Uma Metaheurística GRASP para o Problema de Planejamento de Redes com Rotas Ótimas para o Usuário

Pedro Henrique González
Instituto de Computação
Universidade Federal Fluminense, Niterói, Brasil
pegonzalez@ic.uff.br

Luidi Gelabert Simonetti
Instituto de Computação
Universidade Federal Fluminense, Niterói, Brasil
luidi@ic.uff.br

Carlos Alberto de Jesus Martinhon
Instituto de Computação
Universidade Federal Fluminense, Niterói, Brasil
mart@dcc.ic.uff.br

Edcarllos Santos
Instituto de Computação
Universidade Federal Fluminense, Niterói, Brasil
esantos@ic.uff.br

Philippe Yves Paul Michelon
Laboratoire d'Informatique d'Avignon
Université d'Avignon et des Pays de Vaucluse, Avignon, France
philippe.michelon@univ-avignon.fr

RESUMO

Dado o constante desenvolvimento da sociedade, quantidades cada vez maiores de produtos precisam ser transportadas nos grandes centros urbanos. Devido a este fato, surgem os problemas de planejamento de rede como ferramenta de apoio à tomada de decisão, com o intuito de suprir a necessidade de determinar maneiras eficientes de se realizar tal transporte. Neste trabalho apresenta-se uma formulação matemática do problema de planejamento de redes com rotas ótimas para o usuário como um problema de programação linear discreta mista em dois níveis. Em seguida, discute-se uma formulação inteira em um nível obtida através da aplicação das condições de Karush-Kuhn-Tucker. Foram implementados um algoritmo construtivo randomizado e uma busca local específica, que juntos foram combinados em uma metaheurística GRASP. Os resultados computacionais obtidos foram comparados com os resultados encontrados pelo modelo em um nível e com resultados encontrados na literatura.
Palavras Chaves: Planejamento de Redes; GRASP; Problema em Dois Níveis.

Área Principal: Logística e Transportes

ABSTRACT

Due to constant development of society, increasing quantities of commodities have to be transported in large urban centers. Thanks to that fact, network planning problems arises as tools to support decision-making, aiming to meet the need of finding efficient ways to perform such transportations. This paper presents a mathematical formulation of the network design problem with user-optimal flow as a mixed discrete bilevel linear programming problem. In this work we also discuss a one-level integer formulation obtained by applying Karush-Kuhn-Tucker conditions. We implemented a randomized constructive algorithm, a local search and combined them into a GRASP metaheuristic. In addition, we compare the computational results we obtained with the results found by the one-level formulation and with the results found in the literature.
Keywords: Network Design; GRASP; Bilevel Problem.

Main Area: Logistics and Transport

1 Introdução

O Problema de Planejamento de Redes com Custo Fixo (PPRCF) envolve selecionar um subconjunto de arestas de um grafo, de tal maneira que um certo número de mercadorias possam ser transportadas a partir de suas origens aos seus destinos. O problema consiste na minimização da soma dos custos fixos (em função da escolha das arestas) e custos variáveis (em função dos fluxos de mercadorias sobre as arestas). Custos fixos e variáveis podem ser representados por funções lineares e os arcos não possuem capacidade. O PPRCF pertence a uma grande classe de problemas de projeto de redes (Magnanti e Wong (1984)). Na literatura, pode-se encontrar diversas variações do PPRCF (Boesch (1975)) tais como problema de caminho mais curto, problema de árvore geradora mínima, problema de roteamento de veículos, problema do caixeiro viajante e problema de rede de Steiner (Magnanti e Wong (1984a)), sendo que para cada uma delas restrições são adicionadas ao PPRCF. Demonstrado por vários livros e coleções editadas de documentos (e.g., Boesch (1975); Boyce (1979); Mandl (1981)), o problema de planejamento de redes tem inúmeras aplicações. Este modelo não só representa o PPRCF, mas também problemas de comunicação, transporte, sistemas de esgoto e planejamento de recursos. Ele também surge em outros contextos, tais como sistemas de produção flexíveis e automatizados (Kimenia e Gershwin (1979); Graves e Lamar (1983)). Finalmente, o problema de planejamento de redes surge em muitos problemas de frota de veículos que não envolvem a construção de instalações físicas, ou seja, apenas decide-se o caminho percorrido para cada veículo enviado. (Simpson (1969); Magnanti (1981)).

Nos problemas de planejamento de redes, não só as versões mais simples do problema são NP Difíceis (Johnson, Lenstra e Rinnooy Kan (1978); Wong(1978)), como também mesmo a tarefa de encontrar soluções viáveis (para problemas com restrição de orçamentos sobre o custo fixo) é extremamente complexa (Wong (1980)). Devido às dificuldades naturais do problema, métodos heurísticos e híbridos se apresentam como uma boa alternativa na busca de soluções de qualidade.

Neste trabalho será abordado uma variação específica do PPRCF, o Problema de Planejamento de Redes com Rotas Ótimas para o Usuário (PPR-ROU), que consiste em adicionar múltiplos problemas de caminho mínimo ao problema original. O PPR-ROU pode ser modelado como um problema de programação linear discreta mista em dois níveis. Este tipo de problema envolve dois agentes distintos agindo simultaneamente ao invés de agirem de forma sequencial na hora de tomar decisões. No nível superior, o líder (1^o agente) é encarregado de escolher o conjunto de arestas a serem abertas visando minimizar os custos fixos e variáveis. Em contrapartida, no nível inferior, o seguidor (2^o agente) deve escolher um conjunto de caminhos mínimos na rede, resultando no trajeto pelo qual as mercadorias serão enviadas. O efeito de um agente sobre o outro é indireto: a decisão dos seguidores é afetada pela rede projetada no nível superior, enquanto que a decisão do líder é afetada pelos custos variáveis impostos pelas rotas definidas no nível mais baixo.

A inclusão de restrições de caminho mínimo em um problema de programação linear inteira mista não é direta. As dificuldades surgem tanto na modelagem, bem como na concepção de métodos de solução eficientes. Na literatura, observa-se poucos trabalhos relacionados ao problema ou particularizações do problema, sendo que para a variante abordada destacam-se (Billheimer e Gray (1973); Kara e Verter (2004); Erkut, Erhan, Tjandra e Verter (2007); Mauttone, Labbe e Figueiredo (2008); Erkut e Gzara (2008); Amaldi,Bruglieri e Fortz (2011)) e tem sido tratado como parte de problemas maiores em algumas aplicações (Holmberg e Yuan (2004)).

O PPR-ROU aparece, principalmente, no planejamento de redes para o transporte de materiais nocivos (Kara e Verter (2004); Erkut, Erhan, Tjandra e Verter (2007); Erkut e Gzara (2008); Amaldi, Bruglieri e Fortz (2011)). Na solução deste problema, o governo define uma seleção de estradas a serem utilizadas para o transporte de materiais nocivos, assumindo que o transporte dos materiais nocivos na rede resultante será feito ao longo de caminhos mínimos. Não há custos associados com a seleção de estradas para compor a rede, contudo o governo deseja minimizar a exposição da população no caso de acidente durante um transporte destas mercadorias. Este é um caso particular

do problema PPR-ROU onde os custos fixos são iguais a zero.

Considerando as dificuldades inerentes ao problema, este trabalho tem como objetivo apresentar novas heurísticas para o PPR-ROU e realizar uma comparação de resultados entre as heurísticas desenvolvidas e os modelos de programação matemática. Por fim, compara-se os resultados obtidos com os presentes na literatura para a variante trabalhada.

Este trabalho esta organizado da seguinte forma: na Seção 2, apresenta-se a formulação matemática do PPRCF e do PPR-ROU. Na Seção 3 é abordada a formulação matemática do problema em um nível do PPR-ROU. A Seção 4 detalha as heurísticas desenvolvidas. Já a Seção 5 traz os resultados computacionais alcançados tanto pelas heurísticas, quanto para o modelo de programação matemática. Enfim na Seção 6 tem-se as conclusões e a indicação de trabalhos futuros a serem desenvolvidos.

2 Formulação Matemática: PPRCF e PPR-ROU

Esta Seção abrange os modelos de programação matemática aplicados ao PPRCF e o PPR-ROU. Com o intuito de facilitar a compreensão dos modelos anunciados, uma lista contendo os símbolos utilizados e suas respectivas denominações é apresentado de antemão. Nota-se que (i, j) e (j, i) denotam arcos dirigidos correspondentes a aresta $[i, j]$ não direcionada.

Conjuntos e Parâmetros:

V	Conjunto de nós.
E	Conjunto de arestas bidirecionadas .
K	Conjunto de produtos.
$\delta^+(i)$	Conjunto de todos arcos que saem do nó i.
$\delta^-(i)$	Conjunto de todos arcos que chegam ao nó i.
c_e	Comprimento da aresta $e = [i, j]$.
$o(k)$	Origem do produto k.
$d(k)$	Destino do produto k.
g_{ij}^k	Custo variável de passar com o produto k pela aresta $[i, j] \in E$.
f_{ij}	Custo fixo de abertura da aresta $[i, j] \in E$.

Variáveis:

y_{ij}	Indica se a aresta $[i, j]$ está na rede.
x_{ij}^k	Indica se o produto k passa pela arco (i, j).

2.1 Modelo para o PPRCF

Uma vez que a estrutura do problema pode ser facilmente modelada através de um grafo, tem-se que para a construção da rede são utilizados um conjunto V de nós que representam as facilidades e um conjunto E de arestas não capacitadas representando o deslocamento entre as instalações. Além disso, define-se como K o conjunto de produtos a serem transportados pela rede, sendo que estes podem representar bens físicos, como matéria prima para a indústria, material nocivo ou até mesmo pessoas. Para cada um dos produtos $k \in K$, tem-se um fluxo de entrega a partir de um ponto de origem, denotado por $o(k)$, até um ponto destino denotado por $d(k)$. A formulação aqui apresentada, trabalha com variantes que apresentam produtos com múltiplas origens e destinos, sendo que para tratar tal caso, basta considerar que para cada par $(o(k), d(k))$, existirá um novo produto resultante da dissociação do mesmo em vários (desagregação de fluxo).

O modelo para o PPRCF possui dois tipos de variáveis, uma referente à construção da rede e outra referente à modelagem de fluxo contínuo. Seja y_{ij} uma variável binária, tem-se que $y_{ij} = 1$ quando o arco (i, j) é escolhido como parte do projeto de rede e $y_{ij} = 0$, caso contrário. Neste caso, x_{ij}^k denota o fluxo de produto k no arco (i, j). Apesar das arestas no modelo não possuírem direção, estas podem ser referenciadas como arcos, pois o fluxo para cada produto é direcionado. O conjunto de todos os arcos que saem do nó i é denotado por $\delta^+(i)$ e de forma complementar, $\delta^-(i)$ simboliza o conjunto de todas arestas que chegam ao nó i. Tratando $y = (y_{ij})$ e $x = (x_{ij}^k)$ respectivamente como vetores de adição de aresta e de variáveis de fluxo, tem-se que o problema pode ser modelado da seguinte forma:

$$\begin{cases} \min & \displaystyle\sum_{(i,j)\in E} f_{ij}y_{ij} + \sum_{k\in K}\sum_{(i,j)\in E} g_{ij}^k x_{ij}^k & & \\ \text{s.a.} & \displaystyle\sum_{(i,j)\in\delta^+(i)} x_{ij}^k - \sum_{(i,j)\in\delta^-(i)} x_{ij}^k = b_i^k, & \forall i,j \in V, \forall k \in K, & (1) \\ & x_{ij}^k + x_{ji}^k \leq y_{ij}, & \forall (i,j) \in E, \forall k \in K, & (2) \\ & x_{ij}^k \geq 0, & \forall (i,j) \in E, \forall k \in K, & (3) \\ & y_{ij} \in \{0,1\}, & \forall (i,j) \in E, & (4) \end{cases}$$

onde:

$$b_i^k = \begin{cases} -1 & \text{se } i = d^k, \\ 1 & \text{se } i = o^k, \\ 0 & \text{caso contrário.} \end{cases}$$

O conjunto de restrições (1) representa as equações de conservação de fluxo para cada nó e cada produto. Já o conjunto de restrições denotado por (2), assegura que nenhum fluxo passe por um arco a menos que seu custo de adição seja pago, ou seja, este tem de ser construído para ser utilizado.

2.2 Formulação Matemática do PPR-ROU em dois níveis

O PPR-ROU é uma variante do PPRCF onde cada produto $k \in K$ é transportado por uma rota ótima entre sua origem $o(k)$ e destino $d(k)$. Tal mudança acarreta a adição de novas restrições ao problema geral. No PPR-ROU, além de selecionar um subconjunto de E cujo somatório dos custos fixos e variáveis seja mínimo (problema líder), cada produto $k \in K$ deve ser transportado pelo caminho mínimo entre $o(k)$ e $d(k)$ (problema seguidor). O PPR-ROU pertence à classe dos problemas NP-Difíceis e pode ser modelado como um problema de programação linear em dois níveis (Mauttone, Labbe e Figueiredo (2008)), como segue:

$$\begin{cases} \min & \displaystyle\sum_{(i,j)\in E} f_{ij}y_{ij} + \sum_{k\in K}\sum_{(i,j)\in E} g_{ij}^k x_{ij}^k & & \\ \text{s.a.} & y_{ij} \in \{0,1\}, & \forall e = (i,j) \in E, & (5) \end{cases}$$

onde x_{ij}^k é solução do problema:

$$\begin{cases} \min\limits_{x} & \sum\limits_{k\in K}\sum\limits_{(i,j)\in E} c_{ij}x_{ij}^k & & \\ \text{s.a.} & \sum\limits_{(i,j)\in\delta^+(i)} x_{ij}^k - \sum\limits_{(i,j)\in\delta^-(i)} x_{ij}^k = b_i^k, & \forall i,j\in V, \forall k\in K, & (6)\\ & x_{ij}^k + x_{ji}^k \leq y_e, & \forall e=(i,j)\in E, \forall k\in K, & (7)\\ & x_{ij}^k \geq 0, & \forall e=(i,j)\in E, \forall k\in K. & (8) \end{cases}$$

onde:

$$b_i^k = \begin{cases} -1 & \text{se } i = d^k, \\ 1 & \text{se } i = o^k, \\ 0 & \text{caso contrário.} \end{cases}$$

Analisando o modelo apresentado para o PPR-ROU, pode-se observar que o conjunto das restrições (5) garante que y_e assuma apenas valores binários. Em (6), verifica-se as restrições de fluxo, sendo que b_i^k permanece inalterado uma vez que as arestas permanecem não-capacitadas. O conjunto de restrições (7), impede que haja fluxo em arcos cujas arestas correspondentes estejam fechadas. Por fim, (8) impõe a restrição de não negatividade sobre as variáveis x_{ij}^k. É interessante notar que resolver o problema seguidor é equivalente a resolver $|K|$ problemas de caminho mínimo independentes.

3 Formulação Matemática do PPR-ROU em um nível

O modelo em um nível apresentado a seguir, trata-se de uma variação do modelo apresentado por Kara e Verter[2004], onde diferente do modelo original, um custo fixo é associado a cada aresta. Dado os valores de y_e, o problema interno é unimodular (Wolsey (1998)), sendo assim, a integralidade de x_{ij}^k pode ser substituída por $x_{ij}^k \geq 0$ sem perda de otimalidade. Com a adição desta característica, pode-se representar o problema seguidor pelas condições de Karush-Kuhn-Tucker. Através dos fatores apresentados, é possível representar o PPR-ROU através de uma formulação de programação não-linear inteira mista.

Uma vez que não é o objetivo deste trabalho abordar métodos não-lineares, aplica-se o método de linearização por coeficiente Big-M, de modo que o modelo possa ser escrito como uma formulação de programação linear inteira mista em um nível, como segue:

$$\begin{cases} \min & \sum\limits_{(i,j)\in E} f_{ij}y_{ij} + \sum\limits_{k\in K}\sum\limits_{(i,j)\in E} g_{ij}^k x_{ij}^k & & \\ \text{s.a.} & \sum\limits_{(i,j)\in\delta^+(i)} x_{ij}^k - \sum\limits_{(i,j)\in\delta^-(i)} x_{ij}^k = b_i^k, & \forall i,j\in V, \forall k\in K, & (9)\\ & x_{ij}^k + x_{ji}^k \leq y_{ij}, & \forall e=(i,j)\in E, \forall k\in K, & (10)\\ & c_e - w_i^k + w_j^k - v_{ij}^k + \lambda_{ij}^k = 0, & \forall e=(i,j)\in E, k\in K, & (11)\\ & v_{ij}^k \leq M(1-x_{ij}^k), & \forall k\in K, e=(i,j)\in E & (12)\\ & \lambda_{ij}^k \leq M[1-(y_e - x_{ij}^k)], & \forall k\in K e=(i,j)\in E, & (13)\\ & v_{ij}^k \geq 0, \lambda_{ij}^k \geq 0, w_i^k \in \mathbb{R}, & \forall k\in K, e=(i,j)\in E & (14)\\ & x_{ij}^k \in \{0,1\}, & \forall e=(i,j)\in E, \forall k\in K, & (15)\\ & y_{ij} \in \{0,1\}, & \forall e=(i,j)\in E. & (16) \end{cases}$$

onde:

$$b_i^k = \begin{cases} -1 & \text{se } i = d^k, \\ 1 & \text{se } i = o^k, \\ 0 & \text{caso contrário.} \end{cases}$$

Sendo que as restrições (9),(10),(15) e (16) são mantidas do modelo PPR-ROU em dois níveis abordado na Subseção 2.2, temos como componente adicional deste modelo as restrições (11), (12) e (13) representando as condições de Karush-Kuhn-Tucker para o problema seguidor, ou seja, os problemas de caminho mínimo.

4 Métodos Heurísticos

Este trabalho consiste primordialmente em propor um algoritmo construtivo aleatorizado e uma busca local para o PPR-ROU. Para extrair o melhor de cada componente desenvolvida, combina-se os métodos propostos em uma metaheurística GRASP, com o intuito de encontrar soluções viáveis de melhor qualidade para o problema abordado.

4.1 Desacoplamento Parcial

Uma heurística de desacoplamento total para o PPR-ROU, se baseia na ideia de desacoplar o problema de construção da rede, do problema de caminho mínimo. Entretanto, como discutido em Erkut e Gzara (2008), que o desacoplamento do problema original pode proporcionar resultados piores do que ao se tratar os dois problemas simultaneamente. Assim sendo, este algoritmo propõe o que chamamos de desacoplamento parcial, onde considera-se alguns fatores do problema seguidor ao tentar construir uma solução para o problema líder.

Com a rede previamente construída, aplica-se um algoritmo de caminho mínimo para levar cada produto de sua origem $o(k)$ até seu destino $d(k)$. Para que o procedimento fique claro, é importante ressaltar que $g_{ij}^k = q^k \beta_{ij}$, onde q^k representa a quantidade do produto k e β_{ij} representa custo de transporte pela aresta $e = [i, j]$. O funcionamento do método pode ser observado através do Algoritmo 1.

Algoritmo 1: Desacoplamento Parcial

Entrada: E, K, c, f, g, q, γ
Dados: $MaxCusto \leftarrow \infty, \alpha \leftarrow 1, y \leftarrow 0, x \leftarrow 0;$
inicio
  $OrdenarK(K);$
  $\bar{k} \leftarrow K(1);$
  $K \leftarrow K \backslash \{\bar{k}\};$
  $K_G \leftarrow K;$
  para $e = (i, j)$ *de* $1 \ldots |E|$ **faça**
    $CustoDL(e, \bar{k}) \leftarrow (f_e \times (1 - y_e)) + (\alpha \times g_{ij}^{\bar{k}} + (1 - \alpha) \times c_e);$
    $CustoDS(e) \leftarrow c_e;$
  $DijkstraLider(CustoDL, \bar{k});$
  $DijkstraSeguidor(CustoDS, \bar{k});$
  $SalvaSol \leftarrow [y, x];$
  para $numIterDP$ *de* $1 \ldots MaxIterDP$ **faça**
    enquanto $K \neq \emptyset$ **faça**
      $\overline{K} \leftarrow ListaCandidato(K, \gamma);$
      $k' \leftarrow Random(\overline{K});$
      para $e = (i, j)$ *de* $1 \ldots |E|$ **faça**
        $CustoDL(e, k') \leftarrow (f_e \times (1 - y_e)) + (\alpha \times g_{ij}^{k'} + (1 - \alpha) \times c_e);$
      $DijkstraLider(CustoDL, k');$
      $DijkstraSeguidor(CustoDS, k');$
      $K \leftarrow K \backslash \{k'\};$
    $s \leftarrow [y, x];$
    $FecharArestas(s);$
    se $Custo(s) < MaxCusto$ **então**
      $s_{best} \leftarrow s;$
      $MaxCusto \leftarrow Custo(s_{best});$
    $K \leftarrow K_G;$
    $[y, x] \leftarrow SalvaSol;$
    $\alpha \leftarrow \alpha - \frac{1}{MaxIterDP};$
retorna s_{best}

O desacoplamento parcial consiste, na utilização do algoritmo de *Dijkstra* para o problema de caminho mínimo. A função OrdenarK, ordena de forma decrescente os produtos por suas quantidades q^k. Os procedimentos DijkstraLider e DijkstraSeguidor, resolvem sequencialmente o problema de construção da rede, seguido do problema de caminho mínimo para cada mercadoria $k \in K$, de modo que ao final do procedimento, todas as mercadorias tenham sido transportadas de sua origem até seu destino. Os CustoDL e CustoDS são os custos do procedimento DijkstraLider e DijkstraSeguidor, respectivamente. A função Random retorna aleatoriamente um elemento do conjunto passado como parâmetro. Para realizar a escolha da ordem de inserção dos $|K| - 1$ produtos na solução, utiliza-se uma lista de candidatos, composta por um subconjunto dos produtos ainda não roteados, cuja quantidade seja maior ou igual $\gamma\%$ da maior quantidade do produto ainda não atendido. Por fim, a função FecharArestas, fecha todas as arestas que ao final do procedimento estiverem abertas e não possuírem fluxo.

4.2 Busca Local

Após se obter uma solução viável através da heurística de desacoplamento parcial, inicia-se a segunda fase caracterizada pela aplicação da busca local, com o desígnio de melhorar a solução encontrada na fase de construção. Dada uma solução s, a busca local caracteriza-se pela construção de uma vizinhança $N(s)$ de soluções, obtidas através da aplicação de um movimento pré-definido sobre a solução atual A definição da vizinhança $N(s)$ depende do problema que deseja-se resolver. Neste caso específico, a vizinhança foi definida da seguinte forma:

$$N(s) = \{s' |\ s' \text{ possui o mesmo caminho para pelo menos } 80\% \text{ dos produtos que } s\}.$$

Quando se diz ao mínimo 80%, lê-se $\lceil 0.8|K| \rceil$, uma vez que tal condição garante que a quantidade de produtos que terão seus caminhos destruídos seja inteira.

Visto que a exploração do espaço de busca para a estrutura de vizinhança é inviável computacionalmente ($\frac{|K|!}{(|K|-\lceil 0.8|K| \rceil)!}$), utiliza-se um critério de aceitação *first improvement*, onde ao se encontrar uma solução de qualidade superior a atualmente explorada, substitui-se a solução atual analisada pela de melhor qualidade. O parâmetro, $numIterER$, foi definido após diversos testes, sendo o valor 10 o que apresentou os melhores resultados. O procedimento de busca local, denominado *Ejection Route*, é descrito no Algoritmo 2.

Algoritmo 2: Ejection Route

Entrada: $N(s)$, s_{best}
Dados: $numIterER \leftarrow 0$;
inicio
  enquanto $numIterER < MaxIterBL$ **faça**
    $s' \leftarrow N(s_{best})$;
    se $Custo(s') < Custo(s_{best})$ **então**
      $s_{best} \leftarrow s'$;
      $numIterER \leftarrow 0$;
    senão
      $numIterER \leftarrow numIter + 1$;
retorna s_{best}

Para realizar o procedimento de busca local, $\lfloor 0.2|K| \rfloor$ caminhos são destruídos de forma aleatória e em seguida os caminhos são reconstruídos utilizando o algoritmo de Desacoplamento Parcial. Caso a solução encontrada possua qualidade superior a melhor solução atual, esta é atualizada.

4.3 GRASP

A metaheurística GRASP (Resende e Ribeiro (2003) tem como diferencial para outros métodos a geração de solução inicial que se baseia em três premissas básicas: gulosa (*Greedy*), aleatória (*Randomized*) e adaptativa (*Adaptive*). Enquanto outros algoritmos como a busca tabu e os algoritmos genéticos valem-se de estratégias com grande ênfase na busca local, o GRASP foca seus maiores esforços na geração de uma solução inicial de melhor qualidade para utilizar a busca local apenas para pequenas melhorias. O GRASP é uma metaheurística *multi-start*, o que significa que a cada iteração ele executa sua componente construtiva e sua busca local. As componentes principais do GRASP-DE (Desacoplamento Parcial + Ejection Route) podem ser observados no Algoritmo 3

A componente construtiva do GRASP aqui apresentada, consiste na utilização do Desacoplamento Parcial, onde a construção gera uma solução inicial para o problema através da utilização

Algoritmo 3: GRASP-DE

Entrada: E, K, c, f, g, q, γ, $N(s)$
inicio
  para $numIterGR$ *de* $1 \ldots MaxIterGR$ **faça**
    $s_{best} \leftarrow$ DesacoplamentoParcial$(E, K, c, f, g, q, \gamma)$;
    EjectionRoute$(N(s), s_{best})$;
  retorna s_{best}

do algoritmo de *Dijkstra*. Para construir a solução, aplica-se o algoritmo no intuito de estabelecer um caminho para cada produto entre sua origem $o(k)$, até seu destino $d(k)$. Visando encontrar uma solução de maior qualidade, executa-se $MaxIterDP$ construções e por fim é considerada a solução que proporcionar o melhor resultado. Para compor a etapa de busca local, utiliza-se o procedimento *Ejection Route* para refinar a melhor solução. Este procedimentos são executados sequencialmente $MaxIterGR$ vezes.

5 Resultados Computacionais

Nesta Seção, serão apresentados os resultados computacionais obtidos pelo algoritmo GRASP-DE para o PPR-ROU. O algoritmo foi codificado em Xpress Mosel, através da ferramenta *FICO Xpress Optimization Suite*, em um computador com sistema operacional Windows 7 Professional e processador *Intel (R) Core TM 2 CPU 6400 @ 2.13GHz* com 2GB de memória *RAM*.

Para a bateria de testes foram utilizadas as redes geradas e utilizadas por Mauttone, Labbe e Figueiredo (2008). As redes utilizadas estão agrupadas em função do seu número de nós $(10, 20, 30)$, seguido pela densidade do grafo $(0.3, 0.5, 0.8)$ e por último pela quantidade de produtos distintos a serem transportados. Após diversos testes, os parâmetros $MaxIterDP$, $MaxIterBL$, $MaxIterGR$, γ foram definidos, respectivamente, como 10, 10, 50 e 25.

A Tabela 1 apresenta a comparação entre o algoritmo proposto e a Busca Tabu apresentada por Mauttone, Labbe e Figueiredo (2008), para as 5 instâncias publicadas em seu trabalho. Já a Tabela 2 apresenta os resultados do GRASP-DE para outras instâncias geradas por Mauttone, Labbe e Figueiredo (2008). Para as demais instâncias, os autores não divulgaram nenhum resultado. Nestas tabelas também são apresentadas as comparações entre a qualidade da solução encontrada pelo GRASP-DE, com as soluções ótimas obtidas através do modelo matemático em um nível.

As colunas S*, S, T, GAP, MS, MT, DPS, DPT, BS e BT, representam respectivamente a solução ótima da rede, a solução e o tempo obtidos pela Busca Tabu de Mauttone, Labbe e Figueiredo (2008), o gap em relação a solução ótima, a média das soluções e dos tempos obtidos pelo GRASP-DE, seguidos de seus respectivos desvio padrão e por último, mas não menos importante, a melhor solução e o melhor tempo obtidos pelo GRASP-DE. É importante ressaltar que caso o valor da solução esteja em negrito, a solução ótima foi alcançada.

	Exato	Tabu Search MLF			GRASP						
Instance	S*	S	T	GAP	MS	MT	DPS	DPT	BS	BT	GAP
30-0.800000-30-001	4830	4927	1110	0,020	4871	332,144	0	9,227	4871	330,908	0,008
30-0.800000-30-002	6989	7322	93	0,048	7122,2	328,295	182,39	4,115	**6989**	325,357	0,000
30-0.800000-30-003	7746	8142	565	0,051	8124	337,191	16,43	33,634	8112	321,838	0,047
30-0.800000-30-004	8384	8828	1287	0,053	8384	318,062	0	26,091	**8384**	338,249	0,000
30-0.800000-30-005	7428	7502	794	0,010	7442,8	321,434	33,09	17,889	**7428**	344,367	0,000

Tabela 1: Comparação Busca Tabu X GRASP.

	Exato	GRASP						
Instance	S*	MS	MT	DPS	DPT	BS	BT	GAP
20-0.300000-10-001	5978	6513.58	15.65	136.48	0.34	6411	15.5	0.07
20-0.300000-10-002	10469	10813.3	16.57	185.69	0.58	10664	16.38	0.02
20-0.300000-10-003	7020	7286.4	15.99	132.14	0.34	7200	15.67	0.03
20-0.300000-10-004	5484	5754.74	15.84	116.73	0.33	5598	15.71	0.02
20-0.300000-10-005	7932	8322	16.04	0	0.4	8322	16.01	0.05
20-0.300000-20-001	9488	**9488**	32.1	0	1.36	**9488**	31.84	0
20-0.300000-20-002	11521	11699.86	31.64	201.31	0.91	11607	30.94	0.01
20-0.300000-20-003	8270	8670.82	32.57	222.9	0.72	8568	32.44	0.04
20-0.300000-20-004	11901	12320.58	31.94	300.06	1.07	11985	31.62	0.01
20-0.300000-20-005	9656	10379.38	32.12	178.59	0.46	10297	31.93	0.07
20-0.300000-30-001	12510	13244	49.28	0	0.76	13244	48.69	0.06
20-0.300000-30-002	14216	14854.9	49.81	364.81	1.76	14737	49.41	0.04
20-0.300000-30-003	13393	14687.52	48.18	577.28	1.41	14629	47.79	0.09
20-0.300000-30-004	14452	15420.97	48.62	327.77	0.63	15329	48.32	0.06
20-0.300000-30-005	11419	12599	51.32	0	1.08	12599	51.02	0.1
20-0.500000-10-001	4784	**4784**	21.56	0	0.83	**4784**	21.43	0
20-0.500000-10-002	7689	**7689**	21.86	0	0.57	**7689**	21.73	0
20-0.500000-10-003	6184	**6184**	22.68	0	0.47	**6184**	22.45	0
20-0.500000-10-004	5189	5532.91	22.41	95.2	0.29	5489	22.19	0.06
20-0.500000-10-005	6051	6233.72	22.78	80.47	0.59	6172	22.74	0.02
20-0.500000-20-001	8816	9964	46.5	0	0.95	9964	45.85	0.13
20-0.500000-20-002	8584	8721.34	47.45	150.45	1.83	**8584**	46.89	0
20-0.500000-20-003	7560	8354.83	45.72	214.84	0.92	8305	44.65	0.1
20-0.500000-20-004	7634	7750.74	45.28	100.06	0.84	7674	44.92	0.01
20-0.500000-20-005	8270	8636	44.86	0	1.12	8636	44.77	0.04
20-0.500000-30-001	10156	12600	67.99	0	2.34	12600	67.99	0.24
20-0.500000-30-002	11403	12932	68.66	0	1.91	12932	68.66	0.13
20-0.500000-30-003	11600	13021.4	73.29	334.74	1.35	12867	71.57	0.11
20-0.500000-30-004	11785	12333.56	70.88	317.15	1.32	12260	68.82	0.04
20-0.500000-30-005	9559	10989	69.47	0	1.82	10989	69.33	0.15
20-0.800000-10-001	3947	4120.8	34.32	105.35	0.9	4040	34.32	0.02
20-0.800000-10-002	3743	3915	34.51	0	1.13	3915	34.02	0.05
20-0.800000-10-003	3412	3480.24	34.81	74.75	0.58	**3412**	34.39	0
20-0.800000-10-004	4086	4209	35.27	0	0.8	4209	34.99	0.03
20-0.800000-10-005	4498	4542.98	35.64	97.51	0.77	**4498**	35.28	0
20-0.800000-20-001	5796	6909	70.88	0	1.73	6909	69.22	0.19
20-0.800000-20-002	7037	7635.54	71.48	187.03	1.02	7590	70.34	0.08
20-0.800000-20-003	4596	6251.89	69	89.48	1.84	5422	68.18	0.18
20-0.800000-20-004	4851	5187	70.26	69.01	2.45	5250	69.98	0.08
20-0.800000-20-005	6086	6855.53	72.13	86.23	1.93	6267	71.42	0.03
20-0.800000-30-001	7769	9425	105.01	0	2.17	9425	101.23	0.21
20-0.800000-30-002	7681	8735.33	110.77	126.42	1.98	8666	109.89	0.13
20-0.800000-30-003	5144	5947.89	107.3	201.43	2.67	5889	106.24	0.14
20-0.800000-30-004	7188	8768.08	104.77	177.53	3.74	8630	104.56	0.2
20-0.800000-30-005	7374	8175.16	108.08	127.82	1.46	7942	108.08	0.08

Tabela 2: Resultado demais instâncias.

6 Conclusão e Trabalhos Futuros

Tendo em vista os resultados mostrados, pode-se concluir que o método se comportou de forma eficiente, em geral, para as instâncias testadas. Ao se comparar os resultados entre o GRASP-DE e a Busca Tabu apresentada por Mauttone, Labbe e Figueiredo (2008), é possível observar que o algoritmo desenvolvido neste trabalho proporcionou soluções de qualidade superior em um tempo computacional inferior para 4 das 5 instâncias comparadas. Para a instância 30-0.800000-30-002, foi encontrada solução de maior qualidade, no entanto o algoritmo apresentado levou mais tempo do que o de Mauttone, Labbe e Figueiredo (2008).

Uma vez que este método apresentou resultados competitivos em relação a literatura, pretende-se futuramente agregar novas estruturas de vizinhança tal como *Cycle-based Neighborhood* (Ghamlouche, Crainic, e Gendreau (2003)). Além disso, pode-se também adicionar um procedimento de reconexão por caminhos (Ghamlouche, Crainic, e Gendreau (2004)) de forma a tentar melhorar a qualidade das soluções encontradas.

Referências

[1] Ahuja, R. K., Magnanti, T. L. e Orlin, J. B., Network flows, Prentice-Hall, New Jersey, 1993.

[2] Amandi, E., Bruglieri, M., Fortz,B. , On the hazmat transport network design problem, Network Optimization, UOFinger, 2011.

[3] Bazaraa, M. S., Jarvis, J. e Sherali, H. D. (2004), Linear Programming and Network Flows, 3nd edn,Wiley-Interscience.

[4] Billheimer, J. W. and Gray, P. (1973), Network design with fixed and variable cost elements, Transportation Science 7, 49-74.

[5] Buriol, L., Resende, M. e Thorup, M. (2008), Speeding up dynamic shortest-path algorithms, INFORMS Journal on Computing 20(2), 191-204.

[6] Boesch, F., Large-scale networks: theory and design ,IEEE Press, 1975.

[7] Boyce, D. E. (1979), Transportation Research B 13B(1), 1-3.

[8] Colson, B., Marcotte, P. and Savard, G. (2005), Bilevel programming: A survey, 4OR: A Quarterly Journal of Operations Research 3(2), 87-107.

[9] de Giovanni, L. (2004), The internet protocol network design problem with reliability and routing constraints, PhD thesis, Politecnico di Torino.

[10] Erkut, E. e Gzara, F. (2008), Solving the hazmat transport network design problem. Computers and Operations Research, 35(7), 2234-2247.

[11] Erkut, E., Tjandra, S. A., e Verter, V. (2007), Hazardous Materials Transportation. Handbooks in Operations Research and Management Science (Vol. 14, pp. 539Ű621).

[12] Ghamlouche, I., Crainic, T. G. e Gendreau, M. (2003), Cycle-Based Neighbourhoods for Fixed-Charge Capacitated Multicommodity Network Design, Operations Research, 51(4), 655-667.

[13] Ghamlouche, I., Crainic, T. G. e Gendreau, M. (2004), Path Relinking, Cycle-Based Neighbourhoods and Capacitated Multicommodity Network Design, Annals of Operations Research, 131(1-4), 109-133.

[14] Graves, S. e Lamar, B. (1983), An integer programming procedure for assembly system design problems, Operations Research 31, 522-545.

[15] Holmberg, K. and Yuan, D. (2004), Optimization of internet protocol network design and routing, Networks 43(1), 39-53.

[16] Johnson, D. S., Lenstra J. K. e Rinnooy Kan, A. H. G. (1978), The complexity of the network design problem, Networks 8, 279-285.

[17] Kara, B. Y. e Verter, V. (2004), Designing a road network for hazardous materials transportation, Transportation Science 38(2), 188-196.

[18] Krimenia, J. e Gershwin, S. B. (1979), Network flow optimization in flexible manufacturing systems, Proceedings of 1978 IEEE Conference on Decision and Control, 633-639.

[19] Mandl, C. (1981), A survey of mathematical optimization models and algorithms for designing and extending irrigation and wastewater networks, Water Resource Research 17(1).

[20] Magnanti, T. L. (1981), Combinatorial optimization and vehicle fleet planning: Perspectives and prospects, Networks 11, 179-214.

[21] Magnanti, T. L. e Wong, R. T. (1984), Network design and transportation planning: Models and algorithms, Transportation Science 18(1), 1-55.

[22] Magnanti, T. L. e Wong, R. T. (1984a), Network design and transportation planning: Models and algorithms, Transportation Science 18, 1-56.

[23] Mauttone, A., Labbe, M. e Figueiredo, R. (2008), A Tabu Search approach to solve a network design problem with user-optimal flows, VI ALIO/EURO Conference on Combinatorial Optimization, Buenos Aires.

[24] Resende, M.G.C e Ribeiro, C.C., Greedy randomized adaptive search procedures, Handbook of Metaheuristics (F. Glover e G. Kochenberger, eds.), 2003, 219-249.

[25] Simpson, R., Scheduling and routing models for airline systems, MIT Flight Transportation Laboratory Report, 1969.

[26] Wong, R. T., Accelerating Benders decomposition for network design, Doctoral Dissertation, Department of Electrical Engineering and Computer Science, MIT, 1978.

[27] Wong, R. T. (1980), Worst-case analysis of network design problem heuristics, SIAM Journal on Algebraic Discrete Methods 141-163.