Uma metaheurística *Iterated Local Search* aplicada ao Problema de Correlação de Clusters

Mario Levorato
Petróleo Brasileiro S.A.
Rio de Janeiro, RJ – Brasil
levorato@petrobras.com.br

Rosa Figueiredo
Instituto de Matemática e Estatística, Universidade do Estado do Rio de Janeiro
20550-900 Rio de Janeiro, RJ – Brasil
rosa@ime.uerj.br

Lúcia Drummond e Yuri Frota
Instituto de Computação, Universidade Federal Fluminense
24210-240 Niterói, RJ – Brasil
{lucia, yuri}@ic.uff.br

RESUMO

O nível de equilíbrio em um grupo social pode ser utilizado como ferramenta de estudo pelos pesquisadores de redes sociais para saber de que forma (e se) um grupo evolui para um possível estado de equilíbrio. Neste sentido, uma rede social pode ser representada através de um grafo de sinais e a solução de problemas de *clustering* definidos sobre grafos de sinais pode ser utilizada como um critério para medição do nível de equilíbrio em redes sociais. Em particular, a solução ótima para o Problema de Correlação de Clusters (*Correlation Clustering* ou CC) consiste em uma possível medida desse equilíbrio. Este trabalho visa fornecer uma solução eficiente para o problema CC através do uso da metaheurística ILS. Este algoritmo proposto supera, em tempo de execução, as estratégias apresentadas na literatura, com a mesma qualidade de solução.

PALAVRAS CHAVE. ILS, VND, Correlação de Clusters.

Área Principal: MH - Metaheurísticas

ABSTRACT

One challenge for social network researchers is to evaluate balance in a social network. The degree of balance in a social group can be used as a tool to study whether and how this group evolves to a possible balanced state. The solution of clustering problems defined on signed graphs can be used as a criterion to measure the degree of balance in social networks. In particular, the optimal solution of the Correlation Clustering (CC) Problem arises as one possible measure. In this work, we provide an efficient solution of the CC problem by the use of the ILS metaheuristic. The proposed algorithm outperforms other solution strategies from literature in execution time, with the same solution quality.

KEYWORDS. ILS, VND, Correlation Clustering.

Main Area: MH - Metaheuristics

1. Introdução

Grafos de sinais foram utilizados por Heider (1946) com o objetivo de descrever relações de sentimento entre pessoas que pertencem a um mesmo grupo social e para fornecer uma definição formal para o conceito de equilíbrio social. Posteriormente, Cartwright e Harary (1956) formalizaram a teoria de Heider afirmando que um grupo social equilibrado poderia ser dividido em dois subgrupos mutuamente hostis, cada qual apresentando solidariedade interna. Até hoje os grafos de sinais têm se mostrado uma estrutura discreta bastante atraente para os pesquisadores de redes sociais (Doreian e Mrvar, 1996; Inohara, 1998; Yang et al., 2007; Abell e Ludwig, 2009; Doreian e Mrvar, 2009; Facchetti et al., 2011), os quais enfrentam o desafio de avaliar o equilíbrio em uma rede social. Para tanto, diferentes critérios e abordagens de solução têm sido utilizados na literatura na tentativa de se quantificar e avaliar o equilíbrio em uma rede social representada como um grafo de sinais (Doreian e Mrvar, 2009; Leskovec et al., 2010; Facchetti et al., 2011; Srinivasan, 2011).

Clustering é a atividade de particionar elementos individuais em grupos com base em sua similaridade. Problemas de *clustering* definidos em grafos de sinais aparecem em muitas áreas da ciência (Bansal et al., 2002; Gülpinar et al., 2004; DasGupta et al., 2007; Traag e Bruggeman, 2009; Huffner et al., 2010; Macon et al., 2012). O elemento comum entre essas aplicações é o ambiente colaborativo *vs* conflitante em que elas são definidas. A solução de problemas de *clustering* definidos em grafos de sinais pode ser usada como um critério para medir o grau de equilíbrio em redes sociais (Doreian e Mrvar, 1996, 2009; Figueiredo e Moura, 2013). Ao considerar a definição original (Heider, 1946) de equilíbrio estrutural, a solução ótima do Problema de Correlação de Clusters (CC) surge como uma medida para o grau de equilíbrio em uma rede social. Além disso, medidas alternativas para o equilíbrio estrutural e os problemas de *clustering* associados a elas foram discutidos recentemente em Doreian e Mrvar (2009) e em Figueiredo e Moura (2013).

Até onde pudemos averiguar, o problema CC foi abordado pela primeira vez em Doreian e Mrvar (1996) (não com este nome), onde sua solução por meio de uma heurística foi utilizada como critério para a análise de equilíbrio estrutural em redes sociais. A abordagem heurística proposta pelos autores consiste em um procedimento de busca local que pressupõe um conhecimento prévio do número de clusters na solução, tendo sido implementado no software Pajek (Batagelj e Mrvar, 2008). Recentemente, motivado por um problema de *clustering* de documentos, a versão sem pesos do problema CC foi formalizada em Bansal et al. (2002). Já a versão com pesos do problema foi abordada em Demaine et al. (2006). O problema CC tem sido amplamente investigado do ponto de vista de algoritmos com fator de aproximação constante e tem sido aplicado na solução de muitos problemas. Uma comparação entre estratégias heurísticas (métodos gulosos e de busca local) para este problema é apresentada em Elsner e Schudy (2009), onde é descrita uma aplicação para *clustering* de documentos e processamento de linguagem natural. Em Yang et al. (2007), o problema CC é chamado de *community mining* e uma heurística baseada em agentes é proposta para a sua solução. Por fim, uma abordagem baseada em algoritmos genéticos foi apresentada por Zhang et al. (2008), tendo sido aplicada no agrupamento de documentos (*document clustering*).

De um ponto de vista prático, para resolver o problema de *clustering* tratado neste artigo, abordagens heurísticas são as mais interessantes, uma vez que pode haver a necessidade de se analisar grandes redes sociais (Kunegis et al., 2009; Leskovec et al., 2010; Facchetti et al., 2011). Entretanto, é preciso lembrar que a definição de uma medida para representar o equilíbrio / desequilíbrio de uma rede social envolve por si só um grau de aproximação para a tarefa de avaliar o equilíbrio em uma rede social. Assim sendo, torna-se imprescindível que o problema de *clustering* associado a esta medida seja resolvido de forma eficiente. Do nosso conhecimento, apenas dois trabalhos propuseram abordagens metaheurísticas aplicadas ao problema CC: Zhang et al. (2008) e Drummond et al. (2013). Neste último, foi apresentado um algoritmo GRASP (*Greedy Randomized Adaptive Search Procedure*) (Feo e Resende, 1995) capaz de resolver o problema CC com eficiência em grafos de até 800 vértices. Contudo, observando o tempo excessivo gasto no processamento de grafos maiores, vislumbramos a oportunidade de estender este algoritmo, propondo

uma metaheurística híbrida que fosse capaz de resolver o problema mais rapidamente.

Neste trabalho apresentamos uma metaheurística ILS (*Iterated Local Search*) (Lourenço et al., 2003) com *multistart* para a solução eficiente do problema CC. Em seguida, usando os algoritmos propostos, mostramos a superioridade do ILS em relação ao procedimento GRASP existente.

2. Definição do problema e formulação matemática

Seja $G = (V, E)$ um grafo acíclico não direcionado onde V é o conjunto de n vértices e E é o conjunto de arestas de G. Para um conjunto de vértices $S \subseteq V$, seja $E[S] = \{(i,j) \in E \mid i,j \in S\}$ o *subconjunto de arestas induzidas por* S. Para dois conjuntos de vértices $S, W \subseteq V$, seja $E[S : W] = \{(i,j) \in E \mid i \in S, j \in W\}$. Observa-se, por definição, que $E[S : S] = E[S]$. Considere uma função $s : E \to \{+, -\}$ que atribui um sinal para cada aresta em E. Um grafo não direcionado G, juntamente com uma função s é chamado de *grafo de sinais*. Uma aresta $e \in E$ é *negativa* se $s(e) = -$, e *positiva* se $s(e) = +$. Logo, os conjuntos E^- e E^+ denotam, respectivamente, o conjunto de arestas negativas e positivas em um grafo de sinais.

Uma *partição* de V é uma decomposição de V em subconjuntos não vazios tais que todo elemento de V pertence a um e somente um desses subconjuntos. Considere uma partição $P = \{S_1,S_2,\ldots,S_l\}$ de V. As *arestas de corte* e as *arestas internas* relacionadas com esta partição são definidas, respectivamente, como as arestas dos conjuntos $\cup_{1\le i<j\le l} E[S_i : S_j]$ e $\cup_{1\le i\le l} E[S_i]$. Seja w_e o peso associado a uma aresta $e \in E$. Além disso, para $1 \le i, j \le l$, seja

$$\Omega^+(S_i,S_j) = \sum_{e\in E^+\cap E[S_i:S_j]} w_e \text{ e } \Omega^-(S_i,S_j) = \sum_{e\in E^-\cap E[S_i:S_j]} w_e.$$

O *desequilíbrio* $I(P)$ de uma partição P é definido como o peso total das arestas internas negativas e das arestas de corte positivas, isto é,

$$I(P) = \sum_{1\le i\le l} \Omega^-(S_i,S_i) + \sum_{1\le i<j\le l} \Omega^+(S_i,S_j). \tag{1}$$

Da mesma forma, o *equilíbrio* $B(P)$ de uma partição P pode ser definido como o peso total das arestas internas positivas e das arestas de corte negativas. Claramente, $B(P) + I(P) = \sum_{e\in E} w_e$.

Problema 2.1 (Problema CC) *Seja $G = (V,E,s)$ um grafo de sinais e w_e um peso não-negativo associado a cada aresta $e \in E$. O problema de correlação de clusters consiste no problema de se encontrar uma partição P de V tal que o desequilíbrio $I(P)$ seja minimizado ou, de forma análoga, o equilíbrio $B(P)$ seja maximizado.*

A formulação clássica para o problema CC consiste em um modelo de programação linear inteira (ILP) proposto para os problemas de *clustering* não-capacitados (Mehrotra e Trick, 1998). Nesta formulação, uma variável de decisão binária x_{ij} é atribuída a cada par de vértices $i,j \in V$, $i \neq j$, e definida da seguinte maneira: $x_{ij} = 0$ se i e j estão em um *cluster* comum; $x_{ij} = 1$ caso contrário. O modelo minimiza o desequilíbrio total.

$$\text{minimizar} \sum_{(i,j)\in E^-} w_{ij}(1 - x_{ij}) + \sum_{(i,j)\in E^+} w_{ij}x_{ij} \tag{2}$$

$$\text{sujeito a } x_{ip} + x_{pj} \ge x_{ij}, \qquad \forall\, i,p,j \in V, \tag{3}$$

$$x_{ij} = x_{ji}, \qquad \forall\, i,j \in V, \tag{4}$$

$$x_{ij} \in \{0,1\}, \qquad \forall\, i,j \in V. \tag{5}$$

As desigualdades de triângulo representadas pela equação (3) dizem que, se i e p estão no mesmo *cluster* assim como p e j, então os vértices i e j também devem estar no mesmo *cluster*. O conjunto de restrições (4), escritas para todo $i,j \in V$, estabelece que as variáveis x_{ij} e x_{ji} sempre assumem

o mesmo valor nesta formulação. As restrições em (5) impõem limitações às variáveis, que devem ser binárias, enquanto a função objetivo (2) minimiza o desequilíbrio, definido pela equação (1). Note que, de acordo com a equação (4), metade das variáveis podem ser eliminadas, o que reduz tanto o número de variáveis quanto o número de restrições desta formulação.

Uma formulação *set partitioning* (Mehrotra e Trick, 1998) é proposta na literatura para problemas de *clustering* não-capacitados e também poderia ser utilizada na solução do problema CC. Como podemos esperar, estas duas formulações não constituem abordagens adequadas quando o limite de tempo é uma restrição no processo de solução. Os autores em Figueiredo e Moura (2013) relatam que a formulação clássica começa a falhar (tempo limite fixado em $1h$) com instâncias aleatórias de 40 vértices e densidade das arestas negativas igual a $0{,}5$.

3. ILS aplicado ao Problema de Correlação de Clusters

Metaheurísticas têm sido utilizadas com sucesso para resolver problemas difícies de otimização combinatória por serem capazes de fornecer soluções sub-ótimas em um tempo razoável.

Iterated Local Search (ILS) (Lourenço et al., 2003) é uma metaheurística que tem por objetivo obter melhorias por meio de buscas estocásticas com múltiplos recomeços, conduzidas com a amostragem de soluções candidatas em vizinhanças distantes, aplicando-se, em seguida, uma busca local para refinar tais soluções e chegar ao seu ótimo local. Para atingir essas vizinhancas distantes, o ILS explora uma sequência de soluções criadas através de perturbações aplicadas sobre a melhor solução atual, criadas de acordo com uma heurística pré-definida.

Neste trabalho nós desenvolvemos uma hibridização do ILS, acrescentando o *multistart* (Martí, 2003). Em poucas palavras, o *multistart* propõe uma forma alternativa de atingir diversificação de soluções, por meio do reinício do ILS a partir de uma nova solução inicial sempre que uma região de busca já tiver sido extensivamente explorada.

A heurística proposta ($ILSMultiStartCC$) faz uso de elementos presentes no algoritmo GRASP (*Greedy Randomized Adaptive Search Procedure*) (Feo e Resende, 1995) para a solução do problema CC, algoritmo este proposto por Drummond et al. (2013). Nós aproveitamos as etapas de construção de uma solução inicial e de busca local apresentadas neste trabalho.

O Algoritmo 1 resume o funcionamento do ILS, composto por 4 módulos: o procedimento $ConstructClustering$, responsável por gerar uma solução inicial, a busca local $VariableNeighborhoodDescent$, um algoritmo para perturbar uma solução, implementado na função $Perturbation$, e um critério de aceitação, que define a partir de qual solução a busca será retomada. O parâmetro $iter$ representa o número de iterações *multistart*, ou seja, o número de vezes que o ILS será executado, retornando a melhor solução obtida com base em todas as execuções. O valor $iterMaxILS$ determina o número de iterações do laço principal do ILS, onde serão aplicadas as perturbações e a busca local. Por fim, o parâmetro $perturbationMax$ limita o nível das perturbações a serem aplicadas em uma determinada solução. A seguir descrevemos em detalhes cada etapa do algoritmo.

A primeira tarefa em cada iteração do $ILSMultiStartCC$ é a construção de uma solução de maneira gulosa e aleatória. Sua lógica foi inspirada no trabalho de Nascimento e Pitsoulis (2013), tendo sido modificada a função de ganho. No algoritmo GRASP de Drummond et al. (2013), a função baseou-se em modularidade (comumente usada em grafos sem sinais). Já no ILS passou a ser levado em conta o objetivo do problema CC, isto é, a minimização do desequilíbrio. Para descrever tal construção, são necessárias algumas definições adicionais. Seja $P = \{S_1,S_2,...,S_l\}$ uma *partição parcial* (ou seja, uma partição de um subconjunto próprio de V). Definimos abaixo uma *função* $f : (V \setminus \bigcup_{1\leq k\leq l} S_k) \rightarrow \mathbb{R}$, que irá medir o impacto da inserção de um vértice i na partição parcial P.

$$f(i) = \min\left(I(P \cup \{i\}), \min_{\substack{1\leq k\leq l \\ S_k\cup\{i\}}} I(P)\right). \tag{6}$$

Esta função de minimização compara o custo de inserção do vértice i em um novo cluster ($f(i) = I(P \cup \{i\})$) com o custo de inseri-lo em um dos l clusters de P. Observe que um vértice com um baixo valor da função f provavelmente gerará menos desequilíbrio se for adicionado à partição parcial P.

A tarefa de construção inicial é realizada na fase $ConstructClustering$ descrita no Algoritmo 2. Ela tem início com a ordenação do conjunto L_F (linha 2), definido como o conjunto de vértices $V \setminus \bigcup_{1 \leq k \leq l} S_k$ ordenados de maneira crescente de acordo com a função f. A cada iteração (linhas 3-7) é escolhido um vértice i aleatoriamente dentre os primeiros $\lfloor \alpha . \mid L_f \mid \rfloor$ vértices deste conjunto, sendo este adicionado à partição parcial. Note que o parâmetro α define o grau de aleatoriedade da fase de construção. Este processo é repetido até que uma partição de V seja obtida.

Algoritmo 1: $ILSMultiStartCC$

Entrada: $G = (V,E)$, α, $iter$, $iterMaxILS$ e $perturbationMax$
Saída: partição P^*

```
1  inicio
2    P* = ∅; I(P*) = ∞; i = 1;
3    enquanto i ⩽ iter
4      P = ConstructClustering(G,α);
5      P = VariableNeighborhoodDescent(P,G);
6      j = 1; t = 1;
7      repita
8        P̄ = Perturbation(P,t);
9        P̄ = VariableNeighborhoodDescent(P̄,G);
10       se (I(P̄) < I(P))
11         P = P̄; j = 1; t = 1;
12       senão
13         j = j + 1;
14         se (j ⩾ iterMaxILS)
15           t = t + 1; j = 1;
16         fim-se
17       fim-se
18     enquanto t ⩽ perturbationMax
19     i = i + 1;
20     se (I(P) < I(P*))
21       P* = P;
22     fim-se
23   fim-enquanto
24   retorne P*;
```

Não há garantia de que o método de construção retorne uma solução localmente ótima em relação a alguma vizinhança. Portanto, a solução P obtida na fase $ConstructClustering$ pode ser melhorada pelo procedimento de busca local $VariableNeighborhoodDescent$ descrito na Algoritmo 3.

O método *Variable Neighborhood Search* (VNS), proposto por Mladenović e Hansen (1997), é uma metaheurística que explora progressivamente vizinhanças mais distantes da solução atual, mudando para uma nova solução apenas se uma melhoria for obtida. Sua principal idéia é explorar diferentes estruturas de vizinhança de forma sistemática, com o objetivo de escapar de mínimos locais. Neste trabalho, aplicamos a busca local *Variable Neighborhood Descent* (VND) (Hertz e Mittaz, 2001), uma variante do VNS. A partir de uma solução inicial fornecida, o VND pro-

cura substituí-la, de forma iterativa, por uma outra configuração com desequilíbrio mínimo dentro de sua vizinhança (linhas 4-9), alternando para uma vizinhança maior quando não encontra solução melhor na vizinhança atual, denotada por $N_r(P)$. A busca local é interrompida quando não há melhor solução na vizinhança mais distante da solução atual. A vizinhança $N_r(P)$ é definida como a família de todas as partições obtidas movendo-se r vértices em P de um cluster para outro. Note-se que esta definição de vizinhança permite analisar partições com diferentes quantidades de clusters (ou seja, um vértice pode ser transferido para um novo cluster ou pode ser removido de um cluster unitário). Neste trabalho, foi empregada uma vizinhança com $r \leq 2$ e, como sua travessia por completo mostrou-se muito demorada, optamos pela heurística de primeira melhoria (*first descent* ou *first improvement*). Vale observar também que o VND utilizado no ILS é exatamente o mesmo empregado na etapa de busca local do GRASP.

O passo seguinte a ser realizado pelo algoritmo é gerar uma perturbação, cujo nível é proporcional ao valor de t. A perturbação, listada no Algoritmo 4, consiste em simplesmente t movimentações aleatórias de um vértice qualquer de um cluster para outro.

Conforme descrito por Den Besten et al. (2001), a eficácia do módulo de busca local é de extrema importância para o ILS, uma vez que ele influencia na qualidade da solução final da metaheurística, assim como no tempo total gasto. Por sua vez, as perturbações devem permitir que o ILS efetivamente escape de ótimos locais, mas evitando, ao mesmo tempo, as desvantagens de um reinício aleatório (ou seja, não devem ser fortes em demasia). O critério de aceitação, juntamente com as perturbações, influenciam bastante no tipo de caminhada pelo espaço de solução e podem ser usados para ajustar o equilíbrio do algoritmo, alternando entre intensificação e diversificação da busca. O desafio de se projetar um algoritmo de ILS consiste na necessidade de se encontrar a melhor combinação e configuração possível de seus módulos de tal maneira que o melhor desempenho geral seja atingido.

Algoritmo 2: $ConstructClustering$

Entrada: $G = (V,E)$ e α
Saída: partição P
1 **inicio**
2 $P = \emptyset$; $L_f = Ordena(V)$;
3 **enquanto** $(L_f \neq \emptyset)$
4 Escolha um vértice i aleatoriamente dentre os primeiros $\lfloor \alpha . \mid L_f \mid \rfloor$ elementos de L_f;
5 Atualize $S_k = S_k \cup \{i\}$ onde k é o componente em P que minimiza f;
6 $L_f = L_f - \{i\}$; Reordena(L_f);
7 **fim-enquanto**
8 **retorna** P;

4. Resultados e ganhos obtidos

Os algoritmos descritos na seção anterior foram implementados em ANSI C++. Todos os experimentos, inclusive aqueles com o algoritmo GRASP (a cujo código fonte tivemos acesso), foram realizados em uma máquina dedicada com processador Intel Core i7 4820k 3.70GHz, 32Gb de RAM e sistema operacional Linux Mint 16 (Kernel 3.11.6). A formulação ILP foi codificada em Mosel Xpress 3.2.0 com solver Xpress Optimizer 21.01.00. Para cada execução do modelo e das metaheurísticas, foi definido o limite de tempo de 1 hora. Além disso, por não apresentarem alta variância nos resultados, todos os resultados das heurísticas representam o valor médio de 5 execuções independentes. Experimentos computacionais foram realizados em (i) um conjunto de 29 redes sociais provenientes da literatura, (ii) um conjunto de 63 instâncias de redes que representam votações da Assembleia Geral das Nações Unidas (UNGA), e (iii) um conjunto de 12 instâncias

Algoritmo 3: $VariableNeighborhoodDescent$

Entrada: $G = (V,E)$ e uma partição P
Saída: partição P
inicio
  $r = 1$;
  enquanto $(r \leq 2)$
    para todo $\overline{P} \in (N_r(P))$
      se $(I(\overline{P}) < I(P))$
        $P = \overline{P}$;
        $r = 1$;
      fim-se
    fim-para
    se (P não melhorou)
      $r = r + 1$;
    fim-se
  fim-enquanto
  retorna P;

Algoritmo 4: $Perturbation$

Entrada: $G = (V,E)$, uma partição P e o nível de perturbação t
Saída: partição P
inicio
  $i = 1$;
  enquanto $(i \leqslant t)$
    Escolha aleatoriamente um cluster $c_x \in P$ e um vértice $i \in c_x$;
    Escolha aleatoriamente um cluster $c_y \in P$ tal que $x \neq y$;
    Mova o vértice i do cluster c_x para o cluster c_y;
    $i = i + 1$;
  fim-enquanto
  retorna P;

aleatórias. A seguir, descrevemos brevemente essas instâncias[1].

(i) Este conjunto de instâncias é composto por 22 redes sociais de pequeno porte normalmente utilizadas em abordagens *blockmodeling* para avaliação do equilíbrio estrutural (Brusco, 2003; Doreian e Mrvar, 2009; Figueiredo e Moura, 2013) e 7 redes de sinais extraídas da rede social de grande porte Slashdot, um site de notícias relacionadas com tecnologia (Leskovec et al., 2010; Facchetti et al., 2011). As instâncias de pequeno porte foram utilizadas para parametrizar as heurísticas.

(ii) Geramos 63 redes sociais de médio porte com base em registros de votação da UNGA, provenientes das sessões anuais separadas ano a ano entre 1946 e 2008[2]. Essas instâncias representam versões ponderadas dos grafos de sinais da UNGA, conforme descrito em Figueiredo e Frota (2014). O conjunto de vértices em um grafo de sinais representa o conjunto de países da sessão de votação anual correspondente. O conjunto de arestas ponderadas positivas

[1] todas as instâncias estão disponíveis em http://www.ic.uff.br/∼ yuri/files/CCinst.zip.
[2] Assembléia Geral das Nações Unidas - dados de votação, Anton Strezhnev e Erik Voeten, disponível em http://hdl.handle.net/1902.1/12379. Acessado em 12 de maio de 2014.

/ negativas é definido da seguinte forma. Para cada par de vértices (países) i,j e para cada resolução votada na sessão, são totalizados os pesos associados aos pares de votos de i e j: um acordo tem peso igual a $+1{,}0$ ou $+0{,}5$; um desacordo tem peso igual a $-1{,}0$ ou $-0{,}5$. Seguindo uma observação de Macon et al. (2012), tratamos de forma diferente o desacordo (acordo) em um par de votos para uma mesma resolução: sim-não (sim-sim ou não-não) são tratados de maneira diferente de um sim-abstenção ou não-abstenção (abstenção-abstenção). Além disso, é feita uma normalização dos pesos pelo número total de votos em uma sessão.

(iii) Geramos redes sociais aleatórias com 50 vértices ($n = 50$), variando a densidade da rede $d = 2 \times |E|/(n^2 - n)$ e a densidade negativa do grafo definida aqui como $d^- = |E^-|/|E|$. Consideramos um conjunto de 12 instâncias aleatórias com d e d^- variando, respectivamente, nos conjuntos $\{0{,}1;\ 0{,}2;\ 0{,}5;\ 0{,}8\}$ and $\{0{,}2;\ 0{,}5;\ 0{,}8\}$. Essas instâncias também foram utilizadas por Figueiredo e Moura (2013).

Conforme relatado na literatura (Mehrotra e Trick, 1998), a relaxação linear da formulação ILP apresentada na Seção 2 fornece uma boa representação do problema, permitindo encontrar a solução ótima em muitos casos através da resolução desta relaxação linear. Experiências relatadas em Figueiredo e Moura (2013) confirmam esta afirmação: as 22 pequenas instâncias do conjunto (i) foram resolvidas até a otimalidade em alguns segundos; a formulação falhou em resolver as instâncias aleatórias do conjunto (iii) que possuem $d \geq 0{,}2$ e $d^- = 0{,}5$. Em nossos experimentos, todas as instâncias do conjunto (ii) foram resolvidas à otimalidade pela formulação ILP na raiz da árvore de *branch and bound*. Podemos concluir que essas redes de sinais são quase que perfeitamente equilibradas, com a maior parte do desequilíbrio proveniente das relações negativas.

A desvantagem da formulação ILP aparece quando tentamos resolver as instâncias baseadas na rede social Slashdot: a formulação ILP torna-se muito grande e o Xpress não é capaz de resolver nenhuma instância dentro do tempo limite.

Como já mencionado na seção anterior, em Drummond et al. (2013), foram propostas duas abordagens para a solução dessas instâncias maiores. Um algoritmo GRASP sequencial composto por 400 iterações e um GRASP paralelo usando 8 processadores (GraspPar 8) e 50 iterações por processador. Contudo, foi observado que mesmo a versão em paralelo do GRASP apresentava tempos de processamento demasiadamente longos para instâncias com mais de 800 vértices, chegando a ultrapassar o tempo limite estipulado de 2 horas.

Foi então que, visando contornar esta limitação, nós nos baseamos nos módulos construtivo e de busca local (VND) deste GRASP, assim como em sua natureza *multistart*, e adicionamos um componente responsável por introduzir perturbações na melhor solução atual, chegando ao algoritmo $ILSMultiStartCC$, descrito na seção anterior. Como veremos adiante, a grande vantagem que este algoritmo leva sobre o GRASP é justamente o tempo de execução reduzido. É possível realizar i buscas locais dentro do ILS de maneira muito mais rápida do que se as mesmas i buscas locais fossem executadas dentro da estrutura de reinício aleatório do GRASP, pois provavelmente são necessários menos movimentos na vizinhança para se alcançar o ótimo local a partir de uma solução perturbada do que a partir de uma nova solução gerada por um algoritmo guloso-aleatório. Em outras palavras, o poder potencial do ILS reside em sua amostragem aleatória e tendenciosa (perturbações) do conjunto de ótimos locais encontrados na etapa de busca. Segundo Den Besten et al. (2001), mesmo com as implementações mais simples de perturbações e critérios de aceitação, o ILS mostra-se muito superior do que o simples reinício aleatório. Tal superioridade pode ser demonstrada, por exemplo, no tempo de execução da metaheurística.

Os experimentos feitos com o ILS utilizaram o seguinte conjunto de parâmetros: vizinhança $r = 1$, $iter = 10$ (10 iterações *multistart*), $\alpha = 0{,}4$, $iterMaxILS = 5$ e $perturbationMax = 30$.

A Tabela 1 compara os resultados obtidos com o conjunto de instâncias aleatórias (iii) pelo GRASP sequencial ($r \leq 2$, $iter = 400$ and $\alpha = 0{,}8$) e pelo $ILSMultiStartCC$ ($r = 1$, $iter = 10$

and $\alpha = 0{,}4$, $iterMaxILS = 5$ e $perturbationMax = 30$), ambos em sua melhor configuração, com o tempo limite configurado para 30 minutos. Pode-se notar que o ILS consegue soluções iguais ou melhores que as do GRASP (coluna *Gap* - BestSol), mas com tempos de execução médios consideravelmente menores (coluna *Gap* - AvgTime), atingindo um *gap* médio de $216{,}96$ segundos a menos, onde *gap* é a diferença entre entre o valor de solução, ou de tempo, entre os algoritmos ILS e GRASP, respectivamente.

Por sua vez, a aplicação do algoritmo ILS para as instâncias UNGA demandou a redução do número máximo de perturbações, sem prejuízo do valor da solução encontrado, a fim de se obter um tempo de execução vantajoso. Foram testadas diversas combinações de parâmetros para o ILS, modificando o nível máximo de perturbação até um determinado patamar. Ao final dos testes a configuração a seguir apresentou o melhor resultado: $r = 1$, 10 iterações *multistart*, $\alpha = 0{,}4$, $iterMaxILS = 5$ e $perturbationMax = 7$.

Por se tratarem de instâncias extremamente balanceadas, pequenas perturbações somente fazem mudar para uma outra solução muito parecida com a anterior, ou seja, não foi possível, com pequenas perturbações, dar os saltos que o ILS precisava no espaço de solução viável. Interessante também notar que nas instâncias UNGA, tivemos um resultado semelhante para as metaheurísticas GRASP e ILS. Uma possível explicação reside no fato de que, como tais instâncias já são bem balanceadas, tanto as soluções iniciais pseudo-aleatórias quanto as pertubações estariam levando para o mesmo ponto a ser reotimizado.

O ILS conseguiu alcançar, dentro do tempo limite do GRASP (1 hora), os mesmos valores de solução para todas as instâncias do grupo. Analisando-se a Figura 1, pode-se notar que em apenas 6 instâncias das 63, o ILS apresentou um tempo de execução maior que o do GRASP. Apesar deste fato, na soma dos tempos médios de resolução das instâncias UNGA, o ILS levou $22.026{,}04$ segundos a menos. Isso resulta em uma média de $349{,}61$ segundos a menos por instância, considerando-se a média dos tempos coletados em 5 execuções independentes de cada algoritmo.

Os resultados da execução do ILS sobre as instâncias Slashdot são apresentados na Tabela 2 ("–"significa que nenhuma solução inteira viável foi encontrada no tempo limite). As instâncias que não foram resolvidas à otimalidade pela formulação ILP são marcadas com valores em negrito na coluna (ILP-BestBound); nesse caso a coluna apresenta o valor da melhor solução inteira encontrada no tempo limite. Nota-se que o ILS alcança os mesmos valores de solução do GRASP, exceto por duas instâncias (coluna *Gap* - BestSol), e ganha no tempo de execução em todas elas (coluna *Gap* - AvgTime). Ao todo, foram gastos $25.298{,}46$ segundos a menos para resolver todas as instancias Slashdot, uma média de $3.614{,}06$ segundos a menos por instância e tivemos um *gap* médio (em relação ao valor de solução do GRASP) de $-0{,}57$ (melhor que o GRASP). Esses dados comprovam que, na média, o ILS é muito mais rápido que o GRASP, conseguindo manter a qualidade de suas soluções.

5. Conclusões e trabalhos futuros

Os resultados computacionais obtidos indicam que nossa metaheurística ILS *multistart* consiste em uma abordagem rápida e eficiente para a solução heurística do problema CC, superando, em tempo de processamento, a metaheurística GRASP proposta anteriormente, com qualidade de solução similar. Experimentos numéricos adicionais precisam ser conduzidos com instâncias de outras aplicações. A experiência numérica por meio de redes sociais maiores indica que, a fim de lidar com instâncias grandes como as redes sociais Epinions (131.828 vértices e 841.372 arestas) e Slashdot (82.144 vértices e 549.202 arestas), existe a necessidade de se implementar estratégias de paralelização.

Resultados ainda melhores podem ser obtidos se os módulos do ILS forem otimizados. Primeiramente, o critério de aceitação, definido para aceitar apenas soluções melhores que a atual, pode ser modificado na busca de um equilíbrio maior entre intensificação e diversificação, podendo vir a aceitar soluções piores que a melhor solução atual, o que permitiria ampliar o espaço de busca.

Instâncias Aleatórias	ILP	GRASP		ILS		*Gap*	
	BestSol	BestSol	AvgTime	BestSol	AvgTime	BestSol	AvgTime
$d = 0{,}1$ $d^- = 0{,}2$	48	48,00	51,64	48,00	2,24	0,00	-6,93
$d = 0{,}1$ $d^- = 0{,}5$	55	56,43	395,57	55,00	7,56	-1,43	-43,14
$d = 0{,}1$ $d^- = 0{,}8$	18	18,43	648,25	18,0	8,99	-0,43	-87,14
$d = 0{,}2$ $d^- = 0{,}2$	98	98,00	8,69	98,00	2,47	0,00	-2,83
$d = 0{,}2$ $d^- = 0{,}5$	**159**	149,57	164,24	149,00	9,76	-0,57	-59,61
$d = 0{,}2$ $d^- = 0{,}8$	58	58,20	999,26	58,00	14,35	-0,20	-268,09
$d = 0{,}5$ $d^- = 0{,}2$	245	245,00	7,52	245,00	2,44	0,00	-3,68
$d = 0{,}5$ $d^- = 0{,}5$	**523**	461,00	205,82	461,00	11,72	0,00	-69,92
$d = 0{,}5$ $d^- = 0{,}8$	196	197,67	1800,61	196,00	23,99	-1,67	-689,21
$d = 0{,}8$ $d^- = 0{,}8$	392	392,00	7,52	392,00	2,90	0,00	-4,13
$d = 0{,}8$ $d^- = 0{,}5$	**879**	775,29	228,47	775,00	13,88	-0,29	-86,69
$d = 0{,}8$ $d^- = 0{,}8$	334	335,00	1803,40	334,00	32,52	-1,00	-1292,10
Média	250,42	236,22	526,75	236,30	11,07	-0,47	-216,96

Tabela 1: Resultados obtidos nas instâncias aleatórias ($n = 50$) com as melhores configurações do GRASP sequencial ($r \leq 2$, 400 iterações sem melhora, $\alpha = 0{,}8$) e do algoritmo ILS sequencial ($r = 1$, 10 iterações *multistart*, $\alpha = 0{,}4$, $iterMaxILS = 5$ e $perturbationMax = 30$). BestSol é o valor da melhor solução encontrada dentro do tempo limite; AvgTime é o tempo médio gasto em 5 execuções independentes da heurística. *Gap* é a diferença entre entre o valor de solução, ou de tempo, entre os algoritmos ILS e GRASP, respectivamente.

Slashdot	Formulação ILP		GRASP			ILS			*Gap*	
n	k	BestSol	k	BestSol	AvgTime	k	BestSol	AvgTime	BestSol	AvgTime
200	1	65	3	48,0	15,12	3	48,0	6,46	0,00	-8,66
300	1	85	3	57,0	35,64	4	57,0	21,07	0,00	-14,57
400	1	87	3	60,0	70,05	4	60,0	17,82	0,00	-52,23
600	–	–	3	113,0	223,71	3	114,0	32,99	1,00	-190,73
800	–	–	5	247,0	719,18	4	249,0	60,52	2,00	-658,66
1000	–	–	5	628,0	1618,86	6	627,2	142,63	-0,80	-1476,23
2000	–	–	7	2277,0	7205,07	7	2277,0	605,55	0,00	-6599,52

Tabela 2: Resultados obtidos nos grafos de sinais baseados na rede social Slashdot, com as melhores configurações do GRASP sequencial ($r = 1$, 400 iterações sem melhora, $\alpha = 0{,}4$) e do algoritmo ILS sequencial ($r = 1$, 10 iterações *multistart*, $\alpha = 0{,}4$, $iterMaxILS = 5$ e $perturbationMax = 30$). Número de vértices (n); número de partições (k) na solução ótima $\bar{P}$; BestSol é o valor da melhor solução encontrada dentro do tempo limite; AvgTime é o tempo médio gasto em 5 execuções independentes da heurística. *Gap* é a diferença entre entre o valor de solução, ou de tempo, entre os algoritmos ILS e GRASP, respectivamente.

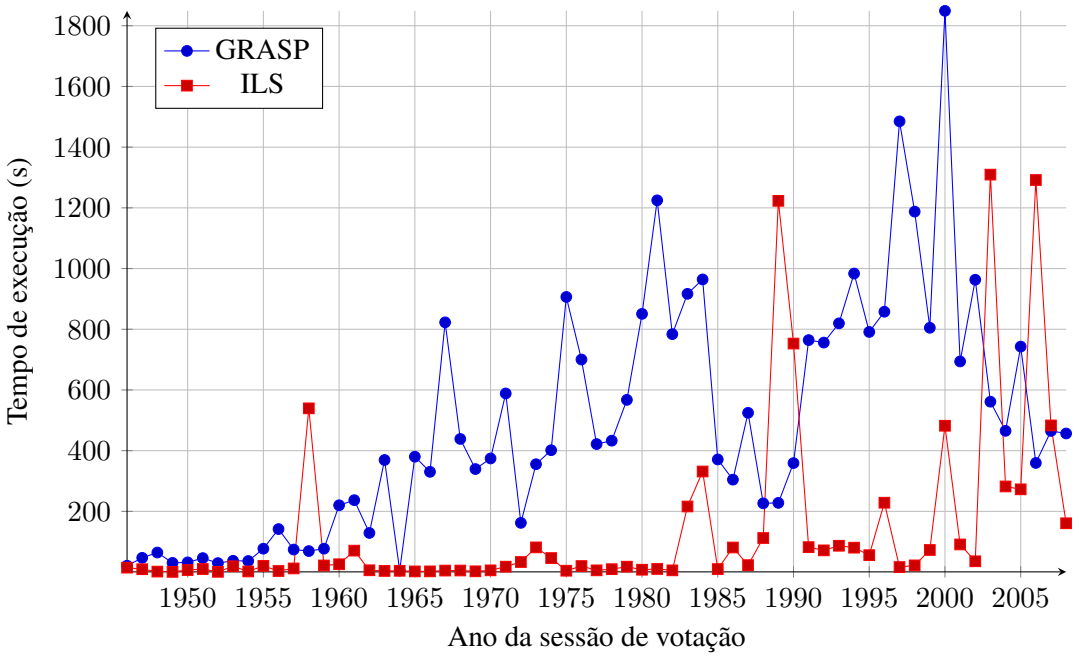

Figura 1: Tempo gasto para resolver as instâncias UNGA com as melhores configurações do GRASP sequencial ($r \leq 2$, 400 iterações sem melhora, $\alpha = 0{,}8$) e do algoritmo ILS sequencial ($r = 1$, 10 iterações *multistart*, $\alpha = 0{,}4$, $iterMaxILS = 5$ e $perturbationMax = 7$). Média de 5 execuções independentes de cada heurística.

Além disso, a rotina de perturbação pode ser ajustada para incorporar diversas informações específicas relativas ao problema, seguindo uma boa prática que diz que uma boa perturbação transforma uma excelente solução em um excelente ponto de partida para uma busca local. Por fim, não apenas novos tipos de vizinhança podem ser acrescentados à busca local (por exemplo, a divisão de *clusters* em duas ou mais partes), como também um módulo de reconexão de caminhos (Laguna e Marti, 1999; Resende e Ribeiro, 2005), em que soluções são combinadas através da geração de "caminhos" entre elas (e passando por elas).

Como já mencionado na introdução, foram propostas medidas alternativas na literatura para o equilíbrio estrutural (Doreian e Mrvar, 2009; Figueiredo e Moura, 2013). Os próximos passos desta pesquisa incluem também a adaptação da metaheurística ILS para lidar com essas diferentes medidas.

Agradecimentos

Os autores agradecem ao Prof. Luiz Satoru Ochi por suas importantes contribuições para a melhoria deste trabalho.

Referências

Abell, P. e Ludwig, M. (2009). Structural balance: a dynamic perspective. *Journal of Mathematical Sociology*, 33:129–155.

Bansal, N., Blum, A., e Chawla, S. (2002). Correlation clustering. In *Proceedings of the 43rd annual IEEE symposium of foundations of computer science*, pages 238–250, Vancouver, Canada.

Batagelj, V. e Mrvar, A. (2008). Pajek wiki. http://pajek.imfm.si/. Acessado em 12.05.2014.

Brusco, M. (2003). An enhanced branch-and-bound algorithm for a partitioning problem. *British Journal of Mathematical and Statistical Psychology*, 56:83–92.

Cartwright, D. e Harary, F. (1956). Structural balance: A generalization of heider's theory. *Psychological Review*, 63:277–293.

DasGupta, B., Encisob, G. A., Sontag, E., e Zhanga, Y. (2007). Algorithmic and complexity results for decompositions of biological networks into monotone subsystems. *BioSystems*, 90:161–178.

Demaine, E. D., Emanuel, D., Fiat, A., e Immorlica, N. (2006). Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361:172–187.

Den Besten, M., Stützle, T., e Dorigo, M. (2001). Design of iterated local search algorithms. In *Applications of Evolutionary Computing*, pages 441–451. Springer.

Doreian, P. e Mrvar, A. (1996). A partitioning approach to structural balance. *Social Networks*, 18:149–168.

Doreian, P. e Mrvar, A. (2009). Partitioning signed social networks. *Social Networks*, 31:1–11.

Drummond, L., Figueiredo, R., Frota, Y., e Levorato, M. (2013). Efficient solution of the correlation clustering problem: An application to structural balance. In Demey, Y. e Panetto, H., editors, *On the Move to Meaningful Internet Systems: OTM 2013 Workshops*, volume 8186 of *Lecture Notes in Computer Science*, pages 674–683. Springer Berlin Heidelberg.

Elsner, M. e Schudy, W. (2009). Bounding and comparing methods for correlation clustering beyond ilp. In *ILP'09 Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 19–27.

Facchetti, G., Iacono, G., e Altafini, C. (2011). Computing global structural balance in large-scale signed social networks. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 108, pages 20953–20958.

Feo, T. A. e Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133.

Figueiredo, R. e Frota, Y. (2014). The maximum balanced subgraph of a signed graph: Applications and solution approaches. *European Journal of Operational Research*, 236(2):473 – 487.

Figueiredo, R. e Moura, G. (2013). Mixed integer programming formulations for clustering problems related to structural balance. *Social Networks*, 35(4):639–651.

Gülpinar, N., Gutin, G., Mitra, G., e Zverovitch, A. (2004). Extracting pure network submatrices in linear programs using signed graphs. *Discrete Applied Mathematics*, 137:359–372.

Heider, F. (1946). Attitudes and cognitive organization. *Journal of Psychology*, 21:107–112.

Hertz, A. e Mittaz, M. (2001). A variable neighborhood descent algorithm for the undirected capacitated arc routing problem. *Transportation Science*, 35(4):425–434.

Huffner, F., Betzler, N., e Niedermeier, R. (2010). Separator-based data reduction for signed graph balancing. *Journal of Combinatorial Optimization*, 20:335–360.

Inohara, T. (1998). On conditions for a meeting not to reach a deadlock. *Applied Mathematics and Computation*, 90:1–9.

Kunegis, J., Lommatzsch, A., e Bauckhage, C. (2009). The slashdot zoo: mining a social network with negative edges. In *WWW'09 Proceedings of the 18th international conference on World wide web*, pages 741–750.

Laguna, M. e Marti, R. (1999). Grasp and path relinking for 2-layer straight line crossing minimization. *INFORMS Journal on Computing*, 11(1):44–52.

Leskovec, J., Huttenlocher, D., e Kleinberg, J. (2010). Signed networks in social media. In *CHI'10 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1361–1370.

Lourenço, H. R., Martin, O. C., e Stützle, T. (2003). *Iterated local search*. Springer.

Macon, K., Mucha, P., e Porter, M. (2012). Community structure in the united nations general assembly. *Physica A: Statistical Mechanics and its Applications*, 391:343–361.

Martí, R. (2003). Multi-start methods. In *Handbook of metaheuristics*, pages 355–368. Springer.

Mehrotra, A. e Trick, M. A. (1998). Cliques and clustering: A combinatorial approach. *Oper. Res. Lett.*, 22(1):1–12.

Mladenović, N. e Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100.

Nascimento, M. C. e Pitsoulis, L. (2013). Community detection by modularity maximization using grasp with path relinking. *Computers Operations Research*. Available online on March 2013.

Resende, M. G. e Ribeiro, C. C. (2005). Grasp with path-relinking: Recent advances and applications. In *Metaheuristics: progress as real problem solvers*, pages 29–63. Springer.

Srinivasan, A. (2011). Local balancing influences global structure in social networks. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 108, pages 1751–1752.

Traag, V. e Bruggeman, J. (2009). Community detection in networks with positive and negative links. *Physical Review E*, 80:036115.

Yang, B., Cheung, W., e Liu, J. (2007). Community mining from signed social networks. *IEEE Transactions on Knowledge and Data Engineering*, 19:1333–1348.

Zhang, Z., Cheng, H., Chen, W., Zhang, S., e Fang, Q. (2008). Correlation clustering based on genetic algorithm for documents clustering. In *IEEE Congress on Evolutionary Computation*, pages 3193–3198.