Sumário

Tese

É possível otimizar a resolução do problema da mochila multiobjetivo através da indexação multidimensional das soluções.

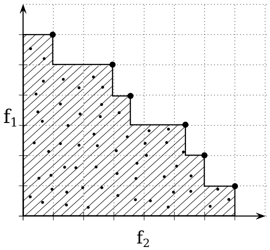
Introdução

- Introdução
- O problema da mochila multiobjetivo (MOKP)
- Algoritmos para o MOKP
- Indexação multidimensional para verificação de dominância
- Experimentos computacionais
- Conclusões e trabalhos futuros

Problemas de Otimização Multiobjetivo

- Otimização simultânea de múltiplos objetivos:

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



- Tipicamente mais de uma solução.

Problemas de Otimização Multiobjetivo

Definição (Dominância)

Diz-se que uma solução $x \in X$ ***domina*** *uma solução $y \in X$, denotado por $x \Delta y$ 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.*

Problemas de Otimização Multiobjetivo

Definição (Dominância)

*Diz-se que uma solução $x \in X$ **domina** uma solução $y \in X$, denotado por $x \Delta y$ 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:*

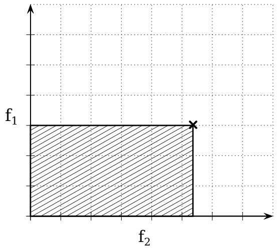
$$x \Delta y \iff \begin{cases} \forall i \in \{1, 2, \dots, m\} : f_i(x) \geqslant f_i(y) \; e \\ \exists j \in \{1, 2, \dots, m\} : f_j(x) > f_j(y) \end{cases}$$

Problemas de Otimização Multiobjetivo

Definição (Dominância)

*Diz-se que uma solução $x \in X$ **domina** uma solução $y \in X$, denotado por $x \Delta y$ 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:*

$$x \Delta y \iff \begin{cases} \forall i \in \{1, 2, \dots, m\} : f_i(x) \geqslant f_i(y) \text{ } e \\ \exists j \in \{1, 2, \dots, m\} : f_j(x) > f_j(y) \end{cases}$$



Problemas de Otimização Multiobjetivo

Definição (Eficiência)

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.*

Problemas de Otimização Multiobjetivo

Definição (Eficiência)

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:

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

Problemas de Otimização Multiobjetivo

Definição (Eficiência)

*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:

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

Definição (conjunto Pareto)

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

Problemas de Otimização Multiobjetivo

Definição (Eficiência)

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:

$$eff(x) \iff \nexists\big(y \in X \land y\Delta x\big)$$

Definição (conjunto Pareto)

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

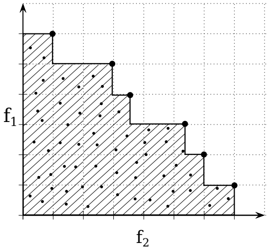
$$Par(X) = \{x \in X \mid eff(x)\}$$

Problemas de Otimização Multiobjetivo

Resolver um problema multiobjetivo consiste em determinar seu conjunto Pareto.

Problemas de Otimização Multiobjetivo

Resolver um problema multiobjetivo consiste em determinar seu conjunto Pareto.



O Problema da Mochila Multiobjetivo

Problema da mochila multiobjetivo (MOKP):

- Generalização do problema da mochila 0-1 ($\mathcal{NP}$-Hard);
- Bastante estudado pela literatura;
- Modela diversos problemas reais:
  - Seleção de projetos;
  - Orçamento de capital;
  - Planejamento de estoque, etc.
- De difícil resolução;
  - Especialmente para mais de 2 objetivos.

O Problema da Mochila Multiobjetivo

Definição formal:

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

onde

$$f_j(x) = \sum_{i=1}^{n} p_i^j x_i \quad j = 1, \ldots, m$$

$$w(x) = \sum_{i=1}^{n} w_i x_i$$

O Problema da Mochila Multiobjetivo

Exemplo de instância:

	Itens									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$\boldsymbol{p^1}$	4	9	3	1	8	7	2	5	6	7
$\boldsymbol{p^2}$	8	4	2	2	3	0	6	8	9	6
$\boldsymbol{w}$	7	8	5	8	3	5	6	2	4	9

$\boldsymbol{W}$	28

O Problema da Mochila Multiobjetivo

Exemplo de instância:

	Itens									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$\boldsymbol{p^1}$	4	9	3	1	8	7	2	5	6	7
$\boldsymbol{p^2}$	8	4	2	2	3	0	6	8	9	6
$\boldsymbol{w}$	7	8	5	8	3	5	6	2	4	9

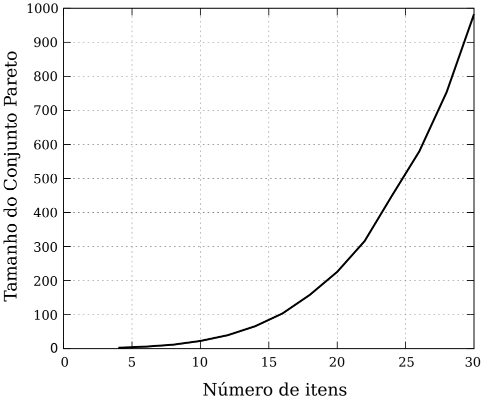
$\boldsymbol{W}$	28

Conjunto Pareto:

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	$\boldsymbol{f_1}$	$\boldsymbol{f_2}$	$\boldsymbol{w}$
$\boldsymbol{x_1}$	x						x	x	x	x	24	37	28
$\boldsymbol{x_2}$	x		x		x		x	x	x		28	36	27
$\boldsymbol{x_3}$	x				x	x	x	x	x		32	34	27
$\boldsymbol{x_4}$		x	x		x		x	x	x		33	32	28
$\boldsymbol{x_5}$		x			x	x	x	x	x		37	30	28
$\boldsymbol{x_6}$		x	x		x	x		x	x		38	26	27

O Problema da Mochila Multiobjetivo

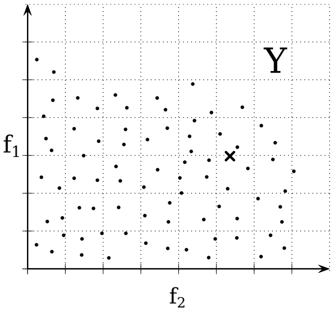
Tamanho do conjunto Pareto para instâncias do MOKP com 3 objetivos.



A operação de verificação de dominância

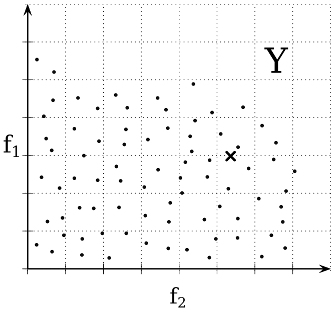
A operação de verificação de dominância

1. Exite alguma solução em Y que **é dominada** por x?



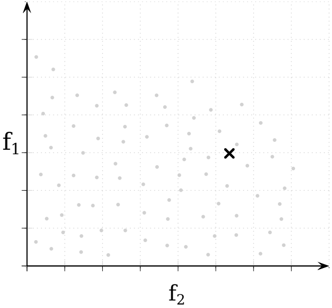
A operação de verificação de dominância

1. Exite alguma solução em Y que **é dominada** por x?



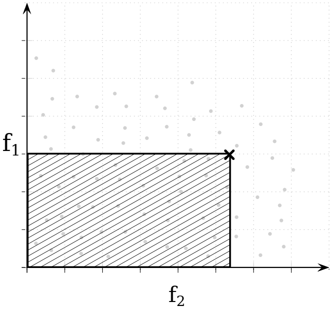
A operação de verificação de dominância

1. Exite alguma solução em Y que **é dominada** por x?



A operação de verificação de dominância

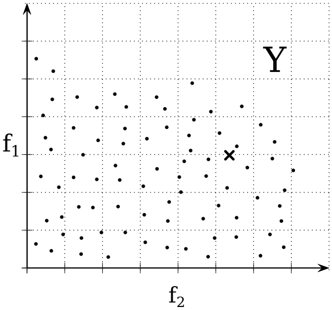
1. Exite alguma solução em Y que **é dominada** por x?



$$\exists(y \in Y)[\, x \Delta y \,]$$

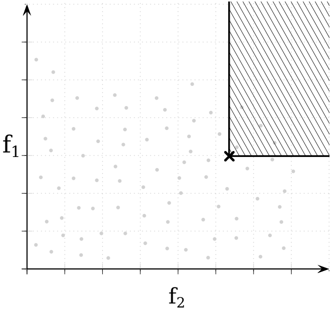
A operação de verificação de dominância

1. Exite alguma solução em Y que **é dominada** por x?
2. Exite alguma solução em Y que **domina** x?



A operação de verificação de dominância

1. Exite alguma solução em Y que **é dominada** por x?
2. Exite alguma solução em Y que **domina** x?



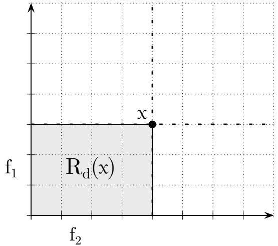
$$\exists(y \in Y)[\, y \Delta x \,]$$

A operação de verificação de dominância

A partir de x pode-se definir duas regiões de interesse:

A operação de verificação de dominância

A partir de x pode-se definir duas regiões de interesse:

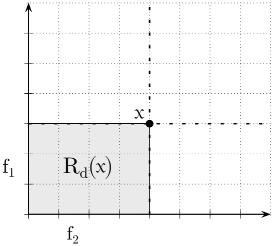


Região **dominada** por x.

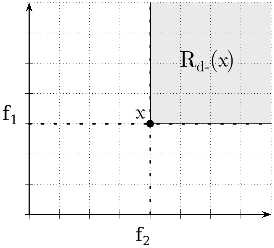
$$R_d(x) = \left\{ y \in \mathbb{R}^m \;\middle|\; y_i \leqslant f_i(x), i \in \{1, \dots, m\} \right\}$$

A operação de verificação de dominância

A partir de x pode-se definir duas regiões de interesse:



Região **dominada** por x.



Região que **domina** x.

$$R_d(x) = \left\{ y \in \mathbb{R}^m \;\middle|\; y_i \leqslant f_i(x), i \in \{1, \ldots, m\} \right\}$$
$$R_{d-}(x) = \left\{ y \in \mathbb{R}^m \;\middle|\; y_i \geqslant f_i(x), i \in \{1, \ldots, m\} \right\}$$

Busca de faixa

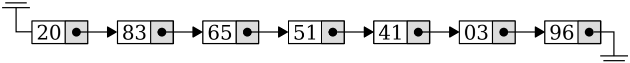
Estruturas de dados para conter soluções:

- Lista encadeada (sem indexação);
- Árvore AVL (unidimensional);
- Árvore KD (multidimensional).

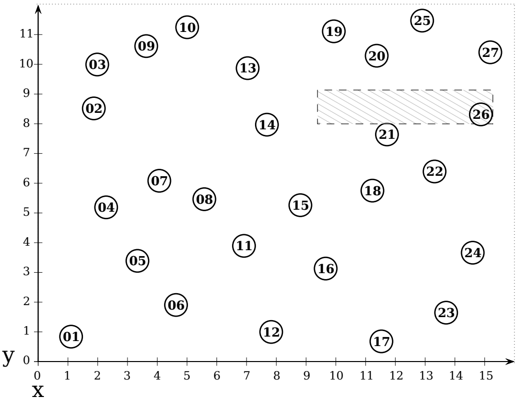
Lista encadeada

Lista Encadeada:

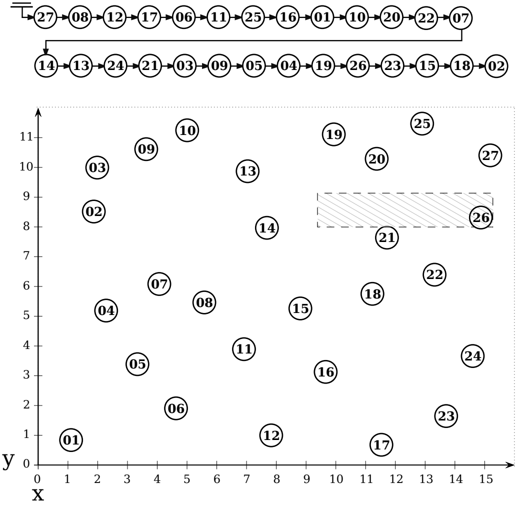
- Implementação simples ▲
- Pouca utilização de memória ▲
- Sem indexação – acesso em tempo linear ▼



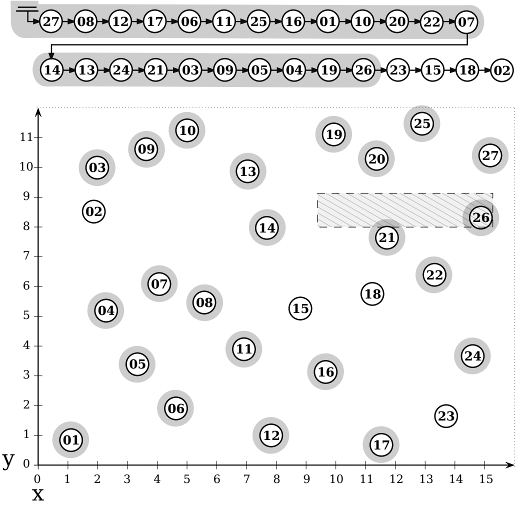
Lista encadeada



Lista encadeada



Lista encadeada



Árvore AVL

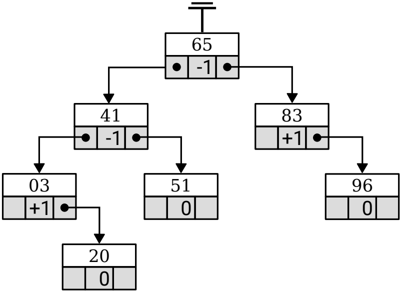
Árvore AVL:

- Implementação complexa ▼
- Pouca utilização de memória ▲
- Indexação unidimensional ●

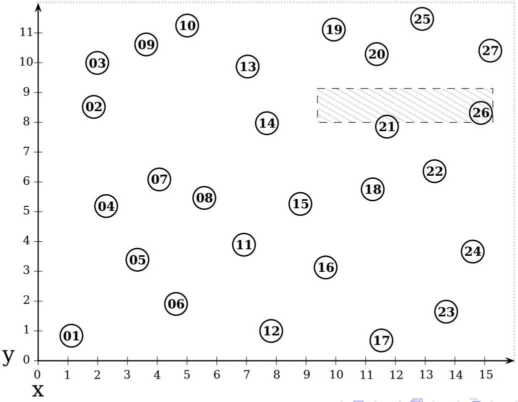
Árvore AVL

Árvore AVL:

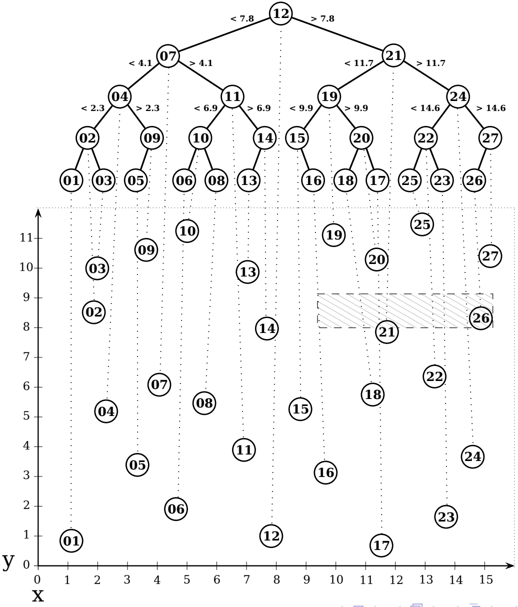
- Implementação complexa ▼
- Pouca utilização de memória ▲
- Indexação unidimensional ●



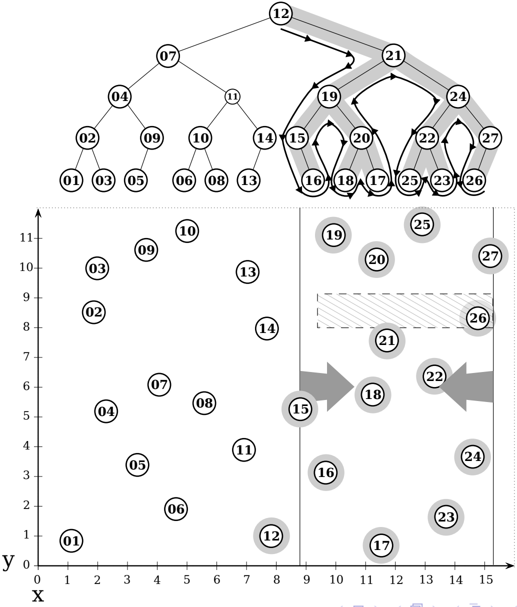
Árvore AVL


01
02
03
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
y
x
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Árvore AVL


12
< 7.8
> 7.8
07
21
< 4.1
> 4.1
< 11.7
> 11.7
04
11
19
24
< 2.3
> 2.3
< 6.9
> 6.9
< 9.9
> 9.9
< 14.6
> 14.6
02
09
10
14
15
20
22
27
01
03
05
06
08
13
16
18
17
25
23
26
y
x
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Árvore AVL


12
07
21
04
11
19
24
02
09
10
14
15
20
22
27
01
03
05
06
08
13
16
18
17
25
23
26
y
x
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Árvore KD

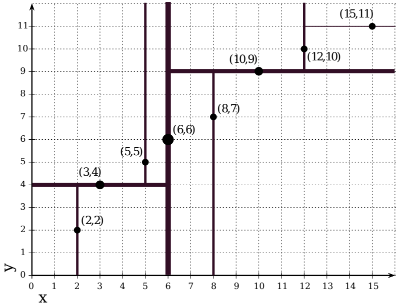
Árvore KD:

- Implementação complexa ▼
- Pouca utilização de memória ▲
- Indexação multidimensional ▲

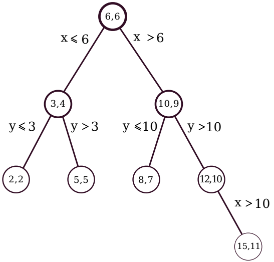
Árvore KD

Árvore KD:

- Implementação complexa ▼
- Pouca utilização de memória ▲
- Indexação multidimensional ▲

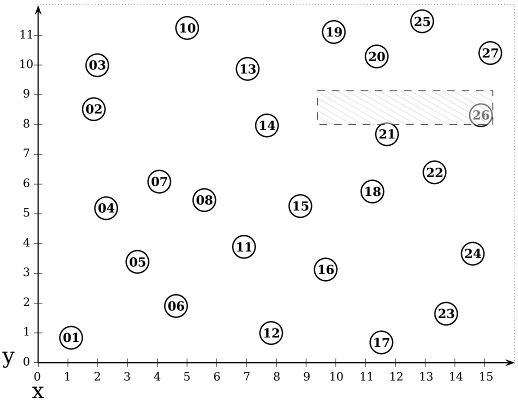


(a) Pontos dispostos num plano bi-dimensional.

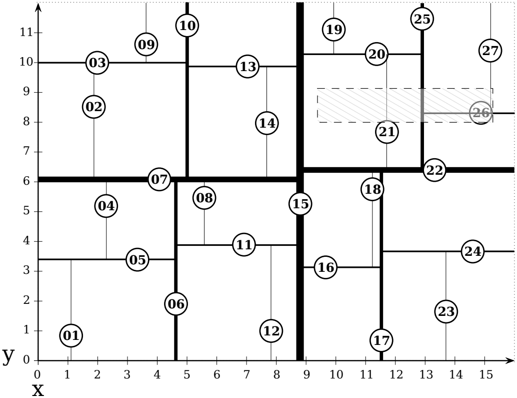


(b) Pontos indexados por uma 2-d tree.

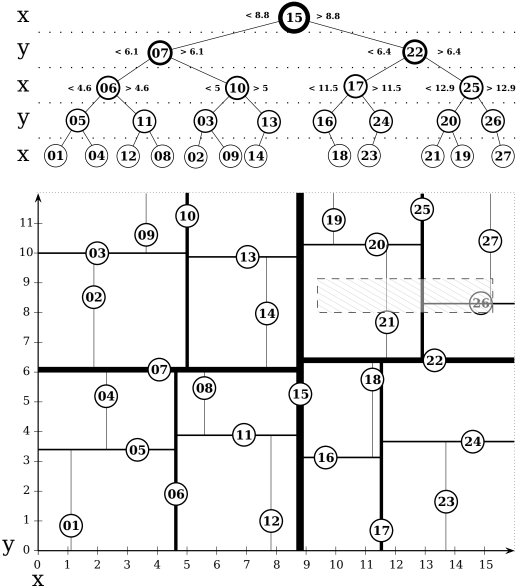
Árvore KD


y
x
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
01
02
03
04
05
06
07
08
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

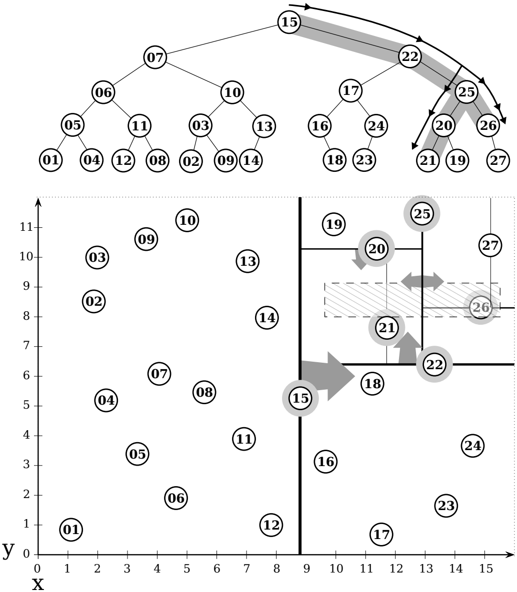
Árvore KD


01
02
03
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
y
x
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Árvore KD


x
y
x
y
x
< 8.8
15
> 8.8
< 6.1
07
> 6.1
< 6.4
22
> 6.4
< 4.6
06
> 4.6
< 5
10
> 5
< 11.5
17
> 11.5
< 12.9
25
> 12.9
05 11 03 13 16 24 20 26
01 04 12 08 02 09 14 18 23 21 19 27
y
x
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Árvore KD


15
07
22
06
10
17
25
05
11
03
13
16
24
20
26
01
04
12
08
02
09
14
18
23
21
19
27
y
x
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Algoritmos para o MOKP

Algoritmo Exato

- Algoritmo de Bazgan – estado da arte;

Algoritmo Heurístico

- SCE para o MOKP – proposta do trabalho;

O Algoritmo de Bazgan

- Algoritmo exato de melhor desempenho;
- Algoritmo de programação dinâmica – variação do Nemhauser-Ullmann;
- Utiliza 3 dominâncias para redução do conjunto de estados.

O Algoritmo de Bazgan

Algoritmo 1: O algoritmo de Nemhauser e Ullmann para o MOKP.

input: $\boldsymbol{p}, \boldsymbol{w}, W$

begin

  $S_0 = \{(0, \ldots, 0)\}$;

  for $k \leftarrow 1, n$ **do**

    $S_k \leftarrow S_{k-1} \cup \left\{(s^1 + p_k^1, \ldots, s^m + p_k^m, s^{m+1} + w_k) \mid s^{m+1} + w_k \leqslant W,\ s \in S_{k-1}\right\}$;

  end

  $P = \{s \in S_n \mid \nexists (a \in S_n \mid a \Delta s)\}$;

  return P;

O Algoritmo de Bazgan

Algoritmo 2: O algoritmo de Nemhauser e Ullmann para o MOKP.

input: $\boldsymbol{p}, \boldsymbol{w}, W$
begin
  $S_0 = \{(0, \ldots, 0)\}$;
  for $k \leftarrow 1, n$ **do**
    $S_k \leftarrow S_{k-1} \cup \left\{(s^1 + p_k^1, \ldots, s^m + p_k^m, s^{m+1} + w_k) \mid s^{m+1} + w_k \leqslant W,\ s \in S_{k-1}\right\}$;
  end
  $P = \{s \in S_n \mid \nexists(a \in S_n \mid a \Delta s)\}$;
  return P;

Relações de dominância utilizadas:

1. D^r: Soluções deficientes;
2. D^Δ: Soluções "*pesadas*";
3. D^b: Soluções não promissoras.

O Algoritmo de Bazgan

1. A relação D^r:

Caso a capacidade residual de uma solução associada a um estado s_k da iteração k seja maior ou igual à soma dos pesos dos itens restantes, o único complemento de s_k que pode resultar em uma solução eficiente é o complemento máximo $I = \{k + 1, \ldots, n\}$.

O Algoritmo de Bazgan

1. A relação D^r:

Caso a capacidade residual de uma solução associada a um estado s_k da iteração k seja maior ou igual à soma dos pesos dos itens restantes, o único complemento de s_k que pode resultar em uma solução eficiente é o complemento máximo $I = \{k+1, \ldots, n\}$.

$$s_k D_k^r s_{k'} \Leftrightarrow \begin{cases} s_{k'} \in S_{k-1}, \\ s_k = (s_{k'}^1 + p_k^1, \ldots, s_{k'}^m + p_k^m, s_{k'}^{m+1} + w_k), \text{ e} \\ s_{k'}^{m+1} \leqslant W - \sum_{i=k}^n w_i \end{cases}$$

O Algoritmo de Bazgan

2. A relação D^{Δ}:

Generalização para o caso multiobjetivo da relação de dominância utilizada no Algoritmo de Nemhauser Ullmann.

O Algoritmo de Bazgan

2. A relação D^{Δ}:

Generalização para o caso multiobjetivo da relação de dominância utilizada no Algoritmo de Nemhauser Ullmann.

$$s_k D_k^{\Delta} s_{k'} \Leftrightarrow \begin{cases} s_k \Delta s_{k'} & \text{e} \\ s_k^{m+1} \leqslant s_{k'}^{m+1} & \text{se } k < n \end{cases}$$

O Algoritmo de Bazgan

3. A relação D^b:

Limite inferior

Vetor objetivo $lb(s) = (lb^1, \ldots, lb^m)$ onde

$$lb^j = s^j + \sum_{i \in J} p_i^j$$

para um complemento J qualquer.

O Algoritmo de Bazgan

3. A relação D^b:

Limite inferior

Vetor objetivo $lb(s) = (lb^1, \ldots, lb^m)$ onde

$$lb^j = s^j + \sum_{i \in J} p_i^j$$

para um complemento J qualquer.

Limite superior

Vetor objetivo $u = (u^1, \ldots, u^m)$ tal que $\forall s_n \in Ext(s_k)$ tem-se que $u^j \geqslant s_n^j, \quad j = 1, \ldots, m.$

O Algoritmo de Bazgan

3. A relação D^b:

$$s_k D_k^b s_{k'} \Leftrightarrow lb(u) \Delta ub(s)$$

O Algoritmo de Bazgan

3. A relação D^b:

$$s_k D_k^b s_{k'} \Leftrightarrow lb(u) \Delta ub(s)$$

O limite superior utilizado:

$$ub^j(s) = s^j + \sum_{i=k+1}^{c_j-1} p_i^j + max\left\{ \left\lfloor \overline{W}(s) \frac{p_{c_j+1}^j}{w_{c_j+1}} \right\rfloor, \left\lfloor p_{c_j}^j - \left(w_{c_j} - \overline{W}(s)\right) . \frac{p_{c_j-1}^j}{w_{c_j-1}} \right\rfloor \right\}$$

O Algoritmo de Bazgan

3. A relação D^b:

$$s_k D_k^b s_{k'} \Leftrightarrow lb(u) \Delta ub(s)$$

O limite superior utilizado:

$$ub^j(s) = s^j + \sum_{i=k+1}^{c_j-1} p_i^j + max\left\{\left\lfloor \overline{W}(s)\frac{p_{c_j+1}^j}{w_{c_j+1}}\right\rfloor, \left\lfloor p_{c_j}^j - \left(w_{c_j} - \overline{W}(s)\right).\frac{p_{c_j-1}^j}{w_{c_j-1}}\right\rfloor\right\}$$

O limite inferior utilizado (complemento J):

$$lb^j(s) = s^j + \sum_{i \in J} p_i^j, \quad \sum_{i \in J} w_i \leqslant \overline{W}(s)$$

O Algoritmo de Bazgan

Algoritmo 3: Algoritmo Bazgan.

input: $\boldsymbol{p}, \boldsymbol{w}, W$

begin

$S_0 \leftarrow \{(0, \ldots, 0)\}$;

$o_1, \ldots, o_n = \mathcal{O}^{max}$;

O Algoritmo de Bazgan

Algoritmo 4: Algoritmo Bazgan.

input: $\boldsymbol{p}, \boldsymbol{w}, W$

begin

$S_0 \leftarrow \{(0, \ldots, 0)\}$;

$o_1, \ldots, o_n = \mathcal{O}^{max}$;

for $k \leftarrow 1, n$ **do**

$S_k^* \leftarrow \{(s^1 + p^1_{o_k}, \ldots, s^m + p^m_{o_k}, s^{m+1} + w_{o_k}) \mid s \in S_{k-1}, s^{m+1} + w_{o_k} \leqslant W\}$

O Algoritmo de Bazgan

Algoritmo 5: Algoritmo Bazgan.

input: $\boldsymbol{p}, \boldsymbol{w}, W$

begin

$S_0 \leftarrow \{(0, \ldots, 0)\}$;

$o_1, \ldots, o_n = \mathcal{O}^{max}$;

for $k \leftarrow 1, n$ **do**

$S_k^* \leftarrow \{(s^1 + p_{o_k}^1, \ldots, s^m + p_{o_k}^m, s^{m+1} + w_{o_k}) \mid s \in S_{k-1}, s^{m+1} + w_{o_k} \leqslant W\}$

$\cup \{s \in S_{k-1} \mid s^{m+1} + w_{o_k} + \ldots + w_{o_n} > W\}$;

O Algoritmo de Bazgan

Algoritmo 6: Algoritmo Bazgan.

input: $\boldsymbol{p}, \boldsymbol{w}, W$

begin

- $S_0 \leftarrow \{(0, \ldots, 0)\}$;
- $o_1, \ldots, o_n = \mathcal{O}^{max}$;
- **for** $k \leftarrow 1, n$ **do**
  - $S_k^* \leftarrow \left\{(s^1 + p^1_{o_k}, \ldots, s^m + p^m_{o_k}, s^{m+1} + w_{o_k}) \mid s \in S_{k-1}, s^{m+1} + w_{o_k} \leqslant W\right\}$
    $\cup \left\{s \in S_{k-1} \mid s^{m+1} + w_{o_k} + \ldots + w_{o_n} > W\right\}$;
  - $S_k^{**} \leftarrow \left\{s \in S_k^* \mid (\nexists u \in S_k^*)\left[u \Delta s\right]\right\}$;

O Algoritmo de Bazgan

Algoritmo 7: Algoritmo Bazgan.

input: $\boldsymbol{p}, \boldsymbol{w}, W$

begin

$S_0 \leftarrow \{(0, \dots, 0)\}$;

$o_1, \dots, o_n = \mathcal{O}^{max}$;

for $k \leftarrow 1, n$ **do**

$S_k^* \leftarrow \left\{(s^1 + p^1_{o_k}, \dots, s^m + p^m_{o_k}, s^{m+1} + w_{o_k}) \;\middle|\; s \in S_{k-1}, s^{m+1} + w_{o_k} \leqslant W\right\}$

$\cup \left\{s \in S_{k-1} \;\middle|\; s^{m+1} + w_{o_k} + \dots + w_{o_n} > W\right\}$;

$S_k^{**} \leftarrow \left\{s \in S_k^* \;\middle|\; (\nexists u \in S_k^*)\left[u \Delta s\right]\right\}$;

$S_k \leftarrow \left\{s \in S_k^{**} \;\middle|\; (\nexists u \in S_k^{**})\left[lb(u) \Delta ub(s)\right]\right\}$;

O Algoritmo de Bazgan

Algoritmo 8: Algoritmo Bazgan.

input: $\boldsymbol{p}, \boldsymbol{w}, W$

begin

  $S_0 \leftarrow \{(0, \ldots, 0)\}$;

  $o_1, \ldots, o_n = \mathcal{O}^{max}$;

  for $k \leftarrow 1, n$ **do**

    $S_k^* \leftarrow \{(s^1 + p^1_{o_k}, \ldots, s^m + p^m_{o_k}, s^{m+1} + w_{o_k}) \mid s \in S_{k-1}, s^{m+1} + w_{o_k} \leqslant W\}$

      $\cup \{s \in S_{k-1} \mid s^{m+1} + w_{o_k} + \ldots + w_{o_n} > W\}$;

    $S_k^{**} \leftarrow \{s \in S_k^* \mid (\nexists u \in S_k^*)[u \Delta s]\}$;

    $S_k \leftarrow \{s \in S_k^{**} \mid (\nexists u \in S_k^{**})[lb(u) \Delta ub(s)]\}$;

  return S_n;

O Algoritmo de Bazgan

Algoritmo 9: Algoritmo Bazgan.

input: $\boldsymbol{p}, \boldsymbol{w}, W$

1 **begin**

2 $\quad S_0 \leftarrow \{(0, \ldots, 0)\}$;

3 $\quad o_1, \ldots, o_n = \mathcal{O}^{max}$;

4 $\quad$ **for** $k \leftarrow 1, n$ **do**

5 $\quad\quad S_k^* \leftarrow \{(s^1 + p^1_{o_k}, \ldots, s^m + p^m_{o_k}, s^{m+1} + w_{o_k}) \mid s \in S_{k-1}, s^{m+1} + w_{o_k} \leqslant W\}$

6 $\quad\quad\quad \cup \{s \in S_{k-1} \mid s^{m+1} + w_{o_k} + \ldots + w_{o_n} > W\}$;

7 $\quad\quad S_k^{**} \leftarrow \{s \in S_k^* \mid (\nexists u \in S_k^*)[u \Delta s]\}$ ◀;

8 $\quad\quad S_k \leftarrow \{s \in S_k^{**} \mid (\nexists u \in S_k^{**})[lb(u) \Delta ub(s)]\}$;

9 $\quad$ **return** S_n;

- verificação da condição $u \Delta s$ (linha 7);

O Algoritmo de Bazgan

Algoritmo 10: Algoritmo Bazgan.

input: $\boldsymbol{p}, \boldsymbol{w}, W$

1 **begin**

2 $S_0 \leftarrow \{(0, \ldots, 0)\}$;

3 $o_1, \ldots, o_n = \mathcal{O}^{max}$;

4 **for** $k \leftarrow 1, n$ **do**

5 $S_k^* \leftarrow \{(s^1 + p^1_{o_k}, \ldots, s^m + p^m_{o_k}, s^{m+1} + w_{o_k}) \mid s \in S_{k-1}, s^{m+1} + w_{o_k} \leqslant W\}$

6 $\cup \{s \in S_{k-1} \mid s^{m+1} + w_{o_k} + \ldots + w_{o_n} > W\}$;

7 $S_k^{**} \leftarrow \{s \in S_k^* \mid (\nexists u \in S_k^*)[u \Delta s]\}$ ◀;

8 $S_k \leftarrow \{s \in S_k^{**} \mid (\nexists u \in S_k^{**})[lb(u) \Delta ub(s)]\}$ ◀;

9 **return** S_n;

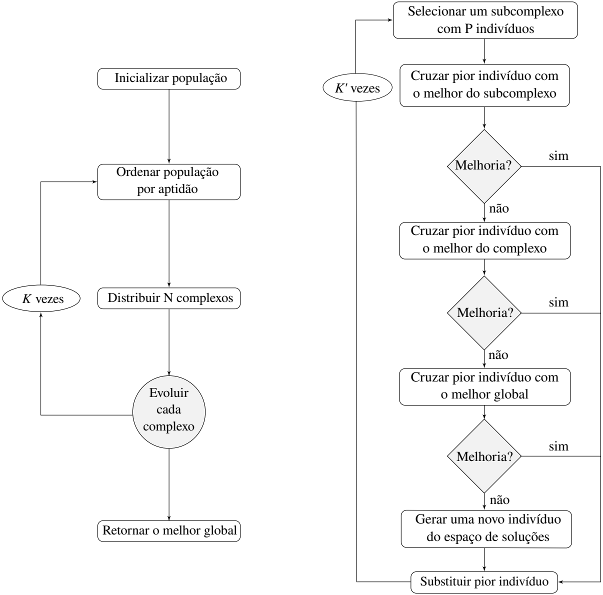
- verificação da condição $u\Delta s$ (linha 7);
- verificação da condição $lb(u)\Delta ub(s)$ (linha 8).

O SCE para o MOKP

O SCE

- Algoritmo populacional evolutivo;
- Evolução simultânea de comunicades independentes;
- Utilizado originalmente para resolver problemas hídricos complexos;
- *Embaralha* a população em N comunidades (complexos);

O SCE


Inicializar população
Ordenar população por aptidão
Distribuir N complexos
Evoluir cada complexo
Retornar o melhor global
K vezes
Selecionar um subcomplexo com P indivíduos
K' vezes
Cruzar pior indivíduo com o melhor do subcomplexo
Melhoria?
sim
não
Cruzar pior indivíduo com o melhor do complexo
Melhoria?
sim
não
Cruzar pior indivíduo com o melhor global
Melhoria?
sim
não
Gerar uma novo indivíduo do espaço de soluções
Substituir pior indivíduo

O SCE

Adaptação para contexto multiobjetivo:

- Aptidão do indivíduo:
- Construção de conjunto Pareto aproximado:

O SCE

Adaptação para contexto multiobjetivo:

- Aptidão do indivíduo:
  - Ordenação em frontes não dominados
- Construção de conjunto Pareto aproximado:
  - Arquivo externo

O SCE

Adaptação para contexto multiobjetivo:

- Aptidão do indivíduo:
  - Ordenação em frontes não dominados
- Construção de conjunto Pareto aproximado:
  - Arquivo externo

Aplicação para o MOKP:

- Construção de solução aleatória;
- Procedimento de cruzamento.

O SCE

Aptidão do indivíduo: ordenação em frontes não dominados.

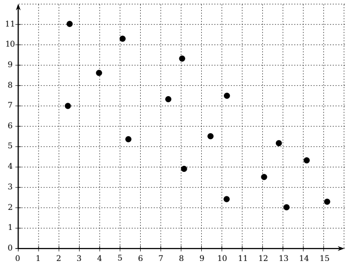


Figure: População sem ordenação.

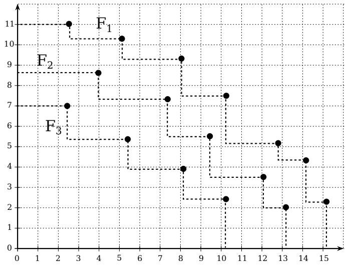


Figure: População ordenada em frontes não dominados.

O SCE

Construção de conjunto Pareto aproximado: utilização de arquivo externo.

Algoritmo 11: Procedimento de atualização de arquivo, dada uma nova solução.

---

input: A : arquivo, x : indivíduo

```
begin
    if ∄(y ∈ A, yΔx) then
        A ← A ∪ {x};                      ▷ Inclusão de x no arquivo
        A ← A\{z ∈ A | xΔz};   ▷ Remoção das soluções dominadas
            por x
    return A;
```

O SCE

Algoritmo 12: Algoritmo SCE adaptado para o MOKP.

```
begin
    Inicializar população de N ∗ M indivíduos gerados aleatoriamente;
    Classificar população em frontes não dominados ;
    Selecionar o 1º fronte para compor arquivo externo;
    for k ← 1 : K do
        Ordenar população por aptidão (desempate por hipervolume);
        Distribuir população em M complexos;
        for i ← 1 : N do
            for k′ ← 1 : K′ do
                Selecionar subcomplexo com P indivíduos retirados
                  do i-ésimo complexo;
                Evoluir pior indivíduo do subcomplexo gerando um
                  novo indivíduo ;
        Classificar toda a população (nova e antiga) em frontes não
          dominados ;
        Propor atualização do arquivo utilizando as soluções do 1º fronte
          F_1 ;
        Selecionar população;
    return Arquivo externo;
```

O SCE

Algoritmo 13: Algoritmo SCE adaptado para o MOKP.

```
1  begin
2      Inicializar população de N * M indivíduos gerados aleatoriamente;
3      Classificar população em frontes não dominados ◄;
4      Selecionar o 1º fronte para compor arquivo externo;
5      for k ← 1 : K do
6          Ordenar população por aptidão (desempate por hipervolume);
7          Distribuir população em M complexos;
8          for i ← 1 : N do
9              for k' ← 1 : K' do
10                 Selecionar subcomplexo com P indivíduos retirados
                     do i-ésimo complexo;
11                 Evoluir pior indivíduo do subcomplexo gerando um
                     novo indivíduo ◄;
12         Classificar toda a população (nova e antiga) em frontes não
             dominados ◄;
13         Propor atualização do arquivo utilizando as soluções do 1º fronte
             F_1 ◄;
14         Selecionar população;
15     return Arquivo externo;
```

- Classificar a população em frontes não dominados (linhas 3 e 12);
- Verificar se o indivíduo teve aptidão melhorada (linha 11);
- Atualização do arquivo, dada uma nova solução (linha 13).

Experimentos Computacionais

Experimentos Computacionais

Abordagem Exata

Instâncias bi-objetivo divididas em 4 tipos:

A) Aleatórias: $p_i^j \in [1, 1000], w_i \in [1, 1000]$.

B) Não-conflitantes:
$p_i^1 \in [111, 1000], p_i^2 \in [p_i^1 - 100, p_i^1 + 100], w_i \in [1, 1000]$.

C) Conflitantes: $p_i^1 \in [1, 1000], p_i^2 \in [max\{900 - p_i^1; 1\}, min\{1100 - p_i^1, 1000\}], w_i \in [1, 1000]$.

D) Conflitantes com pesos correlacionados:
$p_i^1 \in [1, 1000], p_i^2 \in [max\{900 - p_i^1; 1\}, min\{1100 - p_i^1, 1000\}], w_i \in [p_i^1 + p_i^2 - 200, p_i^1 + p_i^2 + 200]$.

Experimentos Computacionais

Abordagem Exata

Instâncias 3-objetivo divididas em 4 tipos:

A) Aleatórias: $p_i^j \in [1, 1000] w_i \in [1, 1000]$

B) Não-conflitantes: $p_i^1 \in [111, 1000], p_i^2 \in [p_i^1 - 100, p_i^1 + 100], p_i^3 \in [p_i^1 - 100, p_i^1 + 100], w_i \in [1, 1000]$.

C) Conflitantes: $p_i^1 \in [1, 1000],\ p_i^2 \in [1, 1001 - p_i^1] p_i^3 \in [max\{900 - p_i^1 - p_i^2; 1\}, min\{1100 - p_i^1 - p_i^2, 1001 - p_i^1\}] w_i \in [1, 1000]$.

D) Conflitantes com pesos correlacionados: $p_i^1 \in [1, 1000] p_i^2 \in [1, 1001 - p_i^1] p_i^3 \in [max\{900 - p_i^1 - p_i^2; 1\}, min\{1100 - p_i^1 - p_i^2, 1001 - p_i^1\}] w_i \in [p_i^1 + p_i^2 + p_i^3 - 200, p_i^1 + p_i^2 + p_i^3 + 200]$.

Experimentos Computacionais

Abordagem Exata

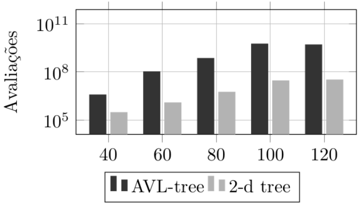
Tempo computacional médio do algoritmo Bazgan para instâncias bi-objetivo:

Instância			AVL tree	árvore 2-d	
Tipo	n	$\|Par\|$	tempo (s)	tempo (s)	speedup
A	40	38.1	**0.06**	**0.06**	1.0
	60	73.1	1.12	**0.88**	1.3
	80	125.6	19.81	**11.89**	1.7
	100	180.4	165.24	**76.50**	2.2
	120	233.9	708.53	**361.87**	2.0
B	100	3.1	**0.02**	0.08	0.3
	200	10.0	**0.80**	5.09	0.2
	300	24.9	**9.45**	88.30	0.1
	400	36.2	**95.39**	730.04	0.1
	500	53.7	**255.57**	2824.65	0.1
C	20	36.6	**0.00**	**0.00**	1.0
	40	102.8	0.65	**0.42**	1.5
	60	231.9	28.98	**14.09**	2.1
	80	358.0	564.10	**241.54**	2.3
	100	513.8	3756.57	**1605.19**	2.3
D	20	174.9	0.15	**0.12**	1.3
	30	269.3	16.82	**7.60**	2.2
	40	478.0	395.76	**186.67**	2.1
	50	553.4	2459.48	**1417.94**	1.7

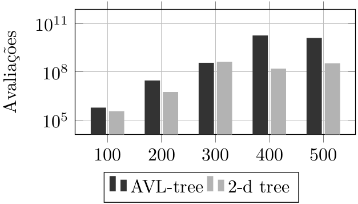
Experimentos Computacionais

Abordagem Exata

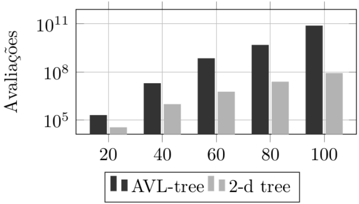
Número de avaliações médio do algoritmo Bazgan para instâncias bi-objetivo:



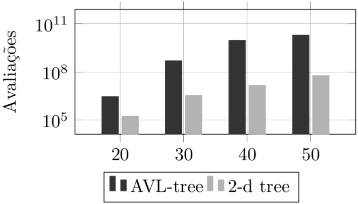
(a) Tipo A



(b) Tipo B



(c) Tipo C



(d) Tipo D

Experimentos Computacionais

Abordagem Exata

Tempo computacional médio do algoritmo Bazgan para instâncias 3-objetivo:

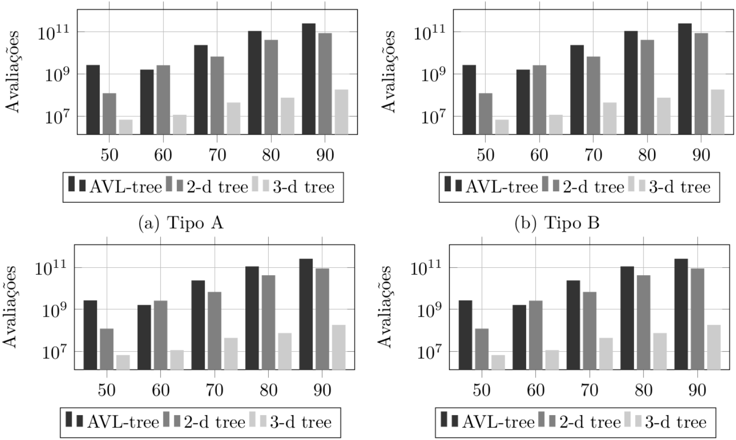
Instância			AVL tree	árvore 2-d		árvore 3-d	
Tipo	n	$\|Par\|$	tempo (s)	tempo (s)	speedup	tempo (s)	speedup
A	50	557.5	41.2	21.3	1.9	**18.5**	2.2
	60	1240.0	485.9	247.8	1.9	**79.9**	6.0
	70	1879.3	3179.5	1038.0	3.0	**614.5**	5.1
	80	2540.5	6667.9	3796.0	1.7	**2943.9**	2.2
	90	3528.5	24476.5	12916.7	1.8	**3683.7**	6.6
B	100	18.0	**0.1**	0.3	0.3	0.3	0.3
	200	65.4	**11.4**	34.4	0.3	29.1	0.4
	300	214.2	**307.7**	631.5	0.5	583.2	0.5
	400	317.0	**4492.9**	8464.9	0.5	5402.2	0.8
C	20	254.4	0.06	0.05	1.2	**0.03**	2.17
	30	1066.6	9.69	4.18	2.3	**1.30**	7.46
	40	2965.5	471.68	153.21	3.1	**30.50**	15.5
D	20	4087.7	23.6	10.9	2.2	**1.9**	12.5
	30	8834.5	8914.2	3625.3	2.5	**1019.5**	8.7

Introdução
O MOKP
A Verificação de Dominância
O Algoritmo de Bazgan
O SCE
Experimentos
Conclusões e Trabalhos Futuros

Experimentos Computacionais

Abordagem Exata

Número de avaliações médio do algoritmo Bazgan para instâncias 3-objetivo:



(a) Tipo A

(b) Tipo B

(c) Tipo C

(d) Tipo D

Experimentos Computacionais

Abordagem Heurística

Valores de parâmetros utilizados no algoritmo SCE:

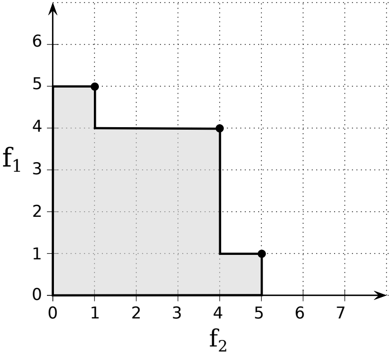
Parâmetro	Valor	Descrição
N	30	Número de complexos
M	30	Número de indivíduos em cada complexo
P	5	Número de indivíduos em cada subcomplexo
K	400	Número de iterações
K'	30	Número de iterações aplicados a cada evolução de complexo
c	$n/20$	Número de genes carregados no procedimento de cruzamento

Experimentos Computacionais

Abordagem Heurística

Métrica para avaliação de qualidade de Pareto: hiper-volume.

Exemplo de conjunto Pareto bi-objetivo possuindo 18 unidades hiper-volume (área):



Experimentos Computacionais

Abordagem Heurística

Hiper-volume médio alcançado por cada heurística:

m	n	SPEA2	NSGA-II	MOEA/D	MOFPA	SCE
2	250	90.4	86.3	96.9	**97.8**	93.6
	500	87.6	81.7	96.9	**97.8**	92.7
	750	85.9	79.2	98.4	**99.2**	92.3
3	250	83.3	77.4	99.0	**99.7**	89.4
	500	72.8	65.9	92.9	**93.6**	79.4
	750	77.5	73.3	94.7	**95.2**	79.8

Experimentos Computacionais

Abordagem Heurística

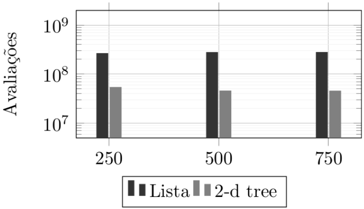
Tempo computacional médio do algoritmo SCE para instâncias Zouache:

Instância			Lista	árvore 2-d		árvore 3-d	
m	n	$\|Par\|$	tempo (s)	tempo (s)	speedup	tempo (s)	speedup
2	250	88.4	**9.3**	13.1	0.71	–	–
	500	106.0	**14.3**	18.3	0.78	–	–
	750	120.4	**18.7**	22.3	0.84	–	–
3	250	705.5	9.8	10.1	0.97	**9.1**	1.08
	500	672.8	15.6	16.0	0.98	**15.2**	1.03
	750	646.0	22.0	24.2	0.91	**21.8**	1.01

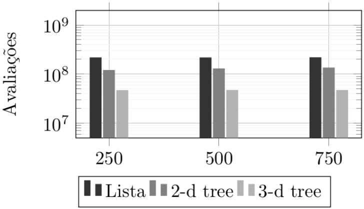
Experimentos Computacionais

Abordagem Heurística

Número de avaliações médio do algoritmo SCE para instâncias Zouache:



(a) Instâncias 2-objetivo



(b) Instâncias 3-objetivo.

Conclusões e Trabalhos Futuros

Conclusões

Principais contribuições:

- Interpretação do problema de verificação de dominância como problema de busca de faixa
- Proposta da utilização da Árvore KD como estrutura de indexação multidimensional
- Análise da proposta em contextos exatos e heurísticos utilizando as principais instâncias da literatura

Conclusões

- Indexação multidimensional foi eficiente no contexto exato:

Conclusões

- Indexação multidimensional foi eficiente no contexto exato:
    - Conjuntos solução de alta cardinalidade

Conclusões

- Indexação multidimensional foi eficiente no contexto exato:
  - Conjuntos solução de alta cardinalidade
  - Speedup de até 2.3 para bi-objetivo

Conclusões

- Indexação multidimensional foi eficiente no contexto exato:
  - Conjuntos solução de alta cardinalidade
  - Speedup de até 2.3 para bi-objetivo
  - Speedup de até 15.5 para 3-objetivo

Conclusões

- Indexação multidimensional foi eficiente no contexto exato:
  - Conjuntos solução de alta cardinalidade
  - Speedup de até 2.3 para bi-objetivo
  - Speedup de até 15.5 para 3-objetivo
  - Drástica redução no número de avaliações de solução

Conclusões

- Indexação multidimensional foi eficiente no contexto exato:
  - Conjuntos solução de alta cardinalidade
  - Speedup de até 2.3 para bi-objetivo
  - Speedup de até 15.5 para 3-objetivo
  - Drástica redução no número de avaliações de solução
- Indexação multidimensional não foi eficiente no contexto heurístico:

Conclusões

- Indexação multidimensional foi eficiente no contexto exato:
  - Conjuntos solução de alta cardinalidade
  - Speedup de até 2.3 para bi-objetivo
  - Speedup de até 15.5 para 3-objetivo
  - Drástica redução no número de avaliações de solução
- Indexação multidimensional não foi eficiente no contexto heurístico:
  - Pouco impacto no tempo computacional

Conclusões

- Indexação multidimensional foi eficiente no contexto exato:
  - Conjuntos solução de alta cardinalidade
  - Speedup de até 2.3 para bi-objetivo
  - Speedup de até 15.5 para 3-objetivo
  - Drástica redução no número de avaliações de solução
- Indexação multidimensional não foi eficiente no contexto heurístico:
  - Pouco impacto no tempo computacional
  - Conjuntos solução de baixa cardinalidade – overhead

Conclusões

- Indexação multidimensional foi eficiente no contexto exato:
  - Conjuntos solução de alta cardinalidade
  - Speedup de até 2.3 para bi-objetivo
  - Speedup de até 15.5 para 3-objetivo
  - Drástica redução no número de avaliações de solução
- Indexação multidimensional não foi eficiente no contexto heurístico:
  - Pouco impacto no tempo computacional
  - Conjuntos solução de baixa cardinalidade – overhead
  - Considerável redução no número de avaliações de solução

Tabalhos Futuros

- Verificar a performance da árvore KD em outros problemas multiobjetivos;
- Considerar outras estruturas de dados para auxílio à operação de verificação de dominância;
- Aprimorar a implementação do SCE para o MOKP;
- Investigar a causa da ineficiência da atual implementação do algoritmo Bazgan.

Obrigado.