Algoritmos Genéticos: uma Introdução[1]

Diogo C. Lucas
dlucas@inf.ufrgs.br

Março, 2002

[1]Apostila elaborada sob a orientação de Luís Otávio Álvares, para a disciplina de Ferramentas de Inteligência Artificial

Conteúdo

Capítulo 1

Introdução

1.1 Seleção Natural

Poucas idéias causaram uma repercussão semelhante ao conceito de seleção natural, proposto por Darwin em 1858[1]. Opondo-se à corrente de pensamento da época (o criacionismo) em favor da idéia de *evolução* das espécies, o darwinismo se firmou, após várias adaptações, como uma das mais importantes teorias científicas da modernidade.

O advento da genética e de inúmeras técnicas de microbiologia trouxeram consigo novas teorias científicas que acabaram sendo agregadas, em conjunto com ????,

portuges

Segundo o neo-darwinismo, os preceitos básicos do processo de evolução das espécies seriam:

- indivíduos de mesma ou diferentes espécies disputam continuamente por limitados recursos presentes no meio ambiente;
- dentre os vários concorrentes presentes em um determinado meio, alguns, por conta de suas características específicas, possuem uma melhor chance (maior probabilidade) de sobrevivência. Tais indivíduos são ditos mais adaptados ao ambiente;
- indivíduos mais adaptados possuem uma maior probabilidade de sobrevivência, e conseqüente reprodução;
- visto que no processo de reprodução um grande número de características do(s) pai(s) são repassadas ao(s) filho(s), indivíduos que se reproduzem mais tendem a propagar mais significativamente suas características nas gerações subseqüentes;

[1] O ano de 1858 marcou a apresentação da teoria de evolução por seleção natural à sociedade *Linæus* de Londres, por Charles Darwin e Alfred Wallace (co-inventor do conceito). O primeiro grande trabalho teórico a respeito deste assunto foi publicado por Darwin no ano seguinte ([DAR 2002]), a partir de notas coletadas ao longo de dez anos, que lhe permitiram reinvidicar o título/mérito de descobridor.

- logo, ao longo do processo de evolução, características mais desejáveis tendem a se propagar na espécie, aumentando assim o grau de adaptação desta como um todo;
- o processo de reprodução não ocorre sem falha — durante a replicação e transmissão dos genes aos novos indivíduos criados o fenômeno conhecido como mutação pode ocorrer. Este fonômeno é geralmente prejudicial ao indivíduo, mas em alguns casos pode incorporar a ele uma característica desejável não contida no conjunto de genes dos seus pais. Desta forma a natureza adquire a capacidade de explorar um número maior de combinações e possibilidades.

1.2 Otimização

Freqüentemente nos deparamos com situações em que devemos decidir determinadas características de um sistema de forma a dele extrair o maior número possível de benefícios. Exemplos práticos deste tipo de preocupação no mundo real são a tomada de decisão de preço de produtos de forma a maximizar o lucro, ou a escolha de um trajeto de ônibus que maximize o número de passageiros servidos e minimize as distâncias percorridas.

Visto que ferramentas matemáticas nos permitem expressar muitos destes problemas na forma de funções, foram desenvolvidos métodos para descobrir que valores devem ser escolhidos para se atingir os pontos máximos ou mínimos destas. A tal processo dá-se o nome de otimização, e o classificamos em dois tipos:

otimização numérica: opera sobre o valor de retorno de funções matemáticas. Exemplo: a descoberta do valor de x que maximize a função:

$$f(x_1, x_2) = 21.5 + x_1 \sin(4\pi x_1) + x_2 \sin(20\pi x_2);$$

otimização combinatória: tem como objetivo descobrir qual a melhor combinação dos recursos disponíveis e suas características para otimizar seu uso. Exemplos: qual a melhor forma de alocar processos concorrentes em uma CPU, qual a melhor seqüência de cidades para o problema do caixeiro viajante.

A própria evolução das espécies pode ser vista como um processo de otimização: ao longo do tempo, os seres vivos se tornam cada vez mais adaptados a um meio ambiente em constante mudança.

Capítulo 2

Algoritmos Genéticos

Os algoritmos genéticos utilizam conceitos provenientes do princípio de *seleção natural* para abordar uma série ampla de problemas, em especial de otimização. Robustos, genéricos e facilmente adaptáveis, consistem de uma técnica amplamente estudada e utilizada em diversas áreas.

2.1 Funcionamento

Inspirado na maneira como o darwinismo explica o processo de evolução das espécies, Holland [HOL 75] decompôs o funcionamento dos AGs nas etapas de inicialização, avaliação, seleção, cruzamento, mutação, atualização e finalização (ver figura 2.1).

Basicamente, o que um algoritmo genético faz é criar uma população de possíveis respostas para o problema a ser tratado (inicialização) para depois submetê-la ao processo de evolução, constituído pelas seguintes etapas:

avaliação: avalia-se a aptidão das soluções (indivíduos da população) — é feita uma análise para que se estabeleça quão bem elas respondem ao problema proposto;

seleção: indivíduos são selecionados para a reprodução. A probabilidade de uma dada solução i ser selecionada é proporcional à sua aptidão;

cruzamento: características das soluções escolhidas são recombinadas, gerando novos indivíduos;

mutação: características dos indivíduos resultantes do processo de reprodução são alteradas, acrescentando assim variedade à população;

atualização: os indivíduos criados nesta geração são inseridos na população;

finalização: verifica se as condições de encerramento da evolução foram atingidas, retornando para a etapa de avaliação em caso negativo e encerrando a execução em caso positivo.

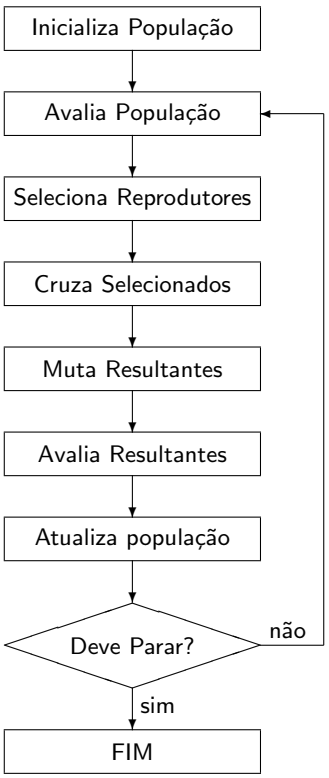

Figura 2.1: Estrutura de funcionamento de um AG tradicional.

2.2 Características

Por causa da maneira particular como os AGs operam, neles se destacam as seguintes características:

busca codificada: Segundo [PÉRE 2000], "os AGs não trabalham sobre o domínio do problema, mas sim sobre representações de seus elementos". Tal fator impõe ao seu uso uma restrição: para resolver um problema é necessário que o conjunto de soluções viáveis para este possa ser de alguma forma codificado em uma população de indivíduos;

generalidade: Os algoritmos genéticos simulam a natureza em um de seus mais fortes atributos: a adaptabilidade. Visto que a representação e a avaliação das possíveis soluções são as únicas partes (de um considerável conjunto de operações utilizadas em seu funcionamento) que obrigatoriamente requisitam conhecimento dependente do domínio do problema abordado [WHI 2000], basta a alteração destas para portá-los para outros casos. A preocupação de um programador de AGs não é então *de que forma* chegar a uma solução, mas sim *com o que ela deveria se parecer*;

paralelismo explícito: o alto grau de paralelismo intrínseco aos AGs pode ser facilmente verificado se considerarmos o fato de que cada indivíduo da população existe como um ente isolado e é avaliado de forma independente.

Se na natureza todo processo de seleção ocorre de forma concorrente, nos AGs essa característica se repete. Os modelos de ilha, descritos na seção 5.1.3 e em [COS 99, WAL 2000], foram criados para explorar tal característica;

busca estocástica: ao contrário de outros métodos de busca de valores ótimos, os algoritmos genéticos não apresentam um comportamento determinístico [GEY 97, MIC 99]. Não seria correto, no entanto, afirmar que tal busca se dá de forma completamente aleatória — as probabilidades de aplicação dos operadores genéticos fazem com que estes operem de forma previsível estatisticamente, apesar de não permitirem que se determine com exatidão absoluta o comportamento do sistema;

busca cega: de acordo com [GEY 97, PÉRE 2000], um algoritmo genético tradicional opera ignorando o significado das estruturas que manipula e qual a melhor maneira de trabalhar sobre estas. Tal característica lhe confere o atributo de não se valer de conhecimento específico ao domínio do problema, o que lhe traz generalidade por um lado, mas uma tendência a uma menor eficiência por outro;

eficiência mediana: por constituir um método de busca cega, um algoritmo genético tradicional tende a apresentar um desempenho menos adequado que alguns tipos de busca heurística orientadas ao problema. Para resolver tal desvantagem, a tática mais utilizada é a hibridização [DAV 91, GEY 97, BUR 95], onde heurísticas provenientes de outras técnicas são incorporadas;

paralelismo implícito: a partir do teorema dos esquemas de Holland (ver seção 6.1), tem-se que ao fazer uma busca por populações, a evolução de um algoritmo genético tende a favorecer indivíduos que compartilhem determinadas características, sendo assim capaz de avaliar implicitamente determinadas combinações ou esquemas como mais ou menos desejáveis, efetuando o que chamamos uma busca por hiperplanos, de natureza paralela [GOL 89];

facilidade no uso de restrições: ao contrário de muitos outros métodos de busca, os AGs facilitam a codificação de problemas com diversos tipos de restrição, mesmo que elas apresentem graus diferentes de importância [BAR 96]. Neste caso, se dois indivíduos violam restrições, é considerado mais apto aquele que viola as mais flexíveis (*soft constraints*) em detrimento do que viola as mais graves (*hard constraints*);

Capítulo 3

Componentes

3.1 Indivíduos

Os indivíduos são a unidade fundamental de um algoritmo genético: eles codificam possíveis soluções para o problema a ser tratado, e é através de sua manipulação (pelo processo de evolução) que respostas são encontradas.

3.1.1 Representação

A escolha de representação para os indivíduos é a etapa mais importante para de desenvolvimento de um AG, visto que ela será a principal responsável para o desempenho do programa.

É de uso comum na área de AGs utilizar os termos *genoma* e mesmo *cromossoma* como um sinônimo para indivíduo. Tal definição nos sugere que um indivíduo se resume ao conjunto de genes que possui (seu *genótipo*), e apresenta um problema: o de que apesar de toda representação por parte do algoritmo ser baseada única e exclusivamente em seu genótipo, toda avaliação é baseada em seu fenótipo (conjunto de características observáveis no objeto resultante do processo de decodificação dos genes).

Tabela 3.1: Exemplos de genótipos e fenótipos correspondentes em alguns tipos de problema.

Genótipo	Fenótipo	Problema
0010101001110101	10869	otimização numérica
CGDEHABF	comece pela cidade C, depois passe pelas cidades G, D, E, H, A, B e termine em F	caixeiro viajante
$C_1R_4C_2R_6C_4R_1$	se condição 1 (C_1) execute regra 4 (R_4), se (C_2) execute (R_6), se (C_4) execute (R_1)	regras de aprendizado para agentes

3.1.2 Características

As características mais importantes

genótipo: consiste na informação presente na estrutura de dados que engloba os genes de um indivíduo;

fenótipo: é o resultado do processo de decodificação do genoma de um indivíduo;

grau de adaptação: representa o quão bem a resposta representada por indivíduo soluciona o problema proposto. É calculado por uma função chamada objetivo (normalmente denotada $f_O(x)$).

grau de aptidão: diz respeito ao nível de adaptação de um indivíduo em relação à população à qual ele pertence. Isto é, se temos que o grau de adaptação de um indivíduo x é dado por $f_O(x)$, então seu grau de aptidão será:

$$f_A(x) = \frac{f_O(x)}{\sum_{i=1}^{n} f_O(i)}$$

sendo n o tamanho da população.

3.2 Populações

A evolução como conhecemos só é possível graças à dinâmica populacional: ao propagar características desejáveis a gerações subseqüentes (cruzamento) enquanto novas são testadas marginalmente (mutação), um AG manipula a freqüência com que determinadas seqüências de genes (ver *esquemas*, na seção 6.1) aparecem nas populações sobre as quais atua.

3.2.1 Características

geração: diz respeito ao número de vezes pelas quais a população passou pelo processo de seleção, reprodução, mutação e atualização;

média de adaptação:

$$M_A = \frac{\sum_{i=1}^{n} f_O(i)}{n}$$

grau de convergência: representa o quão próximo a média de adaptação da atual geração está de suas anteriores. É objetivo dos AGs fazer a população convergir em um valor ótimo de adaptação.

Um fenômeno negativo vinculado a esta medida é a convergência prematura, que se dá quando a população converge em uma média de adaptação sub-ótima, e dela não consegue sair por causa de sua baixa diversidade.

diversidade: mede o grau de variação entre os genótipos presentes na população. A diversidade é fundamental para a amplitude da busca, e sua queda está fortemente vinculada ao fenômeno de *convergência prematura*;

elite: é composta pelos indivíduos mais adaptados da população. Uma técnica comumente usada em AGs é a de *elitismo*, onde os m melhores indivíduos (normalmente $m = 1$) são sempre mantidos a cada geração.

Capítulo 4

Operadores Genéticos

4.1 Inicialização

A inicialização básica de um algoritmo genético clássico se resume à síntese de uma população inicial, sobre a qual serão aplicadas as ações dos passos subseqüentes do processo.

Tipicamente se faz uso de funções aleatórias para gerar os indivíduos, sendo este um recurso simples que visa a fornecer maior *"biodiversidade"*[1], fundamental para garantir uma boa abrangência do espaço de pesquisa. Existem várias alternativas ao método randômico, destinadas a suprir deficiências no que diz respeito a dificuldades existentes quanto à criação aleatória de indivíduos de representação mais complexa e, fator mais considerado, a uma melhora na performance. Podemos citar como exemplo o uso de algoritmos de busca heurística como geradores de populações iniciais, especialmente em casos que apresentem um alto grau de restrições, onde o AG recebe uma população que ainda não possui indivíduos ótimos, mas que apresentam pelo menos algumas das características desejadas. Os operadores de inicialização mais tradicionais são, segundo [GOL 89] e [GEY 97]:

inicialização randômica uniforme: cada gene do indivíduo receberá como valor um elemento do conjunto de alelos[2], sorteado de forma aleatoriamente uniforme;

inicialização randômica não uniforme: determinados valores a serem armazenados no gene tendem a ser escolhidos com uma freqüência maior do que o restante. Um exemplo de tal operador seria o de lançamento de moeda com uma distribuição de probabilidade diferente de 50%.

inicialização randômica com "dope": indivíduos otimizados são inseridos em meio à população aleatoriamente gerada. Esta alternativa apresenta o risco de fazer com que um ou mais super indivíduos tendam a dominar no processo de evolução e causar o problema de convergência prematura.

[1] O termo biodiversidade aqui empregado pertence à tradição existente no meio da computação evolutiva de utilizar, com certa liberdade, termos da biologia.

[2] Também pertencente a biologia, o termo conjunto de alelos diz respeito ao conjunto de valores que um gene pode assumir.

inicialização parcialmente enumerativa: são inseridos na população indivíduos de forma a fazer com que esta comece o processo de evolução possuindo todos os esquemas possíveis de uma determinada ordem.

4.2 Avaliação

Nesta etapa, o primeiro passo da seleção em si é dado: cada indivíduo da população é avaliado para que seja determinado o seu grau de adaptação. Na verdade, este é, em conjunto com a escolha da representação, o ponto do algoritmo mais dependente do problema em si — é necessário aqui que o AG seja capaz de responder sobre quão boa uma resposta é para o problema proposto.

Atualmente, várias formas de avaliação são utilizadas: em casos de otimização de funções matemáticas, o próprio valor de retorno destas tende a ser escolhido, e em problemas com muitas restrições, funções baseadas em penalidades são mais comuns. A função de avaliação também é chamada de função objetivo em um grande número de trabalhos.

Exemplos:

- no caso da busca do valor máximo da função

  $$f(x) = x^3 - x^2 - 10x;$$

  o valor de retorno da função objetivo seria

  $$f_O(G) = f(d(G)),$$

  onde $f_O(G)$ é a função objetivo aplicada sobre o genoma G, $d(g)$ a função que decodifica os genes do genoma em um valor x.

  Se $G = [0001010100101001]$ e $d(x)$ é uma função que decodifica números binários em reais, então: $d(G) = 17,2337$, $f(d(G)) = 361,131$. Logo, $f_o(G) = 361,131$;

- para o problema do caixeiro viajante:

  $$f_O(G) = \frac{1}{\sum_{i=1}^{l_G-1} distancia(g_i, g_{i+1})},$$

  onde g_i representa o i-ésimo gene do genoma G e l_G o comprimento total deste último.

- no caso de um problema de escalonamento, onde a preocupação principal é não permitir que haja violação de restrições, é possível atribuir um valor de penalidade para cada restrição rompida. Neste caso, o valor de adaptação do indivíduo corresponde ao inverso da soma de todas as penalidades por ele recebidas:

  $$f_O(x) = \frac{1}{\sum_{j=1} penalidade_{ij}}$$

  onde i representa o índice do indivíduo avaliado e j a penalidade associada à regra por ele infringida.

Em muitos casos, calcular com exatidão completa o grau de adaptação dos indivíduos pode ser uma tarefa complexa, e se levarmos em conta que esta operação é massivamente repetida ao longo do processo de evolução, seu custo pode ser consideravelmente alto. Em tais situações é comum o uso de funções não determinísticas, que não avaliam a totalidade das características do indivíduo, operando apenas sobre uma amostragem destas.

4.3 Seleção

A seleção é a responsável pela perpetuação de boas características na espécie. Segundo [DAW 96], ela é um processo:

dirigido: ao contrário da mutação, que muitos acreditam ocorrer de forma aleatória, a seleção opera de forma determinística (ou próxima disso). Isto é, enquanto as mutações ocorrem ao acaso, um indivíduo só consegue sobreviver em um ambiente e nele se reproduzir se e somente se for capaz, graças às suas características fenotípicas, de responder de forma adequada a todos fenômenos de seu meio;

cumulativo: os benefícios do processo de seleção são mantidos de uma geração para a outra. Este fator, combinado com a não aleatoriedade da seleção, garante a possibilidade de surgimento de organismos complexos como as formas de vida que hoje conhecemos. A seleção natural consiste então de um longo processo de pequenas tentativas e erros, onde os ganhos são sempre acumulados, de geração para geração.

É no estágio de seleção que os indivíduos são escolhidos para posterior cruzamento. Neste ponto, fazendo uso do grau de adequação de cada um, é efetuado um sorteio onde os mais aptos possuem maior probabilidade de se reproduzirem. Este grau é calculado a partir da função de avaliação de cada indivíduo, e determina quão apto ele está para a reprodução em relação à população a que pertence.

4.3.1 *Scaling*

De acordo com [GOL 89, WAL 2000, GEY 97], o uso de um método de *scaling* sobre o valor de adaptação de cada indivíduo normalmente é útil por reduzir a probabilidade de convergência prematura. Diversas funções podem ser aplicadas sobre a adaptação

***no scaling*:** O valor de retorno da função objetivo é usado sem nenhum tipo de alteração.

***linear scaling*:** O valor de retorno da função objetivo ($f_O(x)$) sofre a seguinte alteração:

$$f_l(f_O(x)) = f_O(x) \times a + b,$$

onde

$$a = (c-1) \times \frac{med}{\Delta} \text{ e } b = med \times \frac{max - c \times med}{\Delta}, \text{ se } \Delta \neq 0,$$

$$\text{ou } a = 1, b = 0 \text{ em caso contrário,}$$

max representa o maior valor de adaptação encontrado, med a média de adaptação da população e c é uma constante pré-definida pertencente aos reais e maior do que 1. Valores negativos de adaptação não são aceitos por este método.

***sigma truncation scaling*:** um múltiplo do desvio médio (denotado por d_m) é subtraído dos valores de adaptação. Todos valores negativos são alterados para zero. A fórmula para seu cálculo é a seguinte:

$$f_{st}(f_O(x)) = f_O(x) - med - c \times d_m,$$

onde c é uma constante pré-definida pertencente ao conjunto dos reais e maior do que zero.

***power law scaling*:** o valor de adaptação do indivíduo é elevado a uma constante pré-definida k:

$$f_{pl}(f_O(x)) = f_O(x)^k$$

Este método não aceita valores negativos de adaptação.

4.3.2 Métodos de seleção

Existem várias formas para efetuar a seleção, dentre as quais destacam-se:

seleção por ranking (*rank selection*): os indivíduos da população são ordenados de acordo com seu valor de adequação e então sua probabilidade de escolha é atribuída conforme a posição que ocupam;

seleção por giro de roleta (*roulette wheel selection*): o método de seleção por giro de roleta funciona da seguinte forma [DAV 91]: calcula-se o somatório da adequação da população ($total$) sorteia-se um valor i tal que pertence ao intervalo [0; $total$] seleciona-se o indivíduo x tal que a ele corresponda à faixa do somatório onde i se localiza;

seleção por torneio (*tournament selection*): Grupos de soluções são escolhidos sucessivamente e as mais adaptadas dentro de cada um destes são selecionadas [GOL 89, WAL 2000, GEY 97];

seleção uniforme: todos indivíduos possuem a mesma probabilidade de serem selecionados [WAL 2000]. Obviamente, esta forma de seleção possui uma probabilidade muito remota de causar uma melhora da população sobre a qual atua;

***remainder stochastic selection*:** a partir da fórmula:

$$s(x) = \frac{f_A(x)}{med}$$

se assume que a parte inteira da função $s(x)$ determina quantas cópias do indivíduo são selecionadas diretamente. O restante de indivíduos a serem selecionados é escolhido aplicando um método de seleção como o giro de roleta sobre a parte decimal do valor $s(x)$ de cada indivíduo [WHI 2000, GOL 89, GEY 97];

4.4 Reprodução

Uma vez selecionados os indivíduos, estes passam com uma probabilidade pré-estabelecida pelo processo de cruzamento (*crossover*), onde partes dos genes dos pais são combinadas para geração de filhos.

4.4.1 Pareamento (*mating*)

Alguns dos principais métodos de escolha dos pares de reprodutores são, segundo [GEY 97]:

escolha aleatória: os pares de indivíduos que devem se reproduzir são escolhidos ao acaso;

***inbreeding*:** parentes são combinados;

***line breeding*:** um indivíduo de alta performance é cruzado com uma subpopulação de indivíduos e os filhos são selecionados como pais;

***outbreeding*:** indivíduos que codificam fenótipos diferentes são combinados;

***self-fertilization*:** o indivíduo é combinado consigo mesmo;

***positive assortive mating*:** indivíduos semelhantes são combinados;

***negative assortive mating*:** indivíduos diferentes são combinados.

4.4.2 Operadores de cruzamento

Na verdade, é possível representar o funcionamento dos operadores de cruzamento como uma seleção por máscara: esta seria representada por um vetor cujos elementos possam assumir valores binários e que possua um comprimento igual ao dos cromossomas a serem combinados . Seu uso pelo operador se daria segundo o seguinte algoritmo:

Figura 4.1: Algoritmo de uso de uma máscara de cruzamento

```
se mascara_i = 0 então
    filho_1 ← pai_1
    filho_2 ← pai_2
senão
    filho_1 ← pai_2
    filho_2 ← pai_1
```

Os mais conhecidos operadores tradicionais de cruzamento para genomas de comprimento fixo são [GOL 89]:

cruzamento de um ponto (1PX): dados dois genomas x e y de comprimento l_g, sorteia-se um número p qualquer tal que $0 < p < l_g$, o primeiro filho f_0 receberá todos os genes x de 1 até p e todos os genes y de $p + 1$ até l_y, e o segundo filho o inverso.

A máscara de cruzamento seria uma série de 0s (zeros) sucedidas de 1s (uns) pertencente ao conjunto $\{0n1m \mid n \geq 0,\ m \geq 0,\ n + m = l_g\}$.

Dentre os operadores de cruzamento tradicionais o de um ponto é o que normalmente apresenta o pior desempenho.

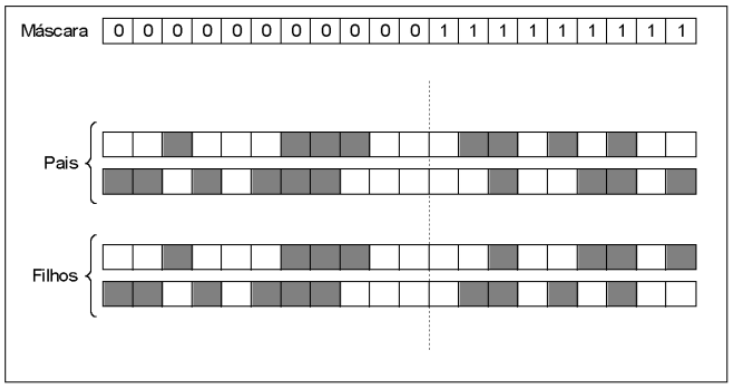

Figura 4.2: O operador de cruzamento de um ponto

cruzamento multiponto (MPX): o cruzamento multi-ponto é uma generalização do operador de um ponto. Nele são sorteados um número fixo n de pontos de corte.

Como regra geral, uma operação de cruzamento de n pontos pode ser vista como um caso especial de uma de $m > n$ pontos em que um ou mais $(m - n)$ pontos de corte selecionados se localizam no final do cromossomo (o ponto de índice $i = l_g$).

Um operador com n pontos de cruzamento apresentaria uma máscara de cruzamento com n mudanças em sua seqüência de zeros e uns;

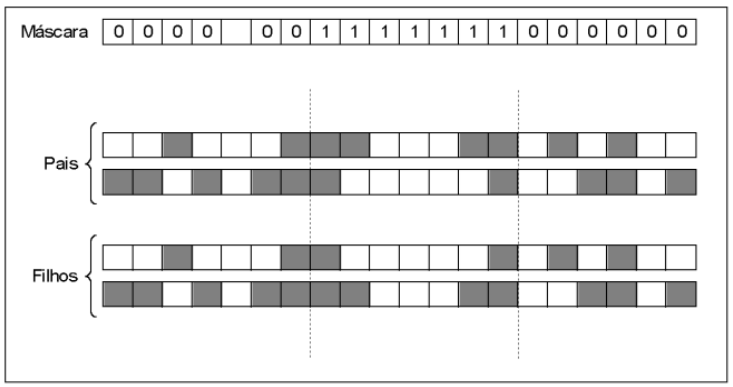

Figura 4.3: O operador de cruzamento multiponto com dois pontos de corte.

cruzamento Segmentado (SX): o cruzamento segmentado funciona de maneira semelhante ao multi-ponto, com a exceção de que sorteia o número de pontos de corte toda vez que é executado;

cruzamento uniforme (UX): para cada gene a ser preenchido nos cromossomos filhos, o operador de cruzamento uniforme sorteia de qual dos pais este deve ser gerado. A máscara de cruzamento de tal operador é uma seqüência qualquer de zeros e uns;

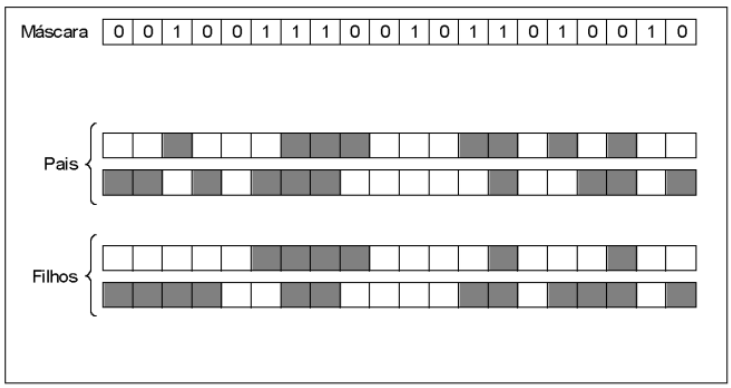

Figura 4.4: O operador de cruzamento uniforme.

cruzamento por combinação parcial (PMX): sorteia dois pontos de corte e faz com que os indivíduos filhos recebam na íntegra os genes do pai ou da mãe (mãe para um e pai para outro) situados entre os pontos de corte, depois disso preenche os genes restantes com os valores considerados mais adequados para cada filho;

4.5 Mutação

A mutação opera sobre os indivíduos resultantes do processo de cruzamento e com uma probabilidade pré-determinada efetua algum tipo de alteração em sua estrutura. A importância deste operador reside no fato de que uma vez bem escolhido seu modo de atuar garante que diversas alternativas serão exploradas, mantendo assim um nível mínimo de abrangência na busca (ver seção 6.2).

Exemplos de operadores de mutação:

mutação *flip*: cada gene a ser mutado recebe um valor sorteado do alfabeto válido;

mutação por troca (*swap mutation*): são sorteados n pares de genes, e os elementos do par trocam de valor entre si;

mutação *creep*: um valor aleatório é somado ou subtraído do valor do gene.

4.6 Atualização

Neste ponto, os indivíduos resultantes do processo de cruzamento e mutação são inseridos na população segundo a política adotada pelo AG. Na forma mais tradicional deste (chamada corriqueiramente de algoritmo genético simples), a população mantém um tamanho fixo e os indivíduos são criados em mesmo número que seus antecessores e os substituem por completo. Existem, porém, alternativas a essa abordagem: o número de indivíduos gerados pode ser menor[3], o tamanho da população pode sofrer variações e o critério de inserção pode ser variado (como, por exemplo, nos casos em que os filhos substituem os pais, ou em que estes só são inseridos se possuírem maior aptidão que o cromossoma a ser substituído), ou o conjunto dos n melhores indivíduos pode sempre ser mantido[4].

4.7 Finalização

A finalização não envolve o uso de nenhum operador genético: ela simplesmente é composta de um teste que dá fim ao processo de evolução caso o AG tenha chegado a algum ponto pré-estabelecido de parada. Os critérios para a parada podem ser vários, desde o número de gerações já criadas até o grau de convergência da atual população (por convergência entende-se o grau de proximidade dos valores da avaliação de cada indivíduo da população).

[3]Para uma abordagem mais profunda do assunto, veja a seção 5.1, sobre populações

[4]Chamamos essa característica de *elitismo*. Esta opção de funcionamento é amplamente utilizada, normalmente com apenas um indivíduo compondo a elite, mesmo em algoritmos genéticos simples. Deve ser aplicada com cuidado para se evitar o problema da *convergência prematura*.

Capítulo 5

Técnicas

Existe uma ampla gama de técnicas de implementação de AGs que servem como alternativa à tradicional. É importante ressaltar aqui que mesmo uma combinação de diversas técnicas geralmente é possível, e em muitos casos, é até mesmo recomendável.

5.1 Populações

Diversas abordagens alternativas à maneira tradicional com que a população evolui a cada geração. Analisaremos a seguir algumas das mais conhecidas.

5.1.1 Estado fixo

Um algoritmo genético de estado fixo (*steady-state*) mantém o tamanho de sua população (*popsize*) constante por todo o processo de evolução. A cada geração são criados indivíduos que serão inseridos na população, e, uma vez feito isso, os menos aptos são retirados de maneira a fazer com que esta retorne a seu tamanho original. A taxa de sobreposição desta população é definida por uma constante, que representa o percentual desta substituído a cada geração. Os novos indivíduos são inseridos antes da exclusão para garantir que os mais fracos da totalidade sejam excluídos.

Algumas abordagens utilizam como constante para orientar na substituição o número n de indivíduos a serem inseridos ao invés do percentual.

5.1.2 Incremental

Segundo [WAL 2000], neste modelo de AG um número pequeno de filhos é criado a cada geração, e o esquema de superposição deste pode ser variável. Algumas das formas mais utilizadas são:

pais: os pais são substituídos;

randômico: os indivíduos a serem substituídos são escolhidos ao acaso;

pior: os piores indivíduos da população são substituídos.

5.1.3 Modelo de Ilhas

O modelo de ilhas é na verdade mais uma forma de comportamento da população de um AG, mas possui um grau tão considerável de importância que deve ser tratado separadamente. Ele tem como objetivo explorar o paralelismo explícito dos algoritmos genéticos visto que o alto grau de independência entre os indivíduos torna facilmente viável induzir sua evolução em paralelo.

Nesta abordagem a população total é dividida em um conjunto de sub-populações ou *ilhas* (normalmente uma sub-população para cada processador disponível) e estas evoluem paralelamente, geralmente fazendo uso de um AG simples ou de estado fixo.A cada geração um número pré-determinado de indivíduos é copiado ou trocado entre as populações, em uma operação conhecida como *migração*. A taxa de migração define este número e deve ter um valor que evite a evolução completamente independente das sub-populações (no caso de a taxa ser igual a zero), onde o problema de convergência prematura tende a ocorrer com maior freqüência, assim como a supressão das diferenças locais entre ilhas. Outro fator a ser considerado no momento de escolha da taxa de migração é o custo que esta operação oferece.

Segundo [COS 99], as principais formas de comunicação utilizadas pelo modelo de ilhas são:

Comunicação em rede: com a disposição em rede todas as ilhas se encontram interconectadas e efetuam um processo de troca de seus melhores indivíduos.

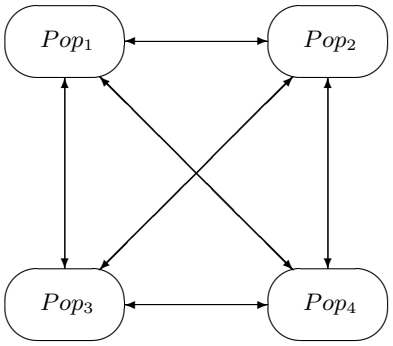

Figura 5.1: Modelo de ilhas em rede [COS 99].

Comunicação em estrela: utilizado em [WAL 2000], na disposição em estrela uma das ilhas (em geral a que apresenta a melhor média de adaptação após a inicialização) se torna a ilha mestre, a única a se comunicar com todas as outras. As restantes são comumente chamadas de escravas e a cada geração efetuam as operações de envio e recepção de indivíduos ótimos apenas com a mestre.

Comunicação em anel: com esta disposição cada ilha envia seus melhores indivíduos para a ilha a seguir e recebe os da anterior, num esquema de comunicação de sentido único.

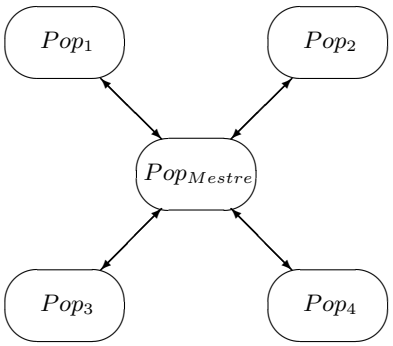

Figura 5.2: Modelo de ilhas em estrela [COS 99].

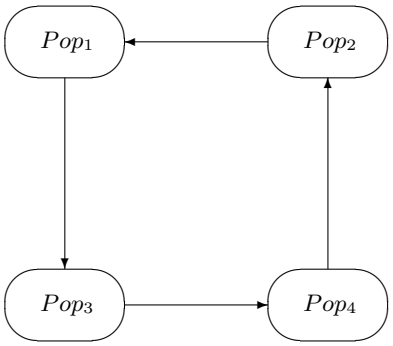

Figura 5.3: Modelo de ilhas em anel [COS 99].

5.2 Representação poliplóide

A representação genética mais utilizada nos AGs é a haplóide, onde o genótipo de cada indivíduo é representado por apenas um cromossoma. Tal escolha se justifica pelo fato de evitar a redundância trazida por representações diplóides e poliplóides, onde o fenótipo relacionado a um certo gene é determinado por uma relação de dominância: uma vez que cada cromossoma a mais oferece um gene extra que codifica a característica desejada e possivelmente um valor (ou alelo) diferente neste, será selecionado aquele considerado dominante em detrimento dos recessivos.

Levando-se em conta que a representação haplóide é usada na natureza apenas para formas de vida mais simples é necessário questionar o porquê do uso de genótipos diplóides ou poliplóides por parte de organismos mais complexos. A justificativa para isto é encontrada no fato de que as relações de dominância entre os alelos não é estática, mas também sujeita à evolução, e que os genes redundantes servem como recipientes de informações que possibilitam uma espécie de "memória" do processo de evolução, reduzindo assim a probabilidade de que determinados alelos que codificam características até o momento pouco desejáveis sejam extintos da população.

5.3 Algoritmos miméticos

Os algoritmos miméticos[1] tem seu funcionamento baseado em uma evolução por pressão social, e não pela tradicional seleção natural[2]. O melhor e o pior indivíduo (o vencedor e o perdedor, respectivamente) localizados durante sua execução são memorizados e servem como modelos para a criação do restante da população. Os operadores de cruzamento e mutação são aqui substituídos por um de mutação social, que torna os indivíduos sobre os quais opera mais próximos ou distantes de seus modelos. Existem diversas estratégias de "comportamento" que definem que atitudes um indivíduo pode assumir com relação a seus modelos (por exemplo: imitar seria equivalente a tentar copiar o maior número possível de características do modelo, rejeitar equivaleria à tentativa de se distanciar delas e ignorar simplesmente corresponderia a não as levar em conta), entre elas estão, segundo [PEY 93]:

empreendedor: imita o vencedor e ignora o perdedor;

rebanho: imita o vencedor e rejeita o perdedor;

fóbico: rejeita o perdedor e ignora o vencedor;

ignorante: ignora ambos modelos, serve como referência para checar a relevância destes;

5.4 Híbridos

Os algoritmos genéticos tradicionais geralmente não são as melhores técnicas de busca na maioria dos domínios aplicáveis, devido à ignorância que possuem de como gerar indivíduos melhores a partir da população sobre a qual atuam. Visto que esse método apresenta entretanto uma série significativa de atributos desejáveis, uma considerável carga de pesquisa foi dedicada à resolução deste problema. A idéia por trás dos algoritmos híbridos se resume a explorar em um AG as principais vantagens pertencentes a seus concorrentes. Esta tarefa se dá importando dos últimos o conhecimento específico sobre o domínio do problema que tratam. De acordo com [DAV 91], as diretivas mais comuns para hibridização consistem em:

- basear-se na forma de codificação da técnica concorrente. Desta forma, se faz uso do conhecimento de domínio presente na técnica, assim como se facilita o uso por parte de especialistas no assunto;
- utilizar o algoritmo concorrente como operador de inicialização — ao se fazer uso do concorrente para gerar a população inicial, garante-se que na pior hipótese o híbrido possuirá um desempenho semelhante a ele. Esta abordagem só é recomendável quando se trata de técnica que forneça respostas em curto espaço de tempo, e é tida como "perfeitamente segura", visto que no pior caso possível o último apresenta um desempenho semelhante ao primeiro;

[1] O termo mimético vem do grego *mimetes*, que quer dizer imitação.

[2] É importante ressaltar que o termo natural não está sendo aqui utilizado de maneira literal, isto é, a pressão social é um fato derivado da natureza e, por conseqüência, pode ser vista como um fenômeno natural.

- converter operações aplicadas com freqüência em operadores genéticos — tais operações podem ser adaptadas para funcionar como um operador genético já conhecido. Por exemplo, considerando-se que a mutação é unária (atua sobre apenas um objeto) e causa variações no cromossoma, uma operação do concorrente que apresente tais características pode ser convertida neste operador. Também é possível gerar um operador com uma nova lógica de funcionamento para substituir ou trabalhar em conjunto com os restantes;

- utilizar o método de decodificação — se o algoritmo concorrente apresenta um método de decodificação eficiente e este é importante para o funcionamento do AG, a melhor alternativa é incorporá-lo ao híbrido;

5.5 Programação genética

Ao invés de manipular possíveis genótipos de soluções para o problema a ser tratado, a lógica de funcionamento dos algoritmos de programação genética é de manipular representações de programas que o resolvam. Para isto, os indivíduos que manipula são na realidade árvores sintáticas como a da Figura 10 e é tomando um cuidado especial com as alterações feitas nestes, visto que facilmente elas podem gerar programas não funcionais.

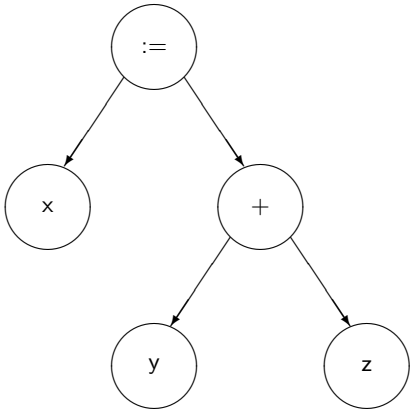

Figura 5.4: Uma árvore sintática usada como indivíduo de um algoritmo de PG.

Um operador de cruzamento por *swap* de sub-árvores escolhe um nodo de cada um dois pais e gera os filhos como resultado da troca destes dois nós (seus eventuais filhos os acompanham no processo), e um operador de mutação flip substitui uma chamada de função por outra de mesmo tipo ou um símbolo terminal por outro do conjunto válido [MIC 99]. Outros exemplos de operadores de mutação e cruzamento para genomas em forma de árvore podem ser encontrados em [WAL 2000].

O cálculo da adaptação de um indivíduo se dá através do teste de sua execução: dada uma entrada pré-definida, a saída do programa representado é avaliada, assim como tamanho deste.

Capítulo 6

Considerações

6.1 Esquemas

Consideremos um algoritmo genético que opere sobre uma população de soluções formadas por uma combinação qualquer de n bits. A linguagem que constrói tais indivíduos poderia usar como alfabeto $S = \{0, 1\}^n$. Utilizando-se o * como um coringa para designar situações *"don't care"*, é possível definir, por exemplo, para $n = 4$ o conjunto C de todos indivíduos gerados a partir S que começam com o número 0 e terminam com o 1 como sendo 0,*,*,1.

Chamamos então esse *string* que define o conjunto, formado a partir de um alfabeto S' (equivalente a um alfabeto básico S básico acrescido do símbolo *), de *esquema*.

6.1.1 Ordem de um esquema

A ordem de um esquema Q, expressa como $O(Q)$, representa o número de posições fixas (elementos pertencentes ao alfabeto original) presentes neste. Por exemplo, sejam os seguintes esquemas:

Q_1	=	{0,1,0,1}
Q_2	=	{*,*,1,*}
Q_3	=	{*,*,*,*}
Q_4	=	{0,*,0,*}

Tabela 6.1: Exemplos de Esquemas

Temos então $O(Q_1) = 4$, $O(Q_2) = 1$, $O(Q_3) = 0$, $O(Q_4) = 2$. Obviamente, esquemas de menor ordem abrangem uma quantidade maior de indivíduos.

6.1.2 Tamanho de um esquema

O tamanho de um esquema Q, denotado por $T(Q)$, é a distância entre o primeiro e o último número fixo deste, para o exemplo anterior temos $T(Q_1) = 4 - 1 = 3$, $T(Q_2) = T(Q_3) = 0$, $T(Q_4) = 3 - 1 = 2$.

O tamanho de um esquema é importante para determinar a probabilidade com que ele se manterá após submetido aos operadores de mutação e cruza-

mento, como veremos adiante. O teorema dos esquemas de Holland diz que bons esquemas tendem a se reproduzir de forma exponencial nas gerações subseqüentes.

6.2 Mutação e Cruzamento

Os operadores genéticos de mutação e cruzamento são os responsáveis por todas as transformações sofridas pela população, mas possuem funções bastante distintas no que diz respeito a seu impacto na evolução.

O operador de cruzamento tem como objetivo propagar os esquemas mais adequados na população (ver seção 6.3). Para isto, os pontos de corte são fundamentais, pois vão determinar quais esquemas sobreviverão ao processo de reprodução. Em problemas altamente combinatórios um operador cego de cruzamento pode facilmente gerar filhos menos adequados a partir dos cromossomas pais, fenômeno causado por uma alta correlação entre genes.

A mutação é fator fundamental para garantir a biodiversidade, assegurando assim que o espaço de busca provavelmente será explorado em uma parte significativa de sua extensão. Apesar de normalmente aplicada com uma probabilidade bastante inferior à de cruzamento, ela é tida por uma série de autores como o operador mais importante para o processo evolutivo, chegando em alguns casos extremos a ser utilizada como o único operador de transformação no curso de evolução do AG.

O operador de mutação possui também um papel fundamental no que diz respeito à necessidade de evitar a convergência prematura, que ocorre quando a população se estabiliza com uma média de adaptação pouco adequada por causa da pressão evolutiva e baixa diversidade. Isto geralmente se dá com o surgimento de um super-indivíduo que domina o processo seletivo e, uma vez incapaz de gerar filhos melhores, transmite suas características por toda população.

6.3 Busca por hiperplanos

Uma das implicações da prova de Holland, baseada no teorema dos esquemas, é de que os algoritmos genéticos efetuam uma busca por hiperplanos.

Se trabalharmos com um problema cuja solução pode ser codificada por uma *string* binária de tamanho 3, nosso espaço de soluções pode ser representado pelo cubo na Figura 6.1. Este cubo representa planos em que os indivíduos podem estar localizados. Por exemplo, nos vértices de sua face frontal localizam-se apenas indivíduos que tenham 0 como valor no gene de *lócus*[1] 1, i.e., que contenham o esquema {0**}, e nos da face superior apenas os com o esquema {*1*}. Esquemas de ordem 2 representam sempre vértices dos segmentos que compoem o cubo: o esquema {1*0} representa os vértices do segmento que é intersecção dos lados esquerdo e traseiro do cubo.

Esta representação torna claro o fato de que esquemas de menor ordem abrangem um número maior de indivíduos. Uma representação de um hipercubo de busca em um espaço de 4 dimensões com um alfabeto binário pode ser vista na Figura 6.2. Nela o gene de *lócus* 1 designa a que cubo pertence

[1] O termo *lócus* designa a posição de um gene dentro do cromossoma.

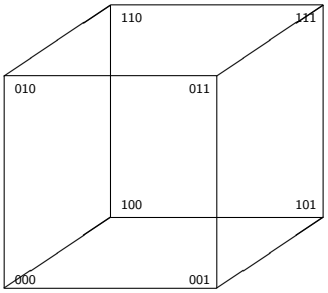

Figura 6.1: Um hipercubo de busca de três dimensões[WHI 2000].

o esquema, e os demais funcionam de maneira análoga à representação tridimensional.

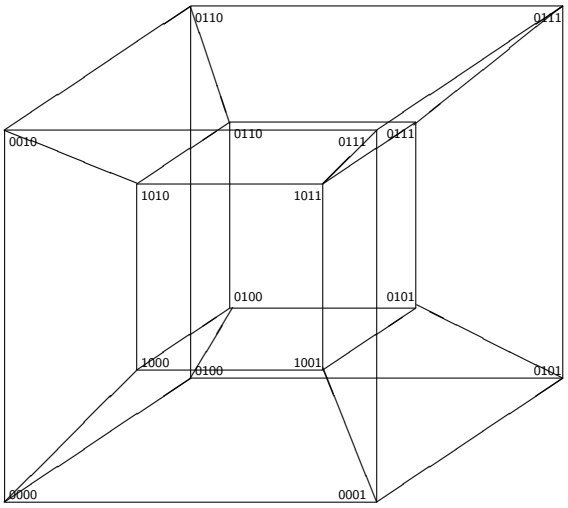

Figura 6.2: Um hipercubo de busca de quatro dimensões[WHI 2000].

Segundo [GOL 89, WHI 2000], cada indivíduo gerado a partir de um alfabeto binário está presente em $2^L - 1$ hiperplanos diferentes (onde L é o comprimento do cromossoma e a *string* formada apenas por símbolos * representa todo espaço de busca e não é considerada uma partição do espaço) e é possível gerar 3^L partições do hiperplano.

A busca por populações realizada por um algoritmo genético na verdade faz uma tomada de amostra dos hiperplanos, e a partir desta efetua uma estimava do grau de adaptação de todos indivíduos pertencentes a eles.

Considerando-se que determinados hiperplanos possuem indivíduos com me-

lhor grau de adaptação do que a média, estes tendem a representar parcelas cada vez maiores da população conforme a execução se dá. Em um algoritmo genético que funcione de maneira adequada, o número de indivíduos pertencentes a cada hiperplano após o processo de seleção deveria estar próximo de um valor que pode ser estimado[WHI00].

6.4 Algoritmos Genéticos × Escalada de Montanha

6.4.1 O problema dos ótimos locais

Algoritmos de escalada de montanha (*hill-climbing*) possuem dificuldades de lidar com funções que apresentem grande oscilação em seus valores, por causa de sua tendência a parar sua busca quando encontram um valor ótimo local. Estes últimos se localizam em picos locais, pontos onde a função atinge seu máximo em um intervalo reduzido, sem, contudo, que este seja o pico de maior altura de todo seu domínio.

Por causa de sua busca por hiperplanos, os AGs conseguem evitar este problema sem grandes dificuldades.

6.4.2 Escalada de montanha em um AG

O esquema de busca por hiperplanos foi demonstrado para o algoritmo genético clássico ou canônico e tem seu funcionamento baseado no operador de cruzamento. Para que este tipo de busca funcione, é necessário que a população apresente uma alta biodiversidade, fator que propicia uma amostragem razoável do espaço de busca. É sabido que algumas heurísticas tendem a reduzir esta característica, assim como afetar a maneira como os esquemas vão se propagar. Em alguns casos, tais métodos (muitas vezes combinados com uma inversão na probabilidade de aplicação dos operadores de transformação, i.e., com baixa probabilidade de cruzamento e alta de mutação) apresentam um comportamento de escalada de montanha, e são defendidos por uma série de autores (para maiores referências, ver [WHI 2000].

Capítulo 7

Aplicações

Por se tratar de uma técnica bastante difundida e geralmente oferecer bons resultados, os AGs são usados em uma ampla série de domínios de problemas. A seguir, ilustramos sua utilização em alguns deles. Mais comentários podem ser encontrados em [HEI 2000].

7.1 Sistemas de classificação

De acordo com [HEI 2000], uma das aplicações propostas por Holland para os AGs seria seu uso em sistemas de classificação (*classifier system*, CFS), capazes de interpretar mudanças em seu meio ambiente e decidir sobre qual a melhor maneira de se adaptar a elas.

Para isto, é necessário que se defina um meio ambiente e a forma como o CFS interage com ele, ou seja, que funcionalidades serão implementadas para que o último seja informado a respeito de alterações e para que possa efetuar alterações no primeiro.

A lógica de funcionamento de um CFS pode ser definida como *"... start from scratch, i.e., from tabula rasa (without any knowledge) using a randomly generated classifier POPULATION, and let the system learn its program by induction, (cf Holland et al. 1986), this reduces the input stream to recurrent input patterns, that must be repeated over and over again, to enable the animat to classify its current situation/context and react on the goings on appropriately."*[HEI 2000].

7.2 Teoria dos Jogos

De acordo com [HEI 2000], os algoritmos genéticos podem ser relacionados à Teoria dos Jogos: para isso, geralmente a evolução de sua população consiste de um processo onde indivíduos são selecionados para jogar um jogo com um número limitado de jogadas. Cada jogador interage com um grupo uma série de vezes para depois jogar com outros oponentes. Dependendo das combinações efetuadas por este em cada jogo será determinado um valor de recompensa, e do somatório destes ao final resultará seu grau de adaptação.

Um indivíduo nesta abordagem codifica a estratégia de um jogador, seja por exemplo representando a probabilidade de escolha de uma jogada A em detri-

mento de uma B (onde a população poderia ser representada por um conjunto de números reais) ou através de Autômatos Finitos que indicariam a sucessão de passos a serem efetuados durante o jogo.

Um exemplo clássico de utilização de AGs para o estudo de tópicos referentes à Teoria dos Jogos foi o trabalho desenvolvido por Axelrod em 1987 (ver [AXE 97] para uma versão atualizada do artigo original), e envolvia uma variação de um problema bastante conhecido da época: o Dilema de Prisioneiro Iterado (DPI).

Em sua forma tradicional, o Dilema do Prisioneiro apresenta a seguinte situação: dois comparsas são presos e interrogados separadamente. Já que as provas coletadas pela polícia são circunstanciais, as penas às quais os criminosos incorrerão são também determinadas também pelo depoimento destes — caso os dois resolvam colaborar entre si, as poucas evidências os indiciaram a um delito leve (utilidade 3), resultando em um ano de prisão, caso os dois resolvam se acusar, ambos passarão três anos na prisão (utilidade 1), caso um colabore e outro denuncie, ao primeiro caberá cumprir 5 anos (utilidade 0) e ao segundo sair livre imediatamente (utilidade 5).

Tabela 7.1: Tabela de utilidades do Dilema do Prisioneiro.

Utilidades: A,B		Indivíduo B	
		coopera	delata
Indivíduo A	coopera	3, 3	0, 5
	delata	5, 0	1, 1

Visto que os jogadores não podem ter certeza a respeito da escolha um do outro, a opção por uma estratégia se dá assumindo *que o outro vê as coisas da mesma maneira*. Já que a jogada de menos risco é a de denunciar (pois na pior da hipóteses se obtem o mesmo resultado medíocre que seu adversário, e na melhor delas o valor máximo de utilidade), e que a única expectativa que um indivíduo tem a respeito de seu oponente é a de que este possua a mesma racionalidade, o equilíbrio do jogo reside na combinação de jogadas ($[a_2, b_2]$), evidentemente não tão boa quanto ($[a_1, b_1]$), mas livre do risco de obtenção da baixa utilidade 1 para qualquer um dos envolvidos (soluções ($[a_1, b_2]$) e ($[a_2, b_1]$)).

Na versão iterada do problema, o jogo é constantemente repetido, tornando ainda mais gritante a possibilidade de obtenção de um desempenho superior à media de um ponto de utilidade por jogada — se os jogadores são capazes de estabelecer alguma forma de cooperação, suas médias podem subir para 3 pontos por rodada. Em um torneio organizado por Axelrod, diversas estratégias baseadas no histórico de jogadas entre os envolvidos foram propostas e confrontadas umas com as outras, e dentre elas uma se destacou como particularmente eficiente: a *tit for tat*.

Baseada na reciprocidade, a *tit for tat* consistia em colaborar na primeira iteração (um gesto de boa fé) e a partir disto optar pela mesma jogada preferida pelo oponente na rodada anterior. Desta forma, jogadores que a adotavam puniam seus denunciantes e incentivavam colaboradores.

Visando descobrir quão universal é o uso da reciprocidade, Axelrod fez uso de um AG para evoluir populações de estratégias contra as oito mais representativas propostas no experimento anterior[1].

[1]Na verdade, ele foi capaz de demonstrar que o desempenho de qualquer estratégia podia

7.2.1 Implementação

Desde que existem 4 resultados possíveis para cada rodada ($[a_1, b_1]$, $[a_1, b_2]$, $[a_2, b_1]$, $[a_2, b_2]$), uma estratégia baseada nas n iterações anteriores deve levar em consideração 4^n combinações. Além disso, visto que ainda não existe um histórico de tamanho n até a rodada $n + 1$, é incorporada ao genoma uma "memória" de n primeiras rodadas hipotéticas (o que lhe acrescenta $2n$ combinações). Assumindo $n = 3$, temos $4^3 = 64$ estratégias possíveis, mais $2 \times 3 = 6$ combinações para o histórico inicial, resultando num total de 70 genes.

Podemos então usar o próprio histórico de jogadas efetuadas entre os envolvidos como índice para definir que ação deve ser seguida: atribuindo ao ato de delatar o valor 0 e ao de cooperar 1, temos que qualquer combinação de jogadas pode ser codificada como uma *string* de bits. Alguns exemplos de strings e seus respectivos históricos podem ser vistos na tabela 7.2:

string	histórico	índice
000000	(DD),(DD),(DD)	0
010011	(DC),(DD),(CC)	19
101001	(CD),(CD),(DC)	41
111010	(CC),(CD),(CD)	58
111111	(CC),(CC),(CC)	63

Tabela 7.2: Strings de bits codificando iterações passadas no DPI.

Segundo este esquema de codificação, teríamos então, para $n = 2$, um genoma semelhante ao da figura 7.1. O indivíduo por ele representado assume que as duas interações que teve com qualquer outro jogador foi cooperativa (denotado pelos pares de Cs correspondentes às duas primeiras rodadas hipotéticas), e tem como política fazer exatamente o que seu oponente fez na rodada anterior, desconsiderando qualquer outra informação (é possível observar esta linha de conduta visto que ele sempre coopera se encontra um C na casa correspondente à última jogada de seu adversário, e delata em caso contrário), implementando assim uma estratégia *tit for tat*[2].

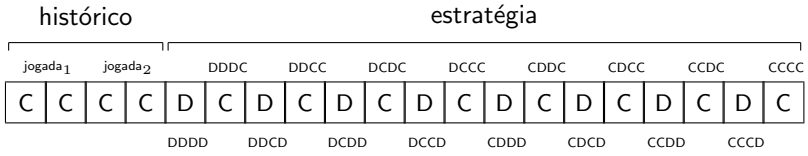

Figura 7.1: Genoma para o Dilema do Prisioneiro Iterado com $n = 2$

ser avaliado levando em consideração os resultados de seu confrontamento com este pequeno conjunto.

[2]Na verdade, a *tit for tat* possui mais de um genoma possível. Enquanto a parte correspondente à estratégia deve ser idêntica para todos, o histório só precisa ter em comum um C no gene correspondente à última jogada do adversário, enquanto os restantes são irrelevantes.

A funcionamento do AG proposto for Axelrod pode ser visto na figura 7.2: os indivíduos gerados são confrontados com as 8 estratégias representantes, e sua probabilidade de reprodução é proporcional ao valor de utilidade acumulado apos x interações.

```
inicializa população
para cada geração
    para cada g_i ∈ população
        para cada uma das estratégias representantes
            para cada uma das x interações
                decide se g_i coopera ou delata em função do histórico
                decide a ação da representante em função de seu histórico
                atualiza o score de g_i
            fim para
        fim para
    fim para
    reproduz indivíduos segundo seu score
fim para
```

Figura 7.2: O Algoritmo Genético proposto por [AXE 97] para o estudo do DPI

7.2.2 Resultados

Após um número razoável de gerações, Axelrod pôde constatar o domínio de indivíduos que baseavam suas jogadas em reciprocidade, de maneira semelhante à *tit for tat*, mas obtendo um maior valor de utilidade: o algoritmo havia encontrado uma maneira de explorar estratégias menos robustas enquanto mantinha o jogo cooperativo com as que não conseguia manipular. A explicação para este desempenho residia na especialização permitida pelo uso de estratégias fixas como adversárias.

O próximo passo foi, então, lançar mão de estratégias dinâmicas para a avaliação de indivíduos. Este objetivo foi alcançado fazendo com que jogadores pertencentes ao AG enfrentassem uns aos outros. Ao rodar o protótipo, foi possível observar uma proliferação inicial de indivíduos que denunciavam cegamente — nas primeiras gerações, onde uma população gerada de maneira aleatória e bem distribuída devia conter um número razoável de indivíduos que cooperavam cegamente, a opção por denunciar sempre se tornava a sem dúvida mais adequada — que acabaram se tornando a grande maioria da população. No entanto surgiram, através de mutações, jogadores que lançavam a mão da reciprocidade: enquanto esta prática não era particularmente adequada para um número muito pequeno de praticantes, seu crescimento na população os tornava, ao contrario dos denunciantes, cada vez mais aptos, permitindo assim a eventual dominação da população por parte destes.

7.3 Escalonamento e grade horária

Segundo [SOA 97], existem diversas definições conflitantes para o problema de escalonamento (*scheduling*) mas uma bastante adequada é a de Fox: *"scheduling selects among alternative plans, and assigns resources and times to each activity so that they obey the temporal restrictions of activities and the capacity limitations of a set of shared resources"*. De acordo com [DAV 91], problemas de scheduling apresentam grande dificuldade em seu tratamento pelos seguintes motivos:

complexidade computacional: os problemas de escalonamento são da classe NP Completo, o que quer dizer que a partir de um tamanho razoável dos dados de entrada torna-se impossível explorar todo espaço de busca. Além disso, métodos que façam uso de heurísticas para limitar este espaço não podem garantir que sempre encontrarão uma resposta ótima;

conhecimento de domínio: problemas de *scheduling* devem levar em consideração restrições que com freqüência estão fortemente vinculadas a conhecimento específico ao domínio de que fazem parte.

O problema de elaboração de grades horárias para cursos é bastante conhecido no meio acadêmico, se não por estudos a ele dedicados, então por ser vivido a cada semestre. Uma típica resolução manual para tal projeto envolve a composição de alguns horários [COR 93, COR 94] que respeitem as regras mais fundamentais, e sua remodelagem conforme neles são descobertos problemas. Este processo envolve muitas vezes a submissão de uma alternativa considerada mais adequada aos indivíduos envolvidos (professores e alunos) para que através do *feedback* proporcionado por estes se encontre uma solução que minimize conflitos. Tal processo efetuado manualmente pode levar vários dias para uma conclusão.

Vários métodos de otimização foram tentados, mas a grande maioria encontra problemas ao lidar com o grande número de restrições e a alta variedade de seu grau de importância, assim como ótimos locais.

O problema de alocação de grade horária em um curso consiste em, para cada hora-aula h pertencente ao conjunto H de horas-aula de todas as disciplinas do curso, ministrada por um docente p a uma turma t e em uma sala s, escolher o melhor *time slot*[3] s de forma a minimizar problemas como:

colisões (*clashes*): por motivos óbvios, o mais importante critério para a avaliação de um horário proposto é que este evite que para um determinado time slot s mais de uma disciplina envolvendo o mesmo professor, turma ou sala ocorra;

semi colisões (*near-clashes*): por preocupações didáticas, é importante evitar a alocação adjacente de blocos de horas-aula do mesmo gênero para a mesma turma, ou seja, evitar que para uma turma t ocorra, após ministrada uma aula de uma disciplina de um gênero g_d no time slot s, outra com mesmo tipo no slot $s + 1$;

[3]Um *time slot* representa um determinado horário da semana em que uma hora-aula pode ser alocada, por exemplo, terça-feira às 10h 00min.

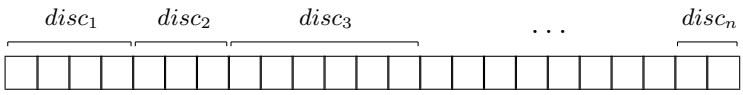

Figura 7.3: A estrutura escohida para o genoma.

tempo de deslocamento: é preciso considerar o tempo gasto no deslocamento entre duas aulas sucessivas que ocorram em salas distantes, tanto para a turma quanto para o professor envolvido. Um horário que também minimize o número de idas e vindas entre locações distantes para aulas é considerado mais adequado;

slots inadequados: determinados slots são menos adequados para aulas que outros. Por exemplo, a alternativa de se ministrar aulas no período de 12h às 14h é viável, mas pouco indicada; ou um determinado turno de um dia da semana de um ou mais professores pode estar reservado a atividades que não lhe permitam dar aulas;

alocação ótima: a vizinhança em que as disciplinas são alocadas deve ser considerada . Para melhor rendimento das aulas são sugeridos "blocos" de duas horas para cada disciplina.

O sistema mais tradicionalmente utilizado para a avaliação dos indivíduos é o de penalidades, no qual utiliza-se a fórmula:

$$f_O(x) = \frac{1}{1 + \sum penalidades(x)}$$

onde $f_O(x)$ representa o valor de retorno da função objetivo aplicada ao indivíduo x, e $penalidades(x)$ as penalidades a ele associadas.

7.3.1 Implementação

A forma de representação escolhida foi a baseada nas horas-aula das disciplinas: neste caso, o genoma representa o conjunto de todas as horas-aula de todas as disciplinas. Por motivos de simplicidade, as horas-aula da mesma disciplina são alocadas em blocos adjacentes, e por sua vez os blocos são alocados ordenadamente segundo o semestre da disciplina a que pertencem. Apesar de permitir todos os tipos de colisão em sua codificação, esta representação apresenta a vantagem considerável de sempre ter como solução horários que respeitem a carga horária de todas as disciplinas.

A função objetivo faz uso do cálculo de penalidades associadas a um cromossoma. Na realidade seu funcionamento se baseia em verificar para cada gene quais penalidades estão associadas à sua combinação com todos os demais.

Na Tabela 7.3 são sugeridas restrições (marcadas com *) que por motivos de viabilidade não foram implementadas. Elas se encontram aqui como uma forma

critério	função	grau
semestre	Impedir colisão entre disciplinas do mesmo semestre	altíssimo
professor	Impedir que disciplinas ministradas pelo mesmo docente sejam alocadas no mesmo horário	altíssimo
deslocamento*	Considerar o respeito ao tempo de demora no deslocamento de um prédio a outro (ex: prédios localizados em dois *campi* distantes)	altíssimo
sala*	Respeitar o limite de uma turma por sala durante as aulas	altíssimo
slots *tabu*	Evitar alocar aulas em horários pouco adequados (ex: das 12h 00min às 14h 00min)	médio
bloco	Dispor as horas-aula das disciplinas em blocos de tamanho que facilitem no estudo. O tamanho tipicamente escolhido para um bom bloco é de duas horas-aula	médio
dia	Evitar a alocação de mais de um bloco de aulas de uma disciplina no mesmo dia	médio
tipo	Inibir que duas disciplinas do mesmo tipo sejam alocadas em sucessão para a mesma turma	baixo
slots ótimos*	Tentar alocar na primeira hora da tarde ou na segunda da manhã	baixo
horários compactos*	Tentar evitar que turnos não tenham a totalidade de seus time-slots alocados	baixo

Tabela 7.3: Tipos de restrição e seu grau de importância

```
soma_das_penalidades← 0
para cada g_i ∈ genoma
    //corta iterações desnecessárias
    para cada (g_j posterior a g_i) ∈ genoma
        //impede que a mesma infração sofra múltiplas penalidades
        se g_j ainda não causou nenhuma penalidade
            então
                soma_das_penalidades← penalidade(g_i, g_j)
        fim se
    fim para
fim para
// soma-se 1 para evitar uma divisão por zero
retorne 1/(1 + soma_das_penalidades)
```

Figura 7.4: Função objetivo de um problema de grade horária

de ilustração de outros fatores que poderiam ser considerados no momento de confecção de uma grade horária e não foram tratadas no aplicativo por não serem consideradas absolutamente fundamentais (como por exemplo a restrição de alocar na primeira hora da tarde ou na segunda da manhã), ou, como no caso da importante colisão de sala, apresentarem um considerável complicador no que diz respeito à dificuldade de se obter os dados de entrada necessários e por ilustrar um conceito (a alta necessidade de evitar colisões) abordado em outros critérios.

7.3.2 Resultados

Os operadores genéticos tradicionais já tiveram sua base explicada na seção 4. Limitamo-nos aqui, então, a ilustrar o comportamento apenas dos que foram criados e a uma avaliação dos resultados apresentados:

cruzamento por um ponto (1PX): apresentou, contrariamente ao normal nas aplicações, o melhor desempenho dentre os operadores tradicionais utilizados. A justificativa para este fato é que por apresentar apenas um ponto de corte, este operador tende a conservar esquemas importantes com maior facilidade;

cruzamento multipontos(MPX): Com um desempenho pouco inferior ao cruzamento por um ponto, os testes realizados confirmam que, quanto menor o número de pontos de corte empregados na operação, melhores os resultados;

cruzamento uniforme(UX): apresentou péssimos resultados, até 1000% inferiores ao cruzamento por um ponto;

uniform random initializer (inicializador aleatório simples): Esta subrotina atribui um valor aleatório para cada gene do indivíduo.

Tem como vantagem a facilidade de implementação, o alto grau de biodiversidade da população que gera e o baixo custo de seu desempenho (neste

caso pouco significativo para os tamanhos de população e indivíduos testados neste trabalho). Sua principal desvantagem é a de gerar com uma probabilidade ínfima indivíduos que satisfaçam as restrições por time slot e alocação ótima;

heuristic slot initializer (inicializador heurístico por *slot*): utiliza uma função aleatória com distribuição não uniforme, que sorteia time slots indesejáveis com uma probabilidade menor.

Seu custo operacional é praticamente igual ao do inicializador randômico simples e possui sobre este a vantagem de raramente alocar horas-aula em time slots tabu[4]. Testes comparativos revelaram uma melhora de 100% no desempenho em relação a seu antecessor. Não se preocupa com uma alocação racional de blocos;

heuristic discipline initializer (inicializador heurístico por disciplina): faz uso da mesma função de sorteio de *time slots* do inicializador de slots, mas aloca preferencialmente as disciplinas em blocos que respeitem as restrições de alocação ótima.

Seu desempenho é comparável dos operadores anteriores, porém proporciona uma melhora considerável na performance do AG como um todo.

multi-point block crossover (MBX): O cruzamento por blocos e em múltiplos pontos é um operador de cruzamento multiponto que efetua o corte dos cromossomas pais de forma a não partir um bloco de uma mesma disciplina.

swap Block Mutation (SWM): Semelhante ao SM, faz a troca de blocos entre os pares de disciplinas que sorteia. Para executar sua tarefa, precisa extrair do genoma uma representação em blocos das disciplinas escolhidas para o swapping e uma vez feito isso escolher quais os melhores blocos para a troca. Este operador apresentou uma melhora de 300% em relação aos operadores de mutação tradicionais, e ajuda a manter a biodiversidade, sem contudo apresentar o mesmo efeito negativo que seus antecessores sobre a convergência.

Em um problema de *timetabling* com a representação escolhida, bons esquemas tendem a possuir um grande comprimento, pois se de um lado é comum que os valores ao redor de um gene sejam importantes para a qualidade final do cromossoma, visto que é isto que determina de que forma as disciplinas foram alocadas em blocos, por outro é muito comum que genes tenham grande importância um para o outro apesar de fisicamente[5] distantes, como no caso de genes pertencentes a disciplinas de semestres diferentes mas de mesmo professor.

Visto que o problema é altamente combinatório, ou seja, que o valor de um determinado gene tende a possuir uma considerável importância para os demais, podemos também afirmar que o grau de um bom esquema geralmente é bastante elevado.

[4] O termo tabu aqui utilizado é emprestado de outra técnica de otimização conhecida: a busca tabu. Um indivíduo ou valor tabu é possível mas inadequado, indesejável.

[5] O termo *fisicamente* aqui se refere à disposição dos genes no genoma.

Considerando-se estas duas características se torna claro que operadores de recombinação mal direcionados podem gerar com alta probabilidade filhos menos adequados que os pais. Dados o alto número de restrições a que está sujeito um horário e a maneira cega como operam o cruzamento e a mutação tradicionais, o AG baseado no uso destes apresentou um desempenho muito pouco satisfatório, principalmente no que diz respeito ao requisito de grupamento ótimo em blocos. Apesar de este requisito poder ser otimizado por esquemas de pequeno comprimento, graças à alocação contígua das horas-aula da mesma disciplina, ele dificilmente é respeitado com o uso dos operadores tradicionais, visto que a probabilidade de que o AG aloque as disciplinas em blocos ótimos é bastante pequena.

O problema de convergência prematura tende a ocorrer com facilidade em AGs para timetabling. Isto se dá pelo fato de que no momento em que se restringe a maneira como os operadores genéticos alteram os indivíduos, também se restringe a biodiversidade da população à qual eles pertencem. Se aliado a isto tivermos o aparecimento de superindivíduos (fenômeno bastante típico) e uma política inadequada de seleção e scaling é possível observar que durante a evolução um grupo reduzido de cromossomas tende a dominar e com isso diminuir o escopo da busca.

Em termos de características da população, o que se observa é um aumento gradual nas médias conforme esta evolui acompanhado de uma queda na diversidade e globalmente no ritmo de adaptação dos melhores indivíduos.

Conforme já descrito, a solução tradicional para este problema é um uso racional do operador de mutação: é necessário que este consiga manter um nível adequado de biodiversidade sem incorrer com grande freqüência no problema de rompimento de bons esquemas. Os operadores tradicionais de mutação deram uma contribuição bastante limitada a estes dois critérios, pois um uso mais freqüente destes (necessário para garantia da diversidade) tende a baixar o grau de adaptação da população.

A importância do elemento aleatório na evolução de um algoritmo genético se faz notar por um fato negativo: os resultados finais de execuções diferentes com os mesmos dados de entrada podem apresentar um desvio extremamente significativo, dependendo do problema tratado. Isto significa que, para determinadas classes de problemas (da qual o de grade horária faz parte), execuções ótimas e medíocres tendem a aparecer intercaladas, e implica em uma dificuldade considerável de previsão de comportamento do algoritmo, fundamental para sua depuração e *benchmarking*.

No protótipo apresentado neste trabalho, variações de 1000% chegaram a ser registradas, fenômeno que requer uma investigação mais profunda do grau de importância da base aleatória para a evolução do AG: através de uma análise dos piores resultados obtidos, é possível observar que seus melhores indivíduos não são ótimos por violarem um pequeno número de restrições de prioridade média ou baixa ligadas à maneira como as horas-aula são alocadas em blocos.

Duas abordagens para a resolução deste problema seriam, primeiramente, limitar a maneira como são gerados os indivíduos iniciais, para reduzir a probabilidade de que apresentem tais características indesejáveis, através da criação de uma nova função heurística de sorteio de *time-slots* para cada bloco de horas-aula, ou, em segundo lugar, fazer uso de uma heurística de escolha que não o simples sorteio uniforme para os pontos do genoma sobre os quais deve atuar o *Swap Block Mutator*, visto que este tende a eliminar o problema quando aplicado

a genes que violem estas restrições.

Lista de Figuras

Lista de Tabelas

Glossary

A

Adaptação no campo da biologia, o nível de adaptação mede o quanto um indivíduo esta apto para sobreviver no meio em que reside. Em sua analogia com a natureza, os AGs fazem uso do grau da adaptação para representar quão bem um determinado indivíduo (solução) responde ao problema proposto.

Alelo os alelos de um gene correspondem ao conjunto de valores que ele pode assumir. Normalmente, o conjunto de alelos é o mesmo para todos genes de um genoma.

Alfabeto um alfabeto consiste em um conjunto de símbolos que são manipulados por uma linguagem. Para a quase totalidade dos problemas tratados pelos algoritmos genéticos é possível definir uma linguagem formal que gere a partir de um alfabeto e uma série de regras pré-definidas todo o conjunto de soluções possíveis para o problema. Um alfabeto, para os AGs é, então, o conjunto união de todos os alelos possíveis para os genes pertencentes ao genoma.

C

Computação Evolutiva campo da Ciência da Computação dedicado a estudar o uso da evolução para a resolução de problemas. A técnica mais conhecida da CE são os Algoritmos Genéticos, mas existem na realidade uma ampla gama de outras abordagens.

Convergência o grau de convergência é característica das populações dos AGs, e diz respeito à diferença entre a média de adaptação da geração atual e suas anteriores. A ascensão deste índice indica que o processo de evolução está efetivamente promovendo a melhora da média de adaptação da população, e sua estabilização em torno de um mesmo valor por muitas gerações normalmente indica que a população se estacionou em um determinado valor médio de adaptação, caso em que a continuação do processo de evolução se torna improdutiva.

Convergência prematura a convergência prematura é um problema bastante recorrente na técnica de algoritmos genéticos: ela se dá quando a diversidade de uma população cai de tal forma que o processo de reprodução gera a cada geração filhos mais semelhantes aos pais,

o que causa a convergência da população em torno de média de adaptação em pouco adequada e retarda (ou até mesmo estanca por completo) a evolução.

Criacionismo corrente de pensamento oriunda da interpretação literal da Bíblia, o criacionismo defende que tanto a terra quanto todas as espécies de seres vivos nela existentes foram criadas no momento da Gênese e desde então permanecem inalteradas (com exceção da possível extinção de espécies que não foram salvas por Noé durante o Grande Dilúvio).

Cromossoma um cromossoma consiste de uma cadeia de genes. Visto que tradicionalmente são utilizados genomas com uma cadeia simples de genes, muitas vezes este termo é utilizado, com certa liberdade, como sinônimo para genoma.

Cruzamento durante o processo de cruzamento os indivíduos previamente selecionados são cruzados com seus pares para gerar filhos.

D

Darwininismo teoria científica para a explicação do fenômeno de evolução das espécies baseada no conceito de seleção natural, proposto por Darwin e Wallace em 1858. Com descobertas posteriores da biologia (notavelmente da genética e biologia populacional) o darwinismo foi reformulado e sofreu a agregação de novas teorias, formando o neo-darwinismo.

Diversidade (ou biodiversidade) característica de uma população de indivíduos, a diversidade diz respeito ao grau de semelhança entre eles.

E

Esquema (ou esquemático) os esquemas servem para definir e representar conjuntos de genomas. Para isto, faz uso do símbolo * para ilustrar situações em que o valor de determinado gene não importa. Por exemplo o esquema {*,*,*,1} representa o conjunto de todos cromossomas que possuem 1 como valor em seu gene de índice 4.

Evolução o processo de evolução é o responsável pela busca efetuada pelos AGs. A cada passo os indivíduos da população a ele submetida passam pelos processos de seleção, reprodução e mutação, criando assim uma nova geração a partir de sua anterior.

F

Fenótipo resultado do processo de decodificação do genótipo de um indivíduo, em um AG o fenótipo é a solução propriamente dita que este representa.

Função objetivo a função objetivo recebe como entrada um indivíduo e retorna o grau de adaptação que este apresenta.

G

Gene um gene serve como um *container*, um espaço para a alocação de um valor. As características dos indivíduos são definidas a partir dos valores contidos no conjunto de um ou mais genes que as codificam.

Genoma um genoma é o conjunto de genes de um determinado indivíduo. Tradicionalmente genomas são formados por uma cadeia única de genes, mas existem representações alternativas como a de árvores (bastante utilizada em problemas de programação genética) e de matrizes multidimensionais.

Genótipo para um algoritmo genético o genótipo de um indivíduo é a informação codificada em um genoma (a estrutura que é manipulada durante o processo de evolução) e o fenótipo resultante do processo de decodificação é a solução que ele representa.

Geração a cada passo do processo de evolução uma nova geração é criada a partir da população anterior, e esta última é atualizada. Para que o processo de evolução possa ser considerado eficiente é necessário que em cada nova geração tenda a ser melhor (mais adaptada) que suas anteriores.

I

Indivíduo um indivíduo pertencente a um algoritmo genético representa uma possível solução para o problema a ser tratado.

L

Lamarckismo teoria anterior ao darwinismo, que propunha que características desenvolvidas ao longo da vida de um indivíduo podiam ser repassadas para a sua prole. Existem técnicas pertencentes à Computação Evolutiva que lançam mão de conceitos lamarckistas.

Lócus (ou índice) o lócus de um gene determina a posição que ele ocupa no genoma. É necessário então uma escolha de determinação do lócus que não permita ambigüidades.

M

Mutação na natureza, falhas no processo de reprodução podem facilmente ocorrer, fazendo com que os filhos apresentem características que não seriam atingíveis através de uma combinação de características dos pais. A este fenômeno dá-se o nome de mutação, e nos algoritmos genéticos esta transformação ocorre sobre indivíduos provindos do processo de reprodução.

P

Pareamento (ou mating) o pareamento tem por objetivo determinar como os indivíduos selecionados para reprodução serão cruzados entre si.

População uma população consiste em um conjunto de indivíduos. Ela apresenta características não observáveis nos indivíduos, tais como grau de diversidade e de convergência.

R

Reprodução o processo de reprodução consiste da alocação em pares (ou em algum método alternativo de mating) dos indivíduos selecionados e do cruzamento entre estes. Ele é o responsável pela perpetuação de características ao longo do processo de evolução.

S

Scaling é sabido que a presença de super indivíduos e a baixa diversidade numa população são fatores que facilmente levam ao problema de convergência prematura. Vários métodos são utilizados para diminuir a probabilidade de que isto ocorra, entre eles os de scaling. Estes últimos operam aplicando uma transformação ao grau de adaptação de cada indivíduo da população, atenuando as discrepâncias nestes valores, e valorizando indivíduos diferentes da média.

Seleção nesta etapa onde indivíduos são escolhidos para a reprodução de acordo com seu grau de adaptação. Para isto cada indivíduo é avaliado e os mais aptos da população possuem maior probabilidade de serem selecionados.

Super-indivíduo uma determinada solução é considerada um super-indivíduo quando ela apresenta um grau de adaptação significativamente superior a média. Por possuir então uma probabilidade muito maior de ser selecionado para reprodução que seus concorrentes, suas características tendem a proliferar pelas gerações subseqüentes, freqüentemente causando queda na diversidade.

Bibliografia

[AXE 97] AXELROD, R. Evolving new strategies. In: **The complexity of cooperation**: agent-based models of competition and collaboration. Princeton, New Jersey: Princeton University Press, 1997. (Princeton Studies in Complexity).

[BAR 96] BARBOSA, H. J. C. Algoritmos genéticos para otimização em engenharia: uma introdução. In: IV SEMINáRIOS SOBRE ELEMENTOS FINITOS E MéTODOS NUMéRICOS EM ENGENHARIA, 1996, Juiz de Fora, MG. **Anais...** [S.l.: s.n.], 1996.

[BUR 95] BURKE, E.; ELLIMAN, D.; WEARE, R. A hybrid genetic algorithm for highly constrained timetabling problems. In: 6TH INTERNATIONAL CONFERENCE ON GENETIC ALGORITHMS (ICGA'95, PITTSBURGH, USA, 15TH-19TH JULY 1995), 1995. **Anais...** Morgan Kaufmann: San Francisco: CA: USA, 1995. p.605–610. Disponível via WWW em http://www.asap.cs.nott.ac.uk/ASAP/papers/pdf/icga95.pdf. (Agosto de 2000).

[COR 93] CORNE, D.; FANG, H.-L.; MELLISH, C. Solving the modular exam scheduling problem with genetic algorithms. In: SIXTH INTERNATIONAL CONFERENCE ON INDUSTRIAL AND ENGINEERING APPLICATIONS OF ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS, 1993. **Anais...** Gordon and Breach Science Publishers, 1993. Disponível via FTP anônimo em ftp://ftp.dai.ed.ac.uk/pub/user/ga/93-001.ps.Z. (Agosto de 2000).

[COR 94] CORNE, D.; FANG, H.-L.; ROSS, P. Fast practical evolutionary timetabling. In: AISB WORKSHOP 1994, 1994. **Anais...** Springer Verlag Lecture Notes in Computer Science, 1994. Disponível via FTP anônimo em ftp://ftp.dai.ed.ac.uk/pub/user/ga/94-004.ps.Z. (Agosto de 2000).

[COS 99] COSTA JR, I. Introdução aos algoritmos genéticos. In: VII ESCOLA DE INFORMáTICA DA SBC REGIONAL SUL, 1999. **Anais...** [S.l.: s.n.], 1999.

[DAR 2002] DARWIN, C. **On the origin of species**. Disponível via WWW em http://www.hn.psu.edu/faculty/jmanis/darwin.htm (Janeiro 2002).

[DAV 91] DAVIS, L. D. **Handbook of genetic algorithms**. [S.l.]: Van Nostrand Reinhold, 1991.

[DAW 96] DAWKINS, R. **A escalada do Monte Improvável**: uma defesa da teoria da evolução. São Paulo – SP: Companhia das Letras, 1996.

[GEY 97] GEYER-SCHULTZ, A. **Fuzzy rule-based expert systems and genetic machine learning**. Heidelberg: Physica-Verlag, 1997.

[GOL 89] GOLDBERG, D. E. **Genetic algorithms in search, optimization & machine learning**. [S.l.]: Addison-Wesley, 1989.

[HEI 2000] HEITKOETTER, J.; BEASLEY, D. **The hitch-hiker's guide to evolutionary computation**. Disponível no newsgroup: comp.ai.genetic. (Agosto de 2000).

[HOL 75] HOLLAND, J. **Adaptation in natural and artificial systems**. Ann Arbor: Univ. of Michigan Press, 1975.

[MIC 99] MICHALEWICZ, Z. **Genetic algorithms + data structures = evolution programs**. Berlin: Springer-Verlag, 1999.

[PEY 93] PEYRAL, M. et al. Mimetic evolution. In: INTERNATIONAL CONFERENCE ON GENETIC ALGORITHMS, 1993, San Mateo. **Anais...** Morgan Kaufmann, 1993.

[PÉRE 2000] PÉREZ SERRADA, A. **Una introducción a la computación evolutiva**. Disponível via WWW em http://www.geocities.com/igoryepes/spanish.zip. (Setembro de 2000).

[SOA 97] SOARES PEDRO; MAMEDE, N. J. Timetabling using demand profiles. In: 8TH PORTUGUESE CONFERENCE IN ARTIFICIAL INTELLIGENCE, EPIA-97, COIMBRA, PORTUGAL, 1997, Berlin Heidelberg. **Anais...** Springer-Verlag, 1997.

[WAL 2000] WALL, M. **Galib**: a C++ library of genetic algorithm components. Massachussets, EUA: [s.n.], 2000. Manual disponível via WWW em http://lancet.mit.edu/ga/ (Agosto de 2000).

[WHI 2000] WHITLEY, D. **A genetic algorithm tutorial**. Disponível via WWW em http://www.geocities.com/igoryepes/ga_tutorial.zip. (Setembro de 2000).

Índice