- N O T A S D E I I A -

Ambiente

Episódico – quando a experiência do agente é dividida em episódios independentes. ≠ *Não Episódico*
Discreto – quando as ações e perceções são perfeitamente distintas e existe um número finito de estados ≠ *Contínuo*
Estático – se o ambiente não se altera durante o tempo de tomada de decisão do agente. ≠ *Dinâmico*
Semi-Dinâmico – se o ambiente não se altera com o tempo mas o desempenho do agente sim.
Determinista – se conseguirmos determinar completamente o próximo estado a partir do estado anterior e da ação a realizar. ≠ *Estocástico / Não determinista*
Acessível – se o agente tem acesso ao estado completo do ambiente.

Agente

Reativo – responde a cada perceção sempre da mesma forma, tem em conta apenas a perceção mais recente. Funciona como um reflexo (if…then…) .

- possui um interpretador de regras e aplicabilidade reduzida;
- cria uma representação abstrata da perceção atual;
- identifica a regra cujo antecedente +e mais semelhante ao atual;

Reativo com Estado Interno (c/ memoria) – responde a cada perceção de forma diferente, combinando a perceção mais recente com a informação acerco do estado anterior do ambiente. A sua atualização requer conhecimento sobre:

- como se modifica o mundo ao longo do tempo;
- efeito que a ação tem no estado do mundo;

Guiado por Objetivos – respondem a uma perceção de forma a atingirem um dado objetivo e combinam essa perceção com informação acerca do estado anterior, considerando o resultado futuro. Se o objetivo for alterado, o agente altera o seu comportamento.

Baseado em Funções de Utilidade - respondem a um perceção de forma a atingirem um dado objetivo maximizando o grau de sucesso obtido na prossecução desse objetivo:

- função utilidade:
  - associa valores numéricos a estados , representando o grau de satisfação;
  - opta pela melhor solução ponderando fatores contraditórios e permitindo medir o grau de sucesso obtido quando um objetivo é conseguido;

Racional – escolhe a ação correta isto é, aquela ação que leva o agente a atingir o maior sucesso. Não existe uma única função de avaliação. O nível de racionalidade depende de 4 fatores:

- conhecimento de ambiente inicial;
- sequencia de perceção (tudo aquilo que já aprendeu);
- ações que pode realizar;
- função usada para a avaliação do sucesso;

- P E S Q U I S A S -

Corresponde á procura de um caminho que conduz á solução, ou á procura do melhor caminho de entre todos os possíveis. Uma pesquisa pode ser avaliada segundo 3 critérios:

- alguma solução foi encontrada?
- o custo do caminho/solução é baixo?
- quando o custo da pesquisa em termos de tempo e memoria necessários?

Uma estratégia +e avaliada em 4 critérios:

- Completude: a estratégia garante que se encontra uma solução quando existe de facto uma ?
- Otimização: encontra a melhor solução quando há várias possíveis?
- Complexidade Temporal: quanto tempo leva a encontrar uma solução?
- Complexidade Espacial: quais os requerimentos de memoria?

Tipos de Pesquisas

Cega / Não Informada – procura uma solução sem recorrer a qualquer informação adicional que o guie, apenas conseguindo comparar o estado atual com o objetivo. As variantes deste tipo de pesquisa variam de acordo com a ordem definida para a expansão de estados.

Heurística / Informada - procura uma solução recorrendo a informação adicional que permite escolher o nó a expandir primeiro.

Pesquisa em Largura:

- é expandida por níveis;
- completa, porque procura todas as soluções possíveis;
- ótima, porque propões sempre a solução com menor número de nós;
- de custo elevado de tempo e espaço devido á sua complexidade exponencial;

Pesquisa em Profundidade:

- cada nó é expandido até ser atingido o último nível da arvore, a menos que a solução seja encontrada entretanto;
- necessita de pouca memoria;
- incompleta, porque pode ser infinita :
- não ótima pois retorna uma solução qualquer que pode não ser a ótima;

Pesquisa Uniforme:

- expande o nó com menor **g** ;
- completa;
- ótima se g(sucessor(n)) = g(n) , ou seja, desde que o custo aumente com a profundidade;

Pesquisa Sôfrega

- expande o nó com menor **h** ;
- custo elevado de tempo;
- não ótima, porque não garante que se encontre o caminho de menor custo;
- incompleta pois pode seguir caminhos infinitos;

Pesquisa A*

- expande o no com menor **f**;
- não ótima **se** má heurística;
- ótima **se** boa heurística;
- completa;
- é ótima e completa se nunca assumir um valor superior ao do custo real – isto constitui uma **Heurística Admissível;**

- M E L H O R A M E N T O I T E R A T I V O -

Os algoritmos de melhoramento iterativo caracterizam-se por:
- não anotam os resultados intermédios que conduzem a uma solução, apresentando apenas a "configuração" valida que a compõe;
- partem de uma configuração inicial completa (que viola as restrições), eventualmente gerada aleatoriamente e melhoram-na sucessivamente até alcançarem uma soluções;

Trepa Colinas

Implementação:
- Parte de um estado inicial dado ou gerado aleatoriamente (todas as variáveis com valores atribuídos);
- Gera os estados sucessores ao estado atual;
- Através da função avaliação, avalia cada estado assim gerado e escolhe o de maior valor;
- Pára quando o estado selecionado tiver um valor inferior ao escolhido na iteração anterior;

Problemas:
- Um máximo local pode ser atingido sem que corresponda ao máximo absoluto (melhor solução);
- Nos planaltos é necessário escolher uma direção aleatoriamente;
- Um cume pode ter lados tao inclinados que o passo seguinte conduz "ao outro lado do cume" nunca atingido o máximo pretendido;

Resolução:
- Reiniciar a pesquisa partindo de um estado inicial diferente , gerado aleatoriamente;
- Guarda o melhor resultado obtido nas pesquisas anteriores;
- Pára quando atinge o número de reinícios máximo ou quando o melhor resultado guardado não for ultrapassado durante **n** interações;

Recristalização Simulada

***** Quando encontra um máximo , como pode ser apenas um máximo local, o algoritmo prossegue durante algum tempo a pesquisa no sentido descendente. Assim, em vez de se escolher o estado seguinte de maior valor, escolhe-se um aleatoriamente:

- se a sua avaliação for superior á do estado anterior, é sempre escolhido;
- se for inferior, é escolhido apenas com uma certa probabilidade (<1) que baixa á medida que o parâmetro **t** tende para zero ao longo das sucessivas iterações;

***** É um algoritmo probabilístico , ou seja, o resultado é não determinista e deve-se executar o algoritmo mais do que uma vez. Se o arrefecimento for "suficientemente" lento é sempre atingido o ótimo global.

Pesquisa Tabu

Durante a pesquisa , força a exploração de novas zonas do espaço de procura, evitando assim ciclos, recorrendo a uma memoria de curso duração. Escolhe sempre o melhor vizinho, desde que seja válido, exibindo assim um comportamento determinista e ao aceitar soluções de pior qualidade pode evitar ótimos locais. No entanto, nem sempre é fácil ajustar o limite de memória e o número máximo de iterações.

Seleção

As "melhores hipóteses" são as de maior "aptidão", que é avaliada por uma função fitness. A função fitness define o critério que avalia cada hipótese de acordo com o objetivo a atingir, de geração em geração.

*** Seleção proporcional –** usa a probabilidade de seleção de uma hipótese que é proporcional ao quociente entre a sua aptidão e a soma das aptidões das restantes, sendo selecionadas as hipóteses com maior valor um maior número de vezes.

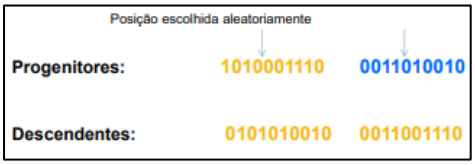
*** Seleção por treino –** escolhe **k** hipóteses (tamanho do torneiro) de entre a população e entre esse grupo seleciona a de maior fitness, e escolhendo com a ajuda de uma probabilidade pré-definida.

***Seleção por posicionamento –** ordena as hipóteses de acordo com a sua aptidão , da melhor para a pior , usando depois o valor da ranking (posição depois da ordenação) por uma função que determinará a probabilidade de seleção de hipótese.

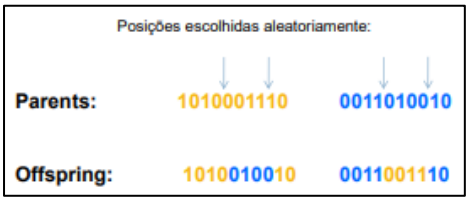
Recombinação

As hipóteses são, muitas vezes, representadas por strings, o que permite um implementação simples das operações de recombinação e mutação.

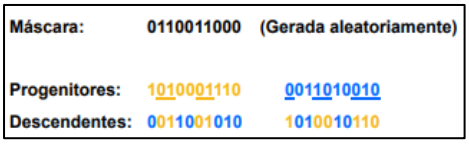
Single-Point Crossover



Double-Point Crossover



Uniform Crossover



Mutação

Gera uma posição aleatória e altera-a, ou seja:

- ALGORITMOS PARA JOGOS –

Os jogos diferenciam-se pela inclusão de um fator de incerteza devido á presença de um adversário, criando assim uma incerteza do tipo não probabilístico, ou seja:
» O adversário B tenterá a melhor jogada para ele, o que implica a pior jogada par o oponente A;
» A aplicação de algoritmos de pesquisa para encontrar a melhor solução de A não funciona porque é necessário contar com os movimento de B ;

Minimax

Aplica-se a jogos determinísticos e observáveis e baseia-se no princípio de "seleção da melhor jogada por parte de cada jogador!. A sua função de utilidade define-se na medição do "proveito" que o estado terminal alcançado representa para cada um do jogadores. Trata-se de um algoritmo recursivo, isto é a distribuição de valore é feita dos nós terminais para a raiz.
Resposta exemplo: Max deverá jogar para C considerando que assim consegue a jogada de maior valor possível e mais vantajosa para si.

Alpha-Beta Pruning

Este algoritmo baseia-se na utilização de parâmetro.
Alfa – representa o valor mínimo garantido que Max poderá obter e , visto que representa um limite inferior e inicializado a **-∞** , vai crescendo sendo atualizado no nó do Max.
Beta – representa o valor máximo que Min consegue impor a Max, assim Max nunca conseguirá jogar para obter um valor superior a Beta e , visto que se trata de um limite superior , é inicializado a **+∞** e posteriormente vai decrescendo, atualizando no nó Mim.
Processo de Pruning – assim ao atingir-se um nó em que alfa >= beta, pode cortar-se o ramo.

Informação para resolução:
α = chão
β = teto
sobe para Min » altera β para o mínimo possível
sobre para Max » altera α para o máximo possível