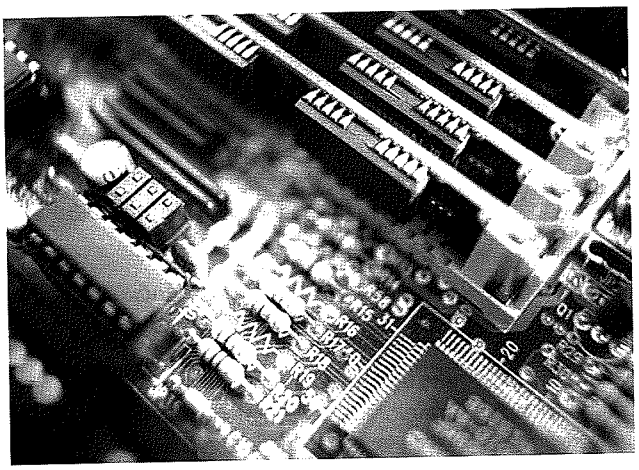
ALGORITMOS GENÉTICOS

Marisol Correia | Prof.ª Adjunta na ESGHT



Este artigo apresenta os algoritmos genéticos de uma forma simplificada. O objectivo é mostrar o que são, para que servem e como funcionam e surge como um primeiro artigo de apresentação ao tema.

O artigo está organizado da seguinte forma: Na secção 2 é apresentado um algoritmo genético simples, que é exemplificado na secção 3. A secção 4 aprofunda os conceitos de: codificação dos indivíduos, população inicial, tamanho da população, a função objectivo, a função *fitness* e, por último, os operadores selecção, cruzamento e mutação. O Teorema dos Esquemas e a Hipótese dos Blocos são desenvolvidos na secção 5. A secção 6 apresenta algumas das áreas de aplicação dos algoritmos genéticos. Por último, são apresentadas as conclusões.

1. Introdução

Os Algoritmos Genéticos (AG) são técnicas de procura e optimização baseados em mecanismos de selecção natural. Foram desenvolvidos por John Holland, seus colegas e alunos da Universidade de Michigan nas décadas de 60 e 70, com o objectivo de estudar formalmente o processo de adaptação que ocorre nos sistemas naturais (natureza) e desenvolver modelos que utilizem os mecanismos de adaptação natural, implementando-os em sistemas computacionais (Holland 1975). Estes algoritmos permitem abordar problemas complexos sobre os quais exista pouca informação, excepto a forma de avaliar uma boa solução. Podem ser aplicados a um conjunto diverso de problemas (Goldberg 1989), permitindo obter melhores resultados que outras técnicas de optimização.

As diferenças dos algoritmos genéticos em relação a outras técnicas de procura e optimização residem em:

- → Não trabalharem directamente com o domínio do problema mas com representações dos seus elementos;
- → Executarem a procura num conjunto de candidatos (população) e não apenas num;
- → Não terem conhecimento específico do problema, utilizando apenas a função objectivo;
- → Utilizarem basicamente regras probabilísticas.

2. Os Algoritmos Genéticos Simples

Nos algoritmos genéticos as variáveis do problema, ou conjunto de parâmetros, são representados como genes num cromossoma (cromossoma esse que se designa por indivíduo). O valor que o gene pode tomar é denominado alelo, enquanto que a posição no cromossoma (de um determinado gene) é denominado locus.

Os algoritmos genéticos utilizam um conjunto de soluções candidatas denominado população. Recorrendo

aos operadores genéticos (selecção, cruzamento, mutação, entre outros) é possível encontrar as melhores soluções, ou seja, os cromossomas mais aptos são escolhidos.

O desempenho ou aptidão (*fitness*, em inglês) de cada indivíduo (cromossoma) é avaliado com base na função objectivo. A selecção natural permite que os indivíduos mais aptos sejam os progenitores da geração seguinte, ou seja, gerem descendentes na população seguinte.

Os indivíduos podem ser considerados a dois níveis: ao nível do fenótipo e ao nível do genótipo. O fenótipo de um indivíduo corresponde ao seu valor no domínio onde a função objectivo é definida. O genótipo de um indivíduo corresponde à representação do seu fenótipo. O fenótipo é codificado no genótipo, tradicionalmente utilizando o código binário, mas podendo ser utilizadas outras estruturas de dados, como os valores reais, matrizes, entre outros.

O operador cruzamento permite que os genes de dois cromossomas progenitores, previamente seleccionados, sejam combinados para formar dois novos cromossomas na população seguinte, os quais, em princípio, serão mais aptos que os seus progenitores, melhorando a população.

O operador mutação permite introduzir diversidade genética na população. Alterando arbitrariamente um ou mais componentes de uma estrutura escolhida entre a descendência, este operador permite introduzir novos elementos à população de forma a assegurar que a probabilidade de se chegar a qualquer ponto do espaço de procura não seja nula. Este operador aplica-se geralmente a seguir ao cruzamento.

Apresenta-se a seguir um algoritmo genético simples:

Gerar a população inicial;
REPETIR
Aplicar a função objectivo a todos os indivíduos da população actual;
Seleccionar os indivíduos para reprodução com base na sua aptidão (fitness);
Aplicar cruzamento e mutação aos indivíduos seleccionados para reprodução;
Substituir população antiga pela nova;
ATÉ acabar.

Como se pode verificar o algoritmo começa por gerar a população inicial, repetindo depois várias acções até atingir um óptimo global ou até que se exceda o número de gerações.

Em cada geração é calculado o *fitness* de cada indivíduo, com base na função objectivo, para todos os indivíduos que constituem a população. Posteriormente são seleccionados, utilizando um dos métodos de selecção, os indivíduos que serão utilizados para a reprodução. Aplicam-se depois os operadores cruzamento e mutação aos indivíduos seleccionados. Os indivíduos obtidos constituem a população da geração seguinte.

3. Exemplificação de um Algoritmo Genético Simples

O algoritmo genético simples apresentado anteriormente será exemplificado utilizando um exemplo retirado de Goldberg (1989).

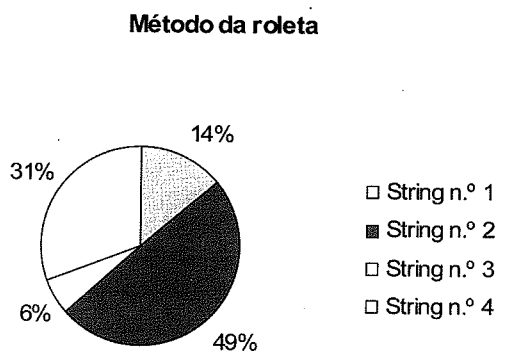
Imagine-se que se pretende maximizar a função $f(x)=x^2$, onde x pode variar entre 0 e 31. Torna-se necessário codificar as variáveis de decisão do problema, que para o exemplo apresentado corresponde à variável x. Esta variável será codificada utilizando o alfabeto binário {0,1} e como uma *string* de 5 bits, uma vez que são necessários apenas 5 bits para representar os números inteiros entre 0 e 31 (2^5 =32, permite representar 32 números onde a *string* 11111 representa o número decimal 31. Repare-se que $1*2^4 + 1*2^3 + 1*2^2 + 1*2^1 + 1*2^0 = 16 + 8 + 4 + 2 + 1 = 31$).

A seguir é gerada aleatoriamente a população inicial, que por uma questão de simplificação, será constituída por apenas 4 indivíduos. Os indivíduos gerados estão representados na tabela 1, juntamente com o valor decimal correspondente e o valor da função objectivo. Como se trata de um problema de maximização e a função objectivo, f(x)= x2, toma sempre valores positivos para todos os valores de x, o fitness de cada indivíduo será calculado directamente com base na função objectivo. A selecção dos indivíduos para reprodução pode ser realizada de diversas formas, neste caso é realizada utilizando uma roleta (vide 4.4.1 e figura 1) que está dividida em função do fitness de cada indivíduo. A roleta foi girada tantas vezes quanto o número de indivíduos que constituem a população, tendo sido seleccionados a string n.º 1 uma única vez, a string n.º 2 duas vezes e a string n.º 4 uma vez. A string n.º 3 não foi seleccionada. Este resultado demonstra que, em princípio, os melhores indivíduos são copiados mais vezes, os que estão na média são mantidos e os piores não são seleccionados.

Tabela 1 - Exemplificação do algoritmo genético simples (uma geração) (adaptado de Goldberg 1989)

Número da *String*	População Inicial	x	f(x)=x²	$\frac{fi}{\sum f}$	$\frac{fi}{\bar{f}}$	N.º de cópias (roleta)	*Strings* seleccionadas para reprodução	Ponto de Cruzamento	Nova População	x	f(x)=x²
1	01101	13	169	0.14	0.58	1	01101	4	01100	12	144
2	11000	24	576	0.49	1.97	2	11000	4	11001	25	625
3	01000	8	64	0.06	0.22	0	11000	2	11011	27	729
4	10011	19	361	0.31	1.23	1	10011	2	10000	16	256
Soma (Σf)			1170	1.00	4.00	4.0					1754
Média			293	0.25	1.00	1.0					439
Máximo			576	0.49	1.97	2.0					729

Figura 1 – Método da roleta



Depois de seleccionados os indivíduos para reprodução procedeu-se à aplicação do operador cruzamento. Como se considerou uma probabilidade de cruzamento de 1.0 (vide 4.4.2), nenhum indivíduo passa directamente para a população da geração seguinte, ou seja, todos os indivíduos são sujeitos ao operador cruzamento. Primeiro foram seleccionados aleatoriamente (moeda ao ar) os pares que se iriam constituir; a seguir foram seleccionados, também aleatoriamente (moeda ao ar), os pontos de cruzamento. Os resultados são os indicados na tabela 1: as *strings* 01101 e 11000 reproduzem-se, tendo sido seleccionado o ponto de cruzamento 4, enquanto que as *strings* 11000 e 10011 têm como ponto de cruzamento o 2. Os descendentes obtidos do cruzamento do primeiro par são as *string* 01100 e 11001, os do segundo par são 11011 e 10000.

O último operador a aplicar é a mutação. Supondo que a probabilidade de mutação para este problema é de 0.001 (vide 4.4.3), e que são transferidos 20 (4*5) bits nesta geração, espera-se que 20*0.001=0.02 bits sejam sujeitos à mutação, ou seja, nenhum bit sofrerá uma mutação (de 0 para 1 ou 1 para 0) nesta geração.

Depois dos operadores selecção, cruzamento e mutação terem sido aplicados, obtém-se uma nova população que será utilizada na geração seguinte. Como se pode verificar pela tabela 1, na 2ª geração a população tornou-se em média mais apta que a população da geração inicial (o valor médio da função objectivo aumentou de 293 para 439). Por outro lado a *string* 11000, que era o indivíduo mais apto (com maior *fitness*), "produziu" um indivíduo ainda mais apto (o valor máximo da função objectivo passou de 576 para 729).

Como se pode verificar, embora as circunstâncias tenham sido favoráveis, não foi por acaso que o melhor indivíduo da primeira geração foi copiado duas vezes devido à sua adaptabilidade elevada e acima da média. Por outro lado quando este indivíduo se combina para cruzamento com o segundo melhor indivíduo dessa geração, um dos descendentes revela-se ser o melhor na geração seguinte.

A seguir são explicados com mais pormenor alguns dos conceitos referidos anteriormente.

4. Desenvolvimento dos conceitos

4.1. Codificação

Para representar os genes dos cromossomas da população é necessário utilizar um alfabeto. Geralmente é utilizado o alfabeto binário {0,1} para representá-los, sendo

que nesta representação cada cromossoma corresponde a uma *string* de 0s e 1s. No entanto, podem ser utilizados outros alfabetos. Por outro lado cada cromossoma deverá representar uma solução factível do problema, ou seja, devem ser cromossomas válidos no espaço de busca. Um dos resultados fundamentais da teoria dos algoritmos genéticos, o Teorema dos Esquemas (será explicado posteriormente), refere que a codificação sobre a qual os algoritmos genéticos funcionam melhor é aquela que utiliza um alfabeto de cardinalidade 2.

4.2. População inicial e tamanho da população

Uma vez definida a codificação dos indivíduos é necessário definir a população inicial. Esta população pode ser criada de forma aleatória ou utilizando técnicas heurísticas que permitam uma aceleração da convergência do algoritmo genético, tendo esta opção a desvantagem de poder conduzir a uma convergência prematura do algoritmo para um óptimo local em vez do global. Estudos empíricos definem que o tamanho da população deverá estar compreendido entre 30 a 100 indivíduos. O tamanho da população é importante no algoritmo genético porque com uma população pequena corre-se o risco de não cobrir adequadamente o espaço de busca, enquanto que uma população numerosa pode conduzir a um elevado custo computacional.

4.3. Funções objectivo e *Fitness*

Nos algoritmos genéticos os indivíduos são avaliados de acordo com a função objectivo, que define o problema em estudo, ou seja, a função objectivo fornece uma medida de como os indivíduos se comportam no domínio do problema. Num problema de maximização, os indivíduos mais aptos terão o maior valor numérico associado à função objectivo, enquanto que num problema de minimização, os indivíduos mais aptos são aqueles que possuem o menor valor numérico da função objectivo correspondente. Esta medida primária da aptidão (*fitness*) é geralmente utilizada para poder determinar a aptidão relativa (*fitness* relativo) dos indivíduos.

A função *fitness* é normalmente utilizada para transformar o valor da função objectivo numa medida do *fitness* relativo, ou seja:

$$F(x) = g(f(x))$$

onde f é a função objectivo, g transforma o valor da função objectivo num número não negativo e F é o *fitness* relativo resultante. Este mapeamento é sempre necessário quando a função objectivo corresponde a um problema de minimização e quando, num problema de maximização, a função objectivo tome valores negativos.

Em muitos casos o valor da função *fitness* corresponde ao número de descendentes que um indivíduo espera "produzir" na próxima geração. Uma transformação muito utilizada é a proporcional ao *fitness*, ou seja, o *fitness* individual $F(x_i)$ de cada indivíduo é calculado em função da performance de cada indivíduo, isto é, da medida primária da aptidão, $f(x_i)$, relativamente a toda a população:

$$F(x_i) = \frac{f(x_i)}{\sum_{i=1}^{NInd} f(x_i)}$$

onde N_{ind} corresponde ao tamanho da população e x_i é o valor do fenótipo do indivíduo *i*.

Existem basicamente duas estratégias de atribuição do *fitness*:

- *Scaling*: o *fitness* é calculado com base na função objectivo para que o *fitness* de todos os indivíduos tome sempre valores não negativos, permitindo controlar a distribuição de *fitness* pelos indivíduos da população e limitar o número de vezes que cada indivíduo é seleccionado para reprodução;
- *Ranking*: o mapeamento é realizado ordenando a população pela função objectivo e, consequentemente, atribuindo o valor *fitness* a cada indivíduo de acordo com a sua posição na população ordenada.

4.4. Operadores genéticos

Os operadores genéticos transformam a população através de sucessivas gerações, sendo necessários para que a população se diversifique e mantenha as características de adaptação propagadas pelas gerações anteriores. Existem vários operadores, sendo considerados mais importantes os que a seguir se descrevem.

4.4.1. Selecção

O operador selecção permite seleccionar os indivíduos que irão ser utilizados na reprodução. O objectivo da selecção é escolher os melhores indivíduos (com maior *fitness*) para que estes originem descendentes ainda mais adaptáveis ao problema. Existem vários métodos de selecção dos quais se explicitam alguns:

- *Método da Roleta (Roulette Wheel Selection ou*

RWS): este método foi proposto por Holland e baseia-se na selecção proporcional ao *fitness*, ou seja, o número de vezes que se espera que um indivíduo seja seleccionado para reproduzir corresponde ao valor obtido na função *fitness* dividido pelo valor médio do *fitness* de toda a população. Uma forma de implementar este método é utilizando uma roleta onde os indivíduos estão representados proporcionalmente ao valor obtido na função *fitness* (vide figura 1). A roleta é girada tantas vezes quanto o tamanho da população, determinando quais os indivíduos que serão utilizados para reprodução. Com este método é possível que o número de descendentes que cada indivíduo origina seja diferente do valor esperado (pode acontecer que o pior indivíduo (com menor *fitness*) seja escolhido em todas as vezes em que a roleta for girada);

→ *SUS ou Stochastic Universal Sampling*: este método foi proposto por Baker (1987) e corresponde a um método de amostragem diferente, embora utilize o mesmo princípio da selecção proporcional ao *fitness*. Neste método não se gira a roleta N vezes para seleccionar os N progenitores, mas apenas uma única vez, tendo a roleta N ponteiros equitativamente espaçados que permitem seleccionar os N progenitores num único giro da roleta. Este método evita o problema indicado no método da roleta, no entanto, continua a não resolver os problemas que surgem quando a selecção é proporcional ao *fitness*, ou seja, no início da procura um número reduzido de indivíduos considerados mais aptos do que outros e os seus descendentes multiplicam-se rapidamente, fazendo com que a convergência do algoritmo seja prematura, impedindo que o algoritmo explore o espaço de busca. Por outro lado, no final da procura, quando todos os indivíduos da população são semelhantes, torna a evolução muito lenta;

→ *Baseado na ordem (Ranking)*: neste método todos os indivíduos da população são ordenados pelo valor obtido na função *fitness* e os primeiros indivíduos são replicados com várias cópias. O número de exemplares replicados por cada indivíduo é proporcional à sua posição no *ranking*;

→ *Selecção por torneio*: este método escolhe aleatoriamente um número T de indivíduos da população e aquele que tem maior pontuação é escolhido para que o seu descendente substitua o que tem menor pontuação. Este processo é repetido até que o número de indivíduos seleccionados corresponda ao tamanho da população;

→ *Elitista*: foi introduzido por De Jong (1975) e é um dos métodos de selecção que obriga o algoritmo genético a levar um número determinado de indivíduos considerados os mais aptos em cada geração, para a geração seguinte. Tais indivíduos podem ser perdidos se não forem seleccionados para reprodução ou se forem destruídos pelo cruzamento ou mutação. Regra geral a elite tem um tamanho reduzido, situando-se entre 1 ou 2 indivíduos numa população de 50. Estes indivíduos podem ser escolhidos por amostragem directa, escolhendo-se os melhores ou aleatoriamente de entre os melhores.

4.4.2. Cruzamento

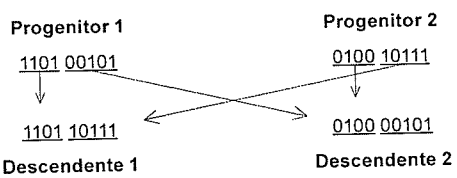
O operador cruzamento é considerado o operador mais importante dos algoritmos genéticos. Permite fazer a troca de segmentos entre os indivíduos progenitores que foram seleccionados para reproduzirem-se, ou seja, permite propagar as características dos indivíduos considerados mais aptos. Este operador é aplicado com uma probabilidade dada pela taxa de cruzamento P_c, que segundo De Jong (1975), deverá situar-se entre os 0.65 e 0.85, para se obterem resultados satisfatórios. Uma taxa de cruzamento de 1.0 indica que todos os indivíduos são seleccionados para reprodução, ou seja, não existem sobreviventes da geração anterior na seguinte. Quanto maior for esta taxa mais rapidamente novas estruturas serão introduzidas na população. No entanto, se for muito alta, a maior parte da população será substituída, podendo ocorrer perda de indivíduos com alta aptidão. Por outro lado com um valor baixo, o algoritmo pode tornar-se muito lento.

Quando a representação utilizada é a binária, o cruzamento pode ser realizado de diversas formas, sendo de destacar as seguintes:

a) N-pontos: os cromossomas são "cortados" em n pontos e trocam-se os segmentos entre os dois indivíduos, gerando dois descendentes. O cruzamento de um ponto e o de dois pontos são as formas mais habituais. De Jong (1975) chegou à conclusão que o cruzamento de dois pontos permite obter valores satisfatórios, não acontecendo o mesmo para mais pontos.

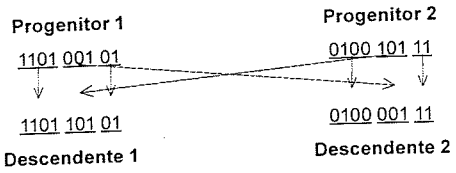
Na figura 2 é apresentado um exemplo de cruzamento de um ponto, tendo o 4º bit sido escolhido para cruzamento.

Figura 2 – Cruzamento de um ponto



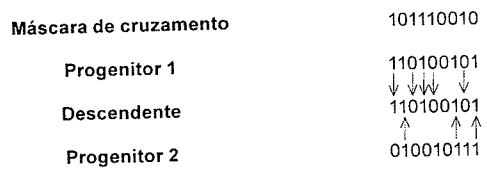
A figura 3 apresenta um exemplo de cruzamento de 2 pontos, tendo sido escolhidos o 4º e 7º bits para cruzamento.

Figura 3 – Cruzamento de dois pontos



a) Uniforme: neste tipo de cruzamento, cada gene dos descendentes resulta da cópia do gene correspondente de um dos progenitores, escolhido em função da máscara de cruzamento gerada aleatoriamente. Assim, quando existe um 1 na máscara de cruzamento, o gene é copiado do primeiro progenitor; se existir um 0, o gene é copiado do segundo progenitor. A figura 4 mostra um exemplo deste tipo de cruzamento.

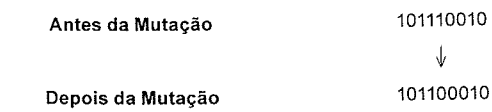
Figura 4 – Cruzamento Uniforme



4.4.3. Mutação

O operador mutação é um operador que permite a introdução e a manutenção de diversidade genética na população. Quando a representação utilizada é a binária, um bit sofre uma mutação quando é alterado de 0 para 1 ou de 1 para 0. De notar que a alteração de um bit num cromossoma pode fazer com que este represente outro ponto, completamente diferente, no espaço de busca. A figura 5 mostra o operador mutação aplicado ao 5º bit.

Figura 5 – Operador mutação



O operador mutação é aplicado depois do cruzamento e geralmente a probabilidade de mutação está compreendida entre 0.001 e 0.1. Este operador possibilita que se chegue a qualquer ponto do espaço de busca, sendo portanto a solução quando o algoritmo genético não avança. Uma taxa elevada torna a busca essencialmente aleatória, além de aumentar a possibilidade de que uma boa solução seja destruída.

Geralmente a probabilidade de mutação mantêm-se constante ao longo de todas as gerações. No entanto, alguns estudos referem que se obtêm melhores resultados modificando a taxa de mutação à medida que se avança nas gerações.

Depois de aplicar os operadores cruzamento e mutação, podem-se obter cromossomas não válidos do espaço de busca, sendo necessário implementar procedimentos para resolver esta situação, que poderão ser:

a) Continuar a aplicar os operadores cruzamento e mutação até obter um indivíduo válido;

b) Atribuir-lhe um valor igual a zero na função objectivo;

c) Reconstruir o indivíduo aplicado-lhe um operador que se costuma designar por Reparador.

Existem outros operadores, menos utilizados, que geralmente se aplicam a problemas específicos e que, por isso, não serão aqui explicitados.

5. Teorema dos Esquemas

A teoria tradicional dos algoritmos genéticos assume que os algoritmos genéticos descobrem e recombinam "blocos" de soluções, ou seja, que as boas soluções são geralmente compostas por "blocos".

Holland introduziu em 1975 a noção de esquemas para formalizar a noção de "blocos". Um esquema é considerado um *template* que permite a exploração de similaridades entre os cromossomas. É construído introduzindo o símbolo * no alfabeto utilizado na codificação dos genes. Um esquema representa todas as *string*s em que coincidem os 0s e 1s em todas as posições distintas das ocupadas por *.

Por exemplo, se considerarmos *string*s de comprimento 4, temos que o esquema S=(1*0*) representa as seguintes *string*s: (1000), (1001), (1100) e (1101).

O esquema (1100) representa uma única *string*, enquanto que o esquema (****) representa todas as *string*s de comprimento 4.

Cada esquema representa exactamente 2^r *string*s, onde r corresponde ao número de * no esquema. Por outro lado, cada *string* de comprimento m é representada por 2^m esquemas. Por exemplo, a *string* 111 pode ser representada pelos seguintes esquemas (2^3=8): (111), (11*), (1*1), (*11), (1**), (*1*), (**1) e (***).

Por sua vez, para *string*s de comprimento m, existem no total 3^m esquemas possíveis.

Os esquemas caracterizam-se por duas propriedades: a Ordem e o Comprimento.

A ordem de um esquema S, denotado por o(S), é o número de 0s e 1s no esquema. Trata-se, portanto, do número de posições fixas no esquema. Por outras palavras, trata-se do comprimento do esquema menos o número de *. Por exemplo, os seguintes esquemas: S1=(1*1*), S2=(0010) e S3=(***1), têm as seguintes ordens: o(S1)=2, o(S2)=4 e o(S3)=1.

A noção de ordem do esquema é necessária para calcular a probabilidade de sobrevivência do esquema em relação ao operador mutação.

O comprimento do esquema S, denotado por δ(S), é a distância entre a primeira e a última posição fixa do esquema. Define a compactabilidade da informação contida no esquema. Em relação aos esquemas S1, S2, e S3 definidos anteriormente, os seus comprimentos são os seguintes: δ(S1)=3-1=2, δ(S2)=4-1=3 e δ(S3)=4-4=0.

A noção de comprimento do esquema é necessária para calcular a probabilidade de sobrevivência do esquema em relação ao operador cruzamento.

O Teorema dos Esquemas estabelece que: *esquemas curtos, de baixa ordem e com adaptação ao problema superior à adaptação média da população, obtêm nas gerações subsequentes do algoritmo genético, um incremento exponencial no número de indivíduos que se associam aos esquemas.*

Um resultado imediato do Teorema dos Esquemas é a Hipótese dos Blocos, que estabelece que: *o algoritmo genético procura o óptimo global através da justaposição de esquemas curtos, de baixa ordem e com uma adaptação superior à média.*

A Hipótese dos Blocos sugere que o problema da codificação num algoritmo genético é crucial para o seu bom desempenho e que essa codificação deverá ter em conta a hipótese enunciada. No entanto, a utilização dos blocos tal como enunciado pela Hipótese dos Blocos, pode desorientar o algoritmo genético e fazê-lo convergir para óptimos locais. Este fenómeno denomina-se por *Deception*. Existem três formas de lidar com este problema:

a) Conhecimento prévio da função objectivo para que a codificação seja realizada da forma mais apropriada;

b) Utilização do operador Inversão que permite a melhor organização de bits para formar os blocos;

c) Utilizando os denominados *Messy Genetic Algorithms* (Goldberg *et al* 1989).

O Teorema dos Esquemas e as suas implicações no comportamento dos algoritmos genéticos tem sido alvo de críticas por parte da comunidade académica, especialmente daqueles que propõem novas abordagens aos algoritmos genéticos.

6. Aplicabilidade

Os algoritmos genéticos foram aplicados em diversas áreas, sendo de destacar as seguintes (Goldberg 1989, Hurley *et al.* 1996, Mitchell 1996):

- ➔ Serviços financeiros – optimização de *portfolio*, atribuição de crédito;
- ➔ Gestão estratégica – previsão dos lucros da empresa;
- ➔ Problemas de localização – planeamento da localização;
- ➔ Problemas de distribuição e de transporte;
- ➔ Planeamento da produção – custo, inventário, produção e mão-de-obra;
- ➔ Problemas de calendarização – horários escolares;
- ➔ Engenharia civil – optimização discreta de estruturas.

7. Conclusões

Como se verificou ao longo deste documento, os algoritmos genéticos são métodos de busca e optimização que simulam os processos naturais de evolução. São técnicas eficientes na busca de soluções óptimas, ou aproximadamente óptimas, numa grande variedade de problemas, uma vez que não impõem algumas das limitações encontradas noutros métodos de busca tradicionais.

Como já foi referido, este artigo surge como introdução ao tema, tendo-se decidido apresentar os conceitos de forma simplificada. Como não foi possível explicar e aprofundar alguns dos termos indicados, por limitações do tamanho do artigo, a autora pretende apresentar um próximo artigo para aprofundar os conceitos abordados.

Referências Bibliográficas

BAKER, J. E. (1987) "Reducing Bias and Inefficiency in the Selection Algorithm" - In *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms.*

CHIPPERFIELD, A. Fleming, P.; Pohlheim, H.; Fonseca, C. M.: *Genetic Algorithm Toolbox for Use with Matlab,* User´s Guide, version 1.2.

DAVIS, L. (ed.) (1991) *Handbook of Genetic Algorithms* – Van Nostrand Reinhold, New York.

DE JONG, K. A. (1975) *An Analysis of the Behaviour of a Class of Genetic Adaptive Systems*, PhD dissertation, University of Michigan.

FONSECA, C. M (1995) *Multiobjective Genetic Algorithms with Application to Control Engineering Problems*, PhD dissertation, University of Sheffield.

FONSECA, C. M. e Fleming, P. J. (1995) "An Overview of Evolutionary Algorithms" In *Multiobjective Optimization – Evolutionary Computation,* pp. 1-16.

GEN, Mitsuo e Cheng, Runwei (1999) *Genetic Algorithms and Engineering Design*, Wiley Series in Engineering Design and Automation.

GOLDBERG, D. E. (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Company, Inc.

GOLDBERG, D. E., Korb, B. e Deb, K. (1989) "Messy Genetic Algorithms: Motivation, Analysis, and First Results" In Complex Systems, 3(5) pp. 493-530.

HOLLAND, John H. (1975) *Adaptation in Natural and Artificial Systems* – University of Michigan, Press Ann Arbor.

HURLEY, S.; Moutinho, L. (1996) "Approximate Algorithms for Marketing Management Problems" - *In Journal of Retailing and Consumer Services*, Vol. 3, N.º 3 pp. 145-154.

LARRAÑAGA, P. (05/11/2001) *Algoritmos Genéticos* - http://www.geocities.com/CapeCanaveral/9802/3d5ca000.htm.

MICHALEWICZ, Zbigniew (1992) *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag.

MITCHELL, Melanie (1996) *An Introduction to Genetic Algorithms*, The MIT Press.

VELDHUIZEN, D. A.; Lamont, G. B. (2000) *Multiobjective Evolutionary Algorithms: Analyzing The State-of-the-Art*, Evolutionary Computation pp. 125-147.