O Problema da Máxima Cobertura com Fortificações e Interdições

Marcos Costa Roboredo
Instituto de Matemática e Estatística - Universidade do Estado do Rio de Janeiro
Rua São Francisco Xavier, 524, 20550-900, Maracanã, Rio de Janeiro, RJ, Brasil
marcos.roboredo@ime.uerj.br

Luiz Aizemberg
Departamento de Engenharia de Produção - Universidade Federal Fluminense
Rua Passo da Pátria, 156, 24210-240, São Domingos, Niterói, RJ, Brasil
luizaizemberg@gmail.com

RESUMO

Neste trabalho apresentamos o problema da cobertura com fortificações e interdições (PCFI). O PCFI considera um conjunto de pontos de demanda e facilidades, onde a demanda de cada ponto é dita estar coberta se existe alguma facilidade a no máximo uma distância pré especificada. Caso uma facilidade seja interditada ela para de funcionar, não cobrindo mais nenhum ponto de demanda. Caso uma ou mais facilidades sofram uma interdição, elas deixam de ser consideradas, o que pode reduzir a demanda total coberta. Para se evitar uma interdição, pode-se fortificar a facilidade. Facilidades fortificadas não podem ser interditadas. O PCFI visa maximizar a demanda coberta escolhendo q facilidades para serem fortificadas, sabendo que r facilidades serão interditadas no pior caso. Nós apresentamos uma formulação minimax bilinear para o PCFI e derivamos um algoritmo *branch-and-cut* para este. Diversos resultados são apresentados visando mostrar a eficiência e a robustez do método proposto.

PALAVRAS CHAVE. Otimização em dois níveis, Algoritmo *branch-and-cut*, Problema da máxima cobertura.

Área principal. Otimização.

ABSTRACT

In this paper we present the r-interdiction covering problem with fortification (RICF). The RICF is composed of customers and facilities, where a customer's demand is covered if there is a facility placed at most a pre-specified distance from this customer. When a facility is interdicted it can not cover any customer. If one or more facilities are interdicted, the total covered demand can decrease. A way to avoid a interdiction is fortifying the facility. A fortified facility can not be interdicted. The RICF aims to maximize the total covered demand, fortifying q facilities knowing that r facilities will be interdicted. We a present a minimax bilinear formulation and derive a branch-and-cut algorithm for the problem. We present several results in order to prove the efficiency and robustness of the method proposed.

KEY WORDS. Bilevel Programming, Branch-and-cut, Maximal covering problem.

Main area. Optimization.

1 Introdução

Em sistemas de abastecimento onde pontos de demanda são atendidos por infraestruturas, estas estão sujeitas a interdições totais ou parciais. As razões para tais acontecimentos são diversas, como por exemplo desastres naturais, quebra de equipamentos, acidentes industriais e ataques intencionais. Interdições a infraestruturas causam impacto no funcionamento do sistema. Algumas infraestruturas são classificadas como críticas pois, quando interditadas, acarretam uma alta perda de eficiência no sistema. Exemplos práticos de infraestruturas críticas são pontes, estradas, terminais de produção ou abastecimento, hospitais, dentre outras. Em resposta as interdições, surgem questões a respeito de decisões preventivas que visam reduzir o impacto no sistema como, por exemplo, construção de reforços estruturais e barreiras, inspeção, monitoramento e manutenção preventiva. Quando alguma facilidade recebe alguma decisão preventiva, dizemos que ela recebeu uma fortificação.

Modelos envolvendo sistemas de abastecimento com decisões de fortificação e interdição tem sido bastante discutidos pela comunidade de Pesquisa Operacional nos últimos anos devido a sua relevância tanto prática como teórica.

Este trabalho lida com um sistema de cobertura não capacitado composto de p facilidades e n pontos de demanda. A demanda de cada um destes pontos é dita estar coberta se existe alguma facilidade a no máximo uma distância pré especificada. Caso uma ou mais facilidades sofram uma interdição, elas deixam de ser consideradas no sistema, o que pode reduzir a demanda total coberta. Uma maneira de se evitar a interdição de uma determinada facilidade é a fortificação desta. Quando uma facilidade está fortificada, ela não pode ser interditada. O problema visa maximizar a demanda coberta, escolhendo q facilidades para serem fortificadas, sabendo que r facilidades serão interditadas no pior caso ($q \leq p$ e $r \leq p$). O pior caso acontece quando as r facilidades interditadas são escolhidas de modo a minimizar a demanda coberta. Nós nos referimos a este problema como Problema da Cobertura com Fortificação e Interdições (PCFI).

O PCFI é um Problema de Otimização em dois Níveis Inteiro (2-PONI) onde o problema de otimização de primeiro nível consiste em escolher as q facilidades que serão fortificadas enquanto que o problema de segundo nível consiste na escolha das r facilidades que serão interditadas. Os 2-PONI em geral são difíceis de se resolver de maneira exata. Moore e Bard (1990) propuseram um algoritmo *branch-and-bound* capaz de resolver instâncias relativamente pequenas. Em geral, os procedimentos exatos para 2-PONI são gerados a partir de características específicas dos problemas, como por exemplo métodos de decomposição baseados em desigualdades super válidas (Israeli e Wood, 2002; O'Hanley e Church, 2011), algoritmos de plano de corte (Taşkın *et al.*, 2009), dentre outros. Mais recentemente, Pessoa *et al.* (2013) propuseram um algoritmo *branch-and-cut* para a classe de 2-PONI que podem ser reformulados como problemas de otimização minimax bilinear. Basicamente, os autores perceberam que o algoritmo *branch-and-cut*, que pode ser gerado substituindo o problema de segundo nível por um conjunto de desigualdades lineares, pode ser bastante eficiente. Os autores tomaram como problema de aplicação o Problema da Mediana com Fortificação e Interdições (PMFI), que será melhor detalhado neste trabalho pela semelhança ao PCFI. Esta ideia, ainda que de forma não generalizada, já havia sido aplicada ao problema $(r,p)-$centroide (Roboredo e Pessoa, 2013) e a um problema de localização competitiva considerando o pior caso (Roboredo e Pessoa, 2012). Em todas as três aplicações, o algoritmo foi capaz de resolver diversas instâncias em aberto da literatura e foi comparado aos melhores métodos exatos, obtendo resultados extremamente satisfatórios.

No melhor do nosso conhecimento, o PCFI está sendo estudado pela primeira vez neste trabalho e é baseado em um dos problemas propostos por Church *et al.* (2004). Estes autores propuseram dois problemas: o Problema da Interdição em uma rede de Mediana (PIM) e o Problema da Interdição em uma rede de Cobertura (PIC). Tanto o PIM quanto o PIC são sob a perspectiva do interditor e visam escolher r facilidades para serem interditadas que causam maior impacto no sistema. A diferença entre os dois problemas está no tipo de rede considerada, uma vez que em uma rede de mediana cada um dos pontos de demanda é atendido pela facilidade mais próxima e portanto o custo do sistema é dado pela distância total ponderada, enquanto na rede de cobertura cada um destes pontos é atendido caso esteja coberto e a eficiência do sistema é dada pela demanda total coberta. Baseado no PIM, Church e Scaparra (2007) incorporaram a decisão de fortificação, propondo o PMFI que visa minimizar o custo de atendimento em uma rede de mediana escolhendo q facilidades para serem fortificadas, sabendo que r facilidades serão interditadas no pior caso. Este trabalho pretende algo similar ao propor o PCFI, incorporando decisão de fortificação ao PIC.

Problemas com forticações e interdições em redes de mediana geraram diversas pesquisas na literatura. Para o PMFI, por exemplo, existem quatro métodos exatos (Church e Scaparra, 2007; Scaparra e Church, 2008b,a; Pessoa *et al.*, 2013). Além dos métodos exatos para o PMFI, variantes e extensões deste problema são bastante estudadas (Aksen *et al.*, 2010; Liberatore *et al.*, 2011; Losada *et al.*, 2012; Liberatore *et al.*, 2012; Aksen *et al.*, 2013; Zhu *et al.*, 2013; Aksen *et al.*, 2014).

Por outro lado, problemas deste tipo em redes de cobertura foram pouco estudados. O'Hanley *et al.* (2007b) desenvolveram um modelo para localizar reservas naturais com o objetivo de maximizar o mínimo nível de cobertura considerando a perda de um subconjunto das reservas no pior caso. O'Hanley *et al.* (2007a) lidou com uma variante do problema prévio onde os autores também localizavam reservas visando minimizar a perda máxima de espécies, sabendo que haveria a interdição de algumas reservas no pior caso. Os dois problemas foram resolvidos por um algoritmo de decomposição baseado em desigualdades super válidas que exploravam características específicas dos problemas. No problema proposto por Berman *et al.* (2009), o decisor visa maximizar a cobertura em uma rede escolhendo p facilidades para localizar, sujeito ao pior caso de falha de um dos arcos. Os autores propuseram heurísticas para o problema. O'Hanley e Church (2011) lidaram com um problema em que o planejador do sistema decide onde localizar p facilidades maximizando a cobertura da demanda, sabendo que r destas serão totalmente interditadas no pior caso. Tal problema se difere do PCFI pelo fato de que no PCFI as facilidades já estão localizadas e o planejador do sistema decide pela fortificação de um subconjunto destas. Mais recentemente Keçici *et al.* (2012) propuseram um problema que generaliza o PCFI, permitindo ao planejador do sistema além da decisão de fortificação, a decisão de localização e realocação de facilidades. Devido a complexidade do problema, os autores propuseram uma heurística de busca tabu para o problema. Para comprovar a eficiência da heurística, os autores também propuseram uma busca exaustiva exata que enumera todas as possibilidades de localização, fortificação e realocação das facilidades.

No presente artigo, nós mostramos que o PCFI admite uma formulação minimax bilinear. Em seguida, nós derivamos um algoritmo *branch-and-cut* substituindo o problema de otimização de segundo nível por um conjunto de desigualdades lineares seguindo a ideia apresentada por Pessoa *et al.* (2013). Diversos resultados computacionais são apresentados visando mostrar a eficiência e robustez do método proposto.

Este artigo está organizado da seguinte maneira. A seção 2 descreve formalmente o problema e apresenta uma formulação minimax bilinear para este. Na seção 3 nós propomos uma formulação minimax bilinear para o problema. Na seção 4 nós apresentamos um algoritmo *branch-and-cut* para o PCFI. A seção 5 mostra os resultados obtidos pelo nosso método. Finalmente, a seção 6 apresenta as conclusões obtidas e as perspectivas de pesquisas futuras.

2 O Problema

Consideremos um sistema de cobertura onde pontos de demanda pertencentes a um conjunto J são atendidos por p facilidades pertencentes a um conjunto I. Um ponto de demanda $j \in J$ possui demanda denotada por w_j e é dito estar coberto pela facilidade $i \in I$ se a distância entre i e j denotada por d_{ij} é menor ou igual a uma distância pré-especificada δ_j. Neste contexto,

Tabela 1: Eficiências para cada par de estratégias de fortificação e interdição

	Interdita i_1	Interdita i_2	Interdita i_3
Fortifica i_1	-	2	3
Fortifica i_2	4	-	3
Fortifica i_3	4	2	-

para cada ponto de demanda $j \in J$, definimos o conjunto de todas as facilidades que cobrem este ponto de demanda dado por $N(j) = \{i \in I | d_{ij} \leq \delta_j\}$. Quando $N(j) \neq \varnothing$, dizemos que a demanda w_j do ponto de demanda j está coberta. A eficiência do sistema é dada pela demanda total coberta. Além disso, quando uma das p facilidades $i \in I$ não pode mais cobrir nenhum ponto de demanda esta é dita estar interditada. Quando uma ou mais facilidades são interditadas, a eficiência do sistema pode ser reduzida consideravelmente. Uma maneira de minimizar esta redução é fortificando algumas facilidades. Quando uma das p facilidades está fortificada, esta não pode ser interditada. O PCFI consiste em escolher q facilidades a serem fortificadas maximizando a demanda total coberta, sabendo que r facilidades serão interditadas no pior caso.

Para ilustrar o problema, nós propomos um exemplo ilustrativo. No exemplo temos três facilidades ($p = 3$ e $I = \{i_1, i_2, i_3\}$), quatro pontos de demanda ($J = \{j_1, j_2, j_3 e j_4\}$) com demanda unitária ($w_j = 1, \forall j \in J$). Além disso, tem-se $N(j_1) = \{i_2\}$, $N(j_2) = \{i_1, i_3\}$, $N(j_3) = \{i_2\}$, $N(j_4) = \{i_3\}$. Como tem-se $N(j) \neq \varnothing$, para todo $j \in J$, todas as demandas são inicialmente cobertas e a eficiência do sistema é 4. Vamos considerar para este exemplo $q = r = 1$, ou seja, uma facilidade será fortificada e uma facilidade será interditada.

Suponhamos que a facilidade i_1 seja fortificada e a facilidade i_3 seja interditada. Neste caso, a demanda de j_4 deixaria de ser coberta e a eficiência do sistema se reduziria para 3. A Tabela 1 mostra a eficiência do sistema para cada par de estratégias de fortificação e interdição.

A solução ótima para o exemplo é fortificar a facilidade i_2 pois no pior dos casos a eficiência se reduziria para 3.

3 Formulação em dois níveis para o PCFI

Nesta seção apresentamos uma formulação minimax bilinear para o PCFI. Tal formulação é descrita formalmente da seguinte maneira. Seja, para cada uma das p facilidades $i \in I$, a variável binária de 1º nível x_i indicando se a facilidade i está fortificada. Seja, para cada ponto de demanda $j \in J$, a variável binária de 1º nível y_j indicando se existe alguma facilidade fortificada em $N(j)$. Seja, para cada uma das p facilidades $i \in I$, a variável binária de 2º nível s_i indicando se a facilidade i está interditada. Seja, para cada ponto de demanda $j \in J$, a variável binária de 2º nível t_j indicando se existe alguma facilidade não interditada em $N(j)$. O modelo segue.

$$\max_{x,y} \sum_{j \in J} w_j \left(y_j + t_j - y_j t_j\right) \tag{1}$$

$$\text{s.a.} \sum_{i \in I} x_i = q \tag{2}$$

$$y_j \leq \sum_{i \in N(j)} x_i, \qquad \forall j \in J \tag{3}$$

$$y_j \in \{0, 1\}, \qquad \forall j \in J \tag{4}$$

$$x_i \in \{0, 1\}, \qquad \forall i \in I. \tag{5}$$

$$\min_{s,t} \sum_{j \in J} w_j \left(y_j + t_j - y_j t_j\right) \tag{6}$$

$$\text{s.a.} \sum_{i \in I} s_i = r \tag{7}$$

$$t_j \geq 1 - s_i, \qquad \forall j \in J, \forall i \in N(j) \tag{8}$$

$$t_j \in \{0, 1\}, \qquad \forall j \in J \tag{9}$$

$$s_i \in \{0, 1\}, \qquad \forall i \in I. \tag{10}$$

As funções objetivo de 1º e 2º níveis (1) e (6) são idênticas mas opostas (uma é de maximização e a outra de minimização). Tais funções representam a demanda total coberta após as fortificações e interdições. A restrição de 1º nível (2) indica que q facilidades devem ser fortificadas. As restrições de 1º nível (3) garantem a consistência entre as variáveis x e y. A restrição de 2º nível (7) indica que r facilidades devem ser interditadas. As restrições de 2º nível (8) garantem a consistência entre as variáveis t e s.

4 Algoritmo *branch-and-cut* para o PCFI

Seguindo a ideia apresentada em Pessoa *et al.* (2013) podemos derivar um modelo de Programação Binária para o PCFI substituindo o problema de otimização do 2º nível (6) - (10) por um conjunto de desigualdades

$$\max_{x,y} z \tag{11}$$

$$\text{s.a. } (2) - (5) \tag{12}$$

$$z \leq \sum_{j \in J} w_j \left(y_j + t_j - y_j t_j\right) \quad \forall s, t \text{ que satisfaz } (7) - (10) \tag{13}$$

Notemos que existe uma desigualdade em (13) para cada estratégia viável de interdição. Como o total de interdições é dado por $\binom{|I|-q}{r}$, tais restrições devem ser geradas durante um algoritmo de *branch-and-cut*.

Seja uma solução relaxada $(\overline{z}, \overline{x}, \overline{y}) \in \mathbb{R} \times [0, 1]^{|I|} \times [0, 1]^{|J|}$ que satisfaz (12) e algumas restrições de (13). O problema da separação consiste em encontrar a estratégia de interdição que minimiza a demanda total coberta. Tal estratégia pode ser encontrada resolvendo o modelo de programação inteira (6) - (10) considerando os valores de $\overline{y}$ na função objetivo. É válido ressaltar que as soluções relaxadas podem ser separadas mesmo quando são fracionárias.

Neste contexto, nosso algoritmo exato é um algoritmo de *branch-and-cut* que resolve o modelo dado por (11) - (13) usando resolvedores comerciais. As restrições (13) são adicionadas através de um *callback* de cortes resolvendo o modelo exato (6) - (10).

5 Resultados computacionais

Foram realizados experimentos computacionais utilizando CPLEX 12.5.1 num computador AMD Phenom X4 9600 de 2,31 GHz, com 3Gb de RAM, utilizando um processador apenas. Foi optado por só separar soluções relaxadas inteiras durante o algoritmo de *branch-and-cut* pois foi observado que isto gera melhores resultados. Os experimentos consistem em uma rede de cobertura com 818 vértices que foi obtida de uma base de dados geográficos relativa à cidade de São José dos Campos em São Paulo. Os vértices representam quadras de alguns bairros centrais e o número de imóveis em cada quadra foi utilizado como valor de demanda. Os dados desta base podem ser encontrados em *http://www.lac.inpe.br/ lorena/instancias.html*. Cada um dos vértices da base de dados representa um ponto de demanda. As p facilidades consideradas foram obtidas resolvendo-se o problema da localização com máxima cobertura proposto por (Church e Revelle, 1974) que consiste em encontrar p facilidades que maximizam a demanda total coberta. Isto facilitará eventuais comparações de outros métodos com o proposto neste trabalho. Os valores testados para p foram 100, 200 e 300. Os valores de q e r testados foram 4, 8 e 12. Os raios de cobertura δ_j foram os mesmos para todos os pontos de demanda e os valores testados foram 400 e 600.

Nós criamos uma instância para cada possível combinação de valores dos parâmetros p, q, r e δ_j descritos anteriormente . Em cada instância, pode ser que um ou mais clientes sejam sempre cobertos independente da estratégia de interdição utilizada, e pode ser que um ou mais clientes não sejam cobertos por nenhuma facilidade. Assim, antes da execução de cada instância, nós aplicamos os seguintes refinamentos:

1. Seja um ponto de demanda $j \in J$ tal que $|N(j)| > r$ então j sempre será coberto independente da estratégia de interdição adotada. Assim, j será retirado da instância.

2. Seja um ponto de demanda $j \in J$ tal que $N(j) = \varnothing$ então j será retirado da instância pois nunca será coberto.

É válido ressaltar que o algoritmo poderia ser aplicado às instâncias sem estes refinamentos. Entretanto, o pré-processamento do resolvedor não detecta os dois casos citados acima por não enxergar o modelo completo. Neste caso, pode-se transformar os refinamentos em um pré-processamento, com a única diferença que no primeiro item, ao deletar o cliente j, a respectiva demanda terá que ser adicionada à função objetivo. Nossos testes mostraram que, sem os refinamentos da instância e sem o pré-processamento, o algoritmo resolve instâncias com no máximo 500 vértices. Somente após a utilização destes refinamentos ou pré-processamento que conseguimos utilizar a base de dados com 818 vértices.

Neste contexto, só são considerados, em cada instância, clientes que são inicialmente cobertos mas que podem deixar de ser dependendo da estratégia de interdição utilizada. As Tabelas 2 e 3 mostram estatísticas do *branch-and-cut* para instâncias com raios $\delta_j = 600$ e $\delta_j = 400$, respectivamente. Estas tabelas possuem as seguintes legendas para as colunas. As colunas 1-7 apresentam as características da instância, onde a coluna *Dens.(%)* apresenta a densidade da matriz de cobertura, que é dada por $\frac{\sum_{j \in J} |N(j)|}{|J| \times p}$. A coluna *OBJ* indica o valor da solução ótima. A coluna *UB Raiz* indica o limite superior obtido no nó raiz. A coluna *Gap Raiz* apresenta em porcentagem a diferença relativa entre o limite superior da raiz (coluna *UB Raiz*) e o custo ótimo (coluna *OBJ*). A coluna *Tempo Raiz* indica o tempo de CPU em segundos gasto para se resolver o nó raiz. A coluna *#Nós* indica o número total de nós da árvore de *branch-and-bound*. A coluna *#Cortes* indica o número total de cortes inseridos associados à separação de (13). A coluna *Tempo Sep* indica o tempo total de CPU em segundos gasto resolvendo problemas da separação. A coluna *Tempo Total* indica o tempo total de CPU em segundos gasto durante todo o processo.

Tabela 2: Estatísticas de instâncias com $\delta_j = 600,\ \forall j \in J$

Características								Raiz			Árvore de B&B				
$\lvert J\rvert$	p	q	r	δ_j	$\sum w_j$	dens.(%)	OBJ	UB Raiz	Gap(%) Raiz	Tempo Raiz	#Nós	#Sep	#Cortes	Tempo Sep(s)	Tempo Total(s)
631	100	4	4	600	24134	4,93	19484	21049,09	8,03	2,20	268	25	24	3,05	10,39
646	100	4	8	600	24508	4,93	16585	18758,47	13,11	11,98	529	56	51	10,25	27,2
684	100	4	12	600	25270	4,93	15119	16931,92	11,99	9,03	317	40	38	7,8	19,56
631	100	8	4	600	24134	4,93	21125	22805,71	7,96	2,89	708	67	62	7,47	26,05
646	100	8	8	600	24508	4,93	19545	21496,59	9,99	6,05	3147	128	126	12,92	90,3
684	100	8	12	600	25270	4,93	19326	20843,12	7,85	7,81	3899	133	132	19,06	124,69
631	100	12	4	600	24134	4,93	22511	23516,67	4,47	2,66	680	55	54	5,59	23,47
646	100	12	8	600	24508	4,93	21943	23208,08	5,77	3,59	2467	128	124	12,42	74,42
684	100	12	12	600	25270	4,93	22382	23573,18	5,32	3,17	4776	114	112	16,55	130,78
525	200	4	4	600	20366	5,29	16569	17892,04	7,99	4,51	1503	20	17	4,31	42,27
540	200	4	8	600	21256	5,29	14294	16333,15	14,27	8,75	1772	104	81	25,47	70,47
557	200	4	12	600	21979	5,29	13297	14957,43	12,49	12,67	326	60	56	14,61	26,03
525	200	8	4	600	20366	5,29	18269	19450,24	6,47	5,11	1376	38	34	6,02	39,22
540	200	8	8	600	21256	5,29	17985	19495,23	8,40	3,47	2207	62	59	7,66	59,66
557	200	8	12	600	21979	5,29	18359	19315,71	5,21	5,44	414	45	42	7,75	20,8
525	200	12	4	600	20366	5,29	19723	20073,28	1,78	3,22	252	35	29	4,22	10,09
540	200	12	8	600	21256	5,29	20460	20789,15	1,61	3,02	260	18	14	2,02	8,25
557	200	12	12	600	21979	5,29	21139	21386,62	1,17	4,00	230	22	17	3,36	9,14
376	300	4	4	600	14674	5,08	12768	13412,82	5,05	2,20	651	7	6	1,53	14,47
422	300	4	8	600	15866	5,08	12547	13645,85	8,76	10,97	170	36	29	12,28	17,17
455	300	4	12	600	17145	5,08	13060	13832,01	5,91	2,45	308	14	12	3,02	9,03
376	300	8	4	600	14674	5,08	14160	14407,57	1,75	4,17	179	26	21	6,13	10,2
422	300	8	8	600	15866	5,08	14988	15375,48	2,59	5,58	495	42	33	6,05	16,05
455	300	8	12	600	17145	5,08	16182	16479,33	1,84	6,64	247	28	25	9,17	14,75
376	300	12	4	600	14674	5,08	14649	14659,35	0,07	4,67	5	33	27	4,45	5,14
422	300	12	8	600	15866	5,08	15719	15803,50	0,54	5,20	72	21	18	3,97	6,05
455	300	12	12	600	17145	5,08	16845	17019,40	1,04	6,69	563	26	20	7,72	17,63

Tabela 3: Estatísticas de instâncias com $\delta_j = 400,\ \forall j \in J$

Características								Raiz			Árvore de B&B				
$\lvert J\rvert$	p	q	r	δ_j	$\sum w_j$	dens.(%)	OBJ	UB Raiz	Gap(%) Raiz	Tempo Raiz	#Nós	#Sep	#Cortes	Tempo Sep(s)	Tempo Total(s)
690	100	4	4	400	25402	2,80	21525	22902,76	6,40	3,63	103	23	21	2,98	5,89
741	100	4	8	400	27015	2,80	20199	22056,55	9,20	2,89	345	54	54	8,14	16,63
781	100	4	12	400	27710	2,80	18436	20286,27	10,04	12,19	1090	84	81	16,83	43,61
690	100	8	4	400	25402	2,80	22754	24016,02	5,55	3,17	895	66	63	8,11	31,25
741	100	8	8	400	27015	2,80	22086	23894,03	8,19	3,64	6970	175	173	26,56	202,22
781	100	8	12	400	27710	2,80	20669	22735,70	10,00	32,56	51222	437	436	102,22	1974,92
690	100	12	4	400	25402	2,80	23307	24540,81	5,29	2,41	2660	131	129	14,91	79,89
741	100	12	8	400	27015	2,80	23103	24903,56	7,79	5,08	29568	529	526	96,13	1102,02
781	100	12	12	400	27710	2,80	22216	24406,25	9,86	42,56	43556	1521	1512	250,88	21192,06
571	200	4	4	400	23030	2,81	20534	21303,27	3,75	2,44	325	10	9	1,81	10,22
620	200	4	8	400	24584	2,81	19960	21225,92	6,34	4,03	967	20	19	4,36	26,36
663	200	4	12	400	25460	2,81	18924	20188,90	6,68	7,74	678	36	33	10,86	27,19
571	200	8	4	400	23030	2,81	21234	22155,16	4,34	7,08	2764	40	38	11,09	77,28
620	200	8	8	400	24584	2,81	21042	22485,14	6,86	4,30	73888	194	189	38,81	1891,41
663	200	8	12	400	25460	2,81	20436	22165,34	8,46	6,95	122634	388	383	92,06	3845,14
571	200	12	4	400	23030	2,81	21526	22445,88	4,27	4,83	140911	118	113	15,97	2891,33
620	200	12	8	400	24584	2,81	21898	23270,54	6,27	3,58	499400	514	510	84,91	11510,47
663	200	12	12	400	25460	2,81	21656	23315,31	7,66	4,30	376020	1212	1203	212,67	21325,22
417	300	4	4	400	14766	2,80	12503	13482,71	7,84	2,58	157	65	63	6,75	10,38
493	300	4	8	400	19508	2,80	15345	16696,46	8,81	3,58	5321	135	134	26,02	129,33
582	300	4	12	400	21671	2,80	15848	17279,53	9,03	6,39	9671	158	156	48,16	239,08
417	300	8	4	400	14766	2,80	13074	13920,83	6,48	2,75	1403	96	96	10,7	36,47
493	300	8	8	400	19508	2,80	16544	17820,48	7,72	5,92	15686	324	318	87,13	417,31
582	300	8	12	400	21671	2,80	17301	18972,61	9,66	41,25	300450	1239	1230	366,88	21470,16
417	300	12	4	400	14766	2,80	13829	14327,97	3,61	3,05	1529	45	43	5,89	33,64
493	300	12	8	400	19508	2,80	17585	18637,90	5,99	4,31	27334	308	304	74,25	622,05
582	300	12	12	400	21671	2,80	19094	20261,82	6,12	6,59	89128	514	508	137,58	2239,08

Analisando as Tabelas 2 e 3 percebemos que o método se mostrou robusto, uma vez que encontrou a solução ótima para instâncias significativamente grandes em razoáveis tempos computacionais. O primeiro resultado que nos chamou atenção é que as instâncias com raios de cobertura $\delta_j = 600$ foram resolvidas significativamente mais rápido do que as com raios de cobertura $\delta_j = 400$. Isto pode ser explicado pelo fato de que quanto menor é o raio, menor é o número de facilidades pertencentes ao conjunto $N(j)$. Consequentemente, em cada separação o número de restrições de 2º nível (8) diminui. Isto faz com que as variáveis t_j possam assumir valor zero na separação com maior facilidade, minimizando a demanda total coberta (objetivo da separação). Neste caso, a tendência é ter que resolver mais problemas da separação durante toda a árvore, aumentando o tempo computacional.

Outro fator que notamos é que as instâncias se mostraram mais difíceis quando o número de interdições r é alto. Isto acontece porque, a medida que o valor de r aumenta, o número de estratégias de interdição aumenta exponencialmente. Assim, o número de nós da árvore de *branch-and-bound* tende a aumentar.

Notamos que, apesar de resolvermos o problema da separação tanto para soluções fracionárias quanto inteiras, o tempo total consumido pela separação foi muito baixo em comparação com o tempo total de execução das instâncias. Isto sugere que poderia haver melhorias se adotássemos uma estratégia em que mais do que um corte da separações seja adicionado a um mesmo nó. Por exemplo, podemos pegar todas as soluções ótimas geradas pela separação e criar um corte diferente para cada uma.

6 Conclusões e trabalhos futuros

Neste trabalho apresentamos o problema da cobertura com fortificações e interdições. Um algoritmo *branch-and-cut* exato foi proposto para o problema obtendo-se resultados extremamente satisfatórios para instâncias razoavelmente grandes.

Como trabalhos futuros, pretendemos testar diferentes estratégias para a execução da árvore de *B&B* uma vez que percebemos que o tempo gasto com a separação é muito pequeno em comparação com o tempo total de execução das instâncias. Além disso, pretendemos generalizar ainda mais o problema, permitindo decisões de localização e realocação de facilidades ao invés de somente fortificações.

Referências

Aksen, D., Aras, N. e Piyade, N. (2013), A bilevel p-median model for the planning and protection of critical facilities. *Journal of Heuristics*, v. 19, n. 2, p. 373–398.

Aksen, D., Piyade, N. e Aras, N. (2010), The budget constrained r-interdiction median problem with capacity expansion. *Central European Journal of Operations Research*, v. 18, n. 3, p. 269–291.

Aksen, D., Şengül Akca, S. e Aras, N. (2014), A bilevel partial interdiction problem with capacitated facilities and demand outsourcing. *Computers & Operations Research*, v. 41, p. 346–358.

Berman, O., Drezner, T., Drezner, Z. e Wesolowsky, G. (2009), A defensive maximal covering problem on a network. *International Transactions in Operational Research*, v. 16, n. 1, p. 69–86.

Church, R. e Revelle, C. R. (1974), The maximal covering location problem. *Papers in regional science*, v. 32, n. 1, p. 101–118.

Church, R. L., Scaparra, M. P. e Middleton, R. S. (2004), Identifying critical infrastructure: the median and covering facility interdiction problems. *Annals of the Association of American Geographers*, v. 94, n. 3, p. 491–502.

Church, R. L. e Scaparra, M. P. (2007), Protecting critical assets: The r-interdiction median problem with fortification. *Geographical Analysis*, v. 39, n. 2, p. 129–146.

Israeli, E. e Wood, R. K. (2002), Shortest-path network interdiction. *Networks*, v. 40, n. 2, p. 97–111.

Keçici, S., Aras, N. e Verter, V. (2012), Incorporating the threat of terrorist attacks in the design of public service facility networks. *Optimization Letters*, v. 6, n. 6, p. 1101–1121.

Liberatore, F., Scaparra, M. P. e Daskin, M. S. (2011), Analysis of facility protection strategies against an uncertain number of attacks: the stochastic r-interdiction median problem with fortification. *Computers & Operations Research*, v. 38, n. 1, p. 357–366.

Liberatore, F., Scaparra, M. P. e Daskin, M. S. (2012), Hedging against disruptions with ripple effects in location analysis. *Omega*, v. 40, n. 1, p. 21–30.

Losada, C., Scaparra, M. P. e O'Hanley, J. R. (2012), Optimizing system resilience: a facility protection model with recovery time. *European Journal of Operational Research*, v. 217, n. 3, p. 519–530.

Moore, J. T. e Bard, J. F. (1990), The mixed integer linear bilevel programming problem. *Operations research*, v. 38, n. 5, p. 911–921.

O'Hanley, J. R. e Church, R. L. (2011), Designing robust coverage networks to hedge against worst-case facility losses. *European Journal of Operational Research*, v. 209, n. 1, p. 23–36.

O'Hanley, J. R., Church, R. L. e Gilless, J. K. (2007a), The importance of *in situ* site loss in nature reserve selection: Balancing notions of complementarity and robustness. *Biological conservation*, v. 135, n. 2, p. 170–180.

O'Hanley, J. R., Church, R. L. e Keith Gilless, J. K. (2007b), Locating and protecting critical reserve sites to minimize expected and worst-case losses. *Biological Conservation*, v. 134, n. 1, p. 130–141.

Pessoa, A., Poss, M., Roboredo, M. e Aizemberg, L. Solving bilevel combinatorial optimization as bilinear min-max optimization via a branch-and-cut algorithm. *Anais do XLV Simpósio Brasileiro de Pesquisa Operacional*, 2013.

Roboredo, M. C. e Pessoa, A. A. A branch-and-cut algorithm for a budget constrained centroid problem. *Anais do XLIV Simpósio Brasileiro de Pesquisa Operacional*. Sobrapo, 2012.

Roboredo, M. C. e Pessoa, A. A. (2013), A branch-and-cut algorithm for the discrete (r|p)-centroid problem. *European Journal of Operational Research*, v. 224, n. 1, p. 101 – 109.

Scaparra, M. P. e Church, R. L. (2008a), A bilevel mixed-integer program for critical infrastructure protection planning. *Computers & Operations Research*, v. 35, n. 6, p. 1905–1923.

Scaparra, M. P. e Church, R. L. (2008b), An exact solution approach for the interdiction median problem with fortification. *European Journal of Operational Research*, v. 189, n. 1, p. 76–92.

Taşkın, Z. C., Smith, J. C., Ahmed, S. e Schaefer, A. J. (2009), Cutting plane algorithms for solving a stochastic edge-partition problem. *Discrete Optimization*, v. 6, n. 4, p. 420–435.

Zhu, Y., Zheng, Z., Zhang, X. e Cai, K. (2013), The *r*-interdiction median problem with probabilistic protection and its solution algorithm. *Computers & Operations Research*, v. 40, n. 1, p. 451–462.