PROGRAMAÇÃO DINÂMICA E MÉTODO GULOSO

LAIRA VIEIRA TOSCANI*
PAULO AUGUSTO S. VELOSO**

RESUMO

A Programação Dinâmica e o Método Guloso são dois métodos de desenvolvimento de algoritmos. Neste trabalho é provado que qualquer problema que pode ser resolvido pelo Método Guloso, pode também ser resolvido por Programação Dinâmica.

ABSTRACT

Dynamic Programming and Greedy Method are methods for developing algorithms. This paper proves that any problem that can be solved by the Greedy Method can also be solved Dynamic Programming.

* Mestre em Informática (PUC-RJ, 73); desenvolvimento e complexidade de algoritmos; Profa. adjunta da UFRGS do Departamento de Informática e Curso de Pós-Graduação em Ciência da Computação, Caixa Postal 1501, 90001, Porto Alegre, RS.

**Ph. D. em Ciência da Computação (Univ. da Califórnia - Berkeley, 75); teoria e metodologia de programação; Prof. associado na PUC/RJ; Deptº de Informática; Rua Marquês de São Vicente, 225, 22453, Rio de Janeiro, RJ.

1. INTRODUÇÃO

Os métodos de desenvolvimento de algoritmos estudados neste trabalho são a Programação Dinâmica e o Método Guloso. A Programação Dinâmica é um método ascendente, que usa a solução de problemas menores para solucionar problemas maiores, enquanto o Método Guloso é especialmente empregado em problemas cuja solução pode ser atingido a partir de uma seqüência de decisões. A cada passo, pelo Método Guloso é escolhido um elemento da entrada e decidido se este elemento faz parte da solução ou não. Um mesmo elemento não é reexaminado, assim, após uma seqüência de decisões é atingida a solução.

Serão apresentados formalizações para os dois métodos, consistindo de um programa abstrato e um conjunto de axiomas. Tomado um problema resolvível pelo Método Guloso e uma instância do método que o resolve é definido uma instância da Programação Dinâmica que resolve também o problema. Assim, será provada uma idéia intuitiva que se tinha que o Método Guloso é um subcaso da Programação Dinâmica.

2. DEFINIÇÕES

Nesta secção serão apresentadas as definições necessárias ao desenvolvimento do trabalho.

De acordo com Veloso em [VEL 83], um problema é uma terna $p = \langle D,R,q \rangle$, onde D é o domínio de dados, R domínio de resultados e q uma relação de D em R que define o problema. A solução de um problema $p = \langle D,R,q \rangle$ é uma função $\alpha: D \to R$ tal que $\forall d \in D$ $(d, \alpha(d)) \in q$.

E algoritmo, segundo Knuth em [KNU 69] é um método abstrato de computar uma função. No caso a função é a solução α. Assim, dada um problema $p = \langle D,R,q \rangle$ e conhecida uma solução α para p, um algoritmo a_α é um método abstrato que computa α.

Um método de desenvolvimento de algoritmos (mda) é um par (Prog, Axio), onde Prog é um programa abstrato definido a partir de funções sintáticamente bem definidos e Axio é um conjunto de axiomas que definem a semântica dessas funções.

Se m = (Prog, Axio) é um mda, uma instância de m é uma instância de Prog que satisfaz Axio. Se p é um problema e α é solução de p, então $i(m,p,\alpha)$ é o conjunto de todas instâncias de m (algoritmos a_α) que computam α. E D(m) domínio de m é o conjunto de todos pares (p,α) tal que p é um problema, α é solução de p e $i(m,p,\alpha) \neq \Phi$.

Se m_1 e m_2 são dois mda's, diz-se que $m_1 \sqsubset m_2$ sss (se $(p,\alpha) \in D(m_1)$ então $\exists\ \alpha'$ t.q. $(p,\alpha) \in D(m_2)$).

Muitas vezes, dado um problema é preciso prepará-lo ou transformá-lo para então, através de um mda projetar um algoritmo que resolva o problema modificado e a partir dessa solução atingir uma solução do problema original. A esta fase do processo de construção de uma solução para um problema chama-se redução.

A redução foi definida em [VEL 84] como: dados dois problemas p = <D,R,q> e p'=<D',R',q'>, uma redução Γ de p em p' é um par de funções <t,v> tal que t:D → D' e v: D' x R' → R. Se diz que Γ é uma redução boa sss para qualquer solução α':D' → R de p', α: D → R definida como $\alpha(d) = v(t(d), \alpha'(t(d))$ é uma solução de p.

Os mda's Método Guloso e Programação Dinâmica requerem uma preparação do problema, uma redução portanto.

Serão usadas as seguintes operações entre funções: (fog)(x) = f(g(x)), (fxg)(x,y) = (f(x), g(y)) e (f,g)(x) = (f(x), g(x)).

3. PROGRAMAÇÃO DINÂMICA

Para resolver um problema por Programação Dinâmica é necessário uma redução $<t_1, v_1>$ tal que t_1 calcula o tamanho do problema, o decompõe em problemas triviais cuja solução é obtida facilmente e resolve-os (t_1=(decompõe, inicializa o decompõe, tamanho)) e v_1 é uma função que recupera a solução de uma variável m dada como saída do programa.

A Programação Dinâmica é definida pelo par (PROG.DINÂMICA, PDAX) como segue:

Sejam Q conjunto de seqüência de problemas, M conjunto de seqüências de soluções.

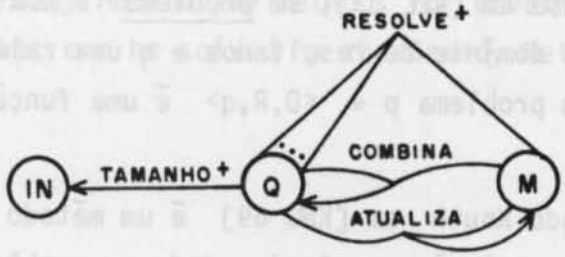

Fig. 1: Diagrama Sintático da Programação Dinâmica

resolve$^+$ é um predicado de Q^2 x M tal que: resolve$^+$ $((p_1, \ldots, p_k), (p'_1, \ldots, p_{k'})$ $(r_1, \ldots, r_{k'}) := p_1, \ldots, p_k$ são problemas triviais, $p'_1, \ldots, p'_{k'}$ foram obtidos a partir de $p_1, \ldots, p_k$ e r_j é solução de p'_j para j = 1, ..., k'.

Programa: PROG.DINÂMICA

ψ(p,m,n) := resolve$^+$(p,p,m)

entrada: p,m,n

1. p' ← p
2. mínimo ← tamanho$^+$(p)
3. para k = mínimo até n-1 faça
4. (p',m) ← (combina, atualiza) (p',m)
5. fim-para

saída: p',m

ψ(p,n,p',m) := resolve$^+$(p,p',m) tamanho$^+$(p') = n

Dada uma seqüência de problemas p de tamanho mínimo, a cada iteração da malha das linhas 3 a 5 é criado (função combina) uma nova seqüência de problemas de tamanho k+1, resolvida (função atualiza) a seqüência de problemas e os resultados guardados em m.

ψ e Ψ são respectivamente as asserções de entrada e saída do programa a função t_1 deve garantir ψ, enquanto Ψ e a função v_1 garantem a solução do problema.

Axiomas: PDAX

APD1: $(\forall p)(\forall p')\ \{resolve^+(p,p',m) \rightarrow resolve^+(p,combina(p',m),$
$atualiza(p',m))\}$

APD2: $(\forall p)(\forall m)\ tamanho^+(combina(p,m)) = tamanho^+(p)+1$

APD1 diz que (combina, atualiza) mantêm a condição de solução de problemas e APD2 que a cada combinação de problemas (função combina) o tamanho dos problemas aumentam de 1.

4. Método Guloso

Antes de se aplicar o Método Guloso a um problema, em geral é necessário classificar os dados para que seus elementos obedeçam uma certa ordem, que permita que a cada passo seja selecionado um elemento e seja possível no momento verificar a viabilidade de sua presença na solução do problema. A redução utilizada é um par $<t_2, v_2>$ onde t_2 é uma função de classificação e v_2 pode ser a função identidade. O Método Guloso é definido pelo par (GULOSO, GUAX), onde GULOSO é um programa abstrato e GUAX um conjunto de axiomas definidos como segue.

Sejam P conjunto de problemas, S conjunto de soluções e E conjunto de elementos da entrada ($E^+ = P$).

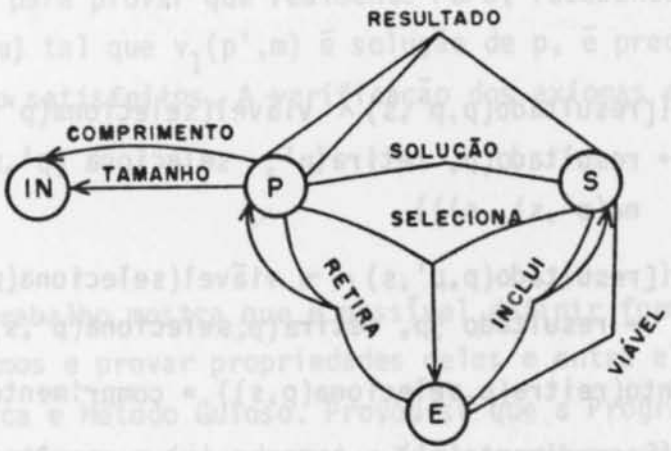

Fig. 2: Diagrama Sintático do Método Guloso

Programa: GULOSO

$\psi(p)$:= comprimento(p)=0 $\wedge$ resultado (p,p,$\emptyset$)

entrada: p

1. s $\leftarrow \emptyset$
2. n $\leftarrow$ tamanho(p)
3. p'$\leftarrow$p
4. para k=0 até n-1 faça

   invariante: resultado (p,p',s) $\wedge$ comprimento(p')=k
5. e $\leftarrow$ seleciona(p',s)
6. p' $\leftarrow$ retira (p',e)
7. se viável (e,s) então
8. s $\leftarrow$ inclui (e,s)
9. fim-se
10. fim-para

saída: s

$\Psi(p,s)$: solução(p,s)

A cada iteração da malha das linhas 4 a 9 é selecionado um elemento da entrada e se este elemento é viável é incluido na solução parcial.

Funções e Predicados auxiliares:

- solução: PxS $\rightarrow$ {T,F} é a condição de solução do problema, i é solução(p,s) :=s é solução de p
- resultado: PxPxS $\rightarrow$ {T,F} é a condição de solução parcial, i.é resultado (p,p',s) := ($\exists$s') [solução(p,s') $\wedge$ s = {x/x $\in$ (p-p') $\wedge$ x $\in$ s'}]
- comprimento: P $\rightarrow$ IN dá o número de elementos da entrada que já foram considerados.

Axiomas: GUAX

AG1:($\forall$p)($\forall$p')($\forall$s){[resultado(p,p',s) $\wedge$ viável(seleciona(p',s),s)] $\rightarrow$ resultado(p, retira(p', seleciona (p',s)),inclui(seleciona(p',s), s))}

AG2.($\forall$p)($\forall$p')($\forall$s){[resultado(p,p',s) $\wedge$ $\neg$ viável(seleciona(p',s),s)] $\rightarrow$ resultado (p, retira(p',seleciona(p',s)), s)}

AG3.($\forall$p) comprimento(reitra(p,seleciona(p,s)) = comprimento(p) + 1

AG4.($\forall$p)($\forall$p')($\forall$s){[comprimento(p') $\geq$ tamanho (p) $\wedge$ resultado (p,p',s)] $\rightarrow$ solução(p,s)}

Os axiomas AG1 e AG2 dão as condições de solução parcial, AG3 define o comportamento da função comprimento e o axioma AG4 dá a condição de solução.

5. MÉTODO GULOSO ⊏ PROGRAMAÇÃO DINÂMICA

Chame m_1=(GULOSO,GUAX) e seja $(p,\alpha) \in D(m_1)$. Então p é um problema resolvível pelo Método Guloso e α uma solução de p ($i(m_1,p,\alpha) \neq \emptyset$). Seja $GU \in i(m_1,p,\alpha)$, GU é uma instância de m_1, que encontra a solução α, para p. GU = (PGGU,AXGU).PGGU é uma instância de GULOSO (obtida pela instanciação das funções: tamanho, seleciona, retira. inclui e do predicado viável) enquanto AXGU é uma instância de GUAX (obtida pelas instanciações de comprimento e resultado, além das instanciações estabelecidos em PGGU).

Seja m_2 = (PROG.DINÂMICA, PDAX). Se quer mostrar que $m_1 \sqsubset m_2$. Assim é preciso definir $PD \in i(m_2,p,\alpha')$, onde α' é uma solução de p. Então PD = (PGPD, AXPD) é uma instância de m_2. PGPD é instância de PROG.DINÂMICA obtida pela instanciação das funções tamanho$^+$, combina e atualiza,e AXPD uma instância de PDAX obtida pela instanciação das funções acima como em PGPD e da instanciação do predicado resolve$^+$.

Considere as seguintes instanciações:

- Q := P, M := S
- combina := retira o (IPxπS, seleciona), onde $\pi X: X \to \{\Lambda\}$ e IX a função identidade em X
- atualiza := se viável o (seleciona, πPxIS)
  então inclui o (seleciona, πPxIS)
  senão (πP,IS)
- tamanho$^+$:= comprimento
- resolve$^+$:= resultado

Redução:

- t_1 := (decompõe, inicializa o decompõe, tamanho), onde:
  decompõe := t_2; inicializa := $\leftarrow\emptyset$; tamanho := tamanho
  v1 := IS

Agora, para provar que realmente PGPD, recebendo $t_1(p)$ como entrada para dando a saída (p',m) tal que $v_1(p',m)$ é solução de p, é preciso mostrar que os axiomas AXPD e $\psi(t_1(p))$ são satisfeitos. A verificação dos axiomas e ψ pode ser facilmente obtida.

6. CONCLUSÕES

Este trabalho mostra que é possível definir formalmente métodos de desenvolvimento de algoritmos e provar propriedades deles e entre eles, como é o caso dos mda's Programação Dinâmica e Método Guloso. Provou-se que a Programação Dinâmica é pelo menos tão geral quanto o Método Guloso, pois qualquer problema resolvível pelo segundo é também resolvido pelo primeiro.

Outros resultados nessa linha estão sendo pesquisados, como Programação Di

nâmica ⊏ Divisão e Conquista e propriedades referentes a complexidade de algoritmos gerados por mda's, publicados em [TOS 85], [TOS 86] e [TOS 87].

BIBLIOGRAFIA

[KNU 83] KNUTH, D,E. Algorithm and program; information and data. CACM 26(1), Jan. 83, p.56.

[TOS 85] TOSCANI, L.V. & VELOSO, P.A.S. Uma especificação formal para programação dinâmica. In: Congresso da SBC, 5., Porto Alegre, 1985. Anais. p.477-86.

[TOS 86] TOSCANI, L.V. & VELOSO, P.A.S. Divisão e conquista: análise da complexidade. In: Congresso da SBC, 6., Olinda, 1986. Anais. p.89-104.

[TOS 87] TOSCANI, L.V. & VELOSO, P.A.S. Análise da complexidade de Programas Abstratos. In: Congresso da SBMAC. 10, Gramado, 1987. Anais. p.978-83.

[TOS 88] TOSCANI, L.V. & VELOSO, P.A.S. Métodos de desenvolvimento de algoritmos: estudo comparativo. In: Congresso do SBMAC, 11., Ouro Preto, 1988. Anais.

[VEL 81] VELOSO, P.A.S. & VELOSO, S.R.M. Problem decomposition and reduction: applicability, soundness, completeness. In: Progess in Cybernetics and Systems Research, R. Trappel, J. Klir and F. Pichler, eds. vol. 8 Hemisphere, Washington, 1981.

[VEL 83] VELOSO, P.A.S. & LOPES, M.A. Problem solvability via Homomorphism and analogy. In: Proc. 10th Internat. Congr. Cybernetics Assoc. Internat. Cybernetique, Namur 1983.

[VEL 84] VELOSO, P.A.S. Outlines of mathematical theory of general problems. Philosophia Naturalis, vol. 21 (nº 2-4). p354-67.

[VEL 88] VELOSO, P.A.S. Problem solving by interpretation of theories. Contemporary Mathematics. vol. 69. 1988. p.241-9.