3^o Encontro Nacional do Colégio de Engenharia Electrotécnica 1997

Ordem dos Engenheiros - Colégio de Eng. Electrotécnica

Título

APRENDIZAGEM DE DCGs UTILIZANDO ALGORITMOS GENÉTICOS

Área

Computadores e Sistemas Informáticos

Autores

Manuel Alcino Cunha
Departamento de Informática
Escola de Engenharia
Universidade do Minho
Largo do Paço, 4709 Braga Codex, PORTUGAL
tel: 351-53-604470
fax: 351-53-604471
emails: alcino@di.uminho.pt

Sumário

Este artigo apresenta um estudo sobre a adequação da aplicação de algoritmos genéticos ao problema da aprendizagem de *Definite Clause Grammars* (DCGs). Dada uma linguagem, caracterizada por um conjunto de frases gramaticais e por um conjunto de frases agramaticais, pretende-se descobrir qual a gramática que caracteriza essa linguagem. São propostas duas variantes de algoritmos genéticos, uma sem espécies e outra com espécies, e analisadas e comparadas as suas capacidades de aprendizagem.

APRENDIZAGEM DE DCGs UTILIZANDO ALGORITMOS GENÉTICOS

Manuel Alcino Cunha

Departamento de Informática, Escola de Engenharia, Universidade do Minho, 4710 Braga, Portugal

Email: alcino@di.uminho.pt

Sumário

Este artigo apresenta um estudo sobre a adequação da aplicação de algoritmos genéticos ao problema da aprendizagem de *Definite Clause Grammars* (DCGs). Dada uma linguagem, caracterizada por um conjunto de frases gramaticais e por um conjunto de frases agramaticais, pretende-se descobrir qual a gramática que caracteriza essa linguagem. São propostas duas variantes de algoritmos genéticos, uma sem espécies e outra com espécies, e analisadas e comparadas as suas capacidades de aprendizagem.

1 INTRODUÇÃO

As DCGs [1] são um dos paradigmas mais utilizados no processamento de linguagem natural, no contexto da programação em lógica [2, 3]. Este formalismo, largamente disponibilizado em muitos sistemas Prolog, parte de uma especificação da gramática em regras de reescrita e gera estruturas, sob a forma de cláusulas definidas, que correspondem às regras gramaticais. Na altura do seu aparecimento constituiram uma séria alternativa ao método das redes de transição, então muito utilizadas [4]. Normalmente, as DCGs são utilizadas em acções de *parsing* ou de análise semântica de textos, pressupondo-se, nestes casos, uma definição, à priori, da gramática a utilizar. Neste artigo, é proposta uma estratégia que permite fazer o contrário, ou seja, dado um conjunto de frases descobrir a DCG que as poderia ter originado.

O problema da descoberta de gramáticas é claramente um problema da classe NP, isto é, com um tempo de execução que cresce exponencialmente em relação ao número de frases exemplo utilizadas. De entre as diversas estratégias existentes na abordagem deste tipo de problemas, os algoritmos genéticos têm sido utilizados com relativo sucesso [5]. No entanto, é sabido que não se adequam a resolver todo o tipo de problemas, nomeadamente devido às dificuldades que apresentam na codifição dos parâmetros do problema na estrutura de um cromossoma. O objectivo principal deste trabalho é precisamente estudar a sua adequação ao processo de descoberta de gramáticas.

A estratégia de aprendizagem proposta, baseada em algoritmos genéticos, pode ser considerada como um caso de *reinforcement learning* [6] se considerarmos que o sucesso de um indivíduo se mede pelo seu número de descendentes. Embora não haja uma tentativa de aprender, explicitamente, a relação entre as acções tomadas (escolha de uma ou de outra gramática) e as recompensas dessas acções (mais ou menos frases certas), o algoritmo converge para uma solução através de uma procura no espaço dos indivíduos, tentando maximizar a função de *fitness* definida. Esta função de *fitness* é calculada a partir dos exemplos de frases gramaticais e agramaticais fornecidos ao algoritmo.

Para além de se estudar a adequação da estratégia tradicional, em que uma população uniforme é sujeita ao processo de melhoramento, estudou-se outro tipo de estratégia na qual convivem várias espécies de indivíduos. Para existirem espécies é necessário introduzir técnicas de acasalamento selectivo [7], isto é, a troca de material genético só é feita entre indivíduos que partilham determinadas caracteristicas. Desta forma, consegue-se que o processo de melhoramento explore várias alternativas de solução. Ocasionalmente, as mutações nos cromossomas levarão a acasalamentos entre indivíduos de diferentes espécies e, consequentemente, à troca do melhor material genético. Este processo tenta reproduzir o que se passa na evolução das diferentes espécies, sendo mais fiel ao mecanismo de selecção natural que os tradicionais algoritmos genéticos.

2 DESCRIÇÃO DOS ALGORITMOS

A concepção de um algoritmo genético para a resolução de um problema implica a resposta a quatro questões [6]:

- Como é calculada a função de *fitness* ?
- Como são representados os indivíduos ?
- Como são seleccionados os casais ?
- Como é que se reproduzem ?

Na secção seguinte estas questões serão abordadas (não necessariamente por esta ordem) para o caso do algoritmo sem espécies. Depois, serão apresentadas as mudanças efectuadas para se introduzir a noção de espécie nesse algoritmo.

Os algoritmos apresentados foram implementados em SWI-Prolog [8].

Algoritmo sem Espécies

Num algoritmo genético cada indivíduo é univocamente representado pelo seu cromossoma. Um cromossoma é uma sequência de genes retirados de um alfabeto finito. O primeiro problema que se coloca consiste em saber como serão obtidas as várias regras de uma DCG a partir de um único cromossoma. A solução escolhida envolve duas estratégias, uma para a separação das regras com diferentes símbolos no lado esquerdo e outra para dividir as regras que derivam do mesmo símbolo.

Como se pode observar na Figura 1, cada cromossoma está dividido em r segmentos, cada um com a genes, dando origem ao comprimento total de $r \times a$ genes. Assim, um cromossoma origina no máximo r regras que derivam de diferentes símbolos não-terminais, tendo cada uma no máximo a símbolos terminais e não-terminais.

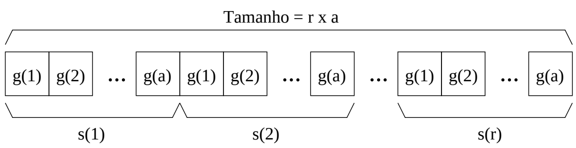

Figura 1: Estrutura do cromossoma sem espécies.

A segunda estratégia consiste na introdução de um tipo de gene especial, denominadado *ou*, que indica uma divisão do segmento na posição onde aparece. Por exemplo, um segmento com um *ou* no meio dá origem a duas regras que derivam do mesmo não-terminal, ficando uma com os símbolos à esquerda do *ou* e a outra com os símbolos à sua direita. Quando a distribução dos genes do tipo *ou* dá origem a regras sem símbolos do lado direito (por exemplo, quando um *ou* aparece num dos extremos ou existem dois *ou* consecutivos), procede-se a uma optimização, não se expandindo essa regra.

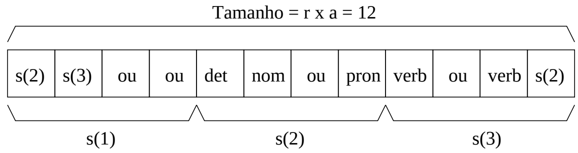

Figura 2: Exemplo de cromossoma.

Os tipos de genes permitidos são, para além do *ou*, as categorias dos terminais envolvidos na aprendizagem (neste caso as categorias das palavras) e tantos símbolos do tipo `s(i)` quantos os segmentos especificados. Este último tipo de gene serve para indicar os símbolos não terminais correspondentes às regras da DCG. A regra de entrada é denominada pelo símbolo `s(1)` na versão sem espécies. Na Figura 2 é apresentado um exemplo de cromossoma com $r = 3$, $a = 4$ e com apenas quatro categorias de terminais (`det`, `nome`, `pron` e `verbo`). Na Figura 3 apresenta-se a DCG correspondente.

```
s(1) --> s(2), s(3).
s(2) --> det, nome.
s(2) --> pron.
s(3) --> verbo.
s(3) --> verbo, s(2).
```

Figura 3: DCG gerada pelo cromossoma exemplo.

Para concluir a questão da representação falta apenas determinar se todos os cromossomas dão origem a indivíduos válidos, ou seja, se é possível determinar a função de *fitness* para qualquer cromossoma construido de acordo com as regras descritas. Como as DCGs dão origem a *parsers top-down* e *left-to-right* é necessário algum cuidado com o problema da recursividade à esquerda: quando o símbolo do qual deriva a regra aparece no inicio do corpo da mesma. Na prática, quando se escreve uma DCG é sempre possível "dar a volta" a esse problema e reescrever a DCG pretendida sem recursividade à esquerda [9]. Neste caso, como os cromossomas vão sendo modificados segundo um processo aleatório, não é possível garantir que a DCG gerada não terá esse problema.

Para lidar com o problema da recursividade nas DCGs optou-se por modificar o processo de *parsing* das frases. O resultado pretendido é, quando o *parser* entra em ciclo devido à recursividade à esquerda, considerar-se a frase que está a ser reconhecida como agramatical, dada a gramática em questão. Para se obter este resultado foi necessário alterar os predicados que transformam as DCGs em predicados Prolog, por forma a ser incluída no início de cada regra uma condição para testar se o algoritmo está a entrar em ciclo. Como exemplo do resultado da nova conversão apresentam-se na Figura 4 os predicados correspondentes à DCG da Figura 3. O primeiro predicado do corpo de cada regra testa se o *parser* está a entrar em ciclo. O segundo predicado regista a informação que indica a tentativa de expansão da regra.

```
s(1, A, B) :-
        not(done(s(1, A, B), 2)),
        assert(done(s(1, A, B), 2)),
        s(2, A, C),
        s(3, C, B).
s(2, A, B) :-
        not(done(s(2, A, B), 2)),
        assert(done(s(2, A, B), 2)),
        det(A, C),
        nome(C, B).
s(2, A, B) :-
        not(done(s(2, A, B), 0)),
        assert(done(s(2, A, B), 0)),
        pron(A, B).
s(3, A, B) :-
        not(done(s(3, A, B), 3)),
        assert(done(s(3, A, B), 3)),
        verbo(A, B).
s(3, A, B) :-
        not(done(s(3, A, B), 0)),
        assert(done(s(3, A, B), 0)),
        verbo(A, C),
        s(2, C, B).
```

Figura 4: Predicados gerados pela DCG exemplo.

Resolvido o problema de gerar uma DCG correcta a partir do cromossoma, as restantes questões apresentam uma resposta praticamente trivial. O *fitness* de cada indivíduo calcula-se subtraíndo ao número de frases gramaticais, reconhecidas pela DCG, o número de frases agramaticais reconhecidas. É absolutamente necessário incluir exemplos de frases agramaticais para se evitar que a população degenere em indivíduos que correspondam a gramáticas triviais (Figura 5).

```
s(1) --> s(2), s(1).
s(1) --> s(2).
s(2) --> pron.
s(2) --> verbo.
s(2) --> det.
     ...
```

Figura 5: DCG de gramática trivial.

A selecção dos casais é feita em duas fases. Primeiro, determina-se qual a probabilidade de um indivíduo ser escolhido dado o seu valor de *fitness*. Depois, se considerarmos que a população é composta por i indivíduos, escolhem-se i indivíduos da população, sendo cada indivíduo seleccionado de acordo com a função de distribuição de probabilidade determinada anteriormente. Estes indivíduos são agrupados em casais, sendo, por isso, fundamental que a população tenha um número par de elementos. Para que o melhoramento aconteça, cada indivíduo pode ser escolhido várias vezes. Assim, os indivíduos com o *fitness* mais elevado acasalarão mais vezes.

O cálculo da probabilidade de um indivíduo ser escolhido deve respeitar as seguintes propriedades:

- um indivíduo com maior *fitness* que outro deve ter sempre uma probabilidade maior;
- as diferenças entre os vários valores de *fitness* devem reflectir-se em diferenças proporcionais nos valores de probabilidade; e
- todos os indivíduos devem ter a hipótese de serem escolhidos.

Assim, sendo $f(i)$ o valor do *fitness* no indivíduo i e $\overline{f}$ a média do *fitness* de toda a população, a probabilidade não normalizada determina-se como

$$p(i) = \begin{cases} f(i) - \overline{f} + 1 & se(f(i) - \overline{f}) \geq 0 \\ \frac{1}{-(f(i)-\overline{f})+1} & se(f(i) - \overline{f}) < 0. \end{cases}$$

Normalizando $p(i)$ obtem-se a probabilidade de um indivíduo ser escolhido para acasalar.

Dados dois indivíduos, a reprodução realiza-se de acordo com o normal nos algoritmos genéticos, ou seja, primeiro faz-se o *cross-over* do material genético (o ponto de *cross-over* é escolhido aleatoriamente) e depois faz-se a mutação dos genes. A probabilidade p de mutação é independente para cada gene do cromossoma.

Algoritmo com Espécies

O segundo algoritmo desenvolvido introduz o conceito de espécie no processo de melhoramento genético. Nesta versão, considerou-se a existência de r espécies, tantas quantos os segmentos, sendo as espécies distinguidas pelo facto de possuirem regras de entrada diferentes. A estrutura do cromossoma teve de ser alterada para se poder identificar a espécie do indivíduo. Como se pode ver na Figura 6, foi acrescentado um gene no inicio do cromossoma para indicar a espécie. Este gene apenas pode ser do tipo $s(i)$.

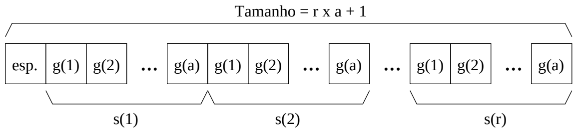

Figura 6: Estrutura do cromossoma com espécies.

Ao nível do processo de melhoramento, apenas se alterou a forma como são seleccionados os casais de indivíduos. Em vez de se escolher i indivíduos de toda a população, faz-se a escolha espécie a espécie, escolhendo-se tantos indivíduos de cada espécie quantos os elementos actuais da espécie. Embora o acasalamento seja selectivo, existem, ocasionalmente, acasalamentos entre indivíduos de espécies diferentes, originando processos de troca de material genético entre as espécies. Esta mistura entre as espécies é necessária para se evitar que um algoritmo com r espécies seja equivalente a r algoritmos sem espécies com i/r indivíduos. A mistura pode acontecer por dois motivos:

- em consequência de uma mutação no gene que define a espécie;
- devido a não se garantir que o número de elementos de cada espécie seja par, o que leva à necessidade de, ocasionalmente, se acasalarem indivíduos de espécies diferentes.

O processo de reprodução é idêntico ao do algoritmo sem espécies.

3 ANÁLISE DOS RESULTADOS OBTIDOS

Para se testar e comparar a adequação dos dois algoritmos ao problema da aprendizagem de DCGs, utilizou-se um conjunto de 12 frases gramaticais e 12 frases agramaticais. As palavras utilizadas foram classificadas manualmente. A utilização de tão poucas frases deve-se às seguintes razões:

- primeiro, o trabalho apresentado constituiu apenas um primeiro estudo da adequação dos algoritmos genéticos ao problema, não tendo por ambição demonstrar a sua eficácia nem construir um algoritmo que aprenda a gramática completa de uma língua;
- segundo, um teste conclusivo necessitaria de um *corpus* considerável de frases gramaticais e agramaticais, já com as palavras classificadas, estando a elaboração desse *corpus* fora do âmbito deste estudo; e
- finalmente, o número de testes necessários para estudar a sensibilidade dos resultados em relação aos parâmetros dos algoritmos impede a utilização de muitas frases, pois a capacidade de cálculo existente não permitiria realizar o teste em tempo útil (não se deve esquecer a enorme complexidade do problema de descoberta de gramáticas).

Nesta primeira fase de testes pretendeu-se, para além de estudar a adequação dos algoritmos, comparar as duas versões desenvolvidas e analisar a sensibilidade dos resultados em relação a p, a probabilidade de mutação dos genes, e i, o número total de indivíduos. Para tal, foram realizados 4 testes para cada versão. Como a gramática que se procurou aprender é igual à da Figura 3 fixou-se $r = 3$ e $a = 5$. Dada a natureza aleatória dos algoritmos genéticos, relizaram-se 3 passagem para cada combinação dos parâmetros. Devido às limitações na capacidade de cálculo, o número máximo de iterações foi fixado em 10000. Assim, nas Tabelas 1 e 2 temos, na primeira coluna, os parâmetros utilizados em cada teste, na segunda, a percentagem de execuções com sucesso (execuções que conseguiram descobrir a gramática antes do limite de iterações), e, na terceira, o número médio de iterações necessário para descobrir a gramática (a média é realizada apenas entre as execuções com sucesso). Uma configuração é mais eficaz do que outra se a percentagem de sucesso é maior, e no caso de serem idênticas, se a média de iterações for menor.

parâmetros	**sucesso**	**iterações**
$i = 60$ $p = 0.04$ $r = 3$ $a = 5$	66.7%	2501
$i = 60$ $p = 0.07$ $r = 3$ $a = 5$	100%	3599
$i = 60$ $p = 0.10$ $r = 3$ $a = 5$	33.3%	2887
$i = 90$ $p = 0.07$ $r = 3$ $a = 5$	100%	2699

Tabela 1: Resultados do algoritmo sem espécies.

O resultado mais surpreendente é a constatação da superioridade do algoritmo sem espécies em todas as configurações testadas. Esta vantagem pode ser explicada de duas formas:

- com poucos indivíduos a eficácia de um algoritmo genético cresce super-linearmente em relação ao tamanho da população, o que implica que uma espécie única, com i indivíduos, apresente melhores resultados do que r espécies com i/r indivíduos;

- é bem conhecido da biologia, que existe um número mínimo de indivíduos necessário para garantir a sustentabilidade e o sucesso de uma espécie, número esse que pode não ter sido atingido na versão com espécies, onde cada uma tinha, em média, 20 ou 30 elementos.

No entanto, este resultado não deve ser considerado definitivo, devendo no futuro ser confrontado com resultados obtidos com grandes populações. O mecanismo de seleção natural é um caso paradigmático do sucesso proporcionado pela biodiversidade e é de estranhar que esse sucesso não se verifique neste caso.

parâmetros	**sucesso**	**iterações**
$i = 60$ $p = 0.04$ $r = 3$ $a = 5$	66.7%	4805
$i = 60$ $p = 0.07$ $r = 3$ $a = 5$	66.7%	2806
$i = 60$ $p = 0.10$ $r = 3$ $a = 5$	33.3%	9255
$i = 90$ $p = 0.07$ $r = 3$ $a = 5$	66.7%	2398

Tabela 2: Resultados do algoritmo com espécies.

Os restantes resultados eram relativamente previsiveis. O número médio de iterações necessário para aprender a gramática decresce com o aumento do tamanho da população. Este resultado não se reflete num aumento proporcional ao nível da eficácia dos algoritmos, já que o tempo de cada iteração aumenta para populações maiores. Os tempos médios das iterações não foram apresentados porque ambas as versões necessitam de ser optimizadas para que a comparação seja justa. Como não se procedeu a essas optmizações optou-se por não os apresentar.

No respeitante à probabilidade de mutação, ficou realçada a importância do afinamento do parâmetro p. Por um lado, valores de p muito baixos implicam maiores tempos para aprendizagem e maior tendência para convergir para máximos locais. Por outro lado, garantem que o algoritmo não "esqueça" as boas soluções já encontradas. Valores de p elevados permitem que se percorra mais rapidamente todo o espaço de procura, mas não garantem, tão bem, a sustentabilidade da população, devido à frequente degeneração dos bons indivíduos. Uma escolha intermédia revelou-se sempre como a melhor opção.

Quanto ao objectivo principal do trabalho, as conclusões revelaram-se bastante positivas. Embora as frases e gramática utilizadas sejam simples, se se levar em conta o baixo número de indivíduos e iterações testados, pode-se prever uma boa adequação dos algoritmos genéticos à descoberta de gramáticas. Com efeito, depois de afinada a probabilidade de mutação, o algoritmo sem espécies revelou uma taxa de sucesso de 100% e, também, ser necessário um baixo número de iterações para descobrir a gramática. No entanto, estes resultados foram obtidos partindo do conhecimento prévio do número e tamanho das regras necessárias. Numa versão definitiva seria necessário permitir que os parâmetros r e a fossem também sujeitos ao processo de evolução, pois seria inviável testar alternadamente todas as combinações possíveis.

4 CONCLUSÕES E TRABALHO FUTURO

Neste artigo demonstrou-se que é possível implementar mecanismos de aprendizagem de DCGs utilizando algoritmos genéticos, atingindo-se assim o principal objectivo proposto. Surpreendentemente, foi demonstrado que para populações pequenas é preferível a utilização de um algoritmo genético tradicional (onde uma população homogénea é sujeita ao processo de melhoramento) do que uma versão com espécies (onde se permite a exploração simultânea de soluções alternativas). Também se confirmou o aumento da eficácia com o aumento da população e a sensibilidade dos resultados em relação à probabilidade de mutação.

Para trabalho futuro fica a optimização dos algoritmos, necessária para se poder testar o processo de aprendizagem com populações maiores e mais variedade de configurações. Complementarmente, também será objecto de trabalho futuro a implementação da possibilidade de evolução dos parâmetros r e a. Para se atingir o primeiro objectivo poderão ser seguidas várias alternativas, nomeadamente: a implementação do processo de aprendizagem numa linguagem mais eficiente, a paralelização dos algoritmos genéticos e a utilização de heurísticas para determinar o *fitness* dos indivíduos.

5 AGRADECIMENTOS

Agradeço ao Eng. Orlando Belo e ao Eng. José João Almeida o apoio prestado durante a realização deste artigo.

Este trabalho recebeu apoio financeiro da JNICT através da bolsa com a referência BD/9269/96.

Referências

[1] F. C. N. Pereira and D. H. D. Warren. Definite clause grammars for language analysis: A survey of the formalism and comparison with augmented transistion networks. *Artificial Intelligence*, 13(3):231–278, 1980.

[2] Fernando C. N. Pereira and Stuart M. Shieber. *Prolog and Natural-Language Analysis*. 1987.

[3] Harvey Abramson and Veronica Dahl. *Logic Grammars*. Symbolic Computation - Artificial Intelligence. 1989.

[4] Helder Coelho. *Inteligência Artificial em 25 Lições*. Fundação Calouste Gulbenkian, 1995.

[5] D.E. Goldberg. *Genetic Algorithms is Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company, Inc., 1989.

[6] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Inc., Upper Saddle River, New Jersey 07458, 1995.

[7] John H. Holland. *Hidden Order - How Adaptation Builds Complexity*. Helix Books, Addison-Wesley Publishing Company, 1995.

[8] Jan Wielemaker. *SWI-Prolog 1.6 Reference Manual*. University of Amsterdam, Dept. of Social Science Informatics, Amsterdam, The Netherlands, 1990.

[9] Gerard Gazdar and Christopher Melish. *Extracts from the book "Natural Language Processing in Prolog"*.