

6

Experimentos Computacionais

Para os experimentos computacionais, os algoritmos foram programados em linguagem C. Foram utilizados os arquivos de entrada e a estrutura de dados do algoritmo implementado por Skorobohatyj [24]. O algoritmo de fluxo máximo também foi aproveitado, sendo este o método de Goldberg e Tarjan [7].

Os algoritmos testados foram:

- A1_AGE: Algoritmo A1 utilizando a função *Árvore Geradora Estrela*;
- A1H_AGE: Algoritmo A1 com a *Heurística de Contração de Grafo* e utilizando a função *Árvore Geradora Estrela*;
- A1_DFS: Algoritmo A1 utilizando a função *Árvore Geradora DFS*;
- A1H_DFS: Algoritmo A1 com a *Heurística de Contração de Grafo* e utilizando a função *Árvore Geradora DFS*;

- A2_AGE: Algoritmo A2 utilizando a função *Árvore Geradora Estrela*;
- A2HH_AGE: Algoritmo A2 com a *Heurística de Contração de Grafo* e a *Heurística de Adição de Arestas*, utilizando a função *Árvore Geradora Estrela*;
- A2_DFS: Algoritmo A2 utilizando a função *Árvore Geradora DFS*;
- A2HH_DFS: Algoritmo A2 com a *Heurística de Contração de Grafo* e a *Heurística de Adição de Arestas*, utilizando a função *Árvore Geradora DFS*;

- Gus: Algoritmo de Gusfield implementado por Skorobohatyj (2011);
- Algo: Algoritmo A1 sem o uso de contrações e utilizando a função *Árvore Geradora Estrela*.

Os resultados estão sintetizados nos gráficos das Figuras 6.1, 6.2 e 6.3, nos quais os testes ocorreram nas instâncias de 10, 45 e 160 nós.

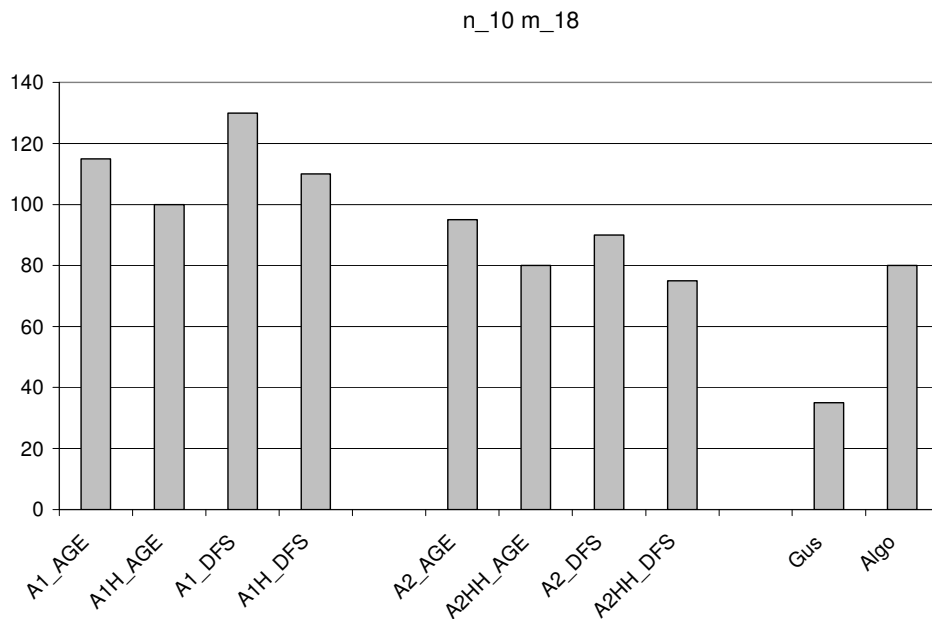


Figura 6.1: Algoritmo x tempo (μ s) de execução na instância de 10 nós e 18 arestas.

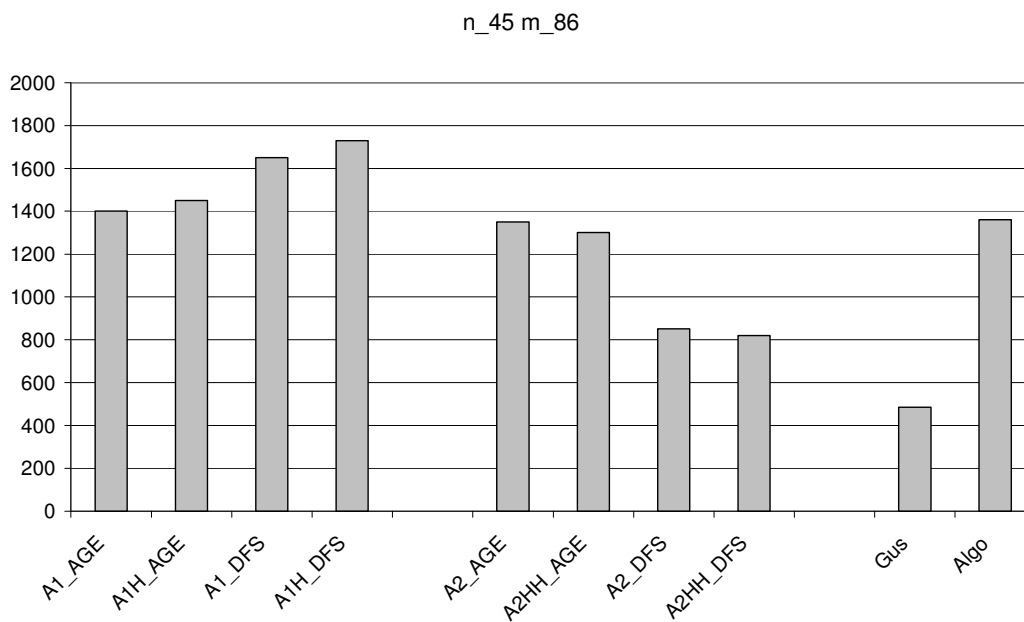


Figura 6.2: Algoritmo x tempo (μ s) de execução na instância de 45 nós e 86 arestas.

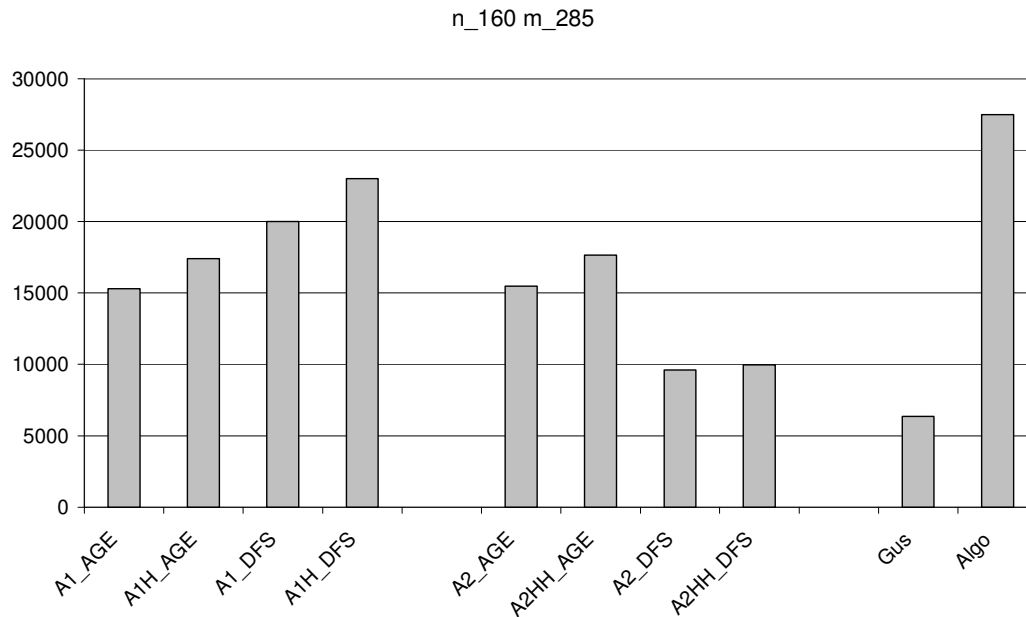


Figura 6.3: Algoritmo x tempo (μ s) de execução na instância de 160 nós e 285 arestas.

Analisando os resultados, chegou-se às seguintes conclusões:

1. As contrações geram maior eficiência para o algoritmo, principalmente em instâncias grandes, visto que, no exemplo de 160 nós, A1_AGE executou em tempo bem menor que Algo;
2. Em A1, a função AGE se mostrou mais eficaz que DFS, possivelmente por propiciar caminhos $i-j$ mais curtos;
3. Em A2, a função DFS se mostrou mais eficaz que AGE, possivelmente por propiciar caminhos $i-j$ maiores;
4. No geral, as heurísticas não obtiveram sucesso em aumentar a eficiência dos algoritmos. Foram testados diversos valores para os tomadores de decisão em ambas as heurísticas, mas em nenhum caso obteve-se um tempo de execução mais baixo.
5. A2 se mostrou, aparentemente, com uma melhor performance que A1, logo, é esperado que a redução no número de iterações aumenta a eficiência do algoritmo.
6. O algoritmo com melhor desempenho foi A2 com a função *Árvore Geradora DFS*, com ou sem heurística. Na instância de 160 nós, A2_DFS executou 391 vezes o algoritmo de fluxo máximo e Gus 159, em uma

proporção de 2,5. No entanto, a proporção dos tempos de execução do algoritmo é de apenas 1,5, demonstrando que executar o algoritmo de fluxo máximo em redes contraídas nos permite executá-lo mais vezes.

7. Principal: o método e todas as suas variações não se mostraram com um bom desempenho em comparação ao algoritmo de Gusfield. Conclui-se, então, que o melhor é proceder inversamente: elaborar um algoritmo que retorne grafos "adaptados" ao método, ou seja, gerar instâncias em que o método atue com maior performance. Por esta razão não foram realizados muitos testes.