DANIEL MARTINESCHEN

ALTERNÂNCIA ENTRE COMPETIÇÃO E COLABORAÇÃO PARA PROMOVER O APRENDIZADO POR MEIO DE HEURÍSTICAS DE JOGOS

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientador: Prof. Dr. Alexandre Ibrahim Direne

CURITIBA

2006

DANIEL MARTINESCHEN

ALTERNÂNCIA ENTRE COMPETIÇÃO E COLABORAÇÃO PARA PROMOVER O APRENDIZADO POR MEIO DE HEURÍSTICAS DE JOGOS

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre. Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientador: Prof. Dr. Alexandre Ibrahim Direne

CURITIBA

2006

AGRADECIMENTOS

A lista de agradecimentos, para ser justa, teria que ser tão grande que ocuparia quase tantas páginas quantas tem esta dissertação (considerando que eu conseguiria agradecer a todos os que merecem menção). Vou me limitar então a alguns, e os demais eu agradeço "no bolo" (a ordem dos fatores não altera o nível de agradecimento).

O primeiro agradecimento vai ao meu orientador, Alexandre. Um poço infinito de paciência, determinação e motivação que conseguiu me manter até o fim deste mestrado, mesmo com todas as dificuldades e picos de desmotivação que sempre assolam um estudante de pós-graduação. É um cara que posso chamar de amigo.

Agradeço aos meus amigos próximos e caros, que não me deixaram enlouquecer (pelo menos não completamente) durante esses dois anos de trabalho. Seja remoto ou presencial, com brincadeiras ou puxões de orelha, o apoio recebido foi imprescindível. Quase2K, pessoal de Letras, da Lingüística, amigos do PET Informática e outros, do mestrado, da Pastoral da Criança, amigos que fiz através de amigos, companheiros de futebol, amigos do Winterkurs: a todos o meu muito obrigado.

Aos meus pais, Dimas e Beta, e irmãos, Rubens e Henrique, a quem dedico esta dissertação, o meu mais especial e cordial obrigado, simplesmente por serem a minha família. Sem vocês eu não seria nada.

Com certeza eu não consegui mencionar todas as pessoas que de algum modo contribuíram nestes dois anos para a minha formação como cientista e como ser humano. Mas cada um desses (mencionados ou não) sabe que de algum modo faz parte da minha vida. Obrigado.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Este trabalho relata a pesquisa sobre a elaboração de conceitos e ferramentas de software para criar um ambiente facilitador da alternância entre atividades competitivas e colaborativas no ensino de Xadrez. A atuação dos alunos na fase de competição consiste na codificação de conceitos matemáticos em heurísticas de jogo, usando uma linguagem formal e ferramentas de software para edição de funções heurísticas, para participar de uma competição automática. A fase de colaboração, por sua vez, consiste no compartilhamento do conhecimento heurístico presente na competição realizada anteriormente. Os competidores interagem entre si para refinar seu conhecimento sobre o jogo, codificando-o em uma nova função heurística.

O trabalho foi executado em duas frentes principais. A primeira delas consistiu em explorar a literatura da área de Informática na Educação em busca de conceitos e técnicas para o uso alternado de competição e colaboração no ensino. Dada a pequena quantidade de trabalhos integrando essas duas abordagens, o presente trabalho vem a contribuir com a elaboração de conceitos e de uma arquitetura de um sistema destinado a viabilizar a alternância entre competição e colaboração. Os conceitos e ferramentas foram focados no ensino de Xadrez no Centro de Excelência de Xadrez, mas podem ser aplicados, a princípio, ao ensino de outros jogos ou áreas do conhecimento. Propomos que a alternância entre as duas atividades pode gerar benefícios maiores do que cada uma isoladamente. A segunda frente de trabalho foi a prototipação em SWI-Prolog de um comparador genérico de funções heurísticas tendo como máquina de busca o algoritmo Minimax na sua variante com podas *alfa-beta*. A implementação visou definir os parâmetros da máquina de busca que flexibilizam sua execução e a tornam adaptável a simulações de campeonatos de maneira integrada ao servidor de Xadrez. Os resultados de simulação (feita para o jogo Mancalla) permitem fazer uma projeção de valores de tempo que atesta a viabilidade da simulação de torneios de Xadrez com centenas de inscritos em poucas horas.

ABSTRACT

This work reports on the research that covers the definition of concepts and software tools to promote the alternation between competitive and collaborative activities applied to Chess teaching. The role of the apprentices in the competition consists of codifying mathematical knowledge in the form of a heuristic function, supported by a formal language and software tools. This heuristic function then takes part in simulated competitions against other students' heuristic functions. In the collaborative phase the heuristic knowledge (that took part in a previous competition) is shared among the competitors so that each can review his/her concepts and (re-)codify his/her heuristic function, which will take part in a next competition.

The work was conducted in two main fronts. The first was to survey the literature of Computers in Education looking for concepts and techniques that deal with the combination of competitive and collaborative activities in educational environments. Since there are few efforts concerning the integration of these two approaches, this work's first and main contribution is to propose concepts and the architecture of a system aimed at promoting the alternation between competition and collaboration. The concepts and software tools are primarily focused on the teaching of Chess in CEX (Centro de Excelência de Xadrez) but are in principle applicable to the teaching of other games or knowledge areas. The second line of work involved the development of a prototype software tool for generic heuristic comparison, which uses as search engine the Minimax algorithm with alpha-beta pruning. The implementation aimed to define the necessary parameters in order to make the comparator flexible and adaptable to the simulation of large competitions (integrated to the CEX Chess server). Simulation results, using the Mancalla game, showed that it is feasible to simulate a competition with hundreds of heuristic functions in a few hours, and predictions indicate that it is also possible to do it with Chess.

CAPÍTULO 1

INTRODUÇÃO

Neste capítulo fazemos uma descrição geral da dissertação, especificando o problema central a ser tratado, os objetivos a alcançar (gerais, secundários e os passos metodológicos) e o contexto de aplicação em que ela se encontra. Ao fim do capítulo apresentamos a distribuição dos capítulos em que a dissertação foi organizada.

1.1 O problema central

Nos últimos tempos, a área de pesquisa em Informática na Educação tem tido muitos avanços, com a criação e experimentação de diversas ferramentas facilitadoras de ensino e com desenvolvimento de conceitos e diretrizes para como conduzir atividades educativas mediadas por computador.

Algumas abordagens enfatizam o trabalho em grupo e o compartilhamento de informação como meios facilitadores para atingir um objetivo comum — são as atividades *colaborativas*. Nesse tipo de ambiente, o desempenho individual não é o mais importante, mas sim a participação de todos na construção de um conhecimento comum e mais elaborado do que seria possível obter sozinho. Conceitos, ferramentas e arquiteturas de sistemas de software para prover ambientes para colaboração são freqüentes na literatura de Informática Educativa.

Os jogos educativos (propondo um foco no indivíduo), por outro lado, incentivam um aprendiz a despender esforço próprio para se superar e superar seus colegas, embarcando em atividades de *competição*. Muitos educadores argumentam que a presença de competição no ambiente educacional é nociva, pois pode promover o isolamento e até a animosidade entre os aprendizes, e os benefícios do trabalho em grupo seriam diminuídos.

Nesta dissertação propomos a idéia de que ambos os tipos de atividades — competição e colaboração — podem ser combinadas e os pontos positivos de ambas podem

ser explorados no ambiente educativo. A *alternância entre competição e colaboração*pode ser pedagogicamente valiosa, dado que o esforço realizado em uma delas (competição) pode ser compartilhado com o grupo na etapa seguinte (colaboração), de modo que todos possam ter uma visão mais ampla do problema a ser resolvido em grupo e possam na próxima etapa de competição ter um desempenho mais satisfatório e um conhecimento (coletivo) mais aprofundado. Propomos também que isoladamente as atividades de competição e colaboração podem não ter seu potencial eficientemente explorado, pois sem um incentivo ninguém se sente motivado a colaborar, e uma competição por si só realça o desenvolvimento somente de um indivíduo e não da comunidade.

No presente trabalho abordamos o problema de definir conceitos e ferramentas de software que viabilizem a alternância sugerida acima. Há esforços grandes concentrados em cada uma das duas abordagens acima, todos com resultados bastante satisfatórios e promissores. Porém, há uma ausência muito grande de trabalhos que procurem explorar os benefícios que a alternância de fases durante o aprendizado pode ter sobre o desempenho dos aprendizes. Para o próprio jogo de Xadrez, que é ensinado e utilizado como ferramenta didática há anos, até hoje não há notícia de esforços para explorar a alternância de diferentes atividades durante o aprendizado. Propomos aqui que essa abordagem é importante porque procura dar uma nova visão, mais integradora, às atividades de ensino mediadas por computador.

Porém, essa integração tem uma série de dificuldades. Primeiro, para tornar os resultados da competição compartilháveis, deve haver uma maneira de codificá-los formalmente de maneira a poderem ser lidos, entendidos e utilizados por outras pessoas posteriormente. De maneira concreta, isso se traduz no problema de criar linguagens e interfaces para definição de elementos heurísticos. As atividades de colaboração são mais difíceis ainda de conduzir. A principal dificuldade concerne ao sincronismo *interno* dos procedimentos feitos durante a colaboração, pois deve haver diretrizes para conduzir o compartilhamento dos conhecimentos e permitir a interação coordenada entre os aprendizes (respeitando a individualidade e o espaço de cada um). Gostaríamos de ressaltar a idéia de que a competição pode ser considerada um excelente compassador (como um

"metrônomo") externo da colaboração, definindo "quando é a hora de pôr em prática o que foi aprendido".

1.2 Objetivos

1.2.1 Objetivo geral

Como uma meta geral deste trabalho efetuamos estudos de pesquisa e o desenvolvimento de conceitos metodológicos de um ambiente computacional que viabilize realizar experimentos com linguagens e ferramentas destinadas a promover a alternância entre competição e colaboração. Em um trabalho conjunto com Alexandre Feitosa (2005), desenvolvemos protótipos de partes do ambiente e procedemos à especificação o mais detalhada possível da arquitetura do sistema e das políticas de acesso ao mesmo. Além disso, os conceitos de como integrar as atividades de colaboração e de competição foram desenvolvidos e elaborados com base na literatura encontrada.

1.2.2 Objetivo secundário

O objetivo secundário (ou mais específico) do trabalho foi projetar, implementar e realizar experimentos em um protótipo de algoritmo de busca heurística. Esse protótipo tomou como base o algoritmo Minimax em sua variação chamada de "Poda Alfa-Beta" (Rich 1983), principalmente pela economia de memória e simplicidade de implementação. Procedemos à implementação de um protótipo na linguagem Prolog, e utilizamos como base o jogo *Mancalla*, dada sua simplicidade de regras e reduzida explosão combinatória.

O principal objetivo dessa prototipação é encontrar os pontos nos quais um algoritmo de busca pode ser parametrizado (com relação à altura da árvore de busca, ordem de início da partida, identidade dos competidores, entre outros) e as dificuldades inerentes a essa parametrização. Resultados de execução permitem, dado que conhecemos o comportamento do algoritmo Minimax, fazer cálculos de previsão de esforço computacional necessário para executar comparações de quantidades grandes de heurísticas e para

utilizar um jogo mais complexo, o Xadrez.

1.2.3 Objetivos específicos

Nesta seção apresentamos as diversas atividades que precisamos desempenhar para colher as informações necessárias para o desenvolvimento das idéias e programas apresentados neste trabalho. Essas atividades foram:

- Entrevistas com enxadristas para obter elementos e conceitos genéricos de como se estimula um aprendiz por meio da competição;
- especificação detalhada dos principais requisitos de software que o ambiente deve prover, tanto os necessários para viabilizar a alternância quanto os necessários para o funcionamento correto de acordo com as regras e costumes da comunidade enxadrística;
- estudo e discussão das capacidades e dos conceitos envolvidos nas atividades de competição e colaboração, quais as dificuldades de conduzi-las separadamente e o problema de integrá-las;
- projeto e implementação de um protótipo do comparador de heurísticas utilizando o algoritmo Minimax com Podas Alfa-Beta (Capítulo 4);
- estudo das diferenças entre a aplicação do MINIMAX em competições humano–máquina e máquina–máquina, principalmente com relação a delegar à máquina toda a responsabilidade de conduzir uma partida (com todas as variáveis envolvidas);
- participação na co-orientação de alunos de bacharelado na implementação do código definitivo do algoritmo Minimax com Podas Alfa-Beta de maneira integrada ao servidor central de Xadrez do CEX.

1.3 O contexto do projeto

Esta dissertação se encontra inserida no projeto de criação de conceitos e ferramentas para apoio ao ensino de Xadrez nas escolas brasileiras, delineado por Direne *et al.*

(2004). No contexto do projeto maior, é apresentada, entre outras, uma proposta de um ambiente para competição automática entre heurísticas e que viabiliza o compartilhamento de conhecimento heurístico entre os participantes da competição. Em conjunto com esta dissertação, está em desenvolvimento, pelo mestrando Alexandre Feitosa (2005), uma ferramenta gráfica para permitir o uso da linguagem de definição de heurísticas. Por meio dela, será possível codificar rapidamente uma heurística e inscrevê-la num campeonato organizado por um instrutor.

Como já dito acima, o jogo de Xadrez tem sido usado já há anos como uma ferramenta de ensino, para desenvolver o raciocínio lógico dos alunos. Com a difusão do uso de computadores e da Internet, abriu-se uma nova possibilidade de ensino do Xadrez, que até o momento não foi muito explorada. Nos trabalhos de Schäfer e Direne (2000) e Direne *et al.* (2004) são propostos conceitos, ferramentas e arquiteturas de sistemas para autoria e uso de material para ensino do Xadrez mediado por computador nas escolas. Com o que se tem hoje, é possível produzir material para aulas de xadrez e disponibilizar no servidor de xadrez do CEX (Centro de Excelência de Xadrez - `http://www.cex.org.br`).

1.4 Estrutura da dissertação

Esta dissertação está organizada em 6 capítulos. Depois desta introdução, no Capítulo 2 fazemos uma avaliação do estado-da-arte das áreas de aprendizado colaborativo, jogos educativos e Sistemas Tutores Inteligentes. Depois, no Capítulo 3 discutimos as idéias por trás de atividades competitivas e colaborativas, e apresentamos uma proposta de alternância entre essas atividades. No Capítulo 4 discorremos sobre a problemática de simular uma partida de um jogo automaticamente, utilizando um algoritmo de busca e duas funções heurísticas. No Capítulo 5 discutimos detalhes de implementação do sistema e, no Capítulo 6 apresentamos conclusões e discutimos idéias sobre extensões futuras.

CAPÍTULO 2

TRABALHOS CORRELATOS

Este capítulo traz a resenha literária dos trabalhos relacionados com esta dissertação. Nas seções a seguir fazemos uma análise crítica procurando integrar as seguintes possíveis visões dentro da Informática Educativa (IE): (i) aprendizado colaborativo; e (ii) jogos educativos. Discutimos também a contribuição de Sistemas Tutores Inteligentes (STIs) para o ensino colaborativo, bem como a contextualização do trabalho na Inteligência Artificial clássica.

2.1 Aprendizagem colaborativa em ambientes inteligentes

Nesta seção fazemos uma discussão sobre as duas abordagens de ensino/aprendizagem que pretendemos integrar neste trabalho e que são comuns na literatura. A aprendizagem colaborativa é uma prática comum na comunidade de Informática na Educação (IE), em especial a de Ensino a Distância (EaD), e enfatiza o compartilhamento de informação entre os aprendizes. Já na área de STIs a atenção é focalizada no aluno, numa perspectiva predominantemente construtivista e tutorial.

2.1.1 Conceitos e ferramentas de aprendizagem colaborativa

O uso de atividades colaborativas é uma abordagem de ensino/aprendizagem muito comum na comunidade de IE e se baseia no compartilhamento de informação entre os aprendizes para se chegar a um objetivo final (que nem sempre precisa ser o mesmo para todos). O conhecimento permeia o ambiente, e tanto aprendizes quanto educadores evoluem em conjunto. O foco da atividade de ensino não é um aluno individual, mas sim a comunidade como um todo, o que contrasta com a idéia construtivista dos STIs (vide próxima seção).

Normalmente os termos *cooperação* e *colaboração* são usados indistintamente na

literatura, mesmo em trabalhos que apresentem abordagens diferentes umas das outras. Resolvemos adotar aqui a distinção proposta por Gava e de Menezes (2003), que diz que as atividades de cooperação se caracterizam pela divisão de trabalho em equipes que trabalham para a resolução de um problema em comum, enquanto que em atividades colaborativas não só o problema é dividido entre os participantes, mas a característica mais marcante é o compartilhamento de informações, criando assim "um conhecimento compartilhado, que nenhum dos participantes tinha previamente ou poderia obter por conta própria" (Gava e de Menezes 2003).

Sistemas que empregam atividades colaborativas têm uma definição precisa dos agentes (humanos e de software) e dos papéis desempenhados por eles. Em trabalhos como da Cruz Neto *et al.* (2003), Guizzadri *et al.* (2003), Pessoa e de Menezes (2003) são apresentadas propostas de arquiteturas e/ou *frameworks* para desenvolvimentos de grupos e atividades colaborativas e/ou cooperativas[1]. Dada a confusão de terminologia e de identificação dos elementos básicos de um sistema colaborativo, alguns trabalhos (Fernandes e Ferreira Jr. 2004, Gava e de Menezes 2003) propõem modelos e ontologias para a construção de sistemas colaborativos.

Em trabalhos como Boff (2000), Borges (2001), Burton *et al.* (2000), Ferreira e Labidi (1998), Johnson *et al.* (1991), Olguín *et al.* (2000), Silveira e Barone (2005) e Johnson e Johnson (1997) são relatadas experiências de uso de atividades colaborativas em ambientes de ensino virtual e "tradicional" (em sala de aula). São poucos os autores porém que procuraram estudar a integração de atividades competitivas e colaborativas no ambiente educativo (Johnson e Johnson (1994) apud Ferreira e Labidi (1998), Johnson e Johnson (1989), Johnson *et al.* (1991), Qin *et al.* (1995)).

2.1.2 O contraponto dos STIs

Os Sistemas Tutores Inteligentes (STIs) são programas de computador utilizados para o ensino, e que se valem de técnicas da Inteligência Artificial (IA) para saberem *o que* ensinar, *como* ensinar e *a quem* o estão ensinando (Nwana 1990). Os STIs têm seu

[1] Ambos os termos são utilizados de maneiras diversas nos trabalhos.

desenvolvimento dividido em módulos, como a representação do conhecimento específico de domínio, a estratégia de ensino (teoria pedagógica adotada), o modelo do aprendiz e o desenvolvimento da interface com o usuário, além de catálogos de erros. No começo havia os sistemas seqüenciais chamados CAI (Computer-Aided Instruction). Os STIs são uma evolução dos sistemas CAI, e em certa época havia duas nomenclaturas para eles: ICAI (Intelligent CAI) e STI (ou ITS em inglês). Porém, o nome STI foi preferido em detrimento de ICAI pois "[os STIs] representam uma mudança de metodologia muito maior do que a simples adição de um 'I' em 'CAI' " (Nwana 1990).

Das diversas possíveis arquiteturas de STIs, uma que é freqüentemente utilizada é a arquitetura de quatro módulos, ilustrada na Figura 2.1. Nessa arquitetura, temos o Módulo do Domínio (também chamado de Módulo do Especialista), o Módulo do Aprendiz, o Módulo do Modelo Pedagógico e o Módulo de Interface. Essa divisão em módulos tem como objetivo adaptar o conteúdo que está sendo ensinado para o ritmo e as preferências individuais do aluno. Essa abordagem segue, como já apontado acima, uma tendência construtivista, focada no aluno, e é de certa maneira oposta à proposta colaborativa. Neste trabalho propomos a combinação de técnicas da área de STIs com os resultados positivos obtidos em experimentos envolvendo atividades colaborativas.

Perspectivas históricas da evolução dos STIs, desde sua origem nos sistemas seqüenciais, bem como descrições e debates comparativos sobre os vários STIs desenvolvidos ao longo dos últimos 50 anos de pesquisas podem ser encontradas em Nwana (1990) e Wenger (1987).

2.1.3 Linguagens e ferramentas de autoria

O fato de os STIs serem compostos de módulos com características e interfaces bem definidas resultou do fato de esses módulos serem suficientemente estanques entre si e poderem ser desenvolvidos separadamente. Como a área de pesquisa em STIs envolve profissionais de várias especialidades diferentes, é irreal esperar que os autores de material de curso saibam programação de computadores. Para evitar que isso se torne um problema, há uma área de pesquisa que se concentra em criar ferramentas e linguagens que

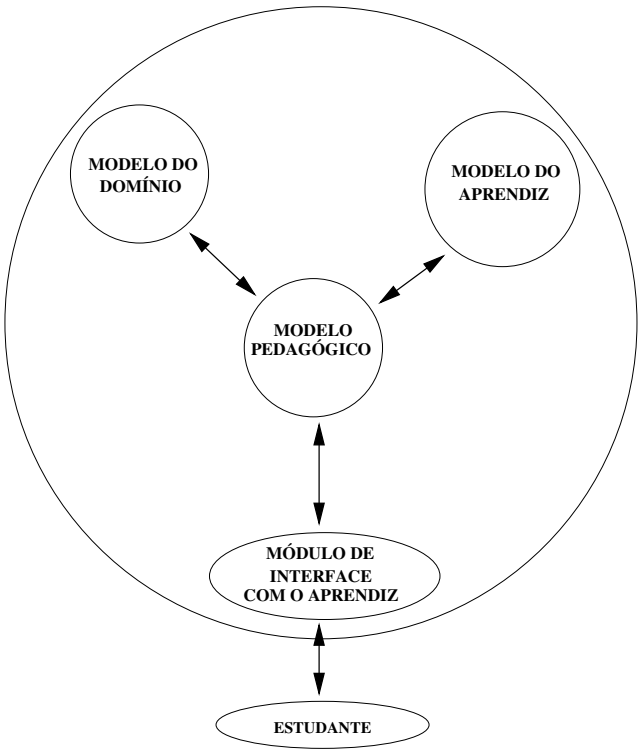

Figura 2.1: Arquitetura de quatro módulos para STIs (Nwana 1990)

permitam o desenvolvimento de conteúdos e de interfaces sem a necessidade de conhecimento especializado em programação de computadores. Esta área é a de Linguagens de Ferramentas de Autoria.

O uso de linguagens e/ou ferramentas de autoria para construção de *software* educativo *inteligente* se justifica pelo salto qualitativo que separa os STIs dos sistemas seqüenciais: estes últimos seguem uma tendência behaviorista e Skinneriana de aprendizado (Nwana 1990), ou seja, o conhecimento é moldado e apresentado em quadros (ou *frames*), de modo a guiar o estudante "passo-a-passo em direção ao comportamento desejado"(Nwana 1990, seção 4.2), e a interação com o sistema é limitada a formas como múltipla escolha ou preenchimento de lacunas. Um sistema feito desse modo é extremamente *ad hoc*, específico para o domínio escolhido, com código de manutenção difícil e pouco reaproveitável em novos sistemas. Para construir um sistema como esse, não são necessárias ferramentas de autoria; na verdade, basta um raso conhecimento de computação e de programação visual para produzi-lo.

Como no modelo do domínio o conteúdo é representado explicitamente e no mo-

delo pedagógico não é definida uma ordem fixa de apresentação do conteúdo (ao contrário, são dadas orientações gerais de avaliação do desempenho do estudante e de adaptação do conteúdo de acordo), o *software* educativo se adapta às necessidades e ao ritmo de progresso do estudante. Uma visão crítica do estado-da-arte de ferramentas de autoria para STIs pode ser vista em Murray (1999).

2.2 Aprendizagem por meio de competição

Tendo analisado a literatura relativa ao ensino por meio de colaboração e o contraponto dos STIs, passamos à discussão de trabalhos que aplicam atividades *competitivas* para o ensino, o que envolve jogos educativos e Inteligência Artificial aplicada à Educação (IAEd).

2.2.1 Jogos Educativos com IAEd

Tem-se utilizado muito jogos (educativos ou não) para tornar o aprendizado mais agradável e incentivar alunos a praticar no jogo os conhecimentos obtidos formalmente em sala de aula. A maioria dos jogos, porém, não consegue realizar na totalidade esse objetivo, uns fixando-se demais no jogo em si, outros no conteúdo que se deseja ensinar, sendo, portanto, apostilas eletrônicas.

Sistemas como WEST (Burton e Brown (1977; 1979) apud Nwana (1990)) — para ensino de operações matemáticas — e QUEST (White e Frederiksen (1985) apud Nwana (1990)) — para diagnóstico de problemas em circuitos elétricos — foram testados experimentalmente, e tiveram resultados satisfatórios. Porém, a abordagem destes sistemas é utilizar um jogo para ensinar um domínio específico de aprendizado (circuitos elétricos ou operações matemáticas). O ensino das estratégias de um jogo (como o Xadrez) como um meio para exercitar conceitos aprendidos em sala de aula, porém, ainda não foi muito explorado. Um dos poucos trabalhos é o sistema SAEX (Schäfer e Direne 2000), no qual uma proposta de definição de conceitos e ferramentas de autoria de material educativo para o ensino de Xadrez é apresentada.

O xadrez já é ensinado nas escolas brasileiras há um bom tempo, com resultados satisfatórios e uma infraestrutura de informática consistente. O CEX (Centro de Excelência de Xadrez) provê cursos, capacita professores e organiza campeonatos entre os alunos. Em Direne *et al.* (2004) é proposta a criação de conceitos e ferramentas para apoio ao ensino de xadrez nas escolas, em que o presente trabalho está incluído. Esse projeto propõe a criação de várias ferramentas para dinamização das aulas, criação de material educativo e desenvolvimento da capacidade dos alunos de codificar sistematicamente o conhecimento sobre o jogo na forma de funções heurísticas.

Ao colocarmos o jogo como objeto central do processo de aquisição do conhecimento, temos uma estratégia de ensino diferente da tradicional. Fundamentos, operações de transformação algébrica, conceitos geométricos e espaciais comumente apresentados pelos STIs são relacionados diretamente com a heurística de jogo, e postos diretamente em prática imediatamente em uma situação "real" — o jogo.

2.2.2 Abordagens tradicionais de busca heurística para jogos

As abordagens tradicionais de busca heurística para jogos tenderam a se concentrar no projeto e na implementação de algoritmos que resolvem a classe dos problemas de cálculo da próxima jogada plausível em jogos heurísticos. Exemplos famosos foram as variações da busca heurística Minimax das quais a mais famosa delas, denominada *Poda Alfa-Beta* (Rich 1983), é capaz de economizar memória ao ponto de precisar armazenar, em qualquer instante, no máximo um ramo da árvore de busca de altura fixa. De maneira semelhante, e o *SSS** (Stockman 1979) também funciona com base em uma árvore de altura fixa mas consegue realizar um número maior de podas nos ramos da árvore, os quais representam jogadas desprezíveis, mesmo com um custo bem maior de ocupação da memória. Também pode ser ressaltada a característica do B^* (Berliner 1979) de realizar a busca com base em uma árvore de altura variável, dependendo da avaliação heurística assim como de outras medidas chamadas de *otimismo* e *pessimismo* das configurações de jogo. Isso se mostrou tão ou mais eficaz que as características de seus concorrentes.

Em pesquisas passadas das quais se tem notícia, freqüentemente encontramos

resultados nos quais as jogadas plausíveis são sempre computadas pela máquina para que esta consiga competir contra um oponente humano (Marsland e Reinefeld 1993). Alguns poucos pesquisadores se lançaram nos desafios de criar algoritmos que permitissem a competição exclusivamente entre duas ou mais máquinas (Hsu 1999, Katsenelinboigen 1989). Todavia, nenhum trabalho conhecido abordou os aspectos acerca da competição conduzida entre máquinas que planejam jogadas segundo funções heurísticas diferentes, utilizando assim as mesmas rotinas de busca.

Quando se delega exclusivamente às máquinas o controle total sobre a comparação entre funções heurísticas, não são apenas os problemas de natureza semântica que são responsáveis por guiar as transformações de estado. Nestes casos, podem surgir também os componentes da pragmática do jogo, inerentes à simulação de aspectos competitivos que são usualmente inseridos na partida por meio do jogador humano ou do juiz humano, mesmo que de forma inconsciente. Por exemplo, no jogo de Xadrez, os componentes da pragmática poderiam estar no tamanho dos registros de lances passados para que haja o monitoramento da ocorrência de ciclos de jogadas. Mesmo a necessidade de adjudicação de uma partida, por qualquer que seja a razão, requer um tratamento especial para ser implementada.

Para que sejam simulados, os componentes da pragmática do jogo exigem a manutenção de grandezas globais inter-lances, o que foge dos padrões de implementação apenas de variáveis locais para o cálculo da próxima jogada plausível. Porém, pode ser muito difícil generalizar os componentes da pragmática para quaisquer jogos, ao contrário do que foi possível criar em termos genéricos para os aspectos semânticos de jogos. Sendo assim, mesmo os algoritmos de busca com aplicação em jogos que sejam de natureza altamente genérica não contemplam as diferenças que cercam a competição Máquina–Máquina com a mesma facilidade que o fazem na competição Humano–Máquina.

Para substanciar as afirmações desta subseção, foi tomada como base a coleção dos principais artigos científicos listados a partir do *Citeseer* (Scientific Literature Digital Library), de acordo com a busca pelas palavras-chave mais representativas da área de pesquisa em questão: *(a)* machine-against-machine / machine-versus-machine; *(b)* game

playing; *(c)* heuristic; *(d)* search algorithm. Obtemos com isso os seguintes temas de pesquisa original e os anos de publicação dos resultados:

- Multi-player alpha-beta pruning (Korf 1991)
- Heuristic search viewed as path finding in a graph (context) (Pohl 1970)
- Searching game trees in parallel (Steinberg e Solomon 1990)
- Parallel alpha-beta versus parallel SSS* (Vornberger e Monien 1987)
- Low overhead alternatives to SSS (Marsland *et al.* 1987)

Em uma rápida avaliação, de fato, pode-se constatar que nenhum dos trabalhos de pesquisa citados enfocou o projeto e a implementação de algoritmos genéricos que visem simular a competição artificial entre duas ou mais funções heurísticas, mas sim otimizações para explorar paralelismo ou otimizações de ajuste fino de desempenho. Inclusive, podemos afirmar que há muito tempo não se pesquisa nas sub-áreas correlatas do tema central aqui abordado. Também encontramos trabalhos que procuram otimizar o algoritmo Minimax ou integrar ao Minimax soluções propostas por outros algoritmos de busca em um algoritmo "híbrido" (Kaindl *et al.* 1991, Korf 1991, Plaat *et al.* 1996). Citamos também trabalhos como os de de Bruin e Pijls (1996) e Korf (1999) para discussões mais formais do estado-da-arte e da complexidade computacional de algoritmos de busca em Inteligência Artificial.

CAPÍTULO 3

ALTERNÂNCIA ENTRE COMPETIÇÃO E COLABORAÇÃO

Neste capítulo apresentamos a proposta de combinar as atividades de competição e de colaboração em um ambiente de ensino e aprendizagem de Xadrez. Depois de uma breve introdução, os conceitos fundamentais de cada tipo de atividade são discutidos e, logo em seguida, a alternância entre as duas atividades é defendida. Uma descrição dos agentes do ambiente e de seus diferentes papéis é feita como proposta avançada deste trabalho de mestrado, e o capítulo é finalizado com uma discussão sobre as limitações da abordagem proposta.

3.1 Visão Propositiva da Dualidade Competição–Colaboração

No mundo de jogos (educativos ou não) utilizados para o ensino, as atividades de *competição* estão presentes sempre, e são basicamente caracterizadas pela disputa para saber quem joga melhor, quem domina melhor as regras do jogo e tem criatividade para saber agir nas mais variadas situações de jogo. A principal vantagem das atividades competitivas é, portanto, o desafio que se impõe a cada competidor para evoluir constantemente.

Em contrapartida — mas não necessariamente de maneira oposta — temos as atividades de colaboração, que são muito comuns em comunidades virtuais de aprendizagem (como em Ensino a Distância). As atividades colaborativas se caracterizam pelo compartilhamento de conhecimento entre os seus participantes, e os aprendizes perdem o papel passivo de simples receptores de conhecimento, mas passam a produzir novas idéias e com isso a comunidade cresce como um todo.

Porém, poucos são os trabalhos encontrados na literatura (como indicado no Capítulo 2) que tratam da combinação de ambas as atividades de modo a explorar as características das duas e promover uma sinergia entre elas. No presente capítulo (desta

dissertação) apresentamos uma proposta de integração entre as duas atividades, aplicada ao ensino de Xadrez. Nas seções seguintes exploramos as características das atividades de competição e colaboração e justificamos nossa proposta de alternância entre elas.

3.1.1 Elementos fundamentais da competição

A competição é um elemento presente em diversas atividades humanas. Não é diferente no ambiente educativo, onde cada aprendiz se esforça para superar seus colegas. Isso é mais evidente ainda quando falamos do ensino do Xadrez, onde os alunos têm aulas em conjunto, têm um acompanhamento individual e põem à prova seus conhecimentos por meio de campeonatos. A competição separa o vencedor do perdedor, e para evoluir cada pessoa tem que se esforçar individualmente para aprender com seus erros, o que não é muito diferente do que ocorre em atividades de natureza predominantemente prática, como a radiologia médica, a programação de computadores e muitas outras. Os aspectos do desafio, da superação dos outros e de si mesmo são as principais características da atividade competitiva.

Sob o ponto de vista do ensino, porém, a competição se caracteriza pelo fato de que cada aprendiz tem seu espaço individual de aprendizagem. Cada pessoa é responsável pelo seu desenvolvimento próprio, e a evolução pessoal não é compartilhada pelos outros competidores. Muitos educadores tendem a ver as atividades competitivas como prejudiciais ao processo educativo, o que explica porque são raros os trabalhos na área de informática educativa que tratem fundamentalmente de jogos (sejam eles educativos ou não) fora do contexto do que chamamos de "atividade lúdica".

Quando o aprendizado do jogo passa a ser um meio e não um fim em si, o objetivo pessoal de vencer perde um pouco a importância para dar espaço ao enriquecimento da comunidade como um todo. Os jogos para os quais propomos aqui a alternância entre etapas de competição e colaboração não são jogos de azar (que dependem de fatores exclusivamente aleatórios), mas sim jogos que envolvem tipicamente o conhecimento profundo das táticas adquiridas por meio de treinamento prolongado. Em outras palavras, são jogos baseados no conhecimento heurístico, o qual pode ser disseminado aos demais membros

da comunidade depois que a competição tiver terminado.

As atividades competitivas no ambiente de ensino são encontradas no uso de jogos educativos e no uso de jogos tradicionais para o desenvolvimento de capacidade de raciocínio e abstração, como é o exemplo do CEX com o Xadrez. Porém, o desenvolvimento de jogos educativos não é embasado necessariamente em teorias educacionais, sendo feito de maneira bastante *ad hoc* ou aleatória. O apelo comercial é grande, pois o que interessa é o jogo em si, e não o uso educacional da atividade competitiva. No plano científico são portanto muito raros os exemplos de trabalhos focalizando o uso de jogos.

3.1.2 Elementos fundamentais da colaboração

Primeiramente, precisamos definir bem o termo *colaboração*. Na literatura os termos *colaboração* e *cooperação* são usados muitas vezes indistintamente, mesmo em trabalhos bastante diferentes. Neste trabalho consideraremos, como foi dito, a distinção apontada por Gava e de Menezes (2003).

Cooperação é a divisão de uma tarefa em partes para que várias pessoas (ou grupos) atuem em conjunto na resolução do problema e obtenham um resultado final (uma solução para o problema original). Num ambiente cooperativo normalmente há um distribuidor de tarefas e um elemento (humano ou de software) que administrará as soluções parciais para uni-las e obter a solução final.

As atividades de *colaboração*, por sua vez, se caracterizam principalmente pelo compartilhamento de conhecimento. Para se chegar ao objetivo final (que pode não ser o mesmo para todos os participantes) deve-se atuar em conjunto, compartilhando idéias e ajudando-se mutuamente. Essa abordagem se encaixa no domínio de ensino de Xadrez, especialmente na atividade de codificação de heurísticas, pois necessariamente cada aprendiz terá que formalizar seus conhecimentos na forma de uma função heurística, contribuindo muito para o crescimento da comunidade.

Em atividades de aprendizado colaborativo, os aprendizes deixam de ser meros receptores e passam a fazer parte da construção do conhecimento. Por meio da prática do compartilhamento de experiências, cada aprendiz pode contribuir com a sua parte na

resolução do problema, e a comunidade cresce como um todo. Os educadores também têm seu papel modificado: deixam de ser apenas fornecedores de conhecimento, mas também aprendem durante a disseminação das idéias surgidas na comunidade. Nas atividades colaborativas, então, cria-se um conhecimento compartilhado, que nenhum dos participantes tinha previamente ou poderia obter por conta própria (Gava e de Menezes 2003).

3.1.3 A Alternância Competição—Colaboração

Nas seções anteriores discorremos sobre as características das duas atividades: *competição* e *colaboração*. No ambiente educacional, nota-se que os dois tipos de atividade estão presentes, porém a idéia que normalmente se tem é a de que elas não são compatíveis. A atividade competitiva, pelo seu caráter individualista, poderia promover o isolamento e até a animosidade entre os aprendizes, pois cada um tem de ser melhor que os outros. Porém, é justamente por meio da interação entre a atividade colaborativa e a competitiva que propomos um novo ambiente de ensino e aprendizagem que explore as características de ambas as atividades.

A idéia central proposta aqui é que, ao serem incitados a competir entre si durante um certo período de tempo, os aprendizes darão o máximo de esforço individual para tentar colocar seu conhecimento das táticas do Xadrez em prática. Ao codificar matematicamente a sua heurística por meio de uma ferramenta de edição (Feitosa 2005), o aprendiz formaliza seu conhecimento na forma de uma função computável, que irá representar sua noção da tática de avaliação de tabuleiros. Em resumo, tal função pode ser comparada automaticamente com as funções dos demais aprendizes por meio da execução de um algoritmo de busca heurística que transforma a comparação em uma verdadeira competição artificial (ver Capítulo 4). Essa atividade de avaliação heurística da posição de um tabuleiro já é mentalmente executada pelos estudantes de Xadrez, pois o contexto para se pôr em prática o aprendizado é o do próprio jogo. A diferença está na formalização prévia do conhecimento sobre a tática de jogo na forma de conceitos matemáticos para que ele possa ser posteriormente compartilhado com os competidores da comunidade por meio de um baixo esforço coletivo. Note-se que os conhecimentos matemáticos aqui

citados (aritméticos, geométricos, algébricos e outros) são o foco de uma outra dissertação de mestrado em desenvolvimento no presente instante (Feitosa 2005).

Na atividade de colaboração, por sua vez, os estudantes se reúnem e interagem entre si para entender a razão do desempenho de suas heurísticas na competição. O professor, então, seleciona as heurísticas de alguns alunos e as usa como exemplo em aulas posteriores. Além disso, no sistema, as heurísticas dos competidores ficam disponíveis para quem quiser vê-las ou copiá-las.

A idéia da alternância entre a competição e a colaboração é fazer com que os resultados de uma etapa do aprendizado se propaguem para a outra, refinando assim o conhecimento. A aprendizagem é baseada no conjunto de conceitos gradualmente desenvolvidos pela comunidade, ou seja, um membro da comunidade não pode evoluir sem a interferência de outros membros. Em particular, tal interferência se dá diretamente por meio do resultado da competição sendo propagado e compartilhado com os demais participantes. Acredita-se, no contexto desta proposta, que este mecanismo original pode ser bem controlado por seres humanos, com o apoio de ferramentas de software, para melhorar o desempenho da capacidade de abstração, principalmente a de natureza matemática.

A seguir, delineamos a dinâmica dessa alternância de atividades.

3.2 A dinâmica da alternância

A dinâmica da alternância entre competição e colaboração é ilustrada na Figura 3.1. Observa-se que a primeira atividade de todas é a de competição ("Torneio1"). Propomos que essa seja a atividade inicial porque logo após as aulas de Xadrez e sobre heurísticas de jogos, não há ainda o que compartilhar entre os alunos, pois estes ainda não testaram seus conhecimentos num torneio[1]. Assim, primeiramente os alunos devem pôr à prova suas heurísticas num campeonato, iniciado e administrado pelo professor (veja Figura 3.2).

Após o campeonato, o professor ou coordenador da competição interage com o

[1]Fazendo um paralelo com a maneira como é praticado hoje, seria o mesmo que logo após as aulas falar para os alunos compararem o que sabem, mesmo sem ter disputado nenhuma partida.

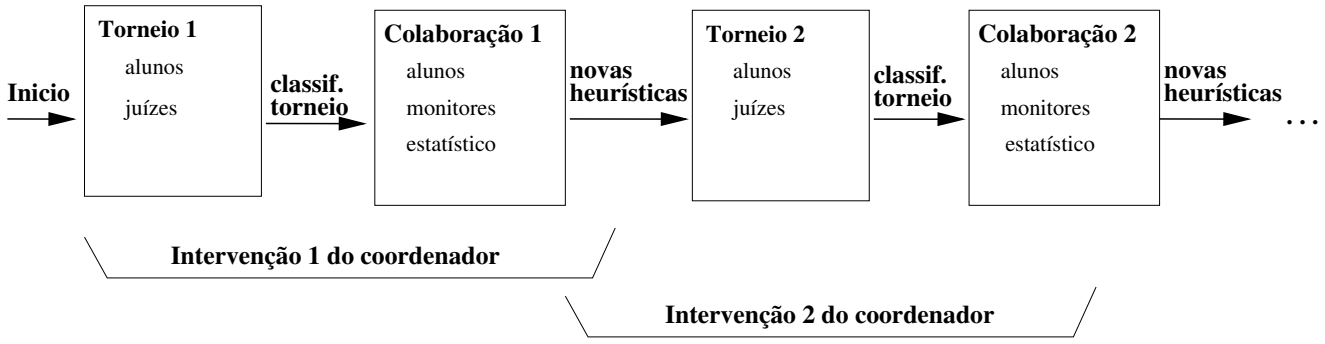

Figura 3.1: Diagrama geral da alternância entre competição e colaboração

sistema, habilitando a leitura das heurísticas dos competidores para todos e abrindo o espaço para discussão. Com os códigos das heurísticas disponíveis ao público, o professor pode então selecionar algumas para exposição e discussão em sala-de-aula (os agentes da alternância, descritos na Seção 3.3.1, têm papel primordial nesta etapa do aprendizado). A dinâmica da atividade de colaboração é mostrada na Figura 3.3, à página 21.

Quando a etapa de colaboração estiver concluída, ou seja, quando a discussão estiver bastante amadurecida, o professor então orienta seus aprendizes a recodificar suas heurísticas, podendo reescrevê-las do zero, modificar as já existentes e mesmo copiar alguns pedaços de heurísticas de colegas[2]. Depois disso, é criado um novo torneio, fecha-se um ciclo da alternância e inicia-se outro. Após esse torneio, uma nova etapa de cooperação acontecerá, e espera-se com isso que o conhecimento seja refinado mais uma vez.

3.3 Benefícios pedagógicos da alternância

Nesta seção vamos comentar alguns dos benefícios pedagógicos que acreditamos que a alternância de atividades de competição e colaboração pode trazer à sala de aula, não nos limitando somente à aplicação na comunidade enxadrística.

Ao contrário do que se pratica tradicionalmente na comunidade de pesquisa e desenvolvimento de STIs — ou mesmo de IE geral que aborde Jogos Educativos como tema central de trabalho— o presente trabalho oferece a vantagem da aplicação de conceitos específicos (exemplo: conceitos matemáticos aprendidos em aula) em um ambiente

[2]A questão de plágio é tratada no Capítulo 5.

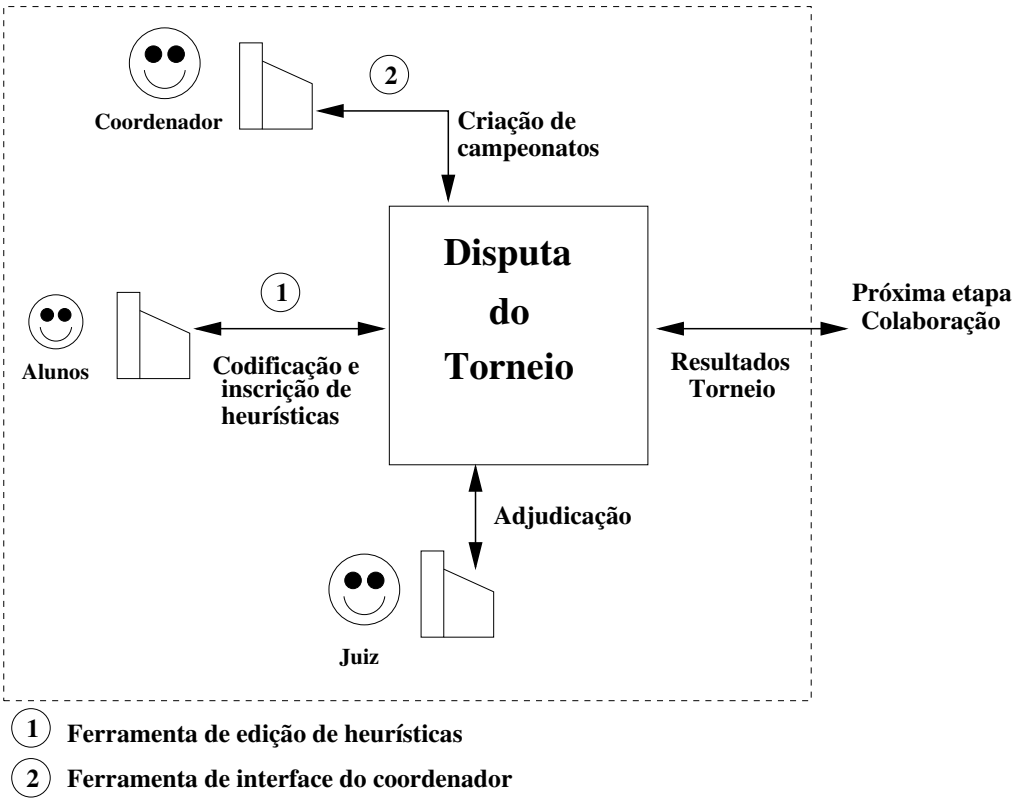

Figura 3.2: Diagrama esquemático da competição

centrado no jogo propriamente dito. Isso não deixa de se caracterizar como Jogo Educativo, mas é uma visão inovadora que, propomos aqui, é de grande expectativa na evolução das formas de ensinar e aprender de maneira mais construtivista (voltada para o aprendiz). Como já foi explicado no Capítulo 2 (resenha literária desta dissertação), sistemas tutores fortemente interventores, chamados "sistemas treinadores" ("coaching systems") foram desenvolvidos e tiveram sucesso limitado (WEST, QUEST entre outros), o que justifica nossa motivação de integrar as atividades.

Um outro benefício é a maior formalização de táticas para a própria comunidade enxadrística, a qual carece de conceitos facilmente documentados, mesmo os de nível intermediário (como os de natureza tática, que não cheguem a ser os de estratégias avançadas). O que propomos aqui é que isso seja obtido por meio do esforço por parte dos alunos de codificarem seu conhecimento de Xadrez numa linguagem formal e sem ambigüidade como é a linguagem de definição de heurísticas.

Com essa visão inovadora sobre a dinâmica de atividades de ensino, pretendemos

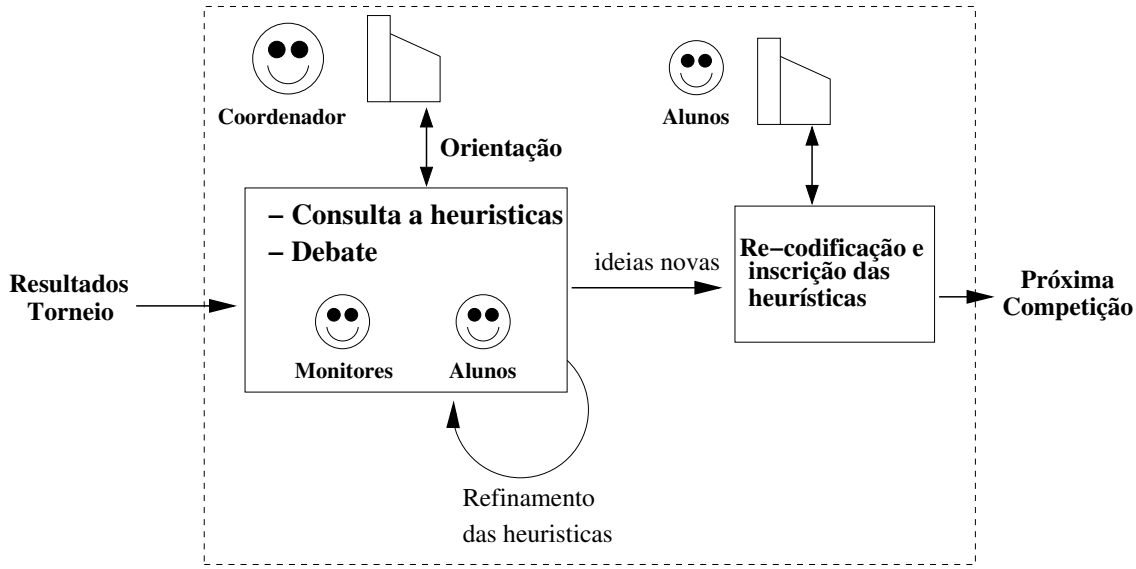

Figura 3.3: Diagrama esquemático da colaboração

abrir uma frente de atuação *suplementar* dos softwares da chamada classe de Softwares Educativos, que permita uma atuação dos alunos como aprendizes voluntários e que, ainda assim, permita que seus desempenhos e aplicações originais de conceitos matemáticos sejam avaliados, direta ou indiretamente. Diretamente, quando o coordenador quiser analisar o código da heurística em si que o aprendiz tenha desenvolvido (ou apresentado de forma evolutiva ao longo de várias competições e colaborações). Indiretamente, por meio da verificação de qual posição o aprendiz atingiu na competição com outros [3].

3.3.1 Agentes promotores da Alternância entre cooperação/competição

No ambiente competitivo podemos identificar pelo menos três agentes ou papéis: o *coordenador*, que é quem organiza os competidores em grupos, ajusta as regras do jogo e coordena as partidas; os *competidores*, que são as pessoas ou grupos que vão disputar entre si; e os *juízes ou avaliadores*, que são aqueles que vão garantir o bom e justo andamento das disputas pela mediação dos competidores ou, se necessário for, da aplicação de regras de solução de conflitos que podem inclusive definir quem serão os vencedores da competição.

[3]Naturalmente para essa avaliação assume-se que as componentes heurísticas dos aprendizes não tenham sido copiadas de outros durante a fase de colaboração.

Esses agentes serão fundamentais também no processo de alternância entre competição e cooperação, e seus papéis serão descritos mais adiante, na seção apropriada, juntamente com os papéis de outros agentes.

Na atividade de colaboração temos a participação dos seguintes agentes: o *aprendiz*, que não tem um papel somente passivo na aquisição de conhecimento, mas participa na construção do mesmo por meio do debate e da disseminação de idéias em grupo; o *coordenador*, que é normalmente um professor, e que divide os aprendizes em grupos, atribui as tarefas aos grupos, as divide entre os integrantes, entre outros; o papel do *avaliador* pode ser atribuído a uma pessoa que desempenhe somente esse papel ou o coordenador pode acumular essa tarefa, que é a de monitorar o desempenho dos alunos e conferir-lhes nota. Esses agentes, principalmente o aprendiz e o coordenador, existirão no ambiente da alternância e seus papéis serão incrementados com as atividades correspondentes no ambiente competitivo.

Nesta seção proporemos papéis para os diversos agentes ou atores no ambiente de ensino com a alternância de competição e colaboração. Mais que uma justaposição dos agentes da competição com os da colaboração, é uma redefinição desses papéis de acordo com a dinâmica das atividades e com o ferramental de software proposto.

3.3.1.1 O Aprendiz/Competidor

O aluno ou aprendiz é a figura principal do ambiente. É quem está aprendendo o jogo de Xadrez e exercitando os conhecimentos formais aprendidos em sala de aula. No contexto atual do CEX cada aluno pode ter um cadastro no servidor de xadrez e jogar partidas isoladas ou campeonatos criados por usuários autorizados para tanto. Neste trabalho propomos uma extensão no papel dos aprendizes. Fazendo uso da ferramenta de definição de heurísticas (Seção 5.1.1), cada aluno poderá codificar a sua própria heurística para xadrez e inscrevê-la em um campeonato ativo. Cada aluno terá à sua disposição uma galeria onde estarão armazenadas as heurísticas que ele construiu no passado. Isso tudo ficará armazenado na base de dados do servidor de Xadrez do CEX.

Na atividade competitiva, a interação do aprendiz com o sistema se limita a sub-

meter uma de suas funções heurísticas para uma competição ativa, e aguardar o resultado dela. Durante a execução do torneio, o aluno (e qualquer pessoa cadastrada no servidor do CEX) também pode acompanhar as partidas que estão sendo disputadas num dado momento. Isso já é possível de ser feito utilizando o software Xboard/Winboard. Ao final do torneio cada aprendiz tem acesso então à sua classificação final na competição.

Durante a atividade de colaboração, por sua vez, o aprendiz pode acessar sua heurística e a de outros competidores, para compreender o porquê de seu desempenho no campeonato. A ferramenta de codificação de heurísticas permite ao aprendiz ler, comentar e copiar partes de heurísticas de outros competidores, e dessa forma se preparar para o próximo torneio.

3.3.1.2 Coordenador de competição

O coordenador (ou professor) normalmente é quem toma conta de uma turma de Xadrez, e é quem vai usar o sistema do servidor de Xadrez para iniciar um campeonato entre seus alunos. Ele terá acesso a uma ferramenta de software que lhe disponibilizará diversas funções para interagir com o servidor, entre elas:

- iniciar o torneio, com datas de inscrição e encerramento;
- escolher o tipo de escalonamento e modalidade de partida;
- editar e comentar as heurísticas dos competidores;
- definir permissões de acesso dos competidores às heurísticas dos demais;
- definir as permissões dos colaboradores;
- adjudicar partidas (se necessário).

O coordenador de competição é uma pessoa chave para a realização da alternância entre competição e colaboração. Seu papel na alternância consiste em, após um torneio, abrir as permissões de leitura para as heurísticas dos competidores e, assim, iniciar o processo de colaboração. Também atuando como professor, ele coordena o debate das

heurísticas e dos resultados dos campeonatos em sala de aula que culminará na escrita de novas heurísticas e no início de um novo torneio.

Uma das funções principais do coordenador de competições, recém-listada acima, é a de definição de permissões para os colaboradores no sistema. Essa função lhe permite conceder direitos especiais de interação com o sistema para pessoas que venham a lhe ajudar no processo de ensino, como o monitor, o juiz ou o estatístico.

No futuro, quando a comunidade estiver confortável com as ferramentas de software, deverá ser possível aos competidores também criar pequenas competições "locais" utilizando suas próprias galerias de heurísticas, para testá-las antes de se inscrever em um torneio.

3.3.1.3 Monitores

Os monitores são competidores que, depois de muito tempo de uso do sistema, o conhecem bem e têm domínio de heurísticas e da própria linguagem de definição de heurísticas. O coordenador, fazendo uso de sua ferramenta de interação com o sistema, configura permissões para que um determinado monitor comente heurísticas de demais competidores, crie campeonatos pequenos, possa intervir em campeonatos que estejam ocorrendo (cancelando, suspendendo, etc), entre outras atividades. O papel principal do monitor é o de assegurar a coesão do grupo, como um nó de difusão de conhecimento, e sua função é principalmente corretora (auxiliando o professor a orientar os alunos mais iniciantes) e catalisadora do conhecimento (fornecendo dicas e macetes de como expressar as idéias na linguagem de heurísticas).

3.3.1.4 Enxadristas Mestres

O enxadrista mestre exerce um papel importante no ambiente de alternância, cooperando com os coordenadores de competição. Eles atuam como referência para os coordenadores no que diz respeito a modalidades de pareamento comuns, dúvidas sobre regras do Xadrez, questões relativas a adjudicação de partidas, e outras questões que demandem o conhecimento profundo sobre Xadrez dos especialistas.

Neste trabalho prevemos também uma atividade extremamente seletiva para os enxadristas mestres, em que eles poderão contribuir com sua experiência e seu conhecimento profundo do jogo de Xadrez. Essa atividade consistirá na utilização da metalinguagem para definição de heurísticas de jogos em geral, de modo a poder definir melhor a chamada linguagem-objeto que os aprendizes virão a utilizar, para que os aprendizes possam especificar suas heurísticas de forma lógico-matemática. Para maiores detalhes sobre essa proposta, apontamos o trabalho de mestrado de Alexandre Feitosa (2005).

3.3.2 Estatístico

O *Estatístico* é mais um dos agentes que vão auxiliar o coordenador a conduzir as atividades de alternância. Seu papel exato ainda não é muito claro, mas certamente, com o avanço da comunidade e com o uso freqüente do sistema, certas tarefas se tornarão necessárias para o bom andamento das atividades.

Como seu nome já diz, o estatístico é alguém que fará levantamentos estatísticos sobre o desempenho dos aprendizes, produzindo relatórios para o coordenador. Fazendo uso de ferramentas de *matching* de heurísticas, ele poderá estimar uma medida de "cópia" de uma dada heurística, ou seja, o quanto um aluno se baseou em heurísticas de colegas ou adversários para construir a sua própria heurística. Isso não retirará o valor do esforço do aluno, pois a evolução do seu aprendizado dependerá da interação com a comunidade, mas permitirá identificar pessoas que possam vir a se aproveitar do esforço de outros para construir heurísticas vencedoras. Outra característica que ele poderia auferir seria uma medida de *complexidade* de uma heurística, para poder perceber se um aprendiz não está "exagerando" na precisão da sua função heurística.

Como dito, o papel do estatístico ainda não está muito claro, e muitas das suas funções poderiam ser acumuladas pelo coordenador ou pelos monitores. Mesmo as ferramentas de *matching* e a de medida de complexidade de heurísticas poderiam estar disponíveis para os aprendizes poderem se auto-avaliar melhor.

3.3.2.1 Juízes

Os *juízes* atuam nas atividades competitivas e colaborativas, basicamente adjudicando partidas e garantindo o bom e justo andamento da competição. Sua participação nas atividades colaborativas derivaria dos detalhes referentes às decisões de adjudicação tomadas em determinados momentos. Se um juiz decidisse que o resultado de uma dada partida deveria ser de empate, a ação de adjudicação poderia conter informações como a quantidade de tempo levada por cada heurística para analisar uma jogada, um comentário do juiz justificando sua decisão, quantidade de jogadas cíclicas possivelmente executadas, entre outras. Essas informações podem ser cruciais para identificar o mau funcionamento de uma heurística, ou mesmo de uma parametrização incorreta do módulo de simulação de partidas (sob controle do coordenador).

3.4 Limitações da abordagem

Nesta seção fazemos uma reflexão acerca das limitações que a abordagem proposta de alternância entre competição e colaboração apresenta.

Um dos fatores que pode vir a dificultar a implantação do ambiente de alternância é o fato de algumas pessoas poderem copiar muitas partes de heurísticas de colegas, aproveitando-se assim do esforço de outros. Em princípio isso pode ser evitado, com a monitoração do coordenador e do estatístico, porém pode se tornar um empecilho ao bom funcionamento da comunidade.

A própria linguagem de definição de heurísticas (embora esta pertença ao escopo de outro trabalho de mestrado) responde por uma enorme parte da motivação, da facilidade de uso do sistema e da abrangência dos conceitos lógico-matemáticos. O sucesso do sistema depende de uma estruturação precisa da linguagem, pois a codificação das heurísticas é o que viabilizaria toda a idéia de alternância. A ação dos enxadristas mestres nessa área se torna, portanto, indispensável.

Associado à linguagem de definição de heurística há que se considerar também as ferramentas de software para apoio das atividades dos agentes da comunidade. Elas não

podem ser muito restritivas (oferecendo poucas funcionalidades ou com fraca ergonomia) nem conceder muitas liberdades aos usuários (pois assim seria difícil o controle sobre o sistema). Adicionalmente, a implementação eficiente das ferramentas se faz necessária para que as mesmas não se tornem empecilhos à alternância ou a façam uma tarefa maçante.

CAPÍTULO 4

A SIMULAÇÃO DE UMA PARTIDA DE XADREZ

Neste capítulo discutimos os aspectos teóricos e práticos do módulo de simulação de partidas de Xadrez. Primeiramente discutimos a problemática de "jogar Xadrez", considerando as hipóteses simplificadoras necessárias para seu estudo e implementação em computador. Em seguida, falamos sobre a explosão combinatória do jogo de xadrez. Por fim, apresentamos argumentos para a escolha de um algoritmo de busca eficiente e que seja de fácil integração com o ambiente *on-line* de Xadrez do CEX, além de discutir a sua parametrização para que o mesmo seja o mais genérico possível.

4.1 Introdução

Um problema recorrente na IA é realizar uma busca em um espaço contendo estados, chamado *espaço de busca*. Diversos algoritmos de busca já foram desenvolvidos, e dentre os mais bem sucedidos estão os algoritmos de busca heurística para jogos. Nestes, o espaço de busca (um grafo *E-OU*, como será visto a seguir) é gerado e percorrido, e uma função de avaliação indica a vantagem (ou desvantagem) material que o conteúdo de um certo estado apresenta no momento. Essa função se chama "função heurística", e se baseia em parâmetros específicos do domínio do problema abordado.

Os algoritmos de busca heurística foram amplamente utilizados em jogos, pois, devido à freqüente explosão combinatória, a função heurística permite guiar a busca em direção a uma solução de modo mais eficiente do que um algoritmo de "força bruta", se é que a força bruta é possível no caso. Em jogos como xadrez a explosão combinatória de possibilidades de jogadas é tão grande que torna a busca exaustiva impraticável em tempo hábil.

O presente módulo aborda o problema de construir um algoritmo de busca heurística genérico, que efetuará a simulação automática de uma partida entre dois com-

petidores artificiais. Cada competidor corresponderá a uma função heurística, definida por um ser humano por meio de uma ferramenta de edição de heurísticas (trabalho de mestrado em desenvolvimento por Alexandre Feitosa).

A principal contribuição deste módulo consiste na generalização e na automação do maior número possível de aspectos do algoritmo de busca. Esses aspectos incluem a máquina de busca em si, a altura da árvore de busca, tempo das partidas, a heurística de cada jogador, entre outras.

4.1.1 Hipóteses simplificadoras

Como foi descrito, precisamos traduzir o complexo problema de "jogar xadrez" em um problema computacional, mais simples e passível de ser implementado. "Jogar xadrez" é, para as pessoas, uma tarefa complexa, que requer concentração, orientação espacial e capacidade de enxergar conseqüências futuras de jogadas efetuadas. Adquirir o grau de Mestre — ou mesmo poder dizer que "joga bem" — demanda muito treino, estudo e reflexão sobre o jogo.

Ao implementarmos em um computador a tarefa de jogar xadrez, devemos reduzir esse problema tão complexo em tarefas menores e mais simples, computáveis. Isso vem sendo feito com sucesso há décadas na comunidade de Inteligência Artificial, adotando uma série de hipóteses simplificadoras como as descritas a seguir. Tomando como base essas hipóteses, o problema de "jogar Xadrez" se reduz primeiramente à dimensão de "calcular a próxima jogada plausível". Todavia, pesquisas anteriores (Lesgold *et al.* 1989) revelam que os grandes-mestres nem mesmo precisam planejar explicitamente os lances para decidir uma jogada. De acordo com experimentos conduzidos nas pesquisas, os grandes-mestres apreendem enormes quantidades de padrões de jogo, de maneira que a classificação de uma dada configuração de tabuleiro ocorre muito rapidamente, e de maneira precisa. Isso inclui também os detalhes relativos ao estágio em que o jogo se encontra (abertura, meio e final do jogo).

Em primeiro lugar, precisamos de uma forma de representação do universo do jogo. Isso é feito por meio de um *grafo de busca*, em que cada vértice é a representação

de uma configuração de tabuleiro (chamada *estado*) e cada aresta ligando dois vértices representa uma *transformação de estado* ou uma jogada. Em jogos esse grafo se chama *grafo E-OU* puro, que pode ser visto como uma árvore (pois permite repetição de jogadas). Isso será discutido com detalhes na Seção 4.3.1.1.

Juntamente à criação da representação do universo do jogo, deve-se pensar nas regras atômicas de transformação de estado, que vão, de fato, viabilizar a criação do grafo de busca. Para cada jogo (ou domínio de aplicação) há um conjunto de regras de transformação específicas. Dependendo do jogo e do algoritmo de busca escolhido, essas regras de transformação podem representar um empecilho quando se pensa em um algoritmo de busca genérico (p. ex. com relação a regras que dependem da não aplicação prévia de outras regras anteriormente, como o roque no xadrez). Essa problemática será discutida na Seção 4.3.1.2.

Por fim, mas não menos importante, deve-se considerar o algoritmo que vai percorrer o espaço de busca. Ele deve fazê-lo de modo eficiente, e usar uma função heurística para guiar a busca — por causa da própria natureza heurística do problema de jogar xadrez. Os requisitos para o algoritmo, principalmente os aspectos que permitem a sua generalização, serão discutidos com detalhe na Seção 4.3.1.4.

4.2 Análise combinatória da busca

Ao modelarmos um problema em um espaço de busca, com a representação dos estados e as regras de transformação dos mesmos, recaímos no problema da explosão de possibilidades de novos estados gerados. Para problemas simples, como o jogo-da-velha ou o jogo dos quadrados deslizantes, a explosão combinatória é limitada, uma vez que a quantidade de transformações plausíveis a partir de um dado estado é limitada (mesmo assim, o jogo dos quadrados deslizantes com 24 quadrados tem uma árvore de jogos com cerca de 10^{20} nodos (Korf 1999)).

No jogo de xadrez, para cada tabuleiro há, em média, 35 possibilidades de jogada plausível. Considerando que, em média, a cada partida um jogador efetua 50 movimentos, temos uma árvore de jogadas com aproximadamente 35^{80} estados. Percorrê-la

inteiramente levaria mais tempo do que o tempo de vida de um dos jogadores (Rich 1983). Por essa razão é necessário que a busca seja guiada por uma função heurística.

4.2.1 Abertura, meio e fim de jogo

O jogo de xadrez (assim como Go e outros jogos de tabuleiro) possui uma peculiaridade: jogadas de início e fim de jogo são muito estilizadas, e têm nomes próprios, indicando o estilo condizente a ela. Essa peculiaridade, associada à busca heurística, poderia permitir uma redução significativa do tempo médio para o cálculo de próxima jogada plausível, dado que nos momentos de abertura e fechamento da partida é possível consultar um catálogo contendo essas jogadas típicas. Desse modo, conhecimento tabulado sobre o jogo pode ser aliado à heurística para obter resultados mais satisfatórios. Na ferramenta de edição de heurísticas deverá haver a possibilidade de seleção de jogadas de catálogo para abertura e fechamento de partidas.

Porém, as fronteiras entre uma fase e outra do jogo são pouco precisas, e mesmo grandes mestres têm dificuldade em especificá-las. A abertura leva em média 10–12 movimentos para ser feita; o meio do jogo é a fase em que são feitos a maior parte das jogadas — e é quando muitos jogos são ganhos ou perdidos; e o fim (ou fechamento) é quando o jogo é efetivamente terminado, com poucos (quando há vitória fácil) ou com muitos lances (um empate, quando ocorre a ciclicidade das jogadas ou uma vitória disputada).

4.2.2 Análise combinatória *versus* jogadores humanos

Como foi dito na seção 4.1, adotamos diversas hipóteses simplificadoras para projetar um jogador automático de xadrez. Uma delas foi representar o universo do jogo (tabuleiros e jogadas) na forma de um grafo de busca. Essa simplificação é a maior delas, pois com ela o problema complexo de jogar xadrez é reduzido à tarefa de percorrer um grafo e encontrar um caminho (seqüência de jogadas) que leve do estado inicial ao estado-meta — e escolher esse caminho como a jogada a ser efetuada.

Essa abordagem, porém, não representa a habilidade de jogar de um humano em toda a sua extensão — como já era de se esperar. Um jogador humano também efetua

mentalmente uma espécie de "busca no grafo", quando pensa nas possibilidades de jogadas e nas suas conseqüências. Mas a habilidade de um especialista — um grande-mestre, por exemplo — não se resume somente a uma busca num grafo. Na verdade, essa busca representa uma fração pequena da atuação de um grão-mestre, pois com o tempo de prática e estudo este adquiriu conhecimentos meta-heurísticos e extra-jogo que lhe permitem jogar conscientemente sem precisar recorrer à busca exaustiva. Essas capacidades, apesar de serem ainda pouco conhecidas, incluem:

1. alta especialização visual, reconhecendo configurações de tabuleiro como favoráveis ou não quase instantaneamente, e capacidade de reconstrução da seqüência de jogadas que levou a esse tabuleiro (Hartmann *et al.* 2005);

2. conhecimento amplo dos estilos de abertura, fim e condução do meio da partida;

3. identificação de harmonia de peças (Hartmann *et al.* 2005);

4. conhecimento da anatomia do tabuleiro e acompanhamento dinâmico das peças por memorização das configurações (Schäfer e Direne 2000);

5. capacidade, inerentemente humana, de "jogar com a sorte", ou seja, confiar em erros do adversário para, por exemplo, sair de uma situação desvantajosa (Rich 1983, p. 129);

6. uso de fatores distrativos, como simulação de erros, desviar a atenção do adversário a outra região do tabuleiro, etc.

As capacidades enumeradas sugerem que humanos conseguem jogar tão bem xadrez porque diversificam a solução do problema, valendo-se de muitas habilidades cognitivas diferentes. Apesar de representarem uma "vantagem" para os humanos, diversas tentativas têm sido feitas (e tem sido razoavelmente satisfatórias) de implementar essas capacidades em computadores. Além disso, com o avanço da microeletrônica produzindo processadores cada vez mais rápidos, é possível fazer análises de bancos de dados de jogadas gigantescos em frações de segundo, conferindo vantagem grande para as máquinas (como exemplificado pelo DeepBlue).

Há experimentos recentes de se implementar capacidades como a de reconhecer estilos de jogo ou de "jogar com a sorte", mas em geral se baseiam em catálogos de partidas ou na inclusão de aleatoriedade no algoritmo. Essas abordagens tentam reduzir o hiato (*gap*) entre a capacidade humana de jogar (genericamente, de resolver problemas) e a busca heurística, mas o fazem por meio de "força bruta" na maioria das vezes. Não trataremos desses experimentos neste trabalho.

4.3 Algoritmo de busca genérico

Nesta seção discorremos sobre a escolha de um algoritmo de busca para a simulação de partidas de xadrez. Primeiramente discutimos os requisitos que tal algoritmo deve ter com relação à representação do conhecimento e com relação ao método de busca em si. O algoritmo Minimax com podas Alfa-Beta é apresentado como a principal opção de algoritmo de busca e, por fim, discute-se a sua flexibilização de uso e os parâmetros necessários para alcançá-la.

4.3.1 Requisitos fundamentais

4.3.1.1 Requisitos de representação do conhecimento

Como delineado na Seção 4.1.1, devemos adotar diversas hipóteses simplificadoras ao implementarmos um jogador automático de xadrez (ou de qualquer outro jogo). Falaremos aqui sobre a representação do problema em uma estrutura de dados apropriada, que permita a busca por uma solução.

Tradicionalmente o espaço de busca de um problema é representado por meio de um grafo *OU* simples, no qual os vértices são chamados de *estados* e as arestas representam *transformações* que levam de um estado a outro (vide seção 4.3.1.2). Em jogos, um estado é entendido como uma disposição das peças no tabuleiro, além de variáveis ou sentinelas globais que indiquem efeitos residuais entre jogadas, permitindo assim o controle sobre situações como o roque e a ciclicidade de jogadas (que caracteriza empate). Uma transformação é entendida como sendo uma jogada. Um *caminho* em um grafo de busca

representa uma seqüência de ações (jogadas) que leva do estado inicial ao final (ou estado-meta).

Um grafo de busca pode ser visto estritamente como uma árvore (permite a repetição de estados), e em cada nível dessa árvore estão os estados resultantes de transformações efetuadas nos estados do nível superior. Em um grafo *E-OU* há alternâncias entre nós *E* e nós *OU*, sendo que os primeiros indicam que todas as ações em seguida são obrigatórias, e os últimos que as ações a serem tomadas a partir deles são opcionais (ou seja, basta tomar uma delas). Os nós *E* representam restrições que devem ser obrigatoriamente atendidas, enquanto que os nós *OU* representam restrições das quais ao menos uma precisa ser satisfeita.

No mundo de jogos (especificamente em jogos de 2 oponentes), a representação do espaço de busca é dada por uma variação do grafo *E*-OU simples, chamada de *grafo E-OU* puro, o qual é uma árvore com alternância garantida de nós *OU* e *E* entre os seus níveis, da raiz até os nodos-folha. Nodos-folha são os últimos estados gerados quando a profundidade máxima da árvore foi atingida. Na figura 4.1 ilustramos um grafo *E-OU* puro representando uma busca por opções para sair na sexta à noite com a namorada. Os nós com um pequeno arco representam os nós *E* e os demais representam nós *OU*.

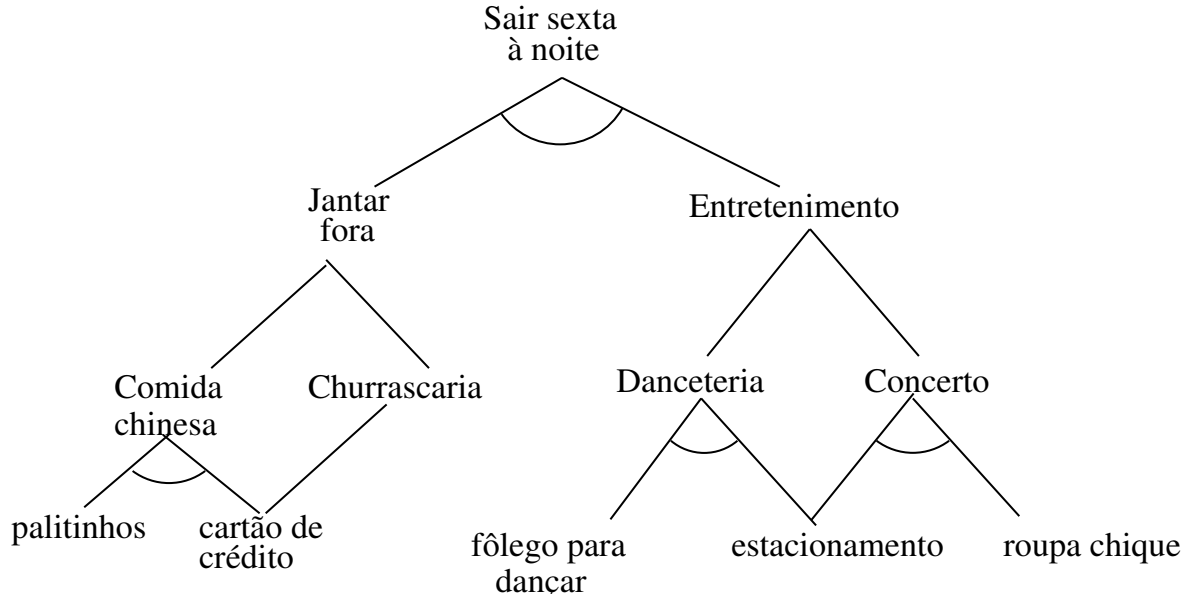

Figura 4.1: Exemplo de grafo *E-OU* puro

4.3.1.2 Regras de transformação atômica de estados

A segunda hipótese simplificadora adotada quando do projeto de um jogador automático de xadrez é a das regras de transformação atômica. Além da representação dos estados de um tabuleiro, para que o grafo de busca possa ser construído é necessário encontrar uma relação entre os estados, e essa relação é expressa pelas regras de transformação atômica.

O jogo de xadrez possui diversas variantes, mas neste trabalho trataremos da variante padrão[1] da FIDE (*Fédération Internationale des Échecs* — Federação Internacional de Xadrez), disponíveis em FIDE (2006). Para cada peça há uma certa quantidade de movimentos plausíveis, e a cada momento do jogo desdobra-se uma grande quantidade de possibilidades de jogadas. A maioria das regras é "estanque", ou seja, independe de outras regras terem sido usadas ou não anteriormente.

Porém, há algumas regras que só podem ser utilizadas em condições específicas, como por exemplo:

- o **roque** (*castling*, em inglês), que consiste em "esconder" o rei atrás de uma das torres, que se deslocará 2 casas em direção ao centro. Para o roque poder ser efetuado, nem o rei nem a torre escolhida podem ter sido movidos anteriormente, e as casas entre essas peças devem estar vazias;

- **captura *en-passant***, que é feita quando, por exemplo, um peão branco é movido 2 casas, e um peão preto está na mesma linha para qual o branco foi movido. O peão preto poderia capturar o peão branco se este tivesse sido movido somente a 1 casa, tendo o direito de efetuar a captura.

Para que ao gerar novas jogadas se consiga determinar a aplicabilidade de regras como essas, é necessário que na representação dos estados (como visto na seção 4.3.1.1) haja a memorização do estado do jogo por meio de variáveis de status, indicando se é possível efetuar movimentos como o roque e a captura *en-passant*. Isso pode ser implementado tanto na representação dos tabuleiros como em variáveis globais visíveis pelo

[1] A escolha de uma variante de xadrez também poderia ser um dos parâmetros do módulo, e é apontada como trabalho futuro.

algoritmo de busca. Além da aplicação de regras, há outras situações que necessitam de memorização de estados anteriores, como a detecção de estados repetidos durante a geração de jogadas, ou a identificação da ciclicidade de jogadas (o que pode caracterizar empate).

Essas medidas para controle de geração de estados resolvem o problema localizadamente, ou seja, são soluções "de software". Porém, elas interferem de maneira negativa na maneira com que o complexo problema de jogar Xadrez foi simplificado. O algoritmo de busca não percorre mais somente o grafo em busca de um caminho, mas deve tomar cuidado para não violar essas "amarras", que influenciam a geração ou não de um novo estado. Como a geração de estados e a busca sobre o grafo são independentes entre si (ou seja, os estados são gerados e depois avaliados), soluções externas como essas têm de ser utilizadas para garantir que o algoritmo de busca não entre em *loop* e que este funcione da maneira correta.

4.3.1.3 Detecção de estado terminal

A aplicação de uma determinada regra pode levar o jogo a um estado a partir do qual não há mais possibilidades, ou seja, o jogo acabou. Assim, o gerador de novos estados deve detectar quando não há mais jogadas plausíveis, ou quando o jogo entrou em ciclo. Os casos básicos de fim de jogo com vitória são o xeque-mate e a configuração de Rei Afogado. Considera-se que o jogo entrou em ciclo se, em 50 lances, uma configuração de tabuleiro se repete três vezes, sem captura de peças, o que caracteriza um empate. No caso de xeque-mate ou de Rei Afogado é simples identificar o estado terminal. No caso da ciclicidade de jogadas, as informações de estado presentes na representação do tabuleiro são imprescindíveis, já que estas vão indicar quantas vezes uma dada seqüência de tabuleiros se repetiu no jogo. Os detalhes das regras de fim de jogo estão definidas no manual de regras de xadrez da FIDE (2006).

4.3.1.4 Requisitos de funcionamento da busca

Tendo representado o universo do xadrez em um grafo de busca, necessitamos de um algoritmo que possa efetuar a busca por uma solução (um caminho) nesse grafo. Tradicionalmente, o algoritmo de busca utilizado em IA para jogos é o Minimax com podas Alfa-Beta (Rich 1983), e este é o escolhido para este módulo, por diversas razões.

Primeiramente, ele é um algoritmo de busca heurística. Guiado por uma função heurística, consegue "driblar" a explosão combinatória típica de jogos como o xadrez e permite obtenção de uma solução em tempo aceitável. Como o presente trabalho propõe a competição entre heurísticas criadas por humanos, é natural que o algoritmo escolhido efetue busca heurística. Outros algoritmos, como o SSS*, podem ser utilizados, mas o Minimax foi escolhido como padrão pela sua simplicidade de compreensão e implementação (sobre escolha de outros algoritmos de busca consulte Seção 4.3.2) .

Além disso, algoritmos como o Minimax promovem uma economia de memória por efetuarem busca em profundidade, ao contrário da busca em largura. Isso é um fator importante dado que a simulação de partidas será executada em modo "off-line", ou seja, automaticamente e sem a interferência de humanos (a não ser por monitoramento externo do andamento da partida). Isso implica o processo executando o algoritmo não poder extrapolar a memória do servidor onde estiver executando, principalmente pelo fato de que pode haver mais de uma partida sendo simulada simultaneamente (conforme o escalonamento do campeonato). Esse servidor é disponível a uma comunidade de jogadores de Xadrez e, exatamente por isso, é necessário haver memória disponível para todos os usuários.

4.3.2 Parametrização do algoritmo

Nas seções anteriores apresentamos os aspectos de simplificação do problema de jogar Xadrez para a implementação de um jogador automático. Nesta seção consideramos os aspectos paramétricos do jogador automático, ou seja, os elementos que permitem a sua flexibilização e interação com os usuários do sistema. Dividiremos os parâmetros em *externos* e *internos*, sendo os primeiros parâmetros definidos externamente à máquina de

busca, e os últimos relativos diretamente com o algoritmo de busca.

4.3.2.1 Parâmetros internos

Estes parâmetros dizem respeito diretamente ao funcionamento interno da máquina de busca. São definidos pelo organizador da competição por meio da ferramenta de gerência de campeonatos, e modificam o comportamento do algoritmo de busca.

Altura da árvore de busca Como falamos anteriormente, o algoritmo de busca percorrerá a árvore de jogos até uma altura h pré-determinada — ou seja, planejará h jogadas à frente. Como é característico de busca heurística, se alterarmos a altura da árvore o algoritmo retornará possivelmente resultados diferentes — pois observou mais ou menos jogadas à frente. Por meio da ferramenta de autoria, o organizador de campeonatos pode alterar esse parâmetro, para tentar capturar diferenças de comportamento das heurísticas dos alunos — por exemplo, se são vulneráveis ao efeito horizonte ou se uma heurística ganhadora se mantém ganhando mesmo planejado menos jogadas à frente. A utilidade desse parâmetro é, portanto, a de avaliar o comportamento da heurística do aprendiz quando esta é aplicada a mais ou menos jogadas planejadas à frente.

Identidade dos competidores Como o objetivo do módulo é efetuar a competição entre heurísticas de dois aprendizes, é necessário que o algoritmo de busca possa selecionar, a partir do banco de dados, a função heurística dos competidores. Como cada um deles estará cadastrado no servidor de xadrez, basta ao algoritmo acessar a base de dados e obter a heurística pertencente ao competidor[2].

Tempo da partida Determinado também pelo organizador do evento, o tempo da partida é um parâmetro interno do algoritmo. Se não for especificado tempo, o algoritmo de busca retornará uma solução usando o tempo que for necessário para tanto. Em partidas com tempo limitado, poder-se-ia usar um relógio externo, ou implementar um

[2]Como o algoritmo tem acesso às duas heurísticas, uma possibilidade seria fazê-lo usar ambas na avaliação dos estados da árvore de busca. Mais detalhes no Capítulo 6.

Minimax que funcionasse em tempo real — orientado a eventos de tempo, usando o tempo que for necessário para tanto.

4.3.2.2 Parâmetros externos

Os parâmetros externos não dizem respeito diretamente ao funcionamento do algoritmo de busca, mas são dados definidos pelo escalonador de partidas e controlam a condução da partida. São eles:

Ordem de início da partida De acordo com o modo de emparelhamento escolhido (p. ex. suíço), é necessário especificar qual dos dois competidores iniciará o jogo (e, por conseguinte, jogará com a cor branca). Definido pelo escalonador de partidas, esse parâmetro é repassado ao algoritmo de busca, e define qual a heurística será aplicada no último nível de maximização na primeira jogada. Além disso, ele já determina a cor das peças de cada oponente.

A própria máquina de busca Se a representação dos estados e as regras de transformação forem bem formalizadas, é possível trocar a própria máquina de busca, possibilitando assim testar o desempenho das heurísticas usando outras metodologias de busca. Naturalmente, essa máquina deve ser de busca heurística. Esse parâmetro, porém, é complicado de ser implementado (devido à representação dos estados e das variáveis globais), e fica como sugestão para uma versão futura deste módulo.

CAPÍTULO 5

ASPECTOS DE IMPLEMENTAÇÃO DO SISTEMA CACAREJE

Neste capítulo discutimos aspectos de implementação do sistema de ensino/aprendizagem alternando competição e colaboração aqui proposto. O sistema se baseia na combinação de diferentes ferramentas que possibilitam a exploração de conceitos de heurísticas de jogos, a submissão das mesmas para competição automática e de ferramentas que viabilizam a assistência humana do instrutor (e colaboradores) na definição desses conceitos. Ressaltamos a originalidade da abordagem de alternância entre atividades de competição e colaboração no ambiente educativo, pouco explorada na literatura.

Apresentamos o sistema CACAREJE (**C**olaboração **A**lternada com **C**ompetição na **A**prendizagem **R**eferenciada por **J**ogos **E**ducativos), que é composto de um núcleo composto por uma base de dados de heurísticas, um comparador genérico de heurísticas (CGDH) e um escalonador de partidas (EGPA). Em torno desse núcleo estão módulos reguladores do acesso a ele, e externamente a ele há as ferramentas de software pelas quais os diversos agentes se comunicam com o sistema. Aqui descrevemos um projeto para o ambiente da comunidade de Xadrez, porém os aspectos aqui apresentados são genéricos e se aplicam, em princípio, a qualquer jogo heurístico de tabuleiro para dois jogadores com alternância de lances.

A implementação foi construída em dois grandes passos: primeiramente, com a construção de um protótipo do comparador de heurísticas para o jogo Mancalla, implementado em Prolog. Resultados concernendo tempo de execução e quantidade de podas efetuadas pelo algoritmo Minimax são apresentados, e deduções sobre a viabilidade para o jogo de Xadrez são efetuadas. Num segundo momento, passamos à implementação em C++ do comparador de heurísticas para o jogo de Xadrez, que está em estágio inicial.

Ao final do capítulo discutimos as limitações do sistema como um todo, bem

como do que precisa ser implementado para estender o protótipo.

5.1 Arquitetura da ferramenta CACAREJE

Nesta seção apresentamos a arquitetura geral do sistema CACAREJE, considerando as interfaces a serem disponibilizadas aos agentes, os módulos responsáveis pela interação das mesmas com o sistema, e os módulos internos principais: o Comparador Genérico de Definições de Heurísticas (CGDH) e o Escalonador Genérico de Partidas Automáticas (EGPA). A arquitetura geral da ferramenta está ilustrada na Figura 5.1.

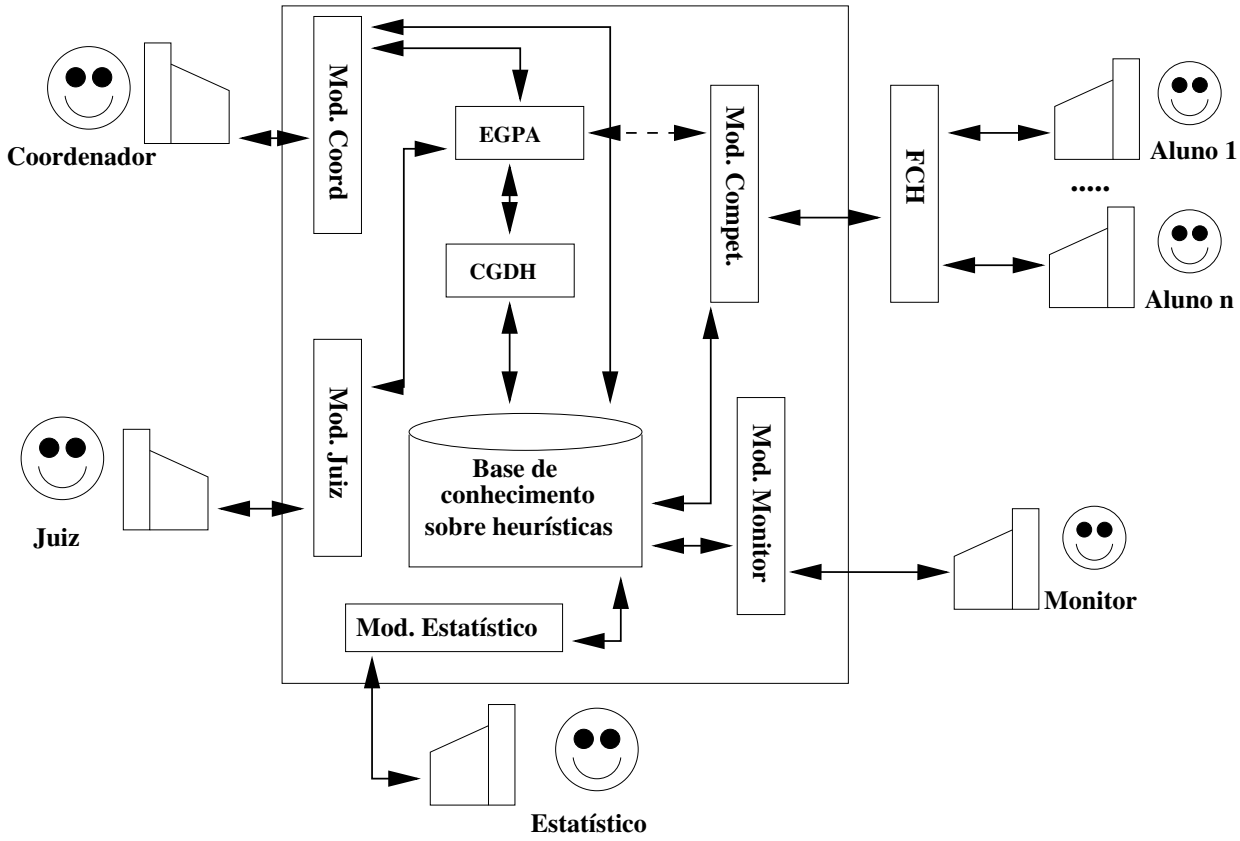

Figura 5.1: Arquitetura geral da ferramenta CACAREJE

Na figura estão ilustradas as interações entre os módulos do sistema, bem como a hierarquia de acessos aos mesmos. O módulo de escalonamento (EGPA) se comunica com o comparador de heurísticas (CGDH) para iniciar as simulações das partidas. O acesso ao CGDH nunca é direto, é feito somente via EGPA, que é acessado pelo coordenador, pelo juiz e (numa versão posterior) pelo aprendiz. Os demais agentes do sistema (aprendiz,

estatístico e monitor) interagem com o sistema por meio de módulos desenvolvidos para os mesmos, e têm acesso somente ao banco de dados de heurísticas.

Nas sub-seções a seguir descreveremos cada um dos módulos e das interfaces apresentados na figura, bem como sua posição na hierarquia de acessos a informações e módulos.

5.1.1 Interfaces dos agentes

Na seção 3.3.1 definimos os papéis desempenhados pelos diversos agentes do sistema (competidores, coordenador, etc). Nesta seção apresentamos a especificação das ferramentas de interface que cada um desses agentes vai ter à sua disposição para atuar no sistema de competição de Xadrez.

5.1.1.1 Interface do aprendiz

A interface do *aprendiz* (ou *competidor*) é o meio pelo qual os participantes do torneio de heurísticas poderão interagir com o sistema. Ela está atualmente em desenvolvimento pelo mestrando Alexandre Feitosa, e faz uso de uma meta-linguagem para definição de heurísticas de Xadrez.

A primeira das atividades disponibilizadas ao usuário é a de criar uma heurística para o jogo de Xadrez. Ela é expressa em uma linguagem formal para definição de heurísticas, que permite codificar aspectos geométricos, algébricos e lógicos na avaliação de um tabuleiro. Para tanto, o usuário poderá fazer uso de uma ferramenta gráfica (estilo planilha) para tratar dos aspectos geométricos, e terá à disposição também um terminal texto para codificar explicitamente as relações algébricas e lógicas do tabuleiro. O produto final dessa fase será um arquivo-texto com o código de uma heurística, que será interpretado e executado pelo comparador de heurísticas (vide seção 5.1.2).

A atividade recém-descrita é complementada pela consulta a heurísticas de outros usuários, ou a heurísticas antigas do próprio usuário. Para tanto, um sistema de permissões e faz necessário, cuja configuração é delegada ao *organizador* da competição e aos monitores que o auxiliam. Entre as permissões disponíveis estão as de leitura e

escrita em uma heurística (essa última só disponível ao próprio autor e ao organizador do campeonato), e as de comentários (um usuário pode, ao visualizar a heurística de outro, adicionar comentários a ela).

As permissões de consulta de heurísticas tornam possível a funcionalidade de *galeria de heurísticas*, que consiste em uma coleção de heurísticas ou desenvolvidas pelo próprio usuário ou consultadas de galerias de outros usuários. Essa galeria é "preenchida" gradualmente com a utilização do sistema por parte do usuário, e permite ter uma visão da evolução do aprendizado do mesmo. A galeria de heurísticas também permite ao usuário executar uma mini-competição entre as heurísticas por ele desenvolvidas, provendo assim um ambiente de teste para as competições oficiais[1].

Remetemos o leitor ao trabalho de mestrado de Alexandre Feitosa (2005) para mais detalhes relativos à interface do aprendiz e do sistema de permissões de acesso à base de dados de heurísticas.

5.1.1.2 Interface do Coordenador

O *coordenador* da competição terá acesso a uma interface diferente da do aprendiz, no que diz respeito aos níveis de permissão de acesso e à interação com o sistema. Diretamente, a interface do coordenador se comunica com o módulo de escalonamento de partidas EGPA (vide seção 5.1.3), para iniciar, agendar, administrar e parar competições de Xadrez. O coordenador pode modificar parâmetros do comparador de heurísticas, indiretamente por meio do módulo EGPA, como altura da árvore de busca, ordem de início das partidas, etc.

Como administrador da competição, o coordenador tem poderes de "super-usuário" no que concerne ao acesso às heurísticas dos competidores. A interface do coordenador permite, então, que este acesse os códigos das heurísticas durante a competição, e possa adicionar comentários às mesmas. Esse acesso é ilustrado pela seta que liga a interface do coordenador à base de dados de heurísticas na Figura 5.1.

[1]Essa funcionalidade consta como trabalho futuro e não necessariamente será desenvolvida numa primeira versão do sistema, como indicado pela seta tracejada na Figura 5.1.

O acesso às heurísticas por parte de outros usuários e as permissões do alcance de atuação dos colaboradores (enxadrista mestre, monitor, estatístico, etc) também são configurados por meio dessa interface. A mudança de *status* de um aluno ou colaborador também é feita por meio dessa interface (por exemplo, dar a um aluno permissões de monitor).

5.1.1.3 Interface do Estatístico

O *estatístico* (vide seção 3.3.2) é o colaborador que auxilia o coordenador na apuração de dados principalmente quantitativos das competições (apesar de considerar dados qualitativos também). Sua interface interage principalmente com a base de dados de heurísticas, dado que o estatístico avalia as definições de heurística dos competidores.

Na interface do estatístico estarão disponíveis dados relativos ao desempenho de um competidor na competição atual bem como o desempenho acumulado das competições em que ele participou. Com isso, o estatístico poderá produzir relatórios para o coordenador com a curva de evolução de um aprendiz. Relatórios como esses permitem identificar o ponto de intervenção para direcionar uma atenção especial a um determinado aprendiz.

De maneira complementar, o estatístico terá à disposição uma ferramenta de *matching*, que permitirá que se estime uma medida de "cópia" de uma heurística. Essa ferramenta compararia uma dada heurística com outras inscritas em campeonatos passados e conseguiria identificar quais porções foram copiadas. Porém, com um número elevado de heurísticas essa tarefa pode se tornar inviavelmente custosa. Uma maneira de contornar esse problema seria manter registros das consultas que um aprendiz fez e permitir que se copiem partes de códigos somente a partir da ferramenta (sem copiar-e-colar em um editor de textos), para poder rastrear as ações de cada aprendiz. Mas é discutível se uma solução dessas não é muito restritiva à liberdade do aprendiz.

Outra atividade interessante para o estatístico seria auferir uma medida de "complexidade" de uma dada heurística, ou seja, o quão detalhadamente ela avalia um tabuleiro. Duplicação de código, contabilização de elementos com pouca relevância, entre outros fatores poderiam ser identificados e até quantificados pelo estatístico, e apresenta-

dos ao aprendiz que criou a heurística. Uma ferramenta de software capaz de desempenhar tal tarefa seria altamente desejável, porém seu desenvolvimento pode ser custoso, além de ter de ser desenvolvida especificamente para o jogo de Xadrez[2].

5.1.1.4 Interface do Monitor

A figura do *monitor* é presente em quase todo ambiente educativo. Um monitor é um aluno com boas notas e conhecimento profundo do domínio em questão, que passa a auxiliar o professor na condução da turma, por ter uma relação mais próxima com os demais alunos e por conhecer bem as dificuldades de aprendizado.

Sua interface é semelhante à do aprendiz, com a diferença de que um usuário monitor tem permissões de acesso (ajustadas pelo coordenador) que o habilitam a abrir e comentar as heurísticas de competidores *durante* a competição, desse modo orientando *in loco* os aprendizes. Adicionalmente um monitor poderia intervir no andamento de um campeonato, suspendendo alguma partida ou criando campeonatos oficiais para desonerar o coordenador dessa tarefa. Porém esse tipo de permissão é discutível, por ser disponibilizada a um aluno. O módulo de apoio ao monitor é responsável por regular essa interação.

5.1.1.5 Interface do Enxadrista Mestre

A atividade do *enxadrista mestre* no ambiente de colaboração e competição é a de trabalhar na especificação e no refinamento da linguagem de definição de heurísticas para Xadrez. Para tanto, ele utilizará uma meta-linguagem genérica de definição de heurísticas para jogos, e por meio de uma ferramenta, vai refinar essa linguagem para a especificidade do Xadrez. Com uma modularização ideal da interface do aprendiz, cada versão nova da linguagem de definição de heurísticas pode ser carregada.

[2]Apesar de que isso não representa uma desvantagem em si.

5.1.1.6 Interface do Juiz

Como a tarefa dos *juízes* é zelar pelo bom andamento das partidas e do torneio, sua interface basicamente lhe permitirá acompanhar as partidas que estiverem sendo simuladas. Serão disponibilizados os controles necessários para poder suspender ou adjudicar uma partida nos casos em que a intervenção humana seja requisitada. A interface do juiz interage com o módulo de apoio ao juiz, que vai se comunicar com o escalonador de partidas. Assim, o juiz tem acesso somente ao que é necessário — as partidas simuladas no momento e o poder de intervir nelas —, respeitando a hierarquia de acessos da Figura 5.1. Ao adjudicar uma partida, o juiz poderá escrever um comentário explicando o porquê de sua decisão, e essa informação ficará armazenada juntamente com a partida.

5.1.2 Módulo CGDH

O módulo CGDH (**C**omparador **G**enérico de **D**efinições de **H**eurísticas) é o módulo do sistema que se encarrega de efetuar a comparação de material com definições de heurísticas por meio da simulação de partidas de Xadrez. Esse material é puramente voltado à avaliação da vantagem material e/ou posicional de peças em uma configuração de tabuleiro, e é utilizado como função de avaliação estática do algoritmo Minimax.

A genericidade do CGDH se refere aos diversos parâmetros que permitem que ele seja configurável e adaptável a várias situações e jogos diferentes. Esses parâmetros são explicados com detalhe na seção 4.3.2, e são sumarizados na lista abaixo:

- altura da árvore de busca;
- identidade dos competidores;
- tempo de relógio de uma partida;
- ordem de início da partida;
- modalidade de partida.

Com esses parâmetros, o comparador genérico de heurísticas se torna maleável para poder simular partidas com diversas condições, como: planejamento de mais ou

menos jogadas à frente, limitação de tempo de relógio (que leva a uma interrupção na busca e retorno do melhor resultado obtido até então), a própria modalidade da partida (que pode fazer com que uma heurística que seja boa mas que demande muito tempo perca uma partida da modalidade relâmpago, por exemplo), etc. Numa parametrização ideal, o comparador de heurísticas poderia trocar até a máquina de busca, porém como discutido na seção 4.3.2 essa opção será considerada em um trabalho futuro.

Na seção supracitada nos referimos também ao problema da representação dos estados do jogo e do registro de elementos globais entre um lance e outro. Para o jogo de Xadrez esses elementos globais se tornam indispensáveis para a simulação de partidas, pois há movimentos (como o roque e a captura *en-passant*) que só podem ser efetuados se certos movimentos tiverem sido efetuados anteriormente. Essa necessidade pode ser contornada por meio do acréscimo de variáveis na representação do estado do tabuleiro para indicar se há condição para o roque (nem o rei nem a torre podem ter sido movidos) ou para a captura *en-passant* (o peão tem que ter se movido duas casas para a frente na jogada imediatamente anterior).

5.1.2.1 Ciclicidade

Um outro elemento global que deve ser armazenado é a indicação se há *ciclicidade* de jogadas, ou seja, se os jogadores fazem os mesmos lances repetidamente e voltam a uma mesma configuração. No Xadrez a ciclicidade é caracterizada quando, depois de alguns lances, se retorna três vezes a uma mesma configuração de tabuleiro, o que qualifica um empate. Para rastrear isso não bastaria acrescentar uma variável à representação dos estados, mas sim criar uma estrutura de dados persistente entre chamadas do algoritmo Minimax que indique se há ciclo no ramo da árvore de busca explorado no momento.

Porém, quando consideramos a simulação automática de uma partida de Xadrez, o tratamento da ciclicidade não é simples, dado que optar por um empate depende de vários fatores e um jogador humano tem que fazer essa escolha conscientemente. Ao tornar essa decisão automática temos que tomar cuidado para não fazermos o algoritmo tomar um caminho que um humano não tomaria. Por exemplo, quando um jogador está

com vantagem e vê o risco de que o oponente force o jogo para um empate, ele vai fazer de tudo para evitar esse empate, que é desvantajoso. Alternativamente, um jogador que estiver em desvantagem tentará conduzir a partida para um empate, evitando lances mais arriscados que o levariam a uma derrota certa.

Uma proposta para solucionar este problema é inserir, juntamente ao teste de estado terminal, um outro teste que avalia a condição de jogada cíclica. Primeiramente, ele testaria se o tabuleiro recém-gerado leva o jogo a uma situação de empate (caracteriza terceiro ciclo). Caso positivo, o algoritmo suspende a busca no ramo da árvore corrente, e aplica uma nova função, h', cujo valor é composto da aplicação da função heurística h e de uma modulação que indique que o tabuleiro caracteriza ciclo. A definição de h' segue:

$$h'(Tab, Jogador) \rightarrow \{+\infty, -\infty\} :$$

$$h(Tab, Jogador) + mod$$

onde

$$mod = \text{modulação de ciclagem tripla}$$

e

$$Jogador = Oponente, Maquina$$

Essa função vai retornar um valor dentre $+\infty$ e $-\infty$, definido pela chamada modulação de ciclagem tripla, e que será atribuído ao tabuleiro no nível corrente da árvore de busca. Esse valor será transportado para os níveis acima com o sinal devidamente trocado, e o ramo correspondente será podado na próxima aplicação da poda alfa ou beta. A escolha dos valores de h' segue a idéia de que quando o jogador está em vantagem, deve-e evitar o empate de qualquer maneira, e que quando em desvantagem, deve-se tentar obter o empate. Na Tabela 5.1 colocamos os valores para a função h' de acordo com as situações de vantagem e desvantagem.

Temos, então, quatro casos distintos (todos ocorrendo quando se caracteriza um ciclo triplo):

- nível de *maximização* e $h \geq 0$: neste caso, o jogador estará em vantagem, e o empate

Valor de h Nível	≥ 0 (vantagem)	< 0 (desvantagem)
MAX	$h' \leftarrow -\infty$	$h' \leftarrow +\infty$
MIN	$h' \leftarrow +\infty$	$h' \leftarrow -\infty$

Tabela 5.1: Valor de h' em caso de ciclo

deve ser evitado. O pior valor possível será atribuído ao tabuleiro;

- nível de *maximização* e $h < 0$: neste caso, o jogador estará em desvantagem, e para o adversário o empate é ruim. O melhor valor possível será atribuído ao tabuleiro;

- nível de *minimização* e $h \geq 0$: o jogador estará em vantagem, mas como este é um nível de minimização, o valor $+\infty$ será atribuído para o tabuleiro, pois será propagado com sinal invertido para o nível acima;

- nível de *minimização* e $h < 0$: o jogador estará em desvantagem, mas como este é um nível de minimização, o valor $-\infty$ é atribuído para o tabuleiro, pois será propagado com sinal invertido para o nível acima;

O plano de recomendação delineado acima é proposto dado o caráter de prototipação dos conceitos e ferramentas apresentados neste trabalho. Essa polarização para os valores $+\infty$ e $-\infty$ é muito extrema e faz o Minimax se comportar como se encontrasse um tabuleiro de fim de jogo. Uma maneira mais apropriada seria adotar, ao invés desses extremos, valores intermediários, porém altos, que refletiriam o caráter altamente vantajoso ou desvantajoso do empate.

Note-se que numa competição em que humanos tomam essa decisão esse problema é minimizado, porém ao projetarmos um comparador genérico que utilize busca heurística não podemos contar com fatores de incerteza (como aleatoriedade e distribuição probabilística) para tomar uma decisão como essas. Esses fatores de incerteza acrescentariam complicação desnecessária ao algoritmo e seriam difíceis de implementar, e implicariam abandonar as hipóteses simplificadoras (Seção 4.1.1) assumidas de início.

5.1.3 Módulo EGPA

O módulo EGPA (**E**scalonador **G**enérico de **P**artidas **A**utomáticas) é o responsável por efetuar o pareamento dos competidores e a formação das chaves do torneio. Além disso, é o módulo com o qual o coordenador e o juiz interagem diretamente, e é por meio dele que são alterados os parâmetros de funcionamento do CGDH e são feitas as intervenções nas partidas (adjudicações).

Existem diversas modalidades de escalonamento para partidas entre 2 jogadores (ou times), mas a versão preliminar do escalonador terá disponíveis o pareamento suíço e o pareamento simples (ou "todos contra todos"). Há também diversas maneiras de condução das partidas (*wild*, sem limite de tempo, relâmpago), mas nessa versão inicial o escalonador contará somente com a modalidade sem limite de tempo, pelo fato de usarmos o algoritmo Minimax puro[3].

O escalonador estará integrado à infra-estrutura existente no servidor de Xadrez do CEX, no que concerne à realização e armazenamento de partidas. Dessa maneira, as partidas realizadas no servidor poderão ser assistidas como já é feito, utilizando as ferramentas *xboard* ou *winboard*. Na figura 5.1 pode-se ver a hierarquia de acesso ao escalonador, restrita somente ao coordenador e ao juiz. O escalonador estará também disponível aos aprendizes para fazerem mini-competições com sua galeria de heurísticas.

5.1.4 Módulos assistentes

Nesta seção discutimos os aspectos relativos aos módulos que vão assistir a ação dos diversos agentes no sistema. Esses módulos vão regular o acesso das interfaces dos agentes somente às partes às quais estes têm permissão. Deve-se assinalar que numa próxima versão do sistema esses módulos poderiam ser substituídos por um módulo único, que controlaria os acessos de cada usuário de acordo com suas permissões. Porém, mantemos essa recomendação de arquitetura pelo fato de as ferramentas dos diversos agentes serem diferentes e se conectarem de maneiras diferentes ao servidor.

[3]Para introduzir a noção de tempo, pode-se usar o chamado Minimax em tempo real — *Real-time Minimax* (Koenig 2001) — mas isso foge ao escopo deste trabalho.

Módulo assistente do aprendiz Este módulo vai permitir ao aprendiz entrar no sistema do servidor de xadrez, carregar suas definições de heurísticas e editá-las, ou criar novas e armazenar na base de dados.

Módulo assistente do coordenador Ao coordenador o acesso é permitido diretamente ao escalonador e à base de dados de heurísticas. O acesso ao CGDH não é feito diretamente, mas sim via EGPA. Este módulo garante essa restrição de acesso.

Módulo assistente do juiz O juiz só necessita acessar o escalonador de partidas, pois sua atividade se restringe a zelar pelo bom andamento do campeonato. Os comentários referentes a adjudicações são repassados por este módulo ao escalonador, para que sejam armazenados juntamente com a partida.

Módulo assistente do monitor Este módulo basicamente possibilita ao monitor o acesso às heurísticas dos competidores durante o campeonato, de acordo com as permissões concedidas a ele pelo coordenador da competição.

Módulo assistente do enxadrista mestre O módulo do enxadrista mestre tem uma funcionalidade um pouco diferente da dos demais módulos, pois esse agente vai interagir diretamente com a linguagem de definição de heurísticas, alterando o "backend" da interface do aprendiz. Para isso, não basta ter o controle de permissões, mas também um acesso exclusivo a um sub-módulo que contém a definição da linguagem. Esse acesso, porém, requer uma modularização da interface do aprendiz, que ainda não está prevista para o protótipo desenvolvido por Alexandre Feitosa (2005).

5.2 Um protótipo do sistema CACAREJE em Prolog

Nas seções anteriores discutimos aspectos metodológicos e filosóficos da implementação do sistema CACAREJE, no que concerne principalmente à escolha do algoritmo de busca, definição de interfaces e módulos e a comunicação entre eles. Nesta seção apresentamos a implementação de um protótipo para o comparador de heurísticas e discutimos

resultados de execução e problemas de implementação.

O primeiro passo para viabilizar a comparação automática genérica de duas heurísticas de jogos é a implementação de um algoritmo de busca heurística para jogos. Desse modo, problemas elementares da implementação de um comparador genérico puderam ser verificados, principalmente no que concerne à sua parametrização. O protótipo foi desenvolvido na linguagem Prolog utilizando o interpretador `swi-prolog`, e teve como domínio de teste o jogo *Mancalla*, um jogo com menor explosão combinatória de jogadas e de fácil codificação[4].

5.2.1 O comparador genérico

O algoritmo escolhido para ser implementado no CGDH foi o algoritmo *Minimax* na sua variação com *podas alfa-beta* (Rich 1983). Esse algoritmo percorre uma árvore de busca fazendo uma busca em *profundidade*, e aplica as chamadas podas alfa-beta para eliminar ramos não promissores da árvore de busca, assim diminuindo o tempo total de execução.

Como foi explicado no Capítulo 4, o algoritmo Minimax com podas alfa-beta foi escolhido com base na reduzida quantidade de memória requerida na sua execução, considerando o compromisso *memória-tempo*. Isso se justifica pelo fato de que, no futuro, o comparador genérico será executado em um servidor de Xadrez acessado via Internet por muitos usuários, e deve-se garantir que haja memória suficiente para todos. O Minimax com podas alfa-beta requer, no seu máximo uso de memória, somente o espaço necessário para armazenar um ramo inteiro da árvore de busca com altura h (Baudet 1978, Rich 1983), parâmetro esse constante e conhecido durante uma partida inteira. Isso, acrescido à sua simplicidade de implementação, faz do Minimax a melhor opção de algoritmo de busca.

Na primeira implementação do protótipo, a título de experimento, não consideramos a parametrização do mesmo, considerando somente uma heurística e altura fixa. Na versão apresentada neste trabalho, o protótipo considera os seguintes parâmetros:

[4]As regras da variante do Mancalla adotada são apresentadas no Apêndice A

- altura da árvore: esse valor é passado como parâmetro para o predicado que vai executar a simulação. Para armazená-lo, o fato do prolog `altura_max(n)` é utilizado;

- identidade dos competidores: antes de executar o comparador, cada competidor deve ter um predicado único no qual estará definida sua heurística. Antes de executar o comparador deve-se acrescentar fatos associando cada predicado a um "lado" do tabuleiro.

Desse modo, o comparador executa uma instância do algoritmo Minimax e seleciona, alternadamente, a função de avaliação estática de cada competidor. São executadas tantas rodadas quantas necessárias forem até um dos dois sair vencedor. Ao final, o vencedor é apontado e a cada lance a jogada efetuada é mostrada na tela.

Esses parâmetros são ajustados manualmente, na chamada do predicado de execução da simulação. Na versão definitiva esses parâmetros serão definidos por meio de uma interface que se comunicará com o escalonador de partidas, e não diretamente com o CGDH.

5.2.1.1 Representação dos estados

Aqui falamos brevemente sobre a escolha da representação do tabuleiro, e a influência que teve no desenvolvimento do comparador.

Para o jogo de Mancalla, foram utilizadas listas dinâmicas, da forma `[[PoteA], [PecasA, PecasB], [PoteB]]`, onde `PoteA e PoteB` representam, respectivamente, a quantidade de peças capturadas pelo jogador A e pelo B, e `PecasA e PecasB` representam as casas que ficam do lado do jogador A e do B, respectivamente. Como o jogo Mancalla é simples, essa representação se mostrou prática e eficiente, tanto para extração de informações quanto para a construção do gerador de novos estados.

Como o Mancalla é um jogo em que não há repetição de configurações de tabuleiro, variáveis globais para registrar ciclicidade de jogadas não se fizeram necessárias. Além disso, no Mancalla cada jogada independe das jogadas efetuadas anteriormente, e se baseia tão-somente na configuração atual do tabuleiro, o que tornou desnecessárias

variáveis para registrar possíveis dependências entre lances. Para o Xadrez, porém, essas variáveis se tornam indispensáveis, pois algumas jogadas dependem de certas condições da partida, e jogadas cíclicas têm que ser detectadas (para caracterizar empate). Desse modo, para o Xadrez a representação do estado deve ser incrementada com essas sentinelas globais para guiar o algoritmo de busca (Como discutido na seção 4.3.1).

5.2.2 Três heurísticas para Mancalla

Para testar o funcionamento do protótipo, foram codificadas (manualmente) três heurísticas, com diferentes níveis de detalhe e maneiras de pontuar um tabuleiro. Consideraremos o tabuleiro da Figura 5.2 para ilustrar a avaliação feita pelas funções heurísticas.

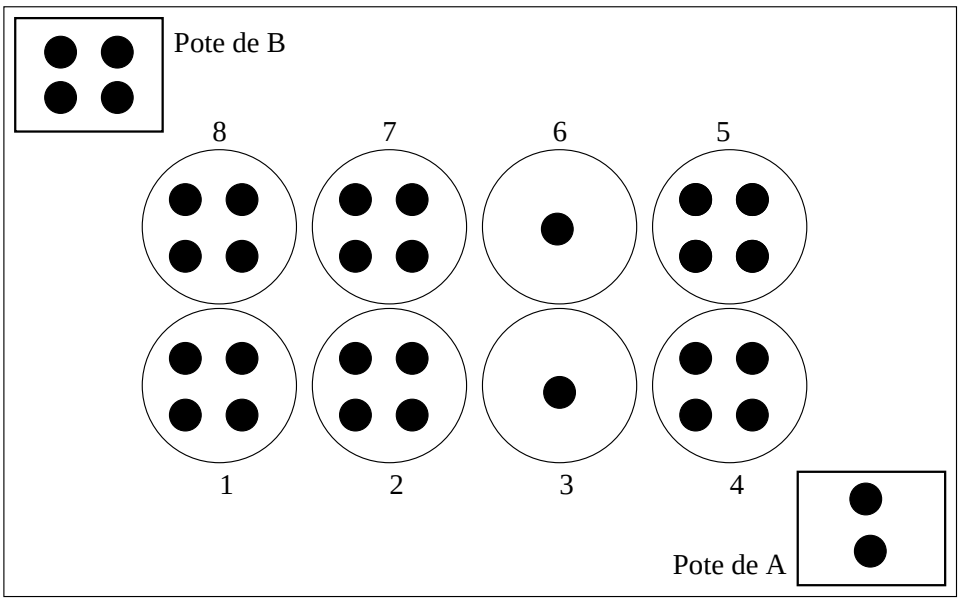

Figura 5.2: Configuração de tabuleiro do jogo Mancalla

A primeira, do jogador "Daniel", considera os seguintes elementos: (i) a diferença entre a quantidade de peças capturadas pelo jogador e pelo adversário; (ii) a quantidade de casas com uma pedra. A diferença de peças leva peso de 20%, enquanto que a quantidade de casas com somente 1 pedra ganha peso maior, 80%. Quanto mais casas com somente uma peça houver no tabuleiro, maiores são as chances de captura, e maior o valor do tabuleiro.

A segunda, do jogador "João", é mais simples e pontua um tabuleiro de acordo

com a diferença entre a quantidade de peças capturadas pelos jogadores. A diferença sempre é calculada como sendo `PeçasJogador -- PeçasAdversário`. A terceira heurística, da jogadora "Maria", pontua um tabuleiro de acordo com a quantidade de peças remanescentes nas casas que estão do seu lado no tabuleiro. Quanto mais peças houver no lado do tabuleiro pertencente a "Maria", maior valor será atribuído a ele.

Considerando o tabuleiro da figura 5.2, os valores atribuídos pelas heurísticas de "Daniel", "João" e "Maria", respectivamente, foram -0.4, -2 e 13. Como se pode notar, a primeira heurística dá um valor mais refinado que as outras, pois no tabuleiro da figura a situação está semelhante para ambos os jogadores. Porém como há uma casa com somente uma peça no lado do jogador e como a diferença entre as peças capturadas é desvantajosa para o jogador, a heurística pontua o tabuleiro como ligeiramente desvantajoso. Na heurística de "João", só o que foi contabilizado foi a diferença de peças capturadas, e o tabuleiro é considerado desvantajoso para o jogador. Por fim, para "Maria" o tabuleiro é bastante vantajoso por ter bastantes peças no seu lado — ou seja, mais possibilidades de jogadas.

5.2.3 Algumas comparações de execução

Com as três simples heurísticas apresentadas na seção anterior já podemos simular algumas partidas e obter alguns resultados de execução. Comparações de tempo e de número de podas podem ser observadas nas Tabelas 5.2 e 5.3.

Na Tabela 5.2 são mostrados os tempos de execução de partidas com os seis pareamentos possíveis das três heurísticas (Daniel×Maria, Daniel×João e João×Maria, e as respectivas partidas invertendo o jogador que inicia a partida). Na tabela, a letra **a** indica que o jogador "a" foi o que ganhou a partida, e a letra **b** indica que foi o jogador "b" que ganhou. Na tabela, o nome no cabeçalho da coluna indica qual dos jogadores iniciou a partida e os resultados são ordenados por ordem crescente da altura da árvore de busca utilizada.

Uma peculiaridade do jogo de Mancalla é que o jogador que inicia a partida tem maiores chances de vencer, pois sua escolha de jogada pode dificultar e muito a

Partida h	Daniel	Maria	Daniel	João	João	Maria
1	**a** 0.01	**a** 0.01	**a** 0.01	**a** 0.01	**b** 0.01	**b** 0.01
3	**a** 0.02	**b** 0.01	**a** 0.01	**a** 0.02	**b** 0.01	**b** 0.01
5	**a** 0.07	**a** 0.07	**b** 0.08	**a** 0.09	**a** 0.06	**b** 0.07
7	**b** 0.31	**b** 0.44	**a** 0.45	**b** 0.43	**b** 0.3	**b** 0.42
9	**a** 1.54	**b** 1.59	**a** 1.52	**b** 1.58	**a** 1.5	**b** 1.51
11	**a** 4.55	**b** 4.75	**a** 4.54	**b** 4.78	**a** 4.43	**b** 4.52
13	**a** 11.54	**b** 12.01	**a** 11.56	**b** 12.08	**a** 11.28	**b** 11.37
15	**a** 24.74	**b** 25.82	**a** 24.77	**b** 25.96	**a** 24.31	**b** 24.57

Tabela 5.2: Resultados de execução (tempo em segundos).

Partida h	Daniel	Maria	Daniel	João	João	Maria
1	0	0	0	0	0	0
3	25	21	28	16	18	20
5	129	166	176	187	127	138
7	549	724	800	743	544	760
9	2040	2021	2006	2010	2027	2004
11	4071	4105	4077	4088	4052	4075
13	7249	7293	7219	7298	7228	7193
15	10491	10561	10434	10560	10471	10415

Tabela 5.3: Resultados de execução (quantidade de podas).

ação do adversário. Isso é mostrado pelo fato de a heurística de "Maria", apesar de ser muito simples e considerar poucos elementos do tabuleiro, ter ganho várias das partidas disputadas, principalmente as partidas em que ela foi a primeira a jogar. Note-se também que esse comportamento se reproduz com alturas da árvore de busca maiores.

Na Tabela 5.3 apresentamos a quantidade de podas alfa ou beta efetuadas pelo algoritmo Minimax durante a simulação da partida. A ordenação dos resultados é a mesma da Tabela 5.3. Como era de se esperar, quanto maior a altura da árvore de busca, mais ramos não-promissores são eliminados.

5.2.4 Escala de valores em uma competição simulada

Considerando os valores de tempo de execução apresentados na seção anterior, podemos fazer uma projeção estimando o tempo necessário para simular uma competição

pessoas h	10	20	30	40	50	100	150	200
1	0.19	0.39	0.59	0.79	0.99	1.99	2.99	3.99
3	0.25	0.51	0.78	1.053	1.32	2.65	3.98	5.31
5	1.39	2.86	4.32666	5.79333	7.26	14.59	21.92	29.26
7	7.44	15.27	23.11	30.94	38.77	77.9417	117.11	156.27
9	29.26	60.06	90.86	121.66	152.46	306.46	460.46	614.46
11	87.30	179.20	271.10	363.01	454.90	914.40	1373.9	1833.4
13	221.16	453.96	686.76	919.56	1152.36	2316.36	3480.36	4644.36
15	475.538	976.104	1476.67	1977.24	2477.8	4980.63	7483.46	9986.29

Tabela 5.4: Projeções de tempo (em segundos) para simulação de campeonatos

com um número maior de heurísticas inscritas, considerando o jogo Mancalla. Tirando a média dos tempos obtidos na simulação acima para cada altura da árvore de busca, e considerando um pareamento dois-a-dois simples (ou seja, uma árvore binária com eliminatória simples), temos a Tabela 5.4, indicando o tempo (em segundos) necessários para a simulação de torneios maiores. Num torneio com n competidores e eliminatória simples realizam-se no máximo $\lceil 2n - 1 \rceil$ partidas.

Dada a simplicidade do jogo Mancalla e da sua limitada explosão combinatória, podemos concluir que uma competição com um número grande de participantes pode ser simulada em poucos minutos (no caso com 200 competidores e árvore de busca de altura 15, a simulação levaria pouco mais de 2 horas e meia). Considerando que muitas partidas podem ser simuladas simultaneamente com outras, esse tempo pode ser reduzido mais ainda (para cerca de 1 hora).

O jogo de Xadrez possui um fator de desdobramento ("branching factor") de uma média de 30 jogadas plausíveis a cada momento da partida. Considerando árvores de busca não muito altas (altura 15, como nos testes) e também levando em conta que várias partidas podem ser executadas simultaneamente, uma competição de Xadrez pode ser simulada em algumas horas. Competições com 200 participantes são comuns na comunidade de Xadrez, mas são competições presenciais; as competições no servidor de Xadrez serão menores, restritas inicialmente a uma turma (aprox. 50 alunos).

5.3 Implementação do sistema CACAREJE no CEX

Na seção anterior relatamos o desenvolvimento de um protótipo para o sistema CACAREJE. Até o momento a prioridade foi para o desenvolvimento do comparador de heurísticas, com sua versão preliminar desenvolvida em Prolog. A versão do comparador que vai ser integrada ao servidor de Xadrez está atualmente em desenvolvimento na linguagem C++. A implementação está sendo conduzida por Antonio Hobmeir e Tiago Sak, na forma de um trabalho de conclusão de curso de graduação, co-orientado pelo autor desta dissertação.

A interface do aprendiz está em desenvolvimento em Java como a dissertação de mestrado de Alexandre Feitosa, a ser defendida em breve. É uma versão de protótipo, mas que permitirá que se experimente a codificação de heurísticas em uma linguagem formal por meio de uma ferramenta gráfica.

Os demais módulos serão desenvolvidos na linguagem C++ para integração com o servidor de Xadrez do CEX, e são apontados como trabalhos futuros no capítulo 6.

5.3.1 Limitações da ferramenta

Neste capítulo apresentamos os requisitos conceituais para a implementação do sistema CACAREJE, e reportamos os resultados da implementação de um protótipo para o comparador de heurísticas CGDH. Como o sistema ainda está em fase de protótipo e foi desenvolvido em Prolog, ele não está integrado ao servidor de CEX. Além disso, ele tem que ser configurado e executado manualmente, e as heurísticas foram codificadas manualmente.

CAPÍTULO 6

CONCLUSÃO E TRABALHOS FUTUROS

Neste trabalho foram apresentados os conceitos e ferramentas de software para permitir a alternância entre atividades competitivas e colaborativas, tendo como pano de fundo a codificação de táticas de Xadrez em uma linguagem formal de definição de heurísticas. Uma breve revisão sobre as pesquisas efetuadas na área revela que há uma carência de trabalhos estudando tanto a combinação de atividades competitivas e colaborativas (em parte por uma crença de muitos educadores de que competição é prejudicial ao ambiente de sala de aula) quanto a aplicação de jogos como ambiente de aplicação de conceitos aprendidos formalmente em sala de aula.

A contribuição da presente dissertação se fez em duas frentes. Primeiro, acreditamos que a alternância entre atividades competitivas e colaborativas, auxiliada por ferramentas de software, pode trazer muitos benefícios para o ensino de Xadrez (e de outras áreas em geral). Apresentamos um plano de trabalho e a concepção da arquitetura de um sistema facilitador dessa alternância, que estará integrado ao servidor de Xadrez do CEX. Esta dissertação propõe-se também a servir como base a trabalhos posteriores (de mestrado e de graduação) que venham a implementar os módulos do sistema e torná-lo utilizável pela comunidade enxadrística.

Em segundo lugar, apresentamos a implementação do protótipo de um comparador genérico de heurísticas, que serviu para identificar as dificuldades que surgem quando se tenta parametrizar e maleabilizar um algoritmo de busca. Apesar de o protótipo ser limitado, os resultados de simulação permitiram projetar uma escala de valores de tempo para o jogo de Xadrez e competições com centenas de competidores. O trabalho também auxiliou o início da implementação da versão definitiva do comparador em C++ (que será conduzida como trabalho de conclusão de curso, a ser co-orientado pelo autor desta dissertação). O trabalho conjunto com Alexandre Feitosa (2005) permitiu definir os rumos

de pesquisa e a divisão de trabalho na realização desta dissertação e do projeto como um todo.

6.1 Trabalhos futuros

Como metas futuras de pesquisa a partir deste trabalho, propomos principalmente duas linhas de investigação. A primeira delas está relacionada com o ambiente de ensino/aprendizagem do modo como é proposto aqui. Principalmente, propomos estudos mais aprofundados de como seres humanos aprendem conceitos de diferentes áreas do conhecimento, e de como esses conceitos podem ser expressos como heurísticas de jogos. Logicamente não seria considerado somente o jogo de Xadrez, mas outros jogos que demandem outros tipos de conhecimento (como Go, que requer orientação espacial e parcimônia, ou jogos que envolvam conhecimento lingüístico, por exemplo).

Além disso, também devem ser analisadas soluções bem-sucedidas de problemas (no caso, efetuar uma jogada ou vencer o jogo), soluções essas que são atingidas de forma totalmente atípica (sem muita "disciplina"). A partir destas soluções corretas e atípicas, poder-se-ia incluir no sistema (mais especificamente num módulo pedagógico) mais um mecanismo genérico, que permita identificar e gerar de explicações de como a solução foi obtida. Por exemplo, explicar como as componentes de uma dada heurística avaliaram um dado tabuleiro, ou, num nível mais alto, explicar a situação de um tabuleiro (qual jogador está em vantagem, quais peças estão ameaçadas, etc).

A segunda linha de investigação está relacionada principalmente com os conceitos e a ferramenta de autoria para definição de heurísticas. Com a contribuição de enxadristas-mestras, a linguagem atual pode ser refinada e aprimorada, permitindo definições mais precisas das heurísticas e que a linguagem seja melhor definida formalmente. Uma expansão possível da linguagem e das ferramentas de autoria poderia incluir principalmente recursos para se definir *contra-exemplos* de componentes heurísticas — dado que em muitas situações é mais fácil definir o que é desvantajoso do que dizer o que é vantajoso (por exemplo, quando esse último requer um nível de detalhe muito grande).

Além disso, este trabalho propõe idéias e especificações para diversas ferramen-

tas de software que virão a facilitar a alternância. A seguir listamos as que devem ser implementadas em trabalhos vindouros.

- um escalonador de partidas de Xadrez, bem como a integração deste e do comparador genérico de heurísticas ao servidor de Xadrez do CEX;
- interface e módulo de acesso do coordenador, para interagir com o escalonador de partidas;
- ferramenta para mostrar a atuação de uma heurística "ao vivo", ao avaliar um tabuleiro;
- ferramentas para o *estatístico*: para acompanhar o desempenho de uma dada heurística ao longo de um campeonato, avaliar o grau de plágio e de complexidade de uma função heurística;
- um sistema centralizado de permissões (faz parte de uma nova versão da arquitetura do sistema), que substituiria os diversos módulos para os agentes.

APÊNDICE A

REGRAS DA VARIANTE DE MANCALLA ADOTADA

No jogo chamado *Mancalla*, dois competidores se enfrentam por meio de um tabuleiro constituído por oito casas de forma circular e dois potes usados como depósitos de peças capturadas durante os lances. A Figura A.1 apresenta o esquema do referido tabuleiro em sua posição de início de jogo. Nela, cada uma das oito casas possui quatro peças iguais, ficando os dois potes totalmente vazios. As casas numeradas de 1 a 4 são consideradas do lado do jogador **"A."** As outras casas são consideradas como estando do lado do jogador **"B."**

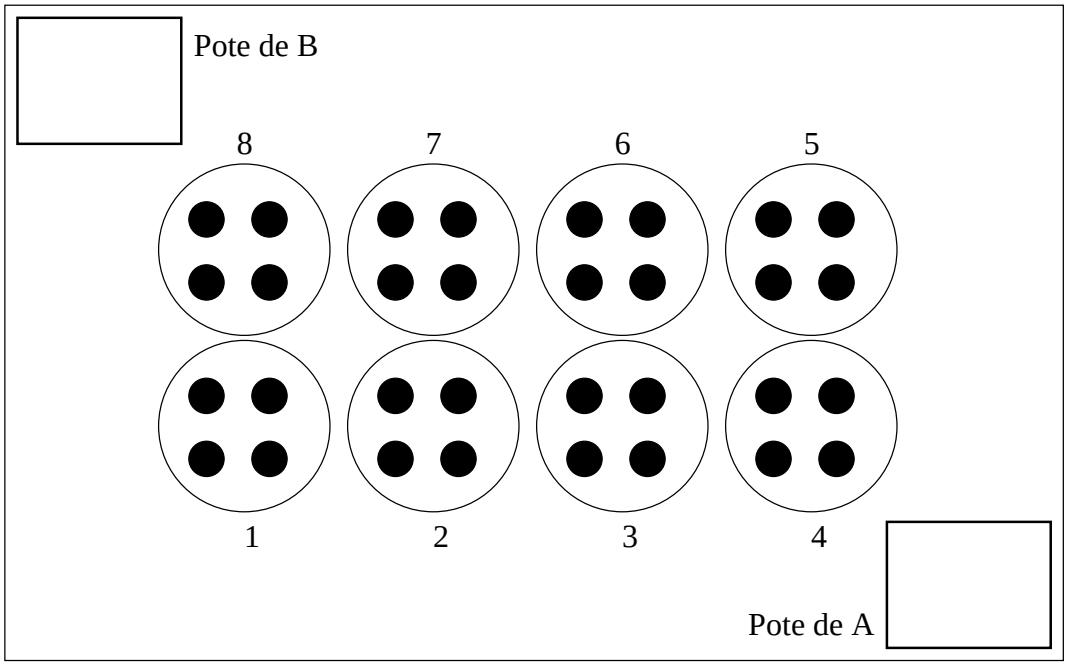

Figura A.1: Tabuleiro de início do jogo Mancala

Os jogadores se alternam a cada lance. Para efeturar um lance, um jogador pega todas as peças de qualquer uma das casas do seu lado e as distribui, uma a uma, no sentido anti-horário, nas casas seguintes àquela de onde as peças foram retiradas. Se, durante a distribuição, o jogador passar por uma casa que possui apenas uma peça, essa peça é

capturada pelo jogador juntamente com a peça que estiver sendo distribuída. Ganha o jogo quem capturar doze ou mais peças.

Nas Figuras A.2 e A.3, por exemplo, vemos o que acontece quando o jogador "A" escolhe a casa 2 para distribuir suas peças.

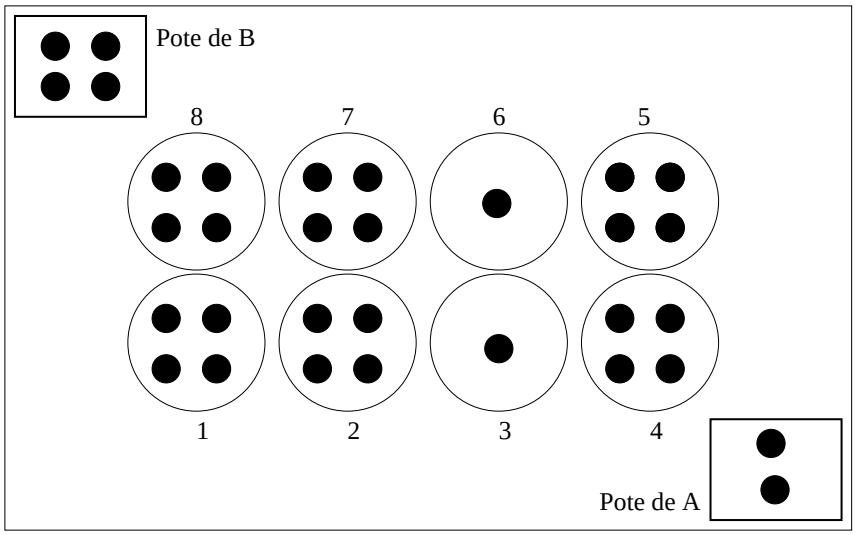

Figura A.2: Tabuleiro antes de efetuada a jogada.

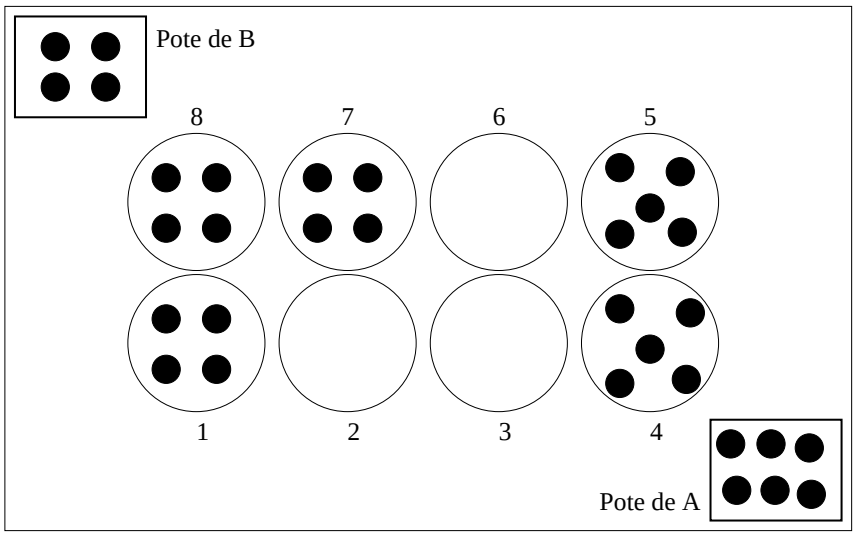

Figura A.3: Tabuleiro depois de efetuada a jogada.

REFERÊNCIAS BIBLIOGRÁFICAS

BAUDET, G. M. (1978). An analysis of the full alpha-beta pruning algorithm. In *STOC '78: Proceedings of the tenth annual ACM symposium on Theory of computing*, p. 296–313, New York, NY, USA. ACM Press.

BERLINER, H. (1979). The b* tree search algorithm: A best-first proof procedure. **Artificial Intelligence**, 12(1):23–40.

BOFF, E. (2000). Construindo um ambiente de ensino-aprendizagem cooperativo: uma experiência interdisciplinar. In *Anais do XI SBIE – Simpósio Brasileiro de Informática na Educação*, p. 112–19, Maceió/AL.

BORGES, M. A. F. (2001). Analysing interaction in a collaborative game: a case study. In *Anais do XII SBIE – Simpósio Brasileiro de Informática na Educação*, p. 185–93, Vitóriaa/ES.

BURTON, M.; BRNA, P., e PILKINGTON, R. (2000). Clarissa: A Laboratory for the Modelling of Collaboration. **International Journal of Artificial Intelligence in Education**, 11:79–105.

BURTON, R. R. e BROWN, J. S. (1977). An tutoring and student modelling paradigm for gaming environments. **SIGCSE Bulletin**, 8:236–246.

BURTON, R. R. e BROWN, J. S. (1979). An investigation of computer coaching for informal learning activities. **International Journal of Man-Machine Studies**, 11:5–24.

DA Cruz Neto, G. G.; GOMES, A. S., e TEDESCO, P. (2003). Elicitação de Requisitos de Sistemas Colaborativos de Aprendizagem Centrada na Atividade de Grupo. In *Anais do XIV SBIE – Simpósio Brasileiro de Informática na Educação*, p. 336–45, NCE-UFRJ.

DE BRUIN, A. e PIJLS, W. (1996). Trends in Game Tree Search. In *SOFSEM: Theory and Practice of Informatics, 23rd Seminar on Current Trends in Theory and Practice of Informatics*, v. 1175 of *Lecture Notes in Computer Science*, p. 255–274, Milovy, Czech Republic. Springer.

DIRENE, A.; BONA, L.; SILVA, F.; DOS SANTOS, G.; GUEDES, A.; CASTILHO, M.; SUNYÉ, M.; HARTMANN, C.; DE ANDRADE NETO, P.; DE MELLO, S.; SUNYÉ NETO, J., e SILVA, W. (2004). Conceitos e ferramentas de apoio ao ensino de xadrez nas escolas brasileiras. In MACÊDO, R., editor, *Anais do XXIV Congresso da Sociedade Brasileira de Computação: WIE - Workshop sobre Informática na Escola*, p. 816–825, Salvador, Brasil. SBC.

FEITOSA, A. R. M. (2005). Ensino e aprendizagem de estatégias de xadrez por meio da autoria incremental de conceitos heurísticos. Proposta de dissertação de mestrado defendida em setembro de 2005.

FERNANDES, C. T. e FERREIRA JR., D. M. (2004). Modelo de Educação a Distância com Atividades de Cooperação e Competição. In *Anais do SBC 2004: XXIV Congresso da Sociedade Brasileira de Computação/ WIE – Workshop de Informática na Escola*, p. 95, Salvador/BA.

FERREIRA, J. S. e LABIDI, S. (1998). Modelagem do aprendiz baseado no paradigma de ensino cooperativo. In *Anais do IX SBIE – Simpósio Brasileiro de Informática na Educação*, p. 494–505, Fortaleza/CE.

FIDE (2006). FIDE Laws of Chess. **http://www.fide.com/official/handbook.asp?level=EE101**. Acessado em janeiro de 2006.

GAVA, T. B. S. e DE MENEZES, C. S. (2003). Uma ontologia de domínio para a aprendizagem colaborativa. In *Anais do XIV SBIE – Simpósio Brasileiro de Informática na Educação*, p. 355–364, Rio de Janeiro/RJ.

GUIZZADRI, R. S. S.; AROYO, L., e WAGNER, G. (2003). Help&Learn: A Peer-to-Peer Architecture to Support Knowledge Management in Collaborative Learning Commu-

nities. In *Anais do XIV SBIE – Simpósio Brasileiro de Informática na Educação*, p. 285–95, Rio de Janeiro/RJ.

HARTMANN, C.; DIRENE, A. I.; BONA, L.; SILVA, F.; DOS SANTOS, G.; CASTILHO, M. A.; SUNYè, M., e GUEDES, A. L. P. (2005). Linguagens e Ferramentas de Autoria para Promover o Desenvolvimento de Perícias de Xadrez. In *Anais do XVI SBIE – Simpósio Brasileiro de Informática na Educação*, p. 656–665, UFJF.

HSU, F. (1999). Ibm's deep blue chess grandmaster chips. **IEEE Micro**, 19(2):70–81.

JOHNSON, D. W. e JOHNSON, T. J. (1989). **Cooperation and competition: Theory and research**. Edina, MA, Interaction Book Company.

JOHNSON, D. W. e JOHNSON, T. J. (1994). **Learning together and alone: Cooperative, competitive, and individualistic learning**. Needham Heights, MA, Allyn & Bacon.

JOHNSON, D. W. e JOHNSON, T. J. (1997). **Joining together: Group theory and group skill**. Needham Heights, MA, Allyn & Bacon.

JOHNSON, D. W.; JOHNSON, T. J., e SMITH, K. A. (1991). **Active Learning: Cooperation in the College Classroom**. Edina, MA, Interaction Book Company.

KAINDL, H.; SHAMS, R., e HORACECK, H. (1991). Minimax Search Algorithms with and without Aspiration Windows. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 13(12):1225–1235.

KATSENELINBOIGEN, A. (1989). **Chess as a Multi-Stage Game**. In *Selected Topics in Indeterministic Systems*, p. 100–23. Intersystems Publ., Seaside, CA.

KOENIG, S. (2001). Minimax real-time heuristic search. **Artificial Intelligence**, 129(1-2):165–197.

KORF (1999). **Artificial Intelligence Search Algorithms**. In *Algorithms and Theory of Computation Handbook*. CRC Press.

KORF, R. E. (1991). Multi-player alpha-beta pruning. **Artificial Intelligence**, 48(1):99–111.

LESGOLD, A. M.; RUBINSON, H.; GLASSER, P. F. R.; KLOPFER, D., e WANG, Y. (1989). **Expertise in a Complex Skill: Diagnosing X-Ray Pictures**. In CHI, M.; GLASSER, R., e FARR, M., editores, *The Nature of Expertise*. Lawrence Erlbaum.

MARSLAND, T. A. e REINEFELD, A. (1993). Heuristic search in one and two player games. *Relatório Técnico* TR 93-02, Edmonton, Canada.

MARSLAND, T. A.; REINEFELD, A., e SCHAEFFER, J. (1987). Low overhead alternatives to SSS. **Artificial Intelligence**, 31(2):185–99.

MURRAY, T. (1999). Authoring intelligent systems: An analysis of the state of the art. **International Journal of Artificial Intelligence in Education**, (10):98–129.

NWANA, H. S. (1990). Intelligent tutoring systems: an overview. **Artificial Intelligence Review**, (4):251–277.

OLGUíN, C. J. M.; DELGADO, A. L. N.; BOTERO, S. W., e RICARTE, I. L. M. (2000). O uso de agentes em ambientes de aprendizagem colaborativos. In *Anais do XI SBIE – Simpósio Brasileiro de Informática na Educação*, p. 236–43, Maceió/AL.

PESSOA, J. M. e DE MENEZES, C. S. (2003). Um Framework para Construção Cooperativa de Ambientes Virtuais de Aprendizagem na Web. In *Anais do XIV SBIE – Simpósio Brasileiro de Informática na Educação*, p. 275–84, Rio de Janeiro/RJ.

PLAAT, A.; SCHAEFFER, J.; PIJLS, W., e DE BRUIN, A. (1996). Best-first fixed-depth minimax algorithms. **Artificial Intelligence**, 87(1-2):255–293.

POHL, I. (1970). Heuristic search viewed as path finding in a graph. **Artificial Intelligence**, 1(3-4):193–204.

QIN, Z.; JOHNSON, D. W., e JOHNSON, T. J. (1995). Cooperative versus competitive efforts and problem solving. **Review of Educational Research**, p. 129–43.

RICH, E. (1983). **Artificial Intelligence**. McGraw-Hill.

SCHÄFER, H. e DIRENE, A. I. (2000). Conceitos e ferramentas para apoiar o ensino de xadrez atraves de computadores. In *Simpósio Brasileiro de Informática na Educação – SBIE2000*, p. 97–104, Maceió/AL.

SILVEIRA, S. R. e BARONE, D. A. C. (Nov. 2005). Formação de Grupos Colaborativos em Cursos a Distância via Web: um estudo de caso utilizando técnicas de Inteligência Artificial. **Revista Novas Tecnologias na Educação**, 3(2).

STEINBERG, I. e SOLOMON, M. (1990). Searching game trees in parallel. In *Proceedings of the International Conference on Parallel Processing*, v. 3, p. 9–17, University Park, PA.

STOCKMAN, G. (1979). A minimax algorithm better than alpha-beta? **Artificial Intelligence**, 12(2):179–96.

VORNBERGER, P. e MONIEN, B. (1987). Parallel Alpha-Beta versus Parallel SSS*. In *Proceedings IFIP Conference on Distributed Processing*, p. 613–25.

WENGER, E. (1987). **Artificial Intelligence and Tutoring Systems**. Los Altos, CA, Morgan Kaufmann.

WHITE, B. Y. e FREDERIKSEN, J. R. (1985). Quest: qualitative understanding of electrical system troubleshooting. **ACM SIGART Newsletter**, 93:34–37.