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

Pablo Felipe Leonhart

Fluxo s-t máximo
(Trabalho 3)

1. Tarefa

- Implementar o algoritmo de Ford-Fulkerson com a estratégia do “caminho mais gordo” (fattest path) s-t.
- Verificar a complexidade do algoritmo experimentalmente.

2. Solução

Foi implementado o algoritmo Ford-Fulkerson utilizando um grafo com a estrutura de dados definida em uma lista de lista, onde cada vértice ‘u’ possui a lista de seus vértices ‘v’ adjacentes, os quais recebem uma aresta de ‘u’. A estratégia do caminho mais gordo foi implementada ao utilizar o algoritmo de Dijkstra com o uso do Hollow Heap como fila de prioridade, desenvolvidos nos trabalhos anteriores.

Para esse problema, o Hollow Heap foi ajustado para manter o maior valor na raiz. O algoritmo de Dijkstra percorre o grafo em busca do caminho que conceder maior capacidade de fluxo. Ao atingir o vértice de destino (t), o algoritmo deixa o caminho percorrido salvo em um vetor auxiliar, para que através deste possa ocorrer a atualização das capacidades no grafo. Enquanto existir um caminho de ‘s’ para ‘t’ o fluxo continua sendo enviado, e quando não existir mais caminhos com capacidade maior que zero, o algoritmo encerra e obtém-se o fluxo máximo.

3. Ambiente de teste

Os resultados foram obtidos numa Intel Core i7-5500U, com 4 processadores de 2.40 GHz e 8 GB de RAM.

4. Resultados

Os testes realizados foram com base no gerador de grafos disponibilizado, new_washington.c. Para cada tipo de grafo, o algoritmo de fluxo máximo foi executado 30 vezes, e então foram mensurados: a corretude do resultado, o fluxo máximo obtido, o número de iterações, isto é, a quantidade de caminhos selecionados para enviar o fluxo, o tempo de execução e o consumo de memória. Os dados estão disponíveis na Tabela 1:

Tabela 1: Resultados obtidos na execução do Ford-Fulkerson para diferentes grafos

Nome	Corretude (%)	Fluxo máximo	Iterações	Tempo (μs)	Memória (kB)
Mesh	-	-	-	-	-
Random level	-	-	-	-	-
Random 2-level	-	-	-	-	-
Matching	100	100	101	39892	4163
Square Mesh	100	300000	450	23169557	212460
BasicLine	-	-	-	-	-
ExpLine	100	7130709	140	17725	3635
DExpLine	100	3087340	75	9117	3371
DinicBad	100	101	3	399	3111
GoldBad	100	100	101	49849	4691
Cheryian	100	200	21	5930	3636

Para os grafos Mesh, Random level, Random 2-level e BasicLine o algoritmo ficou em loop, não obtendo assim resultados satisfatórios. Acredita-se que algum detalhe na implementação do grafo residual possa ter causado o problema.

5. Conclusão

O algoritmo apresentou um bom desempenho ao adotar a estratégia do caminho mais gordo, com o uso do Dijkstra. Apesar de não ter obtido sucesso em todos tipos de grafos solicitados, naqueles em que executou, apresentou um bom resultado, com baixo tempo para atingir o resultado ideal, se comparado a outras técnicas como o push-relabel.