LÓGICAS MODAIS E JOGOS

Carla Amor Divino Moreira Delgado
Mario R. F. Benevides
COPPE Sistemas, UFRJ

A lógica modal é considerada uma ferramenta adequada para o tratamento de sistemas compostos por agentes racionais no contexto de sistemas interativos; operadores modais para capturar transformações globais do sistema ([FL79], [PA85]) e dos processos de raciocínio individuais ([FA95], [VB01]) já foram propostos e sedimentados. Segundo [GB02], o aparato da lógica modal pode ser utilizado para modelar duas visões conceitualmente muito diferentes de teoria dos jogos: uma é a visão acerca do raciocínio de jogadores racionais, e a outra é a visão acerca das possibilidades de resultado que podem ser atingidas por um jogador. Lógicas de Jogos correspondentes a ambas as visões foram estudadas, e cada uma delas captura conceitos específicos de teoria dos jogos. Porém, o tratamento efetivo de sistemas interativos, mesmo dos mais elementares, pressupõe um apanhado de vários destes conceitos. Veremos neste relatório uma revisão dos principais conceitos e trabalhos acerca de lógicas de jogos, especialmente lógicas modais sobre modelos de teoria dos jogos.

RIO DE JANEIRO, RJ – BRASIL
JUNHO DE 2004

Sumário

Capítulo 1

Introdução

A área de computação é constituída por tecnologias novas, em fase de grande expansão, contínuas modificações e estágios de maturidade heterogêneos. A relação da sociedade com esta realidade é pró-ativa, apoiada na facilidade de acesso à informação, mas dificultada pela instabilidade e pelas contradições dos novos modos de conhecimento e regulação social. Esta antítese se reflete nos mecanismos de software que modelam ou suportam processos sociais.

O termo *social software*, aqui traduzido para software social, foi introduzido por Parikh em [PA??] para se referir à tarefa de construir e verificar procedimentos e processos sociais usando os métodos formais de ciência da computação. Parikh ressalta a importância e a grande disseminação atual de sistemas de computação com as características de software social, e observa que os cientistas da computação possuem ao seu dispor ferramentas adequadas para dar início ao seu tratamento formal, citando as Lógicas de Jogos (extensões de Lógica Dinâmica), e a Teoria dos Jogos no sentido de von-Neumann, Morgenstern e Nash.

O interesse nas relações entre lógica e teoria dos jogos tem sido crescente nas últimas décadas. Pesquisadores de lógica encontram na teoria dos jogos um modelo adequado para trabalhar vários conceitos. Os modelos de teoria de jogos, apesar de construídos sobre sólido arcabouço matemático, têm apelo social e uma alta dose de intuição.

Por outro lado, teóricos de jogos encontraram na lógica ferramentas formais poderosas para representar e explorar conceitos de teoria dos jogos. Como exemplo, podemos citar o uso de lógicas de conhecimento associada aos conceitos de informação

perfeita e imperfeita, e o uso de modalidades de coalisão associado a noções de poderes (eficácia em forçar o jogo a certos resultados) dos jogadores individualmente e em grupo.

A interação é a principal características dos processos sociais, e conseqüentemente dos softwares sociais também. Captar sua essência requer admitir a existência de unidades autônomas distintas entre si capazes de atuar sobre o mesmo ambiente com possíveis reflexos para ambas as partes, o que é uma premissa dos paradigmas de sistemas distribuídos e multi-agentes. As correlações entre as pesquisas das comunidades de sistemas multi-agentes e lógica são antigas, e se estreitam ainda mais com as possibilidades surgidas da aproximação com teoria dos jogos.

O impacto social de ambientes distribuídos abertos como a Internet tornam o apelo do paradigma multi-agentes ainda mais efetivo no contexto de software social. A negociação implícita ou explícita, uma constante nos processos sociais, acontece agora também nestes ambientes. Algoritmos e técnicas para possibilitar que agentes de software automáticos atuem nesses ambientes de forma efetiva e produtiva são produtos da pesquisa em sistemas multi-agentes.

Todavia, o estudo de uma computação distribuída é, em algum nível, observar como o individual afeta o coletivo, foco da teoria dos jogos! Fechamos assim o ciclo que envolve as três áreas - Teoria de Jogos, Lógica e Sistemas Multi-Agentes - em torno do tema software social.

A alusão ao emaranhado de contribuições advindo das três áreas citadas aponta no sentido de que as iniciativas de trabalho em torno de software social estejam de alguma forma inseridas neste contexto híbrido. A iniciativa deste trabalho faz jus a esta indicação; apoiaremo-nos no arcabouço de lógicas para sistemas multi-agentes para propor um sistema formal para modelagem e verificação de propriedades relevantes em sistemas de interação racional.

Este trabalho apresenta uma ampla revisão bibliográfica acerca de lógicas para falar sobre jogos. Algumas lógicas de jogos propostas nos últimos anos serão mostradas e discutidas, especialmente no que tange à sua contribuição para a especificação formal de processos interativos em sistemas multi-agentes. Ao final, é apresentada uma discussão sobre a viabilidade e relevância de se construir novas lógicas para

falar sobre jogos, tanto sob a perspectiva dos jogadores quanto sob uma perspectiva externa aos agentes participantes, que corresponde o ponto de vista dos resultados que podem ser obtidos.

No capítulo 2 são apresentados os conceitos fundamentais de teoria dos jogos. O capítulo 3 apresenta em linhas gerais as principais linguagens lógicas construídas sobre modelos de teoria dos jogos. O capítulo 4 apresenta em detalhes a Lógica de Jogos de [PA85], baseada nas ações e poderes dos jogadores. O capítulo 5 apresenta a lógica de [PE87b] para tratar comunicação em processos concorrentes. O capítulo 6 apresenta a Lógica de Coalisão de [MP02], que fala sobre os resultados que podem ser atingidos quando grupos de jogadores unem seus esforços. O capítulo 7 engloba as considerações feitas sobre uma análise do material dos capítulos anteriores, e discute as possibilidades de construção de outras lógicas de jogos.

Capítulo 2

Teoria de Jogos

Este capítulo almeja introduzir conceitos fundamentais de teoria dos jogos, mais especificamente jogos não cooperativos. O texto foi baseado em [MP99] e [OR94]. Uma ampla revisão dos conceitos apresentados e também das relações destes com outros conceitos de teoria dos jogos pode ser vista em [OR94].

O discurso da teoria dos jogos gira em torno da lógica dos processos de decisão entre jogadores racionais, que podem nutrir expectativas sobre a atuação dos outros jogadores. Como o resultado de um jogo é na verdade o resultado de um processo de interação coletivo, sua análise adentra as relações entre colaboração e racionalidade, considerando que não há julgamento moral, nem desfecho universal certo ou errado.

Segundo [GB02], teoria de jogos pode ser considerada como composta por dois módulos.

O primeiro módulo corresponde à descrição formal de situações interativas. A linguagem formulada para este propósito permite a construção de descrições em diferentes níveis de detalhe; desde a forma extensiva, mais detalhada, até a forma estratégica, mais condensada. A linguagem de teoria dos jogos provou ser útil em diversos campos como economia, ciências políticas e militares, biologia evolucionária, ciência da computação, psicologia experimental, sociologia e filosofia social.

O segundo módulo corresponde ao estabelecimento de uma coleção de "conceitos de solução". Os conceitos de solução são métricas para estabelecer, dado um ponto de vista, como atribuir vitória a um ou outro jogador. Cada conceito de solução associa com todo jogo em uma dada classe um resultado ou um conjunto de resultados.

Muito do debate em teoria dos jogos está centrado no segundo módulo, ou seja, em desenvolver conceitos de solução que permitam predizer algo sobre o resultado de um jogo e como os jogadores irão interagir. Em contrapartida, o advento do software social [PA??] e a conseqüente necessidade de construir e verificar processos sociais usando os métodos formais de ciência da computação, impulsionaram os estudos no primeiro módulo.

As abordagens de ambos os módulos são regidas pelo mesmo objetivo central: equacionar conflitos de interesse. No fenômeno que se observa quando pessoas que tomam decisões interagem, a tendência das pessoas, ou seja, dos jogadores, é maximizar o ganho individual mesmo quando agindo coletivamente, recaindo em um dilema entre pessoal e coletivo. Se todos se comportassem de forma altruísta, priorizando o coletivo em detrimento do individual, não haveria dilema, nem jogo.

2.1 Jogos Estratégicos

Jogos estratégicos são o modelo mais sucinto para interação de teoria dos jogos.

Definição 2.1.1 *(Jogo Estratégico)*
Um jogo estratégico é uma tupla $G = (N, (A_i)_{i \in N}, (\succeq_i)_{i \in N})$*, onde:*

- $N = \{1, \cdots, n\}$ *é um conjunto finito de jogadores*
- $(A_i)_{i \in N}$ *associa a cada jogador* i *um conjunto não vazio de ações ou estratégias* A_i
- $\succeq_i$ *é a relação de preferência do jogador* i*: Seja* $A = A_1 \times \cdots A_n$ *o conjunto de configurações de ações, então para todo jogador* $i \in N$*,* $\succeq_i$ *é uma relação transitiva completa em* A*.*

Dizemos que G é finito se e somente se A é finito. Escrevemos $a \sim_i b$ como uma abreviação para $a \succeq_i b$ e $b \succeq_i a$, e $a \succ_i b$ abrevia $a \succeq_i b$ e $b \not\succeq_i a$.

Existem duas formulações alternativas para jogos estratégicos:

Definição 2.1.2 *(Jogo Estratégico baseado em conseqüências de ações)*

$G = (N, (A_i)_{i \in N}, C, g, (\succeq_i)_{i \in N})$, onde C é um conjunto não vazio de conseqüências, e $g : A \to C$ associa a toda configuração de ações uma conseqüência. As preferências são então definidas sobre conseqüências e não sobre as configurações de ações, isto é,$\succeq_i$ é uma relação de preferência em C.

Definição 2.1.3 *(Jogo Estratégico baseado em função de utilidade ou payoff)*

$G = (N, (A_i)_{i \in N}, (u_i)_{i \in N}, (\succeq_i)_{i \in N})$, onde $u_i : A \to R$ é a função de utilidade para o jogador i; configurações de ações com utilidade maior são preferidas: $a \succeq_i b$ se e somente se $u_i(a) \geq u_i(b)$, para $a, b \in A$

Jogos estratégicos com dois jogadores podem ser visualizados como matrizes. Por convenção, as ações do jogador 1 correspondem às linhas e as ações do jogador 2 correspondem às colunas da matriz. Cada célula da matriz corresponde a uma configuração de ações, onde cada jogador escolheu a ação correspondente a uma linha ou coluna.

Exemplo 2.1.1 *(Dilema do Prisioneiro)*

Dois criminosos são colocados em celas separadas e interrogados. Se ambos confessam o crime, cada um é condenado a 3 anos de prisão. Se apenas um deles confessa o crime, é libertado e usado como testemunha contra o outro que não confessou. Este outro é condenado a 4 anos de prisão. Se nenhum deles confessa o crime, eles serão ambos condenados por um delito menor apenas, e cada um será condenado a 1 ano de prisão (figura 2.1).

Figura 2.1: Jogo Estratégico: O Dilema do Prisioneiro

	não confessar	confessar
não confessar	(-1, -1)	(-4, 0)
confessar	(0, -4)	(-3, -3)

Exemplo 2.1.2 *(Falcão e Pomba)*

Dois animais disputam território. Cada um deles pode se comportar como uma pomba ou como um falcão. Para quem deseja clamar o território o melhor cenário é se comportar como falcão enquanto o oponente se comporta como uma pomba. O pior cenário é quando ambos se comportam como falcões, devido aos prejuízos acarretados por uma luta (figura 2.2).

Figura 2.2: Jogo Estratégico: Falcão e Pomba

	pomba	falcão
pomba	(3, 3)	(1, 4)
falcão	(4, 1)	(0, 0)

Exemplo 2.1.3 *(Cara ou Coroa)*

Dois jogadores lançam uma moeda uma vez cada um. Se o resultado for igual em ambos os lançamentos, o jogador 1 paga um real para o jogador 2. Se o resultado for diferente nos lançamentos, o jogador 2 para um real para o jogador 1 (figura 2.3).

Figura 2.3: Jogo Estratégico: Cara ou Coroa

	cara	coroa
cara	(1, -1)	(-1, 1)
coroa	(-1, 1)	(1, -1)

2.1.1 Equilíbrio Nash e Estratégia Dominate

Dada uma configuração de ações $a \in A$ e uma ação $x \in A_i$, seja $a[i : x]$ uma notação para a configuração de ações que é igual a a a menos da ação do jogador i, que será substituída por x.

Definição 2.1.4 *(Equilíbrio Nash)*

Dado um jogo estratégico $G = (N, (A_i)_{i \in N}, (\succeq_i)_{i \in N})$*, um configuração de ações* $a \in A$ *é um equilíbrio Nash se e somente se* $\forall i \in N \ \forall x \in A_i : a \succeq_i a[i : x]$

O equilíbrio corresponde a uma situação onde cada jogador age da melhor forma possível, dadas as ações dos outros jogadores, ou seja, a ação escolhida por cada jogador é a melhor resposta para as ações escolhidas pelos outros jogadores. Um equilíbrio Nash pode ser interpretado como um estado estável do jogo: uma vez atingido, mudar de ação é uma escolha irracional para todos os jogadores.

Nem todos os jogos possuem equilíbrio Nash, e alguns jogos possuem mais de um equilíbrio Nash. O exemplo 2.1.3 não apresenta equilíbrio Nash, enquanto o exemplo 2.1.2 apresenta dois pontos de equilíbrio Nash: os pares (*falcão*, *pomba*) e (*pomba*, *falcão*). Em geral, ser equilíbrio Nash é uma das características levadas em conta na análise dos estados candidatos a solução de um jogo. Nem sempre porém os pontos que são equilíbrio Nash correspondem ao melhor cenário para os jogadores. No dilema do prisioneiro (exemplo 2.1.1), o único equilíbrio Nash é o par (*confessar*, *confessar*). Esta situação é chamada de "tragédia dos comuns", e ocorre porque a colaboração não é prerrogativa do julgamento racional que o jogador faz ao escolher que ação tomar.

Definição 2.1.5 *(Estratégias Dominantes)*

Dadas duas estratégias $x, y \in A_i$ *dizemos que* x *domina fracamente* y *se e somente se* $\forall a \in A : a[i : x] \succeq_i a[i : y]$*. Analogamente,* x *domina fortemente* y *se e somente se* $\forall a \in A : a[i : x] \succ_i a[i : y]$*.*

Enquanto o equilíbrio Nash exige que a ação de cada jogador seja ótima dadas as ações dos outros jogadores, uma estratégia dominante deve ser ótima a despeito das ações escolhidas pelos outros jogadores. Dizemos que uma estratégia x é fortemente dominante se ela domina fortemente todas as outras estratégias $y \neq x \in A_i$ (o similar vale para estratégias fracamente dominates). No exemplo 2.1.3, *confessar* é uma estratégia dominante para os dois jogadores.

2.1.2 Jogos Estritamente Competitivos

Definição 2.1.6 *(Jogos Estritamente Competitivos)*

Um jogo estratégico com dois jogadores $G = (\{1,2\}, (A_1, A_2), (\succeq_1, \succeq_2))$ *é estritamente competitivo se e somente se* $\forall a, b \in A : a \succeq_1 b \Leftrightarrow b \succeq_2 a$.

As preferências dos jogadores são estritamente opostas nos jogos estritamente competitivos, também conhecidos como jogos de soma-zero (soma-zero é o que acontece com os valores das funções de utilidade nesta classe de jogos). O jogo apresentado no exemplo 2.1.3 é estritamente competitivo.

Definição 2.1.7 *(Ação Maxminimizadora)*

Uma ação x *é maxminimizadora para o jogador* 1 *se e somente se* $\forall x' \in A_1 : \min_{y \in A_2} u_1(x, y) \geq min_{y \in A_2} u_1(x', y)$, *e analogamente para o jogador* 2.

Em jogos estritamente competitivos, maxminimizadores são estratégias ótimas para jogadores atenciosos.

Theorema 2.1.1 *Seja* $G = (\{1,2\}, (A_1, A_2), (u_1, u_2))$ *um jogo estritamente competitivo. Se* (a_1, a_2) *é um equilíbrio Nash, então* (a_i) *é maxminimizadora para o jogador* i *e* $u_1(a_1, a_2) = \max_{x \in A_1} \min_{y \in A_2} u_1(x, y)$. *Se* G *possui um equilíbrio Nash, então se* a_1 *é maxminimizadora para para o jogador* 1 *e* a_2 *para o jogador* 2, (a_1, a_2) *é um equilíbrio Nash.*

O teorema apresenta uma caracterização alternativa para equilíbrio Nash no caso de jogos estritamente competitivos, através da maxminimização. O processo de maxminimização pode ser considerado uma justificativa racional para o fato de os jogadores escolherem ações que levam ao equilíbrio.

A validade do teorema 2.1.1 também implica que os equilíbrios Nash de jogos estritamente competitivos correspondem todos ao mesmo valor da função de utilidade, e são intercambiáveis: se (a_1, a_2) e (b_1, b_2) são equilíbrio, (a_1, b_2) e (b_1, a_2) também são, e possuem o mesmo valor para a função de utilidade. O valor da função utilidade nos pontos de equilíbrio de um jogo estritamente competitivo é dito "valor do jogo".

2.2 Jogos Extensivos

Os modelos de jogos extensivos permitem uma caracterização detalhada da interação entre os jogadores, introduzindo estados intermediários a cada interação de um jogador.

Dada uma seqüencia infinita de ações $h = (a_1, a_2, \cdots)$, $h|k = (a_1, a_2, \cdots, a_k)$ denota a subseqüência inicial de ações de tamanho k de h.

Definição 2.2.1 *(Jogo Extensivo de Informação Perfeita)*

Um jogo extensivo de informação perfeita é uma tupla $G = (N, H, P, (\succeq_i)_{i \in N})$ *onde:*

- $N = \{1, \cdots, n\}$ *é um conjunto finito de jogadores.*
- H *é um conjunto de seqüencias (finitas ou infinitas) sobre um conjunto de* A *de ações. Estas seqüencias serão chamadas de* ***histórias*** *do jogo.* H *deve satisfazer:*

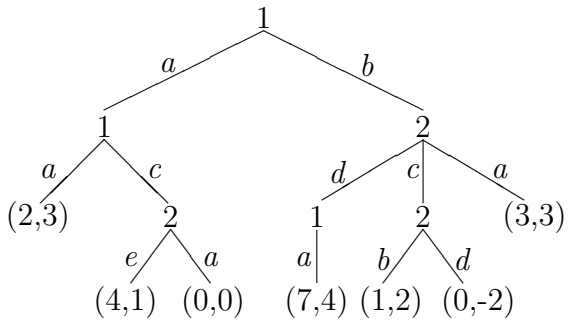
  1. *a seqüencia vazia* $() \in H$*;*
  2. H *é fechado para subseqüências iniciais, isto é, se* $h \in H$ *tem comprimento* l*, então para todo* $k < l$*,* $h|k \in H$*. Se* $h \in H$ *é infinita,* $h|k \in H$ *para todo* k*;*
  3. *Dada uma seqüencia infinita* h *tal que para todo* k $h|k \in H$*, então* $h \in H$*.*

  Seja $Z \subseteq H$ *o conjunto das histórias terminais de* H*, ou seja,* $h \in Z$ *se e somente se para todo* $h' \in H$ *tal que* $h'|k = h$*,* $h' = h$*.*

- $P : H \backslash Z \to N$ *é a função de jogador que associa a toda história não terminal o jogador que está na vez de jogar.*
- $\succeq_i$ *é uma relação de preferência em* Z*.*

Dizemos que um jogo extensivo G é finito se e somente se o conjunto de histórias H de G é finito. Dizemos que G tem horizonte finito se e somente se todas as histórias de H têm comprimento finito.

Figura 2.4: Jogo Extensivo com Dois Jogadores



Assim como nos jogos estratégicos, é possível utilizar, no lugar da relação de preferência sobre histórias terminais, funções de utilidade ou relações de preferência sobre conseqüências.

Definição 2.2.2 *(Estratégia)*

Dada uma história $h = (a_1, a_2, \cdots, a_n)$ *e uma ação* $b \in A$*, seja a história* $(h, b) = (a_1, a_2, \cdots, a_n, b)$ *e* $A(h) = \{x \in A | (h, x) \in H\}$ *o conjunto de ações possíveis depois de* h*. Uma estratégia para o jogador* i *é uma função* $s_i : P^{-1}(\{i\}) \rightarrow A$ *tal que* $s_i(h) \in A(h)$*.*

Dada uma configuração de estratégias $s = (s_i)_{i \in N}$, seja $O(s) \in H$ a história resultante quando os jogadores empregam suas respectivas estratégias.

2.2.1 Forma Estratégica de Jogos Extensivos

É possível condensar um jogo extensivo em um jogo estratégico, porém haverá perda de informações sobre as interações dos jogadores. Na prática, o jogo será representado como se fosse composto de um único evento em que os jogadores determinam, de início, toda a sua estratégia. No jogo estratégico resultante, as ações correspondem às estratégias dos jogadores no jogo extensivo original.

Definição 2.2.3 *(Forma Estratégica de um Jogo Extensivo)*

Seja $G = (N, H, P, (\succeq_i)_{i \in N})$ *um jogo extensivo. A Forma Estratégica de* G *é a tupla* $G^{\circ} = (N, (S_i)_{i \in N}, (\succeq'_i)_{i \in N})$*, onde:*

- *S_i é o conjunto de estratégias do jogador i em G;*
- *$s \succeq'_i s'$ se e somente se $O(s) \succeq_i O(s')$.*

2.2.2 Subjogo e Equilíbrio

Definição 2.2.4 *(Equilíbrio Nash de um Jogo Extensivo)*

Seja $G = (N, H, P, (\succeq_i)_{i \in N})$ um jogo extensivo. Uma configuração de estratégias s é um equilíbrio Nash do jogo extensivo G se e somente se $\forall i \in N \ \forall x \in S_i : O(s) \succeq_i O(s[i : x])$.

O equilíbrio Nash de um jogo extensivo poderia igualmente ser definido como o equilíbrio Nash da forma estratégica do jogo extensivo considerado.

Esta definição alternativa é por si só um indicativo de que o conceito de equilíbrio Nash de um jogo extensivo não leva em consideração todas as informações que o modelo de jogo extensivo tem a oferecer; informações quanto à ordem das jogadas são irrelevantes. O resultado é que configurações de estratégia não plausíveis quando a ordem de jogadas é considerada podem ser equilíbrio Nash do jogo extensivo.

Para formular uma noção de equilíbrio mais adequada a jogos extensivos, é necessário introduzir um fator que leve em conta a ordem das jogadas na análise do equilíbrio.

Dadas duas seqüências h e h', seja (h, h') uma notação para a seqüência resultante da concatenação de de h com h'.

Definição 2.2.5 *(Subjogo de um Jogo Extensivo)*

Seja $G = (N, H, P, (\succeq_i)_{i \in N})$ um jogo extensivo. Para cada história h de G podemos definir um subjogo de G iniciado a partir de h, como uma tupla $G(h) = (N, H|_h, P|_h, (\succeq_i |_h)_{i \in N})$, onde:

- *$H|_h = \{h' | (h, h') \in H\}$*
- *$P|_h(h') = P(h, h')$ para cada $h' \in H|_h$*
- *$h' \succeq_i |_h h''$ se e somente se $(h, h') \succeq_i (h, h'')$*

Estratégias s_i e configurações de estratégias s também podem ser restritas a um subjogo iniciado partir da história h: $s_i|_h(h') = s_i(h, h')$, e $s|_h(h') = s(h, h')$.

Definição 2.2.6 *(Equilíbrio Perfeito de Subjogo)*

Seja $G = (N, H, P, (\succeq_i)_{i \in N})$ um jogo extensivo, e s uma configuração de estratégias para G. Dizemos que s é um equilíbrio perfeito de subjogo se e somente se para todo $h \in H$, $s|_h$ é um equilíbrio Nash de $G(h)$.

Considerando h como a história vazia, percebemos que todo equilíbrio perfeito de subjogo é também equilíbrio Nash.

Theorema 2.2.1 *(Teorema de Kuhn)*

Todo jogo extensivo finito de informação perfeita tem um equilíbrio perfeito de subjogo.

A prova do teorema de Kuhn pode ser vista em [OR94], e é baseada no processo de indução às avessas.

2.2.3 Determinação de Jogos "Ganha/Perde" com Dois Jogadores

Definição 2.2.7 *(Jogos Ganha/Perde)*

Seja $G = (\{1, 2\}, H, P, u_1, u_2)$ um jogo extensivo de informação perfeita. Dizemos que G é um jogo ganha/perde se e somente se G é estritamente competitivo e para toda história h, $u_1(h) = 1$ ou $u_1(h) = -1$.

$u_1(h) = 1$ e $u_1(h) = -1$ são respectivamente interpretados como "o jogador 1 ganha" e "o jogador 1 perde".

Uma conseqüência do teorema de Kuhn é que G é determinado, isto é, um dos jogadores de G possui uma estratégia vencedora.

Uma forma mais geral desse resultado pode ser verificada, apesar de não ser conseqüência do teorema de Kuhn:

Theorema 2.2.2 *Todo jogo ganha/perde de horizonte finito é determinado.*

2.2.4 Jogos de Informação Imperfeita

Nos modelos anteriores, há margem para incertezas apenas acerca de informações futuras, como por exemplo quais serão as próximas jogadas dos outros jogadores. Os jogadores não tem informações privadas e nem se esquecem das coisas que já aconteceram.

O conceito de jogos com informação imperfeita permite que os jogadores possuam informação parcial sobre as ações executadas anteriormente, podendo não ser capazes de determinar com precisão o ponto em que estão na árvore de jogo. Ao se deparar com o dilema de decidir que ação tomar, o jogador constrói uma expectativa acerca daquilo que não sabe, com base no comportamento esperado dos outros jogadores e nas suas noções sobre os eventos que aconteceram no passado.

A definição seguinte generaliza a definição prévia de jogos extensivos de informação perfeita para permitir que os jogadores estejam parcialmente informados sobre os eventos passados ao escolher suas ações.

Definição 2.2.8 *(Jogos Extensivos de Informação Imperfeita)*

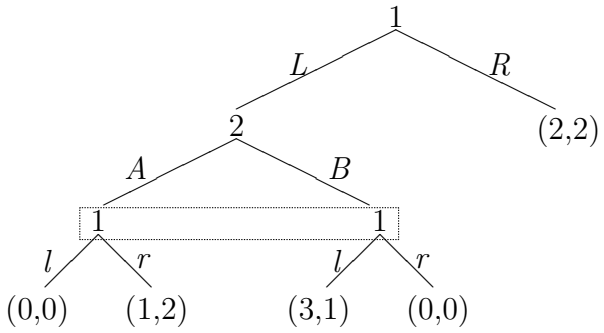
Um jogo extensivo de informação imperfeita $G = (N, H, P, (\mathcal{I}_i)_{i \in N}, (\succeq_i)_{i \in N})$ *é uma tupla onde* $(N, H, P, (\succeq_i)_{i \in N})$ *corresponde a um jogo extensivo e para cada jogador* $i \in N$, $\mathcal{I}_i$ *é uma partição do conjunto* $P^{-1}(\{i\})$ *(das histórias onde o jogador* i *tem a vez de jogar), com a propriedade de que para todo* $h, h' \in I_r \in \mathcal{I}_i$, $A(h) = A(h')$.

Os elementos de $\mathcal{I}_i$ são chamados conjuntos de informação. Se os conjuntos de informações são unitários, então o jogo corresponde ao modelo de informação perfeita.

Se o jogador i tem que fazer uma escolha no ponto do jogo correspondente à história $h \in I_r \in \mathcal{I}_i$, ele não sabe a qual dentre as histórias da partição I_r corresponde a história real. Para sustentar essa interpretação, devemos assumir que as histórias em I_r não podem ser distinguidas pelas ações possíveis. Isto é assegurado pela condição de que para todo $h, h' \in I_r \in \mathcal{I}_i$, $A(h) = A(h')$.

Como o jogador não pode distinguir entre as histórias de um conjunto de informações, suas estratégias têm que ser uniformes em cada conjunto.

Figura 2.5: Jogo Extensivo de Informação Imperfeita com Dois Jogadores



Definição 2.2.9 *(Estratégia para um Jogo Extensivo de Informação Imperfeita)*
Uma estratégia para um jogo extensivo de informação imperfeita é uma função $s_i : \mathcal{I}_i \rightarrow A$ *tal que* $s_i(I_r) \in A(I_r)$.

Uma vez definida a estratégia, podemos associar uma forma estratégica a um jogo extensivo de informação imperfeita do mesmo modo que associamos jogos extensivos de informação perfeita à sua forma estratégica. Os conjuntos de estratégias utilizados na forma estratégica contém estratégias definidas sobre conjuntos de informação.

O conceito da equilíbrio Nash pode ser facilmente estendido para jogos extensivos de informação imperfeita. Esta definição pode ser obtida diretamente através da forma extensiva do jogo, por exemplo.

O conceito de equilíbrio perfeito de subjogo porém não pode ser diretamente aplicado, dada a dificuldade em aplicar indução às avessas sobre os conjuntos de informação. Outros tipos de equilíbrio, levando em conta as peculiaridades acarretadas pela informação imperfeita podem ser vistos em [OR94].

Capítulo 3

Estruturas de Jogos para Linguagens Lógicas

Os modelos de teoria dos jogos tem um apelo imediato aos pesquisadores de lógica, conforme observou Johan van Benthem em [VB01]. Uma árvore de jogo extensivo, por exemplo, descreve um processo interativo multi-agentes (onde os nós são os estados possíveis, com ações disponíveis indicadas por jogador) de forma semelhante a uma estrutura de Kripke.

Além do apelo estrutural, há ainda o apelo prático. A análise de estratégias, pré-requisito da interação racional e objeto de estudo de jogos extensivos, é de grande valia em computação para modelagem e análise de agentes inteligentes e suas interações no tempo.

Este capítulo, baseado em [VB01], desenvolve uma visão lógica acerca dos principais modelos e conceitos de teoria dos jogos.

Segundo [GB02], o aparato da lógica modal pode ser utilizado para modelar duas visões conceitualmente muito diferentes de teoria dos jogos: uma é a visão estratégica correspondente ao raciocínio de jogadores racionais, isto é, como o jogador decide sua forma de jogar. A outra é a visão acerca das possibilidades reais de resultados do jogo, ou uma análise normativa dos poderes do jogador (o que cada jogador é capaz de conseguir em termos de resultados).

Veremos a seguir arcabouços de linguagens lógicas para ambas as visões.

3.1 Jogos Extensivos como Modelos para Lógicas Dinâmicas

As árvores que servem de modelo para jogos extensivos (árvores de jogo) de informação perfeita podem ser diretamente utilizadas como estruturas de Kripke para lógicas dinâmicas.

Duas abordagens para esta situação foram exploradas na literatura. A primeira abordagem propõe modalidades de ação correspondentes às ações disponíveis para os jogadores. A segunda abordagem propõe modalidades de poder para cada jogador ou grupo de jogadores associados, que relaciona a cada estado um conjunto de estados que o jogador ou grupo em questão tem o poder de obter como resultado de suas ações, em detrimento das jogadas alheias. Examinamos em separado cada uma dessas abordagens nas seções seguintes.

3.1.1 Linguagens dinâmicas de ações para vários agentes

Árvores de jogos de informação perfeita podem ser vistas como modelos $\mathcal{M} = (S, \{R_a | a \in A\}, V)$ onde:

- S é um conjunto de estados;
- R_a são relações binárias representando as possíveis transições para ações do tipo a;
- V é uma função de avaliação para letras proposicionais denotando propriedades locais dos estados.

Proposições especiais como $\texttt{turn}_i$ e $\texttt{win}_i$ significam, respectivamente, "o jogador i tem a vez de jogar" e "o jogador i venceu o jogo". `end` é outra proposição especial que vale em todos os estados correspondentes ao fim do jogo. Outras proposições podem indicar valores ou preferências de estados (finais) específicos para os jogadores.

Modelos deste tipo são descritos por uma linguagem modal dinâmica com ações básicas $a, b, \cdots$ e as seguintes operações sobre ações:

- união $\cup$ - escolha

- composição ; - execução em seqüencia
- estrela Kleene * - interação finita arbitrária
- $(\varphi)?$ - teste de φ

Para todas as ações resultantes A, a linguagem tem as modalidades:

- $\langle A\rangle\varphi$ - em algum A-sucessor, φ vale.
- $[A]\varphi$ - em todos os A-sucessores, φ vale.

Isso nos permite falar das relações entre estratégias (seqüencias de ações atômicas ou construídas a partir das atômicas por aplicação dos operadores) e resultados (codificados por proposições), através de fórmulas como: $[c \cup d]\langle\, a \cup b \,\rangle p$.

Considere por exemplo a computação de uma indução às avessas, para descobrir que jogador tem uma estratégia vencedora, em qualquer nó de um jogo soma-zero de dois jogadores onde os resultados são $\texttt{win}_i$ ou $\neg\ \texttt{win}_i$ para cada jogador i. Seja A a união de todos as ações disponíveis. A regra para as posições de vitória pode ser definida pela recursão: $\texttt{WIN}_i \leftrightarrow (\ \texttt{end} \wedge \texttt{win}_i) \vee (\texttt{turn}_i \wedge \langle A\rangle\ \texttt{WIN}_i) \vee (\neg\ \texttt{turn}_i \wedge [A]\ \texttt{WIN}_i)$

A linguagem também pode ser usada para falar sobre estratégias explicitamente. Uma estratégia para o jogador i é uma função parcial $f_i : S \to A$, dos nós onde é a vez do jogador i em ações atômicas. Assim sendo, uma estratégia σ é um conjunto de instâncias de ações atômicas, e pode ser visto como qualquer tipo de ação a que pode ser executado em vários estados. Dizer que σ é uma estratégia vencedora é o mesmo que dizer: $[A^*](\texttt{turn}_i \to \langle\sigma\rangle\ \texttt{WIN}_i)$, onde A^* varia da raiz até qualquer nó do jogo.

A linguagem fornece recursos para raciocinar sobre equilíbrio Nash manipulando estratégias de dois jogadores [DB00].

A Lógica Dinâmica de Ações descrita é decidível em geral, e também sob classes específicas de árvores finitas. Em ambos os casos, há uma axiomatização completa para o conjunto de axiomas correspondentes [BV00].

3.1.2 Linguagem de Poderes dos jogadores e Resultados

Uma visão externa e global do jogo surge ao se pensar em jogos no nível dos poderes dos jogadores. Esta visão corresponde a uma análise de como os jogadores podem influenciar o desfecho do jogo. Outras lógicas de jogos podem ser construídas com este enfoque.

Estratégias em jogos extensivos são definidas da forma usual como funções de todos os nós onde é a vez do jogador i nas ações disponíveis. As "relações de poder" para um jogo codificam o estados que os jogadores podem atingir conforme utilizam suas estratégias:

$\rho G^i s, X$ representa que jogador i tem uma estratégia para jogar (o restante do) jogo G a partir do nó s tal que os estados resultantes estão sempre no conjunto X.

As seguintes restrições se aplicam às relações de poder:

- Fecho sobre superconjuntos: As relações de poder são fechadas sobre superconjuntos.
  (C1) Se $\rho G^i s, Y$ e $Y \subseteq Y'$ então $\rho G^i s, Y'$.

- Condição de consistência: Os jogadores não podem forçar o resultado do jogo em dois conjuntos distintos de resultados.
  (C2) Se $\rho G^1 s, Y$ e $\rho G^2 s, Z$ então $Y \cap Z \neq \emptyset$.

- Determinação: Todos os jogos finitos de dois jogadores são determinados, ou seja, para qualquer convenção acerca das posições de vitória, um dos dois jogadores deve ter uma estratégia vencedora:
  (C3) Se não $\rho G^1 s, Y$ então $\rho G^2 s, S - Y$, e vice versa.

Todo jogo finito de informação perfeita G entre dois jogadores dá origem a relações de poder satisfazendo (C1), (C2) e (C3).

As relações de poder são a base para definir o operador modal de interesse, $\{G, i\}$ onde G é um jogo e i um jogador, nesta visão de jogos:

$\{G, i\}\varphi$ vale em um estado s se e somente se $\rho G^i s, X$ e em todo estado de X vale φ.

A validade da fórmula significa que o jogador i tem uma estratégia para jogar o jogo G a partir de s e forçar que o jogo termine em um conjunto de estados de resultado onde em todos os estados vale φ. Esta é uma visão geral das modalidades de jogo de [PA85] e [MP02].

3.2 Jogos de Informação Imperfeita e Modelos Epistêmicos

Em jogos de informação imperfeita os jogadores podem não saber em que ponto da árvore de jogo estão, durante o curso do jogo. As situações que inspiram esta classe de modelos são várias: os jogadores podem ter limitações cognitivas, como memória, ou percepção limitada dos movimentos dos outros jogadores; a definição do jogo impõe o desconhecimento inicial de certas informações, como em jogos de carta, dados ou charadas.

O recurso utilizado para representar as ausências de informação em teoria dos jogos são os conjuntos de informação. Uma abordagem lógica natural sobre o modelo sugere o uso dos conjuntos de informação na definição de estados indistinguíveis para operadores modais de lógicas epistêmicas.

A falta de informações se reflete nos poderes do jogador na medida em que interfere na formação de estratégias. O mesmo operador modal apresentado anteriormente pode ser utilizado no modelo de informação imperfeita, desde que a relação de poder seja enfraquecida.

3.2.1 Linguagem Epistêmica Dinâmica

Estruturas epistêmicas sobre modelos de Jogos de Informação Imperfeita são construídas a partir dos conjuntos de informação. Os estados no conjunto de informação de um jogador i originam "estados indistinguíveis" para relações binárias de incerteza $\sim_i$.

O modelo resultante contém uma estrutura multi-modal S5: $\mathcal{M} = (S, \{R_a | a \in A\}, \{\sim_i | i \in I\}, V)$ e naturalmente interpreta uma linguagem epistêmica e dinâmica, que introduz operadores de conhecimento para cada jogador.

$\mathcal{M}, s \models K_i\varphi$ se e somente se $\mathcal{M}, t \models \varphi$ para todo t tal que $s \sim_i t$

É possível falar acerca de conhecimento (e ignorância) dos jogadores quando o jogo atinge um certo estado, através de fórmulas como $K_2(\langle a\rangle p \vee \langle b\rangle p)$ e $\neg K_2\langle a\rangle p \wedge \langle b\rangle p$. Os jogadores podem raciocinar sobre o conhecimento (ou ignorância) de outros jogadores por intermédio de fórmulas como $K_1K_2\varphi$, $K_1\neg K_2\varphi$.

Para um raciocínio sistemático sobre as ações disponíveis para cada jogador, conhecimento e ignorância em modelos arbitrários desse tipo, o conjunto completo de axiomas para validade em Lógica Epistêmica Dinâmica é o seguinte:

- A mínima Lógica Dinâmica com operadores modais para as ações $[A]$;
- S5 epistêmico para cada operador modal K_i.

A Lógica Mínima com a inclusão de um operador para conhecimento comum pode ser vista em [FA95]. Não há outros axiomas na Lógica Epistêmica Dinâmica geral.

Algumas restrições interessantes sobre Jogos de Informação Imperfeita induzem axiomas especiais para raciocinar sobre jogadores em jogos de determinado estilo. [OR94] apresenta uma série de restrições com suas respectivas motivações.

Exemplo 3.2.1 *(Jogador da Vez)*

O fato "de quem é a vez de jogar" é de conhecimento comum. Este é um requerimento razoável, especialmente em jogos de tabuleiro ou cartas. $\texttt{turn}_i \rightarrow C_{\{1,2\}}$ `turn_`i

Exemplo 3.2.2 *(Ações Uniformes sobre Conjuntos de Informação)*

Todos os nós no mesmo conjunto de informações tem as mesmas ações possíveis. Esta é uma propriedade razoável ao lidar com jogadores que teriam a capacidade de distinguir os estados pelo conjunto de informações disponíveis. $\texttt{turn}_i \wedge \langle a\rangle\top \rightarrow K_i\langle a\rangle\top$ *ou até mesmo* $\langle a\rangle\top \rightarrow C_{\{1,2\}}\langle a\rangle\top$.

Exemplo 3.2.3 *(Princípio do Intercâmbio - Interchange Principle)*

$K_i[a]\varphi \rightarrow [a]K_i\varphi$. *Se o jogador i sabe exatamente neste momento que fazer a ação a acarretará na validade de* φ*, então realizando a o agente saberá* φ*. Essa*

implicação é razoável para ações sem efeitos colaterais epistêmicos, mas não em geral. Em árvores de jogo, a fórmula só vale sobre a seguinte restrição sobre a ação a e as incertezas de i.

$$\forall xyz(xR_a y \wedge y \sim_i z) \rightarrow \exists u(x \sim_i u \wedge uR_a z)$$

Esta é a propriedade de confluência tal como conhecida, para as relações de ação e incerteza.

Correspondências entre axiomas epistêmico-dinâmicos e restrições em jogos de informações imperfeitas podem ser computados através de algoritmos conhecidos de lógicas modais [BV00].

3.2.2 Poderes e Resultados com Informação Imperfeita

Em jogos de informação imperfeita os jogadores só podem utilizar estratégias uniformes, onde a próxima ação em cada nó é a mesma sobre todo o conjunto de informações ao qual o nó pertence. A relação de poder definida na seção 3.1.2 deve ser adaptada.

$\rho G^i s, X$ representa que jogador i tem uma estratégia uniforme para jogar (o restante do) jogo G a partir de qualquer nó que esteja no mesmo conjunto de informações do nó s tal que os estados resultantes estão sempre no conjunto X. A cada nó, um jogador pode forçar os conjuntos de resultados que são produzidos por uma de suas estratégias uniformes.

As condições (C1) de fecho sobre superconjuntos e (C2) de consistência são válidas tb no caso de poderes em jogos de informação imperfeita. Porém a condição (C3), de determinação, falha.

Modificadas as relações de poder, a definição do operador modal dos poderes do jogador não se altera.

$\{G, i\}\varphi$ vale em um estado s se e somente se $\rho G^i s, X$ e em todo estado de X vale φ.

Capítulo 4

Lógica de Jogos

Nos últimos anos muitas lógicas foram propostas para formalizar jogos, embasadas nos formalismos de teoria dos jogos. Dentre as lógicas propostas, há as que falam sobre o raciocínio dos jogadores, considerando a forma como eles escolhem suas jogadas dados os seus estados de conhecimento. Essa abordagem permite provar propriedades relativas aos resultados do jogo, como por exemplo equilíbrio Nash e indução às avessas [KN96], [GA99], [TC99], [HH02].

Outras lógicas para jogos seguiram uma abordagem diferente: a especificação de jogos, ou de sistemas que se comportam como jogos. Essa classe de lógicas permite observar os jogos sob uma perspectiva de execução e efeito sobre o próprio sistema, evidenciando as formas de interação e os possíveis resultados alcançáveis para o jogo e conseqüentemente para o sistema sendo modelado [MP00]. Vários sistemas econômicos e computacionais se encaixam nesta modelagem.

Neste capítulo apresentaremos uma Lógica de Jogos (GL) proposta por Rohit Parikh em [PA85]. A GL é um formalismo para modelar jogos, ou sistemas que se comportam como jogos. Tal formalismo pode ser usado como um dispositivo computacional, uma vez que conjecturas sobre propriedades de um dado jogo podem ser traduzidas em fórmulas e dadas a um provador automático.

Na Lógica de Jogos, as ações possíveis para cada jogador a cada turno são consideradas jogos. Avaliando os jogos disponíveis para cada jogador em estados (turnos) sucessivos, obtemos uma análise dos desfechos possíveis de uma seqüencia de jogadas. Assim podemos expressar propriedades acerca dos resultados que um jogador pode atingir (poderes do jogador), e de uma forma mais ampla, as formas como

cada um pode influenciar o desenrolar do jogo.

Apresentaremos a seguir a *GL*, sua descrição formal de sintaxe e semântica para jogos envolvendo dois jogadores, e uma extensão para jogos com vários (mais de dois) jogadores.

4.1 Apresentação da Lógica de Jogos

O principal objeto do discurso da Lógica de Jogos são jogos. A *GL* trata basicamente de jogos para duas pessoas, enfatizando jogos em universos abstratos. Outro aspecto relevante da *GL* são as formas de combinar jogos resultando em outros jogos, e a formalização das propriedades axiomáticas destes jogos.

A motivação para a construção deste tipo de lógica de jogos é a percepção de que construções como `if-then-else` e `while-do`, presentes na maioria das linguagens de programação, podem ser interpretadas como jogos. Além disso, as propriedades axiomáticas que relacionam programas compostos com suas componentes, estudadas em Lógica de Hoare e Lógica Dinâmica, também se aplicam a jogos.

A lógica de jogos desenvolvida baseia-se em PDL para programas regulares e μ-calculus, e seu poder de expressão está entre PDL [FL79] e μ-calculus [KD82]. Uma axiomatização completa para a *GL* sem o operador dual é apresentada (a axiomatização para a *GL* com o operador dual está em aberto). O operador dual converte um jogo em seu dual, onde os jogadores trocam de papel. Em certos contextos, o dual pode ser substituído por duas negações.

Conforme observado por [MP00], o modelo adotado afasta-se um pouco da abordagem padrão de teoria dos jogos, especialmente no uso de estados externos do mundo que são independentes das posições internas do jogo, resultando em uma rede (teia) de jogos. [MP00] propõe a partir deste modelo não padrão uma noção de jogo que combina jogos extensivos e formas de jogo, e pode ser usada para expressar jogos com informações imperfeitas.

Um jogo convencional para duas pessoas, como xadrez, dama ou jogo da velha, consiste basicamente de três partes: a posição inicial, os movimentos permitidos, e as posições de vitória para o jogador 1 (consideramos que as outras posições de fim de jogo serão de vitória para o jogador 2).

A *GL* considera cada movimento intermediário como um jogo; um único jogo de xadrez corresponderá a uma rede de jogos encadeados na *GL*. Outra convenção da *GL* é que se um jogador fica bloqueado, sem nenhuma ação disponível (em *deadlock*), ele é automaticamente um perdedor.

Uma típica declaração da *GL* é: $s \models (\alpha)A$, onde s é um estado do modelo, α é um jogo e A é uma fórmula. O estado s é o estado inicial do jogo α, e o conjunto de estados que satisfazem A representam os estados de vitória para o jogador 1. A assertiva completa significa: Jogador 1 tem uma estratégia vencedora, a partir do estado s, de jogar α de tal forma que o jogo termina com A verdadeiro e o jogador um vence, ou fica bloqueado por culpa de 2.

Note que (α) é um transformador de predicados, que converte o predicado A no predicado $(\alpha)A$. Além disso, é um transformador monotônico: se A implica em B então $(\alpha)A$ implica em $(\alpha)B$. Nesse sentido o operador (α) lembra as modalidades $\langle\alpha\rangle$ e $[\alpha]$ de PDL que também são monotônicas. Porém, essas modalidades são disjuntivas e conjuntivas respectivamente (comutam com $\vee$ e $\wedge$). Em teoria dos jogos isso pode ser explicado dizendo que $\langle\alpha\rangle$ assim como $\vee$ representa um jogo onde apenas o jogador 1 move, e o jogador 2 é o único a mover em $[\alpha]$ e $\wedge$. Modalidades mais gerais para jogos onde ambos os jogadores jogam são monotônicas mas não podem ser conjuntivas ou disjuntivas. Essa é a única propriedades que falta ao (α) e tirando o axioma correspondente de uma axiomatização completa para PDL teremos uma axiomatização completa para lógica de jogos [PA85].

4.2 Sintaxe e Semântica

Definição 4.2.1 *(Sintaxe da* GL*)*

Seja $G = \{g_1, ..., g_n\}$ *um conjunto de jogos atômicos e* $\Phi = \{P_1, ..., P_m\}$ *um conjunto de proposições atômicas.*

1. *Cada* P_i *é uma fórmula.*

2. *Se* A *e* B *são fórmulas, então* $A \vee B$ *e* $\neg A$ *também são.*

3. *Se* A *é uma fórmula e* α *é um jogo, então* $(\alpha)A$ *é uma fórmula.*

4. *Se α e β são jogos, então $\alpha;\beta$ (ou $\alpha\beta$), $\alpha+\beta$, $\langle\alpha*\rangle$, e α^d também são. α^d é o dual de α.*

5. *Se A é uma fórmula então $\langle A\rangle$ é um jogo (teste de A).*

Escreveremos $\alpha.\beta$, $[\alpha*]$ e $[A]$ respectivamente para os duais de $\alpha+\beta$, $\langle\alpha*\rangle$ e $\langle A\rangle$.

Intuitivamente $\alpha;\beta$ significa: jogue α e depois jogue β. $\alpha+\beta$ corresponde a: o jogador 1 tem o primeiro movimento e decide qual jogo entre α ou β será jogado, e então o jogo escolhido é jogado (no jogo $\alpha.\beta$ é o jogador 2 quem decide). Em $\langle\alpha*\rangle$ o jogo α é repetidamente jogado, (zero ou mais vezes) até que o jogador 1 decida parar de joga-lo (ele não precisa declarar antecipadamente quantas vezes α será repetido). O mesmo se passa com o jogador 2 e o jogo $[\alpha*]$. Em α^d os jogadores trocam de papel. E finalmente, no jogo $\langle A\rangle$ a fórmula A é avaliada e, se A é falso então o jogador 1 perde. Caso contrário, ele ganha (o mesmo para 2 e $[A]$).

Definição 4.2.2 *(Modelo para a* GL*)*

Um modelo para a Lógica de Jogos é uma tupla $\mathcal{M}=(W,\pi,\rho)$, onde W é um conjunto de mundos, $\pi:\Phi\to P(W)$ associa a cada proposição atômica um subconjunto de W, $\rho:G\to(W\times P(W))$ associa a cada jogo $g\in G$ um subconjunto $\rho(g)$ de pares (s,X), onde s é um mundo de W, e X é o conjunto de mundos que podem ser atingidos ao jogar g em s. $\rho(g)$ deve satisfazer as condições de monotonia: se $(s,X)\in\rho(g)$ e $X\subseteq Y$, então $(s,Y)\in\rho(g)$.

É conveniente pensar em $\rho(g)$ como um operador de $P(W)$ em $P(W)$, dado pela fórmula $\rho(g)(X)=\{s|(s,X)\in\rho(g)\}$.

Definição 4.2.3 *(Extensão das definições de π e ρ para fórmulas e jogos mais complexos)*

2'. $\pi(A\vee B)=\pi(A)\cup\pi(B)$
$\pi(\neg A)=W-\pi(A)$

3'. $\pi((\alpha)A)=\rho(\alpha)\pi(A)$

5'. $\rho(\alpha;\beta)(X) = \rho(\alpha)(\rho(\beta)(X))$
$\rho(\alpha+\beta)(X) = \rho(\alpha)(X) \cup \rho(\beta)(X)$
$\rho(\langle\alpha*\rangle)(X) = \mu Y(X \subseteq Y \ e \ \rho(\alpha)(Y) \subseteq Y)$
$\rho(\alpha^d)(X) = W - \rho(\alpha)(W - X)$

6'. $\rho(\langle A\rangle)(X) = \pi(A) \cap X$

As seguintes propriedades podem ser verificadas:

- $\rho(\alpha.\beta)(X) = \rho(\alpha)(X) \cap \rho(\beta)(X)$
- $\rho([\alpha*])(X) = \nu Y(X \subseteq Y \text{ e } Y \subseteq \rho(\alpha)(Y))$
- $\rho([A])(X) = (W - \pi(A)) \cup X$

4.3 Conexão com PDL

Dada uma linguagem PDL iremos associar a ela uma lógica de jogos onde a cada programa a_i de PDL são associados dois jogos $\langle a_i\rangle$ e $[a_i]$ [PA85].

- $\rho(\langle a\rangle)(X) = \{s|\exists t(s,t) \in R_a \text{ e } t \in X\}$
- $\rho([a])(X) = \{s|\forall t(s,t) \in R_a \rightarrow t \in X\}$

As fórmulas de PDL podem facilmente ser traduzidas para a lógica de jogos. Se o programa a deve ser executado e jogador 1 quer ter A verdadeiro ao seu término, então se é ele executar a, apenas $\langle a\rangle A$ deve ser verdade para que ele vença. Porém, se é o jogador 2 quem vai executar a então $[a]A$ deve ser verdade para que o jogador 1 seja vitorioso. Se não há computações do programa a partir do estado s, então o jogador 2 é impossibilitado de se mover, $[a]A$ é verdadeiro e o jogador 1 ganha. Diferente da situação no jogo de xadrez, uma situação onde o jogador não pode jogar é vista como uma derrota para ele, por definição.

A Lógica de Jogos é mais expressiva que PDL: a fórmula $\langle[b]*\rangle$`Falso` da *GL* diz que não há computações infinitas do programa b, uma noção que não pode ser expressa em PDL. A fórmula $\langle(\langle a\rangle;[b])*\rangle$`Falso`, que diz que o jogador 1 pode fazer a em resposta aos movimentos b do jogador 2 de tal forma que eventualmente o

jogador 2 ficará bloqueado, não pode ser expressa em PDL$^{\Delta}$. Vamos agora mostrar como boa ordenação (*Well-Foundedness*) pode ser definida em lógica de jogos.

Dada uma ordem linear R sobre um conjunto W, considere o modelo para lógica de jogos onde g denota $[a]$ e R_a é a relação inversa de R. Então, R é *well-founded* sobre W se e somente se a fórmula $\langle g* \rangle$`Falso` é verdadeira. O jogador 1 não pode terminar o jogo sem perder, mas ele deve terminá-lo em algum momento, mesmo assim. A única forma de 1 vencer é dizer repetidamente para o jogador 2 - vá jogando..., e esperar que o jogador 2 cedo ou tarde fique bloqueado. O subjogo $[a]$ de $\langle [a]* \rangle$ é um jogo onde o jogador 2 joga, e no jogo principal $\langle [a]* \rangle$ o jogador 1 é responsável por decidir quantas vezes $[a]$ é jogado. Logo, o jogador 1 ganha se e somente se não há seqüências infinitas descendentes de R_a em W.

4.4 Conexão com $\mu-$calculus

A lógica de jogos pode ser traduzida para $\mu-$calculus, e pelo procedimento de decisão de Kozen & Parikh, é decidível.

O $\mu-$calculus, tal como PDL, é baseado em relações binárias. Toda relação binária R pode ser vista como jogo de duas maneiras, $\langle R \rangle$ e $[R]$, porém nem todo jogo pode ser descrito em uma dessas duas formas.

Podemos contudo dar uma tradução da GL para $\mu-$calculus considerando que todo jogo g sobre W pode ser escrito como $\langle a \rangle [b]$ sobre W', onde W' é superconjunto de W e seu tamanho é exponencial em W [PA85].

4.5 Exemplo: "eu corto e você escolhe"

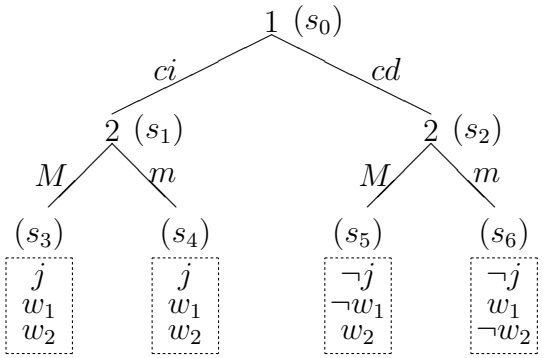
Um típico jogo de duas pessoas é o tradicional método de divisão entre duas pessoas, que consiste na proposta: um faz a divisão e o outro escolhe o primeiro pedaço. Intuitivamente, acreditamos que esta seqüencia de ações leva a um resultado justo dado que se a primeira pessoa fizer um corte desigual gerando um pedaço melhor em detrimento do outro, a segunda pessoa, sendo a ela a primeira a escolher a parte que lhe cabe, optaria pelo melhor pedaço deixando o pior para quem fez tal divisão. Assim sendo, a melhor opção para quem faz o corte é gerar dois pedaços

igualmente atraentes, ou seja, fazer uma divisão justa.

Podemos utilizar a GL para confirmar este raciocínio intuitivo, e ver se o jogo descrito verifica as propriedades:

- Ambos os jogadores estão aptos a ficar com um pedaço atraente;
- Ambos os jogadores possuem uma estratégia para atingir uma divisão justa.

Figura 4.1: Árvore de Jogo: Eu corto e você escolhe



O jogo completo tem dois turnos. No primeiro turno, o jogador 1 tem duas ações disponíveis: cortar dois pedaços iguais ou cortar dois pedaços diferentes. No segundo turno, o jogador 2 tem também duas ações disponíveis: escolher o pedaço mais atraente ou escolher o pedaço menos atraente. A figura 4.1 mostra a árvore de jogo correspondente, onde:

- os estados são $s_0 \cdots s_6$;
- as ações ci, cd, M, m significam respectivamente: "cortar dois pedaços iguais", "cortar dois pedaços desiguais", "escolher o pedaço mais atraente", "escolher o pedaço menos atraente";
- as proposições válidas nos nós terminais estão representadas no retângulo pontilhado. j significa que a divisão foi justa, w_1 e w_2 representam respectivamente que o jogador 1 ficou com um bom pedaço e o jogador 2 ficou com um bom pedaço.

4.5.1 O Jogador 1 tem uma estratégia vencedora?

$s_0 \models (ci + cd); (M.m)w_1$?

$\rho((ci + cd); (M.m))\pi(w_1)$
$\rho((ci + cd); (M.m))\{s_3, s_4, s_6\}$
$\rho(ci + cd)(\rho(M.m)\{s_3, s_4, s_6\})$
$\rho(ci + cd)(\rho(M)\{s_3, s_4, s_6\} \cap \rho(m)\{s_3, s_4, s_6\})$
$\rho(ci + cd)(\{s_1\} \cap \{s_1, s_2\})$
$\rho(ci + cd)(\{s_1\})$
$\rho(ci)(\{s_1\}) \cup \rho(cd)(\{s_1\})$
$\{s_0\} \cup \emptyset$
$\{s_0\}$

Resposta: Sim.

4.5.2 O Jogador 2 tem uma estratégia vencedora?

$s_0 \models ((ci + cd); (M.m))^d w_2$?

$\rho(((ci + cd); (M.m))^d)\pi(w_2)$
$\rho(((ci + cd); (M.m))^d)\{s_3, s_4, s_5\}$
$W - \rho((ci + cd); (M.m))(W - \{s_3, s_4, s_5\})$
$W - \rho((ci + cd); (M.m))\{s_0, s_1, s_2, s_6\}$
$W - \rho(ci + cd)(\rho(M.m)\{s_0, s_1, s_2, s_6\})$
$W - \rho(ci + cd)(\rho(M)\{s_0, s_1, s_2, s_6\} \cap \rho(m)\{s_0, s_1, s_2, s_6\})$
$W - \rho(ci + cd)(\emptyset \cap \{s_2\})$
$W - \rho(ci + cd)(\emptyset)$
$W - \rho(ci)(\emptyset) \cup \rho(cd)(\emptyset)$
$W - (\{s_1, s_2, s_3, s_4, s_5, s_6\} \cup \{s_1, s_2, s_3, s_4, s_5, s_6\})$
$W - \{s_1, s_2, s_3, s_4, s_5, s_6\}$
$\{s_0\}$

Resposta: Sim.

4.5.3 O Jogador 1 pode atingir uma divisão justa?

$s_0 \models (ci + cd); (M.m)j$?

$\rho((ci + cd); (M.m))\pi(j)$
$\rho((ci + cd); (M.m))\{s_3, s_4\}$
$\rho(ci + cd)(\rho(M.m)\{s_3, s_4\})$
$\rho(ci + cd)(\rho(M)\{s_3, s_4\} \cap \rho(m)\{s_3, s_4\})$
$\rho(ci + cd)(\{s_1\} \cap \{s_1\})$
$\rho(ci + cd)(\{s_1\})$
$\rho(ci)(\{s_1\}) \cup \rho(cd)(\{s_1\})$
$\emptyset \cup \{s_0\}$
$\{s_0\}$

Resposta: Sim.

4.5.4 O Jogador 2 pode atingir uma divisão justa?

$s_0 \models ((ci + cd); (M.m))^d j$?

$\rho(((ci + cd); (M.m))^d)\pi(j)$
$\rho(((ci + cd); (M.m))^d)\{s_1, s_3, s_4\}$
$W - \rho((ci + cd); (M.m))(W - \{s_1, s_3, s_4\})$
$W - \rho((ci + cd); (M.m))\{s_0, s_2, s_5, s_6\}$
$W - \rho(ci + cd)(\rho(M.m)\{s_0, s_2, s_5, s_6\})$
$W - \rho(ci + cd)(\rho(M)\{s_0, s_2, s_5, s_6\} \cap \rho(m)\{s_0, s_2, s_5, s_6\})$
$W - \rho(ci + cd)(\{s_2\} \cap \{s_2\})$
$W - \rho(ci + cd)(\{s_2\})$
$W - \rho(ci)(\{s_2\}) \cup \rho(cd)(\{s_2\})$
$W - (\{s_0\} \cup \emptyset)$
$W - \{s_0\}$
$\{s_1, s_2, s_3, s_4, s_5, s_6\}$

Resposta: Não.

4.6 Completude

Os seguintes axiomas e regras são completos para a GL sem o operador dual.

4.6.1 Axiomas

1. Todas as tautologias
2. $(\alpha;\beta)A \Leftrightarrow (\alpha)(\beta)A$
3. $(\alpha+\beta)A \Leftrightarrow (\alpha)A \vee (\beta)A$
4. $(\langle\alpha^*\rangle)A \Leftrightarrow A \vee (\alpha)(\langle\alpha^*\rangle)A$
5. $(\langle A\rangle)B \Leftrightarrow A \wedge B$

4.6.2 Regras de Inferência

1. Modus Ponens:

$$\frac{A \quad A \Rightarrow B}{B}$$

2. Monotonicidade:

$$\frac{A \Rightarrow B}{(\alpha)A \Rightarrow (\alpha)B}$$

3. Indução de Bar:

$$\frac{(\alpha)A \Rightarrow A}{(\langle\alpha^*\rangle)A \Rightarrow A}$$

A corretude e a completude destes axiomas e regras é apresentada em [PA85]. Um procedimento elementar de decisão para a lógica sem o dual é conseqüência da completude. Não se sabe se há um procedimento elementar de decisão para a Lógica de Jogos completa, com o operador dual.

4.7 Jogos de Muitas Pessoas

Conforme visto em [PA85] é possível estender a GL para falar de jogos com várias pessoas, que em geral, são mais complexos que os jogos de duas pessoas. Suponha que temos as ações $a, b, c, ...$ e pessoas $p_1, p_2, ..., p_n$. Suponha que eles estão executando um procedimento (que é jogar um jogo de muita pessoas), onde determinadas pessoas fazem certas ações em certos estados. Exemplo: (p_1 faz a); (p_2 faz b); (p_3 decide se p_1 ou p_2 faz c). Agora suponha que num certo estado uma ação a deve ser feita e que a pessoa p gostaria que a fórmula A fosse válida depois disso. Claramente, se p executa a pessoalmente então a condição $\langle a \rangle A$ é suficiente. Contudo, se outra pessoa executa a, então p necessita que $[a]A$ seja verdade para ter certeza que A será verdade. A questão é: "É possível que, dadas as pessoas $p_1, p_2, ..., p_m$ e objetivos $A_1, A_2, ..., A_m$ fazer um procedimento tal que p_i possa estar certo de alcançar A_i ao fim do procedimento, em detrimento do que os outros farão?". Discutiremos sintaxe e semântica da lógica envolvida, e posteriormente veremos um exemplo.

Definição 4.7.1 *(Sintaxe para Lógica de Jogos de Muitas Pessoas)*

Assumimos, como em PDL, uma coleção de ações atômicas e um conjunto $P_1, P_2, ..., P_k$ de proposições. Assim sendo, os jogos de m pessoas e as fórmulas são definidas como:

1. *Cada P_i é uma fórmula.*
2. *Se A e B são fórmulas, então $A \vee B$ e $\neg A$ também são.*
3. *Se A é uma fórmula, $i \leq m$ e α é um jogo de m pessoas, então $(\alpha, i)A$ é uma fórmula. (A pessoa i tem uma estratégia vencedora, jogando α, de assegurar que A seja válido quando e se o jogo terminar).*
4. *Para cada $i \leq m$, $j \leq m$, (α_i, j) é um jogo.(A pessoa j faz a ação α_i)*
5. *Se α e β são jogos, então $\alpha; \beta$ é um jogo e para cada $i \leq m$, $\alpha \vee_i \beta$ é um jogo. (A pessoa i escolhe se o jogo α ou o jogo β será jogado).*

6. *Se α é um jogo, então para cada $i \leq m$, $\langle \alpha, i, * \rangle$ é um jogo. (O jogo α é repetidamente jogado até que o jogador i decida parar e seguir para o próximo passo, se existir algum).*

7. *Se α e β são jogos e A é uma fórmula então* `If` A `then` α `else` β *é um jogo, e* `While` A `play` α *também é um jogo.*

Chamaremos um jogo de $*$-livre se o operador $*$ não ocorre nele. Todo jogo $*$-livre é limitado em tamanho. Um modelo para essa lógica será simplesmente um modelo PDL. A semântica para o jogo de m pessoas pode ser obtida de forma análoga à de jogo para duas pessoas; a única cláusula indutiva de interesse é definir os valores da fórmula $(\alpha, i)A$ em função dos valores de A. Se podemos definir o transformador de predicados $\rho(a, i)$ então podemos indutivamente definir $\pi((\alpha, i)A)$ como $\rho(\alpha, i)\pi(A)$. Contudo, vemos facilmente que $\rho(\alpha, i)$ é $\rho(a')$ onde a' é o jogo de duas pessoas onde o jogador 1 corresponde a i e todos os outros jogadores correspondem ao jogador 2. No caso particular onde $j \neq i$, (α, i) é $\langle \alpha \rangle$, (α, j) é $[\alpha]$, $\vee_i$ é $\vee$ e $\wedge_j$ é $\wedge$. Valem os seguintes axiomas ($j \neq i$):

1. $(\alpha; \beta, i)A \Leftrightarrow (\alpha, i)(\beta, i)A$
2. $(\alpha \vee_i \beta, i)A \Leftrightarrow (\alpha, i)A \vee (\beta, i)A$
3. $(\alpha \vee_j \beta, i)A \Leftrightarrow (\alpha, i)A \wedge (\beta, i)A$
4. $((\alpha, i, *), i)A \Leftrightarrow A \vee (\alpha, i)((\alpha, i, *), i)A$
5. $((\alpha, j, *), i)A \Leftrightarrow A \wedge (\alpha, i)((\alpha, i, *), i)A$
6. (`If` B `then` α `else` $\beta, i)A \Leftrightarrow (B \wedge (\alpha, i)A) \vee (\neg B \wedge (\beta, i)A)$
7. (`While` B `do` α",$i)A \Leftrightarrow (\neg B \wedge A) \vee (B \wedge (\alpha, i)($`While` B `do` $\alpha, i)A)$

A conexão de jogos de muitas pessoas com PDL é feita através dos axiomas seguintes:

- $((\alpha, i), i)A \Leftrightarrow \langle \alpha \rangle A$

- $((\alpha, j), i)A \Leftrightarrow [\alpha]A$

É imediato que os jogos de muitas pessoas que são $*$-livres podem ser traduzidos para PDL, conseqüentemente nos dando um procedimento de decisão. Podemos também usar a notação PDL e a notação de jogos de muitas pessoas indistintamente em algumas circunstâncias restritas.

As seguintes regras também valem:

1. $$\frac{A \quad A \Rightarrow B}{(\alpha, i)A \Rightarrow (\alpha, i)B}$$

2. $$\frac{A \wedge B \Rightarrow (\alpha, i)A}{A \Rightarrow (While\, B\, do\, \alpha, i)(A \wedge \neg B)}$$

3. $$\frac{A \Rightarrow (\alpha, i)A}{A \Rightarrow ((\alpha, j, *), i)A}$$

4. $$\frac{(\alpha, i)A \Rightarrow A}{((\alpha, i, *), i)A \Rightarrow A}$$

Estas duas últimas regras são generalizações de teoria dos jogos das correspondentes regras para $[]$ e $\langle\rangle$.

4.7.1 Exemplo: Algoritmo para corte do bolo

Um exemplo da aplicação da Lógica de Jogos para Muitas Pessoas baseado no método de divisão "Eu corto e você escolhe" foi apresentado em [PA85].

Veremos a seguir um algoritmo para n pessoas dividindo algo, como por exemplo um bolo. Da mesma forma que no caso de "Eu corto e você escolhe" para duas pessoas, a pessoa que divide tem um forte incentivo para fazer pedaços de tamanho igual.

A primeira pessoa p_1 corta um pedaço que ela afirma ser de tamanho justo. Então o pedaço passa de mão em mão no grupo sendo inspecionado, em turnos, pelas pessoas p_2, p_3, etc. Quem achar que o pedaço não é muito grande, apenas o

passa adiante conforme o recebeu. Quem achar que ele é muito grande pode reduzi-lo, recolocando o excedente junto à parte principal do bolo ainda não dividida. Após ser inspecionado pela última pessoa p_n, quem reduziu o pedaço por último o receberá. Se ninguém o tiver reduzido, p_1 fica com o pedaço. Em qualquer um dos casos alguém fica com o pedaço e o algoritmo continua com o restante do bolo e $n-1$ participantes. Note que incidentalmente o algoritmo descrito anteriormente do "Eu corto e você escolhe", não corresponde a este algoritmo mais geral no caso $n = 2$.

Partimos do ponto de vista que um algoritmo desse tipo é na verdade um jogo para m pessoas que funciona se e somente se cada jogador tem uma estratégia vencedora para atingir um resultado justo. Podemos utilizar a lógica de jogos para muitas pessoas para mostrar sua corretude.

Um estado consistirá nos valores para $n+2$ variáveis. A variável m corresponde ao valor da parte principal do bolo, a variável x será o pedaço sendo considerado e para $i = 1$ até n, x_i terá o valor correspondente ao pedaço da pessoa i, se é que ele já possui algum pedaço. O algoritmo possui três ações básicas:

- c (cortar): Cortar um pedaço de m e atribui-lo a x.
  A ação c é executada apenas se $x = 0$.

- r (reduz): Transfere uma parte do pedaço inspecionado x para a parte principal m do bolo.

- a_i (atribui): Atribui o pedaço x para a pessoa p_i. Assim sendo, a_i equivale a $(x_i, x) := (x, 0)$

Os predicados básicos são $F(u, k)$ onde u é um pedaço e $k \leq n$. Significa que o pedaço u é grande o suficiente para k pessoas. $F(u)$ abrevia $F(u, 1)$ e F_i abrevia $F(x_i)$, o que significa que o pedaço atribuído à pessoa p_i é grande o suficiente para ela.

Assumimos as proposições seguintes para todo $k \leq n$ e todo jogador $i \leq n$:

1. $F(m, k) \Rightarrow \langle c \rangle (F(m, k-1) \wedge F(x))$: Se o pedaço principal é suficiente para k pessoas então é possível retirar um pedaço justo, deixando o bastante para

$k-1$ pessoas. Note que (1) é equivalente à seguinte proposição na lógica de jogos:

1'. $F(m,k) \Rightarrow (c,i)(F(m,k-1) \wedge F(x))$

O seguinte também vale:

2. $F(m,k) \Rightarrow [r*]F(m,k)$: Se a parte principal é suficiente para k pessoas, continua sendo ao acrescentar algo mais a ela.

3. $F(m,k) \Rightarrow [c][r*](F(m,k-1) \vee \langle r \rangle (F(m,k-1) \wedge F(x)))$: Se o pedaço principal é grande o suficiente para k pessoas, então após o corte inicial e várias reduções ou ele é grande o bastante para $k-1$ pessoas, ou então uma redução fará m grande o suficiente para $k-1$ pessoas e deixará x num tamanho justo.

4. $F(x) \Rightarrow [a_i]F_i$: Se x é um pedaço de tamanho justo, então a pessoa que o receber o achará de tamanho justo.

Há algumas afirmativas relevantes, por exemplo, que r e c podem apenas afetar declarações onde m ou x ocorrem. Assumimos ainda que $F(m,n)$ é verdade no início.

O algoritmo principal consiste de n ciclos. Em cada ciclo uma pessoa recebe um pedaço, de forma que ninguém receba mais de um pedaço. Mostramos agora que cada pessoa p_i tem uma estratégia vencedora de forma que, se após o $k-$ésimo ciclo ela ainda estiver no jogo então $F(m,n-k)$, e se ela receber um pedaço então F_i é válido. Isso vale no início uma vez que $k=0$, $F(m,n)$ vale de k a $k+1$. Assumimos por hipótese de indução que $F(m,n-k)$ vale neste ponto.

Se $i=1$ então como p_1 (ou quem quer que seja) faz o corte, então por (1) e (1′) ela pode atingir $F(m,n-k-1) \wedge F(x)$. Se ninguém faz r, ela pega x e F_1 vale já que x não muda. Se alguém faz r, então por (2), $F(m,n-k-1)$ ainda será válido e isso é suficiente porque ele ainda estará participando no próximo turno.

Vamos considerar apenas uma entre as outras pessoas. A última pessoa p_1 a fazer r (se houver alguém que faça r) pode por (3) atingir $F(x)$ e assim quando x

é dado a ele, F_1 valerá. Todos os outros casos são análogos, e o passo de indução segue. Tomando $k = n$ vemos que todo p_i tem a oportunidade de atingir F_i.

Capítulo 5

Comunicação em Lógica Dinâmica Concorrente

Os modelos tradicionais de jogos apresentados no capítulo 2 representam todos os cursos que o jogo pode seguir dadas as escolhas dos jogadores. Basicamente, este é o arcabouço de qualquer processo interativo, e a interação é a principal característica dos processos sociais.

A maioria dos processos sociais interessantes, especialmente os que se dão na Internet, sofrem alta influencia da comunicação entre os agentes participantes. Nos modelos de jogos, as preferências dos jogadores são a única influência considerada nas escolhas das ações que determinam o curso do jogo. Tais preferências são relações estáticas pré-definidas. A comunicação, ou qualquer outra forma de influenciar dinamicamente o processo de decisão racional dos jogadores não é contemplada.

Neste capítulo apresentamos uma linguagem lógica e o modelo correspondente que permite representar comunicação entre dois processos executando em paralelo: *channel-Concurrent Propositional Dynamic Logic* (*channel-CPDL*), proposta por David Peleg em [PE87b].

channel-CPDL é uma dentre as extensões da Lógica Dinâmica Concorrente Proposicional (Concurrent Propositional Dynamic Logic, *CPDL*) que visam incorporar mecanismos de comunicação aos programas concorrentes.

CPDL foi introduzida em [PE87b], como uma extensão da Lógica Dinâmica [PE87a], para modelar programas concorrentes. Todavia, os processos concorrentes modelados em *CPDL* são isolados (não se comunicam) e não sincronizados.

Veremos a seguir uma revisão de *CPDL* e de sua extensão *channel-CPDL*, e mostraremos através de um exemplo como *channel-CPDL* pode ser utilizada para modelar comunicação em processos interativos.

5.1 Lógica Dinâmica Proposicional Concorrente

Lógica Dinâmica Concorrente Proposicional [PE87a] é uma extensão da Lógica Dinâmica regular que tenta prover uma estrutura para raciocinar acerca de programas concorrentes no modelo *and/or*, um dos mais estudados em computação concorrente. *CDL* possui a maioria das propriedades desejáveis de lógica dinâmica, sendo capaz de modelar e discutir concorrência.

Podemos ilustrar um processo no modelo *and/or* como uma árvore cujas arestas representam ações atômicas (árvore *and/or*).

O não-determinismo é representado por um nó com várias ramificações, onde o processo (agente) deve escolher e executar apenas uma dentre as possíveis continuações.

A concorrência também é representada por um nó com várias ramificações, porém neste caso o processo deve executar todas as possíveis continuações em paralelo.

5.1.1 Sintaxe e Semântica

A sintaxe de *CPDL* é a mesma de *PDL*, com a adição de um operador de concorrência $\cap$ nos programas.

Definição 5.1.1 *(Sintaxe de* CPDL*)*

Seja $\Phi = \{P_1, \cdots P_m\}$ *um conjunto de fórmulas atômicas e* $\Psi = \{a_1, \cdots a_n\}$ *um conjunto de programas atômicos:*

1. *Todo* P_i *é uma fórmula;*
2. *Se* A *e* B *são fórmulas e* α *é uma programa então* $A \vee B$, $\neg A$ *e* $\langle\alpha\rangle A$ *são fórmulas;*
3. *Todo* a_i *é um programa;*

4. *Se α e β são programas e A é uma fórmula, então $\alpha \cup \beta$, $\alpha \cap \beta$, $\alpha;\beta$, α^* e $A?$ são programas.*

Definição 5.1.2 *(Modelo para* CPDL*)*

Um modelo para CPDL *é uma tupla $\mathcal{M} =< S, \pi, \rho >$ onde:*

- *S é um conjunto de estados;*
- *π atribui um subconjunto $\pi(P)$ de S a toda fórmula atômica P;*
- *ρ associa um subconjunto $\rho(a) \subseteq S \times 2^S$ para todo programa atômico a.*

Intuitivamente, $(s, U) \in \rho(\alpha)$ para $s \in S$ e $U \subseteq S$ se α pode ser executado a partir de s e atingir todos os estados de U em paralelo. Deste modo, a fórmula $\langle\alpha\rangle A$ vale num estado s se e somente se existe um conjunto $U \subseteq S$ tal que $(s, U) \in \rho(\alpha)$ e cada estado em U satisfaz A.

Definição 5.1.3 *(Extensão de π e ρ para fórmulas e programas mais complexos)*

- $\pi(A \vee B) = \pi(A) \cup \pi(B)$
- $\pi(\neg A) = S - \pi(A)$
- $\pi(\langle\alpha\rangle A) = \{s \mid \exists U((s, U) \in \rho(\alpha) \wedge U \subseteq \pi(A))\}$
- $\rho(A?) = \{(s, \{s\}) \mid s \in \pi(A)\}$
- $\rho(\alpha \cup \beta) = \rho(\alpha) \cup \rho(\beta)$
- $\rho(\alpha \cap \beta) = \{(s, U) \mid \exists V, W((s, V) \in \rho(\alpha) \wedge (s, W) \in \rho(\beta) \wedge U = V \cup W)\}$
- $\rho(\alpha;\beta) = \rho(\alpha).\rho(\beta)$
- $\rho(\alpha^*) = min\{R \mid R \subseteq S \times 2^S \wedge R = \rho(true?) \cup \rho(\alpha).R\}$

onde $R_1.R_2$ é definido (para qualquer $R_1, R_2 \subseteq S \times 2^S$) como:

$$\{(s, U) \mid \exists s_1, U_1, s_2, U_2, ...((s, \{s_1, s_2, ...\}) \in R_1 \wedge \forall i(s_i, U_i) \in R_2 \wedge U = \bigcup_i U_i)\}$$

A definição de $\langle\alpha\rangle A$ diz que existe uma computação de α tal que A vale em todos os seus estados finais.

5.2 Modelo de Computação em Árvore

Veremos agora uma segunda construção para interpretação de programas, equivalente ao modelo definido na seção anterior porém com algumas alterações que constituem as bases técnicas para estender *CPDL* de forma a comportar comunicação. Este modelo é baseado no conceito de *trec* (*tree-like computation*).

Definição 5.2.1 *(*Seq*)*

Uma seq *é um programa* PDL *determinístico que não contém* $\cup$ *ou* $*$.

Intuitivamente, a *seq* é uma seqüencia de programas atômicos e testes concatenados. As *seqs* descrevem execuções determinísticas de programas regulares.

As *trecs* são o análogo das *seqs* para programas concorrentes. Uma *trec* descreve uma execução determinística específica de um programa que pode ou não ser concorrente.

Definição 5.2.2 *(*Trec*)*

Uma trec *é um programa* CPDL *que não possui* $\cup$ *ou* $*$.

O conjunto de *trecs* pode ser definido como o fechamento de programas atômicos e testes sob concatenação e $\cap$.

Definição 5.2.3 *(*Trec *na forma de árvore)*

Dizemos que uma trec α *está na forma de árvore (tree form) se ela pode ser indutivamente definida por:*

1. $true?$ *é uma* trec
2. *se* α *e* β *são* trecs*, a é um programa atômico e A é uma fórmula, então* $\alpha \cap \beta$*,* $a;\alpha$ *e* $A?;\alpha$ *são* trecs*.*

Uma *trec* nesta forma pode ser representada como uma árvore cujos arcos são rotulados pelos programas atômicos e testes da *trec* e cada $\cap$ corresponde a uma divisão em dois ramos. As folhas desta árvore correspondem aos estados finais dos diferentes processos paralelos gerados pela *trec*.

Toda trec pode ser posta na forma de árvore [PE87b][1].

A semântica de um programa *PDL* seqüencial α não determinístico pode ser definida com base em um conjunto $\tau(\alpha)$ de *seqs* determinísticas, descrevendo as possíveis execuções de α tal que $\rho(\alpha) = \bigcup_{\beta \in \tau(\alpha)} \rho(\beta)$ [PE87b].

Analogamente, a semântica de um programa concorrente α não determinístico pode ser baseada numa coleção de *trecs*, determinísticas por definição. Todo programa α está associado a um conjunto $\tau(\alpha)$ de *trecs*, que representam todas as suas execuções possíveis [2].

Definição 5.2.4 *(Conjunto de* trecs *de um programa)*

O conjunto $\tau(\alpha)$ *de* trecs *de um programa* α *é definido por:*

1. $\tau(a_i) = \{a_i\}$, *para programas atômicos* a_i*;*
2. $\tau(A?) = \{A?\}$, *para testes* $A?$*;*
3. $\tau(\alpha \cup \beta) = \tau(\alpha) \cup \tau(\beta)$*;*
4. $\tau(\alpha \cap \beta) = \{\gamma \cap \delta \mid \gamma \in \tau(\alpha) \text{ e } \delta \in \tau(\beta)\}$*;*
5. $\tau(\alpha;\beta) = \{\theta \mid \exists \gamma, \delta_1, ..., \delta_k (\gamma \in \tau(\alpha), \delta_1, ..., \delta_k \in \tau(\beta),$ γ *tem* k *folhas, e* θ *é obtido concatenando cada* δ_i *a uma das folhas de* $\gamma)\}$*;*
6. $\tau(\alpha^*)$ *é o conjunto minimal construído pelas seguintes regras:*
   (a) $\texttt{true}? \in \tau(\alpha^*)$*;*
   (b) *Para todo* γ, δ *e* θ*, se* $\gamma \in \tau(\alpha^*)$*,* $\delta \in \tau(\alpha)$*, e* θ *é obtido concatenando* δ *a uma das folhas de* γ*, então* $\theta \in \tau(\alpha^*)$*.*

[1] Se α e β estão na forma de árvore, então $\alpha;\beta$ (não necessariamente na forma de árvore) pode ser transformada para a forma de árvore concatenando a árvore de β em toda folha da árvore de α. Isto corresponde à repetidas aplicações da regra de transformação: $(\theta \cap \gamma);\delta \Rightarrow \theta;\delta \cap \gamma;\delta$ para qualquer subprograma de $\alpha;\beta$, sempre que possível. Esta regra é válida para programas *CPDL* em geral, e também com $\cup$ no lugar de $\cap$. Porém se δ está à esquerda dos parênteses, então ambas as transformações ($\cup$ ou $\cap$) são inválidas, e a *trec* não pode continuar sendo decomposta em um conjunto de *seqs* paralelas isoladas.

[2] A definição do conjunto $\tau(\alpha)$ de um programa α é uma transformação entre objetos sintáticos, e não inclui interpretação para as primitivas da linguagem.

$\tau(\alpha)$ contém somente *trecs* na forma de árvore e, para qualquer *trec* α, $\tau(\alpha)$ contém algumas *trecs* na forma de árvore equivalentes a α.

Podemos agora dar outra definição para $\rho(\alpha)$ em *CPDL*, equivalente à original.

Definição 5.2.5 *(Função $\rho(\alpha)$ para um programa α)*

Seja $\rho'(\beta)$ para trecs *na forma de árvore β definido segundo a definição original.* $\rho(\alpha) = \bigcup_{\beta \in \tau(\alpha)} \rho'(\beta)$*, para todo programa α.*

5.3 *Channel-CPDL*

Na semântica de *trecs*, um programa *CPDL* pode ser descrito como uma árvore onde não existem ligações entre os ramos. A execução de um programa *CPDL* se inicia em um certo estado s, e avança através de uma seqüencia de estados ou se divide em dois ou mais processos paralelos. Uma vez que a computação se divide, cada um dos processos concorrentes corresponde a um ramo independente na árvore de execução, e a computação avança separadamente em cada ramo, sem sincronização.

Não podemos fazer uso de uma semântica como esta para viabilizar a comunicação entre processos, tendo em vista que a comunicação impõe alguma forma de ligação entre os processos.

As principais funções a serem exercidas pela comunicação num ambiente concorrente são [PE87b]:

- Sincronização de processos;
- Troca de informação entre processos;
- Transmissão de informações, mensagens e resultados finais para algum processo principal.

A comunicação em *Channel-CPDL* é feita através de canais estabelecidos entre dois processos, de forma bem próxima ao que ocorre em várias linguagens concorrentes, e a notação utilizada é semelhante à de *CCS* [M80].

Dadas as necessidades de interação entre processos de ramos distintos em uma trec, [PE87b] propõe para *Channel-CPDL* uma semântica de super-estados. Os

super-estados representam "cortes transversais" na árvore de computação. As definições semânticas envolvem simultaneamente todos os estados no corte e todos os programas operando em diferentes ramos.

Os cortes representam conjuntos de estados locais, sendo que qualquer configuração de estados é permitida (não-determinismo de concorrência), desde que restrições de sincronismo impostas pelos mecanismos de comunicação não sejam violadas.

5.3.1 Sintaxe e Semântica

Definição 5.3.1 *(Sintaxe de* Channel-CPDL*)*

A sintaxe é mesma de CPDL*, porém inclui elementos específicos para a comunicação:*

- *A linguagem possui uma coleção de canais de comunicação* $\{C_i\}$*;*
- *Em adição a programas atômicos e testes, as seguintes operações atômicas são permitidas:* $C!0$*,* $C!1$*,* $C?0$*,* $C?1$*.*

As operações $C!0$, $C!1$ correspondem respectivamente ao envio de um bit com valor 0 e envio de um bit com valor 1 através do canal C. Analogamente, as operações $C?0$ e $C?1$ correspondem ao recebimento de um bit com valor 0 ou 1 em C. Para que um processo possa realizar uma operação de envio é necessário que outro processo realize simultaneamente a respectiva operação de recebimento.

Como cada canal de comunicação C liga apenas dois processos, ocorre que uma operação de comunicação, quando realizada, sincroniza os dois processos envolvidos; se um processo tenta realizar a ação de envio antes de o respectivo processo destinatário da mensagem realizar a ação de recebimento, o primeiro fica bloqueado aguardando que o segundo faça o recebimento da mensagem, e vice versa (comunicação bloqueante).

Os comandos de comunicação podem ser usados por um processo para transferir, por exemplo, o valor de uma proposição P no seu estado corrente (pela execução de $P?; C!1 \cup (\neg P)?; C!0$) e outro processo pode-se tomar uma decisão de acordo com o conteúdo de uma determinada mensagem ($C?0; \alpha \cup C?1; \beta$).

Uma definição formal de semântica para *channel-CPDL* torna-se mais complicada que a de *CPDL*, por ter de lidar com super-estados (conjuntos de estados simultâneos) ao invés de estados individuais de cada ramo de execução.

Esta restrição é uma conseqüência da introdução dos operadores de comunicação, que por serem realizados simultaneamente impedem que sua avaliação seja feita em separado sem introduzir novos objetos semânticos. Se existem canais em comum entre diferentes ramos, pode ser que haja estados a partir dos quais os programas não possam ser executados em separado. Assim sendo, é necessário que as regras semânticas considerem os processos concorrentes que estejam sendo executados em paralelo. Isto é feito através dos super-processos.

O modelo proposto para *CPDL* provê uma interpretação $\rho(a) \subseteq S \times S$ para programas atômicos a, por isso nos limitamos a modelos seqüenciais [PE87a]. Essa definição deve ser estendida para super-processos.

Um super-estado é denotado por $\overline{s} = [s_1 \mid ... \mid s_n]$ (quando $n = 1$ identificamos s_1 com $[s_1]$). Quando $\overline{s}$ é conhecido podemos nos referir diretamente a uma porção dele, como por exemplo $[s_i \mid ... \mid s_{i+k}]$ como $\overline{s}(i, i+k)$. Dois super estados podem ser combinados para formar um super estado maior: $[s_1 \mid ... \mid s_n] \mid [q_1 \mid ... \mid q_m] = [s_1 \mid ... \mid s_n \mid q_1 \mid ... \mid q_m]$. O tamanho de $\overline{s}$, denotado por $\mid \overline{s} \mid$, é o número n de estados paralelos em $\overline{s}$.

Um super-processo é denotado por $\overline{\alpha} = [\alpha_1 \mid ... \mid \alpha_n]$ (identificamos α_1 com $[\alpha_1]$). Podemos descrever uma porção dele por $\overline{\alpha}(i, i+k)$. Para um super processo $\overline{\alpha} = [\alpha_1 \mid ... \mid \alpha_n]$, $\rho(\overline{\alpha}) \subseteq S^n \times (\bigcup_{i \geq n} S^i)$, isto é, $\rho(\overline{\alpha})$ consiste de tuplas $(\overline{s}, \overline{s}')$ com $\mid \overline{s} \mid = n$ e $\mid \overline{s}' \mid \geq n$.

Conservamos a natureza em árvore dos programas determinísticos (as trecs), e a forma de decomposição de programas gerais em trecs. Assim mantemos as transformações descritas na seção 5.2 e obtemos também as seguintes:

1. $(\theta \cap \gamma); \delta \Rightarrow \theta; \delta \cap \gamma; \delta$ e

2. $(\theta \cup \gamma); \delta \Rightarrow \theta; \delta \cup \gamma; \delta$

As definições do conjunto de trecs $\tau(\alpha)$ associado a cada programa α são as mesmas apresentadas em 5.2, com adição dos operadores de comunicação. Defini-

mos agora $\rho(\overline{\alpha})$ para um super-processo $\overline{\alpha}$ consistindo somente de trecs na forma de árvore. Neste ponto a interpretação semântica das operações do canal de comunicação obrigam algumas conexões de sincronização entre ramos separados das trecs, portanto esses ramos não são mais independentes e as trecs perdem sua natureza de árvore.

1. $\rho(true?) = \{(s, s) \mid s \in S\}$.

2. Se $\alpha_i = a; \beta, (s_i, s^{'}) \in \rho(a)$ e $(\overline{s}(1, i-1) \mid s^{'} \mid (\overline{s}(i+1, n), \overline{r}) \in \rho((\overline{\alpha}(1, i-1) \mid \beta \mid (\overline{\alpha}(i+1, n))$ então $(\overline{s}, \overline{r}) \in \rho(\overline{\alpha})$.

3. Se $\alpha_i = C!1; \beta_1$, $\alpha_j = C?1; \beta_2$, $j > i$ e $(\overline{s}, \overline{r}) \in \rho(\overline{\alpha}(1, i-1) \mid \beta_1 \mid \overline{\alpha}(i+1, j-1) \mid \beta_2 \mid \overline{\alpha}(j+1, n))$ então $(\overline{s}, \overline{r}) \in \rho(\overline{\alpha})$ (similarmente quando $j < i$, e quando a mensagem é 0 (zero) em ambos os lados)

4. Se $\alpha_i = A?; \beta$, $s_i \in \pi(A)$ e $(\overline{s}, \overline{r}) \in \rho(\overline{\alpha}(1, i-1) \mid \beta \mid \overline{\alpha}(i+1, n))$ então $(\overline{s}, \overline{r}) \in \rho(\overline{\alpha})$.

5. Se $\alpha_i = \beta \cap \gamma$ e $(\overline{s}(1, i-1) \mid s_i \mid s_i \mid \overline{s}(i+1, n), \overline{r}) \in \rho(\overline{\alpha}(1, i-1) \mid \beta \mid \gamma \mid \overline{\alpha}(i+1, n))$ então $(\overline{s}, \overline{r}) \in \rho(\overline{\alpha})$.

Para todo $\overline{\alpha}$, $\rho(\overline{\alpha})$ contém precisamente aqueles pares $(\overline{s}, \overline{r})$ introduzidos pelas regras acima.

A definição de ρ é estendida para qualquer super processo $\overline{\alpha}$ por:

$$\rho([\alpha_1 \mid ... \mid \alpha_n]) = \bigcup_{\beta_i \in \tau(\alpha_i)} \rho([\beta_{i_1} \mid ... \mid \beta_{i_n}])$$

A interpretação de fórmulas tem que ser modificada de acordo com:

$$\pi(\langle\alpha\rangle A) = \{s^{'} \mid \exists \overline{s}(\overline{s} = [s_1 \mid ... \mid s_n], (s^{'}, \overline{s}) \in \rho(\alpha)\, e\, \forall i, 1 \leq i \leq n(s_i \in \pi(A)))\}$$

A definição semântica de programas apresentada para *channel-CPDL* difere da de *CPDL*; em *channel-CPDL* interpretamos a execução de um programa como partindo de um único estado para um multi-conjunto de estados. Assim permitimos que um programa a se divida e atinja dois ou mais estados em sua execução. A escolha de diferentes semânticas porém não afeta a interpretação de fórmulas. É possível demonstrar que:

1. $\rho(\alpha) \subseteq \rho_{mult}(\alpha)$ para todo programa α, onde $set(U)$ é multi-conjunto de U

2. $(s, U) \in \rho_{mult}(\alpha) \Rightarrow \exists U'(U' \subseteq set(U), (s, U') \in \rho(\alpha)$ para todo programa α

Essas duas observações são úteis para provar que para qualquer fórmula A, $\pi(A) = \pi_{mult}(A)$.

Ressaltamos que em *CPDL* dois programas α e β em contextos separados ($\langle\alpha\rangle A$ e $\langle\beta\rangle B$) estão totalmente desconectados e nenhuma comunicação é possível entre eles. Além disso, mesmo a fórmula anexada A não tem nenhum modo de se referir aos canais presentes em α, e o mesmo se aplica para testes em α. Deste modo, axiomas *CPDL* válidos tais como $\langle\alpha \cap \beta\rangle A = \langle\alpha\rangle A \wedge \langle\beta\rangle A$ e $\langle\alpha;\beta\rangle A = \langle\alpha\rangle\langle\beta\rangle A$ não valem em *channel-CPDL*. Por exemplo: $\langle C!0 \cap C?0\rangle\texttt{true}$ é sempre verdade, enquanto que $\langle C!0\rangle\texttt{true} \wedge \langle C?0\rangle\texttt{true}$ é sempre falso.

5.3.2 Exemplo: Contagem de folha

Este exemplo foi retirado de [PE87b].

Seja a fórmula: $\texttt{even}_1$: $\langle((a \cap b)^*;\texttt{leaf}?;((\neg P)? \cup (P?; C!1))) \cap (C?1; C?1)^*\rangle\texttt{true}$, onde `leaf`: $\neg\langle a \cup b\rangle\texttt{true}$

Considerando modelos na forma de árvores binárias completas finitas (onde cada nó tem apenas zero ou dois ramos, a e b), a fórmula $\texttt{even}_1$ contempla exatamente os modelos nos quais existe um número par de folhas satisfazendo P.

O programa principal de $\texttt{even}_1$ se divide em vários processos paralelos, um para cada ramo da árvore. Cada um desses processos, ao atingir uma folha, testa P e envia uma mensagem se P é verdadeiro. Um processo aparte recebe essas mensagens e computa a paridade do número de mensagens recebidas.

Para fazer esta contagem também sobre árvores binárias parciais (nas quais um nó pode ter zero, um, ou dois filhos), substituímos o subprograma $(a \cap b)^*$ por:

$((\langle a\rangle\texttt{true} \wedge \langle b\rangle\texttt{true})?; (a \cap b) \cup (\langle a\rangle\texttt{true} \wedge \neg\langle b\rangle\texttt{true})?; a \cup (\neg\langle a\rangle\texttt{true} \wedge \langle b\rangle\texttt{true})?; b)^*$

Este programa, quando aplicado a uma árvore binária finita a/b, conta o número de caminhos distintos até as folhas que satisfazem P.

Estes exemplos demonstram processos independentes enviando mensagens (relatórios de execução) para um processo principal (no caso, o que centraliza a contagem). Outros exemplos evidenciando funções de sincronização podem ser vistos em [PE87b].

Capítulo 6

Lógica Modal para Coalisão em Jogos

Pensando em grupos de agentes e ações em conjunto, Marc Pauly propôs em [MP02] uma lógica modal para representar o que grupos de agentes podem atingir ao agir coletivamente, unindo seus esforços em ações conjuntas. Esta ação coletiva feita por um grupo específico corresponde ao que chamamos de coalisão.

A forma natural de modelar ações de agentes é através de um modelo de estados ligados por ações, como foi feito no modelo da Lógica de Jogos no capítulo 4. Um cenário com vários jogadores corresponde a uma estrutura Kripke com várias relações de acessibilidade entre os estados, cada uma delas correspondendo às ações de cada jogador.

Este modelo porém possui duas propriedades não desejadas em um modelo que vise captar o panorama de um grupo de pessoas interagindo: as ações de cada pessoa são independentes (visto que as relações são caracterizadas apenas por pares de estados origem e destino), e não há ações realizadas simultaneamente por agentes distintos em um mesmo estado, ou seja, as ações se sucedem, uma a uma, no decorrer do tempo.

A Lógica de Coalisão (*CL*, *Coalition Logic*) [MP02] baseia-se em um modelo que associa uma tupla de ações às transformações de estado, onde esta tupla pode ter uma ou mais componentes, e cada componente corresponde à ação de um jogador, representando a transformação de estado obtida pela ação conjunta dos jogadores participantes da coalisão representada na tupla.

Um modelo mais genérico que o de Kripke será introduzido. Para representar a coalisão, um jogo estratégico é associado a cada estado do modelo, onde os resultados do jogo são estados do modelo novamente. É possível demonstrar que este modelo corresponde a formas de jogo extensivo com ações simultâneas [OR94], o que permite manipular as propriedades de simultaneidade e associação entre jogadores conforme desejado.

A Lógica de Coalisão pode ser vista como uma generalização da Lógica de Jogos (GL) introduzida por [PA85] e apresentada no capítulo 4. A particularidade do ponto de vista da Lógica de Coalisão é abandonar a pressuposição de determinismo e estender a linguagem de jogadores individuais para grupos.

6.1 Um Modelo para interação

No modelo desejado, em um estado qualquer as ações feitas em conjunto por subgrupos de agentes determinam o próximo estado. Para atender a essa generalização, associamos a cada estado um jogo estratégico.

Definição 6.1.1 *(Forma de um Jogo Estratégico)*

A Forma de um Jogo Estratégico $\mathcal{F} = (N, \Sigma_i | i \in N, o, S)$ *consiste de um conjunto* N *não vazio de agentes ou jogadores, um conjunto não vazio de estratégias ou ações* Σ_i *para cada jogador* $i \in N$*, um conjunto não vazio de estados de resultado* S *e uma função de resultado* $o : \Pi_{i \in N} \Sigma_i \rightarrow S$ *que associa a cada tupla de estratégias dos jogadores (ou perfil de estratégias) um estado de resultado em* S*.*

É importante notar que em teoria de jogos os jogos estratégicos (definidos no capítulo 2) englobam uma relação de preferência entre os resultados $\preceq_i \subseteq S \times S$ para cada jogador $i \in N$ ([OR94], [KB92]). A definição corresponde apenas a formas de jogo, às quais podemos adicionar relações de preferência se desejado.

Para simplificar a notação, $\sigma_C := (\sigma_i)_{i \in C}$ irá denotar a tupla de estratégias para a coalisão $C \subseteq N$ que consiste no jogador i escolhendo a estratégia $\sigma_i \in \Sigma_i$. Assim sendo, dadas as tuplas de estratégias σ_C e $\sigma_{\overline{C}}$ (onde $\overline{C} := N \backslash C$), $o(\sigma_C, \sigma_{\overline{C}})$ denota o estado resultante associado com o perfil de estratégias induzido por σ_C e $\sigma_{\overline{C}}$.

Definição 6.1.2 *(Estrutura de Jogo)*

Seja Γ_S^N o conjunto de todos os jogos estratégicos entre os jogadores do conjunto N sobre o o conjunto de estados S. Definimos uma Estrutura de Jogo para os jogadores em N como um par (S, γ) onde S é um conjunto não vazio de estados e $\gamma : S \to \Gamma_S^N$ é uma função que associa jogos estratégicos a estados.

Na terminologia de teoria dos jogos [OR94], Estruturas de Jogos (*Game Frames*) são essencialmente formas de jogos extensivos com movimentos simultâneos. A única diferença é que em [MP02] assume-se que em qualquer estado existe um jogo que pode ser jogado (não há estados terminais).

Estruturas de Jogo são modelos de interação que generalizam outros modelos baseados em ações. Um primeiro exemplo seria o caso onde os agentes não agem em paralelo, mas em turnos (um após o outro). Isso pode ser capturado por formas de jogos extensivos sem movimentos simultâneos, e pode ser modelado por exemplo por uma estrutura Kripke com uma relação de acessabilidade que liga estados a seus estados sucessores, e uma proposição que indica a cada estado o jogador da vez. Esses jogos extensivos podem ser também caracterizados por uma classe específica de estruturas de jogos:

Definição 6.1.3 *(Jogo Ditatório)*

Uma forma de jogo estratégico $\mathcal{F} = (N, \{\Sigma_i | i \in N\}, o, S)$ é chamada de d-ditatória (d-dictatorship) se e somente se $\forall s \in o \, \exists \sigma_d \forall \sigma_{N \setminus \{d\}} \; o(\sigma_d, \sigma_{N \setminus \{d\}}) = s$.

Note que no caso de existir mais de um d nesta situação, a função de resultado é constante e neste caso todo jogador é um *ditador*.

Definição 6.1.4 *(Estrutura Ditatória)*

Uma Estrutura Ditatória é uma Estrutura de Jogo (S, γ) tal que para todo $s \in S$ existe um $d \in N$ tal que $\gamma(s)$ é um jogo d-ditatório.

As Estruturas Ditatórias correspondem aos modelos Kripke onde os jogadores jogam em turnos.

Processos não determinísticos são outra classe ainda mais restrita de Estruturas de Jogo. Neles, há apenas um jogador que decide em qualquer estado qual seu

sucessor. Este caso não apresenta interação, e pode ser modelado com um modelo de Kripke tradicional. Em termos de Estruturas de jogo:

Definição 6.1.5 *(Processo)*

Um processo é uma Estrutura de Jogo (S, γ) *para um conjunto* N *de jogadores tal que* $|N| = 1$.

Todo Processo é uma Estrutura Ditatória, onde o ditador d é o mesmo em todos os estados.

6.2 Propriedades de Eficácia

A Lógica de Coalisão, que se baseia no modelo definido na seção 6.1, lida com propriedades sobre coalisões em jogos, em especial propriedades de eficácia de uma coalisão.

Dizemos que uma coalisão C é eficaz para um conjunto de estados X no estado s se é possível para o conjunto C de jogadores agir segundo uma estratégia conjunta para, a partir do estado s, atingir o conjunto específico de estados $X \subseteq S$, independente da ações dos jogadores que não participam da coalisão. A noção de eficácia empregada em [MP02] vem da teoria social de escolhas, onde é mostrada com o nome de α-eficácia [AK91, HM83].

Definição 6.2.1 *(Função de Eficácia)*

A Função de Eficácia $E_G : P(N) \rightarrow P(P(S))$ *de uma Forma de Jogo Estratégico* G *é definida como* $X \in E_G(C)$ *se e somente se* $\exists \sigma_C \forall \sigma_{\overline{C}}\ o(\sigma_C, \sigma_{\overline{C}}) \in X$.

Utilizaremos a simplificação $X \in E_G(C)$ para denotar que "a coalisão C pode forçar X".

A noção de eficácia pode ser estudada de forma mais genérica: para um conjunto de agentes N e um conjunto de resultados S, chamamos de função de eficácia uma função $E : P(N) \rightarrow P(P(S))$ qualquer. Intuitivamente, E associa a toda coalisão $C \subseteq N$ o conjunto de resultados para os quais a coalisão é eficaz, isto é, $X \in E(C)$ se e somente se C é eficaz para X.

Pode ser desejável que a função de eficácia satisfaça certas propriedades, de acordo com a situação que se deseja modelar. Dizemos que uma função de eficácia $E : P(N) \to P(P(S))$ é:

- *monotônica em resultado* se e somente se para todo $X \subseteq X' \subseteq S$ e para todo C, se $X \in E(C)$ então $X' \in E(C)$;
- *monotônica em coalisão* se e somente se para todo $C \subseteq C' \subseteq N$, $E(C) \subseteq E(C')$;
- *C-regular* se para todo X, se $X \in E(C)$ então $\overline{X} \notin E(\overline{C})$;
- *C-maximal* se para todo X, se $\overline{X} \notin E(\overline{C})$ então $X \in E(C)$;
- *regular* se e somente se é C-regular para todas as coalisões C;
- *maximal* se e somente se é C-maximal para todas as coalisões C;
- *super-aditiva* se para todo X_1, X_2, C_1, C_2 tal que $C_1 \cap C_2 = \emptyset$, $X_1 \in E(C_1)$ e $X_2 \in E(C_2)$ implica que $X_1 \cap X_2 \in E(C_1 \cup C_2)$.

6.2.1 Eficácia em Jogos Estratégicos

Examinaremos agora as funções de eficácia que podem ser consideradas em Formas de Jogo Estratégico. Isso será útil na definição da lógica de coalisão pois nos permitirá falar diretamente das funções, sem mencionar a Forma do Jogo Estratégico.

Definição 6.2.2 *(Função de Eficácia Aplicável)*

Uma Função de Eficácia $E : P(N) \to P(P(S))$ é Aplicável se e somente se:

1. $\forall C \subseteq N : \emptyset \notin E(C)$,
2. $\forall C \subseteq N : S \in E(C)$,
3. *E é N-maximal,*
4. *E é monotônica em resultado,*

5. E é super-aditiva.

(é possível demonstrar que essas condições são independentes)

Lema 6.2.1 *Toda função de eficácia aplicável é regular e monotônica em coalisão.*

(Prova em [MP02])

Theorema 6.2.1 *(Teorema da Caracterização)*

Uma função de eficácia E é aplicável se e somente se E é função de eficácia de alguma Forma de Jogo.

(Prova em [MP02])

6.2.2 Eficácia em Jogos Estratégicos Ditatórios

Definição 6.2.3 *(Função de Eficácia Individualista)*

Uma Função de Eficácia $E : P(N) \rightarrow P(P(S))$ *é Individualista se e somente se ela é aplicável e* $E(N) = \cup_{i \in N} E(\{i\})$.

A condição garante que se algo pode ser forçado por um grupo, então também o pode ser por um único agente.

Theorema 6.2.2 *Uma função de eficácia E é individualista se e somente se E é função de eficácia de um algum jogo ditatório.*

(Prova em [MP02])

6.3 Lógica de coalisão

6.3.1 Sintaxe e Semântica

Definição 6.3.1 *(Sintaxe para Lógica de Coalisão)*

Dado um conjunto não vazio N de agentes (ou jogadores) e um conjunto de proposições Φ_0*, uma fórmula* φ *pode ser definida como:*

$\varphi := \bot \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid [C]\varphi$ *, onde* $p \in \Phi_0$ *e* $C \subseteq N$.

No caso onde $C = \{i\}$ escrevemos $[i]\varphi$ ao invés de $[\{i\}]\varphi$.

Definição 6.3.2 *(Estrutura de Eficácia Aplicável)*

Chamamos de Estrutura de Eficácia Aplicável a função $E : S \to (P(N) \to P(P(S)))$*, onde para todo estado* $s \in S$*,* $E(s)$ *é uma função de eficácia aplicável.*

Definição 6.3.3 *(Estrutura de Coalisão)*

Uma Estrutura de Coalisão é um par $\mathcal{F} = (S, E)$ *onde* S *é um conjunto não vazio de estados (o universo) e* $E : S \to (P(N) \to P(P(S)))$ *é a Estrutura de Eficácia Aplicável do modelo.*

Para facilitar a leitura, escrevemos sE_CX ao invés de $X \in E(s)(C)$ para denotar que C é eficaz para X no estado s.

Definição 6.3.4 *(Modelo de Coalisão)*

Um Modelo de Coalisão é um par $\mathcal{M} = (\mathcal{F}, V)$ *onde* $\mathcal{F}$ *é uma Estrutura de Coalisão e* $V : \Phi_0 \to P(S)$ *é a função de valoração para as proposições em* Φ_0*.*

É possível demonstrar que sE_CX vale se e somente se a coalisão C é eficaz para X em $G(s)$.

A validade de uma fórmula neste modelo é definida como segue:

- $\mathcal{M}, s \not\models \bot$

- $\mathcal{M}, s \models p$, sse $s \in V(p)$, onde $p \in \Phi_0$

- $\mathcal{M}, s \models \neg\varphi$, sse $\mathcal{M}, s \not\models \varphi$

- $\mathcal{M}, s \models \varphi \vee \psi$, sse $\mathcal{M}, s \models \varphi$ ou $\mathcal{M}, s \models \psi$

- $\mathcal{M}, s \models [C]\varphi$, sse $sE_C\varphi^{\mathcal{M}}$, onde $\varphi^{\mathcal{M}} = \{s \in S | \mathcal{M}, s \models \varphi\}$

A fórmula $[C]\varphi$ vale no estado s se e somente se a coalisão C é eficaz para $\varphi^{\mathcal{M}}$ em $G(s)$. A fórmula φ é válida em um modelo $\mathcal{M}$ com universo S, denotado por $\mathcal{M} \models \varphi$, se e somente se $\varphi^{\mathcal{M}} = S$, e φ é válido na classe de modelos K (denotado por $\models_K \varphi$) se e somente se para todos os modelos $\mathcal{M} \in K$ temos $\mathcal{M} \models \varphi$. Escrevemos

$\Sigma \models_K \varphi$ para indicar que φ é uma conseqüência lógica (local) de Σ: Para todos os modelos $\mathcal{M} \in K$ e todo estado s do universo de $\mathcal{M}$, se $\mathcal{M}, s \models \Sigma$ (isto é, todas as fórmulas de Σ são válidas em s) então $\mathcal{M}, s \models \varphi$.

Seja $\mathcal{M}$ a classe de todos os modelos de coalisão, e seja $\mathcal{M}_d$ a classe dos modelos de coalisão $((S, E), V)$ onde para todo $s \in S$, $E(s)$ é individualista. Dada a caracterização do resultado de jogos ditatórios, podemos pensar em $\mathcal{M}_d$ como a classe de modelos ditatórios.

Devido ao teorema da caracterização podemos representar modelos para coalisão sem nos referirmos diretamente a jogos e estratégias. Isso simplifica o tratamento meta-teórico da Lógica de Coalisão, e também demonstra que um modelo de coalisão é simplesmente uma generalização multi-modal do modelo de vizinhanças, providenciando uma relação de vizinhança para toda coalisão.

Modelos de vizinhança tem sido a ferramenta semântica padrão para investigar lógicas modais não normais, e técnicas utilizadas para prover uma axiomatização completa para tais lógicas podem ser adaptadas para a lógica de coalisão.

6.3.2 Axiomatização

Dado um conjunto N de jogadores, uma Lógica de Coalisão para N é um conjunto de fórmulas Λ que contém:

Todas as instâncias dos axiomas:

1. Todas as tautologias da lógica proposicional
2. $\neg[C]\bot$
3. $[C]\top$
4. $(\neg[\emptyset]\neg\varphi \rightarrow [N]\varphi)$
5. $[C](\varphi \wedge \psi) \rightarrow [C]\psi$
6. $([C_1]\varphi_1 \wedge [C_2]\varphi_2) \rightarrow [C_1 \cup C_2](\varphi_1 \wedge \varphi_2)$, onde $C_1 \cap C_2 = \emptyset$

e que é fechada sobre as regras de Modus Ponens e Equivalência:

1. Modus Ponens:

$$\frac{\varphi \quad \varphi \rightarrow \psi}{\psi}$$

2. Equivalência:

$$\frac{\varphi \leftrightarrow \psi}{[C]\varphi \leftrightarrow [C]\psi}$$

Dada uma lógica de coalisão Λ, escrevemos $\vdash_\Lambda \varphi$ para $\varphi \in \Lambda$ e $\Sigma \vdash_\Lambda \varphi$ se existe $\sigma_1, \cdots, \sigma_n \in \Sigma$ tal que $(\sigma_1 \wedge \cdots \wedge \sigma_n) \rightarrow \varphi \in \Lambda$. Um conjunto de fórmulas Σ é Λ-inconsistente se e somente se $\Sigma \vdash_\Lambda \bot$.

Pode ser demonstrado que a regra da monotonicidade

$$\frac{\varphi \rightarrow \psi}{[C]\varphi \rightarrow [C]\psi}$$

é derivável em qualquer Lógica de Coalisão.

Dizemos que uma lógica é Λ-completa com respeito a uma classe de modelos de coalisão K se $\Sigma \models_K \varphi$ se e somente se $\Sigma \vdash_\Lambda \varphi$.

Theorema 6.3.1 *(Completude)*

A Lógica de Coalisão para N agentes, CL_N*, é completa com respeito à classe de todos os modelos de coalisão:* $\Sigma \models_{\mathcal{M}} \varphi$ *sse* $\Sigma \vdash_{CL_N} \varphi$

(Prova através da construção do modelo canônico [MP02]).

6.3.3 Axiomatização para Jogos Ditatórios

A lógica CL_N é a mais geral e mais fraca dentre as Lógicas de Coalisão. A única restrição é que, a cada estado, o poder de ação da coalisão possa ser modelado por um jogo estratégico. As restrições feitas para frames ditatórios também levam a uma lógica axiomatizável.

Seja DCL_N a menor lógica de coalisão incluindo os seguintes axiomas:

(D1) $[N]\varphi \rightarrow \bigvee_{i \in N}[i]\varphi$

(D2) $[N]\neg\varphi \wedge [N]\psi \rightarrow \bigwedge_{i \in N}([i]\varphi \rightarrow [i]\psi)$

Seja Λ qualquer lógica de coalisão estendendo DCL_N. Consideramos novamente o conjunto de conjuntos maximais consistentes de fórmulas S^Λ.

Theorema 6.3.2 $\Sigma \vdash_{DCL_N} \varphi$ *sse* $\Sigma \models_{M_d} \varphi$

(Prova através da construção do modelo canônico, levando em conta que todo $s \in S^\Lambda$ tem um ditador local [MP02]).

6.4 Aplicações e Exemplos

A Lógica de Coalisão possui várias aplicações na análise de processos e procedimentos sociais e em seus respectivos softwares de apoio, especialmente quando a interação possui papel relevante. Para teoria social, a necessidade é descobrir, dadas as preferências de cada indivíduo na sociedade, como a sociedade como um todo escolhe entre diferentes opções; através da visão de teoria de jogos associada à Lógica de Coalisão, podemos analisar quais métodos de decisão coletiva podem ser manipulados por indivíduos ou grupos. Porém, é importante frisar que a Lógica de Coalisão provê meios para falar do que os agentes, agrupados ou não, podem atingir, mas não entra no mérito do que efetivamente eles irão fazer, pois não leva em conta as preferências dos jogadores. Veremos a seguir dois exemplos de aplicação da Lógica de Coalisão.

6.4.1 Liberdade de Escolha

Abelardo e Eloísa tem cada um duas blusas: uma azul e uma branca. Cada um tem o direito de escolher que blusa deseja usar. Podemos modelar essa situação através da Lógica de Coalisão básica, sendo $N = \{a, e\}$ e $\Phi_0 = \{p_a, p_e\}$, onde p_a representa "Abelardo veste branco" e p_e representa "Eloísa veste branco". Os direitos de Abelardo e Eloísa podem ser representados pela seguinte fórmula escrita na Lógica de Coalisão:

$$\rho^+ := [a]p_a \wedge [a]\neg p_a \wedge [e]p_e \wedge [e]\neg p_e$$

Assumimos que estes são os únicos direitos que ambos tem.

$$\rho^- := \bigwedge_{i \in N} \neg[i]((p_a \wedge p_e) \vee (\neg p_a \wedge \neg p_e)) \vee \bigwedge_{i \in N} \neg[i]((p_a \wedge \neg p_e) \vee (\neg p_a \wedge p_e))$$

Para saber se estes direitos podem ser modelados por um jogo estratégico, devemos checar se $\rho^+ \wedge \rho^-$ pode ser satisfeito por um modelo de coalisão.

O seguinte modelo $\mathcal{M} = (S, \{E_C | C \subseteq N\}, V)$, com o conjunto de estados $S = \{s_0, (w,w), (w,b), (b,w), (b,b)\}$ satisfaz ρ no estado s_0: $V(p_a) = \{(w,w), (w,b)\}$, $V(p_e) = \{(w,w), (b,w)\}$, e $E(s_0)$ é a função de eficácia associada com o jogo estratégico apresentado na figura 6.1:

Figura 6.1: Direitos de Abelardo e Eloísa (Jogo Estratégico)

	w	b
w	(w, w)	(w, b)
b	(b, w)	(b, b)

Para estados diferentes de s_0, E pode ser definida arbitrariamente.

Podemos demonstrar que uma coalisão formada por Abelardo e Eloísa é capaz de forçar qualquer um dos quatro resultados possíveis. Então, se Abelardo e Eloísa têm alguma preferência em relação ao resultado final de suas escolhas, por exemplo, se eles não desejam sair juntos utilizando roupas da mesma cor, podem, unindo seus esforços, combinar que roupa cada um deve vestir.

Capítulo 7

Conclusões

O processo interativo em sistemas distribuídos compostos por agentes autônomos (Multi-Agent Systems-MAS) é o objeto de interesse deste trabalho. O estudo de uma computação distribuída é, em algum nível, observar como o individual afeta o coletivo. Se os agentes são racionais e percebem a dinâmica do sistema onde estão inseridos, decisões individuais passam pela consideração de seu impacto na interação. Isto corresponde a uma análise prévia de que conseqüências, a nível de resultados e respostas, uma atitude suscitará no sistema, ou de forma mais elaborada, como métodos de tomada de decisão coletiva podem ser manipulados por agentes e grupos de agentes (interação racional).

Agentes racionais no contexto de sistemas interativos podem ser caracterizados como entidades autônomas capazes de raciocinar sobre conhecimento próprio e de outros agentes, preferências, objetivos, ações passadas e futuras. A lógica modal é considerada uma ferramenta adequada para o tratamento de sistemas compostos por agentes com estas características; operadores modais para capturar transformações globais do sistema ([FL79], [PA85]) e dos processos de raciocínio individuais ([FA95], [VB01]) já foram propostos e sedimentados.

Uma vez inserido no sistema, um agente pode se confrontar com diversas formas de agir, mesmo quando movido por um objetivo específico. Os processos racionais de tomada de decisão e formação de intenção, inseridos no contexto global de interação racional, são o objeto de estudos da teoria dos jogos, e a princípio estão fora dos domínios tradicionais da lógica modal [HH02]. Um tratamento formal efetivo para esta classe de sistemas deve englobar as potencialidades de ambos os formalismos,

conforme já constatado por vários autores.

[MB94] considera lógica uma ferramenta apropriada para teoria de jogos, uma vez que as obscuridades conceituais desta última envolvem noções como raciocínio, conhecimento e contradição, que são parte do domínio de atuação da lógica modal.

Segundo [GB02], ferramentas da lógica modal enriquecem a linguagem de teoria dos jogos tornando possível expressar os conceitos que previamente eram informais ou vagamente capturados por um conceito de solução. Uma vez que certos conceitos como conhecimento e poderes (ações disponíveis e suas conseqüências possíveis) de um agente ou grupo são modelados explicitamente, viabilizam-se novas discussões: refinamento de estratégias à medida que um agente ganha conhecimento, formação associações a partir da percepção da possibilidade de aumento dos poderes individuais com maior proveito do que custo (julgamento racional).

7.1 Relações entre lógicas modais para jogos

Segundo [GB02], o aparato da lógica modal pode ser utilizado para modelar duas visões conceitualmente muito diferentes de teoria dos jogos: uma é a visão acerca do raciocínio de jogadores racionais, isto é, como o jogador decide sua forma de jogar. A outra é a visão acerca das possibilidades reais, ou seja, uma análise normativa dos poderes do jogador (o que o jogador é capaz de fazer).

A visão epistêmica corresponde a uma abordagem do individual para o coletivo, que busca capturar explicitamente os raciocínios particulares e mútuo reconhecimento do raciocínio dos outros jogadores. A outra abordagem é do coletivo para o individual, ou seja, parte dos resultados globais e procura uma forma de encaixar os resultados com os poderes de cada jogador, levando em conta os objetivos particulares de cada um.

Lógicas de Jogos correspondentes a ambas as visões foram estudadas, todas construídas sobre o mesmo arcabouço modal de *PDL*. A partir daí, cada uma apresenta suas particularidades:

- A Lógica de Jogos (GL) apresentada no capítulo 4 interpreta as modalidades como jogos atômicos e apresenta operadores para combinar a execução de

jogos (de forma seqüencial, não determinística, repetição, inversão de papéis, e até uma forma de teste de fórmulas). Através dela é possível falar como cada jogador pode se comportar, e que resultados é capaz de obter com suas atitudes. É possível descrever situações como jogos e utilizar a lógica para encontrar os modelos que satisfazem as fórmulas que descrevem as ações de cada jogador no jogo. Podemos com isso demonstrar formalmente o que cada jogador é capaz de obter, e o que não é capaz de obter.

- A Lógica de Coalisão (*CL*) apresentada no capítulo 6 interpreta as modalidades como coalisões, ou seja, ações conjuntas de um ou mais jogadores. O modelo por trás reflete, a cada estado, o conjunto de estados que podem ser atingidos a partir das atitudes dos jogadores no estado de origem, que podem ser conjuntas ou não. Assim como na *GL*, não há uma forma de expressar preferências dos jogadores, nem individuais nem de grupo. É possível fazer um estudo de quais resultados cada associação de pessoas pode obter. Ao descrever uma situação através de fórmulas da *CL*, podemos analisar modelos e estudar propriedades do ponto de vista dos poderes de cada indivíduo dentro do grupo, como por exemplo descobrir se uma determinada modalidade de eleição em turnos representa a opinião da maioria ou não.

- Abordagens para o tratamento de jogos de informação imperfeita através de lógica modal de conhecimento e crença foram apresentadas em [VB01]. Os operadores de conhecimento e crença são definidos da forma usual sobre conjuntos de estados indistinguíveis, que são os conjuntos de conjuntos de informação do jogador no modelo de informação imperfeita. Do ponto de vista dos poderes de cada jogador (ações possíveis e seus resultados), os jogos de informação imperfeita requerem ações uniformes sobre os conjuntos de informação. Fora isso, pouco diferem das abordagens da *GL* e da *CL*.

- *Channel-CPDL* trata especificamente de comunicação em modelos concorrentes. As modalidades são programas, que podem ser combinados através de operadores de paralelismo, serialização, não determinismo, repetição e teste, praticamente o mesmo tratamento dado às ações na *GL*. A grande diferença

é a introdução de um novo elemento sintático: um canal de comunicação que pode ser estabelecido entre programas, e duas novas operações que não alteram o estado dos processos que as realizam: envio e recebimento de mensagens através do canal de comunicação. Além da troca de informações, a comunicação impõe também um recurso para sincronização de programas. A semântica de *channel-CPDL* é muito semelhante à da *GL* e da *CL*, permitindo expressar assertivas sobre os resultados que podem ser atingidos através da execução de um programa concorrente.

7.2 Considerações sobre novas lógicas modais de jogos

As lógicas de jogos estudadas capturam, cada uma delas, conceitos específicos de teoria dos jogos. Porém, o tratamento efetivo de sistemas interativos, mesmo dos mais elementares, pressupõe um apanhado de vários destes conceitos. O conceito de estratégia, por exemplo, foi abordado apenas do ponto de vista externo aos agentes. A comunicação, uma das principais e mais peculiares formas de interação, não foi incluída em nenhuma das lógicas de jogos vistas.

Seria interessante buscar outras propostas de lógicas e linguagens para falar sobre jogos, tanto sob o ponto de vista dos jogadores quanto sob o ponto de vista dos resultados que podem ser obtidos. A proposta é estudar e construir lógicas de jogos para representar jogos extensivos com informação imperfeita, onde seja possível expressar informações acerca das estratégias de cada jogador individualmente ou em grupo (coalisão), dado o seu nível de conhecimento (individual ou da coalisão).

7.2.1 Modelos para Lógicas de Jogos

Formas extensivas de jogo ou variações sobre elas podem ser concebidas como estruturas de Kripke para versões de lógicas modais. Além do modelo básico de estados e ações, há outras possibilidades a serem investigadas, especialmente no caso de ações conjuntas (coalisão), informação imperfeita, e comunicação. Os modelos correspondentes devem incluir critérios para definir a validade das fórmulas nos

estados finais e intermediários da estrutura, levando em conta as modalidades e outros mecanismos sintáticos (como no caso da comunicação em *channel-CPDL*) sendo utilizados.

7.2.2 Modalidades para Jogos

A partir das lógicas modais para jogos estudadas, é possível perceber o apelo de diversas modalidades para jogos, algumas muito pouco exploradas e outras ainda não investigadas.

Modalidades de Ação

Para falar das ações, a adoção de uma versão da lógica dinâmica proposicional (*PDL*) é praticamente um consenso.

Modalidades de Coalisão

A formação de coalisão modifica a forma de atuação dos jogadores, que passa a ser uma atuação em grupo e não mais individual. [MP02] apresenta uma lógica modal para analisar os poderes de jogadores isolados e quando em coalisão, onde modalidades específicas para coalisões são investigadas.

Outra abordagem para o tratamento de coalisões em jogos pode ser pensada no nível dos agentes, englobando o julgamento sob o ponto de vista individual de quando e porquê elas devem ser formadas. Essa abordagem pressupõe que a formação de coalisões seja conseqüência de um acordo entre os jogadores, situação onde a sincronização é um mecanismo essencial.

Formar uma coalisão torna-se mais uma opção de jogada disponível para os agentes, e a coalisão, uma vez formada, é o reflexo de um comportamento em bloco de um grupo de jogadores. O tratamento de coalisões sob este enfoque pode ser feito no nível de paralelização de processos, ou seja, no nível das modalidades de ação.

Modalidades de Conhecimento

A abordagem de agentes para modelagem de sistemas é especialmente interessante quando cada unidade autônoma tem suas particularidades de comportamento.

Dentre estas particularidades, está a percepção individualizada que cada agente tem do ambiente onde está inserido, que corresponde à situação onde os agentes possuem apenas conhecimento parcial do ambiente.

Falar de conhecimento parcial certamente implica em adotar modalidades de conhecimento e crença, mas há muitas opções acerca de como estabelecer as relações entre estados indistinguíveis, e qual o escopo das incertezas (história passada, ações alheias, próprias ações, ações futuras, etc). A proposta de estabelecer as relações de indistinguibilidade sobre os conjuntos de informação do jogador não está bem estabelecida, e deve ser investigada.

A formação de coalisão reflete uma colaboração entre agentes que pode se realizar também a nível de conhecimento, o que requer modalidades para conhecimento de grupo.

Modalidades de Ação Reversa

Em linhas gerais, para cada ação a podemos definir a modalidade de ação reversa ${\Box_a}^{-1}$ sobre a relação de acessabilidade R_a que dá a cada estado s o conjunto de estados resultantes da execução de a a partir de s. A interpretação para uma fórmula ${\Box_a}^{-1}\varphi$ é: "se a ação a foi realizada, então imediatamente antes disso acontecer era o caso que φ".

A ação reversa é um recurso valioso nos casos onde o jogador trabalha com informação imperfeita, pois permite a partir de observações sobre o presente fazer considerações sobre o passado e, com o ganho de informação, reformular sua estratégia (análise interativa).

Além das modalidades para jogos, permanece a carência de recursos de comunicação. O papel da comunicação nos processos interativos ainda não foi explorado nos modelos de jogos.

Uma possível abordagem seria incluir a comunicação nas lógicas de jogos de forma semelhante à feita em *channel-CPDL*. Algumas considerações sobre questões temporais específicas do modelo de jogos como a existência de turnos de jogada podem ser experimentadas como simplificações do formalismo original.

Outra questão que deve ser considerada é a influência das preferências dos joga-

dores no desfecho do jogo. Nas lógicas de jogos apresentadas na revisão bibliográfica as preferências dos jogadores não são tratadas explicitamente. As preferências foram exploradas em várias outras lógicas de jogos como por exemplo [HH02], com a intenção de expressar na linguagem conceitos como equilíbrios.

As relações entre equilíbrio e preferência já foram tratados em teoria dos jogos, porém há o interesse prático de relacionar as preferência e os poderes de um jogador ou de um grupo. Incluir preferências na linguagem é um pré-requisito para raciocinar sobre estratégias, um problema pouco explorado no caso de informação imperfeita.

Referências Bibliográficas

[AK91] J. Abdou, H. Keiding, Effective Functions in Social Choice, *Kluer* (1991).

[GA99] G. Asheim, On the epistemic Foundations of Backward Induction, *Logic, Game Theory and Social Choice* (1999) 8–23.

[MB94] M. Bacharach, The epistemic structure of a theory of game, *Theory and Decision, 37* (1994) 7–48.

[VB01] J. van Benthem, Games in Dynamic-Epistemic Logic *Bulletin of Economic Research* (2001) 53 (4)219-48.

[KB92] K. Binmore, *Fun and games, a text on game theory*, Heath (1992).

[BV00] P. Blackburn, M. de Rijke, Y. Venema, *Modal Logic*, 1ª edição (2000), Cambrige, Cambrige University Press.

[GB99] G. Bonanno, The logic of rational play in games of perfect information, *Economics and Philosophy* Spring (1991) 37–65.

[GB02] G. Bonanno, Modal logic and game theory: two alternative approaches, *Risk Decision and Policy, 7*, (2002), 309-324.

[DB00] B. de Bruin, Modeling Knowledge in Games. , *Tese de mestrado, ILLC, University of Amsterdam*, (2000).

[TC99] T. Clausing, Behavioral vs. habitual rationality and backward induction, *Logic, Game Theory and Social Choice* (1999) 46–56.

[FA95] R. Fagin, J. Halpern, Y. Moses, M. Vardi, *Reasoning About Knowledge*, 1ª edição (1995), Massachusetts, EUA, MIT Press.

[FL79] M. Fischer, R. Ladner, Propositional Dynamic Logic of Regular Programs, *J. Comp and System Science* **18**(1979) 194–211.

[HH02] B. Harrenstein, W. van der Hoek, J-J. Meyer, C. Witteveen, On Modal Logic Interpretations of Games, *Proceedings of ECAI 2002, 15th European Conference on Artificial Intelligence* (2002) (pp. 28-32).

[KN96] M. Kaneco e T. Nagashima, Game Logic and its applications, *Studia Logica* **57**(1996) 325–354.

[KD82] D. Kozen, Results on the propositional μ-calculus, *Proceedings of 9th ICALF, Springer LNCS* **140**(1982) 348–359.

[M80] R. Milner, A calculus for communicating systems, *Lecture Notes in Computer Science* Vol. 92, (1980).

[HM83] H. Moulin, The strategy of social choice, *North-Holland*, (1983).

[OR94] M. Osborne, A. Rubinstein, *A course in game theory*, MIT Press, (1994).

[MP99] M. Pauly, Some Game Theory for Logicians, *Notes dor the course Logic and Games* Amsterdam, (1999).

[MP00] M. Pauly, From Programs to games: Invariance and safety for bissimulation, *Computer Science Logic* Vol. 1862, (2000).

[MP00] M. Pauly, Game logic for game theorists, *Report from Centrum voor Wiskunde en Informatica* September, (2000).

[MP02] M. Pauly, A Modal Logic for Coalitional Power in Games *Journal of Logic and Computation* Vol.12(1), **149-166** (2002)

[PA85] R. Parikh, The logic of games and its applications, *Topics in the Theory of Computation* Vol. 24, **57**(1985).

[PA??] R. Parikh, Social software, *commmming soon* , (20??).

[PE87a] D. Peleg, Concurrent Dynamic Logic, *Journal of Assoc. Comput. Mach.* (1987).

[PE87b] D. Peleg, Communication in Concurrent Dynamic Logic, *Journal of Computer and System Sciences* Vol. 35, **23 - 58**(1987).