==========

Algoritmos de Busca Heurística (PARTE 3)

Busca Heurística em Grafos–E–OU

Prof. Alexandre Direne – DInf-UFPR

==========

Recomendação de Bibliografia:

- Introduction to Artificial Intelligence. Eugene Charniak e Drew McDermott. Addison-Wesley, 1989 (do qual a maior parte deste arquivo foi traduzido ou complementado ***).
- Artificial Intelligence: A Modern Approach. Stuart Russell e Peter Novig. Terceira Ed., Prentice Hall, 2010.
- Programming in Prolog. William F. Clocksin and C. S. Mellish. Springer-Verlag, 1987.
- Inteligência Artificial: Ferramentas e Teorias. Guilherme Bittencourt. Terceira Ed., Editora da UFSC, 2006.
- Artificial Intelligence. Elaine Rich e Kevin Knight. Second Edition, McGraw Hill, 1993.
- Artificial Intelligence. Patrick H. Winston. Second Edition, Addison-Wesley, 1993.

Recomendação de Páginas Web:

`http://www.cs.dartmouth.edu/~brd/Teaching/AI/Lectures/Summaries/search.html`

`http://www.decom.ufop.br/prof/guarda/CIC250/index.htm`

`http://aima.cs.berkeley.edu/`

`http://aima.cs.berkeley.edu/newchap05.pdf`

Recomendação de Software:

- Pacote de software para a Programação em Inteligência Artificial (ambiente Poplog) - obtido no seguinte endereço:

  `http://www.cs.bham.ac.uk/research/poplog/freepoplog.html`

- Pacote de software para a Programação em Prolog (SWI–Prolog) - obtido no seguinte endereço:

  `http://www.swi-prolog.org`

Busca Heurística em Grafos–E–OU

Os principais algoritmos de busca desta parte são as variações da busca MINIMAX. Tais variações se baseiam em uma série de formalismos que podem ser definidos como abordagens mais específicas de busca sobre o conceito de árvores de objetivos e Grafos–E–OU. Estes formalismos estão apresentados nas seções seguintes, logo antes das descrições das variações do algoritmo MINIMAX propriamente dito.

1 Árvore de Objetivos

Uma árvore de objetivos descreve a situação na qual um objetivo principal pode ser atingido por meio da aplicação de um método de solução de problema.

Se houver mais de um método disponível, um algoritmo de solução de problemas pode ter que tentar vários deles de forma combinada. Isto se resume a um problema de busca tradicional (e potencial explosão combinatória) - ver o Grafo–OU da Figura 1.

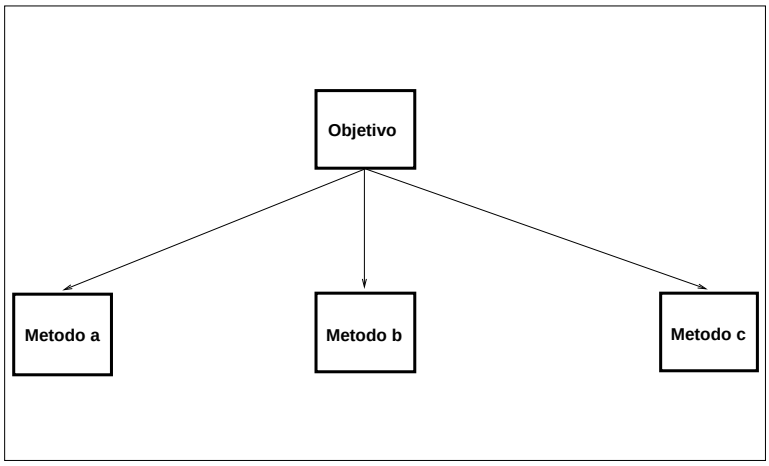

Figura 1: Sub-objetivos e métodos de alcance

A novidade introduzida aqui se explica quando cada método é composto pela agregação de vários fragmentos, todos os quais tendo suas restrições satisfeitas de forma mutuamente consistente. Neste ponto, é importante padronizar a terminologia definindo-se as seguintes correspondências: (1) método é sinônimo de opção; (2) cada fragmento de um método é sinônimo de objetivo conjugado.

Por exemplo, considere a árvore de objetivos da Figura 2 que descreve as possibilidades de como se pode agradar um(a) parceiro(a). Observe o seguinte:

- Como primeiro ponto de solução do problema, existe o objetivo de agradar um(a) parceiro(a), o qual pode ser atingido por meio de duas opções: agradar A OU agradar B;
- Como segundo ponto de solução do problema, dentro da opção adotada, há a seguinte obrigatoriedade: ir jantar E se divertir (não necessariamente nesta ordem);
- Para resolver o problema completamente, é necessário tanto agradar um(a) parceiro(a) quanto gastar menos de 100.

A árvore de objetivos é representada por meio de um grafo E–OU cujos nodos são de dois tipos:

- Tipo "OU" – representando objetivo;
- Tipo "E" – representando opção com solução compatível com as soluções de seus sub-objetivos (note aqui a recursividade mútua).

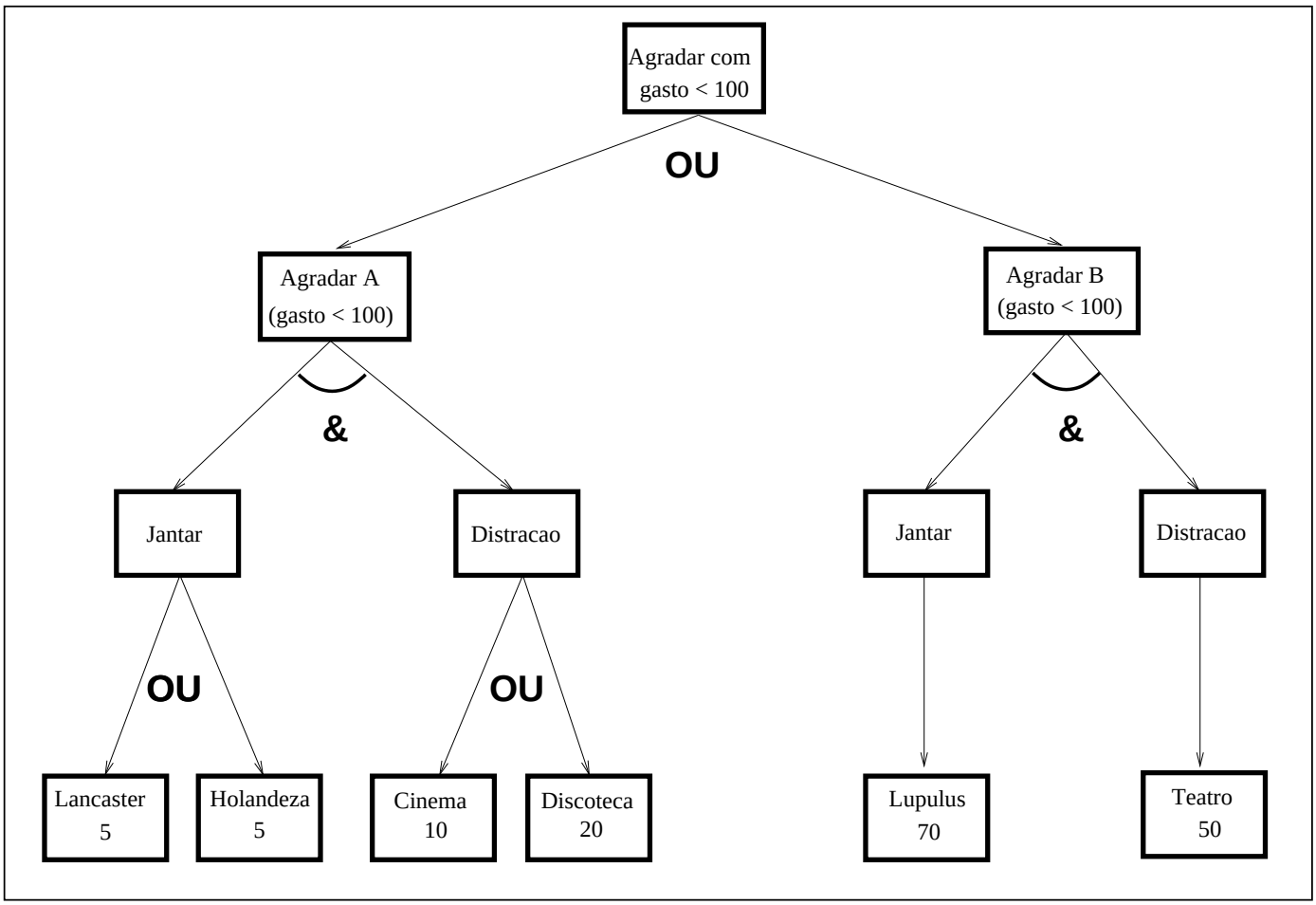

Figura 2: Exemplo do "jogo" de agradar o(a) parceiro(a)

A árvore termina nos nodos de sucesso, os quais representam a solução da menor porção do problema (sub-problema). No exemplo da Figura 2, a escolha de "Lupulus" resolve o sub-problema de jantar com o(a) parceiro(a) "B" (que agrada apenas parcialmente o(a) parceiro(a) e ainda custa menos de 100).

Note que há mais possibilidades de agradar completamente o(a) parceiro(a) "A" do que o(a) parceiro(a) "B." Na verdade, como as exigências de "B" são mais caras, não há solução para o problema de "agradar B" por menos de 100. Então, o nível mais alto da árvore de objetivos só pode ser atingido por meio de "agradar A."

Considere a ávore de objetivos genericamente apresentada na Figura 3. Observe o seguinte:

- Se associarmos os nodos do tipo "E" com letras e os do tipo "OU" com algarismos, então podemos rotular e identificar qualquer nodo da árvore por meio da concatenação alternada de letras e algarismos (sem considerar o rótulo da raiz da árvore).
- O nodo mais à esquerda na árvore recebe o nome de "$a1a1 \ldots a1a$" ou, dependendo da situação, "$a1a1 \ldots 1a1$";
- O nodo raiz recebe o nome de "0";
- Devemos assumir que filhos de um nodo do tipo "E" serão sempre do tipo "OU" (e vice-versa);
- Um descendente de um nodo pode ser definido, de forma recursiva, como sendo um dos filhos do nodo ou um dos descendentes dos filhos do nodo;
- As folhas (nodos de sucesso) da ávore são, comumente, nodos do tipo "E";

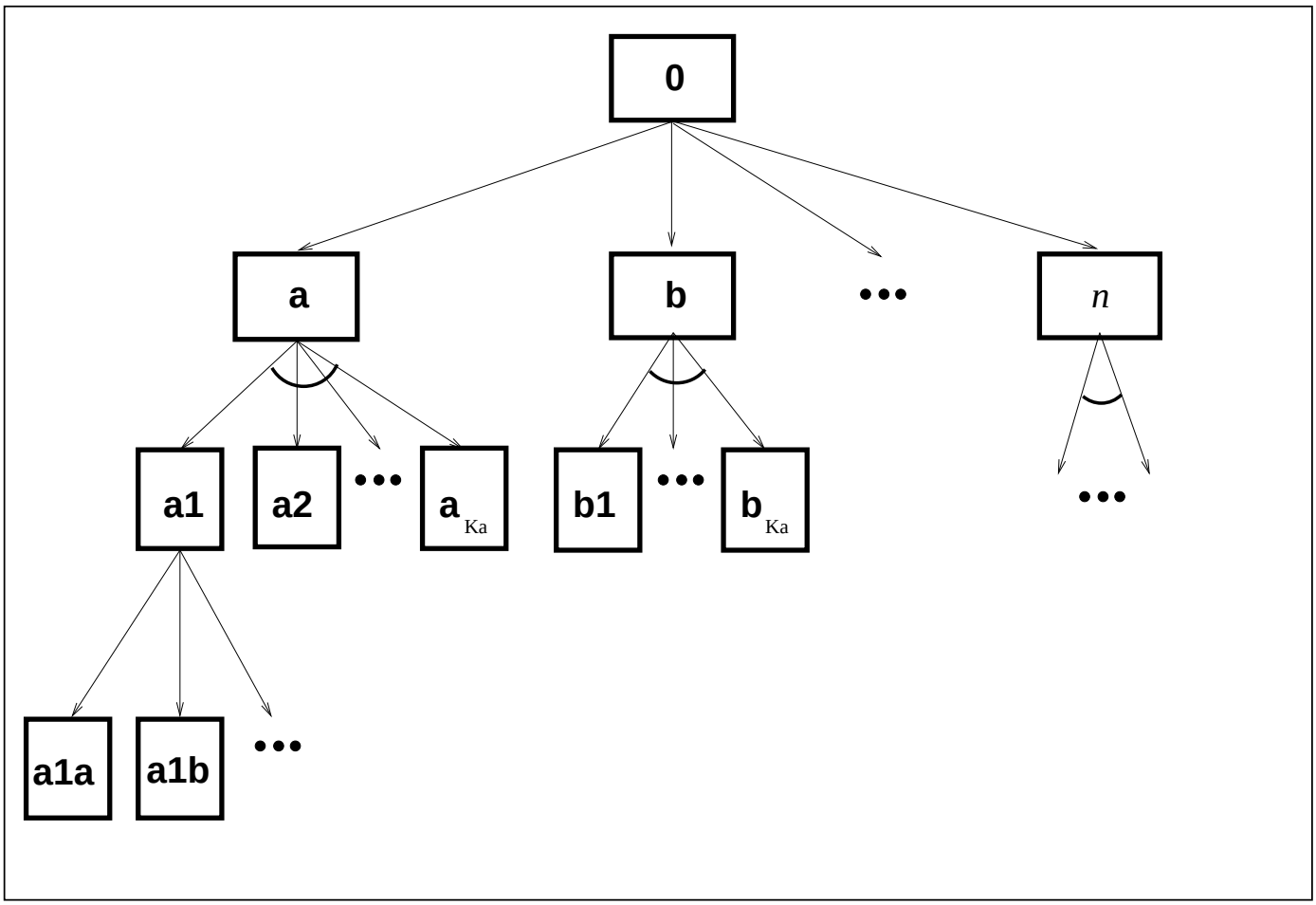

Figura 3: Visão genérica de uma árvore de objetivos

- Por razão de completude da definição, é permitida a existência de nodos do tipo "OU" como folhas da árvore, porém, estes serão considerados nodos de fracasso pois não há maneira concreta de satisfazer suas restrições;
- Cada nodo do tipo "E" que não for folha possui uma restrição a ele associada, C_{nodo}, a qual é uma função das soluções para este nodo (no nosso exemplo, o "$gasto < 100$");
- Se todas as restrições em uma ávore de objetivos forem "vazias" (satisfeitas por qualquer solução), então a árvore é simplesmente um "grafo E–OU puro";
- Um nodo do tipo "E" só é resolvido se todos os seus nodos filhos forem (recursivamente) resolvidos e, além disso, as soluções satisfazem à restrição presente no nodo;
- Um nodo do tipo "OU" só é resolvido se qualquer um de seus nodos filhos for (recursivamente) resolvido;
- A árvore de objetivos é resolvida se o seu nodo raiz for resolvido.

2 Busca tradicional na Árvore de Objetivos

Desde o início da disciplina, a forma com que a busca tradicional foi apresentada contemplou apenas a formação de árvores de busca cujos nodos são do tipo "OU". Tendo conhecido agora a representação e a solução de problemas nas árvores de objetivo, somos levados a concluir (à primeira vista) que seria impossível achar soluções em árvores do tipo tradicional. Todavia, a nossa hipótese original de busca tradicional permanece válida. Além disso, queremos remodelar a forma com que a busca na árvore de objetivos ocorre para permitir que esta seja manipulada pela forma da busca tradicional.

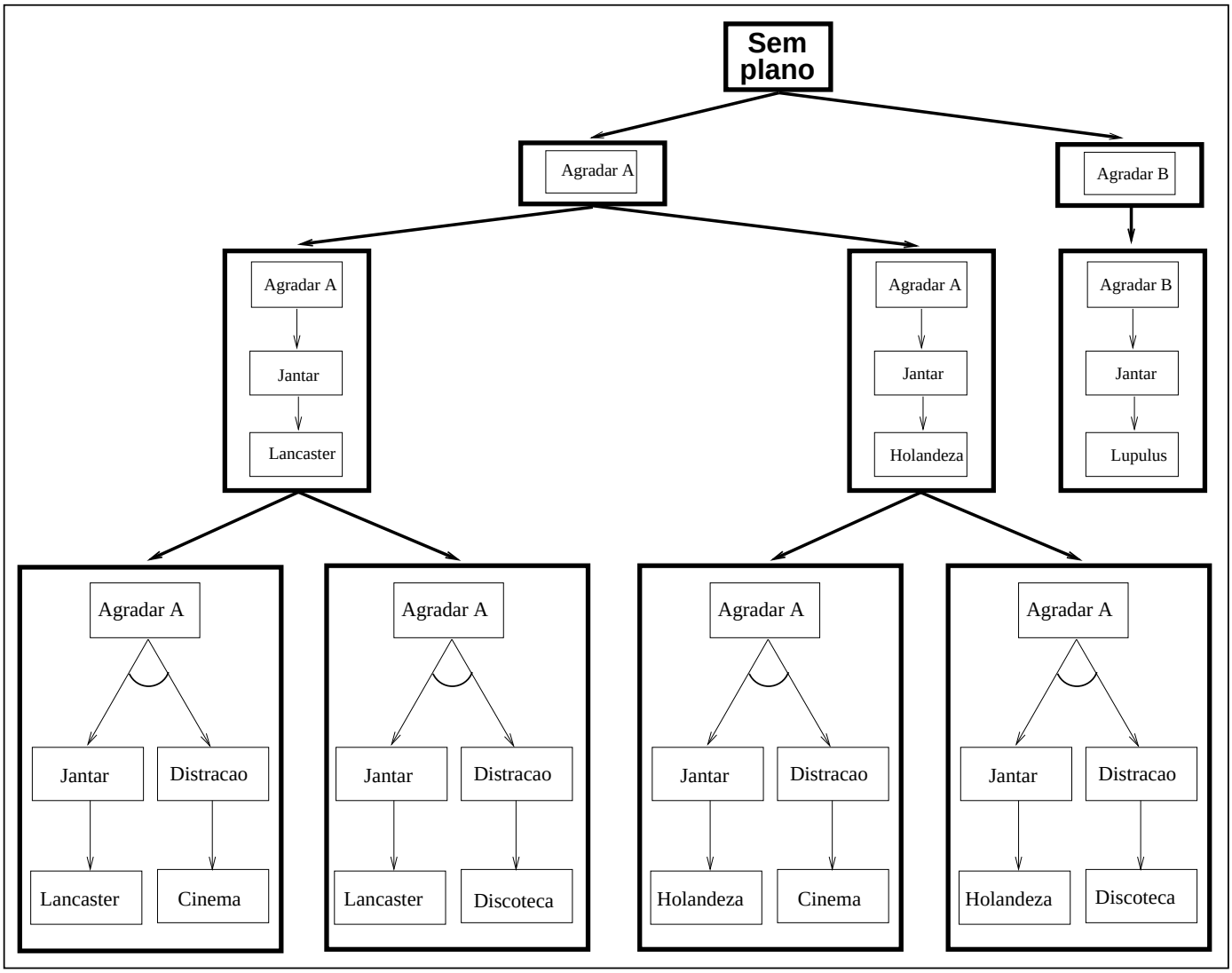

Figura 4: Árvore de busca tradicional expressando planos

Considere novamente o exemplo do "jogo de agradar o(a) parceiro(a)." Como foi mostrado na Figura 2, os nodos da árvore são ações (e não estados de memória como na árvore de busca tradicional). Tais ações são organizadas para que possamos atingir objetivos. Em alguns casos, tais ações são de alto nível, como a do nodo "agradar B," ao passo que em outros casos, a ação é de nível concreto, como a do nodo "ir ao cinema."

Se trocarmos agora o conteúdo dos nodos para que estes não sejam mais ações e sim planos, algumas alterações relevantes resultam disso. Um plano, em si, terá uma estrutura de árvore e conterá os elementos que o planejador pretende montar para atingir a meta de mais alto nível.

A versão do espaço de busca (árvore de busca tradicional), para esta nova representação de planos, aplicada ao "jogo de agradar o(a) parceiro(a)" é mostrada na Figura 4. Observe o seguinte:

- O nodo mais abaixo, à esquerda, mostra o plano de "agradar o(a) parceiro(a) A" por meio de "ir à Lancaster" E "ir ao cinema";
- O nodo raiz da árvore não possui nenhum plano, ao passo que os nodos do tipo folha apresentam planos completos (com exceção de um);
- Os nodos de nível intermediário apresentam planos em estágios de maior ou menor completude;
- No caso do(a) "parceiro(a) B", a árvore é podada em um plano parcial, pois não há um plano completo que satisfaça às restrições.
- Os nodos da árvore são todos do tipo "OU";

- Os nodos do tipo "E" foram todos embutidos nos planos;
- Sendo assim, efetuando a busca em um espaço de planos (ao invés de ações), podemos reduzir a árvore de objetivos a um espaço de busca tradicional e, com isso, aplicar os algoritmos apresentados anteriormente.

No sentido de formalizar mais ainda o lado genérico da abordagem de embutimento de qualquer árvore de objetivos em uma árvore de planos, vamos considerar 3 (três) conceitos adicionais (na árvore de objetivos):

- solução parcial: a solução parcial para um nodo n é a sub-árvore cuja raiz é n, com a propriedade de que, se tal solução tiver um nodo m do tipo "OU", então ela contém exatamente um filho de m.
- solução completa: a solução completa para um nodo n é uma solução parcial para n com a propriedade de, se tal solução contiver um nodo m do tipo "E", então ela contém todos os filhos de m, e as soluções para tais filhos satisfazem $C(m)$ (a restrição em m). Uma solução para a árvore de objetivos inteira é uma solução completa para a sua raiz (o nodo 0).
- extensão: uma solução parcial $s2$ é considerada uma extensão para a solução parcial $s1$ se $s1$ está contida em $s2$; isto é, $s2$ é uma árvore com todos os nodos de $s1$, com zero ou mais opções adicionadas.

Com isso, podemos agora definir o novo espaço de busca como um espaço de soluções parciais para a árvore de objetivos. Os filhos de um nodo são as extensões do nodo nas quais (exatamente) uma opção adicional é adotada.

Se considerarmos a situação da Figura 4 novamente sob este novo ângulo, observamos o seguinte:

- O nodo que possui o plano composto apenas de "agradar A" é uma extensão do nodo "sem plano" e ele representa exatamente um nodo a mais de opção (agradar A ao invés de B).
- De maneira semelhante, o nodo logo abaixo de "agradar A" que contém "ir à Holandeza" representa esta como tendo sido a opção ao invés de "ir à Lanccaster", e um dos nodos folha mais abaixo, contendo "ir à Holandeza" e "ir ao cinema", inclui mais uma opção ainda.
- Mas os nodos da Figura 4 contém informação redundante (as estruturas detalhadas de árvore da figura ajudam a esclarecer o que o plano é mas elas, na verdade, não são necessárias).
- Tudo que o conteúdo de cada nodo da Figura 4 precisa conter é a sequência de opções adotadas (a estrutura arbórea do plano dentro de um nodo não é realmente necessária).

Reestruturando a referida árvore para eliminar as redundâncias, obtemos a nova árvore da Figura 5. Note o seguinte:

- Na verdade, há mais extensões possíveis para os nodos do tipo "E" (exemplo: "agradar A") do que as apresentadas (*e.g.,* Lancaster-Cinema e Cinema-Lancaster).
- Só foram indicadas as extenções que representam opções da parte mais à esquerda da árvore original (as que abordam "ir jantar" primeiro).
- Porém, há outras extensões que representariam opções por outra parte da árvore original (as que abordam "se distrair" primeiro).
- No caso das Figuras 4 e 5, apenas as opções de "se distrair" feitas depois das opções de "ir jantar" foram mostradas.
- Com isso, todos os planos possíveis estão contemplados.

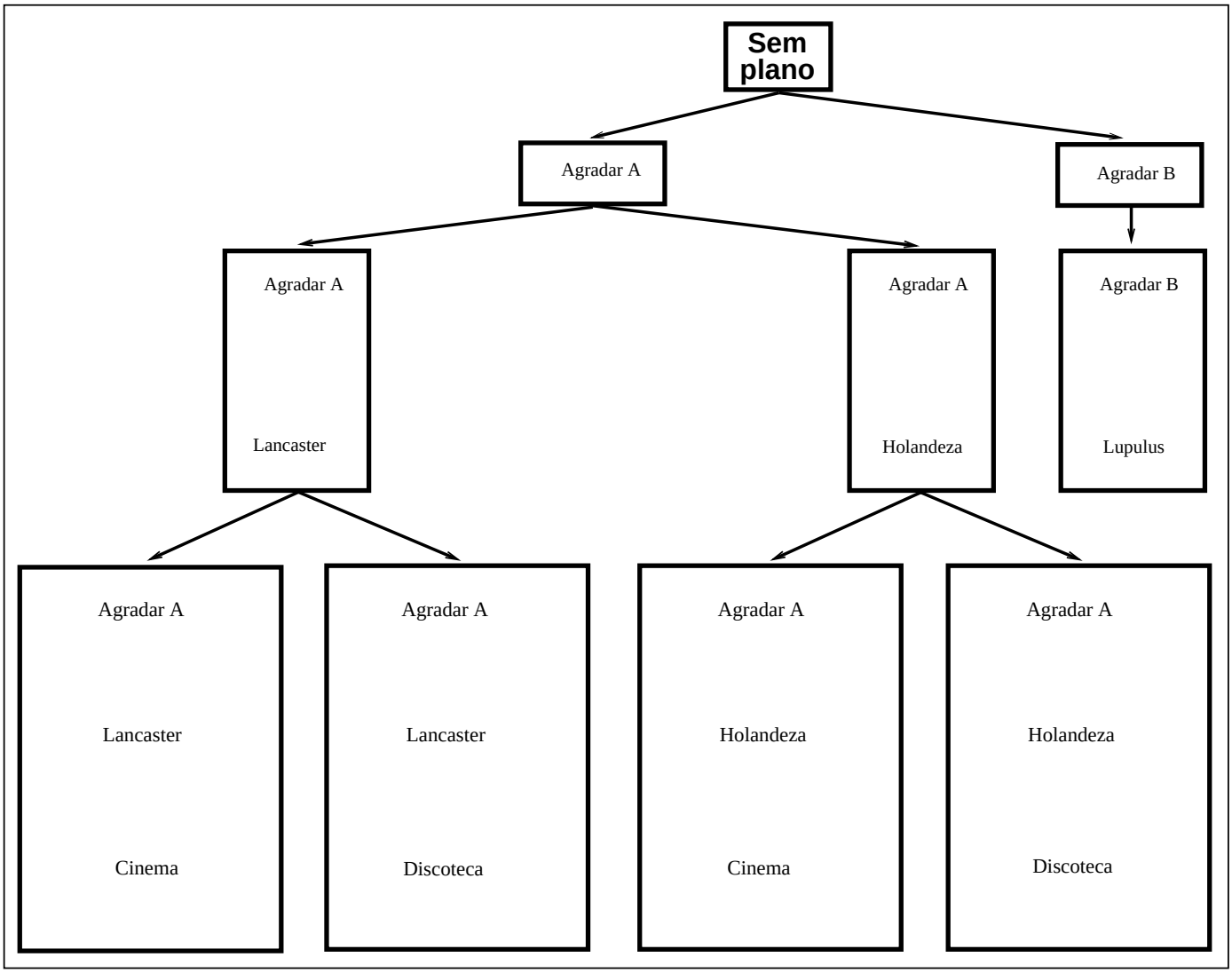

Figura 5: Árvore de busca tradicional sem redundância

- Todavia, a ordem de geração dos planos pode afetar drasticamente o tamanho da busca, ou mesmo a solução encontrada para um sub-problema.

Retomando a abordagem sobre casos genéricos, considere a Figura 6. Será usada a notação $n : \{l_1, l_2, \ldots, l_k\}$ como forma de referência à árvore cuja raiz é o nodo n, terminando nos nodos $\{l_1, l_2, \ldots, l_k\}$ (todos do tipo "E"). Esta notação é derivada daquela usada na Figura 5. A derivação pode ser notada de diversas maneiras:

- a notação omite a estrutura de árvore em cada nodo;
- existe aumento da quantidade de elementos no nome dos nodos mais profundos da árvore;
- existe alternância de letras e algarismos na composição do referido nome;
- a notação também informa, em cada nome de nodo, as opções (nodos "E") adotadas até o nível da árvore onde o nodo se encontra.

Nesta notação, de acordo com a Figura 6, uma solução parcial para 0 é $0 : \{a\}$. Uma solução parcial para o nodo a é $a : \{a\}$. Uma outra é $a : \{a1a, a2b, a3a\}$. Uma solução completa poderia ser $a : \{a1a1a, a1a2b, a2b, a3a\}$, desde que a solução $a1a : \{a1a1a, a1a2b\}$ satisfaça à restrição $C(a1a)$, e a solução completa satisfaça $C(a)$. Nesta sequência de três soluções parciais para a, cada uma é a extensão

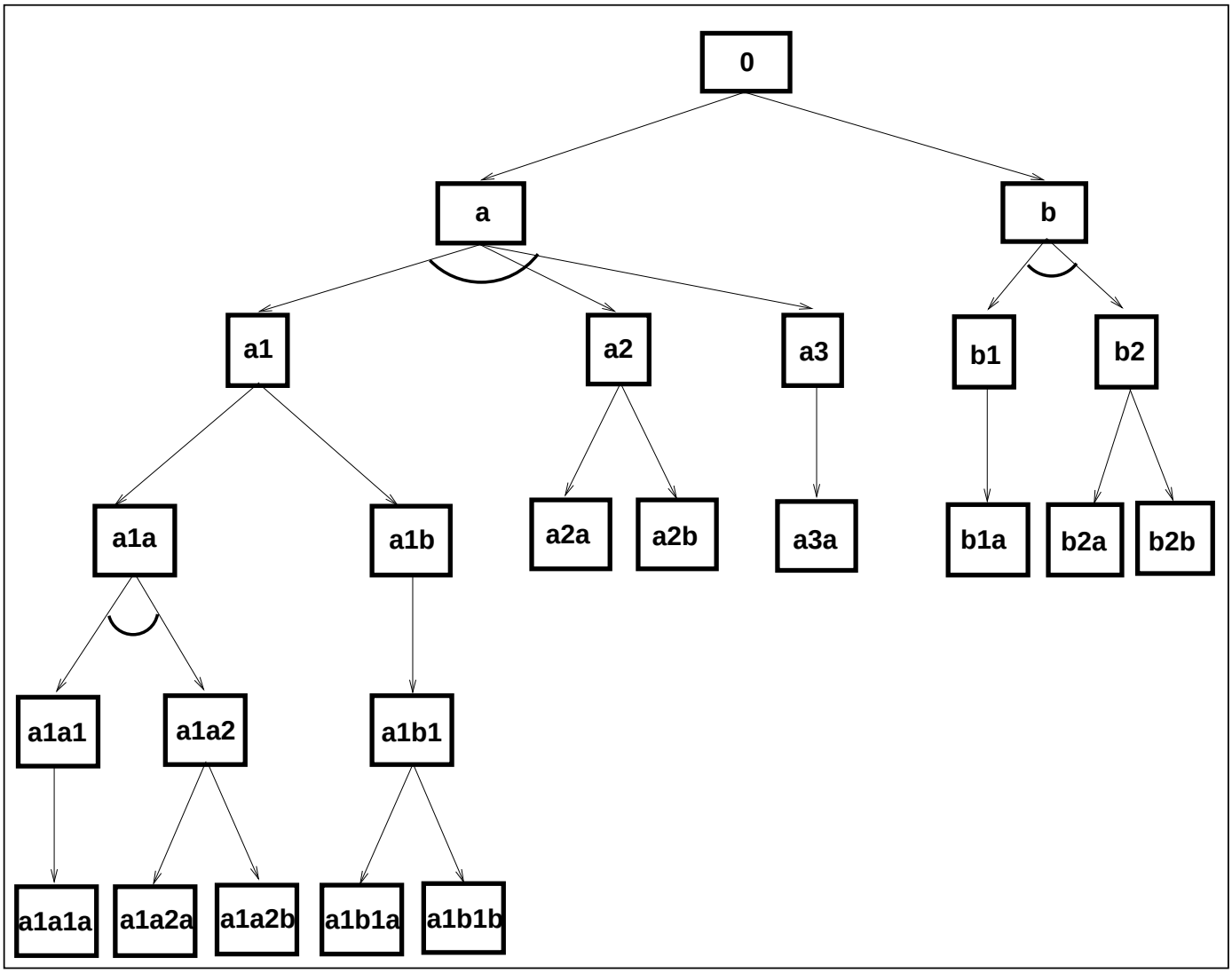

Figura 6: Examplo tipificado de árvore de objetivos

de sua anterior (a transição de $a : \{a\}$ para $a : \{a1a, a2b, a3a\}$ implica em adotar obrigatoriamente os sub-objetivos $a1$, $a2$ e $a3$; a próxima transição, na ordem, implica em adotar ambos os sub-objetivos $a1a1$ e $a1a2$).

É importante ainda observar o seguinte:

- Uma árvore de objetivos é um objeto abstrato (*i.e.*, pode não estar presente na memória real; pode ser infinita; a busca por conjuntos de soluções pode ser bem difícil).

- Uma maneira de pensar em solução parcial se baseia em dois elementos: opções já adotadas e opções a serem adotadas. Quando este segundo conjunto está vazio, a solução está completa. O segundo conjunto deve ser manipulado com base no que já foi construído no primeiro conjunto. Por exemplo, na Figura 6, a solução parcial $0 : \{a, a1a, a3a\}$ requer a adoção de opções para os sub-objetivos $a1a1$, $a1a2$ e $a2$ (nenhuma outra é obrigatória).

A solução do problema de colorir países em um mapa usando apenas 4 (quatro) cores, sem a repetição de cores nas fronteiras contínuas pode ser representado através da busca por soluções parciais de árvores de objetivos. Em qualquer estado da busca, há o conjunto de "países já coloridos" e o conjunto de "países a serem coloridos" (ver Figura 7).

O registro explícito em memória das opções pendentes introduz novas possibilidades na busca em uma árvore de objetivos. Com isso, passa a ser necessária a estimativa da distância do estado–atual até o estado–

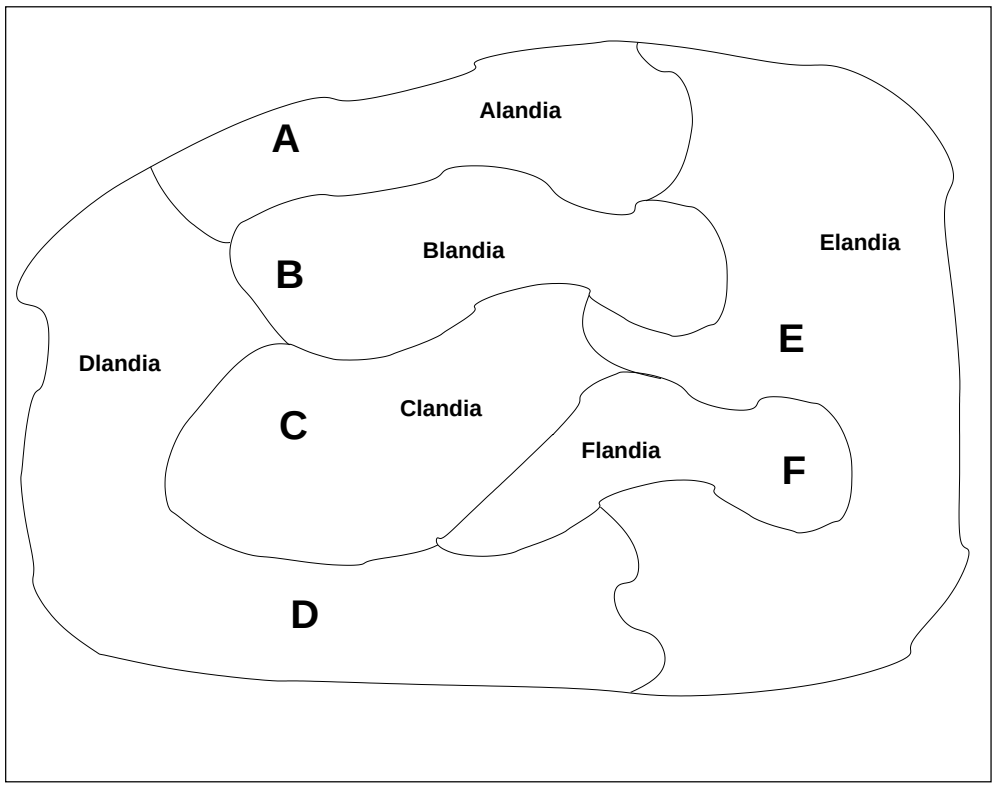

Figura 7: Exemplo de um mapa com países a serem coloridos com 4 cores

solução (avaliação heurística), sendo que o registro destas opções pendentes pode ser uma fonte de dados para compor a estimativa.

Uma forma simples de se adotar esta abordagem heurística seria a de estimar a distância para o estado–solução considerando-a tanto maior quanto mais opções pendentes houver. No exemplo de colorir mapas, esta estimativa sugere que um processo de busca deveria investir primeiro nas soluções parciais que apresentam maior cardinalidade do conjunto de países já coloridos.

Uma idéia melhor seria a de reduzir, com garantia, a complexidade de manipular o conjunto de soluções parciais devido ao seu tamanho. Em alguns domínios, é possível detectar a violação de restrições antecipadamente por avaliação puramente epistemológica. No domínio de "colorir mapas" (ou de "jogar Xadrez"), assim que dois países adjacentes ganharem a mesma cor, a restrição de cor de fronteira é violada (os países do conjunto de "países a serem coloridos" não precisam ter suas cores efetivamente decididas para que a restrição seja aplicada).

Uma forma de elaborar melhor esse conceito de busca é por meio da idéia do maior estreitamento (eliminação ou poda) das combinações de opções pendentes com base nas opções já adotadas. Na Figura 6, por exemplo, a opção $a1a$ como meio de atingir o sub-objetivo $a1$ pode permitir a eliminação de algumas soluções que incluem opções adotadas para objetivos irmãos de $a1$. Em outras palavras, na adoção de uma opção para $a2$, por exemplo $a2a$, pode não existir uma solução completa para a árvore de objetivos que inclui $a1a$. Na presença deste fato, dizemos que $a1a$ eliminou $a2a$ como uma opção possível.

Como exemplo concreto deste maior estreitamento (poda), no mapa da Figura 7, uma vez que Alândia, Blândia e Clândia tenham sido coloridas de vermelho, azul e laranja, respectivamente, todas estas opções podem ser eliminadas como opção de cor para Dlândia.

Se conseguirmos boas estimativas de dificuldade (função heurística) para avaliar as opções pendentes em cada sub-objetivo, então, pode fazer sentido estender primeiro o sub-objetivo com o menor número de opções pendentes (menor dificuldade). Nos casos onde houver 0 (zero) ou 1 (uma) opção pendente, a decisão é bem definida. No caso de 0 (nodo "OU" sem filhos), a solução parcial não pode ser estendida para uma solução completa (e deve ser abandonada). No caso de exatamente 1 (nodo "OU" com 1 filho), é possível adotar a opção agora (tão logo a estimativa se torne disponível), pois isso pode ajudar a restringir (estreitar) outras extensões mais rapidamente ainda.

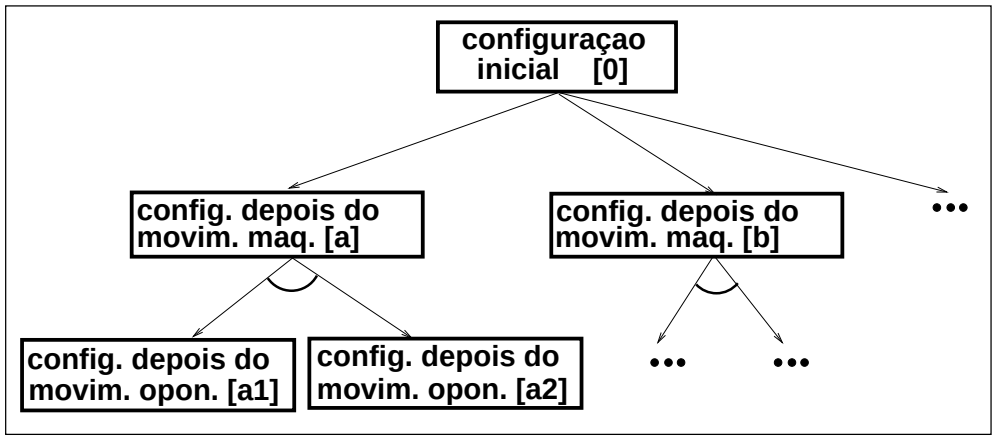

Figura 8: Uma árvore de jogo genérica

3 Árvore de Jogos como Árvore de Objetivos

Uma árvore de jogos é uma árvore de objetivos na forma de um Grafo "E–OU" puro. Ela inclui opções de caminhos de movimentos (de todas as partidas possíveis) do jogo onde, alternadamente, cada jogador faz movimentos discretos. Cada movimento leva sempre a um conjunto finito de estados totalmente previsíveis. Note pela árvore da Figura 8 vários pontos.

- Os nodos "OU" representam a visão do personagem "máquina" e os nodos "E" do seu "oponente" (isomorfismo sintático entre os grafos).
- O sub-problema mais imediato é o de decidir o próximo movimento plausível da máquina.
- Todavia, devido ao fato do oponente atacar a máquina, ela terá que considerar todas as possibilidades de resposta do oponente para o seu próximo movimento plausível (referido acima), incluindo como será o contra-ataque a cada uma dessas possibilidades.
- Isso nos leva a pensar no problema não apenas como o de decidir puramente qual é o próximo movimento plausível mas sim o de achar táticas completas de como ganhar a partida.
- Sendo assim, da mesma maneira que o propósito de uma árvore de objetivos era o de encontrar um plano, o propósito de uma árvore de jogos é o de encontrar uma "tática" para ganhar o jogo.
- Nas folhas da árvore (assumindo que ela é finita) estão as posições de final de jogo, onde a máquina "ganhou" ou "não ganhou".
- Uma tática completa é uma "receita de bolo" que diz qual movimento a máquina deve fazer em cada posição (sub-objetivo), ou melhor, em cada posição que a tática não descartar.
- Com isso, se a tática orientar a máquina a realizar o movimento "a" primeiro, então não é necessário levar em consideração "o que aconteceria se" ela tivesse realizado o movimento "b" no lugar.
- De maneira formal, uma tática completa é uma sub-árvore de jogo com a mesma raiz da árvore de jogo, de tal forma que exatamente um ramo é incluído nos nodos da máquina e todos os ramos são incluídos nos nodos do oponente.
- Em outras palavras, a tática deve antecipar qualquer coisa que o oponente faria.

- Além do isomorfismo sintático, também há equivalência semântica entre a árvore de objetivos e a de jogos, na medida em que podemos usar a árvore de jogos para provar que a máquina pode vencer.
- Para isso, nos nodos da máquina, seria suficiente achar apenas um movimento que comprovadamente funcione, ao passo que, nos nodos do oponente, é preciso provar que todos os movimentos fracassam na tentativa de impedir a máquina.
- Com isso, a tática completa pode ser considerada como uma prova da vitória.
- Se não houver tática que garanta a vitória, é fácil buscar o empate.
- Programas de computador adotam tais táticas e funcionam razovelmente bem.
- Todavia, para jogos como o Xadrez, a árvore de jogo é muito grande para permitir a montagem de táticas completas.
- Ao invés disso, os movimentos são explorados apenas com alguns níveis de profundidade;
- Em seguida, uma função heurística é aplicada para estimar o quão boa a configuração é.
- Nessa condição, a árvore será chamada de árvore incompleta de jogo.
- Em um jogo típico de tabuleiro (ex. Xadrez, Damas, etc), a função heurística é chamada de "Função de Avaliação Estática" (FAE).
- Rotulando-se a máquina de "MAX" e o oponente de "MIN", pode-se dizer que MAX deverá buscar configurações de valores altos da FAE ao passo que MIN os de baixo valor.
- Neste caso, a FAE deveria ser construída de tal maneira que, quando aplicada exclusivamente ao conteúdo da configuração, sob o ponto de vista da máquina, fornecesse uma estimativa da vantagem ($FAE > 0$) ou desvantagem ($FAE < 0$) material e/ou posicional.
- Tais valores pertencem a uma escala inversa àquela que vínhamos considerando até o momento como sendo a dos resultados de uma função heurística (o menor da escala era o melhor).
- Cada movimento de MAX alternado com um movimento de MIN contempla 1 (um) lance ou, por definição, 2 (dois) níveis de alternância.
- Se tivéssemos FAEs "perfeitas", apenas um nível de alternância seria necessário.
- Como FAEs perfeitas são raríssimas (cuidado!), a expectativa de melhoria da estimativa reside na tentativa de trabalhar com mais de um nível de alternância.
- Tal idéia não é nada simples de ser explicada, já que explorar mais de um nível da árvore pode também resultar em um pior próximo movimento.
- Mas, intuitivamente, não parece haver grande mal em seguir o que a tradição tem apresentado como conduta bem sucedida.
- Usa-se o termo "profundidade de visada" para representar o máximo número de níveis de alternância a serem gerados antes de se aplicar a FAE.
- No caso de haver um estado terminal acima da "profundidade de visada", a expansão daquele ramo da árvore deveria ser suspensa e a FAE deveria ser "aplicada" ao estado, atribuindo a ele: (1) um valor muito alto se o nível for de MAX; (2) um valor muito baixo se o nível for de MIN; (3) o valor 0 (zero) se for empate.

- No limite das idéias apresentadas, se MIN cooperasse com MAX (ao invés de se opor a ele) o melhor próximo movimento seria também o que levaria à configuração de maior valor da FAE no nível da "profundidade de visada" (ou mesmo acima dele).

- Todavia, o "próximo movimento" que leva à configuração de maior valor da FAE na profundidade de visada pode levar também a uma configuração péssima para MAX (já que os movimentos sub-sequentes ao "próximo movimento" dependem tanto de MIN quanto de MAX).

- Então, de maneira realista, o que se quer como solução do sub-problema é o "próximo movimento" de MAX que leva à "melhor" configuração no nível da profundidade de visada, assumindo que MIN cause os menores danos possíveis para MAX.

- Adicionalmente, o "valor representativo" de uma tática completa pode ser definido como o "menor" valor de qualquer folha da sub-árvore que representa aquela tática (já que ela deixa abertas apenas as decisões de movimentos de MIN, o qual pode optar pelos piores valores para MAX).

- Com isso, a melhor tática da árvore de jogo é a tática completa que tiver o maior "valor representativo" a ela associado.

- Tradicionalmente, esse maior "valor representativo" é referido como o "valor minimax" da árvore de jogo completa.

- Note que árvores de jogos completas são apenas casos especiais das incompletas, onde cada estado terminal tem, por exemplo, $FAE = 100$ (vitória de MAX), $FAE = -100$ (vitória de MIN), $FAE = 0$ (empate).

- Com isso, a melhor tática seria aquela com $FAE = 100$, se possível, caso contrário, $FAE = 0$.

- Por exemplo, a Figura 9 apresenta um fragmento da árvore do Jogo da Velha.

- Na vez de MAX, se ele marcasse um "X" no canto inferior esquerdo como resposta à configuração do nodo 0, isso o levaria à configuração do nodo a.

- Se MAX efetuasse o referido movimento, MIN teria várias possibilidade de resposta, sendo necessário pesquisar em todas elas.

- Uma delas seria marcar um "O" na casa inferior direita, o que levaria à configuração do nodo $a1$.

- Para isso, MAX tem uma excelente resposta: a configuração do nodo $a1a$ (vitória de MAX, logo a FAE deve ser aplicada e retornar um valor alto, mas ainda abaixo de 100, por exemplo – a explicação de $FAE \leq 100$ continua no texto abaixo).

- Porém, MIN tem outras possibilidades de resposta para a configuração do nodo "a" e antes de MAX se decidir totalmente por seu "próximo movimento", tais possibilidades devem ser pesquisadas.

- Inclusive, antes de MAX poder atingir a configuração do nodo $a1a$, uma configuração muito boa para MIN é a do nodo $a2$, a qual leva à sua vitória na partida, e por isso deve receber um valor heurístico negatigo muito baixo ($FAE \leq -100$, sempre computado sob o ângulo de MAX, a máquina).

- Neste momento, o "próximo movimento" de MAX para a configuração do nodo "a" já não parece nada bom.

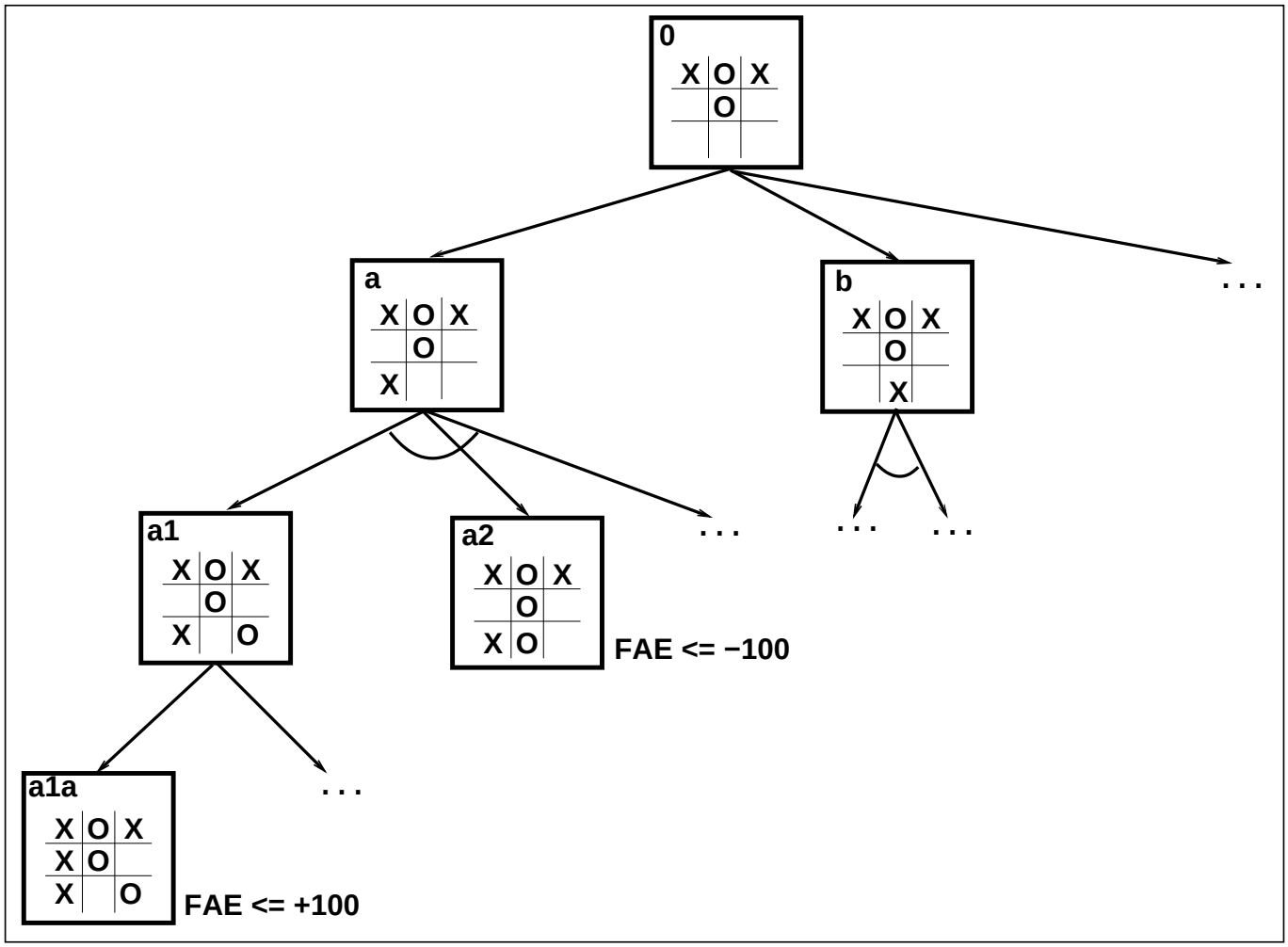

Figura 9: Fragmento da árvore do Jogo da Velha

- A Figura 10 apresenta a Árvore de Táticas (Grafo OU de busca no qual o conteúdo de cada nodo é uma tática) correspondente ao fragmento da árvore do Jogo da Velha da Figura 9.

- A opção de MAX que leva à configuração do nodo "a" corresponde à solução parcial $0 : \{a\}$ na Figura 9.

- Tal solução parcial não contém nenhum nodo folha, logo devemos atribuir a ela um valor $FAE \leq +\infty$ e $FAE > +100$ (o altíssimo valor positivo $+\infty$ garante que a máquina continuará expandindo a tática até encontrar um nodo folha ao passo que o operador relacional $\leq$ nos deixa abertura para uma eventual mudança de opinião – a condição $FAE > +100$ serve para isolar esta faixa das demais faixas na escala completa de valores heurísticos).

- O próximo nível é o de um "nodo–E", logo a tática deve incluir cada resposta de MIN à configuração do nodo "a".

- Como foi visto, a configuração do nodo $a1$ gerada por MIN permite a vitória de MAX um nível abaixo, no nodo $a1a$ (gerando um novo valor da $FAE \leq +100$ e $FAE > 0$ para a tática, já que este será menor que o último, $FAE \leq +\infty$ e $FAE > +100$).

- O valor é $FAE \leq +100$ e $FAE > 0$ pois MIN ainda pode ter configurações melhores para ele que a do nodo $a1$ (gerando nova redução do valor da FAE).

- Reafirmando o que foi dito, a solução parcial $0 : \{a1a\}$ fica com $FAE \leq +100$ e $FAE > 0$ e ela se torna a solução parcial "dominante" em relação e todas as demais alcançadas a partir do nodo $a1$.

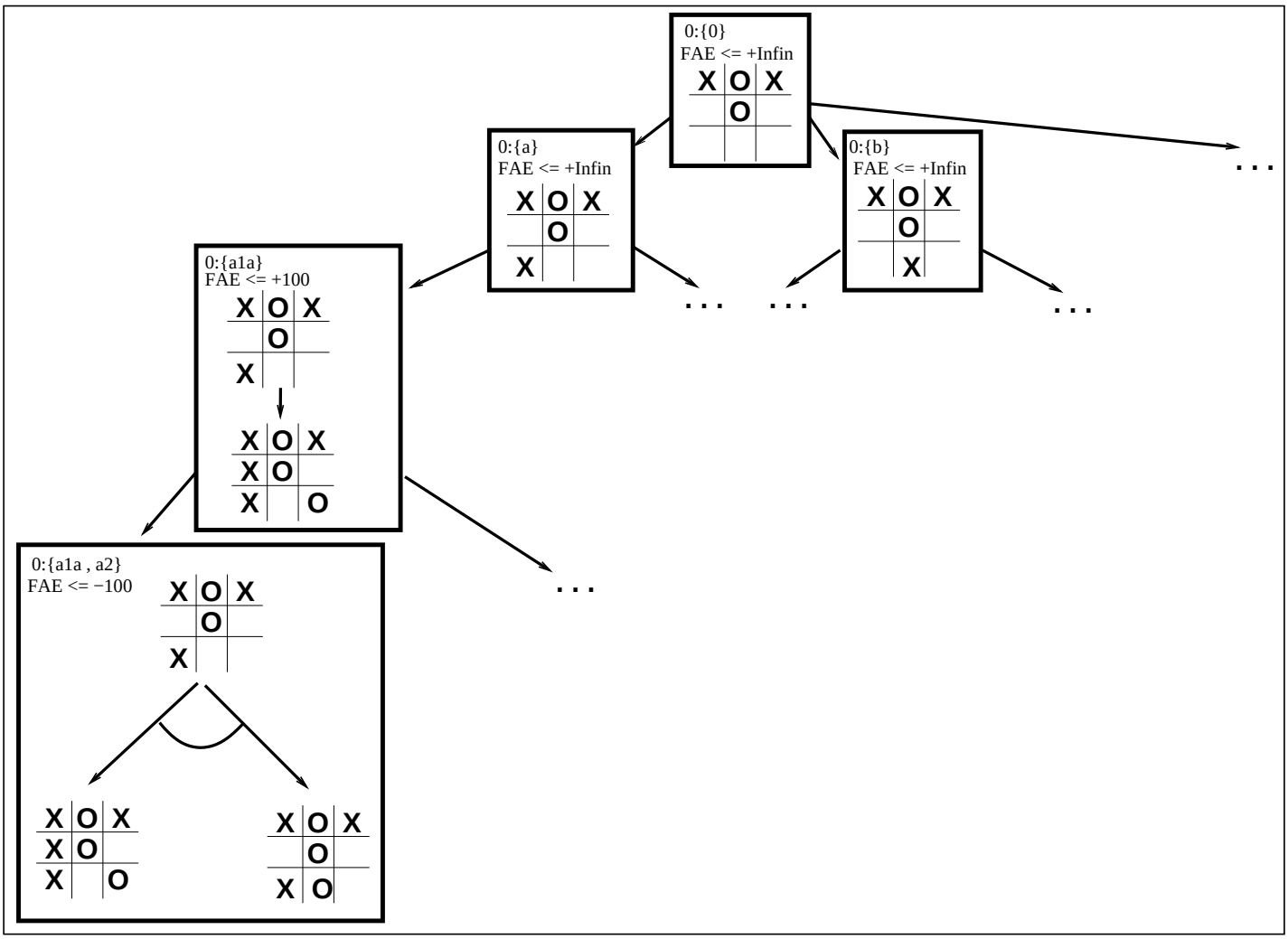

Figura 10: Fragmento da Árvore de Táticas do Jogo da Velha

- Seguindo à frente um pouco mais, é hora de incluir na solução parcial o que MAX deve fazer se MIN gerar a configuração $a2$.
- Como $a2$ é um estado terminal no nível de um "nodo–OU" (vitória de MIN), a avaliação heurística deve sinalizar $FAE \leq -100$.
- Este valor de $FAE \leq -100$ é ainda menor que os do intervalo $FAE \leq +100$ e $FAE > 0$, gerando um novo valor da $FAE \leq -100$ para a solução parcial $0 : \{a1a, a2\}$.
- Tal valor faz com que a máquina perca o interesse na referida solução parcial.
- Infelizmente, o algoritmo irá continuar a expansão desnecessária do nodo $0 : \{a\}$, mesmo que nenhuma delas seja melhor do que a já atingida.
- Serão vistas as formas de reduzir esse desperdício de tempo e memória utilizando mais "conhecimento" genérico para se realizar podas em táticas mas isso só é possível se mais conhecimento heurístico sobre o Jogo da Velha (ou qualquer outro jogo) for adicionado também à escala da FAE.
- Neste ponto, pode-se acusar uma mudança no conceito de "solução", o qual passou de "Árvores Completas de Jogos" para ser agora "Árvores Inompletas com Estimativas".
- No caso atual, procura-se por uma "solução ótima" ao invés de "qualquer solução" (o que pode soar como uma melhoria dos métodos em estudo aqui, mas na verdade não é!!!).

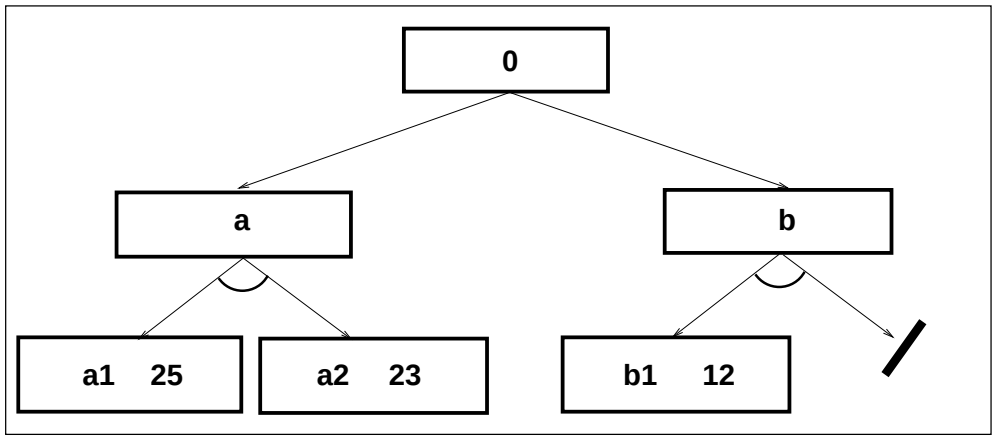

Figura 11: Exemplo da poda de um ramo com o movimento pior para MAX

- Na verdade, a "solução ótima" é apenas a solução para um problema de menor interesse (o de "achar a solução aparentemente melhor para MAX") do que a de "achar a solução que garanta a vitória de MAX".

- Embora em ambos os conceitos (anterior e atual) a busca seja por uma tática "completa" (uma receita exaustiva de como proceder diante de cada configuração), a árvore não mais representa um jogo completo.

- No conceito anterior, uma vez detrminada a tática completa, nenhum processamento posterior era necessário.

- Com o novo conceito, a tática pode até ser "completa", mas apenas em relação a uma árvore incompleta.

- Ou seja, depois do primeiro movimento, e obtido o contra-ataque do oponente, a árvore de estimativas precisa ser re-construída a partir da nova raiz.

- Isso ocorre pois serão gerados agora mais dois níveis em relação à árvore do último cálculo (embora os dois níveis superiores da árvore do último cálculo sejam automaticamente suprimidos).

4 Variações da Busca MINIMAX

Há várias maneiras de "melhorar o desempenho" (tempo de processamento, espaço de memória e plausibilidade de estados) dos algoritmos de busca sobre árvores de jogos e, na sua maioria, isso se dá por meio da adição de conhecimento específico sobre o domínio do jogo em questão. Todavia, há uma maneira genérica de se atingir este fim, fazendo o que poderia ser chamado de "uso epistemológico" dos valores heurísticos já que cada um deles é associado com uma tática parcial. Neste caso, "melhoria de desempenho" significa apenas uma diminuição no tempo de processamanto (mas não necessariamente do espaço instantâneo de memória ocupada) ao considerarmos um número menor de táticas logicamente possíveis, porém mais plausíveis.

4.1 Algoritmo SSS^*

O algoritmo apresentado nesta Sub-seção é chamado de SSS^*. Em poucas palavras, ele permite a realização de podas em táticas da árvore de jogos de acordo com duas grandes categorias (com efeitos correspondentes sobre a árvore de táticas). A primeira delas está ilustrada na Figura 11. Note os pontos abaixo.

- Na referida árvore, não há necessidade de expandir o ramo mais à direita.

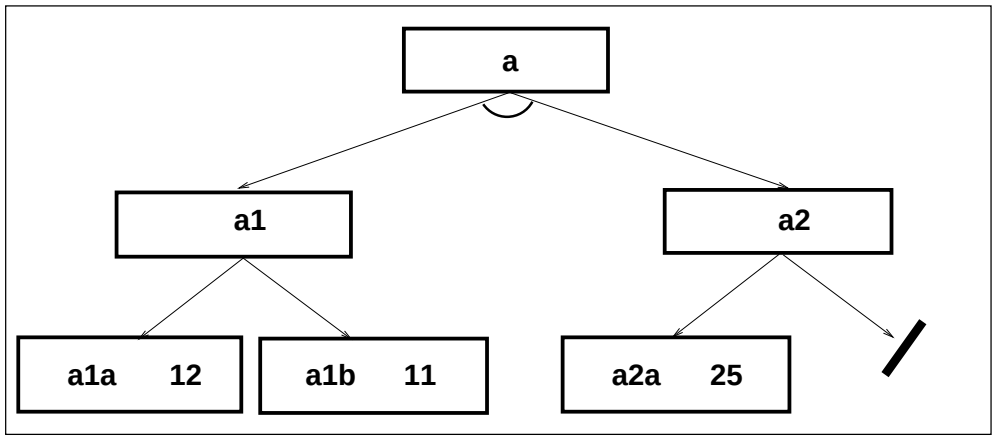

Figura 12: Exemplo da poda de um ramo com o movimento pior para MIN

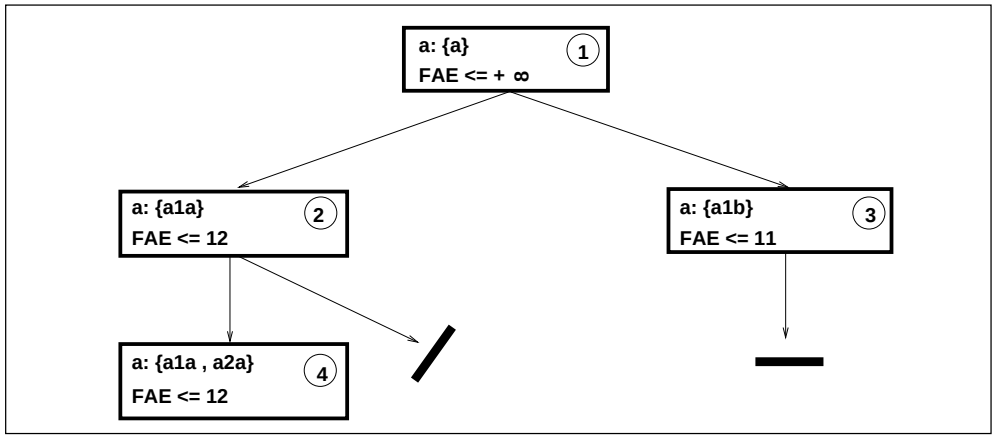

Figura 13: Árvore de táticas correspondente ao movimento pior para MIN

- Já que MIN tem opções de movimentos a partir do nodo b, ele não pode causar a MAX nenhum estrago maior do que o já atingido com a configuração $b1$ (de valor heurístico 12), assumindo que MAX jogue em b.

- Como MAX já tem garantida a tática de valor 23 se ele jogar em a, não seria razoável ele optar por b.

- Consequentemente, assim que $b1$ é conhecido e tiver seu valor heurístico computado, fica determinado que a tática final de MAX não incluirá b, tornando inócua a ivestigação de seus demais efeitos.

O segundo caso onde a avaliação heurística de um nodo indica a possibilidade de poda está ilustrado na Figura 12. Note os pontos abaixo.

- Nesta situação, MIN não escolheria o movimento do nodo $a2$ já que, em optando por $a1$, a pior coisa para ele seria a tática de valor 12.

- É interessante notar que esta segunda categoria de poda já está embutida no método de busca em árvores de táticas visto até o momento.

- Para entender como isso se dá, veja a Figura 13, a qual apresenta a árvore de táticas correspondente à árvore de jogo da Figura 12.

- Na Figura 13 estão registrados os seguintes dados em cada nodo: (1) as opções adotadas para a tática; (2) o valor heurístico da tática; (3) a ordem em que o estado foi gerado na árvore de táticas.

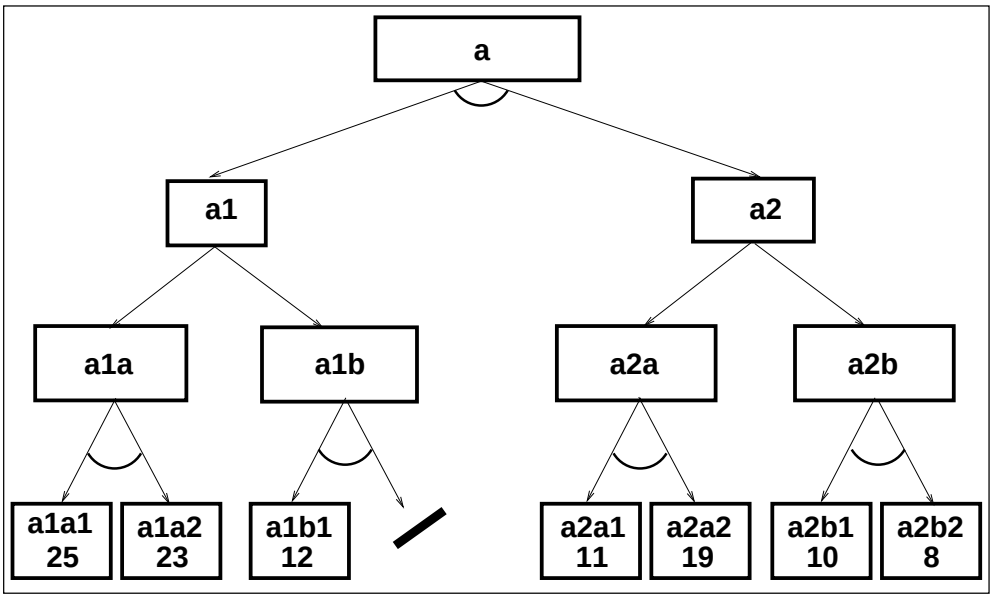

Figura 14: Árvore de jogo ainda maior para se diferenciar a primeira classe de poda

- Pode ser visto que, primeiramente, expande-se a árvore de jogo (Figura 12) gerando e avaliando heuristicamente os nodos $a1a$ ($FAE = 12$) e $a1b$ ($FAE = 11$), correspondendo aos estados "2" e "3", respectivamente, da árvore de táticas (Figura 13).

- A próxima tática a ser expandida é a de maior valor heurístico ($a1a$).

- A tática $a1a$ é então expandida para incluir mais uma opção de MAX para o caso de MIN fazer o movimento $a2$.

- Isso resulta na tática $a : \{a1a, a2a\}$, a qual continua com o valor heurístico 12 apesar do valor heurístico da configuração $a2a$ ser 25 (lembrete: o valor de uma tática é o mesmo da folha de menor valor).

- Logo, ao ser gerado o nodo $a : \{a1a, a2a\}$, não faz sentido expandir a tática $a : \{a1a\}$ por meio de novas combinações entre o nodo $a1a$ e as outras opções de $a2$ (além de $a2a$).

- Da mesma maneira, não faz sentido expandir $a : \{a1b\}$, gerando assim uma poda dupla na árvore OU do espaço de busca.

- Em ambos os casos acima seus valores serão $FAE \leq 12$, e como estes valores só podem diminuir mais ainda (incluindo folhas com $FAE < 12$), não há forma de gerar nenhuma tática melhor do que a tática completa já montada ($a : \{a1a, a2a\}$).

ATENÇÃO!!! Apesar de parecer que este último procedimento de poda dupla na árvore de táticas seria aplicável à primeira categoria de podas (da Figura 11) em táticas apresentada, a rigor, isso não cabe. Para esclarecer melhor a negativa, é preciso considerar uma árvore de jogo com um nível a mais, como a apresentada na Figura 14.

- A árvore da Figura 14 inclui toda a árvore da Figura 11 no ramo à esquerda da raiz.

- A Figura 15 apresenta o início da árvore de táticas correspondente à da Figura 14.

- Se ignorarmos os dois nodos do nível inferior, a árvore da Figura 15 mostra o estado no momento em que acabamos de avaliar os nodos $a1a1$, $a1a2$ e $a1b1$.

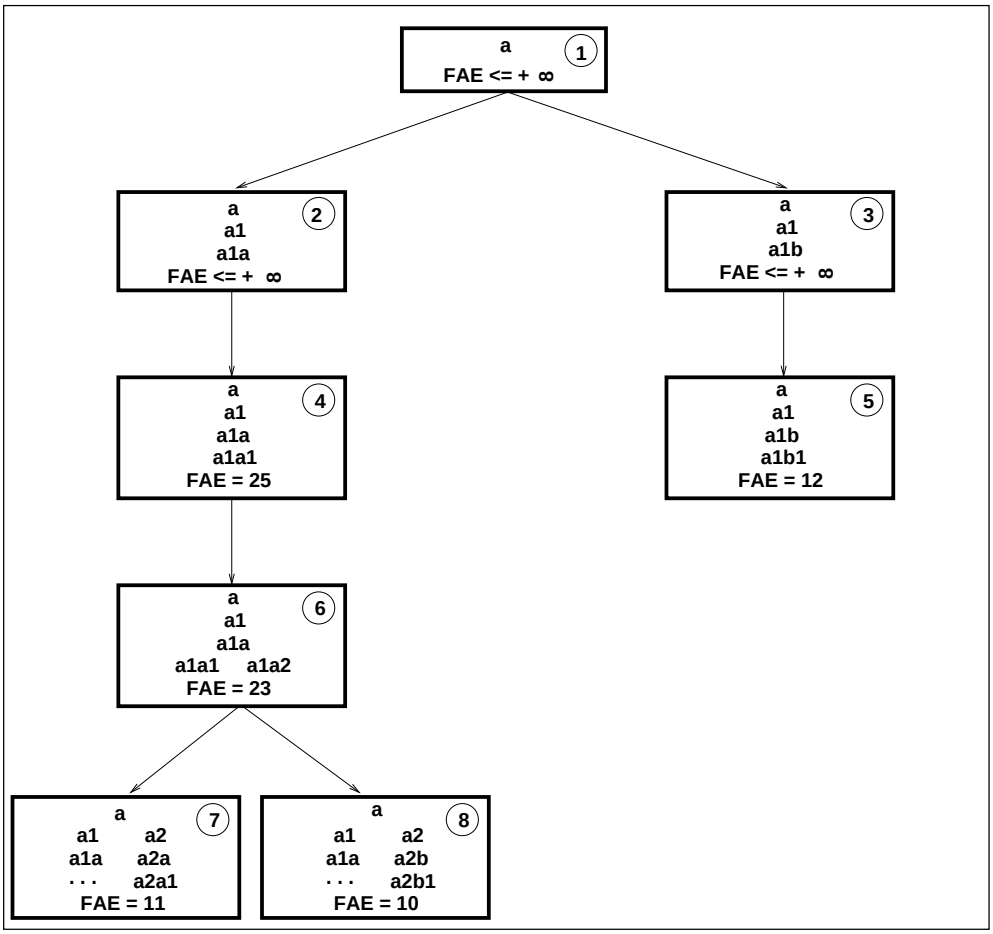

Figura 15: Árvore de táticas para o exemplo da árvore de jogo ainda maior

- Como é o ramo da esquerda da árvore da Figura 15 que tem o maior valor heurístico, 23, é nele que a expansão se dará.
- Além disso, a forma de expansão se dá por meio da investigação do que aconteceria se MIN fizesse o movimento $a2$.
- Isso pode deixar transparecer que o nodo "poda" (Figura 14) não seria mais investigado.
- Todavia, isso não corresponde à realidade.
- De acordo com o que está na parte inferior do ramo esquerdo da árvore de táticas da Figura 15, a exploração da plausibilidade de MIN realizar o movimento para o estado $a2$ mostrará que as duas possíveis respostas para tal movimento levam a táticas parciais cujas FAEs são 11 e 10.
- Como ambas são menores que 12, a tática parcial anteriormente ignorada voltaria a ser interessante.
- Só que (AGORA SIM!!!), como mostra a Figura 14, isso seria inútil.
- Ou seja, a expansão que gerou os estados de FAEs 11 e 10 não tem absolutamente nada a ver com a opção $a1a$ ser melhor que $a1b$ mas apenas com o fato do movimento para o estado $a2$ ser mais plausíivel para MIN.
- O mesmo não aconteceria se tentássemos aumentar a árvore de jogo da Figura 12 (mostre isso por você mesmo!).

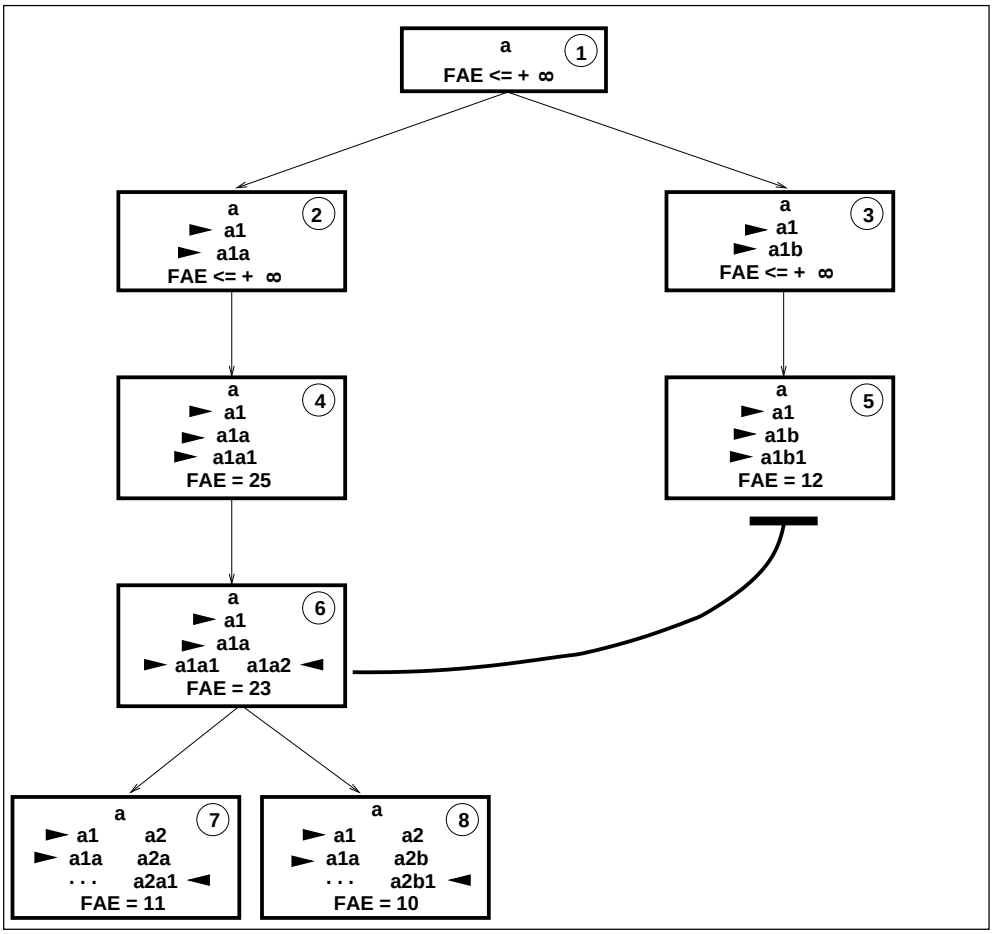

Figura 16: Árvore de táticas com nodos resolvidos

- Para impor prioridade para a poda mostrada na árvore da Figura 14, é necessária a formalização do tratamento automático deste contexto de busca, o que pode ser feito por meio do conceito de nodo resolvido da árvore de jogo.

- Um nodo resolvido é aquele no qual nenhum esforço posterior deve ser investido.

- Sua definição é a seguinte: (1) um nodo folha estará resolvido quando ele tiver sua FAE calculada; (2) um nodo não-folha para movimento de MAX estará resolvido quando o seu descendente de maior FAE estiver resolvido; (3) um nodo não-folha para movimento de MIN estará resolvido quando todos os seus descendentes estiverem resolvidos.

- Na Figura 14 o nodo $a1$ estará resolvido depois que $a1a1$, $a1a2$ e $a1b1$ estiverem resolvidos, já que ter $a1b2$ como resolvido não causaria aumento no valor de $a1$.

- Na Figura 16 foram indicados com setas os nodos resolvidos em cada uma das táticas.

- A vantagem disso é que a partir do ponto em que um nodo esteja resolvido, todas as demais táticas parciais que requerem expansão dependente deste nodo podem ser removidas da fila de prioridades que constitui os estados da árvore de táticas.

- No exemplo da linha curva da Figura 16, a tática parcial do lado esquerdo da linha possui o nodo $a1$ resolvido, logo a tática parcial prestes a ter seu contexto de expansão restaurado a partir de $a1b1$ será removida da fila de prioridades por ainda estar tentando resolver o mesmo nodo $a1$ (já resolvido).

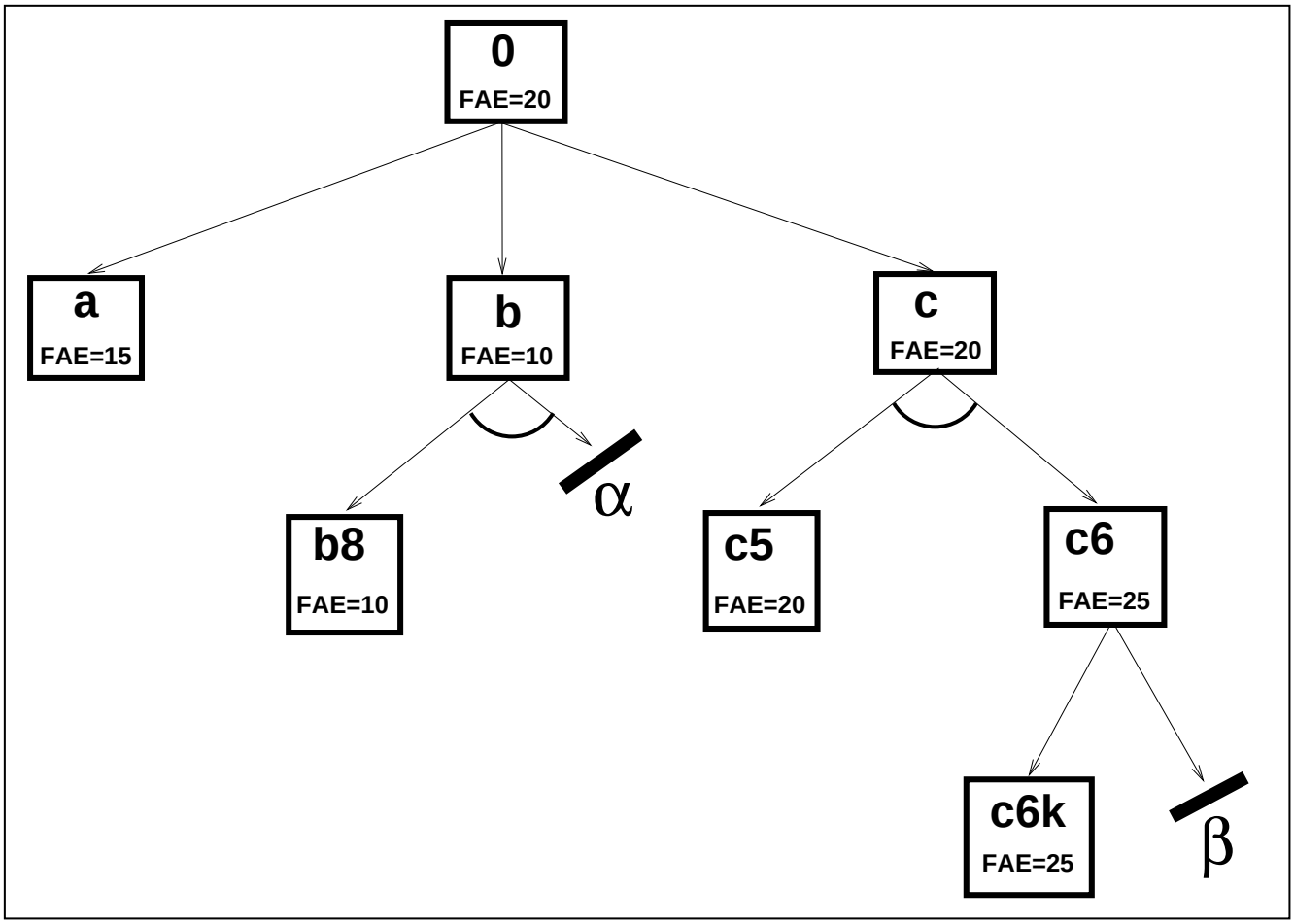

Figura 17: Árvore de jogo com exemplos de podas α e β

4.2 Algoritmo *Poda* α–β

A variação mais amplamente conhecida da busca $MINIMAX$ é o algoritmo *Poda* α–β. Ele é diferente do SSS^* apresentado na Subseção anterior na medida em que caminha exclusivamente em profundidade na árvore de jogo. Porém, de maneira semelhante ao SSS^*, o *Poda* α–β também usa valores da FAE para podar a busca.

Muito do que foi apresentado na Sub-seção anterior vale também para o *Poda* α–β. Com isso, em seguida, estão apresentadas principamente as características do algoritmo, ressaltando mais as diferenças em relação ao anterior.

- Como o *Poda* α–β realiza a busca exclusivamente em profundidade e com valor pré-fixado (N) para a "profundidade de visada", em qualquer instante da busca só há, no máximo, $N + 1$ nodos alocados na memória.

- A FAE só é aplicada no nível máximo ("profundidade de visada") para que seu valor seja transportado para os níveis superiores por meio do final de uma ativação recursiva (retorno de nível na árvore de jogo).

- As podas são de dois tipos, α e β. Elas são sempre tentadas com base no valor de FAE transportado depois de um retorno de nível da árvore de jogo.

- A *Poda*–α é aquela que ocorre em um nível de minimização de valores da FAE (transportados de baixo para cima).

- A Figura 17 mostra um exemplo de *Poda–*α, onde o valor de $FAE = 10$ transportado para a configuração do estado $b1$ já permite decidir que a configuração do estado a é melhor que a configuração do estado b.

- A *Poda–*β é aquela que ocorre em um nível de maximização de valores da FAE (transportados de baixo para cima).

- De forma semelhante, a mesma Figura 17 mostra um exemplo de *Poda–*β, onde o valor de $FAE = 25$ obtido a partir a avaliação heurística da configuração do estado $c2a$ já permite decidir que a configuração do estado $c1$ é melhor que a configuração do estado $c2$.