UM ALGORITMO DE PROGRAMAÇÃO DINÂMICA PARA PARTIÇÕES EM GRAFOS: APLICAÇÃO AO PROBLEMA DOS ANÉIS SONET

Thiago H. F. de Oliveira[1], Christophe Duhamel[2], Dario J. Aloise[1], Andréa Cynthia Santos[3]

[1] Universidade do Estado do Rio Grande do Norte, UERN
Rua Almino Afonso, 478, CEP 59.610-210, Mossoró, RN, Brasil.

[2] LIMOS, Université Blaise Pascal
Campus des Cézeaux, 63173 Aubière CEDEX, France.

[3] ICD-LOSI, Université de Technologie de Troyes
12, rue Marie Curie, CS 42060, 10004 Troyes CEDEX, France.
{thiagohfo,djaloise}@gmail.com, christophe.duhamel@isima.fr, andrea.duhamel@utt.fr

RESUMO

Dado um grafo $G = (V, E)$, o problema de partições em grafos consiste em dividir V em conjuntos disjuntos. Neste trabalho, propomos um método sofisticado para minimizar o número de partições em grafos, cujas partições possuem restrições de capacidade. O procedimento proposto utiliza programação dinâmica e é semelhante ao *split* proposto por Beasley (1983), comumente aplicado a problemas de roteamento de veículos. Para testar a validade do método, aplicamos a idéia ao problema dos anéis SONET (*Synchronous Optical NETwork*), referenciado como SRAP (*Sonet Ring Assignment Problem*). SRAP é um problema NP-difícil e encontra aplicações em topologias de redes de fibras óticas. O método baseado em programação dinâmica é utilizado para gerar partições a partir de sequências aleatórias e também em um algoritmo baseado em BRKGA (*Biased Random Key Genetic Algorithm*), sendo o mesmo empregado como decodificação de chaves aleatórias. Resultados utilizando a decodagem baseada em programação dinâmica são promissores.
PALAVRAS CHAVE. Anéis SONETS, Programação dinâmica, Heurísticas, BRKGA, Projeto de redes.

Áreas Principais: Heurísticas, Telecomunicações, Projeto de redes.

ABSTRACT

Given a graph $G = (V, E)$, the set partitioning problem consists of defining disjoints partitions. In this work, we propose a sophisticated method to optimize the number of partitions in a given graph, considering that every partition are associated with a capacity constraint. The proposed procedure uses dynamic programming and is similar to the *split*, introduced by Beasley (1983). The method is usually applied to vehicle routing problems. Here, the idea is applied to the SONET (*Synchronous Optical NETwork*) rings problem, referred as SRAP (*Sonet Ring Assignment Problem*). SRAP is a NP-hard problem and finds applications for designing optical fiber networks. Two strategies have been tested using the dynamic programming based procedure: in the first strategy, a number of random sequences are generated and partitioned; in the second one, the procedure has been used as a decoder in a BRKGA (*Biased Random Key Genetic Algorithm*) heuristic. Results using the second proposed strategy are very promising.
KEY WORDS. SONET rings, Dynamic programming, Heuristics, BRKGA, Network design.

Main areas: Heuristics, Telecommunications, Network design.

1 Introdução

Dado um grafo $G = (V, E)$, onde V é o conjunto de n vértices e E é o conjunto de m arestas, o problema de partições em grafos consiste em dividir V em conjuntos disjuntos $V = V_1 \cup V_2 \cup \cdots \cup V_l$, de Souza (2002). Neste trabalho, propomos um método sofisticado para minimizar o número de partições em grafos, cujas partições possuem restrições de capacidade. O procedimento proposto utiliza programação dinâmica e é semelhante ao *split* proposto por Beasley (1983), comumente aplicado a problemas de roteamento de veículos, Afsar *et al*. (2014); Prins *et al*. (2014).

Para testar a validade do método, aplicamos a idéia ao problema dos anéis SONET (*Synchronous Optical NETwork*), referenciado como SRAP (*Sonet Ring Assignment Problem*). O SRAP é definido em um grafo não-orientado $G = (V, E)$, definido acima. Além disto, demandas d_{ij} são associadas a cada aresta $[i, j] \in E$. Um anel é formado por um conjunto de nós $i \in V$, cuja capacidade total $Q \in \mathbb{N}^*$ dos anéis é limitada. O problema SRAP consiste em definir partições de V (cada partição corresponde a um anel), respeitando a capacidade Q dos anéis, com o objetivo de minimizar a quantidade de anéis. Após a construção das partições, um anel central, federa os outros anéis da topologia, sendo composto de um vértice pertencente a cada anel. O anel federal também deve respeitar a capacidade Q.

SRAP é NP-difícil Goldschmidt *et al*. (2003) e encontra aplicações em topologias de redes de fibras óticas, onde o anel federal é composto de equipamentos do tipo DCS *(Digital Cross-connect System)*. Devido ao alto custo deste tipo de equipamento, otimizar o número de partições implica minimizar a quantidade de equipamentos DCS, já que existirá um e somente um DCS alocado por partição (anel). Além disto, as demandas d_{ij} correspondem às demandas de tráfego na rede de fibra ótica.

O problema SRAP está relacionado com alguns problemas clássicos da literatura, como o problema de partições em grafos, citado anteriormente, e o problema do *bin packing*. O problema do *bin packing off-line* objetiva alocar em caixas (*bins*) um número pré-definido de objetos com pesos associados, de modo a minimizar a quantidade de caixas, respeitando a capacidade das caixas, Alvim *et al*. (2004). O SRAP difere do *bin packing*, primeiramente porque não tem pesos associados aos vértices, os pesos são associados às arestas e representam as demandas de tráfego. Segundo porque não existe uma relação entre as caixas, como no SRAP, onde o anel federal deve se conectar a todos os outros.

O problema SRAP foi escolhido devido a sua complexidade e interesse. O procedimento baseado em programação dinâmica é empregado para gerar soluções para o SRAP de duas maneiras: na primeira estratégia, simplemente gera-se uma quantidade pré-definida de sequências aleatórias que são particionadas usando o procedimento proposto; na segunda estratégia, o procedimento é utilizado como um decodificador em um algoritmo genético com chaves aleatórias tendenciosas (BRKGA, do inglês *Biased Random Key Genetic Algorithms*), Gonçalves e Resende (2011); Resende (2012).

Este trabalho está organizado da seguinte forma: uma revisão bibliográfica e a definição formal do problema SRAP são apresentadas na Seção 2. Em seguida, o procedimento proposto é descrito na Seção 3, assim como sua aplicação ao SRAP e integração ao BRKGA. Finalmente, resultados preliminares e considerações finais são apresentados respectivamente nas Seções 4 e 5.

2 O problema SRAP

Diversos trabalhos na literatura foram desenvolvidos para o SRAP, sendo o trabalho de Goldschmidt *et al*. (2003) um dos pioneiros. Os autores mostraram a complexidade do problema e apresentaram uma formulação para o SRAP não linear que foi linearizada e outra para uma variação do problema, chamada de *k*-SRAP, onde *k* é uma restrição de capacidade máxima dos anéis. Heurísticas chamadas de *edge-based heuristic*, *cut-based heuristic*, e *node-based heuristic* também foram propostas. Essas heurísticas usam inicialmente a idéia de criar V anéis e depois

uní-los de diferentes maneiras para reduzir o número de anéis. Reformulações para o SRAP são apresentadas em Macambira *et al*. (2006), cujos modelos são baseados no problema de partições em grafos e problema da mochila. Um algoritmo do tipo *Branch-and-Price* é proposto, com duas estratégias para geração de colunas: a primeira introduz uma coluna por vez e a segunda utiliza múltiplas colunas. As soluções primais são geradas utilizando a metaheurística *Greedy Randomized Adaptive Search Procedure* (GRASP). Resultados são apresentados para quatro grupos de instâncias, referenciadas como C1, C2, C3 e C4, contendo respectivamente 157, 230, 12 e 93 instâncias, as quais 111, 225, 3 e 93 possuem soluções viáveis.

Heurísticas e metaheurísticas também são encontradas na literatura para o SRAP, Aringhieri e Dell'Amico (2005); Bastos *et al*. (2005); Bernardino *et al*. (2012). Metaheurísticas do tipo *Tabu Search* e *Scatter Search* são propostas por Aringhieri e Dell'Amico (2005). Diversos mecanismos sofisticados de intensificação (baseado no *Path relinking*), e diversificação (múltiplas vizinhanças) são empregados. Procedimentos heurísticos simples para construção de soluções iniciais são apresentados, tais como o *Best-Fit Decreasing* e *Next-Fit* inspirados dos procedimentos para o *bin packing*, o *Clique Best-Fit* e *Cycle Best Fit* baseados na construção de cliques. Procedimentos baseados em algoritmos genéticos são propostos por Bastos *et al*. (2005). Uma variação no algoritmo genético utilizando *path relinking* também é proposta. A população inicial foi gerada utilizando heurísticas construtivas aleatorizadas, chamadas de *Random Node-Base* e *Relative Neighborhood*. Heurísticas baseadas em um método de reprodução de abelhas *Honey Bees Mating Optimization* e hibridizações foram propostas por Bernardino *et al*. (2012).

Diversas formulações matemáticas compactas e estendidas foram propostas para o SRAP na literatura, Goldschmidt *et al*. (2003); Macambira *et al*. (2006). Até onde sabemos, a formulação compacta de melhor desempenho foi proposta por Macambira *et al*. (2006). A formulação (1) - (12) utiliza o grafo $G = (V, E)$ previamente definido, demandas de tráfego $d_{ij} > 0$ entre cada par de vértice $i, j \in V$. O tráfego total de demandas é dado por $D = \sum_{[i,j]\in E} d_{ij}$. $Q \in \mathbb{N}^*$ corresponde a capacidade total (banda passante) dos anéis locais e federal. O modelo matemático utiliza variáveis y^r que determinam se o anel r é usado ($y^r = 1$), ou não ($y^r = 0$) e as variáveis x_i^r estabelecem se um vértice i é alocado no anel r (x_i^r=1), ou não ($x_i^r = 0$). Além disto, variáveis de decisão auxiliares z_{ij}^r indicam se pelo menos uma das extremidades da aresta $[i, j]$ pertence ao anel i (z_{ij}^r=1), ou não (z_{ij}^r=0).

$$\max \sum_{r=1}^{n} \sum_{i\in V} n x_i^r - \sum_{r=1}^{n} y^r \tag{1}$$

$$\sum_{[i,j]\in E} d_{ij} z_{ij}^r \leqslant Q \quad \forall i = 1 \ldots n \tag{2}$$

$$\sum_{r=1}^{n} \sum_{[i,j]\in E} d_{ij} z_{ij}^r \leqslant D + Q \tag{3}$$

$$\sum_{r=1}^{n} x_i^r \leqslant 1 \quad \forall i \in V \tag{4}$$

$$x_i^r \leqslant y^r \quad \forall i \in V, \forall r = 1 \ldots n \tag{5}$$

$$z_{ij}^r \geqslant x_i^r \quad \forall [i, j] \in E, \forall r = 1 \ldots n \tag{6}$$

$$z_{ij}^r \geqslant x_j^r \quad \forall [i, j] \in E, \forall r = 1 \ldots n \tag{7}$$

$$z_{ij}^r \leqslant x_i^r + x_j^r \quad \forall [i, j] \in E, \forall r = 1 \ldots n \tag{8}$$

$$y^r \leqslant y^{r-1} \quad \forall r = 2 \ldots n \tag{9}$$

$$x_i^r \in \{0,1\} \quad \forall i \in V, \forall r = 1 \dots n \tag{10}$$

$$y^r \in \{0,1\} \quad \forall r = 1 \dots n \tag{11}$$

$$z_{ij}^r \in \{0,1\} \quad \forall [i,j] \in E, \forall r = 1 \dots n \tag{12}$$

A função objetivo (1) maximiza o número de nós incluídos nos anéis e minimiza a quantidade de anéis. As restrições (2) e (3) asseguram, respectivamente, o respeito da capacidade dos anéis locais e federal. As inequações (4) garantem que cada vértice seja alocado a no máximo um anel, enquanto que as restrições (5) evitam que vértices sejam alocados a anéis que não existem. Variáveis x e z são conectadas através das restrições (6), (7) e (8). As inequações (9) reduzem o fenômeno de simetria das variáveis y, evitando que um anel seja aberto, se seu predecessor encontra-se fechado. A definição das variáveis é realizada de (10) a (12).

3 Procedimento por programação dinâmica e sua aplicação ao SRAP

A idéia utilizada neste trabalho é inspirada no princípio "*route first cluster second*" proposto por Beasley (1983) para o problema de roteamento de veículos (VRP, do inglês *Vehicle Routing Problem*). Esse procedimento consiste em gerar uma solução viável para o VRP a partir de uma solução para o caixeiro viajante (TSP, do inglês *Travelling Salesman Problem*). Considerando o grafo $G' = (V', A)$, onde V' é um conjunto de vértices e A é um conjunto de arcos, com custos c_{ij} associados a cada aresta $(i,j) \in A$, o TSP consiste em definir uma rota que sai de um vértice $i \in V$, passa por todos os nós $j \in V \backslash i$ uma única vez e retorna a i, de modo a minimizar o custo total do trajeto. Um algoritmo polinomial é proposto para definir os pontos de corte na solução do TSP de modo a torná-la viável para o VRP. O aspecto mais interessante é que o algoritmo é capaz de determinar os pontos de cortes ótimos de modo a obter um ótimo local para o VRP, considerando a ordem em que os nós aparecem na solução do TSP. Assim, neste trabalho, utiliza-se este princípio para determinar soluções para o SRAP, considerando a capacidade dos anéis e a relação dos anéis com o anel federal.

Seja o grafo auxiliar $H = (N, A)$ obtido a partir de uma sequência S dos vértices de V. Assim, N contém $n+1$ pontos de corte de uma sequência S. O arco $(i,j) \in A$ corresponde a uma subsequência viável de S entre as posições i e j. O problema consiste em calcular um caminho válido induzido por H, do nó 0 até o nó n, de modo a minimizar a quantidade de arcos utilizados. Um caminho de 0 até n é válido se repeita a restrição de capacidade dos anéis e do anel federal. O problema torna-se um problema de caminho mais curtos com restrição de capacidade. O grafo H é acíclico. Assim, usar um algoritmo de programação dinâmica que cálcula rótulos permete encontrar uma solução ótima, se existe uma, para a sequência inicial fornecida.

Quando um vértice é alocado a um anel no problema SRAP, referencia-se aqui como anel gerado ou aberto. Um rótulo l corresponde a um caminho parcial de 0 até um nó j. Sendo definido por $l = (p_l, n_l, t_l)$, onde p_l é o início do arco (p_l, j) que gerou l, n_l é a quantidade de anéis já abertos e t_l é o tráfego inter-anéis dos anéis abertos. Cada nó $i \in H$ tem uma lista L_i de rótulos. Inicialmente, $L_i = \emptyset$ para todos os nós $i > 0$. O nó 0 recebe o rótulo inicial $l_0 = (0,0,0)$. Os rótulos são propagados sob os arcos. Dado um rótulo $l = (p_l, n_l, t_l) \in L_i$ e um arco $(i,j) \in A$; a propagação ocorre da seguinte forma: um rótulo $l' = (i, n_l + 1, t_l + t(i,j))$ é gerado no nó j. O valor $t(i,j)$ corresponde ao tráfego da subsequência $S_i \dots S_{j-1}$ para os nós posteriores $S_j \dots S_n$. Se $t_l + t(i,j) > Q$, a capacidade do anel federal é violada e o rótulo é descartado. O rótulo l' entra na lista L_j, se não tem um rótulo $l'' \in L_j$ que domina l' ($l'' \prec l'$). A condição para que um rótulo seja dominado é dada na Equação (13):

$$l_1 = (p_1, n_1, t_1) \prec l_2 = (p_2, n_2, t_2) \quad \Leftrightarrow \quad (n_1 \leq n_2) \text{ e } (t_1 < t_2) \tag{13}$$

As listas L_i são classificadas em ordem crescente de acordo com o valor n_l dos rótulos $l \in L_i$. Um pseudo-código para o algoritmo de cálculo de partições a partir de uma sequência fornecida é apresentado no Algoritmo (1).

Algoritmo 1 Pseudo-código para determinar anéis para o SRAP.

Entrada: grafo $H = (N, A)$, sequência S
Saída: número mínimo de anéis associados a S

// Inicialização
$L_i \leftarrow \{\}, \forall i = 1 \dots n$
$L_0 \leftarrow \{(0, 0, 0)\}$
// Pontos de corte
para $i = 0 \dots n - 1$ **faça**
  se $L_i \neq \emptyset$ **então**
    // Percorrer os arcos $(i, j) \in A$
    para $j = i + 1 \dots n$ **faça**
      seja $S' = S_{i+1} \dots S_j$ o novo anel
      calcular $I(S')$ e $E(S')$, tráfegos interno e externo de S'
      se $I(S') + E(S') \leq Q$ **então**
        // Propagação de rótulos de L_i
        para todo $l = (p_l, n_l, t_l) \in L_i$ **faça**
          calcular $E^+(S')$, tráfego externo de S' com os nós $S_{j+1} \dots S_n$
          se $t_l + E^+(S') \leq Q$ **então**
            $l' = (i, n_l + 1, t_l + E^+(S'))$
            atualizar L_j com l'
          fim se
        fim para
      fim se
    fim para
  fim se
fim para
se $L_n = \emptyset$ **então**
  retornar n
senão
  retornar $\arg\min_{l=(p_l,n_l,t_l)\in L_n} \{n_l\}$
fim se

Este algoritmo é polinomial uma vez que cada arco de H é examinado uma única vez e a avaliação de $I(S')$, $E(S')$ e $E^+(S')$ tem complexidade assintótica de pior caso $O(m)$.

3.1 BRKGA aplicado ao SRAP

Algoritmos genéticos baseados em chaves aleatórias (RKGA, do inglês *Random-Key Genetic Algorithm*) Bean (1994), introduzem uma representação de indivíduos (cromossomos) independentes do problema. As chamadas chaves aleatórias são sequências de números reais aleatórios no intervalo $[0, 1]$. Assim, um indivíduo (cromossomo) é representado em um vetor de chaves aleatórias. A interpretação das chaves para um problema específico requer um procedimento para transformar o vetor de chaves aleatórias em soluções viáveis. A geração de novos indivíduos a partir de operadores genéticos (por exemplo, cruzamento) no RKGA é realizada a partir de uma seleção aleatória de dois indivíduos da população. O BRKGA, do inglês *Biased Random-Key Genetic Algorithm* Gonçalves e Resende (2011); Resende (2012), difere do RKGA basicamente porque a seleção obriga que um dos indivíduos no cruzamento pertença ao conjunto elite. Este método tem sido aplicado com sucesso em diversos problemas de otimização combinatória, Coco *et al*. (2013, 2014); Morán-Mirabal *et al*. (2014); Resende (2012).

Inicialmente, p vetores de chaves aleatórias são calculados para gerar a população inicial.

Em seguida, em cada iteração do BRKGA, a população é ordenada e separada em dois conjuntos, p_e indivíduos elite e $(p-p_e)$ indivíduos não elite. p_c novos indivíduos são obtidos a partir da aplicação de um operador de cruzamento. Ele é realizado entre um indivíduo pertencente ao conjunto elite e um outro pertencente ao conjunto não elite. A cada posição j do vetor de chaves, a probabilidade do novo indivíduo herdar o valor da chave j é igual a $(rhoe)$ (parâmetro a definir) do indivíduo elite. Consequentemente, a probabilidade de herdar o valor da chave do indivíduo não elite é $(1-rhoe)$. Os outros $p_m = (p-p_e-p_c)$ novos indivíduos (os mutantes) são gerados aleatoriamente para manter um nível de diversidade. A nova população recebe os p_e indivíduos elite et os $(p_c + p_m)$ novos indivíduos. Outra diferença entre RKGA e BRKGA é que a população é dividida em K grupos distintos (as ilhas) que evoluem independentemente. Comumente, três ilhas são utilizadas, Resende (2012). A cada T iterações, o conjunto completo de soluções elite das ilhas são permutados.

Para construir uma solução a partir de um vetor de chaves aleatórias é necessário definir uma função de decodagem. Inicialmente, o vetor de chaves aleatórias é colocado em ordem crescente, guardando o indíce (referência) de cada chave aleatória. Em seguida, as partições (anéis) são construídas, utilizando os vértices que satisfaçam a capacidade do anel. Assim, precisa-se calcular o tráfego interno e externo de cada anel. Dado um anel correspondente a um subconjunto S de vértices, o tráfego interno $I(S)$ é calculado como ilustrado na Equação (14):

$$I(S) = \sum_{[ij] \in E, i,j \in S, j>i} d_{ij} \tag{14}$$

O tráfego $E(S)$ de S para os outros anéis consome a capacidade do anel federal definido na Equação (15):

$$E(S) = \sum_{[ij] \in E, i \in E, j \notin S} d_{ij} \tag{15}$$

O consumo total do anel S é calculado de acordo com a Equação (16):

$$T(S) = I(S) + E(S) \tag{16}$$

A função de decodagem é baseada no procedimento por programação dinâmica descrito anteriormente. A idéia principal consiste em determinar os pontos de corte na sequência que correspondem aos anéis. Estes anéis devem ser viáveis em termos de capacidade. Da mesma forma, a capacidade do anel federal deve ser respeitada considerando o tráfego entre os anéis. O objetivo é então minimizar o número de pontos de corte.

4 Resultados preliminares

Os experimentos computacionais foram realizados em um processador Intel core i7 2.3 GHz com 8 GB de memória RAM, utilizando o sistema operacional Linux. O programa foi implementado em C++. As instâncias utilizadas são pertencentes aos conjuntos C1 e C2, ambas propostas em Goldschmidt *et al.* (2003). Duas estratégias foram utilizadas. Na primeira (referenciada aqui como MS), 1000 sequências aleatórias foram geradas. Em seguida, o procedimento de programação dinâmica foi aplicado. Na segunda, o procedimento proposto é utilizado como decodificação de chaves aleatórias de uma heurística baseada no BRKGA. Parâmetros sugeridos por Resende (2012) foram utilizados no BRKGA : 100 chromossomos, 50 gerações, $p_e = 20$, $p_m = 10$, $rhoe = 0.7$ et $K = 3$.

Resultados para uma amostragem de 48 e 11 instâncias pertencentes respectivamente aos grupos C1 e C2 são apresentadas nas Tabelas 1 e 2. Cada linha corresponde a uma instância. Para cada instância, são indicados o nome da instância, a quantidade total de vértices e o valor ótimo provenientes do trabalho Macambira *et al.* (2006). Os resultados obtidos utilizando as estratégias MS e BRKGA são apresentados nas colunas "Anéis" e "$t(s)$" que indicam, respectivamente, a

quantidade de anéis encontrados por cada estratégia e os tempos de processamento em segundos. Os valores destacados em negrito indicam que o método encontrou a solução ótima.

Tabela 1: Resultados para as instâncias C1

Instância	$\|V\|$	O^*	MS		BRKGA	
			Anéis	t(s)	Anéis	t(s)
GH_15_1	15	3	**3**	0,02	**3**	0,27
GH_15_2	15	3	**3**	0,01	**3**	0,15
GH_15_3	15	2	**2**	0,02	**2**	0,28
GH_15_6	15	2	**2**	0,02	**2**	0,31
GH_15_7	15	2	**2**	0,04	**2**	0,41
GH_15_8	15	3	**3**	0,00	15	0,05
GH_15_9	15	3	**3**	0,01	**3**	0,16
GH_15_10	15	2	**2**	0,03	**2**	0,36
GH_25_1	25	3	**3**	0,03	**3**	1,69
GH_25_2	25	2	3	0,06	**2**	2,00
GH_25_3	25	2	3	0,08	**2**	20,8
GH_25_4	25	3	4	0,03	**3**	10,20
GH_25_8	25	3	4	0,02	**3**	1,04
GH_25_9	25	3	25	0,02	**3**	0,88
GH_30_1	30	3	30	0,04	**3**	1,60
GH_30_10	30	3	**3**	0,07	**3**	3,36
GL_15_1	15	3	**3**	0,01	**3**	0,07
GL_15_4	15	3	**3**	0,00	**3**	0,09
GL_15_7	15	3	15	0,00	**3**	0,09
GL_15_9	15	3	**3**	0,01	**3**	0,21
GL_25_3	25	3	25	0,02	**3**	0,27
GL_25_7	25	3	25	0,02	**3**	1,46
RH_15_1	15	3	**3**	0,01	**3**	0,15
RH_15_3	15	*	15	0,01	**3**	0,05
RH_15_4	15	2	**2**	0,03	**2**	0,36
RH_15_5	15	3	**3**	0,01	**3**	0,20
RH_15_6	15	3	**3**	0,02	**3**	0,22
RH_15_7	15	2	**2**	0,03	**2**	0,35
RH_15_8	15	2	3	0,02	**2**	0,26
RH_15_9	15	3	**3**	0,02	**3**	0,26
RH_25_2	25	3	**3**	0,10	**3**	1,90
RH_25_3	25	3	**3**	0,06	**3**	1,63
RH_25_7	25	3	**3**	0,07	**3**	1,75
RH_25_9	25	2	**2**	0,02	**2**	3,69
RH_25_10	25	3	**3**	0,02	**3**	0,21
RH_30_1	30	3	**3**	0,17	**3**	5,28
RH_30_3	30	3	**3**	0,13	**3**	3,48
RH_30_4	30	3	**3**	0,08	**3**	3,08
RH_30_9	30	3	30	0,06	**3**	1,48
RL_15_1	15	3	**3**	0,01	**3**	0,15
RL_15_3	15	2	**2**	0,03	**2**	0,34
RL_15_4	15	3	**3**	0,03	**3**	0,34
RL_15_6	15	3	**3**	0,00	**3**	0,07
RL_15_8	15	3	**3**	0,01	**3**	0,21
RL_15_9	15	3	**3**	0,01	**3**	0,21
RL_15_10	15	3	15	0,01	**3**	0,06
RL_25_2	25	3	4	0,00	**3**	0,02
RL_25_8	25	3	**3**	0,01	**3**	0,13
RL_30_1	30	3	4	0,01	**3**	0,19

Tabela 2: Resultados para as instâncias C2

Instância	$\|V\|$	O^*	MS		BRKGA	
			Anéis	t(s)	Anéis	t(s)
GL_15_6.3	15	3	15	0	**3**	0,04
GL_15_6.10	15	3	15	0	**3**	0,06
RH_15_10.1	15	3	**3**	0	15	0,04
RH_15_10.2	15	3	**3**	0	15	0,05
RH_15_10.7	15	3	**3**	0	15	0,10
RL_15_2.4	15	3	15	0	**3**	0,04
RL_15_2.5	15	3	15	0	**3**	0,04
RL_15_5.1	15	3	15	0	**3**	0,04
RL_15_5.3	15	3	15	0	**3**	0,04
RL_15_5.7	15	3	15	0	**3**	0,04
RL_15_5.9	15	3	15	0	**3**	0,04

Considerando a amonstragem de instâncias para os grupos C1 e C2, o BRKGA achou soluções ótimas para 47 das 48 instâncias do conjunto C1, enquanto que MS encontrou 33 soluções ótimas. Considerando o conjunto C2, o MS e o BRKGA encontraram respectivamente 3 e 8 soluções ótimas. Resultados similares foram obtidos para todo o conjunto de instâncias C1 e C2. Ressalta-se que o MS e o BRKGA solucionou na otimalidade, para o grupo C1, uma instância não solucionada em Macambira *et al*. (2006). Em termos de tempos de processamento, o MS consumiu menos de 1 segundo, enquanto que o BRKGA consumiu no máximo 3 segundos. Dado a eficiência do método de partições de sequências, muitas oportunidades existem para sofisticar ainda mais as estratégias de soluções, tais como a inclusão de uma busca local e matheurísticas.

5 Conclusões e perspectivas

Neste trabalho, apresentamos um procedimento baseado em programação dinâmica para gerar partições em um grafo, considerando um problema de otimização, com restrições adicionais. O procedimento foi aplicado para gerar soluções para o SRAP de duas maneiras: na primeira estratégia, simplesmente gera-se uma quantidade pré-definida de sequências aleatórias que são particionadas usando o procedimento proposto. Na segunda estratégia, o procedimento é utilizado como um decodificador em um algoritmo genético com chaves aleatórias tendenciosas (BRKGA, do inglês *Biased Random Key Genetic Algorithms*), Gonçalves e Resende (2011); Resende (2012).

A grande vantagem da estratégia proposta é que existe garantia de otimalidade para a sequência inicial fornecida. O mecanismo de decodagem utilizado abre novas perspectivas para problemas de otimização que utilizam partições em grafos. Os experimentos preliminares são promissores: soluções ótimas foram encontradas em alguns poucos segundos.

Estamos investigando estratégias para gerar sequências iniciais de boa qualidade, mecanismos para melhorar o BRKGA. Atualmente, estamos desenvolvendo uma coleção de métodos para o SRAP, tais como buscas locais e matheurísticas, onde um *pool* de anéis viáveis são fornecidos para um *software* comercial de programação linear mista. Experimentos computacionais mais amplos serão realizados com o objetivo de determinar as vantagens e limites da técnica proposta. Como trabalhos futuros, investigaremos variantes determinísticas do SRAP, assim como variantes com incertezas sob os dados de tráfego. Assim como, iremos aplicar o procedimento proposto a outros problemas de partições em grafos.

Referências

Afsar, H. M., Prins, C. e Santos, A. C. (2014), Exact and heuristic algorithms for solving the generalized vehicle routing problem with flexible fleet size. *International Transactions in Operational Research*, v. 21, n. 1, p. 153–175.

Alvim, A. C., Ribeiro, C. C., Glover, F. e Aloise, D. J. (2004), A hybrid improvement heuristic for the one-dimensional bin packing problem. *Journal of Heuristics*, v. 10, n. 2, p. 205–229.

Aringhieri, R. e Dell'Amico, M. (2005), Comparing metaheuristic algorithms for SONET network design problems. *Journal of Heuristics*, v. 11, n. 1, p. 35–57.

Bastos, L. d. O., Ochi, L. S. e Macambira, E. M. GRASP with path-relinking for the SONET ring assignment problem. *Proceedings of the Fifth International Conference on Hybrid Intelligent Systems*, HIS '05, p. 239–244, Washington, DC, USA. IEEE Computer Society, 2005.

Bean, J. C. (1994), Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, v. 6, n. 2, p. 154–160.

Beasley, J. E. (1983), Route-first cluster-second methods for vehicle routing. *Omega*, v. 11, p. 403–408.

Bernardino, E. M., Bernardino, A. M., Sánchez-Pérez, J. M., Gómez-Pulido, J. A. e Vega-Rodríguez, M. A. (2012), Solving large-scale SONET network design problems using bee-inspired algorithms. *Optical Switching and Networking*, v. 9, n. 2, p. 97–117.

Coco, A. A., Júnior, J. C. A., Noronha, T. F. e Santos, A. C. (2014), An integer linear programming formulation and heuristics for the minmax relative regret robust shortest path problem. *To appear in Journal of Global Optimization*. DOI 10.1007/s10898-014-0187-x.

Coco, A. A., Noronha, T. F. e Santos, A. C. Algoritmo baseado em BRKGA para o problema de árvore geradora mínima robusta. *XLV Simpósio Brasileiro de Pesquisa Operacional (SBPO)*, p. 3658–3669, 2013.

de Souza, S. D. *Partições em grafos: caracterizações, algoritmos e complexidade*. Tese de doutorado, COPPE, Universidade Federal do Rio de Janeiro (UFRJ), 2002.

Goldschmidt, O., Laugier, A. e Olinick, E. V. (2003), SONET/SDH ring assignment with capacity constraints. *Discrete Applied Mathematics*, v. 129, n. 1, p. 99–128.

Gonçalves, J. F. e Resende, M. G. C. (2011), Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, v. 17, n. 5, p. 487–525.

Macambira, E. M., Maculan, N. e de Souza, C. C. (2006), A column generation approach for SONET ring assignment. *Networks*, v. 47, n. 3, p. 157–171.

Morán-Mirabal, L., González-Velarde, J. e Resende, M. (2014), Randomized heuristics for the family traveling salesperson problem. *International Transactions in Operational Research*, v. 21, n. 1, p. 41–57.

Prins, C., Lacomme, P. e Prodhon, C. (2014), Order-first split-second methods for vehicle routing problems: A review. *Transportation Research Part C: Emerging Technologies*, v. 40, p. 179–200.

Resende, M. G. C. (2012), Biased random-key genetic algorithms with applications in telecommunications. *TOP*, v. 20, n. 1, p. 130–153.