Universidade Federal de São Carlos
Departamento de Engenharia de Produção

Otimização Linear Contínua e Discreta (Tópicos Avançados em PCSP)

PPGEP, UFSCar - Semestre 01/2022
Prof. Dr. Pedro Munari (munari@dep.ufscar.br)

Tópico 13.1: Decomposição de Dantzig-Wolfe em problemas de Otimização Discreta

Objetivos deste tópico

- Estudar como aplicar a Decomposição de Dantzig-Wolfe quando o problema possui variáveis discretas;
- Conhecer as técnicas de convexificação e discretização;
- Entender o que é o método *branch-and-price* e suas principais características.

Problema de dimensionamento de lotes com capacidade

▷ Problema Mestre

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} \sum_{t=1}^{T} (c_{it}\bar{x}_{qit} + h_{it}\bar{I}_{qit})\lambda_q^i \\
\text{s.a} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} (a_i \bar{x}_{qit})\lambda_q^i \leq b_t, && t = 1, \dots, T, \\
& \sum_{q \in Q^i} \lambda_q^i = 1, && i = 1, \dots, n, \\
& \lambda_q^i \geq 0, && i = 1, \dots, n, \; q \in Q_i.
\end{aligned}
$$

- Poderíamos chegar a essa formulação intuitivamente, apenas com o conhecimento do problema? (ou seja, sem recorrer à DDW)

Problema de dimensionamento de lotes com capacidade

▷ Problema Mestre

- Pensamos em possíveis planos de produção para cada item, por exemplo:

$$\begin{bmatrix} \bar{x}_{11} \\ \bar{x}_{12} \\ \bar{x}_{13} \\ \bar{I}_{11} \\ \bar{I}_{12} \\ \bar{I}_{13} \end{bmatrix} = \begin{bmatrix} 4500 \\ 0 \\ 0 \\ 3600 \\ 1800 \\ 0 \end{bmatrix}, \qquad \begin{bmatrix} \bar{x}_{21} \\ \bar{x}_{22} \\ \bar{x}_{23} \\ \bar{I}_{21} \\ \bar{I}_{22} \\ \bar{I}_{23} \end{bmatrix} = \begin{bmatrix} 400 \\ 1400 \\ 0 \\ 0 \\ 800 \\ 0 \end{bmatrix}.$$

- E transformamos em colunas do problema mestre...

Problema de dimensionamento de lotes com capacidade

▷ Problema Mestre

$$
\begin{aligned}
\text{min} \quad & 6750\lambda_1^1 + 1100\lambda_1^2 + \ldots \\
\text{s.a} \quad & 450\lambda_1^1 + 32\lambda_1^2 + \ldots \leq 240 \\
& 0\lambda_1^1 + 112\lambda_1^2 + \ldots \leq 320 \\
& 0\lambda_1^1 + 0\lambda_1^2 + \ldots \leq 200 \\
& \lambda_1^1 + \ldots = 1, \\
& \lambda_1^2 + \ldots = 1, \\
& \lambda_1^1, \lambda_1^2, \ldots \geq 0.
\end{aligned}
$$

Problema de dimensionamento de lotes com capacidade

▷ Problema Mestre

$$
\begin{aligned}
\text{min} \quad & 6750\lambda_1^1 + 1100\lambda_1^2 + \dots \\
\text{s.a} \quad & 450\lambda_1^1 + 32\lambda_1^2 + \dots \leq 240 \\
& 0\lambda_1^1 + 112\lambda_1^2 + \dots \leq 320 \\
& 0\lambda_1^1 + 0\lambda_1^2 + \dots \leq 200 \\
& \lambda_1^1 + \dots = 1, \\
& \lambda_1^2 + \dots = 1, \\
& \lambda_1^1, \lambda_1^2, \dots \geq 0.
\end{aligned}
$$

- Poderíamos chegar a essa formulação intuitivamente, apenas com o conhecimento do problema? (ou seja, sem recorrer à DDW)

Formulação alternativa para o PDL com preparação

▷ Manne, Management Science 4, 1958

VOLUME 4
NUMBER 2

January 1958

Management Science

PROGRAMMING OF ECONOMIC LOT SIZES*[1]

ALAN S. MANNE[2]

Cowles Foundation for Research in Economics, Yale University

This paper studies the planning problem faced by a machine shop required to produce many different items so as to meet a rigid delivery schedule, remain within capacity limitations, and at the same time minimize the use of premium-cost overtime labor. It differs from alternative approaches to this well-known problem by allowing for setup cost indivisibilities.

As an approximation, the following linear programming model is suggested: Let an activity be defined as a sequence of the inputs required to satisfy the delivery requirements for a single item over time. The input coefficients for each such activity may then be constructed so as to allow for all setup costs incurred when the activity is operated at the level of unity or at zero. It is then shown that in any solution to this problem, all activity levels will turn out to be either unity or zero, except for those related to a group of items which, in number, must be equal to or less than the original number of capacity constraints. This result means that the linear programming solution should provide a good approximation whenever the number of items being manufactured is large in comparison with the number of capacity constraints.

Formulação alternativa para o PDL com preparação

▷ Manne, Management Science 4, 1958

Unknowns, coefficients, and constants for the programming model are defined in the following way:

(a) *unknowns*

x_{ij} = fraction of the total requirement for the i-th part to be supplied by the j-th alternative sequence of inputs. ($i = 1, \cdots, I$; $j = 1, \cdots, J$.)

l_t = number of hours of overtime labor required during time period t($t = 1, \cdots, T$.)

s_t = "slack" variable for straight-time labor during time period t. (all t)

v_t = "slack" variable for overtime labor during time period t. (all t)

(b) *coefficients*

β_{ijt} = labor input required during period t in order to carry out the j-th alternative production sequence for part i. (all i, j, and t)

(c) *constants*

S_t = maximum availability of straight-time labor man-hours during the t-th time period. (all t)

V_t = maximum availability of overtime labor man-hours during the t-th time period. (all t)

With these definitions, the linear programming model becomes:

$$\text{Minimize} \sum_t l_t \tag{2.1}$$

subject to:

$$\sum_j x_{ij} = 1 \qquad (i = 1, \cdots, I) \tag{2.2}$$

$$\sum_i \sum_j \beta_{ijt} x_{ij} - l_t + s_t = S_t \qquad (t = 1, \cdots, T) \tag{2.3}$$

$$l_t + v_t = V_t \qquad (t = 1, \cdots, T) \tag{2.4}$$

$$x_{ij}, l_t, s_t, v_t \geqq 0 \qquad (\text{all } i, j, t) \tag{2.5}$$

PDL com restrição de capacidade

▷ Formulação compacta

$$\min \quad \sum_{i=1}^{n}\sum_{t=1}^{T} c_{it}x_{it} + \sum_{i=1}^{n}\sum_{t=1}^{T} h_{it}I_{it}$$

$$\begin{aligned}
\text{s.a} \quad & \sum_{i=1}^{n} a_i x_{it} \leq b_t, && t = 1, \ldots, T, \\
& x_{it} + I_{i,t-1} = d_{it} + I_{it}, && i = 1, \ldots, n; \quad t = 1, \ldots, T, \\
& I_{i0} = 0, && i = 1, \ldots, n, \\
& x_{it} \geq 0, I_{it} \geq 0, && i = 1, \ldots, n; \quad t = 1, \ldots, T.
\end{aligned}$$

PDL com restrição de capacidade e tempo de preparação

▷ Formulação compacta

$$\min \quad \sum_{i=1}^{n}\sum_{t=1}^{T} c_{it}x_{it} + \sum_{i=1}^{n}\sum_{t=1}^{T} h_{it}I_{it} + \sum_{i=1}^{n}\sum_{t=1}^{T} s_{it}y_{it}$$

$$\begin{aligned}
\text{s.a} \quad & \sum_{i=1}^{n}(a_i x_{it} + st_i y_{it}) \leq b_t, && t = 1,\ldots,T,\\
& x_{it} + I_{i,t-1} = d_{it} + I_{it}, && i = 1,\ldots,n; \quad t = 1,\ldots,T,\\
& x_{it} \leq C y_{it}, && i = 1,\ldots,n; \quad t = 1,\ldots,T,\\
& I_{i0} = 0, && i = 1,\ldots,n,\\
& x_{it} \geq 0,\ I_{it} \geq 0,\ y_{it} \in \{0,1\}, && i = 1,\ldots,n; \quad t = 1,\ldots,T.
\end{aligned}$$

PDL com restrição de capacidade e tempo de preparação

▷ Exemplo

Uma fábrica de refrigerantes produz dois tipos de bebidas, por meio de um único tanque. Para processar 1000 litros da bebida 1 são necessárias 100 horas do tanque, enquanto para 1000 litros da bebida 2, são necessárias 80 horas. A produção de uma bebida em um dado período requer a limpeza e resfriamento do tanque. Esse tempo é de 12 horas para a bebida 1 e 8 horas para a bebida 2. A disponibilidade do tanque para a fabricação destas bebidas nos próximos 3 meses é de 240, 320 e 200 horas. O departamento de vendas fez uma previsão de demanda para os próximos 3 meses. A demanda de cada bebida e os possíveis custos envolvidos são dados na tabela abaixo. Deseja-se determinar quanto produzir e estocar de cada bebida em cada período.

	Bebida 1			Bebida 2		
Período	1	2	3	1	2	3
Demanda (L)	900	1800	1800	400	600	800
Custo prod (R$/L)	1.0	1.5	2.0	0.5	0.5	0.9
Custo estoc (R$/L)	0.5	0.25	—	0.25	0.25	—
Custo prep (R$)	2.0	4.0	4.0	8.0	8.0	8.0

PDL com restrição de capacidade e tempo de preparação

▷ Exemplo: formulação

$$
\begin{array}{ll}
\text{min} & 1.0x_{11} + 1.5x_{12} + 2.0x_{13} + 0.5x_{21} + 0.5x_{22} + 0.9x_{23} \\
& \quad +0.5I_{11} + 0.25I_{12} + 0.25I_{21} + 0.25I_{22} \\
& +2.0y_{11} + 4.0y_{12} + 4.0y_{13} + 8.0y_{21} + 8.0y_{22} + 8.0y_{23} \\
\text{s.a} & 0.1x_{11} + 0.08x_{21} + 12y_{11} + 8y_{21} \leq 240 \\
& 0.1x_{12} + 0.08x_{22} + 12y_{12} + 8y_{22} \leq 320 \\
& 0.1x_{13} + 0.08x_{23} + 12y_{13} + 8y_{23} \leq 200 \\
& x_{11} + I_{10} - I_{11} = 900 \\
& x_{12} + I_{11} - I_{12} = 1800 \\
& x_{13} + I_{12} - I_{13} = 1800 \\
& x_{21} + I_{20} - I_{21} = 400 \\
& x_{22} + I_{21} - I_{22} = 600 \\
& x_{23} + I_{22} - I_{23} = 800 \\
& x_{11} \leq 5000y_{11} \qquad x_{21} \leq 5000y_{21} \\
& x_{12} \leq 5000y_{12} \qquad x_{22} \leq 5000y_{22} \\
& x_{13} \leq 5000y_{13} \qquad x_{23} \leq 5000y_{23} \\
& I_{10} = 0, \;\; I_{20} = 0 \\
& x_{11}, x_{12}, \ldots, x_{23} \geq 0 \\
& I_{11}, I_{12}, \ldots, I_{23} \geq 0 \\
& y_{11}, y_{12}, \ldots, y_{23} \in \{0, 1\}
\end{array}
$$

PDL com restrição de capacidade e tempo de preparação

▷ Formulação compacta

$$\min \quad \sum_{i=1}^{n}\sum_{t=1}^{T} c_{it}x_{it} + \sum_{i=1}^{n}\sum_{t=1}^{T} h_{it}I_{it} + \sum_{i=1}^{n}\sum_{t=1}^{T} s_{it}y_{it}$$

$$\begin{aligned}
\text{s.a} \quad & \sum_{i=1}^{n}(a_i x_{it} + st_i y_{it}) \leq b_t, && t = 1,\ldots,T,\\
& x_{it} + I_{i,t-1} = d_{it} + I_{it}, && i = 1,\ldots,n; \quad t = 1,\ldots,T,\\
& x_{it} \leq C y_{it}, && i = 1,\ldots,n; \quad t = 1,\ldots,T,\\
& I_{i0} = 0, && i = 1,\ldots,n,\\
& x_{it} \geq 0,\ I_{it} \geq 0,\ y_{it} \in \{0,1\}, && i = 1,\ldots,n; \quad t = 1,\ldots,T.
\end{aligned}$$

PDL com restrição de capacidade

▷ Formulação extensiva (estendida)

$$
\begin{aligned}
\text{min} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} \sum_{t=1}^{T} (c_{it}\bar{x}_{qit} + h_{it}\bar{I}_{qit})\lambda_q^i \\
\text{s.a} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} (a_i \bar{x}_{qit})\lambda_q^i \leq b_t, && t = 1, \ldots, T, \\
& \sum_{q \in Q^i} \lambda_q^i = 1, && i = 1, \ldots, n, \\
& \lambda_q^i \geq 0, && i = 1, \ldots, n, \; q \in Q_i.
\end{aligned}
$$

PDL com restrição de capacidade e tempo de preparação

▷ Formulação extensiva (estendida) - adaptada

$$
\begin{aligned}
\text{min} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} \sum_{t=1}^{T} (c_{it}\bar{x}_{qit} + h_{it}\bar{I}_{qit} + s_{it}\bar{y}_{qit})\lambda_q^i \\
\text{s.a} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} (a_i\bar{x}_{qit} + st_i\bar{y}_{qit})\lambda_q^i \leq b_t, && t = 1, \ldots, T, \\
& \sum_{q \in Q^i} \lambda_q^i = 1, && i = 1, \ldots, n, \\
& \lambda_q^i \geq 0, && i = 1, \ldots, n, \ q \in Q_i.
\end{aligned}
$$

PDL com restrição de capacidade e tempo de preparação

▷ Formulação extensiva (estendida) - adaptada

$$\begin{aligned}
\text{min} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} \sum_{t=1}^{T} (c_{it}\bar{x}_{qit} + h_{it}\bar{I}_{qit} + s_{it}\bar{y}_{qit})\lambda_q^i \\
\text{s.a} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} (a_i\bar{x}_{qit} + st_i\bar{y}_{qit})\lambda_q^i \leq b_t, && t = 1, \ldots, T, \\
& \sum_{q \in Q^i} \lambda_q^i = 1, && i = 1, \ldots, n, \\
& \lambda_q^i \geq 0, && i = 1, \ldots, n, \ q \in Q_i.
\end{aligned}$$

- É equivalente à formulação compacta?

PDL com restrição de capacidade e tempo de preparação

▷ Formulação extensiva (estendida) - adaptada

$$
\begin{aligned}
\text{min} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} \sum_{t=1}^{T} (c_{it}\bar{x}_{qit} + h_{it}\bar{I}_{qit} + s_{it}\bar{y}_{qit})\lambda_q^i \\
\text{s.a} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} (a_i \bar{x}_{qit} + st_i \bar{y}_{qit})\lambda_q^i \leq b_t, && t = 1, \ldots, T, \\
& \sum_{q \in Q^i} \lambda_q^i = 1, && i = 1, \ldots, n, \\
& \lambda_q^i \geq 0, && i = 1, \ldots, n, \ q \in Q_i.
\end{aligned}
$$

- É equivalente à formulação compacta?
- Podemos garantir que a preparação será de fato 0 ou 1?

Formulação alternativa para o PDL com preparação

▷ Degraeve and Jans, Operations Research 55, 2007

OPERATIONS RESEARCH
Vol. 55, No. 5, September–October 2007, pp. 909–920
ISSN 0030-364X | EISSN 1526-5463 | 07 | 5505 | 0909

informs®

DOI 10.1287/opre.1070.0404

A New Dantzig-Wolfe Reformulation and Branch-and-Price Algorithm for the Capacitated Lot-Sizing Problem with Setup Times

Zeger Degraeve
London Business School, Regent's Park, London NW1 4SA, United Kingdom, zdegraeve@london.edu

Raf Jans
RSM Erasmus University, 3000 DR Rotterdam, The Netherlands, rjans@rsm.nl

Although the textbook Dantzig-Wolfe decomposition reformulation for the capacitated lot-sizing problem, as already proposed by Manne [Manne, A. S. 1958. Programming of economic lot sizes. *Management Sci.* **4**(2) 115–135], provides a strong lower bound, it also has an important structural deficiency. Imposing integrality constraints on the columns in the master program will not necessarily give the optimal integer programming solution. Manne's model contains only production plans that satisfy the Wagner-Whitin property, and it is well known that the optimal solution to a capacitated lot-sizing problem will not necessarily satisfy this property. The first contribution of this paper answers the following question, unsolved for almost 50 years: If Manne's formulation is not equivalent to the original problem, what is then a correct reformulation? We develop an equivalent mixed-integer programming (MIP) formulation to the original problem and show how this results from applying the Dantzig-Wolfe decomposition to the original MIP formulation. The set of extreme points of the lot-size polytope that are needed for this MIP Dantzig-Wolfe reformulation is much larger than the set of dominant plans used by Manne. We further show how the integrality restrictions on the original setup variables translate into integrality restrictions on the new master variables by separating the setup and production decisions. Our new formulation gives the same lower bound as Manne's reformulation. Second, we develop a branch-and-price algorithm for the problem. Computational experiments are presented on data sets available from the literature. Column generation is accelerated by a combination of simplex and subgradient optimization for finding the dual prices. The results show that

Formulação alternativa para o PDL com preparação

▷ Degraeve and Jans, Operations Research 55, 2007

(Vanderbeck 1998). There is no setup required for initial inventory. Next, we have the setup forcing (3) and capacity constraints (4). Finally, we have the nonnegativity and integrality constraints (5), and the inventory at the end of the final period is set to zero. Let v_{CLST} be the optimal objective value for problem (1)–(5) and $\bar{v}_{\mathrm{CLST}}$ its LP relaxation. We introduce a more compact notation for the variables: $x_i = (x_{i1}, x_{i2}, \ldots, x_{im})$, $s_i = (s_{i0}, s_{i1}, s_{i2}, \ldots, s_{im})$, and $y_i = (y_{i1}, y_{i2}, \ldots, y_{im})$. Further, we define X^i, the single-item lot-size polytope for each product i, as follows:

$$X^i = \left\{ (x_i, s_i, y_i) \;\middle|\; \begin{array}{l} s_{i,t-1} + x_{it} = d_{it} + s_{it} \quad \forall t \in T; \\ x_{it} \leqslant sd_{itm} y_{it} \quad \forall t \in T; \\ y_{it} \in \{0,1\}, \;\; x_{it} \geqslant 0, \;\; s_{it} \geqslant 0 \quad \forall t \in T; \\ \qquad s_{i0} \geqslant 0; \;\; s_{im} = 0. \end{array} \right\}.$$

We first provide a brief literature review on the capacitated lot-sizing problem (§2). The starting point of this research is the well-known observation that the formulation provided by Manne (1958) has a structural deficiency because it is not equivalent to formulation (1)–(5) and provides only a lower bound. It is important to make a distinction between a formulation, in our case a Dantzig-Wolfe (DW) decomposition reformulation, and a solution method, i.e., column generation and branch-and-price (B&P). We have contributions on both accounts. To the best of our knowledge, this is the first time that a formulation has been proposed for the dynamic lot-sizing problem, which is sizing problems. Eppen and Martin (1987) propose a different approach for tightening the formulation using variable redefinition. Kleindorfer and Newson (1975), Thizy and Van Wassenhove (1985), and Trigeiro et al. (1989) propose to dualize the capacity constraint in a Lagrange relaxation approach. A detailed discussion of solution approaches for the CLST can be found in Jans and Degraeve (2007).

Manne (1958) proposes an innovative linear programming formulation for the CLST. He explicitly models all the possible setup schedules. Let $q \in Q^i$, $Q^i = \{(y_{i1}, \ldots, y_{im}) \mid y_{it} \in \{0,1\} \; \forall t \in T\}$ be a feasible setup schedule, and let y_{it}^q be one if there is a setup for item i in period t in setup schedule q, zero otherwise. There are 2^m different setup schedules possible for each product. Exactly one "dominant" production plan corresponds with each setup schedule. Manne considers only dominant production schedules, which have the property that for each period, demand will be met by production in that period if there is a setup, or otherwise from the nearest preceding period with a setup. Manne's dominant production schedules are in fact all the production schedules satisfying the Wagner and Whitin (1958) condition: $s_{i,t-1} x_{it} = 0 \;\forall t \in T, \forall i \in I$. The Wagner-Whitin production plan for item i according to setup schedule q is defined by the production quantities x_{it}^q and is further referred to as the Wagner-Whitin production plan q:

$$x_{it}^q = 0 \quad \text{if } y_{it}^q = 0, = sd_{it,k-1} \text{ otherwise,}$$

Formulação alternativa para o PDL com preparação

▷ Degraeve and Jans, Operations Research 55, 2007

$$\text{Min} \sum_{i\in I}\sum_{p\in P^i}\left(fc_i s_{i0}^p + \sum_{t\in T}(sc_{it}y_{it}^p + vc_{it}x_{it}^p + hc_{it}s_{it}^p)\right)z_p^i, \quad (13)$$

$$\text{s.t.} \sum_{i\in I}\sum_{p\in P^i}(y_{it}^p st_{it} + x_{it}^p vt_{it})z_p^i \leqslant cap_t \quad \forall t\in T, \quad (14)$$

$$\sum_{p\in P^i} z_p^i = 1 \quad \forall i\in I, \quad (15)$$

$$y_{it} = \sum_{p\in P^i} y_{it}^p z_p^i \quad \forall i\in I, \forall t\in T, \quad (16)$$

$$x_{it} = \sum_{p\in P^i} x_{it}^p z_p^i \quad \forall i\in I, \forall t\in T, \quad (17)$$

$$y_{it}\in\{0,1\} \quad \forall i\in I, \forall t\in T, \quad (18)$$

$$z_p^i \geqslant 0 \quad \forall i\in I, \forall p\in P^i. \quad (19)$$

Step 2. *Alternative Description of the Set of Extreme Points*. In the first step, we defined P^i generally as the set of all the extreme points of conv(X^i). In this second step, we define this set more explicitly. Proposition 1 states that P^i consists of all the feasible solutions of X^i that also satisfy the Wagner-Whitin property. This set is much larger than the set of the 2^m dominant plans that Manne used because the set of extreme points also includes nondominant plans satisfying the Wagner-Whitin property, in which it is possible that there is a setup, but no production for

solutions satisfying the Wagner-Whitin property, we can now redefine P^i as follows:

$$P^i = \{(x_i^v, s_i^v, y_i^q) \mid q\in Q^i, v\in Q^{iq}\}.$$

Step 3. *Rewriting the Dantzig-Wolfe Reformulation*. We rewrite the model in terms of convex combinations of extreme points, using the fact that we can define the set P^i in terms of Q^i and Q^{iq}: $\sum_{p\in P^i} z_p^i = \sum_{q\in Q^i}\sum_{v\in Q^{iq}} z_{qv}^i$. The variable z_{qv}^i refers to the extreme point formed by taking setup schedule $q\in Q^i$ and the dominant Wagner-Whitin production plan associated with the setup schedule $v\in Q^{iq}$. The Step 3 formulation, then, is as follows:

$$\text{Min} \sum_{i\in I}\sum_{q\in Q^i}\sum_{v\in Q^{iq}}\left(fc_i s_{i0}^v + \sum_{t\in T}(sc_{it}y_{it}^q + vc_{it}x_{it}^v + hc_{it}s_{it}^v)\right)z_{qv}^i, \quad (20)$$

$$\text{s.t.} \sum_{i\in I}\sum_{q\in Q^i}\sum_{v\in Q^{iq}}(y_{it}^q st_{it} + x_{it}^v vt_{it})z_{qv}^i \leqslant cap_t \quad \forall t\in T, \quad (21)$$

$$\sum_{q\in Q^i}\sum_{v\in Q^{iq}} z_{qv}^i = 1 \quad \forall i\in I, \quad (22)$$

$$y_{it} = \sum_{q\in Q^i}\sum_{v\in Q^{iq}} y_{it}^q z_{qv}^i \quad \forall i\in I, \forall t\in T, \quad (23)$$

$$y_{it}\in\{0,1\} \quad \forall i\in I, \forall t\in T, \quad (24)$$

$$z_{qv}^i \geqslant 0 \quad \forall i\in I, q\in Q^i, v\in Q^{iq}. \quad (25)$$

Step 4. *Integrality Constraints*. Define a new variable

PDL com restrição de capacidade e tempo de preparação

▷ Formulação compacta

$$\min \quad \sum_{i=1}^{n}\sum_{t=1}^{T} c_{it}x_{it} + \sum_{i=1}^{n}\sum_{t=1}^{T} h_{it}I_{it} + \sum_{i=1}^{n}\sum_{t=1}^{T} s_{it}y_{it}$$

$$\begin{aligned}
\text{s.a} \quad & \sum_{i=1}^{n}(a_i x_{it} + st_i y_{it}) \leq b_t, && t = 1,\dots,T, \\
& x_{it} + I_{i,t-1} = d_{it} + I_{it}, && i = 1,\dots,n; \quad t = 1,\dots,T, \\
& x_{it} \leq C y_{it}, && i = 1,\dots,n; \quad t = 1,\dots,T, \\
& I_{i0} = 0, && i = 1,\dots,n, \\
& x_{it} \geq 0,\ I_{it} \geq 0,\ y_{it} \in \{0,1\}, && i = 1,\dots,n; \quad t = 1,\dots,T.
\end{aligned}$$

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição

$$
\begin{aligned}
\mathcal{X}^i = \{\ (x_i, I_i, y_i) \quad | \qquad & x_{it} + I_{i,t-1} = d_{it} + I_{it}, && t = 1, \ldots, T, \\
& x_{it} \leq C y_{it}, && t = 1, \ldots, T, \\
& I_{i0} = 0, && \\
& x_{it} \geq 0, I_{it} \geq 0, y_{it} \in \{0,1\}, && t = 1, \ldots, T\}
\end{aligned}
$$

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição

$$\mathcal{X}^i = \{ (x_i, I_i, y_i) \quad | \quad \begin{array}{ll} x_{it} + I_{i,t-1} = d_{it} + I_{it}, & t = 1, \dots, T, \\ x_{it} \leq C y_{it}, & t = 1, \dots, T, \\ I_{i0} = 0, & \\ x_{it} \geq 0, I_{it} \geq 0, y_{it} \in \{0,1\}, & t = 1, \dots, T \} \end{array}$$

$$\begin{array}{lll} \text{min} & \sum_{i=1}^{n} \sum_{t=1}^{T} c_{it} x_{it} + \sum_{i=1}^{n} \sum_{t=1}^{T} h_{it} I_{it} + \sum_{i=1}^{n} \sum_{t=1}^{T} s_{it} y_{it} & \\ \text{s.a} & \sum_{i=1}^{n} (a_i x_{it} + st_i y_{it}) \leq b_t, & t = 1, \dots, T, \\ & (x_i, I_i, y_i) \in \mathcal{X}^i, & i = 1, \dots, n. \end{array}$$

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição

- Podemos usar o Teorema da Representação?

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição

- Podemos usar o Teorema da Representação?
  Não! Os subconjuntos não são mais convexos agora.

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição

- Podemos usar o Teorema da Representação?
  Não! Os subconjuntos não são mais convexos agora.
- Podemos contornar isso usando *convexificação*.

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição

- Podemos usar o Teorema da Representação?
  Não! Os subconjuntos não são mais convexos agora.
- Podemos contornar isso usando *convexificação*.

Convexificação:

- Substituímos $\mathcal{X}^i$ por $\mathcal{C}^i = conv(\mathcal{X}^i)$,

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição

- Podemos usar o Teorema da Representação?
  Não! Os subconjuntos não são mais convexos agora.
- Podemos contornar isso usando *convexificação*.

Convexificação:

- Substituímos $\mathcal{X}^i$ por $\mathcal{C}^i = conv(\mathcal{X}^i)$, o menor conjunto convexo contendo todos os pontos de $\mathcal{X}^i$;

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição

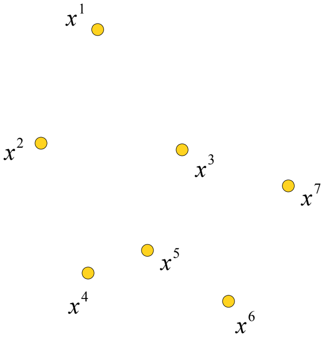
- Podemos usar o Teorema da Representação?
  Não! Os subconjuntos não são mais convexos agora.
- Podemos contornar isso usando *convexificação*.

Convexificação:

- Substituímos $\mathcal{X}^i$ por $\mathcal{C}^i = conv(\mathcal{X}^i)$, o menor conjunto convexo contendo todos os pontos de $\mathcal{X}^i$;
- Quando $\mathcal{X}^i$ é limitado, temos que $\mathcal{C}^i$ é dado pelo envoltório convexo dos pontos de $\mathcal{X}^i$.

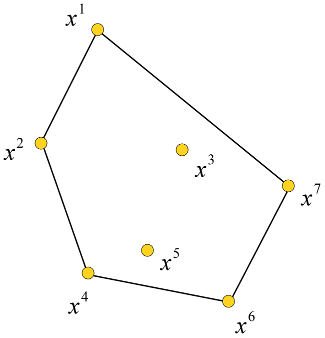
Subproblemas discretos

▷ Convexificação



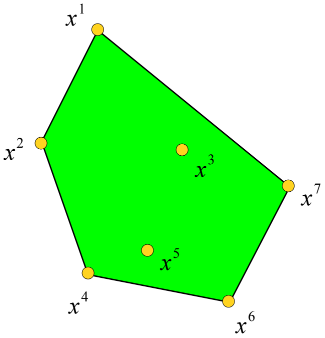
Subproblemas discretos

▷ Convexificação



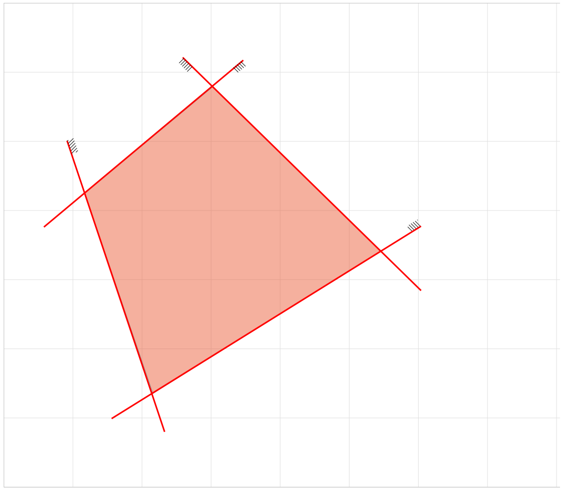
Subproblemas discretos

▷ Convexificação



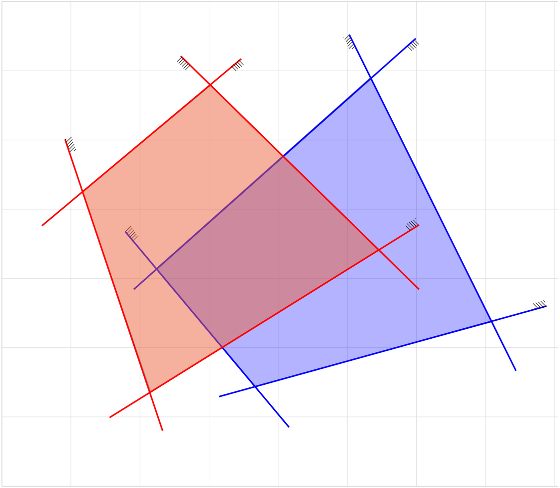
Subproblemas discretos

▷ Convexificação



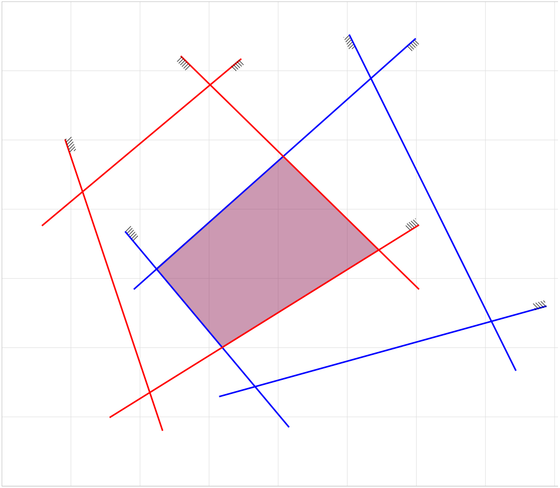
Subproblemas discretos

▷ Convexificação



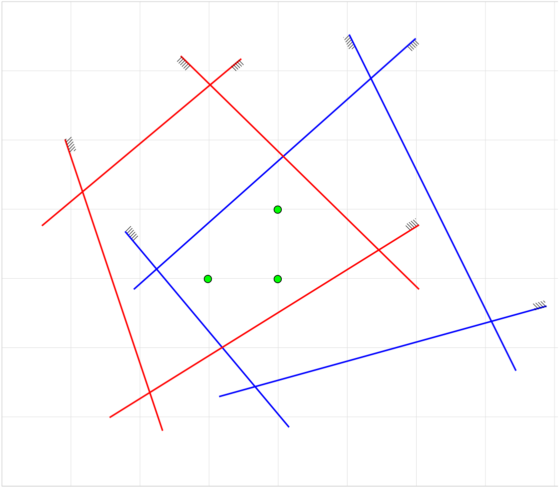
Subproblemas discretos

▷ Convexificação



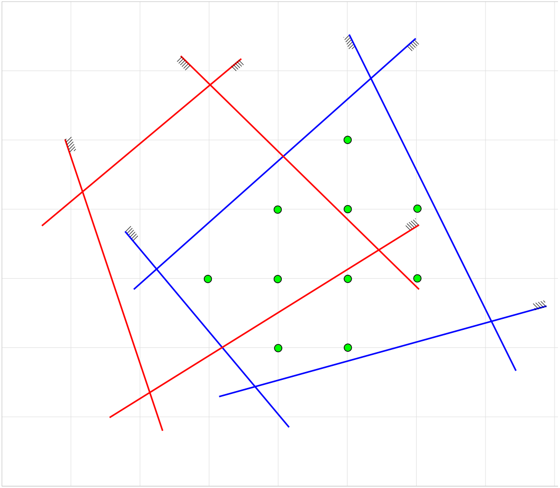
Subproblemas discretos

▷ Convexificação



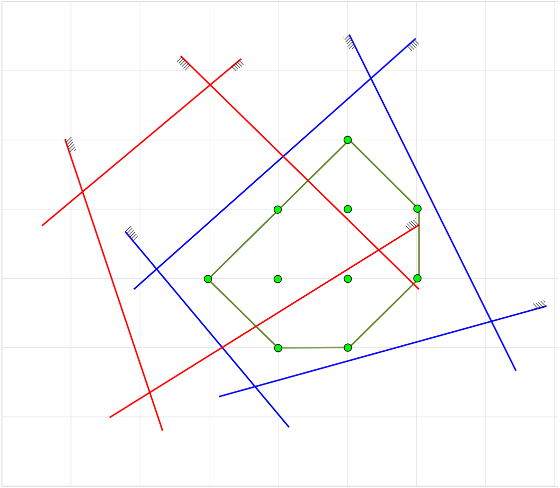
Subproblemas discretos

▷ Convexificação



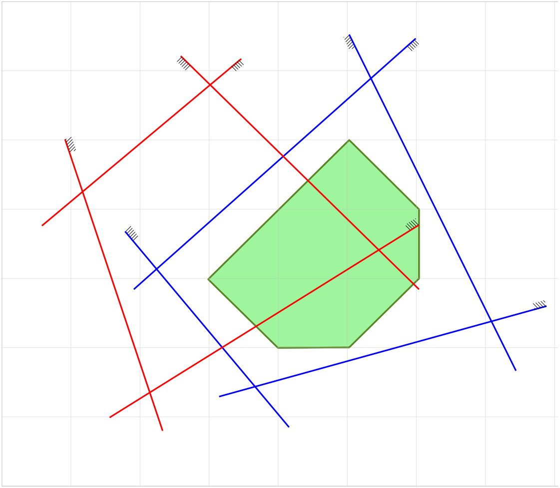
Subproblemas discretos

▷ Convexificação



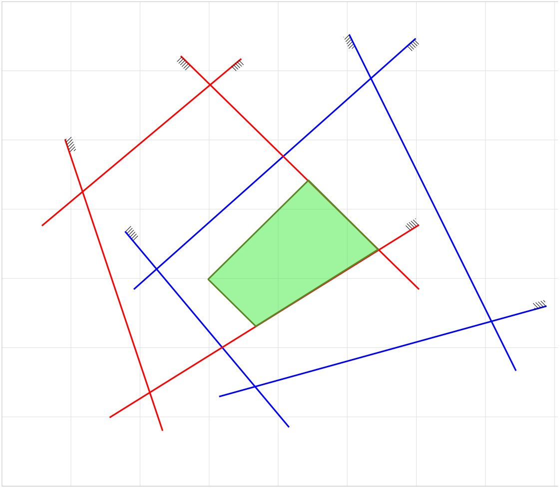
Subproblemas discretos

▷ Convexificação



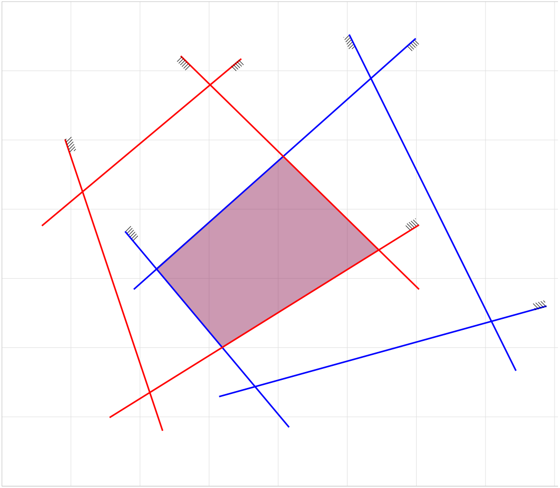
Subproblemas discretos

▷ Convexificação



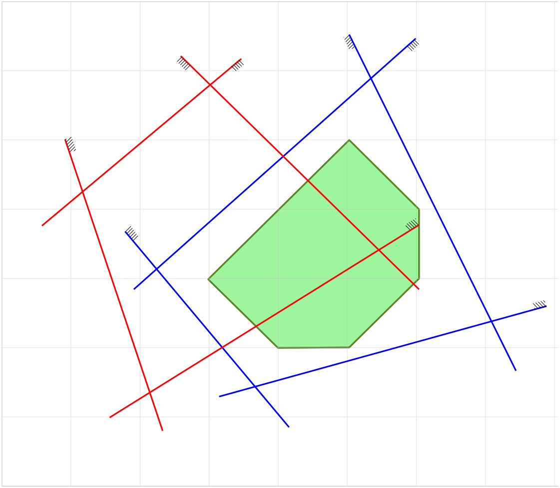
Subproblemas discretos

▷ Convexificação



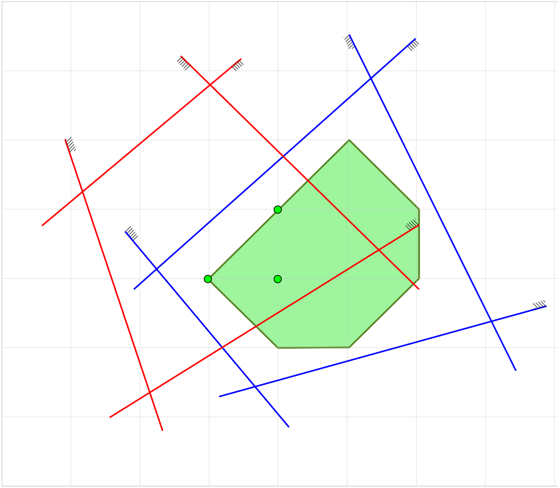
Subproblemas discretos

▷ Convexificação



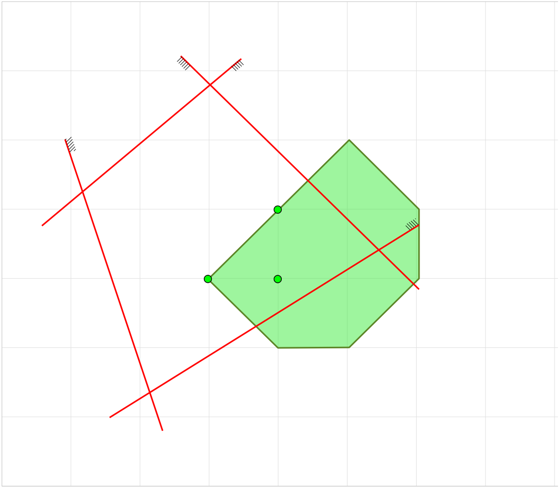
Subproblemas discretos

▷ Convexificação



Subproblemas discretos

▷ Convexificação



PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

- Qualquer $(x_i, I_i, y_i) \in \mathcal{C}^i$ pode ser escrito como:

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

- Qualquer $(x_i, I_i, y_i) \in \mathcal{C}^i$ pode ser escrito como:

$$(x_i, I_i, y_i) = \sum_{q \in Q^i} \lambda_q^i (\bar{x}_{qi}, \bar{I}_{qi}, \bar{y}_{qi}) + \sum_{r \in R^i} \mu_r^i (\bar{x}_{ri}, \bar{I}_{ri}, \bar{y}_{ri}),$$

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

- Qualquer $(x_i, I_i, y_i) \in \mathcal{C}^i$ pode ser escrito como:

$$(x_i, I_i, y_i) = \sum_{q \in Q^i} \lambda_q^i (\bar{x}_{qi}, \bar{I}_{qi}, \bar{y}_{qi}) + \sum_{r \in R^i} \mu_r^i (\bar{x}_{ri}, \bar{I}_{ri}, \bar{y}_{ri}),$$

$$\sum_{q \in Q^i} \lambda_q^i = 1, \lambda_q^i \geq 0, \mu_r^i \geq 0, \ i = 1, \ldots, n,$$

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

- Qualquer $(x_i, I_i, y_i) \in \mathcal{C}^i$ pode ser escrito como:

$$(x_i, I_i, y_i) = \sum_{q \in Q^i} \lambda_q^i (\bar{x}_{qi}, \bar{I}_{qi}, \bar{y}_{qi}) + \sum_{r \in R^i} \mu_r^i (\bar{x}_{ri}, \bar{I}_{ri}, \bar{y}_{ri}),$$

$$\sum_{q \in Q^i} \lambda_q^i = 1, \lambda_q^i \geq 0, \mu_r^i \geq 0, \; i = 1, \ldots, n,$$

sendo $(\bar{x}_{qi}, \bar{I}_{qi}, \bar{y}_{qi})$, $q \in Q^i$, os pontos extremos de $\mathcal{C}^i$

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

- Qualquer $(x_i, I_i, y_i) \in \mathcal{C}^i$ pode ser escrito como:

$$(x_i, I_i, y_i) = \sum_{q \in Q^i} \lambda_q^i (\bar{x}_{qi}, \bar{I}_{qi}, \bar{y}_{qi}) + \sum_{r \in R^i} \mu_r^i (\bar{x}_{ri}, \bar{I}_{ri}, \bar{y}_{ri}),$$

$$\sum_{q \in Q^i} \lambda_q^i = 1, \lambda_q^i \geq 0, \mu_r^i \geq 0, \; i = 1, \ldots, n,$$

sendo $(\bar{x}_{qi}, \bar{I}_{qi}, \bar{y}_{qi})$, $q \in Q^i$, os pontos extremos de $\mathcal{C}^i$ e $(\bar{x}_{ri}, \bar{I}_{ri}, \bar{y}_{ri})$, $r \in R^i$, os raios extremos de $\mathcal{C}^i$;

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

- Qualquer $(x_i, I_i, y_i) \in \mathcal{C}^i$ pode ser escrito como:

$$(x_i, I_i, y_i) = \sum_{q \in Q^i} \lambda_q^i (\bar{x}_{qi}, \bar{I}_{qi}, \bar{y}_{qi}) + \sum_{r \in R^i} \mu_r^i (\bar{x}_{ri}, \bar{I}_{ri}, \bar{y}_{ri}),$$

$$\sum_{q \in Q^i} \lambda_q^i = 1, \lambda_q^i \geq 0, \mu_r^i \geq 0,\ i = 1, \ldots, n,$$

  sendo $(\bar{x}_{qi}, \bar{I}_{qi}, \bar{y}_{qi})$, $q \in Q^i$, os pontos extremos de $\mathcal{C}^i$ e $(\bar{x}_{ri}, \bar{I}_{ri}, \bar{y}_{ri})$, $r \in R^i$, os raios extremos de $\mathcal{C}^i$;

- Neste caso particular do PDL com preparação, o subproblema é limitado e, assim, $\mathcal{C}^i$ é descrito apenas por seus pontos extremos.

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

$$
\mathcal{X}^i = \{ \ (x_i, I_i, y_i) \quad | \quad \begin{aligned} x_{it} + I_{i,t-1} &= d_{it} + I_{it}, & t &= 1, \ldots, T, \\ x_{it} &\leq C y_{it}, & t &= 1, \ldots, T, \\ I_{i0} &= 0, & & \\ x_{it} \geq 0, I_{it} \geq 0, y_{it} &\in \{0, 1\}, & t &= 1, \ldots, T\} \end{aligned}
$$

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

$$\mathcal{X}^i = \{\ (x_i, I_i, y_i) \quad | \quad \begin{array}{rl} x_{it} + I_{i,t-1} = d_{it} + I_{it}, & t = 1, \dots, T, \\ x_{it} \leq C y_{it}, & t = 1, \dots, T, \\ I_{i0} = 0, & \\ x_{it} \geq 0, I_{it} \geq 0, y_{it} \in \{0, 1\}, & t = 1, \dots, T\} \end{array}$$

$$\mathcal{C}^i = conv(\mathcal{X}^i)$$

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

$$\mathcal{X}^i = \{ \ (x_i, I_i, y_i) \quad | \quad \begin{aligned} x_{it} + I_{i,t-1} &= d_{it} + I_{it}, & t &= 1, \dots, T, \\ x_{it} &\leq C y_{it}, & t &= 1, \dots, T, \\ I_{i0} &= 0, & & \\ x_{it} \geq 0, I_{it} \geq 0, y_{it} &\in \{0, 1\}, & t &= 1, \dots, T\} \end{aligned}$$

$$\mathcal{C}^i = conv(\mathcal{X}^i)$$

$$\begin{aligned} \text{min} \quad & \sum_{i=1}^{n} \sum_{t=1}^{T} c_{it} x_{it} + \sum_{i=1}^{n} \sum_{t=1}^{T} h_{it} I_{it} + \sum_{i=1}^{n} \sum_{t=1}^{T} s_{it} y_{it} \\ \text{s.a} \quad & \sum_{i=1}^{n} (a_i x_{it} + st_i y_{it}) \leq b_t, \qquad t = 1, \dots, T, \\ & (x_i, I_i, y_i) \in \mathcal{C}^i, \qquad i = 1, \dots, n \end{aligned}$$

PDL com restrição de capacidade e tempo de preparação

▷ Decomposição: convexificação

$$\mathcal{X}^i = \{ \ (x_i, I_i, y_i) \quad | \quad \begin{array}{rl} x_{it} + I_{i,t-1} = d_{it} + I_{it}, & t = 1, \dots, T, \\ x_{it} \leq C y_{it}, & t = 1, \dots, T, \\ I_{i0} = 0, & \\ x_{it} \geq 0, I_{it} \geq 0, y_{it} \in \{0,1\}, & t = 1, \dots, T\} \end{array}$$

$$\mathcal{C}^i = conv(\mathcal{X}^i)$$

$$\begin{array}{lll} \text{min} & \sum_{i=1}^{n} \sum_{t=1}^{T} c_{it} x_{it} + \sum_{i=1}^{n} \sum_{t=1}^{T} h_{it} I_{it} + \sum_{i=1}^{n} \sum_{t=1}^{T} s_{it} y_{it} & \\ \text{s.a} & \sum_{i=1}^{n} (a_i x_{it} + st_i y_{it}) \leq b_t, & t = 1, \dots, T, \\ & (x_i, I_i, y_i) \in \mathcal{C}^i, & i = 1, \dots, n, \\ & y_{it} \in \{0,1\}, & i = 1, \dots, n, \ t = 1, \dots, T. \end{array}$$

PDL com restrição de capacidade e tempo de preparação

▷ Formulação extensiva equivalente

Reescrevendo (x_i, I_i, y_i) como uma combinação convexa dos pontos extremos de $\mathcal{C}^i$, obtemos o *Problema Mestre*:

PDL com restrição de capacidade e tempo de preparação

▷ Formulação extensiva equivalente

Reescrevendo (x_i, I_i, y_i) como uma combinação convexa dos pontos extremos de $\mathcal{C}^i$, obtemos o *Problema Mestre*:

$$
\begin{aligned}
\text{min} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} \sum_{t=1}^{T} (c_{it}\bar{x}_{qit} + h_{it}\bar{I}_{qit} + s_{it}\bar{y}_{qit})\lambda_q^i \\
\text{s.a} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} (a_i\bar{x}_{qit} + st_i\bar{y}_{qit})\lambda_q^i \leq b_t, && t = 1, \ldots, T, \\
& \sum_{q \in Q^i} \lambda_q^i = 1, && i = 1, \ldots, n, \\
& \lambda_q^i \geq 0, && i = 1, \ldots, n, \ q \in Q_i
\end{aligned}
$$

PDL com restrição de capacidade e tempo de preparação

▷ Formulação extensiva equivalente

Reescrevendo (x_i, I_i, y_i) como uma combinação convexa dos pontos extremos de $\mathcal{C}^i$, obtemos o *Problema Mestre*:

$$
\begin{array}{lll}
\min & \displaystyle\sum_{i=1}^{n}\sum_{q\in Q^i}\sum_{t=1}^{T}(c_{it}\bar{x}_{qit} + h_{it}\bar{I}_{qit} + s_{it}\bar{y}_{qit})\lambda_q^i & \\
\text{s.a} & \displaystyle\sum_{i=1}^{n}\sum_{q\in Q^i}(a_i\bar{x}_{qit} + st_i\bar{y}_{qit})\lambda_q^i \le b_t, & t = 1,\dots,T, \\
& \displaystyle\sum_{q\in Q^i}\lambda_q^i = 1, & i = 1,\dots,n, \\
& \lambda_q^i \ge 0, & i = 1,\dots,n,\ q \in Q_i, \\
& y_{it} \in \{0,1\}, & t = 1,\dots,T,\ i = 1,\dots,n
\end{array}
$$

PDL com restrição de capacidade e tempo de preparação

▷ Formulação extensiva equivalente

Reescrevendo (x_i, I_i, y_i) como uma combinação convexa dos pontos extremos de $\mathcal{C}^i$, obtemos o *Problema Mestre*:

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} \sum_{t=1}^{T} (c_{it}\bar{x}_{qit} + h_{it}\bar{I}_{qit} + s_{it}\bar{y}_{qit})\lambda_q^i & & \\
\text{s.a} \quad & \sum_{i=1}^{n} \sum_{q \in Q^i} (a_i\bar{x}_{qit} + st_i\bar{y}_{qit})\lambda_q^i \le b_t, & & t = 1, \dots, T, \\
& \sum_{q \in Q^i} \lambda_q^i = 1, & & i = 1, \dots, n, \\
& \lambda_q^i \ge 0, & & i = 1, \dots, n,\ q \in Q_i, \\
& y_{it} \in \{0, 1\}, & & t = 1, \dots, T,\ i = 1, \dots, n, \\
& y_{it} = \sum_{q \in Q^i} \lambda_q^i \bar{y}_{qit}, & & t = 1, \dots, T,\ i = 1, \dots, n.
\end{aligned}
$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

$$
\begin{array}{rll}
\text{min} & f(x) = c^T x, & \\
\text{s.a} & Ax = b, & \text{(restrições de acoplamento)} \\
& Dx = d, & \text{(estrutura especial)} \\
& x \in \mathbb{Z}^n_+, &
\end{array}
$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

$$
\begin{array}{lll}
\text{min} & f(x) = c^T x, & \\
\text{s.a} & Ax = b, & \text{(restrições de acoplamento)} \\
 & x \in \mathcal{X}, & \text{(estrutura especial)}
\end{array}
$$

$$\mathcal{X} = \{x \in \mathbb{Z}^n \mid Dx = d, x \geq 0\}$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- $\mathcal{C} = conv(\mathcal{X})$: menor conjunto convexo contendo todos os pontos de $\mathcal{X}$;

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- $\mathcal{C} = conv(\mathcal{X})$: menor conjunto convexo contendo todos os pontos de $\mathcal{X}$;
- Se $\mathcal{X}$ é limitado, então $\mathcal{C}$ é o envoltório convexo dos pontos de $\mathcal{X}$;

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- $\mathcal{C} = conv(\mathcal{X})$: menor conjunto convexo contendo todos os pontos de $\mathcal{X}$;
- Se $\mathcal{X}$ é limitado, então $\mathcal{C}$ é o envoltório convexo dos pontos de $\mathcal{X}$;

$$
\begin{aligned}
\text{min} \quad & f(x) = c^T x, \\
\text{s.a} \quad & Ax = b, \\
& \color{red}{x \in \mathcal{C}}
\end{aligned}
$$

Decomposição de Dantzig-Wolfe

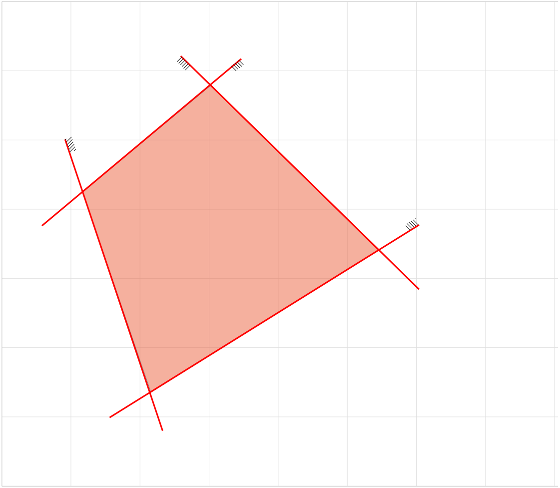
▷ Problemas de otimização discreta: Convexificação

- $\mathcal{C} = conv(\mathcal{X})$: menor conjunto convexo contendo todos os pontos de $\mathcal{X}$;
- Se $\mathcal{X}$ é limitado, então $\mathcal{C}$ é o envoltório convexo dos pontos de $\mathcal{X}$;

$$\begin{aligned} \text{min} \quad & f(x) = c^T x, \\ \text{s.a} \quad & Ax = b, \\ & x \in \mathcal{C}, \\ & x \in \mathbb{Z}_+^n \end{aligned}$$

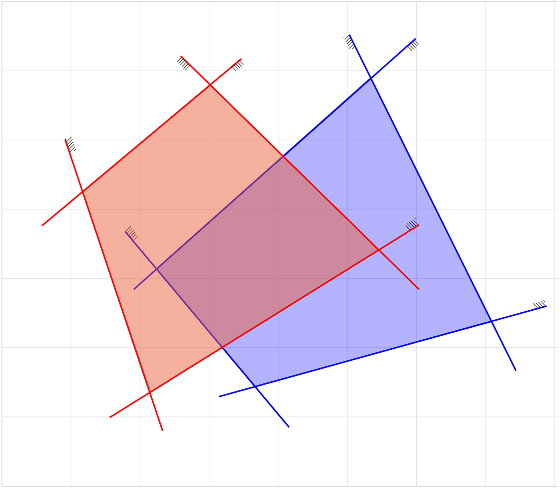
Subproblemas discretos

▷ Convexificação



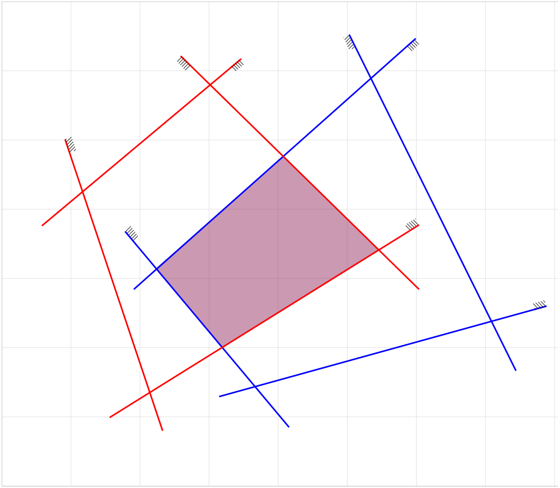
Subproblemas discretos

▷ Convexificação



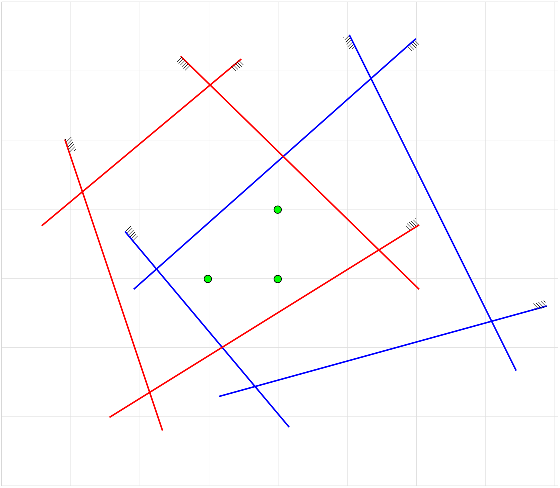
Subproblemas discretos

▷ Convexificação



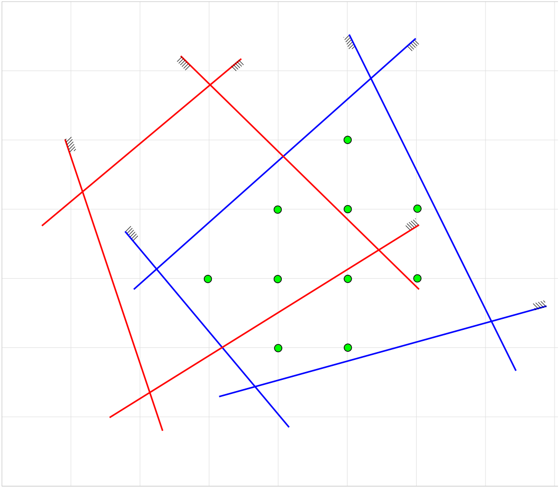
Subproblemas discretos

▷ Convexificação



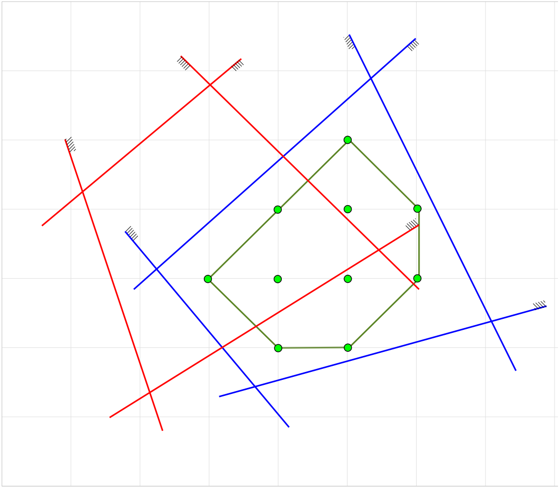
Subproblemas discretos

▷ Convexificação



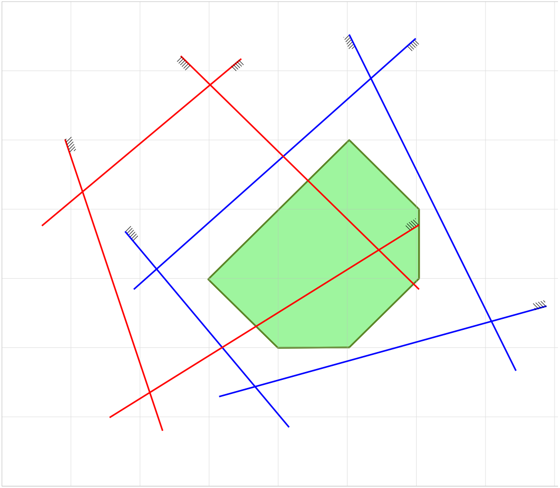
Subproblemas discretos

▷ Convexificação



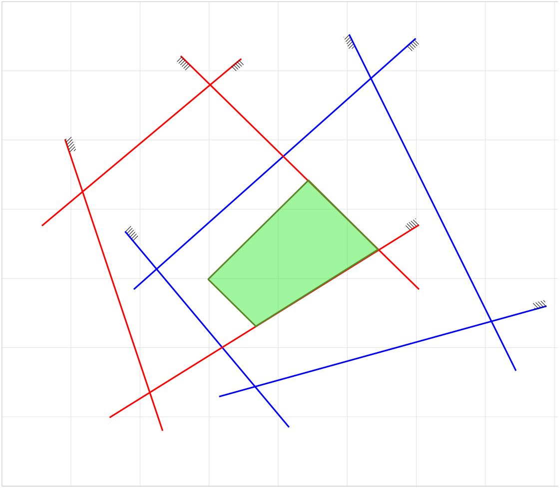
Subproblemas discretos

▷ Convexificação



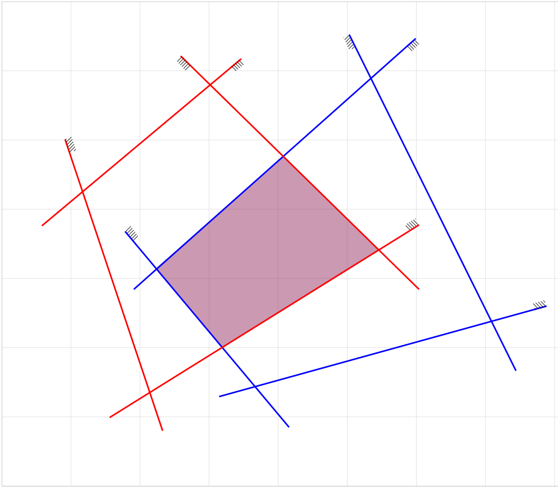
Subproblemas discretos

▷ Convexificação



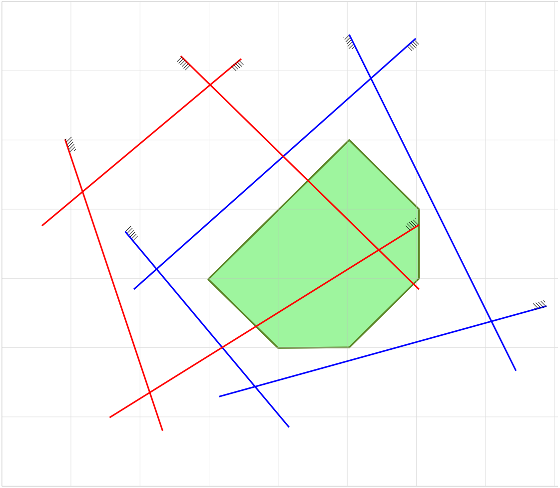
Subproblemas discretos

▷ Convexificação



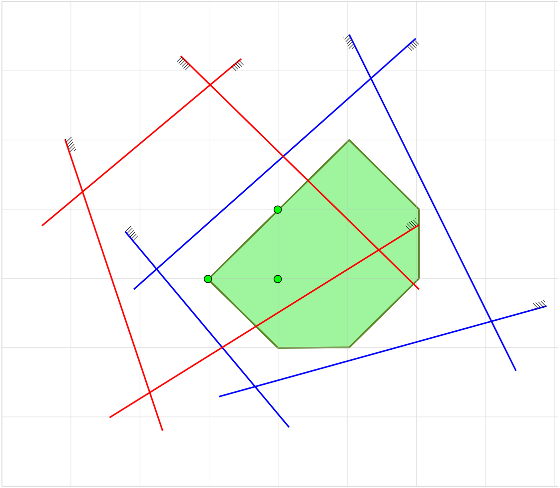
Subproblemas discretos

▷ Convexificação



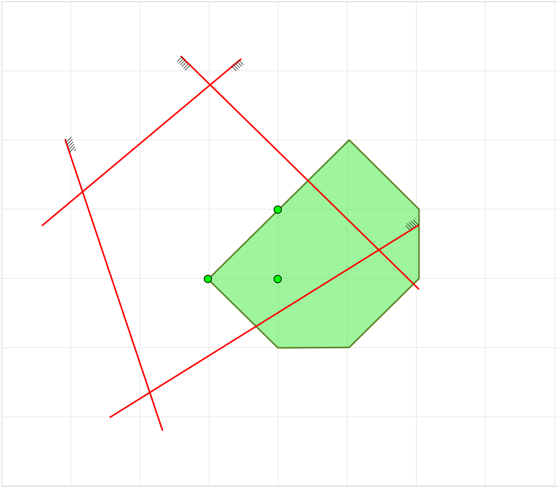
Subproblemas discretos

▷ Convexificação



Subproblemas discretos

▷ Convexificação



Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- Pelo Teorema da Representação, todo $x \in \mathcal{C} = conv(\mathcal{X})$ pode ser escrito como uma combinação de pontos extremos e raios extremos de $\mathcal{C}$:

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- Pelo Teorema da Representação, todo $x \in \mathcal{C} = conv(\mathcal{X})$ pode ser escrito como uma combinação de pontos extremos e raios extremos de $\mathcal{C}$:

$$x = \sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r,$$

$$\sum_{q \in Q} \lambda_q = 1,$$

$$\lambda_q \geq 0, \mu_r \geq 0, \; q \in Q, \; r \in R,$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- Pelo Teorema da Representação, todo $x \in \mathcal{C} = conv(\mathcal{X})$ pode ser escrito como uma combinação de pontos extremos e raios extremos de $\mathcal{C}$:

$$x = \sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r,$$

$$\sum_{q \in Q} \lambda_q = 1,$$

$$\lambda_q \geq 0, \mu_r \geq 0, \; q \in Q, \; r \in R,$$

sendo $\bar{x}_q$, $q \in Q$, os pontos extremos de $\mathcal{C}$ e $\bar{x}_r$, $r \in R$, os raios extremos de $\mathcal{C}$.

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- Pelo Teorema da Representação, todo $x \in \mathcal{C} = conv(\mathcal{X})$ pode ser escrito como uma combinação de pontos extremos e raios extremos de $\mathcal{C}$:

$$x = \sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r,$$

$$\sum_{q \in Q} \lambda_q = 1,$$

$$\lambda_q \geq 0, \mu_r \geq 0, \; q \in Q, \; r \in R,$$

sendo $\bar{x}_q$, $q \in Q$, os pontos extremos de $\mathcal{C}$ e $\bar{x}_r$, $r \in R$, os raios extremos de $\mathcal{C}$. Substituindo no problema original...

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

Problema Mestre:

$$\min \quad c^T \left(\sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r \right)$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

Problema Mestre:

$$\begin{aligned}
\text{min} \quad & c^T \left(\sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r \right) \\
\text{s.a} \quad & A \left(\sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r \right) = b,
\end{aligned}$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

Problema Mestre:

$$
\begin{aligned}
\text{min} \quad & c^T \left(\sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r \right) \\
\text{s.a} \quad & A \left(\sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r \right) = b, \\
& \sum_{q \in Q} \lambda_q = 1,
\end{aligned}
$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

Problema Mestre:

$$
\begin{aligned}
\text{min} \quad & c^T \left(\sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r \right) \\
\text{s.a} \quad & A \left(\sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r \right) = b, \\
& \sum_{q \in Q} \lambda_q = 1, \\
& \lambda_q \geq 0, \mu_r \geq 0, \qquad q \in Q, r \in R
\end{aligned}
$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

Problema Mestre:

$$
\begin{aligned}
\text{min} \quad & c^T \left(\sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r \right) \\
\text{s.a} \quad & A \left(\sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r \right) = b, \\
& \sum_{q \in Q} \lambda_q = 1, \\
& \lambda_q \geq 0, \mu_r \geq 0, \qquad\qquad q \in Q, r \in R, \\
& x \in \mathbb{Z}^n_+
\end{aligned}
$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

Problema Mestre:

$$
\begin{aligned}
\text{min} \quad & c^T\left(\sum_{q\in Q}\lambda_q\bar{x}_q + \sum_{r\in R}\mu_r\bar{x}_r\right) \\
\text{s.a} \quad & A\left(\sum_{q\in Q}\lambda_q\bar{x}_q + \sum_{r\in R}\mu_r\bar{x}_r\right) = b, \\
& \sum_{q\in Q}\lambda_q = 1, \\
& \lambda_q \geq 0, \mu_r \geq 0, \qquad q\in Q, r\in R, \\
& x \in \mathbb{Z}^n_+, \\
& x = \sum_{q\in Q}\lambda_q\bar{x}_q + \sum_{r\in R}\mu_r\bar{x}_r.
\end{aligned}
$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

Problema Mestre:

$$
\begin{aligned}
\text{min} \quad & \sum_{q \in Q} c_q \lambda_q + \sum_{r \in R} c_r \mu_r \\
\text{s.a} \quad & \sum_{q \in Q} a_q \lambda_q + \sum_{r \in R} a_r \mu_r = b, \\
& \sum_{q \in Q} \lambda_q = 1, \\
& \lambda_q \geq 0, \mu_r \geq 0, \qquad q \in Q, r \in R, \\
& x \in \mathbb{Z}^n_+, \\
& x = \sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r.
\end{aligned}
$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- As duas últimas restrições têm o intuito apenas de garantir que a combinação dos pontos extremos e raios extremos resulte em um ponto x que seja inteiro;

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- As duas últimas restrições têm o intuito apenas de garantir que a combinação dos pontos extremos e raios extremos resulte em um ponto x que seja inteiro;
- A relaxação linear do PM pode ser escrita como:

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Convexificação

- As duas últimas restrições têm o intuito apenas de garantir que a combinação dos pontos extremos e raios extremos resulte em um ponto x que seja inteiro;
- A relaxação linear do PM pode ser escrita como:

$$
\begin{aligned}
\text{min} \quad & \sum_{q \in Q} c_q \lambda_q + \sum_{r \in R} c_r \mu_r \\
\text{s.a} \quad & \sum_{q \in Q} a_q \lambda_q + \sum_{r \in R} a_r \mu_r = b, \\
& \sum_{q \in Q} \lambda_q = 1, \\
& \lambda_q \geq 0, \mu_r \geq 0, \qquad q \in Q, r \in R.
\end{aligned}
$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

- $\mathcal{D} = discr(\mathcal{X})$: conjunto contendo todos os pontos inteiros e raios extremos inteiros de $\mathcal{X}$;

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

- $\mathcal{D} = discr(\mathcal{X})$: conjunto contendo todos os pontos inteiros e raios extremos inteiros de $\mathcal{X}$;
  Obs.: Um raio extremo inteiro é um raio extremo com coordenadas inteiras; (é sempre possível mudar a escala de um raio)

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

- $\mathcal{D} = discr(\mathcal{X})$: conjunto contendo todos os pontos <u>inteiros</u> e raios extremos <u>inteiros</u> de $\mathcal{X}$;
  Obs.: Um raio extremo inteiro é um raio extremo com coordenadas inteiras; (é sempre possível mudar a escala de um raio)
- Se $\mathcal{X}$ é limitado, então $\mathcal{D}$ é a <u>enumeração</u> dos pontos factíveis de $\mathcal{X}$;

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

- $\mathcal{D} = discr(\mathcal{X})$: conjunto contendo todos os pontos inteiros e raios extremos inteiros de $\mathcal{X}$;
  Obs.: Um raio extremo inteiro é um raio extremo com coordenadas inteiras; (é sempre possível mudar a escala de um raio)
- Se $\mathcal{X}$ é limitado, então $\mathcal{D}$ é a enumeração dos pontos factíveis de $\mathcal{X}$;
- O problema original pode então ser reescrito como:

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

- $\mathcal{D} = discr(\mathcal{X})$: conjunto contendo todos os pontos <u>inteiros</u> e raios extremos <u>inteiros</u> de $\mathcal{X}$;
  Obs.: Um raio extremo inteiro é um raio extremo com coordenadas inteiras; (é sempre possível mudar a escala de um raio)
- Se $\mathcal{X}$ é limitado, então $\mathcal{D}$ é a <u>enumeração</u> dos pontos factíveis de $\mathcal{X}$;
- O problema original pode então ser reescrito como:

$$\begin{aligned} \text{min} \quad & f(x) = c^T x, \\ \text{s.a} \quad & Ax = b, \\ & x \in \mathcal{D} \end{aligned}$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

- Todo $x \in \mathcal{D} = discr(\mathcal{X})$ pode ser escrito como uma combinação "inteira" de pontos e raios de $\mathcal{D}$:

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

- Todo $x \in \mathcal{D} = discr(\mathcal{X})$ pode ser escrito como uma combinação "inteira" de pontos e raios de $\mathcal{D}$:

$$x = \sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r,$$

$$\sum_{q \in Q} \lambda_q = 1,$$

$$\lambda_q \in \{0,1\}, \mu_r \in \mathbb{Z}_+, \; q \in Q, \; r \in R,$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

- Todo $x \in \mathcal{D} = discr(\mathcal{X})$ pode ser escrito como uma combinação "inteira" de pontos e raios de $\mathcal{D}$:

$$x = \sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r,$$

$$\sum_{q \in Q} \lambda_q = 1,$$

$$\lambda_q \in \{0,1\}, \mu_r \in \mathbb{Z}_+, \; q \in Q, \; r \in R,$$

sendo $\bar{x}_q$, $q \in Q$, os pontos inteiros de $\mathcal{C}$ e $\bar{x}_r$, $r \in R$, os raios extremos inteiros de $\mathcal{C}$.

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

- Todo $x \in \mathcal{D} = discr(\mathcal{X})$ pode ser escrito como uma combinação "inteira" de pontos e raios de $\mathcal{D}$:

$$x = \sum_{q \in Q} \lambda_q \bar{x}_q + \sum_{r \in R} \mu_r \bar{x}_r,$$

$$\sum_{q \in Q} \lambda_q = 1,$$

$$\lambda_q \in \{0, 1\}, \mu_r \in \mathbb{Z}_+, \ q \in Q, \ r \in R,$$

sendo $\bar{x}_q$, $q \in Q$, os pontos inteiros de $\mathcal{C}$ e $\bar{x}_r$, $r \in R$, os raios extremos inteiros de $\mathcal{C}$. Substituindo no problema original...

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

Problema Mestre:

$$
\begin{aligned}
\min \quad & \sum_{q \in Q} c_q \lambda_q + \sum_{r \in R} c_r \mu_r \\
\text{s.a} \quad & \sum_{q \in Q} a_q \lambda_q + \sum_{r \in R} a_r \mu_r = b, \\
& \sum_{q \in Q} \lambda_q = 1, \\
& \lambda_q \in \{0,1\}, \qquad q \in Q, \\
& \mu_r \in \mathbb{Z}_+, \qquad r \in R.
\end{aligned}
$$

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta: Discretização

Problema Mestre:

$$
\begin{aligned}
\min \quad & \sum_{q \in Q} c_q \lambda_q + \sum_{r \in R} c_r \mu_r \\
\text{s.a} \quad & \sum_{q \in Q} a_q \lambda_q + \sum_{r \in R} a_r \mu_r = b, \\
& \sum_{q \in Q} \lambda_q = 1, \\
& \lambda_q \in \{0,1\}, \qquad q \in Q, \\
& \mu_r \in \mathbb{Z}_+, \qquad r \in R.
\end{aligned}
$$

- Observe que a relaxação linear do PM obtido por discretização é igual à do PM obtido por convexificação.

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

- Em ambos os casos, o Problema Mestre é um problema de programação inteira

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

- Em ambos os casos, o Problema Mestre é um problema de programação inteira com um número muito grande de variáveis;

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

- Em ambos os casos, o Problema Mestre é um problema de programação inteira com um número muito grande de variáveis;
- Assim, para resolvê-lo usamos uma combinação do método de geração de colunas

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

- Em ambos os casos, o Problema Mestre é um problema de programação inteira com um número muito grande de variáveis;
- Assim, para resolvê-lo usamos uma combinação do método de geração de colunas com o método *branch-and-bound*,

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

- Em ambos os casos, o Problema Mestre é um problema de programação inteira com um número muito grande de variáveis;
- Assim, para resolvê-lo usamos uma combinação do método de geração de colunas com o método *branch-and-bound*, resultando no método *branch-and-price*;

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

- Em ambos os casos, o Problema Mestre é um problema de programação inteira com um número muito grande de variáveis;
- Assim, para resolvê-lo usamos uma combinação do método de geração de colunas com o método *branch-and-bound*, resultando no método *branch-and-price*;
- A diferença do método *branch-and-price* para o método *branch-and-bound* está no uso do método de geração de colunas para resolver os problemas em cada nó;

Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

- Em ambos os casos, o Problema Mestre é um problema de programação inteira com um número muito grande de variáveis;
- Assim, para resolvê-lo usamos uma combinação do método de geração de colunas com o método *branch-and-bound*, resultando no método *branch-and-price*;
- A diferença do método *branch-and-price* para o método *branch-and-bound* está no uso do método de geração de colunas para resolver os problemas em cada nó;
- A geração de colunas é aplicada à *relaxação linear* do Problema Mestre (nó raiz) modificado com possíveis restrições de ramificação (demais nós);

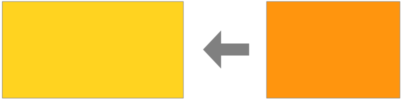
Decomposição de Dantzig-Wolfe

▷ Problemas de otimização discreta

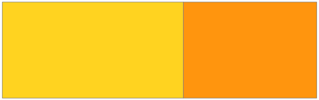
- Em ambos os casos, o Problema Mestre é um problema de programação inteira com um número muito grande de variáveis;
- Assim, para resolvê-lo usamos uma combinação do método de geração de colunas com o método *branch-and-bound*, resultando no método *branch-and-price*;
- A diferença do método *branch-and-price* para o método *branch-and-bound* está no uso do método de geração de colunas para resolver os problemas em cada nó;
- A geração de colunas é aplicada à *relaxação linear* do Problema Mestre (nó raiz) modificado com possíveis restrições de ramificação (demais nós);
- Desigualdades válidas também podem ser usadas, resultando em um método *branch-price-and-cut*.

Branch-and-price e branch-and-cut

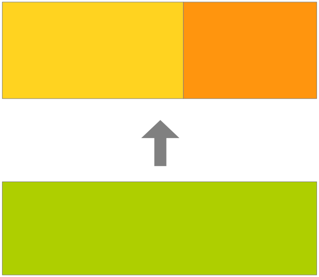
Branch-and-price e branch-price-and-cut



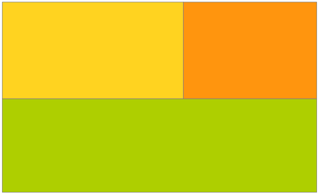
Branch-and-price e branch-price-and-cut



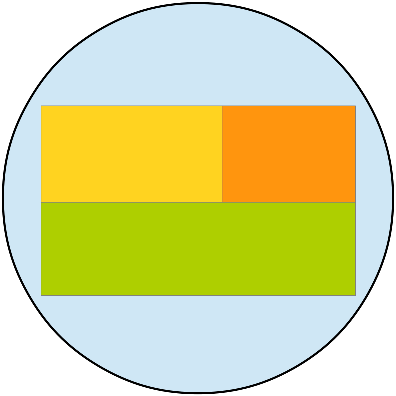
Branch-and-price e branch-price-and-cut



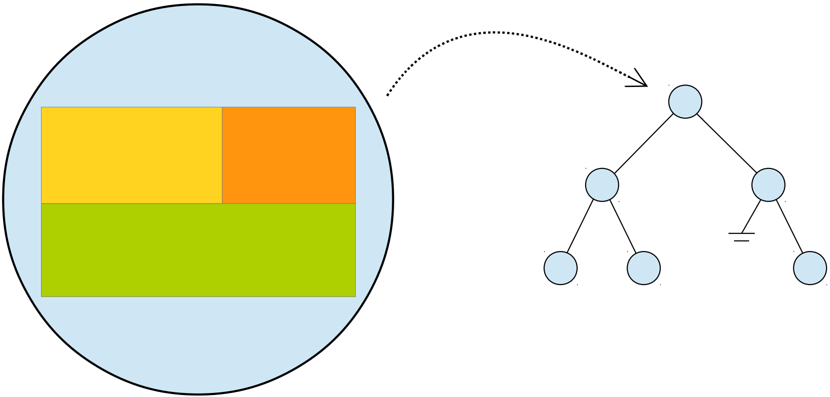
Branch-and-price e branch-price-and-cut



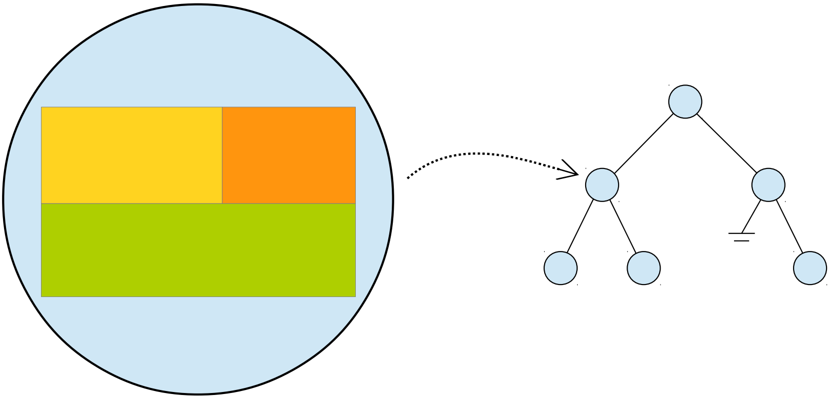
Branch-and-price e branch-and-cut



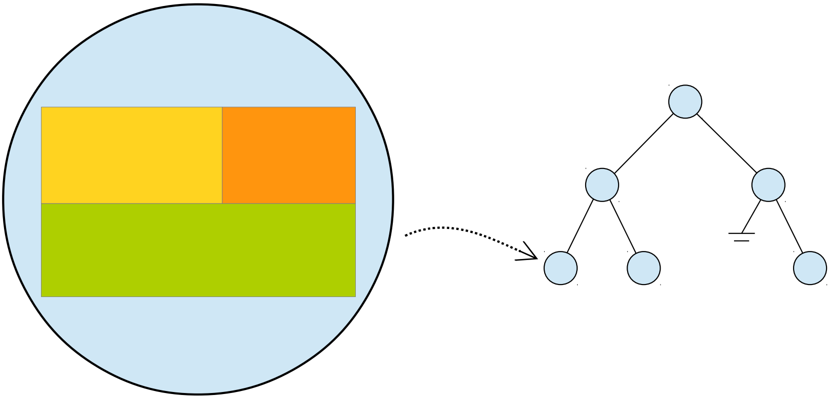
Branch-and-price e branch-price-and-cut



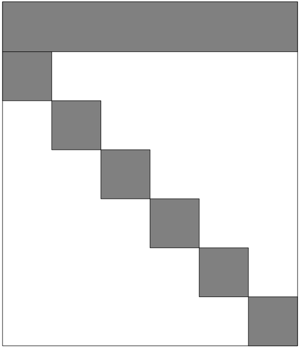
Branch-and-price e branch-price-and-cut



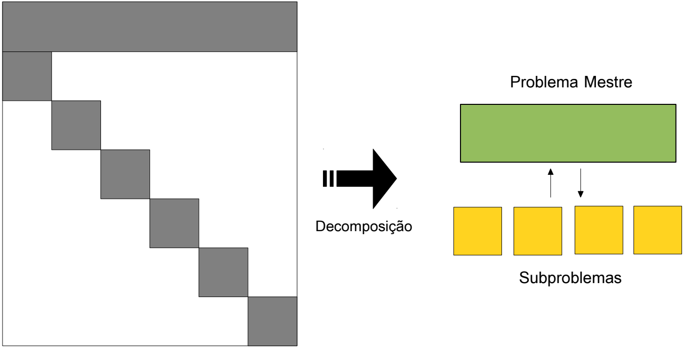
Branch-and-price e branch-price-and-cut



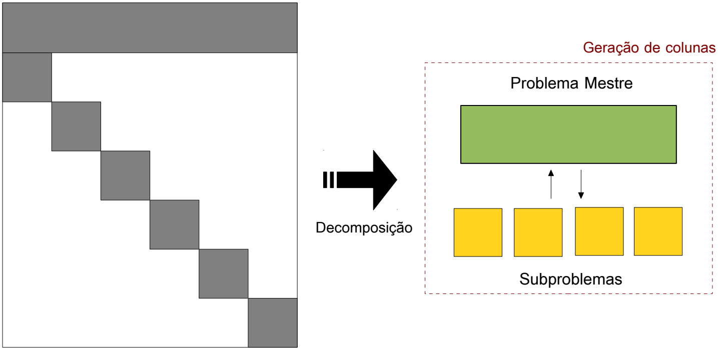
DDW para problemas com variáveis contínuas



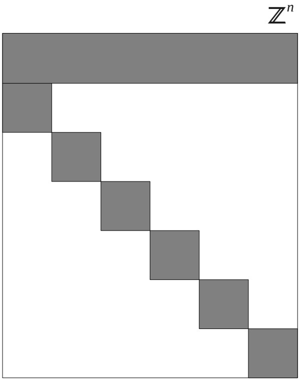
DDW para problemas com variáveis contínuas



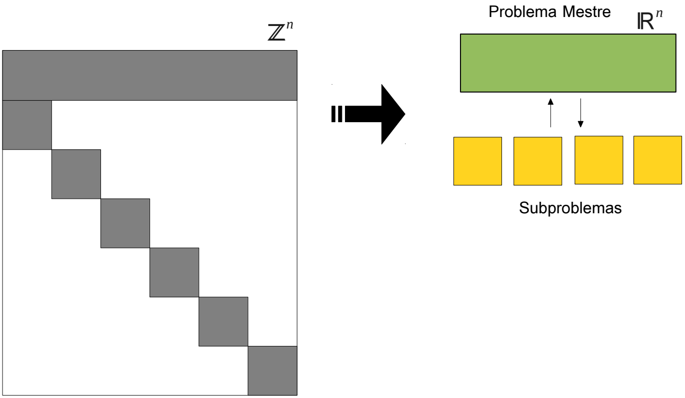
DDW para problemas com variáveis contínuas



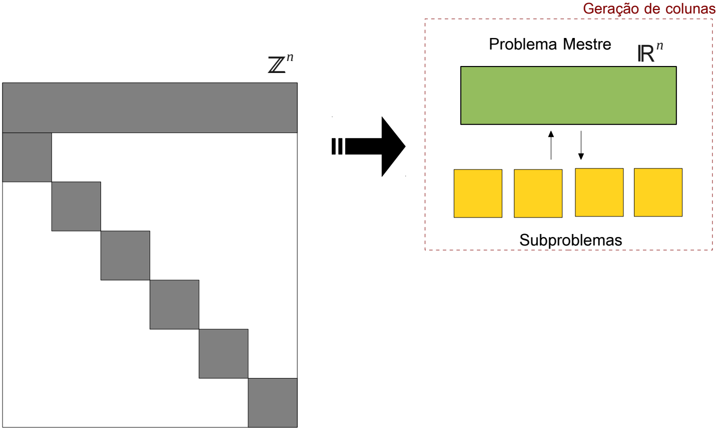
DDW para problemas com variáveis discretas



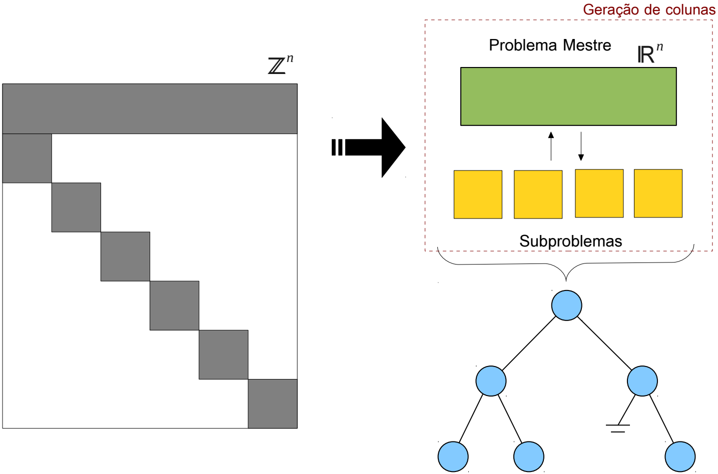
DDW para problemas com variáveis discretas



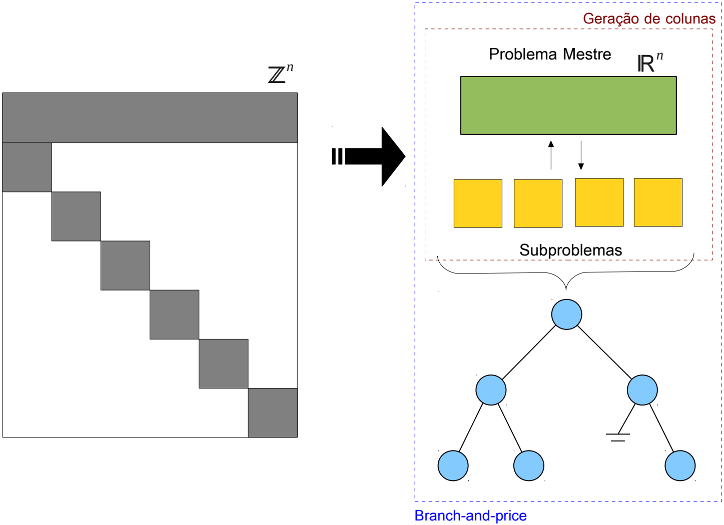
DDW para problemas com variáveis discretas



DDW para problemas com variáveis discretas



DDW para problemas com variáveis discretas



DDW para problemas com variáveis discretas

▷ Barnhart et al., Operations Research 46, 1998

BRANCH-AND-PRICE: COLUMN GENERATION FOR SOLVING HUGE INTEGER PROGRAMS

CYNTHIA BARNHART*, ELLIS L. JOHNSON, GEORGE L. NEMHAUSER, MARTIN W. P. SAVELSBERGH, and PAMELA H. VANCE**

Georgia Institute of Technology, Atlanta, Georgia

(Received February 1994; revisions received May 1995, January 1996; accepted March 1996)

We discuss formulations of integer programs with a huge number of variables and their solution by column generation methods, i.e., implicit pricing of nonbasic variables to generate new columns or to prove LP optimality at a node of the branch-and-bound tree. We present classes of models for which this approach decomposes the problem, provides tighter LP relaxations, and eliminates symmetry. We then discuss computational issues and implementation of column generation, branch-and-bound algorithms, including special branching rules and efficient ways to solve the LP relaxation. We also discuss the relationship with Lagrangian duality.

The successful solution of large-scale mixed integer programming (MIP) problems requires formulations whose linear programming (LP) relaxations give a good approximation to the convex hull of feasible solutions. In the last decade, a great deal of attention has been given to the "branch-and-cut" approach to solving MIPs. Hoffman

enter the basis. If such columns are found, the LP is reoptimized. Branching occurs when no columns price out to enter the basis and the LP solution does not satisfy the integrality conditions. Branch-and-price, which also is a generalization of branch-and-bound with LP relaxations, allows column generation to be applied throughout the

DDW para problemas com variáveis discretas

▷ Lubbecke and Desrosiers, Operations Research 53, 2005

OPERATIONS RESEARCH
Vol. 53, No. 6, November–December 2005, pp. 1007–1023
ISSN 0030-364X | EISSN 1526-5463 | 05 | 5306 | 1007

informs
DOI 10.1287/opre.1050.0234

Selected Topics in Column Generation

Marco E. Lübbecke
Technische Universität Berlin, Institut für Mathematik, Sekr. MA 6-1, Straße des 17. Juni 136,
D-10623 Berlin, Germany, m.luebbecke@math.tu-berlin.de

Jacques Desrosiers
HEC Montréal and GERAD, 3000, chemin de la Côte-Sainte-Catherine, Montréal, Québec, Canada H3T 2A7,
jacques.desrosiers@hec.ca

Dantzig-Wolfe decomposition and column generation, devised for linear programs, is a success story in large-scale integer programming. We outline and relate the approaches, and survey mainly recent contributions, not yet found in textbooks. We emphasize the growing understanding of the dual point of view, which has brought considerable progress to the column generation theory and practice. It stimulated careful initializations, sophisticated solution techniques for the restricted master problem and subproblem, as well as better overall performance. Thus, the dual perspective is an ever recurring concept in our "selected topics."

Subject classifications: integer programming: column generation, Dantzig-Wolfe decomposition, Lagrangian relaxation, branch-and-bound; linear programming: large scale systems.
Area of review: Optimization.
History: Received December 2002; revision received March 2004; accepted October 2004.

DDW para problemas com variáveis discretas

▷ Ralphs and Galati, Math Programmming 106, 2006

Math. Program., Ser. A 106, 261–285 (2006)

Digital Object Identifier (DOI) 10.1007/s10107-005-0606-3

Ted K. Ralphs · Matthew V. Galati

Decomposition and Dynamic Cut Generation in Integer Linear Programming

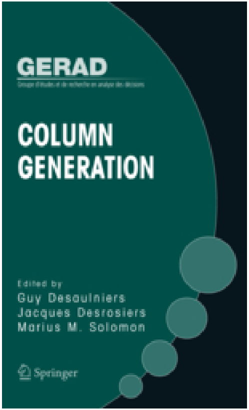
Received: 19 September 2003 / Accepted: 15 April 2005
Published online: August 10, 2005 – © Springer-Verlag 2005

Abstract. Decomposition algorithms such as Lagrangian relaxation and Dantzig-Wolfe decomposition are well-known methods that can be used to generate bounds for mixed-integer linear programming problems. Traditionally, these methods have been viewed as distinct from polyhedral methods, in which bounds are obtained by dynamically generating valid inequalities to strengthen an initial linear programming relaxation. Recently, a number of authors have proposed methods for integrating dynamic cut generation with various decomposition methods to yield further improvement in computed bounds. In this paper, we describe a framework within which most of these methods can be viewed from a common theoretical perspective. We then discuss how the framework can be extended to obtain a decomposition-based separation technique we call *decompose and cut*. As a by-product, we describe how these methods can take advantage of the fact that solutions with known structure, such as those to a given relaxation, can frequently be separated much more easily than arbitrary real vectors.

Key words. Integer Programming – Dantzig-Wolfe Decomposition – Lagrangian Relaxation – Branch and Cut – Branch and Price – Decomposition Algorithms

DDW para problemas com variáveis discretas

▷ Desaulniers et al., Column Generation, 2005



DDW para problemas com variáveis discretas

▷ Ramificação

- Em métodos *branch-and-price* e *branch-price-and-cut*, a ramificação deve garantir que as soluções obtidas tenham valores inteiros;

DDW para problemas com variáveis discretas

▷ Ramificação

- Em métodos *branch-and-price* e *branch-price-and-cut*, a ramificação deve garantir que as soluções obtidas tenham valores inteiros;
- A ramificação nestes métodos é, em geral, mais complicada do que no método *branch-and-bound*;

DDW para problemas com variáveis discretas

▷ Ramificação

- Em métodos *branch-and-price* e *branch-price-and-cut*, a ramificação deve garantir que as soluções obtidas tenham valores inteiros;
- A ramificação nestes métodos é, em geral, mais complicada do que no método *branch-and-bound*;
- Em geral, ramificar diretamente nas variáveis λ tem desvantagens:

DDW para problemas com variáveis discretas

▷ Ramificação

- Em métodos *branch-and-price* e *branch-price-and-cut*, a ramificação deve garantir que as soluções obtidas tenham valores inteiros;
- A ramificação nestes métodos é, em geral, mais complicada do que no método *branch-and-bound*;
- Em geral, ramificar diretamente nas variáveis λ tem desvantagens:
  - Impor $\lambda_j = 0$ diretamente no problema mestre,

DDW para problemas com variáveis discretas

▷ Ramificação

- Em métodos *branch-and-price* e *branch-price-and-cut*, a ramificação deve garantir que as soluções obtidas tenham valores inteiros;
- A ramificação nestes métodos é, em geral, mais complicada do que no método *branch-and-bound*;
- Em geral, ramificar diretamente nas variáveis λ tem desvantagens:
  - Impor $\lambda_j = 0$ diretamente no problema mestre, fará com que a respectiva coluna seja gerada novamente, resultando na mesma solução fracionária (por que?);

DDW para problemas com variáveis discretas

▷ Ramificação

- Em métodos *branch-and-price* e *branch-price-and-cut*, a ramificação deve garantir que as soluções obtidas tenham valores inteiros;
- A ramificação nestes métodos é, em geral, mais complicada do que no método *branch-and-bound*;
- Em geral, ramificar diretamente nas variáveis λ tem desvantagens:
    - Impor $\lambda_j = 0$ diretamente no problema mestre, fará com que a respectiva coluna seja gerada novamente, resultando na mesma solução fracionária (por que?);
      (pra não ser gerada, precisaríamos evitar uma solução específica no subproblema, o que tipicamente compromete sua estrutura).

DDW para problemas com variáveis discretas

▷ Ramificação

- Em métodos *branch-and-price* e *branch-price-and-cut*, a ramificação deve garantir que as soluções obtidas tenham valores inteiros;
- A ramificação nestes métodos é, em geral, mais complicada do que no método *branch-and-bound*;
- Em geral, ramificar diretamente nas variáveis λ tem desvantagens:
  - Impor $\lambda_j = 0$ diretamente no problema mestre, fará com que a respectiva coluna seja gerada novamente, resultando na mesma solução fracionária (por que?);
    (pra não ser gerada, precisaríamos evitar uma solução específica no subproblema, o que tipicamente compromete sua estrutura).
  - Por outro lado, impor $\lambda_j = 1$

DDW para problemas com variáveis discretas

▷ Ramificação

- Em métodos *branch-and-price* e *branch-price-and-cut*, a ramificação deve garantir que as soluções obtidas tenham valores inteiros;
- A ramificação nestes métodos é, em geral, mais complicada do que no método *branch-and-bound*;
- Em geral, ramificar diretamente nas variáveis λ tem desvantagens:
  - Impor $\lambda_j = 0$ diretamente no problema mestre, fará com que a respectiva coluna seja gerada novamente, resultando na mesma solução fracionária (por que?);
    (pra não ser gerada, precisaríamos evitar uma solução específica no subproblema, o que tipicamente compromete sua estrutura).
  - Por outro lado, impor $\lambda_j = 1$ pode ser bastante restritivo, deixando a árvore completamente desbalanceada.

DDW para problemas com variáveis discretas

▷ Ramificação

- Assim, as ramificações são tipicamente feitas em termos das variáveis originais, podendo ser impostas no problema mestre ou nos subproblemas;

DDW para problemas com variáveis discretas

▷ Ramificação

- Assim, as ramificações são tipicamente feitas em termos das variáveis originais, podendo ser impostas no problema mestre ou nos subproblemas;
- Por exemplo, considere uma variável $y_{it} \in \{0, 1\}$ no problema original.

DDW para problemas com variáveis discretas

▷ Ramificação

- Assim, as ramificações são tipicamente feitas em termos das variáveis originais, podendo ser impostas no problema mestre ou nos subproblemas;
- Por exemplo, considere uma variável $y_{it} \in \{0, 1\}$ no problema original.
- Se a solução ótima obtida por geração de colunas, denotada por $\bar{\lambda}$, é fracionária em um dado nó, calculamos a solução em termos da variável original,

DDW para problemas com variáveis discretas

▷ Ramificação

- Assim, as ramificações são tipicamente feitas em termos das variáveis originais, podendo ser impostas no problema mestre ou nos subproblemas;
- Por exemplo, considere uma variável $y_{it} \in \{0, 1\}$ no problema original.
- Se a solução ótima obtida por geração de colunas, denotada por $\bar{\lambda}$, é fracionária em um dado nó, calculamos a solução em termos da variável original, usando:

$$\bar{y}_{it} = \sum_{q \in Q^i} \bar{y}_{qit} \bar{\lambda}_q^i, \quad t = 1, \ldots, T, \; i = 1, \ldots, n.$$

DDW para problemas com variáveis discretas

▷ Ramificação

- Assim, as ramificações são tipicamente feitas em termos das variáveis originais, podendo ser impostas no problema mestre ou nos subproblemas;
- Por exemplo, considere uma variável $y_{it} \in \{0, 1\}$ no problema original.
- Se a solução ótima obtida por geração de colunas, denotada por $\bar{\lambda}$, é fracionária em um dado nó, calculamos a solução em termos da variável original, usando:

$$\bar{y}_{it} = \sum_{q \in Q^i} \bar{y}_{qit} \bar{\lambda}_q^i, \quad t = 1, \ldots, T, \; i = 1, \ldots, n.$$

- Se houver alguma $\bar{y}_{it}$ fracionária, podemos ramificar criando dois nós filhos, impondo $y_{it} = 0$ em um e $y_{it} = 1$ no outro.

DDW para problemas com variáveis discretas

▷ Ramificação

Ramificação no mestre

- Para um dado (i, t), para garantir $y_{it} = 0$, adicionamos a seguinte restrição em um dos filhos:

$$\sum_{q \in Q^i} \bar{y}_{qit} \lambda_q^i = 0,$$

DDW para problemas com variáveis discretas

▷ Ramificação

Ramificação no mestre

- Para um dado (i, t), para garantir $y_{it} = 0$, adicionamos a seguinte restrição em um dos filhos:

$$\sum_{q \in Q^i} \bar{y}_{qit} \lambda_q^i = 0,$$

- No outro filho, para impor $y_{it} = 1$, adicionamos:

$$\sum_{q \in Q^i} \bar{y}_{qit} \lambda_q^i = 1.$$

DDW para problemas com variáveis discretas
▷ Ramificação

Ramificação no mestre

- Para um dado (i, t), para garantir $y_{it} = 0$, adicionamos a seguinte restrição em um dos filhos:

$$\sum_{q \in Q^i} \bar{y}_{qit} \lambda_q^i = 0,$$

- No outro filho, para impor $y_{it} = 1$, adicionamos:

$$\sum_{q \in Q^i} \bar{y}_{qit} \lambda_q^i = 1.$$

- Observe que as duais dessas novas restrição precisam ser incluídas no(s) subproblema(s)!

DDW para problemas com variáveis discretas

▷ Ramificação

Ramificação no subproblema

- Para um dado (i, t), vamos impor no subproblema de um dos nós filhos que $y_{it} = 0$.

DDW para problemas com variáveis discretas

▷ Ramificação

Ramificação no subproblema

- Para um dado (i, t), vamos impor no subproblema de um dos nós filhos que $y_{it} = 0$. Com isso, apenas colunas com essa característica serão geradas.

DDW para problemas com variáveis discretas

▷ Ramificação

Ramificação no subproblema

- Para um dado (i, t), vamos impor no subproblema de um dos nós filhos que $y_{it} = 0$. Com isso, apenas colunas com essa característica serão geradas. **Importante:** no mestre, antes de começar a geração de colunas, precisamos excluir todas as colunas/variáveis que não satisfaçam $y_{it} = 0$;

DDW para problemas com variáveis discretas

▷ Ramificação

Ramificação no subproblema

- Para um dado (i, t), vamos impor no subproblema de um dos nós filhos que $y_{it} = 0$. Com isso, apenas colunas com essa característica serão geradas. **Importante:** no mestre, antes de começar a geração de colunas, precisamos excluir todas as colunas/variáveis que não satisfaçam $y_{it} = 0$;
- No subproblema do outro filho, vamos impor $y_{it} = 1$;

DDW para problemas com variáveis discretas

▷ Ramificação

Ramificação no subproblema

- Para um dado (i,t), vamos impor no subproblema de um dos nós filhos que $y_{it} = 0$. Com isso, apenas colunas com essa característica serão geradas. **Importante:** no mestre, antes de começar a geração de colunas, precisamos excluir todas as colunas/variáveis que não satisfaçam $y_{it} = 0$;
- No subproblema do outro filho, vamos impor $y_{it} = 1$; Também precisamos excluir todas as colunas que não satisfaçam essa restrição, antes de iniciar a geração de colunas;

DDW para problemas com variáveis discretas

▷ Ramificação

Ramificação no subproblema

- Para um dado (i, t), vamos impor no subproblema de um dos nós filhos que $y_{it} = 0$. Com isso, apenas colunas com essa característica serão geradas. **Importante:** no mestre, antes de começar a geração de colunas, precisamos excluir todas as colunas/variáveis que não satisfaçam $y_{it} = 0$;
- No subproblema do outro filho, vamos impor $y_{it} = 1$; Também precisamos excluir todas as colunas que não satisfaçam essa restrição, antes de iniciar a geração de colunas;
- Observação: Devemos evitar danificar a estrutura do subproblema (isto é, evitar que o desempenho do algoritmo usado no nó raiz não se deteriore).

DDW para problemas com variáveis discretas

▷ Ramificação

- Há muitas outras formas de ramificação e, em geral, o desempenho depende do problema que está sendo resolvido;

DDW para problemas com variáveis discretas

▷ Ramificação

- Há muitas outras formas de ramificação e, em geral, o desempenho depende do problema que está sendo resolvido;
- Não há necessidade de usarmos as variáveis originais. Por exemplo, a regra de Ryan-Foster se baseia em subconjuntos disjuntos de variáveis do mestre:

DDW para problemas com variáveis discretas

▷ Ramificação

- Há muitas outras formas de ramificação e, em geral, o desempenho depende do problema que está sendo resolvido;
- Não há necessidade de usarmos as variáveis originais. Por exemplo, a regra de Ryan-Foster se baseia em subconjuntos disjuntos de variáveis do mestre:
    - Dados dois índices r e s de restrições do mestre,

DDW para problemas com variáveis discretas

▷ Ramificação

- Há muitas outras formas de ramificação e, em geral, o desempenho depende do problema que está sendo resolvido;
- Não há necessidade de usarmos as variáveis originais. Por exemplo, a regra de Ryan-Foster se baseia em subconjuntos disjuntos de variáveis do mestre:
  - Dados dois índices r e s de restrições do mestre, essa ramificação consiste em criar dois nós filhos, de modo que em um deles seja permitido apenas colunas com coeficientes não nulos em **ambas** as restrições,

DDW para problemas com variáveis discretas

▷ Ramificação

- Há muitas outras formas de ramificação e, em geral, o desempenho depende do problema que está sendo resolvido;
- Não há necessidade de usarmos as variáveis originais. Por exemplo, a regra de Ryan-Foster se baseia em subconjuntos disjuntos de variáveis do mestre:
  - Dados dois índices r e s de restrições do mestre, essa ramificação consiste em criar dois nós filhos, de modo que em um deles seja permitido apenas colunas com coeficientes não nulos em **ambas** as restrições, e no outro seja permitido apenas colunas em que os coeficientes de pelo menos uma dessas restrições seja nulo.

DDW para problemas com variáveis discretas

▷ Ramificação

- A regra de Ryan-Foster pode ser imposta no subproblema (ideia original), se for possível:

DDW para problemas com variáveis discretas

▷ Ramificação

- A regra de Ryan-Foster pode ser imposta no subproblema (ideia original), se for possível: por exemplo, se o subproblema é um problema da mochila, em um dos nós impomos que os itens r e s estejam sempre juntos; e no outro nó, impomos que nunca podem estar juntos;

DDW para problemas com variáveis discretas

▷ Ramificação

- A regra de Ryan-Foster pode ser imposta no subproblema (ideia original), se for possível: por exemplo, se o subproblema é um problema da mochila, em um dos nós impomos que os itens r e s estejam sempre juntos; e no outro nó, impomos que nunca podem estar juntos;
- Também podemos impor apenas no problema mestre, usando restrições do tipo:

$$\sum_{\substack{q \in Q: \\ \bar{y}_{qir} = \bar{y}_{qis} = 1}} \lambda_q = 0, \qquad \sum_{\substack{q \in Q: \\ \bar{y}_{qir} = \bar{y}_{qis} = 1}} \lambda_q = 1$$

quando a soma do lado esquerdo para uma dada solução seja fracionária ao final da geração de colunas.

DDW para problemas com variáveis discretas

▷ Ramificação

- As ramificações podem ser hierárquicas, começando com uma mais simples e que pode ser heurística (pode não garantir o ótimo) e trocando para uma mais sofisticada e que garanta o ótimo.

- Obrigado pela atenção!
- Dúvidas?