Um algoritmo de *branch and cut* para árvores geradoras mínimas sob restrições de conflito

Phillippe Samer, Sebastián Urrutia
Departamento de Ciência da Computação – Universidade Federal de Minas Gerais
Av. Antônio Carlos, 6627 – Pampulha – Belo Horizonte – MG CEP 31270-901
Email: {samer, surrutia}@dcc.ufmg.br

RESUMO

Este trabalho apresenta abordagens para a solução exata do problema de árvores geradoras mínimas sob restrições de conflito. Dados um grafo $G(V,E)$ e um conjunto $C \subset E \times E$ de pares de arestas, busca-se uma árvore geradora mínima de G livre de conflitos. Isto é, soluções viáveis podem incluir no máximo uma das arestas de cada par em C. O problema é NP–difícil no caso geral, havendo sido recentemente introduzido na literatura. Algumas formulações e algoritmos foram discutidos, mas resultados computacionais com instâncias padrão apresentam *gaps* de dualidade consideravelmente grandes e não fornecem certificados de otimalidade. Neste trabalho introduzimos brevemente uma formulação estendida de fluxo multiproduto, e propomos uma estratégia *branch and cut* usando formulações com uma quantidade exponencial de restrições. Resultados animadores são obtidos, incluindo limites duais consistentemente mais fortes para um conjunto de instâncias padrão, e dois certificados de viabilidade inéditos.

PALAVRAS CHAVE. Árvores ótimas. Restrições de conflito. Formulações em programação inteira. Branch and cut.

ABSTRACT

This paper presents approaches for the exact solution of the minimum spanning tree problem under conflict constraints. Given a graph $G(V,E)$ and a set $C \subset E \times E$ of conflicting edge pairs, the problem consists of finding a conflict-free minimum spanning tree, i.e. feasible solutions may include at most one of the edges from each pair in C. The problem is NP–hard in the general case, and has been introduced recently in the literature. Some formulations and algorithms have been discussed, but computational results for benchmark instances indicate considerably large duality gaps and a lack of optimality certificates. In this work, we briefly introduce an extended multicommodity flow formulation and propose a branch and cut strategy, using formulations with exponentially many constraints. Encouraging results are obtained, including consistently stronger dual bounds for benchmark instances, and two new feasibility certificates.

KEYWORDS. Optimal trees. Conflict constraints. Integer Programming formulations. Branch and cut.

1. Introdução

Relações disjuntivas estão presentes em diversos contextos de otimização combinatória e programação inteira: vários problemas foram estudados sob restrições de conflito ou múltipla-escolha, cortes disjuntivos são implementados em pacotes comerciais para programação inteira mista, além do arcabouço de Programação Disjuntiva introduzido por Egon Balas na década de 1970 (Balas, 2010). Ainda assim, geralizações de alguns problemas clássicos em teoria dos grafos sob restrições disjuntivas foram introduzidas apenas recentemente. Darmann et al. (2011) apresentam variações de problemas em caminhos mínimos, árvores geradoras e emparelhamentos.

Ainda que a maior parte dos resultados seja sobre a complexidade e dificuldade de aproximação de tais problemas em grafos, um interesse particular por árvores geradoras mínimas sob restrições de conflitos (MSTCC, do inglês *minimum spanning tree under conflict constraints*) levou ao desenvolvimento de algoritmos e instâncias padrão de teste.

Dados um grafo $G(V,E)$ e um conjunto $C \subset E \times E$ de pares de arestas, o problema consiste em encontrar uma árvore geradora mínima (MST, do inglês *minimum spanning tree*) livre de conflitos, i.e. uma árvore geradora de G, de mínimo custo e que inclua no máximo uma das arestas e_i ou e_j de cada par $\{e_i, e_j\} \in C$.

De forma equivalente, define-se o problema usando o conceito de um grafo de conflitos $\hat{G}(E,C)$: denotando cada aresta no grafo original como um vértice em $\hat{G}$, representa-se cada restrição de conflito como uma aresta conectando respectivos vértices de $\hat{G}$. Assim, o problema consiste em encontrar um subconjunto de arestas de G, de mínimo custo, que corresponda simultaneamente a uma árvore geradora de G e a um conjunto independente (*stable set*) em $\hat{G}$.

Trabalhos relacionados

Darmann et al. (2009, 2011) introduzem problemas de caminhos, árvores geradoras e emparelhamentos sob restrições de conflito e compulsão (*forcing constraints*, que impõem que soluções incluam pelo menos uma das arestas de cada par em C). Os autores apresentam os problemas e descrevem diversos resultados sobre sua complexidade. Eles demonstram que MSTCC é fortemente NP–difícil, mesmo quando toda componente do grafo de conflitos consiste de um caminho simples de comprimento dois. Ainda mais, mostram que o problema não admite aproximação por um fator constante do ótimo, a menos que $\mathsf{P} = \mathsf{NP}$.

Novos resultados teóricos e computacionais sobre o MSTCC são descritos por Zhang et al. (2011). Os autores discutem casos particulares que podem ser resolvidos em tempo polinomial, testes de viabilidade, heurísticas primais e dois algoritmos baseados em relaxação Lagrangeana. Uma primeira formulação é integral, em que todas restrições disjuntivas são relaxadas e busca-se uma MST clássica no subproblema. Outra formulação relaxa parte destas restrições e resolve-se um subproblema NP–difícil de partição do máximo número de arestas em cliques disjuntas. Resultados computacionais são descritos, mas *gaps* de dualidade consideravelmente grandes são fornecidos pelos algoritmos, que não proveem certificados de otimalidade.

Vale destacar que encontram-se na literatura recente resultados de investigações sobre diferentes problemas combinatórios sob restrições disjuntivas. Pferschy e Schauer (2009) discutem a complexidade de casos particulares do problema da mochila com restrições de conflito. Sadykov e Vanderbeck (2012) apresentam um

algoritmo de *branch and price* para o problema de *bin packing*, resolvendo com programação dinâmica subproblemas da mochila com restrições de conflito. Problemas de fluxo máximo sob restrições de conflito e de compulsão são também estudados do ponto de vista de complexidade por Pferschy e Schauer (2011). Um esquema de relaxação Lagrangeana e heurísticas para o problema de emparelhamentos sob restrições de conflito são apresentados no recente trabalho de Oncan et al. (2013).

Nossa contribuição

Discutimos neste trabalho abordagens para a solução exata do MSTCC. Considera-se na Seção 2 uma formulação polinomial baseada em fluxo multiproduto e resultados preliminares, que indicam o impacto de se utilizar uma descrição tão forte quanto possível do politopo de árvores geradoras na qualidade dos limites duais verificados. Ainda assim, o desempenho computacional deste modelo inviabiliza a solução até mesmo das menores instâncias padrão em tempo razoável.

Alternativamente, obtemos melhores resultados a partir de uma estratégia de decomposição. Apresentam-se na Seção 3 duas formulações em programação inteira com um número exponencial de restrições e algoritmos de *branch and cut*, que permitem obter diversos certificados de otimalidade e soluções viáveis de instâncias cuja viabilidade era desconhecida, vide Seção 4. Finalmente, trabalhos em desenvolvimento e conclusões são indicados na Seção 5.

2. Uma primeira abordagem com formulações polinomiais

Apresentamos brevemente resultados qualitativos sobre formulações polinomiais, uma vez que motivam a abordagem com *branch and cut*. Descrevemos a notação adotada, uma formulação multifluxo e delineamos resumidamente resultados preliminares.

Sejam fornecidos um grafo $G = (V,E)$, com $|V| = n$, $|E| = m$, custos c_e associados a cada aresta e um conjunto $C \subset E \times E$ de pares de arestas em conflito. Defina o vetor de incidência $\mathbf{x} = (x_1, x_2, \ldots, x_{|E|})$ de dada solução, tal que $x_e = 1$ se a aresta e é incluída na solução, $x_e = 0$ caso contrário.

Para consideração de um fluxo f, definimos um nó arbitrário r como raiz; neste trabalho, usamos $r = 1$. Existe um produto diferente associado a cada vértice $k \neq r$. Uma unidade de fluxo de cada produto k emana da raiz e deve alcançar o nó k apropriado. Seja f_{ij}^k a quantidade do produto k conduzida de i a j. Ainda mais, embora o modelo seja definido sobre um grafo não-direcionado, variáveis de fluxo consideram ambos sentidos de uma aresta, i.e. existem variáveis f_{ij}^k e f_{ji}^k para cada $\{i,j\} \in E$ e cada $k \in V \backslash \{r\}$.

Finalmente, defina o *cutset* $\delta(U)$ associado a um subconjunto U de V como é usual: $\delta(U) = \{\{i,j\} \in E : i \in U, j \notin U\}$, tal que $\delta(i) = \delta(\{i\})$ denote o conjunto de arestas incidentes em um vértice i. Em conformidade, indica-se por $\delta^+(i)$ direções de fluxo saindo do vértice i, i.e. variáveis $f_{i,j}^k$ para todo j. Analogamente, $\delta^-(i)$ indica direções de fluxo chegando ao vértice i.

Assim, o problema de MSTCC pode ser formulado como:

$$\min \left\{ \sum_{e \in E} c_e x_e : \mathbf{x} \in \mathscr{P}_c \cap \mathrm{proj}_x(\mathscr{P}_{emflow}) \subseteq \mathbb{B}^{|E|} \right\},$$

em que $\mathscr{P}_c \subset \mathbb{R}_+^{|E|}$ é a região poliédrica correspondente a vetores de incidência livres de conflitos

$$\mathscr{P}_c : \qquad x_{e_1} + x_{e_2} \leq 1, \qquad \forall \{e_1, e_2\} \in C \tag{1}$$

e $\mathscr{P}_{emflow} \subset \mathbb{R}_+^{|E|} \times \mathbb{R}_+^{2|E|\times(|V|-1)}$ é a descrição estendida com fluxos multiproduto do politopo de árvores geradoras, vide Magnanti e Wolsey (1995), dada por $(2)-(8)$ a seguir.

$$\sum_{e\in\delta^-(\{r\})} f_e^k - \sum_{e\in\delta^+(\{r\})} f_e^k = -1, \qquad \forall k \tag{2}$$

$$\sum_{e\in\delta^-(\{k\})} f_e^k - \sum_{e\in\delta^+(\{k\})} f_e^k = 1, \qquad \forall k \tag{3}$$

$$\sum_{e\in\delta^-(\{v\})} f_e^k - \sum_{e\in\delta^+(\{v\})} f_e^k = 0, \qquad \forall k, v \notin \{r,k\} \tag{4}$$

$$\sum_{e\in E} x_e = n-1 \tag{5}$$

$$f_{ij}^k + f_{ji}^{k'} \leq x_e, \qquad \forall k,\ k',\ e=\{i,j\}\in E \tag{6}$$

$$f_{ij}^k \geq 0 \text{ e } f_{ji}^k \geq 0, \qquad \forall k,\ \{i,j\}\in E \tag{7}$$

$$0 \leq x_e \leq 1, \qquad e \in E \tag{8}$$

Equações (2) a (4) indicam balanço de fluxo para cada produto k. Esta condição de conectividade, junto à seleção de $n-1$ arestas na equação (5), garante que a solução corresponde a uma árvore geradora. As desigualdades (7) impõem que valores de fluxo sejam não negativos, enquanto (8) fornece a relaxação linear das variáveis binárias para seleção de arestas. Embora a formulação inclua variáveis f_{ij}^k e f_{ji}^k, Magnanti e Wolsey (1995) mostram que as desigualdades de fluxo bidirecional (6) corretamente vinculam o fluxo de quaisquer produtos k e k' em dada aresta à mesma orientação. Ainda mais, mostram que a formulação alternativa em que (6) são substituídas pelas restrições naturais para um único produto ($f_{ij}^k \leq x_e,\ \forall k,\ e=\{i,j\}\in E$) é mais fraca.

Nosso objetivo em apresentar brevemente esta formulação é motivar a abordagem com *branch and cut*, resumindo resultados computacionais que observamos inicialmente, quando implementamos tanto esta como uma formulação análoga com fluxo de um único produto, descrita por Zhang et al. (2011). Por um lado, nenhuma destas formulações é capaz de resolver instâncias do *benchmark* padrão do problema (instâncias *tipo 1*, apresentadas pelos mesmos autores acima) em menos que dezenas de horas de execução do software CPLEX 12.5, em uma máquina com processador Intel Core i7 980 (3.33GHz) e 16GB de RAM.

Por outro lado, certa intuição é obtida comparando o desempenho destas formulações na solução de um conjunto alternativo de instâncias aleatórias do MSTCC. Construímos e avaliamos um novo conjunto com 56 problemas em que o grafo G é menor (com $n \in \{10,25,50\}$) e o grafo de conflitos $\hat{G}$ é ainda mais esparso, com $|C| \leq \frac{15}{100} \times m$, enquanto o *benchmark* padrão tem $|C| \leq \frac{15}{100} \times \frac{m(m-1)}{2}$ (i.e. até 15% da máxima densidade do grafo de conflitos). Neste caso, a formulação com fluxo de único produto fornece limites duais muito inferiores que a formulação multifluxo, assim como no problema clássico de árvore geradora mínima (Magnanti e Wolsey, 1995). Enquanto o primeiro modelo necessita de um grande número de nós na árvore de enumeração para verificar uma solução ótima, o segundo consegue resolver algumas destas instâncias mais esparsas na própria relaxação linear da raiz.

Destaca-se finalmente que, com respeito às representações poliédricas do politopo de árvores geradoras, Magnanti e Wolsey (1995) mostram que $\mathscr{P}_{emflow}$ tem apenas

vértices inteiros, correspondentes ao conjunto $\mathscr{F}_{tree}$ de árvores geradoras de G, isto é, $\mathscr{P}_{emflow} = conv(\mathscr{F}_{tree})$. Este já não é o caso da formulação com um único produto. Descrevemos a seguir uma estratégia de decomposição com base em formulações equivalentes a $\mathscr{P}_{emflow}$, no sentido de poder resolver instâncias de dimensões razoáveis gerando o modelo dinamicamente, em oposição a carregar toda a formulação *a priori*.

3. Formulações exponenciais e algoritmos de *branch and cut*

Investigamos neste trabalho duas formulações com uma quantidade exponencial de restrições para descrição do politopo de árvores geradoras: uma delas é não-direcionada e utiliza restrições de eliminação de subciclos (SEC – *subtour elimination constraint*); a outra é direcionada, com desigualdades *cutset*.

A seguir, apresentamos ambas formulações, o algoritmo geral com *branch and cut* e a solução dos respectivos problemas de separação.

3.1. Formulações

Seja $\mathscr{P}_{sec} \subset \mathbb{R}_+^{|E|}$ a representação do politopo de árvores geradoras com SECs:

$$\sum_{e \in E(S)} x_e \leq |S| - 1, \ \forall S \subset V, S \neq \emptyset \tag{9}$$

$$\sum_{e \in E} x_e = n - 1 \tag{10}$$

$$0 \leq x_e \leq 1, \ e \in E \tag{11}$$

Enquanto as SECs (9) eliminam ciclos, uma vez que qualquer subgrafo conexo em S induzindo um ciclo tem pelo menos $|S|$ arestas, uma solução viável é garantidamente uma árvore geradora pela escolha de $n-1$ arestas em (10). Como antes, as desigualdades em (11) correspondem à relaxação contínua das variáveis binárias x_e.

Assim, modela-se o MSTCC como:

$$\min \left\{ \sum_{e \in E} c_e x_e : \mathbf{x} \in \mathscr{P}_c \cap \mathscr{P}_{sec} \subseteq \mathbb{B}^{|E|} \right\},$$

em que $\mathscr{P}_c \subset \mathbb{R}_+^{|E|}$ é a região poliédrica anterior (1) de vetores de incidência livres de conflitos.

Alternativamente, consideramos uma formulação com a versão direcionada das clássicas desigualdades de *cutset*. Seja $G' = (V, A)$ a versão direcionada de G, com $A = \{(i,j) \cup (j,i) | \{i,j\} \in E\}$, em que ambos arcos preservam o mesmo custo c_e da aresta original. Neste caso, busca-se uma arborescência geradora em G', a partir da qual é trivial obter uma árvore geradora em G.

Torna-se necessário definir um vértice r como raiz da arborescência. Magnanti e Wolsey (1995) mostram que esta representação poliédrica do politopo de árvores geradoras é simétrica com respeito à escolha da raiz. Neste trabalho, usamos $r = 1$.

Assim, o modelo direcionado é $\mathscr{P}_{dcut}$, dado por:

$$\sum_{a \in \delta^+(U)} y_a \geq 1, \ \forall U \subset V, r \in U \tag{12}$$

$$\sum_{a \in \delta^-(v)} y_a = 1, \ \forall v \in V \backslash \{r\} \tag{13}$$

$$\sum_{a \in A} y_a = n - 1 \tag{14}$$

$$0 \leq y_a \leq 1, \ a \in A \tag{15}$$

Desigualdades *cutset* (12) garantem que o subgrafo selecionado é conexo, impondo que não deve haver uma componente (induzida por um subconjunto próprio de vértices U contendo a raiz) da qual não saia nenhum arco. Já a igualdade (13), que impõe que deve haver exatamente um arco chegando a todo vértice diferente da raiz, é na verdade implicada por (12) e (14). Entretanto, conforme apresentamos em resultados computacionais na seção 4.1, adicionar estas $n-1$ restrições explicitamente no modelo permite resolver sua relaxação linear de forma muito mais eficiente. As demais restrições são análogas ao caso não-direcionado.

Assim, uma definição equivalente de MSTCC é dada por:

$$\min\left\{\sum_{a\in A} c_a y_a : \mathbf{y} \in \mathscr{P}_{c'} \cap \mathscr{P}_{dcut} \subseteq \mathbb{B}^{|A|}\right\},$$

em que $\mathscr{P}_{c'} \subset \mathbb{R}_+^{|A|}$ corresponde à região poliédrica análoga de vetores de incidência livres de conflitos, no espaço das variáveis y_a de seleção de arcos:

$$\mathscr{P}_{c'}: \qquad (y_{a,b}+y_{b,a})+(y_{c,d}+y_{d,c}) \leq 1, \qquad \forall\{\ \{a,b\},\{c,d\}\ \} \in C \qquad (16)$$

3.2. Algoritmo geral

Propomos uma abordagem com *branch and cut* para resolver o problema de MSTCC usando as formulações com representações $\mathscr{P}_{sec}$ e $\mathscr{P}_{dcut}$ para representar o politopo de árvores geradoras. Os algoritmos foram implementados em C++, usando o mecanismo de *callbacks* da interface de programação Concert do software CPLEX 12.5. Desativamos todas as opções de preprocessamento, heurísticas e geração de cortes: separa-se apenas cortes de usuário. Adotamos uma precisão numérica de 10^{-5}, inclusive para verificação de restrições violadas.

A menos de SECs (9) ou desigualdades *cutset* (12), cada algoritmo carrega a formulação completa *a priori*. Isto é, inicia-se como a seguir:

- no caso da formulação com $\mathscr{P}_{sec}$, resolvemos $\min\{\sum_{e\in E} c_e x_e\}$ sujeito a (1), (10) e (11). Seja $\mathbf{x}$ a solução deste programa linear (PL). Claramente, se $\mathbf{x}$ é inteiro e uma busca em profundidade a partir de qualquer vértice i alcança todos os outros vértices em $V\backslash\{i\}$, então $\mathbf{x}$ é também a solução ótima do programa inteiro original.

- no caso da formulação com $\mathscr{P}_{dcut}$, resolve-se $\min\{\sum_{a\in A} c_a y_a\}$ sujeito a (13), (14), (15) e (16). Similarmente, seja $\mathbf{y}$ a solução deste PL. Se $\mathbf{y}$ é inteiro e existe um caminho direcionado da raiz r a todos os outros vértices de G', então a projeção de $\mathbf{y}$ no espaço das variáveis x por $x_{\{i,j\}} = y_{i,j} + y_{j,i}$ é também a solução ótima do programa inteiro.

Se a solução do PL nos casos acima não resolve o programa inteiro original, buscamos desigualdades de eliminação de subciclo (9) ou de *cutset* (12) violadas na solução atual, visando a fortalecer os respectivos poliedros relaxados. Este é o objetivo do procedimento de separação, descrito na próxima seção. Quando o procedimento identifica restrições violadas, adiciona-se respectivos cortes globalmente e resolve-se o novo PL assim fortalecido. Já no caso em que o procedimento de separação falha em obter uma desigualdade violada, prossegue-se na árvore de enumeração com *branching* em variáveis, e o processo reinicia em cada nó.

3.3. Procedimentos de separação

Utilizamos o procedimento exato usual com a solução de problemas de fluxo máximo (corte mínimo), vide Magnanti e Wolsey (1995), através do próprio *solver* de programação linear.

O procedimento para buscar desigualdades violadas em uma solução $\mathbf{y}$ da relaxação de $\mathscr{P}_{dcut}$ inicia definindo capacidades dos arcos de G': ao arco $(i,j) \in A$ atribui-se capacidade igual ao valor atual de $y_{i,j}$. Tem-se que $\mathbf{y}$ satisfaz todas as desigualdades de *cutset* se e somente se é possível enviar uma unidade de fluxo da raiz r a todos outros vértices na rede capacitada. Portanto, resolvendo $n-1$ problemas de fluxo máximo, de r a cada $i \in V \backslash \{r\}$, pode-se verificar em tempo polinomial se a solução $\mathbf{y}$ é viável em $\mathscr{P}_{dcut}$: se o valor de algum corte mínimo é menor que 1, o correspondente conjunto de vértices (i.e. o *cutset*) fornece uma restrição (12) violada por $\mathbf{y}$.

A mesma ideia é adaptada para separar SECs, mas necessitamos primeiramente construir o grafo direcionado correspondente à solução atual x de uma relaxação linear de $\mathscr{P}_{sec}$. A capacidade de ambos arcos (i,j) e (j,i) é definida como $x_{i,j}$, o valor atual da variável correspondente à aresta original.

Destacamos que, em ambos casos, a separação de uma solução inteira pode ser resolvida em menor complexidade computacional. Basta verificar se existe mais de uma componente conexa no subgrafo correspondente à solução atual, e.g. com uma busca em profundidade. Neste caso, a inspeção das componentes conexas fornece um *cutset* ou subciclo correspondente à desigualdade a adicionar. Em nossa implementação, verificamos esta condição em toda execução dos *callbacks*.

4. Resultados computacionais

Descrevemos a seguir os resultados computacionais verificados com algoritmos de *branch and cut* descritos na seção 3. Avaliamos primeiramente o impacto de explicitar restrições (13) na formulação direcionada; então comparamos o desempenho de duas estratégias para a seleção e inclusão de cortes referentes às SECs e desigualdades de *cutset*; por fim, apresentamos resultados gerais comparando ambas formulações entre si e com os resultados descritos por Zhang et al. (2011).

O conjunto de instâncias proposto por tais autores é dividido em dois grupos: *tipo 1* e *tipo 2*. Enquanto o primeiro inclui problemas desafiadores, vários inclusive sem viabilidade confirmada, as instâncias do segundo grupo são de fácil solução na prática, pela sua construção através de uma heurística. Todas as instâncias têm valores inteiros como custos de arestas, então podemos definir o parâmetro de *absolute MIP gap tolerance* do CPLEX como 0.9999. Assim como nos experimentos daqueles autores, adotamos um tempo limite para execução de 5000 segundos. Todos resultados correspondem a avaliações em uma máquina com processador Intel Core i7 980 (3.33GHz), e 16GB de RAM.

4.1. Fortalecendo formulação direcionada

Iniciamos com uma avaliação computacional da inclusão da restrição (13) na formulação direcionada (com $\mathscr{P}_{dcut}$). Conforme explicamos na Seção 3.1, a igualdade é de fato implicada pelas restrições de *cutset* e de seleção de $n-1$ arcos, de forma que o limite dual obtido na relaxação linear é o mesmo.

Consideramos um subconjunto de seis instâncias *tipo 1* viáveis. Valores fora de parênteses na Tabela 1 indicam resultados da relaxação linear (RL) explicitando as restrições (13), enquanto aqueles entre parênteses correspondem aos resultados sem incluí-las. O tempo de execução corresponde ao tempo real (de relógio) decorrido.

Tabela 1: efeito na RL da formulação direcionada de incluir (ou não, entre parênteses) restrições (13), de igualdade sobre grau de entrada de vértices diferentes da raiz.

Instância			Limite de RL	# Cortes na RL	Tempo de RL (s)
$\|V\|$	$\|E\|$	$\|C\|$	de (12)–(16)		
50	200	199	705,5	11 (688)	3,3 (111,6)
50	200	398	761	10 (1831)	3,2 (173,7)
100	300	448	4037,25	2 (8459)	2,3 (3741,6)
100	300	897	4991	1 (433)	1,1 (440,3)
100	500	1247	4275	9 (8728)	18 (7229,7)
100	500	2495	5247,67	2 (187)	4 (396,5)

Claramente, o desempenho computacional melhora muito com a inclusão *a priori* destas $n-1$ restrições no modelo. Ao executar muito menos iterações do procedimento de separação, termina-se o processamento do nó raiz da árvore de enumeração mais rapidamente e com um modelo sensivelmente menor.

4.2. Estratégias de seleção e inclusão de cortes

Pode-se considerar diferentes políticas para fortalecer o poliedro relaxado, uma vez que um corte separando a solução atual é obtido. Consideramos neste trabalho duas estratégias: (i) uma simples, em que termina-se o procedimento de separação assim que um primeiro corte é obtido, e (ii) outra em que executa-se sempre o procedimento por completo e seleciona-se cortes com um critério mais elaborado.

Neste segundo caso, armazenamos os diferentes cortes obtidos e as quantidades $\sum_{e\in E(S)} x_e - (|S|-1)$, dado um conjunto S violando uma SEC, ou $1-\sum_{e\in\delta^+(C)} y_e$, dado um corte $(C, V\backslash C)$ fornecendo uma desigualdade de *cutset* violada. Adicionamos então a desigualdade mais violada e também quaisquer outras próximas o suficiente da ortogonalidade com a mais violada. Neste trabalho, avaliações preliminares indicaram ser eficaz aceitar hiperplanos com produto interno menor ou igual a $0,2$.

Discutimos nas Tabelas 2 e 3 a seguir o impacto de cada estratégia no desempenho geral dos algoritmos de *branch and cut* com a formulação direcionada e a não-direcionada, respectivamente. Utilizamos um subconjunto de instâncias *tipo 1* e *tipo 2* que resolve-se na otimalidade; a seção superior das tabelas corresponde ao primeiro grupo.

Infelizmente, pelo menos com o presente delineamento experimental, a estratégia mais elaborada para seleção e inclusão de cortes não representa melhoria consistente de desempenho. No caso da formulação não-direcionada, tanto o tempo de execução quanto o número de nós na árvore de enumeração são predominantemente maiores.

Tabela 2: desempenho computacional de diferentes estratégias para inclusão de cortes na formulação não-direcionada.

Instância			Primeiro corte			Corte mais violado e ortogonais		
$\|V\|$	$\|E\|$	$\|C\|$	# Cortes	# Nós	Tempo (s)	# Cortes	# Nós	Tempo (s)
50	200	199	134	65	18,6	49	16	3,7
50	200	398	74	68	41,9	72	50	19,6
50	200	597	66	98	56,5	96	111	41,1
50	200	995	212	2448	1239,7	212	2448	1076,9
100	300	448	178	142	202,3	329	219	234,5
100	500	1247	63	20	43,9	117	66	165,2
200	600	34504	25	39	278,2	51	167	1049,8
200	600	42860	18	7	60,2	16	15	96,9
200	600	50984	21	8	44,0	27	9	73,7
200	800	62625	71	63	838,5	89	300	3250,3

Tabela 3: desempenho computacional de diferentes estratégias para inclusão de cortes na formulação direcionada.

Instância			Primeiro corte			Corte mais violado e ortogonais		
$\|V\|$	$\|E\|$	$\|C\|$	# Cortes	# Nós	Tempo (s)	# Cortes	# Nós	Tempo (s)
50	200	199	20	4	1,2	12	4	1,1
50	200	398	19	2	0,6	33	2	8,4
50	200	597	40	21	7,7	70	32	28,1
50	200	995	1851	8067	3623,2	1405	5875	3784,7
100	300	448	2	3	3,4	2	3	3,4
100	500	1247	10	0	2,2	9	0	2,3
200	600	34504	4	23	187,4	4	23	180,6
200	600	42860	8	15	165,5	3	14	125,3
200	600	50984	4	11	100,4	4	11	91,2
200	800	62625	13	105	1197,4	13	105	1241,1

Apenas as primeiras instâncias da Tabela 2 são exceções e, mesmo assim, a diferença é menos representativa. Ainda mais, aparentemente, o ganho de se empregar a estratégia mais simples parece crescer com as dimensões da entrada.

Já no caso da formulação direcionada, o desempenho de ambas estratégias é bastante similar, em geral. Em boa parte das instâncias relacionadas na Tabela 3, a estratégia mais elaborada leva a um menor gasto de tempo ou nós. Curiosamente, o ganho desta segunda estratégia é muito maior em um conjunto de instâncias alternativo, em que os grafos de conflitos são menos densos.

Em demais avaliações, adotamos a estratégia mais simples no caso da formulação não-direcionada, e a estratégia mais elaborada para a formulação direcionada.

4.3. Resultados gerais e certificados de otimalidade

Concluímos apresentando os resultados gerais de algoritmos *branch and cut* com ambas formulações, e comparando com os resultados mais fortes disponíveis na literatura.

Nesta seção, usamos a denominação geral de *limites de Zhang et al. (2011)*. No caso de limites inferiores, os melhores resultados correspondem a um esquema de relaxação Lagrangeana que envolve a solução de um subproblema de partição de arestas em cliques disjuntas. A otimização de limites é feita com um algoritmo de subgradientes. Já quanto aos limites superiores descritos pelos autores, reportamos o melhor resultado obtido para cada instância com alguma das heurísticas que propõem.

No caso do grupo de instâncias mais fáceis (*tipo 2*), heurísticas primais alcançam resultados que aqueles autores reconhecem como ótimos através da solução de um modelo de fluxo com o CPLEX. Já no caso das instâncias *tipo 1*, fornecem apenas gaps de dualidade. A seguir, novos certificados de otimalidade são descritos, e duas instâncias têm limites primais estabelecidos por soluções inteiras viáveis pela primeira vez.

As Tabelas 4 e 5 fornecem resultados com formulações não-direcionada e direci-

Tabela 4: resultados com a formulação não-direcionada e instâncias *tipo 1*.

Instância			Limites de	Custo Ótimo	*Gap* de RL (%)	*Gap* MIP	#Nós	Tempo (s)
$\|V\|$	$\|E\|$	$\|C\|$	Zhang et al. (2011)					
50	200	199	[702,793 , 708]	708	0,57	0	65	18,6
50	200	398	[757,816 , 785]	770	3,38	0	68	41,9
50	200	597	[807,745 , 1044]	917	8,63	0	98	56,5
50	200	995	[877,495 , 1424]	1324	27,79	0	2448	1239,6
100	300	448	[3991,18 , 4102]	4041	1,18	0	142	202,3
100	300	897	[4624,24 , –]	[5339,17 , 5766]	15,54	7,40	2202	4985,2
100	500	1247	[4165,68 , 4293]	4275	0,91	0	20	43,97
100	500	2495	[4805,40 , 6603]	[5550,8 , 6096]	14,21	8,94	1229	4985,8
100	500	3741	[4871,27 , 8787]	[5634 , –]	–	–	1075	4985,24
200	600	1797	[11425,8 , –]	[12598,83 , 14878]	16,39	15,32	394	4993,6
200	800	3196	[17992,6 , –]	[19783,25 , 25482]	22,98	22,36	290	4999,3

Tabela 5: resultados com a formulação direcionada e instâncias *tipo 1*.

Instância			Limites de	Custo Ótimo	*Gap* de RL (%)	*Gap* MIP	#Nós	Tempo (s)
$\|V\|$	$\|E\|$	$\|C\|$	Zhang et al. (2011)					
50	200	199	[702,793 , 708]	708	0,35	0	4	1,1
50	200	398	[757,816 , 785]	770	1,17	0	2	8,4
50	200	597	[807,745 , 1044]	917	5,34	0	32	28,1
50	200	995	[877,495 , 1424]	1324	27,19	0	5875	3784,7
100	300	448	[3991,18 , 4102]	4041	0,09	0	3	3,4
100	300	897	[4624,24 , –]	[5517,75 , 5662]	11,85	2,47	2492	4987,1
100	500	1247	[4165,68 , 4293]	4275	0	0	0	2,3
100	500	2495	[4805,40 , 6603]	[5490,5 , 6238]	15,88	11,97	931	4987,6
100	500	3741	[4871,27 , 8787]	[5616,5 , –]	–	–	1084	4995,7
200	600	1797	[11425,8 , –]	[12549 , –]	–	–	242	4989,7
200	800	3196	[17992,6 , –]	[19921,67 , –]	–	–	155	4999,7

onada, respectivamente, com o subconjunto de instâncias *tipo 1* reconhecidas como viáveis até o momento. Em particular, não eram conhecidas soluções inteiras viáveis para as duas últimas instâncias destas tabelas, mas conseguimos fazê-lo com a formulação não-direcionada. Nestas tabelas, denotamos por RL a relaxação linear, e por *gap MIP* aquele verificado para o problema inteiro, ao fim da execução.

Verificamos que ambas formulações melhoram consistentemente os gaps de dualidade disponíveis, à exceção do limite superior da instância $100-500-3741$, provido por uma heurística de Zhang et al. (2011).

O gap de relaxação linear da formulação direcionada é menor em geral, embora a diferença não seja expressiva – sempre menor que 4%. Ambas formulações fornecem exatamente os mesmos certificados de otimalidade. Entretanto, apenas com o modelo não–direcionado (usando $\mathscr{P}_{sec}$) é possível obter soluções primais viáveis para as instâncias com 200 vértices no tempo limite.

Por fim, a Tabela 6 apresenta resultados com demais instâncias *tipo 1* (na parte superior da tabela) e *tipo 2* (parte inferior). Neste caso, apresentamos os resultados verificados usando a formulação não-direcionada. Todas as instâncias do segundo grupo são resolvidas na otimalidade, em tempo inferior ao limite de 5000 segundos descrito por Zhang et al. (2011), enquanto os limites inferiores obtidos pelo esquema de relaxação Lagrangeana dos autores fornece um gap médio de $25,6\%$ para este conjunto. Conseguimos resolver dez instâncias deste grupo mais fácil na própria relaxação linear. Já para as demais instâncias *tipo 1*, reportamos os melhores limites duais obtidos. Destaca-se que todos são mais fortes que os descritos por aqueles autores, ainda que permitam tempos de execução de até 28421 segundos.

5. Conclusões e trabalho atual

Este trabalho apresenta abordagens para a solução exata do problema de árvore geradora mínima sob restrições de conflito (MSTCC). Descrevemos algoritmos de *branch and cut* com duas formulações envolvendo um número exponencial de restrições: uma não-direcionada, com restrições de eliminação de subciclo (SEC); e outra direcionada, usando desigualdades de *cutset*.

Apresentamos uma avaliação experimental do desempenho computacional desta abordagem, e comparamos as formulações entre si e com os melhores resultados disponíveis na literatura. Na prática, a formulação com SECs obtém um melhor desempenho, permitindo inclusive que duas instâncias tenham soluções inteiras viáveis e gaps de dualidade descritos pela primeira vez. Os resultados computacionais obtidos superam aqueles da literatura, fornecendo limites duais mais fortes e diversos certificados de otimalidade.

Nosso trabalho atual explora principalmente o grafo de conflitos $\hat{G}(E,C)$, para fortalecer relaxações com desigualdades válidas para o politopo $\mathscr{P}_{stab}$ de conjuntos

Tabela 6: resultados com formulação não-direcionada e demais instâncias *tipos 1* e *2*.

Instância			**Zhang et al. (2011)**		**Custo Ótimo**	***Gap* de**	***Gap***	**#Nós**	**Tempo (s)**
$\|V\|$	$\|E\|$	$\|C\|$	Limites	*Gap* (%)		**RL (%)**	**MIP (%)**		
100	300	1344	[4681,27 , –]	–	[5585,5 , –]	–	–	2074	4987,1
100	500	6237	[4968,99 , –]	–	[5794,5 , –]	–	–	1060	4987,4
100	500	12474	[5194,67 , –]	–	[6412 , –]	–	–	1258	4988,7
200	600	3594	[12487 , –]	–	[13460,5 , –]	–	–	343	4990,4
200	600	5391	[12873,2 , –]	–	[14470 , –]	–	–	447	4990,7
200	800	6392	[19705,7 , –]	–	[21083,5 , –]	–	–	239	4988,7
200	800	9588	[20684,8 , –]	–	[21344,5 , –]	–	–	228	4996,7
200	800	15980	[20226,9 , –]	–	[21339 , –]	–	–	237	4999,3
300	800	3196	[30190,1 , –]	–	[33547,5 , –]	–	–	160	4994,5
300	1000	4995	[40732,7 , –]	–	[44282,5 , –]	–	–	96	4998,2
300	1000	9990	[42902,5 , –]	–	[44889,5 , –]	–	–	98	4996,8
300	1000	14985	[44639,1 , –]	–	[44907,5 , –]	–	–	104	4999,6
50	200	3903	[877,467 , 1636]	46,37	1636	37,62	0	135	79,3
50	200	4877	[887,478 , 2043]	56,56	2043	50,05	0	86	53,5
50	200	5864	[1030,25 , 2338]	55,93	2338	56,35	0	40	28,1
100	300	8609	[5754,85 , 7434]	22,59	7434	30,82	0	28	39,1
100	300	10686	[6192,29 , 7968]	22,29	7968	35,46	0	8	15,5
100	300	12761	[6758,57 , 8166]	17,24	8166	37,03	0	9	15,0
100	500	24740	[5104,900 , 12652]	59,65	12652	50,51	0	1042	2703,7
100	500	30886	[5078,820 , 11232]	54,78	11232	51,58	0	1696	4279,9
100	500	36827	[5710,770 , 11481]	50,26	11481	0	0	0	4,3
200	400	13660	[17245,9 , 17728]	2,72	17728	0	0	0	4,2
200	400	17089	[18048,2 , 18617]	3,06	18617	0	0	0	4,0
200	400	20469	[18646,2 , 19140]	2,58	19140	0	0	0	4,0
200	600	34504	[15393,1 , 20716]	25,69	20716	35,69	0	39	278,2
200	600	42860	[13971,5 , 18025]	22,49	18025	26,18	0	7	60,2
200	600	50984	[16708,1 , 20864]	19,92	20864	36,03	0	8	44,0
200	800	62625	[23792,300 , 39895]	40,36	39895	47,37	0	63	838,5
200	800	78387	[22174,200 , 37671]	41,14	37671	44,26	0	59	737,9
200	800	93978	[24907,000 , 38798]	35,8	38798	46,12	0	146	1858,1
300	600	31000	[42720,6 , 43721]	2,29	43721	0	0	0	10,0
300	600	38216	[43486,7 , 44267]	1,76	44267	0	0	0	11,1
300	600	45310	[42149,0 , 43071]	2,14	43071	0	0	0	12,5
300	800	59600	[36629,600 , 43125]	15,06	43125	0	0	0	20,2
300	800	74500	[38069,300 , 42292]	9,98	42292	0	0	0	19,7
300	800	89300	[38843,000 , 44114]	11,95	44114	0	0	0	19,7
300	1000	96590	[56048,300 , 71562]	21,68	71562	37,63	0	128	3244,3
300	1000	120500	[58780,100 , 76345]	23,01	76345	41,54	0	47	1409,5
300	1000	144090	[60810,800 , 78880]	22,91	78880	43,42	0	106	2925,7

independentes (*stable sets*) em $\hat{G}$. Em particular, iniciamos com desigualdades de ciclo ímpar, que podem ser separadas de forma exata em tempo polinomial. Ainda mais, estas podem ser fortalecidas com um procedimento de *lifting* sequencial (Padberg, 1973), fornecendo facetas de $\mathscr{P}_{stab}$.

Agradecimentos
Agradecemos ao incentivo relativo à bolsa de mestrado da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), que viabilizou o presente trabalho.

Referências

Balas, E. (2010), Disjunctive programming, *in* '50 Years of Integer Programming 1958-2008', Springer Berlin Heidelberg, pp. 283–340.

Darmann, A., Pferschy, U. e Schauer, J. (2009), Determining a minimum spanning tree with disjunctive constraints, *in* F. Rossi e A. Tsoukias, eds, 'Algorithmic Decision Theory', Vol. 5783 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 414–423.

Darmann, A., Pferschy, U., Schauer, J. e Woeginger, G. J. (2011), 'Paths, trees and matchings under disjunctive constraints', *Discrete Applied Mathematics* **159**(16), 1726 – 1735.

Magnanti, T. L. e Wolsey, L. A. (1995), Chapter 9 Optimal trees, *in* C. M. M.O. Ball, T.L. Magnanti e G. Nemhauser, eds, 'Network Models', Vol. 7 of *Handbooks in Operations Research and Management Science*, Elsevier, pp. 503 – 615.

Oncan, T., Zhang, R. e Punnen, A. P. (2013), 'The minimum cost perfect matching problem with conflict pair constraints', *Computers & Operations Research* **40**(4), 920 – 930.

Padberg, M. (1973), 'On the facial structure of set packing polyhedra', *Mathematical Programming* **5**(1), 199–215.

Pferschy, U. e Schauer, J. (2009), 'The knapsack problem with conflict graphs', *Journal of Graph Algortihms and Applications* **13**(2), 233–249. DOI: 10.7155/jgaa.00186.

Pferschy, U. e Schauer, J. (2011), 'The maximum flow problem with disjunctive constraints', *Journal of Combinatorial Optimization* pp. 1–11. 10.1007/s10878-011-9438-7.

Sadykov, R. e Vanderbeck, F. (2012), 'Bin packing with conflicts: A generic branch-and-price algorithm', *INFORMS Journal on Computing*. Published online before print May 4, 2012, doi:10.1287/ijoc.1120.0499.

Zhang, R., Kabadi, S. N. e Punnen, A. P. (2011), 'The minimum spanning tree problem with conflict constraints and its variations', *Discrete Optimization* **8**(2), 191 – 205.