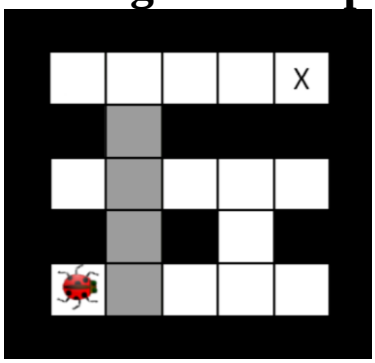
Introdução à Inteligência Artificial — Tipo – 2019/20

Nº Aluno:_____Nome Completo:_____

Exame com consulta. Responda às perguntas nesta própria folha, nos espaços indicados. Duração: ***2h30m***

(I)Paradigma do Espaço de Estados (3.5 valores)



Considera um labirinto quadriculado bidimensional com uma joaninha que parte de uma posição inicial pretendendo chegar ao seu destino, marcado com X, no menor número de passos, e que se pode mover nas 4 orientações ortogonais (norte, sul, este e oeste) ou ficar na mesma casa. Todas as acções têm um custo de 1. A Joaninha tem sempre de passar través de túneis parcialmente inundados (a cinzento na figura). Para passar através desses túneis a joaninha tem de reter a respiração e ela pode retê-la durante um número A de instantes (cada acção corresponde a um instante). Sem ar não pode estar numa casa inundada.

A acção de se mover mover para uma casa inundada custa ao insecto vermelhinho o gasto de 1 unidade de ar enquanto que ao mover-se para uma casa livre lhe permite preencher o seu reservatório de ar. Um exemplo de labirinto é o da figura ao lado esquerdo, em que a joaninha começa na casa mais ao fundo e à esquerda e pretende ir para a casa marcada com um X no menor número de passos

a) Como representarias o problema em termos de um espaço de estados, com a informação mínima necessária? Notem que essa representação deve ser usada para labirintos genéricos, não se podendo restringir à particularidade do labirinto da figura, mas obrigando sempre a que a Joaninha tenha de atravessar túneis inundados. Indique uma representação mínima para os estados (com o estado inicial e final) e os operadores de transição entre estados.

No estado teremos apenas a informação que muda com as acções: neste caso, a posição do Pacman e a quantidade de ar que tem disponível.

Assim, um estado será um par (P,Q):

- P é a posição do Pacman, dada por um par de coordenadas x e y, em que o ponto (0,0) é a casa mais à esquerda e em baixo.
- Q é a quantidade de ar, que é um valor entre 1 e A.

O estado inicial é (Pi,A) em que Pi é a posição inicial do Pacman, no caso da figura seria (1,1).

O estado final será qualquer em que a posição do par seja final(Pf): (Pf,*).

Para além do estado teremos que conhecer as casas navegáveis do labirinto e as inundadas. Mas serão informação estática do problema, estando fora do estado, sendo necessária para as transições entre estados..

Teremos 5 acções: N, S, L, O, F, em que N, S, L e F, todas com o mesmo custo (1) e que correspondem ao Pacman mover-se para a casa adjacente a norte, sul, leste e oeste, respectivamente e F corresponde a ficar na mesma casa. Notem que as acções podem nem sempre existir para certos estados, dado que há paredes e os limites do mundo.

Para qualquer das acções v, sendo Pv a casa vizinha de P, pela acção v:

- (P,Q) -> (Pv,Q-1) se Pv estiver inundada e Q > 1

- (P,Q) -> (Pv,A) se Pv não estiver inundada.

b) Esboça uma dimensão para o espaço de estados

Um limite superior ou valor aproximado será NxMxA em que N é o número de linhas, M o número de colunas e A a quantidade máxima de ar.

c) A heurística que corresponde à distância de manhatan da Joaninha ao objectivo + A é ou não uma heurística consistente? Justifica a tua resposta.

Não é consistente porque no objectivo as heurísticas consistentes terão de ser sempre 0. Não é o caso. Qualquer posição do estado final dista 0 de uma posição final, mas a heurística indica 0+A e sendo A > 0, essa condição não pode ser satisfeita.

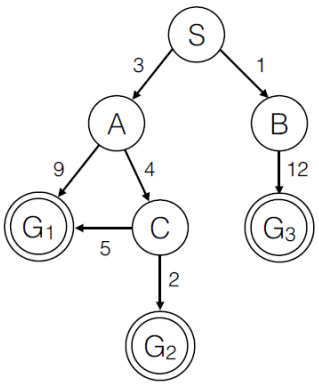
d) O algoritmo de aprofundamento progressivo em árvore seria uma boa opção para este problema? Justifica a tua resposta.

Por ser optimal no caso de custos homogéneos, o que é o caso, seria uma boa solução mas poderia ser muito ineficiente por ser em árvore, Num labirinto grande em espaço aberto, poderia ser muito pesada a procura em termos computacionais, devido à replicação dos estados na árvore de procura.

(II) Procura num Espaço de Estados (2.5+1)

Um aspecto importante nos algoritmos de pesquisa prende-se com a resolução dos casos de empates. Por norma nos exames usa-se a ordem alfabética e a antiguidade. Neste caso, vamos fazer uma variação e tratar dos empates de modo aleatório: em caso de empate, escolhe-se para expansão um dos nós ao acaso.

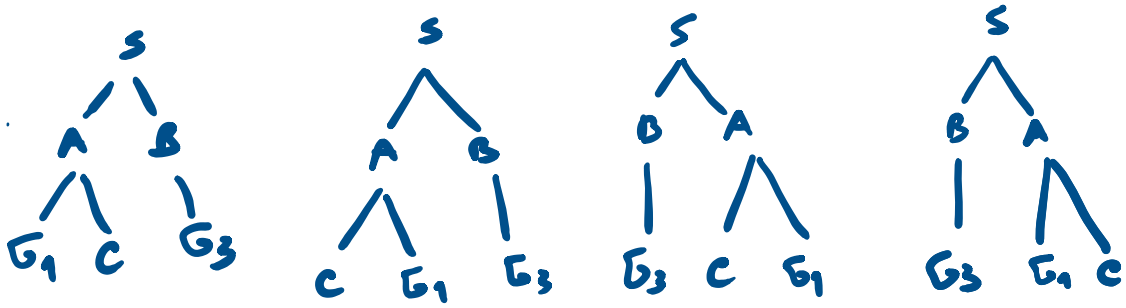
a. Para o grafo e as heurísticas em baixo, considerando que são 0 nos estados finais, seleccione todos os estados finais, G1, G2 ou G3, que podem ser atingidos pelos algoritmos indicados, a partir de S.



	A	B	C	S
H-1	0	0	0	0
H-2	6	7	1	7
H-3	7	7	1	7
H-4	4	7	1	7

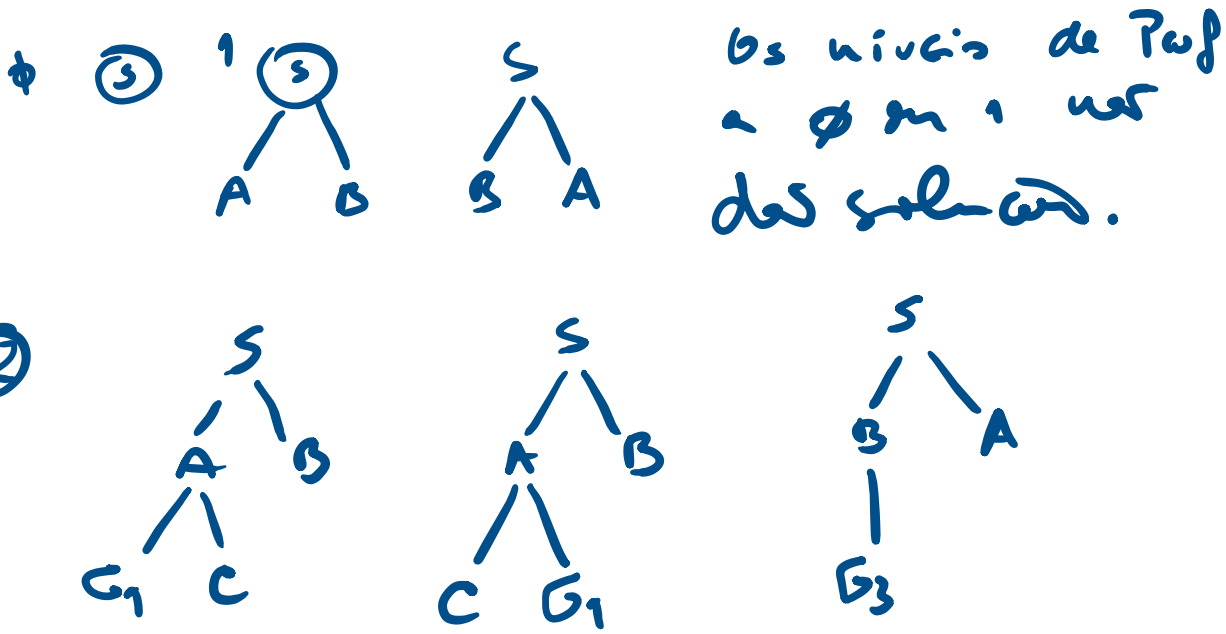
Largura em Grafo	G1	G2	G3
Aprof. Progressivo	G1	G2	G3
A* (H1)	G1	G2	G3
Melhor Primeiro em grafo (H3)	G1	G2	G3
Melhor Primeiro (H4)	G1	G2	G3
A* com (H3)	G1	G2	G3
A* em grafo (H2)	G1	G2	G3
Melhor Primeiro (H2)	G1	G2	G3

Na Largura em Grafo, temos 4 árvores possíveis devido ao desempate aleatório:



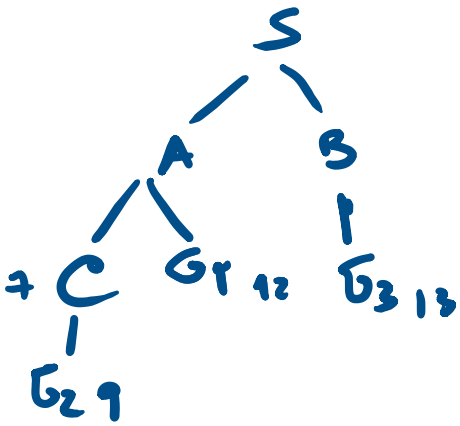
Pelas 4 árvores podemos ver que a largura primeiro poderia atingir G1 ou G3. (Tanto em grafo como em árvore)

No aprofundamento progressivo, acontece a mesma coisa, porque devolve a solução que esteja mais próxima em termos do número de acções.

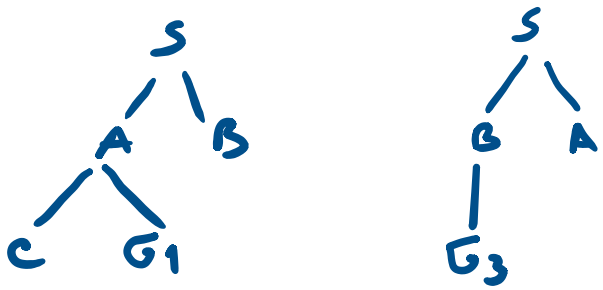


A nível 2, 3 árvores e duas soluções: G1 e G3.

A* com H1 (é o mesmo que o custo uniforme), a única solução é G2

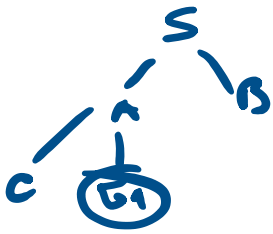


Com Melhor Primeiro (H3) teremos as seguintes árvores:



E pode atingir tanto G1 como G3, devido ao empate de h(A)=h(B)=7.

Com Melhor Primeiro(H4 e H2) teremos:

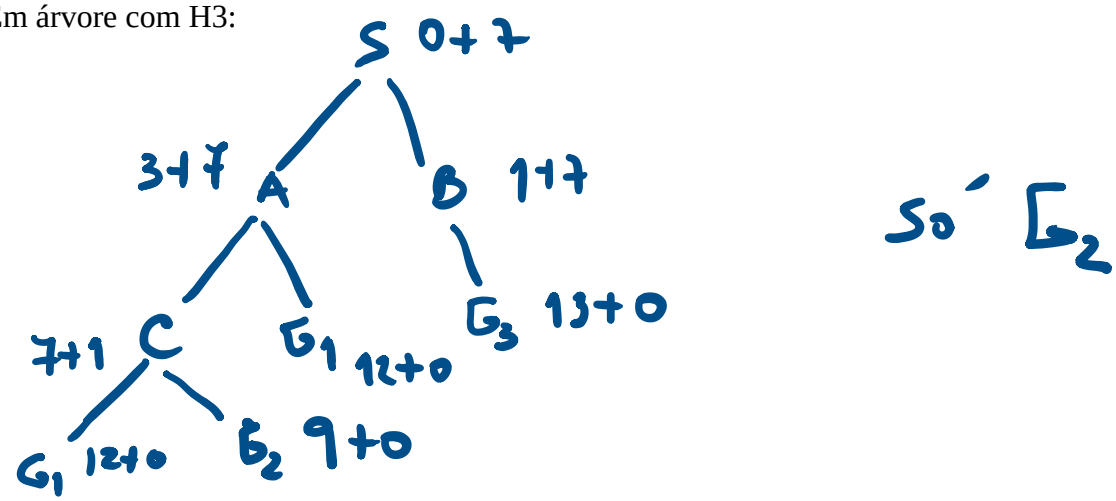


Prefere A

E só pode atingir G1 (não houve empates)

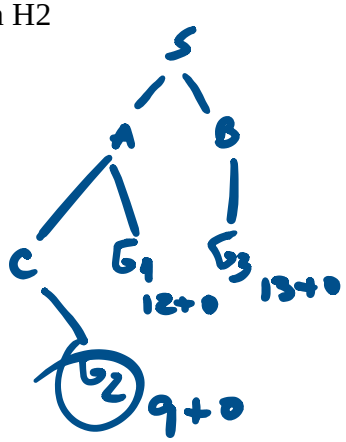
Vejemos os vários A*.

Ém árvore com H3:



Só G2

Em grafo com H2



Só G2

b. Para que heurísticas, H1, H2, H3 ou H4, é que o A* com optimização, i.e. com programação dinâmica, garante sempre a solução óptima, partindo de qualquer dos nós como inicial. Uma solução óptima é a de menor distância a um qualquer estado final.

Para garantir o óptimo na versão optimizada do A*, as heurísticas têm de ser consistentes.

Uma heurística H é consistente se para todos os arcos dirigidos x->y, satisfaz $H(x)-H(y)\leq c(x,y)$

H1 é consistente porque para é trivial, 0 para todos os estados, qualquer arco x->y, $H1(x)-H1(y)\leq c(X,Y)$, porque os custos são positivos sempre e h(s)=0 para todo o s, logo $H1(x)-H1(y)=0-0=0\leq c(x,y)$.

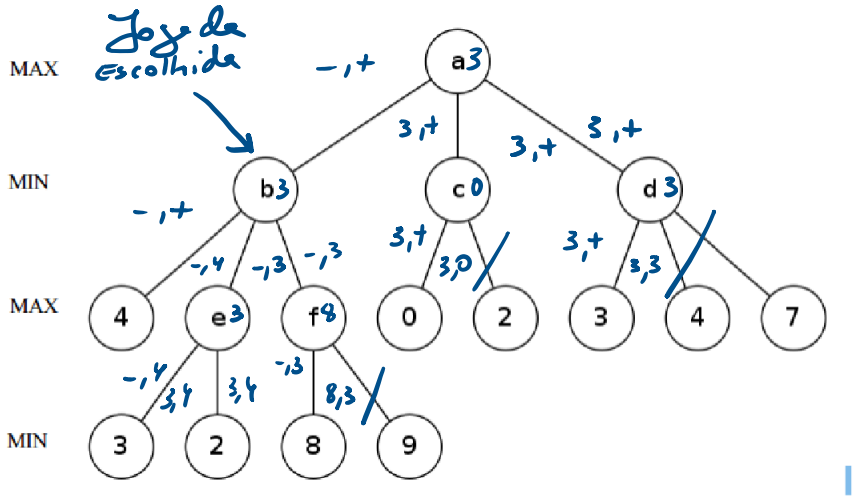
$H_2(A)-H_2(C)>c(A,C)$ porque 6-1>4, negando a consistência de H_2

$H_3(A)-H_3(C)>c(A,C)$ porque 7-1>4, negando a consistência de H_3

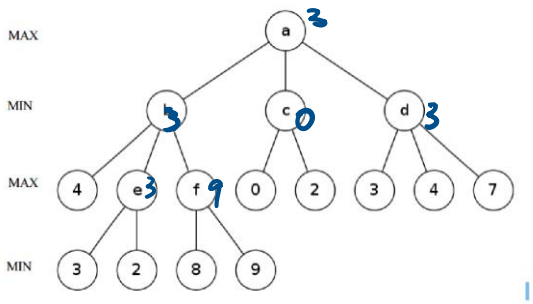
H_4 é consistente porque para todos os arcos dirigidos x->y, satisfaz $H_4(x)-H_4(y)\leq c(x,y)$

(III) Procura com Adversário (2+0.75+0.75)

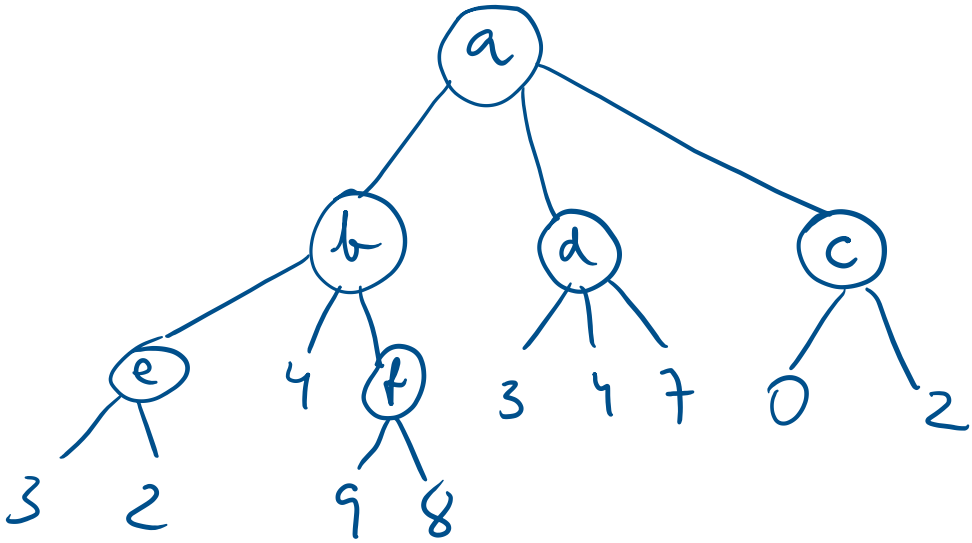
a) Considere a seguinte árvore de jogo e execute o algoritmo alfa-beta:



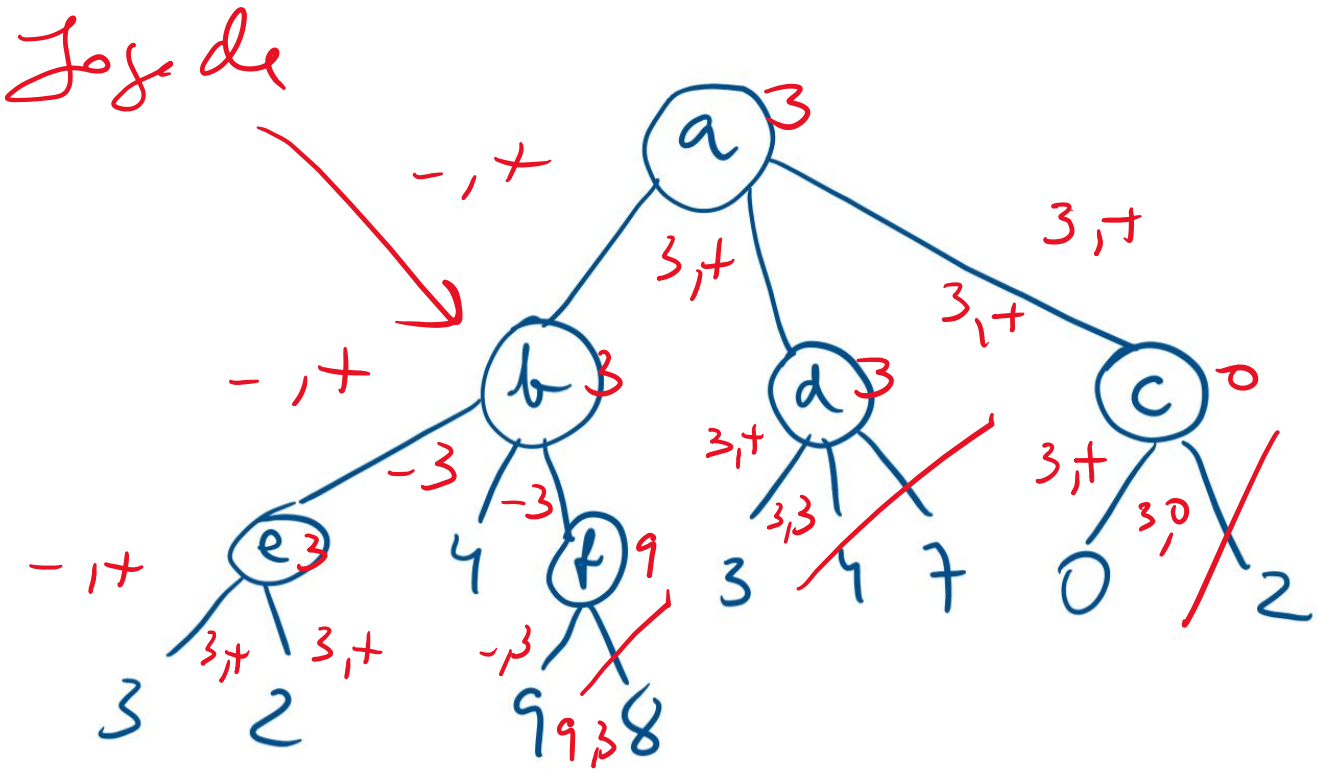
b) Para conseguir um número de cortes máximo, reordenam-se os sucessores dos nós.
Olhando para a árvore percebemos que é impossível haver mais cortes, mas sigamos o protocolo e calculemos os valores minimax dos vários nós.



Ordenamos os descendentes dos max por ordem decrescente do seu valor minimax e os descendentes dos nós min por ordem crescente.

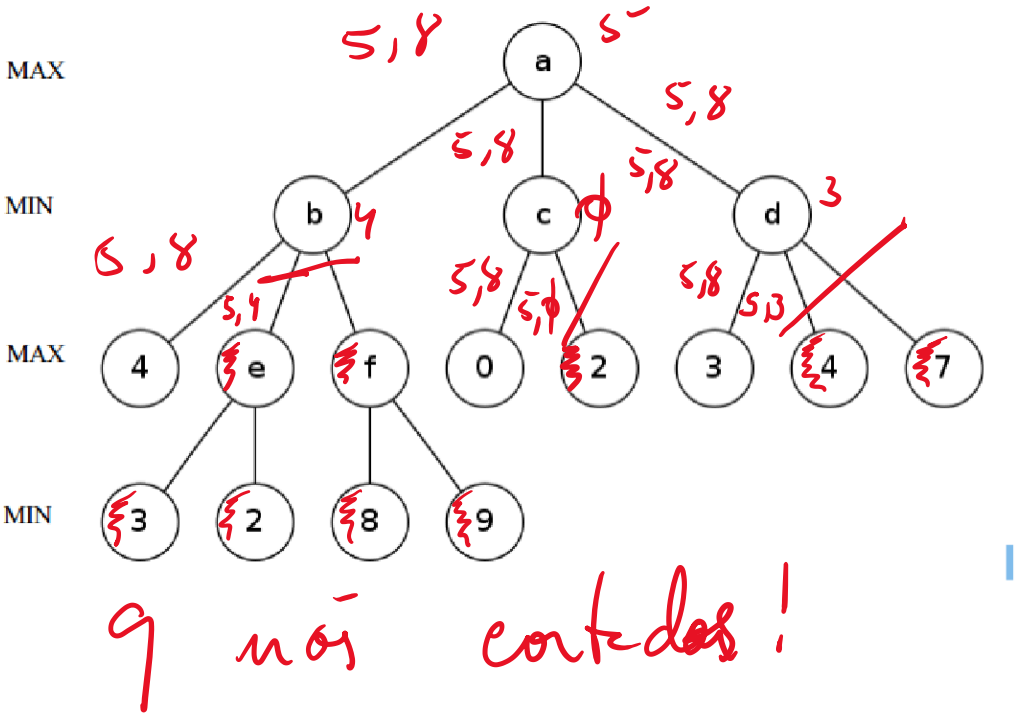


Não era necessário responder mas verifiquemos o que acontece com essa árvore reordenada.



Não houve melhoria, já era máxima, mas a técnica é sempre ordenar os sucessores dos max de dos maiores para os menores e os sucessores dos min por ordem crescente.

c) Aplique o algoritmo alfa-beta supondo um valor de alfa e de beta iniciais de 5 e de 8 respectivamente.



(IV) Problemas de Satisfação de Restrições (1.5+0.75+0.75)

1. O EstudantePac, BébéPac, MãePac, PaiPac, AvôPac, e o amigo fantasma estão alinhados e as posições estão numeradas de 1 a 6, em que 1 tem 2 como vizinha, 2 tem como vizinhas 1 e 3, etc., 5 tem como vizinhas 4 e 6, e 6 tem como vizinha 5. Cada uma ocupa exactamente uma posição. Sabemos que o BébéPac tem que ter o seu pai e a sua mãe ao seu lado. O AvôPac precisa de estar perto do fantasma. O EstudantePac tem de estar na posição 1 ou 2. Formula este problema como um problema de satisfação de restrições: lista as variáveis, os seus domínios e as restrições. Representa as restrições unárias como restrições em vez de filtrares os respectivos domínios.

Vamos ter 6 variáveis: E para EstudantePac, B para BébéPac, M para MãePac, A para AvõPac e F para o Fantasma.

Cada uma delas pode ter como domínio {1,2,3,4,5,6} que são as posições que cada um poderá ocupar.

Restrições:

- Todos são diferentes: alldiff(M,B,F,P,A,E)
- 1 = |A-F|
- 1 = |B – P|
- 1 = |B – M|
- E=1 ∨ E=2

b) Considera um PSR com as variáveis X e Y com os domínios respectivos de {1, 2, 3, 4, 5, 6} e {2, 4, 6} e com as duas restrições seguintes:

X < Y

X + Y > 8.

Lista os valores que permanecem no domínio depois de forçarmos a consistência dos arcos para o arco X → Y.

Para cada valor de X tem de haver pelo menos um valor de Y que satisfaz as restrições entre os dois.

Para X=6 não há nenhum valor de Y tal que X < Y.

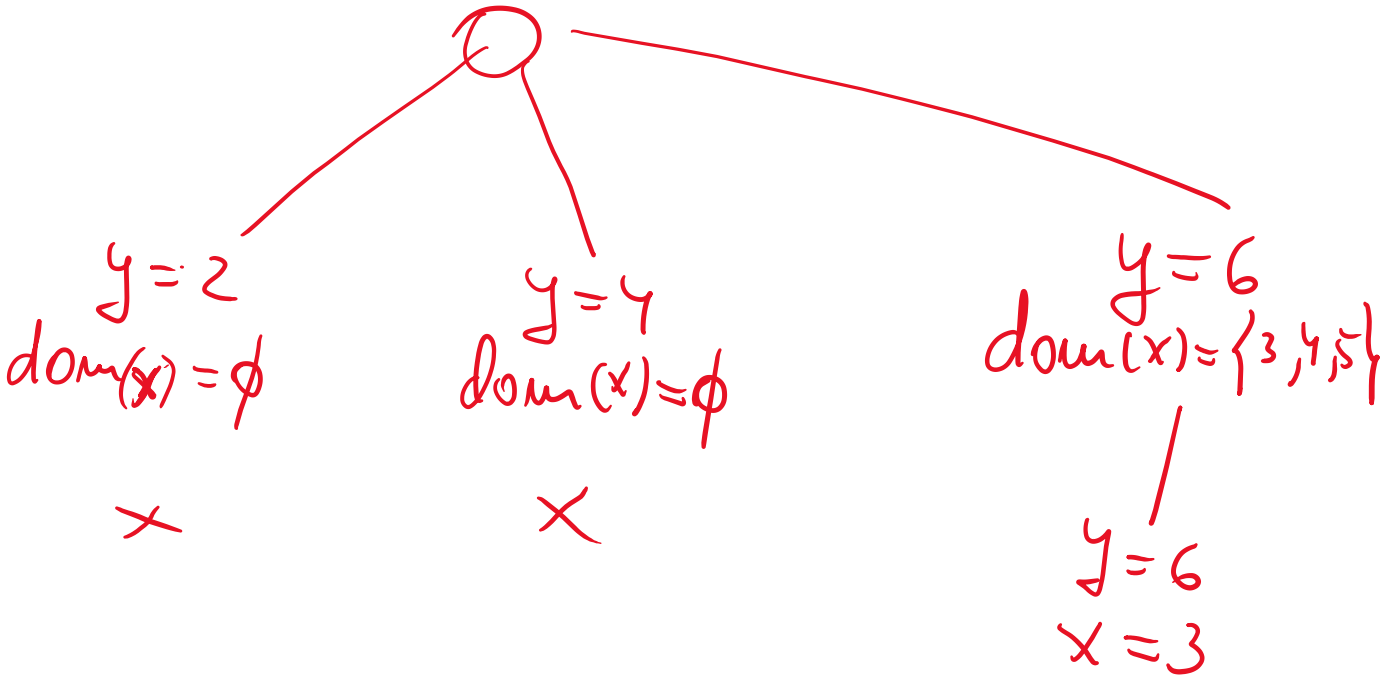
Para X=1 e para X=2 não há nenhum valor de Y que satisfaça X+Y>8

Depois de forçarmos a consistência de X -> Y

Dom(X)={3,4,5}

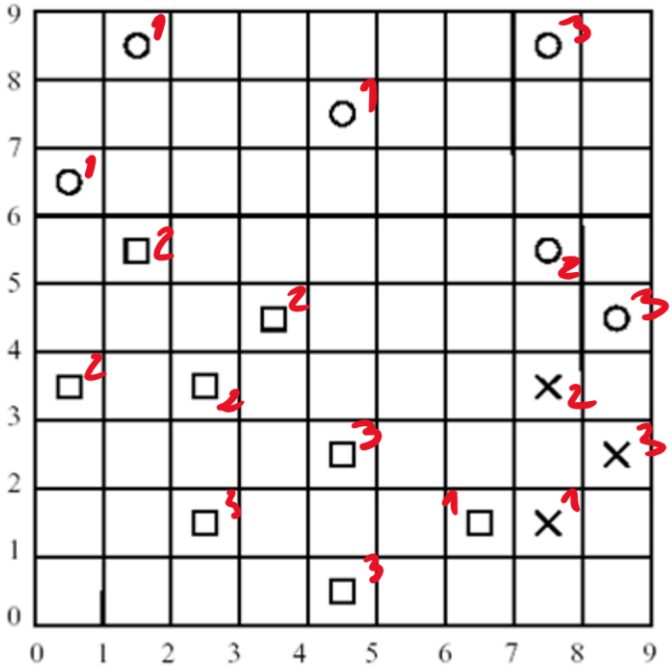
Dom(Y)={2,4,6}

c) Imagine que estamos a usar a procura com retrocesso, com verificação para a frente ("forward checking"), e que escolhemos a variável com menor domínio, para afectação, e os valores por ordem crescente. Desenhe a árvore de procura até à primeira solução.



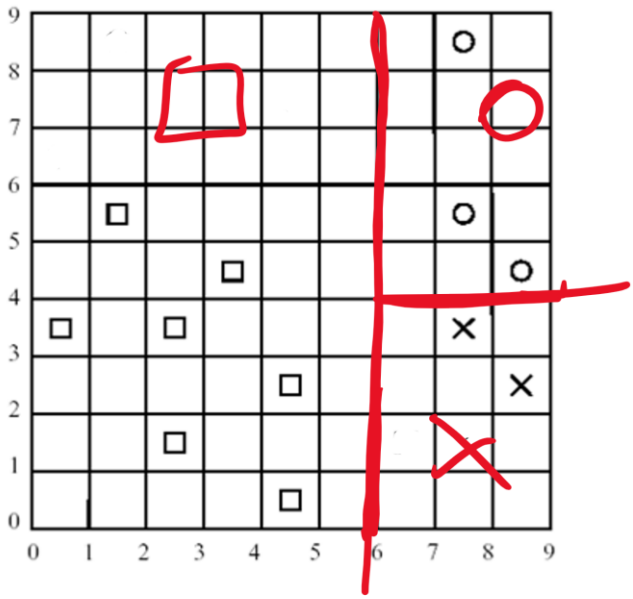
(V) Aprendizagem Automática (1.75+1.75)

Considere os dados apresentados na quadrícula a seguir, descritos por dois atributos (eixos x e y, com valores inteiros entre 0 e 9) e classificados em 3 classes, representadas por quadrados, X's e círculos. Usando o algoritmo ID3 estendido para fazer testes binários sobre atributos numéricos e uma validação cruzada em 3 grupos: 1, 2 e 3, indicados também na figura, construa a fronteira de decisão que o algoritmo daria para cada uma das árvores geradas e também a taxa de precisão total obtida. Não precisa de fazer contas.

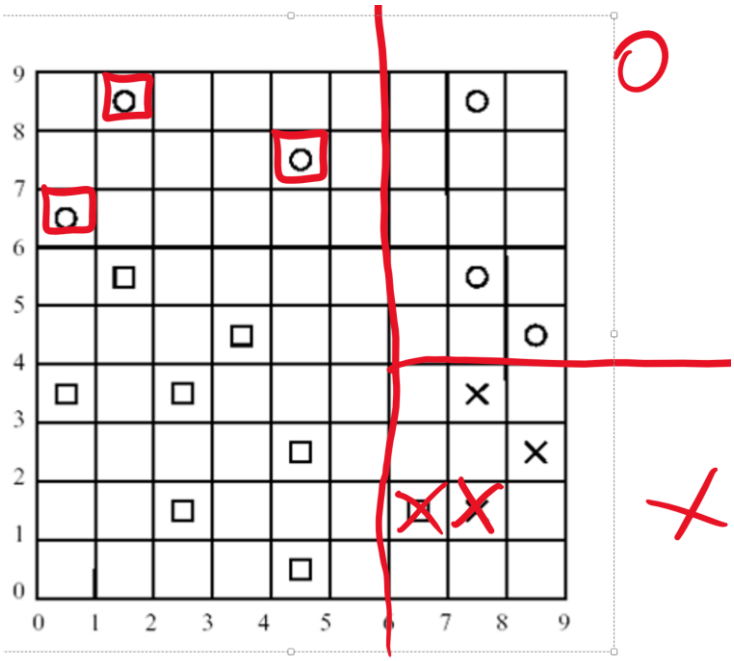


Vamos ver como ficam as grelhas no 3 casos e quais as árvores geradas, a olho.

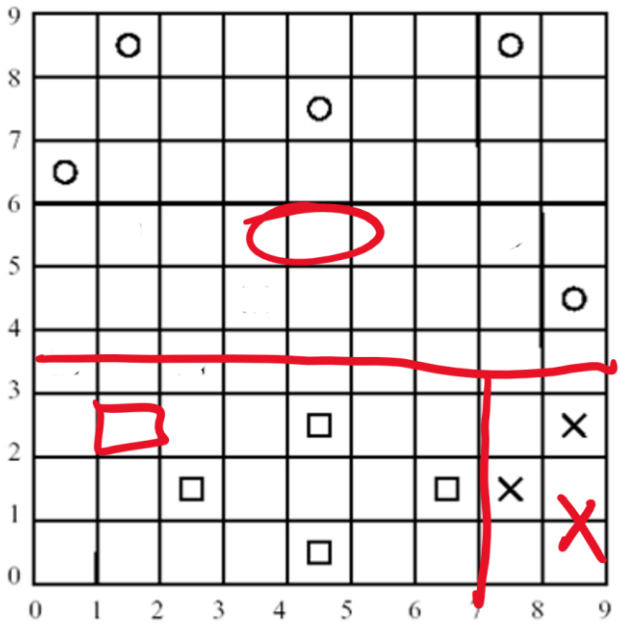
Quando o grupo 1 fica de fora teremos a seguinte fronteira de decisão



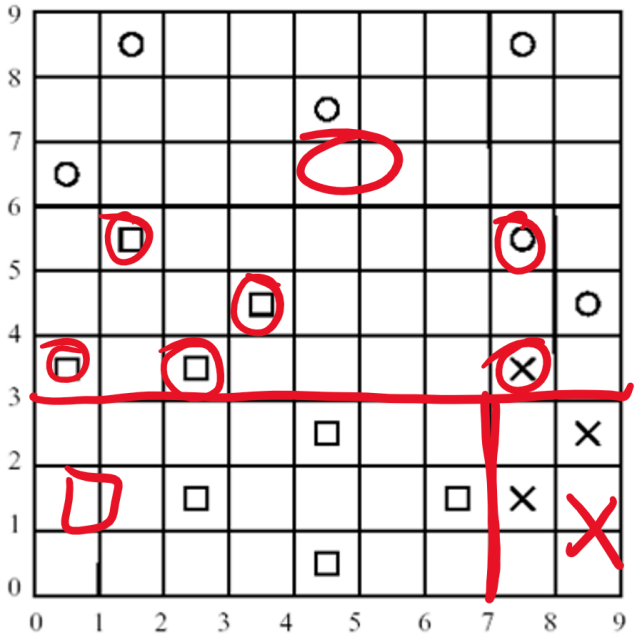
E os elementos de teste serão classificados da seguinte maneira dando origem a



Com o Segundo grupo de for a teremos a fronteira de decisão:

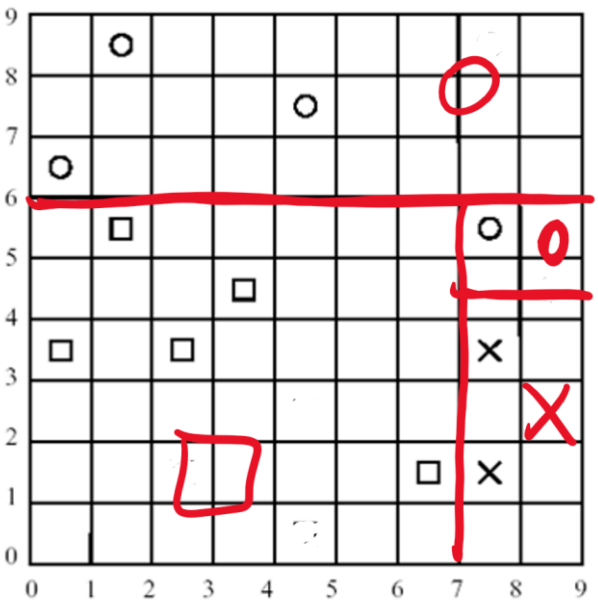


Que dará a seguinte classificação:

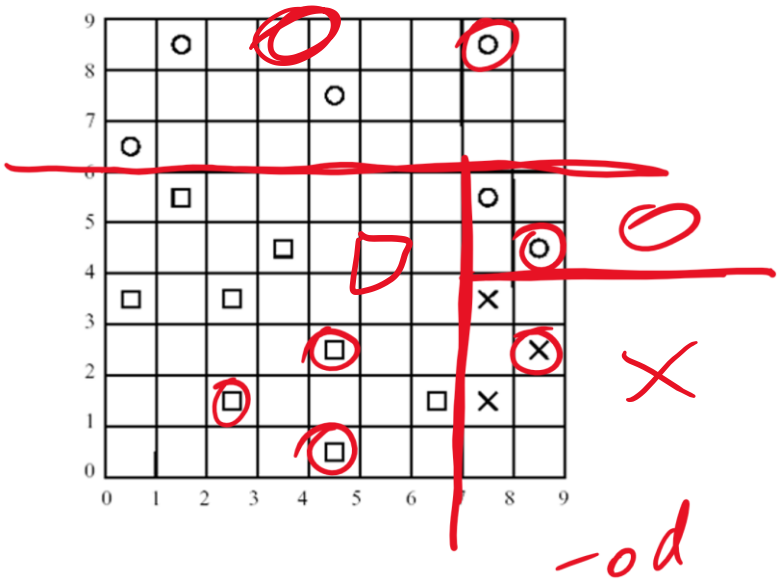


5 erros em 6

Finalmente o ultimo grupo:



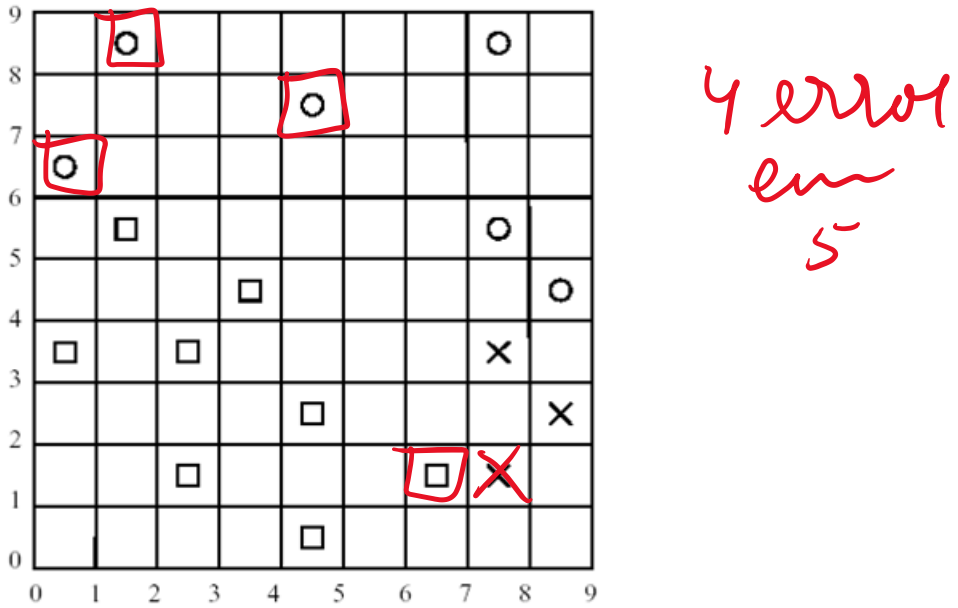
Os últimos dados para teste, do grupo 3 serão classificados:

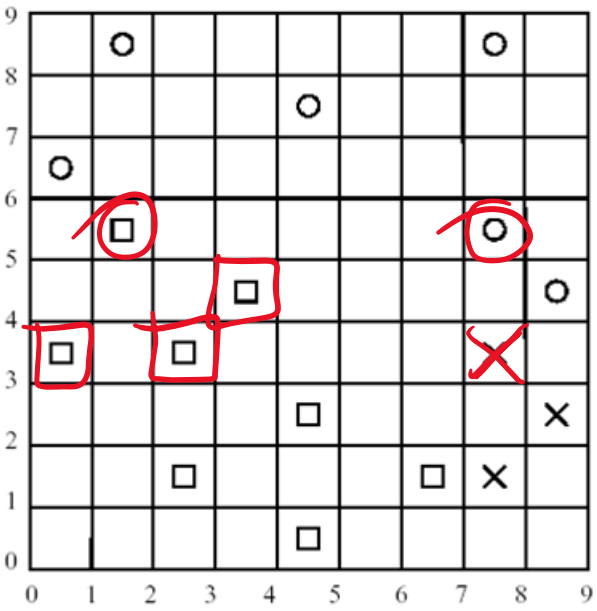


Total: 9 erros em 17

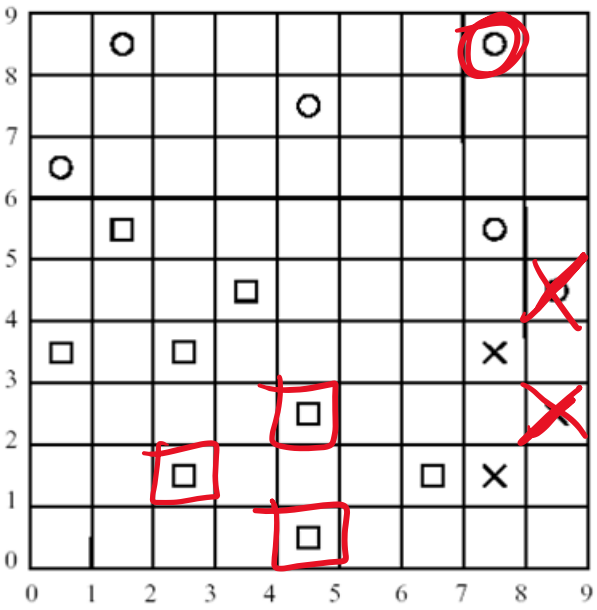
b) Compare essa taxa de precisão com o K-NN para os 3 vizinhos mais próximos, usando a distância de manhatan sem ponderação pela distância. Em caso de empate, decida-se pela classe maioritária.

Vamos fazer a validação cruzada e avaliar os dados de teste, com bola, para cada de treino. A classe prevista está a vermelho. Há dois empates resolvidos pela mais frequente do conjunto de treino,





1 erro em 6

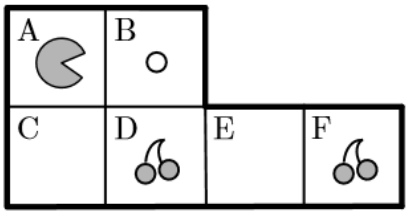


1 erro em 6

Total de 6 erros em 17

O K-NN teve uma melhor performance.

(VI) Processos de Decisão de Markov (2.25+0.75)



a) Considere o seguinte labrinto em que o Pacman deseja comer a pastilha em B e também as cerejas na células D e F. Se o Pacman comer uma cereja recebe 5 pontos e ela desaparece, e se comer a pastilha recebe 1 ponto e também desaparece. O jogo acaba quando o Pacman come a pastilha. O Pacman pode mover-se para norte, sul, leste e oeste, sempre de uma forma determinística, excepto se for impedido pelas paredes exteriores. Cada movimento tem um custo de 1 unidade (recompensa de estar vivo), como no Pacman verdadeiro.

Modelize o problema como um processo de decisão de Markov

Estados:

Vamos ter que representar no estado a posição do Pacman {A, B, C,D, E,F} e também se existe ou não a primeira cereja em D ou a cereja em F. Como o jogo acaba quando se come a pastilha, assumimos que há um estaod final: Exit que é absorvente. Os estados serão triplos formados pela célula e pelos booleanos que indicam se a cereja em D foi comida (T) ou não (F).

Teremos para este labirinto, como limite superior, 6 x 4 + 1 estados, i.e. 25 estados

Teremos como **transições**:

T(Exit,x,exit)=1

T((B, C_D,C_F),x,exit)=1

T((A,C_D,C_F),leste,(B, C_D,C_F))=1

T((A, C_D,C_F),sul,(C, C_D,C_F))=1

T((C, C_D,C_F),norte,(A, C_D,C_F))=1

T((C, C_D,C_F),leste,(D,T, C_F))=1

T((D, C_D,C_F),norte,(B, C_D,C_F))=1

T((D, C_D,C_F),leste,(E, C_D,C_F))=1

T((D, C_D,C_F),oeste,(C, C_D,C_F))=1

T((E, C_D,C_F),leste,(F, C_D,T))=1

T((E, C_D,C_F),oeste,(D,T,CF))=1

T((F, C_D,C_F),oeste,(E, C_D,CF))=1

A função recompensa é a seguinte:

R(X,F,C_F),a,(D,T,C_F)) = 4, para os vários X's e a's, que concretizam a transição entre triplos com esses padrões. Na verdade é só verdade quando X é C e a é leste. (temos de descontar a 5 o custo de estar vivo).

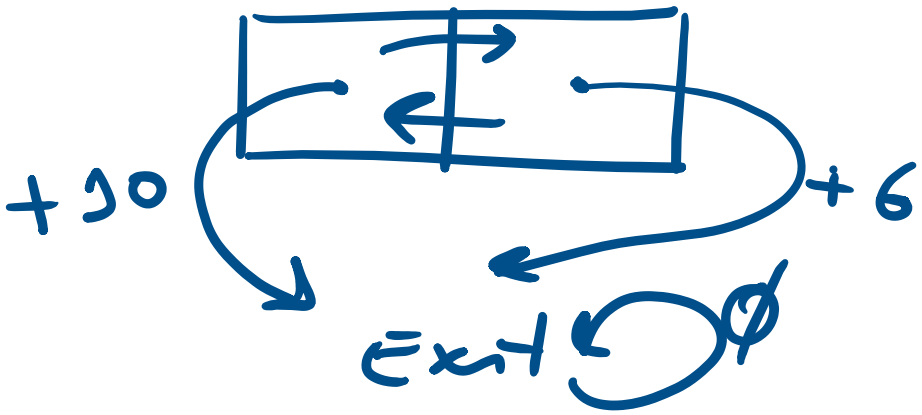
R(E,T,F),leste,(F, T,T,)) = 4

R(X, C_D,C_F),a,(B,T,C_F)) = 0 (1 da pastilha – 1 do custo do movimento)

Todas as outras transições levam a uma recompensa de -1

b) Indique, justificando, se a seguinte questão é verdadeira ou falsa: Se a única diferença entre dois PDMs fôr a taxa de desconto então têm de ter a mesma policy óptima

Esta afirmação é falsa!
Vejemos um contra exemplo na grelha com duas células, com as acções determinísticas de ir para leste ou oeste e de apanhar 10 pela esquerda ou 6 pela direita. Dois estados mais um estado terminal



Quando $\gamma = 1$, melhor Policy:

Quando $\gamma = 0.5$, a melhor policy

A Policy π:

Terá

que é pior