UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA

Daniel Epstein

Heurística para formação de estruturas de coalizão no simulador Robocup Rescue

Trabalho de conclusão de curso.

Ciência da Computação

Orientador: Prof. Dr. Ana Lúcia Cetertich Bazzan

Porto Alegre, julho de 2010

Sumário

LISTA DE ABREVIATURAS E SIGLAS

A	Conjunto de agentes
S	Subconjunto de agentes de A
$\|A\|, n$	Número de agentes
a_i	i-ésimo agente em A
CS	Estrutura de Coalizão
CS^*	Estrutura de Coalizão Ótima
S^A	Conjunto de todas Estruturas de Coalizão
C_i	i-ésima coalizão da estrutura de coalizão CS
k	Número máximo de agentes em qualquer coalizão
$v(C)$	Função característica da coalizão C
$V(CS)$	Valor da estrutura de coalizão CS
Dt	Distância de um agente até sua tarefa
Ap	Área total de um prédio
Fp	Intensidade do fogo em um prédio
Mp	Material do qual o prédio é construído
Vp	Valor do prédio
Pt	Total de pistas em uma rua
Pl	Total de pistas livres em uma rua
Cb	Custo para desbloquear uma rua
Vr	Valor da rua
Hp	Quantidade de vida de um humanóide (Hit Points)
Ev	Expectativa de vida de um civil
St	Intensidade de soterramento de um civil
Vc	Valor do civil
Va	Valor do agente em uma coalizão

LISTA DE FIGURAS

RESUMO

Um dos objetivos principais de um sistema multiagentes é atingir objetivos que estão além da capacidade de um único agentes. Estes objetivos podem ser tarefas que um único agente não tem condições de realizar sozinho, necessitando assim que outros agentes trabalhem junto com ele. Nesses casos, o mecanismo de coordenação de um sistema multiagentes é parte fundamental para o bom funcionamento do sistema, pois permite que os agentes possam agir de maneira coesa em direção aos seus objetivos, sejam eles individuais ou coletivos.

A formação de coalizões (grupos de agentes que concordam em coordenar suas ações por um objetivo comum) é uma questão fundamental nesses sistemas, pois serve de mecanismo de coordenação entre os agentes. Essa tarefa de formação de coalizões entre agentes é um tópico de pesquisa muito ativo que recebe atenção de pesquisadores de diversas áreas. Um dos grandes problemas associados à formação de coalizão de agentes e que representa um importante gargalo no processo, é estabelecer uma partição dos agentes em coalizões denominada de estrutura de coalizão. O que significa, em outras palavras, determinar quais coalizões devem ser formadas de maneira a maximizar a utilidade do sistema.

Neste trabalho, é apresentada uma maneira eficiente de criar estruturas de coalizão no simulador Robocup Rescue. Através de uma série de restrições e um conjunto de regras, demonstra-se como realizar uma busca eficiente por uma CS durante a execução do Simulador. Cada agente do simulador conta com um conjunto de regras e restrições que podem ser trabalhados de forma individual em cada coalizão. Esse conjunto de regras e restrições formam uma heurística para a busca de uma CS que seja satisfatória no contexto do simulador. Além disso, são apresentados experimentos realizados no simulador e uma medição da eficiência desta heurística. Comparando a eficiência e o resultado final desse processo com um algoritmo que não utiliza nenhum tipo de restrição no processo de coalizão, é demonstrado que o aumento na eficiência e diminuição do tempo de processamento não afetam de maneira significativa o resultado final.

ABSTRACT

One of the main goals of a multiagent system is to achieve goals that are beyond the capacity of a single agent. These goals can be tasks that a single agent is unable to accomplish alone, thus requiring that other agents work with him. In such cases, the coordination of a multiagent system is a key to the smooth functioning of the system because it allows agents to act in a manner united toward their goals, whether individual or collective.

The formation of coalitions (groups of agents that agree to coordinate their actions toward a common goal) is a key issue in such systems, once it serves as a coordinating mechanism among agents. This task of coalition formation among agents is a very active research topic receiving attention from researchers in different areas. A major problem associated with the formation of coalition and that represents a major bottleneck in the process is to establish a partition of agents into coalitions called the coalition structure. Which means, in other words, determine which coalitions must be formed so as to maximize the usefulness of the system.

In this paper, we present an efficient way to create structures coalition in RoboCup Rescue simulator. Through a series of restrictions and a set of rules, we show how to perform an efficient search for a *CS* during the execution of the Simulator. Each agent simulator has a set of rules and restrictions that can be worked individually in each coalition. This set of rules and restrictions form a heuristic for finding a *CS* that is satisfactory in the context of the simulator. Furthermore, we present experiments conducted in the simulator and a measurement of the efficiency of this heuristic. Comparing the efficiency and outcome of this process with an algorithm that does not use any type of restriction in the process of coalition, is shown to increase efficiency and decrease processing time does not significantly affect the final result.

1. INTRODUÇÃO

Um agente pode ser definido como um sistema computacional que, com certo nível de autonomia e percepção, possui capacidade de ação no ambiente em que está situado. Quando nos referimos à autonomia, queremos explicitar que o agente é capaz de operar sem intervenção humana ou de outros sistemas. Assim, um agente autônomo possui capacidade de tomar suas decisões sem que seja necessário que uma pessoa ou outro agente diga a ela o que fazer.

Em diversas situações é possível ter não apenas um agente, mas sim diversos agentes em um mesmo ambiente. Nesses casos, temos uma sociedade de agentes ou um sistema multiagentes. Nesses sistemas multiagentes (denominados de SMA no restante do texto), o foco pode deixar de ser em um agente agindo de forma isolada e passar a ser no agente agindo dentro de uma sociedade. Lidar com as questões de relacionamento entre os agentes e de coordenação dos agentes passa a ser um dos focos centrais de um SMA.

Em ambientes aberto, complexos e dinâmicos, onde podem estar presentes questões como restrição de tempo e informações e elementos podem entrar ou sair do sistema, a coordenação entre agentes torna-se um problema ainda mais desafiador e difícil de ser implementado. Porém, ao cooperar e coordenar esforços, os agentes podem muitas vezes atingir um objetivo que seria impossível de ser alcançado através de ações individuais ou mesmo podem perceber benefícios como redução do trabalho necessário do ponto de vista individual de cada agente, redução dos custos e aumento dos benefícios ao realizar um trabalho mais eficiente do que aquele que seria feito pelo agente sozinho.

Um SMA deve ser projetado de maneira que os agentes estejam organizados tal que os papéis, relações e estruturas de autoridade que determinam a dinâmica comportamental dos agentes estejam claramente definidas (HORLING, B.; LESSER, V., 2005). Os diferentes paradigmas organizacionais incluem coalizões, times, hierarquias, federações, congregações, etc.

Neste trabalho, iremos nos focar no paradigma de coalizões para organizar os agentes. Uma coalizão pode ser definida como um grupo de agentes que cooperam para atingir um objetivo comum. Os agentes dentro de uma coalizão irão coordenar o trabalho entre si e não haverá compromisso com agentes externos à coalizão. Assim, a coalizão poderia ser vista como um grande agente único se fosse percebida por um observador externo.

Quando consideramos as aplicações no mundo real, temos um reconhecimento cada vez maior desse tipo de comportamento estratégico tanto no campo teórico como no campo pratico. O uso de métodos de formação de coalizões entre agentes autônomos tem sido usado em uma grande variedade de domínios de aplicação, sendo exemplo deles o comércio eletrônico, aquisição e filtragem distribuída de dados, sistemas de recomendação, sistemas flexíveis de manufatura, células de manufatura, sistemas multi-robôs, grids computacionais, redes de sensores distribuídos, roteamento distribuído de veículos, sistemas de transmissão de energia, organizações virtuais, busca cooperativa entre agentes, entre tantos outros. Devido ao grande número de aplicações, o problema da formação de coalizões entre agentes possui duas abordagens principais distintas dentro da literatura científica: a abordagem da teoria dos jogos e a abordagem de sistemas multiagentes.

De maneira geral, a formação de coalizões em sistemas multiagentes pode ser vista como um processo composto por três atividades entrelaçadas, sendo elas:

- A geração da estrutura da coalizão, que estuda como dividir os agentes em conjuntos ou coalizões de maneira a maximizar a utilidade do sistema multiagentes;

- O cálculo do valor de cada coalizão;

- A divisão do ganho das coalizões entre seus membros.

Historicamente temos a divisão de ganhos entre os membros da coalizão como uma área de pesquisa da teoria dos jogos, enquanto as duas primeiras tem tido uma maior abordagem dentro da área de ciência da computação e SMA. A parte correspondente à geração de estruturas de coalizões é particularmente complicada e tem sido largamente estudada nos últimos anos, já que esta parte representa o maior gargalo no processo como um todo.

Este trabalho de conclusão tem por objetivo propor um conjunto de regras e restrições para a formação de uma heurística a ser aplicada à um algoritmo para ser implementado na plataforma de simulação Robocup Rescue (abreviado por RCR no restante do texto), utilizando estruturas de coalizão para agrupar os agentes do sistema. Através desta heurística, pretende-se garantir um espaço de busca de estruturas de coalizão que seja suficientemente pequeno para ser viável e eficiente sem perder a qualidade das coalizões, a fim de poder ser utilizado em um algoritmo real para o simulador. Através de experimentos no simulador

pretende-se avaliar a qualidade das coalizões geradas e a eficiência desta heurística em relação à um algoritmo que não utilize nenhum tipo de restrição quanto à formação de coalizões, bem como a qualidade final do algoritmo proposto.

1.1 Organização dos capítulos

O texto deste trabalho de conclusão está organizado da seguinte forma:

Capítulo 2: Estabelece o contexto necessário para a compreensão dos conceitos tratados neste trabalho ao apresentar a abordagem tradicional para o estudo de coalizões dentro da teoria dos jogos através das definições usuais, introduzir os sistemas multiagentes, juntamente com o problema associado da coordenação multiagentes, e definir o problema da formação de coalizões em sistemas multiagentes.

Capítulo 3: Trata do problema de geração da estrutura de coalizão e apresenta as diversas abordagens existentes na literatura referentes à esse problema.

Capítulo 4: Apresenta o simulador Robocup Rescue, para o qual será proposta um estratégia de coalizão de agentes.

Capítulo 5: Apresenta a proposta de heurística e o método utilizado para a geração de estruturas de coalizão.

Capítulo 6: Apresenta os experimentos que serão realizados e os parâmetros utilizados nesses experimentos.

Capítulo 7: Apresenta o resultado dos experimentos, bem como uma análise sobre esses resultados e uma comparação entre diferentes tipos de algoritmos para o simulador.

Capítulo 8: Apresenta as conclusões e perspectivas de trabalhos futuros.

2. CONCEITOS RELACIONADOS

Este capítulo contém os conceitos preliminares importantes para a compreensão do restante do texto. Muitas das idéias acerca de coalizões de agentes e noções básicas sobre coalizões em sistemas multiagentes serão descritas nessa seção a fim de permitir uma completa compreensão do restante do texto.

Este capítulo está dividido em quatro sessões. A primeira descreve sobre teoria dos jogos e foca em jogos cooperativos; a segunda descreve os sistemas multiagentes; a terceira introduz conceitos básicos sobre coordenação de agentes; e a quarta descreve conceitos básicos sobre modelos de coalizão.

2.1 Teoria dos jogos

A teoria dos jogos diz respeito à matemática da interação entre entidades racionais e proporciona ferramentas para podermos analisar essa interação do ponto de vista matemático. As entidades envolvidas nessas interações são denominadas jogadores ou agentes e é assumido que esses agentes são capazes de escolher dentro de um conjunto de possibilidades aquela que garanta a ele o maior ganho, sendo este parâmetro definido por uma função de utilidade. Para cada conjunto de ações ou movimentos, é associado um vetor de utilidade que define o ganho correspondente a cada agente naquele momento.

Podemos dividir o estudo dos jogos em três principais modelos: jogos em forma estratégica (ou normal), jogos em forma extensiva e jogos em forma de coalizão. A diferença entre as três formas está na quantidade de informação que há no modelo em relação à interação entre os agentes.

Na forma estratégica de um jogo, cada agente escolhe uma estratégia própria do agente baseado em um conjunto de estratégias possíveis e adiciona a estratégia em seu conjunto de estratégias, ou espaço de ação. A escolha de um agente é baseada não somente nas suas possibilidades de estratégia, mas também nas possibilidades que os outros agentes possuem. Este modelo assume que todas as escolhas são realizadas de maneira simultaneamente e apenas quando todos tiverem escolhidos serão reveladas e então será associado um ganho para cada agente. O cenário de interação descrito acima pode ser formalizado com o uso de uma notação matriz de ganho ou matriz de utilidade.

Um jogo em forma extensa assume que as decisões de ação são tomadas de maneira alternada entre os agentes. Isto adiciona as noções de posição e movimento. A representação usual desde modelo de jogo é através de uma árvore denominada de árvore de jogo.

Por fim, a forma de coalizão é abordada dentro da área de pesquisa de jogos cooperativos da teoria dos jogos. Os jogos cooperativos (também chamados de jogos de coalizão) tiverem inicio nos trabalhos de John Nash, em 1950 (referencia!!!). Neste trabalhamos, estamos apenas interessados na terceira forma de jogos, os jogos de coalizão. A seguir serão explicados os jogos em função característica, que são o modelo básico dentro dos jogos cooperativos.

2.1.1 Jogos cooperativos na forma de função característica

Um dos modelos mais simples para descrever o processo de formação de coalizões entre agentes em um jogo cooperativo é o formalismo dos jogos na *forma de função característica*. Neste modelo cada coalizão *v* possui um valor *v(C)* que depende exclusivamente das ações de seus membros. Por não haver influencia externa no valor da coalizão, podemos analisar apenas o resultado da coalizão em questão, sem se preocupar com as demais. Existem outros modelos que consideram que o valor da coalizão está sujeito a fatores externos a coalizão (chamado de "*externalidades*."), mas esses modelos não podem ser explorados com o uso de funções características e também não serão explorados nesse trabalho.

Abaixo seguem algumas definições básicas usuais dos jogos em função característica. Um grupo de agentes em um jogo é denominado de *coalizão*, e a coalizão formada por todo o conjunto de agentes é usualmente chamado de *grande coalizão*. Formalmente tem-se as seguintes definições.

Definição 1. Seja $A = \{a1, a2, ..., an\}$ o conjunto dos agentes no jogo. Uma *coalizão* é um subconjunto de A, $S \subseteq A$ e o conjunto de todas as coalizões é denotado por 2^A.

Definição 2. A coalizão correspondente ao conjunto de todos os agentes A é denominada de *grande coalizão*.

Definição 3. Um *jogo de coalizão* em forma característica é um par *(A, v)*, onde

• A é o conjunto de agentes; e

• $v : 2^A \mapsto \Re$ é uma função que mapeia cada grupo de agentes S $\subseteq$ N para um valor real de utilidade. Esta função é denominada *função característica* do jogo.

Em algumas situações é possível assumir que a união de duas coalizões permite que esta tenha um valor no mínimo igual à soma dos valores destas duas coalizões iniciais. Neste caso, o jogo é denominado de *super-aditivo*.

Definição 4. Um jogo em forma característica é *super-aditivo* se $v(S \cup T) \geq v(S) + v(T)$, para quaisquer duas coalizões $S, T \subseteq A$ disjuntas ($S \cap T = \phi$).

Também existe a classe dos jogos *sub-aditivos*. Neste caso, o valor da coalizão resultante da união de duas coalizões é igual ou inferior à soma dos valores das coalizões individuais.

Definição 5. Um jogo em forma característica é *sub-aditivo* se $v(S \cup T) \leq v(S) + v(T)$, para quaisquer duas coalizões $S, T \subseteq A$ disjuntas ($S \cap T = \phi$).

A solução de um jogo de coalizão é definida como um vetor de utilidade, ou de pagamento, que define a quantidade que deve ser alocada para cada agente da coalizão. Este vetor é também comumente denominado de configuração.

Definição 6. O vetor de solução de um jogo, ou configuração, é $\vec{x} = (x_1, x_2, \ldots, x_{|A|}) \in \Re^{|A|}$. Ainda, a utilidade do agente ***i*** na solução $\vec{x}$ é denotada por $\vec{x}_i$.

Muitas vezes estamos interessados apenas em soluções "válidas", ou seja, soluções que não distribuem um $\vec{x}$ maior do que a configuração é capaz de produzir (por exemplo, se a configuração tiver um valor *Y*, a soma de $\vec{x}$ não pode ser maior que *Y*).

Definição 7. Uma configuração $\vec{x}$ é válida se existe um conjunto de coalizões $T = \{ S_1, \ldots, S_k \}$ onde $\bigcup_{S \in T} S = A$ tal que $\bigcup_{S \in T} v(S) \geq \sum_{i \in A} \vec{x}_i$.

Uma configuração $\vec{x}$ é válida se podemos encontrar um conjunto de coalizões cujos valores são tanto quanto em $\vec{x}$, de maneira que se pode pagar os agentes de acordo com $\vec{x}$ com v. Este conjunto de coalizões é muitas vezes referido como estrutura de coalizão e usualmente representado como CS.

Para avaliar qual coalizão é melhor que a outra, podemos utilizar diversos métodos que nos permitem ordenar as coalizões em uma ordem de preferência. Muitas vezes o método utilizado é o bem-estar social, calculado pela soma das utilidades recebidas pelos agentes. Essa soma nos fornece um valor global. Utilizamos esse critério de avaliação quando estamos interessados em avaliar o sistema como um todo e não o valor obtido por agentes específicos.

2.2 Sistemas multiagentes

Quando diversos agentes estão situados em um determinado ambiente e estes agentes interagirem entre si, temos um sistema multiagentes, onde muitas vezes podemos nos focar não nos agentes individuais, mas sim no sistema como um todo e na interação entre os agentes no contexto dessa sociedade. Nesses SMA, devemos também considerar que um agente deve ser uma entidade capaz de se comunicar com os outros agentes e que está ciente da existência deles

Existem diversas características desejáveis em um agente que esteja envolvido em um ambiente multiagentes e que são relativas à capacidade de um agente de agir de maneira flexível. Destas, podemos destacar algumas que são relevantes ao trabalho em questão:

- Racionalidade: Um agente pode ser considerado racional se ele procura sempre maximizar seu desempenho, levando em consideração suas percepções do sistema e quaisquer outras informações relevantes que ele possa ter. Ele deve ser capaz de analisar sua base de conhecimento para melhorar seu desempenho.

- Reatividade: é a capacidade do agente de perceber o ambiente onde está e de responder de forma eficiente às mudanças que possam ocorrer nesse ambiente.

- Pró-atividade: Se refere à capacidade do agente de tomar decisões racionais para atingir seus objetivos.

- Habilidade social: Os agentes devem possuir a capacidade de comunicação de interação com outros agentes, seja através de mecanismos diretos como trocas de mensagens ou por mecanismos indiretos com um espaço de memória compartilhada.

Para classificarmos um SMA, temos que pensar em termos dos agentes que o compõem. Se todos os agentes tiverem a mesma arquitetura, podemos considerar esse sistema como sendo homogêneo e que todos agentes tem as mesmas capacidades de realizar tarefas. No caso de haverem agentes de diferentes arquiteturas, o sistema é considerado heterogêneo.

2.2.3 Coordenação em sistemas multiagentes

A coordenação pode ser definida como o ato de trabalhar em conjunto, de forma harmoniosa e coerente, a fim de atingir um objetivo comum. Quando tratamos com SMA, a coordenação é peça chave para que os agentes possam atingir tanto objetivos individuais quanto globais do sistema.

(JENNINGS, N. R., 1996) apresenta três razões principais para explicar a necessidade de coordenar ações em um sistema multiagentes. São elas:

• Existem dependências entre as ações dos agentes. Em algumas situações uma tarefa pode depender de outra. Essa interdependência de ações ocorrer quando a tarefa de um agente pode ser influenciada positivamente ou negativamente pela tarefa a ser realizada por outro agente.

• Os agentes operam sobre restrições globais. Em um ambiente onde há recursos limitados, os agentes devem se coordenar para cumprirem suas tarefas levando em conta a necessidade de recursos uns dos outros. Agentes individuais tentarão utilizar todo o recurso disponível, mas em um SMA isso pode não ser desejado ou pode trazer resultados negativos para a sociedade.

• Os agentes individualmente não possuem informação, recursos ou capacidade para resolver o problema. Os agentes podem não ser capazes de realizar suas tarefas sozinhos. Em

um SMA, cada agente pode possuir um conjunto de informações diferentes e essas informações serem complementares ou mesmo pode haver uma determinada tarefa que um agente não tenha competência para resolver sozinho.

Nem todos SMA são desenvolvidos pensando no bem comum da sociedade. É possível encontrar na literatura diversas ocasiões em que os agentes competem entre si pensando apenas no seu bem estar ao invés do bem estar global (são chamados de agentes auto-motivados). Mesmo nesses sistemas onde o agente pensa apenas em si pode haver a necessidade de coordenação entre eles, pois a tarefa que o agente deseja realizar pode ser impossível para ele fazer sozinho.

2.3 Modelos de coalizão

Em (Horling-Lesser, 2005) afirma-se que coalizões diferem de outros modelos de organização por serem apenas necessários enquanto há um objetivo em comum para ser realizado. Coalizões são formadas tendo como base fundamental um objetivo comum entre os participantes da coalizão em questão e são dissolvidas quando o objetivo já não mais existe ou é impossível de ser realizado. Além disso, a estrutura da coalizão é plana, ou seja, não há uma hierarquia definida dentro da coalizão. Não existem lideres ou agentes que possam decidir as ações dos demais quando há um impasse.

A prática estabelecida na literatura descrita por (Sandholm; Lesser, 1997), (Dang; Jennings 2004) e (Rahwan; Jennings 2007) considera a formação de coalizões em jogos em função característica (JFC)[1], onde o valor de uma coalizão é dado por uma função característica. Neste modelo, assume-se que o valor de uma coalizão não depende das ações de agentes não membros. No presente trabalho, também consideraremos que a formação de coalizões é um JFC.

Sandholm et al. (1999) descreve o processo de formação de coalizão como uma seqüência de três atividades que podem, em alguns casos, serem independentes uma da outra. A formação de coalizão então procede da seguinte maneira:

[1] Tradução do inglês: “characteristic function games (CFG)”.

• Geração da estrutura de coalizão. Esta etapa consiste em particionar exaustivamente um dado conjunto de agentes do sistema em coalizões disjuntas. A partição resultante é denominada estrutura de coalizão. Por exemplo, em um sistema com três agentes {a1, a2, a3}, existem sete possíveis coalizões assim enumeradas: {a1}, {a2}, {a3}, {a1, a2}, {a1, a3}, {a2, a3} e {a1, a2, a3}. Estas coalizões podem ser combinadas em cinco possíveis estruturas de coalizão: {{a1, a2, a3}}, {{a1}, {a2, a3}}, {{a2}, {a1, a3}}, {{a3}, {a1, a2}} e {{a1}, {a2}, {a3}}.

• Otimização do valor da coalizão. Nesta etapa os agentes em cada coalizão devem coordenar seus recursos e ações para resolver o problema conjunto, otimizando o valor a ser recebido. Por exemplo, dada a estrutura de coalizão {{a2}, {a1, a3}}, cada uma das duas coalizões {a2} e {a1, a3} tentará trabalhar de maneira a calcular seu valor, ou ganho.

• Divisão da utilidade entre os integrantes da coalizão. O valor, ou ganho, de cada coalizão deve ser dividido entre os agentes pertencentes de acordo com algum esquema. Usualmente, este esquema deve ser projetado de maneira a garantir eqüidade ou estabilidade.

Em certos casos de aplicações de SMA, o cálculo do valor da coalizão é uma atividade complexa, que pode dificultar ainda mais o processo de formação de coalizões. No âmbito deste trabalho, o processo de calcular *v(c)* não será uma atividade custosa, como será demonstrado mais adiante.

Como mencionado anteriormente, a formação da estrutura de coalizão é o gargalo deste processo e deve ser o principal ponto trabalho a fim de se obter um resultado eficiente. A busca pela estrutura de coalizão ótima já foi provada ser é NP - completo.

3. TRABALHOS RELACIONADOS À GERAÇÃO DE ESTRUTURAS DE COALIZÃO DE AGENTES

Em inteligência artificial, e na ciência da computação de maneira geral, o procedimento que determina em qual ordem deve-se investigar as possíveis soluções de um dado problema é usualmente referido como um procedimento de busca. O procedimento de busca é executado sobre uma representação abstrata e estruturada do universo de soluções, denominada espaço de estados. Este modelo é geralmente implementado no algoritmo de busca através de uma estrutura de dados.

O principal enfoque deste procedimento de busca é encontrar a estrutura de coalizão que maximiza o bem-estar social dos agentes. Em domínios que não são necessariamente super-aditivos ou sub-aditivos, algumas coalizões possuem maior valor trabalhando em separado enquanto outras ganham valor ao se unirem. Em tais situações, a estrutura de coalizão que maximiza o bem-estar social varia e a geração da estrutura de coalizão torna-se não trivial.

Denota-se o conjunto de todas as estruturas de coalizão como S^A. O objetivo é maximizar o bem-estar social dos agentes A no sistema, encontrando a estrutura de coalizão

$$CS^* = \arg\max_{CS \in S^A} V(CS)$$

onde:

$$V(CS) = \sum_{S \in CS} v(S)$$

Sandholm et al. (1999) definem uma estrutura de coalizão como uma partição exaustiva que contém somente coalizões disjuntas. Ou seja, cada um dos agentes deve pertencer a exatamente uma coalizão. Outros autores, como Shehory e Kraus (1998), relaxam esta restrição e consideram casos em que as coalizões podem se sobrepor. Neste trabalho não serão tratadas coalizões que não sejam disjuntas, pois não há como trabalhar com tais coalizões no simulador RCR. Para coalizões disjuntas, a complexidade assintótica do espaço de estados (número de estruturas de coalizão) é $O(|A|^{|A|})$ e $w\left(|A|^{|A|/2}\right)$ (SANDHOLM et al., 1999). Ainda, o número exato de estruturas no conjunto S^A é:

$$\sum_{i=1}^{|A|} Z(|A|, i)$$

onde *Z(|A|, i)* é o número de estruturas de coalizão contendo ***i*** coalizões.

De acordo com Sanholm et al. (1999), existe uma distinção primária que pode ser feita entre métodos desenvolvidos para encontrar uma estrutura de coalizão. Para o caso em que se possui capacidade de observar diretamente o valor de cada coalizão *v(S)*, o problema torna-se equivalente a uma classe de problemas de otimização como o empacotamento e cobertura de conjuntos. É equivalente também ao problema da determinação do vencedor em leilões combinatoriais (SANDHOLM et al., 1999). Em alguns domínios pode não ser possível observar *v(S)* diretamente, pois este depende, por exemplo, de cálculos complexos ou negociação entre os agentes. Ou seja, existe um custo associado à informação. No âmbito da RCR, estamos trabalhando com o caso onde podemos observar diretamente *V(s)*.

A seguir, serão demonstrado algoritmos que provêm garantias de que a solução encontrada está dentro de um limite do ótimo. Esses algoritmos chamados de *anytime* fornecem a solução tanto melhor quanto mais tempo puderam processar as informações. Com um simulador que permite um tempo limitado para cada ciclo não temos tempo infinito para calcular cada *CS* e assim devemos fornecer o melhor resultado que temos num curto espaço de tempo. Algoritmos que utilizam programação dinâmica também se propõe a localizar a melhor *CS* porém a qualidade da resposta do algoritmo não melhora com o tempo, apenas se for possível processar todo o algoritmo, e portanto não poderemos utilizá-los aqui. Necessitamos de um algoritmo que fornece uma solução eficiente no tempo que lhe foi dado e portanto necessitamos utilizar um algoritmo do tipo *anytime*.

Considerando que o espaço de busca das estruturas de coalizão é muito grande para ser exaustivamente enumerado, a busca pela estrutura de coalizão deve ser realizada em um subconjunto $N \subset S^A$ das estruturas de coalizão

$$CS_N^* = \arg\max_{CS \in N} V(CS)$$

com a garantia desejável de que a estrutura de coalizão encontrada está dentro de um limite da solução ótima, ou seja, que existe um k finito e tão pequeno quanto possível

$$k = \min\{k\} \qquad onde \;\; k \geq \frac{V(CS^*)}{V(CS^*_N)}$$

O objetivo deste capítulo é fornecer uma base para o leitor compreender o problema relacionado à formação de estruturas de coalizão e como esse problema é abordado na literatura. O primeiro algoritmo a ser apresentado foi publicado por Sandholm et al. (1999) e foi pioneiro em garantir um desempenho de pior caso. O trabalho mostrou ainda que nenhum outro método publicado até aquele momento era capaz de garantir um limite do ótimo. Na seqüência, aborda-se a contribuição de Dang e Jennings (2004), que melhora os resultados de Sandholm e colegas. Por último, é apresentado o método de Rahwan et al. (2007), que realiza uma busca guiada utilizando uma representação mais compacta do espaço de busca e que corresponde ao atual estado da arte. Todos algoritmos descirtos nesse capítulo garantem este limite k.

3.1 Sandholm et al.

De acordo com Sandholm et al. (1999), o espaço de busca das estruturas de coalizões pode ser visto como um reticulado parcialmente ordenado. Neste reticulado, o elemento supremo é uma estrutura de coalizão onde cada coalizão consiste de apenas um agente e o elemento ínfimo é a grande coalizão contendo todos os agentes. A Figura 3.1 mostra um exemplo deste reticulado, denominado de grafo de estruturas de coalizão, para um sistema com quatro agentes.

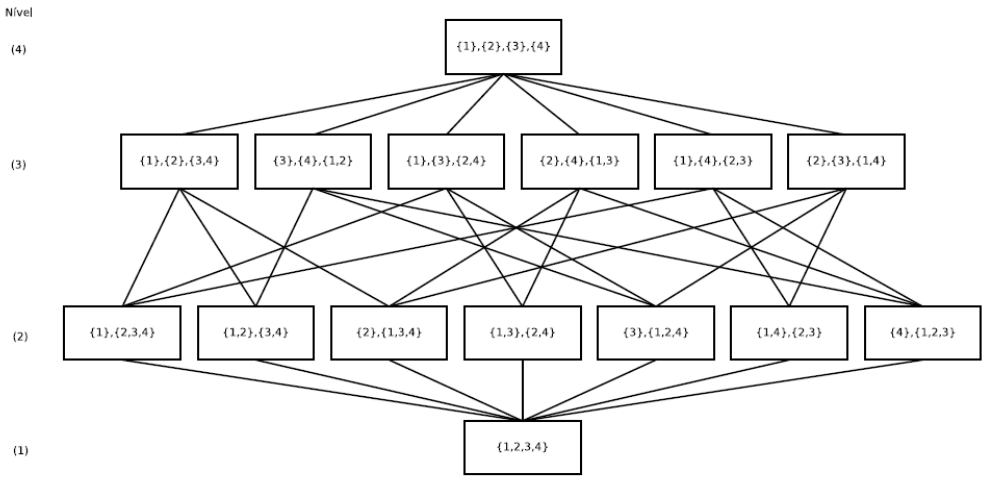

Figura 3.1: Grafo da estrutura de coalizão para um jogo de quatro agentes, com nós representando estruturas de coalizão. Figura adaptada de Sandholm et. al. (1999).

Quando o grafo é percorrido da base em direção ao topo, temos que se uma coalizão *CS1* está acima de uma coalizão *CS2*, a coalizão *CS1* pode ser obtida a partir de *CS2* ao particionar uma de suas coalizões em duas coalizões. Então *CS1* tem exatamente uma coalizão a mais que *CS2*. Esta ordem parcial impõe uma estrutura no reticulado de maneira que podemos descrever o reticulado como tendo ***n*** níveis, onde ***n*** é o número de agentes. Ademais, cada estrutura no nível ***i*** contém exatamente ***i*** coalizões. Algo similar ocorre quando o grafo é percorrido do topo em direção a base, porém neste caso a operação não é uma divisão e sim uma união.

Através do Teorema 1, Sandholm e colegas determinam como um limite ***k*** da solução ótima pode ser estabelecido realizando a busca na menor porção possível do espaço de estados representado pelo grafo de estruturas de coalizão. Adicionalmente, os autores provam outros dois teoremas que garantem que este limite é forte e que nenhum algoritmo diferente de um algoritmo que visite inicialmente os dois níveis inferiores consegue estabelecer um limite visitando apenas $n = 2^{|A|-1}$ estruturas ou menos.

Teorema 1 (Sandholm et al., 1999). Busca mínima para estabelecer um limite. Para limitar ***k***, é suficiente realizar uma busca nos dois níveis inferiores do grafo de estrutura da coalizão. Desta maneira, o limite é ***k*** = |A| e o número de nós visitados é $n = 2^{|A|-1}$.

Segundo os autores, estes resultados podem ser interpretados de maneira positiva ou como um resultado de impossibilidade. Na interpretação positiva do fato tem-se que um limite de desempenho pode ser garantido sem a busca exaustiva do espaço de estados. Ademais, o limite pode ser estabelecido em tempo linear em relação ao tamanho da entrada e com o aumento do número de agentes a fração do espaço de busca que necessita ser visitada tende a zero ($N_{\min}/|M| \to 0$ quando $|A| \to \infty$). Visto como resultado de impossibilidade, estabelece-se que um número exponencial de estruturas de coalizão $2^{|A|-1}$ deve ser buscado antes de se obter alguma garantia de desempenho.

Algoritmo 3.1: "*Coalition-structure-search*" (CSS) (SANDHOLM et al., 1999)

Passo 1: Realizar uma busca nos dois níveis inferiores do grafo de estrutura da coalizão.

Passo 2: Continuar realizando uma busca em largura a partir do nível superior do grafo enquanto houver tempo suficiente ou até visitar todo o grafo.

Passo 3: Retornar a estrutura de coalizão que possui a maior utilidade entre as vistas até agora.

Partindo dos resultados acima mencionados, os autores ainda mostram como o limite pode ser diminuído com a realização de busca adicional. Para tanto, propõem o Algoritmo 3.1. Este algoritmo estabelece o limite |A| realizando inicialmente a busca nos dois níveis inferiores, mínima conforme o Teorema 1, e depois segue com a execução iterada de um passo *anytime* que reduz o limite enquanto visita os espaços dos outros níveis. O Teorema 2 provado pelos autores, estabelece como o limite é reduzido com esta busca adicional. A Tabela 3.2 mostra os limites que podem ser obtidos através da execução do algoritmo de Sandholm et al. e que são garantidos pelo Teorema 2.

Nível	Limite
$\|A\|$	$\|A\|/2$
$\|A\|-1$	$\|A\|/2$
$\|A\|-2$	$\|A\|/3$
$\|A\|-3$	$\|A\|/3$
$\|A\|-4$	$\|A\|/4$
$\|A\|-5$	$\|A\|/4$
...	...
2	$\|A\|$
1	*nenhum*

Tabela 3.2: Redução gradual do limite da solução ótima garantida pelo Teorema 2.

Teorema 2 (Sandholm et al., 1999). Diminuindo o limite com busca adicional. Após realizar uma busca no nível ***l***, o limite ***k*** é $\left\lceil \frac{|A|}{h} \right\rceil$ se |A| ≡ (h − 1) *mod* h e |A| ≡ *l mod* 2. Caso contrário o limite ***k*** é $\left\lfloor \frac{|A|}{h} \right\rfloor$. Onde $h = \left\lfloor \frac{|A|-l}{2} \right\rfloor + 2$.

Sandholm e colegas apresentaram resultados teóricos de seus estudos e focam na análise das garantias de desempenho no pior caso.

3.2 Dang e Jennings

O trabalho de Dang e Jennings (2004) introduz um algoritmo *anytime* para o problema da geração da estrutura de coalizão. Os autores comparam seu algoritmo com o de Sandholm et al. e mostram que, quando são desejados limites do ótimo que são pequenos (baixos), o método proposto é 107 vezes mais rápido (para sistemas com 50 agentes), 1023 vezes mais rápido (para sistemas com 100 agentes) e 10379 vezes mais rápido (para sistemas com 1000 agentes). Ainda, para maiores limites do ótimo, os autores mostram que não há diferença significativa entre o seu algoritmo e o de Sandholm e colegas.

Algumas definições feitas pelos autores são necessárias para a compreensão do algoritmo e, como a intenção deste trabalho não é trabalhar esse algoritmo, serão apresentadas apenas as definições mais importantes para a compreensão do mesmo. Seja L_k o conjunto de

todas as estruturas de coalizão com ***k*** coalizões. Assim, tem-se o conjunto ***L*** de todas as estruturas de coalizão:

$$L = \bigcup_{k=1}^{n} L_k \ ,$$

onde ***n*** é o número de agentes que é igual ao número máximo de coalizões (todos os agentes disjuntos).

Definição 8: Seja *SL*(***n***, ***k***, ***c***) o conjunto de todas as estruturas de coalizão que possuem ***k*** coalizões e no mínimo uma coalizão com cardinalidade não menor que ***c***.

Definição 9: Seja *SL(**n**, **c**)* o conjunto de todas as estruturas de coalizão com cardinalidade entre 3 e $n-1$ e que possuem ao menos uma coalizão com cardinalidade não menor que ***c***. Seria o mesmo que:

$$SL(n,c) = \bigcup_{k=3}^{n-1} SL(n,k,c) \ .$$

Dadas as definições 8 e 9, o algoritmo de Dang e Jennings pode ser explicado da seguinte maneira. Inicialmente, uma busca é efetuada em todas as estruturas de coalizão que possuem uma, duas ou n coalizões. Ou seja, nos conjuntos *L1*, *L2* e *Ln*, respectivamente. Este primeiro passo do algoritmo efetua a busca nos dois níveis inferiores e no nível superior do grafo de estruturas de coalizão (Figura 3.1). Após, o algoritmo de Sandholm et. al. busca seqüencialmente nos conjuntos restantes L_k, com ***k*** variando entre 3 e $n-1$ $(3 \leq \mathrm{k} \leq n-1)$. Porém, o algoritmo de Dang e Jennings busca apenas em alguns subconjuntos específicos de L_k. Mais especificamente, a busca ocorre no conjunto de todas as estruturas de coalizão que possuem ***k*** coalizões e no mínimo uma coalizão com cardinalidade não menor que $\left\lceil \frac{n(q-1)}{q} \right\rceil$, iterando em passos com ***q*** variando de $\left\lfloor \frac{n+1}{4} \right\rfloor$ até 2. A escolha destes valores como limitantes da iteração de ***q*** são explicados posteriormente. Para evidenciar a diferença entre o caminho da busca executada pelos dois algoritmos, o esquema da Figura 3.3 representa uma comparação entre a execução da busca pela estrutura da coalizão feita pelos algoritmos de Dang e Jennings e Sandholm e colegas.

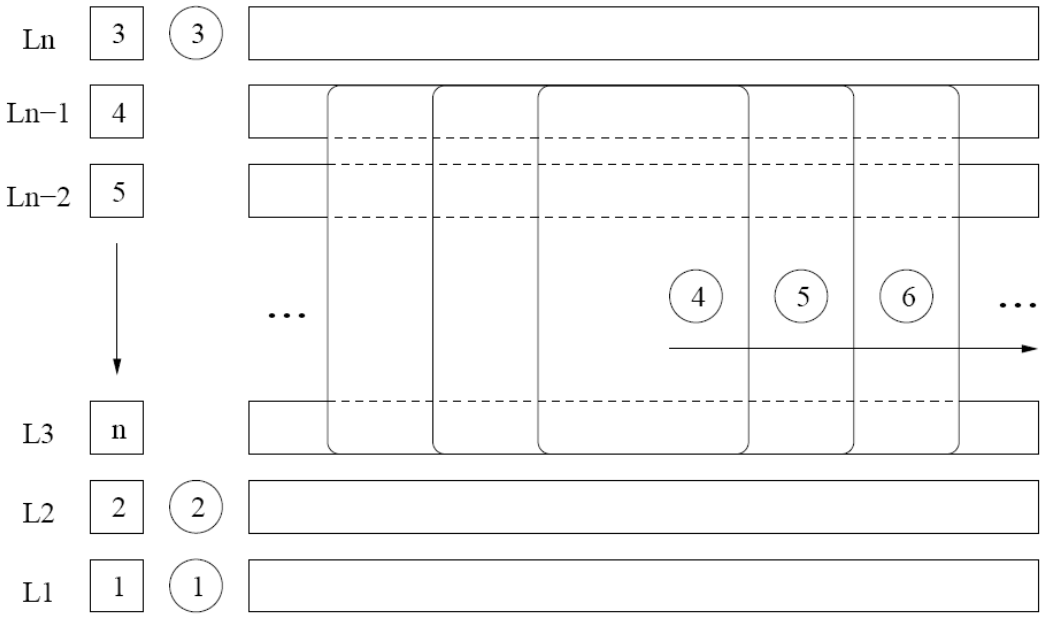

Figura 3.3: Comparação entre os caminhos percorridos pelos algoritmos de busca de Dang e Jennings e Sandholm et al., com os valores dentro de círculos e quadrados representando, respectivamente, os passos dos dois algoritmos. Figura adaptada de (Dang; Jennigs, 2004).

O Teorema 3 garante que a solução retornada está dentro de um limite do ótimo e que este limite é reduzido com a execução de cada passo de busca adicional, o que classifica o algoritmo de Dang e Jennings como *anytime*.

Teorema 3: Após terminar a busca em $SL\left(n, \left\lceil n(q-1)/q \right\rceil\right)$, a solução gerada pelo algoritmo está dentro do limite $b = 2q - 1$ do ótimo.

Com base no Teorema 3, é possível justificar a escolha de $\left\lfloor \frac{n+1}{4} \right\rfloor$ como o primeiro valor de q a ser iterado, pois considerando os Teoremas 1 e 2 de Sandholm et al. que determinam que após buscar em *L1*, *L2* e *Ln* o limite estabelecido é $b = \left\lceil \frac{n}{2} \right\rceil$, deve-se escolher o maior valor possível de q tal que $2q - 1 < \left\lceil \frac{n}{2} \right\rceil$, ou seja $q = \left\lfloor \frac{n+1}{4} \right\rfloor$. A escolha do valor 2 como limitante inferior da iteração é justificada de maneira semelhante.

3.3 Rahwan et al.

Rahwan et al. (2007) apresentaram um novo algoritmo *anytime* para a geração da estrutura de coalizão que utiliza uma nova representação, mais concisa, para o espaço de estados. O método dos autores é mais eficiente do que os algoritmos existentes até aquele momento em termos de tempo de computação. Pode gerar soluções que estão dentro de um limite pequeno do ótimo ou que são ótimas, dependendo do objetivo desejado explicitamente definido na entrada.

Os autores demonstram a eficácia de seu algoritmo empiricamente em comparação com um algoritmo de programação dinâmica, desenvolvido para o problema de particionamento completo de conjuntos (YEH, 1986). Os resultados mostram que o algoritmo de Rahwan et al. foi capaz de obter soluções que estão, no pior caso, dentro do limite de 1% do ótimo utilizando 0, 0043% do tempo e 77% da memória gastos pelo algoritmo de programação dinâmica, em um sistema com 20 agentes.

Ao contrário dos algoritmos de Sandholm et al. e Dang e Jennings vistos anteriormente, que observam os valores *v(C)* das estruturas de coalizão, o método de Rahwan e colegas realiza uma busca observando diretamente os valores *v(C)* das coalizões.

De maneira simplificada, o método consiste em particionar o espaço de estados em um pequeno número de conjuntos de estruturas de coalizão que contém coalizões de determinados tamanhos, chamado de *"configuração"*. Então, sob esta representação do espaço, uma busca "online" (online significa que utiliza a informação obtida ao longo da execução da busca para guiar o processo de busca) é realizada utilizando uma heurística para guiar o processo. Esta heurística realiza cortes no espaço de busca, desconsiderando estruturas não válidas ou redundantes. Após buscar em apenas uma parte relativamente pequena do espaço, a heurística direciona a busca para as melhores configurações estimando a média e o limite superior dos valores das soluções em cada configuração. Ao concluir a busca em todas as estruturas de uma certa configuração, as estimativas sobre as outras configurações são refinadas. Com estas estimativas atualizadas, o algoritmo é capaz de efetuar ainda mais cortes no espaço de busca, ignorando as configurações que possuem um limite superior menor que o valor da melhor solução encontrada até o momento.

$$CL_S = \begin{array}{|c|c|c|c|} \hline CL_1 & CL_2 & CL_3 & CL_4 \\ \hline \{4\} & \{3,4\} & \{2,3,4\} & \{1,2,3,4\} \\ \{3\} & \{2,4\} & \{1,3,4\} & \\ \{2\} & \{2,3\} & \{1,2,4\} & \\ \{1\} & \{1,4\} & \{1,2,3\} & \\ & \{1,3\} & & \\ & \{1,2\} & & \\ \hline \end{array}$$

$$G = \{4\}, F(G) = \{\ \{\{1,2,3,4\}\}$$

$$G = \{2,2\}, F(G) = \left\{ \begin{array}{l} \{\{1,2\},\{3,4\}\} \\ \{\{1,3\},\{2,4\}\} \\ \{\{1,4\},\{2,3\}\} \end{array} \right. \quad G = \{1,3\}, F(G) = \left\{ \begin{array}{l} \{\{4\},\{1,2,3\}\} \\ \{\{3\},\{1,2,4\}\} \\ \{\{2\},\{1,3,4\}\} \\ \{\{1\},\{2,3,4\}\} \end{array} \right.$$

$$G = \{1,1,2\}, F(G) = \left\{ \begin{array}{l} \{\{3\},\{4\},\{1,2\}\} \\ \{\{2\},\{4\},\{1,3\}\} \\ \{\{2\},\{3\},\{1,4\}\} \\ \{\{1\},\{4\},\{2,3\}\} \\ \{\{1\},\{3\},\{2,4\}\} \\ \{\{1\},\{2\},\{3,4\}\} \end{array} \right.$$

$$G = \{1,1,1,1\}, F(G) = \{\ \{\{1\},\{2\},\{3\},\{4\}\}$$

Figura 3.4: Exemplo do espaço de estados mostrando N para 4 agentes com os elementos F(G) expandidos. Figura adaptada de Rahwan et. al. (2007)

A seguir são apresentadas as definições essenciais utilizadas pelo algoritmo de Rahwan e colegas. Seja $CL_s \in 2^A$ a lista de coalizões de tamanho $S \in \{1,2,..,n\}$ tal que CL é o conjunto de todas as listas de coalizões ($CL = \bigcup_{S=1}^{n} CL_S$). Seja $G1, G2, \ldots, G_{|G_{CS}|} \in G_{CS}$ o conjunto de todas as possíveis configurações únicas. Ainda, seja F : $G_{CS} \rightarrow \in_{CS}$ uma função que recebe uma configuração e retorna todas as estruturas de coalizão daquela configuração tal que N = {F(G1), F(G2), . . . , F($G_{|G_{CS}|}$)} representa uma lista onde cada elemento é um conjunto de estruturas de coalizão da mesma configuração. Neste caso, a notação *N1,N2, . . . ,*

$N_{G_{CS}} \in N$ denota os elementos de *N*. A Figura 3.4 (acima) mostra um exemplo com as listas CL_s e *N* para um sistema com quatro agentes.

Como entrada, o algoritmo recebe o conjunto *CL* para o qual o valor da função característica *v(C)* é conhecido para todas as coalizões *C* pertencentes a todas as listas $CL_s \in CL$. Um limite de aceitação do ótimo $\beta \in [0, 1]$ também é informado como entrada, significando que quando $\beta = 1$ o algoritmo deve parar se a solução ótima for encontrada e quando $\beta = 0$ qualquer solução encontrada é retornada. O algoritmo é constituído por três estágios:

1. Estágio de pré-processamento que percorre a entrada para obter os valores máximo e médio de cada $CL_s \in CL$.

2. Utiliza os valores do primeiro estágio para gerar a lista de $G \in G_{CS}$ únicos e selecionar o elemento *F(G)* no qual a busca pela estrutura de coalizão ótima deve ocorrer.

3. Determina os valores de cada estrutura de coalizão $CS \in F(G)$ para encontrar a solução $CS*$.

O estágio de pré-processamento calcula os valores máximo e médio para cada lista $CL_s \in CL.$, denotados por max_s e avg_s. Inicialmente, são verificados os valores das estruturas contendo uma coalizão (a grande coalizão) da lista $CL_{|A|}$ e contendo |A| coalizões (ao somar os valores das coalizões na lista CL_1). A melhor dentre estas duas estruturas é guardada em *CS'*. Após, para cada lista CL_S, observa-se todas as suas coalizões ***C***, guardando os valores máximo e médio. A economia ocorre quando o algoritmo verifica as listas CL_S e $CL_{|A|-S}$ ao mesmo tempo. Para tanto, denota-se uma coalizão de índice ***c*** na lista *CL* como CL^c, onde
$\boldsymbol{c} \in 1, \ldots, |CL|$. A atribuição de índices é feita de tal maneira que resulta que cada elemento $C \in CL_S$ no índice $\boldsymbol{c} \in [1, \ldots, S]$ é igual ao elemento $C' \in CL_{|A|-S}$ no índice $|CL_S| - \boldsymbol{c} + 1$. Outra otimização feita é verificar ao mesmo tempo os valores de todas as estruturas que seguem a configuração $G = \{1, \ldots, 1, i\}$. Com o término deste primeiro estágio uma solução é retornada com a garantia de estar a um limite |A|/2 do ótimo. Este limite é garantido pelo

Teorema 1 de Sandholm et al.. A complexidade computacional do algoritmo de pré-processamento é $O(2^{|A|})$.

O objetivo do segundo estágio é encontrar o *F(G)* que contém a estrutura *CS* ótima ou com valor dentro do limite estabelecido por β. Para tanto, inicialmente são enumeradas todas as configurações em G_{CS} utilizando algum algoritmo que gere partições de um número inteiro. Por exemplo, para a |A| = 4 as partições possíveis são {{4}}, {{3}, {1}}, {{2}, {2}}, {{2}, {1}, {1}} e {{1}, {1}, {1}, {1}}. A seguir, utilizando os valores de máximo e média calculados no pré-processamento, calcula-se o limite superior UB_G e valor médio AVG_G para cada G. Estes valores são passados como entrada para essa etapa. Mas antes, compara-se a solução *V (CS')* encontrada no estágio de pré-processamento com o limite superior considerando β. Caso o valor esteja dentro do limite estabelecido como aceitável, o método pára e retorna a solução *CS'*. Caso contrário, o processo continua para retornar um elemento ***G*** de G_{CS} que possui a maior probabilidade de conter a estrutura de coalizão procurada. Basicamente o algoritmo realiza isto calculando um ganho esperado sobre os elementos ***G***, que representa a quantidade de estados que pode ser evitada após a busca no ***G*** específico.

O terceiro estágio do algoritmo de Rahwan et al. é responsável por realizar a busca de $CS \in F(G)$, caso não tenha ocorrido o caso de uma solução prematura ter sido obtida nos estágios anteriores. O algoritmo gera e verifica todas as estruturas de coalizão dentro do elemento particular ***G*** estabelecido pelo algoritmo do estágio 2. Porém, ao invés de executar uma busca ingênua em um grande número de elementos, o algoritmo evita gerar coalizões não disjuntas (inválidas neste caso) e, mais importante ainda, evita estruturas de coalizão redundantes.

4. 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 infra-estrutura de energia, água e comunicação foi extremamente danificada. Aproximadamente 6.000 pessoas morreram, e mais de 300.000 ficaram feridas. Mais de um quinto da população perdeu sua casa e os danos estruturais foram avaliados em mais de um bilhão de dólares.

Esta catástrofe alertou para a necessidade de se criar um sistema que possa servir de base para elaboração de testes e planos de emergência a serem utilizados para auxiliar os esforços humanos em situações de catástrofes como a de Kobe. Com base nisso, fundou-se a Robocup Rescue Simulation League, onde são centralizados os esforços na busca deste sistema. Esta liga disponibiliza um simulador de desastres (terremotos) e operações de resgate, denominado Robocup Rescue. Neste simulador é possível avaliar a qualidade e eficiência de abordagens multiagentes tendo como base a idéia de salvar o maior número de vidas e minimizar os danos causados pelo terremoto.

A descrição do ocorrido no terremoto de Kobe é apresentada em (Kitano, 2000) e complementada com os pontos mais importantes a serem tratados em situações críticas (como terremotos, por exemplo):

- Obstrução de vias de circulação devido ao grande número de prédios e edificações que tombam nestes cenários. Isso dificulta muito a chegada das equipes de resgate aos locais atingidos pela catástrofe;

- Serviços básicos como fornecimento de energia, água e gás entre outros comprometidos

- Inúmeros focos de incêndio por toda região, que podem rapidamente se espalhar por vastas áreas, principalmente pelo fato do abastecimento de água estar comprometido;

- Inúmeras pessoas feriadas e possivelmente soterradas, que necessitam de tratamento médico imediato;

- Estrutura de comunicação danificada pela queda de linhas de comunicação e destruição de equipamentos sensíveis, tornando as informações necessárias para as equipes de resgate escassas e imprecisas.

A seguir será realizada uma descrição do simulador Robocup Rescue. Inicialmente será explicado o simulador e seu funcionamento e em seguida serão tratados os agentes programáveis existentes no simulador e através do qual será possível estabelecer estratégias de resgate e combate à incêndios.

4.1 O simulador

Inicialmente, o simulador Robocup Rescue recebe como entrada dados geográficos que indicarão onde estão as ruas, os prédios, cruzamentos e todo tipo de informação necessária para se construir um mapa da região a ser investigada. Juntamente com estes dados, o simulador recebe informações acerca do terremoto, com a qual ele reproduz o dano que seria causado em prédios e ruas, bem como incêndios, civis feridos e soterrados e todo outro tipo de dano. Estes dados são todos objetos do simulador, que tem seu estado alterado para identificar incêndios (no caso de prédios), bloqueios (no caso de ruas) ou ferimentos e morte (no caso de humanos). Uma vez iniciada a simulação, o sistema evolui ela gradativamente, espalhando o fogo pela cidade, aumentando o número de escombros e agravando o estado de saúde dos civis.

Para lidar com essa situação, o simulador possui seis tipos de agentes, chamados de "rescue agents" (agentes de resgate). Estes agentes são divididos em dois grupos: agentes centrais (também chamados de Centrais de Comando, que são agentes imóveis capazes de se comunicar entre si e com qualquer agente que esteja sob seu comando) e agentes de campo (platoon agents), agentes capazes de se mover e coletar informações, bem como realizar as tarefas a que estiverem aptos. A tabela 4.1 abaixo contém a designação de cada agente bem como qual a tarefa dele.

Tipo de Agente	Função	Classe
Brigada de incêndio	Combater incêndios em construções.	Campo
Central de Bombeiros	Centralizar a coordenação de bombeiros.	Central
Força policial	Remover escombros e bloqueios de ruas.	Campo
Central de Policiais	Centralizar a coordenação de policiais.	Central
Time de Ambulância	Resgatar civis soterrados e feridos.	Campo
Central de Ambulâncias	Centralizar a coordenação de ambulâncias.	Central

Tabela 4.1: Agentes de resgate existentes no RoboCup Rescue com seus papéis e classes.

A fim de simplificação, as seguintes nomenclaturas serão utilizadas para os agentes de campo:

Brigada de incêndio (Fire Brigade) será chamada de "bombeiro".
Força policial (Police Force) será chamada de "polícia".
Time de Ambulância (Ambulance Team) será chamado de "ambulância".

Os agentes de campo podem interagir com o ambiente e recebem a cada instante de tempo informações do simulador sobre o que eles percebem do ambiente. A percepção de cada agente é limitada a um raio de 10m. É importante ressaltar que o agente não pode perceber apenas um pedaço de um objeto. Quando um agente "enxerga" um prédio, ele recebe informação sobre todo o prédio e o mesmo se aplica às ruas. Para um agente enxergar um prédio, ele deve estar ao alcance do centro deste prédio, pois o nó que determina o objeto prédio se localiza no centro dele.

Inicialmente os agentes conhecem todos os objetos estáticos do sistema (ruas, prédios, nodos, etc.) e a percepção deles altera o estado desses objetos (um prédio que não estava em chamas no ciclo anterior pode ter pego fogo nesse ciclo). A figura abaixo (figura 4.1) demonstra como os agentes percebem o sistema. Nela, o agente recebe informações sobre os prédios *C3* e *C2*, bem como sobre os nodos *N2* e *N3* e os bloqueios *B2* e *B3*. Ele não consegue perceber que há um bloqueio em *B1*, pois isto está fora de seu alcance de visão, mas ele percebe que há um civil dentro do prédio *C2* (*Civ1*) e tem informações sobre a intensidade do incêndio em *C2* e sobre o estado do civil em *Civ1* (o civil pode estar ferido, morto ou mesmo soterrado).

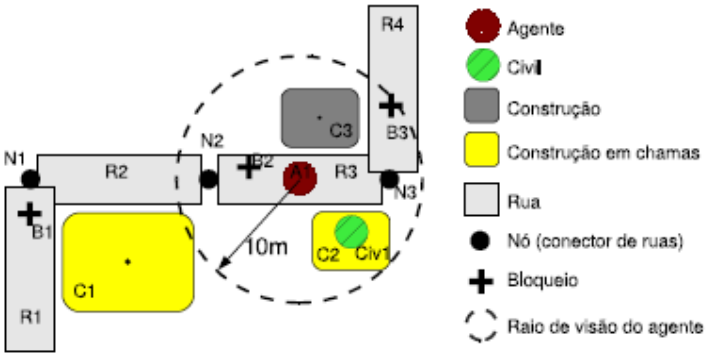

Figura 4.1: Esquema de percepção dos agentes no simulador RoboCup Rescue. Figura adaptada de (Santos, 2009).

Cada tipo agente de campo tem uma ação especifica que somente ele pode realizar (os agentes centrais não podem interagir com o ambiente) e cada agente somente poderá realizar uma ação a cada ciclo. Um agente da brigada de incêndio, por exemplo, não pode combater dois incêndios no mesmo ciclo. A seguir está descrito as ações possíveis, seu significado dentro do simulador e quais agentes podem realizar a ação.

- Mover - Um caminho é percorrido e a localização do agente é alterada: Todos
- Extinguir Incêndio - Uma quantidade de água é despejada: Bombeiro
- Remover bloqueio - Uma parcela de um bloqueio é removida: Polícia
- Resgatar civil - Uma parcela de escombros é removida de sobre um civil soterrado: Ambulâncias
- Carregar civil - Um civil é colocado na ambulância: Ambulância
- Descarregar civil - Um civil é retirado da ambulância: Ambulância

4.1.1 Medição do escore

Para medirmos a eficiência de uma abordagem em relação à outra, foi desenvolvido um método de cálculo para saber a quantidade de pontos obtidos na simulação. Esse cálculo leva em consideração a área total construída preservada (não incendiada) e a quantidade de sobreviventes e é apresentada abaixo:

$$escore = \left(P + HP / HP_{inicial} \right) * \sqrt{B / B_{inicial}} \quad (4.1)$$

Onde:

• P: quantidade total de agentes vivos.

• HP: soma dos níveis de saúde (HP) dos agentes.

• $HP_{inicial}$: 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 melhora o fluxo de agentes.

4.1.2 Características destacadas do simulador RCR

Inicialmente é importante ressaltar que o ambiente é apenas parcialmente observável, uma vez que os agentes dentro do ambiente possuem capacidade limitada de percepção. Assim, o ambiente como um todo não pode ser reconstruído juntando as informações coletadas pelos agentes num dado momentos (mesmo que todos os agentes do cenário juntem seu conhecimento), forçando os agentes a explorarem o cenário e lidarem com situações e estímulos não previstos anteriormente.

O ambiente também tem um comportamento estocástico, tendo um comportamento aleatório em certos atributos (um exemplo são os incêndios, que não é possível determinar a sua propagação com um alto grau de certeza). Os agentes então são forçados a lidar com situações não previsíveis e tem que se adaptar rapidamente às mudanças do cenário. Isso também faz com que o cenário possa ter um número "infinito" de estados possíveis.

Por fim, o ambiente é dinâmico, tendo alterações a todo instante de maneira imprevisível. Os incêndios podem destruir prédios e se espalhar de maneira aleatória, tornando ruas bloqueadas ou mesmo ameaçando a vida de civis. Esse ambiente dinâmico incita o agente a alterar constantemente suas ações e se adaptar às novas condições do ambiente.

4.2 Agentes

Dentro do simulador RCR, temos um conjunto de agentes que são responsáveis pela realização das diversas tarefas existentes no simulador. Esse conjunto de agentes são os únicos objetos dentro do simulador que podem ser alterados sem modificar a estrutura proposta inicialmente para o simulador. Cada tipo de agente deve ser programado individualmente e a seguir serão fornecidas todas as características fundamentais necessárias para a compreensão do funcionamento dos agentes.

4.2.1 Interação entre os agentes

Em um cenário de desastre, como o representado no simulador, é fundamental que as equipes se coordenem para conseguir executar suas tarefas da melhor maneira possível. Assim, é possível que haja comunicação entre os agentes. Devido ao estado da cidade pós-catástrofe, a comunicação não pode ser considerada perfeita. As mensagens enviadas devem obedecer a uma serie de restrições, como a quantidade de mensagens possíveis e seu tamanho máximo, além de estarem sujeitas a perdas e atrasos.

Na RCR existem dois tipos de comunicações possíveis: por radio e por voz. Mensagens de radio podem ser enviadas e recebidas por todos agentes. Os agentes de um certo tipo apenas podem enviar mensagens para agentes do mesmo tipo (além de seus agentes centrais) e agentes centrais podem enviar mensagens entre si. Se, por exemplo, um bombeiro deseja enviar mensagem para uma ambulância, é necessário enviar para seu agente central (Fire Station) e este enviará para o agente central das ambulâncias (Ambulance Center) que repassará a mensagem para as ambulâncias. Essas mensagens têm alcance ilimitado, podendo ser enviadas e recebidas em qualquer parte do mapa.

Mensagens de radio têm tamanho máximo de 256 bytes e cada agente de campo pode enviar e receber no máximo uma mensagem por ciclo. Agentes centrais podem enviar e receber até 2*N mensagens por ciclo, onde N é o número de agentes de campo que esse agente central tem sob sua responsabilidade. As mensagens não possuem destinatário, sendo assim impossível distinguir para qual agente a mensagem foi enviada.

O outro tipo de comunicação (chamado de comunicação por voz) é muito mais restritivo. Seria o equivalente a uma pessoa gritar para a outra. Essas mensagens podem ser

enviadas por qualquer agente ou mesmo por civis e tem o alcance de 30m. Também são limitadas a 256 bytes e são muito mais susceptíveis a ruídos e perdas, sendo isso proporcional a distância entre o agente emitindo a mensagem e o agente recebendo a mensagem. Essa mensagem não pode ser direcionada e o agente apenas pode optar por não recebê-la baseado no identificado de quem enviou a mensagem e não no conteúdo. Mensagens de voz enviadas por civis são ouvidas por todos os agentes e podem ajudar na identificação de um civil soterrado ou preso dentro de um prédio que não esteja visível ao agente.

4.2.2 Ação dos agentes

Dependendo das dimensões de uma tarefa, um único agente pode não ser capaz de realizá-la sozinho de maneira eficientemente. No caso de um incêndio em estado avançado ou em uma grande construção, um único bombeiro não será suficiente para controlar o incêndio. Já um bloqueio de rua será eliminado mais rapidamente se diversas forças policiais removerem várias parcelas dele simultaneamente. Ainda mais grave, um civil pode não resistir aos ferimentos se apenas uma ambulância for ajudá-lo. Em casos onde ele esteja soterrado, vários times de ambulâncias atuando em conjunto removerão mais rapidamente os escombros do civil. Assim, espera-se que em muitos casos seja necessário um grande número de agentes para realizar uma tarefa e que estes trabalhem em conjunto para poderem realizá-la de maneira eficiente.

A seguir será expandido o conceito de agentes de campo e será especificado a função de cada agente no cenário da RCR.

4.2.2.1 Bombeiro (Fire Brigade)

Combate incêndios. O objetivo dele é minimizar a área total destruída por incêndios. Inicialmente o simulador apresenta alguns focos de incêndios que irão se propagar aos prédios vizinhos. Para combater o incêndio, o agente bombeiro irá despejar água no prédio. O alcance limite do jato de água é de 30 metros. Portanto, o bombeiro deve estar situado sobre algum nodo da rua onde haja o prédio em chamas e que esteja a menos de 30m do prédio. 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.

Existem diversas informações associadas a um incêndio que devem ser levadas em consideração pelo bombeiro. Essas informações indicam o quão importante é extinguir um particular incêndio e podem ser usadas para decidir qual incêndio apagar primeiro. Um incêndio em um prédio muito grande, por exemplo, irá alterar o escore de maneira negativa muito mais rapidamente do que um incêndio em um prédio com área pequena. Portanto, os principais atributos que devem ser levados em consideração quando houver mais de uma opção de ação para o bombeiro são:

- Área: quanto maior o tamanho da região afetada pelo fogo, maior será o peso no escore final do simulador. Um prédio muito grande pode ser mais importante do que diversos prédio pequenos se levado em conta o valor que a destruição desse prédio terá no escore final.

- Intensidade do Incêndio: os incêndios podem ter três intensidades: pequeno, médio e alto. Quanto maior a intensidade, maior o dano ao prédio e maior será a perda de pontos no escore. Um prédio que esteja pegando fogo por muito tempo e já esteja com a intensidade do incêndio alto poderá ser completamente destruído, sendo este dano irreparável.

- Custo para apagar o incêndio e material do prédio: nem todos os incêndios necessitam da mesma quantidade de água para se extinguirem. O custo necessário para apagar um incêndio depende da área dele, da intensidade do incêndio e principalmente do material com que o prédio foi construído. Um prédio pode ser destruído pelo fogo mais rapidamente que outro se o material daquele prédio for mais inflamável. Assim, ao medir o custo necessário para apagar o incêndio (o custo medido pode ser a quantidade de água necessária para apagar o incêndio ou o número de bombeiros necessários para apagar o fogo de forma eficiente), o material do prédio deve ser levado em consideração.

- Área ao redor do prédio: Apesar desse atributo não ser um atributo diretamente do prédio, é necessário considerá-lo ao escolher qual prédio terá seu incêndio apagado. Um prédio isolado, sem nenhuma outra construção ao redor, terá, em longo prazo, um efeito no escore muito menor que um prédio em uma área central do mapa. Um prédio em uma área central pode espalhar o fogo e por em risco todos os prédios ao seu redor.

A figura 4.2 mostra o agente bombeiro (circulo vermelho), com seu raio de percepção de 10m enxergando o prédio *C2* em chamas e com capacidade de despejar água num raio de 30m. O agente não consegue despejar água no prédio *C1*, pois o alcance do jato de água não é suficiente para atingir o centro do prédio. Como o nó que representa o prédio está no centro dele, o agente somente irá receber informações sobre o prédio ou poder jogar água no prédio quando o alcance de sua função for suficiente para se estender até o centro do prédio.

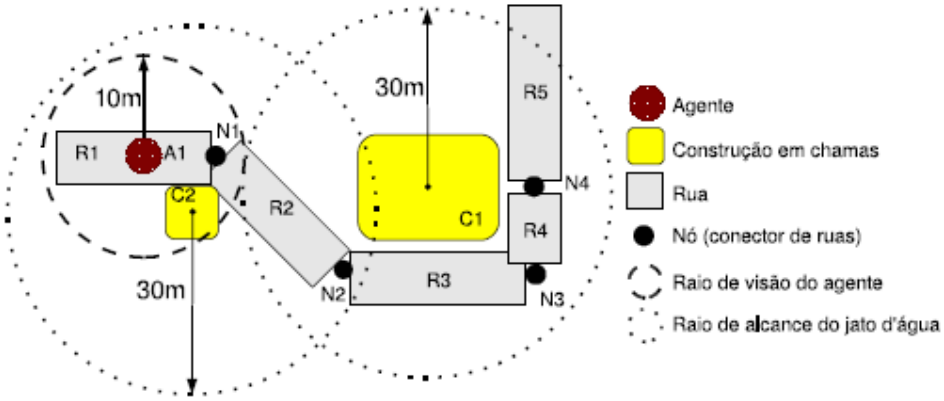

Figura 4.2: Representação do agente, seu raio de visão e distância mínima para extinguir um incêndio. Figura adaptada de (Santos, 2009).

4.2.2.2 Policial (Police Force)

Remoção de bloqueios de ruas. O objetivo é minimizar a quantidade de ruas obstruídas para melhorar o fluxo de bombeiros e ambulâncias. Tarefas de remoção de bloqueios devem ser realizadas por agentes policiais. Para realizar a tarefa, o policial deve localizar-se em cima do bloqueio.

Apesar do bloqueio não interferir diretamente com o escore da simulação, uma estratégia eficiente de remoções de bloqueios permite que ambulâncias e bombeiros circulem pela área do simulador de maneira eficiente e que realizem suas tarefas muito mais rapidamente. Além disso, alguns prédios ou civis podem estar inacessíveis devido à bloqueios na rua onde se encontram. Assim, a tarefa dos policiais se torna de igual importância que as demais.

- Custo do bloqueio: define o número de ciclos que um policial deve trabalhar para remover um determinado bloqueio. Cada policial que se encontra sobre o bloqueio remove uma unidade de custo por ciclo. Assim, um bloqueio de custo cinco pode ser removido em apenas um ciclo se houver cinco policiais trabalhando nele.

- Quantidade de pistas total: Uma rua é dividida em pistas. A rua somente se encontrará completamente bloqueada se todas as suas pistas estiverem bloqueadas. Assim, quanto maior a quantidade de pistas da rua, maiores são as chances de que haja uma pista livre e também maior são as chances de que seja uma rua arterial do cenário.

- Quantidade de pistas livre: Se houver uma pista livre na rua, já será o suficiente para um agente passar por ela. Porém, quanto menos pistas livres houver, mais lento será esse transito e menos o número de agentes que poderão circular naquela rua ao mesmo tempo.

A figura 4.3 representa uma rua com quatro pistas, que estão bloqueadas pelo bloqueio *B1*. O agente *A1* deve então se posicionar sobre o bloqueio e começar a removê-lo. Como o custo de remoção é de 15, o agente levará 15 ciclos para liberar completamente esta rua.

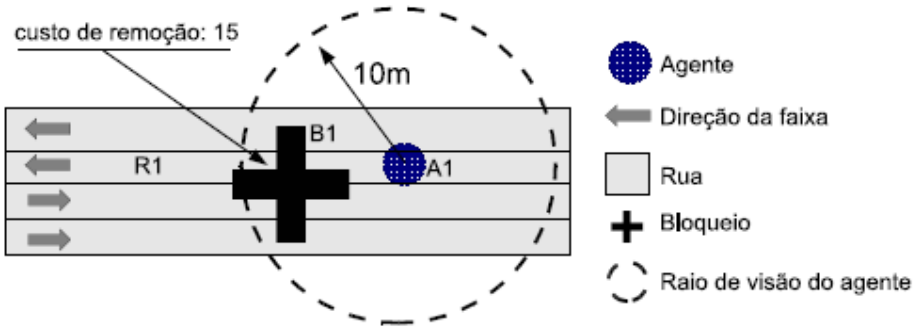

Figura 4.3: Representação do objeto rua pelo simulador bem como o raio de visão de um agente para identificar um determinado bloqueio na rua. Figura adaptada de (Santos, 2009).

4.2.2.3 Ambulância (Ambulance Team)

Resgate de civis feridos. O objetivo é maximizar a quantidade de sobreviventes. Tarefas de resgate de civis devem ser realizadas por agentes do tipo ambulância. Para realizar esta tarefa, a ambulância deve estar situada exatamente na mesma posição do civil ferido. Resgatar

o civil algumas vezes necessita primeiro remover os escombros que estão sobre ele. Assim, um civil soterrado deve ser removido dos escombros e depois carregado para dentro da ambulância, que o levará à um refugio onde estará "salvo".

Um agravante no processo de salvar os civis é que nem sempre o civil estará visível aos agentes e muitas vezes ele só pode ser localizado se a ambulância procurar por ele dentro do prédio onde ele se encontra (o agente pode escutar o pedido de socorro dele e procurar onde está esse civil).

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

- Saúde: Corresponde à "quantidade de vida" do civil. Inicialmente essa quantidade de vida é de 10.000 e o civil perde até 200 pontos de vida a cada ciclo que está soterrado ou ferido. Quando esse atributo chega a zero, o civil está morto.

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

A figura 4.4 mostra as propriedades de diversos civis, com agentes ao redor deles. Como demonstrado abaixo, os civis podem também estar presos dentro de prédios e podem ter dano e soterramento com diversos valores. Cabe aos agentes ambulância escolher qual civil devem salvar primeiro.

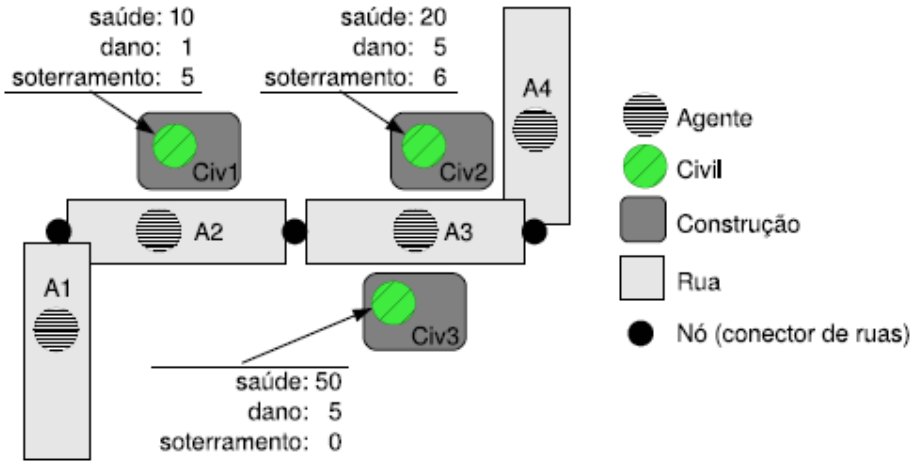

Figura 4.4: Representação de civis presos em prédios com agentes ao redor. Cada civil pode ter um dano e soterramento específico e sua saúde será calculada em função dessas propriedades. Figura adaptada de (Santos, 2009).

5. ESTRATÉGIA PARA O SIMULADOR RCR COM BASE EM COALIZÕES DE AGENTES

Como explicado anteriormente, Sandholm et al. (1999) demonstraram que a tarefa de achar uma coalizão ótima é um problema NP - Completo. Em um cenário como o da RCR, também devemos considerar que não apenas o número de agentes é relevante, mas também devemos considerar que há um enorme número de tarefas possíveis o que torna o número de possíveis coalizões ainda muito maior. Se, por exemplo, houver um número pequeno de tarefas a serem realizadas, provavelmente um ou mais agentes estarão impossibilitados de realizar essa tarefa e o número de coalizões possíveis diminuirá. Em um cenário padrão como o mapa "Kobe", temos apenas 10 agentes bombeiros mas o número de prédios em chamas pode ser varias dezenas. Dessa forma, a escolha da CS ideal deve ser feita com base tanto na divisão dos agentes quanto na escolha da tarefa que estes irão realizar e como isso influenciará no resultado final. Uma coalizão C_1 pode estar associada à uma tarefa T_1 ou T_2 e como isso afetará o resultado final deve ser analisado.

Devido ao elevado número de possibilidades de tarefas a serem resolvidas e um número considerável de agentes possíveis para resolver cada tarefa, tentar achar a melhor coalizão se torna uma tarefa muito difícil de resolver em um curto espaço de tempo. Teríamos que pesquisar entre 2^{10} possíveis coalizões (para o caso citado acima). Isso obviamente não é uma tarefa fácil ou rápida o que foge da proposta do simulador RCR, que é a criação de estratégias que permitam uma resposta rápida aos danos provocados por desastre. Elaborar um plano estratégico com tempo de processamento de vários anos não é o enfoque deste trabalho. Assim, será necessário reduzir o espaço de busca para que a estratégia seja eficiente e valida na proposta do RCR.

Para realizar essa redução, devemos inicialmente analisar as características mais importantes de cada tipo de tarefa possível e as limitações que os agentes possam ter para realizá-las. Devemos poder distinguir as tarefas em uma ordem de importância para que não seja perdido tempo analisando tarefas de menor valor e que possamos nos focar nas tarefas realmente importantes, que farão uma diferença significativa no escore final da simulação. Do ponto de vista dos agentes, muitas vezes eles estarão inaptos para realizar uma tarefa (podem estar bloqueados ou posicionados muito longe, ou mesmo não terem os recursos necessários para a realização da tarefa) e remove-los de possíveis coalizões torna o espaço de busca consideravelmente menor.

Ao buscar uma *CS*, será pesquisado como os agentes podem ser distribuídos entre as tarefas consideradas mais importantes, deixando as tarefas menos importantes para serem tratadas posteriormente ou por agentes fora da coalizão. Um agente que não está apto a realizar nenhuma tarefa considerada prioridade, será responsável por realizar uma tarefa escolhida por ele e não será considerado na *CS*. Isso diminuirá o espaço de busca e tornará certas coalizões inválidas (coalizões válidas são aquelas onde não há sobreposição de agentes e onde todos os agentes são capazes de realizar a tarefa determinada). Quando uma coalizão termina de realizar sua tarefa, ela será desfeita e os agentes realizarão as tarefas determinadas individualmente.

A seguir serão analisadas e descritas as características de cada tipo de tarefa e como cada uma delas influencia no resultado final da simulação. Com base nelas, podemos criar uma heurística para reduzir drasticamente o espaço de busca das tarefas e tornar o algoritmo mais eficiente.

5.1 Tarefa Um: Apagar Incêndios

Um prédio em chamas representa uma tarefa que deve ser realizado pela equipe de bombeiros do RCR. Ao apagar um incêndio, os bombeiros permitem que haja um melhor escore final, já que este é baseado na quantidade de área não queimada que há no cenário, e também podem impedir que esse foco de incêndio se propague para prédios adjacentes, impedindo assim que uma grande área do cenário seja queimada.

Para podermos avaliar a "importância" da tarefa (sendo essa importância uma forma de medirmos o valor da tarefa e decidirmos quais tarefas devem ter prioridades sobre outras), devemos levar em consideração questões como a influencia dela no escore final, a influência dela sobre as demais tarefas e a dificuldade em lidarmos com a tarefa especifica. A seguir será detalhado cada aspecto da tarefa e como usá-lo na hora de decidir qual tarefa executar primeiro.

5.1.1 Relevância da tarefa no escore final do simulador

Como explicado anteriormente, o escore final do simulador é baseado na quantidade de área preservada no final da simulação em relação à quantidade de área existente no inicio. Assim, os prédios do simulador não exercem a mesma influência no escore uns em relação

aos outros. Prédios com áreas maiores serão mais significativos que prédios com áreas menores. A tarefa de apagar incêndios em prédios grandes deve portanto ser priorizada em relação a apagar incêndios em prédios pequenos.

Essa área que o prédio ocupa será portanto uma importante variável no momento de selecionar qual tarefa é mais importante e para qual tarefa devemos enviar mais bombeiros. A relevância que uma tarefa tem sobre o escore final é aspecto mais importante a ser considerado. Lembrando que pela fórmula de escore (equação 4.1), um prédio perde importância a medida que ele queima, pois a área de um prédio queimado é apenas parcialmente contabilizada no escore.

5.1.2 Influência da tarefa no cenário

Um aspecto menos relevante é a influencia que uma determinada tarefa *pode* ter sobre as demais do cenário. Devido a um número consideravelmente grande de variáveis aleatórias (como velocidade e direção do vento, por exemplo), não podemos precisar ao certo como um prédio em chamas pode afetar os demais ao ser redor. Podemos imaginar que um prédio com diversos vizinhos deve ser prioridade em relação a um prédio que se encontre isolado e que o material de que o prédio é construído pode também influenciar nos demais, pois o fogo nesse prédio pode tornar sua condição critica mais rapidamente.

Temos, portanto dois aspectos a serem considerados aqui: o número de vizinhos e o material com que o prédio é construído. Considerar o número de vizinhos pode nos fornecer uma imagem errada da importância dessa vizinhança, pois pode haver um alto número de prédios vizinhos com uma pequena área cada um. Portanto, devemos considerar o tamanho da área dos prédios vizinhos e não o número de prédio. Quanto ao material, aquele que entrar em combustão mais rapidamente terá uma maior prioridade.

5.2 Tarefa Dois: Remoção de bloqueios

Diferentemente das demais tarefas (resgatar civis e apagar incêndios), a tarefa de remoção de bloqueios não tem influência direta no escore final do simulador. Apesar disso, ela é extremamente importante, pois permite que os demais agentes possam se locomover livremente pelo cenário e assim realizar suas respectivas tarefas. Dessa forma, uma estratégia

eficiente de remoção de bloqueios será fundamental para se obter um bom escore e para que os demais agentes realizem suas tarefas com sucesso.

As características fundamentais que serão levadas em consideração são aquelas relativas à melhora na movimentação geral dos agentes e à dificuldade de desbloquear uma determinada rua.

5.2.1 Importância da rua a ser desbloqueada

Em qualquer cidade existe um conjunto de vias que são fundamentais para o deslocamento dos moradores. Muitas vezes, podemos dividir o conjunto de ruas de uma cidade em quatro tipos: arteriais, principais, secundárias e locais. Em Porto Alegre, por exemplo, podemos considerar as ruas Protásio Alves e a avenida Ipiranga como vias artérias. Assim, estas são ruas que precisam estar livres de bloqueios e em bom estado de conservação, pois há um fluxo muito intenso de veículos passando por elas todos os dias.

Assim também no cenário da RCR existem vias que são mais importantes que outras e que devem estar desbloqueadas para permitir um fluxo eficiente de veículos e de agentes de campo. Para definirmos que ruas são essas, podemos utilizar como critério o tamanho da rua e o número de pistas que tem nelas. Estamos assim considerando que uma rua muito grande ou com muitas pistas é fundamental para o fluxo de veículos da cidade e deve ser priorizada em relação a outra rua que seja menor ou que permita menos veículos passando por ela de forma simultânea.

5.2.2 Quantidade de pistas livres

A prioridade ao escolher qual vias desbloquear não é apenas baseada no seu tamanho, mas também em quantas pistas livres ela já tem. Não necessitamos desbloquear uma rua até que ela esteja complemente livre, pois os carros podem passar por ela mesmo que ela tenha apenas uma pista livre. Analogamente com a rua de uma cidade onde houve um acidente, os carros continuam podendo se locomover pelas demais pistas, apesar de muitas vezes os carros terem que reduzir a velocidade e que o número de carros passando por um determinado local seja reduzido.

Nossa prioridade será desbloquear pelo menos uma pista da maior quantidade de ruas possíveis, permitindo assim que os agentes possam se movimentar para qualquer região do

cenário. A prioridade será possibilitar a movimentação dos agentes, mesmo que não seja pelo menor caminho possível.

5.2.3 Dificuldade de desbloquear uma rua

Uma rua que tenha muitos bloqueios pode exigir um tempo consideravelmente grande para desbloqueio. Por mais que a rua seja importante, pode exigir menos tempo desbloquear todas as demais ruas ao redor dela do que ela própria. Temos então que analisar o tamanho do bloqueio da rua e verificar se é uma boa estratégia priorizar ela ou se devemos desbloquear outras ruas. Como queremos liberar o mais rapidamente possível o acesso a todas as áreas da cidade (mesmo que esse acesso não seja o caminho mais curto entre dois pontos) será priorizado liberar o maior número de ruas possível em detrimento de liberar apenas as principais. O aspecto de quão difícil é desbloquear uma rua é uma das principais variáveis na escolha de qual rua desbloquear primeiro.

5.3 Tarefa Três: Resgate de Civis

Talvez a tarefa mais complexa do simulador RCR seja a de resgatar civis. Civis podem estar soterrados e quase morrendo, tornando o número de características a serem avaliadas na tarefa muito numerosas. Devemos considerar o estado do civil, o local onde se encontra, quão difícil é resgatá-lo e se devemos resgatá-lo (ele pode se encontrar tão ferido que pode não ser possível resgatá-lo antes que ele morra).

Além do maior número de características e propriedades dos civis em relação aos prédios e ruas do simulador, outro ponto importante a ressaltar é a grande influência que eles têm no escore final do simulador. Considerando que o escore é diretamente relacionado ao número de agentes e civis que estão vivos e ao seu "bem-estar" (valor do Hit Point (HP) dos agentes e civis ao final da simulação em relação ao HP deles no inicio da simulação), temos que garantir que um maior número de civis será socorrido e estará com vida no final da simulação. As propriedades mais relevantes ao selecionar uma tarefa de resgatar um civil são discutidas a seguir.

5.3.1 HP do civil e nível de soterramento dele

Um civil tem uma quantidade de vida medida pelo HP e um nível de soterramento (pode ser zero caso o civil não esteja soterrado). Ao selecionarmos um civil para socorrer, precisamos ter certeza de que nossos esforços não serão em vão. O civil deve ter um tempo de vida suficientemente grande para que possamos desenterrá-lo, ou seja, seu HP deve ser maior que seu índice de soterramento. A cada ciclo soterrado, o civil perde 200 pontos de HP e ao chegar a zero ele morre. Portanto, os agentes ambulâncias deverão dar prioridade àqueles civis que estão prestes a morrer mas ainda podem ser salvos. De nada adianta tentar salvar um civil que tem um alto nível de soterramento e um HP quase zero.

Para salvarmos o maior número possível de civis, devemos portanto analisar qual civil resgatar e enviar o maior número de ambulâncias para a região, pois o trabalho delas é somado e diminuí o tempo para o resgate. Essa combinação de ambulâncias permite salvar um civil mesmo que o seu HP seja menor que seu índice de soterramento, porém essa hipótese somente será válida se puder garantir que um número mínimo de ambulâncias serão alocadas a esse resgate; nem sempre será a melhor opção pois estará preferindo este civil aos demais que também podem precisar de socorro médico.

5.3.2 Posição do civil em relação à ambulância

Apesar de querermos agrupar um grande número de ambulâncias para resgatar cada civil, não fará sentido chamar uma ambulância muito distante para o resgate, pois ao chegar ao local determinado o civil pode já ter sido resgatado ou estar morto. Portanto, devemos alocar ambulâncias que possam rapidamente chegar ao civil e desconsideramos as demais ao estruturarmos a coalizão. Além disso, um civil pode estar localizado em um prédio em chamas e nesse caso não podemos salva-lo antes que o incêndio seja apagado. As ambulâncias também podem sofrer dano e até morrer caso estejam dentro de um prédio em chamas. Não devemos sacrificar uma ambulância pela possibilidade de salvarmos um civil.

5.4 Propriedades relevantes dos agentes

Além da redução do número de possíveis tarefas a serem escolhidas pelos agentes, podemos também nos focar em reduzir o espaço de busca para a formação de estruturas de coalizão, bastando analisar as propriedades de cada agente (bombeiros, policiais e

ambulâncias) e assim estabelecendo a possibilidade ou impossibilidade de um determinado agente participar de uma coalizão. Uma simples demonstração da importância dessas características seria considerar um agente que demore 10 ciclos para alcançar a tarefa que lhe foi determinada, mas a tarefa deixará de ter valor em cinco ciclos (por exemplo, um civil que morrerá em cinco ciclos não pode ser associado a um agente que esteja a uma distância que consuma mais de quatro ciclos para chegar até ele).

Como a maioria dos agentes possui a mesma capacidade de atacar um objetivo (isso quer dizer que entre dois agentes do mesmo tipo não há distinção no tempo gasto para executar uma tarefa), o ponto principal a ser analisado para que se possa diminuir o espaço de busca sem interferir na qualidade da estrutura de coalizão formada é a distância do agente até a tarefa indicada para a sua coalizão. Para o caso particular bombeiros, precisamos considerar que dois bombeiros possuem a mesma capacidade apenas se possuírem a mesma quantidades de água em seus tanques.

Para policiais, a distância irá influenciar apenas no caso de outro agente já ter realizado a sua tarefa, uma vez que o bloqueio de uma rua não irá se agravar ao longo do tempo. Isso porque se um agente precisa se movimentar até um local onde supostamente há um bloqueio, esse tempo de movimentação estará sendo perdido se ao chegar ao destino este local não estiver realmente bloqueado. Caso um policial esteja distante mais de três ciclos de uma determinada tarefa, ele não será nem considerado para participar da coalizão responsável por realizar a tarefa. O valor de três ciclos foi definido através de experiências e corresponde ao número médio de ciclos necessários para que se remova um bloqueio.

Para ambulâncias temos que considerar o tempo entre um agente e sua tarefa levando em consideração o fato do civil poder morrer antes que o agente chegue nele ou termine de desenterrá-lo. Caso o agente deva se movimentar até certo local onde se encontra o ferido e ainda desenterrá-lo, todo esse tempo deve ser levado em consideração para garantir que agente ambulância não irá desenterrar um civil já morto. Precisamos então garantir que uma ambulância não irá ser associada à uma tarefa se ela está localizada a uma distância que necessite mais de c ciclos para chegar a tarefa, onde c é o tempo de movimentação somado ao tempo restante de vida do civil, calculando a partir de seu HP e de seu soterramento (por exemplo, se o tempo de vida restante do civil for menor que quinze ciclos e o tempo de movimentação for de 6 ciclos, o tempo da tarefa de desenterrar o civil não pode ser superior a 8 ciclos).

Poderíamos considerar que a ambulância não estará trabalhando sozinha na tarefa de desenterrar um civil (que em muitos casos é exatamente o que acontece, havendo várias ambulâncias trabalhando em uma tarefa apenas), mas caso o valor do tempo de movimentação seja muito elevado ou o tempo de vida restante do civil muito baixo, não podemos garantir que ela poderá ajudar e estaríamos desperdiçando tempo de um agente que poderia estar salvando outras vidas. Assim, para um agente individual, temos um limite para a distância máxima que este pode estar do civil.

O agente do tipo bombeiro possui uma característica a mais que os demais agentes: sua capacidade para extinguir fogo depende do fato do agente ter ou não suficiente água em seu tanque. A água do agente pode acabar antes do fogo estar extinto e ele necessitará voltar à um refugio para reabastecer. Assim, além de determinar a distância do agente ao incêndio, é necessário determinar se ele pode ajudar de forma efetiva no combate do incêndio. Não seria eficiente movimentar o bombeiro por 10 ciclos para que ele jogue água por menos de três ciclos em um grande prédio em chamas.

Analogamente ao fator limitante do agente do tipo ambulância, o agente do tipo bombeiro também deve estar perto de seu alvo para que seja considerado na coalizão. Não podemos associar um prédio que estará completamente destruído em cinco ciclos a um bombeiro que demorará 10 ciclos para chegar até ele. Como em média um prédio demora cerca de cinco ciclos para perder mais da metade de seu valor no escore do simulador, limitaremos o raio de ação do bombeiro aos prédios que estiverem a menos de cinco ciclos de distância dele

Para tratarmos a questão da quantidade de água existente no tanque do bombeiro, devemos considerar o tempo que o agente deverá se movimentar em relação ao tempo que ele ficará executando a tarefa requerida. Se um bombeiro tiver que se movimentar por quatro ciclos para jogar água em um prédio por um tempo muito curto (como por exemplo, apenas dois ciclos), estaremos desperdiçando tempo valioso de um agente para executar uma tarefa que não trará resultados significativos. Portanto, quando um bombeiro estiver com menos de cinco ciclos de água no seu tanque (ou seja, poderá apenas jogar água em um prédio em chamas por cinco ciclos ou menos), ele deverá ser alocado ao incêndio mais próximo, a fim de não desperdiçar tempo com deslocamento. Caso haja mais de um alvo possível, escolheremos aquele com menor área, pois a pouca água que será jogada nesse prédio trará um beneficio maior do que a água atirada em um prédio muito grande.

5.5 Definição da heurística aplicada ao simulador RCR

Com base nas analises apresentadas acima, podemos estabelecer uma heurística para aplicarmos aos agentes e tarefas da RCR a fim de diminuirmos drasticamente o espaço de busca por estruturas de coalizão. Essa heurística tem como objetivo eliminar do espaço de busca as tarefas que apresentam uma importância muito menor que as demais e eliminar possíveis coalizões que não podem existir devido às restrições dos agentes.

A seguir será explicada a heurística utilizada para os agentes na seguinte ordem: primeiro será explicado a heurística utilizada pelos agentes bombeiros; em seguida, será explicada a heurística utilizada pelos agentes policiais e por último será explicada a heurística que será utilizada pelos agentes ambulância.

Caso seja necessário ao leitor, na página 4 deste trabalha há uma lista completa de abreviaturas e siglas utilizadas.

5.5.1 Heurística para estruturas de coalizão dos agentes do tipo bombeiros

Conforme descrito acima, o fator principal para definirmos a "importância" de uma tarefa para os bombeiros é o tamanho da área do prédio que está pegando fogo. Essa área será associada à sigla ***Ap*** de agora até o final do texto. Como essa área pode estar parcialmente destruída pelo fogo, temos que associar à esse fator um termo que indique qual a real influência do ***Ap*** no escore final. Esse termo será definido por ***Fp*** e indicará o quão destruído um prédio está. Os demais fatores são a área dos prédios ao redor da determinada tarefa (cuja sigla associada será ***Av***) e o material do qual o prédio é feito (***Mp***) .

Para reduzirmos o número de agentes possíveis para realizar uma determinada tarefa, temos um limite de distância entre o agente e a tarefa (definimos a distância de um agente para uma tarefa por ***Dt***) e o fator da impossibilidade do agente atingir a tarefa. Essa impossibilidade fará com que o agente esteja automaticamente fora de qualquer possível coalizão que tenha essa tarefa como objetivo.

Tendo em vista os fatores descritos acima, temos que a heurística utilizada para definirmos o valor de uma tarefa (Vp) é:

$$Vp = (Ap * Fp) + \sum_{v=0}^{n} Ap_v * Fp_v \quad (5.1)$$

Como podemos ver acima, a área dos prédios ao redor de uma determinada tarefa pode ter uma importância maior que a própria tarefa quando decidirmos qual tarefa é a mais importante dentre todas. Para atenuarmos esse valor, iremos dividir a área ao redor do prédio pela área do próprio prédio, para termos uma percepção de quão maior é essa área vizinha em relação à área do prédio. Essa informação será muito útil no caso de tarefas com áreas muito parecidas ou mesmo iguais. Temos assim o cálculo final do valor da tarefa como:

$$Vp = (Ap * Fp) + \frac{\sum_{i=0}^{n} Ap_i * Fp_i}{Ap} \quad (5.2)$$

Por fim, temos o fator ***Mp*** que não foi considerado na fórmula. Esse fator não pode ser diretamente acessado pelos agentes e é uma propriedade do prédio percebida apenas pelo simulador. Portanto, não podemos utilizá-lo. Caso essa informação se torne disponível em versões futuras do simulador, certamente deve ser incorporado na equação 5.2.

Para analisarmos a participação de um agente em uma determinada coalizão, temos que saber a distância desse agente até a tarefa. Essa distância é calculada pelo simulador como uma seqüência de nodos (todos objetos do simulador estão situados sobre um nodo e qualquer posição que um agente se encontre estará associado à um nodo específico) que o agente deve percorrer entre sua posição e a posição final. Um agente consegue percorrer 12 nodos por ciclo, o que nos indica que a distância do agente para a tarefa deve ser calculada como o total de nodos entre os dois dividido por doze. A recompensa de um agente para a realização de uma determinada tarefa será calculada em função de sua ***Dt***. Temos portanto que o cálculo do valor da recompensa de um agente associado à uma tarefa é:

$$Va = \left((Ap * Fp) + \frac{\sum_{i=0}^{n} Ap_i * Fp_i}{Ap} \right) * \frac{5}{(Dt * 2)} \quad (5.3)$$

Em outras palavras, a distância influência num fator de $5/2$. O número 5 vem do limite máximo de distância que um agente pode estar de sua tarefa e multiplicamos por dois para garantir um peso maior à sua distância, equilibrando o valor do prédio com a distância de um agente até ele.

5.5.2 Heurística para estruturas de coalizão dos agentes do tipo policial

Analogamente ao apresentado para os agentes do tipo bombeiro, iremos quantificar a importância das tarefas e após mostrar como é calculado o valor de um agente na coalizão responsável por executar a tarefa.

Como foi descrito anteriormente, a proposta é tentar liberar pelo menos uma pista do maior número de ruas possível ao invés de tentarmos retirar completamente os bloqueios de uma determinada rua. Assim, a principal informação é se há pistas desbloqueadas ou não. Caso haja, essa rua não deve estar na lista de ruas mais importantes pois ela permite a passagem de veículos, ainda que de forma limitada. Devemos analisar o número total de pistas de uma rua (***Pt***) para saber quão importante a rua é bem como número de pistas livres (***Pl***), o que nos dirá se essa rua deve ser priorizada ou não. Uma rua com muitas pistas e algumas delas já livres não terá prioridade em relação à uma rua com poucas pistas e nenhuma livre. Outro fator a ser considerado é o custo para desbloquear uma pista (***Cb***). Se duas ruas possuem as mesmas características em ***Pt*** e ***Pl***, aquela que tiver menor ***Cb*** terá preferência para ser desbloqueada. O fator ***Cb*** também é muito importante pois determina quão rapidamente uma rua pode ser desbloqueada, sendo assim fundamental para podermos desbloquear um grande número de ruas em um pequeno tempo.

Juntando todas as variáveis, montamos a fórmula para associarmos um valor para cada rua e decidirmos qual deve ser priorizada em relação à outra:

$$Vr = Pt - 2 * Pl - Cb + \frac{1}{Pt} \qquad (5.4)$$

Para avaliarmos a recompensa de cada agente dentro da coalizão, podemos seguir as mesmas idéias apresentadas anteriormente para o agente bombeiro. Representaremos também por ***Dt*** a distância correspondente do agente até sua tarefa e podemos avaliar sua participação como:

$$Va = \left(Pt - 2 * Pl - Cb + \frac{1}{Pt} \right) * \frac{3}{(Dt * 2)} \qquad (5.5)$$

A idéia é análoga àquela apresentada para o bombeiro, mas agora a tempo máximo que o agente usar para se deslocar até uma tarefa é três ciclos (sendo esse o tempo médio que um policial necessita para desbloquear uma rua).

5.5.3 Heurística para estruturas de coalizão dos agentes do tipo ambulância

Como mencionado anteriormente, a tarefa de resgatar civis é a mais complexa dentre as três tarefas do simulador. É preciso garantir que o maior número de civis seja salvo, porém não são conhecidas com precisão a localização deles e nem sempre é possível chegar até eles a tempo de salva-los. Ao associar uma tarefa de resgate à uma ambulância, não adianta fazer apenas metade da tarefa e parar depois, pois o civil continuará soterrado (mesmo que menos soterrado que anteriormente) e perdendo pontos de vida. Se a tarefa for iniciada, deve ser garantido que ela será terminada com sucesso.

Como todos civis têm a mesma importância no escore final, será usado como fator de escolha uma forma de maximizar o número final de civis vivos, mesmo que estejam com baixo HP. Portanto, será priorizado aqueles cuja expectativa de vida é muito baixa porém não nula. Não iremos tentar socorrer um civil que tenha *Hp* perto de 1000 e soterramento acima de cinco, pois a cada ciclo soterrado o civil perderá 200 pontos de vida e ao terminar de socorrer o civil, este já estará morto.

Para calcularmos a expectativa de vida de um civil usamos a seguinte fórmula:

$$Ev = Hp - St * 200 \qquad (5.6)$$

Onde ***Ev*** corresponde à expectativa de vida, *Hp* é a quantidade de vida (Hit Points) que tem o civil e ***St*** é o quão soterrado o civil está, lembrando que a cada unidade de soterramento corresponde a um ciclo de trabalho de uma ambulância.

Ao avaliarmos a ***Ev*** de cada civil, queremos achar aquela que seja a menor e que seja positiva, para não passarmos diversos ciclos desenterrando um civil morto. Ao adicionarmos nessa equação o agente ambulância, temos que a ***Ev*** do civil ainda depende da distância do

agente até o civil, pois essa distância são ciclos que o agente passa se locomovendo e não desenterrando o civil.

Assim, temos no final:

$$Ev = Hp - St * 200 - Dt * 200 \quad (5.7)$$

Como explicado, queremos o menor valor dessa equação que seja maior que zero, pois um resultado negativo será uma indicação de que o civil estará morto quando a ambulância terminar de desenterrá-lo

.

5.6 Uso das tarefas com maior prioridade

Após numerarmos as tarefas por ordem de prioridade, precisamos usar essa ordem para eliminarmos as tarefas menos importantes, ficando assim apenas com as tarefas que apresentam maior relevância para o simulador. A maneira como iremos diminuir o número de tarefas possíveis para os agentes é limitar o número máximo de tarefas a serem analisadas ao número de agentes que realizam a determinada tarefa. Isso significa que, caso haja dez bombeiros no cenário, apenas iremos considerar para análise as dez tarefas mais significativas. Como o número de agentes varia em tipo e mapa, não podemos decidir por um número fixo de tarefas, mas considerarmos que o número é baseado na quantidade de agentes de um determinado tipo nos permite trabalhar com um número reduzido de tarefas e distribuir as tarefas entre os recursos que temos. Também podemos considerar que, se há muitos agente de um tipo, é porque há muitas tarefas desse tipo a serem realizadas no cenário ou que as tarefas necessitam de muitos agentes para realizá-las. Não haveria sentido em criar um cenário com centenas de agentes bombeiro, por exemplo, se há apenas um foco de incêndio.

6 EXPERIMENTOS

Para avaliarmos os efeitos da heurística proposta nesse trabalho, utilizamos o próprio simulador Robocup Rescue, realizando experimentos em seus cenários e avaliando o resultado de cada um desses experimentos. Além disso, será mostrada uma comparação entre os resultados obtidos pelos experimentos com os resultados obtidos por diferentes estratégias aplicadas ao simulador RCR, além de uma análise sobre as conseqüências de tentarmos utilizar uma estratégia de coalizão de agentes sem reduzirmos o espaço de busca de soluções e sem utilizarmos quaisquer tipos de restrições quanto à formação das coalizões.

6.1 Mapas utilizados nos experimentos

Foram selecionados dois mapas próprios do simulador RCR para realizarmos os experimentos. O primeiro mapa é chamado de Kobe. Este mapa representa uma pequena parte da cidade de Kobe, no Japão. Optou-se pela escolha desse cenário por ser o cenário default do simulador e por seus resultados serem bem conhecidos.

Figura 6.1: Mapa Kobe utilizado nos experimentos.

Nesse mapa, contamos com um total de 24 agentes, sendo eles distribuídos entre 10 agentes bombeiros, 8 agentes policiais e 6 agentes ambulâncias, além de conter 72 civis. Há

também um total de 734 prédios e 820 ruas nesse cenário. O cenário está demonstrado figura 6.1.

O segundo mapa escolhido foi o chamado Kobe_4. Este mapa contém um pedaço diferente da cidade de Kobe e é muito maior que o primeiro. Apesar de ser muito maior, o número de agentes nesse mapa não aumenta proporcionalmente ao tamanho do mapa. A razão da escolha desse mapa é justamente o elevado de tarefas frente a um relativamente pequeno número de agentes, tornando assim a busca por uma estrutura de coalizão muito mais demorada que no primeiro caso, se o espaço de busca não for restrito.

Este mapa contém um total de 27 agentes, sendo 12 agentes bombeiros, 9 agentes policiais e 6 agentes ambulâncias e um total de 67 civis. O número de prédios nesse mapa é de 2.155 e o número de ruas é 2.277. Como podemos constatar, o mapa Kobe_4 possui aproximadamente três vezes mais objetos e possíveis tarefas que o mapa Kobe. A figura 6.2 mostra o mapa Kobe_4.

Figura 6.2: Mapa Kobe_4 utilizado nos experimentos.

6.2 Parâmetros para formação de estruturas de coalizão

Uma determinada coalizão apenas terá sentido enquanto a tarefa para a qual foi designada existir e for a tarefa mais importante a ser tratada. Caso a tarefa não precise mais ser realizada (por exemplo, um civil está salvo) ou ela não for mais tão importante quanto

antes (por exemplo, caso um prédio com maior área e prioridade comece a pegar fogo) será necessário reavaliar as coalizões para garantir que as tarefas estão devidamente alocadas. Em contra-partida, cada vez que uma estrutura de coalizão é formada é necessário um considerável esforço computacional e um gasto de recursos e tempo que pode ser prejudicial para a estratégia. Temos portanto que equilibrar a quantidade de vezes que o simulador irá dividir os agentes em grupos para que estes não se tornem nem muito obsoletos nem consumam recursos em demasia.

Como não é possível saber previamente qual o melhor momento para buscar uma nova estrutura de coalizão dos agentes, serão realizados experimentos com diferentes tipos de parâmetros que irão definir o momento que em que ocorrerá a busca por uma nova *CS*. Estes parâmetros serão:

- Diferença do número de ciclos em que ocorreu a última *CS* e o ciclo atual. Essa diferença será denominada de Δ;
- Quantidade de agentes que está realizando uma tarefa determinada pela *CS* em um determinado instante de tempo. Esse parâmetro será chamado de ρ.

Como já mencionado, uma certa tarefa tem validade por um certo tempo e devemos constantemente reavaliar se estamos realizando o melhor para o bem comum ou se devemos procurar outra função para realizar. Será necessário decidir quando reavaliar as coalizões existentes levando em consideração o esforço computacional para tal e a possibilidade de estarmos realizando tarefas já obsoletas por qualquer motivo. Não é possível definir ou justificar a escolha de um valor arbitrário para quando refazer a *CS* e portanto será apresentado experimentos contendo diferentes Δ.

Inicialmente será considerado um pequeno número de ciclos antes de reavaliarmos as coalizões e esse número será posteriormente aumentando. Portanto, será considerado que a cada cinco ciclos será reavaliada a estrutura de coalizão e serão formados novos grupos caso necessário. Após esses experimentos, serão realizados testes com Δ de 5, 10 e 15 ciclos. A intenção é avaliar qual será a diferença no escore final entre os parâmetros de Δ para que se analise se o custo-benefício de modificar os grupos em curtos intervalos de tempo é positivo ou negativo.

O segundo parâmetro que será analisado é o número de agentes que estão realizando uma tarefa que lhes foi atribuída pela *CS* (ρ). *CS* consideram apenas agentes de mesmo tipo (não haverá coalizões contendo bombeiros e policiais, por exemplo) e portanto esse parâmetro ρ será aplicado separadamente para cada tipo de agente. Ou seja, não será considerado o número total de agentes no cenário, mas sim a quantidade de agentes de cada tipo que estão em alguma coalizão em relação ao número total de agentes desse tipo. É possível permitir que os demais grupos terminem suas tarefas antes de se dividir novamente os agentes ou realizar imediatamente uma nova divisão. Serão realizados experimentos para determinarmos se é benéfico realizar a formação de coalizão constantemente ou se é necessário fazê-lo apenas quando uma razoável quantidade de agentes está fora da coalizão.

Será considerada inicialmente a idéia de que pelo menos metade dos agentes de um certo tipo precisam estar agrupados. Assim, quando houver um ρ maior que 50% (ou seja, mais de 50% dos agentes estiverem fora de uma coalizão), será preciso executar novamente o algoritmo para a formação de coalizão. Posteriormente, será testado o desempenho quando o valor de ρ corresponder à 30% e 70%. Esses experimentos também serão úteis para analisar a necessidade de um certo tipo de agente de se reagrupar em relação aos outros tipos agentes.

6.3 Resumo dos experimentos a serem realizados

Será portanto realizado experimentos com os mapas Kobe e Kobe_4. Em cada mapa, será simulado vinte vezes o algoritmo proposto com cada um dos seguintes parâmetros: Δ com valores de cinco, dez e quinze; e ρ com valores de 30%, 50% e 70%.

6.4 Critérios utilizados para avaliação dos resultados

Para a avaliação da estratégia utilizada e o sucesso da heurística para formar estruturas de coalizão entre os agentes, serão utilizados dois fatores: o primeiro será o escore resultante e o segundo será o comportamento que cada tipo de agente teve durante as simulações. O escore resultante será analisado comparando o resultado do algoritmo proposto com os algoritmos de LA-DCOP e Swarm-Gap além do algoritmo próprio do simulador RCR. No capítulo 6.5 há uma breve explicação sobre a funcionalidade desses algoritmos.

6.5 Algoritmos de alocação de tarefas utilizados nos experimentos

A seguir serão explicados os algoritmos utilizados para comparação do escore resultante.

6.5.1 LA-DCOP

Um *Distributed Constraint Optimization Problem* (DCOP) 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.

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. O LA-DCOP é diretamente aplicável em problemas de alocação de tarefas, 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).

6.5.2 SWARM-GAP

O Swarm-GAP (FERREIRA JR.; BOFFO; BAZZAN, 2008) é um algoritmo para alocação de tarefas baseado no LA-DCOP, tendo um comportamento bastante parecido com ele. Ele 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. 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. Esses estímulos permitem que cada tarefa seja escolhida com uma certa probabilidade. O Swarm-GAP também utiliza *tokens* para representar tarefas e repassa *tokens* de tarefas não realizadas para outros agentes.

6.5.3 Algoritmo guloso

O algoritmo guloso é implementado pelo próprio simulador. A estratégia utilizada por ele se resume a buscar a tarefa mais próxima do agente e indicar esta tarefa ao agente. Assim, independente das propriedades que a tarefa possui, cada agente irá realizar a tarefa mais próxima dele.

7. RESULTADOS E ANÁLISE

Este capítulo está disposto como segue: primeiramente será analisado como cada tipo de agente se comporta com uma estratégia baseada em divisão de agentes em estruturas de coalizão. Serão analisados os aspectos benéficos e maléficos dessa estratégia para cada tipo de agente e os principais pontos a serem considerados. Por último, será realizado um comparativo entre os resultados obtidos pela estratégia proposta e demais algoritmos de divisão de tarefas apresentados no capítulo 6.5.

7.1 Análise da influência da *CS* sobre cada tipo de agente

Cada um dos três diferentes tipos de agentes possui características próprias e tarefas específicas para realizar. Essa tarefas possuem propriedades muito diferentes umas das outras e portanto é necessário fazer uma análise individual de cada tipo de agente para investigar quão eficiente é o processo de formação de estruturas de coalizão. A seguir será demonstrada uma análise do comportamento apresentado por cada tipo de agente durante as simulações.

7.1.1 Agentes policiais

Os policiais são os que demonstraram obter menor benefício da estratégia proposta. Isso porque cada tarefa realizada pelos agentes necessita de muito pouco tempo para ser finalizada e não possui grandes melhorias quando realizada em conjunto com outros policiais. As tarefas apresentaram um baixo custo de tempo (geralmente o custo de remoção de bloqueios ficou abaixo de quatro ciclos) e o tempo necessário para deslocar mais de um policial para desbloquear uma rua se torna muito custoso. Foram poucos os casos apresentados onde os policiais de fato se agrupavam e, quando faziam, rapidamente terminavam a tarefa e prosseguiam de forma individual para realizar as demais tarefas.

Outro ponto a ser ressaltado é o grande número de tarefas possíveis no início da simulação e o quase inexistente número de tarefas existentes no final da simulação. Como os bloqueios das ruas não são criados após o início da simulação (ao iniciar a simulação todos os bloqueios já estão dispostos e não serão adicionados novos), temos um enorme número de tarefas desse tipo para analisar logo no principio e isso demanda um grande custo computacional que muito provavelmente não será aproveitado, pois os policiais estão dispersos e não podem trabalhar em conjunto. Na etapa final da simulação não há quase

tarefas e as poucas que temos podem ser realizadas por poucos policiais já que, em teoria, as ruas mais importantes já foram desbloqueadas anteriormente e esses bloqueios existentes são secundários.

É importante ressaltar que, apesar do pouco benefício extraído pelos agentes policiais da formação de estruturas de coalizão, o ponto crítico deles (onde há mais tarefas a serem realizadas) é no início da simulação, sendo o oposto dos demais agentes, pois não há muito conhecimento inicial sobre focos de incêndios ou civis feridos. Assim, o processamento pode ser todo dirigido para estes agentes e qualquer melhoria pode ser considerada válida, mesmo que custe tempo de processamento.

Figura 7.1: Mapa de Kobe após 209 ciclos utilizando apenas a estratégia gulosa e apenas com agentes do tipo policial.

A figura 7.1 mostra claramente que, mesmo utilizando uma estratégia gulosa como foi utilizada, é possível remover praticamente todos os bloqueios do mapa em cerca de dois terços do número máximo de ciclos.

7.1.2 Agentes Ambulância

As ambulâncias são os agentes mais complexos de agrupar devido ao enorme número de características que precisam ser consideradas. Por outro lado, eles são também os que

apresentam um menor número de tarefas e possibilidades de agrupamento. Um dos principais limitantes do trabalho desses agentes é o fato de não muitas vezes não ser possível localizar um civil por estar embaixo de escombros ou dentro de prédios. Portanto, os poucos civis localizados podem ter atenção de múltiplos agentes, facilitando a decisão sobre o processo de formação de *CS*. O grande benefício trazido pela escolha de qual civil cada agente irá socorrer certamente se deve ao método de escolha, que permite salvar o maior número de vidas possível e dividir os agentes priorizando aqueles civis que precisam de atenção médica mais imediata.

Portanto, a estratégia implementada para ambulâncias é benéfica por apresentar possui um baixo custo computacional (devido ao pequeno número de civis que são localizados a cada ciclo) e um alto retorno em termos de escore. Caso o número de civis fosse muito maior, haveria a necessidade de se optar por qual civil salvar e qual deixar morrer, mas ainda assim seria obtida uma distribuição melhor de agentes que no caso de cada agente escolher sozinho, não utilizando mais agentes que necessário para salvar cada civil.

7.1.3 Agentes Bombeiros

Os bombeiros são os que apresentam um maior equilíbrio entre uma estratégia que utiliza formação de estruturas de coalizão uma estratégia puramente individual. Quando há poucos focos de incêndios espalhados pelo cenário, o comportamento desses agentes de forma individual irá ser igual àquele apresentado quando utilizado uma estrutura de coalizão. A diferença entre esses comportamentos acontece quando há um grande número de focos de incêndio próximos uns dos outros e há muitos agentes juntos. Nesses casos, é importante que os agentes sejam separados em grupos pois caso contrário eles atuariam todos no mesmo incêndios uns dos outros. Portanto, a estrutura de coalizão teria o efeito de separar os agentes permitindo que um maior número de tarefas seja realizado simultaneamente.

Ao analisar o comportamento dos agentes bombeiros ao longo da simulação, é possível identificar que há pouca influência da *CS* no início da simulação pois os agentes estão dispersos (assim como as ambulâncias) e o número de focos de incêndios são poucos e bem espalhados. Porém, se o incêndio não é controlado logo no início da simulação, é possível identificar que áreas que possuem muitos prédios em chamas possuem agrupamento de bombeiros e esse agrupamento tende a apagar um determinado incêndio em um determinado prédio ao invés de se dividir em grupos e apagar o maior número de incêndios

possível. São esses os casos onde a divisão dos agentes em grupos é fundamental para a melhoria da distribuição de agentes entre as tarefas.

De maneira geral, podemos considerar que os benefícios oferecidos pela formação de estruturas de coalizão dos bombeiros superam os custos computacionais, pois apenas haverá um custo maior quando realmente existir a necessidade de divisão dos agentes em grupos. Há um baixo custo no início da simulação e um alto custo no final dela, caso os agentes não consigam apagar todos os incêndios no início.

7.2 Análise geral da estratégia e heurística

Inicialmente, os resultados quando utilizamos o fator tempo para reavaliarmos as coalizões são muito parecidos entre si. Podemos entender esse comportamento analisando as ações dos agentes quando não associados a uma coalizão especifica. Como explicado em 7.1, os agentes de maneira geral apresentam uma maior necessidade de formar *CS* quanto maior for o número de tarefas a serem realizadas. Como durante grande parte da simulação o número de tarefas é pequeno e estas são distantes umas das outras, não podemos notar grandes diferenças entre o comportamento que os agentes teriam sem utilizar *CS* e utilizando *CS*. Apenas quando o número de tarefas cresce e os agentes começam a se agrupar em uma região com muitas tarefas possíveis é que podemos notar grande diferença entre os comportamentos.

Em mapas como Kobe, apenas em poucas simulações o número de tarefas para bombeiro e ambulância foram elevadas pois os incêndios foram rapidamente controlados e os civis localizados salvos. Com os policiais, dificilmente se agruparam devido a inexistência de uma grande avenida ou rua que necessitasse ser desbloqueada e que necessitasse de muitos policiais para fazê-lo. Outra importante propriedade do cenário é o fato de suas tarefas, de maneira geral, serem rápidas de se realizar, não havendo uma grande necessidade de juntar muitos agentes para realizá-las.

Quando os experimentos foram realizados utilizando o parâmetro ρ para determinar quando seriam formadas as *CS*, os resultados dos escores melhoraram, porém o custo computacional tornou-se muito mais elevado. Isso porque os agentes conseguiam rapidamente realizar suas tarefas ou mesmo possuíam um número pequeno de tarefas para realizar e inacessíveis para alguns agentes, deixando assim muitos deles sem tarefa designada. Quando um agente não pode ser alocado à nenhuma tarefa, ele é considerado fora de uma coalizão e

isso tem efeitos não desejáveis quando possuímos agentes presos em ruas bloqueadas, pois eles estarão sempre fora de uma coalizão.

Durante a observação das simulações, foi possível constatar que os policiais constantemente precisavam formar novas *CS* devido ao fato das tarefas deles serem muito rápidas de realizar. Independente do valor de ρ, o número de vezes que foi pesquisada uma *CS* foi muito maior que quando foi usado qualquer parâmetro de tempo, pois era necessária uma nova *CS* a cada três ou quatro ciclos. Além disso, a estrutura de coalizão resultante era muitas vezes igual ao que se obteria se os policiais decidissem de forma individual qual tarefa realizar.

Para ambulâncias, o efeito foi o oposto. Devido ao alto número de ciclos necessários para realizar uma tarefa (desenterrar um civil pode levar até sessenta ciclos), uma vez que os agentes tenham sido designados para uma tarefa, demoraria muito tempo até eles serem designados para outra. Esse comportamento mostra uma clara vantagem entre o uso de ρ em relação à Δt, pois elimina a necessidade de buscar uma *CS* quando os agentes já possuem tarefas. Um ponto negativo a ser abordado é o fato do agente não poder realizar uma tarefa de maior importância que tenha sido descoberta após ele estar envolvido em outra tarefa, pois não será realizada uma nova divisão de agentes entre tarefas até que se obtenha um número ρ de agentes sem tarefa. Como ambulâncias possuem geralmente poucas tarefas para realizar, um ρ baixo permitiu que esse ponto negativo fosse mais bem administrado, porém ainda se demonstra um problema caso um grande número de ambulâncias esteja alocado para resolver tarefas.

Para os bombeiros, essa abordagem demonstrou-se positiva no aspecto de melhorar o escore obtido, mas também se demonstrou mais custosa que utilizar um parâmetro de tempo. Houve um aumento do número de estruturas de coalizão ao longo da simulação, tanto por causa da rapidez com que os incêndios são apagados (muitos deles demoram menos de cinco ciclos para serem completamente apagados) quanto pelo número considerável de agentes incapazes de fazer parte de uma coalizão (por estarem bloqueados na maioria dos casos).

7.3 Comparação entre escores

Foram executadas vinte simulações para cada tipo de algoritmo em cada cenário proposto. O algoritmo LA-DCOP foi experimentado com valor de treshold de 0.2 e o Swarm-Gap foi experimentado com estímulo de 0.1 (ambos os parâmetros foram escolhidos por

terem apresentado melhor resultado nos experimentos realizados em Ferreira et. al. 2010). Os resultados estão dispostos na tabela 7.3.

	LA-DCOP	Swarm-Gap	Guloso
Kobe	49.69 ± 6.31	44.97 ± 1.76	43.78 ± 7.19
Kobe_4	68.55 ± 2.66	64.91 ± 3.12	63.35 ± 3.30

Tabela 7.3: Resultado dos experimentos nos mapas Kobe e Kobe_4.

Os resultados dos experimentos utilizando o algoritmo com a heurística proposta também foram colhidos após 20 simulações e estão dispostos na tabela 7.4 e 7.5.

	ρ = 30%	ρ = 50%	ρ = 70%
Kobe	70.28 ± 6.94	67.63 ± 9.68	65.34 ± 7.22
Kobe_4	74.66 ± 5.02	73.62 ± 2.05	73.40 ± 2.05

Tabela 7.4: Resultados dos experimentos nos mapas Kobe e Kobe_4 para o algoritmo proposto.

É possível identificar que a diferença de resultados entre os mapas escolhidos foi muito menor quando utilizando o algoritmo proposto que quando utilizamos outros algoritmos. Isso deve ser analisado extensivamente pois pode indicar que o algoritmo tem dificuldade de lidar com ambientes onde o número de tarefas seja muito grande.

Quando em um cenário como Kobe, a rápida execução das tarefas permitiu que o fogo não se espalhasse muito e que não houvesse portanto uma grande região contendo muitas tarefas a serem realizadas. No mapa Kobe_4 não foi possível conter completamente os focos de incêndio nem socorrer um grande número de civis no início da simulação devido ao alto número de bloqueios. Assim, o fogo se espalhou e diversos civis se encontraram em estado crítico.

Apesar da aparente dificuldade em lidar com um grande número de tarefas, é possível identificar considerável ganho em escore entre o algoritmo proposto e os demais. No melhor caso (com ρ = 30%), foi obtido uma melhora de 40% em relação ao escore do algoritmo LA-DCOP e um ganho ainda maior em relação aos demais algoritmos no mapa Kobe.

8. CONCLUSÕES E TRABALHOS FUTUROS

A utilização de *CS* mostrou-se satisfatória na análise referente ao resultado final do escore do simulador. É possível identificar um ganho considerável em relação aos demais algoritmos de distribuição de tarefas comparados. O processo de busca pela *CS** pode ser completado em muitas ocasiões, apesar de ser mais demorado que os demais algoritmos quando o número de tarefas possíveis cresce muito.

É necessário continuar a análise desse algoritmo utilizando cenários que apresentem um maior número de agentes e de tarefas. Nos cenários propostos, o número de tarefas e de possíveis coalizões foi pequeno e em muitos casos não houve necessidade de buscar uma *CS* em um espaço muito grande de busca.

Como trabalhos futuros, serão testados diferentes parâmetros para indicar quando se deve buscar uma nova *CS*, parâmetros esses que estarão ligados ao número de tarefas existentes ou mesmo a combinação de diversos fatores. Essa proposta tem como objetivo identificar qual o melhor conjunto de parâmetros a ser utilizado (tanto de forma individual para cada tipo de agente como um parâmetro global a ser utilizado por todos os agentes). Será também utilizado cenários com maior número de tarefas e agentes, para que seja possível avaliar o algoritmo frente a diferentes tipos de estratégias, como eXtreme-Ants (SANTOS, 2009) e algoritmos que utilizam clusters, como descritos em (Santos, Bazzan, 2009) e avaliar a eficiência do algoritmo em situações com muitos agentes e tarefas.

Referências

DANG, V. D.; JENNINGS, N. R. Generating coalition structures with finite bound from the optimal guarantees. In: INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS, 3., 2004, New York, USA. **Proceedings.** IEEE Computer Society: Washington: USA, 2004. p.564–571.

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., PAULO R. AND FERNANDO DOS SANTOS AND ANA L. C. BAZZAN AND DANIEL EPSTEIN AND SAMUEL J. WASKOW, 2010**.** ***Robocup Rescue as Multiagent Task Allocation among Teams: experiments with task interdependencies***. aama. vol.20. 3. May. 421-443.

FERREIRA, JR., PAULO R. AND FERNANDO DOS SANTOS AND ANA L. C. BAZZAN AND DANIEL EPSTEIN AND SAMUEL J. WASKOW, 2010**.** ***Robocup Rescue as Multiagent Task Allocation among Teams: experiments with task interdependencies***. aama. vol.20. 3. May. 421-443.

HORLING, B.; LESSER, V. A survey of multi-agent organizational paradigms. **Knowledge Engineering Review**, [S.l.], v.19, n.4, p.281–316, 2005.

IEONG, S.; SHOHAM, Y. Marginal contribution nets: a compact representation scheme for coalitional games. In: ACM CONFERENCE ON ELECTRONIC COMMERCE, 6., 2005, New York, NY, USA. **Proceedings. . .** ACM Press, 2005. p.193–202.

JENNINGS, N. R. Coordination Techniques for Distributed Artificial Intelligence. In: O'HARE, G. M. P.; JENNINGS, N. R. (Ed.). **Foundations of Distributed Artificial Intelligence**. New York: John Wiley & Sons, 1996. p.187–210.

KITANO, H. RoboCup Rescue: a grand challenge for multi-agent systems. In: NTERNATIONAL CONFERENCE ON MULTIAGENT SYSTEMS, 4., 2000, Boston, USA. **Proceedings. . .** Los Alamitos: IEEE Computer Society, 2000. p.5–12.

LARSON, K. S.; SANDHOLM, T. W. Anytime coalition structure generation: an average case study. **Journal of Experimental & Theoretical Artificial Intelligence**, [S.l.], v.12, n.1, p.23–42, 2000.

LESSER, V. Cooperative Multiagent Systems: a personal view of the state of the art. **IEEE Transactions on Knowledge and Data Engineering**, Los Alamitos, v.11, n.1, p.133–142, 1999.

RAHWAN, T.; JENNINGS, N. R. An algorithm for distributing coalitional value calculations among cooperating agents. **Artificial Intelligence**, [S.l.], v.8, n.171, p.535–567 2007.

RAHWAN, T.; RAMCHURN, S. D.; DANG, V. D.; JENNINGS, N. R. Near-optimal anytime coalition structure generation. In: INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE, 20., 2007, Hyderabad, India. **Proceedings. . .** [S.l.: s.n.], 2007. p.2365–2371.

SANDHOLM, T.; LESSER, V. Coalitions among computationally bounded agents. **Artificial Intelligence**, [S.l.], v.1, n.94, p.99–137, 1997.

SANDHOLM, T.; LARSON, K.; ANDERSSON, M.; SHEHORY, O.; ; TOHMÉ, F. Coalition structure generation with worst case guarantees. **Artificial Intelligence**, [S.l.], v.1, n.111, p.209–238, 1999.

SANTOS, DANIELA SCHERER DOS AND BAZZAN, ANA L. C., 2009***. A biologically-inspired distributed clustering algorithm***. Proc. of the 2009 IEEE Swarm Intelligence Symposium. 160-167.

SANTOS, F.. *eXtreme-Ants: Algoritmo Inspirado em Formigas para Alocação de Tarefas em Extreme Teams*, 2009.

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.

SHAPLEY, L. S. A value for n-person games. In: KUHN, H. W.; TUCKER, A. W. (Ed.). **Contributions to the Theory of Games**. Princeton, NJ: Princeton University, 1953. n.28, p.307–317. (Annals of Mathematics Studies, v.2).

SHEHORY, O.; KRAUS, S. Methods for task allocation via agent coalition formation. **Artificial Intelligence**, [S.l.], v.101, n.1–2, p.165–200, 1998.

YEH, D. Y. A dynamic programming approach to the complete set partitioning problem. **BIT Numerical Mathematics**, Lawrence, KS, USA, v.26, n.4, p.467–474, 1986.