3

Programando Agentes em Situações de Desastre: o Caso dos *Extreme Teams* no Ambiente RoboCup Rescue

Ana Lúcia C. Bazzan - bazzan@inf.ufrgs.br[1]

Resumo:

Nosso planeta tem sido alvo de diversos eventos naturais catastróficos como terremotos de magnitude acima de 7 na escala Richter, furacões e tsunamis e, para mencionar um exemplo nacional, as recentes enchentes nos estados de SC, AL e RJ. Lidar com situações de desastre não é uma tarefa fácil para um time de agentes, sejam humanos ou artificiais (e.g. robôs). Coordenação e trabalho em time de agentes é fundamental. De fato, Paul Scerri e colegas cunharam o termo extreme teams para tal caso pois trata-se de times atuando em situações extremas.

Para testar métodos de coordenação destes times foi criada em 2001 a liga de competição *RoboCup Rescue* onde o objetivo é que os métodos propostos pelas equipes competidoras sejam testados visando a resolução de tarefas tais como resgatar civis, apagar incêndios e desbloquear ruas atingidas por escombros.

Neste mini-curso serão introduzidos o ambiente RoboCup Rescue atualmente em uso (http://sourceforge.net/projects/roborescue/) bem como algumas técnicas de programação dos agentes que compõe o cenário (paramédicos, bombeiros

[1]Doutora – Univ. Técnica de Karlsruhe (1997). Atualmente, é professora associada no Instituto de Informática da UFRGS. Sua pesquisa envolve as áreas de inteligência artificial, sistemas multiagentes, simulação baseada em agentes, aplicações de teoria de jogos em coordenação de agentes, aprendizado multiagente e inteligência coletiva e de enxame. Estas técnicas têm sido aplicadas principalmente em problemas de controle e simulação de tráfego urbano, simulação de situações de emergência e desastres, e bioinformática.

e policiais). Será discutida a necessidade de melhorar o desempenho através do uso, por exemplo, de programação de alto desempenho. Um dos objetivos é fomentar atividades futuras em torno deste tema, como a criação de novas equipes e a realização de um campeonato entre estas.

Antes, serão introduzidos conceitos gerais sobre sistemas multiagentes, os quais constituem a base dos métodos de coordenação a serem discutidos. Como será visto, a área de sistemas multiagentes é nova e desafiante. A partir do momento em que um sistema contém mais de um agente, as técnicas de inteligência artificial tornam-se não totalmente adequadas na medida em que não consideram as interações entre os agentes, a necessidade de coordenação, etc.

3.1. O que esperar deste capítulo

Este mini-curso trata de uma aplicação prática de sistemas multiagentes: coordenar times de agentes para lidar com situações de emergência. Serão introduzidos conceitos básicos sobre sistemas multiagentes e posteriormente o enfoque será o ambiente de simulação *RoboCup Rescue* (`www.robocuprescue.org`).

Este texto está estruturado conforme segue. Uma primeira parte (abrangendo as seções 3.2. a 3.4.) trata de conceitos básicos de sistemas multiagentes. A segunda parte (Seção 3.6.) aborda basicamente a questão de coordenação de agentes em simulação de situações de emergência. Após algumas dessas partes, procura-se fornecer ao leitor uma lista de material suplementar onde é possível estender o conhecimento sobre o assunto em questão. O texto conclui com a seção 3.7. que apresenta os principais pontos abordados.

3.2. Agentes autônomos e sistemas multiagentes

Há algumas décadas a inteligência artificial (IA) tem focado o emprego de suas técnicas de resolução de problemas em uma entidade única, seja ela um robô, um sistema especialista, um veículo, etc. Entretanto, a rigor, nenhuma destas entidades deveria ser tratada de forma isolada pois, em geral, trata-se de uma associação de unidades que podem ter inter-relacionamentos complexos. Por outro lado, ao se decompor um determinado problema, são geradas várias partes que interagem umas com as outras. Tal interação deve ser cuidadosamente tratada sob pena de se ter que lidar com complexidade igualmente alta. Esta foi a motivação inicial que levou ao aparecimento de uma nova área da IA, inicialmente denominada "inteligência artificial distribuída" (IAD).

Segundo G. Bittencourt [BIT 2001], "a IAD é uma sub-área da IA que estuda o conhecimento e as técnicas de raciocínio que podem ser necessárias ou úteis para que os agentes computacionais participem de sociedades de agentes". A IAD se preocupa com uma ou mais dentre as seguintes tarefas: decomposição de problemas complexos, alocação de tarefas entre um grupo de agentes de forma que estes melhorem seu desempenho como grupo, distribuição do controle ou das tarefas, evitar interações danosas, comunicação, síntese das soluções parciais ou locais e garantir a solução global do problema, se possível de forma cooperativa. É preciso frisar aqui que cooperação não necessariamente significa que os agentes tenham que se comportar de maneira benevolente ou altruísta; eles podem ser levados a ter comportamentos antagônicos. Neste caso, a cooperação pode emergir quando existe a necessidade dos agentes trabalharem juntos a fim de levar a cabo seus objetivos

privados ou particulares. Na literatura, tais agentes também são denominados intencionais ou ainda agentes motivados individualmente ou auto-motivados (ou seja, motivados por seus objetivos ou por seus estados internos). Igualmente, é preciso frisar que comunicação entre agentes não necessariamente significa que esta deva se dar de forma explícita (como uso de atos de fala). Ao contrário, comunicação pode ser implícita, como por exemplo quando existe alguma forma de conhecimento comum entre os agentes, fato este frequentemente utilizado em interações baseadas em formalismos da teoria de jogos.

A IAD também é normalmente associada com o termo "sociedade de agentes", o que aqui significa uma rede de agentes na qual cada um tem habilidades particulares mas não é capaz de resolver o problema como um todo devido à falta de recursos, informação ou perícia.

Originalmente, as várias técnicas de IAD relatadas na literatura foram divididas em duas grandes áreas de acordo com sua visão: orientadas ao problema (granulometria alta) e orientadas ao agente (granulometria fina). Posteriormente, estas duas visões tornaram-se conhecidas como "resolução distribuída de problemas" (DPS, de Distributed Problem Solving) e sistemas multiagentes.

A DPS ocupa-se preponderantemente com a decomposição de um problema entre uma rede de agentes e com os mecanismos de cooperação e comunicação necessários para resolver o problema.

Já um sistema multiagente ocupa-se principalmente com a coordenação dos agentes, especialmente se forem individualmente motivados. Neste caso, normalmente os agentes devem compartilhar intenções, planos e conhecimentos de forma a poder se coordenar. A coordenação é necessária para que se resolvam conflitos que podem surgir quando se alocam recursos limitados ou quando os agentes têm preferências conflitantes. Assim, a coordenação depende muito da forma de interação que emerge dos agentes. Para ser efetiva, é preciso que haja um certo grau de previsão das atitudes dos demais agentes, o que pode ser obtido através de um modelo das preferências dos outros agentes, por exemplo.

Devido ao relativo baixo interesse atual em torno das DPS (compreensível se verificarmos o seu reduzido escopo frente aos problemas atualmente colocados, por exemplo, pela Internet), o restante desta discussão será focado em sistemas multiagentes. Antes, porém, serão introduzidos os conceitos de agente e algumas de suas características.

A origem da palavra agente vem do latim *agere* ou agir. Entretanto, o termo agente não tem uma definição consensual. Para Wooldridge, "um agente é um sistema computacional situado em algum ambiente, sendo capaz de realizar, de forma independente (autonomamente), ações neste ambiente, descobrindo o que necessita ser feito para satisfazer seus objetivos, ao invés de ter que receber esta informação". Para Franklin e Graesser, "um agente autônomo é um sistema situado dentro de e

parte de um ambiente, ambiente este que o agente percebe e nele age ao longo do tempo, perseguindo um objetivo dentro de sua própria agenda. A ação do agente por sua vez afeta o que ele perceberá no futuro".

Nestas duas definições aparecem alguns conceitos importantes. Por exemplo, agentes devem:

- estar situados (em um ambiente)
- descobrir autonomamente o que fazer (sem ser dito)
- perseguir sua própria agenda
- agir ao longo do tempo

Agentes podem ter uma arquitetura reativa ou deliberativa. Um agente reativo é baseado em modelos simples como o estímulo-resposta e baseado em arquiteturas de subsunção conforme proposto por [BRO 86]. De fato, uma das implicações práticas desta arquitetura foi que a área de agentes se tornou um campo fértil para teste em cenários realistas. Agentes reativos são usados principalmente em ambientes onde eles são numerosos (ordem de centenas, milhares de agentes) e a eficiência do processamento é uma questão chave.

Agentes deliberativos são inspirados em organizações sociais humanas, como empresas e suas hierarquias organizacionais, comunidade científica e mercados no sentido da economia. Neste caso a arquitetura do agente representa explicitamente não apenas o ambiente mas também os demais agentes. Neste tipo de arquitetura o planejamento das ações futuras é em geral no estilo do planejamento clássico[2]. Isto envolve, portanto, uma caracterização formal de atitudes mentais como intenções, objetivos, crenças. Além disso a comunicação entre agentes tem um papel importante neste tipo de arquitetura, o que não é o caso em uma arquitetura reativa. Justamente devido à complexidade das arquiteturas deliberativas, este tipo de agente é encontrado em cenários onde seu número é da ordem de dezenas de agentes.

Outra característica, menos óbvia e mais polêmica, é que um agente pode ser analisado sob uma perspectiva microeconômica, ou seja, pode ser visto como uma entidade racional, auto-motivada e não disposta a mostrar benevolência perante os outros. Os seguidores desta linha colocam a diferença entre agentes e objetos da seguinte maneira: enquanto agentes fazem algo para obter um certo ganho, objetos o fazem sem tal contrapartida. Esta visão de agência é criticada por ser orientada ao ganho e por não dar muito espaço para que o agente coopere com outros ao menos que isso contribua para que seus objetivos sejam satisfeitos. Entretanto, tal

[2]O leitor interessado neste assunto pode consultar os capítulos 11 e 12 do livro de Russell e Norvig [RUS 2004].

abordagem é válida e útil em certos domínios onde os objetivos dos agentes são altamente antagônicos.

Assim como não há uma definição universalmente aceita sobre o que é um agente, também não existe uma definição unânime para sistemas multiagentes. Pode-se dizer que um sistema multiagente é um sistema que consiste em um número de agentes que interagem uns com os outros. Além disso, para interagir de forma eficaz, os agentes deste sistema devem ser capazes de cooperar, se coordenar e negociar entre si [WOO 2002].

Um sistema multiagente possui algumas características, como por exemplo: cada agente tem informação ou capacidades limitadas para resolver o problema; não existe um controle global do sistema; o conhecimento é distribuído.

Uma das motivações para o desenvolvimento de sistemas multiagentes é a possibilidade de resolver problemas que não podem ser resolvidos de forma centralizada (por limitação de recursos ou desempenho). Por outro lado, como os sistemas multiagentes preveem vários agentes envolvidos em um ambiente normalmente complexo, conflitos precisam ser tratados ou na fase de projeto ou durante a execução das tarefas.

3.3. Coordenação e cooperação em um sistema multiagente

3.3.1. Coordenação de agentes

De uma forma geral, a coordenação aumenta o desempenho dos sistemas multiagentes e é um componente-chave, uma vez que cada agente possui apenas uma visão local e incompleta. Ela é fundamental em tarefas de planejamento, conforme detalhado no capítulo 9 de [WOO 2002], e no capítulo 3 de [WEI 99]. Por fim, a coordenação é necessária a fim de se resolver conflitos, alocar recursos limitados, conciliar preferências e buscar soluções de caráter global.

Devido à importância da coordenação, é conveniente se olhar para seu significado em outras áreas. De fato, é possível encontrar vários trabalhos envolvendo o estudo de coordenação sob o aspecto multidisciplinar (computação, teoria de organizações, administração, economia, linguística e psicologia). Um estudo interessante é o de Malone e Crowston [MAL 94] que discute a seguinte questão: como o uso de tecnologia de informação muda a forma como as pessoas colaboram? Neste trabalho são relacionadas várias definições para coordenação, entre as quais as mais relevantes para a área de sistemas multiagentes são: i) coordenação é a tarefa de gerenciar dependências entre atividades [MAL 94]; ii) coordenação é a decisão sobre

como compartilhar recursos, a qual em geral envolve negociação e comprometimentos [ROS 94].

Desta forma fica claro que o projeto de mecanismos adequados de coordenação é uma tarefa fundamental na concepção de um sistema multiagente. Neste texto não serão fornecidos detalhes sobre todos os mecanismos em si por serem de diversas naturezas. O leitor pode consultar [WEI 99] ou [WOO 2002] para este fim.

3.3.2. Cooperação entre agentes

Em uma sociedade ou rede de agentes onde cada um tem conhecimento e perícia limitada, a cooperação é necessária para que o objetivo maior de toda a sociedade seja atingido. O nível e forma de cooperação entre agentes pode variar em um espectro que vai desde totalmente cooperativa até antagônica. Sistemas totalmente cooperativos em geral pagam um alto preço sob a forma de custos de comunicação. Enquanto agentes totalmente cooperativos podem trocar objetivos entre si de forma a solucionar o problema como um todo, em sistemas onde os agentes são antagônicos, estes podem optar por não cooperar de modo algum, além de inclusive tentar impedir as ações dos demais agentes se estas estão em conflito com o objetivo do agente em questão. Se por um lado aqui os custos de comunicação são baixos (ou inexistentes), por outro lado a eficiência pode ser seriamente comprometida.

O papel da cooperação difere nas diversas abordagens propostas. Enquanto a cooperação pode ser vista como um caso especial de coordenação entre agentes não antagônicos (ou seja cooperativos por natureza), alguns pesquisadores defendem que a cooperação pode emergir também entre agentes antagônicos, desde que o benefício de tal cooperação aumente a chance do indivíduo atingir seu objetivo. Por exemplo, duas companhias telefônicas podem fazer um acordo a fim de rotear pacotes que sejam do interesse de ambas [ROS 94]. Esta abordagem (e as inúmeras que dela derivaram) é baseada em ideias advindas da teoria de jogos (cooperativa e não cooperativa), enquanto formalismo matemático para analisar a natureza das interações e cooperações entre agentes antagônicos em sistemas multiagentes.

Para se conseguir raciocinar sobre cooperação, é preciso modelar as intenções dos demais agentes, o que pode ser feito de várias formas. A modelagem explícita dos estados mentais dos agentes é a forma mais sofisticada. Uma introdução a este tópico pode ser encontrada em /citeBazzan2010si.

Uma outra forma de se obter cooperação é via simples troca de informação entre os agentes. Normalmente este processo se apoia nas seguinte etapas: desenvolvimento de um plano que considere o comportamento dos outros agentes, comunicação das partes relevantes do plano, e comunicação do comprometimento de cada agente com suas ações. Dependendo do número de agentes envolvidos, da complexidade dos planos e a da precisão desejada, os custos de comunicação

podem ser inviáveis.

Este fato motivou uma terceira linha de abordagens, a baseada em teoria de jogos, conforme já mencionado. Esta linha se apoia no conceito de conhecimento comum (*common knowledge*). A ideia básica é que as crenças e intenções dos agentes são de conhecimento de todos os agentes (comum) e enquanto este conhecimento for válido, os agentes não necessitam comunicação para cooperarem.

Resumindo, a cooperação está associada com o compartilhamento de objetivos, enquanto a coordenação está associada com o fato de se considerar os planos dos outros. Dessa forma, a menos que os agentes garantidamente tenham o mesmo objetivo, o uso de cooperação apenas não é efetivo; as ações dos indivíduos devem também ser coordenadas.

Na próxima seção será detalhado um aspecto ligados a um sistema multiagente: resolução de problemas em sistemas multiagentes (seção 3.4.). Para outros aspectos (lógicas para sistemas multiagentes, planejamento multiagente, tomada de decisão e aprendizado em ambientes com mais de um agente, o leitor pode consultar [BAZ 2010].

3.4. Resolução de problemas em sistemas multiagentes

Em IA, um foco tradicional é o da resolução de problemas (*problem-solving*). Uma abordagem para resolução de problema é via satisfação de restrições (em inglês *constraint satisfaction problem* ou CSP). CSP é um formalismo bastante utilizado na IA para representação simples e estruturada de problemas, como por exemplo coloração de grafos, escalonamento e alocação de tarefas, agendamento de reuniões, etc. Um CSP envolve três componentes: *variáveis*, *domínios* e *restrições*. Uma variável corresponde a uma parte do problema que pode ter seu valor alterado. Um domínio consiste em um conjunto de valores pré-definidos que uma variável pode assumir. Uma restrição corresponde a uma condição que deve ser satisfeita ao se atribuir valores para as variáveis. O objetivo em um CSP é definir um valor para cada variável de forma a satisfazer todas as restrições.

Em um CSP temos um conjunto $X = \{x_1, \ldots, x_n\}$ de variáveis. A cada variável x_i está associado um domínio D_i não vazio, finito e discreto. O valor que a variável assume é tomado de seu respectivo domínio. Além disto, há um conjunto de restrições $C = \{C_1, \ldots, C_m\}$. Cada restrição C_j envolve um subconjunto de variáveis de X e especifica as combinações de valores permitidas para o subconjunto em questão. Uma solução do problema corresponde a uma *atribuição* de valores a algumas ou a todas as variáveis. Uma atribuição de um valor v_i para uma variável x_i é representada por $\langle x_i, v_i \rangle$. Um estado é portanto um conjunto de atribuições

$\{\langle x_i, v_i\rangle, \langle x_j, v_j\rangle, \ldots\}$.

Determinados CSPs podem apresentar não apenas uma, mas várias soluções. Nestes casos é estabelecido um critério de qualidade para avaliar cada solução, e é preferível aquela com melhor qualidade. Esta qualidade normalmente é definida em termos de funções de custo, podendo-se optar por soluções que maximizem ou minimizem o custo dependendo do problema. Em outros casos, a situação oposta pode ocorrer, ou seja, não é possível satisfazer todas as restrições ao mesmo tempo. Quando isto ocorre, opta-se por uma atribuição de valores às variáveis que também maximizem ou minimizem uma função de custo especificada para o problema. Problemas que possuem estas características são denominados de COP (do inglês *Constraint Optimization Problems*).

Notamos então que o formalismo básico em torno de CSP e COP é útil para certos problemas de planejamento e busca quando colocados de forma *centralizada*. Entretanto, no mundo real existem situações nas quais o problema necessita ser resolvido de forma distribuída, seja porque não há capacidade de processamento central, seja porque os recursos e o conhecimento se encontram de fato distribuídos, ou ainda devido a altos custos de comunicação para centralização de todo o problema em um único local. Além disso podem haver questões intrínsecas de segurança e privacidade das informações.

Um exemplo típico é o de agendamento de reuniões. Neste problema, em geral, as restrições bem como as informações sobre horários não são de domínio público e/ou existem questões de privacidade que implicam que nenhuma entidade central tem todas as informações a fim de resolver o problema usando CSP clássico.

Para especificar um CSP distribuído, foi proposto o chamado DisCSP (do inglês, *distributed constraint satisfaction problem*) [YOK 92]. Já o formalismo que especifica COPs de maneira distribuída é denominado DCOP (do inglês *distributed constraint optimization problem*) [MOD 2003].

Tanto em DisCSP quanto em DCOP, cada variável é gerenciada por um agente. O objetivo global continua sendo o mesmo. Entretanto, cada agente tem agora autonomia para decidir o valor que será atribuído à variável. Este problema está longe de ser trivial já que nenhum agente tem uma visão global. Desta maneira, cada agente tem que se comunicar com outros agentes.

3.5. Para saber mais sobre conceitos e técnicas ligadas a agentes autônomos e sistemas multiagentes

Neste texto foi introduzida de forma breve e seletiva apenas uma pequena parte do ferramental, das técnicas e dos conceitos que embasam a pesquisa tradici-

onal e atual na área de agentes autônomos e sistemas multiagente. Desta forma, o leitor é remetido aos três livros que trazem este material de uma forma mais completa: [WEI 99], [WOO 2009][3], e [SHO 2009]. Para um texto introdutório em língua portuguesa, ver [BAZ 2010a, BAZ 2010].

Em relação aos tópicos aqui abordados, estes podem ser aprofundados da seguinte forma. Cooperação e coordenação são tratados no capítulo 8 de [WOO 2009]. Abordagens baseadas em teoria de jogos aparecem nos capítulos 3–6 de [SHO 2009] e nos capítulos 11 e 13 de [WOO 2009]. Tópicos relacionados a este assunto, como leilões, votação, protocolos de negociação e alocação de tarefas são objeto dos capítulos 12 e 14 de [WOO 2009], 9 e 11 de [SHO 2009], e 5 de [WEI 99]. Aprendizado multiagente é o tópico dos capítulos 6 de [WEI 99] e 7 de [SHO 2009]. Comunicação é discutida no capítulo 7 de [WOO 2009] e em [SHO 2009] no capítulo 8. Para retornar aos conceitos básicos em torno de DPS e IAD, textos clássicos são: [BON 88, GAS 90, JEN 96]. O capítulo 10 de [JEN 96] trata de métodos formais para desenvolvimento de sistemas multiagentes. Este material é clássico porém não atualizado. Sugere-se o livro [BOR 2007] como complementação. Em relação aos métodos em torno de CSP e COP distribuídos, um artigo clássico sobre DisCSP e o asynchronous backtracking algorithm é [YOK 92]. Métodos em torno de DCOP são apresentados em [MOD 2003, MOD 2005] (ADOPT), [MAI 2004] (OptAPO) e [PET 2005] (DPOP). A base para entendimento de CSP's em geral pode ser obtida por exemplo em [KUM 92].

3.6. Aplicação de sistemas multiagentes em simulação de situações de emergência: *RoboCup Rescue*

3.6.1. Visão geral

Em 1995, um terremoto de grandes proporções atingiu a cidade japonesa de Kobe. Milhares de construções foram destruídas, soterrando pessoas e obstruindo ruas. Centenas de focos de incêndio surgiram e se alastraram por várias construções. A partir desta catástrofe, notou-se a necessidade de um sistema que possa criar planos robustos, dinâmicos e inteligentes para busca e resgate que auxilie o esforço humano em situações catastróficas desta escala [KIT 2000]. Em função disto, fundou-se a *RoboCup Rescue simulation league*, onde são centralizados os esforços na busca deste sistema. Esta liga disponibiliza um simulador de desastres (terremotos) e operações de resgate [SKI 2006]. Neste simulador é possível avaliar a

[3] A numeração dos capítulos se refere à segunda edição.

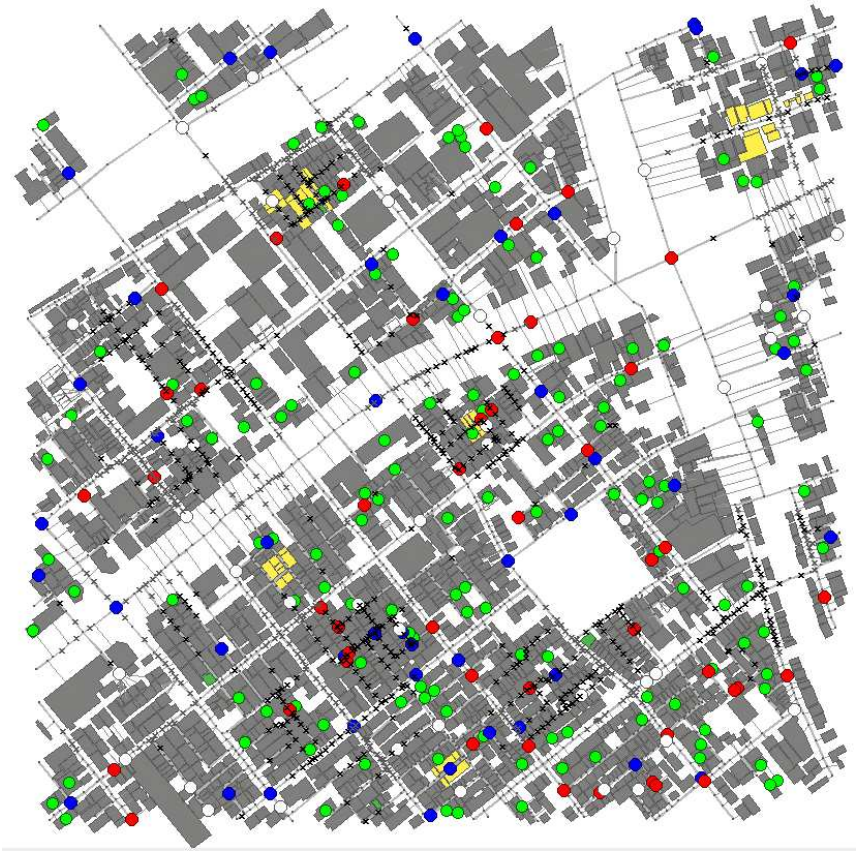

Figura 3.1: Mapa denominado $Kobe\_4$.

qualidade e eficiência de abordagens multiagente no que se refere ao salvamento de pessoas e minimização de danos. Questões como heterogeneidade, acesso limitado à informação, comunicação limitada e planejamento em tempo real caracterizam o ambiente *RoboCup Rescue* como um domínio multiagente complexo [KIT 2000]. A seguir é apresentada uma breve descrição do ambiente de simulação utilizado nas competições da *RoboCup Rescue*[4].

Como entrada, o simulador *RoboCup Rescue* recebe dados geográficos (mapas, ruas, construções, etc.) e informações sobre um terremoto. A partir destas informações, o simulador reproduz o colapso das construções, bloqueio de ruas, incêndios e civis soterrados e/ou feridos no instante imediatamente após a ocorrência do terremoto. Para tanto, o simulador dispõe de entidades que representam os

[4]Esta descrição refere-se ao simulador em uso até 2009; atualmente há uma nova versão deste simulador disponível no site `http://sourceforge.net/projects/roborescue/`, a qual mantem as principais características do anterior mas altera interfaces e modo de operação.

objetos presentes no cenário (agentes, edifícios, ruas, etc). Um exemplo deste mapeamento (no caso representando um dos mapas de Kobe) pode ser visto na Figure 3.1. O simulador reproduz a evolução da catástrofe ao longo de um intervalo de tempo. Esta evolução inclui, por exemplo, propagação de incêndios para construções inicialmente intactas e agravamentos no estado de saúde dos civis etc.

Para lidar com essa situação e minimizar os danos (humanos e materiais), o simulador incorpora alguns tipos de agentes, chamados *agentes de resgate*. Os agentes de resgate são subdivididos em duas classes: *agentes de campo*, que podem perceber e atuar no ambiente; e *agentes de central*, que possuem uma localização fixa e não percebem, diretamente, o ambiente (a percepção, neste caso, se resume a informações repassadas pelos agentes de campo). A Tabela 3.1 apresenta os agentes de resgate, seus papéis, e a respectiva classe a que pertencem.

Agentes de resgate	*Papel*	*Classe*
Brigada de Incêndio	combater incêndios em construções	campo
Posto de Bombeiros	centralizar a coordenação de Brigadas de Incêndio	central
Força Policial	remover escombros e bloqueios de ruas	campo
Delegacia de Polícia	centralizar a coordenação de Forças Policiais	central
Time de Ambulâncias	resgatar civis soterrados e/ou feridos	campo
Central de Ambulâncias	centralizar a coordenação de Times de Ambulâncias	central

Tabela 3.1: Agentes de resgate existentes no *RoboCup Rescue* com seus papéis e classes ([SAN 2009]).

A interação com o ambiente, através de percepção-ação, é feita exclusivamente por agentes de campo. O esquema desta percepção encontra-se na Figura 3.2 A percepção é visual, limitada a um raio de 10 metros. Cada agente pode perceber qualquer entidade localizada dentro de seu raio de visão (construções, bloqueios, civis, etc.). Com relação à ruas e nós que conectam ruas, por ser uma informação estática, são totalmente conhecidos pelo agente. O agente tem acesso aos atributos das entidades percebidas, como por exemplo o estado de uma rua (se livre ou bloqueada), a proporção de um incêndio, o nível de saúde de um civil, etc.

As ações que podem ser realizadas pelos agentes de campo no ambiente são apresentadas na Tabela 3.2. Agentes de central não podem atuar no ambiente, logo, não possuem nenhuma ação associada. É importante mencionar que cada agente pode realizar uma e somente uma ação em cada instante de tempo da simulação. A evolução da situação catastrófica torna o ambiente da *RoboCup Rescue* um ambiente dinâmico, limitando o tempo de decisão disponível aos agentes. Cada agente possui apenas frações de segundo para decidir que ação irá realizar em cada instante de tempo. Isto requer decisões rápidas e eficientes, pois se o agente não decide que

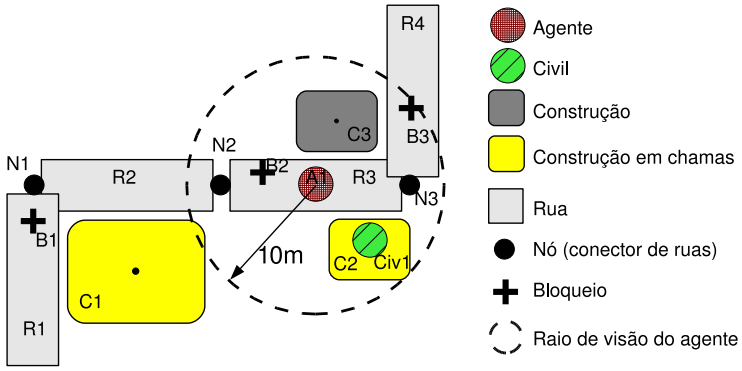

Figura 3.2: Esquema de percepção dos agentes no simulador *RoboCup Rescue* ([SAN 2009]).

Ação	*Efeito*	*Agente de campo*
Mover	Um caminho é percorrido e a localização do agente é alterada	Todos
Extinguir incêndio	Uma quantidade de água é despejada	Brigada de Incêndio
Remover bloqueio	Uma parcela de um bloqueio é removida	Força Policial
Resgatar civil	Uma parcela de escombros é removida de sobre um civil soterrado	Time de Ambulância
Carregar civil	Um civil é colocado na ambulância	Time de Ambulância
Descarregar civil	Um civil é retirado da ambulância	Time de Ambulância

Tabela 3.2: Agentes de campo da *RoboCup Rescue* e suas possíveis ações ([SAN 2009]).

ação irá executar neste intervalo de tempo, o simulador considera que este agente optou por não agir no respectivo instante de tempo.

A interação entre os diferentes tipos de agente é feita através de comunicação. No simulador da *RoboCup Rescue* os agentes podem se comunicar por voz ou por rádio. Em ambos os casos, há severas restrições de quantidade e tamanho das mensagens. Além disto, ruídos podem ocorrer na comunicação, fazendo com que mensagens sejam descartadas ou atrasadas. A seguir são detalhados estes dois tipos de comunicação (voz e rádio).

Mensagens de voz podem ser enviadas apenas por agentes de campo e só são recebidas por agentes de campo do mesmo tipo. Além disto, estas mensagens são de curto alcance, sendo recebidas por agentes localizados em um raio de no máximo 30 metros. Cada mensagem de voz tem o tamanho limitado a 256 *bytes*. Uma mensagens de voz não é endereçável, o que significa que não é possível especificar um

agente para recebê-la, pois todos os agentes localizados no mesmo raio de alcance irão recebê-la. Contudo, os agentes podem se recusar a receber mensagens de voz. Isto se dá exclusivamente com base no identificador do remetente, não no conteúdo das mensagens.

Mensagens de rádio podem ser enviadas e recebidas por qualquer tipo de agente. Estas mensagens são de longo alcance, ou seja, podem ser recebidas em qualquer parte do cenário. Com relação à quantidade, cada agente de campo pode enviar uma mensagem e receber uma mensagem. Cada mensagem de rádio também possui tamanho limitado a 256 *bytes*. Agentes de central podem enviar N mensagens e receber N mensagens, onde N é a quantidade de agentes coordenados pela central em questão. Assim como as mensagens de voz, as mensagens de rádio também não são endereçáveis. Contudo, podem ser definidos canais de rádio para comunicação. Estes canais podem ser utilizados para restringir o recebimento apenas aos agentes que estejam sintonizados no mesmo canal, não interferindo na comunicação dos demais.

Para medir o desempenho dos agentes, o simulador *RoboCup Rescue* define um escore, que leva em consideração a área total construída preservada (não incendiada) e a quantidade de sobreviventes, conforme a Equação 3.1, onde:

- P: quantidade total de agentes vivos;
- H: soma dos níveis de saúde (hp) dos agentes;
- $H_{inicial}$: soma dos níveis de saúde dos agentes no início da simulação;
- B: soma da área construída preservada;
- $B_{inicial}$: soma da área construída no início da simulação.

.

$$escore = \left(P + \frac{H}{H_{inicial}}\right) * \sqrt{\frac{B}{B_{inicial}}} \tag{3.1}$$

O escore não considera diretamente os bloqueios removidos das ruas (tarefas dos policiais). Entretanto, a remoção destes bloqueios afeta diretamente o escore, pois melhora o desempenho dos demais tipos de agentes.

De modo geral, o ambiente *RoboCup Rescue* apresenta as seguintes características:

- Parcialmente observável: A capacidade sensorial dos agentes não permite acesso à informação completa do ambiente. Do ponto de vista do sistema multiagente, o ambiente é coletivamente parcialmente observável, pois mesmo

unindo a percepção de todos os agentes, ainda assim não haverá percepção completa do ambiente.

- Estocástico: Existem incertezas no ambiente que podem afetar as ações dos agentes. Por exemplo, a simulação de incêndios apresenta certo nível de aleatoridade na propagação dos incêndios, podendo fazer com que ações de extinguir incêndio produzam diferentes resultados a partir da mesma quantidade de água despejada.

- Sequencial: As ações dos agentes influenciam as decisões futuras. Por exemplo, ao extinguir com sucesso um incêndio, o agente evita sua propagação nas construções vizinhas.

- Dinâmico: Alterações no ambiente são causadas por fatores outros que não as ações dos agentes. A propagação de incêndios é um exemplo. Além disto, estas alterações podem ocorrer enquanto os agentes estão deliberando, o que exige decisões rápidas por parte destes. Sem rapidez, as decisões podem se tornar rapidamente obsoletas.

- Contínuo: O ambiente apresenta um número infinito de estados possíveis.

Estas características tornam o ambiente *RoboCup Rescue* desafiador, tanto no que se refere às restrições impostas aos agentes a respeito do tempo de deliberação e quantidade de comunicação, quanto à extração de informações relevantes do ambiente.

3.6.2. Alocação de tarefas para coordenação em times

Em ambientes multiagente complexos como o da *RoboCup Rescue*, para que os agentes possam ter um bom desempenho é necessário que eles realizem tarefas em grupo e que cada elemento do grupo tenha uma tarefa para a qual ele é competente.

Como então alocar tarefas aos agentes, de maneira eficiente, em ambientes dinâmicos e de larga escala? Segundo [NAI 2002], o ambiente da *RoboCup Rescue* pode ser modelado como um E-GAP ou seja um GAP (*general assignment problem*) estendido [SCE 2005], onde há a necessidade de realizar as seguintes ações ou tarefas:

- Combate a incêndios: O objetivo é minimizar a área total destruída por incêndios. Tarefas de combate a incêndio devem ser realizadas por agentes brigada de incêndio. A realização desta tarefa consiste em despejar água no foco de incêndio. O alcance do jato de água de uma brigada de incêndio é limitado

a 30 metros. Portanto, é necessário que a brigada de incêndio esteja situada dentro deste raio de alcance do jato. Cada brigada de incêndio pode despejar até 1000 litros de água em cada instante da simulação. Quando a água do reservatório esgota, o agente deve se deslocar até um refúgio para reabastecer.

- Remoção de bloqueios de ruas: O objetivo é minimizar a quantidade de ruas obstruídas para melhorar o fluxo de brigadas de incêndio e times de ambulâncias. Tarefas de remoção de bloqueios devem ser realizadas por agentes força policial. Para realizar a tarefa, a força policial deve estar situada sobre a rua bloqueada. A parcela de bloqueio removida por cada força policial depende das dimensões do bloqueio.

- Resgate de civis feridos: O objetivo é maximizar a quantidade de sobreviventes. Tarefas de resgate de civis devem ser realizadas por agentes do tipo ambulância. Para realizar esta tarefa, a ambulância deve estar situada exatamente na mesma posição do civil ferido. Adicionalmente, a realização desta tarefa é feita em duas etapas: remoção de escombros sobre o civil (caso esteja soterrado); e transportar o civil para um refúgio (caso requeira tratamento médico).

Dependendo das proporções de uma das tarefas acima descritas, um único agente pode não ser capaz de realizá-la eficientemente. No caso de um incêndio em estado avançado ou em uma grande construção, uma única brigada não será suficiente para controlar o incêndio. Já um bloqueio de rua será eliminado mais rapidamente se diversas forças policiais removerem várias parcelas simultaneamente. Por fim, vários times de ambulâncias atuando em conjunto removerão mais rapidamente os escombros sobre um civil soterrado. Em função disto, espera-se que mobilizar simultaneamente uma maior quantidade de agentes para determinadas tarefas implique em uma melhora no desempenho. Para realizar esta mobilização simultânea, adota-se a decomposição da tarefa em subtarefas inter-relacionadas por *and*, que são realizadas simultaneamente pelos agentes.

Além da identificação dos agentes e das tarefas, o modelo E-GAP requer que sejam definidas as competências dos agentes. Estas competências são definidas em função do papel de cada agente de campo.

Scerri e co-autores [SCE 2005] denominam este tipo de ambiente de *extreme teams*. A alocação de tarefas em *extreme teams* está associada a quatro características: (i) ambientes dinâmicos, onde tarefas podem aparecer e desaparecer; (ii) agentes podem realizar múltiplas tarefas, dados os recursos disponíveis; (iii) agentes podem possuir funcionalidades sobrepostas, estando aptos a realizar várias tarefas mas com diferentes níveis de competência; e (iv) podem haver tarefas que requerem esforço conjunto e simultâneo de um grupo de agentes.

O E-GAP é um modelo utilizado para formalizar a alocação de tarefas em *extreme teams*. Um E-GAP é composto por um conjunto $\mathcal{J}$ de tarefas a serem realizadas por um conjunto $\mathcal{I}$ de agentes. Cada agente $i \in \mathcal{I}$ possui uma competência para realizar cada tarefa $j \in \mathcal{J}$, denotada por $Cap(i, j) \rightarrow [0, 1]$. Cada agente i também possui uma quantidade limitada de recursos $i.res$ e despende $Res(i, j)$ unidades de recurso ao realizar a tarefa j. Uma matriz M é usada para representar a alocação, onde m_{ij} é 1 se o agente i realiza a tarefa j, ou 0 caso contrário. O objetivo é encontrar M que maximiza a recompensa do sistema, que é determinada pelas competências dos agentes que participam da alocação. Esta recompensa define portanto a qualidade de uma alocação no E-GAP. Além disto, a alocação M deve respeitar os limites impostos pelos recursos dos agentes, e cada tarefa deve ser alocada por no máximo um agente.

Tarefas que requerem esforço conjunto e simultâneo são representadas no modelo E-GAP através de inter-relacionamentos do tipo *and* lógico. De maneira geral, relações *and* podem ser vistas como a decomposição de uma grande tarefa em subtarefas que devem ser realizadas simultaneamente. A realização de apenas algumas subtarefas não implica na realização da grande tarefa, desperdiçando os recursos dos agentes e não contribuindo para o desempenho do sistema.

Para formalizar relações *and* entre tarefas, o modelo E-GAP define um conjunto $\bowtie = \{\alpha_1, \ldots, \alpha_p\}$ que contém p conjuntos α de tarefas relacionadas por *and*, na forma $\alpha_k = \{j_1 \wedge \ldots \wedge j_q\}$. A quantidade de tarefas x_k de um conjunto α_k que estão simultaneamente alocadas é dado por $x_k = \sum_{i \in \mathcal{I}} \sum_{j \in \alpha_k} m_{ij}$.

Um modelo baseado no E-GAP pode ser convertido em um problema de otimização de restrições, na sua variante distribuída, ou seja um DCOP (Seção 3.4.). Para isto as variáveis do problema são designadas para agentes específicos, enquanto que as tarefas constituem os valores do domínio. Entretanto, esta conversão implica um grafo de restrições denso. Para resolver este problema Scerri e colegas propuseram um algoritmo aproximado denominado Low-communication Approximate DCOP (LA-DCOP) [SCE 2005] que usa um protocolo baseado em token. Os agentes que percebem uma tarefa no ambiente, criam um token para representa-la caso não a realizem. Eventualmente este token é passado a outro agente que avalia se pode ou não realiza-la e assim por diante.

O E-GAP foi a base para diversas extensões realizadas com vistas a: i) melhorar o processo de otimização através de uma abordagem heurística; ii) melhorar a eficiência da realização das tarefas *and*; iii) melhorar a formação de grupos e a alocação de tarefas em geral. Estas extensões são discutidas abaixo.

Ferreira Jr. e colegas propuseram a abordagem Swarm-GAP visando a substituição do processo de otimização da recompensa do E-GAP proposto em [SCE 2005], o qual é computacionalmente caro, por um mecanismo baseado em inteligência de enxames conforme proposto em [FER 2008, FER 2010]. Insetos sociais (e.g. for-

migas) possuem as características de *extreme teams*. Apesar da simplicidade, a natureza dotou-os com as competências necessárias para atuarem efetivamente em *extreme teams*. Para realizar as tarefas relacionadas com a sobrevivência da colônia, os insetos sociais adotam um modelo de divisão de trabalho. Este modelo, formalizado por Theraulaz *et al.* [THE 98], não requer que os indivíduos possuam informação completa a respeito do ambiente e nem que existam indivíduos líderes. Swarm-GAP se assemelha ao LA-DCOP na medida em que também é baseado no E-GAP, é aproximado, usa tokens para comunicação entre os agentes e lida com *extreme teams*. Um agente no Swarm-GAP decide se vai ou não realizar uma tarefa com base no modelo de divisão de trabalho mencionado acima. Um parâmetro-chave nesta abordagem é o estímulo s que cada agente tem em relação às tarefas. Isto possibilita o agente ser seletivo. Por exemplo um bombeiro tem maior estímulo para realizar tarefas do tipo combate à incêndio, sendo mais seletivo em relação às outras.

O modelo baseado em inteligência de enxames foi posteriormente estendido em [SAN 2009, SAN 2009a]. A realização de tarefas que requerem esforço simultâneo é observada em algumas espécies de formigas. A tarefa em questão é o transporte de grandes presas. Ao invés de dividir em partes e transportá-las individualmente, algumas espécies formam grupos que transportam a presa cooperativamente. Estes grupos são formados através de um processo chamado recrutamento [HÖL 78].

Para abordar a questão de formação de grupos, em [SAN 2009b] e [SAN 2009c] é proposto um mecanismo de agrupamento (clustering) baseado em inteligência de enxame, denominado *bee clustering*, o qual tem como objetivo formar, de maneira distribuída, grupos de dados com características similares sem qualquer informação inicial relacionada ao resultado desejado. Um dos cenários utilizado para se verificar a eficácia do algoritmo *bee clustering* foi o da *RoboCup Rescue* visando investigar o desempenho do algoritmo na formação de grupos de agentes para realização de tarefas conjuntas. Deste trabalho concluiu-se que a aplicação do *bee clustering* obtém melhores pontuações do que um algoritmo guloso.

Ainda com vistas à formação de grupos, em [EPS 2011] foi utilizado o formalismo de coalizões. Uma vez que encontrar a estrutura ótima de coalizões é um problema equivalente à partição de conjuntos, é necessário um algoritmo extremamente eficiente (dada a restrição de tempo real do ambiente *RoboCup Rescue*). Neste trabalho foi usada uma abordagem heurística e anytime que visa reduzir o número de possibilidades de coalizões através da introdução de restrições. Um exemplo típico é que uma coalizão de bombeiros não deve ter mais que n deles, onde n é calculado em função do tamanho do incêndio.

Em resumo, nesta seção foram apresentadas diversas abordagens que visam primordialmente resolver o problema de alocação de diversos tipos de tarefas (des-

bloqueio de ruas, resgate de civis, combate a incêndios) entre os diversos tipos de agentes (forças policiais, paramédicos, brigadas de incêndio). As abordagens apresentadas têm como característica comum o fato de serem baseadas no formalismo E-GAP e/ou em métodos que permitem a independência do domínio. Desta forma, estes métodos não utilizam informações específicas como um mapa em particular, ou localização física dos agentes, ou ainda características particulares das tarefas.

Em oposição a este tipo de abordagem, a literatura também reporta métodos utilizados pelos campeões anteriores como [PAQ 2006, YIK 2008, HAR 2008]. Estes focalizam diversos métodos ad-hoc, dependente de domínio que foram ou são eficientes mas que são de difícil generalização. Idealmente deve-se procurar usar uma base independente de domínio, acrescida de algum grau de programação ad-hoc visando melhorar o escore em competição.

3.7. Conclusão

A área de sistemas multiagentes vem colocando desafios crescentes à IA na medida em que exige uma revisão e/ou extensão das técnicas clássicas, como por exemplo aquelas ligadas à representação de conhecimento, resolução de problemas e aprendizado.

Este texto, longe de ser extensivo em relação a todos os conceitos relativos àquela área, introduz e discute alguns aspectos ligados aos desafios que foram colocados à IA a partir da década de 80 quando do surgimento da IA distribuída, os quais basicamente se referem à existência de várias entidades ou agentes em um ambiente, entidades estas que precisam se coordenar a fim de atingir seus objetivos de forma eficiente.

Posteriormente, este texto discutiu uma aplicação desafiadora, especialmente no que se refere à questão de como coordenar times de agentes: a liga *RoboCup Rescue*, surgida para desafiar a comunidade de sistemas multiagentes a formular novas soluções para problemas de alocação de tarefas e outros que surgem em situações de emergência.

Um dos problemas em aberto é a questão da eficiência dos métodos de otimização apresentados na seção 3.6. (especialmente os não heurísticos como o E-GAP). Para melhorar sua eficiência, uma das possibilidades é o uso de computação distribuída e de alto desempenho como por exemplo paralelização dos algoritmos de busca.

Agradecimentos

Agradeço o apoio das agências financiadoras dos projetos de pesquisa que permitiram aprofundar-me nos temas aqui abordados. Sem o suporte destas agências este trabalho teria sido impossível. Desta forma agradeço ao CNPq pelo apoio aos projetos de pesquisa, bem como ao programa de bolsas de pesquisa e bolsas de pós graduação de diversos orientandos. Agradeço ainda à CAPES e à Fundação Alexander von Humboldt pelo apoio ao projeto de cooperação internacional com a Alemanha e à bolsa de pós doutoramento, respectivamente. Não menos importante, agradeço aos meus ex-alunos que contribuíram com suas pesquisas em algumas das áreas abordadas neste texto, notadamente Bruno Castro da Silva, Daniel Epstein, Daniela Scherer dos Santos, Denise de Oliveira, Fernando dos Santos e Paulo Roberto Ferreira Jr. cujas dissertações e teses são próximas ao presente material.

3.8. Bibliografia

[BAZ 2010] BAZZAN, A. L. C. Sistemas multiagentes: introdução e aplicações em simulação e controle de tráfego e simulação de situações de emergência. **Revista de Sistemas de Informação da FSMA**, v.6, p.12–41, 2010, (http://www.fsma.edu.br/si/edicao6/FSMA_SI_2010_2_Principal_3.html).

[BAZ 2010a] BAZZAN, A. L. C. IA multiagente: mais inteligência, mais desafios. In: MEIRA JR., W.; CARVALHO, A. C. P. L. F. de (Eds.). **Atualizações em informática 2010**. Rio de Janeiro: PUC-Rio, 2010. p.111–159.

[BIT 2001] BITTENCOURT, G. **Inteligência artificial**: ferramentas e teorias. 2a..ed. Florianópolis: Editora da UFSC, 2001.

[BON 88] BOND, A. H.; GASSER, L. Readings in distributed artificial intelligence. In: **Readings in distributed artificial intelligence**. San Mateo, California: Morgan Kaufmann, 1988.

[BOR 2007] BORDINI, R. H.; HÜBNER, J. F.; WOOLDRIDGE, M. **Programming multi-agent systems in agentspeak using *jason***. [S.l.]: John Wiley & Sons, 2007. (Wiley Series in Agent Technology).

[BRO 86] BROOKS, R. A robust layered control system for a mobile robot. **Robotics and Automation, IEEE Journal of**, v.2, n.1, p.14–23, January 1986.

[EPS 2011] EPSTEIN, D.; BAZZAN, A. L. C. Dealing with coalition formation in the RoboCup Rescue: an heuristic approach. In: INTERNATIONAL CONFERENCE ON AGENTS AND ARTIFICIAL INTELLIGENCE, 3., 2011, Roma. **Proceedings...** [S.l.: s.n.], 2011. v.2, p.717–720.

[FER 2010] FERREIRA JR., P. R. et al. Robocup rescue as multiagent task allocation among teams: experiments with task interdependencies. **Journal of Autonomous Agents and Multiagent Systems**, v.20, n.3, p.421–443, May 2010.

[FER 2008] FERREIRA JR., P. R. **Coordenação de sistemas multiagente atuando em cenários complexos**: uma abordagem baseada na divisão de trabalho dos insetos sociais. 2008. Tese (Doutorado em Ciência da Computação) — Universidade Federal do Rio Grande do Sul.

[GAS 90] GASSER, L.; HUHNS, M. N. (Eds.). **Distributed artificial intelligence**: vol. 2. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990.

[HAR 2008] HARA, T.; TORIUMU, F. **Robocup rescue 2008 repository - SUNTORI team**.

[HÖL 78] HÖLLDOBLER, B.; STANTON, R. C.; MARKL, H. Recruitment and food-retrieving behavior in *Novomessor* (formicidae, hymenoptera). **Behavioral Ecology and Sociobiology**, v.4, n.2, p.163–181, 1978.

[JEN 96] JENNINGS, N. R. Coordination techniques for distributed artificial intelligence. In: O'HARE, G. M. P.; JENNINGS, N. R. (Eds.). **Foundations of distributed artificial intelligence**. New York: John Wiley & Sons, 1996. p.187–210.

[KIT 2000] KITANO, H. Robocup rescue: a grand challenge for multi-agent systems. In: INTERNATIONAL CONFERENCE ON MULTIAGENT SYSTEMS, 4., 2000, Boston, USA. **Proceedings...** Los Alamitos: IEEE Computer Society, 2000. p.5–12.

[KUM 92] KUMAR, V. Algorithms for constraints satisfaction problems: a survey. **AI Magazine**, v.13, n.1, p.32–44, 1992.

[MAI 2004] MAILLER, R.; LESSER, V. Solving distributed constraint optimization problems using cooperative mediation. In: INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS, 3., 2004, New York. **Proceedings...** New York: IEEE Computer Society, 2004. p.438–445.

[MAL 94] MALONE, T.; CROWSTON, K. The interdisciplinary study of coordination. **ACM Computing Surveys**, v.26, n.1, p.87–119, 1994.

[MOD 2003] MODI, P. J. et al. An asynchronous complete method for distributed constraint optimization. In: SECOND INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS, 2003, New York, USA. **Proceedings...** ACM Press, 2003. p.161–168.

[MOD 2005] MODI, P. J. et al. ADOPT: asynchronous distributed constraint optimization with quality guarantees. **Artificial Intelligence**, v.161, p.149–180, January 2005.

[NAI 2002] NAIR, R. et al. Task allocation in the rescue simulation domain: a short note. In: BIRK, A.; CORADESCHI, S. (Eds.). **Robocup 2001**: robot soccer world cup V. Berlin: Springer-Verlag, 2002. p.751–754. (Lecture Notes in Computer Science, v.2377).

[PAQ 2006] PAQUET, S.; Chaib-draa, B. Learning the required number of agents for complex tasks. In: FIFTH INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS, 2006, Hakodate, Japan. **Proceedings...** New York: ACM Press, 2006. p.736–746.

[PET 2005] PETCU, A.; FALTINGS, B. A scalable method for multiagent constraint optimization. In: NINETEENTH INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 2005, Edinburgh, Scotland. **Proceedings...** Professional Book Center, 2005. p.266–271.

[ROS 94] ROSENSCHEIN, J.; ZLOTKIN, G. **Rules of encounter**. Cambridge (MA): The MIT Press, 1994.

[RUS 2004] RUSSELL, S.; NORVIG, P. **Inteligência artificial**. Rio de Janeiro, RJ: Campus, 2004. 1021p. Tradução da segunda edição.

[SAN 2009c] SANTOS, D. S. d.; BAZZAN, A. L. C. A biologically-inspired distributed clustering algorithm. In: IEEE SWARM INTELLIGENCE SYMPOSIUM, 2009., 2009, Nashville. **Proceedings...** IEEE, 2009. p.160–167.

[SAN 2009b] SANTOS, D. S. d. **Bee clustering**: um algoritmo para agrupamento de dados inspirado em inteligência de enxames. 2009. Dissertação (Mestrado em Ciência da Computação) — PPGC-UFRGS, Porto alegre.

[SAN 2009a] SANTOS, F. d.; BAZZAN, A. L. C. eXtreme-ants: ant based algorithm for task allocation in extreme teams. In: SECOND INTERNATIONAL WORKSHOP ON OPTIMISATION IN MULTI-AGENT SYSTEMS, 2009, Budapest, Hungary. **Proceedings...** [S.l.: s.n.], 2009. p.1–8.

[SAN 2009] SANTOS, F. dos. **eXtreme-ants**: algoritmo inspirado em formigas para alocação de tarefas em extreme teams. 2009. Dissertação (Mestrado em Ciência da Computação) — Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.

[SCE 2005] SCERRI, P. et al. Allocating tasks in extreme teams. In: FOURTH INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS, 2005, New York, USA. **Proceedings...** ACM Press, 2005. p.727–734.

[SHO 2009] SHOHAM, Y.; LEYTON-BROWN, K. **Multiagent systems**: algorithmic, game-theoretic, and logical foundations. [S.l.]: Cambridge University Press, 2009. 483p.

[SKI 2006] SKINNER, C.; BARLEY, M. Robocup rescue simulation competition: status report. In: BREDENFELD, A. et al. (Eds.). **Robocup 2005**: robot soccer world cup IX. Berlin: Springer-Verlag, 2006. p.632–639. (Lecture Notes in Computer Science, v.4020).

[THE 98] THERAULAZ, G.; BONABEAU, E.; DENEUBOURG, J. Response threshold reinforcement and division of labour in insect societies. In: ROYAL SOCIETY OF LONDON SERIES B - BIO-

LOGICAL SCIENCES, 1998. **Anais...** [S.l.: s.n.], 1998. v.265, p.327–332.

[WEI 99] WEISS, G. **Multiagent systems - a modern approach to distributed artificial intelligence**. Cambridge, MA: The MIT Press, 1999.

[WOO 2002] WOOLDRIDGE, M. J. **An introduction to multiagent systems**. Chichester: John Wiley & Sons, 2002.

[WOO 2009] WOOLDRIDGE, M. J. **An introduction to multiagent systems**. Chichester: John Wiley & Sons, 2009. 461p. Second edition.

[YIK 2008] YIKUN, T. et al. **Robocup rescue 2008 repository - ZJUBase team**.

[YOK 92] YOKOO, M. et al. Distributed constraint satisfaction for formalizing distributed problem solving. In: INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS, 12., 1992. **Proceedings...** [S.l.: s.n.], 1992. p.614–621.