UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

FERNANDO DOS SANTOS

eXtreme-Ants: Algoritmo Inspirado em Formigas para Alocação de Tarefas em *Extreme Teams*

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof. Dr. Ana Lúcia Cetertich Bazzan
Orientador

Porto Alegre, maio de 2009

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Santos, Fernando dos

eXtreme-Ants: Algoritmo Inspirado em Formigas para Alocação de Tarefas em *Extreme Teams* / Fernando dos Santos. – Porto Alegre: PPGC da UFRGS, 2009.

69 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2009. Orientador: Ana Lúcia Cetertich Bazzan.

1. Sistemas multiagente. 2. Alocação de tarefas. 3. Insetos sociais. I. Bazzan, Ana Lúcia Cetertich. II. Título.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

Adopt-ng	*Asynchronous Distributed Optimization with Valued Nogoods*
Adopt	*Asynchronous Distributed Optimization*
AFB	*Asynchronous Forward Bounding*
BnB-Adopt	*Branch and Bound Asynchronous Distributed Optimization*
DPOP	*Distributed Pseudotree Optimization*
IA	Inteligência Artificial
IAD	Inteligência Artificial Distribuída
LA-DCOP	*Low-communication Approximate DCOP*
OptAPO	*Optimal Asynchronous Partial Overlay*
CNP	*Contract-Net Protocol*
GAP	*Generalized Assignment Problem*
E-GAP	*Extended Generalized Assignment Problem*
PMB	Problema da Mochila Binário
DCOP	*Distributed Constraint Optimization Problem*
RCA	Recrutamento de Curto Alcance
RDP	Resolução Distribuída de Problemas
RLA	Recrutamento de Longo Alcance
RRSL	*RoboCup Rescue Simulation League*
SMA	Sistema Multiagente

LISTA DE SÍMBOLOS

α	conjunto de tarefas inter-relacionadas
$Cap(i, j)$	competência do agente i para realizar tarefa j
i	agente
$\mathcal{I}$	conjunto de agentes
$i.res$	recursos disponíveis no agente i
j	tarefa
$\mathcal{J}$	conjunto de tarefas
M	matriz de alocação
m_{ij}	valor da i-ésima linha e j-ésima coluna da matriz de alocação M
$Res(i, j)$	recursos consumidos do agente i ao realizar a tarefa j
s_j	estímulo associado à tarefa j
t	instante de tempo
$T_{ij}(s_j)$	tendência do agente i realizar a tarefa j dado s_j
θ_{ij}	limiar de resposta do agente i para a tarefa j
$Val(i, j, \bowtie)$	desempenho de i ao realizar j dadas as tarefas de $\bowtie$
x_α	quantidade de tarefas de um conjunto α simultaneamente alocadas
$\bowtie$	conjunto de todos os conjuntos de tarefas inter-relacionadas α
$\lvert \cdot \rvert$	cardinalidade de um conjunto ou vetor

LISTA DE FIGURAS

RESUMO

Sistemas multiagente são construídos para atingir objetivos complexos e abrangentes, que estão além da capacidade de um único agente. Estes objetivos podem ser representados através de tarefas, que devem ser realizadas pelos agentes de forma a otimizar o desempenho do sistema.

Em muitos ambientes reais, a escala do problema envolve tanto uma grande quantidade de agentes, quanto uma grande quantidade de tarefas. Além disto, os agentes devem lidar com informações incompletas, realizando tarefas em tempo hábil. O termo *extreme teams* foi introduzido na literatura para designar as seguintes quatro características da alocação de tarefas: oa ambientes são dinâmicos; os agentes podem realizar múltiplas tarefas; os agentes podem possuir funcionalidades sobrepostas; e podem existir inter-relacionamentos entre tarefas, impondo, por exemplo, necessidade de realização simultânea. Abordagens existentes na literatura tratam, efetivamente, apenas as três primeiras características de *extreme teams*.

Esta dissertação apresenta um algoritmo para alocação de tarefas, chamado eXtreme-Ants, que trata todas as quatro características de *extreme teams*. O algoritmo é inspirado no sucesso ecológico dos insetos sociais, e utiliza as metáforas de divisão de trabalho e recrutamento para transporte cooperativo. A metáfora de divisão de trabalho proporciona decisões rápidas e eficientes, atendendo as três primeiras características de *extreme teams*. O recrutamento permite formar grupos de agentes comprometidos com a realização simultânea de tarefas que exigem esforço conjunto, atendendo a quarta característica: inter-relacionamentos entre tarefas. Com isto, concretiza-se de fato o conceito completo de *extreme teams*.

Experimentos foram realizados em dois ambientes distintos: um simulador independente de domínio e o simulador RoboCup Rescue. Os resultados obtidos demonstraram que a eficiência do eXtreme-Ants é balanceada com relação ao desempenho, quantidade de comunicação e esforço computacional.

Palavras-chave: Sistemas multiagente, Alocação de tarefas, Insetos sociais.

eXtreme-Ants: Ant Based Algorithm for Task Allocation in Extreme Teams

ABSTRACT

Multiagent systems aim at achieving complex and broad goals, which are beyond the capability of a single agent. These goals can be represented by tasks, which must be performed by the agents in order to optimize the performance of the system.

In many real-world environments, the scale of problems involves both a large number of agents and a large number of tasks. Besides, the agents must reason with incomplete and uncertain information, in a timely fashion. The expression extreme teams was introduced in the literature to describe the following four characteristics regarding task allocation: dynamic environments; agents may perform multiple tasks; agents can have overlapping functionality; and inter-task constraints (such as simultaneous execution requirements) may be present. Existing approaches effectively deal with just the three first characteristics of extreme teams.

This dissertation presents an algorithm for allocating tasks to agents, called eXtreme-Ants, which deals with all the four characteristics of extreme teams. The algorithm is inspired in the ecological success of social insects, and uses the metaphors of division of labor and recruitment for cooperative transport. The metaphor provides fast and efficient decision-making, complying to the first three characteristics. The recruitment ensures the formation of groups of agents committed to the simultaneous execution of tasks that require joint efforts, complying to the fourth characteristic: inter-task constraints. Thus, the full concept of extreme teams is indeed realized.

Experiments were performed in two distict environments: a domain independent simulator, and the RoboCup Rescue simulator. The results shown that eXtreme-Ants achieves a balanced efficiency regarding performance, communication, and computational effort.

Keywords: Multiagent systems, Task allocation, Social insects.

1 INTRODUÇÃO

Um agente pode ser definido como um sistema computacional que possui certa capacidade de percepção e ação no ambiente em que está situado. Agentes são construídos com o propósito de atingir um objetivo. Contudo, objetivos complexos e abrangentes podem estar além da capacidade de um único agente. Nestes casos o uso de um Sistema Multiagente (SMA) surge como alternativa para atingir estes objetivos. Em um SMA, vários agentes encontram-se no ambiente e devem interagir para que os objetivos, sejam eles individuais (de cada agente) ou coletivos (do sistema) sejam atingidos.

Uma das formas de definir os objetivos de um agente é através da especificação de tarefas, criando um ambiente orientado à tarefas. O objetivo do agente passa a ser realizar as tarefas presentes no ambiente. O sucesso de um agente na realização das tarefas é definido por uma medida de desempenho. Em um SMA formado por agentes que realizam tarefas, a medida de desempenho leva em consideração todos os agentes, indicando o sucesso do sistema como um todo. Determinar qual agente irá realizar qual tarefa de forma a otimizar o desempenho é denominado problema de alocação de tarefas.

Em muitos ambientes reais, a escala do problema envolve uma grande quantidade de tarefas, requerendo SMAs de larga escala, com centenas de agentes. Estes ambientes são parcialmente observáveis, pois a percepção dos agentes com relação às tarefas disponíveis é restrita. Também podem haver restrições sobre as interações que necessitam de um canal de comunicação entre os agentes, limitando a troca e centralização de informações.

Scerri et al. (2005) introduziram o termo *extreme teams*[1], que está relacionado com as seguintes características da alocação de tarefas nestes ambientes:

(i). O ambiente é dinâmico, isto é, tarefas podem aparecer e desaparecer;

(ii). Os agentes podem realizar múltiplas tarefas, dados os recursos disponíveis;

(iii). Vários agentes podem possuir funcionalidades sobrepostas (são heterogêneos);

(iv). Podem existir inter-relacionamentos entre tarefas.

O *Generalized Assignment Problem* (GAP)[2] e sua extensão, E-GAP, são modelos que podem ser utilizados para formalizar aspectos da alocação de tarefas a agentes em *extreme teams*. Estes modelos representam, de forma precisa, todas as quatro características de *extreme teams*. A solução de um GAP (e também de um E-GAP) é a alocação que maximiza as competências dos agentes que realizam as tarefas.

[1]Uma tradução para este termo é "times extremos". Entretanto, para manter compatibilidade com a literatura existente, adotou-se o original em inglês

[2]Problema Generalizado de Atribuição

O primeiro passo em direção a concretização do conceito de *extreme teams* foi dado por Scerri et al. (2005), propondo um algoritmo para resolver E-GAPs de maneira aproximada. O algoritmo, chamado *Low-communication Approximate DCOP* (LA-DCOP), utiliza *tokens* para otimizar a comunicação entre os agentes. Entretanto, o LA-DCOP foi testado por seus autores em um ambiente com apenas as características (i) e (ii).

Outro algoritmo que resolve E-GAPs de forma aproximada é o Swarm-GAP (FERREIRA JR.; BOFFO; BAZZAN, 2008). Este algoritmo é baseado na metáfora de divisão de trabalho em insetos sociais. Ferreira Jr. et al. (2009) deram um passo adiante na concretização de *extreme teams*, avaliando seu algoritmo Swarm-GAP em um ambiente com as características (i), (ii) e (iii). Além disto, Ferreira Jr (2008) também apresenta uma abordagem para modelar E-GAPs em TÆMS[3], demonstrando a equivalência entre as representações E-GAP e TÆMS no caso de *extreme teams*. Como pode ser visto, as abordagens existentes até o momento para alocação de tarefas tratam, efetivamente, apenas as três primeiras características envolvidas no conceito de *extreme teams*.

A partir disto, a questão tratada nesta dissertação é:

Como alocar tarefas em *extreme teams* de forma eficiente, em ambientes com restrições de tempo e comunicação, considerando também a característica (iv), isto é, a existência de tarefas inter-relacionadas que requerem esforço simultâneo de um grupo de agentes?

A abordagem proposta é inspirada no sucesso ecológico das colônias de insetos sociais. Uma colônia de insetos sociais com centenas de milhares de membros, apresenta as características de *extreme teams*, operando sem a existência de coordenação explicita. A colônia responde eficientemente a variações de um ambiente dinâmico que ocorrem, por exemplo, em função da época do ano, condições climáticas e disponibilidade de alimentos. Um indivíduo membro da colônia não tem acesso às necessidades e tarefas da colônia como um todo, mas apenas à informação local e simplificada. As interações entre os indivíduos da colônia ocorrem por meios restritos, normalmente através de estigmergia, ou seja, comunicação através do ambiente. Nenhum indivíduo está encarregado de centralizar informações e coordenar os demais. Uma colônia de insetos sociais apresenta princípios da auto-organização, isto é, o comportamento coletivo e complexo emerge a partir de interações entre insetos que exibem comportamento relativamente simples.

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, Bonabeau e Deneubourg (1998), é baseado em limiares de resposta associados aos indivíduos, e em estímulos às tarefas. Os indivíduos que compõem a colônia são heterogêneos. De acordo com sua morfologia ou idade, cada indivíduo é especializado em determinado tipo de tarefa. Esta especialização define o limiar de resposta para a realização das tarefas. Contudo, a divisão de trabalho na colônia apresenta plasticidade, adaptando-se às necessidades da colônia. Esta plasticidade origina-se de uma flexibilidade comportamental dos indivíduos que, apesar de serem especializados, podem se engajar em diferentes tarefas.

Certas espécies de formigas realizam um tipo de tarefa que requer o esforço simultâneo de um grupo de indivíduos para ser realizada com sucesso. Trata-se do transporte cooperativo de grandes presas. O transporte cooperativo, que envolve duas ou mais formigas, tem como objetivo maximizar o compromisso entre a energia adquirida em forma

[3] O TÆMS (DECKER; LESSER, 1993) é uma linguagem concebida para descrever a estrutura de tarefas dos agentes, de forma similar ao modelo E-GAP. Para manter a compatibilidade com a terminologia recente de *extreme teams* adotou-se neste trabalho o modelo E-GAP.

de alimento e a energia empregada no transporte. Estes grupos são formados através de um processo chamado recrutamento (HÖLLDOBLER; STANTON; MARKL, 1978).

1.1 Contribuição

A contribuição desta dissertação é um algoritmo para alocação de tarefas, denominado eXtreme-Ants (SANTOS; BAZZAN, 2009a,b,c). O algoritmo resolve E-GAPs de forma aproximada, sendo inspirado na divisão de trabalho existente em insetos sociais e no processo de recrutamento para transporte cooperativo presente em algumas espécies de formigas. Através do modelo de divisão de trabalho, cada agente decide quais tarefas irá realizar, levando em consideração suas competências e recursos disponíveis. Ao perceber tarefas inter-relacionadas que requerem esforço simultâneo de um grupo de agentes, o agente reproduz o processo de recrutamento. A partir do recrutamento é formado um grupo de agentes comprometidos com a execução simultânea das tarefas. Agentes executando o eXtreme-Ants são eficientes para atuarem em *extreme teams* pois requerem pouca comunicação e pouco esforço computacional.

A eficiência do eXtreme-Ants foi avaliada através de uma série de experimentos em dois ambientes distintos: um simulador independente de domínio e o simulador RoboCup Rescue. Nestes ambientes, o eXtreme-Ants foi comparado com outros dois algoritmos para alocação de tarefas em *extreme teams*: Swarm-GAP e LA-DCOP. As métricas consideradas foram o desempenho em relação às alocações realizadas, a quantidade de comunicação utilizada e o esforço computacional requerido.

No simulador independente, o desempenho do eXtreme-Ants é próximo do LA-DCOP, mas requer menor quantidade de comunicação e esforço computacional. Com relação ao Swarm-GAP, o desempenho do eXtreme-Ants é superior, particularmente na presença de tarefas inter-relacionadas que requerem esforço simultâneo de um grupo de agentes, característica (iv) de *extreme teams*.

No RoboCup Rescue o desempenho do eXtreme-Ants é superior ao Swarm-GAP e LA-DCOP, com menor quantidade de comunicação necessária. Os esforços computacionais estão relacionados com as complexidades de cada algoritmo, sendo que o eXtreme-Ants é inferior ao LA-DCOP e superior ao Swarm-GAP.

1.2 Organização dos capítulos

O texto desta dissertação está organizado da seguinte forma:

Capítulo 2. Introduz conceitos relacionados à área de SMA e alocação de tarefas, e os modelos GAP e E-GAP para formalização da alocação de tarefas em *extreme teams*.

Capítulo 3. Apresenta aspectos acerca dos insetos sociais relevantes a esta dissertação. São apresentados o modelo de divisão de trabalho em colônias de insetos sociais e o processo de recrutamento para transporte cooperativo.

Capítulo 4. Relaciona possíveis abordagens para alocação de tarefas em SMAs e implicações de utilização no cenário específico de *extreme teams*.

Capítulo 5. Apresenta o algoritmo eXtreme-Ants.

Capítulo 6. Investiga o desempenho do eXtreme-Ants através de uma série de experimentos em dois ambientes distintos: um simulador independente de domínio e o simulador RoboCup Rescue.

Capítulo 7. Aponta as conclusões desta dissertação e perspectivas para trabalhos futuros.

2 AGENTES, SISTEMAS MULTIAGENTE E ALOCAÇÃO DE TAREFAS

Este capítulo apresenta conceitos relacionados à área de SMA e que fundamentam este trabalho. Inicia-se com a definição de agente para posteriormente apresentar SMAs. Em seguida, é apresentada a questão da alocação de tarefas em SMA, e os modelos GAP e E-GAP para formalização da alocação de tarefas.

2.1 Agentes e Sistemas Multiagente

Segundo Wooldridge (2002), "Um agente é um sistema computacional que está situado em um ambiente e é capaz de agir, de forma autônoma, para atingir seus objetivos". Outras definições de agente podem ser encontradas na literatura, visto que, devido a variedade de aplicações para agentes, não há um consenso definido. Contudo, esta definição é adequada e suficiente para este trabalho, sendo adotada de agora em diante.

O fato de estar situado significa que o agente recebe informações sensoriais do ambiente e é capaz de modificá-lo a partir de ações. A autonomia do agente está relacionada ao fato de possuir controle sobre seu comportamento, podendo agir sem a intervenção de terceiros. As características envolvidas no conceito de agente autônomo são as seguintes (WOOLDRIDGE, 2002):

Reatividade. Para atingir seus objetivos, o agente é capaz de reagir, em tempo hábil, a mudanças que ocorrem no ambiente.

Pró-atividade. O agente é capaz de tomar a iniciativa, exibindo um comportamento dirigido aos objetivos, em vez de apenas reagir a mudanças que ocorrem no ambiente.

Habilidade social. O agente é capaz de interagir, quando apropriado, com outros agentes ou com seres humanos para atingir seus objetivos.

Afirmar que um agente é um sistema computacional significa que o mesmo existe, fisicamente, em forma de um programa que é executado em um dispositivo computacional. A metodologia utilizada na construção de um agente determina sua arquitetura. Wooldridge e Jennings (1995) apontam três diferentes arquiteturas de agentes:

Deliberativa. Prevê uma representação simbólica e explicita do ambiente. O agente toma decisões a partir de raciocínio sobre esta representação.

Reativa. Não requer representação explicita do ambiente. As decisões do agente são tomadas a partir de regras simples, do tipo percepção-ação.

Híbrida. Combinação da arquitetura deliberativa com reativa. O componente deliberativo oferece melhor fundamentação para as decisões do agente, enquanto que o componente reativo permite ao agente reagir rapidamente a eventos do ambiente sem a necessidade de raciocínios complexos.

Agentes são construídos com o propósito de atingir um objetivo. Entretanto, certos objetivos podem estar além da capacidade de um único agente. Isto é o que normalmente ocorre quando o objetivo é complexo e abrangente. Nestes casos, uma das formas de atingir o objetivo é construir um certo número de agentes, onde cada um deles irá atingir uma parte do objetivo geral (SYCARA, 1998). Chega-se então ao conceito de SMA.

De acordo com Wooldridge (2002), um SMA é formado por um certo número de agentes que interagem entre si. As características de um SMA são as seguintes (JENNINGS; SYCARA; WOOLDRIGE, 1998):

Cada agente pode possuir diferentes competências. As competências dos agentes definem a uniformidade do SMA. Em um SMA homogêneo, todos os agentes possuem as mesmas competências. Já em um SMA heterogêneo, existem agentes com diferentes competências. Em ambos os casos, quando o objetivo estiver além das competências individuais de cada agente, eles precisarão interagir para atingi-lo.

Cada agente pode possuir percepção limitada do ambiente. As informações necessárias para atingir o objetivo do SMA estão descentralizadas, o que limita a percepção. Isto requer interação entre os agentes para reunir as informações necessárias ao objetivo do sistema.

Computação é assíncrona. Esta característica está ligada com a autonomia dos agentes. Por serem reativos e pró-ativos, os agentes atuam assincronamente. Se o objetivo do SMA requer algum sincronismo, os agentes devem interagir para garanti-lo.

Inexistência de controle global/central. Não há entidade central com percepção global do sistema capaz de definir o comportamento adequado que cada agente deve realizar. Os agentes devem interagir para determinar o comportamento adequado que atinge o objetivo do SMA.

Como visto, as interações entre os agentes desempenham papel fundamental em um SMA, tanto que constituem uma das principais questões estudadas na área. Jennings, Sycara e Wooldridge (1998), classificam os tipos mais comuns de interações em:

Cooperação. Nesta interação os agentes possuem um objetivo em comum e trabalham em conjunto para atingi-lo.

Coordenação. Este tipo de interação tem por objetivo organizar os agentes, de forma a evitar comportamentos caóticos e prejudiciais, e explorar comportamentos organizados e benéficos.

Negociação. Nesta interação os agentes buscam atingir um entendimento entre si.

Os agentes de um SMA podem ainda ser cooperativos ou competitivos (ou automotivados). Agentes cooperativos consideram o objetivo do SMA acima dos interesses individuais de cada agente, visando portanto alcançar o melhor desempenho global. Já os agentes competitivos consideram os interesses individuais acima do objetivo do sistema.

2.2 Alocação de Tarefas

Até o momento mencionou-se apenas que cada agente tem como propósito *atingir* seus objetivos, mas não detalhou-se como *definir* estes objetivos. Uma das formas de defini-los é através da especificação de tarefas, criando um ambiente orientado à tarefas. O objetivo do agente é realizar as tarefas presentes no ambiente. Para medir o sucesso de um agente na realização das tarefas, define-se uma medida de desempenho. Quando as tarefas devem ser realizadas não apenas por um agente, mas por um SMA, o desempenho deve levar em consideração o ambiente como um todo. Desta forma indica-se o sucesso do SMA e não apenas de agentes individuais.

Os agentes de um SMA situados em um ambiente orientado à tarefas devem determinar *que* agentes devem realizar *quais* tarefas de forma a otimizar o desempenho. Esta necessidade de determinar que agente realiza qual tarefa é denominado problema de alocação de tarefas. Gerkey e Matarić (2004) apresentam uma taxononomia do problema de alocação de tarefas em sistemas multirobô, que é suficientemente geral para ser aplicada em SMAs. os autores definem três dimensões de classificação para os problemas de alocação de tarefas:

Agentes de única tarefa *versus* agentes de múltiplas tarefas. Um agente de única tarefa é capaz de realizar no máximo uma tarefa por vez. Já um agente de múltiplas tarefas é capaz de realizar múltiplas tarefas simultaneamente. No primeiro caso, o agente só precisa escolher uma dentre todas as tarefas que pode realizar (aquela que otimiza o desempenho). No segundo caso, o agente pode escolher um subconjunto de tarefas. Isto pode tornar o processo de alocação mais difícil pois o agente deve considerar o desempenho de realizar cada um dos possíveis subconjuntos de tarefas, optando por aquele que proporcione desempenho ótimo. Dependendo da quantidade de tarefas disponíveis, avaliar todos os subconjuntos pode ser inviável.

Tarefas de único agente *versus* tarefas de múltiplos agentes. Uma tarefa de único agente requer um e apenas um agente para realizá-la. Neste caso, o processo de alocação deve proporcionar exclusão mútua, para garantir que nenhuma tarefa seja realizada por mais de um agente. Uma tarefa de múltiplos agentes requer que um grupo de agentes seja alocado simultaneamente. O grupo selecionado deve ser aquele que otimize o desempenho. Logo, o processo de alocação deve considerar os possíveis grupos que podem realizar simultaneamente a tarefa. No caso de agentes de única tarefa, estes possíveis grupos são disjuntos. Com agentes de múltiplas tarefas, os grupos podem se sobrepor, aumentando o espaço de combinações.

Alocação instantânea *versus* alocação estendida. Na alocação instantânea as informações percebidas pelos agentes com relação às tarefas disponíveis e ao ambiente só permitem que seja realizada a alocação no instante de tempo atual. Como poucos cenários requerem uma única alocação instantânea, são definidas duas variantes desta dimensão:alocação instantânea iterativa e alocação instantânea *online*. Na alocação iterativa o ambiente apresenta certo dinamismo que modifica as características das tarefas. O objetivo é realizar sucessivas alocações, em um processo iterativo, iniciado a cada alteração no ambiente ou a cada instante de tempo (nos casos onde o tempo é discretizado). Na alocação *online*, o ambiente é parcialmente observável. O conjunto de tarefas a serem alocadas não é revelado por completo de uma só vez, sendo reveladas aos poucos aos agentes. A cada nova tarefa percebida, os agentes devem retomar o processo de alocação até atingir uma que otimize

o desempenho. Na alocação estendida, informações adicionais estão disponíveis aos agentes, como por exemplo um modelo que descreve a dinâmica do ambiente. Logo, os agentes devem conceber uma alocação que leve em consideração o futuro, ou seja, um agendamento de alocações. A alocação estendida é NP-Difícil mesmo nos casos com poucas tarefas ou agentes, devido a quantidade exponencial de possíveis alocações futuras. Uma forma alternativa de tratar esta classe é desconsiderar o componente tempo estendido e tratar como uma instância de alocação instantânea iterativa ou *online*.

Recentemente, um fator adicional tem chamado a atenção nos problemas de alocação de tarefas em SMA: a escala. Este fator diz respeito a quantidade de agentes e de tarefas presentes no ambiente. Esforços tem sido concentrados para conceber métodos eficientes de alocação de tarefas em SMAs de larga escala, com centenas de agentes e tarefas.

Para designar um SMA de larga escala em um ambiente orientado a tarefas, Scerri et al. (2005) introduziram o termo *extreme teams*. Agentes que compõem um *extreme team* possuem competências definidas para realização das tarefas, além de uma quantidade limitada de recursos. As seguintes características estão relacionadas com a alocação de tarefas em *extreme teams*:

(i). O ambiente é dinâmico, isto é, tarefas podem aparecer, desaparecer e terem suas características alteradas com o passar do tempo. Os agentes não possuem, por definição, um modelo que descreve a dinâmica do ambiente. Logo, a alocação de tarefas defe ser instantânea e iterativa. Também deve ser *online*, visto que o ambiente pode ser parcialmente observável. Por fim, a alocação deve ocorrer de forma rápida, pois o dinamismo do ambiente impõe restrições sobre o tempo disponível aos agentes para decidir quais tarefas realizar;

(ii). Os agentes são de múltiplas tarefas. Isto é, cada agente pode realizar várias tarefas ao mesmo tempo, desde que a quantidade de recursos que estas tarefas requerem não ultrapasse a quantidade de recursos que o agente dispõe;

(iii). Os agentes podem possuir funcionalidades sobrepostas. Isto quer dizer que vários agentes podem estar aptos a realizar a mesma tarefa, mas com diferentes níveis de competência. Ou seja, os agentes são heterogêneos;

(iv). Podem existir inter-relacionamentos entre tarefas. Estes inter-relacionamentos impõem, por exemplo, a necessidade de que certas tarefas sejam realizadas simultâneamente para que se obtenha o efeito desejado no ambiente. Neste caso, as tarefas inter-relacionadas podem ser vistas como subtarefas geradas a partir da decomposição de uma tarefa de múltiplos agentes. Portanto, a tarefa de múltiplos agentes só é realizada com sucesso se todas as subtarefas forem executadas simultaneamente.

Para formalização do problema de alocação de tarefas em *extreme teams*, Scerri et al. (2005) propõem a utilização de uma versão estendida do modelo GAP, denominado E-GAP. Estes modelos são apresentados a seguir.

2.2.1 Modelos GAP e E-GAP

O *Generalized Assignment Problem* (GAP) é um modelo que formaliza a alocação de tarefas entre agentes. A seguir, o modelo GAP é descrito. Para detalhes adicionais, recomenda-se consultar (MARTELLO; TOTH, 1990), capítulo 7. Um 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 uma das tarefa $j \in \mathcal{J}$, denotada por $Cap(i, j) \rightarrow [0, 1]$. Cada agente também possui uma quantidade limitada de recursos $i.res$ e consome $Res(i, j)$ recursos ao realizar a tarefa j. Uma matriz M é utilizada para representar a alocação, onde m_{ij} é dado pela Equação (2.1).

$$m_{ij} = \begin{cases} 1 & \text{se } i \text{ realiza } j \\ 0 & \text{caso contrário} \end{cases} \tag{2.1}$$

A solução de um GAP é a alocação M que maximiza o valor do somatório das competências dos agentes que participam da alocação, como mostrado na Equação (2.2). Este valor define a recompensa do sistema, ou seja, seu desempenho.

$$M = \underset{M'}{\operatorname{argmax}} \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} Cap(i, j) \times m'_{ij} \tag{2.2}$$

sujeito a

$$\forall i \in \mathcal{I}, \sum_{j \in \mathcal{J}} Res(i, j) \times m_{ij} \leq i.res \tag{2.3}$$

e

$$\forall j \in \mathcal{J}, \sum_{i \in \mathcal{I}} m_{ij} \leq 1 \tag{2.4}$$

A Equação (2.3) especifica que os agentes são de múltiplas tarefas, onde a quantidade de tarefas que podem realizar ao mesmo tempo é limitada apenas pela quantidade de recursos disponíveis. O modelo GAP considera apenas tarefas de único agente. A Equação (2.4) especifica esta característica, impondo uma restrição de exclusão mútua, isto é, cada tarefa deve ser realizada por no máximo um agente.

O modelo GAP foi estendido por Scerri et al. (2005) para incorporar características relacionadas com *extreme teams*: inter-relacionamentos entre tarefas e ambientes dinâmicos. Este modelo foi chamado de *Extended Generalized Assignment Problem* (E-GAP)[1].

No modelo E-GAP, os inter-relacionamentos são utilizados, por exemplo, para representar tarefas de múltiplos agentes. Estas tarefas podem ser decompostas em subtarefas inter-relacionadas por **E** (*and*) lógico, impondo desta forma a necessidade de realização simultânea. Sem a realização simultânea de todas as subtarefas, a tarefa de múltiplos agentes não é realizada com sucesso e não se produz a recompensa desejada no sistema, desperdiçando os recursos dos agentes.

Para formalizar inter-relacionamentos entre tarefas, o modelo E-GAP define um conjunto $\bowtie = \{\alpha_1, \ldots, \alpha_p\}$ que contém p conjuntos α de tarefas inter-relacionadas por **E**, na forma $\alpha = \{j_1 \wedge \ldots \wedge j_q\}$. Cada conjunto $\alpha \in \bowtie$ representa portanto uma tarefa de múltiplos agentes e contém as subtarefas em que foi decomposta, inter-relacionadas por **E**. Para produzir efeito no sistema, todas as tarefas pertencentes a α devem ser simultaneamente realizadas. A quantidade de tarefas x_α de um conjunto α que estão sendo realizadas simultaneamente é dada pela Equação (2.5).

$$x_\alpha = \sum_{i \in \mathcal{I}} \sum_{j \in \alpha} m_{ij} \tag{2.5}$$

[1] Problema Generalizado de Atribuição Estendido

Seja $v_{ij} = Cap(i,j) \times m_{ij}$. Dadas as relações de $\bowtie$, a recompensa $Val(i,j,\bowtie)$ que o agente i proporciona ao sistema quando realiza j (isto é, seu desempenho) é dado pela Equação (2.6).

$$Val(i,j,\bowtie) = \begin{cases} v_{ij} & \text{se } \forall \alpha \in \bowtie, j \notin \alpha \\ v_{ij} & \text{se } \exists \alpha \in \bowtie \text{ com } j \in \alpha \wedge x_\alpha = |\alpha| \\ 0 & \text{caso contrário} \end{cases} \tag{2.6}$$

O primeiro caso define a recompensa se a tarefa j não está inter-relacionada com outras. O segundo caso especifica a recompensa quando a tarefa j está inter-relacionada com outras tarefas (dadas pelo conjunto α que contém j) que estão sendo realizadas simultaneamente. O último caso especifica a recompensa quando j está inter-relacionada com outras tarefas mas que não estão sendo realizadas simultaneamente.

Ambientes dinâmicos são representados no E-GAP a partir da discretização do tempo. Para tanto, todas as variáveis do modelo E-GAP são indexadas por um instante de tempo t. O objetivo é encontrar uma sequência de alocações $\overrightarrow{M}$, uma para cada instante t de forma a maximizar a recompensa total do sistema $f(\overrightarrow{M})$, Equação (2.7). A recompensa (consequentemente, o desempenho dos agentes) é portanto resultado de uma sequência de alocações, diferententemente do GAP que considera uma única alocação. Um custo de atraso $DC^t(j^t)$ pode ser utilizado para definir uma penalização na recompensa quando uma tarefa j não for realizada no instante t.

$$\begin{aligned} f(\overrightarrow{M}) = & \sum_t \sum_{i^t \in \mathcal{I}^t} \sum_{j^t \in \mathcal{J}^t} Val^t(i^t, j^t, \bowtie^t) \\ & - \sum_t \sum_{j^t \in \mathcal{J}^t} (1 - \sum_{i^t \in \mathcal{I}^t} m_{ij}^t) \times DC^t(j^t) \end{aligned} \tag{2.7}$$

sujeito a

$$\forall t, \forall i^t \in \mathcal{I}^t, \sum_{j^t \in \mathcal{J}^t} Res^t(i^t, j^t) \times m_{ij}^t \leq i^t.res \tag{2.8}$$

e

$$\forall t, \forall j^t \in \mathcal{J}^t, \sum_{i^t \in \mathcal{I}^t} m_{ij}^t \leq 1 \tag{2.9}$$

Igualmente ao GAP, cada agente do E-GAP é de múltiplas tarefas, onde a quantidade de tarefas que pode realizar é limitada pelos recursos disponíveis em cada instante t, conforme define a Equação (2.8). Com a representação indireta de tarefas de múltiplos agentes através de subtarefas inter-relacionadas, cada uma destas subtarefas deve ser realizada por no máximo um agente, conforme define a Equação (2.9)

2.3 Resumo

Este capítulo apresentou conceitos de SMAs e alocação de tarefas em SMAs. Para se definir os objetivos dos agentes usualmente definem-se ambientes orientados à tarefas. Agentes situados nestes ambientes devem realizar as tarefas de forma a otimizar uma medida coletiva de desempenho.

Scerri et al. (2005) introduziram o termo *extreme teams* para designar estes ambientes. A alocação de tarefas em *extreme teams* está relacionada com as seguintes características:

(i) o ambiente é dinâmico; (ii) os agentes podem realizar múltiplas tarefas; (iii) os agentes podem possuir funcionalidades sobrepostas (são heterogêneos); e (iv) podem existir inter-relacionamentos entre tarefas. A alocação de tarefas em *extreme teams* pode ser formalizada através do modelo E-GAP.

Abordagens existentes até o momento para alocação de tarefas tratam, efetivamente, apenas as três primeiras características envolvidas no conceito de *extreme teams*. Neste sentido, esta dissertação apresenta uma abordagem para alocação de tarefas que trata todas as quatro características, contribuindo para a concretização do conceito completo de *extreme teams*.

3 INSETOS SOCIAIS

Insetos sociais, aqueles que vivem em colônias (formigas, abelhas, cupins e algumas espécies de vespas), tem fascinado cientistas ao longo dos anos. Diariamente, uma colônia resolve eficientemente uma série de problemas, por exemplo: encontrar alimento, construir ou estender ninhos, e lidar com alterações no ambiente.

Um inseto, apesar de sua aparente simplicidade, pode processar informações sensoriais, modular seu comportamento de acordo com diversos estímulos e tomar decisões com base em grande quantidade de informações. Contudo, as características destes indivíduos não são suficientes para explicar a complexidade do que uma colônia de insetos sociais é capaz de fazer.

Colônias de insetos apresentam princípios de auto-organização. Em um sistema auto-organizado, o padrão observado em nível macroscópico emerge a partir de processos e interações definidos em nível microscópico. No caso dos insetos sociais, o comportamento coletivo e complexo da colônia emerge a partir de interações entre insetos que exibem comportamentos relativamente simples se comparados com a complexidade da colônia (BONABEAU; THERAULAZ; DORIGO, 1999).

As interações entre os insetos sociais podem ser diretas ou indiretas. Interações diretas ocorrem, por exemplo, através de sons, toques de antenas, troca de fluidos, contato visual ou mandibular. Interações indiretas ocorrem através do ambiente. Dois indivíduos se comunicam por interação indireta quando um deles modifica o ambiente e o outro responde, posteriormente, ao ambiente modificado. A interação através do ambiente é denominada estigmergia e é realizada normalmente através de feromônios (BONABEAU; THERAULAZ; DORIGO, 1999).

Uma das características mais importantes dos insetos sociais é que resolvem os problemas envolvidos na subsistência da colônia de forma flexível e robusta. A flexibilidade permite adaptação às alterações do ambiente. A robustez proporciona à colônia a habilidade de operar mesmo se alguns indivíduos falharem ao realizar suas atividades

A partir de observações do comportamento coletivo das colônias de insetos sociais, pesquisadores tem desenvolvido metáforas válidas para solução de problemas computacionais. Para designar estas metáforas e pesquisas, utiliza-se o termo inteligência de enxames. Na inteligência de enxames, a solução do problema é uma propriedade que emerge da coletividade, ou seja, da interação entre agentes simples (BONABEAU; THERAULAZ; DORIGO, 1999). A seguir são apresentadas duas metáforas criadas a partir dos insetos sociais e que fundamentam a abordagem proposta nesta dissertação: divisão de trabalho e recrutamento para transporte cooperativo.

3.1 Divisão de Trabalho

Uma divisão de trabalho eficiente é responsável pelo sucesso ecológico das sociedades de insetos. Uma colônia de insetos sociais com centenas de milhares de membros opera sem a existência de coordenação explicita. Um indivíduo não têm acesso às necessidades da colônia como um todo, mas apenas à informação local e simplificada.

Em uma colônia, os indivíduos se especializam em determinados conjuntos de tarefas. Esta especialização pode ser ocasionada por três fatores, que podem se sobrepor:

Polietismo. A idade do indivíduo determina as tarefas que irá realizar. Indivíduos na mesma faixa etária tendem a realizar tarefas idênticas.

Polimorfismo. A forma (morfologia) do indivíduo é que determina as tarefas que irá realizar. Indivíduos com diferentes morfologias tendem a realizar diferentes tarefas.

Variabilidade individual. Tanto no polietismo quanto no polimorfismo, podem existir diferenças entre os indivíduos na mesma faixa etária ou com a mesma morfologia. Estas diferenças podem afetar o desempenho ao realizar as tarefas.

A divisão de trabalho entre indivíduos com base em sua especialização permite que diferentes atividades sejam realizadas simultaneamente por grupos de indivíduos especializados, ao invés de por indivíduos sem nenhuma especialização. Isto aumenta a eficiência da colônia na realização das tarefas.

Outro fator que contribui para a eficiência da colônia é a plasticidade da divisão de trabalho. A plasticidade, que corresponde a uma flexibilidade comportamental dos indivíduos, permite que se engajem em diferentes tarefas de acordo com as necessidades da colônia, colaborando para a flexibilidade e rubustez da colônia.

Observações deste comportamento de divisão de trabalho são a base do modelo teórico descrito por Theraulaz, Bonabeau e Deneubourg (1998). Neste modelo, a interação entre os membros da colônia e a percepção, por parte dos indivíduos, de necessidades pontuais, resultam em uma distribuição dinâmica das tarefas. Cada indivíduo da colônia possui um limiar de resposta a estímulos para realizar determinada tarefa. Um indivíduo passa a realizar uma tarefa quando o estímulo associado à tarefa ultrapassa seu limiar de resposta.

Seja s_j a intensidade de um estímulo associado a uma tarefa j em particular, onde s_j pode ser o número de encontros com outros indivíduos, uma concentração química ou qualquer outro fator quantitativo que possa ser sentido por um indivíduo. Cada indivíduo i possui um limiar de resposta θ_{ij}, expresso em unidades de intensidade de estímulo, relacionado com a pré-disposição de reagir ao estímulo associado à tarefa j. Este limiar de resposta está relacionado com as três especializações mencionadas: polietismo, polimorfismo ou variabilidade individual.

No modelo de Theraulaz, Bonabeau e Deneubourg (1998), o limiar de resposta θ_{ij} e o estímulo s_j formam a tendência (probabilidade) $T_{ij}(s_j)$ do indivíduo i realizar a tarefa j. Esta tendência é mostrada na Equação (3.1).

$$T_{ij}(s_j) = \frac{s_j^2}{s_j^2 + \theta_{ij}^2} \tag{3.1}$$

A equação de tendência comporta-se da seguinte maneira. Se o valor do limiar de resposta θ_{ij} for muito menor que o estímulo s_j, a tendência tende a 1. No caso oposto, quando o limiar θ_{ij} for muito maior em relação ao estímulo s_j, a tendência tende a 0. Com $\theta_{ij} = s_j$, a tendência é exatamente 1/2.

O uso desta tendência como base para o modelo de divisão de trabalho introduz a flexibilidade e robustez da colônia. Qualquer indivíduo está apto a realizar qualquer tarefa, desde que o estímulo correspondente seja suficientemente alto para ultrapassar o limiar de resposta do indivíduo. Em uma eventual ausência de indivíduos especializados, outros indivíduos iniciam a realização das tarefas assim que os estímulos ultrapassam seus limiares de resposta.

A Figura 3.1 apresenta o comportamento da equação de tendência considerando a variação do limiar de resposta θ_{ij} e com o estímulo $s_j = \{0.1, 0.5, 0.9\}$. Pode-se verificar que quando uma tarefa oferece alto estímulo ao indivíduo (por exemplo, 0.9), a tendência de realizar esta tarefa é alta para $0 \leq \theta_{ij} < 0.9$. Com isto, uma tarefa com alto estímulo tem uma chance muito maior de ser realizada, mesmo que por agentes não especializados, ou seja, com limiar de resposta alto.

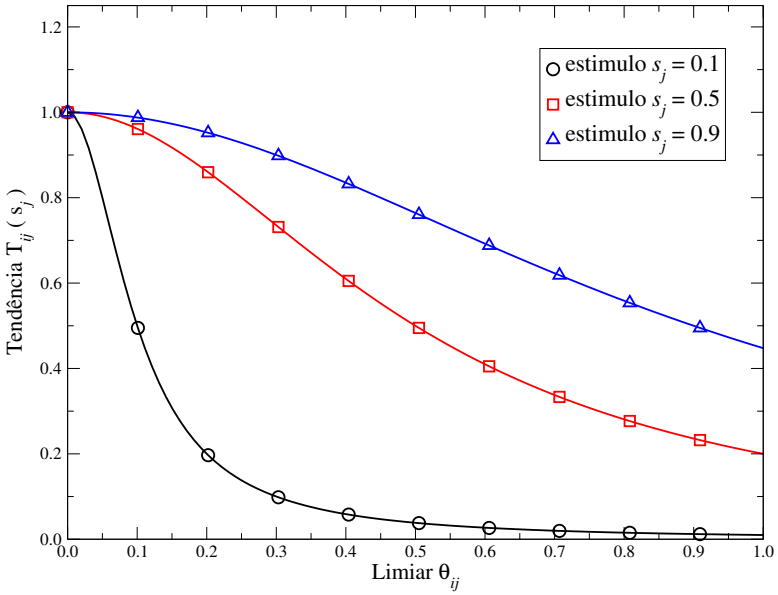

Figura 3.1: Exemplo da tendência $T_{ij}(s_j)$ considerando variação no limiar θ_{ij}, para os valores de estímulo $s_j = \{0.1, 0.5, 0.9\}$.

Quando o estímulo assume um valor intermediário, no caso 0.5, a tendência de a tarefa ser realizada é influenciada pelo estímulo e pelo limiar de resposta do indivíduo. Indivíduos com limiares baixos estarão mais pré-dispostos para realizar a tarefa. Indivíduos com limiares altos também podem realizar a tarefa, mas com menor chance.

Quando uma tarefa oferece baixo estímulo (por exemplo, 0.1), o limiar do indivíduo é determinante para a tendência de a tarefa ser realizada ou não. Indivíduos com alto limiar de resposta, ou seja, não especializados na tarefa, tendem a recusar sua realização.

3.2 Recrutamento para Transporte Cooperativo

Certas espécies de formigas transportam cooperativamente presas para o ninho. É o caso, por exemplo, das espécies *Novomessor cockerelli* e *Novomessor albisetosus* (HÖLLDOBLER; STANTON; MARKL, 1978), *Oecophylla longinoda* (HÖLLDOBLER; WILSON, 1972), *Eciton burchelli* (FRANKS, 1986), e *Formica schaufussi* (ROBSON; TRANIELLO, 1998). O transporte cooperativo de presas envolve duas ou mais formigas que transportam de maneira conjunta uma presa que não poderia ser

transportada por uma única formiga (ROBSON; TRANIELLO, 1998). A Figura 3.2 apresenta uma situação de transporte cooperativo na espécie *Oecophylla longinoda*.

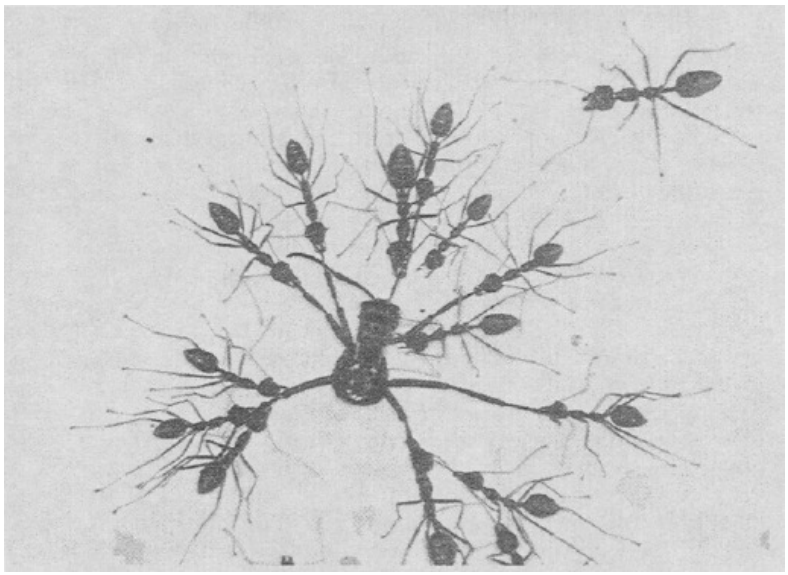

Figura 3.2: Exemplo de um grupo de formigas imobilizando e transportando cooperativamente uma presa (HÖLLDOBLER; STANTON; MARKL, 1978)

Nas espécies mencionadas, o principal propósito do transporte cooperativo é maximizar o compromisso entre a energia adquirida (em forma de alimento) e a energia empregada no transporte para o ninho. Hölldobler, Stanton e Markl (1978) reportam que um grupo de três a cinco operárias *Novomessor* são facilmente capazes de transportar cooperativamente um gafanhoto com peso seco de 200mg. Um peso equivalente é obtido com aproximadamente 240 cupins. Entretando, seriam necessárias aproximadamente 240 operárias *Novomessor* para transportar individualmente estes cupins. No caso da espécie *E. burchelli*, a formação de grupos para transportar as presas também minimiza a energia empregada (FRANKS, 1986). Mesmo havendo operárias forte o bastante para transportar grandes presas, a energia empregada por esta única operária precisa superar o peso e as forças rotacionais da presa para equilibrá-la no trajeto para o ninho. Com mais operárias envolvidas no transporte, as forças rotacionais são balanceadas, fazendo com que as formigas necessitem empregar apenas a energia necessária para superar o peso da presa.

Outro propósito do transporte cooperativo é aumentar a velocidade com que a presa é transportada (HÖLLDOBLER; STANTON; MARKL, 1978; ROBSON; TRANIELLO, 1998; HÖLLDOBLER; WILSON, 1972). A velocidade é importante para saciar rapidamente a fome da colônia após longos períodos de falta de comida. Além disto, previne que a presa seja descoberta por espécies que competem pela sobrevivência.

O grupo envolvido no transporte cooperativo é formado por um processo chamado recrutamento. Quando uma formiga exploradora (deste ponto em diante será chamada de *scout*) descobre uma presa, ela inicialmente tenta agarrá-la e transportá-la individualmente. Após algumas tentativas sem sucesso, ou quando consegue erguer a presa mas o transporte não atinge uma certa velocidade, o processo de recrutamento é iniciado.

Para recrutar companheiras, todas as espécies estudadas empregam um mecanismo que Hölldobler et al. (1978) denominam Recrutamento de Longo Alcance (RLA). Algumas espécies também empregam um segundo mecanismo, denominado Recrutamento de Curto Alcance (RCA). Ambos os mecanismos utilizam estigmergia (comunicação através

do ambiente).

No RCA a *scout* que descobriu a presa libera uma secreção. O odor da secreção flui através do ar. Formigas localizadas na vizinhança, em um raio de até dois metros, são atraídas até a presa pelo odor, e tentam transportar em conjunto a presa. Hölldobler et al. (1978) reportaram que na maioria das vezes o RCA é suficiente para formar um grupo de formigas que conseguem transportar com sucesso a presa.

No RLA, a *scout* que descobriu a presa volta para o ninho para recrutar companheiras. No trajeto até o ninho, a *scout* deixa um rastro de feromônio (substância utilizada na comunicação por estigmergia). Formigas encontradas no trajeto até o ninho são estimuladas pela *scout*. Quando uma companheira é estimulada, ela também passa a deixar um rastro de feromônio, mesmo ainda não tendo experimentado o contato direto com a presa. Desta forma é estabelecido uma cadeia de comunicação entre companheiras. Após a *scout* chegar no ninho, companheiras são atraídas pelo feromônio e movem-se em direção à presa. Assim que as formigas recrutadas chegam na presa, o transporte cooperativo é iniciado.

A quantidade de formigas envolvidas de fato no transporte cooperativo é regulada de acordo com as características da presa, como massa, forças rotacionais e dificuldade de movimentação. Enquanto o número de formigas não for suficiente para transportar a presa, mais formigas são recrutadas através de RCA ou RLA. Apesar de que, em uma abordagem mais econômica, a *scout* deveria recrutar a quantidade exata de companheiras necessárias, Robson e Traniello (1998) sugerem que a *scout* não tem capacidade de definir esta quantidade. Portanto, a estratégia mais eficiente é recrutar um número constante de formigas, seguida por uma adaptação desta quantidade à presa durante o transporte.

O mecanismo de recrutamento possibilita que as formigas transportem presas que seriam impossíveis de serem transportadas por uma formiga solitariamente. Um fato interessante reportado por Franks (1986) na espécie *E. burchelli* é a supereficiência do grupo de transporte. Em alguns casos, a presa transportada é tão grande que se fosse fragmentada, os membros do grupo seriam incapazes de transportar todos os fragmentos.

3.3 Resumo

Este capítulo apresentou as metáforas de divisão de trabalho e de recrutamento para transporte cooperativo em insetos sociais. Uma divisão de trabalho eficiente é responsável pelo sucesso ecológico das sociedades de insetos. O recrutamento permite que presas grandes sejam transportadas de forma coooperativa, minimizando o compromisso entre a energia adquirida em forma de alimento e a energia empregada no transporte.

Estas metáforas fundamentam a abordagem para alocação de tarefas proposta nesta dissertação. Os agentes utilizam o modelo de divisão de trabalho para decidir quais tarefas devem realizar, atendendo as características (i), (ii), e (iii) de *extreme teams*. O recrutamento é utilizado para atender a característica (iv), pois permite que sejam formados grupos de agentes comprometidos com a realização simultânea de tarefas interrelacionadas.

4 TRABALHOS RELACIONADOS

Segundo Sycara (1998), a alocação de tarefas é um dos problemas mais antigos abordados em SMA. Uma alternativa para a alocação de tarefas seria definir *a priori*, pelo projetista do sistema, uma alocação fixa das tarefas aos agentes. Contudo, esta solução é limitada e inflexível para ambientes de tarefas dinâmicos e parcialmente observáveis.

A seguir são apresentadas algumas possíveis abordagens flexíveis para alocação de tarefas em SMA. Uma revisão completa sobre cada abordagem está fora do escopo deste trabalho. Em cada abordagem será apresentada a idéia geral e implicações de utilização no caso específico de *extreme teams*. As abordagens LA-DCOP e Swarm-GAP recebem maior destaque, pois são objeto de comparação com a abordagem proposta.

4.1 *Contract-Net Protocol* e Leilões

Um dos primeiros métodos flexíveis para alocação de tarefas em SMA foi o *Contract-Net Protocol* (CNP) (SMITH, 1980). O CNP requer que cada agente esteja conectado a todos os demais através do canal de comunicação. No CNP, o agente que percebe uma tarefa (denominado de controlador) dispara um processo para contratar agentes que irão realizar a tarefa. Este processo consiste em fazer difusão de uma oferta de contrato para a tarefa. Os demais agentes podem recusar ou aceitar a oferta. Ao aceitar, o agente informa o controlador suas competências para realizar a tarefa, não podendo voltar atrás (cancelar a aceitação). Ao receber aceitações suficientes, o controlador contrata os agentes com melhor competência para realizar a tarefa. As deficiências do CNP incluem o fato de requerer que todos os agentes estejam inter-conectados, e também a intensa comunicação que ocorre entre os agentes durante o processo de contratação (SYCARA, 1998), inviabilizando a aplicação em *extreme teams*.

Um leilão é um mecanismo de compra e venda de itens baseado em preços. Seu modo de operação é relativamente similar ao CNP, por este motivo são apresentados em conjunto. Em um leilão, os compradores fazem lances sobre os itens à venda levando em consideração suas preferências e recursos. Os lances podem ser cancelados pelos compradores caso se interessem por outros itens. Um leiloeiro, após rodadas de lances por parte dos compradores, define quais vencem o leilão de forma que o lucro seja maximizado. Os compradores podem fazer lances sobre itens individuais ou combinações de itens. No caso de lances sobre combinações, caracteriza-se um leilão combinatorial.

Hunsberger e Grosz (2000) apresentam um mecanismo para alocação de tarefas em SMA baseado em leilões combinatoriais. Os itens à venda são as tarefas e os compradores são todos os demais agentes, que ofertam a compra de tarefas a partir de suas competências e recursos. Um agente leiloeiro deve ser escolhido para decidir quais compradores vencem o leilão. Decidir quais compradores vencem o leilão com base em suas ofertas é

conhecido como problema de determinar o vencedor[1]. Este problema é NP-Completo no caso de leilões combinatoriais devido ao número exponencial de combinações de itens à venda que devem ser avaliadas. Maiores detalhes sobre leilões e sobre o problema de determinar o vencedor podem ser econtrados em (CRAMTON; SHOHAM; STEINBERG, 2006) e (LEHMANN; MÜLLER; SANDHOLM, 2006), respectivamente.

Na alocação de tarefas através de leilões, o anúncio das tarefas que estão à venda e as ofertas de compra são feitas através de comunicação por mensagens. O anúncio é feito pelo leiloeiro através de difusão das tarefas para os demais agentes. Cada agente oferta sua competência ao leiloeiro para uma tarefa ou um conjunto delas. No modelo E-GAP, um agente pode, no melhor caso, possuir competências para realizar todas as tarefas e pode portanto fazer muitos lances. Considerando as ofertas através de mensagens e a questão da escala presente em *extreme teams*, os leilões requerem grande quantidade de comunicação. Além disto, um leilão acaba sendo um mecanismo centralizado, pois as informações relacionadas com as competências dos demais agentes são centralizadas no leiloeiro para determinar os vencedores. Este fato pode suscitar questões de escalabilidade em SMAs de larga escala.

Em trabalhos recentes, Ham e Agha (2007; 2008) avaliam o desempenho de diferentes mecanismos de leilão simples (não combinatorial) para alocação de tarefas. O ambiente considerado possui agentes homogêneos, com percepção limitada do ambiente. Entretanto, a comunicação dos agentes é limitada em relação ao alcance, mas não em relação a quantidade (isto é, um agente pode se comunicar livremente com outros desde que estejam no raio de alcance de comunicação). Apesar de apresentarem resultados apenas qualitativos das comparações entre os mecanismos, a necessidade do uso de difusão para anunciar as tarefas à venda sugere grande quantidade de comunicação. De forma geral, os resultados apresentados estabelecem que limitações na percepção e alcance da comunicação diminuem a complexidade dos leilões realizados com relação a tempo e quantidade de mensagens. Isto deve-se a diminuição na quantidade de tarefas à venda percebidas.

4.2 Coalizões

Uma coalizão pode ser definida como um grupo de agentes que concordam em cooperar para atingir um objetivo comum. Este objetivo possui uma medida de desempenho, portanto, a coalizão responsável por realizar o objetivo deve ser aquela que maximize o desempenho. Na literatura sobre coalizões, esta medida de desempenho é chamada de valor da coalizão. Caso o objetivo seja realizar tarefas, pode-se pensar na alocação de tarefas como um problema de formação de coalizões (SHEHORY; KRAUS, 1995). Isto fica ainda mais evidente ao considerar tarefas de múltiplos agentes, que requerem um grupo (coalizão) de agentes simultaneamente alocados. Após as coalizões serem formadas, elas são atribuidas às tarefas, para que sejam realizadas pelos agentes.

Segundo Shehory e Kraus (1995), o processo de formação de coalizões para alocação de tarefas em SMAs é composto de três etapas. Na primeira etapa deve-se gerar as possiveis coalizões, isto é, as combinações de agentes. Na segunda etapa deve-se computar o valor destas coalizões. Por último, na terceira etapa, deve-se escolher as coalizões que apresentaram melhor valor e que deverão realizar as tarefas.

No método proposto por Shehory e Kraus (1995), cada agente deve ter ciência da existência dos demais agentes e de todas as tarefas. Para a primeira etapa, cada agente gera todas as coalizões possíveis que contenham o próprio agente. Esta geração das coalizões

[1] *Winner Determination Problem*

possui complexidade de tempo e espaço exponenciais em função do número de agentes. Através de negociação, os agentes definem quais deles serão responsáveis por calcular o valor de quais coalizões. Esta negociação envolve troca de mensagens, que contém as coalizões que cada agente se candidatou a calcular o valor. Como a quantidade de possíveis coalizões é exponencial, estas mensagens normalmente são de tamanho exponencial. A determinação do valor das coalizões envolve interação entre os agentes, para obter as suas competências. Para cada uma das tarefas existentes, o agente avalia o desempenho de cada coalizão, considerando as competências dos agentes envolvidos. Por fim, deve-se escolher qual coalizão irá realizar cada tarefa. Este processo também é feito por negociação, com base nos valores calculados para as coalizões. Se o sistema for composto por agentes de múltiplas tarefas, estes podem participar de diferentes coalizões. Neste caso, ao se escolher uma coalizão que contém um agente de múltipla tarefa, o valor das demais coalizões devem ser recalculados para refletir os recursos utilizados pelo agente.

Como pode ser visto, o processo de formação de coalizões é computacionalmente caro, inviabilizando sua utilização em ambientes parcialmente observáveis, dinâmicos e com grande quantidade de agentes e tarefas. Recentemente, Rahwan e Jennings (2007) propuseram um novo método para distribuir o cálculo dos valores das coalizões entre os agentes. Este método define índices para as coalizões com base nos identificadores dos agentes. As coalizões que cada agente irá calcular o valor estão relacionadas com o identificador do agente, não requerendo portanto negociação adicional nesta etapa. Contudo, ainda é necessário que cada agente conheça a quantidade total de agentes. Além disto, o cálculo do valor das coalizões realizado por cada agente também requer que conheça as competências dos envolvidos nas coalizões.

4.3 LA-DCOP

Um *Distributed Constraint Optimization Problem* (DCOP)[2](MODI et al., 2003) consiste de um conjunto de variáveis que devem assumir cada qual um valor. Cada variável possui um domínio discreto de possíveis valores e está associada a um agente que possui o controle sobre seu valor. O objetivo dos agentes é selecionar valores para as variáveis de forma a otimizar uma função objetivo global. Esta função pode ser descrita como uma agregação de funções de custo definidas sobre pares de variáveis. Um DCOP pode ser representada por um grafo de restrições, onde os vértices representam variáveis e as arestas funções de custo entre variáveis.

Como pode ser visto, existem semelhanças entre as características de DCOPs e E-GAPs. Em ambos, as informações estão distribuídas entre agentes, e é necessário uma otimização. Um DCOP deve otimizar a função de objetivo global. Em um E-GAP, deve-se otimizar a recompensa do sistema (obtida a partir das competências dos agentes) na realização das tarefas. Estas semelhanças induzem tratar a alocação de tarefas em *extreme teams* como DCOPs. Para tanto, é necessário mapear o E-GAP para um DCOP de forma que a solução do DCOP seja a solução do E-GAP. Contudo, este mapeamento apresenta duas questões críticas quanto a complexidade do DCOP gerado: o tamanho dos domínios e a densidade do grafo de restrições.

O tamanho dos domínios cresce exponencialmente em função do número de tarefas. No pior caso, onde os agentes possuem recursos suficientes para realizar todas as tarefas simultaneamente, o tamanho do domínio de cada variável será $2^{|\mathcal{J}|}$. A densidade do grafo de restrições cresce exponencialmente em função do número de agentes. Isto ocorre

[2]Problema de Otimização de Restrições Distribuídas

pela necessidade de se especificar uma função de custo entre cada par de agentes, para estabelecer a restrição de exclusão mútua existente na alocação de tarefas do modelo E-GAP. O resultado é um grafo completo, onde existe função de custo entre todas as variáveis, gerando um total de $\frac{|\mathcal{I}|(|\mathcal{I}|-1)}{2}$ funções de custo.

Diversos algoritmos completos para DCOPs foram propostos recentemente, como por exemplo: Adopt-ng (SILAGHI; YOKOO, 2006, 2007), Adopt (MODI et al., 2003), AFB (GERSHMAN; MEISELS; ZIVAN, 2006), BnB-Adopt (YEOH; FELNER; KOENIG, 2007a), DPOP (PETCU; FALTINGS, 2005) e OptAPO (MAILLER; LESSER, 2004). Entretanto, as questões apontadas acima tornam inviáveis a aplicação destes algoritmos em E-GAPs em função do tempo que levariam para obter a solução e do espaço que requerem. Esta constatação é comprovada por Sultanik et al. (2007), que realizaram experimentos utilizando o algoritmo Adopt para alocação de tarefas em um ambiente estático, com apenas 4 agentes e 3 ou 4 tarefas, reportando dias de processamento para obter a alocação. Tendo em vista que a inviabilidade é causada pela complexidade do DCOP gerado, estes resultados sugerem que os demais algoritmos sofrem limitações semelhantes.

Para lidar com as características específicas do modelo E-GAP, Scerri et al. (2005) apresentam um algoritmo aproximado, chamado *Low-communication Approximate DCOP* (LA-DCOP). Seus autores o classificam como um algoritmo para DCOP por utilizar conceitos ligados a definição de DCOP, como a distribuição da informação e necessidade de maximizar uma função de objetivo global, que corresponde à recompensa no modelo E-GAP. O LA-DCOP é diretamente aplicável em problemas de alocação de tarefas formalizados como E-GAP, não havendo necessidade de qualquer tipo de mapeamento.

O LA-DCOP utiliza um protocolo baseado em *tokens*. Ao perceber uma tarefa, o agente é responsável por criar um *token* para representá-la. O agente também pode receber *tokens* enviados através do canal de comunicação. Para lidar com inter-relacionamentos entre tarefas, o LA-DCOP utiliza um tipo adicional de *token*, chamado *potential token*. A decisão de realizar ou não uma tarefa (retendo seu respectivo *token*) é tomada com base em um valor de *threshold*[3] associado ao *token*. O *threshold* representa a competência mínima que um agente deve possuir para realizar a tarefa. Se a competência do agente é superior ao *threshold*, o *token* é retido. Caso contrário, o *token* é enviado para outro agente aleatoriamente selecionado. Com o ajuste de valores adequados de *threshold*, a alocação é obtida em tempo hábil e com recompensa satisfatória (SCERRI et al., 2005).

Ao reter mais de um *token*, o agente deve decidir quais das tarefas (representadas pelos *tokens*) irá de fato realizar. Esta decisão é tomada de forma a maximizar as competências do agente, respeitando seus recursos. Isto é, o agente seleciona um subconjunto de *tokens* cuja soma de recursos necessários seja menor ou igual aos recursos disponíveis, e ao mesmo tempo maximize as competências do agente. Esta escolha é um problema de maximização, que pode ser reduzido para um Problema da Mochila Binário (PMB), que é NP-Completo. Portanto, a complexidade do LA-DCOP depende do método adotado para resolver PMBs. Este fato pode afetar seu desempenho em situações onde a quantidade de *tokens* é muito superior a de agentes. Uma vez que o número de *tokens* retidos pelos agentes aumenta, o tempo consumido para decidir quais de fato alocar também aumenta.

Considerando uma abordam gulosa de resolução de PMBs (adotada nos experimentos desta dissertação), a complexidade do LA-DCOP é da ordem $O(n + n\ log\ n\ + n)$, sendo n a quantidade de *tokens* retidos pelo agente. O primeiro termo da soma origina-se da

[3] A palavra *threshold* pode ser traduzida como limiar. Entretanto, para tornar evidente estar se tratando do algoritmo LA-DCOP, será mantido o uso do original em inglês.

comparação entre a competência do agente e o valor de *threshold*, para cada um dos n *tokens* retidos. Nesta comparação são separados os *tokens* em que a competência do agente é inferior ao *threshold*, sendo repassados a outros agentes. O segundo termo corresponde à complexidade da abordagem gulosa para o PMB formado pelos *tokens* restantes. Nesta abordagem gulosa, a ordenação dos *tokens* em função das competências do agente requer $O(n\ log\ n)$. Após esta ordenação, segue-se no máximo mais n operações para selecionar os *tokens* que estão dentro do limite de recursos disponíveis e que serão de fato alocados, correspondendo ao terceiro termo da soma. Unindo os termos da soma, a complexidade do LA-DCOP é $O(2n + n\ log\ n)$.

O LA-DCOP constitui o primeiro passo em direção à concretização do conceito de *extreme teams*, por ser capaz de resolver E-GAPs em tempo satisfatório. Entretanto, seus autores o avaliaram em um ambiente com apenas as características (i) e (ii) de *extreme teams*. Ferreira Jr. et al. (2009) consideraram o LA-DCOP em um ambiente com as características (i), (ii) e (iii), comparando-o com o algoritmo Swarm-GAP (maiores detalhes sobre esta comparação são apresentados a seguir). A eficiência do LA-DCOP ao se considerar todas as quatro características de *extreme teams* não foi verificada até o momento.

4.4 Swarm-GAP

O Swarm-GAP (FERREIRA JR.; BOFFO; BAZZAN, 2008) é também um algoritmo para alocação de tarefas capaz de resolver E-GAPs. O Swarm-GAP é inspirado na inteligência de enxames, mais especificamente, na metáfora de divisão de trabalho. Agentes no Swarm-GAP decidem quais tarefas realizar com base no modelo de divisão de trabalho dos insetos sociais (Seção 3.1). O limiar de resposta de cada agente é definido em função de suas competências. A partir dos limiares de resposta e de estímulos associados às tarefas, cada agente decide probabilisticamente por realizar ou não as tarefas a partir da tendência, Equação (3.1). O Swarm-GAP também utiliza *tokens* para representar tarefas e repassa *tokens* de tarefas não realizadas para outros agentes.

As tarefas inter-relacionadas, que requerem esforço simultâneo de grupos de agentes, são tratadas no Swarm-GAP de forma implícita, através de um coeficiente de execução. O valor do coeficiente de execução x_j de uma tarefa j é computado pela razão entre o número n_j de tarefas inter-relacionadas com j que estão sendo realizadas simultâneamente e o total N_j de tarefas inter-relacionadas com j, conforme apresentado na Equação (4.1). Este coeficiente é utilizado para incrementar a tendência do agente alocar a tarefa j.

$$x_j = \begin{cases} \frac{1+|n_j|}{|N_j|} & \text{se } |n_j| \neq |N_j| \text{ e } |n_j| > 1 \\ 0 & \text{caso contrário} \end{cases} \tag{4.1}$$

O uso do coeficiente de execução porém, não garante que todas as tarefas inter-relacionadas sejam simultaneamente realizadas. Pode ocorrer de que apenas uma parte das tarefas inter-relacionadas sejam realizadas. Neste caso, os agentes desperdiçam seus recursos com tarefas que não deveriam ser realizadas. Isto porque tarefas inter-relacionadas só podem ser realizadas simultaneamente. Este desperdício de recursos afeta o desempenho do sistema, pois os recursos desperdiçados poderiam estar sendo utilizados para realizar tarefas não inter-relacionadas.

A complexidade do Swarm-GAP é da ordem de $O(n)$, sendo n a quantidade de tarefas percebidas pelo agente. Esta complexidade origina-se da decisão probabilística, baseada na tendência de realizar ou não uma tarefa. A decisão do agente a respeito de quais

tarefas realizar é tomada percorrendo-se a lista com as n tarefas percebidas ou recebidas, e decidindo por alocar ou não cada tarefa, com base na tendência e nos recursos disponíveis.

Ferreira Jr. et al. (2009) deram um passo adiante na concretização do conceito de *extreme teams*, avaliando o Swarm-GAP em um ambiente (o simulador RoboCup Rescue[4]) considerando as características (i), (ii) e (iii). Nesta avaliação, o desempenho do Swarm-GAP também foi comparado com o LA-DCOP. A conclusão dos autores é que o desempenho do Swarm-GAP e LA-DCOP são similares nestas condições. Da mesma forma como ocorre com o LA-DCOP, a eficiência do Swarm-GAP ao se considerar todas as quatro características de *extreme teams* não foi verificada até o momento.

4.5 Outras Abordagens Baseadas em Inteligência de Enxames

O uso de metáforas provenientes da inteligência de enxames para alocação de tarefas já é reportado na literatura. Cicirello e Smith (2004) utilizaram a metáfora de divisão de trabalho para alocar tarefas de pintura à máquinas de uma fábrica de caminhões. As tarefas surgem de uma linha de produção e possuem como característica a cor que deve ser pintada. Os agentes (máquinas) podem pintar apenas uma tarefa por vez. Para realizar tarefas com diferentes características (cores), o agente requer que sua configuração (tinta utilizada) seja alterada. Cada agente decide quais tarefas irá realizar com base em seu limiar de resposta, definido em função da necessidade ou não de alterar sua configuração e da fila de tarefas que já estão atribuidas ao agente. Nesta abordagem, o ambiente é completamente observável (todo agente percebe todas as tarefas). Além disto, a dinâmica do ambiente não atua sobre tarefas já existentes (após sair da linha de produção, as características das tarefas não se alteram). Por fim, não existem tarefas que requerem esforço simultâneo de um grupo de agentes para serem realizadas com sucesso.

Kube e Bonabeau (2000) utilizaram a metáfora do transporte cooperativo em robôs homogêneos que devem forragear[5]. O transporte dos itens deve ocorrer de forma cooperativa e coordenada, visto que os itens utilizados não podem ser transportados por um único robô. Entretanto, não há uso de processos de recrutamento, pois os agentes não necessitam se comunicar, visto que o ambiente é totalmente observável (todos os robôs sabem exatamente a localização do item a ser transportado). Os autores exploram apenas a adaptação da quantidade de agentes ao item, para que seja transportado com sucesso.

Agassounon e Archerio (2002) utilizaram a metáfora de divisão de trabalho em times de agentes homogêneos que devem forragear e agrupar itens. O limiar interno do agente é definido como um tempo máximo de exploração por itens. O estímulo para continuar forrageando considera o tempo que o agente está sem encontrar algum item (maior o tempo, menor o estímulo para forragear). O ambiente é dinâmico, pois novos itens podem aparecer ao longo da simulação. Para lidar com a observação parcial do ambiente, agentes podem interagir quando estiverem próximos, trocando informações sobre o estímulo percebido. Krieger et al. (2000) utilizam, além da divisão de trabalho, a metáfora de recrutamento em times de agentes homogêneos. O agente decide forragear de acordo com seu limiar interno, definido em termos do nível energético da colônia. Quando o nível energético é inferior ao limiar, o agente possui grande tendência de forragear. O ambiente avaliado é estático, havendo sempre a mesma quantidade e localização de itens ao redor do ninho. Nestas duas abordagens, os itens não estão inter-relacionados, isto

[4]Detalhes sobre este simulador serão apresentados na Seção 6.2

[5]Forragear é uma tarefa que compreende a localização de itens, que devem ser transportados para um local específico, normalmente chamado de ninho

é, não requerem esforço simultâneo e coletivo para serem transportados. Neste sentido, o recrutamento da abordagem de Krieger et al. (2000) é utilizado apenas para informar outros robôs da existência de itens em um determinado local. Apesar da relativa simplicidade destes ambientes, os autores reportaram sucesso com uso da divisão de trabalho e recrutamento no desempenho.

4.6 Resumo

A maioria das abordagens apresentadas para alocação de tarefas serão ineficientes ao se considerar *extreme teams*. CNPs requerem que todos os agentes estejam interconectados através de canais de comunicação. Leilões e CNPs, fazem intenso uso de comunicação entre os agentes. Coalizões são computacionalmente caras devido à necessidade de geração e avaliação das possíveis coalizões (combinações de agentes). DCOPs representam E-GAPs de forma complexa, inviabilizando a aplicação dos algoritmos completos devido ao tempo/espaço que requerem.

Abordagens desenvolvidas especificamente para alocação de tarefas em *extreme teams* mostram-se mais eficientes, como é o caso do LA-DCOP (SCERRI et al., 2005) e Swarm-GAP (FERREIRA JR.; BOFFO; BAZZAN, 2008). Entretanto, no LA-DCOP os agentes devem resolver vários PMBs ao longo da execução, possuindo complexidade da ordem $O(2n + n\ log\ n)$ ao ser adotada uma abordagem gulosa para resolução de PMBs (sendo n a quantidade de tarefas percebidas por um agente). O Swarm-GAP apresenta limitações no tratamento de tarefas inter-relacionadas que requerem realização simultânea. Contudo, a complexidade do Swarm-GAP é baixa, sendo da ordem $O(n)$. A eficiência destas abordagens ao se considerar todas as quatro características de *extreme teams* não foi verificada até o momento.

A abordagem proposta nesta dissertação é uma alternativa para tratar da alocação de tarefas em *extreme teams*, pois possui baixa complexidade, requerendo baixo esforço computacional por parte dos agentes. Além disto, a abordagem trata corretamente da alocação de tarefas inter-relacionadas que requerem esforço simultâneo de grupos de agentes, concretizando desta forma o conceito de *extreme teams* em suas quatro características.

5 ALGORITMO eXtreme-Ants

O algoritmo apresentado nesta dissertação, denominado eXtreme-Ants (SANTOS; BAZZAN, 2009a,b), se propõe a alocar tarefas em *extreme teams* de forma eficiente, em ambientes com restrições de tempo e comunicação, considerando todas as quatro caracteristicas envolvidas no conceito de *extreme teams*. Por ser baseado no modelo E-GAP, o eXtreme-Ants pode ser aplicado em qualquer problema de alocação de tarefas que seja formalizado através deste modelo.

Agentes no eXtreme-Ants são autônomos, sendo capazes de reagir rapidamente a mudanças no ambiente sem a intervenção de terceiros (reatividade). Além disto, podem interagir com outros agentes através de comunicação (habilidade social). Esta comunicação tem por objetivo lidar com as limitações de percepção dos agentes. A pró-atividade é suportada pelo eXtreme-Ants, porém não é exigida, ficando a cargo do projetista desenvolvê-la de acordo com as necessidades do ambiente.

Os agentes do eXtreme-Ants são especificados de acordo com uma arquitetura reativa, pois os agentes decidem reativamente quais tarefas irão realizar. Estas decisões seguem o modelo de divisão de trabalho em insetos sociais, que não pressupõe representação simbólica e explicita do ambiente. De acordo com Sycara (1998), esta arquitetura é apropriada para ambientes dinâmicos. Por não possuirem uma representação do ambiente, os agentes reativos não precisam revisar esta representação toda vez que mudanças ocorram no ambiente. Esta revisão exigiria tempo adicional para tomada de decisão.

Para decidir quais tarefas realizar, os agentes no eXtreme-Ants usam o modelo de divisão de trabalho dos insetos sociais (apresentado na Seção 3.1). Os limiares de resposta dos agentes são definidos através do conceito de polimorfismo, isto é, as competências dos agentes são interpretadas como morfologias para especificar seu limiar. Para isto, o limiar de resposta θ_{ij} de um agente i para uma tarefa j corresponde ao inverso da sua competência $Cap(i,j)$, como mostrado na Equação (5.1). Se o agente não é capaz de realizar a tarefa, seu limiar de resposta é definido como infinito, para evitar a realização da tarefa pelo agente. Desta forma, um agente com maior competência para determinada tarefa terá baixo limiar para esta tarefa.

$$\theta_{aj} = \begin{cases} 1 - Cap(i,j) & \text{se } Cap(i,j) > 0 \\ \infty & \text{caso contrário} \end{cases} \tag{5.1}$$

No eXtreme-Ants, cada tarefa $j \in \mathcal{J}$ possui um estímulo s_j associado. O estímulo controla a realização das tarefas pelos agentes. Com baixos valores para os estímulos, as tarefas só serão realizadas por agentes com baixos limiares (ou seja, com maiores competências). Altos valores para os estímulos aumentam a chance das tarefas serem realizadas, mesmo por agentes com altos limiares (com menores competências).

A decisão de realizar ou não uma tarefa é probabilística, com base na tendência $T_{ij}(s_j)$ de realização, Equação (3.1), respeitando os recursos disponíveis no agente. A tendência considera o limiar de resposta do agente e o estímulo associado à tarefa. O eXtreme-Ants normaliza as tendências de realização das tarefas, isto é, cada valor de tendência é dividido pelo somatório das tendências de todas as tarefas percebidas e recebidas. Esta normalização visa fornecer ao agente uma visão mais geral a respeito de todas as tarefas disponíveis. Isto aumenta a chance do agente decidir realizar as tarefas para as quais sua competência é maior.

O modelo E-GAP define uma restrição de exclusão mútua na alocação das tarefas, Equação (2.9). Para lidar com esta restrição de maneira eficiente, o eXtreme-Ants utiliza *tokens* para representar as tarefas. Um agente, de posse de um *token*, tem exclusividade na realização das tarefas nele contidas. Se alguma tarefa contida no *token* não for realizada, o *token* é repassado para outro agente. Desta forma, o eXtreme-Ants evita conflitos na alocação, reduz a comunicação necessária para garantir a exclusão mútua (pois apenas o agente que retém um *token* pode decidir se realiza ou não suas tarefas) e aumenta a percepção dos agentes através dos *tokens* recebidos (aumentando a chance das tarefas serem realizadas, pois um agente que não as percebeu sensorialmente irá percebê-las através da recepção de *tokens*)

O eXtreme-Ants é apresentado no Algoritmo 1. Cada agente i responde a dois eventos: percepção de tarefas e recebimento de mensagem. Ao perceber um conjunto de tarefas $\mathcal{J}_i$ não inter-relacionadas (linha 1), um *token* é criado para armazená-las. Em seguida, o agente decide por realizar ou não as tarefas contidas no *token*, a partir da tendência e dos recursos disponíveis (linhas 27-40). Ao realizar uma tarefa, os recursos do agente são decrementados pela quantidade requerida pela tarefa. Se alguma tarefa não foi realizada, o *token* é enviado para outro agente selecionado aleatoriamente. Ao receber um *token* (linha 25), o agente executa este mesmo processo de decisão para realizar ou não as tarefas nele contidas. Assim como em Scerri et al. (2005), para evitar a ida e vinda de um *token* (o que faria com que ele circulasse apenas entre dois agentes), cada *token* mantém uma lista de agentes visitados e só pode revisitar um agente após todos terem sido visitados.

O uso do modelo de divisão de trabalho proporciona decisões rápidas e eficientes, o que permite que os agentes atuem em ambientes dinâmicos, atendendo a característica (i) de *extreme teams*. A característica (ii), que especifica que os agentes podem realizar múltiplas tarefas respeitando os recursos disponíveis, é atendida ao se considerar, na tomada de decisão, não apenas a tendência, mas também os recursos disponíveis no agente (linhas 7 e 36). A característica (iii), que define que os agentes podem possuir funcionalidades sobrepostas (sendo heterogêneos), é atendida pelo limiar de resposta, Equação (5.1), que reflete as competências do agente para realização de tarefas.

Para lidar com tarefas inter-relacionadas que requerem realização simultânea, atendendo a característica (iv) de *extreme teams*, os agentes no eXtreme-Ants reproduzem o processo de recrutamento para transporte cooperativo. Através do recrutamento é formado um grupo de agentes comprometidos com a realização simultânea das tarefas inter-relacionadas. Ao perceber um conjunto de tarefas α inter-relacionadas (linha 6), o agente comporta-se como uma formiga *scout*. Inicialmente tenta realizar sozinho todas as tarefas (linhas 7-19). Se não conseguir, ele inicia um processo de recrutamento.

O processo de recrutamento para transporte cooperativo presente nos insetos sociais (apresentado na Seção 3.2), é composto por três etapas:

1. A *scout* que descobriu a presa sinaliza o recrutamento através de feromônios.

Algoritmo 1: eXtreme-Ants para um agente i

```
1  quando perceber conjunto de tarefas J_i
2      token := novoToken();
3      para cada j ∈ J_i faça
4          token.tarefas := token.tarefas ∪ j;
5      avaliarToken(token);
6  quando perceber conjunto α de tarefas inter-relacionadas
7      se i.res é suficiente para realizar todas as tarefas de α
8          /* calcular tendências */
9          para cada j ∈ α faça
10             T_ij = s_j^2 / (s_j^2 + θ_ij^2);
11         /* normalizar tendências */
12         para cada j ∈ α faça
13             T_ij = T_ij / Σ_{j∈α_k} T_ij;
14         /* decidir por realizar tarefas */
15         para cada j ∈ α faça
16             se roleta() < T_ij
17                 agente aceita realizar tarefa j;
18         se agente aceitou realizar todas as tarefas de α
19             realiza tarefas e decrementa i.res;
20         senão
21             descarta realização das tarefas de α (linhas 7-19);
22             realizarRecrutamento(α);
23     senão
24         realizarRecrutamento(α);
25 quando receber token
26     avaliarToken(token);
27 procedimento avaliarToken(token)
28     /* calcular tendências */
29     para cada j ∈ token.tarefas faça
30         T_ij = s_j^2 / (s_j^2 + θ_ij^2);
31     /* normalizar tendências */
32     para cada j ∈ token.tarefas faça
33         T_ij = T_ij / Σ_{j∈token.tarefas} T_ij;
34     /* decidir por realizar tarefas */
35     para cada j ∈ token.tarefas faça
36         se roleta() < T_ij e i.res ≥ Res(i, j)
37             realiza tarefa j e decrementa i.res;
38             token.tarefas := token.tarefas − j;
39     se alguma tarefa de token.tarefas não foi realizada
40         enviar token para um agente selecionado aleatoriamente;
```

2. Formigas que aceitaram participar do recrutamento deslocam-se até a presa.

3. O grupo de formigas de fato envolvidas no transporte é adaptado à presa.

A forma como estas três etapas ocorrem no eXtreme-Ants está relacionada com as características do ambiente quanto a comunicação e movimentação dos agentes. A seguir são apresentadas duas propostas para realização de recrutamento. A primeira delas é voltada para ambientes em que a interação entre os agentes só é possível através de comunicação. A segunda é voltada para ambientes em que, além da comunicação, os agentes

podem interagir através de movimentação no ambiente. Nestas duas abordagens, assume-se que os agentes não são dotados de mecanismos para liberar feromônio no ambiente. Portanto, estas abordagens utilizam o canal de comunicação para enviar mensagens, que assumem o papel do feromônio. O uso de feromônios, ao invés de mensagens, para realizar o recrutamento é sugerido como trabalho futuro.

5.1 Recrutamento com comunicação

Em ambientes onde as interações entre os agente só são possíveis através do canal de comunicação, as etapas envolvidas no processo de recrutamento devem ser adaptadas para utilizarem mensagens. Estas mensagens formam um protocolo, apresentado no Algoritmo 2, que simula as etapas do recrutamento.

O protolo utiliza cinco tipos de mensagens:

recrutamento: para anunciar o recrutamento, requisitando que um agente se comprometa com a realização de uma tarefa inter-relacionada $j \in \alpha$;

comprometido: para informar que um agente aceitou participar do recrutamento e se comprometeu a realizar a tarefa $j \in \alpha$;

engajar: para informar que um agente foi, de fato, selecionado para realizar a tarefa $j \in \alpha$;

liberar: para informar que um agente não foi o selecionado para realizar a tarefa $j \in \alpha$ e deve descomprometer-se de realizá-la;

timeout: para informar que uma requisição de recrutamento de uma tarefa $j \in \alpha$ atingiu seu *timeout*.

Na primeira etapa, o agente *scout* envia, para cada tarefa inter-relacionada $j \in \alpha$ que foi percebida, um certo número de mensagens **recrutamento** (linhas 1-5). Estas requisições são enviadas a agentes escolhidos aleatoriamente. Como ocorre com a formiga *scout*, que recruta um número fixo de companheiras, o recrutamento com comunicação do eXtreme-Ants define um número máximo de requisições de recrutamento que devem ser enviados para cada tarefa inter-relacionada. Este número deve ser determinado experimentalmente de forma a maximizar o desempenho dos agentes na realização das tarefas.

Na segunda etapa, os agentes devem decidir se participam ou não do recrutamento. Ao receber uma requisição de recrutamento para uma tarefa j (linhas 6-15), o agente deve decidir se aceita ou não participar do recrutamento. Esta decisão é tomada com base na tendência de realização da tarefa, Equação (3.1), evitando se comprometer com a mesma requisição duplamente. Se a requisição de participação for aceita, o agente se compromete com uma eventual futura realização da tarefa, reservando a quantidade necessária de recursos. Uma mensagem **comprometido** é enviada ao remetente da requisição para indicar o comprometimento. Se a requisição de recrutamento não for aceita, ela é encaminhada a outro agente selecionado aleatoriamente, formando a cadeia de comunicação existente no recrutamento dos insetos.

Na terceira etapa deve-se definir o grupo de agentes que irá realizar simultaneamente as tarefas inter-relacionadas. Este grupo é definido a partir daqueles agentes que decidiram participar do recrutamento e se comprometeram com a eventual realização das

Algoritmo 2: Recrutamento com comunicação do eXtreme-Ants (agente i)

```
1  procedimento realizarRecrutamento(conjunto α de tarefas inter-relacionadas)
2      repita
3          j := seleciona uma tarefa de α;
4          enviar("recrutamento", j) para um agente selecionado aleatoriamente ;
5      até atingir o número máximo de requisições enviadas para cada tarefa j ∈ α ou o
       recrutamento para α estiver finalizado ou abortado.
6  quando receber ("recrutamento", j) de i_s
7      /* decide comprometer-se ou não com o recrutamento*/
8      se roleta() < T_ij e i.res ≥ Res(i, j) e i não está comprometido com j
9          participa do recrutamento e compromete-se com j;
10         enviar("comprometido", j) para i_s;
11     senão
12         se atingiu o timeout da requisição de recrutamento
13             enviar("timeout", j) para i_s;
14         senão
15             encaminhar("recrutamento", j) para um agente selecionado aleatoriamente;
16 quando receber ("comprometido", j) de i_c
17     α := conjunto de tarefas inter-relacionadas que contém j
18     se recrutamento para α está finalizado ou abortado
19         enviar("liberar", j) para i_c; retorna
20     se ao menos um agente se comprometeu com cada j ∈ α
21         recrutamento para α está finalizado!
22         /* formar o grupo de agentes engajados na realização simultânea*/
23         para cada j ∈ α faça
24             selecionar agente i_p com prob. proporcional à Cap(i_p, j);
25             enviar("engajar", j) para i_p;
26             enviar("liberar", j) para cada agente não selecionado;
27 quando receber ("engajar", j)
28     realiza tarefa j e decrementa i.res;
29 quando receber ("liberar", j)
30     descompromete-se de realizar j;
31 quando receber ("timeout", j)
32     α := conjunto de tarefas inter-relacionadas que contém j
33     se a quantidade de timeouts recebidos para cada tarefa j ∈ α é igual a quantidade de
       requisições enviadas
34         recrutamento para α está abortado!
35         para cada j ∈ α faça
36             enviar("liberar", j) para agentes que se comprometeram;
```

tarefas. Quando o *scout* recebe comprometimentos suficientes para cada tarefa inter-relacionada $j \in \alpha$ (linha 20), ele define o grupo de agentes que irão realizar simultaneamente as tarefas de α. Seguindo a definição E-GAP, apenas um agente deve ser selecionado para cada tarefa. O agente *scout* realiza uma escolha probabilística, selecionando, para cada tarefa $j \in \alpha$ um agente i_p com probabilidade proporcional a sua competência $Cap(i_p, j)$. O *scout* então informa i_p que ele foi o selecionado e portando deve se engajar na realização de j (através de mensagem **engajar**, linha 25). Os agentes não selecionados são liberados (através de mensagens **liberar**, linha 26). Agentes que se comprometem com tarefas já alocadas também são liberados, evitando *deadlocks* (linha 18). Neste momento o recrutamento é finalizado. Como resultado, um grupo de agentes está formado, onde cada agente está engajado na realização de uma tarefa $j \in \alpha$, habilitando a realização simultânea de todas as tarefas de α.

Após o grupo de agentes ter sido formado, as requisições de recrutamento que ainda não foram aceitas por algum agente tornam-se obsoletas. Para evitar que os agentes passem adiante requisições obsoletas, o recrutamento com comunicação do eXtreme-Ants introduz um mecanismo de *timeout*. O *timeout* é definido através de uma quantidade de agentes que uma requisição pode visitar. Quando o *timeout* é detectado (linha 12), o agente *scout* que originou a requisição deve ser notificado. Ao receber *timeout* de todas as requisições enviadas para uma tarefa $j \in \alpha$, o recrutamento de todo o conjunto α é abortado e os agentes que eventualmente haviam se comprometido são liberados (linhas 31-36).

5.2 Recrutamento com comunicação e movimentação

Em ambientes onde os agentes podem interagir através de comunicação e movimentação no ambiente, as etapas envolvidas no processo de recrutamento não requerem grandes adaptações, como ocorreu com o recrutamento apenas com comunicação. A única adaptação requerida está na primeira etapa, que utiliza mensagens para sinalizar o recrutamento. O Algoritmo 3 apresenta o recrutamento com comunicação e movimentação do eXtreme-Ants.

Algoritmo 3: Recrutamento com comunicação e movimentação do eXtreme-Ants (agente i)

```
1  procedimento realizarRecrutamento(conjunto α de tarefas inter-relacionadas)
2      para cada j ∈ α faça
3          enviar("recrutamento", j) para um agente selecionado aleatoriamente;
4  quando receber ("recrutamento", j)
5      se roleta() < T_ij e i.res ≥ Res(i, j) e i não está comprometido com j
6          participa do recrutamento, movendo-se até a tarefa j;
7      senão
8          encaminhar("recrutamento", j) para um agente selecionado aleatoriamente;
```

Nas demais etapas do processo de recrutamento, os agentes comportam-se exatamente da mesma forma que as formigas, visto que podem se movimentar no ambiente. Na segunda etapa, os agentes que aceitam participar do recrutamento deslocam-se até a tarefa. Na terceira etapa, os agentes que chegaram até as tarefas iniciam sua realização, eventualmente adaptando-se ao espaço físico existente ao redor das tarefas.

5.3 Análise de Complexidade

A complexidade de tempo do eXtreme-Ants é determinada em função das operações envolvidas na tomada de decisão do agente a respeito de quais tarefas irá realizar. Seja n a quantidade de tarefas com e sem inter-relacionamentos percebidas pelo agente. Esta quantidade engloba as tarefas percebidas sensorialmente e também as tarefas recebidas através do canal de comunicação (a partir do recebimento de *tokens* e de requisições de recrutamento). A complexidade de tempo do eXtreme-Ants é da ordem de $O(n+n+n) = O(3n) = O(n)$.

O primeiro termo da soma corresponde aos passos necessários para calcular as tendências de realização, Equação (3.1) das n tarefas (Algoritmo 1, linhas 29-30). O segundo termo corresponde à etapa de normalização destas tendências (Algoritmo 1, linha 31). Por

fim, o terceiro termo corresponde à decisão de realizar ou não as tarefas (ou comprometer-se com o recrutamento) considerando as tendências normalizadas e os recursos disponíveis (Algoritmo 1, linhas 35-38).

A complexidade de comunicação do eXtreme-Ants é determinada em função da quantidade de mensagens enviadas por um agente para cada tarefa percebida. Esta quantidade depende da existência ou não de inter-relacionamentos.

Para um conjunto $\mathcal{J}_i$ de tarefas não inter-relacionadas percebidas, a quantidade de mensagens enviadas é exatamente 1, pois apenas um *token* é criado para representá-las (Algoritmo 1, linha 2). Neste caso, a complexidade de comunicação é $O(1)$.

Para um conjunto α de tarefas inter-relacionadas percebidas, a quantidade de mensagens enviadas pelo processo de recrutamento é dada em função da cardinalidade de α. No recrutamento com comunicação e movimentação, são enviadas exatamente 1 requisição de recrutamento para cada tarefa $j \in \alpha$ (Algoritmo 3, linha 3), gerando complexidade de comunicação $O(|\alpha|)$. No recrutamento apenas com comunicação, a quantidade de requisições de recrutamento enviadas para cada tarefa inter-relacionada deve ser determinada experimentalmente, de forma a maximizar a qualidade das alocações (Algoritmo 2, linha 4). Sendo qtd_req a quantidade de requisições enviadas para cada tarefa no processo de recrutamento com comunicação, então complexidade de comunicação neste recrutamento é da ordem de $O(|\alpha| \times qtd\_req)$.

5.4 Resumo

Este capítulo apresentou o algoritmo eXtreme-Ants para alocação de tarefas em *extreme teams*. O algoritmo é capaz de atuar em situações onde existem tarefas que requerem esforço simultâneo de um grupo de agentes, além de restrições de tempo e comunicação. Por ser baseado no modelo E-GAP, o eXtreme-Ants pode ser aplicado em qualquer problema de alocação de tarefas formalizado através deste modelo.

Agentes do eXtreme-Ants são autônomos e especificados utilizando uma arquitetura reativa. Para decidir quais tarefas realizar, os agentes utilizam o modelo de divisão de trabalho dos insetos sociais. Por ser uma decisão probabilística, o eXtreme-Ants apresenta baixa complexidade, sendo da ordem de $O(n)$. Esta baixa complexidade habilita os agentes a atuarem em cenários com restrições no tempo disponível para tomar decisões. Para reduzir a comunicação, o eXtreme-Ants utiliza *tokens* para representar as tarefas. Os *tokens* resolvem naturalmente a restrição de exclusão mútua na alocação, evitando a necessidade de comunicação adicional, e otimizando o uso do canal de comunicação.

Para lidar com tarefas inter-relacionadas que requerem realização simultânea, os agentes no eXtreme-Ants reproduzem o processo de recrutamento para transporte cooperativo. Duas abordagens para a realização de recrutamento foram apresentadas. A primeira é orientada à ambientes em que a interação entre os agentes só é possível através de comunicação. Consequentemente, é necessário uso adicional do canal de comunicação para realizar o recrutamento. A segunda é voltada para ambientes em que os agentes podem interagir por comunicação e por movimentação, como é o caso do RoboCup Rescue.

A computação no eXtreme-Ants é assíncrona. Os agentes reagem aos eventos na medida em que ocorrem. Além disto, enquanto o processo de recrutamento está em curso, os agentes podem realizar outras tarefas. Assim como ocorre com os insetos socias, não existe controle global ou central no eXtreme-Ants. Do ponto de vista das interações entre os agentes, o eXtreme-Ants pode ser visto como um mecanismo de coordenação, que organiza o comportamento dos agentes para realizar eficientemente as tarefas.

O modelo de divisão de trabalho proporciona decisões rápidas e eficientes, atendendo as características (i), (ii), e (iii) de *extreme teams*. O recrutamento permite formar grupos de agentes comprometidos com a realização simultânea de tarefas que exigem esforço conjunto, atendendo a característica (iv). Com isto, concretiza-se de fato o conceito completo de *extreme teams*.

6 RESULTADOS EXPERIMENTAIS

O eXtreme-Ants foi avaliado através de uma série de experimentos em dois ambientes distintos: um simulador independente de domínio e o simulador RoboCup Rescue. Nestes ambientes, o eXtreme-Ants foi comparado com outros dois algoritmos para alocação de tarefas em *extreme teams*: Swarm-GAP (FERREIRA JR.; BOFFO; BAZZAN, 2008) e LA-DCOP (SCERRI et al., 2005).

As métricas consideradas visam avaliar a eficiência dos algoritmos na alocação de tarefas em E-GAPs. Esta eficiência está relacionada com desempenho dos agentes, sendo uma das métricas avaliadas (o desempenho diz respeito ao sucesso dos agentes na realização das tarefas e é detalhado na sequência, em cada um dos ambientes avaliados). Em ambientes com restrições de comunicação, a eficiência também está relacionada com a otimização do uso do canal de comunicação. Neste sentido, a segunda métrica avaliada é a utilização do canal de comunicação. Além disto, em ambientes com restrições de tempo os agentes devem decidir quais tarefas irão alocar de forma ágil. O esforço computacional dos agentes reflete esta agilidade, sendo a terceira métrica avaliada.

6.1 Simulador independente de domínio

O desenvolvimento e utilização de um simulador independente de domínio para realizar experimentos tem por objetivo avaliar o eXtreme-Ants nas mesmas condições em que o LA-DCOP e Swarm-GAP foram avaliados por seus respectivos autores, com E-GAPs de grande quantidade de agentes e tarefas. A motivação para adotar as mesmas condições de experimentação é permitir comparação adequada entre os algoritmos.

Cada experimento consiste de um E-GAP com 2000 tarefas, agrupadas em 5 classes, sendo que cada classe determina as características das tarefas. A quantidade de agentes variou de 500 a 4000. Cada agente tem 60% de probabilidade de ter competência diferente de zero para cada classe de tarefas. Neste caso, as competências são uniformemente distribuídas com valores no intervalo $[0, 1]$. Cada agente também possui uma unidade de recurso, isto é, $i.res = 1.0$. A quantidade de recursos que cada classe de tarefa consome é uniformemente distribuida entre os valores $\{0.25, 0.50, 0.75\}$. 60% de todas as tarefas são separadas em grupos de 5 e inter-relacionadas por **E**, ou seja, precisam ser realizadas simultaneamente. A implementação do eXtreme-Ants neste simulador utiliza o recrutamento com comunicação (Seção 5.1) para alocar as tarefas inter-relacionadas.

A comunicação entre os agentes é confiável (toda mensagem enviada é recebida) e todos os agentes estão inter-conectados pelo canal de comunicação. Cada simulação considera 1000 instantes de tempo. A quantidade total de tarefas ao longo da simulação é mantida constante. Para simular a dinâmica do ambiente, cada tarefa possui uma probabilidade de 10% de ser substituída por outra (através da alteração de sua classe) a cada

instante de tempo. Com a alteração de classe, a tarefa irá requerer uma competência diferente para ser realizada.

As tarefas são persistentes, isto é, tarefas não realizadas em um instante de tempo são mantidas no próximo. Em cada instante de tempo, cada *token* ou mensagem pode se mover de um agente a outro apenas uma vez. Apesar de que cada tarefa pode possuir estímulo ou *threshold* diferente das outras, adota-se o mesmo valor para todas. Cada ponto nos gráficos que seguem representam uma média de 20 execuções. A análise dos resultados foi feita com base em testes t com 99% de confiança. A seguir são apresentados os resultados obtidos em cada métrica avaliada.

6.1.1 Desempenho relativo às recompensas totais

Conforme definido no modelo E-GAP (Seção 2.2.1), o desempenho dos agentes na realização das tarefas é dado pela recompensa total do sistema. Esta recompensa total corresponde à soma das competências dos agentes que participam da alocação em cada instante de tempo ao longo da simulação, conforme definido na Equação (2.7).

Cada um dos algoritmos avaliados possui um parâmetro específico (*threshold* no LA-DCOP e estímulo no eXtreme-Ants e Swarm-GAP), cujo valor deve ser determinado experimentalmente de forma a maximizar a recompensa total. Cada um dos algoritmos foi testado com valores de parâmetro no intervalo $[0.0, 0.9]$ para determinar qual valor proporciona a melhor recompensa total em cada situação (quantidade de agentes). Os valores que proporcionaram melhor recompensa total, e que são utilizados nos demais experimentos, são apresentados na Tabela 6.1. Adicionalmente, no caso do eXtreme-Ants as recompensas totais foram obtidas com cinco requisições de recrutamento para cada tarefa inter-relacionada e com um *timeout* de 20 agentes visitados. A Figura 6.1 apresenta os desempenhos obtidos com os parâmetros da Tabela 6.1.

Tabela 6.1: Valores de parâmetros que proporcionam melhor recompensa total.

Algoritmo	Parâmetro	Quantidade de agentes							
		500	1000	1500	2000	2500	3000	3500	4000
eXtreme-Ants	estímulo	0.3	0.3	0.2	0.2	0.2	0.2	0.2	0.2
Swarm-GAP	estímulo	0.2	0.3	0.2	0.2	0.2	0.2	0.2	0.2
LA-DCOP	*threshold*	0.0	0.4	0.6	0.6	0.6	0.6	0.7	0.7

O eXtreme-Ants obteve recompensas totais superiores ao Swarm-GAP para todas as quantidades de agentes. Entretanto, o LA-DCOP obteve recompensas totais superiores ao eXtreme-Ants. Para elucidar a diferença entre as recompensas totais, a Tabela 6.2 apresenta a relação, em termos percentuais, entre as recompensas do Swarm-GAP e LA-DCOP e a recompensa do eXtreme-Ants. Por exemplo, no caso com 500 agentes, a recompensa total do Swarm-GAP e LA-DCOP correspondem, respectivamente, a 70.62% e 110.76% das recompensas do eXtreme-Ants.

As menores recompensas totais do Swarm-GAP estão relacionadas com a maneira que este lida com tarefas inter-relacionadas. O uso do coeficiente de execução não é suficiente para garantir a realização simultânea destas tarefas. Os agentes utilizam seus recursos realizando apenas parcialmente os grupos de tarefas inter-relacionadas. Isto prejudica a recompensa total, que só considera grupos onde todas as tarefas estejam sendo realizadas simultaneamente, conforme define a Equação (2.6). O eXtreme-Ants e o LA-DCOP obtiveram melhores recompensas totais devido a existência de um mecanismo de coordenação explícita para lidar com tarefas que apresentam inter-relacionamentos, garantindo

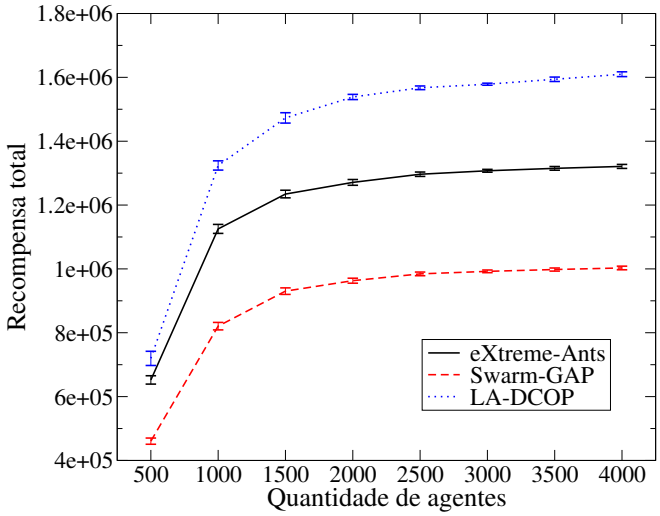

Figura 6.1: Recompensas totais obtidas no simulador independente de domínio. A recompensa corresponde à soma das competências dos agentes que participaram das alocações ao longo da simulação.

Tabela 6.2: Recompensas totais do Swarm-GAP e LA-DCOP em relação ao eXtreme-Ants (em percentual).

Agentes	eXtreme-Ants	Swarm-GAP	LA-DCOP
500	$0.652 \times 10^6 \pm 0.013 \times 10^6$	70.62%	110.76%
1000	$1.125 \times 10^6 \pm 0.014 \times 10^6$	72.95%	112.97%
1500	$1.234 \times 10^6 \pm 0.012 \times 10^6$	75.42%	116.98%
2000	$1.271 \times 10^6 \pm 0.009 \times 10^6$	75.72%	117.24%
2500	$1.297 \times 10^6 \pm 0.007 \times 10^6$	75.93%	116.43%
3000	$1.307 \times 10^6 \pm 0.004 \times 10^6$	75.90%	117.85%
3500	$1.315 \times 10^6 \pm 0.006 \times 10^6$	75.90%	118.51%
4000	$1.321 \times 10^6 \pm 0.006 \times 10^6$	75.95%	118.83%

realização simultânea.

As maiores recompensas totais do LA-DCOP estão relacionados com a maximização das competências dos agentes ao decidir quais tarefas serão realizadas. Por outro lado, agentes no eXtreme-Ants decidem de maneira simples, com base na tendência. Entretanto, no LA-DCOP há um compromisso entre as recompensas totais obtidos e a quantidade de comunicação/esforço computacional requeridos.

6.1.2 Comunicação

Este experimento, mostrado na Figura 6.2, compara a utilização do canal de comunicação por cada algoritmo. A utilização corresponde a quantidade total de mensagens enviadas pelos agentes ao longo da simulação, independente do tipo de mensagem (e.g. *token*, requisições de recrutamento, etc.).

A quantidade de mensagens enviadas pelo eXtreme-Ants é superior ao Swarm-GAP e inferior ao LA-DCOP. A Tabela 6.3 apresenta a utilização do canal de comunicação no Swarm-GAP e LA-DCOP em relação ao eXtreme-Ants.

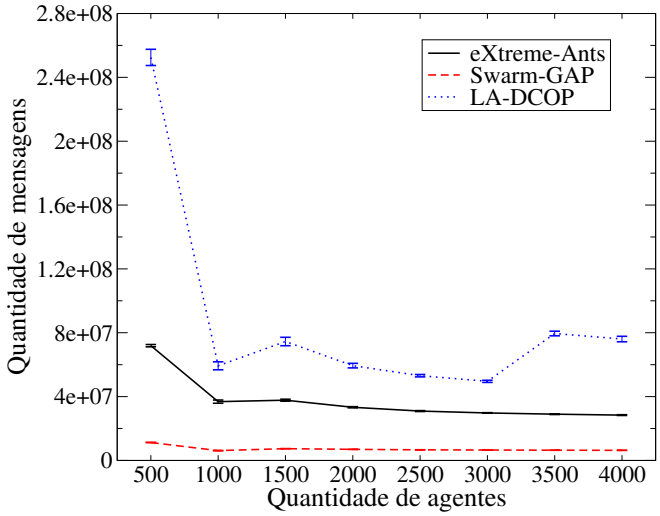

Figura 6.2: Uso do canal de comunicação, medido como a quantidade total de mensagens enviadas pelos agentes no simulador independente de domínio.

Tabela 6.3: Uso do canal de comunicação no Swarm-GAP e LA-DCOP em relação ao eXtreme-Ants (em percentual).

Agentes	eXtreme-Ants	Swarm-GAP	LA-DCOP
500	$7.191 \times 10^7 \pm 0.074 \times 10^7$	15.63%	351.10%
1000	$3.683 \times 10^7 \pm 0.101 \times 10^7$	16.69%	160.95%
1500	$3.774 \times 10^7 \pm 0.063 \times 10^7$	19.53%	197.42%
2000	$3.331 \times 10^7 \pm 0.038 \times 10^7$	20.84%	178.31%
2500	$3.094 \times 10^7 \pm 0.030 \times 10^7$	21.50%	171.64%
3000	$2.976 \times 10^7 \pm 0.019 \times 10^7$	21.88%	166.55%
3500	$2.898 \times 10^7 \pm 0.022 \times 10^7$	22.12%	274.37%
4000	$2.841 \times 10^7 \pm 0.025 \times 10^7$	22.31%	267.78%

A menor diferença do eXtreme-Ants em relação ao LA-DCOP ocorre no caso com 1000 agentes, onde o LA-DCOP requer 160.95% do total de mensagens utilizadas pelo eXtreme-Ants. A diferença mais crítica ocorre no caso com 500 agentes, onde o LA-DCOP requer 351.10% das mensagens enviadas pelo eXtreme-Ants. As quantidades de mensagens enviadas pelo LA-DCOP são maiores que as do eXtreme-Ants em função da sua decisão determinística, baseada na maximização das competências dos agentes. Este processo de decisão pode fazer com que cada agente realize poucas tarefas cujas competências são máximas, mas que consomem todos os seus recursos. As tarefas recusadas são repassadas para outros agentes, o que explica a maior utilização do canal de comunicação. Já no eXtreme-Ants cada agente pode não realizar tarefas cujas competências são máximas (pois a decisão é probabilística) e que eventualmente consumiriam todos os seus recursos. Desta forma, cada agente pode realizar maior quantidade de tarefas que consomem menos recursos e cujas competências são razoáveis. A consequência disto é a diminuição da quantidade de tarefas que devem ser repassadas a outros agentes.

O algoritmo Swarm-GAP envia menor quantidade de mensagens que o eXtreme-Ants e o LA-DCOP devido a forma como lida com tarefas inter-relacionadas. A ausência de um mecanismo de coordenação explícita para garantir a realização simultânea leva a uma

menor quantidade de mensagens, porém tem um grande impacto na recompensa total, como mostrado anteriormente. Só é possível garantir a realização simultânea de tarefas inter-relacionadas no simulador independente de domínio através do uso de comunicação adicional.

6.1.3 Esforço computacional

Este experimento avalia o esforço computacional dos agentes em cada algoritmo. Este esforço computacional corresponde à quantidade de operações realizadas pelo agente para tomar decisões a respeito de quais tarefas realizar, em cada instante de tempo.

Para cada tarefa (ou conjunto de tarefas) percebida, a quantidade de operações realizadas por um agente para decidir realizá-la(s) ou não corresponde a complexidade de cada algoritmo. O mesmo se aplica às tarefas recebidas através de mensagens pelo canal de comunicação. No caso do simulador independente de domínio, os agentes são totalmente assíncronos e não há limite sobre a quantidade de mensagens que cada agente pode enviar/receber em cada instante de tempo. Quanto maior a quantidade de mensagens recebidas (isto é, tarefas), maior será a quantidade de vezes que o agente deve tomar decisões. Isto quer dizer que o esforço computacional é também influenciado pela comunicação que ocorre entre os agentes.

Um baixo esforço computacional significa que os agentes são mais eficientes para atuarem em ambientes onde o tempo disponível para tomar decisões é restrito. A Figura 6.3 apresenta o esforço computational de cada agente em cada instante de tempo. O gráfico externo enfatiza a área que concentra a maioria dos pontos, enquanto o gráfico interno mostra a área total, apenas para exibir os pontos não exibidos no gráfico externo.

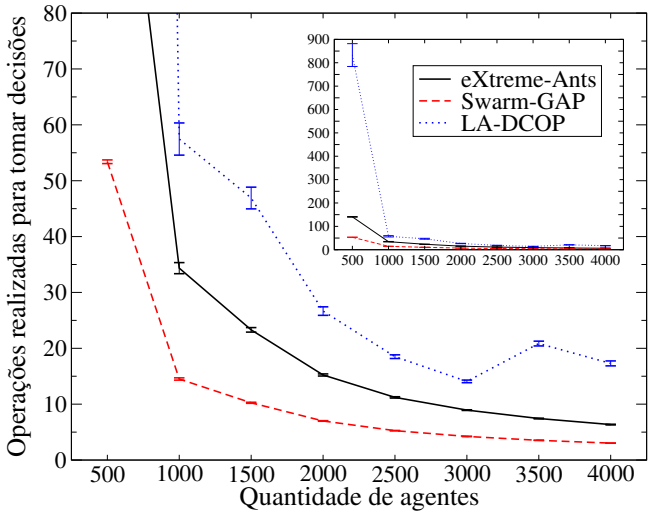

Figura 6.3: Esforço computational, medido pela quantidade de operações realizadas por cada agente para tomar decisões em cada instante de tempo no simulador independente de domínio.

Os esforços computacionais do eXtreme-Ants são superiores ao Swarm-GAP e inferiores ao LA-DCOP. A Tabela 6.4 apresenta os esforços computacionais do Swarm-GAP e LA-DCOP em relação ao eXtreme-Ants.

A menor diferença entre o eXtreme-Ants e o LA-DCOP ocorre no caso com 3000 agentes, onde o esforço computacional de cada agente no LA-DCOP corresponde a 157.51% dos agentes no eXtreme-Ants. A diferença mais significativa ocorre no caso com

Tabela 6.4: Esforço computacional do Swarm-GAP e LA-DCOP em relação ao eXtreme-Ants (em percentual).

Agentes	eXtreme-Ants	Swarm-GAP	LA-DCOP
500	$1.405 \times 10^2 \pm 0.014 \times 10^2$	37.99%	592.69%
1000	$0.344 \times 10^2 \pm 0.010 \times 10^2$	42.23%	167.27%
1500	$0.233 \times 10^2 \pm 0.004 \times 10^2$	43.98%	201.29%
2000	$0.152 \times 10^2 \pm 0.002 \times 10^2$	45.93%	175.09%
2500	$0.112 \times 10^2 \pm 0.001 \times 10^2$	46.89%	164.98%
3000	$0.089 \times 10^2 \pm 0.001 \times 10^2$	47.28%	157.51%
3500	$0.074 \times 10^2 \pm 0.001 \times 10^2$	47.52%	280.62%
4000	$0.064 \times 10^2 \pm 0.001 \times 10^2$	47.70%	272.29%

500 agentes, onde o esforço computacional de cada agentes no LA-DCOP corresponde a 592.69% dos agentes no eXtreme-Ants. Estas diferenças são reflexo da complexidade e da quantidade de comunicação do LA-DCOP. A maximização das competências dos agentes na decisão de quais tarefas realizar leva à melhores recompensas totais, porém exigem maior esforço computacional dos agentes em relação ao eXtreme-Ants.

Os esforços computacionais do Swarm-GAP são menores que os do eXtreme-Ants novamente em função da maneira como lida com tarefas inter-relacionadas. Como não há mecanismo de coordenação explicita para lidar com inter-relacionamentos entre tarefas, os agentes não necessitam fazer avaliações adicionais relativas à realização simultânea, reduzindo desta forma os esforços computacionais do Swarm-GAP. Entretanto, a ausência deste mecanismo afeta as recompensas totais, como apresentado previamente.

De modo geral, os algoritmos apresentam diminuição do esforço computacional com o aumento da quantidade de agentes. Este comportamento deve-se ao fato de que maiores quantidades de agentes reduzem a carga (isto é, a relação tarefas por agente) do sistema. Isto reduz a quantidade de tarefas que cada agente deve decidir por alocar ou não, diminuindo o esforço computacional.

6.2 Simulador RoboCup Rescue

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 infraestrutura de energia, água e comunicação foi extremamente danificada. Aproximadamente 6.000 pessoas morreram, e mais de 300.000 ficaram feridas.

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 (KITANO, 2000). Em função disto, fundou-se a *RoboCup Rescue Simulation League* (RRSL), onde são centralizados os esforços na busca deste sistema. Esta liga disponibiliza um simulador de desastres (terremotos) e operações de resgate, denominado RoboCup Rescue (SKINNER; BARLEY, 2006). Neste simulador é possível avaliar a qualidade e eficiência de abordagens multiagente no que tange ao salvamento de pessoas e minimização de danos. Questões como heterogeneidade, acesso limitado à informação, comunicação limitada e planejamento em tempo real caracterizam o RoboCup Rescue como um domínio multiagente complexo (KITANO, 2000). A seguir

é apresentada uma breve descrição do simulador RoboCup Rescue.

Como entrada, o simulador RoboCup Rescue recebe dados geográficos (mapas, ruas, construções, etc.) e informações sobre o 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 objetos presentes no cenário (agentes, ruas, etc). Em seguida, 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.

Para lidar com essa situação e minimizar os danos (humanos e materiais), o simulador incopora 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 6.5 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 Ambulância	resgatar civis soterrados e/ou feridos	campo
Central de Ambulâncias	centralizar a coordenação de Times de Ambulância	central

Tabela 6.5: Agentes de resgate existentes no RoboCup Rescue com seus papéis e classes

A interação com o ambiente, através de percepção-ação, é feita exclusivamente por agentes de campo. A percepção é visual, limitada a um raio de 10 metros. A Figura 6.4 ilustra graficamente o esquema de percepção dos agentes. Cada agente pode perceber qualquer entidade localizada dentro de seu raio de visão (construções, bloqueios, civis, etc.). No exemplo da Figura 6.4, o agente A1 percebe as construções C2 e C3, os bloqueios B2 e B3 e o civil Civ1. Com relação a 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, a saúde de um civil, etc.

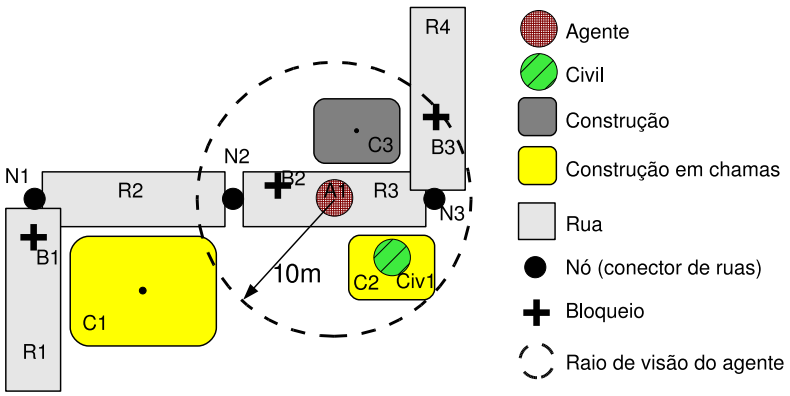

Figura 6.4: Esquema de percepção dos agentes no simulador RoboCup Rescue

As ações que podem ser realizadas pelos agentes de campo no ambiente são apresentadas na Tabela 6.6. 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 RoboCup Rescue um ambiente dinâmico, limitando o tempo de decisão disponível aos agentes. Cada agente possui apenas meio 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 ação irá executar neste meio segundo, o simulador considera que este agente optou por não agir no respectivo instante de tempo.

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 6.6: Agentes de campo da RoboCup Rescue e suas possíveis ações

A interação entre os diferentes tipos de agente é feita através de comunicação. No 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 em 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, mas com base exclusivamente 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 em 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 construida preservada (não incendiada) e a

quantidade de sobreviventes, conforme apresenta a Equação (6.1).

$$escore = \left(P + \frac{H}{H_{inicial}} \right) * \sqrt{\frac{B}{B_{inicial}}} \tag{6.1}$$

sendo:

- 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.

O escore não considera diretamente os bloqueios removidos das ruas. Entretanto, a remoção destes bloqueios afeta diretamente o escore, pois otimiza o fluxo 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 SMA, o ambiente é coletivamente parcialmente observável, pois mesmo unindo a perceção 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 incêncios (NÜSLE; KLEINER; BRENNER, 2004), 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. Com isto, a decisão sobre qual outro incêndio irá combater não levará em consideração as construções vizinhas ao incêndio extinto, já que não incendiaram.

Dinâmico. Alterações no ambiente são causadas por fatores adicionais além da ação 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 obsoletas rapidamente.

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

Estas características tornam o ambiente RoboCup Rescue desafiador, tanto no que tange as restrições impostas aos agentes a respeito do tempo de deliberação e quantidade de comunicação, quanto na extração de informações relavantes do ambiente.

Segundo (NAIR et al., 2002), o ambiente da RoboCup Rescue pode ser modelado como um E-GAP, onde há a necessidade de realizar as seguintes tarefas:

Combate a incêndios. O objetivo é minimizar a área total destruida 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 d'água de uma brigada de incêndio é limitado em 30 metros. Portanto, é necessário que a brigada de incêndio esteja situada exatamente sobre algum nó de rua que esteja localizado 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 ambulância. 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 time de ambulância. Para realizar esta tarefa, o time de ambulância deve estar situado 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 carregar o civil dentro da ambulância e movê-lo para um refúgio (caso requeira tratamento médico).

Dependendo das proporções de uma tarefa, 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 **E**, 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 resgate. Apresentamos a seguir as abordagens adotadas para decompor tarefas em subtarefas inter-relacionadas por **E** e para definir as competências dos agentes.

6.2.1 Tarefas de combate a incêndios

A decomposição de uma tarefa de combate a incêndio poderia utilizar propriedades do simulador RoboCup Rescue, como por exemplo a quantidade de água despejada por uma brigada de incêndio em cada instante de tempo e também a dinâmica envolvida na evolução do incêndio (temperatura, tipo de material da construção, etc). Contudo, esta abordagem frequentemente resultaria em uma quantidade de subtarefas impossível de ser tratada ao se considerar a questão de congestionamentos presente no RoboCup Rescue. Ou seja, de nada serve uma decomposição muito precisa, se não houver espaço suficiente ao redor da construção para abrigar a quantidade correspondente de brigadas de incêndio.

Em função disto, optou-se por adotar uma abordagem pragmática: a quantidade de subtarefas inter-relacionadas por **E** geradas pela decomposição de uma tarefa de combate

a incêndio corresponde ao espaço físico existente nas proximidades da construção. Este espaço é definido em termos da quantidade de nós de ruas próximos da construção e dentro do raio de alcance do jato d'água. Por exemplo, no caso apresentado na Figura 6.5, existem duas construções em chamas: C1 e C2. A construção C2 foi percebida pelo agente brigada de incêndio A1 (pois está em seu raio de visão) e será decomposta em duas subtarefas inter-relacionadas por **E**, pois os nós N1 e N2 estão dentro do raio de alcance do jato d'água. Já a construção C1 será decomposta em três subtarefas inter-relacionadas por **E**, a partir dos nós N2, N3 e N4.

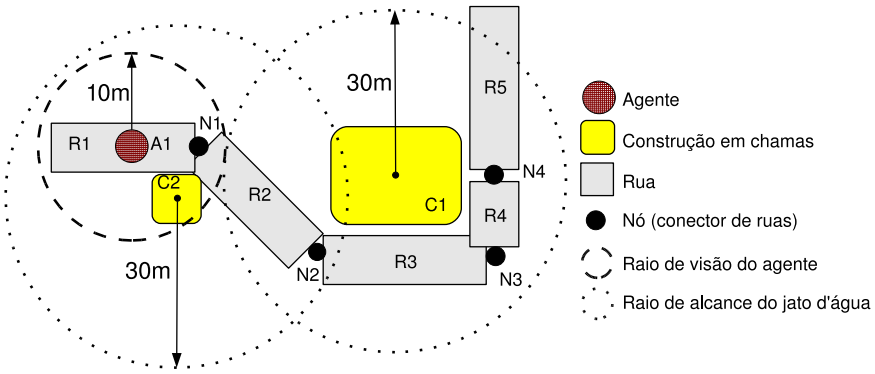

Figura 6.5: Esquema para decomposição de tarefas de combate a incêndio em subtarefas inter-relacionadas por **E**.

Cada agente brigada de incêndio $brig_i$ tem sua competência definida em função da distância Euclideana do agente até a tarefa de combate a incêndio. Seja $\mathcal{J}$ o conjunto de todas as construções em chamas percebidas pelo agente $brig_i$. Seja também $\mathcal{J}_{30} \subseteq \mathcal{J}$ o subconjunto de construções em chamas que estão ao alcance do jato d'água do agente $brig_i$, isto é, cujas distâncias sejam menores ou iguais a 30 metros. As competências do agente $brig_i$ para as construções em chamas $j \in \mathcal{J}$ são definidas de acordo com a Equação (6.2), onde $d(x, y)$ é a distância Euclideana entre x e y.

$$Cap(brig_i, j) = \begin{cases} 1 & \text{se } j \in \mathcal{J}_{30} \\ 1 - \frac{d(brig_i,j) - \min\limits_{k \in \mathcal{J}-\mathcal{J}_{30}} d(brig_i,k)}{\max\limits_{k \in \mathcal{J}-\mathcal{J}_{30}} d(brig_i,k) - \min\limits_{k \in \mathcal{J}-\mathcal{J}_{30}} d(brig_i,k)} & \text{caso contrário} \end{cases} \tag{6.2}$$

O primeiro caso atribui competência máxima para as tarefas situadas dentro do raio de alcance do jato d'água. Isto porque estas tarefas não exigem movimentação do agente, que pode realizá-las imediatamente. O segundo caso define as competências para as tarefas fora do raio de alcance. Para estas tarefas, a competência é inversamente proporcional a distância, sendo máxima para a tarefa menos distante e mínima para a tarefa mais distante. Isto porque tarefas distantes exigem maior deslocamento do agente. Consequentemente, mais instantes de tempo serão utilizados para movimentação.

6.2.2 Tarefas de remoção de bloqueios

Ao perceber uma rua bloqueada, o agente tem acesso a um atributo do bloqueio denominado custo de remoção. O valor deste atributo é definido pelo simulador em função da dimensão do bloqueio. Este valor corresponde à quantidade de instantes de tempo que um único agente força policial irá necessitar para remover completamente o bloqueio. Este valor também pode ser visto como a quantidade de forças policiais que devem atuar simultaneamente para remover por completo o bloqueio em um único instante de tempo.

A utilização do atributo custo de remoção para definir a quantidade de subtarefas inter-relacionadas por **E** (já que este atributo especifica a quantidade de agentes necessários para remover o bloqueio em um único instante de tempo), apesar de tentadora, não é eficiente pois não é possível que dois ou mais agentes ocupem exatamente a mesma posição. Deve-se portanto considerar também o espaço físico existente na rua, definido em termos de faixas. Por exemplo, no caso apresentado na Figura 6.6, a rua R1 é formada por quatro faixas, sendo duas para a direita e duas para a esquerda. Esta rua está bloqueada por B1 com custo de remoção 15 e que foi percebido pelo agente A1. Considerando apenas o custo de remoção seriam necessários 15 agentes atuando simultâneamente para remover o bloqueio em um instante de tempo. Entretanto, só é possível alocar no máximo 4 agentes simultâneamente, que é a quantidade de faixas disponíveis na rua. Neste sentido, a Equação (6.3) especifica a divisão de uma tarefa de remoção de bloqueio em subtarefas inter-relacionadas por **E**.

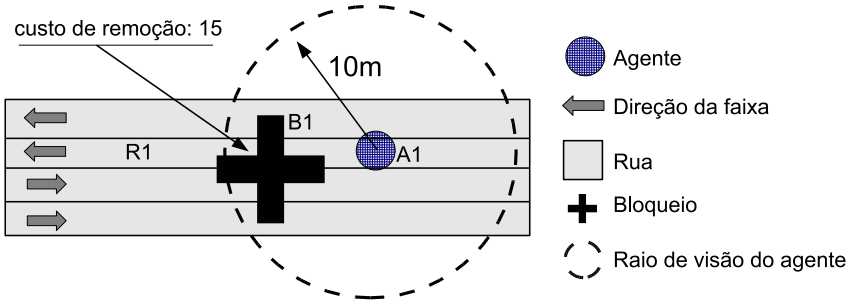

Figura 6.6: Esquema para decomposição de tarefas de remoção de bloqueios em subtarefas inter-relacionadas por **E**.

$$subtarefas\ \ bloqueio = \begin{cases} faixas & \text{se } custo\,remocão > faixas \\ custo\,remocão & \text{caso contrário} \end{cases} \tag{6.3}$$

As competências dos agentes força policial são definidas em função da distância Euclideana até as tarefas. Isto porque tarefas que estão próximas exigem menor movimentação do agente. Desta forma, instantes de tempo eventualmente gastos com deslocamento até o bloqueio são utilizados para removê-lo. Além disto, quanto mais distante está o bloqueio, maiores as chances de o trajeto até ele estar bloqueado, impedindo o acesso do agente. Seja $\mathcal{J}$ o conjunto de todos os bloqueios percebidos pelo agente força policial $fpol_i$. As competências do agente $fpol_i$ para os bloqueios $j \in \mathcal{J}$ são definidas de acordo com a Equação (6.4), onde $d(x, y)$ é a distância Euclideana entre x e y.

$$Cap(fpol_i, j) = \begin{cases} 1 & \text{se } d(fpol_i, j) = 0 \\ 1 - \frac{d(fpol_i,j) - \min\limits_{k\in\mathcal{J}} d(fpol_i,k)}{\max\limits_{k\in\mathcal{J}} d(fpol_i,k) - \min\limits_{k\in\mathcal{J}} d(fpol_i,k)} & \text{caso contrário} \end{cases} \tag{6.4}$$

O primeiro caso atribui competência máxima para as tarefas com distância zero, pois não exigem movimentação e o agente pode executá-las imediatamente. O segundo caso define competência inversamente proporcional a distância para as demais tarefas. Assim como ocorre com as brigadas de incêndio, tarefas distantes exigem maior deslocamento do agente, utilizando mais instantes de tempo para movimentação.

6.2.3 Tarefas de resgate de civis

Ao perceber um civil, o agente tem acesso a alguns atributos que definem o seu estado. Os valores destes atributos são atualizados pelo simulador no decorrer da simulação. Os atributos utilizados nestes experimentos são o soterramento, dano e saúde.

Soterramento. Indica o quanto o civil está soterrado na construção que colapsou. O valor deste atributo é a quantidade de instantes de tempo que um único agente time de ambulância irá necessitar para remover os escombros de sobre o civil. Este valor também pode ser interpretado como a quantidade de times de ambulância que devem atuar simultaneamente para remover os escombros em um único instante.

Saúde. Corresponde à vitalidade (nível de saúde) do agente, que morre quando o valor deste atributo torna-se zero.

Dano. Indica que o agente está ferido e requer tratamento médico. Um civil que requer tratamento médico deve ser transportado para um refúgio. Um agente ferido tem sua saúde decrementada pelo valor deste atributo a cada instante de tempo. Assim que o agente atinge um refúgio, o dano do agente passa a ser zero.

A quantidade de times de ambulâncias que estão removendo os escombros de um civil não sofre com restrições de espaço físico. Isto significa que um civil pode ter infinitos times de ambulância atuando simultâneamente para remover os escombros. Conforme já mencionado, a quantidade exata de times de ambulância requeridos para remover os escombros em um único instante de tempo é o valor do atributo soterramento. Logo, este atributo pode ser utilizado para definir com exatidão a quantidade de subtarefas inter-relacionadas por **E**.

Entretanto, apenas o uso do atributo soterramento para definir as subtarefas não é racional, pois pode ocasionar que poucos civis sejam resgatados por muitos agentes simultaneamente. Isto não é interessante, pois o objetivo é maximizar a quantidade de civis resgatados. Portanto, a divisão em subtarefas inter-relacionadas por **E** deve considerar a expectativa de vida do civil, que define quantos instantes de tempo ele irá sobreviver, dados a sua saúde e dano. A Equação (6.5) define a expectativa de vida de um agente qualquer i.

$$expectativa(i) = \frac{saúde(i)}{dano(i)} \tag{6.5}$$

Um agente com menor expectativa de vida requer maior quantidade de times de ambulâncias atuando simultâneamente para resgatá-lo a tempo. Por outro lado, se a expectativa de vida for maior do que a quantidade de instantes de tempo que um único time de ambulância necessita para resgatar o agente, é racional utilizar apenas um time de ambulância. Desta forma, os demais times de ambulâncias podem resgatar outros civis.

Por exemplo, na situação apresentada na Figura 6.7, o civil Civ1 possui soterramento 5, indicando que 5 times de ambulâncias são necessários simultâneamente para resgatá-lo em um único instante de tempo. Contudo, sua expectativa de vida é de 10 instantes de tempo. Logo, apenas um time de ambulâncias é suficiente para resgatá-lo, necessitando 5 instantes de tempo, duração que está dentro da expectativa de vida. Já o civil Civ2 possui expectativa de vida de 4 instantes de tempo. Neste caso, apenas um time de ambulância não é suficiente para resgatá-lo, pois necessitará 6 instantes de tempo. Com dois times de ambulâncias simultâneamente, o tempo de resgate será de 3 instantes, fazendo com que

o civil sobreviva. O civil Civ3 não está soterrado, sendo necessário apenas um time de ambulância para transportá-lo até um refúgio. A Equação (6.6) define a divisão de uma tarefa de resgate de civis em subtarefas inter-relacionadas por **E**.

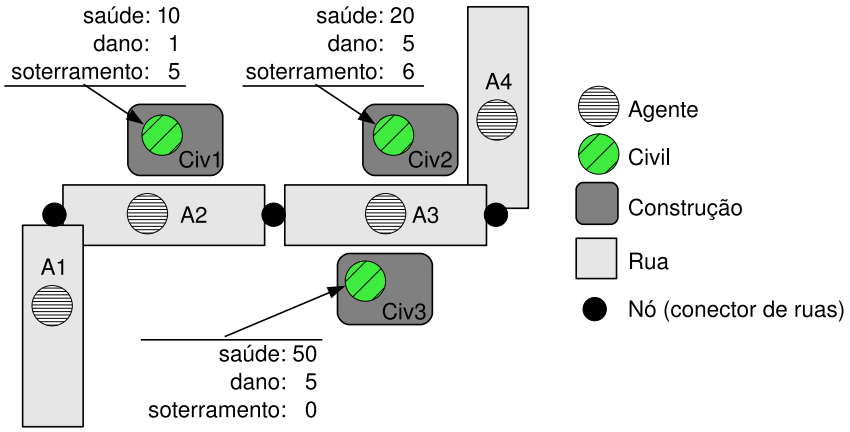

Figura 6.7: Esquema para decomposição de tarefas de resgate de civis em subtarefas inter-relacionadas por **E**.

$$subtarefas\ resgate = \begin{cases} 1 & \text{se não está soterrado ou} \\ & expectativa(civil) > \text{restante da simulação} \\ \left\lceil \frac{soterramento(civil)}{expectativa(civil)} \right\rceil & \text{caso contrário} \end{cases} \tag{6.6}$$

O primeiro caso define apenas uma tarefa (ou seja, não são criadas subtarefas) para um civil que não está soterrado. Nesta situação é necessário apenas transportar o civil ferido até o refúgio, atividade que requer apenas um agente time de ambulância. Além disto quando a expectativa de vida do civil é maior do que o tempo restante da simulação, apenas um time de ambulância é suficiente para resgatá-lo e transportá-lo até um refúgio. Isto porque este civil estará ferido, mas vivo, ao término da simulação. Desta forma os outros agentes poderão concentrar esforços em civis com menor expectativa de vida.

O segundo caso define uma quantidade de subtarefas inter-relacionadas por **E** que seja suficiente para manter o civil vivo enquanto durar o resgate. Para isto, determina-se quantos times de ambulâncias são necessários por instante de tempo da expectativa de vida, de forma que os escombros sejam totalmente removidos e o civil sobreviva. Tendo em vista que o atributo soterramento é interpretado como a quantidade de times de ambulâncias necessários para remover os escombros em um único instante, divide-se seu valor pela quantidade de instantes disponíveis (expectativa de vida) e obtém-se a quantidade de agentes necessária por instante de tempo. Esta é portanto a quantidade de subtarefas inter-relacionadas por **E** neste caso.

As competências dos agentes time de ambulâncias são definidas em função da distância Euclideana até as tarefas. O objetivo do uso da distância é o mesmo do caso dos agentes força policial: minimizar a quantidade de instantes de tempo necessários para movimentar o agente. Seja $\mathcal{J}$ o conjunto de todos os civis percebidos pelo agente time de ambulância $tamb_i$. As competências do agente $tamb_i$ para os civis $j \in \mathcal{J}$ são definidas

de acordo com a Equação (6.7), onde $d(x, y)$ é a distância Euclideana entre x e y.

$$Cap(tamb_i, j) = \begin{cases} 1 & \text{se } d(tamb_i, j) = 0 \\ 1 - \frac{d(tamb_i,j) - \min\limits_{k \in \mathcal{J}} d(tamb_i,k)}{\max\limits_{k \in \mathcal{J}} d(tamb_i,k) - \min\limits_{k \in \mathcal{J}} d(tamb_i,k)} & \text{caso contrário} \end{cases} \tag{6.7}$$

O primeiro caso atribui competência máxima para tarefas com distância zero. O segundo define competência inversamente proporcional à distância para as demais tarefas.

6.2.4 Experimentos

Os experimentos foram realizados em um dos mapas largamente utilizado na RRSL, denominado *Kobe_4*, que representa uma parte da cidade japonesa Kobe. Para caracterizar um *extreme team*, com larga escala na quantidade de agentes e tarefas, optou-se por aumentar significativamente estas quantidades em relação às utilizadas na competição. A Tabela 6.7 apresenta as quantidades de agentes e demais entidades utilizadas nos experimentos. Para demonstrar o aumento, também são apresentadas as quantidades (mínima e máxima) utilizadas na competição da RRSL de 2008 (RAMCHURN; VETSIKAS; ITO, 2008).

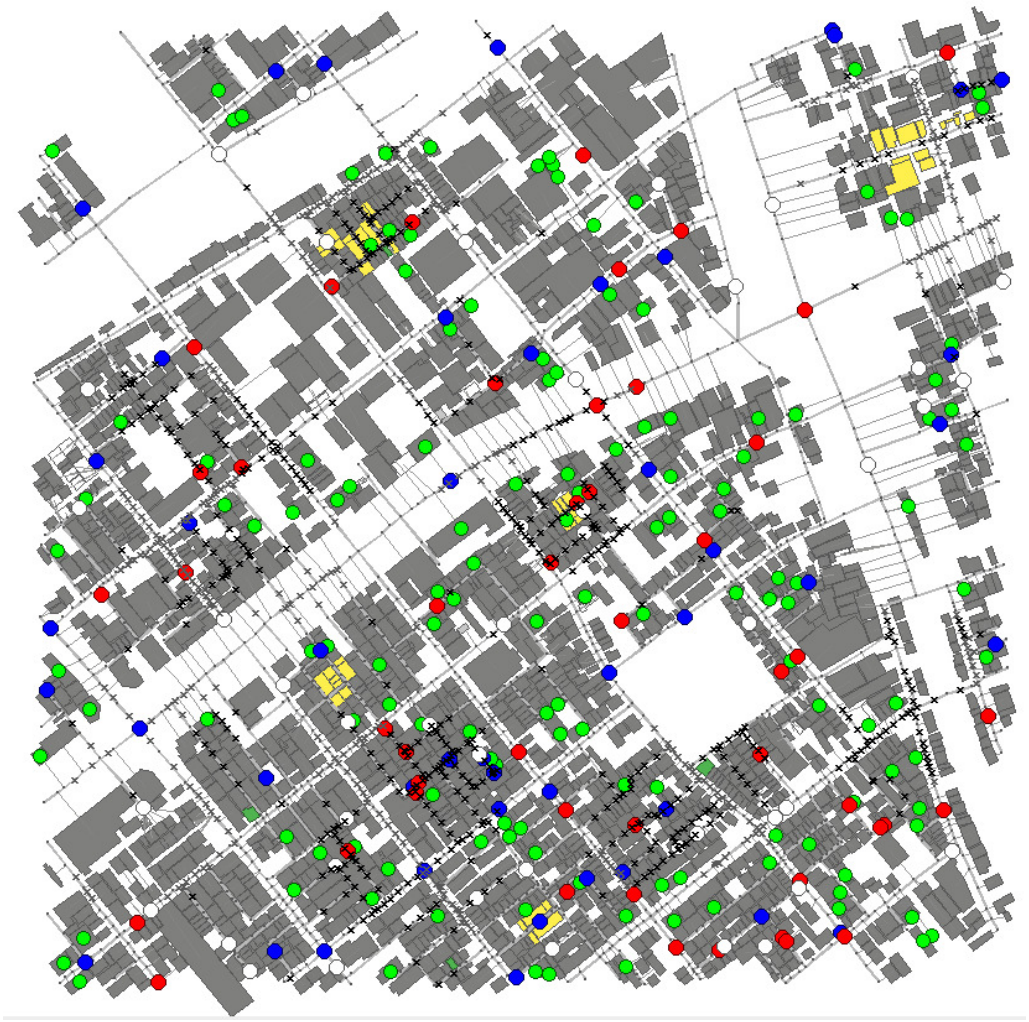

Figura 6.8: Mapa *Kobe_4* utilizado nos experimentos.

Entidade	*Experimentos*	*RRSL 2008 (min–max)*
Agentes Brigada de Incêndio	50	0–15
Agentes Força Policial	50	0–15
Agentes Time de Ambulância	50	0–8
Focos de incêndio iniciais	50	2–30
Civis	150	50–90

Tabela 6.7: Quantidade das entidades utilizadas nos experimentos e na RRSL 2008

O objetivo dos experimentos é confrontar os algoritmos eXtreme-Ants, Swarm-GAP e LA-DCOP. Cada experimento avalia a eficiência destes algoritmos em relação a três fatores: desempenho dos agentes na realização das tarefas, esforço computacional e utilização de comunicação. Além disto, também é avaliado o impacto de utilizar ou não tarefas inter-relacionadas por **E** (através da decomposição de tarefas em subtarefas, da maneira descrita anteriormente). Em todos os experimentos, foram avaliados *thresholds* (LA-DCOP) e estímulos (eXtreme-Ants e Swarm-GAP) no intervalo $[0.1, 0.9]$. Cada simulação considera 300 instantes de tempo (padrão utilizado na RRSL). Cada ponto nos gráficos que seguem representa uma média de 20 execuções, excluindo-se o maior e menor valor. A análise dos resultados foi feita com base em testes t com 99% de confiança.

A percepção de cada agente não se restringe apenas às tarefas que pode realizar (por exemplo, uma brigada de incêndio percebe civis feridos e ruas bloqueadas). Para as tarefas que o agente está apto a realizar, sua competência é definida como descrito anteriormente. Já para tarefas que não está apto, a competência do agente é zero. Além das competências, critérios adicionais relacionados com as simulações foram utilizados. Estes critérios são mencionados a seguir, como forma de garantir que os experimentos sejam reprodutíveis.

Os agentes são pró-ativos, exploram o ambiente para perceber as tarefas. Para realizar esta exploração, adotou-se uma estratégia de setorização do mapa. Cada agente possui um setor de exploração. Quando o agente não estiver realizando tarefas, ou não houver percebido tarefas, ele passa a explorar seu setor (ruas e construções). Se todos os objetos de um setor forem explorados, o agente escolhe aleatoriamente outro setor.

As tarefas percebidas são armazenadas na memória do agente. Tarefas não realizadas são mantidas na memória do agente ao longo da simulação, até que sejam realizadas ou deixem de existir. Desta forma, ao terminar de realizar uma tarefa, o agente pode imediatamente iniciar a realização de outra previamente armazenada em sua memória. Isto evita exploração adicional e, consequentemente, melhora o desempenho dos agentes.

Ao dedicir realizar uma tarefa, o agente se compromete a realizá-la por completo. Isto evita que os agentes fiquem trocando de tarefas à medida que se deslocam pelo ambiente. Esta troca é prejudicial pois os instantes de tempo utilizados para deslocamento poderiam estar sendo investidos na realização de tarefas. Tarefas que deixam de existir são abandonadas (por exemplo, uma construção que já foi totalmente consumida por incêndios, ou um civil morto). Para evitar que o agente fique preso em uma rua bloqueada no trajeto até a tarefa, ele possui um *timeout* de inatividade. Se permanecer inativo por uma quantidade de instantes de tempo superior ao *timeout*, a tarefa é abandonada.

Os agentes se auto-monitoram, verificando sua saúde em cada instante. Isto deve ser realizado pois os agentes podem se ferir ao realizar as tarefas. Por exemplo, um time de ambulâncias pode sofrer queimaduras caso esteja resgatando um civil em uma construção que incendiar. Se o agente verificar que seus ferimentos põem em risco sua vida, ele desloca-se até um refúgio e lá permanece. Isto é para evitar um impacto negativo causado

pela morte do agente no escore, que considera a quantidade total de agentes vivos.

A implementação do eXtreme-Ants no RoboCup Rescue utiliza recrutamento com comunicação e movimentação (Seção 5.2), devido ao fato de que os agentes podem se movimentar pelo ambiente para atender às requisições de recrutamento.

6.2.4.1 Desempenho relativo ao escore

Para avaliar o desempenho dos agentes foi utilizado o escore definido pelo simulador RoboCup Rescue, Equação (6.1). Como já mencionado, este escore considera a área total construída preservada e a quantidade de sobreviventes. A Figura 6.9 apresenta os escores obtidos por cada algoritmo,

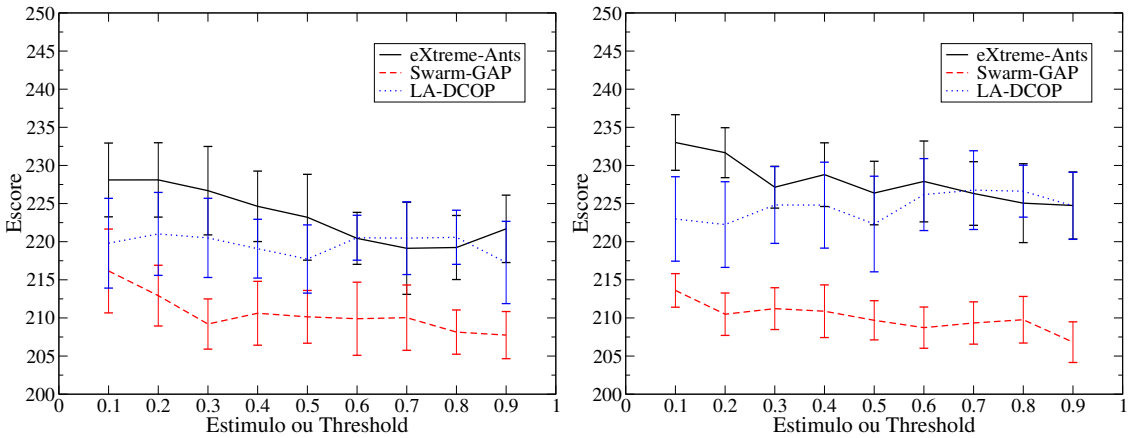

Figura 6.9: Escore ao final de 300 instantes de tempo da simulação no RoboCup Rescue. (a) Sem tarefas inter-relacionadas por **E**. (b) Com tarefas inter-relacionadas por **E**.

Com relação aos valores de estímulo e *threshold* avaliados, os resultados mostram que os escores são similares nas duas situações, com e sem tarefas inter-relacionadas por **E**, havendo um leve aumento no escore do eXtreme-Ants e LA-DCOP na presença de tarefas inter-relacionadas. O escore do eXtreme-Ants com tarefas inter-relacionadas é estatísticamente superior ao eXtreme-Ants sem tarefas inter-relacionadas para os estímulos 0.1, 0.4, 0.6, 0.7 e 0.8. Já o LA-DCOP com tarefas inter-relacionadas é superior ao LA-DCOP sem tarefas inter-relacionadas para os *thresholds* 0.4, 0.6, 0.7, 0.8. 0,9. Não há diferença significativa nos escores do Swarm-GAP com/sem tarefas inter-relacionadas.

A Tabela 6.8 apresenta o melhor e pior escore de cada algoritmo, com o respectivo estímulo ou *threshold*. O melhor de cada algoritmo, independente da presença de tarefas inter-relacionadas, é destacado em negrito. O melhor escore global nos casos com e sem inter-relacionamentos, independente de algoritmo, está sombreado. Como pode ser visto, os melhores escores do LA-DCOP não superam estatísticamente os piores escores do eXtreme-Ants.

O escore do LA-DCOP é influenciado pela forma como os agentes decidem quais tarefas realizar. Esta decisão é determinística e ótima, visto que cada agente sempre realiza as tarefas em que é mais competente. Desta forma, tarefas com competências subótimas não são realizadas. Já no eXtreme-Ants, a decisão probabilística acaba por causar um melhor "espalhamento" dos agentes no ambiente, pois o algoritmo permite que realizem tarefas cujas competências sejam subótimas. Este espalhamento demonstra-se

Tabela 6.8: Escores para melhor e pior valor de parâmetro (estímulo no eXtreme-Ants e Swarm-GAP, e *threshold* no LA-DCOP), sem e com tarefas inter-relacionadas.

Algoritmo	Sem tarefas inter-relacionadas				Com tarefas inter-relacionadas			
	Melhor		Pior		Melhor		Pior	
	Escore	Parâm.	Escore	Parâm.	Escore	Parâm.	Escore	Parâm.
eXtreme-Ants	228.1±4.8	0.1	219.1±6.0	0.7	**233.0±3.6**	**0.1**	224.7±4.3	0.9
Swarm-GAP	**216.1±5.4**	**0.1**	207.7±3.0	0.9	213.6±2.1	0.1	206.8±2.6	0.9
LA-DCOP	221.0±5.4	0.2	217.2±5.3	0.9	**226.7±5.1**	**0.7**	222.2±5.6	0.2

mais eficiente no RoboCup Rescue pois evita a propagação de incêndios, ocasionando menor destruição das construções e diminuindo a gravidade dos ferimentos em civis.

Os escores do eXtreme-Ants e do LA-DCOP são superiores ao Swarm-GAP em ambos os casos, sem e com tarefas inter-relacionadas. Os seguintes fatores influenciam negativamente o escore do Swarm-GAP:

Tendência de realização. Como originalmente consta no Swarm-GAP (FERREIRA JR.; BOFFO; BAZZAN, 2008), o limiar de resposta do agente é 1 (valor máximo) no caso em que não possui competência para realizar a tarefa. Entretanto, utilizar este valor na tendência, Equação (3.1), estabelece uma chance de o agente realizar uma tarefa para a qual não possui competência. Por exemplo, com os estímulos 0.5 e 0.9, as tendências de realizar uma tarefa com limiar de resposta 1 são 0.2 e 0.45, respectivamente (ver Figura 3.1). Realizar uma tarefa sem possuir competência é prejudicial já que, por definição, esta é a tarefa mais distante em relação à posição do agente. Isto fará com que sejam consumidos vários instantes de tempo apenas para se deslocar até a tarefa. Por exemplo, no caso de uma brigada de incêndio ou time de ambulância, enquanto o agente se desloca, o incêndio pode se propagar e a saúde do civil diminuir, respectivamente. Isto dificulta a realização das tarefas. No eXtreme-Ants isto não acontece, pois o limiar interno é definido como ∞ nos casos em que o agente não possui competência para evitar que estas tarefas sejam alocadas, Equação (5.1).

Normalização da tendência. O algoritmo Swarm-GAP não prevê a normalização das tendências, Equação (3.1), quando o agente percebe várias tarefas simultaneamente. Isto evita que o agente tenha uma "visão geral" de todas as tarefas disponíveis para realização, pois a tendência de realizar uma tarefa acaba não se relacionando com as demais tendências. Desta forma existe uma maior chance de o agente realizar uma tarefa para a qual sua competência seja reduzida.

Tratamento de tarefas inter-relacionadas. Conforme já discutido nos experimentos da Seção 6.1, há uma certa deficiência por parte do Swarm-GAP para lidar com tarefas inter-relacionadas por **E**. Esta deficiência influencia o escore do Swarm-GAP no caso de decomposição em subtarefas, já que estas subtarefas são inter-relacionadas por **E**.

6.2.4.2 Comunicação

Em todos os três algoritmos, quando um agente decide por não realizar uma tarefa, ela é repassada para outro agente através do canal de comunicação. Este experimento avalia a utilização do canal de comunicação pelos agentes.

O simulador RoboCup Rescue impõe limites no uso do canal de comunicação em cada instante de tempo, com relação a quantidade e tamanho das mensagens. Em função disto, este experimento adota uma abordagem diferente da adotada no simulador independente (Seção 6.1). Em vez de gerar uma mensagem para cada tarefa não realizada, estas tarefas são codificadas e enviadas em uma única mensagem, dentro do limite de tamanho estabelecido (máximo 256 *bytes*). Desta forma, este experimento avalia a quantidade total de *bytes* enviados ao longo da simulação, e não a quantidade total de mensagens.

Outra limitação imposta pelo simulador RoboCup Rescue é o fato de que um agente não pode endereçar uma mensagem para outro agente. Entretanto, isto é necessário nos algoritmos avaliados, pois quando uma tarefa não é alocada, ela é repassada para um único agente aleatoriamente selecionado. Para contornar esta limitação, são utilizados os agentes de central, estabelecendo-se um canal de comunicação entre a central e cada agente. O papel da central é de única e exclusivamente rotear as mensagens, recebendo de um agente e selecionando aleatoriamente outro agente para enviá-la. Não há desta forma nenhuma inteligência na central que poderia influenciar os algoritmos. A Figura 6.10 apresenta a utilização do canal de comunicação.

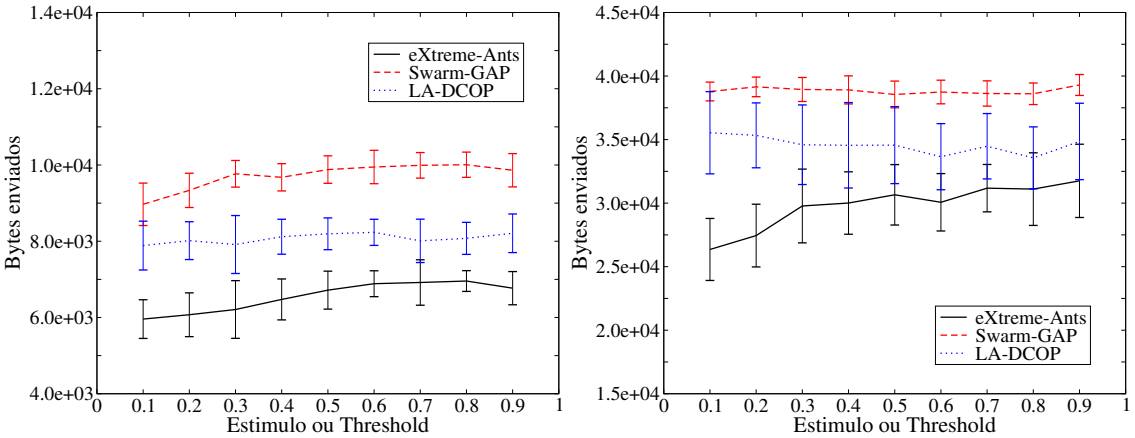

Figura 6.10: Uso do canal de comunicação, medido como o total de *bytes* enviados por todos os agentes no RoboCup Rescue. (a) Sem tarefas inter-relacionadas por **E**. (b) Com tarefas inter-relacionadas por **E**.

Nos dois casos, sem e com tarefas inter-relacionadas, o total de *bytes* enviados pelo eXtreme-Ants é inferior ao LA-DCOP, que por sua vez é inferior ao Swarm-GAP. Tendo em vista que cada agente repassa as tarefas que não consegue alocar, o uso do canal de comunicação no caso com tarefas inter-relacionadas é maior justamente porque são criadas muito mais tarefas.

É interessante notar a relação entre o uso do canal de comunicação e o escore dos algoritmos (Figura 6.9). Quanto maior o escore, menor é o uso do canal de comunicação. Isto é explicado pela eficiência dos algoritmos. A eficiência do algoritmo reflete-se no uso do canal de comunicação pois influencia a dinâmica do ambiente. Por exemplo, ao extinguir eficientemente um incêndio, evita-se que se propague, o que aumentaria a quantidade de tarefas. Um algoritmo eficiente na alocação reduz o surgimento de tarefas no longo prazo, com consequente redução na comunicação necessária.

6.2.4.3 *Esforço computacional*

Em razão da dinâmica do ambiente RoboCup Rescue, o tempo que cada agente dispõe para decidir qual tarefa realizar é limitado em meio segundo. Portanto, as decisões devem ser ágeis, ou então a alocação torna-se facilmente obsoleta.

Para avaliar a agilidade das decisões adotou-se o esforço computacional, que corresponde à quantidade de operações realizadas pelo agente para tomar decisões a respeito de quais tarefas realizar, em cada instante de tempo. Um baixo esforço computacional aumenta a agilidade dos agentes para atuarem em ambientes com restrições no tempo disponível para tomar decisões. No caso do simulador independente de domínio (Seção 6.1), por ser assíncrono, os agentes tomavam a decisão várias vezes em um mesmo instante de tempo, isto é, toda vez que uma tarefa (ou conjunto) fosse percebida ou recebida pelo canal de comunicação. Já o simulador RoboCup Rescue é síncrono, ou seja, o agente pode tomar a decisão a respeito de qual tarefa realizar apenas uma vez em cada instante de tempo. Porém, não são em todos os instantes de tempo que um agente precisa tomar esta decisão. Conforme mencionado anteriormente, convencionou-se que ao dedicir realizar uma tarefa, o agente se compromete a realizá-la por completo. Sendo assim, durante os instantes de tempo em que um agente estiver realizando alguma tarefa, ele não precisa tomar decisão se realiza ou não outras tarefas percebidas/recebidas. Nos instantes de tempo em que o agente não precisa tomar decisão são consideradas zero operações para efeito de esforço computacional. Quando o agente precisa tomar decisões, é considerada uma quantidade de operações correspondente a complexidade do algoritmo.

A Figura 6.11 apresenta o esforço computacional de cada agente em cada instante de tempo, para cada algoritmo. As curvas apresentam a mesma tendência nas situações sem e com tarefas inter-relacionadas por **E**, havendo diferença na escala do esforço computacional (eixo y), sendo significativamente maior no caso com tarefas inter-relacionadas. Isto porque no caso com tarefas inter-relacionadas são criadas e disponibilizadas muito mais tarefas aos agentes em função da decomposição de tarefas em subtarefas.

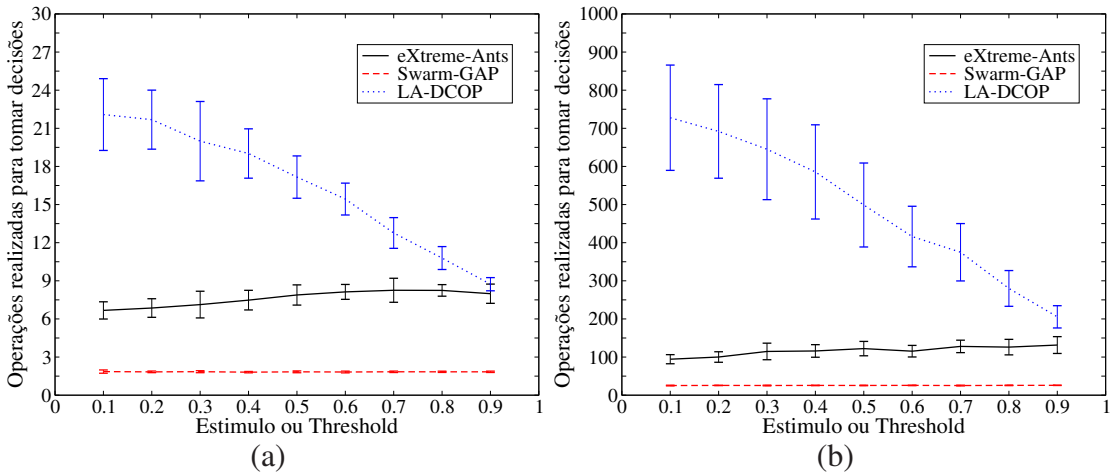

Figura 6.11: Esforço computational, medido pela quantidade de operações realizadas por agente para tomar decisões em cada instante de tempo no simulador RoboCup Rescue. (a) Sem tarefas inter-relacionadas por **E**. (b) Com tarefas inter-relacionadas por **E**.

Nos dois casos, o Swarm-GAP apresenta esforços computacionais inferiores ao eXtreme-Ants. Isto é explicado por dois fatores. O primeiro é a baixa complexidade do

Swarm-GAP, que é da ordem de $O(n)$. O segundo fator está relacionado com baixo escore do Swarm-GAP. Conforme mencionado na Seção 6.2.4.1, o escore do Swarm-GAP é baixo porque os agentes desperdiçam muitos instantes de tempo se deslocando quando decidem realizar tarefas distantes, ao invés de realizar tarefas próximas. Durante o deslocamento, o agente não toma decisões a respeito de realizar outras tarefas, pois ele já está comprometido com a tarefa previamente escolhida. Sendo assim, a quantidade de tomadas de decisão de um agente no Swarm-GAP é menor (ocorrendo apenas quando não está se deslocando), consequentemente, diminuindo as operações realizadas para tomar decisões. É importante notar que há um compromisso no Swarm-GAP entre o esforço computacional e o escore. Apesar de possuir menores esforços computacionais que o eXtreme-Ants e o LA-DCOP, em nenhum caso o escore do Swarm-GAP é superior a estes dois algoritmos.

Em relação ao LA-DCOP, nos dois casos, o esforço computacional é superior ao eXtreme-Ants, sendo fortemente influenciado pela complexidade do algoritmo. O esforço computacional dos agentes diminui a medida que o *threshold* aumenta. Isto deve-se ao fato de que no LA-DCOP cada agente utiliza o *threshold* para decidir quais tarefas é capaz de realizar (aquelas em que a competência é superior ao *threshold*). As tarefas de fato realizadas são escolhidas dentre aquelas que é capaz de realizar. Logo, quanto menor o *threshold*, maior é a quantidade de tarefas que os agentes são capazes de realizar e, consequentemente, maior é o esforço computacional requerido para avaliar quais, de fato, serão realizadas.

6.2.4.4 Comparação com time vice-campeão da liga de simulação em 2008

A escolha do RoboCup Rescue para realização dos experimentos desta dissertação deve-se ao fato de ele ser uma plataforma de teste que tenta representar problemas reais de busca e resgate, além de já ser conhecido pela comunidade de SMA. O objetivo do eXtreme-Ants não é, neste momento, ser competitivo a ponto de participar das competições da RRSL, pois só aborda a alocação de tarefas. Apesar disto, confrontou-se o escore do eXtreme-Ants (e dos outros dois algoritmos) com o escore do time vice-campeão da RRSL 2008[1] (HARA; TORIUMU, 2008). Este confronto tem caráter meramente ilustrativo. A escolha do time vice-campeão foi necessária devido ao fato de que a implementação do time campeão apresenta restrições que a impede de lidar com as quantidades de agentes utilizadas nestes experimentos.

A Figura 6.12 apresenta o melhor escore de cada algoritmo (em negrito na Tabela 6.8) e o escore do time vice-campeão. As diferenças são estatisticamente significantes (teste t, 99% de confiança). O escore do vice-campeão é 11.1% superior ao eXtreme-Ants, 19.8% superior ao Swarm-GAP e 14.2% superior ao LA-DCOP.

O escore superior do time vice-campeão era esperado. Isto porque os times participantes da RRSL exploram ao máximo as características peculiares do RoboCup Rescue para construir heurísticas e métodos *ad hoc* com o objetivo de vencer. A grande desvantagem deste tipo de abordagem é a dificuldade de generalizar métodos que sejam aplicáveis em outras circunstâncias e ambientes. Por outro lado, o eXtreme-Ants foi concebido com o objetivo de ser um método geral para alocar tarefas em *extreme teams* formalizados através de E-GAP, independentemente de ambiente.

[1] As implementações dos times participantes da RRSL 2008 encontram-se em `https://roborescue.svn.sourceforge.net/svnroot/roborescue/robocup2008data/SourceCodes/finals.zip`

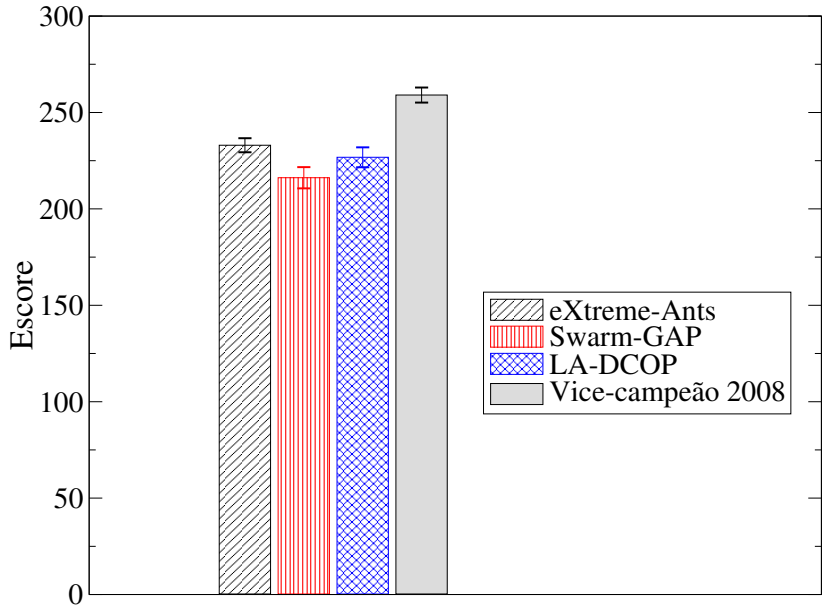

Figura 6.12: Escore de cada algoritmo em relação ao vice-campeão da RRSL 2008.

6.2.5 Resumo

Este capítulo apresentou a avaliação da eficiência do eXtreme-Ants através de uma série de experimentos em dois ambientes distintos: um simulador independente de domínio e o simulador RoboCup Rescue. Nestes ambientes, o eXtreme-Ants foi comparado com outros dois algoritmos: Swarm-GAP (FERREIRA JR.; BOFFO; BAZZAN, 2008) e LA-DCOP (SCERRI et al., 2005). As métricas consideradas foram o desempenho em relação às alocações realizadas, a quantidade de comunicação utilizada e o esforço computacional requerido.

No simulador independente de domínio, o desempenho do eXtreme-Ants é próximo do LA-DCOP, mas requer menor quantidade de comunicação e esforço computacional. Com relação ao Swarm-GAP o desempenho do eXtreme-Ants é superior, particularmente na presença de tarefas inter-relacionadas, que requerem esforço simultâneo de um grupo de agentes.

O simulador RoboCup Rescue requer que as competências dos agentes sejam definidas em função de seus papeis e de características do ambiente. Nos experimentos desta dissertação, as competências foram definidas em função do tipo de agente e da distância Euclideana das tarefas. O uso da distância Euclideana tem por objetivo fazer com que os agentes invistam maior tempo na realização de tarefas do que em deslocamentos.

Para mobilizar simultaneamente uma maior quantidade de agentes em determinadas tarefas, adotou-se o uso de tarefas inter-relacionadas por **E**, que são realizadas simultaneamente pelos agentes. As tarefas inter-relacionadas por **E** são criadas a partir da decomposição de uma tarefa em subtarefas. A abordagem adotada para decomposição é pragmática no sentido de gerar subtarefas que sejam viáveis de serem realizadas considerando questões de congestionamento. A decomposição em subtarefas inter-relacionadas adotada considera os seguintes fatores:

Tarefas de combate a incêndio. O espaço físico capaz de receber brigadas de incêndio nas proximidades das construções.

Tarefas de remoção de bloqueios. O tempo necessário para remover o bloqueio e a quantidade de faixas da rua.

Tarefas de resgate de civis. A expectativa de vida do civil, que considera seu nível de saúde, dano e soterramento.

Os resultados obtidos no RoboCup Rescue mostraram que a eficiência do eXtreme-Ants é superior aos outros algoritmos com relação ao desempenho e utilização do canal de comunicação. O uso de tarefas inter-relacionadas proporcionou um leve aumento no desempenho, apesar de requerer maiores esforços computacionais e comunicação. Comparando-se os resultados obtidos nos dois ambientes, o desempenho do eXtreme-Ants foi superior ao LA-DCOP no RoboCup Rescue, mas não no simulador independente. Isto deve-se ao modo diferenciado de como ocorre a dinâmica nestes ambientes.

No simulador independente, a dinânica é simulada através de alterações nas características das tarefas, sendo que a quantidade total de tarefas é mantida fixa. Isto significa que a quantidade de tarefas não realizadas em um instante de tempo não influencia a quantidade de tarefas que existirão no futuro. Sendo assim, é viável a abordagem utilizada pelo LA-DCOP, que realiza apenas as tarefas que maximizam o desempenho em detrimento àquelas cuja participação no desempenho não é acentuada.

Por outro lado, no RoboCup Rescue as características e a quantidade das tarefas são variáveis ao longo da simulação, sendo influenciadas pelo comportamento dos agentes (realização das tarefas). Sendo assim, as tarefas não realizadas em um instante de tempo podem contribuir para o surgimento de novas tarefas, como ocorre por exemplo com um incêndio não controlado que se propaga nas construções próximas. O desempenho do eXtreme-Ants neste caso é superior pois sua decisão probabilística causa melhor "espalhamento" dos agentes no ambiente ao permitir que realizem tarefas cujas competências sejam subótimas. Este espalhamento demonstra-se mais eficiente no RoboCup Rescue pois diminui a quantidade de tarefas que surgem ao longo da simulação.

De forma geral, os experimentos mostraram que a alocação probabilística do eXtreme-Ants, baseada no modelo de divisão de trabalho dos insetos sociais, reduz a quantidade de comunicação e o esforço computacional. Entretanto, é importante enfatizar que a escolha de um algoritmo em particular está relacionada com as características do cenário. Quando não há restrições na comunicação ou no tempo que os agentes dispõem para decidir, a quantidade de tarefas é fixa, e as tarefas não realizadas em um instante de tempo não contribuem para o surgimento de novas tarefas, o LA-DCOP pode ser uma boa escolha. Por outro lado, em cenários com restrições de comunicação ou tempo, e onde a quantidade de tarefas é variável e influenciada pelas tarefas não realizadas, o eXtreme-Ants é mais apropriado.

7 CONCLUSÕES E TRABALHOS FUTUROS

Em muitos ambientes reais, a escala do problema envolve tanto uma grande quantidade de agentes, quanto uma grande quantidade de tarefas. O termo *extreme teams* foi introduzido na literatura para designar as seguintes características da alocação de tarefas nestes ambientes: são dinâmicos; agentes podem realizar múltiplas tarefas; agentes podem possuir funcionalidades sobrepostas; e podem existir inter-relacionamentos entre tarefas. Abordagens existentes na literatura tratam, efetivamente, apenas as três primeiras características de *extreme teams*.

Esta dissertação apresentou um algoritmo, geral e independente de ambiente, para alocação de tarefas em *extreme teams* formalizados através do modelo E-GAP. O algoritmo, chamado eXtreme-Ants, é inspirado no sucesso ecológico dos insetos sociais, e utiliza as metáforas de divisão de trabalho e recrutamento para transporte cooperativo. A metáfora de divisão de trabalho proporciona decisões rápidas e eficientes, atendendo as três primeiras características de *extreme teams*. O recrutamento permite formar grupos de agentes comprometidos com a realização simultânea de tarefas que exigem esforço conjunto, atendendo a quarta característica: inter-relacionamentos entre tarefas. Com isto, esta dissertação contribui para o estado-da-arte no estudo e desenvolvimento de SMAs ao concretizar, de fato, o conceito completo de *extreme teams*.

A eficiência do eXtreme-Ants foi avaliada através de uma série de experimentos em dois ambientes distintos: um simulador independente de domínio e o simulador RoboCup Rescue. Nestes ambientes, o eXtreme-Ants foi comparado com outros dois algoritmos para alocação de tarefas em *extreme teams*: Swarm-GAP e LA-DCOP. Os resultados obtidos demonstram que a eficiência do eXtreme-Ants é balanceada com relação ao desempenho, a quantidade de comunicação e ao esforço computacional. Isto sugere que técnicas inspiradas em inteligência de enxames podem ser consideradas com sucesso para alocação de tarefas em *extreme teams*.

Considera-se que investigar como a eficiência do eXtreme-Ants possa ser melhorada, mantendo sua base no modelo de divisão de trabalho e no processo de recrutamento, constitui a principal sugestão de sequência deste trabalho. A seguir são apresentadas possíveis direções para aprimorar o eXtreme-Ants.

Conforme definido no modelo de divisão de trabalho, o valor de estímulo reflete a necessidade e urgência da realização da respectiva tarefa. Desta forma, sugere-se estudar como dinâmicas podem ser introduzidas aos valores de estímulo das tarefas, para refletir, por exemplo, o tempo transcorrido sem que as tarefas tenham sido realizadas. Além disto, o valor de estímulo inicial de uma tarefa pode melhor refletir a importância desta tarefa ser realizada. Isto seria particularmente útil no domínio da RoboCup Rescue. Por exemplo, o combate rápido e eficiente de um incêndio evita sua propagação.

Estimar quais tarefas podem surgir no futuro, sejam elas novas ou derivadas de tare-

fas não realizadas, pode auxiliar os agentes na tomada de decisão. Neste sentido, outra possibilidade de trabalho futuro é incorporar no agente a utilização de um modelo que descreve, ao menos aproximadamente, a dinâmica do ambiente. Com isso, os agentes podem detectar quais são as tarefas mais importantes ou críticas, que se não forem realizadas brevemente, prejudicarão o desempenho do sistema. A partir desta informação, o agente poderia modificar, ou o seu limiar de resposta, ou o estímulo associado à tarefa, para permitir que estas tarefas sejam realizadas prioritariamente. É importante enfatizar que o que se propõe é incorporar o uso de um modelo de dinâmica do ambiente para auxiliar a tomada de decisão (e.g., através de modificações no limiar ou estímulos). O modelo em si, é peculiar a cada ambiente. Por exemplo, no ambiente RoboCup Rescue, um agente que possui um modelo capaz de estimar a propagação de um incêndio (baseado no material de construção e em princípios termodinâmicos), decidirá extinguir o incêndio de uma construção com maior potencial de propagação, diminuindo a quantidade de tarefas que surgiriam no futuro.

A decisão de realizar ou não cada tarefa no eXtreme-Ants é tomada pelo agente com base na tendência de realização. Esta tendência é baseada no limiar de resposta do agente (que considera suas competências) e no estímulo da tarefa. Outra possibilidade de trabalho futuro consiste em incorporar os recursos disponíveis no agente e requeridos pela tarefa no limiar de resposta do agente. Desta forma, um melhor compromisso entre as competências do agente e a utilização de seus recursos seria obtida, proporcionando consequentemente melhoria na eficiência de realização das tarefas.

A eliminação, por completo, da comunicação direta entre os agentes também pode ser considerada. Para substituir a comunicação direta, propõe-se utilizar estigmergia, ou seja, comunicação indireta, através do ambiente. Isto pode ser atingido incorporando-se no eXtreme-Ants a capacidade de depositar e detectar feromônios no ambiente. O uso de feromônios seria particularmente interessante no recrutamento, pois evitaria a necessidade de adaptar as etapas do processo para utilizarem comunicação direta. Contudo, a contrapartida de utilizar comunicação por estigmergia é a necessidade de que os agentes possam se movimentar pelo ambiente para liberar os feromônios.

Os experimentos realizados nesta dissertação adotaram um modelo de rede completa, onde todos os agentes estão interconectados. Contudo, é possível que a rede existente no SMA adote outros modelos, por exemplo, redes de mundos pequenos, redes aleatórias e redes sem escala. Neste sentido, outro possível trabalho futuro compreende a avaliação do desempenho do eXtreme-Ants nestes outros modelos de rede. Acredita-se que o desempenho do eXtreme-Ants não será drasticamente afetado. Isto porque a comunicação entre os agentes no eXtreme-Ants não é baseada em difusão de mensagens, e sim em uma cadeia de mensagens que são repassadas de um agente a outro.

REFERÊNCIAS

AGASSOUNON, W.; MARTINOLI, A. Efficiency and robustness of threshold-based distributed allocation algorithms in multi-agent systems. In: AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS (AAMAS 2002), 2002, New York, NY, USA. **Proceedings...** ACM, 2002. p.1090–1097.

BONABEAU, E.; THERAULAZ, G.; DORIGO, M. **Swarm Intelligence**: from natural to artificial systems. New York, USA: Oxford University Press, 1999. 307p.

CICIRELLO, V.; SMITH, S. Wasp-like Agents for Distributed Factory Coordination. **Journal of Autonomous Agents and Multi-Agent Systems**, Amsterdam, v.8, n.3, p.237–266, May 2004.

CRAMTON, P.; SHOHAM, Y.; STEINBERG, R. Introduction to Combinatorial Auctions. In: CRAMTON, P.; SHOHAM, Y.; STEINBERG, R. (Ed.). **Combinatorial Auctions**. Cambridge: The MIT Press, 2006. p.1–13.

DECKER, K. S.; LESSER, V. R. Quantitative Modeling of Complex Computational Task Environments. In: INTERNATIONAL WORKSHOP ON DISTRIBUTED ARTIFICIAL INTELLIGENCE, 12., 1993, Hidden Valley, Pennsylvania. **Proceedings...** [S.l.: s.n.], 1993. p.67–82.

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.

FERREIRA JR., P. R.; BOFFO, F.; BAZZAN, A. L. C. Using Swarm-GAP for Distributed Task Allocation in Complex Scenarios. In: JAMALI, N.; SCERRI, P.; SUGAWARA, T. (Ed.). **Massively Multiagent Systems**. Berlin: Springer, 2008. n.5043, p.107–121. (Lecture Notes in Artificial Intelligence).

FERREIRA JR., P. R.; SANTOS, F. dos; BAZZAN, A. L. C.; EPSTEIN, D.; WASKOW, S. J. Robocup Rescue as Multiagent Task Allocation among Teams: experiments with task interdependencies. **Journal of Autonomous Agents and Multiagent Systems**, [S.l.], 2009. to appear.

FRANKS, N. R. Teams in social insects: group retrieval of prey by army ants (*Eciton burchelli*, hymenoptera: formicidae). **Behavioral Ecology and Sociobiology**, [S.l.], v.18, n.6, p.425–429, 1986.

GERKEY, B. P.; MATARIĆ, M. J. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. **The International Journal of Robotics Research**, [S.l.], v.23, n.9, p.939–954, 2004.

GERSHMAN, A.; MEISELS, A.; ZIVAN, R. Asynchronous Forward-Bounding for Distributed Constraint Optimization. In: EUROPEAN CONFERENCE ON ARTIFICIAL INTELLIGENCE (ECAI 2006), 17., 2006, Riva del Garda, Italy. **Proceedings...** IOS Press: Inc., 2006. p.103–108.

HAM, M.; AGHA, G. Market-based coordination strategies for large-scale multi-agent systems. **System and Information Sciences Notes**, [S.l.], v.2, n.1, p.126–131, 2007.

HAM, M.; AGHA, G. A Study of Coordinated Dynamic Market-Based Task Assignment in Massively Multi-Agent Systems. In: JAMALI, N.; SCERRI, P.; SUGAWARA, T. (Ed.). **Massively Multiagent Systems**. Berlin: Springer, 2008. n.5043, p.43–63. (Lecture Notes in Artificial Intelligence).

HARA, T.; TORIUMU, F. **Robocup Rescue 2008 Repository - SUNTORI team**. 2008. Disponível em <https://roborescue.svn.sourceforge.net/svnroot/roborescue/robocup2008 data/SourceCodes/finals.zip>. Acesso em: 13 maio 2009.

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

HÖLLDOBLER, B.; WILSON, E. O. The multiple recruitment systems of the african weaver ant *Oecophylla longinoda* (Latreille) (Hymenoptera: formicidae). **Behavioral Ecology and Sociobiology**, [S.l.], v.3, n.1, p.19–60, 1972.

HUNSBERGER, L.; GROSZ, B. J. A combinatorial auction for collaborative planning. In: FOURTH INTERNATIONAL CONFERENCE ON MULTIAGENT SYSTEMS (ICMAS), 2000, Boston. **Proceedings...** [S.l.: s.n.], 2000. p.151–158.

JENNINGS, N.; SYCARA, K.; WOOLDRIGE, M. A Roadmap of Agent Research and Development. **Autonmous Agents and Multi-Agent Systems**, [S.l.], n.1, p.7–38, 1998.

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.

KRIEGER, M. J. B.; BILLETER, J.-B.; KELLER, L. Ant-like task allocation and recruitment in cooperative robots. **Nature**, [S.l.], v.406, n.6799, p.992–995, 2000.

KUBE, R.; BONABEAU, E. Cooperative Transport by Ants and Robots. **Robotics and Autonomous Systems**, [S.l.], v.Volume 30, n.Issue 1/2, p.85–101, 2000. ISSN: 0921-8890.

LEHMANN, D.; MÜLLER, R.; SANDHOLM, T. The Winner Determination Problem. In: CRAMTON, P.; SHOHAM, Y.; STEINBERG, R. (Ed.). **Combinatorial Auctions**. Cambridge: The MIT Press, 2006. p.217–317.

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.

MARTELLO, S.; TOTH, P. **Knapsack Problems**: algorithms and computer implementations. New York, NY, USA: John Wiley & Sons, 1990. 308p.

MODI, P. J.; SHEN, W.-M.; TAMBE, M.; YOKOO, M. 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.

NAIR, R.; ITO, T.; TAMBE, M.; MARSELLA, S. Task Allocation in the Rescue Simulation Domain: a short note. In: BIRK, A.; CORADESCHI, S. (Ed.). **RoboCup 2001**: robot soccer world cup V. Berlin: Springer-Verlag, 2002. p.751–754. (Lecture Notes in Computer Science, v.2377).

NÜSLE, T.; KLEINER, A.; BRENNER, M. **Approaching Urban Disaster Reality**: the ResQ Firesimulator. Freiburg: Institut für Informatik, Universität Freiburg, 2004.

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.

RAHWAN, T.; JENNINGS, N. R. An Algorithm for Distributing Coalitional Value Calculations among Cooperating Agents. **Artificial Intelligence Journal**, [S.l.], v.171, n.8-9, p.535–567, 2007.

RAMCHURN, G.; VETSIKAS, Y.; ITO, N. **Rescue Simulation League - Agent Competition**: rules and setup. 2008. Disponível em <http://www.robocuprescue.org/wiki/images/Rules2008-draft.pdf>. Acesso em: 13 maio 2009.

ROBSON, S. K.; TRANIELLO, J. F. A. Resource Assessment, Recruitment Behavior, and Organization of Cooperative Prey Retrieval in the Ant *Formica schaufussi* (Hymenoptera: formicidae). **Journal of Insect Behavior**, [S.l.], v.11, n.1, p.1–22, 1998.

SANTOS, F. dos.; BAZZAN, A. L. C. Ant-Based Task Allocation Among Teams. In: INT. JOINT CONF. ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS, 8., 2009, Budapest, Hungary. **Proceedings...** IFAAMAS, 2009. p.1183–1184.

SANTOS, F. dos; BAZZAN, A. L. C. Ant Based Multiagent Algorithm for Task Allocation in Large Scale and Dynamic Scenarios. In: ANNUAL CONFERENCE ON GENETIC AND EVOLUTIONARY COMPUTATION (GECCO-09), 11., 2009, Montréal, Canada. **Proceedings...** ACM Press, 2009. To Appear.

SANTOS, F. dos; BAZZAN, A. L. C. Alocação Eficiente de Tarefas em Sistemas Multiagente Dinâmicos e de Larga Escala. In: VII ENCONTRO NACIONAL DE INTELIGÊNCIA ARTIFICIAL (ENIA), 2009. **Anais...** SBC, 2009. To appear.

SCERRI, P.; FARINELLI, A.; OKAMOTO, S.; TAMBE, M. 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.

SHEHORY, O.; KRAUS, S. Task allocation via coalition formation among autonomous agents. In: FOURTEENTH INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 1995, Montréal, Canada. **Proceedings...** Morgan Kaufmann, 1995. p.655–661.

SILAGHI, M. C.; YOKOO, M. Nogood Based Asynchronous Distributed Optimization (ADOPT ng). In: FIFTH INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS, 2006, New York, NY, USA. **Proceedings...** ACM Press, 2006. p.1389–1396.

SILAGHI, M. C.; YOKOO, M. Revisiting ADOPT-ing and its Feedback Schemes. In: IEEE INTL. CONF. ON INTELLIGENT AGENT TECHNOLOGIES (IAT), 2007, Silicon Valley. **Proceedings...** [S.l.: s.n.], 2007. p.3–10.

SKINNER, C.; BARLEY, M. Robocup Rescue Simulation Competition: status report. In: BREDENFELD, A.; JACOFF, A.; NODA, I.; TAKAHASHI, Y. (Ed.). **RoboCup 2005**: robot soccer world cup IX. Berlin: Springer-Verlag, 2006. p.632–639. (Lecture Notes in Computer Science, v.4020).

SMITH, R. G. The Contract Net Protocol: high-level communication and control in a distributed problem solver. **IEEE Transactions on Computers**, Washington, DC, USA, v.C-29, n.12, p.1104–1113, Dec. 1980.

SULTANIK, E.; MODI, P. J.; REGLI, W. C. On Modeling Multiagent Task Scheduling as a Distributed Constraint Optimization Problem. In: TWENTIETH INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, (IJCAI 2007), 2007. **Proceedings...** [S.l.: s.n.], 2007. p.1531–1536.

SYCARA, K. P. Multiagents Systems. **AI Magazine**, California, v.19, n.2, p.79–92, 1998. Disponível em<http://www.aaai.org/AITopics/assets/PDF/AIMag19-02-2-article.pdf>.Acesso em: 13 maio 2009.

THERAULAZ, G.; BONABEAU, E.; DENEUBOURG, J. Response Threshold Reinforcement and Division of Labour in Insect Societies. In: ROYAL SOCIETY OF LONDON SERIES B - BIOLOGICAL SCIENCES, 1998. **Anais...** [S.l.: s.n.], 1998. v.265, p.327–332.

WOOLDRIDGE, M. J. **An Introduction to MultiAgent Systems**. New York: John Wiley & Sons, 2002. 348p.

WOOLDRIGE, M.; JENNINGS, N. Intelligent Agents: theory and practice. **Knowledge Engineering Review**, [S.l.], v.10, n.2, p.115–152, 1995.

YEOH, W.; FELNER, A.; KOENIG, S. BnB-ADOPT: an asynchronous branch-and-bound DCOP algorithm. In: NINTH INTERNATIONAL WORKSHOP ON DISTRIBUTED CONSTRAINT REASONING (DCR2007) - 13TH INTERNATIONAL CONFERENCE ON PRINCIPLES AND PRACTICE OF CONSTRAINT PROGRAMMING (CP2007), 2007. **Proceedings...** [S.l.: s.n.], 2007.