Gramáticas Adaptativas – Estado Atual da Pesquisa aplicada em Autômatos Adaptativos

César Bravo

***Resumo*—Este artigo apresenta um roteiro de pesquisa em autômatos adaptativos baseado em resultados obtidos para gramáticas livres de contexto adaptativas com verificação de aparência. Os resultados gramaticais são descritos e exemplificados e é proposta uma estratégia para obter resultados análogos para o caso dos autômatos adaptativos. Essa estratégia é logo aplicada com sucesso para deduzir uma versão restrita, para o caso dos autômatos adaptativos, do resultado geral para gramáticas livres de contexto adaptativas com verificação de aparência.**

Palavras-chave: **Gramáticas adaptativas, autômatos adaptativos.**

I. Introdução

Este trabalho descreve os resultados da pesquisa do autor em gramáticas adaptativas **AG** (do inglês *Adaptive Grammars* [2]) e apresenta algumas alternativas de pesquisas para obter resultados análogos para autômatos adaptativos. A principal contribuição na área das gramáticas adaptativas foi uma simplificação baseada em produções livres de contexto, a qual, além de funções adaptativas simplificadas, utiliza uma noção especial de passo de derivação. Essa versão do formalismo foi batizada como *gramáticas livres de contexto adaptativas com verificação de aparência*, **CFAG WAC** (do inglês *Context-free Adaptive Grammars With Appearance Checking* [1]) e possui poder computacional de máquina de Turing.

O conceito de *verificação de aparência* estende a definição de passo de derivação segundo a hierarquia de Chomsky e é usado para "dar mais um passo de derivação" em configurações nas quais não seria possível fazê-lo, segundo o conceito clássico ([1], [3], [4], [5]).

O resultado principal sobre gramáticas livres de contexto adaptativas com verificação de aparência pode ser resumido da seguinte forma:

Teorema 1

As gramáticas livres de contexto adaptativas com verificação de aparência identificam a classe das linguagens recursivamente enumeráveis L_0. Além disso, basta usar funções adaptativas sem geradores, nem variáveis, nem ações adaptativas elementares de consulta, e as funções adaptativas trabalham sempre sobre os subconjuntos de um conjunto fixo de produções livres de contexto e o único parâmetro requerido é uma produção.

□

O objetivo deste trabalho é focalizar a questão: é possível obter, para autômatos adaptativos, um resultado similar ao teorema acima?

Isto é, quem já utilizou dispositivos adaptativos em algum problema computacional está ciente de que, em geral, as ações adaptativas elementares de consulta retornam conjuntos como respostas, e o processamento do dispositivo deve tratar cada um dos elementos desses conjuntos, o que agrega dificuldades adicionais aos problemas usais de não-determinismo (no caso dos autômatos) e da ambigüidade (no caso das gramáticas).

Assim, o fato de ter sido eliminado o uso das ações elementares de consulta resulta em uma vantagem (simplificação) desta versão das gramáticas adaptativas com respeito à definição original. É desejável, portanto, pesquisar a possibilidade de obter uma simplificação similar para o caso dos autômatos adaptativos.

Outra simplificação das gramáticas livres de contexto adaptativas com respeito à versão original é o fato de elas não precisarem de geradores ou variáveis, isto é, trabalharem sempre sobre um conjunto fixo de produções livres de contexto, do qual as funções adaptativas selecionam alguns subconjuntos que vêm a ser usados como "conjunto de produções corrente". Pode-se perceber que essa característica tem muito impacto na simplificação da implementação de um dispositivo para manipular esse tipo de gramáticas adaptativas e, no caso dos autômatos adaptativos, viria a significar que bastaria usar um conjunto fixo e finito de estados, entre os quais as funções adaptativas atuariam criando e removendo transições durante o exame de uma cadeia de entrada.

Uma simplificação desse tipo parece muito improvável para o caso de autômatos finitos adaptativos, mas veremos um resultado que mostra que existem linguagens dependentes de contexto que podem ser reconhecidas por um autômato (de pilha) adaptativo com um número prefixado de estados.

Há a questão também de complexidade mínima para poder computacional máximo. Isto é, pode-se mostrar, usando métodos clássicos ([3], [4]), que as gramáticas regulares

Recebido em 2007-07-26

César Bravo: Universidade de São Paulo. Escola Politécnica da Universidade de São Paulo. Caixa Postal 668. 13560-970~~São Paulo -- SP. cesarabravop@hotmail.com.

adaptativas não podem expressar a mesma classe de linguagens formais do que as gramáticas livres de contexto adaptativas. Nesse sentido, as produções livres de contexto vêm a ser o menor tipo, na hierarquia de Chomsky, para atingir poder computacional máximo.

Para que este trabalho possa servir como um roteiro para tentar obter um resultado similar para autômatos adaptativos, é necessário: a) conhecer as ferramentas usadas para provar o teorema 1; b) perceber que essas são ferramentas gramaticais; c) pesquisar a disponibilidade de ferramentas similares para autômatos e d) tentar reproduzir a prova do teorema 1 para o caso dos autômatos adaptativos, utilizado as ferramentas para autômatos.

Neste ponto, parece conveniente recomendar uma estratégia "bottom-up": construir exemplos específicos a partir dos quais se possa tentar inferir resultados gerais. Deve ser claro, também, que é conveniente manter uma posição minimalista com respeito às generalizações possíveis.

A. Um exemplo simples.

Consideremos a gramática livre de contexto **G = ({ S, A, B, C }, { a, b, c }, S, P)**, com conjunto de produções **P = { p_0 = S→ ABC, p_1 = A→ aA, p_2 = B→ bB, p_3 = C→ cC, p_4 = A→ a, p_5 = B→ b, p_6 = C→ c }**, a qual gera a linguagem **L(G) = { $a^+b^+c^+$ }**.

Para este caso, a estratégia recomendada pode ser descrita da seguinte forma: como se pode estender a gramática **G** para uma nova gramática $\mathbf{G^{++}}$, usando-se apenas produções livres de contexto e de modo que $\mathbf{L(G^{++}) = \{ a^n b^n c^n : n \geq 1 \}}$? Como a restrição de usar apenas produções livres de contexto limita o poder de expressão da gramática resultante a linguagens livres de contexto (tipo 2) e, no exemplo, a gramática estendida deve reconhecer uma linguagem dependente de contexto (tipo 1), deve-se entender que a extensão da gramática deve passar por algum mecanismo que permita ampliar esse poder de expressão.

Esse exemplo da extensão da gramática **G** pode ser visto como um caso particular do seguinte problema: como estender gramáticas livres de contexto para gerar linguagens tipo 1 e tipo 0, usando apenas produções livres de contexto?

Em [1], foram estudados quatro possíveis mecanismos de extensão, os quais passamos a descrever na próxima seção.

O artigo é composto pelas seguintes seções: II Gramáticas com derivações controladas; III Gramáticas Livres de Contexto Adaptativas (sem verificação de aparência); IV Autômatos com mecanismos de controle; V Conclusões e, finalmente, Agradecimentos e Referências.

II. GRAMÁTICAS COM DERIVAÇÕES CONTROLADAS

A. Reescritura controlada

O tema central desta seção também é conhecido como gramáticas com derivações controladas. Na referência [1], foram estudados estes quatro mecanismos:

- Context-Free Time-Variant grammars `T`
- Context-Free Programmed grammars `P`
- Context-Free Matrix grammars `M`
- Context-Free Grammars with regular control language `R`

Usaremos as siglas **CFTVG**, **CFPG**, **CFMG**, **CFGWRCL** para referir-nos a cada uma dessas classes de gramáticas. A seguir, vamos discutir, de modo geral, o poder computacional dessas classes de gramáticas e exemplificar somente os casos de **CFPG** e **CFGWRCL**, mas o leitor pode encontrar os detalhes completos e exemplos para os outros dois casos em [1]. Deve-se notar também que existem vários outros mecanismos de extensão de gramáticas livres de contexto que podem ser consultados em [3], [4] e [5].

Os resultados clássicos sobre o poder de expressão dessas classes de gramáticas controladas com respeito à máquina de Turing podem ser resumidos no seguinte diagrama

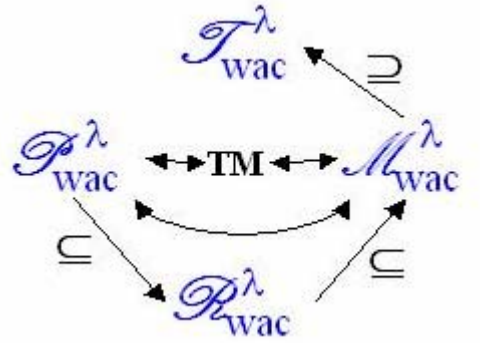


Figura 1. Reescritura controlada versus máquina de Turing.

Na Figura 1, o sobrescrito **λ** indica que são consideradas gramáticas com produções que geram a palavra vazia. O subscrito **wac** indica que as derivações podem ser aplicadas em modo de verificação de aparência; as setas duplas entre o formalismo da maquina de Turing e das gramáticas programadas [4] e das gramáticas matriciais ([3], [5]) indicam que há prova direta e construtiva da equivalência entre os formalismos apontados pelas pontas da dupla seta. As setas simples etiquetadas com a relação de inclusão indicam que o formalismo apontado pela ponta da seta simula o formalismo do qual a seta parte.

A condição de as gramáticas com mecanismos de controle considerarem ou não produções que geram a palavra vazia tem efeito no poder de expressão do modelo resultante, pois, no caso de não serem consideradas produções que geram a palavra vazia, as classes de linguagens geradas pelas gramáticas com mecanismos de controle ficam contidas dentro da classe das linguagens dependentes de contexto. Isso pode ser expresso com um diagrama análogo ao anterior, como mostra a Figura 2.

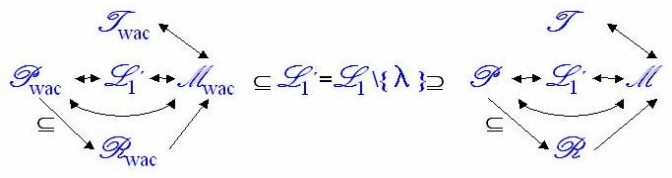


Figura 2. Classes obtidas desconsiderando-se produções que produzem lambda.

Assim mesmo, se utilizarmos somente a noção clássica de passo de derivação, então teremos que **CFAG** $\subseteq \mathtt{L}_1' = \mathtt{L}_1 \setminus \{ \lambda \}$, onde $\mathtt{L}_1 \setminus \{ \lambda \}$ é a classe das linguagens dependentes de contexto do qual foram retiradas aquelas linguagens que podem ser geradas por gramáticas dependentes de contexto que têm produções que geram a palavra vazia λ.

A estratégia usada em [1] foi definir a classe das gramáticas adaptativas baseadas em produções livres de contexto **CFAG**, muni-las com o mecanismo de aplicação de produções em modo de verificação de aparência **CFAG WAC** e provar, em forma construtiva, que elas podem simular cada um dos formalismos **CFTVG**, **CFPG**, **CFMG**, e **CFGWRCL**. A Figura 3 posiciona o novo formalismo com respeito aos outros quatro mecanismos.

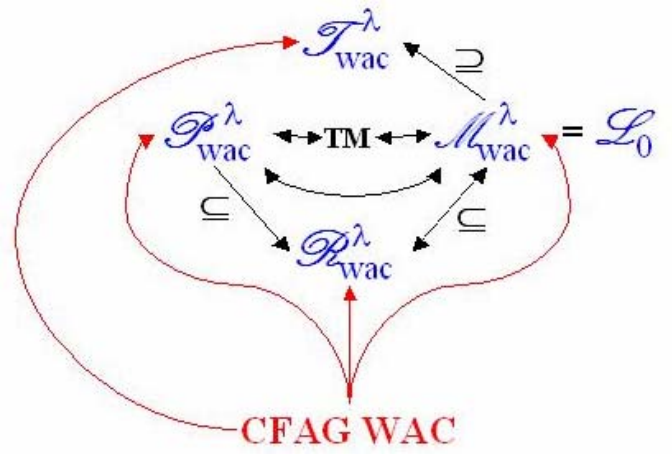


Figura 3. Gramáticas Livres de Contexto Adaptativas simulam gramáticas com mecanismo de controle.

Qual a relação entre esta especial classe de gramáticas adaptativas e as gramáticas adaptativas como definidas originalmente em [2]? Embora ambos os formalismos sejam equivalentes no que se refere ao fato de conseguirem gerar qualquer linguagem gerada por uma gramática de tipo 0 (ou seja, qualquer linguagem formal), do ponto de vista da teoria de conjuntos, pode-se dizer que essas classes possuem interseção e também diferença simétrica não vazia. Ou seja, existem gramáticas adaptativas que não pertencem à classe das **CFAG WAC**: aquelas com produções dependentes de contexto que não são livres de contexto. Também existem gramáticas da classe **CFAG WAC** que não pertencem à classe das **A**G: aquelas que possuem produções que são aplicadas em modo de verificação de aparência. A Figura 4 apresenta esta comparação.

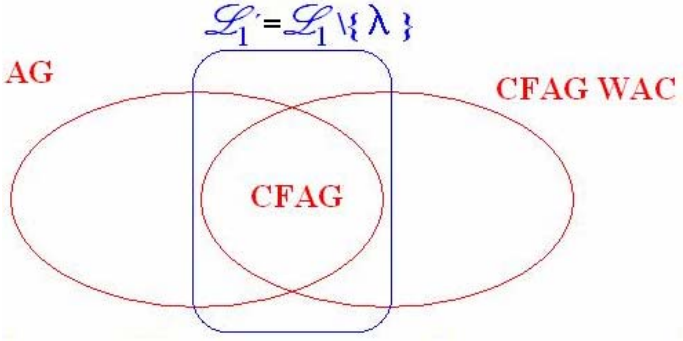


Figura 4. Comparação por conjuntos entre AG e CFAG WAC.

B. Exemplo: Extensão de G para gramática programada.

Para definir uma gramática programada baseada em uma gramática livre de contexto, deve-se definir uma função *sucesso* s(p): $P \rightarrow 2^P$, a qual controla quais são as produções aplicáveis depois da aplicação de uma dada produção. No caso do exemplo I.A, temos a seguinte função.

Tabela 1. A função *sucesso* para a gramática programada $\mathbf{G_p}$.

p	s(p)
p_0	$\{ p_1, p_4 \}$
p_1	$\{ p_2 \}$
p_2	$\{ p_3 \}$
p_3	$\{ p_1, p_4 \}$
p_4	$\{ p_5 \}$
p_5	$\{ p_6 \}$

Talvez o melhor argumento para chamar estas gramáticas de programadas resida no fato de ser possível representar a função sucesso com um diagrama de fluxo, como mostrado na Figura 5.

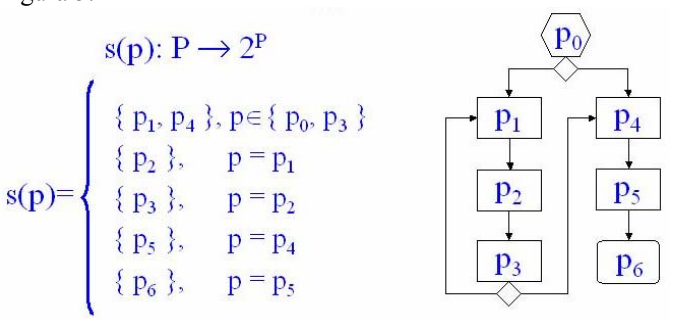
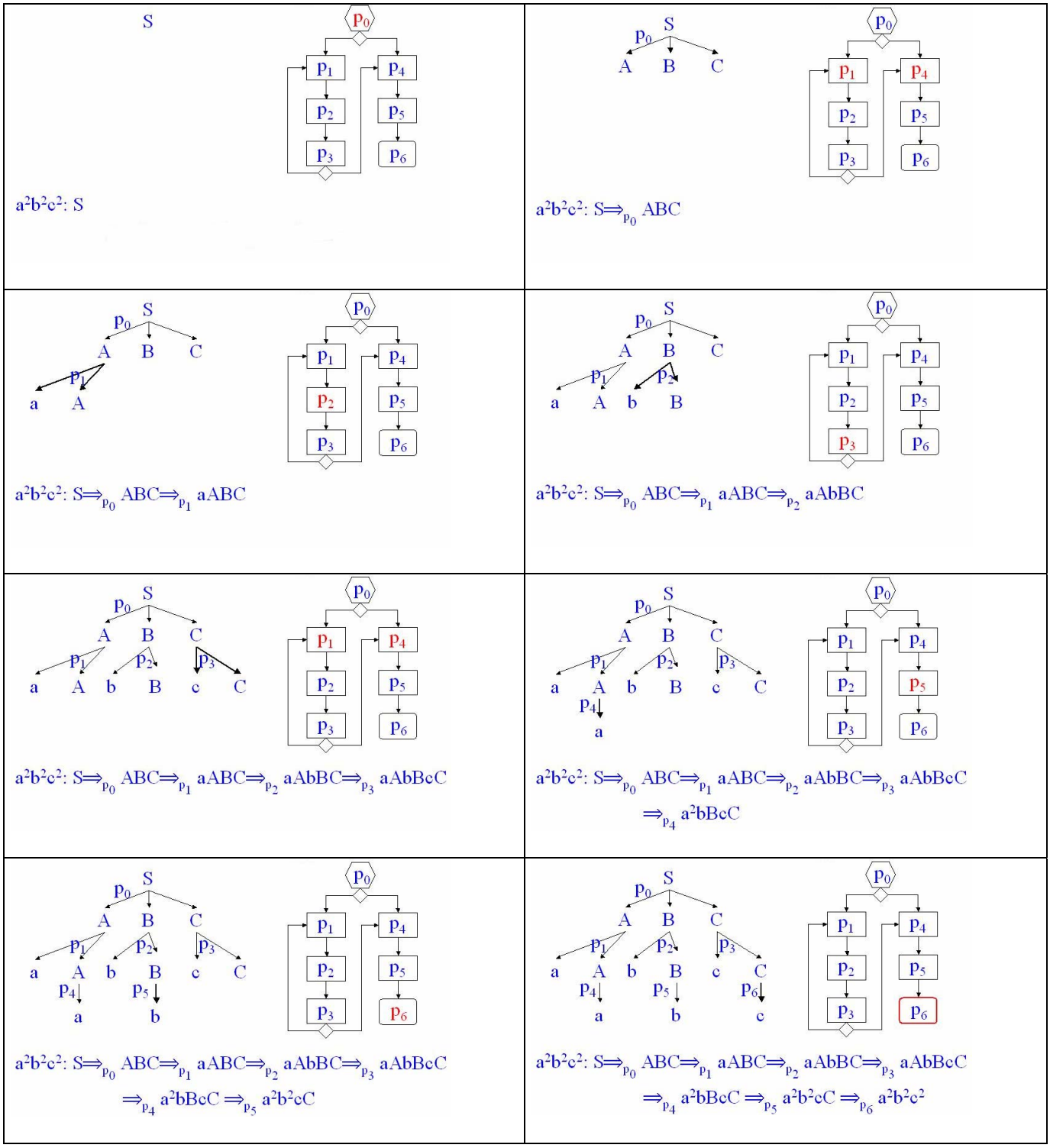


Figura 5. O diagrama de fluxo da **CFPG** $\mathbf{G_p}$.

E, para continuar com a analogia, a derivação de uma palavra segundo uma gramática livre de contexto programada $\mathbf{G_p=(G, s)}$ pode ser vista como a execução do programa descrito pelo diagrama de fluxo respectivo, como ilustrado na sucessão de figuras a seguir, na Tabela 2, lidas da direita para a esquerda e de cima para baixo.

As produções em vermelho no diagrama de fluxo indicam quais produções podem ser aplicadas; os subscritos na relação $\Rightarrow$ indicam qual produção foi aplicada. Um bloco em vermelho indica que não há mais produções aplicáveis.

Tabela 2. Derivação de uma palavra como execução de um programa.

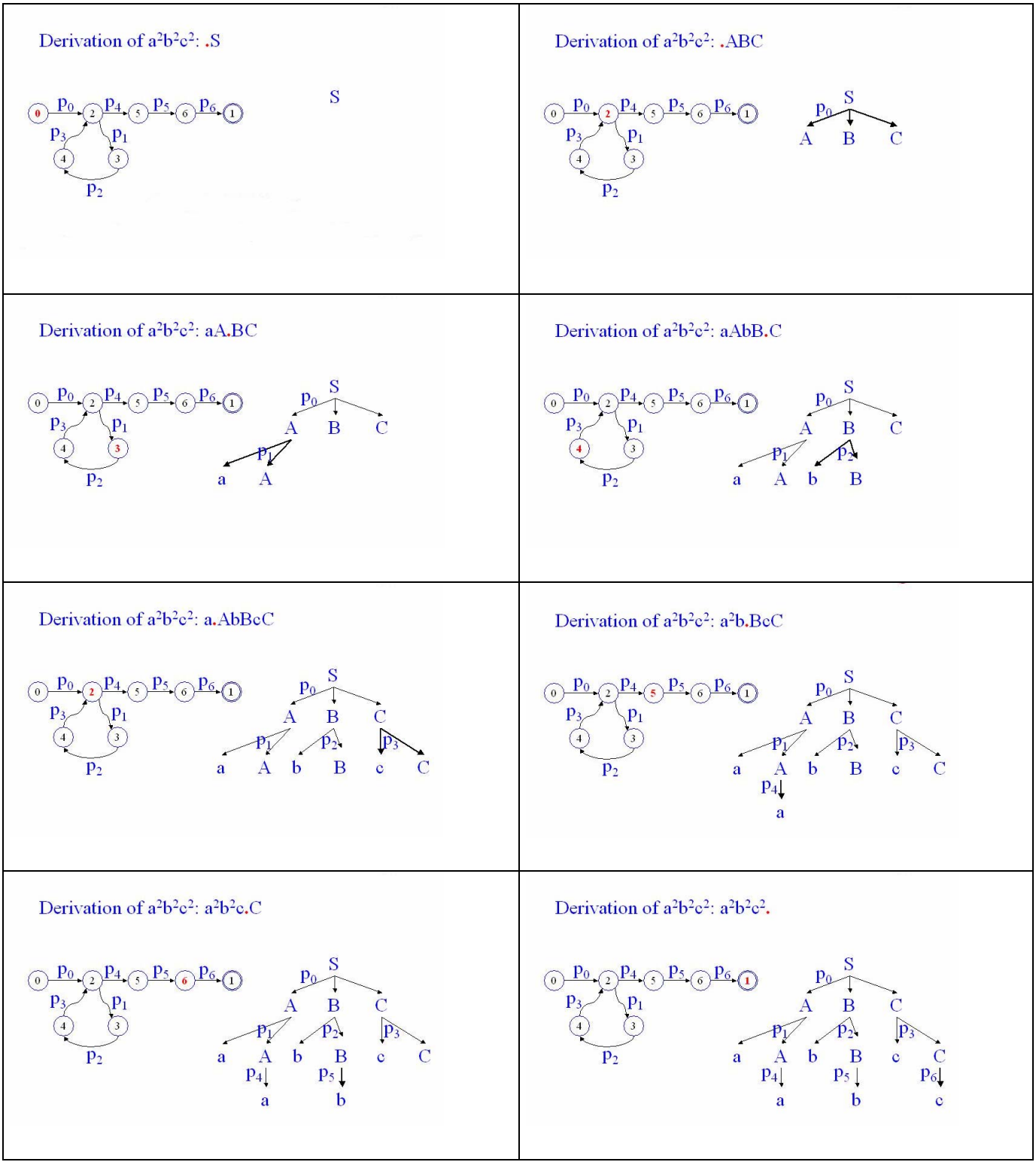


C. Exemplo: Extensão de G para gramática com linguagem de controle regular.

Para definir uma gramática com linguagem de controle regular baseada em uma gramática livre de contexto, deve-se definir uma palavra de controle sobre o alfabeto formado pelos nomes das produções da gramática livre de contexto. As derivações serão então realizadas lendo-se a expressão regular da esquerda para a direita e aplicando-se as produções na ordem em que aparecem na expressão regular. Evidentemente, por se tratar de uma expressão regular, esse processo pode ser governado por um autômato finito e, desse modo, as derivações podem ser vistas como a leitura da palavra pelo autômato finito, como mostrado na seqüência de figuras da Tabela 3 para $\mathbf{G_{reg}=(G,\ p_0(p_1p_2p_3)^*p_4p_5p_6)}$. O estado em vermelho, no autômato, indica o estado corrente; na forma sentencial, o ponto vermelho foi colocado na esquerda de um

não-terminal, que é o lado esquerdo de alguma das produções disponíveis.

Tabela 3. Derivações como leitura de uma palavra por um autômato.



III. GRAMÁTICAS LIVRES DE CONTEXTO ADAPTATIVAS

Uma gramática livre de contexto adaptativa é uma tripla **AG = (G, [P$_0$], A)**, onde:

- **G = (N, T, S, P)** é uma gramática livre de contexto;
- **P$_0$** é um subconjunto de **P** e **[P$_0$]** é o conjunto de produções adaptativas associados às produções de **P$_0$**;
- $A : P \times 2^P \rightarrow 2^P$ é uma função adaptativa.

Para definir as funções adaptativas, é necessário apresentar antes o conceito de *ação adaptativa elementar*. Estas ações permitem adicionar ou remover produções no conjunto de produções correntes, durante a derivação de uma palavra, e são definidas a seguir.

Definição

Dados $p \in P$ e $Q \subseteq P$, definimos a *ação adaptativa elementar de adição* (de produções), denotada com o símbolo "+", como:

$+(p, Q): P \times 2^P \rightarrow 2^P$

$+(p, Q) = Q \cup \{p\}$

e a ação *adaptativa elementar de remoção* (de produções), denotada com o símbolo "-", como:

$-(p, Q): P \times 2^P \rightarrow 2^P$

$-(p, Q) = Q \setminus \{p\}$

Nós denotamos +(p, Q), -(p, Q) abreviadamente com *eaa* (do inglês *elementar adaptive action*).

□

Com a definição anterior, definimos agora as funções adaptativas como seqüências de ações adaptativas elementares.

Definição

Dada uma gramática livre de contexto **G = (N, T, S, P)**, definimos uma *função adaptativa*, denotada com A, como:

$A: P \times 2^P \rightarrow 2^P$

que é definida por uma seqüência finita de "i" ações adaptativas elementares e é denotada como $A(\ ,\ ) = \{ eaa_h(\ ,\ ): 0 \leq h \leq i \}$, cujo valor é calculado como segue:

$A(p, Q) = eaa_i(p_i, Q_i)$

$Q_{h+1} = eaa_h(p_h, Q_h)$, onde $0 \leq h \leq i$.

$Q_0 = Q$

□

Observe que, na definição anterior, a ordem na qual aparecem as ações adaptativas elementares é importante: isso é uma restrição com respeito à definição original das gramáticas adaptativas ([2]).

A semântica para o cálculo do valor da função adaptativa pode ser expressada dizendo que, depois de aplicar a produção p_h, é calculado o novo conjunto de produções Q_h. Por outro lado, o segundo parâmetro Q, em A(p, Q), representa sempre o "conjunto de produções corrente" e esse conjunto é modificado durante o cálculo do valor da função e essas modificações devem ser levadas em consideração durante o cálculo do valor da função. Desse modo, o único parâmetro fixo é o primeiro. Por isso, escrevemos A(p) para denotar A(p, Q) , +[p] para +(p, Q) e -[p] para -(p, Q) e, se $p = X \rightarrow Y$, então escrevemos $[p] = [X \rightarrow A()Y]$.

A. Exemplo: Convertendo CFPG para CFAG.

Do algoritmo exposto no Lema 3.2.8 de [1], podemos obter a função adaptativa correspondente à gramática programada $\mathbf{G_p}$:

p	A(p)
p_0	{ -[p_0], -[p_3], +[p_1], +[p_4] }
p_1	{ -[p_1], +[p_2] }
p_2	{ -[p_2], +[p_3] }
p_3	{ -[p_0], -[p_3], +[p_1], +[p_4] }
p_4	{ -[p_4], +[p_5] }
p_5	{ -[p_5], +[p_6] }

O conjunto de produções iniciais é $P_0 = \{p_0\}$. Denotamos também $P_1 = \{p_1, p_4\}$, $P_2 = \{p_2\}$, $P_3 = \{p_3\}$, $P_4 = \{p_4\}$, $P_5 = \{p_5\}$ e $P_6 = \{p_6\}$.

B. Exemplo: Convertendo CFWRCL para CFAG.

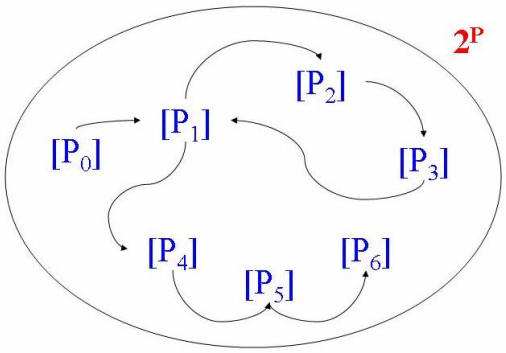
Do algoritmo exposto no Lema 3.2.4 de [1], podemos obter que a função adaptativa correspondente à gramática com linguagem de controle regular $\mathbf{G_{reg}}$ é:

p	A(p)
p_0	{ -[p_0], +[p_1], +[p_4] }
p_1	{ -[p_1], +[p_2] }
p_2	{ -[p_2], +[p_3] }
p_3	{ -[p_3], +[p_1], +[p_4] }
p_4	{ -[p_4], +[p_5] }
p_5	{ -[p_5], +[p_6] }

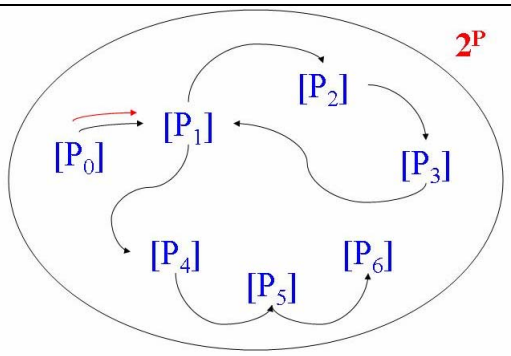
Neste caso, a função adaptativa pode ser considerada a função *follow* da expressão regular $p_0(p_1p_2p_3)^*p_4p_5p_6$.

Em ambos os casos, a derivação de uma palavra segundo uma gramática livre de contexto adaptativa pode ser vista como um passeio entre os subconjuntos P_0, P_1, P_2, P_3, P_4, P_5 e P_6 do conjunto de produções P, como mostra a Tabela 4.

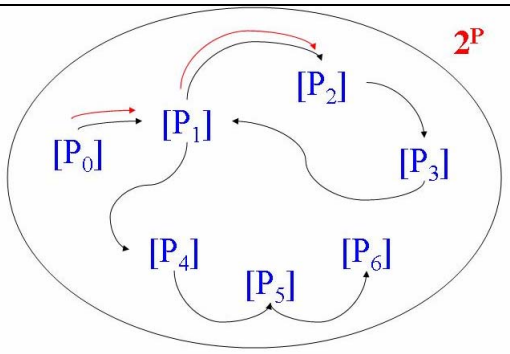
Tabela 4 Derivações como passeios entre subconjuntos de produções



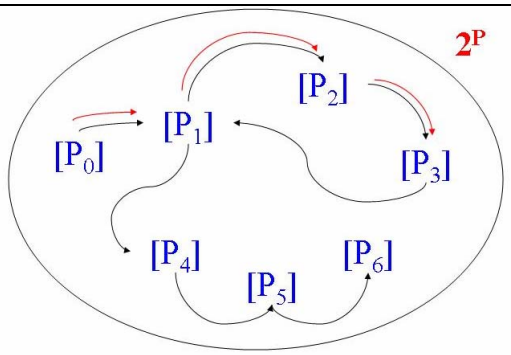
Derivation of $a^2b^2c^2$: .S



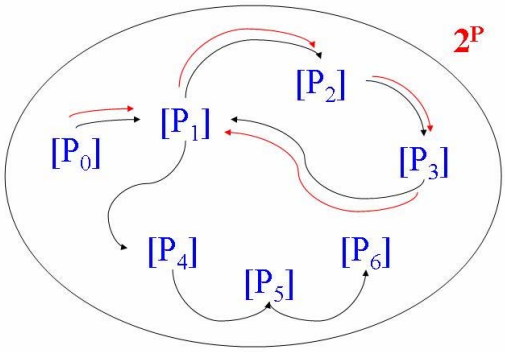
Derivation of $a^2b^2c^2$: .ABC



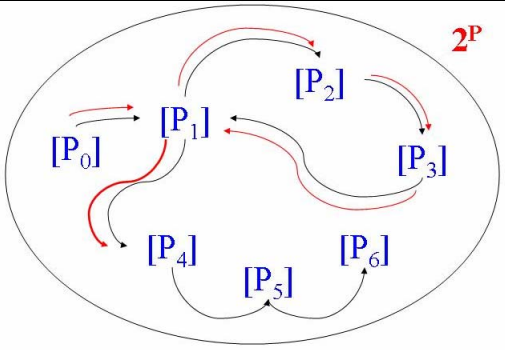
Derivation of $a^2b^2c^2$: aA.BC



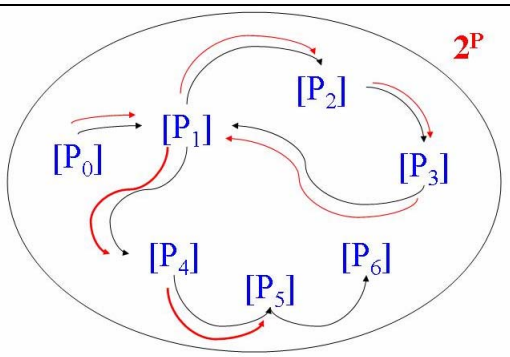
Derivation of $a^2b^2c^2$: aAbB.C



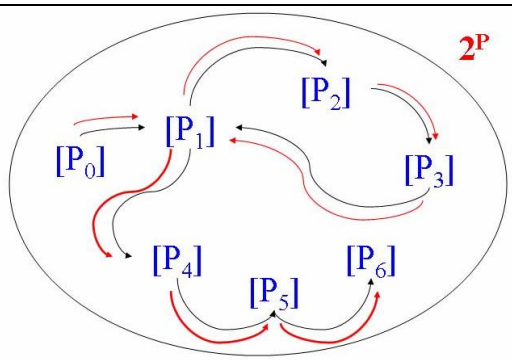
Derivation of $a^2b^2c^2$: a.AbBcC



Derivation of $a^2b^2c^2$: a^2b.BcC



Derivation of $a^2b^2c^2$: a^2b^2c.C



Derivation of $a^2b^2c^2$: $a^2b^2c^2$.

IV. AUTÔMATOS COM MECANISMOS DE CONTROLE

Embora existam diversos tipos de autômatos com mecanismos de controle similares aos descritos na seção II para gramáticas (por exemplo, *timed-automata*), nós vamos tratar somente de um tipo que corresponde às gramáticas variantes no tempo ([1], [3], [4]). Restringimo-nos a essa classe de maquinas porque elas possuem características promissoras para a obtenção, em relação aos autômatos adaptativos, de um resultado similar ao teorema da seção I.

De fato, nesse tipo de autômato, o conjunto de transições é modificado enquanto o autômato examina uma palavra, mas o conjunto de estados permanece fixo. Existem dois tipos de autômatos variantes no tempo: Autômatos (Finitos) Variantes no Tempo **TVA** (do inglês *Time Variant Automata* [6]) e Autômatos de Pilha Variantes no Tempo **TVPDA** (do inglês *Time Variant Pushdown Automata* [8]). Há ainda uma versão generalizada dos **TVA**, denominada **GTVA** (do inglês *Generalized Time Variant Automata* [7]), mas que não tem uma correspondência exata com o conceito original. A função de variação para os **TVA** é descrita a seguir:

$$\mathbf{v(i) = \delta_i : \ Q \times \Sigma \rightarrow Q, \ i = 0, 1, \ 2, ...}$$

onde **Q** é o conjunto de estados e **Σ** é o alfabeto do autômato. Evidentemente, um autômato (finito) adaptativo pode simular um **TVA**, definindo, para cada transição possível na máquina original, uma transição adaptativa com a função adaptativa a seguir:

$$\mathrm{A}(\mathbf{i}) = \{-[\mathbf{v(i-1)}], +[\mathbf{v(i)}] \}, \ \mathbf{i = 1, \ 2, ...}$$

onde -[**v(i-1)**] significa retirar todas as transições do tempo "**i-1**" e +[**v(i)**] significa adicionar todas as transições do tempo "**i**". Para o caso do **TVPDA**, basta levar em consideração a pilha para obter o autômato adaptativo correspondente.

Os resultados principais para essas classes de máquinas são resumidos a seguir:

- $\mathrm{L}_2 = \mathrm{L}$ (**PTVPDA**) $\subset \mathrm{L}$ (**TVPDA**)
- L (**TVDPDA**) $\subset \mathrm{L}$ (**TVPDA**) $= \mathrm{L}_0$
- L (**TVDPDA**) $\cap \mathrm{L}_1 \neq \varnothing$
- $\mathrm{L}_3 \subset \mathrm{L}$ (**TVA**), e L (**TVA**) $\cap \mathrm{L}_2 \neq \varnothing$
- L (**GTVA**) $= \mathrm{L}_0$

Na primeira dessas equações, a sigla **PTVPDA** denota **TVPDA**, onde a função de variação é periódica. Como temos um método algorítmico para obter um autômato adaptativo que reconhece a mesma linguagem que um **TVDA**, podemos agora enunciar o seguinte resultado.

Teorema 2

Qualquer linguagem livre de contexto pode ser reconhecida por um autômato adaptativo cujas funções adaptativas não usam geradores, nem requerem ações adaptativas elementares de consulta. As funções adaptativas trabalham sempre sobre os subconjuntos de um conjunto fixo de transições e o percurso sobre esses subconjuntos transições é de natureza periódica.

□

Embora à primeira vista possa parecer que a classe de autômatos adaptativos descritos pelo teorema anterior coincide com a classe dos **PDA**, isso implicaria que os percursos das transições dos **PDA**, na análise de uma palavra, sempre (para qualquer linguagem livre de contexto) possuem natureza periódica e esta afirmativa teria que ser provada.

Deve-se frisar que o resultado acima corresponde a máquinas não determinísticas em geral, o que imediatamente coloca as seguintes perguntas:

- Qual é a subclasse de autômatos adaptativos que corresponde à subclasse de **PTVPDA** que reconhece linguagens livres de contexto ambíguas?
- Qual é a subclasse de autômatos adaptativos que corresponde à subclasse de **PTVPDA** que reconhece linguagens **LL[k]**?
- Qual é a subclasse de autômatos adaptativos que corresponde à subclasse de **PTVPDA** que reconhece linguagens **LR[k]**?.

Na segunda e na terceira das equações anteriores, a sigla **TVDPDA** denota **TVPDA** determinísticos. É conhecido que a classe **TVPDA** tem interseção com a classe das linguagens dependentes de contexto ([8]), de modo que pode ser interessante estudar problemas clássicos de linguagens de programação (verificação de tipos, mudança de contexto, etc.) com máquinas dessa classe e seus correspondentes autômatos adaptativos.

V. CONCLUSÕES

Neste trabalho, foram resumidas as estratégias que permitiram criar uma versão simplificada de gramáticas adaptativas sem perda do poder computacional [1], e foi proposta uma estratégia similar, baseada em resultados para máquinas de estado, para obter uma simplificação análoga para autômatos adaptativos. Em particular, a estratégia foi aplicada com sucesso para o caso das linguagens livres de contexto, obtendo-se uma subclasse de autômatos adaptativos para reconhecer a classe L_2, os quais operam sempre sobre um conjunto fixo de estados e cujas funções adaptativas não fazem uso de funções adaptativas elementares de consulta, nem de geradores.

AGRADECIMENTOS

O autor agradece o convite do prof. Dr. João José Neto para ministrar a palestra na qual se baseia este texto durante o I WTA 2007, Workshop em Tecnologia Adaptativa, e para participar na publicação das memórias do mesmo.

REFERÊNCIAS

[1] C. Bravo, *Gramáticas Livres de Contexto Adaptativas com verificação de aparência. PhD dissertation*. EPUSP 2004. Disponível em: http://www.teses.usp.br/teses/disponiveis/3/3141/tde-31032005-134438/publico/CESARPARIENTE.pdf

[2] M. Iwai. *Um formalismo Gramatical Adaptativo para Linguagens Dependentes de Contexto. PhD dissertation*. EPUSP 2000. Disponível em: [Iw00]: http://www.pcs.usp.br/~lta/artigos/margarete_tese.zip

[3] J. Dassow; G. Paun; A. Salomaa. Grammars with controlled derivations. In Rozenberg, Salomaa A., editors, *J Handbook of Formal Languages*, pages 101–154. Springer-Verlag, 1997.

[4] A. Salomaa, *Formal Languages*. Academic Press, 1973.

[5] J. Dassow; G. Paun. *Regulated Rewriting in Formal Language*. EATCS Monographs on Theoretical Computer Science, Vol. 18, 1989. Springer-Verlag.

[6] A. Salomaa. On Finite Automata with a Time-Variant structure. *Information and Control, 13, pp. 85-98, 1968*.

[7] K. Krithivasan. Time Variant Finite Automata. *International Journal of Computer Mathematics, Vol. 19, pp. 103-123, 1986*.

[8] K. Krithivasan. Time Variant Pushdown Automata. *International Journal of Computer Mathematics, Vol. 24, pp. 223-236, 1988*.

César Bravo graduou-se em Pesquisa Operacional na "Facultad de Ciencias Matemáticas de la Universidad Nacional Mayor de San Marcos", em 1989. Recebeu o título de mestre pelo Instituto de Matemática e Estatística da Universidade de São Paulo em 1996. Recebeu o titulo de doutor pela Escola Politécnica da Universidade de São Paulo em 2004.