Um Algortimo Genético para a o Problema da Mochila Compartimentada

Pedro Henrique Neves da Silva

18 de Dezembro de 2009

Introdução

O Problema da Mochila

- Desperta muito interesse devido a sua vasta gama de aplicações
- Surge como subproblema de inúmeros outros problemas
- Possui diversas variantes agrupadas no que podem ser chamadas de Classes de Problemas da Mochila
- Se caracteriza, basicamente, pela escolha de um subconjunto de itens que irá otimizar um objetivo
- Cada item deve possuir um "peso" e um "benefício" e a mochila deve possuir uma capacidade máxima
- Deseja-se escolher um subconjunto de itens que maximize o benefício da mochila, sem que a soma dos pesos dos itens selecionados ultrapasse a capacidade da mochila

Problema da Mochila

Classes de Problemas da Mochila

- Todas as mochilas que serão apresentados nessa seção utilizam um conjunto de itens S com n itens, cada item i $(1 \leq i \leq n)$ deve estar associado a um valor de benefício b_i e um peso l_i. A capacidade da mochila é representada por L.

Problema da Mochila

Problema da Mochila 0-1

- Para cada item $i \in S$ é associada a variável x_i que pode assumir um dos dois valores: 0 ou 1. x_i será 0 quando o item i não for escolhido para compor a mochila, e será 1 quando o item fizer parte da solução.

$$Maximizar \quad \sum_{i=1}^{n} b_i x_i \tag{1}$$

$$Sujeito\ a \quad \sum_{i=1}^{n} l_i x_i \leq L \tag{2}$$

$$x_i = 0\ ou\ 1,\ \ i = 1, ..., n$$

Problema da Mochila

Problema da Mochila Restrita

- Variáveis deixam de ser binárias e passam a indicar o número de repetições do respectivo item na mochila
- Cada variável tem associada a ela um limitante inferior (t_i) e um superior (d_i)

$$Maximizar \quad \sum_{i=1}^{n} b_i x_i \tag{3}$$

$$Sujeito\ a \quad \sum_{i=1}^{n} l_i x_i \leq L \tag{4}$$

$$t_i \leq x_i \leq d_i \ e\ inteiro, i = 1, ..., n$$

Problema da Mochila

Problema de Múltiplas Mochilas 0-1

- n conjuntos de itens, disjuntos entre si
- Conjunto com m mochilas, cada uma com capacidade L_j, $(1 \leq j \leq m)$
- Itens de um determinado conjunto podem ser atribuídos a, no máximo, uma única mochila

$$\text{Maximizar} \quad \sum_{j=1}^{m} \sum_{i=1}^{n} b_i x_{ij} \tag{5}$$

$$\text{Sujeito a} \quad \sum_{i=1}^{n} l_i x_{ij} \leq L_j, j = 1, ..., m \tag{6}$$

$$\sum_{j=1}^{m} x_{ij} \leq 1, i = 1, ..., n \tag{7}$$

$$x_{ij} = 0 \; ou \; 1, i = 1, ..., n, j = 1, ..., m$$

Problema da Mochila

Estratégias para Resolução de Mochilas

- Embora o Problema da Mochila 0-1 e suas variações sejam pertencentes à classe NP-difícil, muitos deles surgem em aplicações práticas na vida real ou como sub-problemas de outros problemas mais complexos e por isso justifica-se a tentativa de encontrar soluções exatas ou aproximadas, mesmo que isso custe muito tempo.

Problema da Mochila

Branch-and-Bound

- Funciona como o método de força bruta, porém, utiliza um mecanismo para diminuir o conjunto de configurações, descartando a verificação daquelas que não tenham chance de serem soluções ótimas.
- Para descartar configurações não promissoras, calcula-se um *limitante* que é o valor máximo (ou mínimo) que uma solução pode atingir a partir da configuração em questão. Se esse valor limitante não for satisfatório, essa configuração será descartada.

Problema da Mochila

Branch-and-Bound

- Seja s_c é o somatório dos pesos de todos os itens que compõem a configuração c
- Seja k o maior valor tal que $\sum_{j=i+1}^{k} s_j \leq L - s_c$
- Os itens de $i+1$ a k são os melhores itens que ainda cabem na mochila
- Para calcular o limite superior para c, consideramos a adição de todos esses elementos a c mais tudo o que for possível do item $k+1$

$$upper(c) = l_c + \sum_{j=i+1}^{k} l_j + (L - s_c - \sum_{j=i+1}^{k} s_j)(l_{k+1})/(s_{k+1}) \quad (8)$$

Problema da Mochila

Programação Dinâmica

- Os itens de S serão numerados como 1,2,3,...,n
- Para cada $k \in \{1, 2, ..., n\}$, define-se S_k como o subconjunto contendo itens S rotulados de 1 até k
- Teremos $B[0, w] = 0$ para cada $w \leq L$ e derivaremos a seguinte relação para o caso geral:

$$B[k, w] = \begin{cases} B[k-1, w] & : l_k > w \\ max\{B[k-1, w], B[k-1, w-l_k] + b_k\} & : l_k \leq w \end{cases}$$

Problema da Mochila

Algoritmo Genético

- Os algoritmos genéticos são uma família de modelos computacionais inspirados na evolução
- Algoritmos genéticos não garantem que a melhor solução será encontrada
- Indicados para encontrar soluções aproximadas para problemas de otimização difíceis que envolvem um grande número de variáveis
- Trabalham com descrições de entrada formadas por cadeias de bits de tamanho fixo
- Começa com uma população aleatória de cromossomos
- Essas estruturas são avaliadas e associadas a uma probabilidade de reprodução

Problema da Mochila

Algoritmo Genético

gene um ou mais símbolos do alfabeto

cromossomo conjunto de genes que corresponde a um indivíduo

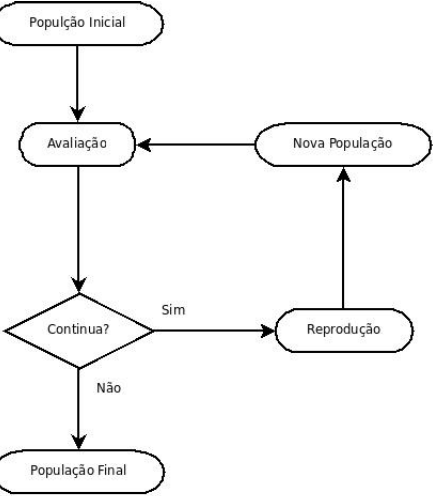
população conjunto de indivíduos que corresponde ao conjunto de pontos no espaço de busca

geração iteração completa do algoritmo genético que gera uma nova população

aptidão saída gerada pela função objetivo para um indivíduo da população.

Problema da Mochila

Fluxograma do Algoritmo Genético



Problema da Mochila

Algoritmo Genético

- **Cálculo de Aptidão** determinada por meio do cálculo da função objetivo
- **Fase de Seleção** os indivíduos mais aptos da geração atual são selecionados.
- **Fase de Cruzamento** um novo cromossomo é gerado permutando-se a partes de um cromossomo com partes do outro.
- **Fase de Mutação** utilizada para garantir uma maior varredura do espaço de busca.
- **Outros parâmetros** existem vários parâmetros que podem melhorar o seu desempenho.

Problema da Mochila

Algoritmo Genético

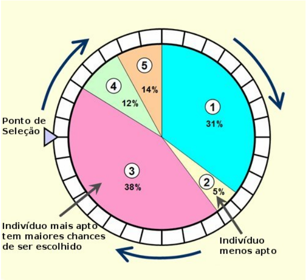


Figura: Exemplo de seleção por roleta.

Problema da Mochila Compartimentada

Definição

- É uma variação do Problema da Mochila
- Itens estão divididos em classes
- Mochila pode ser dividida em compartimentos com capacidades flexíveis
- Consiste em determinar a melhor distribuição dos itens em compartimentos, visando maximizar o benefício total, levando em consideração as capacidades máximas e mínimas de cada compartimento e a capacidade máxima da mochila

Problema da Mochila Compartimentada

Definição

- $M = \{1, ..., m\}$ conjunto dos tipos de itens;
- K quantidade de classes distintas (ou partições);
- C_k subconjunto de M, contendo itens de mesma classe, $k = 1, ..., K$ (para $i \neq j$, $C_i \bigcap C_j = \emptyset$);
- c_k custo de incluir um compartimento para itens da classe k na mochila $(c_k \geq 0)$, $k = 1, ..., K$;
- S perda decorrente da inclusão de um novo compartimento na mochila;
- L capacidade da mochila;
- N_k número total de possíveis compartimentos para a classe k;

Problema da Mochila Compartimentada

Definição

- L_{max} capacidade máxima de cada compartimento;
- L_{min} capacidade mínima de cada compartimento ($L_{min} < L_{max} < L$);
- l_i peso do item i ($l_i > 0$), $i = 1, ..., m$;
- b_i benefício ou utilidade do item i ($b_i \geq 0$), $i = 1, ..., m$;
- d_i limite máximo de itens i na mochila, $i = 1, ..., m$;
- α_{ijk} número de itens do tipo i, da classe k, no compartimento do tipo j ($i = 1, ..., m, k = 1, ..., K$ e $j = 1, ..., N_k$); e
- β_{jk} número de repetições do compartimento do tipo j alocados com a classe $k, k = 1, ..., K$ e $j = 1, ..., N_k$.

Problema da Mochila Compartimentada

Definição

- Capacidade ocupada dada por:

$$L_{jk} : \sum_{i \in C_k} l_i \alpha_{ijk}, \quad k = 1, ..., K \ e \ j = 1, ..., N_k. \tag{9}$$

- Valor de utilidade, ou benefício, dado por:

$$V_{jk} : \sum_{i \in C_k} b_i \alpha_{ijk}, \quad k = 1, ..., K \ e \ j = 1, ..., N_k. \tag{10}$$

Problema da Mochila Compartimentada

Definição

$$Maximizar: \quad \sum_{k=1}^{K} \sum_{j=1}^{N_k} (V_{jk} - c_l)\beta_{jk} \tag{11}$$

$$Sujeito\ a: \quad V_{jk} = \sum_{i \in C_k} b_i \alpha_{ijk} \tag{12}$$

$$L_{jk} = \sum_{i \in C_k} l_i \alpha_{ijk} \tag{13}$$

$$L_{min} \leq L_{jk} \leq L_{max} \tag{14}$$

Problema da Mochila Compartimentada

Definição

$$\sum_{k=1}^{K}\sum_{j=1}^{N_k}\alpha_{ijk}\beta_{jk} \leq d_i, i = 1, ..., m \tag{15}$$

$$\sum_{k=1}^{K}\sum_{j=1}^{N_k}(L_{jk} + S)\beta_{jk} \leq L \tag{16}$$

$$\alpha_{ijk} \geq 0,\ \ inteiro\ e$$

$$\beta_{jk} \geq 0,\ \ inteiro,$$

$$para\ i = 1, ..., m,\ \ k = 1, ..., K\ \ e\ \ j = 1, ..., N_k.$$

Problema da Mochila Compartimentada

Simplificações adotadas

- As simplificações adotadas não alteram a natureza do problema nem a sua complexidade computacional

- $N_K = 1$, ou seja, existe apenas 1 compartimento para a classe k;
- $L_{min} = 0$, ou seja, uma classe k de itens pode estar presente, ou não, em algum compartimento na mochila; e
- $0 \leq d_i \leq 1$, ou seja, cada item pode aparecer no máximo 1 vez na solução.

Problema da Mochila Compartimentada

Simplificações adotadas

$$Maximizar: \quad \sum_{k=1}^{K} (V_{jk} - c_l) \tag{17}$$

$$Sujeito\ a: \quad V_k = \sum_{i \in C_k} b_i \alpha_{ik} \tag{18}$$

$$L_k = \sum_{i \in C_k} l_i \alpha_{ik} \tag{19}$$

$$0 \leq L_{jk} \leq L_{max} \tag{20}$$

$$\sum_{k=1}^{K} (L_k + S) \leq L \tag{21}$$

$$\alpha_{ik} \geq 0,\ inteiro\ e$$

$$para\ i = 1, ..., m\ e\ k = 1, ..., K$$

Problema da Mochila Compartimentada

- Existem várias heurísticas para se resolver o Problema da Mochila Compartimentada
- Destacarei a heurística da decomposição e o uma possível implementação de um algoritmo genético para a mochila compartimentada do caso restrito.

Problema da Mochila Compartimentada

Heurística da Decomposição

- Consiste de duas fases:
  - Na primeira, são resolvidos $(K-1)$ Problemas da Mochila de capacidade L_{max}, um para cada agrupamento
  - Na segunda fase, um problema clássico da mochila é resolvido

$$Maximizar: \quad V_k = \sum_{i \in C_k} p_i \alpha_{ik} \tag{22}$$

$$Sujeito\ a: \quad \sum_{i \in C_k} l_i \alpha_{ik} + S \leq L_{max} \tag{23}$$

$$0 \leq \alpha_{ik} \leq d_i\ e\ inteiro,\ i = 1, ..., m.$$

Problema da Mochila Compartimentada

Heurística da Decomposição

$$Maximizar: \quad \sum_{k=1}^{K} (V_k - c_k)\beta_k \tag{24}$$

$$Sujeito\ a: \quad \sum_{k=1}^{K} L_k \beta_k \leq L - S \tag{25}$$

$$\alpha_{ik}\beta_k \leq d_i, \; i \in C_k \; e \; k = 1, ..., K$$

$$\beta_k \geq 0, \; inteiro \; para \; k = 1, ..., K.$$

Problema da Mochila Compartimentada

Algoritmo Genético

- Funciona como a heurística de decomposição
- Em cada indivíduo são computados os pesos e é verificado se, para cada compartimento, o valor obtido é menor que o limite do compartimento, depois é verificado se o peso total ultrapassa o limite da mochila
- Duas aptidões: uma para o peso e outra para o benefício
- Denominaremos de *razão de aproximação* a divisão do valor da solução ótima pelo valor retornado pelo algoritmo genético
- Quanto mais próximo de 1, mais próximo do valor ótimo será a resposta do algoritmo genético

Implementações e Resultados Computacionais

Problema da Mochila 0-1

- Força-bruta, programação dinâmica, branch-and-bound e algoritmo genético
- Parâmetros utilizados no algoritmo genético:
    - Número de gerações: 50.
    - Tamanho da população: 30.
    - Taxa de *cross-over*: 75%.
    - Taxa de mutação: 5%.

Implementações e Resultados Computacionais

Problema da Mochila 0-1

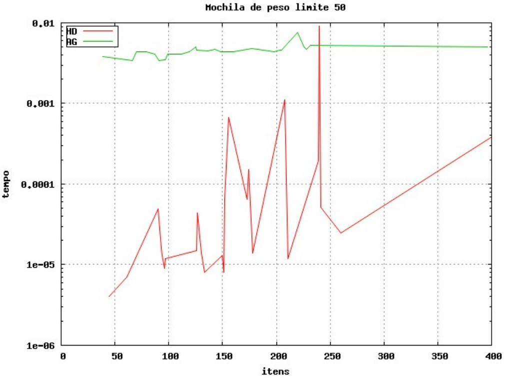


Figura: Gráfico comparativo dos tempos de execução dos quatro métodos: força bruta(FB), branch-and-bound(BB), programação dinâmica(PD) e algoritmos genéticos(AG), para mochilas com capacidade 50.

Implementações e Resultados Computacionais

Problema da Mochila 0-1

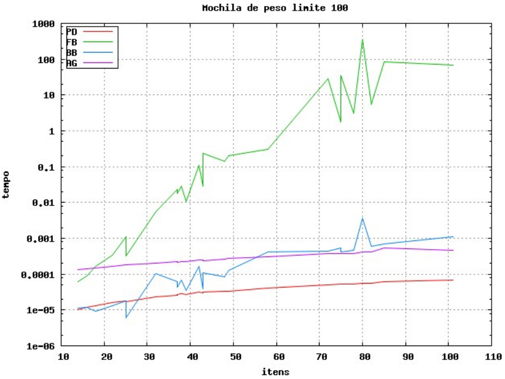


Figura: Gráfico comparativo dos tempos de execução dos quatro métodos: força bruta(FB), branch-and-bound(BB), programação dinâmica(PD) e algoritmos genéticos(AG), para mochilas com capacidade 100.

Implementações e Resultados Computacionais

Problema da Mochila 0-1

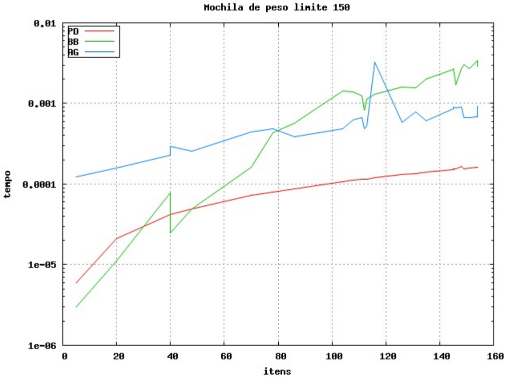


Figura: Gráfico comparativo dos tempos de execução dos três métodos mais rápidos: branch-and-bound(BB), programação dinâmica(PD) e algoritmos genéticos(AG), para mochilas com capacidade 150.

Implementações e Resultados Computacionais

Problema da Mochila 0-1

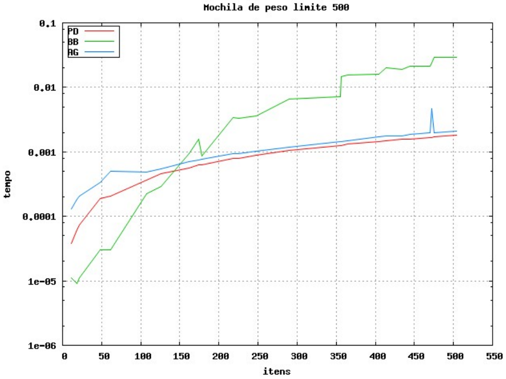


Figura: Gráfico comparativo dos tempos de execução dos três métodos mais rápidos: branch-and-bound(BB), programação dinâmica(PD) e algoritmos genéticos(AG), para mochilas com capacidade 500.

Implementações e Resultados Computacionais

Problema da Mochila 0-1

Figura: Gráfico comparativo dos tempos de execução dos três métodos mais rápidos: branch-and-bound(BB), programação dinâmica(PD) e algoritmos genéticos(AG), para mochilas com capacidade 1000.

Implementações e Resultados Computacionais

Problema da Mochila Compartimentada

- Heurística de Decomposição:
  - Utiliz $K-1$ chamadas ao algoritmo que utiliza força bruta para que escolher os melhores compartimentos
  - Faz uma última chamada ao mesmo algoritmo, para que sejam escolhidos os compartimentos que entraram na mochila
- Algoritmo Genético, com parâmatros:
  - Número de gerações: 50.
  - Tamanho da população: 30.
  - Taxa de *cross-over*: 75%.
  - Taxa de mutação: 5%.

Implementações e Resultados Computacionais

Problema da Mochila Compartimentada

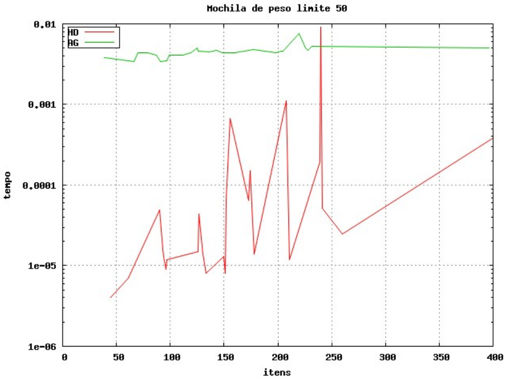


Figura: Gráfico comparativo dos tempos de execução dos dois métodos: heurística da decomposição(HD) e algoritmos genéticos(AG), para mochilas com capacidade 50.

Implementações e Resultados Computacionais

Problema da Mochila Compartimentada

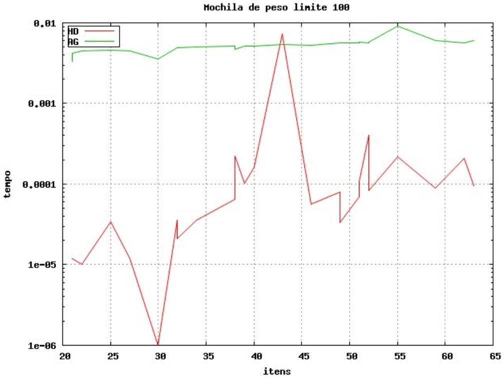


Figura: Gráfico comparativo dos tempos de execução dos dois métodos: heurística da decomposição(HD) e algoritmos genéticos(AG), para mochilas com capacidade 100.

Conclusão

- Para o Problema da Mochila 0-1, o algoritmo genético apresentou bons resultados
- A razão de aproximação obtida foi igual a 1,13
- Com relação ao Problema da Mochila Compartimentada, o algoritmo genético apresentou bom desempenho
- Porém não há como avaliar com precisão quão próxima da solução ótima está a resposta do algoritmo
- As soluções tem razão de aproximação próxima a 2.
- Os algoritmos genéticos tem apresentado boas razões de aproximação, porém não podemos dizer que sejam algoritmos aproximativos

Referências I

M. R. Garey and D. S. Johnson.
Computers and Intractability: A Guide to the Theory of NP-Completeness.
W. H. Freeman and Co, 1979.

Michael T. Goodrich and Roberto Tamassia.
Projeto de Algoritmos.
Bookman, 2002.

Robinson Hoto and Nelson Maculan Filho.
Um provável branch-and-bound para uma versão simplificada do problema da mochila compartimentada.
In *XXXII Simpósio Brasileiro de Pesquisa Operacional*, 2000.

R. Hoto, N. Maculan, F. Marques, and M. N. Arenales.
Um problema de corte com padrões compartimentados.
Pesquisa Operacional, 23:169–187, 2003.

Robinson Samuel Vieira Hoto, Fernando Luis Spolador, and F. Marques.
Resolvendo mochilas compartimentadas restritas.
XXXVII Simpósio Brasileiro de Pesquisa Operacional, 1:1756–1766, 2005.

Referências II

Rajeev Kumar, Ashwin H. Joshi, Krishna K. Banka, and Peter I. Rockett.
Evolution of hyperheuristics for the biobjective 0/1 knapsack problem by multiobjective genetic programming.
In *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 1227–1234, New York, NY, USA, 2008. ACM.

Fabiano do Prado Marques and Marcos Nereu Arenales.
O problema da mochila compartimentada e aplicações.
In *Pesquisa Operacional*, 2002.

Fabiano do Prado Marques.
O problema da mochila compartimentada.
Technical report, Instituto de Ciências Matemáticas e de Computação - ICMC-Usp, 2000.

Silvano Martello and Paolo Toth.
Knapsack Problems - Algorithms and Computer Implementatios.
John Wiley & Sons, 1990.

Fernando Luis Spolador.
O problema da mochila compartimentada.
Technical report, Universidade Estadual de Londrina, 2005.

Referências III

Thomas Weise.
Global optimization algorithms - theory and application, 2009.