

Programação de Expressão Genética: um Novo Algoritmo Adaptável para Resolver Problemas

Cândida Ferreira

1. *candidaf@gene-expression-programming.com*

2. *http://www.gene-expression-programming.com*

3. *Departamento de Ciências Agrárias, Universidade dos Açores, Terra Chã, 9701-851, Angra do Heroísmo, Portugal*

A programação de expressão genética, um algoritmo genético com genótipo/fenótipo (linear/não-linear), é apresentada aqui pela primeira vez como uma nova técnica para criação de programas de computador. A programação de expressão genética usa cromossomas lineares de caracteres compostos por genes organizados estruturalmente numa cabeça e numa cauda. Os cromossomas funcionam como genoma e estão sujeitos a modificação por meio de mutação, transposição, transposição para a raiz, transposição génica, recombinação génica, recombinação num e dois pontos. Os cromossomas codificam árvores de expressão, sendo estas o objecto da selecção. A criação destas duas entidades distintas (genoma e árvore de expressão) com funções diferentes, permite que o algoritmo tenha um grande desempenho: nos problemas de regressão simbólica, indução de sequências e empilhamento de blocos, o algoritmo supera a programação genética em mais de duas ordens de grandeza, enquanto que no problema da classificação da densidade ele supera a programação genética em mais de quatro ordens de grandeza. A galeria de problemas escolhida para ilustrar o poder e a versatilidade da expressão de programação genética inclui, além dos problemas mencionados acima, dois problemas de síntese lógica: o multiplexer de 11 bits e o problema da regra PG.

1. Introdução

A programação de expressão genética (PEG) é, à semelhança dos algoritmos genéticos (AGs) e da programação genética (PG), um algoritmo genético pois usa populações de indivíduos, selecciona os indivíduos de acordo com a sua aptidão e introduz variação genética usando um ou mais operadores genéticos [1]. A diferença fundamental entre os três algoritmos reside na natureza dos indivíduos: nos AGs os indivíduos são cadeias lineares de tamanho fixo (cromossomas); na PG os indivíduos são entidades não-lineares de diferentes tamanhos e formas (árvores analíticas); e na PEG os indivíduos são codificados em cadeias lineares de tamanho fixo (o genoma ou cromossoma) que são expressas posteriormente como entidades não-lineares com diferentes tamanhos e formas (representações esquemáticas simples ou árvores de expressão).

Se tivermos em conta a história da vida na Terra [2], podemos ver que a diferença entre os AGs e a PG é superficial: ambos os sistemas utilizam somente um tipo de entidade que funciona tanto como genoma como corpo (fenoma). Estes sistemas estão condenados a ter uma de duas limitações: se são fáceis de manipular geneticamente, perdem em complexidade funcional (o caso dos AGs); se possuem um certo grau de complexidade funcional, são extremamente difíceis de reproduzir com modificação (o caso da PG).

Os GAs, com o seu genoma simples e uma diversidade estrutural e funcional limitadas, assemelham-se a um

Mundo de ARN primitivo [2], enquanto que a PG com a sua diversidade estrutural e funcional se assemelha a um Mundo Proteico hipotético. Somente quando moléculas capazes de replicação se associaram a moléculas com actividade catalítica foi possível criar sistemas mais complexos e, eventualmente, a primeira célula. Desde essa altura, o genoma e o fenoma presumem-se um ao outro e nenhum pode funcionar sem o outro. Similarmente, os cromossomas e as árvores de expressão da PEG presumem-se mutuamente e nenhum existe isoladamente.

As vantagens dum sistema como a PEG estão patentes na natureza, no entanto convém salientar as mais importantes: Primeiro, os cromossomas são entidades simples: lineares, compactas, relativamente pequenas, fáceis de manipular geneticamente (replicar, mutar, recombinar, transpor, etc.). Segundo, as árvores de expressão (AEs) são exclusivamente a expressão do cromossoma respectivo; elas são as entidades sobre as quais a selecção opera, sendo, de acordo com a sua aptidão, seleccionadas para se reproduzirem com modificação. Durante a reprodução são os cromossomas dos indivíduos e não as AEs que são reproduzidos com modificação e transmitidos à geração seguinte.

A acção conjunta dos cromossomas e das AEs implica a existência dum sistema de tradução inequívoco para traduzir a linguagem dos cromossomas para a linguagem das AEs. A organização estrutural dos cromossomas da PEG apresentada neste trabalho permite este tipo de interacção, pois qualquer modificação feita no genoma resulta sempre em AEs ou programas sintacticamente

correctos. De facto, o conjunto variado de operadores genéticos desenvolvido para introduzir diversidade genética nas populações da PEG produz sempre AEs válidas. Assim, a PEG é um sistema de vida artificial muito simples capaz de adaptação e evolução.

Graças a estas características, a PEG é extremamente versátil e supera imensamente as técnicas evolutivas existentes. De facto, no problema mais complexo apresentado neste trabalho, a evolução de regras para o problema da classificação da densidade em autómatos celulares, a PEG supera a PG em mais de quatro ordens de grandeza.

No trabalho presente apresenta-se a organização estrutural e funcional dos cromossomas da PEG; mostra-se de que forma a linguagem dos cromossomas é traduzida para a linguagem das AEs; como os cromossomas funcionam como genótipo e as AEs como fenótipo; e como um programa individual é criado, amadurecido e reproduzido, deixando descendentes com novas propriedades, logo capazes de adaptação. O trabalho procede com uma descrição detalhada da PEG e uma ilustração desta técnica com seis exemplos vindos de campos diferentes, comparando ao mesmo tempo o desempenho da PEG com a PG.

2. Algoritmos de expressão genética: uma resenha

O fluxograma dum algoritmo de expressão genética (AEG) está representado na Figura 1. O processo começa com a criação aleatória dos cromossomas da população inicial. Depois os cromossomas são expressos e a aptidão de cada indivíduo é calculada. Os indivíduos são posteriormente seleccionados de acordo com a aptidão para se reproduzirem com modificação, deixando descendentes com características novas. Os indivíduos desta nova geração são, por sua vez, sujeitos ao mesmo processo de desenvolvimento: expressão dos genomas, confrontação com o ambiente de selecção e reprodução com modificação. O processo é repetido por um número determinado de gerações ou até se encontrar uma solução.

Note-se que a reprodução inclui não só a replicação mas também a acção dos operadores genéticos capazes de criar diversidade. Durante a replicação, o genoma é copiado e transmitido à geração seguinte. Obviamente a replicação por si só é incapaz de introduzir variação: somente com a acção dos restantes operadores é que a variação genética é introduzida na população. Estes operadores seleccionam aleatoriamente os cromossomas a ser modificados. Assim, um cromossoma pode tanto ser modificado por um ou vários operadores ao mesmo tempo como não ser sequer modificado. Os detalhes da implementação dos operadores da PEG estão descritos na secção 5.

3. O genoma dos indivíduos da PEG

Na PEG, o genoma ou cromossoma é uma cadeia linear simbólica de tamanho fixo, composta por um ou mais genes. Veremos que apesar do seu tamanho fixo, os cromossomas da PEG codificam para AEs com diferentes tamanhos e formas.

3.1. Grelhas de leitura aberta e genes

A organização estrutural dos genes da PEG é mais facilmente compreendida em termos de grelhas de leitura aberta (GLAs). Em biologia, uma GLA ou sequência

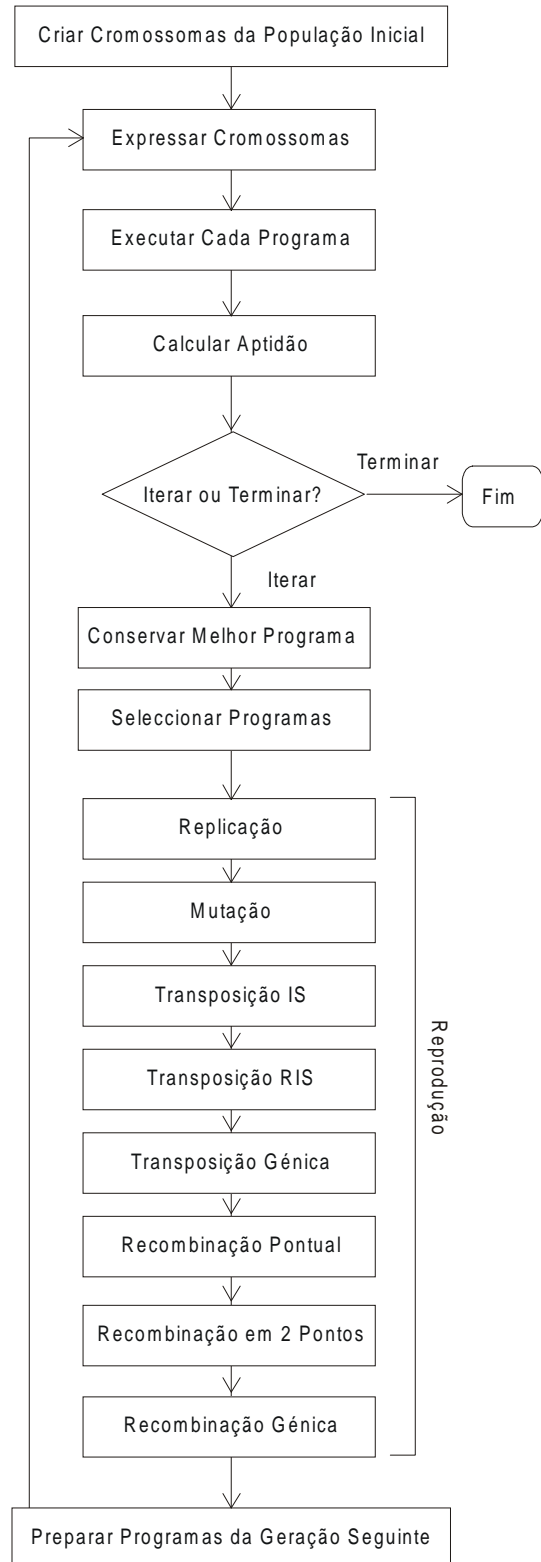


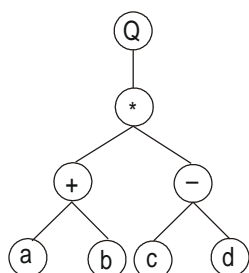
Figura 1. Fluxograma dum algoritmo de expressão genética.

codificadora dum gene, começa com o codão de iniciação, continua com os codões dos aminoácidos, e acaba no codão de terminação. No entanto, um gene é mais do que a GLA respectiva, com sequências a montante do codão de iniciação e sequências a jusante do codão de terminação. Na PEG, apesar do codão de iniciação ser sempre a primeira posição dum gene, o ponto de terminação nem sempre coincide com a última posição do gene. É comum os genes da PEG terem regiões não-codificadoras a jusante do ponto de terminação. (Por enquanto não vamos considerar estas regiões não-codificadoras já que elas não interferem com o produto da expressão.)

Considere, por exemplo, a expressão algébrica:

$$\sqrt{(a+b)} \times (c-d) \quad (3.1)$$

Também pode ser representada como um diagrama ou AE:



onde 'Q' representa a função raiz quadrada. Este tipo de diagramas constituem de facto o fenótipo dos indivíduos da PEG, sendo o genótipo facilmente inferido a partir do fenótipo como se indica:

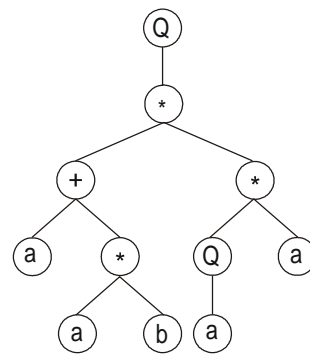
$$\begin{array}{l} 01234567 \\ Q^*+-abcd \end{array} \quad (3.2)$$

isto é, a leitura directa da AE da esquerda para a direita e de cima para baixo. A expressão 3.2 é uma GLA, começando em 'Q' (posição 0) e terminando em 'd' (posição 7). Estas GLAs foram designadas expressões K (de *Karva*, o nome que eu escolhi para a linguagem da PEG). Note-se que este ordenamento difere tanto das expressões postfix como prefix utilizadas em implementações da PG com filas ou pilhas [3].

O processo inverso, isto é, a tradução duma expressão K para uma AE, também é muito simples. Considere outra GLA, a expressão K seguinte:

$$\begin{array}{l} 01234567890 \\ Q^*+*a*Qaaba \end{array} \quad (3.3)$$

O ponto de iniciação (posição 0) na GLA corresponde à raiz da AE. Depois, por baixo de cada função são colocados tantos ramos quantos os argumentos da função. A montagem fica completa quando se forma uma linha de base composta somente por terminais (as variáveis ou constantes utilizadas num problema). Neste caso, forma-se a seguinte AE:



Olhando somente para a estrutura das GLAs da PEG, é difícil ou mesmo impossível perceber as vantagens desta representação, com excepção talvez da sua simplicidade e elegância. No entanto, quando se analisam as GLAs no contexto dum gene, as vantagens desta representação tornam-se óbvias. Como já referi, os cromossomas da PEG têm tamanho fixo e são compostos por um ou mais genes de tamanho igual. Portanto o tamanho do gene também é fixo. Assim, na PEG, o que varia não é o tamanho dos genes, mas sim o tamanho das GLAs. De facto, o tamanho duma GLA pode ser igual ou menor que o tamanho do gene. No primeiro caso, o ponto de terminação coincide com o fim do gene e no último caso, o ponto de terminação está algures a montante do fim do gene.

Qual é então a função destas regiões não-codificadoras nos genes da PEG? De facto, elas são a essência da PEG e da evolução, pois elas permitem a modificação do genoma usando qualquer operador genético sem restrições, produzindo sempre programas sintacticamente correctos sem necessitar de um processo complicado de edição ou de formas extremamente restritivas de implementar os operadores genéticos. De facto, esta é a diferença crucial entre a PEG e implementações anteriores da PG, com ou sem genomas lineares (ver [4] para uma revisão sobre PG com genomas lineares).

3.2. Os genes da PEG

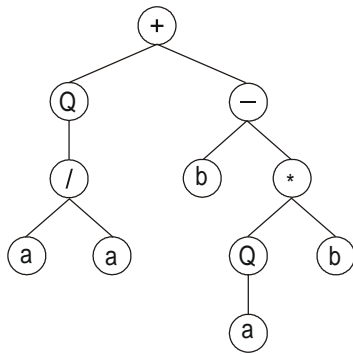
Os genes da PEG são compostos por uma cabeça e uma cauda. A cabeça contém símbolos que representam tanto funções como terminais, enquanto que a cauda contém somente terminais. Para cada problema, o tamanho da cabeça h é escolhido, enquanto que o tamanho da cauda t é uma função de h e do número de argumentos da função com mais argumentos (n), sendo calculada pela equação:

$$t = h(n - 1) + 1 \quad (3.4)$$

Considere um gene composto de $\{Q, *, /, -, +, a, b\}$. Neste caso $n = 2$. Por exemplo, para um $h = 10$, $t = 11$ e o tamanho do gene é $10+11=21$. Um gene deste tipo está indicado abaixo (a cauda está assinalada a cheio):

$$\begin{array}{l} 012345678901234567890 \\ +Q- /b^*aaQbaabaabbaaab \end{array} \quad (3.5)$$

Ele codifica a seguinte AE:

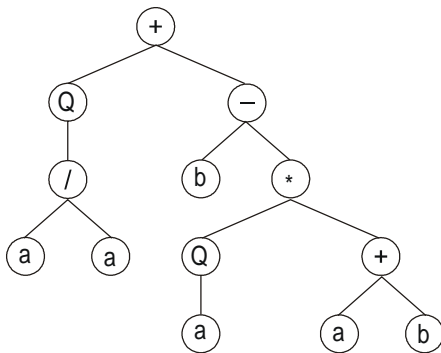


Neste caso, a GLA termina na posição 10, enquanto que o gene acaba na posição 20.

Suponha agora que uma mutação ocorre na posição 9, modificando o 'b' num '+'. Então forma-se o gene seguinte:

012345678901234567890
+Q- /b*aaQ+**aabaabbbaaab** (3.6)

E a sua expressão resulta em:

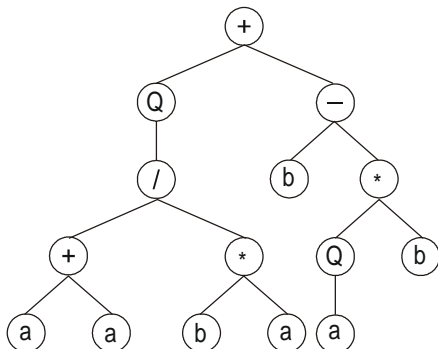


Neste caso o ponto de terminação desloca-se duas posições para a direita (posição 12).

Suponha agora que ocorre uma modificação mais radical e os símbolos nas posições 6 e 7 no gene 3.5 acima, são substituídos respectivamente por '+' e '*', criando o gene seguinte:

012345678901234567890
+Q- /b*+*Qb**aabaabbbaaab** (3.7)

A sua expressão dá:

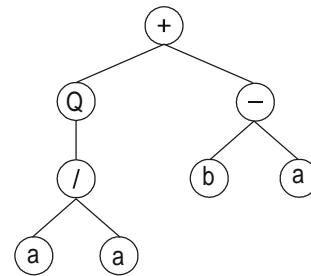


Neste caso o ponto de terminação desloca-se várias posições para a direita (posição 14).

Obviamente, também acontece o contrário, sendo a GLA encurtada. Por exemplo, considere o gene 3.5 acima e suponha que uma mutação ocorre na posição 5, modificando o '*' num 'a':

012345678901234567890
+Q- /baaaQb**aabaabbbaaab** (3.8)

A sua expressão resulta na AE seguinte:



Neste caso a GLA termina na posição 7, encurtando a AE original em três nós.

Apesar do seu tamanho fixo, cada gene tem potencial para codificar para AEs com diferentes tamanhos e formas, sendo a mais simples composta somente por um nó (quando o primeiro elemento de um gene é um terminal) e a maior composta por tantos nós quantas as posições do gene (quando todos os elementos da cabeça são funções com o número máximo de argumentos n).

Os exemplos apresentados acima mostram claramente que qualquer modificação feita no genoma, independentemente do seu radicalismo, resulta sempre numa AE válida. Obviamente, a organização estrutural dos genes tem que ser preservada, mantendo sempre as fronteiras entre a cabeça e a cauda e não permitindo que símbolos representantes de funções sejam introduzidos na cauda. Na secção 5 mostra-se o modo de funcionamento dos operadores genéticos e de que forma eles modificam o genoma dos indivíduos da PEG durante a reprodução.

3.3. Cromossomas multigénicos

Os cromossomas da PEG são normalmente compostos por mais do que um gene de tamanhos iguais. Para cada problema ou corrida, o número de genes e o tamanho da cabeça têm que ser escolhidos. Cada gene codifica para uma sub-AE e as sub-AEs interagem umas com as outras formando uma AE mais complexa com múltiplas subunidades. Os detalhes destas interações serão explicados plenamente na secção 3.4.

Considere, por exemplo, o cromossoma seguinte com 27 de tamanho e composto por três genes (as caudas estão assinaladas a cheio):

012345678012345678012345678
-b*b**abbab***Qb+**abbba**-*Q**abbaba** (3.9)

O cromossoma tem três GLAs, codificando cada GLA para uma sub-AE (Figura 2). A posição zero marca o início de cada gene; o fim de cada GLA, no entanto, só se torna evidente depois da construção da sub-AE respectiva. Como se mostra na Figura 2, a primeira GLA acaba na posição 4 (sub-AE₁); a segunda GLA acaba na posição 5 (sub-AE₂); e a última GLA também acaba na posição 5 (sub-AE₃). Assim, os cromossomas da PEG codificam para uma ou mais GLAs, cada uma expressando uma sub-AE particular. Dependendo do problema em questão, as sub-AEs podem ser seleccionadas individualmente de acordo com a aptidão respectiva (por exemplo, em problemas com saídas múltiplas), ou então podem formar uma AE mais complexa constituída por várias subunidades e ser seleccionadas de acordo com a aptidão da AE final composta por múltiplas subunidades. Os padrões de expressão e os detalhes da selecção serão discutidos ao longo deste trabalho. No entanto, tenha presente que cada sub-AE é ao mesmo tempo uma entidade separada e uma parte duma estrutura hierárquica mais complexa e, como em todos os sistemas complexos, o todo é mais do que a soma das partes.

3.4. Árvores de expressão e o fenótipo

Na natureza, o fenótipo tem diversos níveis de complexidade, sendo o organismo o mais complexo. Mas os ARNs de transporte, as proteínas, os ribossomas, as células, etc., também são produtos de expressão, e todos eles são, em última análise, codificados pelo genoma.

Ao contrário do que acontece na natureza, na PEG a expressão da informação genética é muito simples. Mesmo assim, os cromossomas da PEG são compostos por uma ou mais GLAs e, por conseguinte, os indivíduos codificados exibem diferentes níveis de complexidade. Os indivíduos mais simples são codificados por um único gene e, neste caso, o ‘organismo’ é o produto dum só gene - uma AE. Noutros casos, o ‘organismo’ é uma AE de múltiplas subunidades, estando as subunidades unidas por uma função particular. Noutros casos, o ‘organismo’ emerge da organização espacial de diferentes sub-AEs (por

exemplo, em problemas de planeamento e problemas com saídas múltiplas). E ainda noutros casos, o ‘organismo’ emerge das interacções entre sub-AEs convencionais com domínios diferentes (por exemplo, em redes neuronais). Em todos os casos, no entanto, o ‘organismo’ completo é codificado por um genoma linear.

3.4.1. Modificações pós-traducionais

Já vimos que a tradução resulta na formação de sub-AEs de diferente complexidade, mas a expressão completa da informação genética requer que estas sub-AEs interactuem entre si. Uma das interacções mais simples, consiste na ligação das sub-AEs com uma função particular. Este processo é semelhante à interacção das diferentes subunidades nas proteínas com múltiplas subunidades.

Quando as sub-AEs são expressões algébricas ou expressões booleanas, qualquer função matemática ou booleana com mais do que um argumento pode ser usada para ligar as sub-AEs numa AE final composta por múltiplas sub-AEs. As funções mais escolhidas são a adição para sub-AEs algébricas e OR ou IF para sub-AEs booleanas.

Na versão usada neste trabalho, para cada problema, a função de ligação é escolhida *a priori*, mas ela pode ser facilmente introduzida no genoma, por exemplo, na última posição do cromossoma, e também ser sujeita a adaptação. De facto, resultados preliminares sugerem que este sistema funciona muito bem.

A Figura 3 ilustra a ligação de duas sub-AEs pela adição. Note-se que a raiz da AE final (+) não é codificada pelo genoma. Note-se também que a AE final poderia ser facilmente linearizada numa expressão K:

$$0123456789012 \\ +Q** -bQ+abbba \quad (3.10)$$

No entanto, para evoluir soluções para problemas complexos, os cromossomas multigénicos são mais eficientes, pois eles permitem a construção modular de estruturas complexas e hierárquicas, em que cada gene

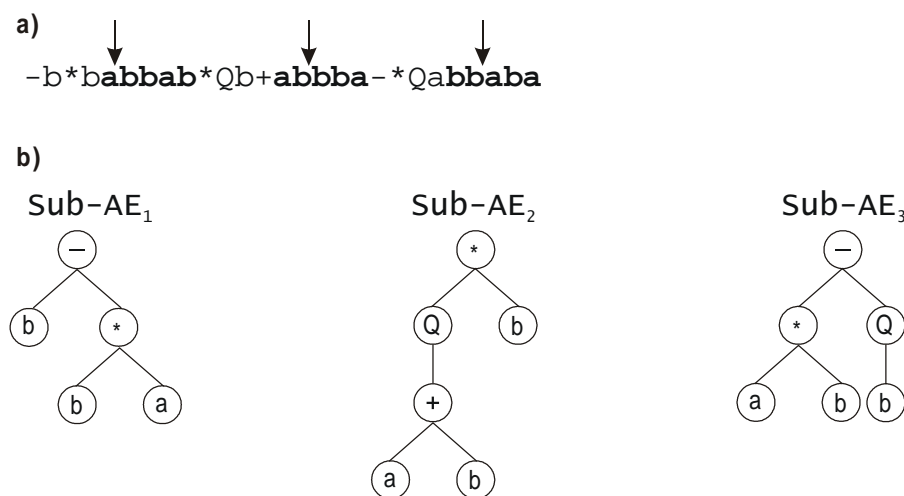


Figura 2. Expressão dos genes da PEG como sub-AEs. a) Um cromossoma tri-génico com as caudas salientadas a cheio. As setas indicam o ponto de terminação de cada gene. b) As sub-AEs codificadas por cada gene.

codifica para um pequeno bloco de construção. Estes pequenos blocos estão separados uns dos outros e, portanto, podem evoluir independentemente. Por exemplo, se tentássemos evoluir uma solução para o problema da regressão simbólica apresentado na secção 6.1 utilizando cromossomas uni-génicos, a taxa de sucesso cairia significativamente (ver secção 6.1). Neste caso, a descoberta de blocos pequenos está mais limitada pois eles não estão livres para poderem evoluir de forma mais independente. Este tipo de experiências mostra que a PEG é de facto um sistema de invenção poderoso e hierárquico capaz de evoluir facilmente blocos simples e usá-los posteriormente na formação de estruturas mais complexas. A Figura 4 mostra um outro exemplo de modificação pós-traducional, onde três sub-AEs booleanas estão ligadas pela função IF. Mais uma vez, a AE com múltiplas subunidades poderia ser linearizada na expressão K seguinte:

$$\begin{aligned} &01234567890123456789012 \\ &IINAIAINu1ca3aa2acAOab2 \end{aligned} \quad (3.11)$$

Na Figura 5 está indicado um outro exemplo de modificação pós-traducional, onde as sub-AEs são do tipo mais simples (sub-AEs com um só elemento). Neste caso, as sub-AEs são ligadas 3 a 3 pela função IF, depois estes agrupamentos são por sua vez também ligados 3 a 3 com outra função IF, e estes últimos agrupamentos também são ligados por um IF, formando uma AE final gigantesca composta por múltiplas subunidades. Este tipo de arquitectura cromossómica foi utilizado para evoluir soluções para o problema do multiplexer de 11 bits apresentado na secção 6.5.2 e também para evoluir regras para o problema da densidade em autómatos celulares (resultados não apresentados). Mais uma vez, o indivíduo da Figura 5 poderia ser convertido na expressão K seguinte:

$$IIIIIIIIIIIIIIII13lu3ab2ubab23c3ua3la333au3 \quad (3.12)$$

E, por último, a expressão completa de um cromossoma requer a execução sequencial de planos pequenos, em que a primeira sub-AE faz uma parte do trabalho, a segunda continua a partir daquele ponto, etc. O plano final resulta da acção ordenada de todos os sub-planos (ver o problema do empilhamento de blocos apresentado na secção 6.3).

O tipo de função de ligação, o número de genes e o tamanho da cabeça, são escolhidos *a priori* para cada problema. Por isso, pode-se sempre começar por usar um cromossoma de um só gene e aumentar gradualmente o tamanho da cabeça; se se tornar excessivamente grande, pode-se aumentar o número de genes, escolhendo evidentemente uma função para os ligar. Pode-se começar com a adição ou OR, mas noutros casos poderá ser mais conveniente utilizar outra função de ligação. O importante é encontrar uma solução boa, e a PEG oferece os meios para o fazer.

4. Funções de aptidão e selecção

Nesta secção são dados dois exemplos de funções de aptidão. Outros exemplos de funções de aptidão serão

oportunamente apresentados com os problemas da secção 6. O sucesso dum problema depende consideravelmente da forma como a função de aptidão é desenhada: o objectivo deve ser clara e correctamente definido de forma a fazer o sistema evoluir no sentido pretendido.

4.1. Funções de aptidão

Uma aplicação importante da PEG é a regressão simbólica, em que o objectivo consiste em descobrir uma função que tenha um bom desempenho para todos os casos de aptidão, isto é, devolva um valor que esteja dentro dum certo erro relativamente ao valor alvo. Em algumas aplicações matemáticas é útil utilizarem-se erros relativos ou absolutos relativamente baixos para se poderem descobrir soluções adequadas. Mas se a margem de selecção for demasiado apertada, as populações evoluem muito devagar e são incapazes de descobrir uma solução correcta. Por outro lado, se se alargar a margem de selecção, aparecerão inúmeras soluções com aptidão máxima que estarão longe de ser soluções adequadas.

Para resolver este problema, arquitectou-se uma estratégia evolutiva que permite a descoberta de soluções muito boas sem, no entanto, impedir a evolução. Assim, permite-se que o sistema encontre por si só a melhor solução possível dentro dum erro mínimo. Para isso atribui-se à selecção uma margem de manobra bastante ampla, por exemplo, um erro relativo de 20-100%, que permite que o processo evolutivo arranque. De facto, estes indivíduos iniciais de pouco servem, mas nas gerações futuras os seus descendentes adaptam-se lindamente, encontrando soluções muito boas que progressivamente se aproximam da solução perfeita. Matematicamente, para um caso de aptidão, a aptidão de um indivíduo é determinada pela equação:

$$f = M - |E| \quad (4.1)$$

em que M é a margem de selecção e E é o erro (relativo ou absoluto) entre o valor gerado pela AE e o valor alvo. Por exemplo, para um conjunto de 10 casos de aptidão e um $M = 100\%$, $f_{max} = 1000$ se todos os valores tiverem sido calculados exactamente ou se estiverem dentro dum certo valor (a precisão escolhida para o erro), por exemplo, 0,01%.

Noutra aplicação importante da PEG, aprendizagem de conceitos booleanos ou síntese lógica, a aptidão de um indivíduo é o número de casos de aptidão para os quais ele apresenta um desempenho correcto. No entanto, para a maior parte das aplicações booleanas, é importante penalizar os indivíduos capazes de resolver cerca de 50% dos casos de aptidão, pois isto reflecte, quase pela certa, os 50% de probabilidade de acertar numa função booleana. Por conseguinte, é aconselhável seleccionarem-se somente indivíduos capazes de resolver mais de 50-75% dos casos de aptidão. Abaixo deste limite, pode atribuir-se um valor simbólico à aptidão, por exemplo, um ponto de aptidão. A maior parte das vezes o processo de evolução arranca com estes indivíduos pouco úteis, pois eles são facilmente criados na população inicial. Mas nas gerações futuras começam a aparecer indivíduos mais adequados, espalhando-se facilmente na população. Para problemas

a)
 012345678012345678
 Q*Q+**bb**aa*-b**abaabb**

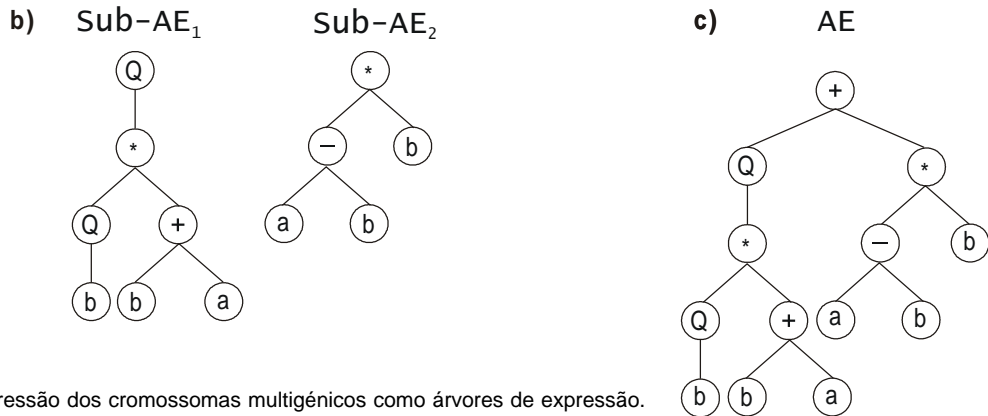


Figura 3. Expressão dos cromossomas multigénicos como árvores de expressão. a) Um cromossoma bi-génico com as caudas salientadas a cheio. b) As sub-AEs codificadas por cada gene. c) O resultado da ligação pós-traducional com a adição.

a)
 IIAI**ca3aa2acu**NNA**Oab2u3c31c**Au12**ua3112cac**

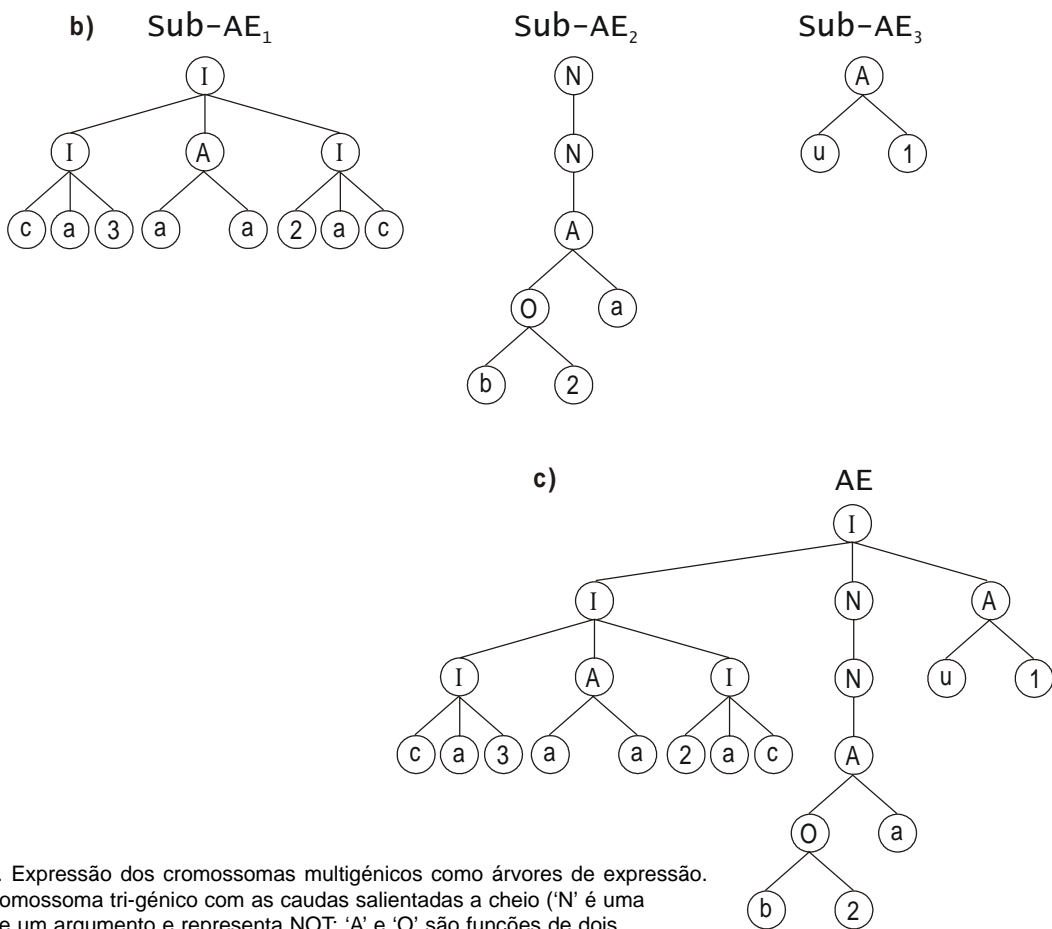


Figura 4. Expressão dos cromossomas multigénicos como árvores de expressão. a) Um cromossoma tri-génico com as caudas salientadas a cheio ('N' é uma função de um argumento e representa NOT; 'A' e 'O' são funções de dois argumentos e representam, respectivamente, AND e OR; 'I' é uma função de três argumentos e representa IF; os símbolos restantes são terminais). b) As sub-AEs codificadas por cada gene. c) O resultado da ligação pós-traducional com IF.

a)
131u3ab2ubab23c3ua31a333au3

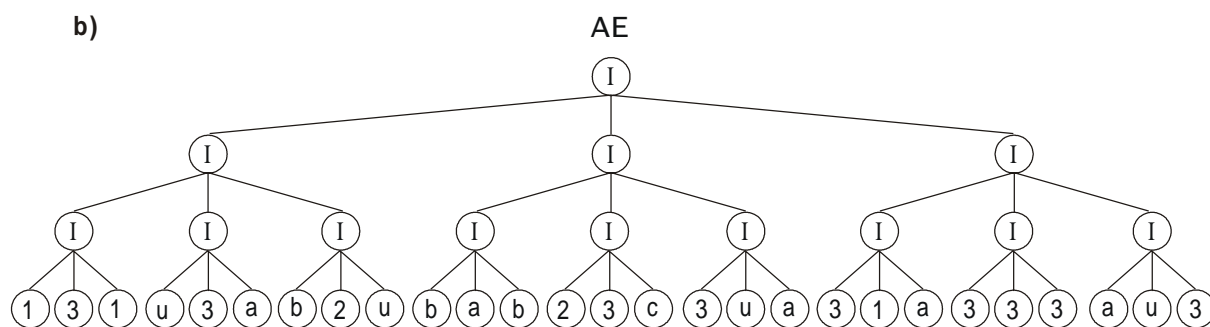


Figura 5. Expressão dos cromossomas multigénicos como árvores de expressão. a) Um cromossoma 27-génico composto por genes de um só elemento. b) O resultado da ligação pós-traducional com IF.

simples, como por exemplo funções booleanas de 2-5 argumentos, isto não é muito importante, mas para problemas mais complexos é conveniente escolher-se um patamar para a selecção. Para esses problemas pode utilizar-se a seguinte função:

$$\text{Se } i \geq \frac{3}{4}C, \text{ então } f = i; \text{ de contrário } f = 1 \quad (4.2)$$

onde i é o número de casos de aptidão correctamente determinados e C é o número total de casos de aptidão.

4.2. Selecção

Na PEG, os indivíduos são seleccionados de acordo com a aptidão, sendo o sorteamento feito por roleta [5]. Na verdade, ainda não foram testados outros métodos de selecção, tendo este sido escolhido por simular a natureza mais fielmente. É verdade que com este método se perdem frequentemente os melhores indivíduos, mas isto até pode trazer algumas vantagens e fazer as populações saltar para picos de aptidão distantes. Claro que isto merece um estudo detalhado, mas o alto desempenho da PEG indica que este algoritmo consegue andar (eu diria até voar) eficientemente pela paisagem da aptidão. De todas as formas, foi implementado um tipo de elitismo bastante simples onde o melhor indivíduo duma geração é clonado, nunca se perdendo a melhor característica.

5. Reprodução com modificação

De acordo com a aptidão e a sorte na roleta, os indivíduos são seleccionados para se reproduzirem com modificação, criando a diversidade genética necessária que permite a adaptação a longo prazo.

À excepção da replicação, que copia os genomas de todos os indivíduos seleccionados, todos os outros operadores escolhem aleatoriamente os cromossomas a ser sujeitos a modificação. No entanto, e à excepção da mutação, cada operador só modifica um cromossoma determinado uma só vez. Por exemplo, para uma taxa de transposição de 0,7, sete cromossomas diferentes dum total de 10 são escolhidos aleatoriamente.

Acrescente-se, no entanto, que na PEG um cromossoma

particular pode ser escolhido por nenhum ou por vários operadores genéticos capazes de introduzir variação na população. Esta característica também distingue a PEG da PG, onde uma entidade nunca é modificada por mais do que um operador ao mesmo tempo [6]. Desta forma, na PEG, as modificações dos vários operadores genéticos vão-se acumulando durante a reprodução, produzindo descendentes muito diferentes dos progenitores.

A secção procede com a descrição detalhada dos operadores genéticos da PEG, começando obviamente com a replicação.

5.1. Replicação

Apesar de vital, a replicação é o operador menos interessante: por si só ele não contribui com nada para a diversidade genética. (De facto, a replicação, juntamente com a selecção, só é capaz de causar deriva genética.) De acordo com a aptidão e a sorte da roleta, os cromossomas são exactamente copiados para a geração seguinte. Quanto mais apto o indivíduo, maior a probabilidade de deixar mais descendência. Assim, durante a replicação, os genomas dos indivíduos seleccionados são copiados tantas vezes quantas as vezes que foram sorteados na roleta. A roleta gira tantas vezes quantos os indivíduos da população, mantendo sempre o mesmo número de indivíduos na população.

5.2. Mutação

As mutações podem ocorrer em qualquer ponto do cromossoma. No entanto, a organização estrutural do cromossoma tem que ser conservada. Nas cabeças qualquer símbolo pode ser trocado por outro (função ou terminal); nas caudas os terminais só podem trocar com terminais. Desta forma, a organização estrutural do cromossoma é mantida e todos os indivíduos novos produzidos pela mutação são programas estruturalmente correctos. Tipicamente é utilizada uma taxa de mutação (p_m) equivalente a 2 mutações pontuais por cromossoma. Considere o cromossoma seguinte, composto por 3 genes:

012345678012345678012345678
-+-+abaaa/bb/ababb*Q*+aaaba

Suponha que uma mutação alterou o elemento na posição 0 do gene 1 para 'Q'; o elemento na posição 3 no gene 2 para 'Q'; e o elemento na posição 1 no gene 3 para 'b', obtendo:

```
012345678012345678012345678
Q+-+abaaa/bbQababb*b*+aaaba
```

Note-se que se uma função for trocada por um terminal ou vice versa, ou se uma função de um argumento for trocada por outra de dois argumentos ou vice versa, a AE é modificada drasticamente. Note-se também que a mutação no gene 2 é um exemplo de uma mutação neutra, pois ocorreu na região não-codificadora do gene.

Convém salientar que na PEG não há qualquer tipo de restrição tanto no tipo de mutação como no número de mutações por cromossoma: em todos os casos os indivíduos criados de novo são programas sintacticamente correctos.

Na natureza, uma mutação pontual na região codificadora dum gene, pode modificar ligeiramente a estrutura dum proteína, ou então pode nem sequer alterá-la, pois as mutações neutras são bastante frequentes (por exemplo, mutações em intrões, mutações que resultam no mesmo aminoácido devido à redundância do código genético, etc.). Aqui, apesar de existirem mutações neutras, uma mutação na sequência codificadora dum gene tem um efeito muito mais profundo, modificando quase sempre a AE de forma drástica.

Ao contrário do pensamento corrente em computação evolutiva, esta capacidade para transformar profundamente a AE é fundamental para a evolução. Uma análise exaustiva dos operadores da PEG está aquém do âmbito deste trabalho, no entanto, os resultados aqui apresentados mostram claramente que o desejo humanamente compreensível de não desmantelar os blocos funcionais que aparecem nas AEs e de os recombinar cuidadosamente (como é feito na PG) é conservador e funciona mal. Num sistema de genótipo/fenótipo como a PEG, o sistema é capaz de encontrar formas muito mais eficientes de criar e de usar estes blocos funcionais. As formas do sistema, no entanto, só se tornam evidentes quando emergem na árvore de expressão.

5.3. Transposição e sequências de inserção

Os elementos de transposição da PEG são fragmentos do genoma que podem ser activados e saltar para outro sítio no cromossoma. Na PEG existem três tipos de elementos de transposição: i) pequenos fragmentos com uma função ou terminal na primeira posição que saltam para a cabeça dos genes, à excepção da raiz (elementos IS, do inglês *insertion sequence elements*); ii) pequenos fragmentos com uma função na primeira posição que saltam para a raiz (elementos RIS, do inglês *root IS elements*); e iii) genes inteiros que saltam para o início dos cromossomas.

A existência de elementos IS e RIS é um vestígio do processo de desenvolvimento da PEG, pois o primeiro algoritmo de expressão genética tinha somente cromossomas uni-génicos e, naqueles sistemas, um gene com um terminal na raiz de pouco servia. Após a introdução

de cromossomas multigénicos, esta característica foi conservada pois estes operadores são importantes para a compreensão dos mecanismos de variação genética. De facto, o poder de transformação destes operadores mostra claramente que não faz sentido ser-se conservador em computação evolutiva. Por exemplo, a inserção na raiz (o operador com um efeito mais profundo) por si só é capaz de encontrar soluções criando padrões repetitivos (este é um dos padrões observados, mas outros existem certamente).

5.3.1. Transposição de elementos IS

Qualquer sequência no genoma se pode tornar num elemento IS. Por conseguinte, estes elementos são seleccionados aleatoriamente ao longo de todo o genoma. É feita uma cópia do transposão que é inserida em qualquer ponto na cabeça dum gene, à excepção da primeira posição.

Usualmente utiliza-se uma taxa de transposição IS (p_{is}) de 0,1 e um conjunto de três elementos IS de tamanhos diferentes. Este operador escolhe aleatoriamente o cromossoma, o início do elemento IS, o sítio alvo e o tamanho do transposão. Considere o cromossoma abaixo indicado, composto por dois genes:

```
012345678901234567890012345678901234567890
*--+a--+a*bbabbaabababQ**+abQbb*aabbaaabba
```

Suponha que a sequência 'bba' no gene 2 (posições 12-14) era escolhida para ser um elemento IS e que o sítio alvo era a ligação 6 no gene 1 (entre as posições 5 e 6). Então faz-se um corte na ligação 6 no gene 1 e o bloco 'bba' é copiado e inserido neste sítio, obtendo-se:

```
012345678901234567890012345678901234567890
*--+a-bba+babbaabababQ**+abQbb*aabbaaabba
```

Durante a transposição, a sequência a montante do sítio de inserção fica inalterada, enquanto que a sequência a jusante do elemento IS copiado perde, no fim da cabeça, tantos símbolos quantas as posições no elemento IS (neste caso a sequência 'a*b' foi removida). Note-se que, apesar desta inserção, a organização estrutural dos cromossomas foi mantida e portanto todos os indivíduos criados por este operador são programas sintacticamente correctos. Note-se também que a transposição pode transformar radicalmente a AE e quanto mais acima for a inserção mais profunda é a transformação.

5.3.2. Transposição para a raiz

Todos os elementos RIS começam com uma função, sendo por conseguinte escolhidos de entre as sequências das cabeças. Assim, é escolhido aleatoriamente um ponto numa cabeça e o gene é percorrido para jusante até que se encontre uma função. Esta função passa a ser a primeira posição do elemento RIS. Se não encontrar nenhuma posição, não faz nada.

Usualmente utiliza-se uma taxa de transposição para a raiz (p_{ris}) de 0,1 e um conjunto de três elementos RIS de tamanhos diferentes. Este operador escolhe aleatoriamente

o cromossoma, o gene a ser modificado, o início do elemento RIS e o seu tamanho. Considere o cromossoma seguinte, composto por dois genes:

```
012345678901234567890012345678901234567890
-ba*+--Q/abababbbaaaQ*b/bbabbaaaaaabbb
```

Suponha que a sequência '+bb' no gene 2 foi escolhida para ser um elemento RIS. Neste caso, é feita uma cópia do transposão para a raiz do gene, obtendo:

```
012345678901234567890012345678901234567890
-ba*+--Q/abababbbaaabbQ*b/bbabbaaaaaabbb
```

Durante a transposição para a raiz, a cabeça inteira é deslocada para acomodar o elemento RIS, perdendo, ao mesmo tempo, os últimos símbolos da cabeça (tantos quanto o tamanho do transposão). Tal como acontece com os elementos IS, a cauda do gene modificado e todos os restantes genes ficam inalterados. Repare-se também que todos os programas criados por este operador são sintacticamente correctos, pois a organização estrutural do cromossoma é mantida.

As modificações causadas pela transposição RIS são extremamente radicais porque é a própria raiz que é modificada. Na natureza, se um transposão for inserido no início da região codificadora, causando uma mutação de deslocação de grelha, a proteína correspondente é modificada radicalmente. À semelhança da mutação e da transposição IS, a inserção na raiz tem um poder de transformação tremendo e é ótima para criar diversidade genética. Este tipo de operadores impede as populações de ficarem estagnadas em ótimos locais, permitindo descobrir fácil e rapidamente soluções muito boas.

5.3.3. Transposição génica

Na transposição génica, um gene inteiro funciona como transposão e é transferido para o início do cromossoma. Ao contrário do que acontece com as outras formas de transposição, na transposição génica o transposão (o gene) é removido na origem. Desta forma, o tamanho do cromossoma é conservado.

O cromossoma sujeito a transposição génica é escolhido aleatoriamente e um dos seus genes (com excepção de primeiro, obviamente) é escolhido aleatoriamente para transpor. Considere o cromossoma seguinte, composto por três genes:

```
012345678012345678012345678
*a-*abbab-QQ/aaabbQ+abababb
```

Suponha que o gene 2 foi escolhido como transposão. Então, obtém-se o seguinte:

```
012345678012345678012345678
-QQ/aaabb*a-*abbabQ+abababb
```

Note-se que para aplicações numéricas em que a função de ligação é a adição ou a multiplicação, a expressão calculada pelo cromossoma não é modificada. Mas a

situação é diferente em aplicações em que a função de ligação não é comutativa, por exemplo, a função IF escolhida para ligar as sub-AEs no problema do multiplexer de 11 bits (secção 6.5.2). No entanto, o poder transformador deste operador manifesta-se quando ele é conjugado com a recombinação. Por exemplo, se dois cromossomas contendo um gene idêntico em posições diferentes recombinarem, o novo indivíduo pode ficar com um gene duplicado. Sabe-se que a duplicação de genes desempenha um papel importante em biologia e na evolução (ver [7] para uma referência geral). Curiosamente, na PEG, é comum o aparecimento de genes duplicados durante a resolução de problemas.

5.4. Recombinação

Existem três tipos de recombinação na PEG: recombinação num ponto, em dois pontos e recombinação génica. Em todos os casos, dois cromossomas progenitores são escolhidos aleatoriamente e emparelhados para trocar entre si algum material.

5.4.1. Recombinação pontual

Durante a recombinação pontual, os cromossomas cruzam-se num ponto escolhido aleatoriamente para formar dois cromossomas novos. Considere os progenitores abaixo indicados:

```
012345678012345678
-b+Qbbabb/aQbbbaab
/-a/ababb-ba-abaaa
```

Suponha que a ligação 3 no gene 1 (entre as posições 2 e 3) foi escolhido aleatoriamente para ponto de recombinação. Então, os cromossomas emparelhados são cortados por esta ligação, trocando entre si o material a jusante do ponto de recombinação, formando a descendência seguinte:

```
012345678012345678
-b+/ababb-ba-abaaa
/-aQbbabb/aQbbbaab
```

Com este tipo de recombinação, a maior parte das vezes, a descendência criada exhibe características diferentes das dos parentes. A recombinação pontual é, como os operadores apresentados acima, uma fonte muito importante de variação genética, sendo, após a mutação, um dos operadores mais escolhidos na PEG. A taxa de recombinação pontual (p_p) utilizada depende das taxas dos outros operadores. Normalmente utiliza-se uma taxa de recombinação global (a soma das taxas dos três tipos de recombinação) de 0,7.

5.4.2. Recombinação em dois pontos

Na recombinação em dois pontos, os cromossomas são emparelhados, escolhendo-se aleatoriamente dois pontos para recombinação. O material entre os pontos de recombinação é trocado posteriormente entre os dois

cromossomas, formando-se dois cromossomas novos. Considere os seguintes cromossomas progenitores:

```
0123456789001234567890
+*a*bbccac*baQ*acabab-[ 1 ]
*cbb+cccbcc++*bacbaab-[ 2 ]
```

Suponha que a ligação 7 no gene 1 (entre as posições 6 e 7) e a ligação 3 no gene 2 (entre posições 2 e 3) foram escolhidas como pontos de recombinação. Então, os cromossomas são cortados por estas ligações e o material entre os pontos de recombinação é trocado, formando a descendência seguinte:

```
0123456789001234567890
+*a*bbc c b c c++*Q*acabab-[ 3 ]
*cbb+cc c c a c *ba*ba c b a a b-[ 4 ]
```

Note-se que o primeiro gene é, em ambos os progenitores, cortado a jusante do ponto de terminação. De facto, as regiões não-codificadoras dos genes da PEG são regiões ideais por onde os cromossomas podem ser cortados para recombinação sem interferir com as GLAs. Note-se também que o segundo gene do cromossoma 1 foi igualmente cortado a jusante do ponto de terminação. Mas o gene 2 do cromossoma 2 foi cortado a montante do ponto de terminação, modificando profundamente a sub-AE. Note-se ainda que quando estes cromossomas recombinaram, a região não-codificadora do cromossoma 1 foi activada e integrada no cromossoma 3.

O poder transformador da recombinação em dois pontos é maior que o da recombinação pontual, sendo bastante útil para evoluir soluções para problemas mais complexos, especialmente se se usarem cromossomas multigénicos compostos por vários genes.

5.4.3. Recombinação génica

Durante a recombinação génica é trocado um gene inteiro. Os genes trocados são escolhidos aleatoriamente e ocupam a mesma posição nos cromossomas parentais. Considere os progenitores seguintes:

```
012345678012345678012345678
/aa-abaaa/a*bbaaab/Q*+aaaab
/-*/abbabQ+aQbabaa-Q/Qbaaba
```

Suponha que o gene 2 foi escolhido para ser trocado. Neste caso forma-se a descendência seguinte:

```
012345678012345678012345678
/aa-abaaaQ+aQbabaa/Q*+aaaab
/-*/abbab/a*bbaaab-Q/Qbaaba
```

Os indivíduos criados contêm genes vindos de ambos os parentes. Note-se que com este tipo de recombinação podem trocar-se genes semelhantes, mas na maior parte dos casos os genes trocados são muito diferentes entre si, introduzindo-se material novo na população.

É importante salientar que este operador é incapaz de criar genes novos: os indivíduos criados são re-arranjos

diferentes de genes existentes. De facto, quando a recombinação génica é utilizada como a única fonte de variação genética, só é possível resolver problemas complexos se se utilizarem populações muito grandes, garantindo, deste modo, a diversidade de genes necessária. No entanto, o poder criador da PEG baseia-se não só no baralhar de genes ou blocos mas também na criação permanente de material genético.

6. A programação de expressão genética na resolução de problemas: seis exemplos

O conjunto de problemas escolhido para ilustrar o funcionamento deste algoritmo novo é extremamente variado, incluindo não só problemas de áreas muito diversas (regressão simbólica, planeamento, síntese lógica e autómatos celulares) mas também problemas de grande complexidade (regras para o problema da classificação da densidade em autómatos celulares).

É costume compararem-se diferentes algoritmos evolutivos usando problemas com um grau de complexidade semelhante aos problemas da regressão simbólica, da indução de sequências, do empilhamento de blocos ou do multiplexer de 11 bits [8]. As comparações são normalmente feitas em termos da probabilidade de sucesso e em termos do número médio de cálculos da função de aptidão necessário para encontrar uma solução correcta. Apesar das diferenças entre a PEG e a PG, o desempenho destas duas técnicas pode ser facilmente comparado já que, graças à representação em árvore, problemas semelhantes podem ser implementados de forma idêntica.

As comparações são feitas em cinco dos problemas e, sempre que possível, o desempenho da PEG e da PG é avaliado em termos do número médio de cálculos da função de aptidão (F_z) necessário para encontrar, com uma probabilidade z , uma solução perfeita. O F_z é calculado pela equação:

$$F_z = G \cdot P \cdot C \cdot R_z \quad (6.1)$$

onde G é o número de gerações; P o tamanho da população; C o número de casos de aptidão; e R_z o número de corridas independentes necessário para se encontrar um solução perfeita ao fim de G gerações com $z = 0,99$ [6]. O R_z é calculado pela fórmula:

$$R_z = \frac{\log(1-z)}{\log(1-P_s)}, \text{ e } P_s \neq 1 \quad (6.2)$$

onde P_s é a probabilidade de sucesso; se $P_s = 1$, então $R_z = 1$.

6.1. Regressão simbólica

O objectivo deste problema consiste na descoberta duma expressão que satisfaça um conjunto de casos de aptidão. Suponha que nos davam uma amostra de valores numéricos da função

$$y = a^4 + a^3 + a^2 + a \quad (6.3)$$

ao longo de 10 pontos escolhidos e que pretendíamos descobrir uma função que satisfizesse esses valores dentro de 0,01 do valor correcto.

Antes de tudo, temos que escolher o conjunto de funções F e o conjunto de terminais T . Neste caso o $F = \{+, -, *, /\}$ e o $T = \{a\}$. Depois escolhe-se a organização estrutural dos cromossomas, designadamente o tamanho da cabeça e o número de genes. É aconselhável começar com cromossomas pequenos e de um só gene, aumentando gradualmente a cabeça. Na Figura 6 mostra-se uma análise deste tipo para este problema. Utilizou-se uma p_m equivalente a 2 mutações pontuais por cromossoma e uma $p_r = 0,7$ em todas as experiências para simplificar a análise. O conjunto de casos de aptidão está indicado na Tabela 1 e a aptidão foi calculada pela equação 4.1, sendo M um erro absoluto de 100. Se o E for igual ou menor que 0,01 (a precisão escolhida para este problema), então $E = 0$; assim para $C = 10, f_{max} = 1000$.

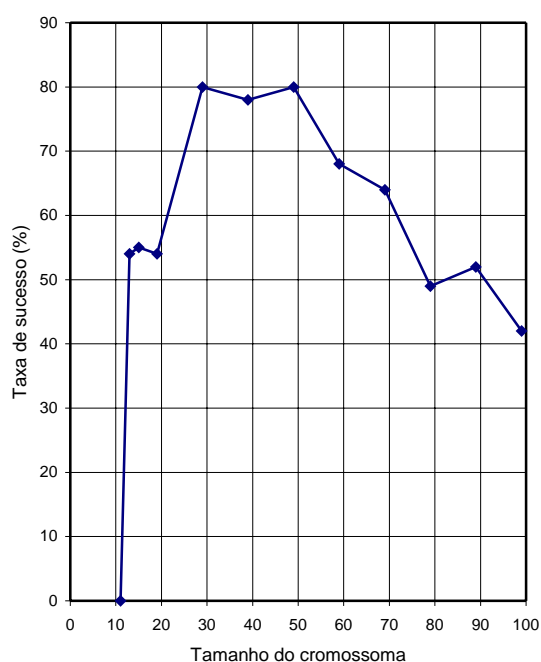


Figura 6. Variação da taxa de sucesso (P_s) com o tamanho do cromossoma. Para esta análise $G = 50$ e $P = 30$. A P_s foi determinada para 100 corridas idênticas.

Note-se que a PEG pode ser útil para se procurar a solução mais parcimoniosa para um problema. Por exemplo, o cromossoma

```
0123456789012
*++/**aaaaaa
```

com $h = 6$ codifica para a AE:

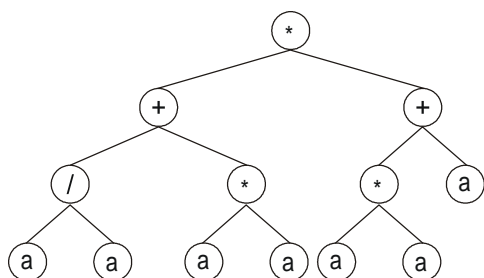


Tabela 1. Conjunto de casos de aptidão usados no problema da regressão simbólica.

a	f(a)
2,81	952,425
6	1554
7,043	2866,55
8	4680
10	11110
11,38	18386
12	22620
14	41370
15	54240
20	168420

que é equivalente à função alvo. Note-se também que a PEG é capaz de evoluir eficientemente soluções utilizando sub-AEs grandes e complexas. Como se mostra na Figura 6, para cada problema existe um tamanho de cromossoma que é ótimo para evoluir eficientemente soluções. Convém salientar que os genomas mais compactos não são os mais eficientes. De onde se conclui que uma certa redundância é fundamental para se evoluírem eficientemente bons programas.

A relação entre a taxa de sucesso e P também foi analisada (Figura 7). Neste caso escolheu-se um $h = 24$. Estes resultados mostram a supremacia duma representação genótipo/fenótipo, já que este sistema unigénico, mais próximo da PG, supera consideravelmente esta

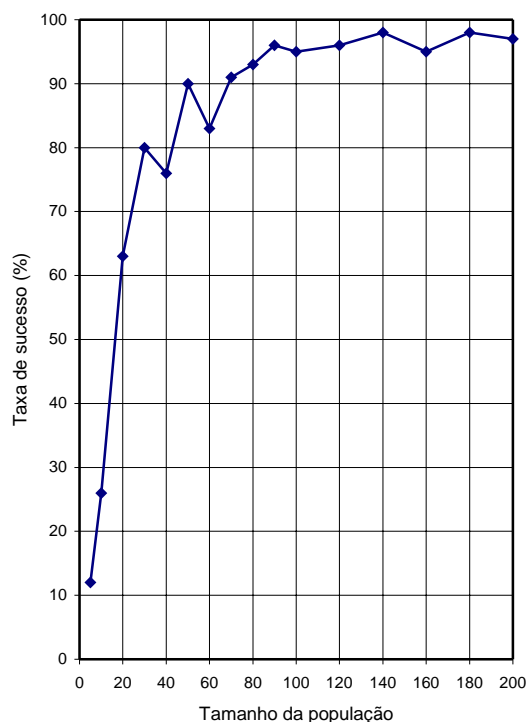


Figura 7. Variação da taxa de sucesso (P_s) com o tamanho da população. Para esta análise $G = 50$ e um valor médio de 49 foi utilizado como tamanho do cromossoma ($h = 24$). A P_s foi determinada para 100 corridas idênticas.

técnica [6]. No entanto, a PEG é muito mais complexa que um sistema uni-génico, pois os cromossomas da PEG codificam para mais do que um gene.

Suponha que após a análise apresentada na Figura 6, ainda não se tinha encontrado uma solução correcta. Então poderíamos aumentar o número de genes e escolher uma função para os ligar. Por exemplo, podíamos escolher um $h = 6$ e aumentar o número de genes gradualmente. A Figura 8 mostra a variação da taxa de sucesso com o número de genes. Nesta análise a p_m foi equivalente a duas mutações pontuais por cromossoma, $p_{lr} = 0,2$, $p_{2r} = 0,5$, $p_{gr} = 0,1$, $p_{is} = 0,1$, $p_{ris} = 0,1$, $p_{gt} = 0,1$ e foram utilizados três transposões (tanto elementos IS como RIS) de 1, 2 e 3 de tamanho. Note-se que a PEG lida perfeitamente com um excesso de genes: a taxa de sucesso para o sistema de 10 genes é ainda bastante elevada (47%).

Na Figura 9 apresenta-se outra relação muito importante: a variação da taxa de sucesso com o tempo evolutivo (G). Ao contrário do que acontece na PG, em que a norma são 51 gerações pois nada de útil se descobre depois disso [4], na PEG, as populações vão-se adaptando e evoluindo indefinidamente porque está sempre a ser introduzido material novo no reservatório genético.

Por último, suponha que o sistema multigénico com as sub-AEs ligadas pela adição também não tinha conseguido evoluir uma solução satisfatória para o problema. Então podíamos escolher outra função de ligação, por exemplo, a multiplicação. Este processo continuava até que se encontrasse uma solução perfeita.

Como já referi, os cromossomas da PEG podem ser facilmente modificados de forma a codificar também a função de ligação. Neste caso, a função de ligação ideal para cada problema seria encontrada durante o processo de adaptação.



Figura 8. Variação da taxa de sucesso (P_s) com o número de genes. Para esta análise $G = 50$, $P = 30$ e $h = 6$ (um gene com 13 de tamanho). A P_s foi determinada para 100 corridas idênticas.

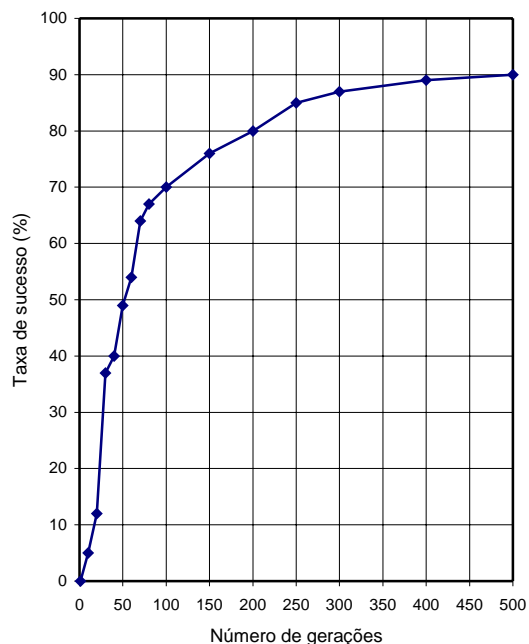


Figura 9. Variação da taxa de sucesso (P_s) com o número de gerações. Para esta análise $P = 30$, tendo sido utilizado um cromossoma com 79 de tamanho (um cromossoma uni-génico com $h = 39$). A P_s foi determinada para 100 corridas idênticas.

Considere, por exemplo, um sistema multigénico composto por três genes ligados pela adição. Como se mostra na Figura 8, a taxa de sucesso tem neste caso o valor máximo de 100%. Na Figura 10 mostra-se a progressão da aptidão média da população e a aptidão do melhor indivíduo de cada geração para a corrida 0 da experiência sumariada na Tabela 2, coluna 1. Nesta corrida, foi encontrada na geração 11 uma solução perfeita (as sub-AEs estão ligadas pela adição):

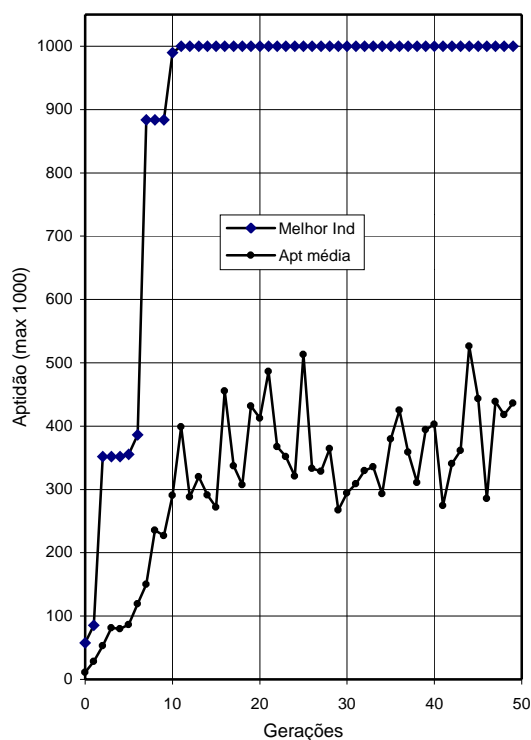


Figura 10. Progressão da aptidão média da população e aptidão do melhor indivíduo da geração para a corrida 0 da experiência sumariada na Tabela 2, coluna 1 (regressão simbólica).

Tabela 2.

Parâmetros usados nos problemas da regressão simbólica (RS), indução de seqüências (IS), empilhamento de blocos (EB) e 11-multiplexer (11-M).

	RS	IS	EB	11-M
Número de corridas	100	100	100	100
Número de gerações	50	100	100	400
Tamanho da população	30	50	30	250
Número de casos de aptidão	10	10	10	160
Tamanho da cabeça	6	6	4	1
Número de genes	3	7	3	27
Tamanho do cromossoma	39	91	27	27
Taxa de mutação	0,051	0,022	0,074	0,074
Taxa de recombinação pontual	0,2	0,7	0,1	0,7
Taxa de recombinação em 2 pontos	0,5	0,1	--	--
Taxa de recombinação génica	0,1	0,1	0,7	--
Taxa de transposição IS	0,1	0,1	0,1	--
Tamanho dos elementos IS	1,2,3	1,2,3	1	--
Taxa de transposição RIS	0,1	0,1	0,1	--
Tamanho dos elementos RIS	1,2,3	1,2,3	1	--
Taxa de transposição génica	0,1	0,1	--	--
Margem de selecção	100	100	--	--
Erro absoluto	0,01	0,0	--	--
Taxa de sucesso	1	0,79	0,7	0,57

012345678901201234567890120123456789012
 -*a+aaaaaa+a*aaaaaa*+-a/aaaaaaa

que a PEG supera a PG em 374 vezes, portanto, mais de duas ordens de grandeza.

Matematicamente ela corresponde à função alvo (a contribuição de cada sub-AE está indicada entre parêntesis):

$$y = (a^4) + (a^3 + a^2 + a) + (0) = a^4 + a^3 + a^2 + a$$

A análise detalhada deste programa, mostra que algumas das acções são redundantes para o problema em questão, como por exemplo, a adição de zero ou a multiplicação por um. No entanto, a existência destes blocos desnecessários ou mesmo pseudogenes como o gene 3, é importante para a evolução de indivíduos mais aptos (comparar, nas Figuras 6 e 8, a taxa de sucesso dum sistema compacto com um único gene e um $h = 6$ com outros sistemas menos compactos).

A comparação dos valores de F_z obtidos pela PEG e pela PG [6] para este problema (Tabela 3, coluna 1), mostra

6.2. Indução de seqüências

O problema da indução de seqüências é um caso particular de regressão simbólica em que o domínio da variável independente consiste nos números naturais. No entanto, a seqüência escolhida para este problema é mais complicada que a expressão usada na regressão simbólica, devido à existência de coeficientes diferentes.

Na seqüência 1, 15, 129, 547, 1593, 3711, 7465, 13539, 22737, 35983, 54321, ..., o termo $n(N)$ é dado pela expressão:

$$N = 5a_n^4 + 4a_n^3 + 3a_n^2 + 2a_n + 1 \quad (6.4)$$

onde a_n consiste nos números naturais 0, 1, 2, 3, ...

Para este problema $F = \{+, -, *, /\}$ e $T = \{a\}$. O conjunto dos casos de aptidão C está indicado na Tabela 4. A aptidão foi calculada pela equação 4.1, sendo $M = 100$. Assim, se

Tabela 3.

Comparação da PEG com a PG nos problemas da regressão simbólica, indução de seqüências e empilhamento de blocos.

	Regressão simbólica		Indução de seqüências		Empilhamento de blocos	
	PEG	PG [6]	PEG	PG [6]	PEG	PG [8]
G	50	51	100	51	100	51
P	30	500	50	500	30	500
C	10	20	10	20	10	167
Ps	1	0,35	0,79	0,15	0,7	0,767
Rz	1	11	3	29	4	4
Fz	15.000	5.610.000	150.000	14.790.000	120.000	17.034.000

Tabela 4.
Conjunto de casos de aptidão para o problema da indução de seqüências.

a	N
1	15
2	129
3	547
4	1593
5	3711
6	7465
7	13539
8	22737
9	35983
10	54321

os 10 casos de aptidão forem calculados exactamente, a $f_{max} = 1000$.

A Figura 11 mostra a progressão da aptidão média da população e a aptidão do melhor indivíduo para a corrida 0 da experiência sumariada na Tabela 2, coluna 2. Nesta corrida foi encontrada uma solução perfeita na geração 24 (as sub-AEs estão ligadas pela adição):

```
0123456789012012345678901201234567890120123456789012...
**+--+-----aaaaaaa*+/-+a*aaaaaaa*+*+*+aaaaaaa*-*+*+aaaaaaa...
...012345678901201234567890120123456789012
...*a/+a-aaaaaaa+--/*aaaaaaa**+a*+aaaaaaa
```

Matematicamente ela corresponde à seqüência alvo (a contribuição de cada sub-AE está indicada entre parêntesis):

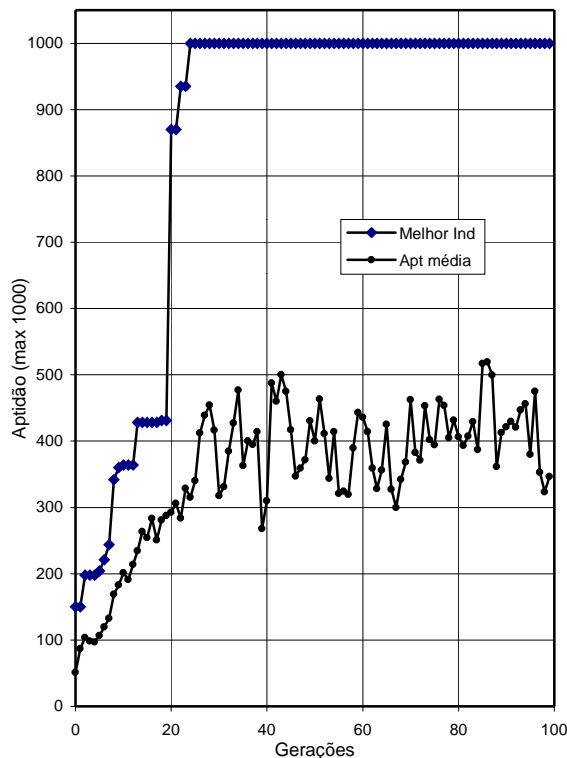


Figura 11. Progressão da aptidão média da população e aptidão do melhor indivíduo da geração para a corrida 0 da experiência sumariada na Tabela 2, coluna 2 (indução de seqüências).

$$y = (0) + (3a^2) + (2a^4 + 4a^3) + (0) + (a) + (1 + a) + (3a^4)$$

Como se mostra na coluna 2 da Tabela 2, a probabilidade de sucesso para este problema é de 0,79. A comparação dos valores de F_z obtidos para a PEG e para a PG [6] (Tabela 3, coluna 2) mostra que a PEG supera a PG em 98,6 vezes. Deve salientar-se, no entanto, que a PEG não só é capaz de resolver este tipo de problemas de forma muito mais eficiente que a PG com também o faz sem usar a constante efêmera aleatória R , que consiste num conjunto de números escolhidos que limita consideravelmente a utilidade da técnica. Por exemplo, para este problema o R escolhido cobria os números inteiros 0, 1, 2 e 3 [6]. As vantagens da PEG são óbvias porque, primeiro, em aplicações reais nunca se sabe de antemão que tipo de constantes são necessárias e, segundo, o número de terminais é muito menor, reduzindo-se a complexidade do problema.

6.3. Empilhamento de blocos

No problema do empilhamento de blocos, o objectivo consiste em descobrir um plano que aceita qualquer configuração de blocos distribuídos aleatoriamente entre a mesa e a pilha e os coloca na pilha na ordem correcta. Neste caso, os blocos são as letras da palavra “universal”. (Apesar de ter sido usada a palavra “universal” como ilustração, na minha versão do problema os blocos podem ser idênticos, como por exemplo na palavra “individual”.)

As funções e os terminais utilizados neste problema consistem num conjunto de acções e sensores, sendo $F = \{C, R, N, A\}$ (carregar para a pilha, remover da pilha, NOT, e fazer até verdadeiro, respectivamente), onde as primeiras três são funções de um argumento e ‘A’ é uma função de dois argumentos. Nesta versão, os ciclos ‘A’ são executados logo ao princípio, são processados numa ordem particular (de baixo para cima e da esquerda para a direita), o argumento da acção é executado pelo menos uma vez independentemente do estado do predicado e cada *fazer até verdadeiro* é executado uma só vez, terminando ao fim de 20 iterações. O conjunto de terminais consiste em três sensores {u, t, p} (último bloco na pilha, bloco do topo correcto e próximo bloco necessário, respectivamente). Nesta versão, o ‘t’ refere-se somente ao bloco no topo da pilha e se ele está correcto ou não; se a pilha estiver vazia ou tiver alguns blocos, todos correctamente empilhados, o sensor devolve *Verdadeiro*, de contrário devolve *Falso*; e o ‘p’ refere-se obviamente ao bloco necessário logo a seguir ao ‘t’.

Para este problema foi utilizado um sistema multigénico composto por três genes. A ligação das sub-AEs consiste na execução sequencial de cada sub-AE ou sub-plano. Por exemplo, se a primeira sub-AE esvaziar todas as pilhas, a segunda pode proceder com o seu preenchimento, etc. A aptidão foi determinada contra 10 casos de aptidão (configurações iniciais de blocos). Em cada geração é criada uma pilha vazia, mais nove configurações iniciais aleatórias contendo entre uma a nove letras na pilha. A pilha vazia foi utilizada para impedir a terminação prematura das corridas (ver abaixo). No entanto, a PEG é capaz de resolver este problema eficientemente utilizando

exclusivamente 10 configurações iniciais aleatórias (resultados não apresentados).

A função de aptidão utilizada foi a seguinte: a cada pilha vazia foi atribuído um ponto de aptidão; a cada pilha parcial e correctamente empilhada (isto é, com 1 a 8 letras no caso da palavra “universal”) foram atribuídos dois pontos de aptidão; e por cada pilha completa e correctamente preenchida foram atribuídos três pontos de aptidão. Assim, a aptidão máxima é igual a 30. A ideia era fazer a população de programas evoluir hierarquicamente soluções no sentido duma maior complexidade até chegar a um plano perfeito. E de facto, os primeiros planos úteis descobertos normalmente esvaziam todas as pilhas, depois surgem outros programas que já são capazes de preencher parcialmente as pilhas previamente esvaziadas e, por último, é descoberto um plano perfeito que preenche as pilhas correcta e completamente (ver a Figura 12).

A Figura 12 mostra a progressão da aptidão média da população e a aptidão do melhor indivíduo de cada geração da corrida 2 da experiência sumariada na Tabela 2, coluna 3. Nesta corrida foi descoberto um plano perfeito na geração 50:

```
012345678012345678012345678
ARCuptppuApNCptuutNtpRppptp
```

Note-se que o primeiro sub-plano esvazia todas as pilhas e coloca uma letra correctamente no sítio; o segundo sub-plano procede com o empilhamento correcto das restantes letras; e o último plano não faz nada. Convém salientar que todos os planos descobertos com aptidão máxima são, de facto, planos perfeitos e universais: em cada geração eles são testados contra nove configurações

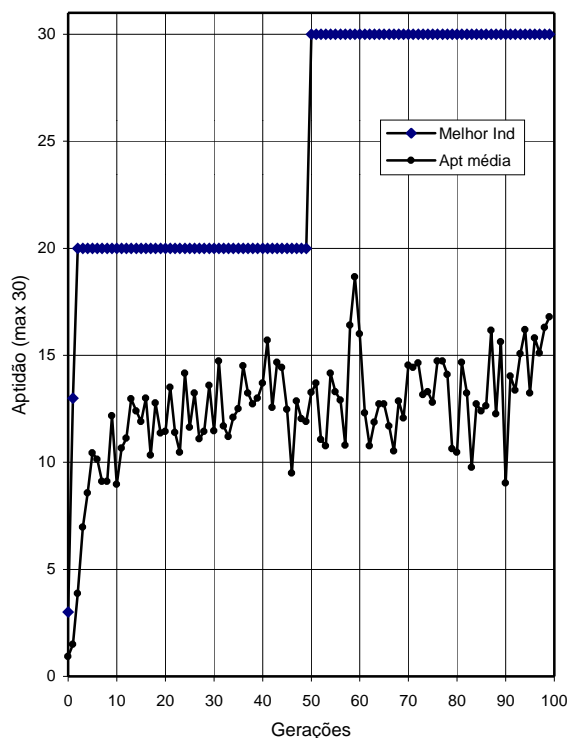


Figura 12. Progressão da aptidão média da população e aptidão do melhor indivíduo da geração para a corrida 2 da experiência sumariada na Tabela 2, coluna 3 (empilhamento de blocos).

iniciais completamente aleatórias, o que é mais do que suficiente para permitir uma generalização do problema (como se pode ver na Figura 12, uma vez alcançada, a aptidão máxima mantém-se). De facto, com a função de aptidão utilizada e o tipo de casos de aptidão usados, todos os planos com aptidão máxima são planos universais.

Como indicado na terceira coluna da Tabela 2, a probabilidade de sucesso para este problema é 0,70. A comparação dos valores de F_z obtidos pela PEG e pela PG para este problema (Tabela 3, coluna 3) mostra que a PEG supera a PG em 142 vezes, portanto mais de duas ordens de grandeza. Convém salientar que a PG usa 167 casos de aptidão, extremamente bem arquitectados para cobrir as várias classes de possíveis configurações iniciais, enquanto que a PEG utiliza 9 (dum total de 10) configurações totalmente aleatórias. De facto, nas aplicações reais nem sempre é possível prever o tipo de casos que farão o sistema descobrir uma solução. Por isso, os algoritmos capazes de generalizar facilmente mediante casos de aptidão aleatórios são mais vantajosos.

6.4. Evolução de regras para o problema da classificação da densidade em autómatos celulares

Os autómatos celulares (ACs) têm sido amplamente estudados, pois tratam-se de versões idealizadas de sistemas computacionais maciçamente paralelos e descentralizados, capazes de comportamentos emergentes. Estes comportamentos complexos resultam da execução simultânea de regras simples em múltiplos pontos locais. Na tarefa da classificação da densidade, uma regra simples envolvendo uma vizinhança restrita e operando simultaneamente em todas as células dum AC de uma dimensão, tem que ser capaz de fazer o AC convergir para um estado de tudo 1's se a configuração inicial (CI) tiver uma maior densidade de 1's, ou para um estado de tudo 0's se a CI tiver um maior número de 0's.

A capacidade dos AGs para evoluírem regras para o problema da classificação da densidade em ACs foi intensivamente investigada [9, 10, 11, 12], mas as regras descobertas pelos AGs têm rendimentos muito baixos e estão longe de se aproximar da exactidão da regra GKL, uma regra desenhada à mão. A PG também foi utilizada para evoluir regras para a tarefa da densidade [13], tendo conseguido descobrir uma regra que supera a regra GKL e outras regras desenhadas à mão.

Nesta secção, mostra-se como a PEG foi aplicada com sucesso a este problema difícil. As regras descobertas pela PEG têm níveis de precisão de 82,513% e 82,550%, portanto são superiores a todas as regras escritas à mão e à regra descoberta pela PG.

6.4.1. A tarefa da classificação da densidade

O AC mais simples é uma cadeia circular de N células de estado binário, em que cada célula está ligada a r vizinhas de ambos os lados. O estado de cada célula é actualizado por uma regra particular. A regra é aplicada simultaneamente em todas as células e o processo é iterado por t passos.

Na versão mais estudada deste problema, $N = 149$ e a vizinhança é 7 (a célula central é representada por 'u'; as

$r = 3$ células para a esquerda são representadas por ‘c’, ‘b’ e ‘a’; as $r = 3$ células para a direita são representadas por ‘1’, ‘2’ e ‘3’). Portanto, a dimensão do universo de regras onde procurar uma solução para este problema, é o número astronómico de 2^{128} . A figura 13 mostra um AC com um $N = 11$ e o estado actualizado para o autómato celular ‘u’ depois da aplicação duma regra de transição determinada.

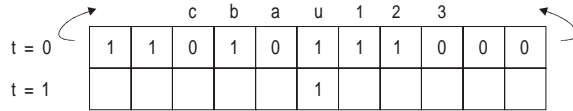


Figura 13. Um autómato celular unidimensional, binário, com $r = 3$ e $N = 11$. As setas representam as condições periódicas de ligação. O estado actualizado da célula central está indicado. Os símbolos utilizados para representar a vizinhança também estão indicados.

A tarefa da classificação da densidade consiste em determinar correctamente se as CIs contêm uma maioria de 1's ou uma maioria de 0's, fazendo o sistema convergir, respectivamente, para um estado de tudo 1's (células pretas ou “ligadas” no diagrama espaço-temporal) e para um estado de tudo 0's (células brancas ou “desligadas”). Sendo a densidade duma CI uma função de N argumentos, as acções de células locais com informação e comunicação limitadas têm que estar coordenadas umas com as outras para que possam classificar correctamente as CIs. De facto, a descoberta de regras com um alto desempenho é um desafio, tendo sido utilizados vários algoritmos para descobrir regras cada vez melhores [10, 12, 13, 14]. As melhores regras existentes têm desempenhos de 86,0% (coevolução 2) e 85,1% (coevolução 1) e foram descobertas utilizando uma abordagem coevolutiva entre regras evoluídas pelos AGs e classes de CIs [14]. No entanto, o objectivo desta secção é comparar o desempenho da PEG com os outros algoritmos genéticos (AGs e PG) quando aplicados a um problema difícil. E de facto, a PEG conseguiu evoluir regras melhores que a regra descoberta pela PG (regra PG), usando recursos computacionais que são mais de quatro ordens de grandeza inferiores aos utilizados pela PG.

6.4.2. Duas regras descobertas pela PEG

Numa experiência $F = \{A, O, N, I\}$ ('A' representa a função booleana AND, 'O' representa OR, 'N' representa NOT e 'I' representa IF) e o $T = \{c, b, a, u, 1, 2, 3\}$. Os parâmetros utilizados por corrida estão apresentados na Tabela 5, coluna 1. A aptidão foi determinada contra 25 CIs imparciais (casos de aptidão). Neste caso, a aptidão é uma função do número de CIs para as quais o sistema estabiliza correctamente numa configuração de tudo 0's ou 1's após $2 \times N$ passos, tendo sido desenhada com o intuito de privilegiar os indivíduos capazes de classificar correctamente CIs tanto com uma maioria de 1's como com uma maioria de 0's. Assim, se o sistema convergir indiscriminadamente, em todos os casos, para uma configuração de tudo 1's ou 0's, é atribuído somente um ponto de aptidão; se, nalguns casos, o sistema convergir correctamente ou para uma configuração de 0's ou para uma configuração de 1's, então $f = 2$; além do

Tabela 5.

Parâmetros usados no problema da classificação da densidade.

	PEG ₁	PEG ₂
Número de gerações	50	50
Tamanho da população	30	50
Número de CIs	25	100
Tamanho da cabeça	17	4
Número de genes	1	3
Tamanho do cromossoma	52	39
Taxa de mutação	0,038	0,051
Taxa de recombinação pontual	0,5	0,7
Taxa de transposição IS	0,2	--
Tamanho dos elementos IS	1,2,3	--
Taxa de transposição RIS	0,1	--
Tamanho dos elementos RIS	1,2,3	--

mais, se as regras convergirem para uma configuração alternada de tudo 0's e tudo 1's são eliminadas, pois estas regras são facilmente descobertas e invadem as populações, impedindo a descoberta de soluções boas; e, por último, se um programa particular classificar correctamente CIs tanto com maiorias de 1's como com maiorias de 0's, é atribuído um bónus igual ao número de CIs, C , sendo neste caso $f = i + C$. Por exemplo, se um programa classificasse correctamente duas CIs, uma com uma maioria de 1's e outra com uma maioria de 0's, receberia $2 + 25 = 27$ pontos de aptidão.

Nesta experiência foram feitas sete corridas. Na geração 27 da corrida 5, foi descoberto um programa com uma aptidão de 44:

```
0123456789012345678901234567890123456789012345678901
OAI1AucONObAbtANIb1u23u3a1.2aacb3bc21aa2baabc3bccuc13
```

Note-se que a GLA termina na posição 28. Este programa tem um desempenho de 0,82513 testado contra 100.000 CIs imparciais num mosaico de 149×298 , sendo portanto melhor que os 0,824 da regra PG testada num mosaico de 149×320 [14, 13]. A tabela de verdade desta regra (regra PEG₁) está representada na Tabela 6. Na Figura 14 mostram-se três diagramas espaço-temporais obtidos com esta regra.

Comparativamente, a PG utilizou populações de 51.200 indivíduos e 1000 CIs por 51 gerações [13], portanto foram feitos $51.200 \times 1.000 \times 51 = 2.611.200.000$ cálculos de aptidão, enquanto que a PEG fez somente $30 \times 25 \times 50 = 37.500$ cálculos de aptidão. Portanto, a PEG supera a PG em mais de quatro ordens de grandeza (69.632 vezes). E como John Holland diz no seu livro *Emergence: from chaos to order*, “In the sciences, three orders of magnitude is enough to call for a new science.” (Nas ciências, três ordens de grandeza é suficiente para a criação duma nova ciência.) De facto, na natureza, o aparecimento duma entidade única, consistindo num genótipo e num fenótipo levou ao aparecimento da vida.

Uma regra ligeiramente superior à PEG₁ foi descoberta noutra experiência, tendo um desempenho de 0,8255. Como a anterior, o seu desempenho foi determinado usando 100.000 CIs imparciais num mosaico de 149×298 . Para esta experiência $F = \{I, M\}$ ('I' representa IF e 'M' representa a

Tabela 6.

Descrição de duas regras novas (PEG₁ e PEG₂) para o problema da classificação da densidade descobertas pela PEG. A regra PG também se indica. Os bits de saída estão assinalados por ordem lexicográfica, começando em 0000000 e acabando em 1111111.

PEG ₁	00010001	00000000	01010101	00000000	00010001	00001111	01010101	00001111
	00010001	11111111	01010101	11111111	00010001	11111111	01010101	11111111
PEG ₂	00000000	01010101	00000000	01110111	00000000	01010101	00000000	01110111
	00001111	01010101	00001111	01110111	11111111	01010101	11111111	01110111
Regra PG	00000101	00000000	01010101	00000101	00000101	00000000	01010101	00000101
	01010101	11111111	01010101	11111111	01010101	11111111	01010101	11111111

função maioria com três argumentos), sendo o T obviamente o mesmo. Neste caso, foram usadas 100 CIs imparciais e cromossomas tri-génicos com as sub-AEs ligadas pela função booleana IF. Os parâmetros utilizados por corrida estão indicados na segunda coluna da Tabela 5.

A função de aptidão foi ligeiramente modificada pela introdução dum sistema hierárquico em que os indivíduos capazes de classificar correctamente entre 2 e 3/4 das CIs recebem um bónus igual a C ; se classificarem correctamente entre 3/4 e 17/20 das CIs recebem 2 bónus; e se classificarem correctamente mais de 17/20 das CIs recebem 3 bónus. Além do mais, nesta experiência, os indivíduos capazes de classificar correctamente somente um tipo de situação, mas não de forma indiscriminada, são diferenciados e têm uma aptidão de i .

Na geração 43 da corrida 10 foi descoberto um indivíduo com aptidão 393:

012345678901201234567890120123456789012
MIuua1113b21cMIM3au3b2233bM1MIacc1cblaa

A sua tabela de verdade está apresentada na Tabela 6. Na Figura 15 mostram-se três diagramas espaço-temporais obtidos com esta regra (PEG₂). Mais uma vez, a comparação com a PG mostra que a PEG supera a PG em 10.444 vezes.

6.5. Síntese lógica

A regra PG e o multiplexer de 11 bits são, respectivamente, funções booleanas de sete e de 11 actividades. Enquanto que a solução para o 11-multiplexer é uma função booleana bem conhecida, a solução da regra PG é praticamente desconhecida, pois o programa evoluído pela PG [13] é tão complicado que não é possível perceber o que ele faz.

Nesta secção mostra-se que a PEG pode ser eficientemente aplicada para evoluir expressões booleanas de vários argumentos. Além do mais, a organização estrutural dos cromossomas utilizada para evoluir soluções para o multiplexer de 11 bits, é um exemplo duma organização muito simples que pode ser utilizada eficientemente para resolver certos problemas. Por exemplo, esta organização (genes de um elemento ligados por IF) foi aplicada com sucesso na evolução de regras para o problema da classificação da densidade em ACs, tendo sido descobertas regras melhores que a regra GKL (resultados não apresentados).

6.5.1. O problema da regra PG

Para este problema $F = \{N, A, O, X, D, R, I, M\}$ (representando, respectivamente, NOT, AND, OR, XOR,

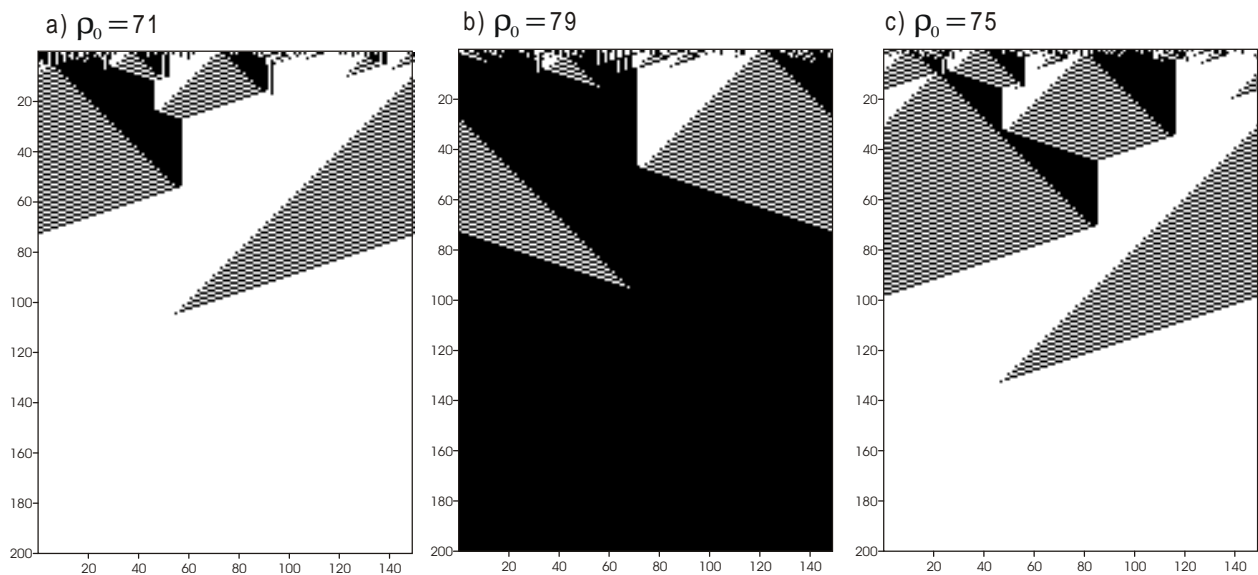


Figura 14. Três diagramas espaço-temporais da evolução dos estados dos ACs obtidos com a regra PEG₁. O número de 1's (ρ_0) na CI está indicado sobre cada diagrama. Em a) e b) os ACs convergiram correctamente para um padrão uniforme; em c) convergiram incorrectamente para um padrão uniforme.

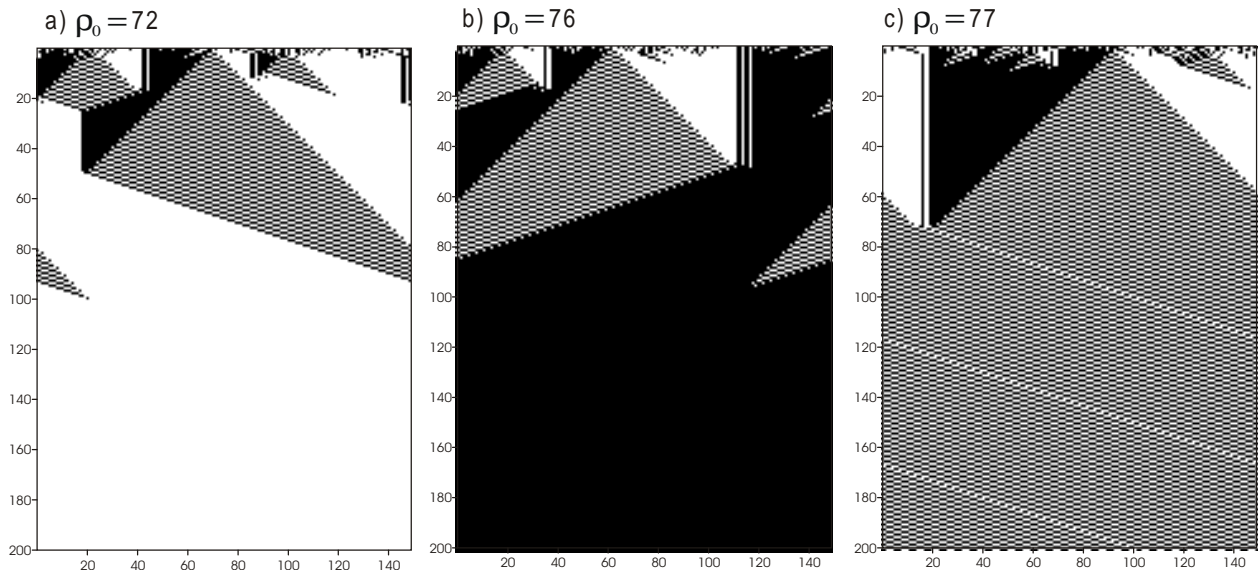


Figura 15. Três diagramas espaço-temporais da evolução dos estados dos ACs obtidos com a regra PEG₂. O número de 1's (ρ_0) na CI está indicado sobre cada diagrama. Em **a)** e **b)** os ACs convergiram, respectivamente, para uma configuração correcta de tudo 0's e tudo 1's; em **c)** não conseguiram convergir para um padrão uniforme.

NAND, NOR, IF e Maioria, sendo a primeira uma função de um argumento, a segunda até à quinta funções de dois argumentos e as duas últimas funções de três argumentos) e $T = \{c, b, a, u, 1, 2, 3\}$. A tabela de verdade ($2^7=128$ casos de aptidão) está representada na Tabela 6 e a aptidão foi calculada pela equação 4.2. Assim, $f_{max} = 128$.

Numa experiência foram descobertas três soluções:

```
MA300AMOauOMRa1cc3cubcc2cu11ba2aacb331ua122uu1
X3RRMIMODIAIAAI3cauuc313bub2uc33ca12u233c22bcb
MMOIOcXOMa3AXAu3cc112ucbb3331uac3cu3auubuu2ab1
```

A análise cuidada destes programas mostra que a regra PG é, como a regra GKL, uma função de cinco argumentos: c, a, u, 1 e 3.

6.5.2. O problema do multiplexer de 11 bits

A tarefa do 11-multiplexer consiste na descodificação dum endereço binário de 3 bits (000, 001, 010, 011, 101, 110, 111) e na devolução do valor do registo correspondente ($d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7$). Assim, o 11-multiplexer é uma função de 11 argumentos: três (de a_0 a a_2) determinam o endereço e oito (de d_0 a d_7) determinam a resposta. Como os cromossomas da PEG são constituídos por símbolos de um só caractere, o $T = \{a, b, c, 1, 2, 3, 4, 5, 6, 7, 8\}$, correspondendo, respectivamente a $\{a_0, a_1, a_2, d_0, d_1, d_2, d_3, d_4, d_5, d_6, d_7\}$.

Existem $2^{11} = 2048$ combinações possíveis dos 11 argumentos da função booleana do 11-multiplexer. Para este problema, em cada geração, é utilizada uma amostra aleatória das 2048 combinações como casos de aptidão. Os casos de aptidão foram agrupados por endereço, tendo sido utilizado por cada endereço um sub-conjunto de 20 combinações aleatórias. Portanto, o ambiente de selecção muda todas as gerações e consiste num total de 160 casos de aptidão aleatórios. Neste caso, a aptidão de um programa é o número de casos de aptidão para os quais o valor booleano devolvido é o correcto, mais um bónus de 180 pontos por cada sub-conjunto de casos de aptidão

resolvido correctamente na sua totalidade. Assim, é atribuído um total de 200 pontos de aptidão por cada endereço correctamente descodificado, sendo a aptidão máxima igual a 1600. A ideia era fazer o algoritmo descodificar um endereço de cada vez. E, de facto, os indivíduos aprendem primeiro a descodificar primeiro um endereço, depois outro, até ao último (ver Figura 16).

Para resolver este problema foram utilizados cromossomas multigénicos compostos por 27 genes, consistindo cada gene num terminal. Assim, não foram utilizadas funções nos cromossomas, mas as sub-AEs foram ligadas pós-traducionalmente por IF.

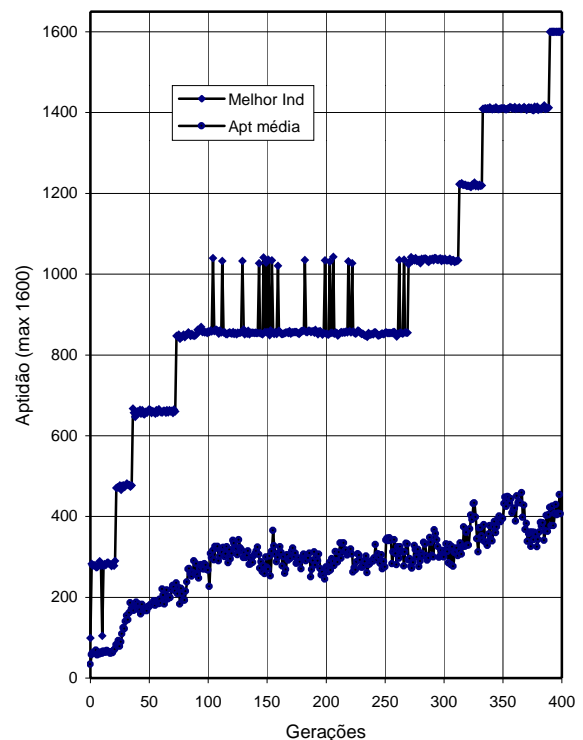


Figura 16. Progressão da aptidão média da população e aptidão do melhor indivíduo da geração para a corrida 1 da experiência sumariada na Tabela 2, coluna 4 (11-multiplexer).

Os parâmetros utilizados por corrida estão indicados na coluna 4 da Tabela 2. A primeira solução correcta foi encontrada na geração 390 da corrida 1 (os terminais são unidos 3 a 3, formando uma AE com profundidade de 4, composta de 40 nódulos, correspondendo os primeiros 14 nódulos a IFs e os restantes aos caracteres do cromossoma; ver a expressão K 3.12 e a Figura 5):

```
3652bb5bbba4c87c43bcc62a51
```

sendo uma solução universal para o problema do 11-multiplexer. Na Figura 16 mostra-se a progressão da aptidão média da população e a aptidão do melhor indivíduo de cada geração para a corrida 1 da experiência sumariada na Tabela 2, coluna 4.

Como se mostra na quarta coluna da Tabela 2, a PEG resolve o 11-multiplexer com uma taxa de sucesso de 0,57. É de salientar que a PG foi incapaz de resolver o 11-multiplexer com populações de 500 indivíduos por 51 gerações [8], tendo só conseguido resolvê-lo utilizando populações de 4.000 indivíduos [6].

7. Conclusões

Os detalhes da implementação da PEG foram explicados detalhadamente, permitindo que outros investigadores implementem este novo algoritmo. Além do mais, os problemas escolhidos para ilustrar o funcionamento da PEG, mostram que este novo paradigma pode ser utilizado para resolver vários problemas de áreas muito diversas com a vantagem de funcionar eficientemente num computador pessoal. O novo conceito por detrás dos cromossomas lineares e das AEs permitiu que a PEG superasse consideravelmente a PG: mais de duas ordens de grandeza nos problemas de regressão simbólica, indução de sequências e empilhamento de blocos, e mais de quatro ordens de grandeza no problema da classificação da densidade. Portanto, a PEG oferece novas possibilidades na resolução de problemas tecnológicos e científicos mais complexos. Igualmente importante e original, é a organização multigénica dos cromossomas da PEG, que faz da PEG uma técnica de descoberta verdadeiramente hierárquica. Por último, os algoritmos de expressão genética representam a natureza mais fielmente, podendo, por isso, ser usados como modelos computacionais de processos evolutivos naturais.

Agradecimentos

Estou imensamente grata ao José Simas por ter ajudado com o hardware, por ter lido e comentado o manuscrito e pelo seu entusiasmo e apoio enquanto eu me debatia com as ideias básicas e os conceitos da PEG.

Bibliografia

1. M. Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, 1996.
2. J. Maynard Smith and E. Szathmáry, *The Major Transitions in Evolution*, W. H. Freeman, 1995.
3. M. J. Keith and M. C. Martin, *Genetic Programming in C++: Implementation Issues*. In K. E. Kinnear, ed., *Advances in Genetic Programming*, MIT Press, 1994.
4. W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and its Applications*, Morgan Kaufmann, 1998.
5. D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
6. J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, Cambridge, MA: MIT Press, 1992.
7. C. K. Mathews, K. E. van Holde, and K. G. Ahern, *Biochemistry*, 3rd ed., Benjamin/Cummings, 2000.
8. U.-M. O'Reilly and F. Oppacher, A comparative analysis of genetic programming. In P. J. Angeline and K. E. Kinnear, eds., *Advances in Genetic Programming 2*, MIT Press, 1996.
9. M. Mitchell, P. T. Hraber, and J. P. Crutchfield, 1993. Revisiting the edge of chaos: Evolving cellular automata to perform computations. *Complex Systems* 7, 89-130.
10. M. Mitchell, J. P. Crutchfield, and P. T. Hraber, 1994. Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D*: 75, 361-391.
11. J. P. Crutchfield, and M. Mitchell, 1995. The evolution of emergent computation. *Proceedings of the National Academy of Sciences, USA*, 82, 10742-10746.
12. R. Das, M. Mitchell, and J. P. Crutchfield, 1994. A genetic algorithm discovers particle-based computation in cellular automata. In Y. Davidor, H.-P. Schwefel, and R. Männer, eds., *Parallel Problem Solving from Nature - PPSN III*. Springer-Verlag, 1994.
13. J. R. Koza, F. H. Bennett III, D. Andre, and M. A. Keane, M. A. *Genetic Programming III: Darwinian Invention and Problem Solving*. San Francisco: Morgan Kaufmann Publishers, 1999.
14. H. Juillé, and J. B. Pollack. Coevolving the "ideal" trainer: Application to the discovery of cellular automata rules. In J. R. Koza, W. Banzhaf, K. Chellapilla, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba, and R. L. Riolo, eds., *Genetic Programming 1998: Proceedings of the Third Annual Conference*. Morgan Kaufmann, San Francisco, CA, 1998.