MARCOS VINICIUS POLICARPO CÔRTES

UEBDI : UMA ARQUITETURA PARA AGENTES EMOCIONAIS BDI

Dissertação apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Mestre na área de Concentração Engenharia de Software.

Orientadora: Profª. Drª. VIVIANE TORRES DA SILVA

Niterói

2013

Ficha Catalográfica – Esta página deve ser removida na versão a ser entregue para a banca, mas deve ser reinserida na versão final, com a ficha catalográfica fornecida pela biblioteca. Informações sobre este processo devem ser obtidas na secretaria da pós-graduação.

MARCOS VINICIUS POLICARPO CÔRTES

UEBDI : UMA ARQUITETURA PARA AGENTES EMOCIONAIS BDI

Dissertação apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Mestre. Área de Concentração: Engenharia de Software.

Aprovada em janeiro de 2013

BANCA EXAMINADORA

Prof. Dr. Viviane Torres da Silva – Orientadora
UFF - Universidade Federal Fluminense

Prof. Dr. Anselmo Antunes Montenegro
UFF - Universidade Federal Fluminense

Prof. Dr. Ricardo Choren Noya
IME - Instituto Militar de Engenharia

Niterói
2013

Agradecimentos

Ao meu pai e meu irmão por me acompanharem neste período. Aos colegas que fiz na pós-graduação, minha orientadora Viviane Torres Silva, aos professores e funcionários da UFF que me ajudaram no decorrer deste trabalho. A Jomi Hübner, coautor do Jason, pelo suporte a ferramenta. Ao CNPq, pelo apoio financeiro dado através do processo nº 146846/2010-9.

Muito Obrigado

Resumo

Agentes Emocionais BDI são uma classe de agentes que estende a arquitetura BDI para resolver os problemas de capturar, expressar e simular o fenômeno emocional em sistemas computacionais. Algumas abordagens de Agentes Emocionais BDI simulam a emoção de forma explícita, construindo uma base de emoções que influencia as outras instâncias da arquitetura e funções que revisam o estado emocional sobre a arquitetura BDI.

Visto que não existe abordagem prévia que tenha realizado todas as possíveis influências usando um embasamento teórico válido, esta Dissertação de Mestrado propõe uma arquitetura abstrata para lidar com a influência emocional sobre todas as outras instâncias da arquitetura. Além disto, esta proposta foi mapeada e implementada sobra a Plataforma de Agentes Jason e uam série de experimentos foram feitos para validá-la.

Palavra chave: Agentes emocionais, Agentes BDI, Arquitetura de Agente, Agentes Emocionais BDI.

Abstract

Emotional BDI Agents are a class of agents that extend BDI architecture to solve problems of capturing, expressing and simulating emotional phenomena on computational systems. Some approaches of Emotional BDI Agents simulate emotion in an explicit way, building an emotional base that influences other architecture's instances and functions to review the emotional state over the BDI architecture.

Since no previous approach has fulfilled all possible influences using a valid theorical basis, this Master's Thesis proposes an abstract architecture to deal with emotional influence on every other architecture's instance. Furthermore, the proposed approach was mapped and implemented using Jason Agent Platform and a serie of experiments has been performed to validate it.

Keywords: Emotional Agents, BDI Agents, Agent Architecture, Emotional BDI Agents.

Lista de Figuras

Lista de Tabelas

Sumário

1 Introdução

Recentes estudos evidenciam que a emoção desempenha importante papel no comportamento inteligente e participa ativamente na tomada de decisão (Picard 2000 p. 2), no processo de adaptação do indivíduo ao ambiente (relacionando os diversos aparatos cognitivos e suas respostas) (Sloman 1999) e na economia de recursos (LeDoux 2002). Além de participar de inúmeros processos cognitivos, como memória, percepção, aprendizagem e fala (Oliveira and Sarmento 2003, Picard 1997).

Atualmente é vastamente aceito que as emoções são importantes nas atividades humanas (Hernández et al. 2004). Isto motiva vários pesquisadores a investigarem os fenômenos emocionais a fim de simulá-los em sistemas computacionais. (Picard 1997 p. 50) propõe que adaptar emoções humanas em computadores tornará a tomada de decisão destes sistemas mais flexível e dará maior capacidade para lidar com múltiplos interesses e preocupações. A habilidade de reconhecer afeto e de expressar afeto ajudará a estes sistemas a ajustar seu comportamento, melhorando a interação com seus usuários humanos.

Um agente de software é uma entidade (computacional) autônoma que está inserida em um ambiente, muitas vezes incerto, havendo imprecisão na sua percepção e incerteza na efetuação de suas ações (Wooldridge 2009). Este conceito é muito importante para a área de Inteligência Artificial (IA), pois estas características facilitam a modelagem de problemas reais e permite que o agente lide melhor com cenários onde a IA clássica, lógica e formal tem dificuldade de atuar (Russel and Norvig 2004 p. 6). Da mesma forma, a comunidade de Engenharia de Software (ES) vem adotado os conceitos de "agência" para representar e implementar sistemas de software onde são exigidos requisitos de ubiquidade, interligação, delegação, humanização e inteligência (Wooldridge 2009 p. 4).

Dentro do estudo de agentes existe uma classe distinta chamada *Agentes BDI* (Rao and Georgeff. M. 1995) que procura modelar e construir agentes adotando os conceitos de crenças, desejos e intenções (destes termos vem a sigla que o representa: *Beliefs, Desires and Intentions*), o que facilita sua representação.

É observado que existe uma série de trabalhos que adotam agentes que implementam algoritmicamente o fenômeno emocional para tratar os problemas da computação afetiva. A Computação Afetiva busca simular o fenômeno emocional em sistemas computacionais, sendo o uso da teoria de agentes natural neste contexto, visto que é preciso representar a causa, a responsabilidade dos fenômenos emocionais como uma entidade externa ao sistema que a contempla (Marsella and Gratch 2009 p. 75). Uma lista exaustiva destes trabalhos pode ser encontrada em (Picard 1997) e uma versão mais atualizada em (Picard 2003). Os trabalhos (Silva et al. 2006), (Moridis and Economides 2008) e (Rumbell et al. 2011) também apresentam outras propostas focadas em problemas específicos.

Ainda são poucos os trabalhos que se dedicam exclusivamente a estudar a influência da emoção sobre os processos cognitivos do agente de software (Neto and Da Silva 2010 p. 103, Neto 2010 p. 9). Este problema é difícil de se lidar, visto que o embasamento da teoria da emoção ainda é excipiente e impreciso (Picard 2003).

Especificadamente, para o problema de modelagem e simulação do fenômeno emocional encontramos trabalhos sobre interação com o usuário, visando: (i) o entretenimento (Bates et al. 1994), (ii) a simulação de emoções robôs (Breazeal 1998, Hernández et al. 2004, Velásquez 1996), (iii) a criação de um sistema computacional para interação musical com o usuário (Camurri and Coglio 1998), (iv) a otimização e adaptação do agente ao ambiente (Oliveira and Sarmento 2003, Signoretti 2012, Signoretti et al. 2010) e (v) a tomada de decisão (Jiang and Vidal 2006, Neto and Da Silva 2010, Oliveira and Sarmento 2003, Pereira et al. 2005, Velásquez 1998b), seja para otimização de desempenho (Morgado 2006, Signoretti et al. 2010) ou para simulação de comportamento humano (Neto 2010).

Convencionou-se a chamar os agentes que implementam os fenômenos emocionais de *Agentes Emocionais*. Não existe uma definição formal aceita (Scheutz 2002), mas elas convergem para a seguinte: *Agentes Emocionais* são agentes que buscam solucionar os problemas da computação afetiva (Camurri and Coglio 1998) de perceber, expressar, simular internamente emoções (Meyer 2004, Oliveira and Sarmento 2003, Steunebrink 2010) e para tomar decisões que sejam úteis ao agente (Pereira et al. 2005). Inclusive, algumas propostas de agentes emocionais vêm adotando como base a arquitetura BDI, chamados de *Agentes Emocionais BDI* (Hernández et al. 2004, Jiang and Vidal 2006, Meyer 2004, Neto and Da Silva 2010, Padgham and Taylor 1997, Pereira et al. 2005, Signoretti 2012, Steunebrink 2010), conforme discutiremos mais afundo a seguir.

1.1 Motivação

Embora estes trabalhos de agentes emocionais BDI tenham obtido êxito em desenvolver soluções baseadas na arquitetura BDI, eles isoladamente satisfazem apenas alguns aspectos do fenômeno emocional, e principalmente lidam somente com algumas das possíveis influências do fenômeno emocional sobre módulos da arquitetura BDI. Por exemplo, uma dada proposta trabalha bem com modelos emocionais de confiança entre agentes (Jiang and Vidal 2006), porém a arquitetura não lida com algoritmos de foco de atenção influenciados pelas emoções (Morgado 2006, Oliveira and Sarmento 2003, Signoretti 2012), pois não implementa a influência da emoção sobre o módulo de percepção do agente. A seguir mencionamos brevemente alguns destes trabalhos e suas limitações.

O trabalho de (Hernández et al. 2004) apenas contempla a influência do estado emocional sobre crenças e desejos, e as abordagens de (Jiang and Vidal 2006, Oliveira and Sarmento 2003) focam seus modelos na influência das emoções sobre as intenções. Já (Neto 2010) cria uma arquitetura concreta sobre a plataforma Jason (Bordini et al. 2007) onde contempla influência das emoções na seleção de percepções, crenças e intenções, mas não define uma arquitetura abstrata que possa ser implementada em outra plataforma. (Steunebrink 2010) define uma fomalização lógica BDI extendida com um modelo bem definido, porém apenas considera a influência das emoções sobre as intenções do agente. (Signoretti 2012) define um framework que só usa emoções para filtrar as emoções existentes e implementa tal *framework* utilizando Jason. O autor também preferiu criar um conjunto de bibliotecas ao invés de estender o Jason. Por fim, (Pereira et al. 2005) cria uma arquitetura onde muitos módulos BDI são influenciados pela emoção, mas os autores não justificam as motivações para tais influências.

Como vemos, utilizando apenas uma proposta não é possível desenvolver um agente emocional onde as emoções sejam consideradas em todo processo de raciocínio dado que cada proposta contempla apenas a influência das emoções em um subconjunto dos módulos BDI. Além disto, como todos os estudos são parciais quanto a influência emocional sobre as instâncias BDI, a literatura carece de um estudo e proposta de um agente emocional BDI que possibilite todas as influências possíveis do fenômeno emocional sobre o resto da arquitetura.

1.2 Solução Proposta

Como dito no tópico anterior, a impossibilidade das propostas anteriores em representar apenas algumas das influências emocionais em sua arquitetura, motivou o

presente trabalho a desenvolver uma proposta **unificada** que contemple estas influências do fenômeno emocional sobre todos os módulos BDI.

O objetivo com esta nova proposta é permitir modelar e implementar fenômenos emocionais e as diversas influências das emoções nos módulos BDI através de uma única abordagem. Um desenvolvedor de agentes emocionais poderá programar agentes que utilizem diferentes influências do fenômeno emocional lidando com cenários emocionais diversos e utilizando para isto uma única arquitetura abstrata e um único *framework* de implementação.

Para desenvolver a nossa abordagem, adotamos a seguinte metodologia: (i) levantar o estado da arte das propostas que implementam agentes emocionais se baseando na arquitetura BDI; (ii) propor uma arquitetura para agentes emocionais BDI chamada UEBDI que unifique as características apresentadas nas propostas estudadas, de tal forma que o resultado seja consistente e compatível com a arquitetura BDI e com a teoria emocional; (iii) mapear a UEBDI sobre uma implementação de agente BDI – a escolhida foi o Jason; e (iv) demonstrar, via cenários de uso, que é possível utilizar as características adotadas pela arquitetura em diferentes contextos explorando as diferentes influências das emoções nos módulos BDI.

Adotou-se a arquitetura BDI como base principalmente porque (i) a arquitetura BDI é um padrão *de fato* nas comunidades de IA e ES para desenvolver agentes de software (Jiang and Vidal 2006 p. 2, Neto and Da Silva 2010 p. 34, Pereira et al. 2005 p. 1, Signoretti 2012); (ii) ela foi construída para lidar com os problemas de reconsideração e planejamento, problemas clássicos de IA (Bordini et al. 2007, Wooldridge 2009) e (iii) ela é baseada em psicologia informal (Pereira et al. 2005), instâncias intencionais (Wooldridge 2009) e simula a tomada de decisão humana (Hernández et al. 2004), o que facilita relacionar conceitos do fenômeno emocional à arquitetura.

Já a plataforma Jason foi adotada porque (i) ela é um interpretador da linguagem formal AgentSpeak(L) (Rao 1996) que preserva os conceitos formais de BDI; (ii) ela é *open-source*, o que nos ajudou na economia de custos do projeto e no acesso ao código fonte, possibilitando compreender melhor como ele funciona internamente; e (iii) Jason possui razoável documentação de apoio.

Para desenvolver a arquitetura unificada focou-se, principalmente, nas propostas de agentes emocionais explicitamente desenvolvidas sobre BDI (Hernández et al. 2004, Jiang and Vidal 2006, Neto and Da Silva 2010, Pereira et al. 2005, Signoretti 2012) e nas propostas que usam os conceitos BDI e que adicionavam algum algoritmo importante de influência emocional sobre a arquitetura, embora algumas não tenham seguido o fluxo de execução BDI explicitamente (Breazeal 1998, Lino 2006, Morgado 2006, Oliveira and Sarmento 2003,

Velásquez 1998b). Preferiu-se usar como base as arquiteturas explicitamente desenvolvidas sobre BDI pela sua semelhança com esta proposta e pelo objetivo deste trabalho de unificar as diversas propostas. Também se preferiu as propostas que construíram algoritmos de influência do fenômeno emocional sobre arquitetura, pois elas contribuíram com nosso objetivo de desenvolver uma arquitetura focada na influência emocional e não na simulação do estado e dinâmica emocional.

Outras propostas não foram exaustivamente utilizadas, pois algumas focavam na simulação de um modelo emocional específico (Bazzan and Bordini 2001, Steunebrink 2010), o que difere da escolha que fizemos de deixar esta tarefa a projetista de agente. Outras não se preocupavam com a influência da emoção no resto da arquitetura (Marsella and Gratch 2009), que é justamente a meta principal desta dissertação. E ainda algumas se preocupavam com a formalização lógica do fenômeno emocional (Meyer 2004, Steunebrink 2010), o que colide com nosso objetivo de deixar o modelo emocional em aberto na arquitetura.

1.3 Contribuições

Como principais contribuições deste trabalho, podemos citar:

1.3.1 Arquitetura abstrata UEBDI

Foi desenvolvida uma arquitetura abstrata para o desenvolvimento de agentes emocionais estendendo a arquitetura BDI. A extensão se deu de tal forma (i) que as implementações feitas em UEBDI possam usar conceitos e ferramentas conceituais já desenvolvidas para BDI; (ii) que a nova arquitetura unifique os conceitos propostos pelas abordagens anteriores para o desenvolvimento de agentes emocionais com o mesmo propósito, permitindo desenvolver diferentes fenômenos emocionais em uma só abordagem; (iii) que a arquitetura abstrata fosse independente do fenômeno emocional; e (iv) que a nossa abordagem focasse em explicitar a influência da emoção sobre os diversos módulos da arquitetura, e não na implementação do estado e da dinâmica emocional.

1.3.2 Implementação da arquitetura UEBDI sobre o Jason

A arquitetura abstrata foi mapeada para a arquitetura concreta Jason e depois implementada sobre esta plataforma. Como resultado, foi criado um *framework* para a implementação dos agentes emocionais definidos de acordo com a arquitetura abstrata. O

framework deixa explícitos os pontos que podem e devem ser estendidos para a implementação de um agente emocional.

1.3.3 Conjunto de experimentos

Foi criado um conjunto de experimentos para verificar a viabilidade da utilização da arquitetura e do *framework* UEBDI para desenvolver agentes emocionais. Nosso critério de completude foi criar um experimento para cada influência do módulo emocional sobre outra instância BDI. Os experimentos são: (i) simulação de incêndio (influência sobre a percepção), (ii) simulação de teste dado por um professor a um aluno (influência sobre o módulo de crenças), (iii) simulação de *drive* (pulsão) diversão (influência sobre o módulo de desejos), (iv) dilema do prisioneiro iterado – IPD – (influência sobre o módulo de intenções) e (v) alocação de tarefas (influência sobre o módulo de ações).

1.4 Organização do texto

Esta dissertação será divida nos seguintes capítulos:

Capítulo 2 – Referencial teórico: Aborda uma série de conceitos básicos envolvidos com nosso trabalho, como a Teorias da Emoção, Agentes de Software, Agentes BDI, Computação Afetiva e Agentes Emocionais;

Capítulo 3 – Trabalhos Relacionados: Trata das abordagens de agentes emocionais relacionadas com nossa proposta;

Capítulo 4 – Unified Emotional BDI Architecture: Propõe a nossa proposta de arquitetura abstrata para agentes Emocionais BDI unificando as propostas anteriores;

Capítulo 5 – Mapeando UEBDI no Jason: Mapea a UEBDI na plataforma Jason, demonstrando a viabilidade de implementação da arquitetura;

Capítulo 6 – Experimentos: Experimenta a proposta com uma série de cenários de fenômeno emocional, focando em exercitar as influências do módulo emocional sobre o resto da arquitetura;

Capítulo 7 – Conclusão: Por fim, é discutido quais as contribuições, limitações e possíveis trabalhos futuros deste trabalho.

2 Referencial Teórico

Neste capítulo será abordado um conjunto de temas que servem de apoio teórico para esta dissertação. O tópico 2.1 apresenta o que são agentes de software, servindo de teoria base para nossa pesquisa. O tópico 2.2 aborda uma classe especial de agentes chamada de Agentes BDI, que se preocupa em representar o raciocínio simbólico com instâncias de crenças, desejos e intenções. O uso desta representação torna simples a modelagem de problemas de deliberação e planejamento, por exemplo, que são problemas clássicos de IA. O tópico 2.3 discute a implementação BDI Jason. O tópico 2.4 discutirá as teorias da emoção, apontoando conceitos importantes adotados em nossa dissertação, como eliciação, emoções multicamadas, entre outros. O tópico 2.5 discute sobre a computação afetiva, área de pesquisa interessada em relacionar emoções e computação. Por fim, no tópico 2.6 apresentamos o que são agentes emocionais, agentes capazes de simular, expressar e perceber emoções, e uma subclasse deste, que estende a arquitetura BDI para que seja capaz de representar emoções, chamados de agentes emocionais BDI.

2.1 Agentes

Em computação, podemos dizer que agentes são uma classe de sistemas que exigem certo grau de autonomia, sendo capazes de atingir uma tarefa por si mesmos (Bordini et al. 2007 p. 2, Wooldridge 2009 p. 15). Em Inteligência Artificial, este conceito junto com o de ambiente servem de arcabouço conceitual para estudar como desenvolver sistemas com desempenho ótimo (Russel and Norvig 2004 p. 7).

Considera-se um agente de software como um sistema computacional **autônomo** inserido em um ambiente, que recebe **percepções** deste e é capaz de realizar **ações**. Considera-se que as qualidades desejáveis para este sistema são: autonomia, proatividade, reatividade e capacidade de sociabilização (Bordini et al. 2007 p. 3). Nota-se que é incomum e muitas vezes irreal existir um único agente em um ambiente. Neste caso, um ambiente é

usualmente repleto de agentes que podem interagir, trocando mensagens e recursos. Neste caso, considera-se o conjunto ambiente e os seus agentes um **Sistema Multi-Agente**.

Segundo os autores (Russel and Norvig 2004, Wooldridge 2009) uma arquitetura genérica de agente pode ser definida por três funções, como exibido na Figura 5: (i) o comportamento de um agente é implementado por uma **função de agente** (*Agent Function*), (ii) a percepção do ambiente por sensores (*Sensor*) e (iii) sua atuação por atuadores (*Actuator*). O agente "trabalha" em ciclos, de tal forma que para operar ele executa as funções *Sensor*, *Agent Function* e *Actuator*, nesta ordem. A áreas de IA e Engenharia de Software se preocupam principalmente em descrever a *Agent Function*, visto que nela reside o raciocínio propriamente dito. A *Agent Function* é dada pela (Eq. 1), onde *S* é o conjunto de sensações que atingem os sensores e *A* é a ação (ou conjunto de ações) tomada pelo agente.

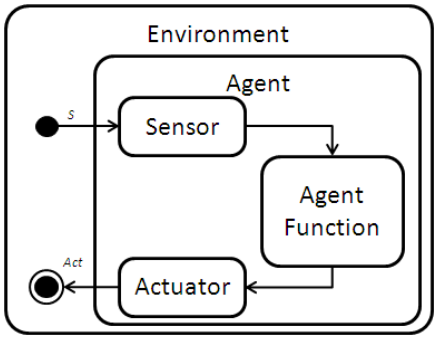


Figura 1: *Agent Function*

$$Agf: S \rightarrow A \qquad \text{(Eq. 1)}$$

Na literatura, convencionou-se classificar os agentes em "tipos" de acordo com sua constituição, seus objetivos e como atingem suas tarefas. Falaremos de quatro tipos importantes para nosso estudo: (2.1.1) Racional, (2.1.2) Reativo, (2.1.3) Deliberativo e (2.1.4) BDI.

2.1.1 Agente Racional

Agentes racionais são agentes que fazem tudo certo (Russel and Norvig 2004). Agente racional é aquele que ao realizar uma tarefa obtém o melhor resultado/pontuação e – preferencialmente – em menor tempo (ou menor custo de forma geral). Formalmente, um agente racional é aquele que maximiza (ou minimiza) sua **medida de desempenho** (Russel and Norvig 2004 p. 36). Esta é uma medida especificada pelo problema que representa o quão "bom" foi o resultado gerado por um agente que o solucionou. Por exemplo, imagine que temos o problema de *arrumar uma sala*. Para medirmos o quanto uma sala está arrumada podemos contar quantos objetos estão jogados no chão. Esta medida – *quantos objetos estão*

no chão – é a métrica de desempenho do problema. Os agentes podem solucionar este problema de várias maneiras, mas aquele(s) que ao final obtiverem menor valor da medida (menos objetos no chão) é um agente racional.

Nota-se que agente racional é meramente uma definição que serve de ferramenta de análise de um problema. As outras definições de agente abaixo irão discutir a constituição do agente, e não seu desempenho em um problema. Por isto, um agente dos tipos discutidos abaixo podem ser racionais ou não, isto depende se o algoritmo que eles implementam obtém melhor desempenho ou não.

2.1.2 Agente Reativo

Agentes reativos reagem diretamente aos estímulos do ambiente, sem raciocinar sobre isto (Wooldridge 2009 p. 85). Funcionalmente, um agente reativo é aquele que age de acordo com suas percepções atuais, ignorando as percepções anteriores e não armazenando qualquer informação (Russel and Norvig 2004 p. 36). Um agente reativo pode ser representado como uma função que mapeia diretamente uma configuração das percepções do agente na ação adequada a ser realizada. Uma visão mais geral propõe que um agente reativo é aquele que não raciocina simbolicamente, não transforma suas percepções em símbolos de uma linguagem, e – como não tem símbolos – não consegue aplicar operações sobre estes símbolos (Morgado 2006 p. 15). Normalmente estes tipos de agentes são usados quando o problema não exige soluções complexas ou quando se espera que o comportamento ótimo não venha de um único agente, mas sim da interação de vários agentes simples (Bordini et al. 2007 p. 86, Morgado 2006 p. 15).

2.1.3 Agente Deliberativo

Agentes deliberativos processam e armazenam informações obtidas do ambiente usando símbolos e raciocinam sobre os símbolos com regras lógicas (Wooldridge 2009 p. 50). Ao contrário do agente reativo, um agente deliberativo representa explicitamente seu ambiente, as percepções, as ações e toda a tomada de decisão por símbolos e os opera através de regras lógicas pré-estabelecidas (Morgado 2006 p. 14). Além disto, devemos considerar literalmente que um agente deliberativo *delibera* utilizando seus símbolos e operações, é capaz de analisar o ambiente e gerar uma resposta adequada. Também se utiliza o termo *agentes cognitivos* para designar este tipo de agente (Signoretti 2012).

Normalmente, para que estes agentes consigam ser racionais, eles internalizam a medida de desempenho, podendo assim medir quão bom será agir de tal forma para que maximize a medida. Por estas propriedades, duas características são possíveis e desejáveis em agentes deliberativos: (i) a primeira é que tenha memória e armazene seu passado, permitindo que consiga aperfeiçoar sua tomada de decisão de acordo com esta informação e (ii) como podem conter a medida de desempenho, sejam capazes de prever o resultado de suas ações e com isto aperfeiçoem seu comportamento, procurando a racionalidade (maximizar a medida de desempenho).

2.1.4 Agente BDI

*São agentes deliberativos que seguem um estilo de modelagem baseada nos conceitos de **crenças**, **desejos** e **intenções*** (Bordini et al. 2007 p. 16, Morgado 2006 p. 14). Não iremos entrar em detalhes aqui, pois o tópico 2.2 abaixo será dedicado a sua análise.

2.2 Agentes BDI

Ao falarmos de "agentes BDI", podemos estar nos referindo a três conceitos diferentes que são usadas nesta classe de agentes. O primeiro é o *modelo BDI*, que propõe um arcabouço conceitual para modelar agentes, baseado nos conceitos de *crença*, *desejos* e *intenções*. Estes conceitos nos ajudam a representar a solução de problemas, similarmente como o diagrama de classes de UML nos ajuda a construir um sistema orientado a objetos. O segundo é a a*rquitetura BDI* que define a semântica lógica que um agente adotará para raciocinar sobre o modelo BDI. Por fim, o terceiro é a *Implementação* BDI, que representa a implementação da arquitetura em uma linguagem real que permitirá concretizar um agente BDI.

Para modelar o comportamento de agentes complexos (inclusive humanos) foi desenvolvida por (Bratman 1987 apud JIANG 2006) uma ferramenta de abstração chamada *intentional stance*. O princípio básico desta abordagem é permitir usar termos de psicologia informal (*folk psycology*[1]), como crenças, desejos e intenções (Bordini et al. 2007 chap. 2) para a modelagem de problemas complexos. Estes termos e suas associações são eficientes para descrever um problema ou um sistema de forma abstrata, permitindo compreender sua

[1] A tradução literal seria "psicologia popular". Mas este termo é muito usado na literatura com o sentido de *psicologia informal*. Sendo aceito pela comunidade científica para representar certas teorias, mas que sem não tem embasamento teórico.

estrutura. Consideraremos neste trabalho que o modelo BDI é a forma de representação de um problema utilizando os conceitos de crença, desejo e intenções.

Esta ferramenta é útil para descrever sistemas não determinísticos e que não possuem seu funcionamento completamente conhecido. Porém é fraca e, às vezes inapropriada, quando a lógica de funcionamento do sistema é claramente conhecida e definida através de modelos formais, lógicos ou físicos (Wooldridge 2009 sec. 2.4).

Devemos lembrar também que o modelo *intentional instance* serve para predição e explicação do comportamento de sistemas, sem se preocupar como o sistema realmente trabalha (Braubach and Pokahr 2010 sec. 1.2.1). Então é preciso construir um "mecanismo de raciocínio" que opere sobre este modelo de descrição do problema/sistema para podermos desenvolver um sistema útil. Na literatura, este "mecanismo" é representado por uma linguagem lógica que define como operará o raciocínio do agente de forma genérica. Chama-se na literatura este "mecanismo" de *arquitetura BDI*.

Em tese, poderiam existir várias arquiteturas, pois sua função é meramente operacionalizar o modelo BDI. Entretanto, o padrão adotado pela literatura é o *Raciocínio Prático* (*Practical Reasoning*), proposto em uma série de trabalhos e implementações de Wooldridge e seus contribuidores (Wooldridge 2009).

Esta arquitetura se preocupa em modelar o raciocínio do agente através de ações (Wooldridge 2009 p. 66). Para isto, ele realiza duas atividades básicas: (i) a primeira é descobrir *o que* se deseja obter e (ii) a segunda é descobrir *como* obter estes desejos. Além disto, esta tarefa deve ser realizada em um tempo limitado e com recursos limitados (processador, memória, energia, entre outros) (Rao and Georgeff. M. 1995). Observe que a primeira atividade reflete o problema de tomada de decisão (deliberação) e a segunda o problema de planejamento, ambos são problemas clássicos de IA. (Bordini et al. 2007 p. 18, Rao and Georgeff. M. 1995, Wooldridge 2009 p. 66). Também observe que lidar com estas atividades em separado é importante, pois podemos rever tanto o deliberado quanto o planejado para podermos lidar com a incerteza e o dinamismo do ambiente através da reconsideração dos desejos e intenções.

Podemos resumir o fluxo de execução da arquitetura BDI da seguinte forma, como apresentado na Figura 2. A informação que o agente percebe do ambiente, isto é, suas percepções são utilizadas para atualizar as crenças do agente utilizando *Belief Review Function* (*Brf*). Em seguida, ele gera um conjunto de desejos baseado nas suas crenças atuais e nas suas intenções utilizando a *Option Function* (*Of*). Depois disto, o agente seleciona alguns destes desejos que irão formar o conjunto de intenções. As intenções serão associadas

a planos, permitindo ao agente atingir suas intenções com a *Filter Function* (*Ff*). Por fim, o agente executa as ações dos planos com a *Execute Function* (*Ef*). Isto modifica o ambiente e refletirá nas percepções do agente no próximo fluxo de execução.

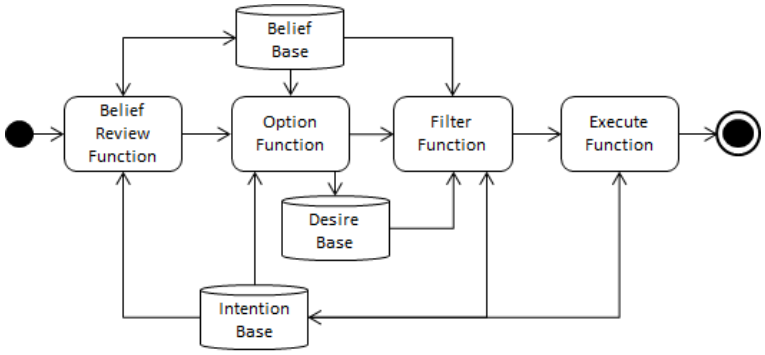


Figura 2: Arquitetura BDI

Embora a arquitetura BDI já seja bem difundida e validada para o desenvolvimento de agentes, ela não contempla os fenômenos emocionais. Os agentes emocionais ainda carecem de um formalismo para tratar o raciocínio, tomada de decisão e deliberação. Devido a isto, muitos trabalhos se dedicam à criação de agentes BDI com componentes emocionais (Hernández et al. 2004, Jiang and Vidal 2006, Neto 2010, Oliveira and Sarmento 2003, Padgham and Taylor 1997, Pereira et al. 2005, Signoretti 2012), sendo estes chamados de agentes Emocionais BDI (*Emotional BDI Agent*) (Jiang and Vidal 2006, Pereira et al. 2005), discutidos no tópico 2.6.2.

Dentre as implementações BDI, que concretizam a arquitetura BDI em uma linguagem ou *framework*, uma das mais conhecidas e talvez a primeira do paradigma BDI é o sistema PRS (*Procedural Reasoning System*), de acordo com (Wooldridge 2009 p. 83). PRS influenciou várias outras propostas que vieram a seguir (Braubach et al. 2005 p. 46). Entre elas, iremos discutir a implementação Jason a seguir.

2.3 Jason

Existem várias implementações de agentes BDI, entretanto, julgamos mais adequado adotar a plataforma Jason, pois (i) ela é um interpretador da linguagem formal AgentSpeak(L) (Rao 1996) que preserva os conceitos formais de BDI; (ii) ela é *open-source*, o que nos ajudou na economia de custos do projeto e no acesso ao código fonte, possibilitando compreender melhor como ele funcionava internamente; e (iii) Jason possui razoável documentação de apoio. Outros autores chegaram a conclusão similar (Neto 2010, Signoretti 2012). Um resumo de várias outras propostas pode também ser encontrado em (Bordini et al. 2006, Neto 2010).

O Jason é um interpretador da linguagem AgentSpeak(L) proposta por (Rao 1996). Ele possui embutido um *middleware* que permite adotar seu ciclo BDI assim como a linguagem sobre várias plataformas de agente, além de possuir sua própria plataforma. Inclusive, possui um conjunto de pacotes de desenvolvimento, IDE própria e plugins para desenvolvimento dentro da IDE Eclipse (Bordini and Hübner 2007, Hübner and Bordini 2009).

Observando a Figura 5, podemos considerar que um agente Jason é um agente deliberativo que internamente conta com um módulo de raciocínio, uma base de crenças e uma base de planos, sendo possível ao desenvolvedor especificar estas duas bases em AgentSpeak(L). Internamente ao módulo de raciocínio existem duas outras bases: a de eventos e de intenções, que refletem os eventos internos e externos ocorridos com o agente e quais as intenções ele está realizando em dado momento.

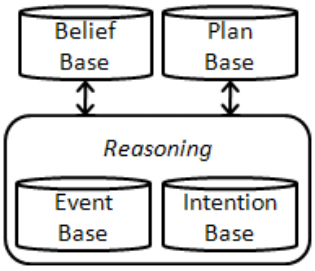


Figura 3: Estrutura geral do Jason

Para desenvolver um agente em Jason se manipula quatro artefatos: (i) código em AgentSpeak(L) especificando quais as crenças e planos do agente, (ii) extensão da classe Agent em Java que representa a classe de raciocínio em si, permitindo que o desenvolvedor customize as principais funções da arquitetura, como a função de revisão de crenças; (iii) extensão da classe AgArch em Java, que especifica características da infraestrutura na qual o agente está residindo, de tal forma que para cada infraestrutura (Centralised, Jade ou JACI) se deve implementar uma classe derivada da AgArch. Sendo esta o "corpo" do agente dentro do sistema[2]. E parte da percepção e atuação dos agentes está contida nesta extensão da AgArch. Por último (iv) a classe Environment, responsável pelo ambiente, determinando as percepções e ações que o agente terá neste ambiente.

Na Figura 4 é exibido o fluxo de execução Jason (Bordini and Hübner 2007 p. 68) usando o diagrama por nós proposto no Anexo A. Veja que as caixas em cinza indicam

[2] O modelo descrito foi simplificado e é anterior a versão 1.3.6. A partir desta versão a arquitetura de agente é representada por uma implementação do padrão *chain of responsability* onde o último item é a infraestrutura propriamente dita.

funções que não são customizáveis pela extensão da classe Agent. Descreveremos o comportamento de cada função a seguir:

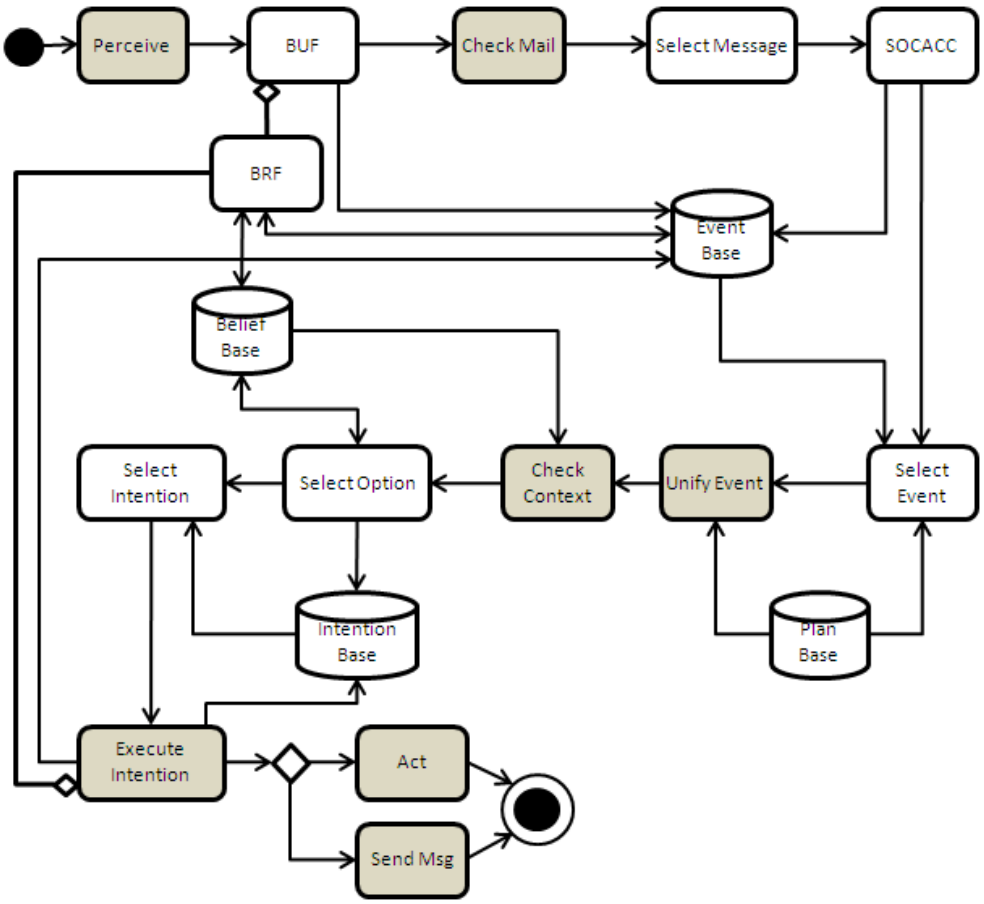


Figura 4: Fluxo de execução Jason

1. *Perceive*: Retorna as percepções do ambiente através de uma implementação da classe AgentArch.
2. *BUF* (*Belief Update Function*): Esta função recebe as percepções do ambiente e atualiza a base de crenças. Esta atualização utiliza a função *BRF* para revisar as operações de adição e exclusão. Além disto, quando é aceita a operação, esta função gera eventos representando a inserção ou remoção de crenças na base de crenças.
3. *BRF* (*Belief Review Function*): Quando ocorrer uma remoção ou adição de crenças, o sistema utiliza esta função para verificar a integridade destas operações e para gerar crenças derivadas, mantendo a integridade dos dados. Por exemplo, pode-se inibir que uma crença seja inserida contradizendo outra já existente, evitando paradoxos. Ou ainda se pode criar a crença "estou doente" quando se recebe a crença "estou com febre" e "estou com coriza". É importante destacar que esta operação sempre ocorre em uma única crença a ser inserida ou removida, ao contrário do que ocorre com a *BUF*, que

opera sobre um conjunto de percepções. A função *BRF* não faz parte do fluxo principal, sendo chamada apenas pelas funções *BUF* e *Execute Intention*.

4. *Check Mail*: Obtém uma coleção de mensagens do ambiente para que o agente possa interpretá-las. As mensagens trocadas entre os agentes não são recebidas através de *Perceive*, mas através desta.
5. *Select Message*: Dada as mensagens que o agente recebeu, seleciona uma para ser interpretada.
6. *SocAcc*: Dada a mensagem selecionada pela *Select Message*, verifica se ela deve ser aceita ou rejeitada. Após a confirmação, o Jason automaticamente torna a mensagem recebida uma espécie de meta que deve ser interpretada por um plano. Por padrão, o Jason interpreta todas as mensagens através de planos pré-especificados que manipulam a mensagem de acordo com seu conteúdo, tornando-a uma crença, uma meta ou ainda um plano, dependendo de seu conteúdo.
7. *Unify Event*: Esta função seleciona todos os planos que tratam o evento selecionado pela *Select Event*.
8. *Check Context*: Esta função verifica dentre o conjunto de planos selecionados pela *Unify Event* quais são válidos de acordo com a base de crenças (indicando que os pré-requisitos foram atendidos). Esta verificação gerará um subconjunto de planos que tratam o evento e possuem pré-requisitos válidos de acordo com as crenças do agente.
9. *Select Option*: Com os planos já selecionados, esta função seleciona qual dos planos deve ser adotado para aquele evento. Por padrão, o Jason seleciona o primeiro que chegar à lista de opções. Depois de selecionado, o par composto pelo evento e plano se torna uma intenção que é adicionada à base de intenções.
10. *Select Intention*: Como a base de intenções pode possuir várias intenções ao mesmo tempo, em cada ciclo de execução o Jason usa esta função para selecionar apenas uma intenção para ser processada. Por padrão, o Jason utiliza a política *Round Robin* (Bordini et al. 2007 p. 80) e faz um rodízio da execução da intenções, tentando fazer com que todas sejam executadas uniformemente.
11. *Execute Intention*: Dada a intenção selecionada, o agente tentará executar um passo do plano. Este plano é composto por ações internas, que atualizam o estado mental do agente ou realizam operações elementares, ou por ações externas, que modificam o ambiente. Vale lembrar que o envio de mensagens (*SendMsg*) é realizado por uma ação interna. Já as ações externas deixam o plano inativo até que seja concluída e sua execução ocorre na classe *Environment*.

2.4 Teoria da Emoção

O conceito de emoção é de senso-comum, entretanto é difícil desenvolver uma teoria científica (Morgado 2006 p. 7) e concisa (Sarmento 2004 p. 19) sobre o termo e outros fatores relacionados, não havendo uma definição padrão aceita (Jiang and Vidal 2006 p. 3, Picard 2000 p. 21, Sloman 1999 p. 1). Além disto, o número de teorias que abordam a emoção cresceu demasiadamente (Scherer 2005 p. 696) dificultando encontrar uma interseção entre elas, assim como detectar conflitos teóricos (Gadanho and Hallam 2001 p. 8).

De qualquer forma, o tema e sua investigação possuem grande importância devido a sua relação com outros processos cognitivos (Oliveira and Sarmento 2003 p. 305). O fenômeno emocional desempenha um papel crucial na habilidade de tomar decisões de forma inteligente (Damásio 2004, Sloman 1999). Alguns pesquisadores obtiveram significantes avanços experimentais na relação entre emoção e seus mecanismos sobre cognições sofisticadas, como tomada de decisão, aprendizagem, planejamento, gerência de riscos, memória e criatividade (Gadanho and Hallam 2001 p. 6, Oliveira and Sarmento 2003 p. 305).

Por conveniência chamaremos de **Teorias da Emoção** as que descrevem a relação do fenômeno emocional sobre determinados aspectos, com o objetivo de elucidar a dinâmica e as funções deste fenômeno. Não é foco deste trabalho realizar um levantamento exaustivo das teorias da emoção, visto que existem mais de 92 definições diferentes (Jiang and Vidal 2006 p. 3). Entretanto vamos abaixo descrever três classes de teorias que são adotadas por agentes emocionais: (i) teorias apreciativas, (ii) teorias multicamadas e (iii) teorias neurológicas. Julgamos adequado discutir estas três classes devido a seu exaustivo uso nos agentes emocionais.

2.4.1 Teorias apreciativas

O princípio central das teorias apreciativas (ou teorias de avaliação, do inglês *appraisal theories*) é que as emoções são eliciadas e diferenciadas através da avaliação ou apreciação subjetiva do indivíduo sobre uma situação, objeto, evento ou um agente usando um conjunto de critérios (ou dimensões) (Scherer 1999 p. 637).

O primeiro modelo adotado pelas pesquisas em IA foi o modelo OCC (Ortony et al. 1990) e acabou se tornando o modelo padrão para sistemas computacionais (Picard 2000 p. 196). O modelo OCC é um modelo cognitivo que define 22 classes de emoções, descreve formalmente quais as regras que as eliciam e propõe o relacionamento entre estes grupos. Isto vem se refletindo até os trabalhos atuais, onde alguns usam o modelo OCC original (Bates et

al. 1994, Bazzan et al. 2002, Steunebrink 2010) ou em conjunto com outros modelos para complementar partes que ele não lida, como dinâmica emocional, temperamento e personalidade (Breazeal 1998, Hernández et al. 2004, Neto 2010, Signoretti 2012, Velásquez 1998b). Deve-se destacar que este modelo não foi feito para geração (simulação) de emoções, mas sim apenas a sua descrição (Neto 2010, Picard 2000 p. 196), o que força os autores que os usam a adicionarem novos conceitos ao modelo, como regras para decaimento da intensidade emocional e da sua influência sobre o comportamento do agente. O trabalho de (Steunebrink 2010 chap. 2) apresenta uma releitura deste modelo sobra a ótica da computação, observando o modelo proposicionalmente, e não mais agrupado por eventos eliciadores como o original, o que facilita a compreensão por pesquisadores de IA e ES.

Já o modelo proposto por (Mehrabian 1996) estende o modelo OCC adicionando um modelo para modelar humor chamado PAD (*pleasure – arousal – dominance*) e um modelo para representar a personalidade chamado OCEAN. Comumente este modelo é chamado de "modelo afetivo", pois representa outras instâncias além da emoção. Alguns autores de computação tem preferido este modelo em relação ao OCC puro, devido a sua maior representabilidade (Neto 2010, Signoretti 2012).

2.4.2 Teorias multicamadas

Existe um problema nas teorias da emoção que é decidir se um determinado fenômeno emocional refere-se a estados somáticos do organismo ou também a suas avaliações cognitivas. E se sabe também que não é possível definir uma distinção absoluta (Wilson and Keil 1999 p. 272).

As teorias multicamadas (ou multiníveis) da emoção visam tratar este problema criando uma hierarquia de tratamento da informação percebida pelo indivíduo (agente), onde cada camada opera processos de avaliação em diferentes escalas de tempo e/ou diferentes camadas de sofisticação (Marsella and Gratch 2009 p. 71). Nesta ótica, a principal tarefa do estudo da emoção seria entender estas camadas e descrever a relação deles com o fenômeno emocional (Teasdale 1999 p. 665). Tal paradigma já é bem consolidado em outras áreas de pesquisa, como a psicologia clínica e estudos sobre a memória (Scherer 1999 p. 646).

O trabalho de Leventhal e Scherer (1987) (apud Van Reekum and Scherer 1997 pp. 262–263, apud Scherer 1999) sugere a adoção de "camadas de processamento". Onde uma primeira camada trataria da parcela fisiológica, reativa e inata da emoção. Enquanto que uma segunda trataria da resposta emocional aprendida. E uma terceira de respostas que exijam obter generalizações a partir dos eventos percebidos. A definição destas camadas está

relacionada a complexidade fisiológica de cada um, oriunda de uma evolução genealógica e a evolução do seu uso em relação a idade dos seres humanos, onde a primeira camada está relacionada a "capacidade emocional" de um recém nascido, a segunda camada às emoções adquiridas no decorrer do desenvolvimento e a terceira a capacidade emocional adulta (Teasdale 1999 p. 668).

2.4.3 Teorias neurológicas

Por último, as teorias neurológicas propõem uma visão multicamadas do processamento da emoção, porém, diferente da anterior, esta se embasada na fisiologia cerebral e em experimentos neurológicos. Consideram que certas áreas do cérebro responsáveis pela tomada de decisão são também responsáveis pelo processamento emocional. Os principais autores, Damásio (Damásio 2004) e Ledoux (LeDoux 2002) propõem através destas evidências que a emoção está relacionada com a tomada de decisão e raciocínio.

Damásio em (Damásio 2004) propõe a hipótese do marcador somático, que seria a rotulação de imagens mentais com o estado fisiológico do indivíduo. Por exemplo, imagine que uma pessoa se depara com um cachorro raivoso que quer ataca-la. A pessoa perceberá o cachorro pelos seus sentidos (visão, audição e olfato) e estas percepções serão rotuladas com o seu estado fisiológico que seria: sudorese e aumento do batimento cardíaco. Com isto, o cérebro tenderia a associar a percepção com as percepções geradas, criando um *cluster* de informações que podem ser resgatados posteriormente em conjunto. Caso esta mesma pessoa observar novamente outro cachorro, só que manso e brincalhão, poderá ter as mesmas sensações de sudorese e aumento do batimento cardíaco.

De acordo com Damásio, os dois episódios são fenômenos emocionais, porém de "classes" distintas, os quais chama de "dois tipos de emoções"[3]: (i) as primárias são respostas fisiológicas do organismo a um evento sendo disposições inatas a resposta de um estímulo, controladas pelo sistema límbico; e (ii) as secundárias seriam a recordação do estado fisiológico associado à imagens mentais relacionadas a eventos passados similares ao evento atual. Assim estas relações são aprendidas pelo indivíduo e envolvem categorização destes estímulos, em especial a sua avaliação boa ou ruim. O córtex cerebral, junto com o sistema

[3]Da forma como (Damásio 2004) define emoção, o termo "tipo" é mal empregado pela literatura. Um nome melhor seria "estado emocional" primário (reativo) e secundário (deliberativo). Já que não são duas emoções – primária e secundária – que são desencadeadas por um evento, mas sim uma mesma emoção que evolui no processo emocional.

límbico, é então o responsável pelas emoções secundárias (Morgado 2006 p. 54, Tomaz and Giugliano 1997 p. 410). Muitos autores de agentes emocionais usam esta teoria para dar suporte a sua proposta de emoção multicamadas (Hernández et al. 2004, Jiang and Vidal 2006 p. 1, Velásquez 1998b) ou como substrato conceitual para memória emocional (Morgado 2006, Neto 2010).

Já Joseph Ledoux realiza investigações sobre a emoção medo e descreve quais são os possíveis "percursos" desta emoção dentro do cérebro. O primeiro – caminho inferior – descreve como um estímulo de medo passa diretamente do tálamo sensorial e vai direto a amígdala, gerando respostas rápidas e simplistas no comportamento e alterações fisiológicas autônomas[4] e hormonais. Já o segundo – o caminho superior – leva a informação para o córtex sensorial que tem a capacidade de gerar uma resposta mais elaborada e precisa (Teasdale 1999 p. 677), porém de forma mais lenta. E só depois esta informação é levada para a amígdala (LeDoux and Rogan 1999, Morgado 2006 p. 56).

Um exemplo prático disto é realizar uma experiência de condicionamento utilizando um estímulo e um choque. Se este estímulo condicionado for simples (por exemplo, uma campainha) o indivíduo em teste irá usar o caminho inferior. Caso seja um estímulo mais complexo que requeira categorização, ele é realizado pelo caminho superior (LeDoux and Rogan 1999). O segundo caminho também está relacionado com a parada da elicitação de estado emocional, promovendo o autocontrole do indivíduo, de tal forma que o processo de decaimento da intensidade de uma emoção seja condicionado pelo segundo caminho (Teasdale 1999 p. 677).

Algumas críticas cabíveis as teorias neurofisiológicas são devidas a sua abordagem *bottom-up* (partem de explicações neurológicas para explicar fenômenos cognitivos complexos), o que as limitam a cenários particulares. O próprio Ledoux (LeDoux and Rogan 1999) reconhece a limitação do seu trabalho vendo que seus estudos sobre o medo não podem, sem a devida elucidação, serem generalizados para todas as emoções. O trabalho de (Damásio 2004) embora de grande influência, faz devidas propostas sobre relação entre emoção e tomada de decisão com argumentos em relação à topologia cerebral e aos experimentos realizados, mas não diz explicitamente que emoções ou quais processos emocionais são requeridos para a inteligência (Sloman 2004).

[4] p269 - autonomic

2.5 Computação Afetiva

Segundo (Neto 2010), a computação afetiva é "o estudo de sistemas computacionais (incluindo-se sistemas robóticos, de software e/hardware) capazes de reconhecer, sintetizar e expressar emoções". A relação entre o fenômeno emocional e sistemas computacionais ocorre desde o momento em que foi postulada que uma teoria geral do pensamento e da solução de problemas (planejamento / tomada de decisão), para serem completas, deverão lidar com o fenômeno emocional (Picard 1997 p. 1).

Muitos autores (Hernández et al. 2004, Jiang and Vidal 2006, Neto 2010, Steunebrink 2010 p. 9) consideram ou usam o trabalho (Picard 2000) de Rosalind Picard como a principal referência neste campo, sendo o livro que cunhou o nome *Affective Computing*. Neste trabalho, a autora define um conjunto de conceitos bases, como tipos de emoções, emoções fisiológicas e cognitivas, emoções primárias e secundárias, entre outros, que são largamente adotados atualmente no estudo de agentes emocionais.

Inclusive, ela determina quais são os limites de pesquisa da Computação Afetiva definindo quatro grupos de tarefas que envolvem a relação de seres humanos (usuários) e computadores: (i) reconhecimento de emoções; (ii) expressão de emoções; (iii) fazer os computadores *terem* emoções; e (iv) além de *ter* emoções um computador deve ter *inteligência emocional* para que seja útil ao ser humano.

Veja que *ter* emoções, inclusive em seres humanos, é um conceito de difícil definição, pois envolve vários conceitos que são ainda abertos a discussão. Picard em (Picard 2000 p. 60) para evitar cair nesta discussão, define um conjunto de cinco componentes que devem estar presentes em um sistema para considerá-lo como *tendo* emoções:

1. Possuir emoções emergentes e comportamento emocional (preferencialmente sem a representação explícita das emoções – nós discutimos melhor este conceito no tópico 2.6.1);
2. Possuir emoções primárias (rápidas) – discutimos este conceito no tópico 2.4.2;
3. Possuir emoções secundárias (cognitivas) – discutimos este conceito no tópico 2.4.2;
4. Possuir experiência emocional – isto é, o sistema é capaz de "perceber" que está no meio de um fenômeno emocional através da percepção cognitiva, fisiológica e subjetiva;
5. Ocorrer interação entre o fenômeno emocional a nível cognitivo e fisiológico – isto é, fatores emocionais ocorridos primariamente de forma cognitiva (ex. pensamos em um evento do passado que é emocionalmente relevante) eliciarão um fenômeno emocional fisiológico (ex. esta lembrança causa aumento do batimento cardíaco).

Por fim, *ter inteligência emocional*, como usado pela autora refere-se, ao mesmo conceito homônimo de *inteligência emocional* em humanos: Um ser humano adulto é "inteligente emocionalmente" se ele souber lidar com suas emoções, seja (i) através da capacidade de perceber e expressar suas emoções (ele é capaz de perceber que outro indivíduo esta triste ou feliz e é capaz de expressar suas próprias emoções de forma nem exagerada e nem contida), (ii) regulando-as para que não sejam "nocivas" a si mesmo ou ao próximo (por exemplo, caso o indivíduo sofra uma agressão, ele será capaz de conter a raiva, não "explodindo"); ou ainda (iii) utilizando-as como um motivador (por exemplo, o indivíduo frente a um fracasso em um jogo de futebol, não verá isto como "o fim do mundo" e irá treinar mais para que no próximo jogo tenha melhor desempenho).

Logo, segundo (Picard 2000), "inteligência emocional" em um computador é a capacidade deste sistema de reconhecer e expressar emoções de forma coerente e não exagerada/contida em relação ao usuário. E, principalmente, sua simulação de emoções deve ser usada para aperfeiçoar o sistema. Por exemplo: caso o computador falhe em uma tarefa, as emoções geradas – por exemplo, tristeza – devem ser usadas para que ele melhore seu desempenho alocando, por exemplo, mais recursos para conseguir realizar a tarefa que falhou. E não para que ele piore seu desempenho – por exemplo, o computador desistir da tarefa e considerar-se inapto para realizá-la.

Vamos apresentar a seguir os tópicos da computação afetiva sobre o enfoque da computação. Na Figura 5 vamos considerar quatro subáreas da computação afetiva em um gráfico ilustrativo usando o conceito de agentes. As áreas são (i) percepção do fenômeno emocional; (ii) expressão do fenômeno emocional; (iii) simulação do fenômeno emocional; e (iv) estudo do fenômeno emocional real. Neste trabalho, consideraremos que computação afetiva é a área de pesquisa interessada na investigação da interação de um sistema computacional em um ser humano ou com outro sistema que lide com o conceito de emoções sobre pelo menos uma das subáreas.

Veja que consideramos o item (iii) como simulação da emoção ao invés de *ter* emoções. Consideramos *simular* mais adequado que *ter*, pois notamos que, na prática, os autores pesquisados estavam mais preocupados com a simulação (implementar um algoritmo que simule o fenômeno emocional) do que seguir os postulados dados pela autora. A própria autora, em trabalho posterior (Picard 2003), comenta que a modelagem de afeto é um problema em aberto, de tal forma que nenhuma definição de "ter emoções" é aceita pelos pesquisadores de Computação Afetiva.

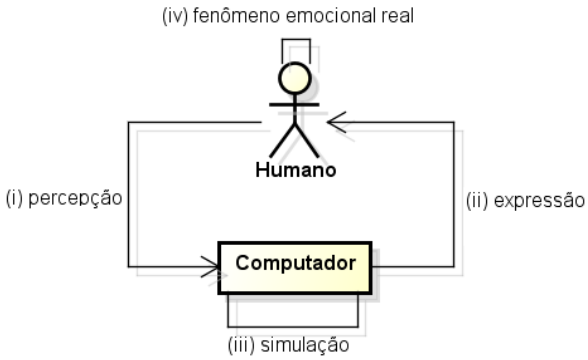


Figura 5: Tópicos da Computação Afetiva

Também se considera como área de interesse da computação afetiva o estudo do fenômeno emocional real (em humanos). Embora esta seja de responsabilidade primária da Psicologia e outras ciências humanas, consideramos necessário citá-la como área de estudo da Computação Afetiva visto que, de fato, uma série de autores consultados dedicaram alguns tópicos ao estudo da teoria da emoção (Morgado 2006, Neto 2010, Picard 2000, Scheve et al. 2006, Signoretti 2012, Sloman 1999, 2004, Steunebrink 2010).

Nossa proposta de agente emocional, assim como os outros trabalhos que propõe um agente deste tipo se encontram justamente na parte (iii) de simulação, pois eles não estão preocupados em expressar emoções para um usuário (subárea ii) e nem obtê-las (subárea i), mas estudar a melhor forma de implementar um fenômeno emocional, orientando-se na definição de uma arquitetura de agente.

2.6 Agentes Emocionais

Uma série de trabalhos começaram a adotar agentes inteligentes para solucionar os problemas da computação afetiva (Moridis and Economides 2008, Rumbell et al. 2011, Silva et al. 2006). É natural que isto aconteça, pois a interação entre um ser humano emocional com outros seres humanos, animais ou o ambiente está diretamente relacionada ao paradigma orientado a agentes (Wooldridge 2009). Inclusive, é comum ver nos modelos emocionais o uso dos conceitos de agência quando é preciso determinar a causa e a responsabilidade de dado evento (Marsella and Gratch 2009 p. 75, Ortony et al. 1990, Velásquez 1998b).

Não existe uma definição formal aceita do que seriam agentes emocionais (Scheutz 2002), mas elas convergem para a seguinte: *Agentes Emocionais* são agentes que buscam solucionar os problemas da Computação Afetiva (Camurri and Coglio 1998) de perceber, expressar, simular internamente emoções (Meyer 2004, Oliveira and Sarmento 2003) e para tomar decisões que sejam úteis ao agente (Pereira et al. 2005).

Estes agentes são usados na literatura para desenvolver personagens interativos emocionalmente credíveis para entretenimento (Bates et al. 1994, Rodrigues 2007), simular o fenômeno emocional em robôs (Breazeal 1998, Gadanho and Hallam 2001), e explorar a influência do estado emocional sobre o comportamento dos agentes (Velásquez 1998b). E dentro desta proposta de controle comportamental, tenta-se usar a influência do estado emocional sobre a capacidade de adaptação do agente e sobre os algoritmos de tomada de decisão (Jiang and Vidal 2006, Neto 2010, Oliveira and Sarmento 2003, Pereira et al. 2005, Signoretti 2012, Steunebrink 2010, Velásquez 1998b). Alguns estudos estão adotando o conceito de emoção em nível macro, auxiliando o controle de sistemas multi-agentes através de normas (Fix et al. 2006, Scheve et al. 2006, Staller and Petta 2001) e de reputação (Jiang and Vidal 2006).

2.6.1 Aspectos do fenômeno emocional

Existem três aspectos sobre o fenômeno emocional que são de uso comum nos trabalhos sobre agentes emocionais: estado emocional, dinâmica emocional e influência emocional.

O primeiro aspecto, **estado emocional**, é a representação das emoções que estão ocorrendo no agente em dado momento. Grande parte dos agentes emocionais (Breazeal 1998, Hernández et al. 2004, Neto and Da Silva 2010, Padgham and Taylor 1997, Signoretti et al. 2010, Velásquez 1998a) define explicitamente um estado emocional para o agente, diferenciando apenas na sua representação. Esta representação que define o conjunto de estados possíveis é chamada **espaço emocional**, de tal forma que o estado emocional é um elemento (ou subconjunto) deste espaço emocional. Além disto, considera-se o estado emocional como um subconjunto do estado mental por conveniência, já que faz parte da "mente" do agente.

Existem basicamente duas formas de representação do espaço emocional: um conjunto de rótulos, utilizado em (Padgham and Taylor 1997, Velásquez 1998a), ou um híbrido de rótulos e tuplas numéricas, utilizado em (Breazeal 1998, Hernández et al. 2004, Neto and Da Silva 2010, 2010, Signoretti et al. 2010). Quando o estado emocional é explicitamente definido na arquitetura, dizemos que este estado emocional é **explícito**.

Entretanto, existem outras propostas como (Morgado 2006, Staller and Petta 2000) que procuram não determinar explicitamente a emoção e são principalmente sustentadas pela ideia de que o fenômeno emocional é um fenômeno referente às alterações em vários componentes, incluindo componentes cognitivos e motivacionais (Scherer 2000 p. 71).

(Picard 1997 p. 24) define estado emocional como sendo o estado interno e dinâmico do agente que inclui o estado mental e físico. Também o trabalho (Morgado 2006) considera que existe uma classe de agentes emocionais, chamada de modelos arquitetônicos, cujo fenômeno emocional não é explicitamente definido, mas sim uma consequência da forma que a arquitetura foi definida. Assim, o fenômeno emocional é aplicado nestas arquiteturas pela dinâmica destes componentes, mas não há uma representação explícita do estado emocional. Quando se adota esta perspectiva de fenômeno emocional, dizemos que o estado emocional é **implícito**.

Entretanto, os próprios autores que propõe o estado emocional implícito reconhecem que, em alguns momentos, é necessário saber qual é este estado emocional e propõem técnicas para obtenção deste estado derivado.

Por último, algumas propostas de agente BDI (Jiang and Vidal 2006, Oliveira and Sarmento 2003, Pereira et al. 2005) concordam na existência de um estado emocional explícito, entretanto não definem qual o seu "formato", isto é, qual a natureza do espaço emocional, deixando esta tarefa a cargo do projetista. Notoriamente, como visto na Tabela 1 e levantado por (Picard 2000 p. 196) e (Signoretti 2012 p. 80), normalmente se adota o modelo OCC (Ortony et al. 1990) da emoção sozinho, com modificações informais ou em conjunto com algum outro modelo. Isso ocorre pois ele não define o aspecto dinâmico da emoção (Bazzan et al. 2002) e não representa outras entidades importantes no processo afetivo, como a personalidade (Silva et al. 2006). E mais recentemente, vêm se adotando um modelo misto feito por (Mehrabian 1996), que concatena um modelo de emoções (OCC), um modelo de humor (PAD) e um modelo de personalidade (OCEAN) (Neto and Da Silva 2010, Neto 2010, Signoretti 2012, Signoretti et al. 2010), a qual chamam de "modelo afetivo".

Arquitetura	Autor	Espaço Emocional	Teoria Base
Tok (Oz Project)	(Bates et al. 1994 p. 4)	Discreto, definição informal	Rótulos do OCC, mas a dinâmica e a influência emocional foram definidas informalmente (Bates et al. 1994)
Sem nome	(Bazzan et al. 2002 p. 114)	Discreto (Felicidade, Ressentimento, Compaixão e Raiva) com limiar de ativação	OCC
Kismet	(Breazeal 1998, 2000)	Três dimensões (Valência, Excitação, Postura) com regiões rotuladas	Emoções discretas de Ekman e espaço de emoções (Ekman and Davidson 1994, Izard 1993)
Agent Flow Model	(Morgado 2006, Morgado and	Duas dimensões e quatro rótulos	Modelo emocional de Scherer (Reekum and Scherer 1997)

	Gaspar 2005)		apoiado pelas memórias emocionais (marcador somático) de Damásio (Damásio 2004)
Sem nome	(Steunebrink 2010, Steunebrink et al. 2009)	Discreto (OCC)	OCC
Cathexis	(Velásquez 1998a, 1996, 1998b)	Discreto (seis emoções de Ekman)	Emoções discretas de Ekman e teoria da mente de Minsky (Minsky 1988)
BDIE	(Hernández et al. 2004)	Duas dimensões (Valência, Excitação) com regiões rotuladas	Mesmo de (Breazeal 1998)
Emotional BDI	(Jiang and Vidal 2006)	Não definido (Tarefa do projetista)	Não definido (Tarefa do projetista)
Sem nome	(Oliveira and Sarmento 2003)	Não definido (Tarefa do projetista)	Não definido (Tarefa do projetista)
Sem nome	(Padgham and Taylor 1997)	Discreto: Felicidade, Tristeza, Agradecido, Esperançoso, Desapontado e Culpado	(Dyer 1987)
Emotional BDI	(Pereira et al. 2005)	Não Definido (Tarefa do projetista)	Não Definido (Tarefa do projetista)
ALEC	(Gadanho and Hallam 2001)	Discreto - Felicidade, Tristeza, Medo e Raiva	(Wilson 1991)
Sem nome	(Neto and Da Silva 2010, Neto 2010)	Discreto (OCC) mapeado em um modelo contínuo PAD (Mehrabian 1996) e parametrizado por fatores de personalidade do OCEAN (McCrae and John 1992)	(Mehrabian 1996)
AAFA	(Signoretti 2012, Signoretti et al. 2010)	Discreto (OCC) mapeado em um modelo contínuo PAD e parametrizado por fatores de personalidade do OCEAN	(Mehrabian 1996)

Tabela 1: Estados emocionais adotados nas arquiteturas

O segundo aspecto do fenômeno emocional a ser definido neste trabalho é a dinâmica emocional. A partir do momento que o fenômeno emocional possui características temporais, ele possui um comportamento (dinâmica) que varia de acordo com o estado mental do agente. Então, a **dinâmica emocional** é definida como uma função que verifica o estado mental do

agente e altera seu estado emocional. Arquiteturas com estado emocional explícito tendem a possuir uma ou mais funções que avaliam certos aspectos do estado mental e atualizam o estado emocional (Scheutz 2004 p. 2). Já nas arquiteturas com estado emocional implícito, a dinâmica emocional é realizada por aspectos essenciais da arquitetura, como mecanismos de interrupção (Sloman 1999), métricas derivadas da capacidade do agente em alcançar suas metas (Morgado 2006) ou atender às normas impostas (Staller and Petta 2000). As funções que modificam o estado emocional são compostas por um conjunto de sub-funções, regras de produção ou funções reativas chamadas de **eliciadores** (*elicits*) ou lançadores (*releasers*). Elas monitoram o estado mental e as percepções enquanto modificam a base de emoções quando determinada alteração ocorre.

Alguns autores definem também um comportamento emocional chamado de **decaimento emocional**. Este comportamento faz com que uma emoção, depois de eliciada, decaia sua intensidade com o tempo (Bazzan et al. 2002, Breazeal 1998, Hernández et al. 2004, Neto 2010 p. 106, Oliveira and Sarmento 2003, Padgham and Taylor 1997, Signoretti 2012, Velásquez 1998b). Entretanto, nem todas as abordagens considerem isto (Jiang and Vidal 2006, Morgado and Gaspar 2005).

O terceiro aspecto do fenômeno emocional é a **influência emocional**, que é a influência do estado emocional em outras cognições do agente. Em particular, em alguns agentes BDI (Jiang and Vidal 2006, Neto 2010, Oliveira and Sarmento 2003, Pereira et al. 2005, Signoretti 2012) esta influência acontece pela parametrização das funções que o compõe por uma base de dados que armazena o estado emocional.

Acredita-se que estes três aspectos sejam suficientes para caracterizar e estudar o fenômeno emocional em agentes emocionais. Além disto, consideraremos que o conjunto dos três é chamado de **modelo emocional computacional**. A tarefa de modelar um agente emocional consiste em obter uma teoria que explique o fenômeno emocional, desenvolver um modelo emocional computacional em cima desta e posteriormente mapear este modelo em uma arquitetura de agente.

Usaremos estes aspectos daqui por diante para analisar os trabalhos relacionados (capítulo 3) e na definição da arquitetura (capítulo 4).

2.6.2 Agentes Emocionais BDI

Mais recentemente alguns autores vêm estendendo a arquitetura BDI para dar suporte ao desenvolvimento de agentes BDI emocionais. A maioria destes autores propõe novas

arquiteturas focando na incorporação de um "módulo emocional" na arquitetura BDI que se relaciona com os outros módulos a fim de simular o fenômeno emocional.

Eles apontam uma série de vantagens em modelar agentes emocionais baseados na arquitetura BDI, tais como:

- A arquitetura BDI possui clara distinção entre componentes importantes em um agente inteligente, facilitando a modelagem e a implementação (Hernández et al. 2004);
- A arquitetura BDI permite lidar com planos tão complexos quanto necessário (Hernández et al. 2004);
- A arquitetura BDI é inspirada em psicologia informal (Pereira et al. 2005) e permite o desenvolvimento de tomada de decisão similar a humana (Hernández et al. 2004), pois reflete o processo prático de raciocínio humano (Jiang and Vidal 2006 p. 2) ;
- É uma arquitetura já amplamente aceita para agentes e existe uma gama de implementações e frameworks implementados (Jiang and Vidal 2006 p. 2, Neto 2010 p. 34, Pereira et al. 2005 p. 1, Signoretti 2012);
- Existem *framerworks* lógicos formais que descrevem a arquitetura BDI e permitem a verificação de seu funcionamento (Pereira et al. 2005);
- Permite desenvolver comportamento adaptativo (Neto 2010 p. 34);
- A arquitetura BDI permite implementar dois conceitos fundamentais em IA: deliberação e planejamento (Bordini et al. 2007), o que permite relacionar emoções a estes dois conceitos;
- A arquitetura BDI é compatível com outras arquiteturas de agentes já propostas, como a reativa, em camadas e lógico (Jiang and Vidal 2006, Neto 2010); de tal forma que se for preciso criar um agente emocional e reativo, poderá se usar uma ferramenta que misture BDI e um módulo Emocional;
- É natural e formalmente permitido estender a arquitetura BDI, já que ela é baseada em conceitos de psicologia informal. No caso de emoções, é vantajoso adicionar um "módulo emocional" que contenha explicitamente representações do estado emocional, pois isto é embasado em várias teorias de emoções (Norling 2004).

Entretanto, os autores de agentes emocionais BDI apenas contemplam em suas propostas uma faceta do fenômeno emocional. Por exemplo, o trabalho de (Hernández et al. 2004) somente contempla a influência do estado emocional do agente sobre as crenças e os desejos, enquanto as abordagens de (Jiang and Vidal 2006, Oliveira and Sarmento 2003) focam seus modelos na influência das emoções do agente sobre as intenções. O proposto em

(Pereira et al. 2005) chega a criar um modelo de influência das emoções sobre os outros componentes da arquitetura BDI, todavia não deixa explícitas as razões para tal feito. Por último, o trabalho de (Neto 2010) é bastante completo e apresenta bons resultados no uso de um módulo emocional na arquitetura de agentes Jason, entretanto apresenta uma série de limitações devido a ser uma proposta menos generalista.

Esta parcialidade das propostas indica a necessidade de uma arquitetura unificada e genérica, que abarque todas as possíveis influencias que um módulo emocional possa realizar sobre o resto da arquitetura BDI.

3 Trabalhos Relacionados

Iremos analisar as abordagens de agentes emocionais relacionadas com nossa proposta. A maioria destas abordagens são arquiteturas Emocionais BDI (Hernández et al. 2004, Jiang and Vidal 2006, Neto 2010, Padgham and Taylor 1997, Signoretti et al. 2010, Steunebrink 2010), enquanto outras são apenas arquiteturas emocionais, mas que contribuem efetivamente para nossa com algoritmos ou modelos que serão mapeados no modelo BDI (Breazeal 1998, Morgado 2006, Oliveira and Sarmento 2003, Sloman 1999, Velásquez 1996).

Outras propostas não foram exaustivamente utilizadas, pois algumas focavam na simulação de um modelo emocional específico (Bazzan and Bordini 2001, Steunebrink 2010), o que difere da escolha que fizemos de deixar esta tarefa a projetista de agente. Outras não se preocupavam com a influência da emoção no resto da arquitetura (Marsella and Gratch 2009), que é justamente a meta principal desta dissertação. E ainda algumas se preocupavam com a formalização lógica do fenômeno emocional (Meyer 2004, Steunebrink 2010), o que colide com nosso objetivo de deixar o modelo emocional em aberto na arquitetura.

Nossa análise será orientada a discutir cada módulo da arquitetura BDI e cada tópico abaixo discutirá uma abordagem e seus subtópicos discutirão um componente BDI ou possíveis módulos de percepção e atuação, além do módulo emocional. Quando esta arquitetura não for BDI, será feita a mesma organização e será apontado como esta proposta trata do respectivo módulo em discussão. Quando um subtópico de um módulo não for especificado, indica que a abordagem não trata deste módulo.

Após apresentar os trabalhos, no tópico 3.11, iremos fazer uma comparação destas abordagens.

3.1 BDIE (Hernández et al. 2004)

A arquitetura BDIE (Hernández et al. 2004) apresenta uma proposta não conservativa, incorporando um componente emocional que influencia as percepções do agente, a geração de seus desejos e também a execução de seus comportamentos. Em adição, as crenças, metas e

emoções correntes do agente contribuem com a revisão destas emoções. Todavia, esta arquitetura não se preocupa com a influência das emoções sobre a F*ilter Function*, como outros autores fizeram (Jiang and Vidal 2006, Pereira et al. 2005, Steunebrink 2010).

Um ponto importante nesta proposta é que eles deliberadamente não criaram uma influência da emoção sobre as intenções e as ações, pois consideraram que mesmo que seja pertinente ter esta influência para obter uma resposta rápida do sistema, isto iria prejudicar a modularidade do sistema e seria teoricamente incompatível com a teoria emocional adotada.

3.1.1 Módulo de percepções

Os autores consideram que o estado emocional interfere no processo perceptivo. Por exemplo, dado o estado emocional *raiva*, o sensor de visão deve funcionar limitadamente (focando somente no objeto que causou raiva). Entretanto, os autores não definem explicitamente uma função e base de percepções, não dão maiores detalhes de como é realizada esta influência e a influência não é demonstrada em seu exemplo.

Inversamente, consideram que a percepção influencia o estado emocional, mas para isto ele insere esta percepção na base de crenças. De tal forma que só é possível influenciar as emoções se esta percepção for "envelopada" em uma crença.

3.1.2 Módulo de crenças

Este módulo foi alterado para suportar a teoria emocional multicamadas. Na sua abordagem, as emoções secundárias e ternárias seriam compostas por emoções primárias (fisiológicas/reativas) e crenças. Para dar suporte a esta característica, o autor considera que o conceito de emoções multicamadas é implementado na base de crenças e as crenças podem ser de três tipos, refletindo três camadas: a primeira camada são as percepções dos agentes em sua forma original, a segunda camada são as crenças compostas por relações e transformações em cima das crenças do primeiro nível e a terceira camada seria a combinação das crenças do segundo nível (Hernández et al. 2004 p. 4). A base de crenças é monitorada por avaliadores classificados também por níveis: os avaliadores primários são simples funções de verificação associados com um valor de limiar, tal como o modelo de (Velásquez 1998b). Os avaliadores secundários, que monitoram as crenças de segundo nível, são mais complexos e possuem comportamento deliberativo. Segundo autor, também deveriam existir avaliadores ternários, entretanto não foram implementados.

Entretanto, os autores não especificam como os outros elementos da arquitetura BDI (por exemplo, os planos do agente) influenciam na revisão das crenças. Além disso, a proposta de estender o conceito de níveis às crenças foi uma representação criada pelos autores e não possui uma motivação clara, não apresentou nenhum ganho à arquitetura e aumentou a complexidade do problema analisado. De acordo com o exemplo usado para validação da arquitetura o motivo é apenas classificar as crenças similarmente como as emoções são.

3.1.3 Módulo de desejos

Quanto aos desejos do agente, eles são suas metas (*goals*) e suas variáveis homeostáticas (*homeostatic variables*). Uma variável homeostática é um valor que varia com o decorrer do tempo. A ela estão associados limiares inferiores e superiores, que indicam um estado "indesejável" do agente, promovendo algum tipo de resposta. O caso clássico de variável homeostática é a fome. Que aumenta com o tempo e quando extrapola certo limiar, promove um conjunto de fenômenos e comportamentos para que o indivíduo se alimente, diminuindo a fome e a fazendo voltar a valores entre os limiares (estado de homeostase).

Os desejos são classificados por suas *prioridades* e *importâncias*[5]. O agente sempre selecionará os desejos com maior prioridade e depois selecionará os com maior importância. Isto é, se um desejo A é mais prioritário que um desejo B, ele será selecionado, mesmo que B seja mais importante que A. Além disto, quando um desejo não consegue ser satisfeito, todos os desejos que possuem prioridade igual ou menor a dele serão desativados. Segundo o autor, isto permite que o sistema atenda aos desejos que realmente se relacionam com o atual contexto. Os níveis de prioridade são dinâmicos e podem variar no decorrer da execução do agente.

Quanto à função de geração de desejos desta arquitetura, ela é definida tal como a BDI original e os autores aludem que as emoções primárias ativam certos desejos para gerar resposta rápida na arquitetura e que todas as emoções são potenciais fontes de desejos. Os desejos também influenciam as emoções, principalmente durante sua satisfação e insatisfação (promovendo alegria ou tristeza, respectivamente).

Este esquema do módulo de desejos é inovador, pois unifica o conceito de motivações – ou *drives*, segundo (Velásquez 1996, 1998b) e (Breazeal 1998) – e o conceito tradicional de desejos da arquitetura BDI. Entretanto, podemos enumerar as seguintes problemáticas desta

[5] A diferença semântica entre importância e prioridade não está clara em (Hernández et al. 2004).

proposta: (i) o uso de prioridades é confuso, (ii) não é explicada qual é a diferença entre prioridade e importância e (iii) não há indício que esta abordagem melhore a capacidade do agente de atenção em determinado problema ocorrendo no mundo.

3.1.4 Módulo emocional

Esta proposta define um modelo emocional constituído por três classes de emoções. Emoções primárias são as relacionadas com tarefas fundamentais para a vida. São rápidas e pré-definidas, sendo ativadas quando uma situação particular ocorre. Elas são importantes dentro do sistema já que podem gerar desejos prioritários que interrompem planos correntes para priorizar a execução de planos que lidem com esta emoção. Medo e surpresa podem ser consideradas emoções primárias. As emoções secundárias são relacionadas com cognições de alto-nível que interpretam as percepções do agente e promovem a alteração do estado emocional. Este tipo de emoção possui funções como a comunicação e interação com outros agentes. Normalmente a alegria, tristeza e raiva podem ser consideradas secundárias. Por último, as emoções ternárias são resultado e cognição reflexiva como vergonha e apreço, por exemplo. Entretanto tais emoções não foram implementadas na arquitetura proposta. A principal motivação para esta divisão das emoções é melhorar o desempenho da avaliação das mesmas, onde as emoções primárias gerariam respostas rápidas a eventos, tomando o controle do sistema em situações de emergência/fundamentais (Hernández et al. 2004 p. 3). E como comentado, para os autores emoções constituem a base de crenças e elas próprias podem ser aglomerados de crenças, o que exigiu alterar a base de crenças, como comentado acima.

O **espaço emocional** é híbrido e composto por um espaço contínuo com os eixos Excitação e Valência/Favorabilidade (*Arousal* e *Valence*) particionado por regiões discretas que representam emoções discretas, como o proposto por (Breazeal 2000). Um **estado emocional** é um ponto neste espaço contínuo e a região que o contém.

A **dinâmica emocional** é representada por eliciadores que monitoram as crenças do agente e que são divididos de acordo com o tipo de crença (veja o tópico crença acima) e por um *fator de decaimento* associado a partição onde a emoção está representada decrementando a emoção automaticamente de acordo com este fator.

Os eliciadores primários monitoram as crenças primárias e alteram as emoções primárias, sendo normalmente simples computações que monitoram se um dado fator excedeu um limiar. Quando há uma eliciação primária atribui-se um ponto absoluto no espaço de Excitação x Valência. Já os eliciadores secundários monitoram as crenças secundárias e alteram as emoções secundárias. Estes, após eliciados, podem consultar outras bases da

arquitetura, como a de desejos, e modificar o estado emocional atual. Quando um eliciador primário é acionado, não se faz a computação dos secundários, sobre a justificativa de que uma emoção primária representa uma situação de emergência que o sistema deve lidar com prioridade.

Entretanto, o autor não define explicitamente se existe uma ou mais funções de revisão de emoções, havendo apenas determinado que existam estes eliciadores que devem ser executados após a revisão de crenças. Entendemos então que ele embora use os conceitos de emoções multicamadas, adota somente uma função de revisão de emoções.

O autor também considera que esta eliciação, além de monitorar as crenças, pode ser influenciada pelas metas (Hernández et al. 2004 p. 3). Entretanto não faz nenhuma formalização disto, definindo os parâmetros da função. Entendemos então que esta função de revisão de revisão de emoções é influenciada pelas emoções, desejos e crenças.

O modelo proposto em (Hernández et al. 2004) é inovador, porém se limita em vários aspectos:

(i) O uso do espaço emocional híbrido é mais versátil que o uso das emoções discretas, porém manter os rótulos discretos facilita a computação da influência das emoções sobre as outras instâncias psíquicas. Mas a definição das partições não é trivial e foi realizada de forma informal.

(ii) A definição do processo de eliciação utilizando apenas crenças inicialmente e dando prioridade às emoções primárias, embora faça com que o agente lide melhor com situações de urgência, o impede de fazer computação de emoções secundárias. O próprio conceito de emoções secundárias é diferente do usado por outras abordagens, visto que está relacionado com a complexidade do algoritmo de eliciação e suas prioridades, e não com sua estrutura como feito por (Velásquez 1998b). Tal proposta parece interessante, já que os níveis de emoções visam obter economia de recursos computacionais e melhor respostas emergenciais, mas a fundamentação teórica para esta adoção é precária e não houve experimentos para comprovar sua efetividade.

(iii) O exemplo apresentado descreve que o agente deve reagir à iluminação do ambiente é muito simples e não permite uma melhor investigação das avaliações cognitivas nesta arquitetura. O exemplo indica a adoção da função emocional de comunicação. Por último, o autor considera que apenas as crenças e os desejos influenciam as emoções ignorando assim a influência das intenções (tal com feito por (Jiang and Vidal 2006, Neto 2010)) e que as emoções só influenciam nos desejos e nas ações do agente.

3.2 EBDI (Jiang and Vidal 2006)

Em (Jiang and Vidal 2006), os autores apresentam outra extensão conservativa da arquitetura BDI com componente emocional. Eles adicionam um novo módulo emocional com duas camadas emocionais e associam estas camadas a funções de revisão para que o componente emocional seja capaz de acessar os outros componentes e inferir o novo estado emocional. Seu foco foi expandir o trabalho de (Pereira et al. 2005), validando as influências propostas e procurando construir uma arquitetura independente do modelo emocional adotado.

Os autores citam que a emoção controla a priorização dos desejos e que o sucesso ou o fracasso de um desejo pode eliciar emoções. Entretanto não apresentam nenhuma relação entre desejos e emoções na sua arquitetura. Desta forma, a emoção apenas influencia as crenças e as intenções. As percepções, ações e desejos não são influenciados.

O interessante desta arquitetura é que ela propõe considerar emoções primárias e secundárias de forma separada, o que permite representar o gerenciamento de recursos e a diferenciação das emoções primárias e secundárias explicitamente. E, embora defina esta separação, não define um modelo emocional específico, permitindo o desenvolvedor embutir qualquer modelo que deseja e seja mais adequado ao seu cenário.

3.2.1 Módulo de percepções

Os autores consideram que a função de revisão de crenças deva ser dividida em três: *brf-see*, que lida com as percepções do agente; a *brf-msg*, que lida com as mensagens trocadas entre agentes e a *brf-in*, que realiza a revisão das crenças do agente, considerando o estado emocional e as intenções para isto. Neste caso, ele explicita uma função revisão de percepções e mensagens, mas não há influência de nenhuma base influencia estas funções.

3.2.2 Módulo de crenças

Como comentado em 3.2.1, função de revisão de crenças é segmentada em três e somente a *brf-in* é influenciada pela base de crenças, de intenções e de emoções para realizar o seu trabalho. E o comportamento desta função um *hot-spot* a ser definido pelo projetista.

3.2.3 Módulo de intenções

Quanto ao módulo de intenções, ele mantém as influências da arquitetura BDI e adiciona a influência da base de emoções. Segundo os autores isto serviria principalmente para criar uma relação entre intenções e crenças, permitindo prevenir, pelas emoções, a inconsistência entre estas. Por exemplo, se um agente está realizando uma intenção, porém crê que esta não lhe trará benefícios, ele "se sentirá" triste (Jiang and Vidal 2006 p. 4).

Eles consideram que as intenções irão influenciar as duas Funções de Revisão de Emoções. Entretanto, a influência da intenção sobre as crenças se apresenta confusa no decorrer do trabalho (Jiang and Vidal 2006 p. 4): os autores descrevem que as intenções não irão influenciar as crenças diretamente, mas sim indiretamente pelas emoções. Apesar disto, no diagrama apresentado em seguida a função de revisão de crenças contemplativa é influenciada pela base de intenções e no exemplo proposto as intenções são utilizadas como forma de priorização das crenças.

3.2.4 Módulo de emoções

O módulo emocional é inspirado na teoria de emoções primárias e secundárias de (Damásio 2004), onde os autores interpretaram que as emoções primárias são as "sentidas primeiro" representando uma primeira resposta a uma situação/evento. Já as secundárias são geradas a partir das emoções primárias e de inferências mais complexas, geradas com o resultado da tomada de decisão do agente e podendo substituir uma emoção primária. O conceito de marcador somático não é representado na arquitetura, apenas a divisão das camadas. Entretanto foi parcialmente desenvolvida no protótipo, visto que uma emoção é associada ao agente de origem, e esta associação será usada posteriormente para alterar o comportamento do agente.

Devido a isto, a **dinâmica emocional** é representada por duas funções, cada uma responsável por um tipo de emoção e sofrendo influência de diferentes fontes de dados. A *Primary Emotion Update Functon* recebe as percepções do agente, as emoções anteriores e as intenções e as verifica para atualizar as emoções primárias. Já a *Secondary Emotion Update Function* verifica o estado emocional anterior, as crenças e as intenções para gerar o novo estado emocional. Veja que não é definido como ocorrerá a eliciação nem qualquer outro processo de controle temporal da emoção, pois os autores consideram isto dependente de domínio.

Inversamente ao que (Hernández et al. 2004) realiza, este trabalho não ignora a computação das emoções secundárias caso uma primária seja eliciada. Ao invés, ele sempre gera o novo estado emocional com as emoções secundárias e, caso este novo estado seja diferente do anterior e haja tempo para realizar uma nova deliberação, repete-se a revisão das crenças, a revisão dos desejos e a seleção das intenções utilizando o novo estado emocional.

Quanto ao **espaço emocional**, os autores não definem um a priori, podendo ser utilizada qualquer forma de representação. No seu exemplo, ele considera as emoções como sentenças lógicas em primeira ordem, associando uma emoção ao agente ao qual é dirigida a emoção (o agente odeia o outro agente, por exemplo) ou ao estado do mundo (o agente está com raiva do seu concorrente, por exemplo). Da mesma forma, o **estado emocional** será derivado de um elemento deste espaço customizado.

A **influência emocional** é bem definida e representada pela influência de uma base de emoções sobre as funções de revisão de crenças e de filtro de intenções. Além disto, no exemplo descrito no artigo o sistema emocional possui a função avaliativa, trabalhando como um sistema de reputação distribuído (De Weerdt et al. 2007) onde um agente tem um sentimento em relação aos outros, que pode ir do amor até o ódio, representando o quanto este outro agente foi confiável durante as interações. A função de atualização de emoções primárias verifica se o que um agente vizinho informa condiz com o que o agente está percebendo do mundo e a função de atualização de emoções secundárias realiza a mesma inferência, só que utilizando as crenças do agente.

Por exemplo, em um primeiro instante, um agente A recebe uma mensagem do agente B dizendo o dólar caiu e custa R$ 1,6. Porém o agente A percebe que o dólar custa na verdade R$ 1,7. Na primeira função de revisão de emoções, ele recebe o dado percebido (U$ 1,7) e a mensagem de B (dólar caiu para U$ 1,6) e como B errou, ele passa a "desconfiar" de B. Posteriormente, na segunda função de revisão de emoções, o agente tem acesso a suas crenças sobre o preço do dólar no decorrer do tempo e descobre que o dólar estava custando R$ 1,3. Isto é, o dólar está aumentando de preço e o agente B mentiu quanto ao dólar ter caído, o que fará o agente A "desconfiar muito" de B.

Esta proposta segue uma visão também proposta por em (Oliveira and Sarmento 2003, Pereira et al. 2005) de que o espaço emocional deve ser algo definido pelo desenvolvedor e não pela arquitetura. Isto aumenta a abrangência da arquitetura, visto que uma proposta de modelo é mais adequada que outra dependendo do problema a ser atacado.

O exemplo apresentando no artigo (Jiang and Vidal 2006) utiliza tanto a emoção quanto a intenção como formas de priorização de crenças (dando uma ordem a elas para

facilitar seu acesso pelo resto do sistema). O uso das emoções, embora validado, não foi comparado com o desempenho de agentes não-emocionais.

3.3 Abordagem proposta em (Neto and Da Silva 2010, Neto 2010)

A proposta de Neto e da Silva (Neto and Da Silva 2010, Neto 2010) procura adicionar um modelo emocional que engloba emoções, humores e personalidade (Mehrabian 1996) e os conceitos de marcador somático de Damásio na implementação BDI Jason. Sua modificação é conservativa, pois insere um módulo ao Jason e cria um conjunto de regras de eliciação chamadas durante a revisão de crenças e cria três filtros influenciados pelas emoções: um para crenças, outro para planos e um terceiro para percepções, que eliminam itens não compatíveis com o estado emocional atual.

Esta proposta apresenta a desvantagem de usar o modelo emocional fixo. Os autores também implementam sua solução diretamente sobre o Jason, o que dificulta transferir sua abordagem para outra plataforma. A base de desejos não é influenciada pelas emoções, como proposto por (Hernández et al. 2004, Pereira et al. 2005). E por fim, sua dinâmica emocional só é influenciada pelas percepções e usando apenas uma função, não aproveitando as metodologias multicamadas, como propostas em (Hernández et al. 2004, Jiang and Vidal 2006, Velásquez 1998b).

3.3.1 Módulo de percepções

As percepções irão eliciar um conjunto de emoções e serão rotuladas por elas. Estas percepções, após rotuladas, serão filtradas de acordo com este rótulo e sua similaridade com o estado emocional corrente. Logo, o filtro de percepções é influenciado pelas emoções e as percepções, somente.

Esta proposta de influência emocional sobre as percepções é clara e funcional. Além de ser embasada em teorias que explicam a percepção em seres humanos, diferente do realizado por (Lino 2006), que faz uma abordagem informal desta influência, e de (Signoretti 2012) que apenas usa a intensidade emocional na influência. Não há detalhes se outras instâncias cognitivas influenciam a percepção, como a intenção ou as crenças.

3.3.2 Módulo de crenças

A base de crenças é a mesma do Jason. Os autores não definem uma influência das crenças sobre as emoções e as emoções não participam na criação de crenças. Por outro lado,

ele tem uma visão similar a de (Oliveira and Sarmento 2003) de que as crenças possuem rótulos contendo informação sobre o estado emocional. Porém, da mesma forma que as percepções, são filtradas de acordo com a similaridade deste rótulo com a base de emoções. Devido a isto, podemos considerar que a base de emoções influência a revisão de crenças.

Esta interpretação do rótulo de crenças é inovadora em relação ao trabalho que (Oliveira and Sarmento 2003) já que o usa como mecanismo de seleção de crenças e não como um algoritmo de *clusterização* e indexação das crenças para futuro resgate. O exemplo que propõe usa o filtro de crenças para que o agente tenha comportamento mais próximo que um ser humano e não diz nada se tal filtro ajudaria na tomada de decisão ou outro quesito de otimização.

3.3.3 Módulo de intenções

Quanto ao módulo de intenções, foi criado um filtro na seleção de planos baseado, segundo o autor, na teoria do marcador somático de (Damásio 2004). Assim um plano possui um marcador representando um valor positivo ou negativo, indicando a qualidade do estado emocional atingido após as várias execuções deste plano (Neto and Da Silva 2010 p. 50).

Na proposta teórica deste filtro os autores propõem a seleção dos planos com marcação mais positiva (Neto 2010 p. 68) e na implementação se seleciona o plano com marcação mais afim com o estado emocional (Neto 2010 p. 83). Esta relação de afinidade entre marcador somático e estado emocional é dependente do domínio já que reflete relações definidas no modelo emocional adotado. No exemplo oferecido pelo autor em (Neto 2010 p. 74), caso o estado emocional do agente seja arrogante, o agente irá selecionar um plano entre "testemunhar" e "não testemunhar" com maior número de marcações positivas, entretanto multiplicará o número de marcações do plano "testemunhar" por três, dando mais chances deste ser selecionado.

Com isto podemos dizer que a Função de Execução é influenciada pela base de intenções e pela base de emoções.

Este filtro de planos é uma proposta concreta e operacional. Os seus experimentos verificaram a efetividade desta no ambiente do "Dilema do Prisioneiro Iterado" em alterar o comportamento do agente. Entrementes, existem dois problemas na sua abordagem:

O primeiro é que não ficou claro em qual momento do ciclo de execução um marcador somático de um plano é atualizado e como é detectado. Na definição da arquitetura ele diz que no filtro de planos são selecionados os planos com marcação mais positiva. Entretanto na

implementação, em vez disto, ele seleciona um plano com marcação positiva ou negativa dependendo do estado emocional corrente.

A segunda é a fuga do conceito de emoção. O marcador somático foi mal definido e a forma como foi utilizada se entende apenas como uma métrica indicando qual plano foi mais eficiente que outro em alcançar suas metas (Neto 2010 p. 68). A própria implementação realiza simplificações em relação ao modelo proposto: não há associação exata do marcador ao plano, mas sim um marcador somático geral que indica a desempenho do agente no problema atacado. Além disto, se a seleção do plano utilizar apenas a métrica de sucesso, nada garante que só porque um plano não foi bem sucedido até agora, não o será no próximo cenário.

3.3.4 Módulo de emoções

Foi adicionado um módulo afetivo baseado no modelo afetivo de (Mehrabian 1996), conforme discutido no tópico 2.4.1. Considera-se então que o **estado/espaço emocional** é composto por estes três componentes (emoção, humor e personalidade), sendo adotado três modelos cognitivistas para cada componente.

Quanto a **dinâmica emocional**, se define que a eliciação das emoções ocorre logo após a atualização das crenças junto com a atualização do humor e usa regras de produção derivadas do modelo OCC. Entretanto, considera-se que somente as percepções influenciam as emoções, pois o autor considera que apenas os eventos do Jason gerados pelas percepções eliciam emoções. E o decaimento da emoção ocorre com o tempo utilizando uma função de declínio.

Esta abordagem apresenta a implementação do modelo emocional bastante completa, se comparada aos trabalhos de arquiteturas anteriores (Gadanho and Hallam 2001, Hernández et al. 2004, Lino 2006, Padgham and Taylor 1997, Velásquez 1996). O modelo é embarcado em uma implementação de agente BDI bem consolidada, neste caso o Jason, o que não acontece nas outras arquiteturas BDI emocionais.

A interação entre humores e emoções é parecida com a proposta de (Breazeal 1998, Hernández et al. 2004) em seu modelo emocional híbrido. Entretanto nesta arquitetura a eliciação ocorre na parcela discreta do modelo através das regras de produção do modelo OCC e a parcela discreta do modelo que possui associada valores que indicam o quanto aquela emoção é pertinente, diferentemente do adotado pelos outros modelos, que não armazenam esta emoção. Isto permite que o agente esteja em um estado emocional

momentâneo (alegre), mas seu humor permaneça entediado (valores de prazer, excitação e dominância baixos).

3.4 Emotional-BDI (Pereira et al. 2005)

O trabalho (Pereira et al. 2005) apresenta uma extensão da arquitetura BDI conservativa, adicionando um módulo de percepção e um módulo emocional. Além destes, ele desenvolve um módulo responsável por interpretar o estado mental e determinar quais os recursos e capacidades o agente possui. Segundo os autores, é necessário explicitar estas entidades para permitir saber quais os meios (*means*) os agentes possuem para atuar sobre o ambiente.

Um problema central nesta abordagem é construir um agente emocional onde o módulo emocional desta proposta influencia todos os outros, porém não existem bases teóricas nem justificativas para explicar estas influências. Da mesma forma, os autores não apresentam um exemplo de uso da abordagem, o que impede chegarmos a alguma conclusão sobre a sua validade. Por isto, não abordaremos a influência entre a emoção e as crenças, desejos e intenções.

Por outro lado, eles propõem um módulo explícito de percepções com justificativas razoáveis e com influência emocional. E um módulo emocional distinto e sem modelo definido, como (Jiang and Vidal 2006), para não tornar a proposta dependente de domínio. Por isto, discutimos estes dois pontos a seguir.

3.4.1 Módulo de percepções

Esta abordagem considera que é preciso interpretar um estímulo vindo do ambiente para construir conceitos que façam sentido ao raciocínio do agente ao invés de trabalhar diretamente com estes estímulos. Então eles decidem criar um módulo de sensação e percepção (*Sensing and Perception Module*). Este módulo é composto por uma função de percepção (*Perception Filter*) responsável por atribuir rótulos semânticos a uma sensação e uma função de sensação (*Sensing Filter*) responsável por transformar as sensações obtidas em outras mais coerentes ao raciocínio do agente.

Este módulo de percepções é influenciado pelas crenças, emoções, capacidades e recursos do agente. As crenças são importantes para a interpretação semântica da informação recebida pela *Perception Filter* e os recursos e capacidades são usadas pela *Sensing Filter*

para realizar filtro de percepções de acordo com os recursos disponíveis e capacidades que o agente possui no momento.

Com esta função explicitamente definida, o projetista do agente possuirá melhores bases para especificar o processo perceptivo do agente. Todavia tal modelo não possui um embasamento teórico e sua escalabilidade é limitada. Como ele centraliza a função de percepção em dois componentes, que tratam de todos os sensores do agente (aquele que realiza a seleção e aquele que atribui informação semântica), isto tornará tal módulo sobrecarregado, tendo de lidar com sensações de naturezas diferentes.

3.4.2 Módulo de Emoções

O módulo emocional é uma caixa preta que deve ser definida pelo projetista, não especificando como são o **estado** e a **dinâmica emocional**. Logo, é o projetista de agente, em tempo de projeto, que deve definir qual o modelo emocional. Novamente, como não foi levantada nenhuma referência sobre modelo emocional, o autor não nos dá pistas de como implementar este módulo e como ocorreria a **influência emocional**, embora ele considere que este módulo influencie todos os outros módulos e é influenciado por todos os outros.

Devido a este problema fundamental, o trabalho (Jiang and Vidal 2006) relê este trabalho e atribui conteúdo teórico a algumas influências e elimina outras.

3.5 AAFA (Signoretti 2012)

A arquitetura proposta por (Signoretti 2012, Signoretti et al. 2010) chamada *AAFA – Affective Attention Focus Agent –* é implementada sobre a plataforma BDI Jason (Bordini et al. 2007). Seu foco foi adotar o modelo afetivo de (Mehrabian 1996) para fazer filtro de percepções, tanto qualitativo (preferir certas percepções de acordo com a emoção) quanto quantitativo (eliminar algumas percepções menos significantes de acordo com o estado emocional), baseado principalmente no trabalho já feito por (Morgado 2006) e similarmente como feito por (Neto 2010). Embora bem sucedido, sua proposta se limita ao filtro de percepções e não busca investigar a influências das emoções nas outras entidades BDI. Por isto, apenas discutiremos o módulo de percepção.

A autora de (Lino 2006) propõe um agente similar anteriormente a este proposta, também baseado em (Morgado 2006). Porém sua proposta é muito mais simples e limitada que esta, sem se preocupar com os detalhes formais do modelo emocional e sem ter um rigor para experimentação.

3.5.1 Módulo de percepções

Este módulo foi desenvolvido pelos autores baseado na proposta de (Morgado 2006). Eles também adotam o Jason como arquitetura base e o modelo emocional (Mehrabian 1996), como (Neto 2010) realizou. Entretanto, os autores ao invés de focarem na seleção por pertinência do estado emocional eliciado em as percepções do agente, eles usam uma métrica derivada do estado do humor do agente simbolizando quanto o agente está angustiado ou relaxado com o ambiente para determinar quantos elementos percebidos irão para a base de crenças. Quanto mais próximo o valor desta métrica estiver de zero, maior será esta parcela das percepções a serem consideradas (Signoretti 2012 p. 177). Além disto, eles definem que o agente rotule suas percepções com o que eles chamam de *behavior* – conjunto de ações que o agente executa em dado momento e seus relacionamentos de dependência (Signoretti 2012 p. 64). E consideram também que um *behavior* pode influenciar a percepção do agente, procurando selecionar quais são mais pertinentes para o *behavior* atual. Se considerarmos um *behavior* é uma espécie de plano orientado a metas em conjunto com informação semântica sobre as percepções do ambiente, podemos dizer que em sua proposta a percepção é influenciada pelas intenções, crenças e emoções do agente.

Esta proposta obteve bons resultados na otimização do tempo gasto pelos agentes em uma situação estressante, onde a poda das percepções usadas permitiu que o raciocínio fosse mais rápido, sem perda drástica de desempenho dos agentes. Entretanto, deve ser lembrado que no modelo proposto pelos autores não é só a emoção que realiza a poda das percepções, há anteriormente todo um processo de priorização das percepções utilizando como critério de priorização tanto a relevância da percepção para o agente quanto a taxa de mudança da percepção em relação ao instante anterior. Ambas as priorizações são importante no processo de seleção e não sofrem influência da emoção.

3.6 Abordagem proposta em (Oliveira and Sarmento 2003)

Embora os autores em (Oliveira and Sarmento 2003) não proponham uma extensão explícita da arquitetura BDI, eles descrevem a influência da emoção sobre os algoritmos de percepção do agente, e sobre os algoritmos de planejamento. Em adição, propõem a influência dos desejos sobre a função de revisão de emoções.

Não iremos discutir o módulo de desejos, pois os autores, embora considerem que deva existir a influência das emoções neste módulo, eles não definem como isto deva ocorrer.

3.6.1 Módulo de crenças

Não é proposto um módulo de crenças em si, mas sim o uso de duas otimizações para a base de crenças do agente: Rotulação Emocional e Congruência de Memória por Humor (*Emotional Tagging* e *Mood-Congruent Memory*). A primeira visa rotular as crenças com o estado emocional no momento que se adquiriu aquela crença. Esta rotulação permite a seleção de crenças que deveriam ser mantidas por longo tempo, por curto tempo ou serem eliminadas. A motivação para isto é a conjectura de que memórias com maior intensidade emocional deveriam ser persistidas por mais tempo. Já a Congruência de Memória por Humor, dado que as crenças estão rotuladas, visa agrupar as crenças, formando *clusters*, para aperfeiçoar a busca de dados nesta base, com a conjectura de que eventos que promovam um estado emocional exigirão informações obtidas em outro estado emocional similar.

O caso de teste proposto pelos autores para estas otimizações foi fracamente discutido e somente a Rotulação de Crenças utilizada. Não existe em (Oliveira and Sarmento 2003) resultados de desempenho desta arquitetura e nem foi comentado se a Rotulação de Crenças modificou o comportamento do agente, comparado a não usar esta estratégia para julgar a persistência de crenças. Neste trabalho também não foi feita nenhuma consideração sobre a Função de Revisão de Crenças e muito menos qual a influência das emoções neste processo.

3.6.2 Módulo de intenções

Esta proposta considera o conceito de planejamento para que o agente realize suas atividades. Não há seleção explícita de intenções, mas sim a execução de planos de acordo com um desejo do agente e uma influência da emoção sobre este no algoritmo de planejamento.

Consideram que um algoritmo de planejamento pode ser influenciado pelo estado emocional através da modificação dos seus seguintes atributos: (i) granularidade, indica se é preferível um plano com poucos passos ou muitos passos; (ii) tempo de execução, indica o tempo permitido ao sistema de planejamento para elaborar um plano; (iii) conjunto de ações (*Action Set*), indica qual conjunto de ações o plano poderá utilizar; e (iv) estilo de algoritmo/heurística usado, i.e., se ele deve ser um algoritmo exato ou uma heurística aproximativa. Por exemplo, (segundo os autores) durante o período que o humor “ansiedade” está ativo o algoritmo de planejamento (i) deveria diminuir a granularidade dos planos, pois a emoção ansiedade prevê um maior detalhamento das atividades sendo realizadas pelo agente, (ii) deve incrementar o conjunto de ações, pois isto ajudará a descobrir boas oportunidades de

planos; (iii) deve usar heurísticas conservativas, visto que o agente deve ser pessimista diante das suas ações; e (iv) deve aumentar o tempo permitido ao planejamento, permitindo melhor refinamento dos planos.

No mesmo trabalho também é proposto que os algoritmos adotados por um agente para solucionar um problema podem ser classificados pela sua complexidade e seu nível de influência pela emoção. Por exemplo, um algoritmo heurístico possui baixa complexidade computacional, mas pode ter grande influência da emoção. De acordo com os autores, dependendo do estado emocional um agente poderá preferir um dado tipo de algoritmo (se em pânico, ele preferirá algoritmos reativos).

No exemplo de (Oliveira and Sarmento 2003) é adotado tanto o uso de calibragem do algoritmo de planejamento quanto a seleção dos algoritmos de acordo com sua classe através do estado emocional. Estas duas técnicas, embora bem explicadas, são postas muito abstratamente em relação ao estado emocional. Isto é, utiliza quatro estados emocionais distintos e define qual a influência de cada um sobre o algoritmo de planejamento e o de seleção de classe de algoritmos, entretanto não há indicação de literatura que possa ser usada pelo projetista para relacionar as emoções com as técnicas adotadas. Além disto, a técnica de calibragem de planejamento é informal. Também não é provado se tal técnica promove alteração do comportamento ou melhor desempenho do agente.

3.6.3 Módulo de emoções

Sua proposta para o módulo emocional foca em dois pilares básicos: (i) a emoção tem função básica de orientar o sistema de geração de metas do agente (função regulação de comportamento); e (ii) não é definido um estado/espaço emocional próprio, mas sim apenas as influências da emoção em outros aspectos da arquitetura e a dinâmica emocional.

Para os autores, a emoção reflete a capacidade do agente de satisfazer seus desejos/metas em relação ao estado do ambiente e é representada por uma visão funcional conhecida como potencial de satisfação (*potential copping*) (Oliveira and Sarmento 2003 p. 306). Em outras palavras, a eliciação das emoções depende das chances do agente em satisfazer suas metas. Como por exemplo, se o agente recomendador tem como meta encontrar livros que um usuário do site possivelmente esteja interessado, mas o agente crê não haver nenhum livro no estoque que esteja de acordo com o perfil do usuário, o agente pode desencadear um processo emocional, como a frustração.

Quanto a **dinâmica emocional**, eles consideram que a eliciação emocional é parametrizada por uma meta, o estado mental do agente e suas percepções do ambiente. Por

isto, se mapearmos sua proposta para a BDI, consideramos que as crenças, metas e percepções influenciam a emoção.

Também não é definido o **estado/estado emocional**, é uma generalização da proposta de (Velásquez 1996, 1998b), pois os autores consideram que o estado emocional em si como algo flexível, devido ao fato de existirem várias propostas para eliciação das emoções, similarmente ao concluído por (Jiang and Vidal 2006, Pereira et al. 2005). Entretanto, o decaimento emocional é fixo e implementado como um fator numérico que decrementará a intensidade da emoção eliciada, sendo fixo e definido pelo projetista do agente. Importante notar que os autores consideram que um fenômeno afetivo pode ser uma emoção, um humor ou um temperamento, dependendo da sua intensidade emocional e do tempo gasto para a intensidade decair.

Esta proposta apresenta um avanço em relação as anteriores, visto que generaliza o estado emocional. Entretanto, quanto a dinâmica emocional, não está claro se as regras de eliciação são literalmente parametrizada pelas metas ou se somente as regras devem ser definidas de acordo com as metas. Além disto, o próprio uso da meta como fator chave para a eliciação da emoção limita a arquitetura para casos que não necessariamente exijam uma meta explícita.

Considere por exemplo que estamos fazendo uma simulação onde um agente deseja ir a um teatro assistir a uma peça. Ele possui a meta "Assistir peça" e está executando o plano "Ir ao Teatro; Comprar Bilhete; Entrar na sala". Quando vai executar a ação "Entrar na sala", ele é notificado pelo recepcionista (outro agente) que é proibido o uso das suas vestimentas – por exemplo, short – e não poderá entrar na sala. O agente, devido a esta restrição, sentirá a emoção vergonha. Veja que a emoção vergonha não está relacionada com a falha em concluir o plano nem de alcançar a meta "Ver peça", mas sim com a restrição que lhe foi atribuída. Em uma arquitetura de agente, a recepção desta restrição poderia gerar diretamente uma emoção, sem que haja relação com as metas que estavam sendo realizadas nem a necessidade de criar uma meta "artificial", que nosso exemplo poderia ser a meta de "Não sofrer restrições ou punições".

Outro detalhe é que os autores deixam fixo o decaimento da intensidade emocional usando um valor que é subtraído do valor da emoção por instante de tempo. Embora a maioria dos trabalhos considerem isto (Hernández et al. 2004, Neto 2010, Padgham and Taylor 1997, Signoretti 2012, Steunebrink 2010, Velásquez 1998b), não há bases teóricas para corroborar com o uso desta abordagem para todos os domínios. Os trabalhos (Jiang and Vidal 2006, Lino 2006, Morgado 2006) são exemplos que não adotam este comportamento.

Além disto, os autores propõem uma taxonomia que possui importante valor para a área de agentes já que permite uma melhor análise do problema, em especial no momento em que o projetista deve especificar as emoções do agente. Por outro lado, a definição desta taxonomia não tem embasamento teórico e existem outras propostas mais concretas e gerais, como a de (Scherer 2005).

Por último, a rotulação de crenças e o agrupamento por humor não são bem definidos e possuem uma série de problemas quando são implementados. Detalhes são explicados no tópico (4.3).

3.7 Abordagem proposta em (Steunebrink 2010)

Esta abordagem se preocupa primeiramente em estudar o modelo emocional OCC e modelá-lo logicamente, permitindo que se derivem teoremas e se consiga assim provar consequências sobre este modelo (Steunebrink 2010 p. 9). Além disto, ele define uma dinâmica emocional com intensidade contínua e uma influência emocional baseada nos conceitos de tendência de ação e Marcador Somático de Damásio. Isto tudo sobre uma variante da arquitetura abstrata BDI KARO.

A influência emocional proposta foca principalmente no módulo de intenções. Entretanto, como é uma definição lógica, é possível que esta emoção influencie os outros módulos sem maiores problemas, porém não haverá um embasamento teórico/semântico para isto. Por isto, apenas iremos discutir sua influência sobre o módulo de intenções e o seu módulo de emoções.

Talvez o trabalho de Steunebrink (2010) seja o mais completo e denso de agentes emocionais, pois propõe soluções para o estado, a dinâmica e a influência emocional usando um formalismo estritamente lógico. O seu ponto forte é fazer toda sua proposta logicamente e propor minuciosamente a definição do modelo OCC sobre a arquitetura BDI. Entretanto, justamente por sua complexidade, torna difícil a interpretação das suas propostas, o que pode desencorajar um projetista de agentes emocionais a adotá-la. Como fica preso ao modelo OCC, sua abordagem não lida com humor e personalidade explicitamente, como as abordagens (Neto 2010, Signoretti 2012), e nem é independente de domínio, como os autores (Jiang and Vidal 2006, Oliveira and Sarmento 2003, Pereira et al. 2005) tentaram realizar. A sua influência emocional é parcial, sem propor uma solução para a influência da emoção sobre o módulo perceptivo e, principalmente, o de desejos.

3.7.1 Módulo de intenções

Os autores consideram que as emoções influenciam primordialmente a manipulação de ações e intenções, e por isto podemos considerar que na sua proposta as emoções influenciam a *Execute Function* e a *Filter Function*. Eles se baseiam no conceito de regulação emocional de e tendência de ação de (Frijda 1987). Sua proposta realiza duas coisas: a primeira cria uma relação entre as classes de emoções e as ações, e definindo quanto uma ação é capaz reduzir a intensidade de uma emoção, para voltar a um valor homeostático. A segunda é propor que o estado emocional, mais precisamente a emoção esperança ativa a geração de planos, a emoção medo ativa a análise dos planos correntes para prever falhas de ações, afim de eliminar o plano, a emoção alegria promove a execução de planos e a emoção angústia promove a reconsideração de planos, interrompendo a execução do plano atual para permitir o agente "ter atenção" a outro plano ou substituir ou interromper o plano atual.

Embora estas propostas sejam importantes, a primeira é puramente teórica e não é possível implementá-la diretamente devido ao seu indeterminismo. Já a segunda pode-se implementar, porém lida apenas com a influência de quatro tipos de emoções sobre o módulo de intenções. E, segundo os próprios autores, se poderia explorar outras influências.

3.7.2 Módulo de emoções

Quanto ao **estado emocional**, ele adota o modelo emocional OCC, que, segundo o autor, possui as qualidades de (i) possuir uma classificação clara dos "tipos" de emoção; (ii) uma descrição concisa dos eliciadores emocionais e (iii) são adotados conceitos similares ao do modelo BDI, o que tornaria mais simples seu uso. Entretanto, como a lógica adotada no seu trabalho é proposicional, o trabalho prefere redefinir todo o modelo focando sua descrição das emoções em termos proposicionais, ao invés de focar-se em quais eventos as eliciam.

Quanto a **dinâmica emocional**, elas são eliciadas pelas percepções do agente e pelas motivações (*cocerns*). Devemos lembrar que as percepções para este trabalho são as percepções do modelo OCC, que podem ser consequências de eventos ações de agentes ou atributos destes objetos/agentes. E as motivações – sobre sua ótica – não são metas, mas sim são literalmente o raciocínio sobre estas percepções em relação aos desejos do agente. Por exemplo, se determina que uma percepção é positiva ou negativa se esta percepção é desejável (está na base de desejos). Por isto, podemos considerar que as emoções são influenciadas pelas percepções e desejos.

O processo de atualização e decaimento emocional são definições lógicas de um valor de intensidade não negativo atribuído a uma classe de emoção. Definindo, inclusive, funções de decaimento e valores de limiares. Tais conceitos já foram explorados nos trabalhos de emoções fisiológicas como (Breazeal 1998, Gadanho and Hallam 2001, Velásquez 1996) e largamente adotado na maioria dos modelos de agentes emocionais BDI (Hernández et al. 2004, Neto and Da Silva 2010, Pereira et al. 2005, Signoretti 2012).

3.8 Cathexis (Velásquez 1996) e Kismet (Breazeal 1998)

Iremos tratar do trabalho de agente emocional proposto por Velásquez em seus trabalhos (Velásquez 1996, 1998b) e do trabalho de Breazeal em (Breazeal 1998, 2000) conjuntamente. Somos levados a fazer isto porque o trabalho de Breazeal é uma extensão do de Velásquez, com algumas adições. Iremos abaixo descrever como as propostas contribuíram para agentes emocionais de acordo com o módulo BDI que pode ser beneficiado com esta contribuição.

3.8.1 Módulo de desejos

A proposta de (Velásquez 1996) possui o conceito de *Drives* que representam o desejo que impulsiona o agente a realizar uma ação (Velásquez 1998b p. 2). Embora seja um desejo, um *drive* tem a semântica diferenciada. Por exemplo, se um agente tem o *drive* fome ele possui a necessidade de comida. A fome de tempos e tempos será ativada e este momento dependerá de vários fatores, como as ações que o agente tomou –o agente gastou muita energia em suas tarefas– ou as informações que recebeu do ambiente –ele pode ter visto um bife saboroso, o que aumentaria sua fome. O conceito de *drive* aqui tem uma semântica genérica, que vai desde as necessidades inatas da espécie, aos moldes da etologia, necessidades fisiológicas, necessidades sociológicas ou psíquicas.

Desta forma, os autores propõem que um *drive* seja influenciado pelas percepções do agente e quando ativo influencie os comportamentos que satisfazem este *drive*. Um *drive* também pode influenciar as emoções, visto que a urgência de uma necessidade pode promover emoções, como medo e raiva. Já o trabalho apresentado em (Breazeal 1998) propõem uma abordagem similar, porém neste caso o *drive* possui um fator de incremento, fazendo-o aumentar com o decorrer do tempo, mesmo que não haja percepções que promovam isto.

Tal proposta de uso de *drives* tem duas importâncias. A primeira é que tal conceito é relevante em agentes emocionais, principalmente para os que implementam a função de expressão facial, visto que o uso de valores associados às necessidades do agente e a ativação de comportamentos quando eles extrapolam um limiar é uma alternativa simples e mais flexível do quer usar meras regras de produção e máquinas de estados. A segunda relevância do uso de *drives* é o fato de contemplar um aspecto não contemplado diretamente pelos desejos da arquitetura BDI. Um *drive* possui uma dinâmica homeostática que o associa a um valor e é permanente ao longo da execução do agente, enquanto que os desejos da arquitetura BDI não podem representar esta dinâmica diretamente. Representar um *drive* na arquitetura BDI original é algo complexo, já que é preciso modelar este comportamento com crenças (para armazenar o valor do *drive*) e ações do agente (para alterar este valor).

3.8.2 Módulo de ações

Estes trabalhos representam as ações do agente pelas entidades "comportamento"[6]. No caso de (Velásquez 1996), as percepções, as motivações (que consideramos metas) e as emoções podem ativar um comportamento através de uma influência numérica. Isto é, quando uma percepção ocorrer e uma motivação ou emoção estiver ativa, elas gerarão um valor numérico equivalente a sua intensidade e comunicarão os comportamentos influenciados. Estes comportamentos também possuem relações de excitação e anulação com outros, de tal forma que se um comportamento estiver ativo, ele inibe ou excita outros (também numericamente).

Ambos os autores consideram que para cada emoção existe uma expressão facial que a representa de tal forma que quando uma emoção é ativada, a expressão facial associada deve ser exibida. Quando várias emoções estão ativas, deve-se usar uma política de seleção para selecionar apenas uma, pois o modelo adota uma forma adaptada de emoções discretas. Podemos considerar que uma expressão facial é uma ação e que as emoções influenciam estas ações.

[6] Consideramos um comportamento como uma ação devido a seu caráter elementar de atuação nestas arquiteturas. Isto é, o comportamento é a ação mínima e indivisível que um agente pode representar, raciocinar e executar.

3.8.3 Módulo de emoções

Velásquez em (Velásquez 1996) considera o **estado emocional** composto por um conjunto discreto de famílias de emoções que compartilham similaridades quanto a sua ativação e reação, tais como eventos antecedentes, expressão, resposta comportamental e padrões fisiológicos. O autor definiu seis famílias de emoções: raiva, medo, angústia/tristeza, alegria/felicidade, desgosto e surpresa. A escolha destas famílias foi feita devido às evidências que sugerem sua universalidade. No trabalho posterior (Velásquez 1998a) ele faz uma proposta para implementação do Marcador Somático de Damásio, considerando que emoções primárias seriam as emoções discretas já propostas e emoções secundárias seriam associações entre estas emoções com outras ou com crenças. A criação destas relações tentam relacionar as emoções primárias com os estímulos e memórias do agente que geraram a emoção. Com estas associações, quando um estímulo ocorrer novamente, o agente busca a entidade estimuladora na memória e irá ativar as emoções associadas a elas, priorizando as associações com maior peso.

Quanto a **dinâmica emocional**, o valor das emoções é incrementado por lançadores (*releasers*) equivalentes a eliciadores. Por exemplo, um agente presa percebe a presença de um predador e elicia a emoção medo. E este valor é decrementado dado um fator a cada instante de tempo. Por exemplo, com o passar do tempo, o medo da presa pelo predador diminuiu caso este não venha a atacá-la.

Os eliciadores/lançadores podem ser de quatro tipos: (i) neural, como fatores neurofisiológicos que afetam o agente, como temperatura, taxa de hormônios, dieta, entre outros; (ii) sensório-motor, como o estado físico do agente, como expressão, postura-corporal, potencialidade de ativação de músculos, etc; (iii) motivacional, tal como os *drives* (variáveis homeostáticas) e outras emoções; e (iv) cognitivo, que trata de liberadores que realizam cognição sobre as crenças, desejos, memória e eventos.

Podemos então considerar, de acordo com estes lançadores, que as emoções são influenciadas pelas percepções, crenças e desejos.

O modelo emocional definido por Velásquez é simples e baseado em modelos fisiológicos primitivos. Por um lado, isto permite modelar sistemas pequenos rapidamente e de forma simples. Entretanto, como discutido em (Morgado and Gaspar 2005 p. 64), há uma dificuldade inerente na especificação das relações entre emoções, motivações e ações, visto que não há uma relação simbólica, mas sim numérica, exigindo uma calibração complicada do sistema, em especial quando o número de emoções e motivações cresce. Além disto, a rede

associativa entre emoções e suas entidades eliciadoras não apresenta detalhamento nem verificação para que se possa ser replicada.

3.9 Abordagem proposta em (Padgham and Taylor 1997)

Esta proposta foi uma das primeiras a relacionar emoção e agentes deliberativos e a primeira arquitetura BDI estendida com um módulo emocional. É conservadora, pois mantém a arquitetura BDI intacta e implementa os requisitos de um agente emocional sem alterá-la, representando tais requisitos como crenças e regras de produção. Logo, não há uma influência explícita da emoção sobre o módulo de desejos e intenções. Somente para o módulo de crenças.

Como é conservativa, a proposta permite adicionar aspectos emocionais a qualquer agente baseado em BDI sem grande impacto. Além disto, ela se torna a torna simplista e intuitiva, facilitando assimilação e desenvolvimento.

3.9.1 Módulo de crenças

A base de crenças armazena as emoções do agente. Já a função de revisão de crenças possui regras que gerarão crenças de acordo com o estado emocional para ativar os desejos, seguindo as teorias de (Frijda 1987). Logo, o estado emocional influencia os desejos encapsulados em crenças que os representam.

3.9.2 Módulo de emoções

Os autores propõe um modelo simples de emoção que considera o **estado emocional** composto por pares de emoções, como os pares vergonha e orgulho, felicidade e tristeza ou amor e ódio. Estes pares possuem um valor associado que funciona como um "medidor"[7] das duas emoções a cada instante de tempo. Uma emoção associada ao lado negativo do "medidor" está ativa quando o valor do "medidor" extrapola um limiar negativo. Já a sua antagonista positiva é ativada quando o "medidor" extrapola um limiar positivo. Assim, o estado emocional é o conjunto das emoções derivadas destes "medidores". Estes pares são implementados como crenças e regras de produção dentro da base de crenças.

A **dinâmica emocional** define como os valores dos pares de emoção decaem ou incrementam com o tempo, voltando para o valor neutro (zero). A eliciação das emoções se dá

[7] Do inglês *Gauge*.

pelo tratamento de três tipos de eventos. O primeiro tipo agrupa eventos que ocorrem com o agente que desencadeiam uma emoção, como a morte de um ente querido, por exemplo. Já o segundo tipo são eventos de sucesso ou falha na obtenção de uma meta, podendo causar, por exemplo, felicidade ou tristeza. O terceiro tipo são eventos que são reconhecidos como oportunidades ou empecilhos para o agente atingir suas metas. Como exemplo deste tipo de evento podemos citar a quebra do carro quando o agente está indo (ou está com a meta de ir) ao trabalho. Vendo esta classificação, podemos concluir que as percepções (eventos) e desejos eliciam as emoções.

Todavia, esta proposta de modelo emocional possui um problema conceitual importante: o uso de polaridade entre emoções positivas e negativas é restritivo, porque impede que emoções ditas "antagônicas" coexistam e permutem sua ativação. Em muitas situações é importante que o agente seja capaz de representar emoções antagônicas ao mesmo tempo, normalmente relacionadas a contextos diferentes. Por exemplo, na dualidade "felicidade-tristeza", um agente pode estar feliz dado a ocorrência de um evento e triste dado a ocorrência de outro evento. Utilizando a abordagem de (Padgham and Taylor 1997), o segundo evento anularia a felicidade anterior ou a diminuiria, mas não seria possível armazenar a informação sobre a emoção de felicidade relacionada a um evento e a de tristeza relacionada a outro evento. Além do mais, pelo fato do seu modelo possuir emoções discretas e valores associados, sofre dos mesmos problemas dos modelos de emoções fisiológicas, como comentado por (Morgado 2006), onde o aumento da complexidade do modelo emocional acarreta em uma dificuldade na implementação devido a calibração dos pesos das influências das emoções sobre outras instâncias da arquitetura.

3.10 *Agent Flow Model* (Morgado 2006)

O autor (Morgado 2006 p. 196) desenvolve um algoritmo genérico de seleção de itens do estado mental de acordo com a afinidade deste estado emocional associado a este item e a variação do estado emocional no decorrer do tempo. Levando como parâmetro de comparação o caráter positivo (é benéfico a algum fator do agente) ou negativo (é maléfico a algum fator do agente) deste elemento, este algoritmo pode ser utilizado em duas entidades importantes: as percepções, tentando selecionar as com mais afinidade com o estado emocional; ou as intenções, que só serão executadas se possuírem afinidade com o estado emocional. Desta forma, podemos dizer que as emoções influenciam na seleção de percepções e de intenções.

O autor utiliza um modelo emocional implícito. Isto é, ele não define uma base e funções de revisão emocional, mas sim considera que o estado emocional é derivado do estado mental do agente. Mais precisamente, a emoção é derivada da afinidade dos componentes mentais com as metas do agente.

A vantagem desta proposta se encontra no seu caráter implícito e genérico: este algoritmo de seleção não é dependente de domínio devido ao fato de que o estado emocional associado a dada sensação é determinado pela "distância" da sensação em relação às metas do agente (apenas exigindo que exista uma função de comparação entre uma motivação e o outro elemento). As desvantagens são as seguintes. A proposta, por ser puramente matemática, (i) trás complexidade a implementação, pois necessita mapear os conceitos psicológicos neste modelo matemático exige maior conhecimento do projetista ou desenvolvedor, (ii) a definição desta função de distância pode não ser trivial e (iii) apenas se considera o caráter positivo-negativo da emoção em relação as metas. O que não permite lidar, por exemplo, com a priorização de percepções relacionadas a uma meta crítica.

Os autores de (Signoretti et al. 2010) também questionam que a proposta de (Morgado 2006) apenas lidam com modelos fisiológicos e apreciativos, e menosprezam modelos que lidam com humores (*mood*) e personalidade. É indevido afirmar isto, visto que podemos desenvolver humores e personalidade usando modelos emocionais, já que, como vários autores já citam (Morgado 2006, Neto 2010, Oliveira and Sarmento 2003), humores e personalidades podem ser interpretados como espécies de emoções com maior período de pertinência.

3.11 Abordagem proposta em (Sloman 1999)

O autor deste trabalho procura definir uma arquitetura teórica de agente biológico e artificial dividindo os processos cognitivos em camadas e fluxos. Os fluxos determinam processos de interação com o ambiente que passam pela percepção, cognição e atuação. E as camadas se relacionam com a natureza e a complexidade da informação que está sendo tratada. Segundo os autores a emoção seria um mecanismo de interrupção que auxilia esta arquitetura a se adaptar ao ambiente e a tomar decisões.

A proposta deste autor é complexa e abrange todo o processo de eliciação emocional e considera todas as etapas de percepção, raciocínio e atuação do agente de uma forma homogênea. Porém é restrita, pois considera que a emoção é mero mecanismo de alarme e modela as emoções mais complexas como mecanismos da arquitetura, seguindo a Teoria do

Marcador Somático de Damásio (veja tópico 2.4.3). Na perspectiva de ES, ela pouco ajuda, visto que é uma proposta teórica e não voltada para uma formalização explícita.

Entretanto, vamos comentar a visão deste autor sobre o processo perceptivo do agente.

3.11.1 Módulo de Percepção

Para o autor, em agentes muito simples, as percepções podem ser realizadas por algoritmos rudimentares, pertencentes ao que o autor chama de camada reativa da percepção. Percepções mais complexas ou ricas devem ser tratadas por algoritmos mais elaborados, como processamento simbólico, sendo estes pertencentes a camada deliberativa da percepção. Por último, em caso de percepções muito complexas, é preciso adotar algoritmos mais ricos como aprendizagem ou sistemas especialistas para transformar a percepção. Estes últimos são pertencentes a camada meta-gerencial da percepção.

Embora o autor considere que estes algoritmos e processos não são isolados, mas sim um aglomerado de operações interconectadas, ele também considera que se pode isolar fluxos de dados para fins de modelagem. Estes fluxos estão relacionados a natureza e origem da percepção e, muitas vezes, podem atravessar uma ou mais camadas da percepção, de tal forma que parcelas diferentes da percepção sejam tratadas por camadas diferentes, dependendo da complexidade desta parcela.

É lembrado em (Sloman 1999 p. 6) que este processo perceptivo desempenha principalmente a função de filtro de atenção existente entre a camada reativa e deliberativa, de forma que a camada reativa apenas passaria informação para a deliberativa, com o objetivo principal de poupar recursos limitados da camada deliberativa (Sloman 1999 p. 19). Autores como (Morgado 2006, Neto 2010, Signoretti 2012) adotaram esta perspectiva em suas arquiteturas.

3.12 Comparação

A Tabela 2 destaca resumidamente como cada um dos trabalhos relacionados apresentados acima aborda as influências das bases de dados nas funções da arquitetura BDI, e nas funções de revisão de emoções e percepções. São elas: *Brf – Belief Review Function*. *Of – Option Function*, *Ff – Filter Function, Pf – Perception Function*, *1ª Emof* e *2ª Emof – 1ª* e *2ª Emotional Function*. Foram definidas duas funções emocionais, como proposto por (Jiang and Vidal 2006) para podermos representar influência sobre a segunda função emocional. As outras abordagens que possuem influência sobre o módulo emocional apenas consideram que

existe a *1ª Emof*. As colunas indicam as bases de crenças, desejos, emoções e a informação sentida (ou base de percepções). Nas células estão os números dos tópicos das abordagens que consideraram que existe a influencia da base na função.. As células inscritas com BDI indicam que esta influência da base sobre a função já existe na arquitetura BDI original. E as células inscritas com BASIC são influências triviais, pois referem-se à influência de uma determinada base sobre a função cujo objetivo é a atualização desta própria base.. No nosso caso, a influência da *Emotional Base* sobre as funções *1ª Emof* e *2ª Emof*, por exemplo.

	Base				
	Percepções	Crenças	Desejos	Intenções	Emoções
Pf	BASIC	3.4 3.5	-	3.5	3.4 3.3 3.5 3.6 3.10
1ª Emof	3.2 3.3 3.4 3.6 3.7 3.8 3.9	3.1 3.6 3.8	3.1 3.6 3.7 3.8 3.9	3.2	BASIC
2ª Emof	-	3.2	-	3.2	BASIC
Brf	BDI	BDI	-	BDI	3.3 3.6 3.9
Of	3.8	BDI	BDI	BDI	3.1
Ff	-	BDI	BDI	BDI	3.2 3.3 3.6 3.7 3.10
Ef	-	-	-	BDI	3.7 3.8

Tabela 2: Comparação Função x Base

4 Unified Emotional BDI Architecture

Vamos neste capítulo definir nossa proposta de arquitetura de agente emocional unificada. A metodologia que usaremos será *top-down*. Isto é, primeiro iremos propor uma visão macro desta arquitetura para posteriormente ir analisando os módulos que a constituem. Achamos esta abordagem mais adequada visto que já é adotada por outros autores (Jiang and Vidal 2006, Neto 2010, Pereira et al. 2005) e é mais adequada do que analisar algoritmos pontuais para a implementação da emoção (Hernández et al. 2004, Oliveira and Sarmento 2003, Padgham and Taylor 1997, Velásquez 1996), pois permite uma melhor visualização dos componentes sobre uma perspectiva de ES.

Nossa arquitetura define a função *Agent Function* descrita no tópico 2.1. No início do ciclo de execução a arquitetura recebe um conjunto de sensações e, após passar for um conjunto de funções, gera uma ou mais ações a serem executadas. Na Figura 6 temos um diagrama para ilustrá-la. Tal arquitetura é derivada do modelo BDI básico discutido no tópico 2.2, composto pelas funções *Belief Review Function*, *Option Function* e *Filter Function* e pelas bases *Belief Base*, *Desire Base* e *Intention Base*. A arquitetura segue ainda a proposta dada por (Jiang and Vidal 2006), onde se insere duas funções emocionais (*First Review Function* e *Second Review Function*) e uma base para guardar as emoções (a *Emotion Base*). A ordem destas funções e as influências entre os módulos serão explicadas nos próximos tópicos.

Foram adicionados ainda dois módulos à arquitetura original BDI: o módulo de percepção composto pela *Perception Function* e pela base *Perception Base*, que visa explicitar a influência da emoção sobre a percepção, como citado por vários autores (Hernández et al. 2004, Jiang and Vidal 2006, Lino 2006, Morgado 2006, Neto 2010, Pereira et al. 2005, Signoretti 2012); e o módulo de atuação composto pela *Execute Function* e pela base *Action Base*, que visam explicitar a execução de planos e suas ações, assim como a influência da emoção sobre as ações. Alguns autores descrevem esta influência direta da emoção na atuação, em especial (Velásquez 1996) e (Breazeal 2000), mas nenhum explicita

um módulo a parte responsável por esta tarefa. Embora a arquitetura original BDI proponha uma função de execução de ações oriundas de planos, ela não trata da representação explícita da operacionalização dos planos e nem da influência da emoção sobre a execução das ações e planos. Além disto, foi adicionada a base *Resource Base* para auxiliar o processo de percepção e atuação discutidos a seguir.

Nos tópicos a seguir iremos explicar cada uma das entidades envolvidas nesta arquitetura e seus relacionamentos.

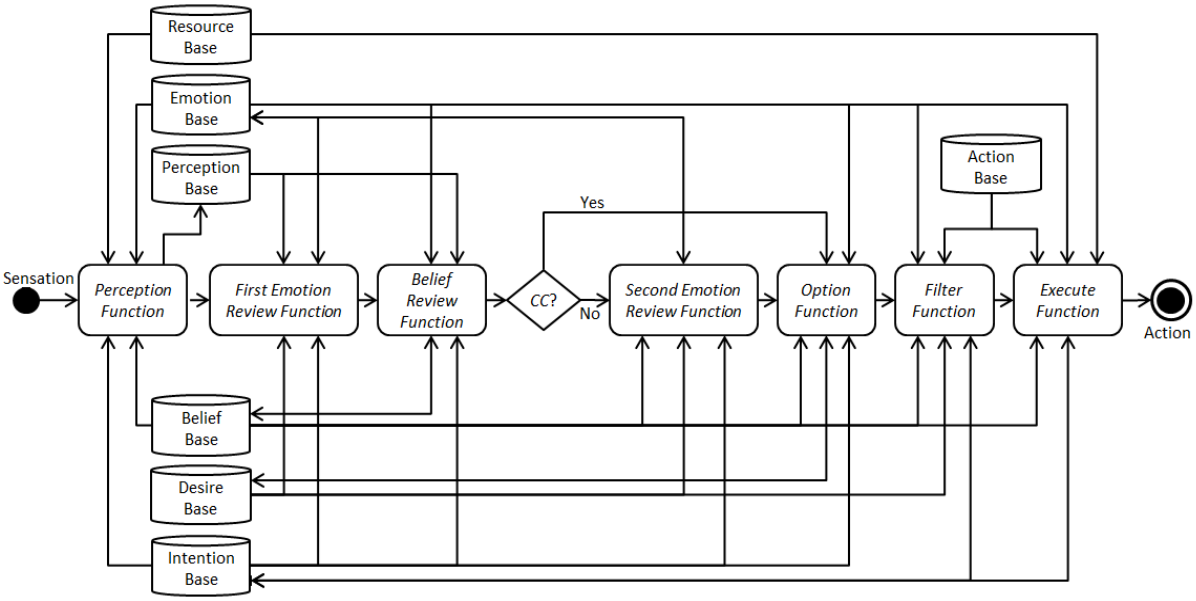


Figura 6: Arquitetura UEBDI internamente

4.1 Módulo Perceptivo

Como no modelo BDI, algumas propostas de agentes emocionais não designam um componente distinto para tratar as percepções, como em (Hernández et al. 2004, Oliveira and Sarmento 2003, Padgham and Taylor 1997). Porém é importante notar que as percepções de eventos externos são uma importante fonte de eliciação emocional e a emoção influencia no conjunto de informações percebidas (Signoretti 2012).

Na arquitetura tradicional BDI se for preciso fazer um pré-processamento sobre as percepções é preciso atribuí-lo a função de Revisão de Crenças (*Brf*), como principal exemplo temos a tarefa de eliminar as sensações que não interessam ao agente no momento. Usar a *Brf* para isto causa problemas quando o número de sensores e a quantidade de dados captados por eles aumentam, tornando a função de Revisão de Crenças complexa e pouco eficiente.

Embora a arquitetura BDI original não defina um módulo a parte para lidar com as percepções, algumas propostas para o desenvolvimento de agentes emocionais como (Jiang

and Vidal 2006, Pereira et al. 2005, Signoretti 2012) determinam explicitamente um módulo que realiza o processamento da percepção promovido principalmente pela influencia do estado emocional no aparato perceptivo do agente, como abordado em (Hernández et al. 2004, Oliveira and Sarmento 2003, Velásquez 1996). Desenvolveremos também um módulo de percepção a parte, visto que esta separação é vantajosa, pois muito do processamento semântico que no modelo original BDI deveria ser feito na *Belief Review Function*, pode agora ser realizado neste (Pereira et al. 2005). Além disto, as intenções (Morgado 2006, Neto 2010) e emoções (Lino 2006, Morgado 2006, Signoretti 2012) desempenham um importante papel na atenção do agente promovida principalmente pela seleção de percepções, e estes processamentos são de natureza perceptiva, o que torna inconveniente coloca-los na função de revisão de crenças.

Além disto, os autores (Jiang and Vidal 2006, Sloman 1999) propõem que o tratamento da percepção do agente deve ser feito de forma segmentada. Os autores (Jiang and Vidal 2006) consideram que percepções de origens diferentes devem ser tratadas por funções diferentes e (Sloman 1999) considera percepções que exigem tratamentos mais complexos deveriam ser tratadas por funções específicas que adotam algoritmos mais complexos.

Com estes argumentos, julgamos vantajoso definir explicitamente uma função de percepção para atingir os seguintes ganhos: (i) diminuir a complexidade da Função de Revisão de Crenças e (ii) criar um projeto de sistema perceptivo explícito e segmentado, descrevendo como será realizada a seleção das sensações, como serão atribuídos os rótulos semânticos e como serão realizadas as transformações semânticas através de uma série de funções, agrupadas pela origem da percepção e pela complexidade de seus algoritmos.

Em nossa arquitetura, a geração de percepções do agente será realizada por uma função composta chamada *Perception Function* (Eq. 2), apresentada na Figura 7. Ele recebe influência da base de emoções (*E*), das crenças (*B*) e das intenções (*I*) do agente. Recebe as sensações obtidas do ambiente (*S*) e retorna as percepções geradas para a base de percepções (*P*). Observe que esta base de percepções foi criada como mero artifício de facilitar o entendimento do modelo, visto que não precisamos dela para fazer nada além da comunicação do conteúdo gerado por esta função *Perception Function* com a *First Emotion Review Function* e a *Belief Review Function*.

$$Pf : S \times B \times E \times I \rightarrow P \qquad \text{(Eq. 2)}$$

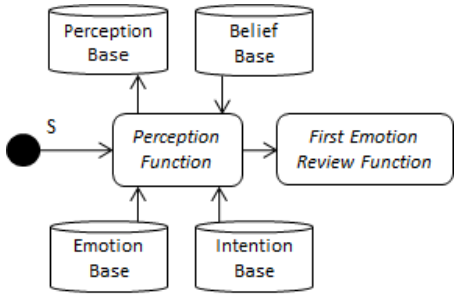


Figura 7: *Perception Function*

Os três principais papéis deste módulo são: (i) selecionar as percepções desejáveis dado o estado mental do agente (por exemplo, um agente selecionará percepções que tenham relação com suas intenções e eliminará as outras); (ii) atribuir informação semântica à percepção (o agente pode rotular as percepções com informação do estado emocional, como por exemplo rotular as memórias de um acidente traumático com as emoções de medo que ocorreram no momento de sua percepção) ; e (iii) transformar as percepções em uma estrutura de informação mais útil ao resto do sistema (como por exemplo, modificar as percepções visuais – imagens – em caracteres via OCR).

Seguindo a proposta de (Sloman 1999) que vê que a percepção pode ser tratada em camadas de acordo com sua complexidade, a consideração de (Jiang and Vidal 2006) sobre o uso de várias funções para tratar percepções de acordo com sua origem e de funções de filtro e atribuição semântica de (Pereira et al. 2005), consideraremos que a *Perception Function* possuirá vários **fluxos** e várias **etapas**. Cada fluxo deve estar associado a um sensor do agente. Se o agente possui os sensores humanos (visão, audição, tato, paladar e olfato) ele possuirá cinco fluxos na função de percepção. Na Figura 8 os círculos numerados indicam os fluxos para cada sensor, podendo haver quantos forem necessários. Esta segmentação lógica é feita para que seja sugestivo ao projetista que tais operações podem ser realizadas potencialmente por módulos independentes e remotos, permitindo processamento paralelo e especializado para cada tipo de informação proveniente de cada sensor.

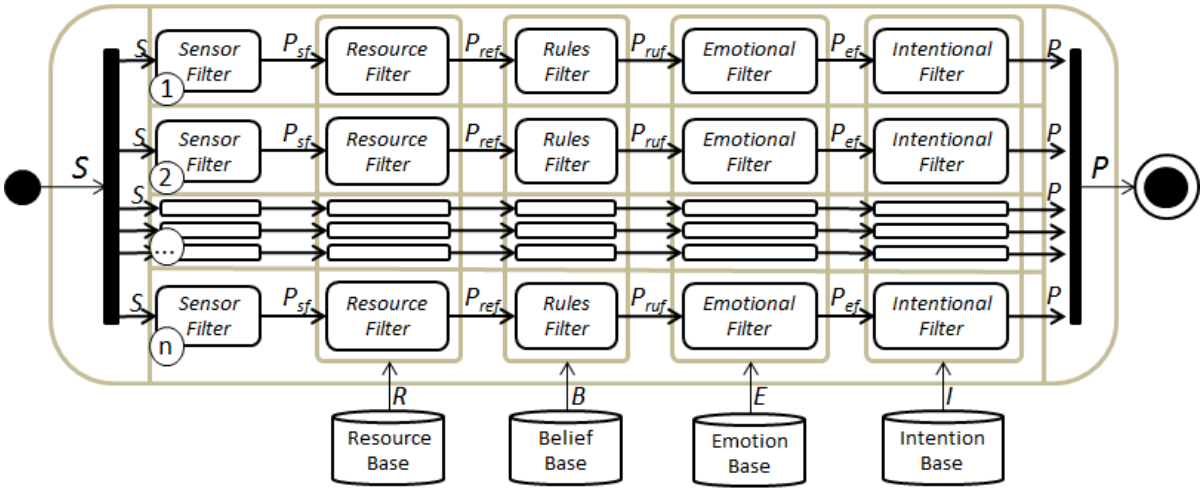


Figura 8: *Perception Function* (internamente)

Cada fluxo é dividido em etapas, cada uma responsável por um tipo de operação e que desempenha um ou mais dos três papéis citados. Estas etapas são: (i) *Sensor Filter*, (ii) *Resource Filter*, (iii) *Rules Filter*, (iv) *Emotional Filter* e (v) *Intentional Filter*. Tal como uma linha de produção, cada função gera a entrada da sua subseqüente e a informação gerada no decorrer do processo por cada função chamaremos de *pré-percepção*, sendo representada por $P_{função}$, onde *função* é a função que deu origem a esta pré-percepção. Por exemplo, P_{sf} representa a pré-percepção depois de passar pela *Sensor Filter*.

A *Sensor Filter*, que representa os sensores em si, desempenha o papel de seleção das sensações, captando somente as sensações possíveis de serem captadas pelo sensor associado, assim representando o sensor físico propriamente dito. Por exemplo, a *Sensor Filter* que representa o sensor da audição capta apenas sons e não imagens e nem odores. Estes devem captados, respectivamente, pelos sensores da visão e do olfato. Além disso, a *Sensor Filter* ainda filtra a percepção de acordo com as limitações do sensor. Por exemplo, o sensor de audição de um ser humano não é capaz de perceber sons em frequências tão baixas quanto o sensor de audição de um cachorro. Sendo assim, qualquer som fora de uma determinada faixa definida nas características do sensor deve ser eliminado.

Esta função é representada matematicamente pela equação Eq. 3, onde S representa as sensações recebidas do ambiente e P_{sf} as sensações já selecionadas de acordo com o tipo de sensor e suas limitações.

$$Sf: S \rightarrow P_{sf} \qquad \text{(Eq. 3)}$$

A *Resource Filter* desempenha o papel de seleção verificando se existem, naquele momento, recursos disponíveis para armazenar/manipular aquela percepção. Por falta de recurso disponível pode ser que uma determinada percepção não seja propagada para as

próximas etapas (por exemplo, o agente irá eliminar percepções caso seu processador esteja sobrecarregado). Sua fórmula é dada pela função Eq. 4, onde P_{sf} é a pré-percepção gerada pela *Sensor Filter*, *R* é a *Resource Base* e P_{ref} é a pré-percepção gerada pela *Resource Filter*. A *Resource Filter* é parametrizada por uma base de informação chamada *Resource Filter*, que contém informação sobre os recursos do agente.

$$Ref: P_{sf}\ x\ R \rightarrow P_{ref} \qquad \text{(Eq. 4)}$$

Já a *Rules Filter* utiliza uma base de regras para realizar seleções, atribuições de semânticas e transformações nas percepções que estão sendo geradas. Seu papel de seleção é equivalente ao desempenhado pela função *Sensing Filter* da arquitetura (Pereira et al. 2005), utilizando as crenças sobre o mundo para saber se uma percepção é necessária. Por exemplo, uma pessoa comum ficará incomodada com os barulhos de uma fábrica, enquanto que um operário que trabalhe a anos no local irá ignorá-los. O papel de atribuição de semântica é similar ao desempenhado pela função *Semantic Association Rules* definida em (Pereira et al. 2005) e usa as crenças para determinar quais qualidades o objeto observado possui. Por exemplo, ao perceber um agente o observador pode reconhecer este agente e lhe atribuir um nome e um conjunto de informações associadas à relação que possui com o agente observado. E, por último, o papel de transformação transforma o dado percebido em outro mais útil ao agente. Por exemplo, transforma um som obtido pelo agente observador em uma sequência de palavras que o representa.

Neste contexto, a base *Resource Base* armazena as informações sobre os recursos do agente. Não é representada a atualização de suas informações, pois considera-se que estas informações sejam fornecidas para a arquitetura abstrata pelo resto do sistema que embarca o agente e não é possível ao projetista manipulá-las.

A representação matemática da *Ruf* é dada pela função Eq. 5, onde P_{ref} são as pré-percepções da *Resource Filter*, *B* é a base de crenças e P_{ruf} o conjunto das pré-percepções geradas após a aplicação da *Rules Filter*.

$$Ruf: P_{ref} x\ B \rightarrow P_{ruf} \qquad \text{(Eq. 5)}$$

Após este processamento, o filtro *Emotional Filter* é executado desempenhando o papel de seleção das percepções de acordo com o estado emocional do agente tal como proposto em (Lino 2006, Pereira et al. 2005, Signoretti et al. 2010) e sugerido por (Gadanho and Hallam 2001, Hernández et al. 2004, Marsella and Gratch 2009, Neto 2010, Oliveira and Sarmento 2003, Pereira et al. 2005, Velásquez 1996). Por exemplo, um agente com a emoção felicidade poderá ignorar percepções agressivas, ou ainda um agente com raiva, poderá

eliminar percepções supérfluas e focar no objeto que lhe causa raiva. Além disto, os algoritmos (i) de filtragem de percepção relacionadas ao estado emocional corrente (Neto and Da Silva 2010 p. 201) – uma percepção pode ser eliminada caso não seja emocionalmente relevante naquele momento, como ignorar um erro de outro agente quando este outro agente é querido – e (ii) de filtragem de percepções de acordo com a intensidade emocional corrente (Signoretti et al. 2010) – uma percepção é eliminada caso a intensidade emocional corrente seja muito baixa ou alta, como por exemplo eliminar a sensação de dor quando se está com raiva alta.

Assim, a fórmula da função *Ef* é dada pela equação Eq. 6, onde P_{ruf} é o conjunto das pré-percepções geradas pela *Rules Function*, *E* o conjunto dos estados emocionais e P_{ef} o conjunto de pré-percepções geradas.

$$Ef: P_{ruf} x\ E \rightarrow P_{ef} \qquad \text{(Eq. 6)}$$

Por último, o módulo perceptivo faz uma última seleção das percepções utilizando as intenções do agente com a função *Intentional Filter*. Utilizando esta função, o agente tem sua percepção orientada às intenções que possui no momento, tal como sugerido por (Jiang and Vidal 2006). Em outras palavras, se o agente possui a intenção de realizar uma tarefa, irá selecionar as percepções que auxiliem a realização de sua intenção. Por exemplo, um agente que representa uma loja virtual que tem a intenção de recomendar livros a um usuário, irá atentar para outros livros em seu ambiente (o estoque), ao invés de sapatos ou eletrodomésticos. De acordo com a equação Eq. 7, a *Intentional Filter* recebe o conjunto P_{ef} e as intenções contidas na base de intenções (*I*) como parâmetros e retorna um conjunto de percepções finais *P*.

$$If: P_{ef} x\ I \rightarrow P \qquad \text{(Eq. 7)}$$

Nossa meta com esta proposta de função de percepção foi permitir a influência da emoção sobre a percepção como proposto por (Lino 2006, Neto and Da Silva 2010, Signoretti et al. 2010), em especial quanto a seleção de percepções, e segmentar o processamento em fluxos e etapas, como proposto pelos autores (Jiang and Vidal 2006, Sloman 1999). Além disto, nos preocupamos em adicionar a influência das crenças, que é algo já implicitamente explorado pela própria arquitetura BDI original (Wooldridge 2009), visto que as percepções eram diretamente processadas pela função de revisão de crenças. E também nos preocupamos em adicionar a influência das intenções, permitindo o agente selecionar percepções mais afins com seu estado intencional, permitindo simular o fenômeno de atenção no agente. (Signoretti 2012, Signoretti et al. 2010) propõem algo similar em sua arquitetura, mas ele considera que

as percepções são selecionadas de acordo com seus *behaviors*, que a nosso ver são uma espécie de ações possíveis do agente que constituem os seus planos em execução, em vez de utilizar as intenções diretamente.

Nota-se que a base de desejos não influencia nossa *Perception Function*. Os dois principais motivos foram: (i) não haver na literatura de agentes emocionais uma proposta que adotasse isto com embasamento teórico; e (ii) julgarmos que a influência das intenções é suficiente para promover a atenção do agente às percepções mais adequadas a estas intenções. Em nossa arquitetura, se for preciso realizar alguma forma de raciocínio utilizando os desejos sobre as percepções, ele deve ser feito na revisão de crenças, após a análise das percepções. Outra possibilidade seria realizar o raciocínio utilizando os desejos indiretamente via emoções, de tal forma que um desejo gere uma emoção e esta emoção venha a modificar a percepção do agente pela *Emotional Filter*.

A ordem das etapas da *Perception Filter* representada na Figura 8 não é aleatória. É preciso considerar a *Sensor Filter* como primeira, visto que ela representa as limitações e ajustes do sensor físico. A *Resource Filter* também está intimamente relacionada com o sensor físico, visto que a falta de recursos poderá interferir "fisicamente" no sensor, eliminando percepções que estejam imprecisas ou de baixa confiança. Já *Rules Filter* e a *Emotion Filter* devem ser executadas anteriormente a *Intention Filter*, pois regras semânticas e emoções tentem a ter uma estrutura mais simples que a estrutura de intenções, desta forma, espera-se que estas primeiras funções executem mais rápido e eliminem informação a ser processada pela *Intention Filter*, que tende a ser mais custosa pela natureza da informação das intenções.

4.2 Módulo Emocional

Nas implementações de agentes emocionais se convencionou definir um módulo para gerenciar o fenômeno emocional (Bates et al. 1994, Breazeal 1998, Hernández et al. 2004, Jiang and Vidal 2006, Neto and Da Silva 2010, Oliveira and Sarmento 2003, Padgham and Taylor 1997, 1997, Pereira et al. 2005, Signoretti et al. 2010, Velásquez 1998a). Iremos seguir esta mesma abordagem e definir um módulo emocional próprio em nossa arquitetura.

4.2.1 Nossa Proposta

Em nosso trabalho seguiremos a proposta de (Jiang and Vidal 2006, Oliveira and Sarmento 2003, Pereira et al. 2005) em adotar um estado emocional explícito a ser definido

pelo desenvolvedor da aplicação. Não consideraremos um estado emocional implícito como (Morgado 2006) sugere por fugir da proposta inicial de BDI que é encontrar um conjunto de entidades que compõe um agente e realizar a modelagem explícita destas entidades. Além disto, não utilizaremos um modelo próprio embutido na arquitetura, similar a (Jiang and Vidal 2006, Oliveira and Sarmento 2003, Pereira et al. 2005). Como nos lembra (Signoretti 2012 p. 82), implementações práticas de agentes emocionais devem definir o modelo emocional de acordo com as exigências da aplicação e não como parte da arquitetura, visto que os autores de teoria da emoção não têm um consenso sobre como representar a emoção (por lógica ou numericamente) e nem qual seria o conjunto de emoções que representaria o fenômeno emocional.

4.2.1.1 Multicamadas

Veja que na literatura se adotou propostas multicamadas devido aos seguintes requisitos: (i) diminuir o custo de raciocínio e permitir resposta reativa, utilizando emoções primárias como gatilho para comportamento reativo (Hernández et al. 2004, Staller and Petta 2000 p. 5); (ii) tentar economizar recursos, apenas tomando decisão sobre um estado emocional secundário quando houver tempo e recursos (Jiang and Vidal 2006); (iii) criar associação entre um estado emocional e entidades (Jiang and Vidal 2006, Velásquez 1998b); (iv) facilitar a tomada de decisão através de redes associativas (emoções secundárias) (Morgado 2006, Velásquez 1998b); (v) facilitar a representação da eliciação das emoções em diferentes cenários (Teasdale 1999); e (vi) permitir a integração de modelos emocionais fisiológicos com os modelos cognitivos (Leventhal and Scherer 1987, 1987, Reekum and Scherer 1997, Scherer 1999, 2005).

Devido a estas características seguiremos a perspectiva multicamada implementando duas funções de revisão do estado emocional, tal como (Jiang and Vidal 2006) realizaram e está demonstrado na Figura 9. A *First Emotion Review Function* eliciará as emoções primárias e a *Second Emotion Review Function* eliciará as emoções secundárias.

Podemos exemplificar a diferença destas duas funções da seguinte forma: um agente presa está dormindo e foi surpreendido por um agente predador. Em um primeiro momento, ele eliciará a emoção medo, agindo defensivamente e correndo. Em um segundo momento, quando ele estiver em uma posição segura em relação ao predador, ele notará que o predador é menor do que ele e não aparenta ser um adversário perigoso, então ele perderá medo e terá confiança. Note que a eliciação da emoção medo é primária e realizada pela *First Emotion Review Function* e é eliciada por processos reativos a percepções do agente. Já o decaimento

da emoção medo exige maior elaboração e cognição em relação as informações do ambiente, do agente predador e das crenças do agente presa, o que exige maior processamento e algoritmos mais ricos, o que indica que este decaimento deveria ser realizado na *Second Emotion Review Function*. Da mesma forma, a emoção confiança é eliciada pela avaliação do contexto do agente em um nível mais complexo, o que indica que também é uma eliciação a ser realizada na *Second Emotion Review Function*,

Não adotaremos uma função de emoção ternária visto que elas estão associadas diretamente a capacidade de abstração e aprendizagem do agente – seguindo a teoria de (Leventhal and Scherer 1987, Reekum and Scherer 1997), e, portanto, não seria vantajoso ter esta função a parte, já que não é nosso objetivo projetar agentes com a capacidade de aprendizagem. Caso seja preciso estender o modelo proposto com estas características, pode-se usar a função secundária para esta tarefa ou ainda criar funções similares a esta dentro da arquitetura.

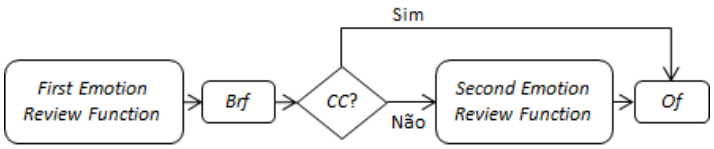


Figura 9: Funções de Revisão Emocional

Explicadas as motivações para a adoção de uma dinâmica emocional multicamada, iremos descrever quais as responsabilidades que estas duas funções devem ter para promover a dinâmica emocional, que são a de eliciação e de atualização emocional.

4.2.1.2 Eliciação e atualização emocional

Eliciadores são como gatilhos que promovem certas alterações no organismo/sistema caso ocorra um evento. Na literatura de agentes são quase sempre implementados como regras de produção baseadas no domínio do problema esperando que certo estado mental ocorra.

Alguns trabalhos ainda tentam identificar quais classes de eliciadores existem. Normalmente, os eliciadores são relativos a eventos no ambiente (Padgham and Taylor 1997, Velásquez 1996), alterações nas metas e no cenário do ambiente, inclusive com previsão de cenários oportunos ou inconvenientes (Oliveira and Sarmento 2003, Padgham and Taylor 1997), ou ainda fatores fisiológicos (Velásquez 1996). Todas as abordagens são simplistas e pouco úteis na hora de desenvolver um agente, comparadas com a literatura base, como (Ortony et al. 1990, Reekum and Scherer 1997), que especificam classificações bem mais densas e funcionais.

Devido a dificuldade de se especificar um conjunto de eliciadores genérico e não dependentes de domínio e a não existência de uma classificação/estrutura genérica na área de agentes, não especificaremos um conjunto de eliciadores em nossa arquitetura, mas sim deixaremos o desenvolvedor adotar um conjunto próprio que seja baseado em uma teoria base ou informal de acordo com o domínio da aplicação.

Já o processo de atualização emocional define como a emoção se comportará com o passar do tempo após sua eliciação. Alguns autores teóricos estão convencidos que o caráter dinâmico da emoção se deve às várias reavaliações de eventos ocorridos no agente (Reekum and Scherer 1997, Scherer 2000) e que a finalização de um fenômeno emocional é promovida por deliberações para modularizar o fenômeno (Teasdale 1999 p. 677), deliberações típicas de emoções secundárias. Entretanto, o que vem sendo usado com mais afinco em agentes é que a emoção, após eliciada, terá sua intensidade decaída com o tempo por certo fator (Bazzan et al. 2002, Mehrabian 1996, Oliveira and Sarmento 2003, Padgham and Taylor 1997, Velásquez 1998b). Assim, o processo de decaimento emocional será tratado dentro da nossa arquitetura como uma atualização emocional.

Devido a esta diversidade de opções e supondo que uma proposta é mais indicada para dado cenário que outra, também não consideraremos uma função de atualização emocional específica, ao contrário do feito por (Oliveira and Sarmento 2003) por uma função a parte e o feito por (Bazzan et al. 2002, Hernández et al. 2004, Neto 2010, Padgham and Taylor 1997, Signoretti 2012) implicitamente em uma função de dinâmica emocional.

Por fim, estes dois principais fenômenos (eliciação e atualização) irão refletir nas tarefas que as funções emocionais realizarão. A seguir demonstra-se como são implementadas internamente as funções de *First Emotion Review Function* e *Second Emotion Review Function* e como elas refletirão as tarefas de dinâmica e atualização emocional.

4.2.1.3 First Emotion Review Function

Esta função atualiza o estado emocional utilizando a percepção do agente (Hernández et al. 2004, Neto 2010, Signoretti 2012), seu estado emocional atual, as suas intenções (Jiang and Vidal 2006, Pereira et al. 2005) e as metas (Morgado and Gaspar 2005, Oliveira and Sarmento 2003).

As percepções influenciarão principalmente a aliciação de emoções primárias. Podemos citar dois exemplos: (i) eventos inesperados gerarão surpresa e (ii) a aparição de pessoas queridas trará felicidade e a aparição de pessoas perigosas, medo.

As metas, em especial, possuem uma restrição. Conforme o proposto na teoria multicamada (tópico 2.4.2), emoções primárias são eliciadas por metas fundamentais e necessidades fisiológicas. Então nós consideraremos que apenas *drives*, como fome, sede e socialização podem ser utilizados como metas por esta função. Impede-se também o acesso desta função primária às crenças, visto que as emoções primárias devem tender a respostas reativas (Hernández et al. 2004, Reekum and Scherer 1997) e as emoções geradas por ela não são associativas (Hernández et al. 2004, Jiang 2007, Velásquez 1998b). Sendo assim, a função *First Emotion Review Function* (Eq. 8) leva em consideração a percepção do agente, o estado emocional atual, as metas em forma de *drives* e as intenções do agente para atualizar o estado emocional.

As intenções participarão da eliciação das emoções primárias principalmente pela reação ao fracasso ou sucesso de intenções. Por exemplo, quando o agente ganha uma competição, ele conclui a intenção que lidava com este objetivo e esta informação pode ser usada pela *First Emotion Review Function* para eliciar a emoção felicidade.

$$Ferf: P \times E \times D \times I \rightarrow E \qquad \text{(Eq. 8)}$$

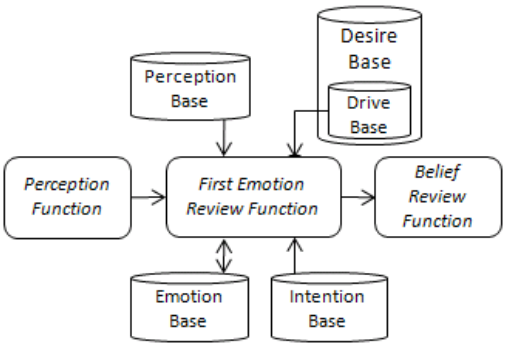


Figura 10: *First Emotion Review Function*

Esta função será composta pela *First Emotional Elicitors* (Eq. 9), que representa um conjunto de eliciadores que devem disparar as emoções em seu estágio primário, e pela *First Emotional Update* (Eq. 10), que representa a atualização do estado emocional. Note que esta abordagem é contraditória aos indícios teóricos propostos por (LeDoux 2002), que afirmam que o controle da dinâmica emocional, diminuindo sua intensidade, se dá por estruturas deliberativas e mais complexas que as que eliciaram (caracterizando uma dinâmica de emoção secundária). Entretanto precisamos definir esta função de atualização primária, pois, se só existisse atualização do estado emocional na *Second Emotion Review Function*, esta atualização poderia nunca ser realizada, já que a execução da *Second Emotion Review Function* ocorre só quando a função *Critical Context* avalia que há recursos disponíveis.

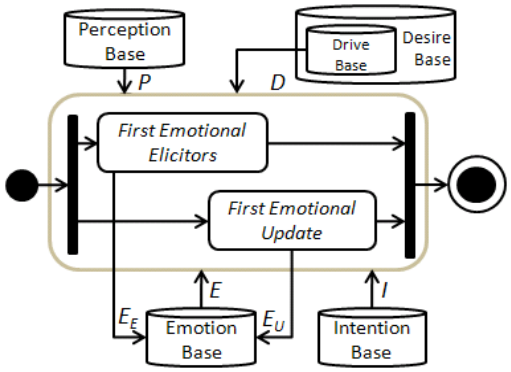


Figura 11: *First Emotion Review Function* internamente

$$FERFFee: P\ x\ E\ x\ D\ x\ I \rightarrow E_E \quad \text{(Eq. 9)}$$

$$FERFFeu: P\ x\ E\ x\ D\ x\ I \rightarrow E_U \quad \text{(Eq. 10)}$$

Como visto pelas barras de bifurcação, a *First Emotional Elicitors* e a *First Emotional Update* podem ser executadas sem uma ordem específica. Embora em todas as arquiteturas consultadas o processo de eliciação ocorra antes do processo de atualização, no nosso caso particular não há nenhuma justificativa, seja teórica ou de implementação, para que haja tal precedência.

4.2.1.4 Critical Context

Como visto na Figura 9 o sistema só executará a *Second Emotion Review Function* se houverem recursos suficientes para isto ou o estado emocional sugerir que o contexto atual é apropriado para realizar as computações de dinâmica emocional secundárias. Por exemplo, um agente de "recomendação de livros" recebeu uma solicitação para sugerir um livro a um novo usuário e só lhe é dado uma janela de tempo pequena para isto. Outro exemplo seria um cenário onde o agente "presa" é surpreendido por um "predador" e deve gerar uma resposta rápida (como por exemplo, fuga).

Então, para lidar com esta questão, define-se uma função chamada *Critical Context* (representada na Figura 9 por um losango e descrita na Eq. 11) que recebe influência da base de emoções e da base de recursos que deve responder verdadeiro ou falso a pergunta: "Estou em uma situação crítica?".

$$CC: E\ x\ R \rightarrow \{true, false\} \quad \text{(Eq. 11)}$$

Veja que as arquiteturas de (Hernández et al. 2004, Jiang and Vidal 2006) adotaram emoções multicamadas e adotaram uma política diferente para evitar a revisão das emoções secundárias. No trabalho (Hernández et al. 2004) sempre que ocorre a eliciação de uma emoção primária não se realiza a eliciação das emoções secundárias. Tal política melhora a

resposta do agente a situações emergenciais, mas não evita que o sistema entre em um cenário onde as emoções secundárias nunca serão eliciadas.

Já (Jiang and Vidal 2006) propõe que existe uma função similar a *Critical Context*, porém lidando somente com o recurso tempo. Ele sempre calcula as emoções secundárias, realizando os passos de geração de objetivos (*Option Function)* e de escolha das intenções (*Filter Function*) propostos em sua arquitetura. Em seguida, caso haja tempo disponível e o estado emocional gerado pela função secundária for diferente do gerado pela função primária, ele executa novamente os passos de *Option* e *Filter Function* com o novo estado emocional. Existem dois problemas com esta proposta: primeiro ele sempre calcula o novo estado emocional e não considera que a eliciação de um estado emocional possui um custo e este custo pode ser demasiado; segundo, não deveria ser preciso realizar duas deliberações só por que foi alterado o estado emocional.

A forma que propomos é similar a de (Jiang and Vidal 2006), mas verificamos antes da deliberação se o agente tem recurso suficiente para achar este novo estado emocional, utilizando a *Critical Context*. Se sim, chama-se a *Second Emotion Review Function*. Caso não possua, continua-se a deliberação.

4.2.1.5 Second Emotion Review Function

Conforme discutido, é nesta função que ocorrerá eliciação das emoções secundárias, sendo mais complexas (Hernández et al. 2004, Jiang and Vidal 2006) e podendo fazer relações a outras entidades do ambiente (Velásquez 1998b).

Ela, diferentemente da anterior, sofrerá influência da base de crenças, permitindo que o agente use informações das crenças para eliciar emoções. Por exemplo, um agente A passa a acreditar que um agente B está muito doente. Isto fará o agente A eliciar a emoção pena.

A função também recebe influência de toda a base de desejos, e não somente dos *drives*. Isto permite que esta função note se as metas estão muito tempo na base ou se elas foram excluídas, gerando emoções relacionadas. Por exemplo, um agente deseja viajar para o Japão. Entretanto esta meta nunca se torna uma intenção, pois ele sempre prioriza outras metas. Isto pode causar a emoção desapontamento ou tristeza. Já a base de intenções influenciará este função similarmente como influencia a *First Emotion Review Function*.

Tal como ilustrado na Figura 12, esta função (Eq. 12) é influenciada por todas as bases da arquitetura BDI mais a base de emoções. Veja que com acesso às crenças, ela conseguirá fazer associações entre emoções e crenças do agente, em especial com outras entidades do ambiente, caracterizando um estilo de emoção secundária proposto por (Velásquez 1998b).

$$Serf: B\ x\ D\ x\ I\ x\ E \rightarrow E \quad \text{(Eq. 12)}$$

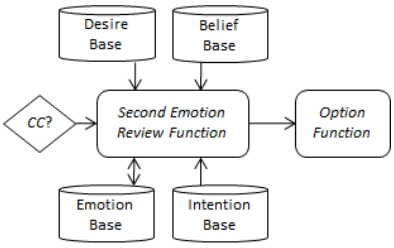


Figura 12: *Second Emotion Review Function*

Esta função de revisão será composta internamente também por duas funções, como ilustrado na Figura 13: uma função representando os eliciadores secundários chamada *Second Emotional Eliciators* (Eq. 13) e a outra é a função responsável pelo processo de atualização das emoções com o tempo chamada *Second Emotional Update* (Eq. 14).

$$SERFSee: B\ x\ D\ x\ I\ x\ E \rightarrow E_E \quad \text{(Eq. 13)}$$

$$SERFSeu: B\ x\ D\ x\ I\ x\ E \rightarrow E_U \quad \text{(Eq. 14)}$$

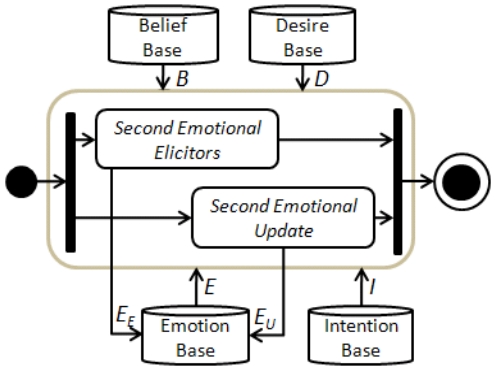


Figura 13: *Second Emotion Review Function* internamente

Igualmente às sub-funções da *First Emotion Review Function*, a *Second Emotional Elicitors* e a *First Emotional Update* não possuem alguma ordem de execução predefinida e podem ser executadas independentemente.

4.3 Módulo de Crença

As crenças são elementos definidos na arquitetura BDI original que representam a informação parcial do ambiente que o agente foi capaz de obter e armazenar, sejam oriundas das percepções ou de inferências sobre o estado mental (Rao and Georgeff. M. 1995 p. 3). Do ponto de vista de definição lógico-formal, o conceito de crenças representa a concretização da lógica doxástica (Steunebrink 2010 p. 59). Este conceito permite definir agentes que representam o conhecimento do mundo, o estado em que este mundo se encontra durante a

última percepção do agente (Georgeff et al. 1998 p. 3) e sobre si mesmo (Signoretti 2012 p. 66), de forma subjetiva. Em outras palavras, o agente obtém/gera esta informação, mas ela não é necessariamente correta em relação a realidade do ambiente.

Existem dois elementos relacionados com a manipulação de crenças: a base de crenças (*Belief Base*), que armazena as crenças adotando alguma forma de indexação e estruturação que permita manipulá-las e resgatá-las no futuro. E a função de revisão de crenças (*Belief Review Function*), que possui basicamente o papel de atualizar a base de crenças, adicionando, removendo e transformando suas instâncias.

A partir da literatura levantada iremos definir a Função de Revisão de Crenças (*Belief Review Function - Brf*). Esta função é influenciada pelas percepções do agente, pelas crenças atuais do agente – como já descrito na arquitetura BDI original –, por suas emoções (Jiang and Vidal 2006, Neto 2010, Oliveira and Sarmento 2003, Pereira et al. 2005) e por suas intenções (Bordini et al. 2007, Bordini and Hübner 2007, Jiang 2007), tal como definido na Eq. 15 na Figura 14. Esta função foi subdividida em três sub-funções tal como descreve a Figura 15 e que serão detalhadas a seguir.

$$Brf : B \; x \; I \; x \; E \; x \; P \rightarrow B \qquad \text{(Eq. 15)}$$

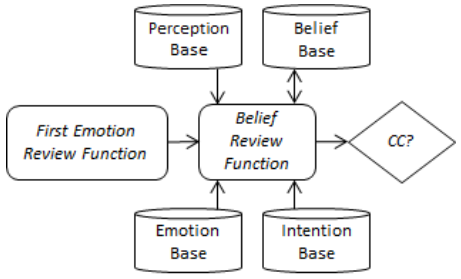


Figura 14: *Belief Review Function*

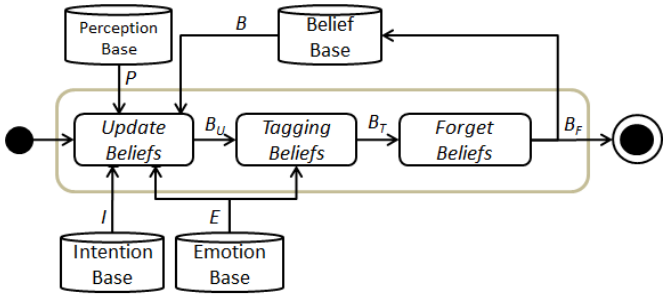


Figura 15: *Belief Review Function* (internamente)

A *Update Beliefs* (Eq. 16) representa a operação esperada da função de revisão de crenças na arquitetura BDI tradicional levando em consideração o estado emocional para realizar a inferência e também as intenções correntes do agente. Sua responsabilidade é a de incluir, excluir e transformar as crenças na *Belief Base*. Nesta função, *B* representa a base de

crenças do agente, *I* a base de intenções, *E* a base de emoções e *P* as percepções geradas pela *Perception Function*. *Bu* é então o conjunto de novas crenças que deve conter a *Belief Base*.

A *Emotional Base* influenciará a atualização de crenças adicionando ou removendo crenças de acordo com o estado emocional. Por exemplo, um agente A que gosta do agente B pode adicionar crenças relacionadas ao B, como, por exemplo, acreditar que B realizará certas tarefas em seu favor. Da mesma forma, o agente A pode adicionar crenças que prevejam informações futuras se ele estiver confiante consigo mesmo.

$$UBf : B\ x\ I\ x\ E\ x\ P \rightarrow B_U \qquad \text{(Eq. 16)}$$

Dentre as novas adições analisadas na literatura, a que demonstrou possuir uma funcionalidade importante e que possui um embasamento válido é a de rotulação de crenças proposto por (Oliveira and Sarmento 2003). Ela permite ao agente "esquecer" as crenças após certo período, e este período é dependente do estado emocional ocorrido no momento da assimilação da crença. Devido a isto, se adicionou a *Tagging Beliefs* (*TBf* - Eq. 18) que utiliza informação proveniente da emoção para rotular a crença. Não é objetivo deste trabalho definir um algoritmo para realizar esta rotulação, cabendo ao projetista escolher um na literatura ou construir um próprio. Os trabalhos de (Oliveira and Sarmento 2003), (Neto 2010) apresentam propostas para esta rotulação.

A *Tagging Beliefs* recebe a B_U, que são as crenças já atualizadas pela *Update Beliefs*, e o estado emocional do agente (*E*) e retorna um conjunto de crenças rotuladas, B_T.

$$TBf: B_U\ x\ E \rightarrow B_T \qquad \text{(Eq. 17)}$$

A outra função necessária para contemplar o processo de esquecimento proposto por (Oliveira and Sarmento 2003) é a *Forgetting Beliefs*, definida na Eq. 18, que é responsável por analisar a informação que rotulou uma crença e determinar se ela já pode ser esquecida (apagada) ou não, removendo-a da base caso positivo. Da mesma forma que a *Ubf*, não definiremos um algoritmo ou alguma heurística para realizar o esquecimento das crenças, cabendo ao desenvolvedor do agente emocional definir estes algoritmos.

Esta função recebe as crenças já rotuladas pela *Tagging Belief* (B_T) e retorna um subconjunto delas, correspondente as crenças ainda não esquecidas (B_F). Por fim a Base de crenças receberá o seu novo conjunto de crenças.

$$FBf: B_T \rightarrow B_F \qquad \text{(Eq. 18)}$$

Não foram adicionadas outras características discutidas pela literatura devido aos seguintes detalhes. A proposta de crença multicamadas de (Hernández et al. 2004) não

apresenta nenhum ganho à arquitetura e aumenta a sua complexidade. Já a proposta de (Oliveira and Sarmento 2003) de congruência de memória por temperamento (*Mood-Memory Congruent*) não é interessante, pois ela agrupa as crenças de acordo com sua rotulação emocional para que o sistema ao acessar uma crença tenha um caminho mais rápido para adquirir às outras. Este conceito seria análogo ao de princípio de proximidade da memória cachê, uma espécie de "princípio de proximidade emocional" das crenças. Contudo, os autores de (Oliveira and Sarmento 2003) não fazem experimentos para comprovar a eficiência desta proposta em nenhum cenário, nem ao menos verificam sua efetividade.

Entretanto, na prática, as crenças são armazenadas e endereçadas por campos chave. E bons algoritmos de estrutura de dados são eficazes, como o mapeamento *hash* que possui complexidade O(1). Mesmo se supormos a hipótese de congruência de memória por temperamento funciona, já temos algoritmos eficientes para acesso a crenças, o que nos levou a abandonar a investigação desta proposta.

Também se deve lembrar que a responsabilidade de processamento das percepções para seleção, transformação e atribuição semântica que era da função de revisão de crenças ficou a cargo da *Perception Function* (tópico 4.1), algo similar ao proposto por (Jiang and Vidal 2006, Pereira et al. 2005, Signoretti 2012).

4.4 Módulo de Desejo

Os desejos são o conjunto de objetivos que o agente "gostaria" de alcançar associados às suas prioridades, vantagens adquiridas quando são alcançados e desvantagens adquiridas quando não o são (Rao and Georgeff. M. 1995 p. 3). De acordo com (Jiang and Vidal 2006 p. 4), os desejos são opções que guiam o agente ao realizar sua deliberação e definem cursos possíveis de atuação. Um desejo representa usualmente um estado futuro do mundo que o agente deseja alcançar (Wooldridge 2009 p. 4). Ou ainda, para a teoria de planejamento, uma meta é uma descrição parcial do mundo que se deve satisfazer dentro de certas restrições (Gratch 1999 p. 101). Neste trabalho, não iremos fazer distinção entre a definição de desejos e metas[8]. Em nossa arquitetura, as duas palavras indicarão a mesma coisa: os estados futuros que o agente almeja alcançar.

Definimos uma função de geração de desejos, que chamaremos de *Option Function*. Esta função é dada na equação Eq. 19 e é influenciada pelas crenças e intenções, tal como já

[8] Ambas as palavras aparecem na literatura como *desires* e *goals*, normalmente representando os desejos do agente.

conhecido na arquitetura BDI original (Wooldridge 2009). Primeiramente, adicionaremos uma base de desejos que poderá persisti-los, como vemos nas arquiteturas concretas (Bordini et al. 2006, 2007). Os autores de agentes emocionais adicionam uma base de desejo para permitir sua persistência e porque é preciso armazenar informações relativas à implementação dos *drives* (veja os tópicos 3.1 e 3.8), que naturalmente persistem de um ciclo de execução para outro.

Adicionaremos também a influência das emoções para podermos realizar sua influência sobre os desejos, como descrito a seguir:

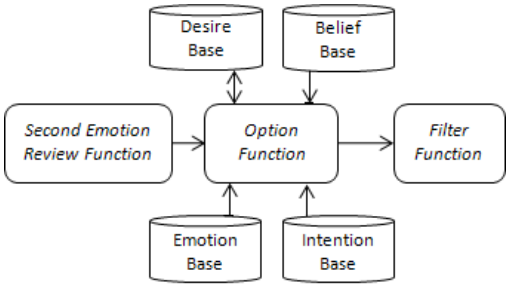


Figura 16: *Option Function*

$$Of : B\,x\,D\,x\,I\,x\,E \rightarrow D \qquad \text{(Eq. 19)}$$

Consideramos que o estado emocional do agente influencia o módulo de desejos de três diferentes maneiras: (i) inclusão/exclusão de desejos (Hernández et al. 2004, Padgham and Taylor 1997); (ii) uso de variáveis homeostáticas/*drives* (Breazeal 1998, Hernández et al. 2004, Velásquez 1996); e (iii) priorização dos desejos (Hernández et al. 2004, Jiang and Vidal 2006, Marsella and Gratch 2009, Padgham and Taylor 1997).

A primeira maneira considera que quando certas emoções ocorrem, o sistema pode criar ou excluir alguns desejos para lidar com estas emoções. Por exemplo, as emoções relacionadas com algum objeto e que sejam negativas (como medo de outro agente ou vergonha de outro agente) podem fazer o agente desejar dado comportamento (como fugir de outro agente ou ajudar outro agente).A segunda considera que a base de desejos é composta por desejos propriamente ditos e por *drives*, seguindo o proposto por (Hernández et al. 2004, Velásquez 1996). Estes se comportam como variáveis homeostáticas do agente possibilitando a representação de necessidades fisiológicas, como fome e sede, e motivações em geral (como desejo de socialização, por exemplo). Conforme o modelo de (Breazeal 1998), os *drives* possuem um valor que acumula por instante de tempo e um limiar superior (e/ou inferior). Ambos, desejos e *drives*, podem estar ativos ou não. Os desejos se tornam ativos quando a *Option Function* os gera devido a um estado mental do agente e o *drive* se torna ativo quando seu valor extrapolar seu limiar. A *Option Function* também pode aumentar ou diminuir o

valor de um *drive* dependendo do estado mental e deve incrementar este valor no decorrer do tempo, para simular sua intensidade, como ocorre com a fome e sede em seres humanos.

Um exemplo prático e simples de *drive* é a fome: a fome possui uma intensidade que pode ser representada por um valor numérico. A fome possui um limiar superior. Quando a sua intensidade ultrapassa o limiar superior, o agente "toma consciência" que tem fome, passa a ter uma série de fenômenos fisiológicos e psicológicos (exemplo a dor na barriga). Para lidar com estes fenômenos ele deve comer. Quando este agente se alimentar, a intensidade do *drive* fome diminuirá e não mais ultrapassará o limiar superior, então o agente não sentirá mais fome.

A terceira diz respeito a priorização dos desejos de acordo com o estado mental do agente, incluindo as emoções, permitindo que ele selecione os desejos mais prioritários durante o processo de seleção ocorrido na *Filter Function*. Por exemplo, se o agente gostar de outro agente, poderá priorizar os desejos que favoreçam este outro agente.

Assim, nossa função de geração de opções será composta por três funções, uma para cada forma de influência emocional discutida, como mostrado na Figura 17. A primeira, *Update Desires* (Eq. 20) trata da inserção, remoção e transformação de desejos, levando em conta as crenças (*B*), os próprios desejos (*D*), as intenções (*I*) e as emoções (*E*). Este último é utilizado para contemplar a capacidade de incluir ou excluir desejos caso o estado emocional exija isto, como discutido anteriormente. A função retorna então um conjunto com o novo conteúdo da *Desire Base*, D_{UDe}. Também se deve lembrar que *drives*, embora seja incomum, podem ser incluídos ou excluídos em tempo de execução, e este procedimento seria de responsabilidade desta função.

$$UDef : B\ x\ D\ x\ I\ x\ E \rightarrow D_{UDe} \qquad \text{(Eq. 20)}$$

A segunda função, a *Update Drives* (Eq. 21) atualiza o valor de intensidade dos *drives* contidos no conjunto D_{UDe} , ativando-os quando extrapolarem seu limiar ou desativando-os. Isto de acordo com as crenças (*B*), desejos (*D*), intenções (*I*) e emoções (*E*) do agente gerando assim o conjunto $D_{UDr.}$

$$UDrf : B\ x\ D_{UDe}\ x\ I\ x\ E \rightarrow D_{UDr} \qquad \text{(Eq. 21)}$$

A terceira, *Prioritize Desires* (Eq. 22) lhes atribui um valor de prioridade aos desejos já tratados pelas equações anteriores (contidos no conjunto D_{DUr}). Este valor será usado nas etapas seguintes, utilizando as crenças, desejos, intenções e emoções gerando assim o conjunto D_P. Os *drives* também são priorizados por esta função. Só que eles possuem prioridade insignificante quando estão inativos (por exemplo, o menor valor possível de

prioridade), ou possuem uma prioridade também dependente do seu valor atual quando estão ativos.

$$PDf : B\ x\ D_{UDr}\ x\ E\ x\ I \rightarrow D_P \qquad \text{(Eq. 22)}$$

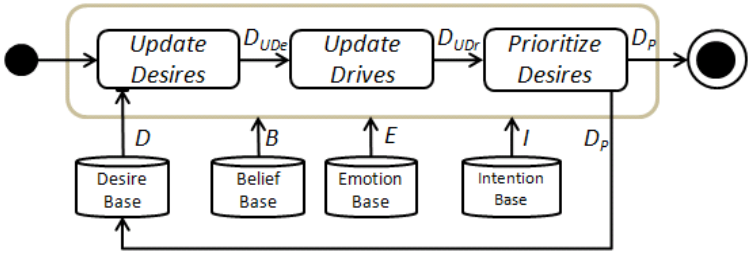


Figura 17. *Option Function* (internamente)

A ordem proposta para a execução destas funções não é aleatória. A função *Update Desires* deve ser executada primeiro porque ela atualiza os desejos de acordo com a nova realidade do agente, permitindo que tanto a *Update Drives* possa atualizar adequadamente os *drives* com suas informações atualizadas, tanto a *Prioritize Desires* possa usar informações sobre os desejos/*drives* atualizados para fazer sua priorização. *Update Drive* deve ser executada anteriormente a *Prioritize Desires*, pois esta precisará saber se um *drive* está ativo ou não, podendo decidir se sua prioridade será insignificante para caso de inativo ou dependente deste valor, caso ativo.

4.5 Módulo de Intenção

As intenções são metas que o agente irá tentar satisfazer em determinado momento utilizando um plano (Rao and Georgeff. M. 1995). Assim, as intenções permitem que o agente saiba quais as metas está deliberando em dado instante, podendo-se verificar se elas estão sendo alcançadas com êxito pelos planos e, caso não estejam ou exista outra meta mais importante, reconsiderar estas intenções correntes. (Wooldridge 2009 p. 67) discute um significado mais genérico propondo que intenções são estados mentais que direcionam atitudes (*pro-attitudes*), isto é, que tendem a conduzir um agente a ação.

Podemos considerar que existem duas entidades que compõe este módulo de intenção: uma função de seleção de intenções, que comumente é chamada de *Filter Function* e uma base de dados que armazena as intenções do agente conjuntamente com um ou mais planos que visam satisfazer esta intenção, que chamaremos de *Intention Base*. A arquitetura abstrata BDI considera o planejamento como processo necessário para o raciocínio e tomada de decisão, mas abstrai como ocorre este processo. Inclusive, considera que o ato de gerar planos

pode literalmente construir um plano do zero ou pode ser a busca de planos em uma base de planos (Wooldridge 2009 p. 74).

Entretanto, pelas questões de (i) compatibilidade com sistemas com capacidade de gerar planos (Meneguzzi et al. 2004), (ii) a ausência de planejamento trazer restrições aos agentes autônomos quanto a sua capacidade de lidar com situações não previstas em tempo de projeto (Meneguzzi et al. 2004) e (iii) a existência de propostas que levam em conta a relação entre emoção e capacidade de planejamento (Gratch 1999, Oliveira and Sarmento 2003), não podemos desconsiderar o planejamento dentro de nossa arquitetura. Especificaremos, portanto uma função geradora de planos, que chamaremos de *Generate Plans*[9]. Esta função exige uma base de ações (*Action Base*) que irão compor o plano. Esta base pertence ao módulo de atuação, que será discutida no tópico 4.6. Porém, como não é foco deste trabalho, não iremos tratar do planejamento em si e nem como a emoção poderia influenciá-lo.

As modificações no modelo BDI encontradas nos modelos emocionais BDI quanto ao módulo de intenção são (i) a própria influência do estado emocional sobre a seleção das intenções (Bazzan et al. 2002, Jiang and Vidal 2006, Neto 2010, Pereira et al. 2005); (ii) a configuração do algoritmo de planejamento (Oliveira and Sarmento 2003); (iii) a seleção do estilo de algoritmo de acordo com o estado emocional (Oliveira and Sarmento 2003); e (iv) a rotulação dos planos com marcador somático para selecionar planos cujo marcação somática é mais afim com o estado emocional atual (Neto 2010).

Nós abstrairemos os detalhes teóricos dos processos de influência propostos, devido ao fato deles serem dependentes de domínio e alguns – como os de (Oliveira and Sarmento 2003) – não possuírem experimentos comprovando sua eficiência. Consideraremos ainda que a emoção influencia principalmente na própria atualização de intenções e no processo de geração e seleção de planos.

Desta forma, definiremos nossa *Filter Function* (Eq. 23) apresentada na Figura 18 da seguinte forma: ela sofre influência da *Intention Base*, da *Desire Base* e da *Belief Base*, tal como no BDI original e postulamos a influência da *Action Base* porque supomos que nossa arquitetura abstrata poderá ser munida de um sistema de planejamento que potencialmente usará esta base. Por último, a *Emotion Base* também a influenciará, seguindo o proposto por (Jiang and Vidal 2006, Neto 2010, Oliveira and Sarmento 2003, Pereira et al. 2005).

$$Ff : B \, x \, D \, x \, I \, x \, E \, x \, A \rightarrow I \qquad \text{(Eq. 23)}$$

[9] Entretanto, faremos como a arquitetura BDI e tentaremos explicar como se realiza este processo e como a emoção pode promove-lo e influenciá-lo, pois foge do escopo deste trabalho.

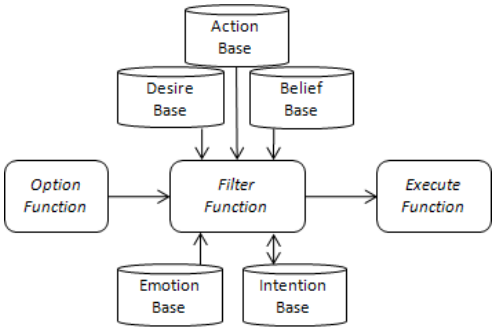


Figura 18: *Filter Function*

Seu funcionamento interno é exibido na Figura 19. A primeira operação a ser realizada, *Update Intention* (Eq. 24), utiliza as bases de desejos, intenções, emoções e crenças para selecionar o conjunto de intenções atuais do agente. Veja que é naturalmente necessário a base de desejos e intenções para que se realize o processo de seleção e reconsideração de intenções. Já a de crenças é usada para verificar se uma intenção está condizente com as crenças do agente sobre o mundo e sobre si mesmo, evitando assim, por exemplo, paradoxos (uma intenção é alcançar um estado do mundo que o agente crer ser impossível), que foram discutidos por (Jiang and Vidal 2006). Consideramos também que a base de emoções também será exigida para que o agente possa reconsiderar a intenção de acordo com o estado emocional, de tal forma que, possa reordená-las ou até mesmo removê-las caso o estado emocional não esteja favorável a esta intenção. Podemos citar como exemplo: um agente A passa a não gostar do agente B. Assim ele passa a eliminar as intenções que tinha de ajudar B. Os autores (Jiang and Vidal 2006, Oliveira and Sarmento 2003, Pereira et al. 2005, Steunebrink 2010) também consideram esta influência.

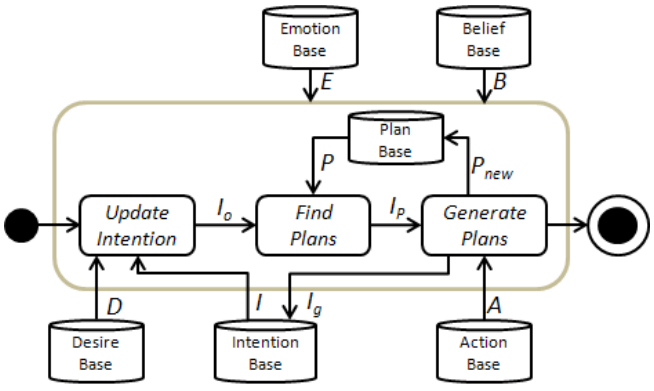


Figura 19: *Filter Function* internamente

$$FUIf : B\ x\ D\ x\ I\ x\ E \rightarrow I_o \qquad \text{(Eq. 24)}$$

A próxima função chamada *Find Plans* (Eq. 25) tem o objetivo de associar planos às intenções do conjunto I_o buscando-os em uma base de planos chamada *Plan Base* os planos definidos em tempo de projeto ou gerados pela função *Generate Plans*. O resultado desta

função é um conjunto de intenções associadas a um conjunto de planos que podem ser utilizados para alcançar estas intenções. A este conjunto damos o nome de I_p. Para realizar esta tarefa, a *Find Plans* precisa da *Plan Base* (*P*) para saber quais planos pode utilizar e das bases de crenças e emoções para saber se o contexto de execução do plano condiz com o estado emocional do agente e suas crenças.

Como visto, a emoção influencia na *Find Plans*, de tal forma que se selecione planos ou não planos para dada meta de uma intenção. Um exemplo é ignorar planos custosos e demorados quando se está ansioso e considerar planos custosos quando se está relaxado. Podemos dizer também que a hipótese do marcador somático de Damásio (tópico 2.4.3) pode ser implementada nesta função, visto que se pode usar os resultados passados destes planos e estados emocionais gerados por estes planos para sua seleção posterior, como feito por (Neto 2010).

$$FFPf : B\ x\ E\ x\ P\ x\ I_o \rightarrow I_P \qquad \text{(Eq. 25)}$$

Caso exista alguma intenção ainda sem plano após a execução de *Find Plans*, a *Generate Plans* (Eq. 26) tentará gerar um ou mais planos para estas intenções. Para isto ela utiliza a base de crenças, para saber qual o contexto atual e para gerar planos com pré-requisitos de acordo com este contexto. A base de emoções, além de servir para gerar planos com pré-requisitos adequados com o estado emocional, influenciará o planejamento tal como (Oliveira and Sarmento 2003) sugere. Por exemplo, quando o agente está no estado emocional "com medo", ele pode forçar o seu algoritmo de planejamento a gerar planos curtos para tentar lidar rapidamente com a fonte do medo e limitar o tempo que o algoritmo terá para operar. Por último, a base de ações é exigida para que o algoritmo de planejamento tenha as informações das ações que podem constituir para gerar os planos.

No final, a *Generate Plans* irá tentar associar novos planos para cada intenção que continuou sem planos contida na I_P, obtendo a nova coleção I_G. Esta função também adicionará todos os novos planos gerados (P_{new}) à base de planos, permitindo que eles sejam reutilizados em ciclos posteriores.

$$FGPf : B\ x\ E\ x\ A\ x\ I_P \rightarrow I_G, P_{new} \qquad \text{(Eq. 26)}$$

Quanto à ordem adotada para as funções, observa-se que existem duas tarefas básicas representadas: (i) a seleção e reconsideração de intenções e (ii) a geração/associação de planos. Adotou-se esta ordem (primeiro seleciona/reconsidera intenções), pois como selecionamos primeiro as intenções antes de associar/criar planos, a segunda tarefa se torna mais rápida, visto que teremos menos metas exigindo planos para satisfazê-las. Esta escolha

também é a mais adotada pelas abordagens BDI e Emocionais BDI (Bordini et al. 2007, Jiang and Vidal 2006, Neto 2010), o que nos indica sua efetividade. Entretanto existem trabalhos que propõe a ordem inversa, onde primeiro se seleciona/cria planos para metas e só depois realiza a seleção/reconsideração de intenções (Meneguzzi et al. 2004). Esta ordem inversa é desejável quando é preciso gerar planos para avaliar a necessidade de manter metas na base ou utilizá-los para selecionar as futuras intenções, com o custo de ter de planejar/buscar metas que ainda não se sabe se tornarão intenções.

4.6 Módulo de Atuação

Considera-se neste trabalho a atuação de um agente como sendo a forma que ele alterará seu ambiente e se comunica com outros agentes. A arquitetura BDI de (Rao and Georgeff. M. 1995), não faz nenhuma proposta ao processo de atuação do agente, já a definição de (Wooldridge 2009) define uma função que executa ações de plano, porém ela não trata da operacionalização dos planos. Além disto, o processo de atuação é notoriamente dependente de domínio com atuadores que podem ser físicos (uma garra de um robô) ou virtuais (um módulo de comunicação).

Refletindo sobre isto, o Jason não considera em detalhes a atuação em sua arquitetura. Da mesma forma, as arquiteturas emocionais BDI não discutem sobre a atuação do agente, assim como sua influência com o resto da arquitetura e com o módulo emocional, salvo uma tímida discussão em (Pereira et al. 2005) relacionando o estado emocional e a informação dos recursos do agente com a calibração das ações.

Entretanto, a atuação tem importante papel em arquiteturas emocionais de origem fisiológica – tal como as propostas em (Breazeal 1998, Velásquez 1996, 1998b p. 199) – como discutido nos trabalhos relacionados. De acordo com as propostas de agentes emocionais encontradas na literatura, as influências que o módulo de atuação pode sofrer são: (i) uma ação é potencialmente parametrizada pelo estado emocional e pelas motivações do agente, de acordo com (Breazeal 1998, Velásquez 1996, 1998b); e (ii) uma ação é parametrizada pelos recursos do agente, tal como (Pereira et al. 2005) sugere.

Em virtude disto, iremos definir um módulo explícito de atuação, contendo uma função de execução de ações, chamada de *Execute Function*, e uma base de ações chamada *Action Base*.

A *Action Base* possuirá informações sobre as ações para o planejamento, como (i) as pré e pós-condições das ações; (ii) informação se ela está sendo executada no momento; (iii)

informação se está indisponível por algum motivo; e (iv) equivalência entre ações, i.e., se uma ação diferente possui os mesmos pré-requisitos e gera os mesmos pós-requisitos, permitindo sua troca. As alterações desta base não são realizadas diretamente pela arquitetura, mas sim pelo processo de atuação em si, que é dependente de domínio.

Já a *Execute Function* será definida tal como Eq. 27 e executada após a *Filter Function*, como visto na Figura 20**.** Esta função é influenciada pela base de intenções, crenças, emoções e recursos. Consideramos que a base de crenças influencia esta função, pois os planos podem utilizar informações das crenças do agente para realizar cálculos e parametrizar ações. Também consideraremos que a base de ações influencia este módulo para que ele possa observar o estado de uma ação (sua disponibilidade, por exemplo).

$$Ef : A \; x \; B \; x \; E \; x \; I \; x \; R \rightarrow Act \qquad \text{(Eq. 27)}$$

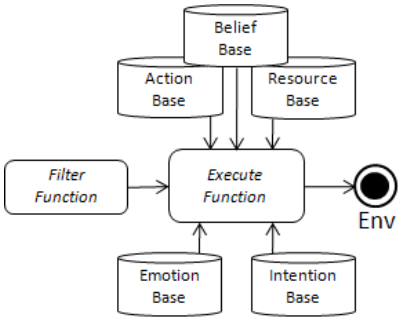


Figura 20: *Execute Function*

Consideraremos que nossa arquitetura tentará realizar uma intenção qualquer por um ciclo de execução, adotando algum critério de seleção desta intenção e tentará obter seu objetivo executando todos os planos associados, um de cada vez até obter êxito ou acabarem os planos possíveis. Nossa motivação para adotar esta política é a sua simplicidade, facilitando o estudo do problema e a implementação da proposta.

Visando esta política, a *Execute Function* será composta por duas outras funções, como visto na Figura 21. A primeira é a *Select Intention* (Eq. 28) que, dada a coleção de intenções, seleciona qual será a intenção executada no respectivo ciclo, levando em conta a base de emoções para saber qual intenção é mais adequada para um dado estado emocional, tal como (Pereira et al. 2005) sugere. É importante existir esta influência para que se possa representar situações similares a de comportamentos influenciados por emoções como visto nas propostas de (Breazeal 1998, Velásquez 1996).

Um exemplo de como a emoção influencia na seleção da intenção é o seguinte: um agente tem a intenção de fazer um relatório antes de limpar sua mesa de trabalho. Outro

agente observa a bagunça em sua mesa. Isto faz o agente ter vergonha e ele passa a preferir realizar a intenção de limpar a mesa.

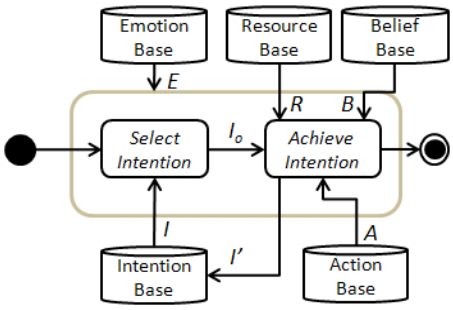


Figura 21: *Execute Function* Internamente

$$ESIf : I\ x\ E \rightarrow I_o \qquad \text{(Eq. 28)}$$

A segunda função, a *Achieve Intention* (Eq. 29) deve receber a intenção selecionada (I_o) e executar os planos associados até satisfazer a intenção. Os planos são executados com uma ordem de prioridade que pode ser obtida também pela informação do estado emocional e ações. Caso um plano mais prioritário falhe, a função irá tentar o próximo mais prioritário, até que satisfaça a intenção ou todos os planos falhem. Veja que neste processo pode ser utilizada uma série de algoritmos para priorização de planos propostos em (Neto 2010, Oliveira and Sarmento 2003, Pimentel and Cravo 2009) usando as métricas do marcador somático e similaridade com o estado emocional, respectivamente (veja tópicos 2.4.3 e 3.3.3 para mais detalhes). Inclusive, devemos levar em conta que o próprio plano pode querer informações sobre as emoções no decorrer de sua execução, e também para isto é necessário que a *Execute Function* tenha acesso a base de emoções.

$$EAIf : I_o\ x\ B\ x\ E\ x\ R\ x\ A \rightarrow I' \qquad \text{(Eq. 29)}$$

Além disto, dentro do processo de execução de um plano, a *Achieve Intention* deverá ter a capacidade de substituir ações de um plano por ações equivalentes de acordo com o estado emocional, o que exige acesso à base de ações. Isto é útil principalmente para que se possa implementar a expressão das emoções, tal como (Breazeal 1998, Velásquez 1996, 1998b) propõem.

Por último ela gera I' que será vazio caso o sistema tenha conseguido atingir a intenção I_o (que deve ser removida da base de intenções) ou conterá I_o caso não tenha atingido a intenção associada.

5 Mapeando UEBDI no Jason

Com nossa arquitetura UEBDI definida, nosso próximo passo será propor uma implementação para a mesma. Para isto, em vez de começarmos uma implementação do zero, vamos estender a parte de arquitetura do Jason. Será visto que algumas características possuem representantes diretos no Jason, outras precisam de algum ajuste e outras não podem ser adotadas devido a incompatibilidade da nossa arquitetura com a proposta no Jason. Chamaremos esta tarefa de mapeamento.

5.1 Fluxo de execução UEBDI

A título de revisão, iremos descrever na Tabela 3 os principais módulos, bases e funções da arquitetura UEBDI. A ordem das funções na tabela reflete a ordem de execução das mesmas.

Arquitetura UEBDI	
Nome	**Descrição**
Resource Base	Base descrevendo os recursos do agente
Action Base	Base contendo dados referentes as ações do agente
Módulo de Percepção	
Perception Base	Base contendo as percepções do agente em um ciclo de execução
Perception Function	Tratamento das percepções antes de se tornarem crenças
Sensor Filter	Filtra as sensações de acordo com as limitações do sensor
Resource Filter	Filtra as percepções de acordo com os recursos disponíveis
Identity Rules Filter	Filtra as percepções através de regras de produção
Emotional Filter	Filtra as percepções de acordo com o estado emocional
Intentional Filter	Filtra as percepções de acordo com as intenções
Módulo de Percepção	
Belief Base	Base de crenças
Belief Review Function	Revisa as crenças
Update Beliefs	Atualiza a base de crenças, inserindo ou removendo crenças e mantém a integridade da base

Tag Beliefs	Rotula a crença de acordo com o estado emocional e o momento da criação/modificação
Forget Beliefs	Verifica, de acordo com o rótulo, se a crença pode ser esquecida (deletada)
Módulo de Desejos	
Desire Base	Base de desejos
Option Function	Determina os desejos a serem considerados neste ciclo de execução
Update Desires	Atualiza os desejos, inserindo ou removendo-os da base de desejos
Update Drives	Realiza o comportamento dos *drives*, ativando-os ou não, de acordo com seu valor. Além de atualizar este valor com o incremento natural com o tempo e com os eventos ocorridos
Prioritize Desires	Prioriza os desejos de acordo com sua importância em relação ao estado mental
Módulo de Intenções	
Intention Base	Base de Intenções
Filter Function	Seleciona quais intenções devem persistir no novo ciclo e quais desejos devem se tornar intenções
Update Intention	Revisa as intenções e seleciona os desejos que se tornarão intenções
Find Plans	Encontra planos para as intenções sem planos
Generate Plan	Gera planos para intenções sem planos
Módulo de Execução	
Execute Function	Seleciona uma intenção e executa os planos associados
Select Intention	Seleciona a intenção
Achieve Intention	Executa os planos associados em ordem de prioridade
Módulo de Emoções	
Emotion Base	Base de Emoções
First Emotion Review Function	Revisa primariamente o estado emocional
Critical Context	Verifica se o sistema está em um contexto crítico. Caso verdadeiro, não executa a *Second Emotion Review Function*
Second Emotion Review Function	Revisa secundariamente o estado emocional

Tabela 3: Resumo das bases e funções da arquitetura UEBDI

5.2 Descrição do mapeamento

Vamos agora proceder com o mapeamento. Para isto iremos fazer a correspondência de uma função UEBDI em outra do Jason. Devemos considerar que a função UEBDI deve ter acesso às bases planejadas e a ordem de execução das funções deve ser mantida. Se os critérios forem atingidos dizemos que o mapeamento de uma função foi **completo**. Caso um ou outro não seja atingido, dizemos que o mapeamento é **incompleto**. Além disto, não

planejamos alterar a estrutura do Jason e nos limitamos a alterar somente seus pontos flexíveis (ou *fronzen-spots*).

Iremos realizar esta tarefa por módulos. Isto é, o mapeamento será realizado a cada módulo da arquitetura. Com exceção do mapeamento de bases de recursos, ações, emoções e intenções, que será realizado previamente. No final elaboraremos uma tabela resumindo o resultado final do mapeamento. Pode-se obter informações mais detalhadas sobre a arquitetura do Jason no tópico 2.3.

5.2.1 Bases de dados atípicas (*Resource*, *Action* e *Emotion Base*)

Existem três bases de dados na arquitetura UEBDI não encontradas BDI: (i) a *Action Base* é responsável por armazenar informações sobre os atuadores do agente, assim como seu estado em dado instante; (ii) a *Resource Base* armazena informações referentes a recursos necessários para o funcionamento do agente, como por exemplo, o nível de bateria, quantidade de projéteis, quantidade de material para construir seu objeto de trabalho, entre outros; e (iii) a *Emotion Base* utilizada para armazenas as emoções do agente. Tanto a *Action Base* quanto a *Resource Base* podem ser representadas como crenças sem muitos problemas. Já a *Emotion Base* será mapeada na base de crenças adotando-se algumas convenções (explicadas no tópico 5.2.8 abaixo). Logo, todas estas bases serão mapeadas na base de crenças do Jason.

5.2.2 Base de Intenções

A base de intenções não tem acesso trivial no Jason, pois as intenções são implementadas no Jason como um conjunto de pilhas de execução destas intenções e não existe explicitamente um objeto "Base de Intenções" como existe a *Belief Base*, com métodos próprios para consulta.

Entretanto, existe uma ação interna chamada ".intend" que permite aos planos consultar a base de intenções, verificando se existe uma intenção que seja iniciada por determinada meta.

Para facilitar nosso mapeamento, foi construída uma base de intenções explícita que permite consultar a base de intenções com sentenças (metas) que iniciaram as intenções. Para realizar isto, o comportamento da ação interna ".intend" foi encapsulado dentro da nova base de intenções. Com isto, conseguimos acessar, via Java, as intenções de um agente de forma mais simples, facilitando nosso mapeamento.

Lembramos que as pilhas de execução das intenções no Jason são acessíveis de qualquer função da classe Agent. Por isto, nossa base de intenções explícita também será acessível de qualquer função da classe Agent.

5.2.3 Mapeamento do módulo de percepções

Como visto, o módulo de percepções do UEBDI é composto pela *Perception Function* e pela *Perception Base*. A *Perception Base* foi introduzida como artifício para representar a ligação entre a etapa de percepção e a de revisão de crenças. Esta base não é essencial visto que os dados de percepção não precisam ser persistidos entre dois ciclos de execução. Então não é preciso mapeá-la no Jason.

Já a *Perception Function* é dividida em vários fluxos, um para cada sensor do agente. No Jason, percepções do ambiente são filtradas e inseridas no fluxo padrão da arquitetura BDI, o que torna quase direto o mapeamento da *Perception Function* do UEBDI na função *Perceive* Jason. Já as mensagens do Jason são interpretadas por funções separadas e inseridas diretamente como metas no agente na base de eventos. Isto faz com que a mensagem no Jason não siga o fluxo esperado de uma percepção da UEBDI, que é de inserção na base de percepções para depois passar pelas *Perception Function*, e por fim a *Belief Review Function*. Em tese, é possível tratar estas mensagens que caem na *Event Base*, entretanto fazê-lo irá inverter a ordem de execução original da UEBDI e não será possível acessar a base de percepções pela *Select Event*. Por isto, não iremos considerar as mensagens e consideraremos o mapeamento das mensagens **impossível**.

De qualquer forma, ainda podemos tratar as mensagens se a considerarmos uma meta a ser tratada na *Select Event* do Jason (ou a *Update Desires* da UEBDI). E quanto as percepções normais, iremos mapeá-las abaixo.

Na Figura 22 representamos o mapeamento. A *Sensor Filter* da arquitetura faz a seleção de sensações pertinentes àquele fluxo de percepção e representa o sensor propriamente dito. Sendo assim, esta função deve ser mapeada a função *Perceive* em Jason que possui objetivo similar.

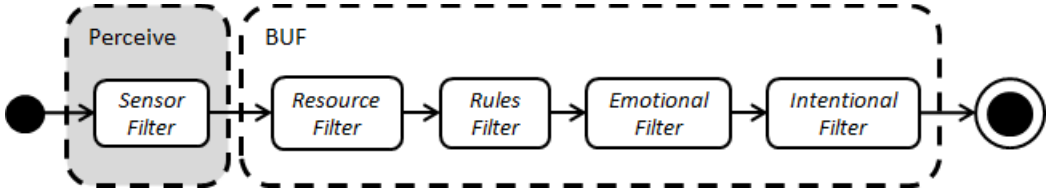


Figura 22: Mapeamento *Perception Function*

Já as funções *Resource Filter*, *Rules Filter*, *Emotional Filter* e *Intentional Filter* da arquitetura podem ser mapeadas para a função *BUF* do Jason. Embora estas funções adicionem comportamento à função *BUF*, (i) elas também possuem o objetivo de tratar as percepções antes de virarem crenças; e (ii) utilizam bases de dados que foram mapeadas como sub-bases da base de crenças as quais a função *BUF* tem acesso, além a base de intenções. A *Resource Filter* utiliza a *Resource Base* enquanto a *Emotional Filter* utiliza a *Emotion Base*, ambas mapeadas como sub-bases de *Belief Base* em Jason.

Com isto, podemos considerar o mapeamento **completo**, pois a *Sensor Filter* foi mapeada na função *Perceive* do Jason e as outras na função *BUF*, e destas funções temos acesso a todas as bases necessárias.

5.2.4 Mapeamento do módulo de crenças

O módulo de crenças da UEBDI é composto pela *Belief Review Function* e *Belief Base*. A *Belief Base* pode naturalmente ser mapeada na base de crenças do Jason diretamente.

A *Belief Review Function* da nossa arquitetura visa realizar a inclusão, exclusão e transformação/revisão da base de crenças (feito pela *Update Beliefs*) e dar suporte ao sistema de rotulação e esquecimento (chamadas na nossa arquitetura de *Tag Beliefs* e *Forget Beliefs*). No Jason, as tarefas de inclusão e exclusão pertencem à função *BUF* e aos planos do agente. Já a tarefa de transformação/revisão de crenças é realizada pela *BRF*. Porém existe uma diferença, a *BRF* realiza somente a revisão de crenças individuais e não de um conjunto de crenças à medida que cada crença é incluída ou excluída da base de dados.

Dadas similaridades funcionais, podemos mapear a *Update Beliefs* na *BUF* e na *BRF* do Jason, respeitando a ordem de execução original. Também é possível mapeá-la nos planos, pois estes podem atualizar a base de crenças. Entretanto, neste caso a ordem de execução é diferente da UEBDI, pois os planos irão executar a atualização na *Execute Function* do Jason. Além disto, nos planos e na *BUF*, pode-se acessar a *Belief Base*, a *Emotion Base* (contida na *Belief Base*) e a *Intention Base*, permitindo acessar todas as bases necessárias por esta função. Devido a isto, consideraremos que este mapeamento é **completo** se implementarmos agentes estendendo a *BUF*/*BRF* do Jason. Caso usemos os planos para atualizar as crenças, o mapeamento será **incompleto**.

A *Tag Beliefs* pode ser mapeada na *BRF* do Jason, pois as duas funções operam durante a inclusão de crenças e a *BRF* já pressupões acesso à base de crenças, o que é também uma necessidade da *Tag Beliefs* que precisa acessar o estado emocional do agente. A informação de rotulação feita para *Tag Beliefs* pode ser inserida como outras crenças na

própria base de crenças ou como anotações sobre a crença rotulada, um recurso existente no Jason. Neste caso, o mapeamento é **completo**.

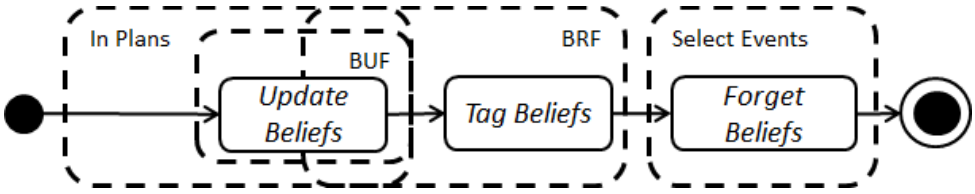


Figura 23: Mapeamento *Belief Review Function*

Por último, a *Forget Beliefs* não pode ser mapeada para a função *BRF* do Jason, pois ela deve ser executada a cada ciclo de execução, forçando com que crenças não estejam mais disponíveis logo após o seu tempo de vida expirar. Tampouco pode ser mapeada para a função *BUF*, pois podem existir crenças oriundas das percepções que não foram rotuladas pela *Tag Beliefs*.

O local mais indicado para incluir a funcionalidade de *Forget Beliefs* é no início da função *Select Event*, pois ela é executada logo em seguida a *BUF*. Mesmo *Select Event* não tendo a função de tratamento de crenças, o início desta função é o local ideal para o procedimento de esquecimento de crenças, que deve ser executado logo no final de todo o processo de revisão de crenças. A função só precisa saber qual a crença deve esquecer, assim como as informações que a rotulam, ambas contidas na base de crenças e acessíveis pela *Select Event*. Logo, o mapeamento da *Forget Beliefs* na *Select Event* é **completo**.

5.2.5 Mapeamento do módulo de desejos

Metas no Jason são eventos especiais que são inseridos pelos planos na base de eventos para notificar ao sistema que ele deseja alcançar dada sentença. Posteriormente este evento é selecionado pela função *Select Event*.

Na nossa arquitetura existe o módulo de desejos que é composto pela *Desire Base* e pela *Option Function*. A base pode ser mapeada como subconjunto da base de eventos do Jason contendo os eventos relativos a metas. Como já discutido no tópico 2.3, o Jason possui seu sistema de metas limitado, não permitindo implementar metas persistentes diretamente sem o uso de artifícios. Entretanto iremos considerar que através dos padrões de projeto (Hübner et al. 2006) é possível implementar metas persistentes em Jason. Os *drives*, que até então estávamos considerando como parte das metas do agente, são mapeados para a base de crenças, como discutiremos a seguir.

A Figura 24 demonstra como poderá ocorrer o mapeamento. A função *Update Desires* realiza a tarefa de inserção e a remoção de metas é influenciada pelas crenças, emoções,

desejos e intenções. Podemos considerar que esta função pode ser mapeada nos planos do agente, onde as emoções e crenças são acessíveis pela *Belief Base* do Jason e desejos e intenções são acessíveis pelas ações internas ".desire(x)" e ".intend(x)" do Jason. Veja que este mapeamento é **incompleto**, pois – apesar de conseguirmos acessar as bases de crenças emoções e intenções pelos planos – esta função será executada na função *Execute Intention* do Jason junto com os planos, e não logo depois da revisão de crenças e revisão e emoções pela *Second Review Emotion Function* (discutida em 5.2.8). Isto inverte a ordem de execução original da UEBDI.

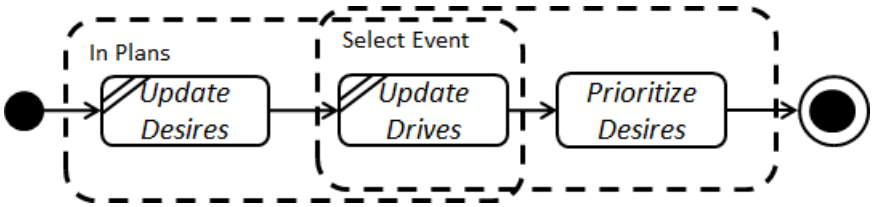


Figura 24: Mapeamento *Option Function*

Posteriormente temos a função *Update Drives*. Veja que ela desempenha três papéis: dado um conjunto de *drives*, (i) atualizar o valor dos *drives* e (ii) verificar quais *drives* extrapolaram seus limites de ativação e os ativar, assim como verificar se o seu valor voltou a estar contido entre seus limites de ativação e o desativar; e (iii) atualizar o valor dos *drives* caso ocorra certo evento (como por exemplo, diminuir a cede caso o agente tenha ingerido água).

Para implementar a *Update Drives* no Jason, vamos definir a descrição dos *drives* como crenças na forma *drive*(*nome*, *valor*, *limiteSup*, *limiteInf*, *incremento*), onde *nome* é o nome do *drive* que será usado para identificá-lo no sistema, *limiteSup* é o valor limiar superior do *drive* (quando o *valor* do *drive* for maior que este, o *drive* é considerado ativo superiormente), *limiteInf* é o limiar inferior do *drive* (quando o *valor* do *drive* for menor que este, ele é considerado ativo inferiormente) e *incremento* é um valor usado para incrementar o *valor* do *drive* a cada ciclo de raciocínio.

Além disto, para atualizar o *valor* dos *drives* podemos adotar duas abordagens. A primeira é mapear a atualização para a função *Select Event* do Jason, monitorando o estado mental do agente e alterando as crenças relativas ao *drive*. A base de crenças é acessível, assim como a base de desejos (mapeada como base de eventos) e a base de emoções (contida na base de crenças). Veja que neste cenário nenhum critério de mapeamento é ferido, pois a ordem de execução pode ser mantida e a *Update Drives* terá acesso a todas as bases necessárias.

A segunda abordagem é utilizar os planos, de tal forma que quando um evento desejado ocorrer, podemos alterar o valor da crença que representa o *drive*. Nesta segunda será possível ter acesso às crenças e às emoções (pela base de crenças), desejos e intenções (por ações internas do Jason). Entretanto, usar planos para atualizar os *drives* mudará a ordem de execução da atualização dos *drives* (ela será feita na *Execute Intention* do Jason, o que é distante da função *Select Event* do Jason, onde ocorre o resto das atualizações dos desejos) e isto torna o mapeamento **incompleto**.

Nós optamos por uma proposta híbrida destas duas, onde o incremento do valor dos *drives* e a ativação (ou não) deles quando passam um limiar é realizada pela extensão da *Select Event*. Já o monitoramento de eventos que alterem seu valor é realizado pelos planos. A motivação disto é que justamente o processo de incremento e ativação/desativação de *drives* é independente do domínio, podendo ser reaproveitado em vários agentes. Já o processo de modificação de seus valores de acordo com eventos ocorridos é dependente de domínio e por isto é implementado através dos planos, permitindo rápida modificação de um agente para o outro. Veja que, infelizmente, este mapeamento é **incompleto**, pois como monitoramos os eventos nos planos, a atualização dos *drives* terá uma ordem diferente da original.

Quanto à tarefa de verificar quais *drives* extrapolam ou não seu valor, podemos inserir um algoritmo na função *Select Event* do Jason para realizar tal tarefa. O algoritmo deve verificar para todo *drive* – da forma **drive**(**nome**, **valor**, **limiteSup**, **limiteInf, incremento**) contido na base de crenças – se seu **valor** é maior que o seu **limiteSup** e não está se desejando nenhuma meta identificada por **nome**, então deve-se inserir um desejo **nome(sup)** na base de eventos do Jason. Similarmente, se o **valor** é menor que o **limiteInf** e não existir meta com o nome **nome(inf)**, ele deve inserir um desejo **nome(inf)**. Por outro lado, se o **valor** estiver entre os **limiteSup** e **limiteInf** e se deseja uma meta identificada por **nome(sup)** ou **nome(inf)**, então remove-se esta(s) meta(s) da base de eventos. Por último, deve-se atualizar o **valor** do **drive**, incrementando-o com o valor **incremento**.

Por último, temos que mapear a função *Prioritize Desires* que deverá atualizar a base de desejos rotulando os desejos de acordo com sua prioridade dado o contexto atual. Esta operação é papel da *Select Event* do Jason, visto que ela detecta qual o evento mais apto a se tornar uma intenção, priorizando-o com sua seleção. Logo, *Prioritize Desires* pode ser mapeada na *Select Event* devido à similaridade. Nosso mapeamento é **completo**, pois colocar a *Prioritize Desires* na *Select Event* manterá a ordem de execução da UEBDI e já que a *Select Event* tem acesso a todas as bases exigidas pela função *Prioritize Desires*.

5.2.6 Mapeamento do módulo de intenções

A UEBDI define o módulo de intenções composto pela *Intention Base* e pela *Filter Function*. A *Intention Base* pode ser mapeada na base de intenções do Jason, adicionando-se as devidas extensões já discutidas no tópico 5.2.2.

No Jason a seleção e reconsideração de intenções não ocorrem em uma função discriminada, mas sim separadamente. A inclusão de intenções ocorre implicitamente na *Select Event*, pois o sistema sempre tentará associar um plano ao evento selecionado para torná-lo uma intenção. A reconsideração se dá pelo processo de exclusão de uma intenção por ações internas do Jason a serem executadas em um plano.

Devido a isto, observa-se na Figura 25 o possível mapeamento da *Filter Function* da UEBDI sobre o Jason. Vamos considerar que a função *Update Intention* da arquitetura UEBDI, que possui o papel de atualização das intenções, pode ser mapeada na função de *Select Event* (inclusão) e nos planos do Jason (exclusão), e será possível ter as influências dos desejos, das crenças e do estado emocional (que está na base de crenças). A *Select Event* tem acesso a todas as bases exigidas pela *Update Intention*. Entretanto, quando houver exclusão de intenções pelos planos, a ordem de execução será mudada em relação a UEBDI, o que torna o mapeamento **incompleto**.

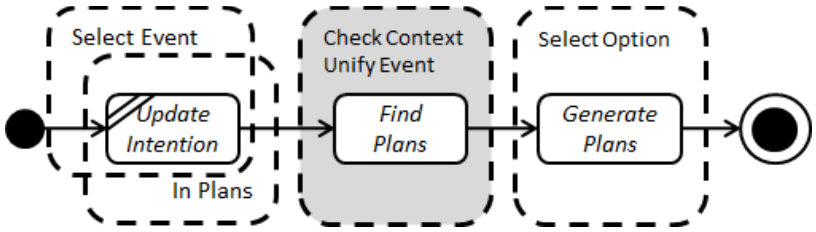


Figura 25: Mapeamento *Filter Function*

Quanto a próxima sub-função do UEBDI, a *Find Plans*, ela representa exatamente as funções *Check Context* e *Unify Event* do Jason. Consideramos este mapeamento **completo**, pois a ordem de execução da UEBDI foi respeitada e a *Find Plans* só exige as bases de crenças e planos, ambas acessíveis pela *Check Context* e *Unify Context* do Jason.

A *Generate Plans* naturalmente não poderá ser mapeada em nenhuma função do Jason visto que o Jason não lida com planejamento. Consideramos que, caso se queira esta propriedade, deve-se anexar um terceiro módulo de geração de planos na *Select Option* do Jason. Este mapeamento é **completo**, pois a ordem de execução é mantida e a *Belief Base*, a *Emotion Base* e a *Action Base* são acessíveis pela *Select Option*.

Outra questão importante é que a UEBDI ao término da *Filter Function* especifica um conjunto novo de intenções onde cada uma contém uma coleção de planos, enquanto que o

Jason ao término da *Select Option* especifica uma nova intenção, sempre associada a um único plano. Isto faz com que o Jason não tente executar vários planos para uma intenção até satisfazê-la. Isto é contornável a partir do momento que existem padrões de projeto (Hübner et al. 2006) e diretrizes de execução (Bordini et al. 2007 chap. 8) que implementam estas propriedades às intenções. Não consideraremos este fato um problema neste módulo. Mas isto se tornará um inconveniente no módulo de atuação.

5.2.7 Mapeamento do módulo de atuação

Na arquitetura UEBDI temos a função *Execute Function* que visa satisfazer uma intenção executando os planos associados e promovendo a atuação do agente. Ela é composta por duas subfunções: a primeira, a *Select Intention*, irá selecionar uma intenção da *Intention Base* para ser efetuada. E a segunda, a *Archieve Intention*, que irá executar os possíveis planos desta intenção até satisfazê-la.

O mapeamento se dá como na Figura 26. O Jason possui também uma função chamada *Select Intention* que irá desempenhar o mesmo papel que sua homônima na UEBDI, então esta pode ser mapeada na *Select Intention*. Veja que chamar este função na sua homônima mantém a ordem de execução da função na UEBDI. Além disto, através da *Select Intention* do Jason, é possível acessar a base de intenções e a base de emoções (através da base de crenças). O que torna o mapeamento **completo**.

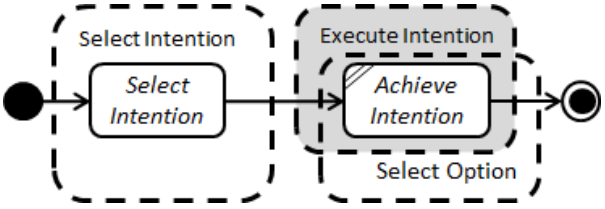


Figura 26: Mapeamento *Execute Function*

Antes de continuar, deve-se ressaltar uma diferença fundamental entre o Jason e a UEBDI: AUEBDI seleciona um conjunto de intenções com seus possíveis planos na *Filter Function* e persiste esta estrutura até a *Select Intention*, que selecionará qual a intenção a ser executada e até a *Archieve Intention*, que selecionará a ordem de execução dos planos. Já o Jason faz a seleção de planos antecipadamente. Primeiro ele seleciona o plano associado a intenção recém criada com a *Select Option*; posteriormente ele irá selecionar a intenção corrente através da *Select Intention*. Esta diferença faz com que não seja possível realizar o comportamento da *Archieve Intention* (de executar todos os planos possíveis até alcançar a intenção) de forma direta no Jason. Entretanto é possível simular os múltiplos planos do

UEBDI no Jason através de padrões de projeto (Hübner et al. 2006). O único problema disto é que a ordem da responsabilidade de selecionar os planos é invertida.

Outra diferença no módulo UEBDI em relação ao Jason é a forma de executar um plano: A UEBDI executa atomicamente os planos associados a uma intenção, enquanto que o Jason executa, para dada intenção, apenas partes do plano a cada ciclo de execução. Novamente, não consideraremos isto um problema maior, pois é possível forçar a execução de um plano por vários ciclos através da seleção da intenção que o interessa pela *Select Intention* ou de diretivas dadas a um plano no código asl do agente.

Posteriormente, a *Archieve Intention* (UEBDI) executa os planos associados a intenção em uma ordem de prioridade. No Jason quem faz a seleção de planos é a *Option Function* e quem executa propriamente um plano associado é a *Execute Intention*.

Assim, a *Archieve Intention* será mapeada na *Execute Intention* e na *Select Option*, considerando que é possível simular os múltiplos planos no Jason. Ela exige as *Emotion*, *Resource* e *Action Base* e todas elas estão contidas na *Belief Base* e são acessíveis. Porém a ordem de execução é invertida, conforme discutido. Por isto consideramos o mapeamento da *Archieve Intention* é **incompleto**.

5.2.8 Mapeamento do módulo de emoções

Naturalmente não existem representantes do módulo emocional UEBDI no Jason. Consideramos que este módulo é composto pela *Emotion Base*, pela *First Emotion Review Function*, pela *Second Emotion Review Function* e pela *Critical Context*. Como discutido, podemos mapear a *Emotion Base* para dentro da *Belief Base*. De tal forma que uma emoção seja uma crença da forma **emotion(componente$_1$, componente$_2$, ..., componente$_n$)** onde a natureza dos componentes e o número de componentes (n) dependem da aplicação que está sendo desenvolvida. O acesso às emoções pode ser feito diretamente pela base de crenças ou através de uma classe envelope (*wrapper*) que transformará as crenças em objetos relativos ao domínio da emoção e mais fáceis de manipular durante a dinâmica emocional (as funções de revisão).

Quanto às funções de revisão, vê-se seu mapeamento na Figura 27. Iremos adotar a seguinte estratégia para mapeá-las: adicionaremos estas funções dentro das funções do Jason de tal forma que preserve a ordem de execução definida na UEBDI, mantendo as mesmas funções predecessoras e sucessoras. Desta forma, a *First Emotion Review Function* é adicionada na *Belief Update Function* logo após a *Emotional Filter* da *Perception Function*. Nesta posição, ela terá acesso às percepções geradas pelas sub-funções da *Perception*

Function, acesso às emoções (através da *Belief Base*), aos *drives* (que estão contidos na *Belief Base* também) e às intenções. Como a ordem foi mantida é possível acessar todas as bases exigidas, consideramos o mapeamento **completo**.

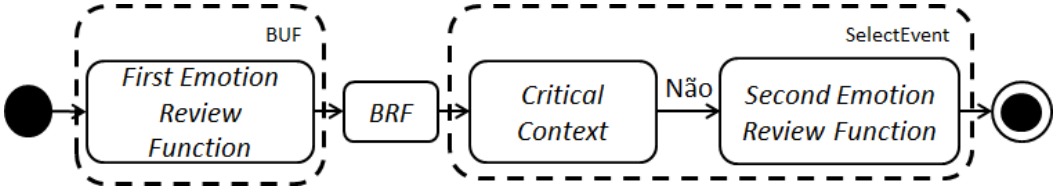


Figura 27: Mapeamento das funções de emoção

A *First Emotion Review Function* é constituída por duas sub-funções: a *First Emotional Elicitors* e a *First Emotional Update*. Serão implementadas também na *Buf*. Como vimos, a ordem de execução destas não importa, então não nos preocuparemos com isto no nosso mapeamento e na nossa implementação. O mesmo se aplica a *Second Emotional Review Function*.

Posteriormente existe a função *Critical Context* e a função *Second Emotion Review Function*. Ambas podem ser colocadas nesta ordem na *Select Event* do Jason, após a função *Forget Beliefs* mapeada do UEBDI. A *Critical Context* terá acesso as suas duas bases exigidas, a *Emotion Base* e a *Resource Base*, contidas na *Belief Base* do Jason. Já a *Second Emotion Review Function* terá acesso a *Belief Base*, a *Emotion Base* (contida na *Belief Base*), a *Desire Base* (contida na *Event Base* –metas– e na *Belief Base* –*drives*– do Jason) e a *Intention Base*. Em ambas as funções, a ordem foi mantida e é possível acessar todas as bases exigidas. Logo, o mapeamento é **completo**.

5.3 Resumo do Mapeamento

Conforme o mapeamento realizado, podemos observar que a maior fonte de incompatibilidade entre o Jason e a UEBDI foi a ordem de execução das funções. Destacamos quatro itens:

1) Não é possível tratar o fluxo de mensagens do Jason antes dele ser introduzido na base de crenças. Então não é possível tratá-lo pela *Perception Function*;
2) Quando um meta ou *drive* é atualizado em um plano pelos planos, a ordem da *Update Desires* e *Update Drives* é trocada em relação a orinal definida na UEBDI;
3) A *Update Intention* da UEBDI, quando executada nos Planos, tem sua ordem de execução invertida;

4) Os fluxos de execução de planos no Jason e na UEBDI são diferentes. No primeiro executa-se apenas um plano por intenção, em pequenas partes por ciclo. Já na segunda, se executa todos os planos por intenção completamente em um único ciclo de execução.

Algumas das incompatibilidades encontradas podem ser contornadas se não forem usados planos para a implementação da funcionalidade (item (2) e (3)). A questão de mensagens do Jason (item 1) não é um problema maior, pois como elas são eventos, podem ser tratadas como metas na UEBDI. E a questão da ordem de execução dos planos que é mais fundamental (item 4), visto que a ordem do Jason diverge em relação a UEBDI e não havendo uma forma de contornar problema para manter a ordem da UEBDI.

Na **Erro! Autoreferência de indicador não válida.** é exibido o resumo do mapeamento entre a arquitetura UEBDI e o Jason. Os números como expoente indicam o item discutido acima relacionado a esta extensão. Particularmente a *Perception Base* não precisa ser mapeada porque ela foi um artifício conceitual introduzido pela UEBDI que não exige implementação (veja tópico 4.1 sobre o assunto).

Arquitetura UEBDI	**Jason sem extensão**
Resource Base	*Belief Base*
Action Base	*Belief Base*
Emotion Base	*Belief Base*
Módulo de Percepção	
Perception Base	(não necessário)
Perception Function	(Somente Percepções e não Mensagens do Jason)[1]
Sensor Filter	*Perceive*
Resource Filter	*BUF*
Rules Filter	*BUF*
Emotional Filter	*BUF*
Intentional Filter	*BUF*
Módulo de Crença	
Belief Base	*Belief Base*
Belief Review Function	
Update Beliefs	*Planos* + *BUF* + *BRF*
Tag Beliefs	*BRF*
Forget Beliefs	*Select Event*
Módulo de Desejos	
Desire Base	*Event Base* (*metas*) + *Belief Base* (*drives*)
Option Function	
Update Desires	*Planos*[2]

Update Drives	*Select Event + Planos*(2)
Prioritize Desires	*Select Event*
Módulo de Intenções	
Intention Base	*Intention Base*
Filter Function	
Update Intention	*Select Event + Planos*(3)
Find Plans	*Check Conflicts + Unify Event*
Generate Plans	*Select Option*
Módulo de Execução	
Execute Function	(fluxo compatível)
Select Intention	*Select Intention*
Achieve Intention	*Execute Intention*(4)
Módulo de Emoções	
Emotion Base	*Belief Base*
First Emotion Review Function	*BUF*
Critical Context	*Select Event*
Second Emotion Review Function	*Select Event*

Tabela 4: Resumo do mapeamento da UEBDI para o Jason

6 Experimentos

Para validarmos nossa proposta, iremos neste capítulo apresentar uma série de experimentos que exercitam as influências do módulo emocional sobre um dos outros módulos da arquitetura BDI. Cada experimento focará a influência sobre um módulo em específico e terá um sub-tópico chamado "mapeamento e implementação" que mostra como o problema foi solucionado com a arquitetura UEBDI e como esta solução foi mapeada e implementada no Jason.

Serão ao todo cinco experimentos: (6.1) Incêndio, que tratará da influência da emoção sobre o módulo de percepção; (6.2) Professor-Aluno, que tratará da influência da emoção sobre o módulo de crenças; (6.3) Diversão, que tratará da influência da emoção sobre o módulo de desejos; (6.4) Dilema do Prisioneiro Iterado (IPD), que tratará da influência da emoção sobre o módulo de seleção de intenções e (6.5) Alocação de Tarefas, que tratará da influência da emoção sobre o módulo de atuação.

Iremos utilizar um modelo emocional simples para cada experimento. Nós somos encorajados a fazer isto visto que (i) nosso foco não é o modelo emocional, mas a influência e (ii) um modelo simples facilitará a compreensão dos exemplos. Julgamos este conjunto de experimentos suficiente, pois ele demonstra que as cinco influências do módulo emocional sobre o resto da arquitetura BDI podem ser efetuadas e causarão alteração significativa dentro do comportamento do agente.

6.1 Incêndio: Influência sobre o módulo de percepção

O experimento será embarcado no seguinte cenário: vários agentes estão em uma sala e são surpreendidos por um incêndio. Para sobreviverem eles devem fugir o mais rápido possível. Este experimento e proposta de agente são uma forma simplificada dos realizados por Signoretti em (Signoretti 2012, Signoretti et al. 2010 p. 201).

O ambiente é uma grade 2D de dimensão 30x30 e cercada por paredes. Nas células (6, 15), (23, 15), (15, 6) e (15, 23) serão colocados focos de incêndio que se propagarão por todo

o cenário de forma radial[10]. Uma célula pode estar com fogo (em chamas) com uma intensidade de 0 até 100. Um agente morre quando está em uma célula onde o fogo atinge intensidade acima de 50. O ambiente também possui quatro saídas nas células (0,0), (29,29), (29,0) e (0, 29). O objetivo dos agentes é chegar a uma destas saídas, onde poderão escapar da sala e serem salvos do fogo. Um agente pode se mover para as células adjacentes a que reside e um agente não colide com o outro (podendo uma célula conter dois agentes).

Quanto as suas percepções, todos os agentes são "oniscientes", isto é, eles recebem do ambiente percepções de todas as células do ambiente da forma: *cell*(*x*, *y*, *deltaFire*, *exit*), onde *x* e *y* são a posição da célula, *deltaFire* indica qual foi a variação de fogo nesta célula e *exit* é igual *true* caso exista uma saída nesta célula ou *false* caso contrário. Observe que nesta configuração, um agente irá perceber 30x30 = 900 percepções do ambiente.

Um agente irá utilizar uma versão do algoritmo A* proposta por (Lester 2004) e implementada para o Jason (como ação interna) para criar o caminho de sua posição atual até a saída mais próxima utilizando todas as percepções, permitindo a ele fugir dos focos de incêndio e se salvar quando chegar à saída.

6.1.1 Agente Emocional

O agente emocional proposto por (Signoretti 2012) tinha como princípio usar um conjunto de emoções para derivar uma métrica que representasse o quanto o ambiente está crítico para o agente. Com esta métrica o agente diminui o número de percepções a serem passadas para o seu processo de deliberação, acelerando então a execução deste processo.

Nosso modelo emocional simplificado é constituído por estado emocional com uma dimensão emocional chamada de *Relaxado-Ansioso*, que varia no raio [-1, +1]. O valor zero é considerado o seu valor neutro. O valor negativo -1 indica que o agente está totalmente ansioso por causas que lhe causaram danos ou por apreensão devido a previsão de um evento prejudicial e o valor +1 indica que o agente está totalmente relaxado por causas que lhe causaram ganhos ou devido a previsão de um evento benéfico. A cada ciclo de raciocínio a emoção *Relaxado-Ansioso* é decrementada (caso sua intensidade seja positiva) ou incrementada (caso sua intensidade seja negativa) de 0,002, até que volte ao valor neutro 0. Quanto a eliciação emocional, nós definimos duas regras:

[10] Utilizamos um modelo informal para simular o fogo. Resumidamente cada célula tem certa quantidade de combustível e quando ela está adjacente a outra que possui fogo, "pega" fogo também até acabar o combustível, quando o fogo será extinto.

i) Calcula-se o valor de *importance*(*x*, *y*) de todas as células percebidas com valor de *deltaFire* negativo e seleciona-se o maior valor, que chamaremos de *maxIncEmotion*. Este valor será usado para incrementar o valor de *Estresse*. Veja que quando o fogo está diminuindo (sendo indicado pelo *deltaFire* negativo), indica que o estado do ambiente está melhorando para o agente, e isto deve incrementar a intensidade da emoção *Relaxado-Ansioso*.

ii) Calcula-se o valor *importance*(*x*, *y*) de todas as células percebidas com o valor de *deltaFire* positivo e seleciona-se o maior valor que será chamado de *maxDecEmotion*. Este valor será usado para decrementar a emoção *Relaxado-Ansioso*. Veja que como o valor de *deltaFire* está aumentando, quer dizer que o estado do ambiente está se tornando pior para o agente, e isto deve negativar o valor de *Relaxado-Ansioso*.

A função *importance*(*x*, *y*) é dada na Eq. 30. Observe que *distance* é uma função que acha a distância entre dois pontos, *diagonal* é o tamanho da diagonal do ambiente e (x_0,y_0) a posição da célula onde o agente está. O objetivo desta função é obter um valor que represente a importância de dada célula, levando em conta a variação do fogo e sua distância em relação ao agente. Isto é, quanto maior for o valor de *importance*, mais próxima é a célula da posição atual do agente e maior foi a sua variação de fogo. Vale ressaltar que esta função é normalizada entre [0, 1], pois dividimos o valor de *deltaFire* pelo maior valor possível (100) e o valor da distância pelo maior valor possível (*diagonal*, que é a distância entre dois cantos do ambiente 2D).

Observe também que sempre é selecionado o maior valor de *importance* obtido entre o conjunto de células com deltas negativos e positivos. Isto é feito pelo seguinte motivo: a célula com maior valor de *importance* é a que está mais próxima *e* que possui maior variação de *deltaFire*, isto indica que ela é a mais relevante para o agente dentre todas do conjunto. Desta forma, o agente sempre está "atento" emocionalmente à percepção mais significativa.

$$importance(x,y) = \left|\frac{deltaFire}{100}\right| . \left(1 - \frac{distance(x,y,x_0,y_0)}{diagonal}\right) \qquad \text{(Eq. 30)}$$

Depois de obtidos, deve-se incrementar o valor *maxIncEmotion* e decrementar o valor *maxDecEmotion* da intensidade emocional *Relaxado-Ansioso*, limitando o seu valor entre [-1, 1], concluindo a eliciação da emoção sobre o estado emocional.

Agora é preciso definir a influência emocional. No nosso agente, o estado emocional irá influenciar o filtro de percepções, reduzindo o número de percepções que serão enviadas para a *Belief Update Function* e ao resto do ciclo BDI, em seguida.

A primeira coisa a ser feita é ordenar de forma crescente as percepções sobre células utilizando a equação *importance*. Após isto, deve-se calcular o valor *L* que indicará a quantidade de elementos que devem permanecer na lista e serem enviados para o processo de raciocínio.

Para o cálculo de *L*, os autores de (Signoretti 2012, Signoretti et al. 2010) consideraram que *L* deve ter uma relação com a intensidade do estado emocional (intensidade de *Relaxado-Ansioso*). Porém deve ser definida de tal forma que quando a intensidade da emoção *Relaxado-Ansioso* for neutra (zero) o valor de *L* deve ser máximo, fazendo o agente considerar todas as percepções possíveis, sobre a conjectura que quando o agente se encontra neste estado pode gastar recursos processando todos estes elementos. Quando a intensidade de *Relaxado-Ansioso* variar para valores positivos ou negativos, o valor de *L* deve diminuir, formando uma curva "U" invertida (gaussiana) em relação ao valor a intensidade de *Relaxado-Ansioso*. O princípio teórico de utilizar esta curva é que existem evidências que indivíduos com humor moderado tendem a ter maior desempenho nas capacidades cognitivas, incluindo a percepção. Enquanto que indivíduos muito relaxados ou ansiosos tentem a diminuir o desempenho (Signoretti 2012 p. 92).

Isto fará o agente considerar menos percepções no processo de raciocínio, seja porque o ambiente está inóspito, pois o fogo está aumentando (*deltaFire* positivo) ou seja porque o ambiente está próspero, pois o fogo está diminuindo (*deltaFire* negativo). A função que dá o valor de *L* é definida na Eq. 31. *N* é o tamanho da lista de percepções que será podada por *L* e *Int* é a intensidade do *Relaxado-Ansioso*. A Figura 28 exibe um exemplo de comportamento de L com $N = 100$.

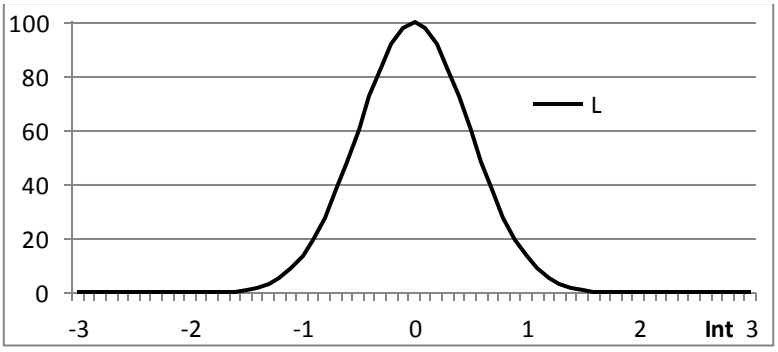


Figura 28: *L* por intensidade de *Relaxado-Ansioso*

$$L = N.e^{-2.Int^2} \quad \text{(Eq. 31)}$$

Na prática, dado como nosso ambiente foi definido, o agente terá valores de *deltaFire* positivos, fazendo aumentar a intensidade da emoção *Relaxado-Ansioso* dos agentes aumentar e, consequentemente, fazendo *L* diminuir.

6.1.2 Mapeamento e Implementação

As regras de eliciação da emoção que incrementam/decrementam a intensidade da emoção *Relaxado-Ansioso* de acordo com a percepção corrente são implementadas em nossa arquitetura na *First Emotion Review Function* e na sub-função *First Emotional Elicitors*. Elas devem ser implementadas na primeira função, pois elas usam a informação de *deltaFire* proveniente das percepções, que só são acessíveis por esta função de revisão de emoções. Como previsto no mapeamento, estas funções serão implementadas na função *Belief Update Function* do Jason.

Quanto à regra de decremento emocional, que fará a intensidade emocional retornar a zero, decidimos mapeá-la na sub-função *Second Emotional Update* contida em *Second Emotion Review Function*. Em tese poderíamos implementar esta operação na *First Emotional Update* (na *First Review Emotional Function*) ou na *Second Emotional Update*, pois esta operação exige somente a base de crenças, acessada por ambas. Entretanto é melhor usarmos a *Second Emotional Update*, pois está mais de acordo com a teoria emocional que diz que o decréscimo emocional ocorre como uma operação cognitiva, típica da *Second Emotion Review Function*, como (Hernández et al. 2004, Jiang and Vidal 2006) sugerem. Esta função foi mapeada na *Select Event* do Jason, como previsto no mapeamento.

Por fim, quanto a influência emocional, as percepções serão filtradas naturalmente na *Perception Function* e na sua sub-função *Emotional Filter*. Esta *Emotional Filter* foi implementada na *Belief Update Function* do Jason, como previsto no mapeamento.

6.1.3 Experimento

A hipótese que trataremos para este problema é a seguinte: um grupo de agentes emocionais conseguirá ter mais sobreviventes ao final da simulação se comparado a um grupo de não emocionais. Espera-se que o uso do modelo emocional e sua influência sobre as percepções, filtrando-as, irá diminuir o número de percepções manipuladas pelo agente, tornando este mais rápido e facilitando sua fuga. A hipótese já foi provada pelos autores de (Signoretti 2012, Signoretti et al. 2010), no que eles chamam de foco temporal, ganhando significado no número de agentes sobreviventes ao final da simulação.

Para verificar a hipótese, usaremos as medidas:

1) Número de agentes sobreviventes ao final da simulação;
2) Tempo total de simulação em milissegundos (quando todos os agentes se salvam ou morrem);

3) Número de percepções consideradas na tomada de decisão (valor de *L*).

A medida (1) será a principal, pois indica diretamente quantos agentes sobreviveram ao final da simulação. A (2) será usada para indicar a rapidez com que os agentes conseguiram tomas as suas decisões. A (3) indica a média de percepções consideradas pelo agente após a filtragem das percepções. Deve-se lembrar que o agente não emocional não irá filtrar as percepções e por isto sabemos previamente que este valor será igual ao número de células (30x30 = 900).

Observe também que a interpretação de (2) carece de um detalhe: O sistema terá seu tempo de simulação diminuído se os agentes demorarem menos para interagir OU se eles morrerem mais rápido. Então esta métrica só indica uma melhora no desempenho para nós se ela for acompanhada por um maior número de agentes sobreviventes, e não o contrário.

Serão colocados 10 agentes dentro do ambiente já especificado para se realizar uma instância do experimento. Serão executadas 30 instâncias deste experimento e obtidas as médias das medidas (1) e (2) para o agente não emocional (onisciente) e emocional (com filtro de percepção) e a média da medida (3) para o agente emocional. A Tabela mostra os resultados:

	Emocional	Não Emocional
Agentes sobreviventes	9,77	7,40
Tempo de simulação (ms)	8041	9479
Número de percepções consideradas (L)	330,46	900

Tabela 5: Média de sobreviventes e tempo de simulação

Como podemos observar, o uso do módulo emocional com filtro das percepções conseguiu fazer as instâncias com agentes emocionais terem maior número de sobreviventes (9,77 contra 7,4 do não emocional) e que a simulação durasse menos tempo na média (8041 milissegundos contra 9479 milissegundos do não emocional). Inclusive, o agente emocional demorou menos tempo e teve maior número de sobreviventes, como nós indicamos desejável para o bom desempenho de acordo com a medida (2). Além disto, foi constatado que o valor de *L* médio usado em toda a simulação dos agentes emocionais foi igual a 330,46 percepções (onde o total gerado pelo ambiente é 900), demonstrando que houve uma queda significativa das percepções a serem tratadas. Com isto, reafirmando os resultados dos trabalhos (Signoretti 2012, Signoretti et al. 2010), há indícios que devido a queda do número de elementos processados durante a simulação (baixo valor de *L*), o agente emocional conseguiu melhorar o número de agentes salvos (medida (1)) e o tempo de simulação (medida (2)).

6.2 ProfessorAluno: Influência sobre o modulo de crenças

Este experimento consiste na simulação de uma avaliação feita por um professor a um aluno. O professor irá fazer várias perguntas, em sequência da forma: "Qual a soma de todos os números associados a sentença x_i?", onde x_i é o nome de uma sentença (no Jason, chamamos isto de *functor*) feita no turno *i*. O aluno, por sua vez, possui em sua base de crenças várias crenças da forma $x_i(Y)$, onde Y é um valor numérico positivo diferente de zero. O aluno então deve somar todos estes valores Y e retornar a soma na forma $sum(x_i, S_{resposta})$, onde $S_{resposta}$ é o valor da soma achada pelo aluno. É garantido que o professor sempre perguntará por um x_i cujo existam N elementos na base de crenças do aluno e esta quantidade de elementos é a mesma para qualquer x_i. Note que dentro destas restrições, o valor de soma sempre será positivo maior que zero, pois sempre existe pelo menos um elemento que é maior que zero. Neste agente, sabe-se que o custo da consulta, comparado com o custo da soma de um elemento, é insignificante[11]. Também é garantido que o professor sabe previamente o resultado da soma, dado por S_{certo}.

Por exemplo, sendo x_i igual a "vendas", e o agente aluno possui em sua base de crenças os valores {vendas(10), vendas(6), vendas(1)}. Se o professor fizer a pergunta "Qual a soma de vendas?", o aluno deve responder "*sum*(vendas, 17)", cujo 17 é a soma dos valores associados as crenças em sua base de crenças (10 + 6 + 1).

O professor esperará no turno *i* o tempo $wait_i$, em milissegundos, pela resposta do aluno para a sentença $x_i(Y)$. Se o aluno responder a tempo, ele receberá uma *Nota* dependente da soma respondida $S_{resposta}$ e da resposta certa S_{certo}, dada pela Eq. 32. A Figura 29 demonstra o gráfico de nota por $S_{resposta}$, com S_{certo} igual a 1. Se o aluno não responder a tempo, ele receberá $NOTA = -1$. O professor irá enviar a nota ao aluno com a sentença $nota(x_i, NOTA)$. Após enviar a nota ao aluno e o aluno confirmar o recebimento com *ack(i)*, o professor irá fazer a próxima pergunta x_{i+1}. Por fim, o professor acumula a soma destas notas em uma crença chamada *totalNotas*.

$$Nota = \left[1 - \min\left(1, \frac{|S_{resposta} - S_{certo}|}{S_{certo}}\right)\right] \qquad \text{(Eq. 32)}$$

[11] Na implementação, operações falsas e sem resultado durante a soma para garantir esta característica do problema.

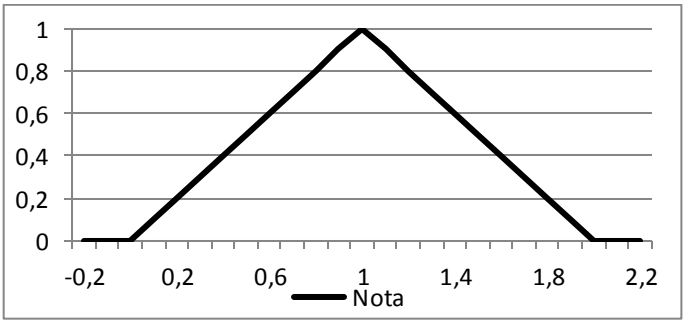


Figura 29: Nota por $S_{resposta}$

O valor de $wait_i$ irá variar da seguinte forma: $wait_o$ começará com um valor determinado e o próximo $wait_1$ será igual a $wait_0$ -1. De tal forma que $wait_{i+1} = wait_i$ -1. Quando o valor de $wait_i$ chegar a zero, o professor interromperá o teste. Entretanto, o aluno não sabe exatamente qual será o comportamento do professor, apenas que ele esperará menos no próximo turno.

Observe que um agente pode não responder a tempo devido ao custo da soma. Para lidar com isto um agente pode somar apenas os L elementos $x_i(Y)$ com Y maiores valores (que são mais significativos na soma). Desta forma, ele diminui o custo do somatório e pode responder a tempo. Como serão os L primeiros Y maiores números, a soma deles será mais próxima do valor real esperado do que se o agente escolhesse aleatoriamente os Y. O problema de decisão do agente consiste em justamente determinar este L, o tamanho da lista.

Vamos definir um agente não emocional simples que irá seguir a seguinte heurística: Ele no início da simulação constrói uma lista com todos os elementos em ordem decrescente de valor, onde o tamanho da lista é $L = N$. Ele usará esta lista para determinar quais elementos irá somar para responder o professor. Quando o agente falhar, ele decrementará *decValue* de L e reconstruirá a lista. Assim, no próximo turno, como a lista possui menos elementos, ele somará menos.

Vamos agora definir um agente emocional: este agente possuirá uma única emoção chamada *Estresse* com intensidade igual a E_{int}. O seu valor nunca extrapola o raio de [0, E_{max}], onde $E_{max} = 100$. A cada resposta atrasada reportada pelo professor, o *Estresse* é incrementado *incEmoValue*. E a cada resposta certa, o estresse decrementa um valor igual a *decEmoValue*.

A emoção *Estresse* influenciará o tamanho da lista, sendo o número de elementos inversamente proporcional a intensidade E_{int}.. De tal forma que $L = \max\left(1, N.\left[1 - \left(\frac{E_{int}}{E_{max}}\right)\right]\right)$. Note que o valor de L nunca será menor do que 1 e deverá ser no máximo igual a N. A cada turno que passar, o agente calculará L de acordo com a emoção corrente e reconstruirá a lista. Como a reconstrução ocorre após o professor dizer se o aluno respondeu a tempo e

antes do aluno responder ao professor com *ack(i)*, isto não atrasará o aluno nas próximas perguntas.

Observe também que quando o aluno atrasar a resposta, ele irá decrementar seu estresse, e, proporcionalmente, diminuir o valor de *L*. Nos próximos turnos, se ele obter sucesso na resposta, irá decrementar seu estresse e, proporcionalmente, aumentar o valor de *L*.

6.2.1 Mapeamento e Implementação

A emoção *Estresse* é incrementada e decrementada a partir das crenças do agente recebidas como mensagens do Professor. Logo, para realizar a eliciação emocional é preciso acessar a base de crenças e, por isto, implementamos este processo na *Second Emotion Review Function*. Como só definimos o processo de eliciação e não de decremento, só será preciso modificar a sub-função *Second Emotion Elicitors*. Conforme o mapeamento, esta função é implementada na função *Select Event* do Jason.

Já a influência emocional ocorre justamente na construção da lista de maiores valores baseado na Intensidade de *Estresse*, de tal forma que a emoção influencia a atualização das crenças. Na nossa arquitetura isto é feito pela *Belief Review Function* e pela sub-função *Update Beliefs*. Esta foi mapeada e implementada como um plano no Jason.

6.2.2 Experimentos

Nossa hipótese é que o agente emocional consiga obter melhor nota que o agente não-emocional, visto que ele além de decrementar seu valor de falha – como o agente não-emocional – também irá ajustar o valor de *L* para um valor que lhe dê maior nota, visto que sua emoção irá decrementar com o tempo, aumentando o valor de *L*.

Vamos para isso realizar dois experimentos: O primeiro (6.2.2.1) irá achar um valor ideal de *decValue* para o agente não-emocional que maximize sua nota final. Já o segundo (6.2.2.2) irá determinar o valor de *decEmoValue* ideal para um agente emocional que use um valor de *incEmoValue* igual ao valor ideal achado para *decValue*.

Os experimentos foram realizados em um computador AMD Athlon X2 4800 com 2 Gigas de RAM e Java 1.7.

6.2.2.1 Agente não-emocional (decValue)

Quanto ao primeiro experimento, será realizado da seguinte forma: um professor irá fazer perguntas ao aluno com um $wait_0$ = 100 milissegundos e irá decrementando de 1 a cada

turno e o professor só realizará perguntas sobre a sentença $a\ (x_i)$. O agente terá um total de 100 crenças onde da forma: {a(100), a(99), a(98), ...}.

Serão realizadas 30 instâncias desta configuração para um dado valor de *decValue*, para que possamos obter a média de *totalNotas* nas 30 execuções. Isto irá eliminar possíveis interferências nos valores. Serão testados os valores de *decValue* variando entre [0, 99], de tal forma que no final serão realizadas 30*100 = 3000 instâncias do experimento.

Os resultados são vistos na Tabela 6 e na Figura 30. Observa-se que os melhores valores de nota obtidos ocorrem com *decValue* entre [10, 25]. Logo, valores dentro desta maximizam a nota do aluno.

6.2.2.2 Agente Emocional (decEmoValue)

Já o segundo experimento irá executar um agente emocional variando o valor de *decEmoValue* entre os valores [0, 15] e com valor de *incEmoValue* igual a 15, que é o valor mediano entre os valores que otimizaram a nota do aluno no experimento 1. Veja que não faz sentido usarmos valores de *decEmoValue* iguais ou maiores que 15, pois isto faria o agente não ter estresse (visto que nosso *incEmoValue* é 15) e logo não diminuir seu valor *L* quando não consegue responder a tempo. Uma instância deste experimento será executada 30 vezes para cada valor de *decEmoValue* e obteremos a média de *totalNotas* com dado *decEmoValue*. Estes valores são exibidos na Tabela 7 e na Figura .

Como podemos ver, o algoritmo do agente emocional não conseguiu otimizar o seu desempenho, para nenhum valor de *decEmoValue*, o que nega nossa hipótese inicial. Inclusive, vê-se que a nota foi inversamente proporcional que o valor de *decEmoValue*, assim se pode considerar que quanto maior o *decEmoValue*, pior será o desempenho do agente. Vamos proceder com a análise dos resultados para tentar compreender por que o agente emocional não conseguiu otimizar o resultado.

Comecemos pelo valor de *decEmoValue* igual a 0. Em tese, se o *Estresse* não vai diminuir, o valor de *L* nunca irá aumentar novamente, o que torna nosso agente emocional similar ao agente não emocional. Entretanto, seu desempenho foi inferior. Consideramos que isto ocorreu pelo próprio custo da arquitetura UEBDI, que fez o agente agir mais lentamente e a não responder tão rápido quanto o não emocional, que não é estendido pelo *framework*.

Antes de analisar valores de *decEmoValue* não nulos, vamos considerar o seguinte: neste cenário *decEmoValue* equivale a um "*incValue"*, que incrementa o valor de *L* a cada turno, desta forma podemos desconsiderar a relação entre a emoção *Estresse* e *L*.

Vamos considerar também a seguinte situação perfeita: o custo de acessar os *L* elementos é igual a 1 milissegundo. Também vamos considerar *incValue* igual a 1 e *decValue* igual a 20. Note que o aluno emocional irá atrasar novamente duas vezes mais rápido que o agente não emocional, pois ele está incrementando a lista em 1 a cada pergunta. Em outras palavras, enquanto o agente não emocional demora mais ou menos 20 turnos para atrasar novamente a resposta, o emocional demora só 10, pois irá atrasar mais rápido. Note que o ganho em nota do agente emocional quando ele faz a inclusão de um elemento em um turno é no máximo 100/5050 ~=0,0198, quando ele inclui o elemento mais significativo 100 na soma (entretanto sabemos que ele nunca chegará a um valor tão alto, pois o valor 100 é o que tem mais prioridade e que nunca sai da lista.

Supondo que o agente emocional obtém o lucro máximo pelo aumento de *incValue*, ele terá após 10 turnos um lucro máximo de 0,0198*10 = 0,198. Sendo que ele terá o dobro de punição (pois errará duas vezes mais) em relação ao não emocional. No nosso caso, em um turno ele ganha, no melhor caso, 0,198 -1 (lucro com o incremento da lista menos a punição). Logo, ele terá prejuízo. Isto ocorrerá nas próximas falhas, fazendo o agente acumular, no melhor caso, 0,912 a menos de nota, em relação ao não emocional. O problema se agrava quando se aumenta *incValue*, pois como o agente irá aumentar mais rápido o seu valor de *L* após um atraso, ele irá também chegar antes a um valor de *L* igual ao valor de *wait* que o professor aguarda, gerando mais atrasos com este comportamento, piorando a relação de custo x benefício com o incremento de *L*.

decValue	*média totalNotas*	*decValue*	*média totalNotas*	*decValue*	*média totalNotas*	*decValue*	*média totalNotas*
0	-1,00	25	51,20	50	41,17	75	41,83
1	-1,00	26	51,17	51	42,38	76	40,88
2	-1,00	27	50,40	52	43,14	77	40,15
3	-0,97	28	49,72	53	43,22	78	39,20
4	29,49	29	49,68	54	45,36	79	37,97
5	37,79	30	47,98	55	45,07	80	37,36
6	41,79	31	46,57	56	45,28	81	36,17
7	45,80	32	44,18	57	43,79	82	35,03
8	48,38	33	44,11	58	44,74	83	33,37
9	50,87	34	43,97	59	46,08	84	32,76
10	52,86	35	44,41	60	46,06	85	31,32
11	52,08	36	44,62	61	46,05	86	30,16
12	53,32	37	46,89	62	46,26	87	28,93
13	53,23	38	46,26	63	47,32	88	27,45
14	53,44	39	46,86	64	46,84	89	25,95
15	53,30	40	46,84	65	45,14	90	24,21
16	52,48	41	46,42	66	45,93	91	22,87
17	52,81	42	45,31	67	45,50	92	21,43
18	52,80	43	44,82	68	45,03	93	19,51
19	52,31	44	43,52	69	43,91	94	17,99
20	52,62	45	44,04	70	44,53	95	16,05
21	53,33	46	41,90	71	43,68	96	14,50
22	53,17	47	40,85	72	43,62	97	12,71
23	52,24	48	42,24	73	43,09	98	11,86
24	50,88	49	41,05	74	42,62	99	9,87

Tabela 6: Notas por valor de *decValue*

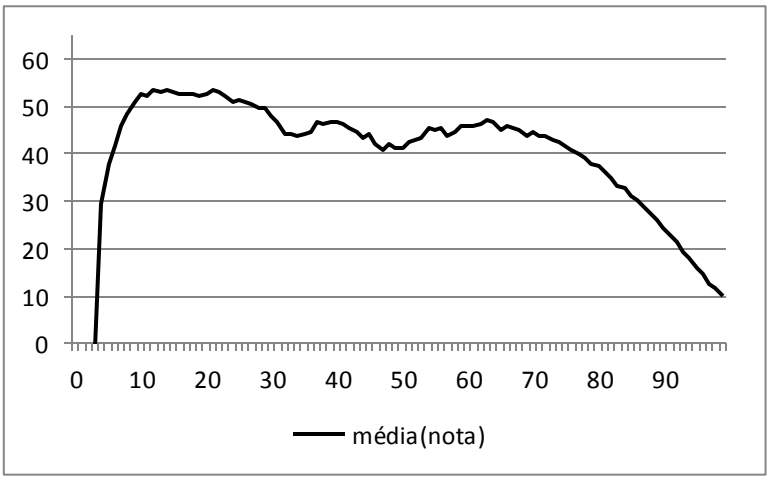


Figura 30: Média de Nota por *decValue*

decEmoValue	*média totalNotas*	*decEmoValue*	*média totalNotas*
0	52,03	8	5,20
1	46,43	9	1,30
2	38,80	10	-0,54
3	30,94	11	-0,40
4	24,84	12	-0,29
5	19,29	13	-0,30
6	14,15	14	-0,58
7	9,17	15	-0,10

Tabela 7: Notas por valor de *decEmoValue*

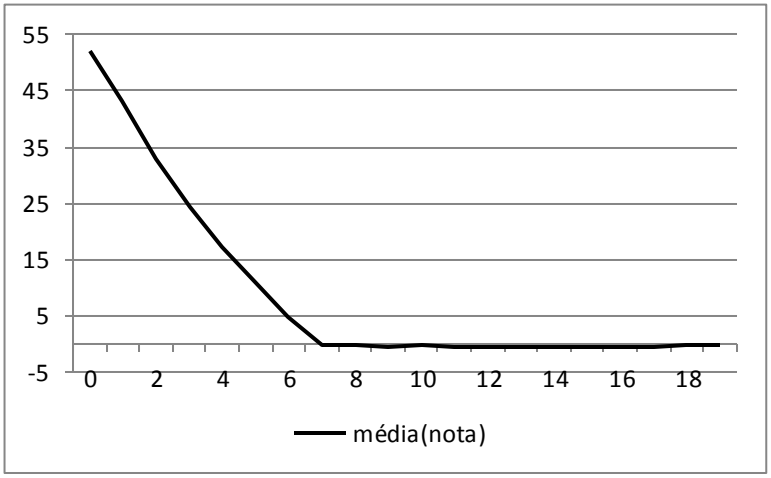


Figura 31: Nota por valor de *decEmoValue*

6.3 Diversão: Influência sobre o módulo de desejos

Exemplificarmos esta influência sobre o módulo de desejo através de um problema simples de contingência e satisfação de desejos, que chamaremos de Problema Diversão. Consiste no seguinte cenário: todos os dias o agente possui uma vontade de se divertir e alcança este desejo indo ao parque público ou indo ao cinema. Para representar este fenômeno, o agente possui um *drive* chamado **Diversão** que aumenta 1 ponto[12] a cada dia, representando o seu desejo de se divertir. O agente pode satisfazer o seu desejo de se divertir (i.e., diminuir o seu *drive* **Diversão**) alcançando uma entre duas metas: (i) ir ao parque público (**MParque**) ou (ii) ir ao cinema (**MCinema**). Entretanto, quando o agente vai ao **MParque**, ele satisfaz 1 ponto da **Diversão**, porém sua ida ao **MCinema** satisfaz somente um valor *satCinema* da *Diversão* (que reside entre [0, 1] e é especificado no experimento). Sendo assim, todas as vezes que o agente vai ao parque ele satisfaz o seu desejo de se divertir

[12] Não entraremos em detalhes sobre o conceito *drive* implementado na arquitetura para simplificar nossa discussão. Para maiores detalhes veja os tópicos 4.4, 3.1.3 e 3.8 desta dissertação.

completamente, i.e, o seu *drive* **Diversão** decai de 1 ponto. Já quando ele vai ao cinema ele satisfaz o seu desejo parcialmente, i.e., o seu *drive* **Diversão** decai somente *satCinema*.

Porém, quando o agente vai ao parque ele pode ser assaltado, o que não ocorre quando ele vai ao cinema. Quando ocorre um assalto, o *drive* **Diversão** do agente não é decrementado e ele ainda recebe o custo de 1 ponto (representando as perdas ocorridas no assalto, como por exemplo, a perda de dinheiro). Durante o processo de **MParque**, existe a chance *assaultTax* do agente sofrer um assalto, valor especificado no momento do experimento. Note que nesta configuração em **MCinema** nunca ocorrerá um assalto e sempre satisfará o *drive* **Diversão** com *satCinema* pontos.

Os objetivos dos agentes no experimento são: (i) minimizar a intensidade registrada no *drive* **Diversão** e (ii) minimizar o **Custo**. Isto é, ele deseja satisfazer completamente sua **Diversão** evitando ao máximo suas perdas com o assalto, representadas pelo **Custo.** Consideremos neste experimento que os agentes desejam minimizar os dois valores com igual importância. Isto nos permite definir uma métrica de desempenho que chamaremos de **Contra-Desempenho** igual a **Custo** + **Diversão**. Esta métrica representa igualmente a importância das duas medidas para o agente e minimizá-la indica que o agente está otimizando igualmente as duas medidas.

Observe que um agente que leve em conta somente a satisfação do *drive* **diversão** irá preferir sempre ir ao parque. Contudo, como há assaltos, isto pode não ser vantajoso quando esta taxa de assalto (*assaultTax*) é muito alta e quando a satisfação de ir ao cinema (*satCinema*) não é tão baixa.

6.3.1 Agente Emocional

Para tentar lidar com esta situação, iremos definir um agente emocional simples baseado na emoção medo. O estado emocional é constituído pela emoção *medo*(x, i), onde x indica uma meta e i a intensidade do medo em relação a esta meta que reside entre $[0, \infty]$. Esta emoção indica que "O agente tem *medo* de tentar realizar a meta x com intensidade i".

De acordo com este estado emocional, podemos definir a seguinte regra de eliciação e decremento: Quando este agente for assaltado na realização da meta **MParque**, ele irá incrementar seu *medo* em relação a esta meta em 1 ponto. A cada dia o agente irá decrementar este medo usando um fator *decMedo* que deve ser especificado no momento da simulação.

A emoção influencia a seleção de metas da seguinte forma: O agente irá preferir gerar a submeta que tenha menor valor de medo. Em caso de empate, ele irá selecionar a meta que lhe traga maior decréscimo do *drive* Diversão. No nosso caso, um agente preferirá **MParque**

sempre que não tiver medo dele (devido a um assalto), e preferirá **MCinema** caso contrário, já que a intensidade do medo em relação a meta **MCinema** é sempre 0.

Note que existem três variáveis não definidas no problema: a satisfação de ir ao cinema (*satCinema*), o decréscimo do medo (*decFear*) e a taxa de assalto (*assaultTax*). Chamaremos estes valores de **configuração do experimento**.

No caso de *satCinema*, caso este valor seja 1, ir ao cinema é tão vantajoso quanto ir ao parque, então um comportamento racional seria ir só ao cinema, já que ir ao parque pode causar prejuízo. O inverso, *satCinema* igual a 0, ir ao cinema não trás nenhuma satisfação o que tornar ir ao cinema desvantajoso. Neste caso é sempre melhor ir ao parque, mesmo que haja alta chance de assalto, pois existe uma chance real de satisfazer o drive **Diversão**.

Quanto a variável *assaultTax*, quando ela é mínima (0), a chance de ser assaltado no parque é zero, então para o agente é melhor ir ao local onde haja maior satisfação, que é ir ao parque (a não ser que *satCinema* seja 1, o que torna ir ao cinema tão bom quanto ir ao parque). Por outro lado, *assaultTax* igual a 1 faz com que o agente sempre seja assaltado ao ir ao parque. Logo ele não terá satisfação alguma realizar esta submeta. O que torna mais vantajoso ir ao cinema, visto que existirá pelo menos algum decréscimo do *drive* **Diversão** (a não ser que *satCinema* seja igual a 0, neste caso tanto faz ir ao parque ou ao cinema, visto que ir ao parque nunca satisfaz o *drive* devido aos assaltos e ir ao cinema nunca satisfaz o *drive* devido a *satCinema* ser zero).

Por fim, *decMedo* irá influenciar a "memória" do agente quanto aos assaltos que ele sofrer. Quando o agente é assaltado, sua emoção "medo de ir ao parque" aumenta em 1 ponto. Isto o fará preferir ir ao cinema enquanto o medo não passar (i.e., voltar a intensidade zero). O valor *decMedo* decrementará um dado valor a cada dia que passar. Logo, quanto mais alto for *decMedo*, mais rápido o agente deixará de ter medo e mais rápido ele voltará a ir ao parque. Neste contexto, se *decMedo* for igual a 1, ele nunca irá ao cinema, pois logo após dele sofrer um assalto, o agente irá revisar o medo e decrementá-lo de 1, o que o fará não ter mais medo logo no dia seguinte. Este também é o caso de um agente que não se importa com os assaltos que sofre (i.e., que não tem medo) e que sempre prefere ir ao parque. O contrário, *decMedo* igual a 0, fará o agente nunca ir ao parque após o primeiro assalto, pois ele nunca deixará de ter medo de ir ao parque, visto que o valor de decremento é nulo.

6.3.2 Mapeamento e Implementação

A eliciação causada por um assalto irá ocorrer na *First Emotion Review Function* e na sub-função *First Emotional Elicitors*, pois ela exige processar a percepção do assalto. Conforme o mapeamento, esta regra foi implementada na *Belief Update Function* do Jason.

Já a regra de decremento emocional do medo tem caráter cognitivo, visto que representa o agente lidando com o trauma do assalto. Consideramos mais coerente colocá-la na *Second Emotion Review Function* (na sub-função *Emotional Update*) visto que esta é responsável pelo caráter cognitivo da emoção.

O desejo de se divertir foi implementado como um *drive* que tem limiar inferior -1[13]e superior +1. Possui valor inicial igual a 0 e ele possui valor de incremento igual a 1. Sendo que este valor é incrementado a cada dia (informação dada pelo ambiente) e não a cada ciclo. Por fim, quando o *drive* extrapolar o limiar superior, ele se tornará ativo e o agente tentará criar uma intenção para trata-lo. Os *drives* são implementados de forma híbrida, como proposto no mapeamento, sub-tópico 5.2.5.

Por fim, quando o agente deseja se divertir, ele irá selecionar entre a sub-meta **MParque** e **MCinema** através da avaliação da intensidade da emoção *medo* em um plano executado para o *drive* **Diversão**. A inserção de uma submeta por decisão tomada sobre influência de uma emoção é implementada na função *Option Function* e na sub-função *Update Desires* da UEBDI. E isto foi mapeado nos planos do Jason, conforme discutido no mapeamento.

6.3.3 Experimento

Nosso objetivo neste experimento é ressaltar quais são as melhores configurações de agentes emocionais em cada uma das situações analisadas.

Serão executadas 30 instâncias do problema para dada configuração. Em cada instância passam-se 1000 dias e em cada dia o agente decide qual submeta realizar: ir ao parque ou ir ao cinema. No final dos 1000 dias obtemos os valores de **Custo** e **Diversão** finais para se calcular o **Contra-Desempenho** do agente. Após isto, é feita a média dos 30 valores de **Contra-Desempenho** obtidos das configurações. Estas médias serão usadas para comparação dos resultados de cada configuração.

[13] Observe que neste exemplo só é importante o limiar superior, pois não é tratado o estado do *drive* quando extrapola o limiar inferior.

Este experimento foi realizado em um computador com processador Core i5-2450M com 6Gb de memória e Windows 7 64 bits. Rodando na JDK 1.7.

As configurações adotadas são: a taxa de assalto *assaultTax* teve os valores {0.1; 0.3; 0.5}. A satisfação de ir ao cinema *satCinema* esteve entre os valores {0.0; 0.2; 0.4; 0.6; 0.8; 1.0}. Já o valor de decréscimo do medo *decMedo* variou no mesmo conjunto de *satCinema*: {0.0; 0.2; 0.4; 0.6; 0.8; 1.0}.

Os resultados serão agrupados por *assaultTax* em três tabelas, uma para cada valor usado desta variável. As linhas destas tabelas indicarão um dado valor de *satCinema* e as colunas um dado valor de *decMedo*. De tal forma que uma célula de uma tabela indica o valor de média de **Contra-Desempenho** obtido para o valor de *satCinema* de sua linha e o valor de *decMedo* de sua coluna. Para facilitar nossa análise, as células da tabela foram pintadas com tons de cinza, onde os mais claros indicam os valores menores de **Contra-Desempenho** e valores mais escuros indicam valores maiores de **Contra-Desempenho** em relação às outras células na mesma linha (quanto menor o valor melhor desempenho do agente). Os valores marcados em negrito+itálico são os melhores valores por linha.

assaultTax 0.1		*decMedo*					
		0,0	0,2	0,4	0,6	0,8	1,0
satCinema	0,0	994,77	462,30	327,87	269,33	276,87	***206,93***
	0,2	795,95	401,85	291,21	253,13	254,72	***197,55***
	0,4	595,16	328,81	260,01	238,35	239,92	***202,55***
	0,6	397,20	265,55	239,11	222,65	224,67	***197,56***
	0,8	199,70	202,58	200,67	199,82	199,49	***199,21***
	1,0	***2,00***	134,00	166,80	185,87	183,07	197,80

Tabela 8: Contra-Desempenho para *assaultTax=0.1*

assaultTax 0.3		*decMedo*					
		0,0	0,2	0,4	0,6	0,8	1,0
satCinema	0,0	998,47	841,00	744,37	684,30	690,07	***615,20***
	0,2	799,41	720,68	672,83	650,61	642,49	***608,16***
	0,4	***599,66***	603,99	601,06	606,89	601,27	599,86
	0,6	***400,53***	480,51	519,75	547,76	560,57	582,17
	0,8	***201,37***	359,08	453,01	511,91	507,66	607,33
	1,0	***2,00***	240,27	370,00	461,73	463,73	602,13

Tabela 9: Contra-Desempenho para *assaultTax*=0.3

assaultTax 0.5		*decMedo*					
		0,0	0,2	0,4	0,6	0,8	1,0
satCinema	0,0	1000,27	1004,27	1005,27	1009,00	1003,97	***996,17***
	0,2	***800,27***	854,76	898,37	936,37	930,41	998,33
	0,4	***600,60***	712,23	797,94	867,23	873,21	1000,81
	0,6	***401,29***	570,68	695,29	795,60	796,44	1000,36
	0,8	***201,66***	424,67	596,93	729,20	731,43	1006,66
	1,0	***2,00***	286,20	500,07	663,67	665,73	1005,60

Tabela 10: Contra-Desempenho para *assaultTax=0.5*

Vamos observar nestas tabelas como a taxa de assalto influencia o desempenho de um dado par de *decEmotion* e *satCinema*.

Observa-se na Tabela 8 que apenas quando a satisfação do cinema (*satCinema*) é igual a 1 a submeta **MCinema** se torna a melhor opção**,** visto que o melhor **Contra-Desempenho** nesta linha é quando o valor de *decMedo* é zero, forçando o agente a nunca perder o medo e forçando-o a ir sempre ao cinema após um assalto no parque. Ele obtém melhor desempenho nesta situação (**Contra-Desempenho** igual a 2), pois, como já prevemos, o cinema causa mesma satisfação que o parque e, então, é melhor visitá-lo para satisfazer o *drive* **Diversão** sem ter custo adicional pelos assaltos. Nas outras linhas para valores menores de *satCinema* é melhor seguir a submeta **MParque**, pois a taxa de assalto é baixa e a satisfação de ir ao cinema não compensa a troca.

Já na Tabela 9 podemos observar que o desempenho do agente é bastante dependente de *satCinema*. Se *satCinema* for baixo (i.e., <0,4), mesmo ocorrendo algum assalto quando o agente vai ao parque, ainda vale mais a pena ir ao parque do que ir ao cinema. Neste caso, quanto menor for o medo do agente em ser assaltado (i.e., quanto maior for *decMedo*) melhor. Porém, se satCinema >= 0,4, o agente passa a preferir ir ao cinema pois a sua satisfação indo ao parque considerando a possibilidade de ser assaltado não compensa. Neste caso, quanto maior for o medo do agente em ser assaltado (i.e., quanto menor for *decMedo*), maior será o seu ganho final. Como a taxa de assalto é mais alta neste caso, estas configurações passam a ter desempenho melhor justamente porque evita os assaltos e conseguem alguma satisfação indo ao cinema.

Por fim, na última tabela, onde a taxa de assalto é bastante alta (0,5), é melhor ter menor decréscimo emocional (*decMedo*) para praticamente todos os valores de satisfação ao ir ao cinema (*satCinema*). Este não é o caso onde a satisfação é igual a 0, onde, notoriamente, não adianta ir ao cinema, pois ele não satisfaz o *drive* **Diversão**, então é melhor ir ao Parque, onde pelo menos se irá obter alguma satisfação quando não assaltado.

Note que o resultado **Contra-Desempenho** igual a 2 repete-se na configuração *decEmotion* = 0 e *satCinema*=1 nas três tabelas. Isto se deve ao mesmo motivo: nesta configuração, não importa a taxa de assalto (basta que seja maior que 0), quando o agente sofrer o primeiro assalto, ele terá um valor de insatisfação de **Diversão** acumulado pelo assalto e um valor de **Custo** acumulado, e nunca mais tentará a meta **MParque**, totalizando o valor 2 encontrado nesta configuração.

Com isto, podemos observar uma tendência que quanto maior for a taxa de assalto, melhor se sairá um agente cuja emoção medo demore mais tempo a baixar sua intensidade. Para *assaultTax* igual a 0.3, agentes emocionais com menor valor de *decMedo* se tornam melhores que agentes com maior valor de *decMedo* quando a satisfação de ir ao cinema é maior que 0.4. E quando o valor de *assaultTax* for igual a 0.5, um agente emocional com valores menores de *decMedo* se tornam melhores do que os com valores maiores em todas as configurações, exceto quando a satisfação do cinema é zero.

6.4 IPD: Influência sobre o módulo de intenções

Iremos demonstrar a influência sobre o módulo de intenções através da seleção de um determinado plano entre vários que tentam atingir uma mesma meta. Usaremos para isto um problema clássico de teoria dos jogos e sistemas multi-agentes chamado Dilema do Prisioneiro Iterado – IPD (do inglês *Iterable Prisioner Dilemma*).

Informalmente este dilema descreve um cenário onde duas pessoas são presas por um crime e estão sofrendo um interrogatório. Elas só possuem duas ações possíveis: confessar ou não um crime que cometeram juntas. Se um prisioneiro confessar e o outro não, o réu confesso é libertado e o outro cumprirá pena total de 3 anos. Se ambos confessarem, ficarão detidos por 2 anos. Por fim, se nenhum confessar, ambos ficarão presos apenas 1 ano. Após a tomada de decisão dos dois agentes, eles são informados de qual foi a pena lhe atribuída (isto permite detectar qual foi a resposta da outra pessoa, cruzando o valor da pena com a sua própria resposta). Por se iterado, o interrogatório é repetido e os prisioneiros devem tomar a decisão novamente, durante um número finito de vezes.

Este cenário possui três características principais: (i) descreve uma situação de cooperação/competição, onde o ato de confessar o crime indica que o agente competiu [14] com o outro e o ato de não confessar indica que o agente cooperou com o outro; (ii) o cenário é

[14] Consideraremos que "competir" equivale a "não cooperar". Achamos este termo mais adequado a ser usado aqui. Embora não seja usado nos trabalhos de (Wooldridge 2009) e (Neto 2010).

modelado de tal forma que um agente "racional"[15] deveria sempre não cooperar (confessar), visto que o pior caso obtido com esta escolha é ficar preso por no máximo 2 anos, enquanto que no caso de cooperar (não confessar) poderá fazê-lo ficar detido mais tempo (3 anos) (Wooldridge 2009 p. 237). Logo, é pior cooperar/confessar do que competir/não confessar; e por fim (iii) a ação de cooperar embora não seja ótima para o indivíduo, é ótima quando se deseja minimizar a pena de ambos os participantes.

Embora simplório este dilema representa dilemas reais e importantes na vida real, como por exemplo, tratados de não construção de armas de destruição em massa. Neste exemplo, não construir armas (cooperar) implica em estar vulnerável a uma possível não cooperação da outra parte (Wooldridge 2009 p. 237). Entretanto, não cooperar indica uma situação perigosa em termos coletivos, já que traz maiores perdas para todas as nações em caso de guerra.

Outro exemplo, mais corriqueiro, é a negociação entre duas pessoas por uma troca. Uma primeira pessoa quer trocar uma mercadoria por dinheiro com uma segunda pessoa. E esta troca é realizada de tal forma que tanto a mercadoria quanto o dinheiro estarão ocultos por um envelope e só é possível saber se o outro colocou o objeto desejado (mercadoria/dinheiro) no envelope após a negociação. Nesta situação, uma pessoa pode cooperar (colocar o dinheiro/mercadoria no envelope) concretizando a negociação ou pode competir (deixar o envelope vazio), traindo a outra pessoa (Colman 2003 p. 146).

A aplicação deste dilema a agentes emocionais ocorreu no trabalho (Neto 2010) para demonstrar as vantagens da arquitetura proposta no desenvolvimento de agentes emocionais, pois, segundo os autores, este dilema permite avaliar a influência de um módulo afetivo no desempenho do comportamento de agentes. Por isto, adotaremos o mesmo problema usando uma proposta similar a dele para validar a influência sobre o módulo de intenções. Como será visto, a principal diferença é que usamos um modelo emocional mais simples (que chamamos de modelo emocional de Jiang (Jiang and Vidal 2006)) do que o proposto pelos autores em (Neto 2010) (baseado no modelo afetivo de (Mehrabian 1996)). Nosso foco também será diferente: os autores de (Neto 2010) se preocuparam unicamente em analisar a capacidade do modelo emocional de alterar o comportamento do agente. Nós, por outro lado, iremos discutir mais afundo o problema IPD e demonstrar que, conforme discutido em (Wooldridge 2009 pp. 238–240), o comportamento unicamente competitivo é racional quando apenas consideramos

[15] Entendemos que "racional" neste trecho significa "racional quanto a métrica de obter menos anos de cadeia individualmente".

nossa medida de desempenho a diminuição da pena dos agentes. Quando passamos a considerar também uma medida de cooperação, premiando os agentes que cooperam, pois isto é esperado para gerar menos danos aos agentes semelhantes, a estratégia racional passa ser a que avalia o comportamento do adversário e tenta ser cooperativo quando o outro é e competitivo caso contrário. Sendo esta estratégia chamada de TFT (*Tit-ForTat* – Olho por olho, dente por dente). Além disto, veremos que o modelo emocional proposto é uma forma generalizada deste algoritmo ótimo.

Já o trabalho (Bazzan and Bordini 2001) utiliza agentes emocionais com modelo emocional baseado no OCC para aumentar a taxa de agentes cooperadores no IPD. Entretanto este trabalho considera que os agentes estão em um mundo 2D, como uma matriz de vizinhanças, e que eles se multiplicam de acordo com a sua pontuação em relação aos vizinhos. Além disto, seu foco foi demonstrar que o uso de agentes emocionais aumenta o número de agentes cooperativos nestes cenários, comparado com outros agentes puramente cooperativos ou competitivos. Isto é diferente do cenário adotado, não nos permitindo uma comparação direta.

Seguiremos abaixo na definição do problema e na modelagem do cenário que será experimentado.

6.4.1 Definição

Neste trabalho usaremos como referência a proposta do IPD contida em (Neto 2010) e (Wooldridge 2009) e a implementação já existente no Jason do problema. Este dilema é representado por um sistema multi-agente composto por **n** agentes prisioneiros e por um agente **árbitro**. Durante **m** iterações (rodadas), o agente **árbitro** irá selecionar dois agentes quaisquer e irá realizar um interrogatório. Primeiro o **árbitro** irá perguntar a ambos se cooperam (não confessam) ou não cooperam (confessam) com o outro agente. Os agentes responderão ao **árbitro** que irá avaliar a resposta e comunicar a pontuação aos agentes (de acordo com a Tabela 11). Em seguida o ciclo é reiniciado. Quanto a esta nota obtida a cada rodada chamaremos de **Pontos Individuais da Rodada (PIR)**.

Existem quatro possíveis conjuntos de ações em uma rodada: (i) os dois agentes cooperam e não confessam, (ii) o agente A não coopera com o agente B confessando o crime e o agente B coopera com o agente o A não confessando, (iii) o agente B não coopera com o agente A confessando e o agente A coopera com agente B não confessando, e (iv) os dois agente não cooperam e confessam o crime. De acordo com as respostas obtidas, os agentes são pontuados pelo árbitro seguindo a Tabela 11.

PIR		Agente A	
		Compete	**Coopera**
Agente B	**Compete**	A recebe 2 pontos B recebe 2 pontos	A recebe 0 pontos B recebe 5 pontos
	Coopera	A recebe 5 pontos B recebe 0 pontos	A recebe 3 pontos B recebe 3 pontos

Tabela 11: Matriz de Pontos Individuais da Rodada

Veja que ao final do interrogatório o agente terá acesso a duas informações: (i) quem foi seu oponente e (ii) qual foi sua punição. Pela pontuação e pela resposta obtida é possível saber qual foi a resposta dada pelo oponente traçando estas informações com a Tabela 11.

Ao final dos **n** interrogatórios, os agentes terão uma pontuação total equivalente a soma dos PIR obtido em cada interrogatório que chamaremos de **Pontos Individuais** (**PI**). Observe que, de acordo com estas definições, um agente terá desempenho melhor ou igual a qualquer outro agente se ele sempre não cooperar (confessar), seguindo exatamente o que prevê a teoria dos jogos, (a verificação disto é simples e exposta em (Wooldridge 2009 pp. 236–240) e (Colman 2003)). Contudo, esta situação é ótima se apenas quisemos obter agentes individualmente racionais, pois não privilegia a cooperação entre os agentes, minimizando a pena de todos.

Como comentado, iremos definir outra métrica para poder medir uma "taxa de cooperação" dentro do sistema multi-agente definido. A cada rodada/interrogatório será gerada uma segunda pontuação data pela Tabela 12 que reflete o quão cooperativos os agentes foram em uma rodada[16]. Chamaremos esta pontuação de **Pontos de Cooperação da Rodada (PCR)**. Ao final das **n** rodadas, podemos obter uma medida de cooperação do experimento igual ao somatório de PCR, que chamaremos de **Pontos de Cooperação (PC)**. Veja que esta pontuação não é dada a um agente individualmente, mas a todo o sistema de acordo com as ações tomadas pelos agentes. Se ambos cooperarem, o sistema ganhará 2 pontos. Se apenas um cooperar, ganhará 1 ponto. Se nenhum cooperar, ganhará 0 pontos.

PCR		Agente A	
		Compete	**Coopera**
Agente B	**Compete**	Sistema recebe 0 pontos	Sistema recebe 1 ponto
	Coopera	Sistema recebe 1 ponto	Sistema recebe 2 pontos

Tabela 12: Matriz de Pontos de Cooperação da Rodada

[16] A tabela foi definida de tal forma que o valor de pontos de um interrogatório seja igual a soma de respostas cooperativas obtidas.

6.4.1.1 Métrica de desempenho

Com nossas medidas definidas iremos propor uma métrica de desempenho que nos indique quanto um agente foi apto em maximizar o nível de cooperação do sistema (**PC**), mas sem perder pontos quanto ao seu desempenho individual (**PI**). Chamaremos esta métrica de **Pontos Gerais** – **PG**, sendo proporcional a multiplicação de PI por PC (Eq. 1), de tal forma que esta pontuação **PG** seja máxima quando PI e PG forem máximas e decaia quando alguma delas for menor que a outra ou as duas forem baixas.

$$PG \approx PI \, x \, PC \qquad \text{(Eq. 1)}$$

6.4.2 Estratégias

Na literatura consultada, já existem vários algoritmos definidos para o comportamento de um agente no problema IPD. Na teoria dos jogos, chama-se estes algoritmos de "estratégias". Selecionamos cinco estratégias devido a sua importância dentro do problema disponíveis nos diferentes trabalhos consultados que adotaram este problema (Bazzan and Bordini 2001, Neto 2010, Wooldridge 2009). A Tabela 13 descreve as estratégias adotadas no nosso experimento e a motivação para selecioná-las. Cada estratégia em nossos agentes reflete a um plano que tenta atingir a mesma (sub)meta que é responder ao Árbitro.

Agente/Estratégia	Sigla	Estratégia	Motivação
Olho por Olho	TFT	Repetir a última ação o adversário	Melhor estratégia no experimento empírico original
Pavlov	PAVLOV	Se PIR >= 3, repetir última jogada. Senão trocá-la	Algoritmo guloso que tenta otimizar o desempenho do agente
Nunca cooperar	ALLD	Sempre não coopera	Maximiza PG / Melhor estratégia
Sempre cooperar	ALLC	Sempre coopera	Minimiza PG / Pior estratégia
Aleatório	RANDOM	Acusa ou não acusa randomicamente	Utilizado para simular uma situação não determinística

Tabela 13: Agentes/estratégias adotados

6.4.3 Agente Emocional Jiang-IPD

Desenvolvemos um agente emocional utilizando uma versão adaptada do modelo proposto por Jiang. A principal motivação para escolha deste modelo é que ele é o modelo mais simples que lida com avaliação de outras entidades adotado na literatura consultada. A forma como ele será adaptado fará o agente emocional cooperar quando o outro agente cooperar e a não cooperar quando o outro agente também não cooperar.

O modelo de Jiang representa o estado emocional como um conjunto de tuplas da forma **emotion**(*nome*, *alvo*), onde *nome* é uma das emoções contidas em {*Love, Happy, Like, Dislike, Unhappy, Hate* } e *alvo* é o agente referenciado pelo qual o agente emocional "sente" a emoção. Por exemplo, se o agente **A** possui a tupla **emotion**(*hate*, *B*), isto indica que o agente A sente *hate* em relação ao agente B.

Inicialmente, todo agente sente *like* pelos outros (**emotion**(*like*, _)). Como regra de eliciação usaremos o resultado dado pelo agente árbitro. Dados os agentes Árbitro, A e B, se o Árbitro comunicar a A que B cooperou (não confessou o crime), A aumentará a emoção que sente por B. Por exemplo, irá de **emotion**(*like*, *B*) para **emotion**(*happy*, *B*)). Caso o contrário, B não cooperou (e confessou o crime), então A decrementará a emoção que sente por B (por exemplo: irá de **emotion**(*like*, *B*) para **emotion**(*unhappy*, *B*)).

Na influência emocional consideramos que cada emoção do modelo ativará uma estratégia, tal como descrito na Tabela 14. Lembremos que uma estratégia para nós é representada por um plano que a implementa. Por isto, nosso agente fará influencia emocional sobre a seleção de planos, que reside no módulo de intenções.

Emoção em relação ao agente X	Estratégia adotada	Explicação
Love	ALLC	O agente confia que o outro também irá cooperar
Like	ALLC	O agente confia que o outro também irá cooperar
Happy	TFT	O agente não possui confiança suficiente para cooperar cegamente com o outro, então prefere adotar a política "olho por olho"
Unhappy	TFT	O agente não possui confiança no outro agente para cooperar, então adota a política "olho por olho"
Dislike	PAVLOV	Não confia no outro agente, mas se a última política com ele foi boa (pontuação >=3), então a mantém, senão a troca
Hate	ALLD	Não confia no outro agente e irá sempre não cooperar

Tabela 14: Estratégia adotada em dado estado emocional

O mapeamento adotado para a influência emocional tenta refletir a relação de confiança que um agente desenvolveu pelo outro, de forma similar ao feito por (Neto 2010) em seu primeiro experimento (sem marcador somático). A diferença é que a dinâmica e o estado emocional do modelo de Jiang é muito mais simples que o modelo misto de (Neto 2010). O modelo de Jiang possui apenas seis emoções básicas discretas enquanto que o modelo de (Neto 2010) trabalha com um subconjunto do modelo OCC em conjunto com o modelo de humor PAD e de personalidade OCEAN.

6.4.4 Mapeamento e Implementação

Observe que a emoção em relação a um agente se altera quando o agente emocional recebe informação sobre qual foi o resultado de uma iteração. Isto é, a emoção é atualizada quando o agente trata a informação sobre o resultado de uma interação. Na implementação do problema em Jason, o agente trata esta informação como se fosse uma meta chamada *!take_records*. Por isto, para eliciarmos a emoção neste contexto, precisamos considerar que esta meta *!take_records* influencia a emoção. Vamos tratar então esta regra de eliciação na *Second Emotion Review Function*, pois é uma meta e não é um *drive* (se fosse um *drive*, ela poderia ser tratada na *First Emotion Review Function*). E será implementada na sub-função *Second Emotional Update*, visto que o que ocorrerá é uma alteração da intensidade e não propriamente uma eliciação da emoção. Esta sub-função é mapeada na função *Select Event* do Jason, conforme tratado no mapeamento (tópico 5.2.5).

A influência emocional foi considerada da seguinte forma: o agente emocional tem a meta de responder ao árbitro se irá confessar ou não o crime (competir ou cooperar) e deve selecionar entre cinco planos (um para cada uma das 5 estratégias). Para decidir isto, ele utiliza a emoção em relação ao outro agente que interage com ele neste turno, como discutido anteriormente.

Logo, esta influência trata-se de uma seleção de plano e foi mapeada na função *Filter Function* e na sub-função *Find Plans* da UEBDI. Esta última foi mapeada nas funções *Check Context* e *Unify Context* do Jason.

6.4.5 Experimento

Nosso experimento foi especificado da seguinte forma: executou-se uma instância do experimento IPD somente com dois agentes em 100 rodadas, de tal forma que **m** é igual a 2 e **n** é igual a 100. O valor 100 foi adotado devido ao seu uso já em trabalhos passados (Neto 2010, Wooldridge 2009) e o fato de que é mais do que o suficiente para verificar uma tendência dos agentes na seleção de suas estratégias.

Com o uso do número de agentes igual a 2, força-se o Árbitro a selecionar apenas os mesmos dois agentes em todas as rodadas. Com o objetivo de testar todas as estratégias, utilizamos cinco agentes implementando as cinco diferentes estratégias descritas na Tabela 7 (ALLC, ALLD, RANDOM, PAVLOV e TFT) e um agente emocional utilizando a estratégia JIANG apresentada na Tabela 14. Executamos instâncias do experimento IPD para todos os

pares possíveis destes agentes, inclusive entre agentes com a mesma estratégia, totalizando 21 execuções.

O equipamento adotado para o experimento foi um computador com processador Athon 2X 4800, com 2,8 Gb de RAM, Windows XP SP3. Usando a JVM 6 e rodando via linha de comando um pacote JAR.

A Tabela 15 mostra o resultado dos experimentos ilustrando as estratégias utilizadas por cada um dos agentes e PI e PC obtidos. Os valores de PI dos agentes que ganharam os combates estão em negrito e os empates estão em itálico.

Já a Tabela 16 exibe para cada agente qual foi a soma dos PIs em todos os experimentos (∑PI) daquele agente, a soma dos PCs em todos os experimentos daquele agente (∑PC) e o somatório dos PGs obtidos por aquele agente em seus experimentos ($\sum PG10^{-3}$). Por conveniência o valor de PG foi multiplicado por 10^{-3}, visto que o que nos interessa proporção dos PGs entre os agentes e não sua ordem de grandeza. Além disto, a tabela exibe os somatórios de PI e PC sem os experimentos realizados com RANDOM (∑PI-∑PI(RANDOM) e ∑PC-∑PC(RANDOM)), para evidenciar a diferença de desempenho dos agentes com ou sem o fator não-determinístico.

As figuras a seguir: Figura 32, Figura 33 e Figura 34 exibem os dados das tabelas Tabela 15 e Tabela 16 de forma visual.

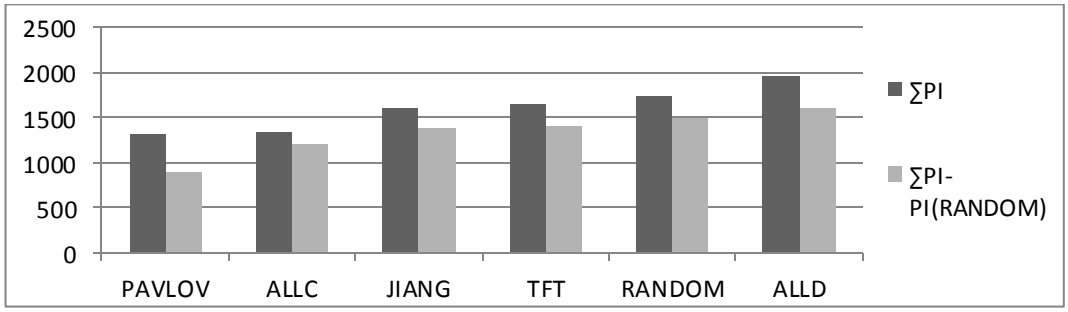


Figura 32: PI total dos agentes

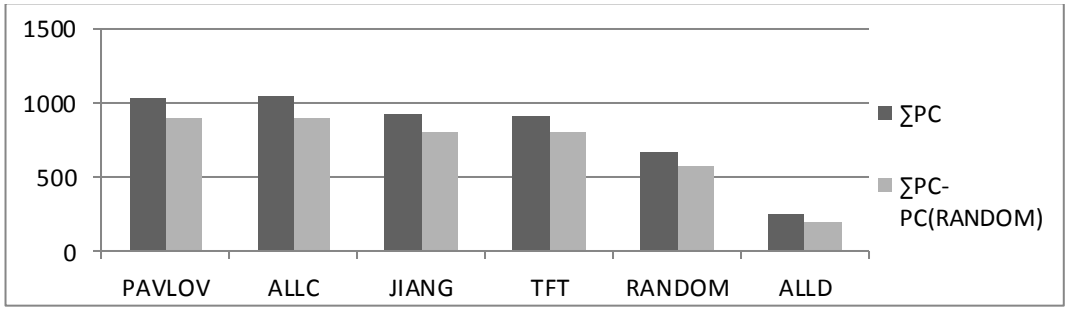


Figura 33: PC total dos agentes

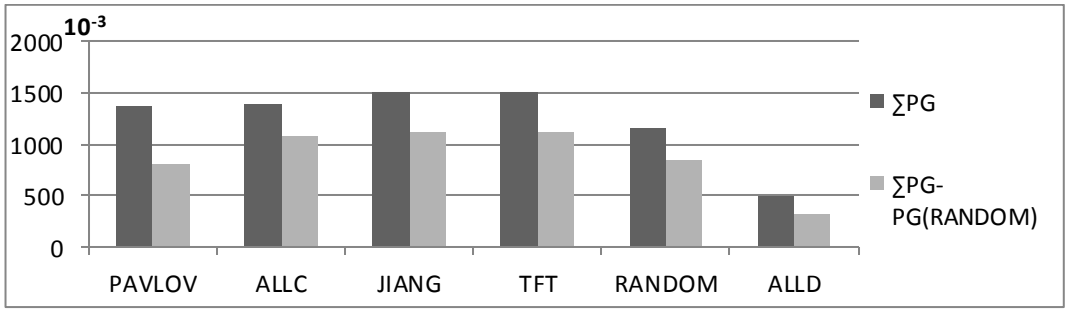


Figura 34: PG dos agentes

Agente A	Agente B	PI(A)	PI(B)	PC		Agente A	Agente B	PI(A)	PI(B)	PC
JIANG	JIANG	*300*	*300*	200		TFT	PAVLOV	*300*	*300*	200
JIANG	PAVLOV	*300*	*300*	200		ALLC	TFT	*300*	*300*	200
PAVLOV	PAVLOV	*300*	*300*	200		TFT	ALLD	198	**203**	1
JIANG	ALLC	*300*	*300*	200		TFT	TFT	*300*	*300*	200
PAVLOV	ALLC	*300*	*300*	200		JIANG	RANDOM	221	**306**	127
ALLC	ALLC	*300*	*300*	200		RANDOM	PAVLOV	114	**424**	138
ALLC	ALLD	0	**500**	100		RANDOM	ALLC	**410**	135	145
JIANG	ALLD	196	**206**	2		RANDOM	ALLD	100	**350**	50
PAVLOV	ALLD	0	**500**	100		RANDOM	TFT	252	**257**	109
ALLD	ALLD	*200*	*200*	0		RANDOM	RANDOM	241	**261**	102
JIANG	TFT	*300*	*300*	200						

Tabela 15: Resultado de cada combate do experimento IPD.A

Agente	∑PI	∑PC	$\sum PG 10^{-3}$	∑PI-PI(RANDOM)	∑PC-PC(RANDOM)	$\sum PG\text{-}PG(RANDOM)\ 10^{-3}$
PAVLOV	1314	1038	1363,93	890	900	801,00
ALLC	1335	1045	1395,07	1200	900	1080,00
JIANG	1617	929	1502,19	1396	802	1119,59
TFT	1655	910	1506,05	1398	801	1119,80
RANDOM	1733	671	1162,84	1492	569	848,95
ALLD	1959	253	495,63	1609	203	326,63

Tabela 16: Resultado geral do experimento IPD.A

6.4.5.1 Discussão

Iremos primeiro começar comparando o Ponto Individual (PI) de JIANG em relação aos outros agentes. Como esperado, vemos na Tabela 16 e a Figura 32 que ALLD obteve maior valor desta métrica, pois ele sempre "não coopera", obtendo pontuações iguais ou maiores que a do seu adversário. O agente emocional Jiang ficou com pontuação intermediária (1617) entre os adversários e próximo ao valor de TFT (1655). Vemos também que os valores de PI sem os combates de RANDOM não alteram significativamente o valor geral.

Agora olhando a medida de Pontos de Cooperação (PC) exibida na Figura 33, observamos que a situação se inverteu: ALLD é o último colocado enquanto que seu antagonista, ALLC, obteve melhor pontuação. Este resultado também é esperado por um princípio análogo ao anterior: como ALLC sempre coopera, ele sempre ganha as pontuações de cooperação maiores ou iguais ao seu adversário. O agente emocional JIANG novamente fica no meio da lista (929) e próximo de TFT (910), apenas sendo superior em 19 pontos.

Observe que quanto ao agente TFT, ambos os resultados medianos podem ser esperados, visto que ele é um agente que privilegia a cooperação caso o adversário também seja cooperativo. Isto é, se o adversário for cooperativo, ele tenta ganhar Pontos de Cooperação e manter mediano o ganho com pontos Individuais através da cooperação (não acusa). Caso o adversário não coopere, ele também não coopera, evitando a perda de Pontos Individuais (empatando nesta medida com o adversário) e perdendo Pontos de Cooperação. Isto faz com que a estratégia TFT mantenha PI e PC equilibrados, evitando que haja perda exacerbada de alguma das métricas.

Já o agente emocional JIANG, por sua vez possui princípio similar a estratégia TFT e tenta adotar políticas cooperativas caso ele goste do adversário (o adversário coopere) e adota políticas competitivas caso não goste do adversário (o adversário compita), de forma muito similar a TFT, o que no final faz com que ambos tenham resultados similares em ambas as medidas.

Com esta explicação e com a definição da métrica de Pontos Gerais, é esperado que esta métrica seja maximizada com TFT e com JIANG, visto que eles tendem a equilibrar os valores de PI e PC, o que maximiza PG. E isto é comprovado na Figura 34, onde tanto JIANG quanto TFT obtém maiores valores. Mais precisamente, JIANG obtém 1502 mil pontos e TFT 1506 mil pontos. A diferença entre eles, embora seja dos milhares, não é tanta comparando TFT com os outros agentes.

De fato, com os dados obtidos e a relação que criamos entre TFT e JIANG, podemos concluir que JIANG equivale a TFT e obtém desempenho (PG) próximo ao ótimo (no experimento). Uma forma de comprovar isto é imaginar a seguinte simplificação: Cria-se um agente JIANG com apenas duas emoções: gostar e não gostar. Ele gosta do adversário caso ele coopere e não gosta caso ele não coopere. Caso ele goste do adversário ele coopera, caso contrário não coopera. E tal dinamismo é o mesmo proposto por TFT. Em outras palavras, o

agente JIANG é equivalente ao TFT, a diferença é que como ele demora mais para "não gostar" e "gostar" de um adversário, ele demora mais a alterar seu comportamento.

6.5 TaskAlloc: Influência sobre o módulo de ações

Neste problema um cliente deve escolher o melhor servidor para executar uma tarefa. Nós consideramos "o melhor servidor" aquele agente que (i) oferece menor custo para execução da tarefa e (ii) a maior probabilidade de sucesso em sua execução. Estes parâmetros – baixo custo e alta probabilidade de sucesso – devem ser considerados quando se seleciona o melhor servidor. Este problema foi inspirado no trabalho de (De Weerdt et al. 2007) de alocação de tarefas em sistemas multi-agentes usando o protocolo *contract-net* (Wooldridge 2009 pp. 156–158). O estudo minucioso de leilões para solução do problema é encontrada em (Ramchurn et al. 2009). Não cabe comparar suas abordagens com a nossa, visto que nosso problema é uma versão muito resumida do problema atacado por eles e nossos agentes não foram desenvolvidos para gerarem uma solução ótima. Sendo meramente ilustrativos para a exemplificação do agente emocional.

Sabendo que os clientes não sabem inicialmente qual a probabilidade de sucesso dos servidores, eles devem inferir este valor usando informação passada. A Eq. 33 apresenta a função usada para selecionar o melhor servidor, onde *mfail*(*server*) é a média da probabilidade de falha de um *server*, *cost*(*server*, *t*) é o custo que o *server* pede para executar a tarefa *t* e *selected* é o servidor selecionado. Os clientes que selecionam seus servidores por esta equação serão chamados aqui de **clientes racionais**. Isto está teoricamente correto, pois, como veremos, eles conseguem maximizar seu desempenho de acordo com a informação que tem acesso.

$$selected = min\big(mfail(server).cost(server,t)\big) \qquad \text{(Eq. 33)}$$

O algoritmo dos agentes racionais pode ser aperfeiçoado se nós considerarmos que estes agentes também usam a experiência passada de seus colegas quando interagiram com um dado servidor. Nós assumimos que os colegas participaram de interações com servidores que o cliente ainda não interagiu e que devem ter participado em interações mais recentes com servidores que já interagiram com o cliente. Note que isso é importante, pois assumimos que a probabilidade de falha dos servidores pode alterar com o passar do tempo.

Desta forma, um cliente pode perguntar aos outros clientes qual a probabilidade de falha dos servidores com os quais este já interagiu. Esta informação representa a **reputação** dos servidores. Neste caso, o agente faz sua seleção usando a Eq. 34, onde *mrfail*(*server*) irá

considerar ambas opiniões do próprio cliente quando a de seus colegas sobre a probabilidade de falha do servidor *server* em questão. Assim, *mrfail*(*server*) é a media das reputações do servidor mais o valor *mfail*(*server*). Os agentes que usam a reputação dos servidores são chamados de **cliente reputação**.

$$selected = min\big(mrfail(server).cost(server,t)\big) \tag{34}$$

6.5.1 Cliente Emocional

Como modelo emocional, consideremos que um agente pode *gostar* ou *desgostar* (modelo *like*/*dislike*) de outro agente, dependendo de suas interações passadas (que equivale ao modelo JIANG apresentado no tópico 6.4.3. Devemos ressaltar que existe uma diferença entre emoções *like/dislike* e a informação de reputação: embora a dinâmica emocional tenha semântica similar ao valor de reputação (ambos qualificam o que um agente *acha* de outro agente), ela não é idêntica, tendo o agente emocional um comportamento não linear quanto a sua avaliação dos outros agentes. Além disto, a forma como o agente emocional influencia sua seleção é diferente da do agente reputação. O segundo irá ranquear simbolicamente seus servidores, o segundo tentará escolher numericamente o servidor.

Um cliente possui uma *Base Emocional* que é composta por sentenças da forma *like*(*x*,*i*), onde *x* é um servidor e *i* é a intensidade do sentimento de *gostar* (*like*) em relação a este agente. Podemos classificar a relação entre o cliente e dado servidor *x* em três classes: Se *i* estiver entre [7, 10], o cliente *likes* (*gosta*) do servidor *x*. Se *i* estiver entre [4, 6], então o cliente é *neutral* (*indiferente*) em relação ao servidor *x*. E se *i* estiver entre [0, 3] o cliente *dislike* (*desgosta*) do servidor *x*.

Quando o cliente recebe a informação sobre o sucesso ou falha na execução de uma tarefa pelo servidor *x*, ele muda a intensidade de *i* na sentença *like*(*x*, *i*) relacionada a *x*. A *função de mudança* é dada pela Tabela 17, onde a coluna *like*, *neutral* e *dislike* representa a classificação prévia para a sentença *like*(*x*, *i*), as linhas *success* e *fail* indicam o sucesso ou a falha na respostada pelo servidor *x* e as células indicam o valor de incremento ou decremento usado para alterar a intensidade *i*. Como mostrado pelos incrementos e decrementos, a *função de mudança* prioriza as emoções do agente e não a informação sobre o sucesso e a falha do servidor.

	if client … server		
	like	**neutral**	**dislike**
success	+2	+1	+1
fail	-1	-1	-2

Tabela 17: Incremento/Decremento da *função de mudança*

O cliente emocional é definido sobre o agente reputação usando o modelo emocional *like/dislike*. Embora o cliente emocional também use a Eq. 34 para selecionar o melhor servidor assim como faz o cliente reputação, ele irá preferir primeiro os servidores que *goste*, ignorando os outros no momento de aplicar a Eq. 34. Se não existirem servidores pelos quais *goste*, ele irá preferir servidores pelos quais sinta *indiferente*. Caso este tipo também não exista em sua base de emoções, ele irá selecionar um servidor entre os quais *desgoste*.

6.5.2 Mapeamento e Implementação

A dinâmica emocional ocorre da seguinte forma: o agente emocional irá esperar a resposta do servidor. Quando isto ocorrer, a informação de sucesso e falha irá se tornar um evento de inclusão de crenças, que é considerada uma meta. O Jason trata as mensagens como eventos / crenças, o que nos impede de interceptar esta mensagem na *Belief Update Function* do Jason e logo na nossa *Perception Function* que é implementada neste local.

Por isto consideramos que uma emoção é atualizada pela influência da base de crenças. Neste caso, temos de mapear a *função de mudança* na *Second Emotion Review Function*, que tem acesso a base de crenças, e na sub-função *Second Emotional Update* que está a cargo de atualizar o estado das emoções.

Quanto a influência emocional, nota-se que a emoção influencia na hora de selecionar o grupo de servidores que serão avaliados pela função da Eq. 34 para se selecionar o servidor escolhido. Note que o estado emocional irá parametrizar uma função de ordenação dos servidores, que é um algoritmo bem determinado e que pode ser implementado dentro de um plano (ou uma ação interna). Logo, esta influência é mapeada na *Execute Function*, que é responsável por executar os planos, mais precisamente na *Archieve Plans*, e no próprio plano.

No Jason, isto é implementado na seguinte forma: A execução do plano em si está na função *Execute Intention*, e o algoritmo de ordenação foi implementado em uma ação interna do Jason que é chamada pelo plano, conforme discutido no mapeamento.

6.5.3 Experimento

Agora iremos realizar um experimento para comparar o comportamento de três tipos de clientes: racional, reputação e emocional. Nós iremos definir uma medida chamada *score*, dada pela Eq. 35, que retorna tão ruim é um servidor com respeito a sua probabilidade de falha – *pfail*(*server*) – e seu custo. Desta forma, o servidor com menor *score* é o *melhor* servidor, e o servidor com maior *score* é o *pior* servidor. A Eq. 36 normaliza o valor das

pontuações dos servidores e os ordena de forma ascendente. Repare que *nscore*(*best*), a pontuação normalizada do melhor agente, é igual a 1 e *nscore*(*worst*), a pontuação normalizada do pior agente, é igual a zero.

$$score(server) = pfail(server).cost(server, t) \quad \text{(Eq. 35)}$$

$$nscore(\ server) = \frac{score(server) - score(worst)}{score(best) - score(worst)} \quad \text{(Eq. 36)}$$

A cada rodada do experimento, nós usamos 5 clientes do mesmo tipo, 5 servidores e 10 tarefas do mesmo tipo. Para simplificar o experimento, todos os servidores cobram o mesmo custo para executar todas as tarefas que lhe são solicitadas. As probabilidades de sucesso dos servidores são descritas na Tabela 18. Nós podemos ver que o *melhor* agente é o serv5 porque seu *nscore* é igual a 1 e o *pior* servidor é o *serv1* porque seu *nscore* é igual a 0. A máquina que usamos para rodar os experimentos foi: Intel Pentium Dual CPU; 2,99 Gb RAM; Windows XP SP3 e Java SE 1.7.

Para investigar a influência do modelo emocional proposto *like*/*dislike* sobre os agentes quando selecionam seus servidores nós desenvolvemos quatro variantes de agentes emocionais com diferentes estados emocionais iniciais, como demonstrado na Tabela 19. O primeiro (cliente emocional neutro) começa sentindo *neutro* em relação a todos os servidores, assim seu valor de intensidade é igual a 5. O segundo cliente (cliente emocional favorável) tem o valor inicial de *like/dislike* proporcional ao valor de *nscore* do servidor, de tal forma que ele goste mais dos *melhores* agentes para se alocar uma tarefa. O terceiro cliente (cliente emocional desfavorável) tem valor inicial de intensidade inversamente proporcional a *nscore*, de tal forma que ele goste mais dos *piores* agentes para se alocar uma tarefa. Já o quarto agente (cliente emocional randômico) inicia com valores aleatórios de intensidade emocional em relação a todos os servidores, e por isso não está representado na tabela.

server	cost	pfail	score	nscore
serv1	1	1.00	1.00	0.00
serv2	1	0.75	0.75	0.25
serv3	1	0.50	0.50	0.50
serv4	1	0.25	0.25	0.75
serv5	1	0.00	0.00	1.00

Tabela 18: Configuração dos Servidores

Emotional client	serv1	serv2	serv3	serv4	serv5
neutral	5	5	5	5	5
favorable	0	3	5	8	10
unfavorable	10	8	5	3	0

Tabela 19: Intensidade Emocional dos Clientes pelos Servidores

Nossa intenção neste experimento é obter as seguintes hipóteses:

1. Depois de várias interações, a pontuação *nscore* de todos os agentes será a mesma. Após várias interações com vários servidores, um cliente será capaz de encontrar qual o *melhor* servidor, maximizando *nscore*.
2. O desempenho do agente reputação deve ser melhor que o desempenho do agente racional. O cliente reputação será capaz de encontrar primeiro o *melhor* agente. Isto deve acontecer visto que o cliente reputação baseia sua decisão não somente em suas próprias interações, mas também na reputação reportada por outros clientes.
3. O cliente emocional favorável obtém melhor desempenho que os outros nas primeiras interações. Isto deve ocorrer visto que ele *gosta* mais dos *melhores* agentes servidores. Logo, irá preferir selecioná-los.
4. Inicialmente, o desempenho do cliente emocional neutro será similar ao do cliente reputação. Sabendo que este cliente emocional neutro sente-se *indiferente* em relação a todos os agentes, a emoção não irá influir na seleção dos servidores e o cliente emocional neutro irá adotar o mesmo algoritmo que o agente reputação, tornando-o similar a este.
5. O desempenho do agente randômico depois de algumas interações será similar ao do agente reputação e do agente neutro. A probabilidade do cliente emocional randômico gostar de um servidor com baixo desempenho e de gostar de um servidor com alto desempenho é igual. Assim, o agente só será capaz de trocar sua emoção para algo coerente com a realidade após algumas interações.
6. Clientes emocionais desfavoráveis serão os que tem pior performance. Sabendo que estes clientes gostam dos piores servidores, isto fará com que o agente tenda a selecioná-los durante as primeiras interações e terá baixo desempenho.

6.5.3.1 Resultados

Para cada tipo de cliente, nós rodamos 30 experimentos e tiramos a média dos *nscore* obtidos para cada tarefa, assim removendo possíveis *outliers* e encontrando o desempenho médio de cada tipo de cliente. Os valores da média de *nscore* são dados na **Erro! Autoreferência de indicador não válida.**.

O experimento confirma as hipóteses 2, 3, 4 e 6. Porém, o desempenho dos agentes não foi exatamente com esperado pela hipótese 1. Embora todos os agentes tenham alcançado a mesma performance nas últimas duas tarefas por selecionar os melhores servidores, o *agente emocional desfavorável* não conseguiu aumentar sua performance. Nós entendemos que este agente pode melhorar seu desempenho se ele executar mais rodadas de alocação de tarefas para encontrar quem são os melhores servidores. Este agente leva mais tempo que os

outros para encontrar os melhores servidores, pois (i) no início ele desgosta de todos os melhores servidores e gosta dos piores, e (ii) o algoritmo para seleção do servidor prioriza o que o cliente sente pelo servidor ao invés do seu desempenho.

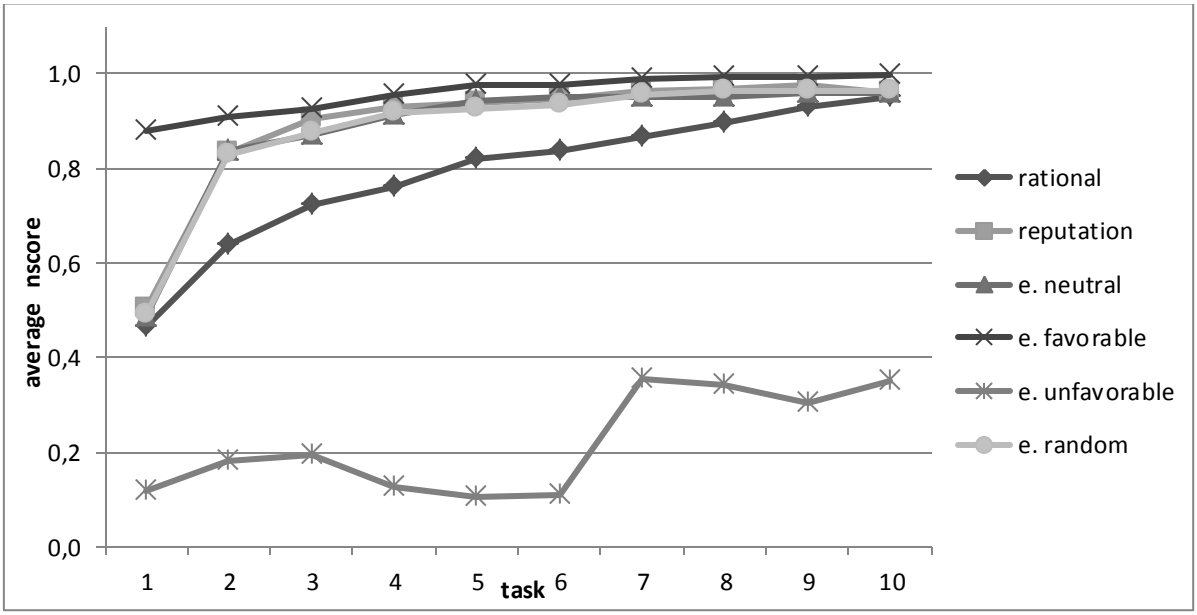


Figura 35: Média de *nscore* por tarefa *t*

7 Conclusão

Este trabalho apresentou a arquitetura de Agentes Emocionais UEBDI. Esta nova abordagem procura unificar as características de propostas anteriores e construir uma arquitetura genérica que contemple todas as possíveis influências do fenômeno emocional sobre o resto da arquitetura.

O Capítulo 2 procurou investigar os temas de computação afetiva, agentes inteligentes e agentes emocionais, levantando quais tópicos são necessários para a construção de uma arquitetura de agente emocional.

o Capítulo 3 analisou trabalhos que construíram um agente emocional BDI ou que tenham contribuído com modelos ou algoritmos que podem ser mapeados na arquitetura BDI, levantando-se quais seus pontos fortes e fracos e obtendo-se o que existe de estado da arte em agentes emocionais BDI, permitindo a posterior construção da nossa proposta.

O Capítulo 4 propõe a arquitetura abstrata propriamente dita. A metodologia adotada foi *top-down*, propondo uma arquitetura de agente a nível macro, observando os componentes de percepção, atuação e raciocínio, e "descendo" para o nível intermediário, que é a releitura das funções clássicas BDI (por exemplo, *Belief Review Function*) e a adição de novas funções para apoiar as funções emocionais (*Perception Function*, *First Emotion Review Emotion*, etc...), tal como já feito por alguns autores (Hernández et al. 2004, Jiang and Vidal 2006, Neto and Da Silva 2010, Pereira et al. 2005, Signoretti 2012). Por fim, "desceu-se" mais um nível, observando as funções do nível anterior internamente pelas suas sub-funções, determinando qual o papel de cada uma, de qual autor/trabalho foi trazida a proposta e qual sua relação com as outras funções e bases.

Observa-se que a proposta UEBDI não tem preocupação em explicar o fenômeno emocional ou em implementar uma teoria emocional em específico, mas sim em construir uma arquitetura abstrata que operacionalize a lógica do agente BDI e emocional em seus *fronzenspots* e ceda *hotspots* para que o projetista de agentes emocionais ou de modelos emocionais possam desenvolver seus trabalhos.

O capítulo 5 mapeou a arquitetura abstrata UEBDI na plataforma de agentes Jason, permitindo assim implementar uma versão da UEBDI, verificando a viabilidade técnica da proposta.

Por fim, o capítulo 6 apresentou cinco experimentos com o objetivo de demonstrar o uso da arquitetura para implementar agentes emocionais. O critério adotado para a escolha dos cenários foi exercitar as influências do módulo emocional sobre o módulo de percepções, crenças, desejos, intenções e ações, validando a proposta dada por este trabalho.

7.1 Contribuições

A principal contribuição deste trabalho foi propor uma arquitetura de Agente Emocional BDI abstrata que unifica as arquiteturas já pospostas anteriormente e permite definir influências do fenômeno emocional sobre todas as outras instâncias da arquitetura justificando estas propostas, seja por teorias emocionais, seja por justificativas computacionais. Nenhuma proposta anterior encontrada consegue atingir tal característica.

Como contribuições secundárias, nossa proposta foi mapeada e implementada sobre o Jason. E foi experimentada utilizando cinco experimentos diferentes que exercitavam as cinco influências possíveis do fenômeno emocional sobre as instâncias BDI (crenças, desejos, intenções, percepções e ações). O mapeamento demonstra a viabilidade técnica da proposta e os experimentos demonstram a efetividade da proposta em alcançar a meta de permitir as influências sobre as outras instâncias da arquitetura.

7.2 Limitações

Como todo trabalho, nossa proposta apresenta limitações que devem ser levadas em conta por outros pesquisadores e por projetistas de agentes quem queira adotá-la. Elas são:

- Reconsideração: A arquitetura BDI original (Wooldridge 2009) prevê reconsideração de intenções e planos durante a atuação do agente. Por simplicidade, não consideramos a reconsideração dentro da nossa arquitetura abstrata. O trabalho (Jiang and Vidal 2006) tenta atacar este problema timidamente, mas sem experimentá-lo. Acreditamos que emoções sirvam também para auxiliar este problema, visto que ajudariam ao agente ponderar se devem manter uma intenção ou eliminá-la ou replanejá-la.
- Formalismo Lógico: Existem trabalhos que se preocupam em criar uma lógica formal para representar agentes emocionais BDI, como os (Meyer 2004, Pereira et al. 2006) e, em especial, a série de trabalhos de Steunebrink (Steunebrink 2010, Steunebrink et al. 2008,

2009). Não procuramos definir um formalismo próprio, pois: (i) não era objetivo central do trabalho; (ii) a implementação Jason implementa a AgentSpeak(L), que é uma linguagem criada embasada em um formalismo lógico. A partir do momento que mapeamos nossa proposta no Jason, ela foi mapeada para a AgentSpeak(L) e para este formalismo lógico;

- Usabilidade: Não foi realizado um experimento para verificar a facilidade de uso da ferramenta com um conjunto de projetistas de agentes. Conhecer a usuabilidade da proposta seria um ponto a favor de sua adoção. Além disto, não foi analisada a usabilidade de nenhum trabalho relacionado. Considera-se então este um bom trabalho futuro.

Referências

Bates, J. and Loyall, A. B. and Reilly, W. S., (1994), "An Architecture for Action, Emotion, and Social Behavior"

Bazzan, A. and Adamatti, D. and Bordini, R., (2002), "Extending the computational study of social norms with a systematic model of emotions", *Advances in Artificial Intelligence*, p. 1–19.

Bazzan, A. L. C. and Bordini, R., (2001), "A framework for the simulation of agents with emotions". In: *Proceedings of the fifth international conference on Autonomous agents*, p. 292–299

Bordini, R. and Braubach, L. and Dastani, M. and Seghrouchni, A. and Gomez-Sanz, J. and Leite, J. and O'Hare, G. and Pokahr, A. and Ricci, A., (2006), "A Survey of Programming Languages and Platforms for Multi-Agent Systems", *Informatica*, v. 30, p. 33–44.

Bordini, R. H. and Hübner, J. F. and Wooldridge, M., (2007), *Programming Multi-Agent Systems in AgentSpeak using Jason*. 1 ed. West Sussex, England, John Wiley & Sons.

Bordini, R. and Hübner, J., (2007), *Jason Tutorial*.

Bratman, M. E., (1987), *Intention, Plans, and Practical Reason*. Cambridge, MA, Harvard University Press.

Braubach, L. and Pokahr, A., (2010), Jadex: BDI Agent System. *Jadex (Overview)*. Disponível em: <"http://jadex-agents.informatik.uni-hamburg.de/xwiki/bin/view/About/Overview">. Acesso em: 26 Apr 2010.

Braubach, L. and Pokahr, A. and Moldt, D. and Lamersdorf, W., (2005), "Goal representation for BDI agent systems", *Programming Multi-Agent Systems*, p. 44–65.

Breazeal, C. F., (1998), "A motivational system for regulating human-robot interaction". In: *Proc. of the fifteenth national/tenth conference on AI/Innovative applications of AI*, p. 54–61, Madison, Wisconsin, United States.

Breazeal, C. L., (2000), *Sociable machines: Expressive social exchange between humans and robots*, MIT

Camurri, A. and Coglio, A., (1998), "An Architecture for Emotional Agents", *IEEE MultiMedia*, v. 5, n. 4 (Oct.), p. 24–33.

Colman, A. M., (2003), "Cooperation, psychological game theory, and limitations of rationality in social interaction", *Behavioral and Brain Sciences*, v. 26, n. 02, p. 139–153.

Damásio, A., (2004), *Erro de Descartes, O*. São Paulo, Brazil, Companhia das Letras.

Dyer, M. G., (1987), "Emotions and their computations: Three computer models", *Cognition & Emotion*, v. 1, n. 3, p. 323–347.

Ekman, P. E. and Davidson, R. J., (1994), *The nature of emotion: Fundamental questions*. Oxford University Press.

Fix, J. and Scheve, C. von and Moldt, D., (2006), "Emotion-based norm enforcement and maintenance in multi-agent systems: foundations and petri net modeling". In: *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, p. 105–107, Hakodate, Japan.

Frijda, N. H., (1987), *The Emotions*. Cambridge, United Kingdom, Cambridge University Press.

Gadanho, S. and Hallam, J., (2001), "Robot learning driven by emotions", *Adaptative Behavior*, v. 9, n. 1, p. 42.

Georgeff, M. and Pell, B. and Pollack, M. and Tambe, M. and Wooldridge, M., (1998), "The belief-desire-intention model of agency". In: *Proceedings of Intelligent Agents V. Agent Theories, Architectures, and Languages: 5th International Workshop, ATAL'98*, p. 630–630, Paris, France.

Gratch, J., (1999), "Why you should buy an emotional planner". In: *Proceedings of the AgentsProceedings of the Agents'99 Workshop on Emotion-based Agent Architectures (EBAA'99)*, p. 99–465

Hernández, D. J. and Déniz, O. and Lorenzo, J. and Hernández, M., (2004), "BDIE: a BDI like Architecture with Emotional Capabilities", *American Association for Artificial Intelligence*, n. 2004, p. 8.

Hübner, J. and Bordini, R. and Wooldridge, M., (2006), "Programming declarative goals using plan patterns", *Declarative Agent Languages and Technologies IV*, p. 123–140.

Hübner, J. F. and Bordini, R. H., (2009), Jason: a Java-based interpreter for an extended version of AgentSpeak. *Jason Home*. Disponível em: <"http://jason.sourceforge.net/JasonWebSite/Jason%20Home.php">. Acesso em: 26 Apr 2010.

Izard, C. E., (1993), "Four systems for emotion activation: Cognitive and noncognitive processes", *Psychological Review*, v. 100, n. 1 (Jan.), p. 68–90.

Jiang, H., (2007), *From Rational to Emotional Agents*, University of South Carolina

Jiang, H. and Vidal, J. M., (2006), "From rational to emotional agents". In: *Proceeding of the AAAI Workshop on Cognitive Modeling and Agentbased Social Simulation*

LeDoux, J., (2002), "Emotion, memory and the brain", *Special Editions*

LeDoux, J. and Rogan, M., (1999), "Emotion and the Animal Brain", *The MIT Encyclopedia of Cognitive Sciences*, p. 269–270.

Lester, P., (2004), A * Pathfinding para Iniciantes. *A * Pathfinding para Iniciantes*. Disponível em: <"http://www.policyalmanac.org/games/aStarTutorial_port.htm">. Acesso em: 31 Oct 2012.

Leventhal, H. and Scherer, K. R., (1987), "The relationship of emotion to cognition: A functional approach to a semantic controversy", *Cognition & Emotion*, v. 1, n. 1, p. 3–28.

Lino, N. de L., (2006), *Modelo de Percepção de Agentes Inteligentes baseados em Emoções*, Universidade Federal de Pernambuco

Marsella, S. C. and Gratch, J., (2009), "EMA: A process model of appraisal dynamics", *Cogn. Syst. Res.*, v. 10, n. 1 (Mar.), p. 70–90.

McCrae, R. R. and John, O. P., (1992), "An Introduction to the Five-Factor Model and Its Applications", *Journal of Personality*, v. 60, n. 2, p. 175–215.

Mehrabian, A., (1996), "Pleasure-arousal-dominance: A general framework for describing and measuring individual differences in temperament", *Current Psychology*, v. 14, n. 4, p. 261–292.

Meneguzzi, F. R. and Zorzo, A. F. B. and Móra, M. D. A. C. and Luck, M., (2004), "Incorporating Planning into BDI Systems", *SCPE*, v. 2006, p. 2007–2010.

Meyer, J. J. C., (2004), "Reasoning about emotional agents". *Proceedings of 16th Eureopean Conference on Artificial Intelligence*, p. 22–27, Valenca Spain.

Minsky, M., (1988), *The Society of Mind*. Pages Bent ed. Simon & Schuster.

Morgado, L. F. G., (2006), *Integração de emoção e raciocínio em agentes inteligentes*. Thesis, Universidade de Lisboa Disponível em: <http://docs.di.fc.ul.pt/handle/10455/3314>.

Morgado, L. and Gaspar, G., (2005), "Emotion based adaptive reasoning for resource bounded agents". In: *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, p. 921–928, New York, NY, USA.

Moridis, C. N. and Economides, A. A., (2008), "Toward Computer-Aided Affective Learning Systems: A Literature Review", *Journal of Educational Computing Research*, v. 39, n. 4 (Jan.), p. 313–337.

Neto, A. F. . and Da Silva, F. S. ., (2010), "On the Construction of Synthetic Characters with Personality and Emotion". In: *Advances in Artificial Intelligence–SBIA 2010SBIA2010*, p. 102, São Bernado do Campo, Brazil.

Neto, A. F. B., (2010), *Uma arquitetura para agentes inteligentes com personalidade e emoção*, USP Disponível em: <www.ime.usp.br/~fcs/LIDET/bressane.pdf>.

Norling, E., (2004), "Folk Psychology for Human Modelling: Extending the BDI Paradigm". In: *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, p. 202–209, New York, New York.

Oliveira, E. and Sarmento, L., (2003), "Emotional advantage for adaptability and autonomy". In: *Proc. of the Second International Joint conference on Autonomous Agents and Multiagent Systems*, p. 305–312, Melbourne, Australia.

Ortony, A. and Clore, G. L. and Collins, A., (1990), *The cognitive structure of emotions*. Cambridge, USA, Cambridge University Press.

Padgham, L. and Taylor, G., (1997), "A System for Modelling Agents having Emotion and Personality", *PRICAI Workshop on intelligent agent systems*, p. 59–71.

Pereira, D. and Oliveira, E. and Moreira, N., (2006), "Modelling emotional BDI agents". In: *Workshop on Formal Approaches to Multi-Agent Systems (FAMAS 2006), Riva Del Garda, Italy (August 2006)*

Pereira, D. and Oliveira, E. and Moreira, N. and Sarmento, L., (2005), "Towards an architecture for emotional BDI agents". In: *Proc. of the Twelfth Portuguese Conference on AI*, p. 40–47

Picard, R. W., (1997), *Affective Computing*, Technical 321, MIT.

Picard, R. W., (2000), *Affective Computing*. 1 ed. Cambridge, USA, The MIT Press.

Picard, R. W., (2003), "Affective computing: challenges", *International Journal of Human-Computer Studies*, v. 59, n. 1, p. 55–64.

Pimentel, C. F. and Cravo, M. R., (2009), "“Don’t think too much!” - Artificial somatic markers for action selection". *3rd International Conference on Affective Computing and Intelligent Interaction and Workshops, 2009. ACII 2009*, p. 1 –8

Ramchurn, S. D. and Mezzetti, C. and Giovannucci, A. and Rodriguez-Aguilar, J. A. and Dash, R. K. and Jennings, N. R., (2009), "Trust-based mechanisms for robust and efficient task allocation in the presence of execution uncertainty", *Journal of Artificial Intelligence Research*, v. 35, n. 1, p. 119.

Rao, A., (1996), "AgentSpeak (L): BDI agents speak out in a logical computable language", *Agents Breaking Away*, p. 42–55.

Rao, A. and Georgeff. M., (1995), "BDI agents: From theory to practice", *Proceedings of the first international conference on multi-agent systems (ICMAS-95)*, p. 312–319.

Reekum, C. M. and Scherer, K. R., (1997), "Levels of processing in emotion-antecedent appraisal", In: Matthews, G. [ed.] (eds), *Advances in Psychology*, , chapter 124, North-Holland, p. 259–300.

Van Reekum, C. M. and Scherer, K. R., (1997), "Levels of processing in emotion-antecedent appraisal", *Advances in Psychology*, v. 124, p. 259–300.

Rodrigues, P. S. L., (2007), *Um Sistema de Geração de Expressões Faciais Dinâmicas em Animações Faciais 3D com Processamento de Fala*, PUC-Rio

Rumbell, T. and Barnden, J. and Denham, S. and Wennekers, T., (2011), "Emotions in autonomous agents: comparative analysis of mechanisms and functions", *Autonomous Agents and Multi-Agent Systems*, p. 1–45.

Russel, S. J. and Norvig, P., (2004), *Inteligência Artificial: tradução da segunda edição*. 2 ed. Rio de Janeiro, Brazil, Elsevier.

Sarmento, L. M., (2004), *An emotion-based agent architecture*, Universidade do Porto

Scherer, K. R., (1999), "Appraisal theory", *Handbook of cognition and emotion*, , p. 637–663.

Scherer, K. R., (2000), "Emotions as episodes of subsystem synchronization driven by nonlinear appraisal processes", *Emotion, development, and self-organization: Dynamic systems approaches to emotional development*, p. 70–99.

Scherer, K. R., (2005), "What are emotions? And how can they be measured?", *Social Science Information*, v. 44, n. 4, p. 695 –729.

Scheutz, M., (2002), "Agents with or without emotions". In: *Proceedings FLAIRS*, p. 89–94

Scheutz, M., (2004), "How to determine the utility of emotions", *IN PROC. OF AAAI SPRING SYMPOSIUM*

Scheve, C. von and Moldt, D. and Julia Fix and R. von Luede, (2006), "My agents love to conform: Norms and emotion in the micro-macro link", *Computational & Mathematical Organization Theory*, 2 ed, chapter 12, Springer

Signoretti, A., (2012), *Agentes Inteligentes com Foco de Atenção Afetivo em Simulações Baseadas em Agentes*, UFRN

Signoretti, A. and Feitosa, A. and Campos, A. M. and Canuto, A. M. and Fialho, S. V., (2010), "Increasing the Eciency of NPCs Using a Focus of Attention Based on Emotions and

Personality". In: *Proceedings of the 2010 Brazilian Symposium on Games and Digital Entertainment*, p. 171–181, Washington, DC, USA.

Silva, D. R. D. and Tedesco, P. R. and Ramalho, G. L., (2006), "Usando Atores Sintéticos em Jogos Sérios: O Case SmartSim". In: *V Brazilian Symposium of Games and Digital Entertainment (SBGAMES 2006)SBGames*, Pernambuco, Brazil.

Sloman, A., (1999), "How many separately evolved emotional beasties live within us?". In: *Presented at workshop on Emotions in Humans and Artifacts Vienna*

Sloman, A., (2004), "What are theories of emotion about". In: *Proceedings of the AAAI Spring Symposium: Architectures for Modeling Emotion: cross-disciplinary foundations. AAAI Technical report SS-04-02. AAAI Press*

Staller, A. and Petta, P., (2000), "Introducing emotions into the computational study of social norms". In: *In Proceedings of the 2000 Convention of the Society for the Study of Artificial Intelligence and the Simulation of Behaviour (AISB'00)*, p. 17–20, Birmingham, UK.

Staller, A. and Petta, P., (2001), "Introducing emotions into the computational study of social norms", *Journal of Artificial Societies and Social Simulation*, v. 4, n. 1

Steunebrink, B. R., (2010), *The Logical Structure of Emotions*, Utrecht University

Steunebrink, B. R. and Dastani, M. and Meyer, J. J. ., (2009), "A Formal Model of Emotion-Based Action Tendency for Intelligent Agents". In: *Proc. of the 14th Portuguese Conference on Artificial Intelligence: Progress in AI*, p. 174–186

Steunebrink, B. R. and Dastani, M. and Meyer, J. J. C., (2008), "A Formal Model of Emotions: Integrating Qualitative and Quantitative Aspects"

Teasdale, J., (1999), "Multi-level theories of emotion", *Handbook of cognition and emotion*, Chichester: Wiley

Tomaz, C. and Giugliano, L. G., (1997), "A razão das emoções: um ensaio sobre "O erro de Descartes"", *Estudos de Psicologia (Natal)*, v. 2, p. 407–411.

Velásquez, J., (1998a), "Modeling emotion-based decision-making", *Emotional and Intelligent: The Tangled Knot of Cognition*, p. 164–169.

Velásquez, J. D., (1996), *Cathexis: a computational model for the generation of emotions and their influence in the behavior of autonomous agents*, MIT Disponível em: <http://dspace.mit.edu/handle/1721.1/10651>.

Velásquez, J. D., (1998b), "When robots weep: emotional memories and decision-making". In: *Proceedings Of The National Conference On Artificial Intelligence*, p. 70–75

De Weerdt, M. and Zhang, Y. and Klos, T., (2007), "Distributed task allocation in social networks". In: *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, p. 76:1–76:8, New York, NY, USA.

Wilson, R. A. and Keil, F. C., (1999), *The MIT Encyclopedia of Cognitive Sciences*. Cambridge, Massachusetts, USA, MIT Press.

Wilson, S. W., (1991), "The Animat Path to AI". In: *From animals to animatsProcedings of the First Conference on Simulation of Adaptive Behavior*, p. 15–21, Cambridge, Massachusetts, USA.

Wooldridge, M., (2009), *An Introduction to MultiAgent Systems*. 2nd ed. West Sussex, England, John Wiley & Sons.

Anexo A – Diagrama de Arquitetura de Agentes

Vamos definir um diagrama de descrição de arquitetura dos agentes. Nossa motivação para definir tal diagrama veio de uma observação sobre a literatura levantada: não existe um padrão largamente aceito e definido formalmente para a descrição de arquiteturas de agentes. Porém existe uma tendência a utilizar diagramas derivados do diagrama de fluxo de dados e/ou do diagrama de atividades da UML, normalmente constituídos pelo relacionamento entre entidades funcionais e bases de dados (Bordini et al. 2007, Hernández et al. 2004, Jiang and Vidal 2006, Meneguzzi et al. 2004, Neto 2010, Pereira et al. 2005, Wooldridge 2009). Este tipo de diagrama acaba sendo um padrão *de facto*. Devido a isto, nossa proposta nada mais é do que uma formalização deste modelo que já são adotados informalmente.

O diagrama é composto por duas entidades básicas, chamadas **funções** e **bases de dados**. As funções são entidades que realizam modificações em dados e podem receber de 0 (zero) a N (várias) bases de dados como entrada e gerar um novo conjunto de dados para serem depositados em 1(uma) ou N (várias) bases de dados. Também se pode considerar que uma função pode receber dados de uma "fonte", por exemplo: uma função de percepção recebe dados não de uma base, mas do ambiente. Neste caso, o ambiente é uma **fonte de dados**. As bases contêm um conjunto de elementos de um tipo específico. Uma função, por padrão, sempre substitui o conteúdo de uma base por inteiro, salvo quando dito o contrário.

Pode-se descrever as bases por seu nome, começando com a primeira letra da palavra em maiúscula (como, por exemplo, *Belief*) ou simplesmente as primeiras letras que diferem seu nome das demais bases (como *B*). Já as funções podem ser descritas pela forma $f : X_1, X_2,..., X_n \rightarrow Y_1, Y_2,..., Y_m$, onde f é o nome da função, $X_1, X_2,..., X_n$ são as bases que são usadas como parâmetro de f e $Y_1, Y_2,..., Y_m$ são as bases que são alteradas por f.

Estas entidades se relacionam através de **setas direcionadas**. Quando uma seta parte de uma função e vai a uma base, indica que a função altera o conteúdo da base. Quando uma seta parte de uma base em direção a uma função, indica que a base é parâmetro da função. Quando uma seta parte de uma função e vai a outra função, indica após a execução da

primeira, será executada a segunda. Não pode existir uma seta que parta de uma base e vá para outra base.

Existem duas entidades especiais chamadas de **ponto inicial** e **ponto final**. O ponto inicial indica onde começa o fluxo de execução e o final onde termina. Elas são representadas por um ponto preto preenchido e por um ponto preto preenchido com um círculo preto em volta, respectivamente.

Os relacionamentos entre funções e entre os pontos iniciais e finais podem ter como descrição o nome de um tipo de dado – por exemplo *P* – que indica um tipo de dado que é passado de uma função a outra. Isto é equivalente a dizer que existe uma **base de dados oculta** que armazena os dados temporariamente de uma função a outra. Quando esta descrição está em um relacionamento partindo de um ponto inicial, indica que antes do início o fluxo recebeu dados de alguém que o chamou externamente. E em um relacionamento indo para um ponto final indica que a última função gerou dados que devem ser oferecidos a quem chamou este fluxo.

Existe uma função especial chamada **ponto de decisão** que indica que naquele local o sistema deve decidir, em tempo de execução, qual outra função será a próxima. Esta função pode receber parâmetros como as outras, mas não pode alterar as bases. Nos diagramas pode-se não indicar quais bases influenciam esta função, para simplificar a descrição. Também, nos relacionamentos para as funções posteriores, pode-se adicionar um texto descrevendo a condição para que aquele caminho seja seguido. Entretanto, é obrigatório que se siga por um e somente um caminho.

Quando for preciso bifurcar o fluxo de execução em dois independentes usa-se a **barra de sincronização**, cuja semântica similar a do diagrama de atividades: uma seta direcionada parte de uma função em direção a um dos lados da barra. Do outro lado partirão outras duas ou mais setas representando os fluxos independentes. Quando for preciso juntá-los novamente, as setas do final do fluxo incidirão sobre um lado de outra barra, e no outro lado desta barra partirá uma única seta, unificando os fluxos anteriores.

Uma **função composta** é uma função composta por outras sub-funções ou sub-bases. Para representar isto usaremos um retângulo representando a função composta em si que conterá em seu interior os retângulos de suas sub-funções e/ou os cilindros de suas sub-bases. O início e o fim do fluxo dentro desta função serão representados por pontos iniciais e finais, respectivamente.

Outra forma de representar uma função composta por outra é usar uma **seta-losango** (similar ao relacionamento de composição da UML), onde a função do lado do losango é

composta pela outra do outro lado. A diferença desta representação em relação a anterior é que ela perde a ordem de execução das funções que compõe a função composta.

Às vezes é preciso dizer que uma função equivale a outra (ou outras). Quando isto ocorre dizemos que estamos *mapeando* uma função em outra(s) e existe uma **função mapeada** que é mapeada em outra **função alvo**.

Para representar isto graficamente, desenhamos a função mapeada como uma função qualquer (retângulo arredondado) e desenhamos em volta dela outro retângulo tracejado que representa a função alvo. Pode-se mapear uma função em uma ou mais outras, desde que parte do retângulo da função mapeada esteja "dentro" das outras funções alvo.

Quando, semanticamente, conseguimos mapear todas as responsabilidades de uma função em outra, dizemos que este mapeamento é um **mapeamento completo**, caso contrário, o mapeamento é **incompleto.** Para representarmos isto, devemos desenhar uma faixa diagonal no canto superior esquerdo do retângulo da função alvo.

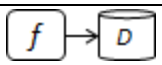
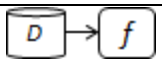
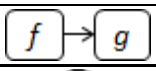
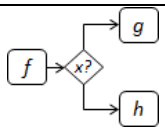
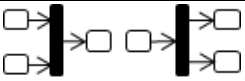
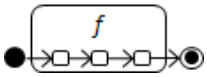
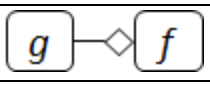
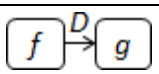
Nome	Imagem	Descrição
Retângulo arredondado		Função *f*
Cilindro		Base de dados do tipo *Data* (*D*)
Relacionamento *f* para *D*		Função *f* alterará a base *D*
Relacionamento *D* para *f*		Base *D* serve de parâmetro para a função *f*
Relacionamento *f* para *f*		Função *g* é executada após a função *f*
Ponto inicial		Ponto onde começa o fluxo do diagrama
Ponto final		Ponto onde termina o fluxo do diagrama
Ponto de decisão		Especifica uma condição para decidir entre dois fluxos posteriores
Barra de sincronização		Determina a bifurcação e junção de fluxos de execução
Função composta		Função *f* é composta por outras
Seta-losango de *g* para *f*		Função *f* é composta por função *g*
Base de Dados oculta		*D* é uma base de dados temporária que armazena a saída de *f* e para ser entrada em *g*
Retângulo tracejado		Função alvo
Faixa diagonal		Função *f* mapeada incompletamente

Tabela 20: Figuras usadas no diagrama