Imprimir Fechar

Referência Completa

Tipo da Referência	Conference Proceedings
Repositório	sid.inpe.br/iris@1905/2005/07.27.01.11
Metadados	sid.inpe.br/iris@1905/2005/07.27.01.11.27
Site	mtc-m05.sid.inpe.br
Rótulo	5236
Chave Secundária	INPE-3645
Chave de Citação	YanasseSoma:1985:AlExPr
Autor	1 Yanasse, Horacio Hideki 2 Soma, Nei Yoshihiro
Grupo	1 LAC-INPE-BR
Título	Um algoritmo exato para o problema da Mochila
Nome do Evento	Ermac, 2.
Ano	1985
Data	16-17 maio 1985
Localização do Evento	Sao Jose dos Campos
Palavras-Chave	COMPUTACAO APLICADA, ALGORITMOS.
Idioma	Pt
Tipo Secundário	PRE CN
Area	COMP
Projeto	120448
Ultima Atualização dos Metadados	2015:03.18.16.18.16 sid.inpe.br/bibdigital@80/2006/04.07.15.50 administrator
Estágio do Documento	concluido
e-Mail (login)	marciana
Grupo de Usuários	administrator
Visibilidade	shown
Transferível	1
Tipo do Conteudo	External Contribution
Unidade Imediatamente Superior	8JMKD3MGPCW/3ESGTTP
Conteúdo da Pasta source	não têm arquivos
Conteúdo da Pasta agreement	não têm arquivos
Histórico	2015-03-18 16:18:16 :: administrator -> marciana :: 1985
Campos Vazios	abstract accessionnumber affiliation archivingpolicy archivist booktitle callnumber copyholder copyright creatorhistory descriptionlevel dissemination documentstage doi e-mailaddress edition editor electronicmailaddress format isbn issn lineage mark mirrorrepository nextedition notes numberoffiles numberofvolumes organization pages parameterlist parentrepositories previousedition progress publisher publisheraddress readergroup readergroup readpermission resumeid rightsholder secondarydate secondarymark serieseditor session shorttitle size sponsor subject targetfile tertiarymark tertiarytype type url versiontype volume
Data de Acesso	24 jul. 2015

atualizar

Fechar

1. Publicação nº	2. Versão	3. Data	5. Distribuição
INPE-3645-PRE/815		*Set., 1985*	☐ Interna ☒ Externa
4. Origem *DIN/DEP*	Programa *POPES/INFOR*		☐ Restrita

6. Palavras chaves - selecionadas pelo(s) autor(es)

PROBLEMA DA MOCHILA
ALGORITMO PSEUDOPOLINOMIAL
ENUMERAÇÃO

7. C.D.U.: *519.87*

8. Título *INPE-3645-PRE/815*

UM ALGORITMO EXATO PARA O PROBLEMA DA MOCHILA

10. Páginas: *09*

11. Última página: *07*

12. Revisada por

L. Antonio N. Lorena

9. Autoria *Horacio Hideki Yanasse*
Nei Yoshihiro Soma

Assinatura responsável

13. Autorizada por

p/Marco Antonio Raupp
Diretor Geral

14. Resumo/Notas

Apresenta-se neste trabalho um novo esquema de enumeração para resolver o problema da mochila unidimensional. Este novo algoritmo é pseudopolinomial e, dadas suas características especiais, há possivelmente uma redução nos requisitos de memória computacional quando comparado com outros métodos exatos.

15. Observações *Este trabalho foi apresentado no 2º ERMAC, de 16 a 17 de maio de 1985, INPE, São José dos Campos-SP, e aceito para publicação em anais.*

UM ALGORITMO EXATO PARA O PROBLEMA DA MOCHILA

Horacio Hideki Yanasse
Nei Yoshihiro Soma

Instituto de Pesquisas Espaciais - INPE
Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq
C.P. 515 - 12200 - São José dos Campos - SP - Brasil

RESUMO

Apresenta-se neste trabalho um novo esquema de enumeração para resolver o problema da mochila unidimensional. Este novo algoritmo é pseudopolinomial e, dadas suas características especiais, há possivelmente uma redução nos requisitos de memória computacional quando comparado com outros métodos exatos.

ABSTRACT

This paper presents a new enumeration scheme to solve the one dimension knapsack problem. This new algorithm is pseudopolinomial and, given its special features, there is probably a reduction on the computational memory requirements when compared to other exact methods.

1 - INTRODUÇÃO

O problema da mochila (PM) considerado é da forma:

$$\text{Max } Z = \sum_{J=1}^{N} C(J)X(J),$$

$$\text{sujeito a: } \sum_{J=1}^{N} A(J)X(J) = B,$$

$$X(J) \in \mathbb{N}, \; J = 1, \ldots, N.$$

Admite-se que B, C(J) e A(J), J=1, 2,..., N sejam inteiros não-negativos. Esta não é uma condição muito restritiva, pois na maioria das situações práticas estes valores são inteiros não-negativos (Veja Kluyver e Salkin, 1975; Bulfin et alii, 1979). Sem perda de generalidade, admite-se que os A(J)s estejam em ordem crescente. Caso isto não aconteça, será necessário um ordenamento deles e isto requer O(N log N) operações.

O PM aparece em várias situações como podem ser vistas em Dantzig (1957), Lorie e Savage (1955), Chvátal (1980), Graham (1969), Gilmore e Gomory (1963), Harvey (1979), Onyekwelu (1983), Kannan (1983).

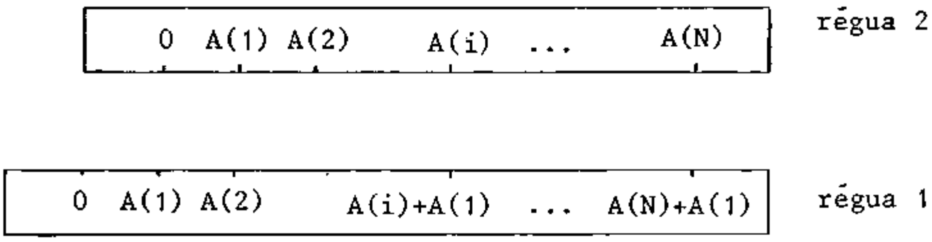
Algoritmos exatos para a solução do PM foram sugeridas por Gilmore e Gomory (1965), Gilmore e Gomory (1966), Horowitz e Sahni (1974), Martello e Toth (1979), Shapiro (1979), Denardo e Fox (1979) para citar apenas alguns. Algoritmos heurísticos também têm sido propostos, por exemplo, por Senju e Toyoda (1968), Martello e Toth (1984), Lawler (1979), Sahni (1975) e Ibarra e Kim (1975).

Os métodos exatos baseiam-se em técnicas tradicionais em programação inteira e programação dinâmica. O PM é NP-difícil (Garey e Johnson, 1979) sendo os melhores algoritmos existentes todos pseudopolinomiais. O melhor algoritmo existente, do conhecimento destes autores, é o proposto por Gilmore e Gomory (1965), que é O(NB) e requer O(B) de requisitos de memória.

Desenvolve-se aqui um novo esquema de enumeração que foge um pouco das abordagens tradicionais. Na Seção 2 apresenta-se a idéia que motivou a elaboração do novo algoritmo. Na Seção 3 são apresentados alguns comentários adicionais sobre este novo esquema.

2 - O ALGORITMO

Considere o princípio da régua de cálculo. Se ao invés da escala logarítmica houver escalas lineares, é possível somar-se quaisquer duas quantidades. Considere o esquema a seguir:



Ao fazer coincidir a marca 0 da régua 2 com a marca A(i) da régua 1, tem-se indicado pela marca A(j) da régua 2 o valor A(i)+A(j) na régua 1. Se no lugar destas marcas fosse carregado o valor C(i) correspondente, poder-se-ia levantar os custos associados a esta combinação. Usando esta idéia, formulou-se o algoritmo 1 para resolução do PM.

Algoritmo 1

PASSO 0: Inicialização

Construa uma lista Z(A(1)), Z(A(1)+1),..., Z(B-A(1)); Z(B) e faça:

$$Z(I) = \begin{cases} C(J) & \text{para } I=A(J), \quad J=1, 2,\ldots, N \\ -1 & \text{caso contrário.} \end{cases}$$

Comentários: Z(k) carrega o melhor valor objetivo encontrado até aquele momento para o lado direito igual a k. Se Z(k) é negativo para algum k no final do algoritmo, o problema é inviável para o lado direito igual a k.

Construa uma lista SOLIN(A(1)), SOLIN(A(1)+1),..., SOLIN(B-A(1)), SOLIN(B), e faça:

$$\text{SOLIN}(I) = \begin{cases} J & \text{se } I=A(J), \quad J=1, 2, \ldots, N \\ 0 & \text{caso contrário} \end{cases}$$

Comentários: SOLIN(k) guarda o índice da última variável incorporada para compor a solução viável, com o lado direito igual a k, que dá o melhor valor objetivo Z(k) encontrado até aquele momento.

Faça POINTER = A(1)

Comentário : POINTER é um ponteiro que guarda a posição da marca 0 da régua 2 relativo à régua 1.

PASSO 1: Faça para J=1 até N,

Se POINTER+A(J) ≤ B-A(1) on POINTER+A(J)=B, então ZLIN(POINTER+A(J))=Z(POINTER)+C(J). Se ZLIN(POINTER+A(J)) > Z(POINTER+A(J)), então Z(POINTER+A(J))=ZLIN(POINTER+A(J)), SOLIN(POINTER+A(J))= J.

Se POINTER+A(J) > B, então vá para o PASSO 2.

Comentário : ZLIN(k) guarda o valor objetivo relativo à solução viável obtida neste passo para o lado direito igual a k.

PASSO 2: POINTER= POINTER + 1

Se POINTER > B-A(1), então vá para o PASSO 3.

Se Z(POINTER) < 0, então vá para o PASSO 2, caso contrário, vá para o PASSO 1.

PASSO 3: Se Z(B) < 0, então o problema é inviável, PARE. Caso contrário, o valor ótimo está em Z(B).

A solução ótima pode ser obtida com os seguintes passos adicionais:

PASSO 4: Para I=1 até N, faça X(I)=0
POINTER=B

PASSO 5: Enquanto POINTER > 0, faça:

X(SOLIN(POINTER))= X(SOLIN(POINTER))+1
POINTER = POINTER - A(SOLIN(POINTER))

Comentário.: A solução ótima pode ser obtida imprimindo X(I), I=1,..., N.

Este algoritmo 1 é $O((B-A(1))\ N-\sum_{J=1}^{N} A(J))$ e requer $O(B-2A(1))+O(N)$ requisitos de memória computacional. Portanto, em geral, tem-se uma redução nos requisitos de memória comparado com o algoritmo de Gilmore e Gomory (1965).

Uma ilustração deste algoritmo, uma prova da finitude do algoritmo, a otimilidade da solução quando encontrada e o cálculo da ordem do algoritmo são apresentados em Yanasse e Soma (1985).

3 - COMENTÁRIOS

Neste novo esquema de enumeração, os valores dos C(J)s não afetam a ordem do algoritmo. A única restrição da função objetivo para a aplicabilidade do algoritmo é que ela seja aditiva.

Em Yanasse e Soma (1985) são feitos alguns comentários adicionais comparando este esquema com a abordagem de programação dinâmica e outros métodos. Baseado nesta mesma idéia de enumeração, Soma e Yanasse (no prelo) sugerem mais um algoritmo para o problema da equação linear diofantina.

4 - REFERÊNCIAS BIBLIOGRÁFICAS

BULFIN, R.; PARKER, R.; SHETTI, C. Computational results with a branch-and-bound algorithm for the general knapsack problem. *Naval Research Logistics Quarterly*, 26(1):41-46, 1979.

CHVÁTAL, V. Hard knapsack problems. *Operations Research*, 28(6),1402-1211, 1980.

DANTZIG, G.B. Discrete extremum problems. *Operations Research*, 5:266-277, 1957.

DENARDO, E.; FOX, B. Shortest-route methods: 2. Group Knapsack, expanded networks, and branch-and-bound. *Operations Research*, 27(3):548-566, 1979.

GAREY, M.R.; JOHNSON, D.S. *Computers and intractability - A guide to the theory of NP-completeness*. San Francisco. W.H. Freeman. 1979.

GILMORE, P.; GOMORY, R. A linear programming approach to the cutting stock problem II. *Operations Research*, 11(6):863-888, 1963.

——— Multistage cutting stock problems of two and more dimensions. *Operations Research*, 13:94-120, 1965.

——— The theory and computation of knapsack functions. *Operations Research*, 14:1045-1074, 1966.

GRAHAM, R.L. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*. 17:416-429, 1969.

HARVEY, C.M. *Operations research - An introduction to linear optimization and decision analysis*. Amsterdan, Elsevier North Holland, INC. 1979.

HOROWITZ, E.; SAHNI, S. Computing partitions with applications to the knapsack problem. *Journal of ACM*, 21:277-292, 1974.

IBARRA, O.H.; KIM, C.E. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the Association for Computer Machinery*, 22(2):463-468, 1975.

KANNAN, R. Polynomial - time aggregation of integer programming problems. *Journal of ACM*, 30(1):133-145, 1983.

KLUYVER, C.A.; SALKIN, H.M. The knapsack problem: A survey. *Naval Research Logistics Quarterly*, 2(1):127-144, 1975.

LAWLER, E.L. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4(4):339-356, 1979.

LORIE, J.; SÁVAGE, L. Three problems in capital rationing. *Journal of Business*, 28:229-239, 1965.

MARTELLO, S.; TOTH, P. The 0-1 knapsack problem. In: Christofides, N.; Mingozzi, A.; Toth, P.; Sandi, C. *Combinatorial Optimization*, New York, John Wiley and Sons, 1979.

MARTELLO, S.; TOTH, P. Worst case analysis of greedy algorithms for the subset - sum problem. *Mathematical Programming*, 28:198-205, 1984.

ONYEKWELU, D.C. Computational viability of a constraint aggregation scheme for integer linear programming problems. *Operations Research*, 31(4):795-861, 1983.

SAHNI, S. Approximate algorithms for 0/1 knapsack problem. *Journal of the Association for Computer Machinery*, 22(1):115-124, 1975.

SENJU, S.; TOYODA, Y. An approach to linear programming with 0-1 variables. *Management Science*, 15(4):196-207, 1968.

SHAPIRO, J. *Mathematical Programming: Structures and Algorithms*. New York, John Wiley, 1979.

SOMA, N.Y.; YANASSE, H.H. A pseudopolinomial time algorithm for linear diophantine equations - São José dos Campos, INPE in press.

YANASSE, H.; SOMA, N.Y. *A new enumeration scheme for the knapsack problem*. INPE - São José dos Campos (in press). To be presented at the School of Combinatorial Optimization on July 8-19, 1985. Rio de Janeiro, RJ.