Uma Infraestrutura de Teamwork para Ambientes Dinâmicos com Requisitos de Tempo-Real

Ivan Medeiros Monteiro[1], **Luis Otavio Alvares**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{immonteiro,alvares}@inf.ufrgs.br

Abstract. *Although there are many tools for teamwork, the maintenance of the team coherence, in dynamic environment with real-time requirements, is still a challenge. In order to overcome the limitations of the previous tools, this paper introduces TWProxy, a new infrastructure for multi-agent coordination. The validation experiments show how the teamwork efficiency may change according to the coordination tool used.*

Resumo. *A manutenção da coerência de times, com agentes situados em ambientes dinâmicos com requisitos de tempo real, ainda representa um desafio para as ferramentas atuais de teamwork. Por isso, este trabalho apresenta o TWProxy, uma nova infraestrutura que permite a coordenação multiagente para tais ambientes, superando limitações apresentadas por outras ferramentas e introduzindo características importantes. Com resultados de experimentos expressivos, este trabalho evidencia as vantagens do TWProxy, além de demonstrar a influência que uma ferramenta de teamwork tem sobre a eficácia de um time.*

1. Introdução

Diversos domínios de aplicação operam com requisitos de tempo real. Em jogos de computador a demanda por comportamentos convincentes tem crescido, forçando a diminuição da distância entre o comportamento humano e o comportamento sintetizado [de Byl 2004]. Em jogos baseados em times a situação é ainda mais crítica, visto que freqüentemente os *bots*[1] não assumem papéis coerentes dentro do time. Assim, apesar dos avanços ocorridos na determinação do comportamento individual dos *bots*, pouco tem sido feito pelo comportamento social dos mesmos.

Um paradigma de time, conhecido em sistemas multiagentes como *teamwork*[2], aborda o problema do comportamento social com soluções para ambientes dinâmicos. Um *teamwork* flexível, promovido por um modelo explícito, é capaz de definir comprometimentos e responsabilidades para os membros do time, permitindo uma coordenação robusta e mantendo a coerência mesmo que haja falhas individuais ou mudanças imprevisíveis no ambiente [Schurr et al. 2005]. Por isso, neste trabalho o paradigma de *teamwork* é utilizado para suprir a demanda por comportamento social em jogos de computador. Neste contexto, um time é um grupo de entidades autônomas, pró-ativas e com capacidades de reflexão sobre as próprias habilidades e as habilidades do grupo, que compartilham um objetivo comum.

[1]Um personagem controlado por um agente artificial.

[2]Esforço cooperativo realizado por membros de uma equipe para alcançar um objetivo comum.

Teamwork tem emergido como o paradigma para coordenação de agentes de forma cooperativa em ambientes dinâmicos e tem se mostrado capaz de levar a um comportamento robusto e flexível. Entretanto, não existe uma solução geral para construção de times de agentes e cada novo domínio pode ter novos requisitos. Uma classe de aplicação que se mantém como desafio para a área de *teamwork* é aquela na qual o ambiente é altamente dinâmico e que requer um tempo de resposta muito curto.

Em ambientes dinâmicos, o estado do ambiente pode sofrer alteração entre o agente perceber tal ambiente e atuar sobre ele. Ou seja, o tempo que o agente demora para agir pode afetar a qualidade de suas ações. Isso se torna mais grave quando o ambiente exige uma reatividade muito alta, com intensas atualizações de seu estado. Em domínios altamente dinâmicos a manutenção da coerência de time é também dificultada, dado que a mudança de postura do time precisa acompanhar as mudanças de estado do ambiente. Nesses ambientes frequentemente é necessário sacrificar alguns aspectos gerais, como comunicação assíncrona ou decisões distribuídas, em favor do tempo de resposta.

A fim de superar essas dificuldades, este trabalho introduz o *TWProxy*, uma nova ferramenta que habilita agentes individuais a integrarem times e agir de forma coerente em ambientes altamente dinâmicos com requisitos de tempo-real[3]. Ele utiliza algumas das boas idéias apresentadas pelo *Machinetta*[Scerri et al. 2003], contornando algumas de suas limitações e adicionando novas características.

O restante deste trabalho está organizado da seguinte forma: a Seção 2 apresenta os trabalhos de *teamwork* relacionados, a Seção 3 detalha o funcionamento do *TWProxy*, a Seção 4 relata os experimentos e avaliação da ferramenta aqui apresentada e na Seção 5 são feitas as considerações finais deste trabalho.

2. Trabalhos Relacionados

Modelos teóricos de *teamwork* como *Joint Intentions* [Levesque et al. 1990] e *SharedPlans* [Grosz and Kraus 1996] inspiraram o surgimento de diversas ferramentas que auxiliam na manutenção da coerência de time, tais como: *COLLAGEN* [Rich and Sidner 1997], *STEAM* [Tambe 1997], *RETSINA* [Sycara et al. 2001], *Machinetta* [Scerri et al. 2003]. Entretanto, nenhuma dessas ferramentas representa uma solução geral de *teamwork*, devido às particularidades de cada novo domínio.

O *Machinetta* surgiu como uma evolução do modelo de *teamwork* do *STEAM*, utilizando uma abordagem baseada em *proxy*, que visa habilitar agentes não comprometidos socialmente ao comportamento social. Assim como o *STEAM*, ele também demonstrou sucesso em diversos domínios e por isso é utilizado neste trabalho de forma comparativa com a ferramenta aqui proposta.

Devido às limitações das ferramentas existentes em fornecer suporte de *teamwork* para ambientes altamente dinâmicos, com requisitos de tempo-real, este trabalho apresenta o *TWProxy*.

3. TWProxy

TWProxy é baseado no formalismo de *Joint Intentions* e inspirado no *Machinetta*, adicionando novas características importantes como: (i) uma linguagem simples e extensível para descrição de crenças e planos, (ii) dois processos eficientes de alocação de tarefas parcialmente distribuídos, (iii) manutenção de consistência através de comunicação atômica, (iv) reuso de planos terminados, (v) planos invariantes ao número de agentes.

[3]Considera-se aceitável neste trabalho um tempo de resposta dentro de uma janela de 200ms

Apesar de existirem soluções de *teamwork* para ambientes dinâmicos, não existia até agora uma preocupação evidente com o tempo de resposta no processo de mudança de atividades dentro do time, a ponto de atender requisitos de tempo real. Essa característica é muito importante quando o ambiente é altamente dinâmico e a qualidade das ações do time pode cair muito com o atraso na mudança de uma estratégia. Por isso, o *TWProxy* mantém o foco na diminuição do tempo de resposta, sem perder a qualidade do gerenciamento das ações de time.

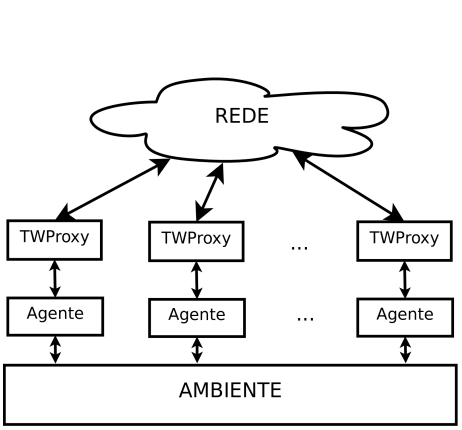
Na abordagem apresentada neste trabalho, cada agente é associado com um *proxy*, como visto na Figura 1(a). O *proxy* fornece ao agente uma interface de time, permitindo que este participe de uma equipe sem se preocupar com os detalhes de coordenação. Tudo que o agente precisa para compor um time é agir conforme as sugestões do *proxy* e informar a seu *proxy* sobre a atualização de determinadas crenças. Assim, os *proxies* ficam responsáveis por manter a coerência de time através da comunicação *inter-proxy* e pelo planejamento realizado pelo grupo de *proxies*. O acoplamento simbiôntico do agente individual com o seu *proxy* é o que forma o agente social. Toda a comunicação e planejamento de time são feitos através dos *proxies*, sem necessidade de consciência do agente individual.

A forma utilizada pelo *TWProxy* para alcançar a coordenação multiagente é através da alocação de papéis, que pode ser vista também como a alocação de tarefas ou atividades de time. Essa maneira de coordenar os agentes é flexível o suficiente para permitir que agentes heterogêneos componham a equipe, já que o *proxy* não diz como o agente deve executar as atividades. O conhecimento de como executar a tarefa fica por conta do próprio agente e este deve ter consciência do que ele está capacitado para fazer. Assim, quando ele informar ao *TWProxy* sobre suas capacidades estará habilitando o *proxy* a realizar uma alocação adequada. Como os *proxies* comunicam-se entre si, cada *proxy* conhece as habilidades dos demais membros da equipe.

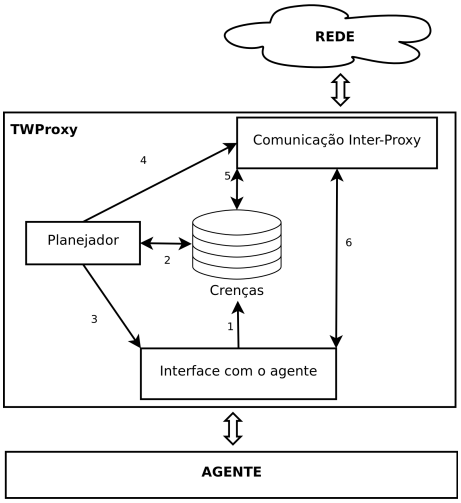
O processo de alocação de papéis é disparado por planos, que são compostos por pré-condições, pós-condições e papéis a serem alocados. Quando as pré-condições de um plano são alcançadas, inicia-se o processo de alocação dos papéis especificados pelo plano. Quando uma pós-condição é alcançada, os membros da equipe são liberados dos papéis associados ao plano. Devido a esta coordenação em alto nível, é possível alcançar um *teamwork* flexível, deixando os detalhes de como executar o papel para o agente individual.

A Figura 1(b) mostra a organização interna do *TWProxy*, composta pelo seu planejador, sua base de conhecimento, um módulo de comunicação com o agente, um módulo de comunicação inter-*proxy* e a interação entre esses elementos. Os módulos de comunição são facilmente extensíveis para atender a novos requisitos de organização, e também para possibilitar a integração do *TWProxy* com novos agentes sem a necessidade de alterar o planejador ou a base de crenças. O planejador e a base de conhecimento são dirigidos pela linguagem de definição de planos e crenças apresentados na subseção 3.1. As interações internas do *TWProxy*, representadas por setas na Figura 1(b), são: **1 -** Chegada de uma nova crença ou a atualização de alguma crença existente que foi percebida pelo agente e que irá atualizar a base de conhecimento; **2 -** Uso da base de conhecimento pelo planejador, que faz uso das crenças armazenadas na base para verificar as condições de ativação e término dos planos; **3 -** Deliberação sobre a alocação de um determinado papel para o agente associado ao *proxy*; **4 -** Deliberação sobre a alocação de um determinado

papel para um outro membro do time; **5 -** Sincronização de crenças com outro *proxy*; **6 -** Canal que permite a comunicação entre agentes utilizando o *TWProxy* apenas como um meio.



(a) Modelo de *teamwork* baseado em *proxy*.



(b) Organização.

Figura 1. *TWProxy*.

3.1. Descrição de planos e crenças

No *TWProxy*, a base de crenças armazena igualmente as crenças iniciais e aquelas criadas em tempo de execução. As crenças iniciais são carregadas a partir de um arquivo estruturado que define um conjunto de crenças básicas para o funcionamento do *proxy*. As crenças geradas em tempo de execução são criadas pelos agentes e podem representar, além de novas crenças, atualizações das crenças iniciais.

Esta seção trata das crenças iniciais, juntamente com os planos de time, que são descritos por um arquivo estruturado e carregado na inicialização do *proxy*. Este arquivo contém crenças sobre as capacidades do agente, a composição do time e o conhecimento do domínio específico da aplicação. Tal arquivo também armazena os planos de time em alto nível, formando um programa de time[Scerri et al. 2003]. Cada crença é composta por um identificador precedido de "belief"e seguido por um conjunto de pares atributo-valor delimitado por chaves. De forma semelhante, cada plano possui um identificador precedido de "plan"e seguido por: um conjunto de papéis a serem alocados, expressões de pré-condições para ativação e expressões de pós-condições para finalização do plano.

A listagem 1, apresenta um trecho do arquivo de crenças e planos que descreve uma crença sobre a capacidade do agente defender a sua base, duas crenças de domínio e um plano para a captura da bandeira inimiga. A crença *capability_agent001_CTFProtectTheBase* descreve a capacidade do agente *agent001* executar o papel *CTFProtectTheBase* através de um nível de habilidade, que neste caso é 76. As crenças de domínio *flag_enemy* e *flag_friend* descrevem o estado das bandeiras presente no ambiente. Já o plano *p1* define um conjunto de papéis que devem ser executados *(CTFProtectTheBase, CTFCaptureTheFlag)* quando a pré-condição de *p1*, formada por crenças de domínio, for alcançada. A satisfação da pós-condição marca o término do plano, liberando os agentes dos papéis envolvidos. O plano permite ainda a aplicação de

modificador de flexibilidade para um determinado papel, expresso pelo símbolo + posfixado ao nome do papel. Isto permite que um determinado papel seja executado por um ou mais agentes, enquanto os papéis especificados sem modificador são executados exclusivamente por um único agente.

Listing 1. Trecho do arquivo de crenças e planos utilizado pelo proxy.

```
belief capability_agent001_CTFProtectTheBase {
        type: capability;
        rapId: agent001;
        roleId: CTFProtectTheBase;
        ability: 76;
}
belief flag_enemy{
        type: flag;
        owner: enemy;
        have: false;
}
belief flag_friend{
        type: flag;
        owner: friend;
        have: true;
}
plan p1{
        roles: CTFProtectTheBase, CTFCaptureTheFlag+;
        precondition  { ( flag_friend.have == true  ) & ( flag_enemy.have == false ) }
        postcondition { ( flag_friend.have == false ) | ( flag_enemy.have == true  ) }
}
```

3.2. Comunicação

O módulo de comunicação inter-*proxy* usa dois tipos de comunicação, a difusão atômica [Defago et al. 2004] e a comunicação direta (*unicast*). Cada tipo de comunicação é separada em diferentes canais. A difusão atômica é usada para compartilhar crenças do time e a comunicação direta é usada no processo de alocação de papéis. A difusão atômica é utilizada quando uma nova crença precisa ser compartilhada com a equipe. Em outras palavras, quando um membro do time percebe uma nova situação que é relevante para a equipe, ele compartilha essa informação mantendo o conhecimento do time consistente. Entretanto, os membros do time não precisam compartilhar seus conhecimentos individuais, diminuindo a quantidade de mensagens trocadas.

O canal de *unicast* é usado para executar uma variação do Contract Net Protocol[Smith 1981], a fim de realizar a alocação de papéis. Neste processo, o líder utiliza comunicação direta para requisitar aos membros que atualizem as crenças do grupo sobre suas capacidades; uma vez atualizadas essas crenças, ele decide sobre a atribuição de papéis, enviando, via *unicast*, o papel que cada membro deve executar.

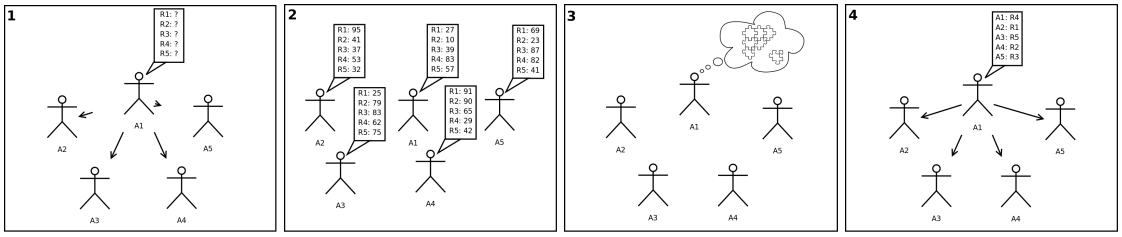
3.3. Coordenação

A coordenação sobre o paradigma de *teamwork*, fornecida aos agentes pelo modelo do *TWProxy*, é resultado do processo de alocação de papéis executado pelo conjunto de *proxies*. Este processo é feito de forma parcialmente distribuída, utilizando um líder do time para decidir sobre a alocação. O principal benefício por utilizar um processo de alocação de papéis apenas parcialmente distribuído, centralizando no momento da decisão da alocação, é que existem algoritmos ótimos e quase ótimos com tempo polimonial para resolver a alocação de forma centralizada, possibilitando a economia de tempo no processo de decisão comparando com os processos totalmente distribuídos, apesar de ainda se gastar tempo com comunicação para a execução do processo.

O planejador do *TWProxy* utiliza o líder do time para lançar um novo plano. Este líder pode ser prefixado estaticamente ou definido dinamicamente em tempo de execução. O uso de líder de time evita a necessidade de resolver problemas de conflitos na alocação de papéis, economizando tempo para reagir às mudanças do ambiente. O líder do time

tem as crenças atualizadas e pode lançar de forma consistente um plano de time, porque todos estão comprometidos a compartilhar informações relevantes para o time. A escolha do líder pode ser feita de duas formas. A primeira, mais simples, é uma escolha estática, onde um líder é definido no momento da inicialização do *TWProxy*, com base em seu arquivo de configuração. A segunda forma é a escolha dinâmica, onde o agente que percebe uma modificação de crença relevante ao time será o líder naquele momento.

A Figura 3.3 apresenta, em quatro passos, o processo de alocação de papéis utilizado pelo *TWProxy*, considerando que um líder já foi escolhido. No primeiro passo, apresentado no quadro um, o líder (agente A1) requisita aos demais membros do grupo que atualizem as informações sobre suas capacidades no quadro-negro distribuído. No segundo passo, cada membro atualiza as crenças do time sobre suas capacidades por meio de difusão atômica. Em seguida, o líder executa localmente um algoritmo de alocação de tarefas e então atribui, no quarto passo, os papéis para os demais membros.



(a) Passo 1 (b) Passo 2 (c) Passo 3 (d) Passo 4

Figura 2. Processo de alocação de papéis.

Para o alocação de papéis, o *TWProxy* utiliza por padrão o algoritmo de Kuhn-Munkres [Kuhn 1955, Munkres 1957], que é ótimo e possui a complexidade $O(n^3)$. Para um configuração com centenas de agentes, também é possível utilizar no *TWProxy* um algoritmo heurístico de complexidade $O(n^2 log(n^2))$, que produz resultados bem próximos ao ótimo com o crescimento do número de agentes [Monteiro 2009].

3.4. Interface com o agente

Cada agente individual é anexado ao *TWProxy* formando assim um agente social. A comunicação entre o agente individual e o *TWProxy* é definida por uma interface flexível, permitindo interação com vários tipos de agentes. Para a utilização do *TWProxy* em um novo domínio são necessárias as implementações da interface com o agente e da descrição do programa de time. Assim, é possível criar um time para um novo domínio com muito pouco esforço. O agente não precisa ter um comprometimento social além do compartilhamento de suas crenças, porque o *TWProxy* faz isso por ele. Desta maneira, o desenvolvimento do agente também fica bastante simplificado.

4. Experimentos e Avaliações

Dois tipos de experimentos foram realizados a fim de avaliar o *TWProxy*. O primeiro visa avaliar a performance, medindo o tempo de resposta da ferramenta. O segundo tem o objetivo de avaliar a eficiência, juntamente com a manutenção da coerência de time. Para isso foram realizadas disputas de *Capturar a Bandeira* entre diversos times utilizando o jogo Unreal Tournament 2004.

Na modalidade de jogo *Capturar a Bandeira*, um time deve obter pontuação capturando a bandeira inimiga na base do oponente e levando-a para a base aliada. Se o

personagem que carrega a bandeira é morto, a bandeira cai ao chão e qualquer um pode pegá-la. Se a bandeira aliada é roubada, o time deve resgatá-la antes que a equipe adversária obtenha pontuação com a bandeira capturada.

4.1. Avaliação do período de inconsistência de time

O experimento sobre o período de inconsistência de time tem o objetivo de avaliar a habilidade do *TWProxy* de reagir em tempo-real às mudanças do ambiente. A fim de comparar essa performance, o *Machinetta* foi exposto à mesma situação. Um grupo de agentes enviava atualizações de crenças e o tempo de reação das ferramentas era medido, isto é, o período entre a informação da mudança no ambiente e a adoção de uma nova estratégia pelo time. Este tempo transcorrido entre uma mudança no ambiente e a finalização do processo de alocação de papéis é definido aqui como período de inconsistência de time.

Tanto o *TWProxy* como o *Machinetta* foram expostos à mesma seqüência de mudanças de estado do ambiente. Esta seqüência foi composta por 25 atualizações de ambiente com relevância para o time, representando uma simulação do ambiente de *Capturar a Bandeira*. Cada *proxy* executou 100 vezes essa sequência com a finalidade de obter um comportamento geral do período de inconsistência de time. A latência de rede não está sendo considerada neste experimento.

O resultado apresentado na Tabela 1 descreve a média (μ) e o desvio padrão (σ) do conjunto de execuções. A unidade de medida é milisegundos e o tempo representa o período transcorrido entre uma nova crença ser recebida e a modificação da estratégia. Nos experimentos o *TWProxy* alcançou a média de $77.91ms$, enquanto o *Machinetta* obteve $12303.34ms$. Em outras palavras, o *TWProxy* foi, em média, mais de cem vezes mais rápido do que o *Machinetta* para alcançar o estado consistente. O desvio padrão também expressa o quão estável foi a execução do *TWProxy* comparada com a do *Machinetta*.

Tabela 1. Média e desvio padrão do Período de Inconsistência de Time.

	Tempo transcorrido(ms)	
	μ	σ
TWProxy	77,91	12,95
Machinetta	12303,34	6459,54

4.2. Eficiência de teamwork

Neste segundo tipo de experimento é avaliada a eficiência dos times de agentes em um situação real, aplicada ao jogo de computador *Unreal Tournament 2004 (UT2004)*, um jogo multijogador de primeira pessoa (*First-Person Shooter*). A modalidade de jogo é a de *Capturar a Bandeira* e a forma de interação com o jogo é através da interface de *GameBOTs* [Kaminka et al. 2002]. Para o desenvolvimento dos agentes foi utilizado o framework *IAF* [Monteiro and dos Santos 2007], que implementa uma arquitetura de agente híbrida [Monteiro 2005].

O ambiente fornecido pelo jogo é bastante complexo, sendo classificado como **parcialmente observável**, **estocástico**, **não episódico**, **dinâmico**, **contínuo** e **multiagente**, o que aumenta o desafio da manutenção de coerência de time.

Para estes experimentos, tanto o time que utiliza o *TWProxy* (*TWProxy-Team*) como o time que utiliza o *Machinetta* (*Machinetta-Team*), possuem a mesma implementação dos agentes e consequentemente as mesmas capacidades. Assim, apenas o trabalho em equipe faz a diferença na disputa entre eles.

Para a avaliação da eficência de *teamwork* conseguida com o *TWProxy*, foram executados diversos experimentos de combates entre times distintos. Os times que se enfrentaram foram: *TWProxy-Team* vs *UTBots-Average* (time de bots do próprio jogo UT2004 no nível de habilidade *Average*), *TWProxy-Team* vs *UTBots-Experienced* (time de bots do próprio jogo UT2004 no nível de habilidade *Experienced*), *Machinetta-Team* vs *UTBots-Average*, *TWProxy-Team* vs *Machinetta-Team* e *TWProxy-Team* vs *Human-Team* (time de humanos voluntários jogadores de UT2004). Cada combate tinha a duração de quinze minutos e foram executados trinta combates para cada combinação de time que se enfrentava. Nesses combates, cada time possuía cinco jogadores, com excessão dos combates entre *TWProxy-Team* e *Human-Team* quando foram utilizados apenas quatro jogadores. O resultado dos combates está resumido na Tabela 2.

Nos combates entre *TWProxy-Team* e *UTBots-Average*, apesar do uso de controle centralizado e acesso a estados ocultos de ambiente pelo *UTBots-Average*, o *TWProxy-Team* ganhou $93,33\%$ das partidas e empatou apenas $6,67\%$, mantendo um média de $2,42$ bandeiras capturadas por jogo. O desempenho dos agentes individuais, representado na Tabela 2 pela soma das pontuações individuais, também foi bem superior ao desempenho dos bots do próprio jogo no nível padrão de dificuldade.

Tabela 2. Resultado das partidas.

	TWProxy-Team		**UTBots-Average**	
Percentual de vitórias	$93,33\%$		0%	
Pontuação de Time	$\mu = 2,42$	$\sigma = 1,45$	$\mu = 0,03$	$\sigma = 0,18$
Soma das Pontuações Individuais	$\mu = 230,97$	$\sigma = 31,11$	$\mu = 100,77$	$\sigma = 20,27$
	TWProxy-Team		**UTBots-Experienced**	
Percentual de vitórias	$76,67\%$		0%	
Pontuação de Time	$\mu = 1,37$	$\sigma = 1,24$	$\mu = 0$	$\sigma = 0$
Soma das Pontuações Individuais	$\mu = 227,03$	$\sigma = 27,32$	$\mu = 117,87$	$\sigma = 27,72$
	Machinetta-Team		**UTBots-Average**	
Percentual de vitórias	20%		0%	
Pontuação de Time	$\mu = 0,27$	$\sigma = 0,64$	$\mu = 0,03$	$\sigma = 0,18$
Soma das Pontuações Individuais	$\mu = 185,47$	$\sigma = 48,28$	$\mu = 73,87$	$\sigma = 31,84$
	TWProxy-Team		**Machinetta-Team**	
Percentual de vitórias	$43,33\%$		0%	
Pontuação de Time	$\mu = 0,6$	$\sigma = 0,81$	$\mu = 0$	$\sigma = 0$
Soma das Pontuações Individuais	$\mu = 217,27$	$\sigma = 57,15$	$\mu = 176,07$	$\sigma = 29,61$
	TWProxy-Team		**Human-Team**	
Percentual de vitórias	20%		$26,67\%$	
Pontuação de Time	$\mu = 0,23$	$\sigma = 0,43$	$\mu = 0,5$	$\sigma = 0,97$
Soma das Pontuações Individuais	$\mu = 108,73$	$\sigma = 25,02$	$\mu = 154,2$	$\sigma = 39,23$

Os resultados das partidas entre *TWProxy-Team* e *UTBots-Experienced* também são apresentados na Tabela 2, onde ainda é possível perceber uma grande diferença no desempenho individual dos jogadores. Neste experimento o *UTBots-Experienced* não conseguiu capturar nenhuma bandeira do *TWProxy-Team*, mesmo com aumento do nível de dificuldade do jogo. No resultado geral deste experimento, o *TWProxy-Team* venceu $76,67\%$ e empatou $23,33\%$ das partidas, com uma média de $1,37$ bandeiras capturadas por jogo.

No experimento em que o *Machinetta-Team* enfrentou o *UTBots-Average*, o *Machinetta-Team* teve um desempenho melhor que o *UTBots-Average*, com 20% das vitórias e 80% de empates, como visto na Tabela 2. Entretanto, numa comparação indireta com o *TWProxy-Team*, ele possui um desempenho bastante baixo, com uma média

de captura de bandeiras de $0,27$ por jogo. Também houve decréscimo no desempenho individual, apesar dos agentes serem os mesmos utilizados pelo *TWProxy-Team*.

No experimento em que o *TWProxy-Team* confronta o *Machinetta-Team*, o *TWProxy-Team* teve uma média de $0,6$ capturas por jogo, enquanto o *Machinetta-Team* não conseguiu pontuar, como pode ser visto na Tabela 2. No resultado geral, $43,33\%$ das partidas resultaram em vitória do *TWProxy-Team* enquanto $56,67\%$ das partidas representaram empate entre as duas equipes. Com base nas comparações entre os times *TWProxy-Team* e *Machinetta-Team* , é possível identificar que o *TWProxy* apresenta vantagens claras sobre o *Machinetta* para o domínio utilizado, que representa um ambiente dinâmico com requisitos de tempo-real[4].

Também foram realizados jogos com jogadores humanos, onde a principal motivação era identificar as possíveis incoerências a que o time poderia estar se submetendo e que jogadores humanos experientes poderiam explorar com maior facilidade. Tais inconsistências de time não foram encontradas e os resultados das partidas foram bastante próximos, como pode ser visto na Tabela 2. Dos trinta jogos disputados, o *TWProxy-Team* ganhou 20%, perdeu $26,67\%$ e empatou $53,33\%$. A pontuação individual mostra que os jogadores humanos atuaram em média melhor que os jogadores do *TWProxy-Team*, mas ainda assim os jogos foram bastante equilibrados.

5. Considerações Finais

Teamwork tem se tornado um paradigma padrão para a coordenação de agentes de forma cooperativa, mostrando-se robusto e flexível para domínios complexos. Com o desenvolvimento de ferramentas de suporte a *teamwork*, tornou-se possível a reutilização do modelo de *teamwork* através de diferentes domínios. Entretanto, nenhuma das ferramentas pré-existentes fornece uma solução satisfatória para domínios que possuem ambientes altamente dinâmicos com requisitos de tempo real.

O *TWProxy*, apresentado neste trabalho, viabiliza o uso do paradigma de *teamwork* em domínios com requisitos de tempo real. Ele introduz novas características importantes para o desenvolvimento de time em ambientes complexos, tais como: eficiência do processo decisório, com soluções ótimas ou próximas da ótima em tempo real; reativação de planos terminados, evitando a criação de planos redundantes; simplificação do projeto de time com planos flexíveis e linguagem de descrição intuitiva; e facilidade na reutilização do *proxy* para novos domínios.

O *TWProxy* teve bons resultados, sendo caracterizado de maneira geral como estável e eficiente. Na avaliação do período de inconsistência de time, o *TWProxy* é comparado com o *Machinetta* em um ambiente simulado, onde o *TWProxy* obtém períodos de inconsistência muito menores que o *Machinetta*. Quanto à eficiência, avaliada sobre uma aplicação real, foi possível constatar que o *TWProxy* consegue manter a coerência do time mesmo em um ambiente altamente dinâmico, competindo com abordagens centralizadas e contra times de humanos. As comparações com o *Machinetta* também foram importantes para demonstrar a variação de eficiência que um time pode sofrer devido a mudança da sua ferramenta de coordenação.

Os próximos passos são: a implementação do suporte a novas organizações de agentes; o desenvolvimento de uma camada de comunicação tolerante a falhas, tentando

[4]O jogo Unreal Tournament 2004, possui janelas de 200ms para ação dos agentes, desta forma a cada 200ms chegam novas percepções com o novo estado do ambiente.

minimizar o impacto do atraso inserido por mensagens perdidas; e a investigação sobre a autonomia ajustável em ambientes altamente dinâmicos, com agentes semi-autônomos que permitam a participação de humanos em times híbridos.

Referências

de Byl, P. B. (2004). *Programming Believable Characters for Computer Games*. Charles Development Series.

Defago, X., Schiper, A., and Urban, P. (2004). Total order broadcast and multicast algorithms: Taxonomy and survey. *ACM Comput. Surv.*, 36(4):372–421.

Grosz, B. J. and Kraus, S. (1996). Collaborative plans for complex group action. *Artificial Intelligence*, 86(2):269–357.

Kaminka, G. A. et al. (2002). Gamebots: a flexible test bed for multiagent team research. *Commun. ACM*, 45(1):43–45.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2:83–97.

Levesque, H. J., Cohen, P. R., and Nunes, J. H. T. (1990). On acting together. In *Proc. of AAAI-90*, pages 94–99, Boston, MA.

Monteiro, I. M. (2005). Uma arquitetura modular para o desenvolvimento de agentes cognitivos em jogos de primeira e terceira pessoa. In *Anais do IV Workshop Brasileiro de Jogos e Entretenimento Digital*, pages 219–229.

Monteiro, I. M. (2009). Twproxy: Uma ferramenta de teamwork para ambientes dinâmicos com requisitos de tempo-real. Master's thesis, Universidade Federal do Rio Grande do Sul, Porto Alegre, Rio Grande do Sul.

Monteiro, I. M. and dos Santos, D. A. (2007). Um framework para o desenvolvimento de agentes cognitivos em jogos de primeira pessoa. In *VI Brazilian Symposium on Computer Games and Digital Entertainment - Computing Track*, pages 107–115.

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38.

Rich, C. and Sidner, C. L. (1997). COLLAGEN: When agents collaborate with people. In Johnson, W. L. and Hayes-Roth, B., editors, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 284–291, New York. ACM Press.

Scerri, P. et al. (2003). Team oriented programming and proxy agents: The next generation. In *In: Proc. of the 1st Inter. Workshop on Prog. MAS at AAMAS¿03*, pages 131–138. Springer.

Schurr, N. et al. (2005). Evolution of a teamwork model. In *Cognition and Multi-Agent Interaction: From Cognitive Modeling to Social Simulation*, pages 307–327. Cambridge University Press.

Smith, R. G. (1981). The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113.

Sycara, K. et al. (2001). The RETSINA MAS infrastructure. Technical Report CMU-RI-TR-01-05, Robotics Institute Technical Report, Carnegie Mellon.

Tambe, M. (1997). Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124.