) \wedge (Int\, i\, I) \wedge plano(\alpha, I) \wedge \forall j \neg TrustB(i, j)))$$

$$\forall i(impossivel(I, B, T) \equiv impossivel(I, B) \vee \exists\alpha(interagir(\alpha) \wedge (Int\, i\, I)$$
$$\wedge plano(\alpha, I) \wedge \forall j \exists c(\neg TrustPB(i, j, c) \wedge c \in capacidades(\alpha))))$$

$$\forall i(impossivel(I, B, T) \equiv impossivel(I, B) \vee \exists\alpha(interagir(\alpha) \wedge (Int\, i\, I)$$
$$\wedge plano(\alpha, I) \wedge \forall j(TrustV(i, j, v) \wedge v < VAL)))$$

$$\forall i(impossivel(I, B, T) \equiv impossivel(I, B) \vee \exists\alpha(interagir(\alpha) \wedge (Int\, i\, I)$$
$$\wedge plano(\alpha, I) \wedge \forall j \exists c(TrustPV(i, j, c, v) \wedge v < val(c) \wedge c \in capacidades(\alpha))))$$

$$\forall i(impossivel(I,B,T) \equiv impossivel(I,B) \vee \exists\alpha(interagir(\alpha) \wedge (Int\, i\, I) \\ \wedge plano(\alpha,I) \wedge \forall j(TrustF(i,j,n,v) \wedge v < VAL)))$$

$$\forall i(impossivel(I,B,T) \equiv impossivel(I,B) \vee \exists\alpha(interagir(\alpha) \wedge (Int\, i\, I) \\ \wedge plano(\alpha,I) \wedge \forall j\exists c(TrustPF(i,j,n,c,v) \wedge v < val(c) \wedge c \in capacidades(\alpha))))$$

Para as sentenças acima, um exemplo prático seria se a ação α for o envio de informações bancárias para atingir a intenção de se realizar uma compra de determinado produto pela Internet. Se nenhum *site* de comércio eletrônico disponível que vende o produto for confiável, o agente não fará a compra supondo que ele seja racional. Ou seja, a intenção de comprar o produto pela Internet é impossível.

concis()

Sem levar a confiança em consideração, um plano será consistente com as crenças e as intenções atuais de um agente se a pré-condição do plano for consistente com as crenças e a pós-condição do plano for consistente com as intenções. Essa última parte é importante porque o agente pode reconsiderar as intenções e o plano que está sendo executado pode não ser compatível com o atual conjunto de intenções do agente. Logo, tem-se que:

$$\forall i(concis(\pi,I,B) \equiv (Bel\, i\, B) \wedge (Int\, i\, I) \wedge (B \rightarrow pre(\pi)) \wedge (I \rightarrow pos(\pi)))$$

Quando se insere a confiança, um plano será inconsistente se a ação a ser executada precisar de interação com um determinado agente e ele não for confiável o suficiente. Obviamente, quando um agente faz um plano, ele escolhe apenas agentes confiáveis para com eles interagir por ser racional. Contudo, enquanto o plano é executado, os agentes escolhidos podem se comportar inadequadamente e o nível de confiança que o agente que quer interagir com eles tem neles diminui. Nas sentenças abaixo, o predicado $plano(\alpha,\pi)$ indica que a ação α está no plano π. Já o predicado binário $interagir(\alpha,j)$ diz que a ação α precisa do agente j para ser concluída.

$$\forall i(concis(\pi,I,B,T) \\ \equiv concis(\pi,I,B) \wedge \forall\alpha\exists j(interagir(\alpha,j) \wedge plano(\alpha,\pi) \rightarrow TrustB(i,j)))$$

$$\forall i(concis(\pi,I,B,T) \equiv concis(\pi,I,B) \wedge \forall\alpha\exists j\forall c(interagir(\alpha,j) \\ \wedge plano(\alpha,\pi) \wedge c \in capacidades(\alpha) \rightarrow TrustPB(i,j,c)))$$

$$\forall i(concis(\pi,I,B,T) \equiv concis(\pi,I,B) \wedge \forall\alpha\exists j(interagir(\alpha,j) \wedge plano(\alpha,\pi) \\ \rightarrow TrustV(i,j,v) \wedge v > VAL))$$

$$\forall i(concis(\pi, I, B, T) \equiv concis(\pi, I, B) \wedge \forall \alpha \exists j \forall c(interagir(\alpha, j) \wedge plano(\alpha, \pi) \wedge c \in capacidades(\alpha) \rightarrow TrustPV(i, j, c, v) \wedge v > val(c)))$$

$$\forall i(concis(\pi, I, B, T) \equiv concis(\pi, I, B) \wedge \forall \alpha \exists j(interagir(\alpha, j) \wedge plano(\alpha, \pi) \rightarrow TrustF(i, j, n, v) \wedge v > VAL))$$

$$\forall i(concis(\pi, I, B, T) \equiv concis(\pi, I, B) \wedge \forall \alpha \exists j \forall c(interagir(\alpha, j) \wedge plano(\alpha, \pi) \wedge c \in capacidades(\alpha) \rightarrow TrustPF(i, j, c, n, v) \wedge v > val(c)))$$

5.9 Dois Exemplos Simples

Considera-se agora dois exemplos simples, mas práticos. Estes têm como finalidade mostrar a utilidade da lógica aqui definida.

Quanto ao primeiro, ao se fazer compras através da Amazom.com Marketplace, precisa-se avaliar o vendedor depois que o processo de compra e venda estiver supostamente concluído. Nesse caso, o usuário dá uma nota de 1 a 5 ao vendedor, de maneira global. Essa nota é usada para compor a reputação de um vendedor. A reputação final é a média aritmética de todas as notas dadas.

Como toda a nota é computada, pode-se considerar que a Amazon.com (considerada aqui como um agente) confia nos compradores quanto a capacidade de dar notas, chamada de n. Para a avaliação, considera-se que um agente comprador passa a confiar no agente vendedor no valor da nota depois da transação. Deve-se observar que os modelos de confiança usados são diferentes. Para se dar as notas, o modelo de confiança não é dividido em capacidades, já que a nota é global para o agente sendo avaliado. Já o comprador tem como modelo de confiança aquele que tem níveis fixos, mais precisamente 5. Já o do sistema é com um valor variável, já que o valor de reputação ou confiança final será a média aritmética. A seguir, os casos onde o vendedor está recebendo a sua primeira avaliação e uma n-ésima avaliação:

$$\forall i \forall j((Happens\, says(i, TrustF(i, j, 5, v)) \wedge TrustPB(A, i, n) \wedge \neg conhece(A, j) \rightarrow \bigcirc TrustV(A, j, v) \wedge conhece(A, j))$$

$$\forall i \forall j((Happens\, says(i, TrustF(i, j, 5, v_1)) \wedge TrustPB(A, i, n) \wedge TrustV(A, j, v_2) \wedge conhece(A, j) \wedge calculo(v_1, v_2, v_3) \rightarrow \bigcirc TrustV(A, j, v_3))$$

No segundo caso, tem-se um comprador i interessado em um determinado produto. Para simplificar, supõe-se que todos os vendedores o vendem pelo mesmo valor. Nesse caso, o comprador obviamente vai optar por aquele vendedor em que ele confia mais ou por um dos que ele confia no caso da confiança binária. Como na prática apenas têm-se a avaliação global de

um vendedor, usa-se apenas os casos de confiança global. Note-se que não necessariamente ele irá comprar o produto. Também, tem-se a ação *comprar*(), onde o agente i compra um produto do agente j.

$$\forall j(TrustB(i,j) \rightarrow E\Diamond(Happens\, comprar(i,j)))$$

$$\forall j \exists k(TrustV(i,j,v) \wedge TrustV(i,k,v_1) \wedge v_1 > v \rightarrow E\Diamond(Happens\, comprar(i,k)))$$

Também deve-se levar em conta que o comprador não vai comprar de um agente em quem não confia ou confia pouco:

$$\forall j(\neg TrustB(i,j) \rightarrow A\Box\neg(Happens\, comprar(i,j)))$$

$$\forall j(TrustV(i,j,v) \wedge v < MIN \rightarrow A\Box\neg(Happens\, comprar(i,k)))$$