Reformulação para o Problema da Mediana 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, Artur Alves Pessoa
Departamento de Engenharia de Produção - Universidade Federal Fluminense
Rua Passo da Pátria, 156, 24210-240, São Domingos, Niterói, RJ, Brasil
luizaizemberg@gmail.com, artur@producao.uff.br

RESUMO

Nós lidamos com o problema da mediana com fortificações e interdições, que considera um sistema composto por n pontos de demanda e p facilidades. Cada ponto é atendido pela facilidade mais próxima. Uma facilidade é dita estar interditada quando não pode mais atender nenhum ponto. Uma maneira de se evitar a interdição de uma facilidade é fortificando-a. Uma facilidade fortificada não pode ser interditada. O problema visa decidir quais q facilidades deverão ser fortificadas sabendo que r facilidades não fortificadas serão interditadas. Após isso, os pontos que estavam sendo atendidos por facilidades agora interditadas, passam a ser atendidos pela facilidade não interditada mais próxima, aumentando a distância percorrida por este ponto. A decisão de fortificação visa, considerando o pior caso, minimizar a soma ponderada deste aumento para todos os pontos. Nós reformulamos a melhor formulação para instâncias grandes do problema reduzindo significativamente o número de variáveis e restrições. Os resultados mostram que a nova formulação é adequada também para instâncias pequenas, bem como resolveu instâncias grandes que estavam em aberto.

PALAVRAS CHAVE. Otimização em dois níveis, Algoritmo *branch-and-cut*, Problema da mediana com fortificações e interdições.

Área principal. Otimização.

ABSTRACT

In this paper we deal with r-interdiction median problem with fortification. This problem considers a system composed of n customers and p facilities, where a customer's demand is served by the closest facility. When a facility is interdicted it can not serve any customer. A way to avoid a interdiction is fortifying the facility. A fortified facility can not be interdicted. The problem consists of fortifying q facilities knowing that r facilities will be interdicted. After these decisions, the customers that were being served by interdicted facilities are served now by the closest not interdicted facility, increasing the distance traveled by this customer. In this context, the fortification decision is made aiming to minimize the total weighted increase. We reformulate the main formulation for large instances of the problem reducing significantly the number of variables and constraints. The results show that the new formulation is suitable for both small and large instances. Besides we solve several open instances.

KEY WORDS. Bilevel Programming, Branch-and-cut, The r-interdiction median problem with fortification.

Main area. Optimization.

1 Introdução

Modelos de interdição são usados em sistemas onde são consideradas possíveis interdições totais ou parciais de infraestruturas do sistema. As razões para tais acontecimentos são diversas, como por exemplo desastres naturais, quebra de equipamentos, acidentes industriais e ataques intencionais. Dentre as estruturas do sistema, existem as chamadas infraestruturas críticas, que são aquelas que, 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 de otimização envolvendo decisões de fortificações e interdição de infraestruturas possuem grande relevância tanto prática quando teórica e, portanto, tem sido cada vez mais discutidos pela comunidade acadêmica. Algumas pesquisas que envolvem o tema foram propostas por Wood (1993), Washburn e Wood (1995), Israeli e Wood (2002), Liberatore *et al*. (2012) e Church e Scaparra (2007).

Neste trabalho, será estudado o Problema da Mediana com fortificações e Interdições (PMFI) introduzido por Church e Scaparra (2007). Neste problema, é considerado um sistema composto por n pontos de demanda e p facilidades onde cada ponto de demanda é atendido pela facilidade mais próxima. O problema consiste em decidir quais q facilidades deverão ser fortificadas sabendo que r facilidades não fortificadas serão interditadas. Após as fortificações e interdições, os pontos de demanda que estavam sendo atendidos por facilidades que foram interditadas passam a ser atendidos pela facilidade não interditada mais próxima, aumentando a distância percorrida por este ponto de demanda. Nestre contexto, a decisão de fortificação visa, considerando o pior caso, minimizar a soma deste aumento para todos os pontos de demanda ponderada pelo valor da demanda de cada ponto.

O PMFI é um Problema de Otimização em dois Níveis Inteiro (2-PONI) onde a decisão de primeiro nível consiste em decidir quais facilidades serão fortificadas enquanto a decisão do segundo consiste em escolher as 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) observou que quando um 2-PONI admite uma formulação minmax bilinear, uma reformulação de Programação Inteira Mista pode ser obtida com número polinomial de variáveis, substituindo o problema de otimização do 2º nível por um conjunto de desigualdades lineares. Isto possibilita o uso eficiente de resolvedores comerciais para resolução exata. Como exemplo de aplicação, os autores utilizaram o próprio PMFI e, portanto, este trabalho será melhor detalhado mais adiante. Esta técnica já havia sido aplicada para resolver, embora de maneira implicita, a dois problemas de localização competitiva (Roboredo e Pessoa, 2012, 2013). Para todas as três aplicações prévias, o método foi comparado aos melhores métodos exatos da literatura, obtendo resultados computacionais satisfatórios. Além disso, várias soluções ótimas de instâncias grandes foram apresentadas pela primeira vez.

A respeito especificamente do PMFI, as pesquisas são recentes. Church e Scaparra (2007) introduziram o problema e propuseram um modelo de Programação Linear Inteira (PLI) exato. Nós encontramos ainda mais três métodos exatos (Scaparra e Church, 2008b,a; Pessoa *et al*., 2013) mas nenhum método aproximativo. Os resultados apresentados por estes quatro trabalhos mostram, como era de se esperar, que a medida que os valores de n, p, q e r aumentam, a dificuldade das

instâncias também aumenta, onde o maior impacto é causado pelo número de facilidades total p e pelo número de facilidade a serem interditadas r. Isto acontece pois, para se avaliar o custo de uma simples estratégia de fortificação, é necessário se resolver um problema NP-difícil que visa otimizar a estratégia de interdição e com dificuldade significativamente sensível a estes dois parâmetros. O modelo PLI apresentado por Church e Scaparra (2007) possui um número exponencial tanto de variáveis quanto de restrições, onde os autores conseguiram resolver instâncias com no máximo $n = 150$, $p = 20$, $q = 10$ e $r = 4$. Scaparra e Church (2008b) propuseram um modelo alternativo de PLI baseado em uma formulação para problema de máxima cobertura. Os autores mostraram ainda que o tamanho do modelo original pode ser reduzido. Os autores apresentaram resultados computacionais para instâncias com até $n = 150$, $p = 30$, $q = 7$ e $r = 7$. Nos dois modelos prévios, existe a necessidade da enumeração completa de todas as possíveis maneiras de se interditar r das p infraestruturas, deixando o tempo de resolução bastante sensível a aumentos nestes dois parâmetros.

Já nos métodos propostos por Scaparra e Church (2008a) e Pessoa *et al*. (2013), este problema não acontece. No primeiro trabalho, os autores formularam o problema como um 2-PONI resolvido através de uma árvore de enumeração em que no máximo $\frac{r^{q+1}-1}{(r-1)}$ problemas do segundo nível são resolvidos. Tal método, chamado pelos autores de Enumeração Implícita (EI), encontrou a solução ótima para instâncias com até $n = 150$, $p = 60$, $q = 12$ e $r = 5$. Já no trabalho proposto por Pessoa *et al*. (2013), uma formulação minimax bilinear foi proposta, possibilitando a criação de um algoritmo *branch-and-cut* para o problema. Tal algoritmo foi comparado a EI proposta por Scaparra e Church (2008a), se mostrando mais adequado para instâncias com $r > 5$. Pessoa *et al*. (2013) apresentaram ainda a solução ótima para instâncias com até $n = 150$, $p = 50$, $q = 8$ e $r = 10$.

Variantes do problema também são estudadas na literatura. Liberatore *et al*. (2011) trataram uma variante do problema com incerteza uma vez que considera uma distribuição de probabilidade sobre o número de instalações a serem interditadas. Já na variante proposta por Aksen *et al*. (2010), cada infraestrutura possui um custo para ser fortificada e assim o número de infraestruturas fortificadas é limitado por uma restrição orçamentária. Além disso, os autores também consideraram que para cada uma infraestrutura não interditada existe um custo unitário por cada ponto de demanda que esta venha a atender que era antes atendido por uma infraestrutura que foi interditada.

A principal contribuição deste trabalho é uma reformulação para a formulação proposta por Pessoa *et al*. (2013). Primeiramente, vamos propor uma nova formulação minmax bilinear equivalente a proposta pelo trabalho prévio com uma redução de mais do que $\frac{p-r}{p}$ e np variáveis e restrições, no primeiro e no segundo nível respectivamente. Uma outra consequência importante da reformulação é que o parâmetro p passa a não influenciar mais no número de variáveis da formulação. Para avaliar o impacto da reformulação, nós a resolvemos via um algoritmo de *branch-and-cut* similar ao proposto por Pessoa *et al*. (2013). Além disso, o método foi comparado ao proposto por Scaparra e Church (2008a), onde foi mais rápido em praticamente todas as instâncias, independentemente do tamanho destas. A reformulação em conjunto com o algortimo de *branch-and-cut* permitiu comprovar o ótimo de diversas instâncias em aberto.

Este trabalho está dividido da seguinte maneira. Seção 2 descreve formalmente o PMFI. Seção 3 apresenta a reformulação proposta neste artigo. Seção 4 mostra o algoritmo *branch-and-cut* utilizado para resolver a formulação. Seção 5 relata os experimentos computacionais desta pesquisa. Seção 6 sumariza nossas conclusões.

2 Descrição do PMFI

O ambiente do PMFI é um sistema formado por um conjunto J de n pontos de demanda e um conjunto I de p facilidades. A demanda de cada ponto $j \in J$ é denotada por w_j. A distância de cada ponto de demanda $j \in J$ a cada facilidade $i \in I$ é denotada por d_{ij}. O custo para que o cliente j seja servido pela facilidade i, denotado por c_{ij}, é dado por $c_{ij} = w_j d_{ij}$. Todo cliente j deve

ser servido por exatamente uma facilidade. Existe a possibilidade de uma facilidade ser interditada. Neste caso, esta não poderá atender nenhum cliente e os clientes que eram atendidos por esta passam a ser atendidos pela facilidade não interditada mais próxima, aumentando o custo do sistema. Uma maneira de se evitar a perda de uma facilidade por interdição é fortificando esta facilidade. Quando uma facilidade é fortificada, esta pode continuar atendendo os pontos de demanda mesmo que seja interditada.

Neste contexto, o PMFI consiste em decidir quais q facilidades a serem fortificadas sabendo que r facilidades serão interditadas. Tal decisão é tomada considerando o pior caso, que acontece quando o conjunto de r facilidades a serem interditadas é o que causa maior prejuízo ao sistema. Este problema pode ser interpretado como um problema de otimização em dois níveis onde a decisão do líder, consiste em escolher quais facilidades serão fortificadas enquanto o problema do seguidor consiste em decidir quais facilidades serão interditadas.

3 Reformulação minimax bilinear para o PMFI

Nesta seção, é mostrado que o PMFI admite uma formulação minimax bilinear. Este novo modelo reformula o modelo de segundo nível encontrado em Pessoa *et al*. (2013), reduzindo consideravelmente a quantidade de variáveis e restrições.

O modelo usa os conjuntos de variáveis x, y, s, t. Para cada $i \in I$, a variável binária x_i é igual a 1 se e somente se a facilidade i é fortificada e a variável binária y_i é igual a 1 se e somente se a facilidade i está interditada. Para cada $i \in I$ e $j \in J$, a variável binária s_{ij} é igual a 1 se e somente se i é a facilidade fortificada mais barata para o cliente j e a variável binária t_{ij} é igual a 1 se e somente se a facilidade i é interditada e qualquer facilidade mais barata para o cliente j do que i é também interditada. O custo de servir um dado cliente $j \in J$ é igual à distância deste cliente até a facilidade não interditada mais perto ponderada pela demanda do cliente, e dado por c_{ij}.

Nos artigos anteriores que estudaram o PMFI, todas as variáveis com indice i de facilidade foram criadas até p. No entanto, foi observado que, para as variáveis s_{ij} e t_{ij}, a quantidade de variáveis criadas pode ser drásticamente reduzida.

Considere novamente a definição da variável binária t_{ij} que resulta em 1 caso a facilidade i e todas as facilidades mais próximas de j do que i sejam interditadas. Como o seguidor pode interditar no máximo r facilidades, somente as variáveis t_{ij} cuja facilidade i esteja entre as r mais perto de j inclusive podem ter valor 1. Assim, podemos descartar todas as t_{ij} com i mais distante que a r-ésima facilidade mais perto de j. O raciocínio é análogo para s_{ij}, com apenas uma diferença. Como o modelo obriga o líder a selecionar uma facilidade para ser a fortificada mais perto de cada cliente, e pode ser que o líder não fortifique nenhuma facilidade das r mais perto de um dado cliente, será necessário criar s_{ij} até a $(r+1)$-ésima mais próxima. Esta última facilidade poderá sempre ser selecionada como a mais perto, e caso seja selecionada, não irá interferir no custo da solução.

A observação acima leva a uma redução de mais do que $\frac{p-r}{p}$ e np variáveis e restrições, no primeiro e no segundo nível respectivamente. Para cada cliente $j \in J$ é definida a constante $\varphi(k, j)$ denotando a $k-$ésima facilidade mais barata para o cliente j. Empates são quebrados arbitrariamente. A formulação minimax bilinear para o PMFI é a que segue.

$$\min_{x,s} \quad C(s,t) = \sum_{j \in J} c_{\varphi(1,j),j} + \sum_{j \in J} \sum_{i=1}^{r} t_{ij} \left(c_{\varphi(i+1,j),j} - c_{\varphi(i,j)j} \right) \sum_{\substack{k \in I: \\ c_{kj} \geq c_{\varphi(i+1,j),j}}} s_{kj} \qquad (1)$$

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

$$s_{\varphi(i,j),j} \leq x_{\varphi(i,j)}, \quad \forall j \in J, \forall i = 1, 2, ..., r \tag{3}$$

$$s_{\varphi(r+1,j),j} \leq 1, \quad \forall j \in J \tag{4}$$

$$\sum_{i=1}^{r+1} s_{\varphi(i,j),j} = 1, \quad \forall j \in J \tag{5}$$

$$x_i \in \{0, 1\}, \quad \forall j \in J, \forall i \in I \tag{6}$$

$$s_{\varphi(i,j),j} \in \{0, 1\}, \quad \forall j \in J, \forall i = 1, 2, ..., r+1 \tag{7}$$

$$\max_{y,t} \quad C(s, t) \tag{8}$$

$$\text{s.a.} \quad \sum_{i \in I} y_i = r \tag{9}$$

$$t_{\varphi(i,j),j} \leq y_{\varphi(i,j)}, \quad \forall j \in J, \forall i = 1, 2, ..., r \tag{10}$$

$$t_{\varphi(1,j)j} = y_{\varphi(1,j)}, \quad \forall j \in J \tag{11}$$

$$t_{\varphi(i,j)j} \geq t_{\varphi(i+1,j)j}, \quad \forall j \in J, \forall i = 1, 2, ..., r-1 \tag{12}$$

$$y_i \in \{0, 1\}, \quad \forall j \in J, \forall i \in I \tag{13}$$

$$t_{\varphi(i,j),j} \in \{0, 1\}, \quad \forall j \in J, \forall i = 1, 2, ..., r \tag{14}$$

A função objetivo do líder (1) minimiza a distância total ponderada pela demanda. Note que o primeiro termo da função objetivo é uma constante cujo valor é o custo mínimo do problema, com todos os clientes sendo atendidos pela facilidade mais próxima. O segundo termo da função objetivo junta a estratégia da líder (s_{ij}) com a estratégia do seguidor (t_{ij}). O custo extra devido às interdições aparece neste segundo termo.

A restrição (2) garante que q facilidades devem ser fortificadas. As restrições (3) garantem que a variável s_{ij} só pode ser 1 caso a facilidade i seja fortificada. As restrições (4) permitem que a $(r+1)$-ésima facilidade mais perto do cliente j seja considerada a fortificada mais perto, caso nenhuma das r mais perto sejam fortificadas. Isto não altera custo da solução. As restrições (5) garantem que, para cada cliente j, existe exatamente uma facilidade fortificada mais barata. A restrição de segundo nível (9) garante que r facilidades devem ser interditadas. As restrições (10) garantem que a variável t_{ij} só pode ser 1 se a facilidade i for interditada. As restrições (11) garantem que se a facilidade mais barata para o cliente j for interditada, então a variável correspondente t associada a esta facilidade é igual a 1. Restrições (12) garantem que, para cada cliente j e facilidade i, se $t_{ij} = 1$ então as variáveis t associadas ao cliente j e qualquer facilidade mais barata do que i são também iguais a 1.

4 Algoritmo exato para o PMFI

Desenvolvemos um algoritmo de *branch-and-cut*, substituindo o problema de otimização do 2º nível (8) - (14) por um conjunto de desigualdades, seguindo a ideia proposta por Pessoa *et al*. (2013):

$$\min_{x,s} \ z \tag{15}$$

$$\text{s.a. } (2) - (7) \tag{16}$$

$$z \geq C(s, t) \quad \forall s, t \text{ que satisfaz } (9) - (14) \tag{17}$$

Notemos que existe uma desigualdade em (17) para cada estratégia viável de interdição. Como o total de interdições é dado por $\binom{p-q}{r}$, tais restrições devem ser geradas sob demanda.

Seja uma solução relaxada $(\overline{z}, \overline{x}, \overline{s}) \in \mathbb{R} \times [0,1]^{|I|} \times [0,1]^{|J|\times|I|}$ que satisfaz (16) e algumas restrições já adicionadas de (17). 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 (9) - (14) considerando os valores de $\overline{s}$ na função objetivo. É válido ressaltar que as soluções relaxadas podem ser separadas mesmo quando são fracionárias.

Neste contexto, o algoritmo de *branch-and-cut* resolve o modelo dado por (15) - (17) usando resolvedores comerciais. As restrições (17) são adicionadas sob demanda resolvendo o modelo exato (9) - (14). Uma vantagem do método proposto é que nosso algoritmo de *branch-and-cut* permite o uso de resolvedores comerciais. Assim, seus algoritmos heurísticos e de planos de corte podem ser utilizados tanto no problema de primeiro nível quanto no problema da separação.

Com o objetivo de acelerar o método, é definido um parâmetro ϵ para reduzir o número total de cortes separados. Enquanto o *gap* for maior do que ϵ, os cortes são separados para qualquer solução da relaxação linear encontrada durante o *branch-and-bound*. Quando o *gap* se torna menor do que ϵ, os cortes são separados apenas para soluções inteiras.

5 Resultados Computacionais

Nesta seção, são apresentados resultados computacionais do método proposto aplicado ao PMFI. Os experimentos incluem testes sobre o usual conjunto de dados *150-node London* (Goodchild e Noronha, 1983) composto de 150 nós ($n = 150$) que é frequentemente usado como *benchmark* para o problema da p-mediana. Cada um dos 150 nós é um cliente ($n = 150$). Os outros parâmetros da instância variam da seguinte maneira: $p \in \{40, 50, 60\}$, $q \in \{4, 5, 6, 7, 8, 9, 10, 12\}$ e $r \in \{2, 3, 4, 5, 6, 7, 8, 9, 10\}$. Em todos os testes, o conjunto de p facilidades existentes é representado pela p-mediana ótima. Tal consideração já é adotada pelos outros artigos que tratam do PMFI, facilitando a comparação entre os resultados. Depois de uma calibragem, estipulamos $\epsilon = 0,05$. Todos os testes foram executados em um computador com um processador Intel Core i7 duo 3,60 GHz e com 3 Gb de memória RAM. O resolvedor utilizado foi o CPLEX 12.5.

Esta seção é dividida em outras três subseções da seguinte forma. Na seção 5.1, vamos avaliar o impacto da reformulação, comparando os resultados obtidos neste trabalho com os obtidos por Pessoa *et al*. (2013). Na seção 5.2, comparamos nosso método ao proposto por Scaparra e Church (2008a). Finalmente, a seção 5.3 apresenta estatísticas detalhadas do método proposto e apresenta as soluções ótimas para instâncias com até $n = 150$, $p = 60$, $q = 10$ e $r = 10$, onde diversas destas ainda não estavam comprovadas na literatura.

5.1 Resultados associados a reformulação.

Nesta seção, mostramos qual o impacto no tempo de resolução do algoritmo de *branch-and-cut* com a reformulação ao resolvermos as instâncias da literatura. Para isso, comparamos os tempos computacionais obtidos pelo nosso método com os obtidos por Pessoa *et al*. (2013). É válido ressaltar que possuimos acesso ao código, possibilitando a execução dos testes comparativos em um mesmo computador. A Tabela 1 mostra a comparação entre os dois métodos e possui a seguinte legenda para as colunas. As colunas n, p, q e r apresentam as características da instância. As colunas *Este Trabalho* e Pessoa *et al*. (2013) mostram respectivamente os tempos computacionais em segundos de cpu para o nosso método e o método proposto por Pessoa *et al*. (2013). A coluna $\frac{Pessoa \text{ et al. } (2013)}{Este\ Trabalho}$ apresenta a razão entre as colunas Pessoa *et al*. (2013) e *Este Trabalho*.

Observando a tabela 1, podemos concluir que o uso da reformulação contribui de maneira significativa para a redução do tempo computacional uma vez que na média, este método mostrou-se mais de cinco vezes mais rápido. Esta aceleração permitiu a comprovação da otimalidade das instâncias deixadas em aberto por Pessoa *et al*. (2013), como é visto com detalhes na seção 5.3.

Tabela 1: Análise do impacto da reformulação..

Características da instância				Tempo(s)		
n	p	q	r	Este trabalho	Pessoa *et al*. (2013)	$\frac{\text{Pessoa } et\ al.\ (2013)}{\text{Este Trabalho}}$
150	60	4	6	9,11	105,3	11,56
150	60	4	7	32,67	175,77	5,38
150	60	4	8	31,47	239,72	7,62
150	60	4	9	42,22	117,13	2,77
150	60	4	10	104,41	342,06	3,28
150	60	6	6	79,94	350,75	4,39
150	60	6	7	82,98	583,5	7,03
150	60	6	8	144,91	544,64	3,76
150	60	6	9	261,66	846,02	3,23
150	60	6	10	488,08	597,36	1,22
150	60	8	6	128,45	959,23	7,47
150	60	8	7	134,89	1873,19	13,89
150	60	8	8	821,38	4003,78	4,87
150	60	8	9	1741,80	6285,39	3,61
150	60	8	10	2476,39	6064,56	2,45
150	60	10	6	417,89	2957,31	7,08
150	60	10	7	1032,84	7281,41	7,05
150	60	10	8	4388,27	17065,53	3,89
150	60	10	9	6320,55	$\geq$ 25200,00	$\geq$ 3,99
150	60	10	10	24778,83	$\geq$ 25200,00	$\geq$ 1,02
Média				2175,94	$\geq$ 5039,63	$\geq$ 5,28

5.2 Comparação entre o nosso método e o proposto por Scaparra e Church (2008a).

Nesta subseção é apresentada uma comparação entre o método proposto e a Enumeração Implícita (EI) proposta por Scaparra e Church (2008a). Os autores deste artigo testaram novamente em uma máquina e resolvedor mais modernos e nos enviaram os resultados das instâncias. As configurações da máquina e do resolvedor utilizados pelos autores são as seguintes: HP2500 *workstation*, com um Intel(R) Xeon(R), CPU E5630, 2,53 GHz, 6Gb de memória Ram e CPLEX 12.5.

A Tabela 2 mostra a comparação para instâncias com $p \in \{40, 50, 60\}$ e pequenos valores de r ($r \leq 5$). Já a Tabela 3 mostra a comparação para instâncias com $p = 40$ e altos valores de r ($r > 5$). O seguinte cabeçalho foi utilizado para as colunas: n, p, q e r indicam as características da instância, *Tempo(s)* indica o tempo computacional em segundos. Coluna $\frac{IE}{Este\ Trabalho}$ indica a razão entre o tempo computacional consumido pelo IE e o tempo consumido pelo método proposto. Assim, quando este valor é maior do que um, tem-se que o método proposto foi mais rápido e este valor é marcado em negrito. O resultado do tempo computacional da instância com $n = 150$, $p = 40$, $q = 10$ e $r = 10$ foi omitido uma vez que o método EI não encontrou a solução ótima para esta instância em tempo computacional aceitável.

Observando as tabelas 2 e 3, podemos notar que o nosso método só não foi mais rápido apenas para 1 das 56 instâncias comparadas. Além disso, para 39 das 56 instâncias nosso método foi mais de 10 vezes mais rápido, onde em 8 destas 39, o nosso método foi mais de cem vezes mais rápido. Com base nestas observações, podemos concluir que, apesar da ligeira diferença entre as máquinas, o nosso método se mostrou mais robusto, independentemente do tamanho das instâncias.

5.3 Estatísticas computacionais detalhadas do método proposto.

Nesta seção, apresentamos estatísticas do método para diversas instâncias com ($p \geq 40$ e $r \geq 6$). Tabelas 4 e 5 mostram os resultados para $p = 40$ e $p = 60$ respectivamente. Devido a falta

Tabela 2: Comparando o tempo computacional consumido pelo método proposto e pelo método proposto por Scaparra e Church (2008a) (EI) para instâncias com $p \in \{40, 50, 60\}$ e pequenos valores de r.

Instância Características				Valor ótimo	Tempo(s)		
n	p	q	r		Este trabalho	EI	$\frac{\text{EI}}{\text{Este Trabalho}}$
150	40	4	2	75676,41	0,22	0,28	**1,27**
150	40	4	3	81766,35	2,19	2,31	**1,05**
150	40	4	4	88495,16	1,03	7,83	**7,60**
150	40	4	5	94687,71	6,42	21,31	**3,32**
150	40	6	2	75418,05	0,34	0,83	**2,44**
150	40	6	3	81424,85	2,13	8,80	**4,13**
150	40	6	4	87178,67	1,41	55,60	**39,43**
150	40	6	5	93286,72	23,41	206,30	**8,81**
150	40	8	2	74847,58	1,75	2,09	**1,19**
150	40	8	3	80370,63	7,69	46,35	**6,03**
150	40	8	4	86182,77	9,78	436,46	**44,63**
150	40	8	5	91664,38	45,73	1683,56	**36,82**
150	50	5	2	60168,5	0,38	0,42	**1,11**
150	50	5	3	65160,41	0,86	4,65	**5,41**
150	50	5	4	69918,29	4,91	16,08	**3,27**
150	50	5	5	74694,85	8,91	58,13	**6,52**
150	50	8	2	59225,56	1,59	1,73	**1,09**
150	50	8	3	63552,59	2,75	28,88	**10,50**
150	50	8	4	68302,73	13,69	163,47	**11,94**
150	50	8	5	73055,22	55,97	967,65	**17,29**
150	50	10	2	58553,08	3,50	3,29	0,94
150	50	10	3	62261,17	5,91	93,57	**15,83**
150	50	10	4	67026,53	38,27	628,05	**16,41**
150	50	10	5	71140,4	46,48	5478,18	**117,86**
150	60	6	2	46563,64	1,13	1,15	**1,02**
150	60	6	3	50809,54	0,91	10,56	**11,60**
150	60	6	4	54621,16	3,66	33,12	**9,05**
150	60	6	5	58615,76	21,69	122,84	**5,66**
150	60	9	2	45889,14	2,33	2,36	**1,01**
150	60	9	3	49697,61	8,69	69,64	**8,01**
150	60	9	4	53509,22	30,06	333,38	**11,09**
150	60	9	5	56932,06	80,38	1575,12	**19,60**
150	60	12	2	45310,07	3,13	3,93	**1,26**
150	60	12	3	48814,47	7,94	329,41	**41,49**
150	60	12	4	52011,79	63,36	1730,80	**27,32**
150	60	12	5	55469,28	362,53	11107,79	**30,64**

de espaço, os resultados para $p = 50$ estão sendo omitidos. O seguinte cabeçalho foi usado para as colunas: n, p, q, e r indicam as características da instância. *Opt* indica o custo ótimo da solução. *LB raiz* indica o limite inferior no nó raiz, enquanto *Gap Raiz(%)* indica o *gap* entre as colunas *LB raiz* e *Opt*, *Nós #B&B* indica o total número de nós criados pela árvore de *branch-and-cut*, *#Sep* indica o número de separações, *#Cortes* indica o número total de cortes separados. *Tempo Raiz*, *Tempo Sep* e *Tempo Total* indicam o tempo de execução total acumulado em segundos consumidos no nó raiz, pelo algoritmo de separação e o algoritmo de *branch-and-cut* completo respectivamente. Na coluna *Opt*, os valores que estão marcados em negrito são referentes a instâncias cuja a otimalidade está sendo comprovada pela primeira vez neste artigo.

Observando as tabelas 4 e 5, podemos notar que 7 instâncias estão sendo resolvidas pela

Tabela 3: Comparando o tempo computacional consumido pelo método proposto e pelo método proposto por Scaparra e Church (2008a) (EI) para instâncias com $p = 40$ e grandes valores de r.

Instância Características				Valor ótimo	Tempo(s)		
n	p	q	r		Este trabalho	EI	$\frac{\text{EI}}{\text{Este trabalho}}$
150	40	4	6	101598,44	8,70	59,03	**6,79**
150	40	4	7	108225,05	13,23	109,12	**8,25**
150	40	4	8	115080,07	7,92	211,26	**26,67**
150	40	4	9	122170,27	33,11	386,55	**11,67**
150	40	4	10	130408,32	39,33	631,91	**16,07**
150	40	6	6	100078,89	44,25	690,93	**15,61**
150	40	6	7	106352,95	34,92	1496,06	**42,84**
150	40	6	8	113960,94	183,95	3531,39	**19,20**
150	40	6	9	120606,98	412,52	7101,38	**17,21**
150	40	6	10	126403,82	423,58	12041,69	**28,43**
150	40	8	6	97508,17	62,78	6589,98	**104,97**
150	40	8	7	102380,26	179,72	16845,28	**93,73**
150	40	8	8	108230,84	477,13	44151,18	**92,53**
150	40	8	9	113464,87	697,59	100684,59	**144,33**
150	40	8	10	118595,86	1155,14	146913,77	**127,18**
150	40	10	6	94526,17	492,95	47658,43	**96,68**
150	40	10	7	99124,14	447,41	139906,16	**312,70**
150	40	10	8	103738,66	887,80	370635,16	**417,48**
150	40	10	9	108677,52	1467,13	836554,54	**570,20**

primeira vez na otimalidade. Além disso, apesar da complexidade do problema, todas as instâncias foram resolvidas em tempos computacionais razoáveis, onde apenas 9 das 40 instâncias testadas necessitaram mais de 1000 segundos. Por outro lado, os resultados indicam ainda perspectivas de melhorias. Ao observarmos as colunas *Gap Raiz(%)* e *Nós #B&B*, por exemplo, percebemos que em apenas 7 das 40 instâncias testadas, o *gap* da raiz foi inferior a 10% e também percebemos um elevado número de nós. Isto sugere que, se encontrarmos cortes fortalecidos para o problema de primeiro nível, podemos reduzir substancialmente o tamanho da árvore de B&B.

6 Conclusões

Neste artigo, propusemos uma reformulação para o PMFI capaz de diminuir um cojunto bastante extenso tanto de variáveis quanto de restrições. Para comprovar o impacto da reformulação, derivamos um algoritmo *branch-and-cut* e os resultados foram comparados com a antiga formulação e com a literatura existente para o problema. Os resultados mostraram que a nova formulação aliada ao algoritmo se mostrou bem mais rápida que a literatura além de ter resolvido 7 instâncias que estavam em aberto.

A nível de trabalhos futuros, uma das perspectivas é explorar uma das fraquezas apresentadas pelo método que foi o grande tamanho da árvore de B&B. Para reduzir este tamanho, pretendemos encontrar cortes fortalecidos para o problema de primeiro nível.

Referências

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.

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.

Tabela 4: Estatísticas do método para instâncias com $n = 150$, $p = 40$, $q \in \{4, 6, 8, 10\}$ e $r \in \{6, 7, 8, 9, 10\}$.

n	p	q	r	Opt	LB Raiz	Gap Raiz(%)	Nós #B&B	#Sep	#Cortes	Tempo Sep	Tempo Raiz	Tempo Total
150	40	4	6	101598,44	94277,25	7,21	292	22	18	4,31	7,59	8,70
150	40	4	7	108225,05	106352,95	1,73	11	19	16	11,05	13,03	13,23
150	40	4	8	115080,07	115080,07	0,00	0	15	12	7,92	7,70	7,92
150	40	4	9	122170,27	112953,61	7,54	751	31	27	13,72	29,92	33,11
150	40	4	10	130408,32	118384,23	9,22	1804	35	30	14,06	31,05	39,33
150	40	6	6	100078,89	89881,99	10,19	6945	61	58	5,28	22,41	44,25
150	40	6	7	106352,95	95030,64	10,65	2802	55	52	9,89	25,17	34,92
150	40	6	8	113960,94	99690,55	12,52	17239	214	210	5,61	110,66	183,95
150	40	6	9	120606,98	104469,32	13,38	28602	404	389	11,36	265,38	412,52
150	40	6	10	126403,82	109369,26	13,48	35230	355	349	7,92	218,39	423,58
150	40	8	6	97508,17	86936,59	10,84	8703	130	124	5,97	32,08	62,78
150	40	8	7	102380,26	90932,36	11,18	23950	212	208	17,63	85,25	179,72
150	40	8	8	108230,84	94867,46	12,35	73555	462	456	8,53	144,91	477,13
150	40	8	9	113464,87	98417,95	13,26	87928	599	591	7,02	228,45	697,59
150	40	8	10	118595,86	102746,33	13,36	107733	1053	1045	7,23	343,39	1155,14
150	40	10	6	94526,17	83889,01	11,25	116660	336	328	7,39	59,70	492,95
150	40	10	7	99124,14	87372,17	11,86	62219	722	714	9,25	159,09	447,41
150	40	10	8	103738,66	91183,28	12,10	143144	764	753	8,06	161,52	887,80
150	40	10	9	108677,52	94200,71	13,32	189557	1108	1096	7,19	242,98	1467,13
150	40	10	10	**113149,70**	97677,25	13,67	237774	1690	1684	7,94	390,16	2546,48

Tabela 5: Estatísticas do método para instâncias com $n = 150$, $p = 60$, $q \in \{4, 6, 8, 10\}$ e $r \in \{6, 7, 8, 9, 10\}$.

n	p	q	r	Opt	LB Raiz	Gap Raiz(%)	Nós #B&B	#Sep	#Cortes	Tempo Sep	Tempo Raiz	Tempo Total
150	60	4	6	63761,97	58881,82	7,65	323	24	22	4,03	7,97	9,11
150	60	4	7	69139,79	61621,28	10,87	1070	35	32	7,17	29,39	32,67
150	60	4	8	74730,36	65615,16	12,20	1019	44	42	7,55	27,98	31,47
150	60	4	9	79351,34	70264,56	11,45	649	32	29	17,91	39,75	42,22
150	60	4	10	84507,98	74741,99	11,56	3455	74	70	16,58	93,36	104,41
150	60	6	6	62915,66	55417,94	11,92	5577	189	183	5,42	62,55	79,94
150	60	6	7	66681,65	60207,74	9,71	2873	114	107	20,67	73,05	82,98
150	60	6	8	70687,26	62464,13	11,63	19126	124	116	9,31	76,25	144,91
150	60	6	9	74504,22	66257,17	11,07	23455	167	158	18,13	176,11	261,66
150	60	6	10	79387,74	69772,40	12,11	53433	245	238	11,97	273,91	488,08
150	60	8	6	61129,04	54322,59	11,13	17723	210	206	8,34	69,20	128,45
150	60	8	7	64769,70	57319,16	11,50	16229	188	182	10,33	78,14	134,89
150	60	8	8	69195,09	60530,80	12,52	109311	580	576	9,91	385,00	821,38
150	60	8	9	**73254,84**	63333,78	13,54	206256	953	944	12,17	744,50	1741,80
150	60	8	10	**77280,88**	66055,70	14,53	298843	1031	1022	8,55	861,53	2476,39
150	60	10	6	60201,28	53141,24	11,73	82208	543	530	9,67	132,84	417,89
150	60	10	7	**64273,21**	55535,07	13,60	166504	1048	1044	11,23	363,03	1032,84
150	60	10	8	**67807,04**	58226,41	14,13	429940	2735	2708	11,67	1344,89	4388,27
150	60	10	9	**71453,90**	60751,64	14,98	601488	2750	2740	9,63	1441,13	6320,55
150	60	10	10	**75006,42**	62728,75	16,37	980599	5702	5656	7,42	3076,36	24778,83

Goodchild, M. F. e Noronha, V. T. *Location-allocation for small computers*. Department of Geography, University of Iowa, 1983.

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

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.

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.

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.

Washburn, A. e Wood, K. (1995), Two-person zero-sum games for network interdiction. *Operations Research*, v. 43, n. 2, p. 243–251.

Wood, R. K. (1993), Deterministic network interdiction. *Mathematical and Computer Modelling*, v. 17, p. 1–1.