Marcos Daniel V. Baroni

A Hybrid Heuristic for the Multi-objective Knapsack Problem

Vitória - Espírito Santo - Brasil
November 27, 2017

Marcos Daniel V. Baroni

A Hybrid Heuristic for the Multi-objective Knapsack Problem

Tese de Doutorado apresentada de acordo com o regimento do Programa de Pós-graduação em Informática da Universidade Federal do Espírito Santo.

Universidade Feredal do Espírito Santo – UFES
Departamento de Informática
Programa de Pós-Graduação em Informática

Advisor: Dr. Flávio Miguel Varejão

Vitória - Espírito Santo - Brasil
November 27, 2017

Marcos Daniel V. Baroni

A Hybrid Heuristic for the Multi-objective Knapsack Problem

Tese de Doutorado apresentada de acordo com o regimento do Programa de Pós-graduação em Informática da Universidade Federal do Espírito Santo.

Work approved. Vitória - Espírito Santo - Brasil, November 27, 2017:

Dr. Flávio Miguel Varejão
Advisor

Dr.ª Maria Claudia Silva Boeres
Member

Dr. Arlindo Gomes de Alvarenga
Guest

Dr.ª Simone de Lima Martins
Guest

Vitória - Espírito Santo - Brasil
November 27, 2017

Resumo

Many real applications like project selection, capital budgeting and cutting stock involves optimizing multiple objectives that are usually conflicting and can be modelled as a multi-objective knapsack problem (MOKP). Unlike the single-objective case, the MOKP is considered a NP-Hard problem with considerable intractability. This work propose a hybrid heuristic for the MOKP based on the shuffled complex evolution algorithm. A multi-dimensional indexing strategy for handling large amount of intermediate solutions are proposed as an optimization, which yields considerable efficiency, especially on cases with more than two objectives. A series of computational experiments show the applicability of the proposal to several types of instances.

Keywords: Multi-objective Knapsack Problem, Metaheuristic, Shuffled Complex Evolution, Multi-dimensional indexing

Contents

1 Introdução

Intro...

2 O Problema da Mochila Multi-objetivo

Em problemas reais é comum a existência de situações em que deseja-se otimizar mais de um objetivo os quais, geralmente, são conflitantes. Estes problemas são chamados multi-objetivos e tipicamente não possuem uma solução sendo a melhor em todos os objetivos, mas as possuem várias soluções de interesse chamadas *soluções eficientes*.

Um problema de otimização multi-objetivo com m objetivos pode ser descrito como uma função vetorial $f(x) = \big(f_1(x), \ldots, f_p(x)\big)$ para a qual deseja-se encontrar um vetor $x \in X$ que maximize simultaneamente as m funções objetivo. Formalmente:

$$\begin{aligned} \max\ f(x) &= \big(f_1(x), f_2(x), \ldots, f_m(x)\big) \\ \text{sujeito a } & x \in X \end{aligned}$$

Definição 1 (Dominância, Eficiência e conjunto Pareto)**.** *Considere um problema de otimização multi-objetivo. Diz-se que uma solução $x \in X$* domina *uma solução $y \in X$, denotado por $dom(x, y)$ se, e somente se, x é ao menos tão boa quanto y em todos os objetivos e melhor que y em ao menos um dos objetivos. Formalmente:*

$$dom(x, y) = \begin{cases} \forall i \in \{1, 2, \ldots, m\} : f_i(x) \geqslant f_i(y) \ \ e \\ \exists j \in \{1, 2, \ldots, m\} : f_j(x) > f_j(y) \end{cases}$$

Uma solução $x \in X$ é dita eficiente*, denotado por $eff(x)$, se, e somente se, x não é dominada por nenhuma outra solução pertencente a X. Formalmente:*

$$eff(x) \iff \nexists\big(y \in X \wedge dom(y, x)\big)$$

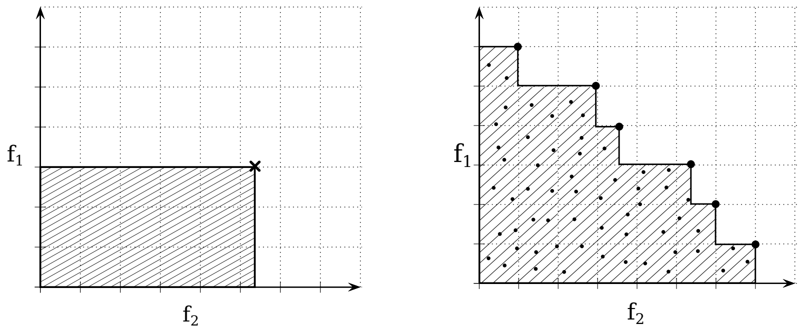
O conjunto de todas as soluções eficientes de um problema multi-objetivo, denotado por $Par(X)$, é chamado de conjunto Pareto *ou* conjunto Pareto-ótimo*. Formalmente:*

$$Par(X) = \{x \in X \mid eff(x)\}$$

Resolver um problema multi-objetivo consiste em determinar seu conjunto Pareto. Este conceito foi primeiramente elaborado por Vilfredo Pareto em 1896, que enunciou a relação Pareto-Ótima que diz: "não é possível melhorar uma característica do problema sem piorar outra", o que caracteriza a relação conflitante entre os objetivos na otimização multi-objetivo.

Na Figura 1a ilustra o conceito de dominância. A solução marcada domina todas as soluções existentes na área hachurada. As soluções em destacadas na Figura 1b formam um conjunto Paretopor dominarem sobre todas as outras soluções.

Um dos problemas multiobjetivos mais importantes da literatura é o problema da mochila multiobjetivo (MOKP). Muitas problemas reais podem ser modelados como uma



(a) Região de dominância de uma solução.

(b) Exemplo de conjunto Pareto.

Figura 1: Exemplos de solução dominante e conjunto Pareto.

instância do MOKP como seleção de projetos (TENG; TZENG, 1996), orçamento de capital (ROSENBLATT; SINUANY-STERN, 1989), carregamento de carga (TENG; TZENG, 1996) e planejamento de estoque (ISHIBUCHI; AKEDO; NOJIMA, 2015).

Comentar sobre a dificuldade de problemas MObj. Exploão do pareto com o aumento da quantidade de objectivos. Poucos métodos extados eficientes, geralmente utiliza-se métodos heurísticos.

O problema da mochila multi-objetivo pode ser descrito como uma função vetorial f que mapeia uma variável de decisão (solução) a uma tupla de m valores (objetivos). Formalmente:

$$\begin{aligned} \max\ y = f(x) &= \big(f_1(x), f_2(x), \ldots, f_m(x)\big) \\ \text{sujeito a } x &\in X \end{aligned}$$

onde x é a *variável de decisão*, X denota o conjunto de soluções viáveis e y representa o *vetor de objetivos* para os quais deseja-se maximizar.

Vale resaltar que o tamanho do conjunto Paretopara o problema em questão tende a crescer rapidamente com o tamanho do problema, especialmente com o número de objetivos.

Uma instância de um problema da mochila multi-objetivo (MOKP) com m objetivos consiste em uma capacidade inteira $W > 0$ e n itens. Cada item i possui um peso inteiro positivo w_i e m lucros inteiros $p_i^1, p_i^2, \ldots, p_i^m$ não negativos. O lucro p_i^k representa a contribuição do i-ésimo item para com o k-ésimo objetivo. Uma solução é representada por um conjunto $x \subseteq \{1, \ldots, n\}$ contendo os índices dos itens incluídos na mochila. Uma solução é viável se o peso total incluído na mochila não ultrapassa a capacidade da mochila. Formalmente a definição do problema é a seguinte:

$$\begin{aligned} \max\ f(x) &= \big(f_1(x), f_2(x), \ldots, f_m(x)\big) \\ \text{subject to } w(x) &\leqslant W \\ x &\in \{0,1\}^n \end{aligned}$$

where

$$\begin{aligned} f_j(x) &= \sum_{i=1}^{n} p_i^j x_i \quad j = 1, \ldots, m \\ w(x) &= \sum_{i=1}^{n} w_i x_i \end{aligned}$$

O MOKP é considerado um problema $\mathcal{NP}$-Hard visto set uma generalização do bem conhecido problema da mochila $0-1$, para o qual $m = 1$. É consideravelmente difícil determinar o conjunto Paretopara um MOKP, especialmente para vários objetivos. Até mesmo para casos bi-objetivos, problemas pequenos podem se apresentar intratáveis. Por este motivo interessa-se no desenvolvimento de métodos eficientes para manipular uma grande quantidade de soluções, o que pode eventualmente trazer tratabilidade a instâncias antes intratáveis.

A literatura contém várias propostas para resolver o MOKP de forma exata. Porém, nenhum método tem provado ser eficiente para grande instâncias com mais de dois objetivos. Mesmo para problemas bi-objetivo, algumas instâncias de tamanho considerado médio têm aprestando difculdades na determinação da solução exata, o que tem motivado o desenvolvimento de métodos heurísticas que buscam determinar um conjunto Paretoaproximado em tempo computacional razoável.

2.1 Métodos Exatos

Comentar sobre background de propostas de métodos exatos.

Comentar sobre o método da Bazgan como sendo considerado o melhor, citar melhorias propostas.

O algoritmo de Nemhauser e Ullmann é um algorimto de programação dinâmica que resolve problemas da mochila de forma genérica aplicando o conceito de dominância da mochila para remover soluções parciais que não resultarão em soluções eficientes, ou seja, soluções que irão compor o conjunto Pareto (conjunto solução).

O algoritmo inicia definindo uma solução inicial S^0 contendo apenas a solução vazia (linha 2). Na k-ésimo iteração o algoritmo recebe um conjunto S^{k-1} contendo soluções exclusivamente compostas pelos primeiros $k-1$ itens, ou seja, $\forall x \in S^{k-1}, x \subseteq \{1, \ldots, k-$

Algorithm 1 O algoritmo de Nemhauser e Ullmann para o MOKP.

1: **function** DP($\boldsymbol{p}, \boldsymbol{w}, W$)
2: $\quad S^0 = \left\{\emptyset\right\}$
3: $\quad$ **for** $k \leftarrow 1, n$ **do**
4: $\quad\quad S_*^k = S^{k-1} \cup \{x \cup k \mid x \in S^{k-1}\}$ $\quad$ ▷ extensão de soluções
5: $\quad\quad S^k = \{x \mid \nexists a \in S_*^k : dom_k(a, x)\}$ $\quad$ ▷ filtro de dominância parcial
6: $\quad$ **end for**
7: $\quad P = \{x \mid \nexists a \in S^n : dom(a, x) \mid w(x) \leqslant W\}$ $\quad$ ▷ dominância/viabilidade
8: $\quad$ **return** P
9: **end function**

1}. O conjunto S^{k-1} é então expandido adicionando-se uma cópia de cada uma das suas soluções mas desta vez incluindo o k-ésimo item (linha 4), formando o conjunto S_*^k, o qual possui o dobro da cardinalidade de S^{k-1}. O conjunto S_*^k é então reduzido, retirado-se todas as soluções que são dominadas (segundo a dominância da mochila) por alguma outra (linha 5). Após a conclusão das iterações um passo final remove as soluções inviáveis e também as dominadas por alguma outra, dessa vez porém, considerando apenas os valores de lucro (linha 7).

Apesar de sua simplicidade o Algoritmo 1 é consideravelmente poderoso. Contudo o potencial crescimento exponencial do conjunto Pareto para o MOKPcompromete severamente o seu desempenho. Uma forma de atacar este problema é tentar reduzir ainda mais a quantidade de soluções parciais manuseadas durante as iterações do algoritmo.

> Falar sobre as 3 propostas de redução de número de soluções da bazgan e justificar as definições seguintes.

> Explicar que o algoritmo Bazgan considera um conceito generalizado de dominância aplicado a cada iteração.

O processo sequencial executado pelo algoritmo de programação dinâmica consiste de n iterações. A cada k-ésima iteração é gerado o conjunto de estados S^k, que representa todas as soluções viáveis compostas de itens exclusivamente pertencentes aos k primeiros itens ($k = 1, \ldots, n$). Um estado $s_k = (s_k^1, \ldots, s_m^k, s_{m-1}^k) \in S_k$ represena uma solução viável que tem valor s_k^i como i-ésimo objetivo ($i = 1, \ldots, m$) e s_k^{m-1} de peso. Portanto, temos $S_k = S_{k-1} \cup \{(s_{k-1}^1 + p_k^1)\}$

> Comentar sobre as estratégias de redução dos conjuntos de estados, motivando as definições a seguir.

> Definir conjunto cobertura, conjunto independente, etc.

Definição 2 (Extensão, Restrição e Complemento)**.** *Considere o Algoritmo 1 e qualquer estado* $s_k \in S_K(k < n)$*. Um* complemento *de* s_k *é qualquer subconjunto* $J \subseteq \{k+1, \ldots, n\}$ *tal que* $s_k^{m+1} + \sum_{j \in J} w_j \leqslant W$*. Assumiremos que qualquer estado* $s_n \in S_n$ *admite o conjunto vazio como único complemento. Um estado* $s_n \in S_n$ *é uma* extensão *de* $s_k \in s_k(k \leqslant n)$ *se, e somente se, existe um complemento* J *de* s_k *tal que* $s_n^i = s_k^i + \sum_{j \in J} p_j^i)$*para* $i = 1, \ldots, m$ *e* $s_n^{p+1} = s_k^{p+1} + \sum_{j \in J} w_j$*. O conjunto de extenções de* s_k *é denotado por* $Ext(s_k)(k \leqslant n)$*. Um estado* $s_k \in S_k(k \leqslant n)$ *é uma* restrição *do estado* $s_n \in S_n$ *se, e somente se,* s_n *é uma extenão de* s_k*.*

Introdução às relações de dominância?

Definição 3 (Relação de dominância entre soluções)**.** *Uma relação* R_k *sobre* $S_k, k = i,\ ldots, n$*, é uma relação de dominância se, e somente se, para todo* $s_k, s_{k'} \in S_k$*,*

$$R_k(s_k, s_{k'}) \Rightarrow \forall s_{n'} \in Ext(s_{n'}), \exists s_n \in Ext(s_k), dom(s_n, s_{n'}) \tag{2.1}$$

Apesar das relações de dominância não serem transitivas por definição, costumam ser transitivas por construção, como é o caso das três relações de dominância da Seção 2.1.1. Vale notar que se $R_k^i,\ i = 1, \ldots, p$ são relações de dominância então $R_k = \bigcup_{i=1}^p R_k^i$ é também uma relação de dominância, geralmente não transitiva mesmo se $R_k^i,\ i = 1, \ldots, p$ forem transitivas.

Para se ter uma implementação eficiência do algoritmo de programação dinâmica é recomendável utilizar múltiplas relações de dominância $R_k^1, \ldots, R_k^p(p \leqslant 1)$ a cada execução da k-ésima iteração $k = 1, \ldots, n$ uma vez que cada relação R_k^i explora características específicas.

Algorithm 2 Algoritmo de programação dinâmica utilizando múltiplas relações de dominância.

function DP
    $C^0 \leftarrow \{(0, \ldots, 0)\}$
    for $k \leftarrow 1, n$ **do**
        $C_k^0 \leftarrow C_{k-1} \cup \left\{(s_{k-1}^1 + p_k^1, \ldots, s_{k-1}^m + p_k^m, s_{k-1}^{p+1} + w_k) \mid s_{k-1}^{p+1} + w_k \leqslant W, s_{k-1} \in C_{k-1}\right\}$
        for $i \leftarrow 1, p$ **do**
            Determinar um conjunto C_k^i cobertura do conjunto C_{k-1}^i com respeito a R_k^i
        end for
        $C_k \leftarrow C_k^p$
    end for
    return C_n
end function

Proposição 1. *Para quaisquer relações de dominância* $R_k^1, \ldots, R_k^p(p \leqslant 1)$ *sobre* S_k*, o conjunto* C_k^p *obtido pelo Algoritmo 2 em cada iteração é uma cobertura de* C_k^0 *com respeito a* $R_k = \bigcup_{i=1}^p R_k^i$ $(k = 1, \ldots, n)$*.*

Demonstração. Considere $s_k \in C_k^0 \backslash C_k^p$, este foi removido quando selecionado um conjunto cobertura na iteração da linha 6. Seja $i_1 \in \{1, \ldots, p\}$ a iteração da linha 6, tal que $s_k \in C_k^{i_1-1} \backslash C_k^{i_1}$. Uma vez que $C_k^{i_1}$ é um conjunto cobertura de $C_k^{i_1-1}$ com respeito a R_k^{i-1}, existe $s_{k'}^{(1)} \in C_k^{i_1}$ tal que... $\square$

2.1.1 As relações de dominância

Relações de dominância

2.2 Métodos Heurísticos

3 A k-d tree

4 Experimentos

5 Conclusão

References

ISHIBUCHI, H.; AKEDO, N.; NOJIMA, Y. Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 19, n. 2, p. 264–283, 2015.

ROSENBLATT, M. J.; SINUANY-STERN, Z. Generating the discrete efficient frontier to the capital budgeting problem. *Operations Research*, INFORMS, v. 37, n. 3, p. 384–394, 1989.

TENG, J.-Y.; TZENG, G.-H. A multiobjective programming approach for selecting non-independent transportation investment alternatives. *Transportation Research Part B: Methodological*, Elsevier, v. 30, n. 4, p. 291–307, 1996.

WEINGARTNER, H. M.; NESS, D. N. Methods for the solution of the multidimensional 0/1 knapsack problem. *Operations Research*, INFORMS, v. 15, n. 1, p. 83–103, 1967.