UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

THIAGO BELL

Laboratório 4 - Pareamento

Porto Alegre, 2017

1.1 Tarefa

Implementar o algoritmo de Hopcroft-Karp para o problema de pareamento de grafos bipartidos.

1.2 Implementação

Os algoritmos foram implementados em C++. A implementação de grafo usa listas de adjacências. Cada vértice contém uma variável com o indíce do vértice com o qual está pareado. Também contém uma variável para distância usada na execução do algoritmo. Essa última é desnecessária para ambos os subgrafos, pois, apenas se calculará a distância para um deles. Não se modificou isso para manter a simplicidade.

1.3 Ambiente de Teste

Os experimentos foram realizados usando um processador Intel i7 2600k acompanhado de 8 GiB de RAM. O sistema operacional utilizado foi Ubuntu Linux 16.10.

1.4 Análise de Complexidade do Algoritmo de Hopcroft-Karp

A complexidade teórica do algoritmo foi comparado com resultados experimentais. Conclui-se que a implementação respeita as previsões teóricas.

1.4.1 Complexidade

A complexidade do algoritmo é de:

$$O(sqrt(n)(n+m))$$

Para demonstrar que a implementação respeita esse limite, fez-se dois testes. Um fixando o número de arestas e variando o de vértices. Outro, fixando o de vértices e variando o de arestas.

Também, o número de fases do algoritmo é limitado por

$$O(sqrt(n))$$

Em todas os testes, esse limitante superior foi respeitado.

1.4.2 Fixando o Número de Arestas

Fixou-se o número de arestas em aproximadamente 1.67 milhões. O número de vértices foi variado entre 2^{12} a 2^{15}. Comparando-se o tempo de execução e o custo teórico esperado, conclui-se a adesão da implementação a complexidade prevista.

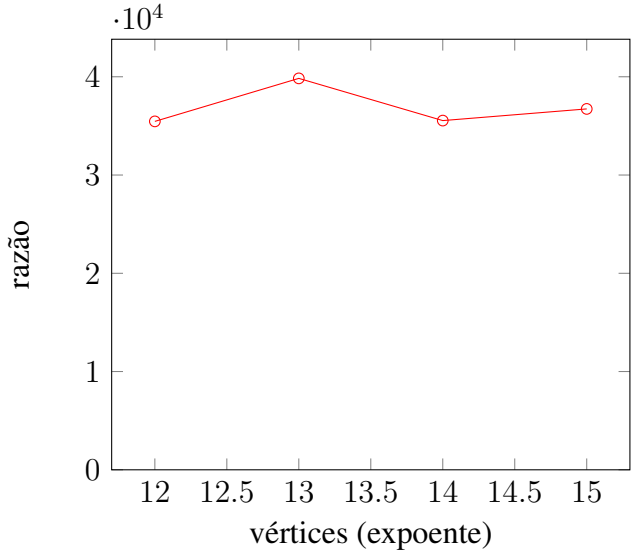

Figura 1.1: Mostra a razão entre o custo teórico esperado e o tempo de execução

1.4.3 Fixando o Número de Vértices

Fixou-se o número de vértices em 16384. O número de arestas foi variado entre 838971 a 6710515. Comparando-se o tempo de execução e o custo teórico esperado, conclui-se a adesão da implementação a complexidade prevista.

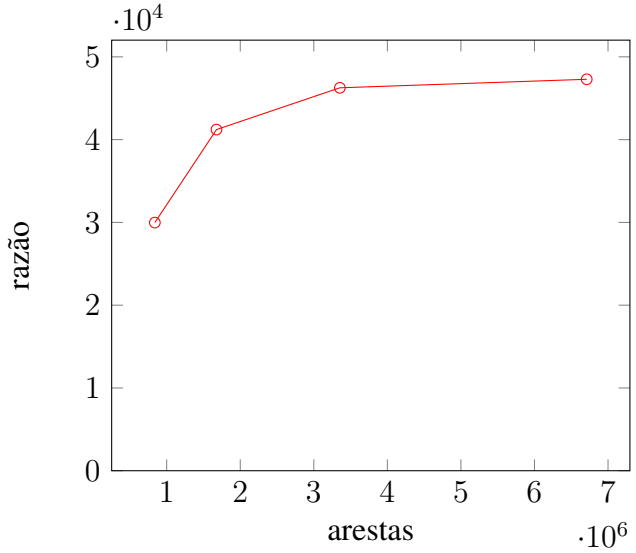

Figura 1.2: Mostra a razão entre o custo teórico esperado e o tempo de execução

1.5 Comparação entre Hopcroft-Karp e Redução a Maxflow

Comparou-se os tempos de execução entre o algoritmo implementado e a execução do algoritmo de Ford-Fulkerson com estratégia de caminho mais gordo com a mesma instância reduzida ao problema do fluxo máximo. O algoritmo de Hopcroft-Karp foi significativamente mais rápido. Para 4096 vértices e 1679005 arestas, o tempo de execução para o primeiro foi de $13.8ms$ (repetiu-se a execução 300 vezes para evitar problemas de precisão de relógio e obter uma medida precisa). Para o Ford-Fulkerson o tempo foi

de $94.3s$. Dessa forma, o algoritmo de Hopcroft-Karp tem uma performance ordens de grandeza melhor.

1.6 Conclusão

Implementou-se o algoritmo de Hopcroft-Karp . Verificou-se que a implementação respeita a complexidade do algoritmo. Além disso, esse algoritmo tem melhor performance que o de Fork-Fulkerson com uma mesma instância reduzida ao problema de fluxo máximo.