**Exame 2007 - Época Normal**

0

**a) Em que ordem a pesquisa em largura expande os nós?**

Solução caso usemos o Breadth-First Search sem uma melhoria na complexidade temporal/espacial, ou seja, um nó só é comparado com a solução quando é seleccionado para ser expandido.  
Seja Q a lista de nós na fila para expandir e E a lista de nós expandidos. Como o espaço de pesquisa é um grafo um estado que se encontre na fronteira, Q, não é adicionado novamente. Temos por isso:

Q=[S] E=[]

Q=[A,B] E=[S]

Q=[B,C,E] E=[S,A]

Q=[C,E,D] E=[S,A,B]

Q=[E,D,G] E=[S,A,B,C]

Q=[D,G] E=[S,A,B,C,E]

Q=[G] E=[S,A,B,C,E,D]

Q=[] E=[S,A,B,C,E,D,G]

Tendo sido seleccionado para expansão o nó solução encontramos a ordem pela qual os nós são expandidos, S,A,B,C,E,D,G. Complexidade espacial e temporal O(bd+1)

Solução caso usemos o Breath-First Search com melhoria na complexidade temporal/espacial, ou seja, quando um nó é gerado é comparado com a solução.

Seja Q a lista de nós na fila para expandir e E a lista de nós expandidos. Assumindo a mesma cena dos grafos que disse acima. Temos:

Q=[S] E=[]

Q=[A,B] E=[

Q=[B,C,E] E=[S,A]

Q=[C,E,D] E=[S,A,B]

Q=[E,D,G] E=[S,A,B,C]

Neste caso ao expandir-mos C gerou-se o estado G e sendo G a solução termina o algoritmo. Neste caso a lista de nós expandidos é S,A,B,C. Penso que esta é a resposta mais correcta pois esta tem uma complexidade e temporal e espacial O(bd).

**b) Em que ordem a pesquisa por aprofundamento progressivo expande os nós?**

p=1 : S,A,B ;

p=2 : S,A,C,E,B,C,D;

p=3 : S,A,C,G (após encontrar a solução, para)

depth=0

Q=[S] E=[]

Q=[] E=[S]

depth=1

Q=[S] E=[]

Q=[B,A] E=[S]

Q=[A] E=[S,B,A]

depth=2

Q=[S] E=[]

Q=[B,A] E=[S]

Q=[D,C,A] E=[S,B]

Q=[C,A] E=[S,B,D]

Q=[G,A] E=[S,B,D,C]

Gerou-se o estado final, G, por isso os estados que foram expandidos até se obter a solução foram:

S; S,B,A; S,B,D,C;

**c) Em que ordem o hill-climbing expande os nós?**

S,A,E -> se E gerado 1º que C

**d) Em que ordem a pesquisa gulosa (greedy) expande os nós?**

S,B,C,G

**e) Em que ordem o algoritmo A\* expande os nós?**

S,A,E,B,C,G

**1.2)Analise a possível aplicação dos métodos anteriores a um problema semelhante, mas com uma dimensão elevada, isto é, com alguns milhões de nós.**

Se para o problema em questão não interessa garantir uma solução óptima e se a completude do algoritmo for essencial, assim como existir preocupação na complexidade temporal e espacial então o algoritmo mais vantajoso seria o Hill-Climbing porque garante completude apesar de a solução encontrada poder não ser óptima. Em termos de complexidade espacial como não guarda a árvore de pesquisa consome pouca memória e normalmente tende a convergir rapidamente para uma solução. Uma variante deste algoritmo, Random-Restart Hill-Climbing, poderia ser usada p ara garantir optimalidade e completude mesmo para árvores de pesquisa com milhões de nós.

Se não existirem restrições de complexidade espacial nem temporal então a melhor escolha seria o A\* que garante optimalidade e é completo.

**2. Aprendizagem (4 Val)**

**a) Construa a árvore de decisão, para determinar se é concedido crédito a um**

**cliente, usando o critério da razão do ganho. Indique todos os cálculos que**

**efectuar. Escreva as regras obtidas para a classificação.**

info(C) = - 4/8 \* log2(4/8) - 4/8 \* log2(4/8) = 1

**1ª iteração**

info(C/est.civil) = 2/8 \* [- 1/2 \* log2(1/2) - 1/2 \* log2(1/2)] +

6/8 \* [- 3/6 \* log2(3/6) - 3/6 \* log2(3/6)]

= 1

*Ganho* = info(C)-info(C/est.civil) = 1-1 = 0

*infoSeparação* = - ( 2/8 \* log2(2/8) + 6/8 \* log2(6/8)) = 0.811

*Critério da Razão do Ganho* = Ganho/infoSeparação = 0

info(C/nº filhos) = 4/8 \* [- 3/4 \* log2(3/4) - 1/4 \* log2(1/4)] +

3/8 \* [- 1/3 \* log2(1/3) - 2/3 \* log2(2/3)] +

1/8 \* [- 1/1 \* log

= 0.75

*Ganho* = info(C)-info(C/nºfilhos)= 1-0.75 = 0.25

*infoSeparação* = - ( 4/8 \* log2(4/8) + 3/8 \* log2(3/8) + 1/8 \* log2(1/8))

= 1.406

*Critério da Razão do Ganho* = Ganho/infoSeparação = 0.1778

info(C/ordenado) = 4/8 \* [- 3/4 \* log2(3/4) - 1/4 \* log2(1/4)] +

2/8 \* [- 1/2 \* log2(1/2) - 1/2 \* log2(1/2)] +

2/8 \* [- 2/2 \* log2(2/2)]

= 0.66

*Ganho* = info(C)-info(C/ordenado)= 1-0.66 = 0.34

*infoSeparação* = -(4/8 \* log2(4/8) +2/8 \* log2(2/8)+2/8 \* log2(2/8)) = 1.5

*Critério da Razão do Ganho* = Ganho/infoSeparação = 0.34/1.5 = 0.227

**2ªIteração (escolhendo ordenado uma vez que tem maior razão do ganho)**

info(C/est.civil)(ordenado=baixo) = 1/4 \* [-1/1 \* log2(1/1)] +

3/4 \* [-2/3 \* log2(2/3) - 1/3 \* log2(1/3)]

= 0 + 0.689 = 0.689

*info(C)* = - 1/4 \* log2(1/4) - 3/4 \* log2(3/4) = 0.811

*Ganho* = info(C)-info(C/est.civil)= 0.811-0.689 = 0.122

*infoSeparação* = - ( 1/4 \* log2(1/4) + 3/4 \* log2(3/4)) = 0.811

*Critério da Razão do Ganho* = Ganho/infoSeparação = 0.150

info(C/est.civil)(ordenado=medio) = 2/2 \* [-1/2 \* log2(1/2) - 1/2 \* log2(1/2)]

= 1

*info(C)* = - 1/2 \* log2(1/2) - 1/2 \* log2(1/2) = 1

*Ganho* = info(C)-info(C/est.civil)= 1-1 = 0

*infoSeparação* = - 2/2 \* log2(2/2) = 0

*Critério da Razão do Ganho* = Ganho/infoSeparação = 0

info(C/est.civil)(ordenado=alto) = 1/2 \* [-1/1 \* log2(1/1)] +

1/2 \* [-1/1 \* log2(1/1)]

= 0

*info(C)* = - 2/2 \* log2(2/2) = 0

*Ganho* = info(C)-info(C/est.civil)= 0-0 = 0

*infoSeparação* = - ( 1/2 \* log2(1/2) + 1/2 \* log2(1/2)) = 1

*Critério da Razão do Ganho* = Ganho/infoSeparação = 0

info(C/nºfilhos)(ordenado=baixo) = 3/4 \* [-3/3 \* log2(3/3)] +

1/4 \* [-1/1 \* log2(1/1)]

= 0

*info(C)* = - 1/4 \* log2(1/4) - 3/4 \* log2(3/4) = 0.811

*Ganho* = info(C)-info(C/nºfilhos)= 0.811-0 = 0.811

*infoSeparação* = - ( 1/4 \* log2(1/4) + 3/4 \* log2(3/4)) = 0.811

*Critério da Razão do Ganho* = Ganho/infoSeparação = 1

* + ordenado = baixo & nºfilhos = 1 ---> crédito = nao
  + ordenado = baixo & nºfilhos = 2 ---> crédito = sim

info(C/nºfilhos)(ordenado=medio) = 1/2 \* [-1/1 \* log2(1/1)] +

1/2 \* [-1/1 \* log2(1/1)]

= 0

*info(C)* = - 1/2 \* log2(1/2) - 1/2 \* log2(1/2) = 1

*Ganho* = info(C)-info(C/nºfilhos)= 1-0 = 1

*infoSeparação* = - ( 1/2 \* log2(1/2) + 1/2 \* log2(1/2)) = 1

*Critério da Razão do Ganho* = Ganho/infoSeparação = 1

* + ordenado = medio & nºfilhos = 1 ---> crédito = sim
  + ordenado = medio & nºfilhos = 2 ---> crédito = nao

info(C/nºfilhos)(ordenado=alto) = 1/2 \* [-1/1 \* log2(1/1)] +

1/2 \* [-1/1 \* log2(1/1)]

= 0

*info(C)* = - 2/2 \* log2(2/2) = 0

*Ganho* = info(C)-info(C/nºfilhos)= 0-0 = 0

*infoSeparação* = - ( 1/2 \* log2(1/2) + 1/2 \* log2(1/2)) = 1

*Critério da Razão do Ganho* = Ganho/infoSeparação = 0

****

**Regras:**

SE ordenado = baixo E nºfilhos = 1 ENTÃO crédito = nao

SE ordenado = baixo E nºfilhos = 2 ENTÃO crédito = sim

SE ordenado = medio E nºfilhos = 1 ENTÃO crédito = sim

SE ordenado = medio E nºfilhos = 2 ENTÃO crédito = nao

SE ordenado = alto ENTÃO crédito = sim

**b) Determine o valor máximo da razão do erro nas folhas da árvore construída na alínea anterior.**

Erro = (e+1)/(n+2)

Erro (Ordenado baixo e nº filhos = 2) = (0+1)/(1+2) = ⅓

Erro (Ordenado baixo e nº filhos = 1) = (0+1)/(3+2) = ⅕

Erro (Ordenado médio e nº filhos = 1) = (0+1)/(1+2) = ⅓

Erro (Ordenado médio e nº filhos = 2) = (0+1)/(1+2) = ⅓

Erro (Ordenado alto) = (0+1)/(2+2) = ¼

Erro = 0 em todas as folhas. porque

Sempre que conseguimos chegar a uma conclusão deixa de existir razão de erro. A razão de erro só faz sentido se a razão de ganho for diferente de 1.

**c) Explique um método de poda usado pelo C4.5. Quais as suas vantagens e inconvenientes? Justifique.**

O C4.5 usa um método de poda, este poda a árvore usando a informação do conjunto de treino. Seja S uma sub-árvore da Árvore de Decisão, calcula repetidamente para cada sub-árvore de S, a sua razão entre o aumento de erro e a complexidade e, sendo a razão inferior à da média das outras sub-árvores, substitui-a por uma folha.

As vantagens do C4.5 é o facto de tratar casos com valores não completamente especificados atribuindo-lhes um valor probabilístico e trata atributos que podem tomar valores em domínios contínuos. E combate o problema de overfitting, utilizando a estratégia de poda da árvore.

As desvantagens ….... ??

----//----

Existem dois métodos de podar a árvore em C4.5: sub-tree replacement ou sub-tree raising.

O sub-tree replacement faz exactamente aquilo que o nome indica: Substitui todos os nós/folhas que se encontram abaixo do nó/folha actual por um único nó. Existem vantagens, e uma delas torna-se na leitura simplificada da árvore. No entanto, é também possível obter uma desvantagem, visto que inúmeros nós da árvore são cortados, podendo haver inconsistências de resultado.

Já o sub-tree raising parte da “elevação” das árvores, ou seja, todos os caminhos que vão dar a um nó podem ser simplificados por um só caminho, e muito mais simples. Imagine-se que existem 10 caminhos que levam sempre ao mesmo resultado. É fácil simplificar a árvore e dizer que um único caminho é o mais acertado. Uma das desvantagens centra-se no facto de algumas informações se perderem da árvore, assim como conhecimento valioso que mais tarde poderia ser usado para a adição de nova informação.

**3. Algoritmos para a Evolução (5 Val.)**

**Suponha um puzzle semelhante aos dominós mas em que as peças, para além do número (0-6), têm também uma cor (A-C). Dado um conjunto de 10 peças previamente definido, pretende-se encontrar a solução óptima para o problema. Numa solução admissível, todas as peças em contacto têm o mesmo número. A solução óptima, respeita esta restrição mas ao mesmo tempo maximiza o número de peças com a mesma cor em contacto. Pretende-se resolver este problema utilizando algoritmos genéticos.**

**a) Defina o tipo de cromossomas a utilizar, sua representação e as características e um possível modo de obtenção da população inicial. (1 Val.)**

Em cada peça é necessário representar duas cores e dois números. Para representar cada cor será necessário 2bits , uma vez que existem 3 cores possíveis e 21 < 3 < 22, já para representar cada número será necessário 3bits, uma vez que existem 7 números possíveis e 22 < 7 < 23. Assim cada peça é representada por 10bits (Exemplo: 110 00 010 01 -> Peça 6A 2B).

Cada cromossoma inicial será constituído por 10peças, ou seja 10\*10bits, e poderá ser representado por um vector de strings, de tamanho 10.

A população inicial será constituída por N cromossomas cada um com 10peças previamente definidas e dispostas de forma aleatória.

**b) Qual a medida de avaliação (adaptação) a utilizar. Apresente o pseudo-código que permita calcular esta medida de avaliação (a partir da representação definida para a medida de avaliação). (1.5 Val.)**

fitness=0

ciclo da primeira peca até a ultima

se o seg num == primeiro num da peça seguinte

fitness+1

se segunda cor = primeira cor da peça seguinte

fitness+1

else

fitness = 0

break

fim do ciclo

retorna valor fitness

Se (n1 = n2) {

fitness.Valor++;

Se (cor1 = cor2) {

fitness.Cor++;

}

return fitness;

Na função de avaliação a importancia é o fitness.Valor e em caso de empate então aí é que entra o fitness.Cor

btw estou a considerar como melhor fitness o que tem o valor mais alto.

Não concordava com a função de cima porque é obrigatório que todas as peças em contacto tenham o mesmo número. A cor é uma coisa secundária, portanto fiz esta, vejam se concordam:

fitness = 0; i = 0;

ciclo por todos os pares de peças em contacto

i++;

if(peça1.n2 = peça2.n1)

fitness++;

end ciclo;

if(fitness==i)

ciclo por todos os pares de peças em contacto

if(cor1=cor2)

fitness++;

end ciclo;

return fitness;

else

return false;

**c) Suponha que utiliza um tipo de estratégia de selecção elitista e um método de cruzamento à sua escolha. Apresente o pseudo-código do algoritmo que lhe permite aplicar algoritmos genéticos na resolução do problema. (1.5 Val.)**

seleccao elitista=ordenar os cromossomas pelo melhor fitness e cruzar dois a dois pelos melhores

minha escolha=aleatorio

cruzamento: escolhia partir os cromossomas ao meio e depois fazer o cruzamento, pois desta forma não se misturava informação de cores e numeros de peças diferentes durante o cruzamento

Gerar um numero entre 1 e n-1(porque não vou partir no inicio e fim do chromossoma) multiplicava por 10 (tamanho de cada cromossoma se fossem de tamanho 7 multiplicava por 7) e realizava o cruzamento.

**d) Discuta a aplicação de outro(s) método(s) de resolução deste problema que lhe pareçam também adequados. (1 Val.)***número de cores que estão juntas.*

**4. Inteligência Artificial. (8 Val).**

**1) Qual o comportamento do algoritmo A\* se h(n) for uma heurística perfeita que retorna sempre o custo exacto para o objectivo? Justifique.**

“usando este algoritmo com uma heurística apropriada atinge-se um comportamento ótimo”->wikipedia

**2) Quais as diferenças entre os algoritmos “Arrefecimento Simulado” e “Subir a Colina”? Porque é que o efeito próprio do algoritmo de Arrefecimento Simulado se faz sentir sobretudo na fase inicial da pesquisa?**

|  |  |
| --- | --- |
| **Arrefecimento Simulado** | **Subir a Colina** |
| complexidade temporal -> depende da probabilidadex | complexidade temporal -> O(r\*p) |
| complexidade espacial -> depende da probabilidade | complexidade espacial-> O(r) |
| “retira” os mínimos e máximos | fica preso em máximos e mínimos locais |
| escolhe aleatoriamente o sucessor do estado actual | escolhe o sucessor + próximo da solução ou o sucessor melhor |
| Melhoria do subir a colina | ----- |

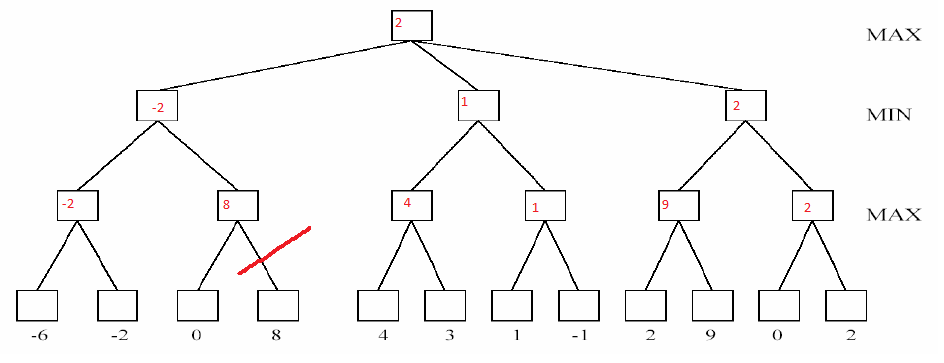
**3) O que se entende por uma estratégia de pesquisa óptima? Será que garante encontrar a solução num tempo óptimo. Explique e exemplifique.**

Uma estratégia de pesquisa óptima consiste no uso de técnicas que permitem encontrar, sempre, a melhor solução para o dado problema. Devido à complexidade que um método destes pode trazer, de forma a chegar à melhor solução em todas as situações, poderá não chegar a ela numa complexidade temporal linear. Por exemplo, no caso do algoritmo “Minimax”, é necessário recorrer a backtracking para fazer avaliação prévia de resultados candidatos; o que, num que não garanta a solução óptima, apenas interessa chegar à solução final, sem se preocupar com o que foi avaliado durante a sua operação.

**4) Explique como é utilizada a aleatoridade nos algoritmos genéticos?**

A aleatoridade está presente nos algoritmos genéticos nas fases de: Selecção, Emparelhamento, Cruzamento e Mutação. Na 1ª, não sendo esta forma de escolha imperativa (pode-se usar a selecção elitista), o gene pode vir a ser seleccionado por uma probabilidade, que se baseia na função de adaptação e no número total de indivíduos. A 2ª requer mesmo esta característica: baseia-se nela para verificar quais os genes, depois de seleccionados, farão o cruzamento propriamente dito. Opcionalmente, a forma aleatória pode ser usada para escolher os pares para esta operação. A 3ª faz uso da aleatoridade para escolher um (ou mais) ponto(s) de cruzamento para a reprodução dos filhos dos genes. A última recorre a esse factor, para adicionar um acontecimento que é uma perfeita analogia do que acontece no meio natural: Existe uma pequena possibilidade (vinda de diversas fontes) de que os genes possam vir a sofrer mutações.

**5) Calcule na árvore seguinte os valores minimax para cada estado, indicando o movimento escolhido por MAX (o computador) como o seu primeiro movimento a efectuar.**

****

(Pus cortes Alfa-Beta)

O computador segue o caminho: ele segue o caminho que o leva a por o 2 no top

**6) Uma companhia de seguros tem a seguinte tabela: Se nº filhos de fai = 2 ou 3 Então prémio de seguro = 1000 Segundo a teoria de Dempster-Shaffer, quanto é Plausível a companhia de seguros receber?**

S -> ter 2 ou 3 filhos

Bel(S) = 2/6

Bel(~S) = 2/6

Plausibilidade(S)= 1 - Bel(~S) ) 4/6

É plausível que a companhia de seguros venha assim a receber 4000 euros (4 casos plausíveis x 1000€).

PS- não sei se é só isto. parece-me demasiado fácil

**7) Suponha que deseja utilizar redes neuronais para dadas 1000 fotografias (de 5 objectos, i.e. 200 fotografias de cada um) efectuar o seu reconhecimento a partir de dez características a definir. Proponha uma arquitectura de rede, uma definição das características a extrair e forma de extracção e uma metodologia de treino/teste adequada.**

Arquitectura:

camada de entrada: 10 neurónios (1 para cada feature)

camada intermedia: 5 ou 6 no máximo

camada de saida: 5 -> um neurónio para cada imagem (5 imagens = 5 saidas)

Definição de caracteristicas a extrair:

eles nao falem disso em lado nenhum, supostamente esta parte não tem nada haver com a cadeira, no entando um exemplo, penso que pode ser o numero de pixeis pretos.

treino:

Treinar a rede com vários sets de cada tipo de imagem (200 sets de 5 imagens [uma de cada tipo diferente]) até o neurónio da saída com o valor máximo, ser a saída que a imagem deveria activar, e todas as outras saídas terem um erro superior a -0.5 e inferior a 0.5, evitando assim o overtrainnig da rede.

treino: utilizar 90% dos dados que temos para treinar os 5 tipos de imagens, ou seja, 180 fotografias cada.

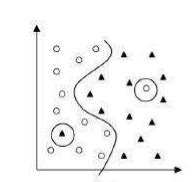
testes: utilizar os restantes 10% para validar o valor obtido vs o valor real.

**8) O que significa "overfitting" em Redes Neuronais? Apresente exemplos e relacione o conceito com a paragem do treino tendo em conta o erro de teste e treino.**

Este problema ocorre quando:

1. é utilizado um número excessivo de neurônios na camada oculta, que levarão a uma perda da capacidade de generalização (fora-da-amostra)

exemplo:



2. A rede é treinada com casos demasiado parecidos e concretos perdendo também a capacidade de abstração.

Pode-se parar de treinar uma rede quando o erro obtido for suficientemente baixo para o problema em questão. De notar que dificilmente se obtém um erro de 0.0 pelo que é necessário aceitar uma margem de erro.