O Papel de Buscas Locais Estocásticas na Modularidade por Densidade

Alex Luciano Roesler Rese, Fernando Concatto, Rafael de Santiago

[1]Laboratório de Inteligência Aplicada - LIA
Universidade do Vale do Itajaí (UNIVALI)
Caixa Postal 360 – CEP 88302-202 – Itajaí – SC – Brasil

alexrese@outlook.com, fernandoconcatto@gmail.com, rsantiago@univali.br

Abstract. *Modularity Density Maximization is a computational problem about solving the community detection problem in networks, which attracted recent contributions in the field of algorithms and heuristics. Results using stochastic local searches was not found or did not exist in the literature. The aim of this work is to identify the role of local searches, especially stochastic local searches, for improving Modularity Density Maximization solutions. The tested searches were a local search that stops in local optimum, Iterated Local Search, Randomised Iterative Improvement, and Tabu Search. All our searches used 1-neighborhood strategy. Tests showed that Iterated Local Search, Randomised Iterative Improvement, and Tabu Search obtained improvements larger than* 16% *for solutions from state-of-the-art heuristic.*

Resumo. *Maximização da Modularidade por Densidade é um problema computacional para a detecção de comunidades em redes que tem recebido uma série de contribuições recentes na literatura sobre algoritmos e heurísticas. Apesar disto, resultados sobre buscas locais estocásticas não foram encontrados na literatura ou inexistem. O objetivo do trabalho reportado neste artigo foi identificar o papel de buscas locais, mais especificamente busca locais estocásticas, na melhoria de soluções para o problema da Maximização da Modularidade por Densidade. As buscas experimentadas foram uma busca local que pára em ótimos locais, a Busca Local Iterada, a Busca Local Monótona Randomizada e a Busca Tabu. Todas as buscas utilizando estratégias de 1-vizinhança. Os experimentos demonstraram que a Busca Local Iterada, Busca Local Monótona Randomizada e a Busca Tabu obtiveram melhoria superior a* 16% *em soluções obtidas por uma heurística considerada estado-da-arte na literatura.*

1. Introdução

Identificar padrões em redes (ou grafos) tem demonstrado importância para diversas áreas na literatura. No contexto biológico, redes que com muitos vértices tornam o trabalho de um analista humano praticamente impossível. Neste sentido, no mapeamento metabólico, agrupar *metabolites* (vértices) de acordo com suas interações pode auxiliar o analista, pois ele passaria a analisar uma "meta-rede" com vértices agrupados, como acontece no trabalho de [Guimera and Amaral 2005]. Um exemplo no contexto social seria identificar com quais grupos um criminoso interagem em redes de contato telefônico. Mesmo sem ter

acesso a conversas ou trocas de mensagens, pode-se identificar quais grupos pertencem a família, amigos ou pessoas com ficha criminal suspeita e desmantelar uma quadrilha [Ferrara et al. 2014, Calderoni et al. 2017].

Identificar grupos em redes é um problema conhecido na literatura como de "Detecção de Comunidades", onde o termo comunidade se refere ao agrupamento de vértices que representam mesma funcionalidade ou maior interação. De acordo com [Radicchi et al. 2004, Girvan and Newman 2002], estes grupos são caracterizados por uma alta "modularidade", ou seja, alta densidade de conectividade entre os membros do grupo. Lê-se esta conectividade como uma alta proporção de arestas internas em relação a quantidade de arestas que conectam uma comunidade a vértices de comunidades vizinhas.

Os problemas de "Detecção de Comunidades" são geralmente classificados de duas formas. A primeira delas considera que os vértices de uma rede podem pertencer a uma ou mais comunidades. Estes problemas são chamados de "Detecção de Comunidades com Sobreposição" [Xie et al. 2013]. A outra classificação e mais comum é a detecção "sem Sobreposição", onde cada vértice pertence a uma comunidade [Fortunato and Hric 2016].

Um dos problemas que de "Detecção de Comunidades sem Sobreposição" é a Maximização da Modularidade por Densidade. Este problema utiliza uma função objetivo que quantifica a diferença da densidade interna das comunidades em relação à externa para medir a modularidade de uma solução. Este problema foi introduzido por [Li et al. 2008] depois do artigo [Fortunato and Barthélemy 2007] relatou uma deficiência para um problema amplamente utilizado na literatura chamado de Maximização da Modularidade [Newman and Girvan 2004]. Logo, a Maximização da Modularidade por Densidade evita esta deficiência ocorra, ou seja, evita que pequenas comunidades não sejam separadas de comunidades maiores na detecção.

A função objetivo da Maximização da Modularidade por Densidade pode ser visualiza na Equação 1, onde C é uma solução de uma detecção de comunidades (uma partição disjunta de vértices), E_c é o conjunto de arestas que conectam dois vértices na comunidade $c \in C$, d_v é o grau do vértice v.

$$D(C) = \sum_{c \in C} \left(\frac{4|E_c| - \sum_{v \in c} d_v}{|c|} \right) \tag{1}$$

Recentemente, trabalhos na literatura relativa ao problema da Maximização da Modularidade por Densidade tem demonstrado eficiência de métodos construtivos, de busca local, e de multinível [Santiago and Lamb 2017a, Santiago and Lamb 2017b]. Dentre estas buscas, duas que se destacam são CM+LNM (Coarsening Merger Local Node Moving) e a HLSMD (Hybrid Local Search Modularity Density) por serem rápidas e encontrarem soluções com alto valor para a função objetivo D. Estas buscas são baseadas em heurísticas bem sucedidas na literatura sobre detecção de comunidades [Rotta and Noack 2011]. No entanto, não se encontra na literatura resultados sobre buscas locais clássicas como a Busca Local Iterada, Busca Local Monótona Randomizada, e a Busca Tabu aplicadas a intensificação de soluções relacionada ao problema.

Com o objetivo de verificar da Busca Local Iterada, Busca Local Monótona

Randomizada e a Busca Tabu, o trabalho reportado neste artigo avaliou-as sobre a intensificação de soluções com pouca identidade com o problema (comunidades com apenas um vértice) e intensificação sobre soluções de alta qualidade obtidas por heurísticas. Os resultados demonstraram que as buscas conseguem melhorar em média 16% as soluções obtidas por buscas como CM+LNM e HLSMD que são consideradas heurísticas estado-da-arte para o problema.

O presente artigo se divide em mais três seções. A próxima seção relata as buscas locais utilizadas no trabalho, relatando detalhes de projeto necessários para replicação das mesmas. Em seguida, uma seção com a análise de resultados é apresentada. Ao final, os principais resultados são apresentados e trabalhos futuros são sugeridos.

2. Buscas Locais Utilizadas

Para a pesquisa reportada neste artigo, foram utilizadas quatro buscas locais, sendo que em apenas uma delas a busca é finalizada quando um ótimo local é encontro. As outras três buscas são uma Busca Local Iterada, uma Busca Local Monótona Randomizada, e uma Busca Tabu. Por questões de facilidade de representação de resultados em tabelas e gráfico, elas serão chamada de BLI, BLMR e BT respectivamente. A outra busca será chamada de BLS (Busca Local Simples).

De forma geral, uma busca local utiliza-se de uma estratégia que especifica vizinhança entre soluções em uma paisagem de otimização. Iterativamente, buscas locais vão substituindo uma solução atual por uma solução vizinha utilizando um determinado critério para selecionar qual vizinho deverá ser assumido. Em uma busca local que pára sobre um ótimo local, seleciona-se iterativamente vizinhos que sejam melhores que a solução atual até que se pare em uma solução sem vizinhos melhores (um ótimo local). Denota-se $N(s)$ o conjunto de vizinhos da solução s.

Neste trabalho, utilizou-se uma estratégia de busca chamada de *1-vizinhança*. Considera-se que uma solução para o problema de Maximização de Modularidade por Densidade um conjunto de subconjunto disjuntos de vértices. Cada subconjunto representa uma comunidade. Por exemplo, para uma rede com os vértices $\{1, 2, 3, 4, 5\}$, uma solução seria $\{\{1, 3, 5\}, \{2, 4\}\}$. Na estratégia de *1-vizinhança*, as soluções vizinhas são todas aquelas que alteram um vértice de sua comunidade para uma outra ou para uma nova comunidade. Para a solução $s = \{\{1, 3, 5\}, \{2, 4\}\}$, duas soluções vizinhas seriam $s' = \{\{3, 5\}, \{1, 2, 4\}\}$ e $s'' = \{\{1, 5\}, \{3\}, \{2, 4\}\}$.

Nas seções que se seguem, as buscas locais utilizadas neste trabalho são descritas.

2.1. Busca Local Simples

A Busca Local Simples (BLS) utilizada neste trabalho é uma implementação de um "Hill Climbing" para a Maximização da Modularidade por Densidade que utiliza a estratégia de *1-vizinhança*.

No Algoritmo 1, pode-se visualizar o BLS. Recebe-se uma solução inicial $s_{inicial}$ e um *max*-Heap-Fibonacci $heap$ que mantém as melhores soluções vizinhas da solução atual s. s_{best} corresponde a melhor solução encontrada e é atualizada caso a solução atual for melhor que a melhor encontrada. Caso contrário, a busca pára e a melhor solução encontrada é retornada. O método $atualizarNovosVizinhos$ atualiza o $heap$ com os novos vizinhos para uma nova solução atual s.

Algorithm 1: Busca Local Simples (BLS)

Input : $s_{inicial}$
$s \leftarrow s_{inicial}$
$s_{best} \leftarrow \emptyset$
$heap \leftarrow iniciarHeapFibonacci(s_{inicial})$
while $D(s_{best}) < D(s)$ **do**
  $s_{best} \leftarrow s$
  $atualizarNovosVizinhos(heap, s)$
  $s \leftarrow melhorVizinho(heap)$
end
return s_{best}

2.2. Busca Local Iterada

Uma Busca Local Iterada (BLI) é uma busca local que sempre tenta melhorar a solução atual s através de uma vizinho s' melhor que s. Quando encontra-se um ótimo local a busca não pára (como acontece com a BLS). Aplica-se uma perturbação sobre a solução s para que passe a intensificar uma solução de outra parte do espaço de busca [Hoos and Stützle 2004].

No contexto deste trabalho, a perturbação é alterar k vértices da solução para uma comunidade aleatória ou uma nova comunidade. Utiliza-se um parâmetro chamado de $\alpha \in [0, 1]$, que define a proporção de vértices atualizados em s, logo $k = |V| \cdot \alpha$, onde V é o conjunto de vértices da rede.

A busca utilizada neste trabalho pode ser visualizada no Algoritmo 2. Neste algoritmo, utiliza-se a busca local presente no Algoritmo 1, chamado através da função $BLS(s)$. Atualiza-se a solução atual s com o ótimo local a cada iteração. Depois, verifica-se se deve ser alterada a melhor solução encontrada s_{best}. No ótimo local, aplica-se a perturbação através da função $perturbarSolucao$. O procedimento é repetido por $maxIt$ iterações.

Algorithm 2: Busca Local Iterada (BLI)

Input : $s_{inicial}, maxIt, \alpha$
$s \leftarrow s_{inicial}$
$s_{best} \leftarrow s_{inicial}$
$it \leftarrow 0$
while $it < maxIt$ **do**
  $s \leftarrow BLS(s)$
  if $D(s) > D(s_{best}$ **then**
    $s_{best} \leftarrow s$
  end
  $s \leftarrow perturbarSolucao(s, \alpha)$
  $it \leftarrow it + 1$
end
return s_{best}

2.3. Busca Local Monótona Randomizada

A Buscal Local Monótona Randomizada (BLMR) atualiza a solução atual s por um vizinho s' melhor que s com uma chance $1 - \alpha$, onde $\alpha \in [0, 1]$ é um parâmetro da busca. Com probabilidade α, troca-se s por um vizinho aleatório. Este procedimento se repete até que uma condição de parada seja satisfeita [Hoos and Stützle 2004].

O Algoritmo 3 apresenta o pseudo-código BLMR utilizada neste trabalho. A solução atual s é atualizada por um vizinho aleatório com a probabilidade α, ou por um vizinho melhor que ela com a probabilidade $1 - \alpha$. Este procedimento é repetido até o número máximo de iterações seja atingido.

Algorithm 3: Busca Local Monótona Randomizada (BLMR)

Input : $s_{inicial}, maxIt, \alpha$
$s \leftarrow s_{inicial}$
$s_{best} \leftarrow s_{inicial}$
$heap \leftarrow iniciarHeapFibonacci(s_{inicial})$
$it \leftarrow 0$
while $it < maxIt$ **do**
  if $random() < \alpha$ **then**
    $s \leftarrow vizinhoAleatorio(s)$
  end
  if $random() \geq \alpha$ **then**
    $s \leftarrow melhorVizinho(heap)$
  end
  $s_{best} \leftarrow argmax\{D(s), D(s_{best})\}$
  $atualizarNovosVizinhos(heap, s)$
  $it \leftarrow it + 1$
end
return s_{best}

2.4. Busca Tabu

A Busca Tabu (BT) é uma busca local que atualiza a solução atual s pela melhor solução vizinha não-tabu. Uma solução tabu possui alguma característica que está presente numa "lista" (não necessariamente uma lista) Tabu [Glover 1986]. Esta lista é sempre atualizada com características da solução recém selecionada para ser a atual. Esta característica pode permanecer na lista Tabu por um número específico de iterações, ou a lista pode ser uma espécie de fila. Neste último, considera-se uma capacidade máxima para a lista Tabu, quando ela está cheia, a característica tabu que está a mais tempo na lista é removida. O principal objetivo da Busca Tabu utilizar esta estratégia é evitar que uma mesma solução seja selecionada mais de uma vez durante a trajetória da busca no espaço de otimização.

No contexto deste trabalho, considerou-se o vértice e a comunidade que geraram uma solução (*1-vizinhança*) atual como característica Tabu. Logo, uma característica Tabu é uma dupla, composta por um vértice e uma comunidade. Uma solução não-tabu é aquela cujo sua composição não tenha um vértice em uma comunidade especificada na lista Tabu. As características permanecem na lista tabu por um período igual a $\alpha \cdot |V|$, onde $\alpha \in [0, 1]$ e V é o conjunto de vértices da rede.

No Algoritmo 4 pode ser visualizado o pseudo-código da BT utilizada no trabalho reportado neste artigo. A cada iteração, seleciona-se a melhor solução vizinha não-tabu. Quando este vizinho é selecionado, atualiza-se a solução atual s, e o vértice v e a comunidade c que correspondem a diferença da solução passada para a recém atualizada são considerados como uma característica Tabu e são armazenados na lista Tabu através da função $adicionarNaTabu$. Após isto, atualiza-se a melhor solução encontrada, caso seja necessário. Então o heap é atualizado com as soluções vizinhas da solução atual e a verifica-se se algum elemento deve ser removido da lista Tabu (ficou na lista por $\alpha \cdot |V|$ iterações.

Algorithm 4: Busca Tabu (BT)

Input : $s_{inicial}, maxIt, \alpha$
$s \leftarrow s_{inicial}$
$s_{best} \leftarrow s_{inicial}$
$heap \leftarrow iniciarHeapFibonacci(s_{inicial})$
$listaTabu \leftarrow \emptyset$ $it \leftarrow 0$
while $it < maxIt$ **do**
  $(s, (v, c)) \leftarrow melhorVizinhoNaoTabu(heap, tabu)$
  $adicionarNaTabu(tabu, (v, c))$
  $s_{best} \leftarrow argmax\{D(s), D(s_{best})\}$
  $atualizarNovosVizinhos(heap, s)$
  $atualizarTabu(tabu)$
  $it \leftarrow it + 1$
end
return s_{best}

3. Resultados e Análise

Esta seção apresenta os resultados das buscas locais e a análise realizada sobre eles. Aseção está dividida em duas de acordo com a metodologia utilizada nos experimentos. Na primeira etapa de experimentos, um conjunto restrito de instâncias foi utilizado para conhecer a relação dos parâmetros das buscas com o valor da função objetivo (D). Nesta mesma etapa, é possível avaliar a distância dos resultados obtidos por uma busca local que pára em ótimos locais em relação aos melhores resultados da literatura. Na segunda etapa de experimentos, os melhores parâmetros são utilizados para intensificar as buscas heurísticas estado-da-arte na literatura.

3.1. Análise de Parâmetros

Nesta seção, a primeira etapa dos experimentos é descrita e seus resultados são apresentados e analisados. Nesta parte dos experimentos, optou-se por utilizar um conjunto restrito de instâncias, geralmente utilizadas na literatura como *benchmarks* para problemas de detecção de comunidades em redes. As soluções iniciais utilizadas nas buscas locais BLS, BLI, BLMR e BT são compostos por comunidades de apenas um vértice cada (*singletons*). Para as buscas locais que não páram sobre ótimos locais (BLI, BLMR e BT), os parâmetros $\{0,1; 0,3; 0,5; 0,7; 0,9\}$ foram experimentados. Os experimentos foram repetidos 30 vezes para cada combinação de rede, busca e parâmetro, por motivos de avaliar os resultados médios nas buscas probabilísticas.

Tabela 1. Valores D médios obtidos em análise preliminar nas buscas locais reportadas neste artigo em relação ao estado da arte na literatura.

Busca	Parâmetro	Karate $\|V\| = 34$	Dolphins $\|V\| = 62$	Lesmis $\|V\| = 77$	Polbooks $\|V\| = 105$	Adjnoun $\|V\| = 112$	Football $\|V\| = 115$	Jazz $\|V\| = 198$
BLS	**Nenhum**	-2,694	5,485	-4,817	-21,486	-9,490	-40,042	12,066
BLI	**0,1**	6,854	8,594	7,695	1,530	-1,251	-7,572	26,831
	0,3	7,013	8,652	6,137	-0,602	-0,977	-4,671	28,154
	0,5	7,154	8,485	6,844	2,473	-2,797	-3,224	27,352
	0,7*	7,163	8,671	7,309	2,156	-2,087	-1,488	29,344
	0,9	7,063	8,541	8,133	0,318	-3,078	-3,460	28,151
BLMR	**0,1**	4,330	6,365	4,479	-5,150	-9,360	-21,837	18,359
	0,3*	6,530	7,979	6,086	1,839	-2,380	-15,725	25,113
	0,5	6,138	8,024	5,961	0,796	-8,610	-18,831	-30,948
	0,7	-9,447	-34,460	-113,916	-153,531	-129,124	-321,479	-1056,272
	0,9	-42,313	-108,813	-315,822	-343,049	-312,882	-594,140	-2366,403
BT	**0,1**	3,083	4,288	1,050	-1,860	-15,090	4,562	27,269
	0,3	1,617	5,985	4,033	1,733	-20,214	4,562	27,044
	0,5	3,083	6,190	3,050	4,478	-2,257	-4,079	27,320
	0,7	3,083	9,247	5,624	5,968	-2,315	15,562	26,510
	0,9*	6,880	8,946	5,624	1,918	-3,496	4,262	32,048
HLSMD	**Nenhum**	**7,544**	8,476	11,072	17,250	4,701	34,014	31,902
CM+LNM	**Nenhum**	7,224	**10,590**	**12,636**	**18,703**	**7,589**	**44,340**	**44,002**

Na Tabela 1, podem ser visualizados os valores da função objetivo (D). Além das buscas produzidas pelo trabalho reportado neste artigo, a tabela traz resultados reportados na literatura para duas buscas consideradas estado-da-arte para o problema da Maximização da Densidade por Modularidade. Elas são a CM+LNM (Coarsening Merger Local Node Moving) de [Santiago and Lamb 2017a] e a HLSMD (Hybrid Local Search for Modularity Density) de [Santiago and Lamb 2017b]. Pode-se observar que as buscas locais HLSMD e CM+LNM obtiveram melhores resultados médios para a função objetivo, com a predominância dos melhores valores pela busca CM+LNM (melhores valores destacados em negrito). Nesta mesma tabela, pode-se visualizar a importância de se evitar os ótimos locais, pois a busca que não realiza esta operação (BLS) obteve os piores resultados. Quanto aos parâmetros das outras buscas apresentadas neste artigo, destaca-se os melhores na tabela com o símbolo "*". Deste modo, para a BLI, o melhor parâmetro foi $0,7$, enquanto para BLMR e BT foram $0,3$ e $0,9$ respectivamente. Estes valores foram usados em um segundo experimento para avaliar a melhoria passível de ser obtida sobre uma das buscas da literatura (CM+LNM).

3.2. Experimentos com Melhores Parâmetros

A segunda etapa de experimentos utilizou a análises e seleção dos parâmetros reportada na Seção "Análise de Parâmetros". Nesta segunda etapa, as buscas locais reportadas neste artigo foram utilizadas para intensificar os resultados obtidos pela melhor instância da literatura. Então, utilizou-se as soluções obtidas pela busca CM+LNM como solução inicial das buscas BLI, BLMR e BT. O principal objetivo deste experimento foi avaliar se é possível melhorar as buscas obtidas na literatura. Nesta etapa de experimentos foram repetidos 30 vezes os experimentos para cada combinação de rede e busca. Ampliou-se a quantidade de redes de *bechmark* utilizadas para ampliar os resultados.

Na Tabela 2, pode-se visualizar os resultados médios dos valores obtidos para a função objetivo (D). Em negrito, os melhores valores são apresentados. Na última linha (identificada como "% Melhoria"), a melhoria média obtida nos experimentos realizados,

calculada através da fórmula

$$\frac{D_{busca} - D_{CM+LNM}}{D_{CM+LNM}} \cdot 100, \tag{2}$$

onde D_{busca} é o valor médio D obtido por BLI, BLMR ou BT, e D_{CM_LNM} é o valor obtido pela busca CM+LNM. Pode-se observar na tabela que as três buscas locais BLI, BLMR e BT obtiveram melhoria superior a 16% de resultados de buscas estado-da-arte na literatura, evidenciando o importante papel das buscas locais na intensificação para o problema da Maximização da Modularidade por Densidade.

Tabela 2. Valores D médios obtidos nas buscas locais reportadas neste artigo em relação ao estado da arte na literatura.

Rede	#Vértices	#Arestas	BLI	BLMR	BT	CM+LNM	HLSMD
Strike	24	38	8,374	8,465	**8,861**	7,544	7,224
Galesburg f	31	63	**8,286**	**8,286**	**8,286**	**8,286**	7,329
Galesburg d	31	67	**6,841**	**6,841**	**6,841**	**6,841**	5,691
Karate	34	78	7,575	**7,605**	6,824	6,036	7,544
Korea1	35	69	10,663	**10,669**	10,557	10,554	9,930
Korea2	35	84	10,835	**10,915**	10,682	10,014	10,572
Mexico	35	117	8,579	**8,670**	8,558	8,557	7,329
Sawmill	36	62	8,305	**8,414**	8,221	7,908	7,343
Dolphins	62	159	10,756	11,230	**11,239**	8,476	10,590
Lesmis	77	479	**12,995**	12,854	12,636	11,072	12,636
Polbooks	105	441	**20,416**	19,328	18,703	17,250	18,703
Adjnoun	112	425	7,589	**7,598**	7,589	4,701	7,589
Football	115	613	44,305	44,310	**44,340**	34,014	44,340
Jazz	198	2742	**49,339**	46,951	44,003	31,902	44,002
Email	1133	5451	**32,974**	27,108	26,633	23,039	-36,977
%Melhoria →			**20,21%**	18,23%	16,11%		

Na Figura 1, pode-se observar os tempos demandados em milisegundos para que as buscas BLI, BLMR e BT obtivessem os resultados reportados na Tabela 2. Pode-se observar que BLI demandou menos tempo que as demais. Como esta busca obteve a melhor melhoria em relação aos resultados de CM+LNM, se mostrou a de maior benefício nesta análise.

4. Considerações Finais

Este trabalho teve como objetivo avaliar algumas buscas locais estocásticas e tentar identificar se elas geram melhorias em soluções obtidas por estado da arte na literatura. Foram analisadas a Busca Local Iterada, a Busca Local Monótona Randomizada e a Busca Tabu. Em um primeiro momento, identificou-se parâmetros com as buscas partindo de soluções onde cada vértice estava isolado em uma comunidade. No segundo experimento, as soluções iniciais das buscas locais partiram dos resultados da busca estado-da-arte CM+LNM ([Santiago and Lamb 2017a]).

As buscas em geral conseguiram melhorar em média 16% da avaliação das soluções obtidas por CM+LNM. A Busca Local Iterada apresentou uma melhora média superior a 20% em relação aos demais experimentos, além de apresentar o menor tempo

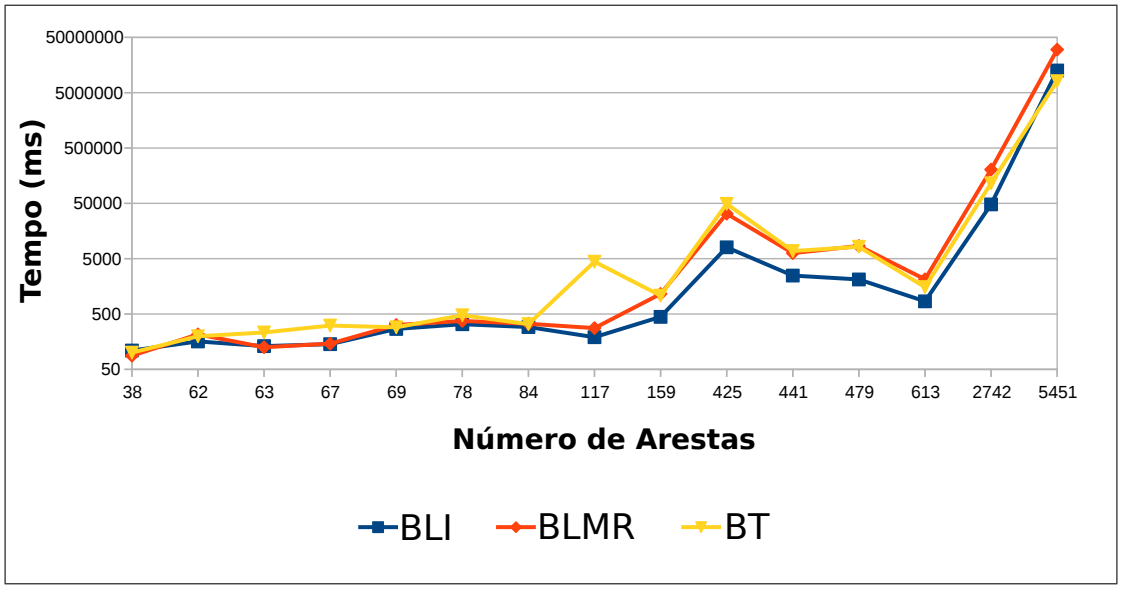

Figura 1. Tempo em milisegundos demandado pelas buscas BLI, BLMR e BT na segunda etapa de experimentos.

médio. Estes resultados reforçam que as buscas locais tem um papel importante na busca por soluções da Maximização da Modularidade por Densidade.

Referências

Calderoni, F., Brunetto, D., and Piccardi, C. (2017). Communities in criminal networks: a case study. *Social Networks*, 48:116–125.

Ferrara, E., De Meo, P., Catanese, S., and Fiumara, G. (2014). Detecting criminal organizations in mobile phone networks. *Expert Systems with Applications*, 41(13):5733–5750.

Fortunato, S. and Barthélemy, M. (2007). Resolution limit in community detection. *Proceedings of the National Academy of Sciences of the United States of America*, 104(1):36–41.

Fortunato, S. and Hric, D. (2016). Community detection in networks: A user guide. *Physics Reports*, 659:1–44.

Girvan, M. and Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(12):7821–7826.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13(5):533–549.

Guimera, R. and Amaral, L. (2005). Functional cartography of complex metabolic networks. *Nature*, 433(February):895–900.

Hoos, H. and Stützle, T. (2004). *Stochastic Local Search: foundations and applications*. Morgan Kaufmann.

Li, Z., Zhang, S., Wang, R.-S., Zhang, X.-S., and Chen, L. (2008). Quantitative function for community detection. *Physical Review E*, 77(3):036109.

Newman, M. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., and Parisi, D. (2004). Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(9):2658–2663.

Rotta, R. and Noack, A. (2011). Multilevel local search algorithms for modularity clustering. *Journal of Experimental Algorithmics*, 16(2):2.1.

Santiago, R. and Lamb, L. C. (2017a). Efficient modularity density heuristics for large graphs. *European Journal of Operational Research*, 258(3):844–865.

Santiago, R. and Lamb, L. C. (2017b). Efficient Quantitative Heuristics for Graph Clustering. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 117–118, Berlin. ACM New York.

Xie, J., Kelley, S., and Szymanski, B. K. (2013). Overlapping community detection in networks. *ACM Computing Surveys*, 45(4):1–35.