4
Algoritmo

O algoritmo proposto neste trabalho está baseado na teoria da análise de sensibilidade já exposta, e nas respectivas técnicas de Gomory e Hu e de Gusfield, quais sejam, contração e construção de árvores. Utilizar as informações de uma árvore de cortes existente para computar a próxima é a idéia central do algoritmo, permitindo, assim, que os fluxos máximos/cortes mínimos sejam aplicados sempre a redes menores que a original.

Para calcular o fluxo máximo entre todos os pares de nós de uma rede $G = (V, E)$ não direcionada e com capacidades nas arestas, o algoritmo inicia gerando uma "espécie" de solução inicial, técnica utilizada por muitas heurísticas. No caso, a solução inicial será uma árvore geradora (*spanning tree*) T de G, ou seja, um subgrafo de G contendo todos os seus vértices e que, também, é uma árvore. Esta poderá ser qualquer árvore geradora de G, não necessitando ser mínima ou máxima.

Chama-se o grafo T de G_0. Sobre sua árvore de cortes (cut tree) CT_0, como G_0 é uma árvore, ou seja, entre qualquer par de nós há somente um único caminho, CT_0 é o próprio grafo G_0.

Considera-se E' o conjunto das arestas contidas em G, mas não contidas em G_0. É adicionada, então, uma aresta $e = (i, j)$ pertencente a E' ao grafo G_0, transformando-o em G_1.

Para o cálculo de CT_1 lançamos mão dos seguintes lema e teorema desta dissertação:

- Lema 3: existe uma CT_1 idêntica a CT_0 exceto pelo caminho i-j.

- Teorema 4: são necessárias apenas $(P-1)$ aplicações do algoritmo de fluxo máximo/corte mínimo, sendo P é o número de vértices no caminho i-j.

De acordo com a demonstração do teorema 4, que se utiliza do método de Gomory e Hu, as $(P-1)$ execuções do algoritmo de fluxo máximo/corte mínimo poderão ser aplicadas a um grafo contraído de G_1.

De posse das extremidades *i* e *j* da aresta adicionada, o algoritmo identifica o caminho *i-j* em CT_0. A seguir, mapeia todas as subárvores T_a^b de CT_0. Cada subárvore será contraída a somente um vértice no grafo em que será aplicado o algoritmo de fluxo máximo. Identificado o caminho e mapeadas as subárvores, gera-se, então, utilizando as técnicas de contração do algoritmo de Gomory e Hu, o grafo contraído de G_1.

Por fim, o algoritmo, fazendo uso da rotina algorítmica de Gusfield, calcula o fluxo máximo entre todos os pares de nós pertencentes ao caminho *i-j* de CT_0. Vale ressaltar que o algoritmo de fluxo máximo é sempre aplicado no grafo corrente contraído, do que se obtém as duas informações seguintes: o valor do fluxo máximo entre o par de nós e o lado em que se encontra cada um dos nós do caminho *i-j* em relação ao corte mínimo.

A árvore de cortes, resultante do algoritmo de Gusfield, substituirá, então, o caminho *i-j* em CT_0. Desta forma, obtém-se CT_1.

Em um laço **para**, estes procedimentos são executados até que todas as arestas de *E*' tenham sido adicionadas. Com as adições, serão gerados G_2, G_3, ..., G_y e suas respectivas árvores de corte CT_2, CT_3, ..., CT_y, onde $y = (m - n + 1)$ sendo *m* o número de arestas e *n* o número de nós do grafo *G*. O valor de *y* é simples de ser encontrado visto que o número de arestas de G_0 é sempre $n-1$. Note que apenas uma aresta é adicionada a cada iteração.

Ao término do laço, a árvore de cortes remanescente é CT_y, ou, apenas, *CT*, que é, portanto, a resposta ou a solução final do problema. A figura 4.1 exibe um pseudocódigo do algoritmo (A1).

procedimento A1 (G);

1 $T \leftarrow$ crie uma árvore geradora (*spanning tree*) do grafo G;

2 $G_0 = T$;

3 $CT_0 = G_0$;

4 $E' \leftarrow$ conjunto de arestas que estão em G mas que não estão em G_0;

5 **para** $e_k \in E'$ onde $k = 1, ..., m-n+1$ **faça**

6 Adicione e_k ao grafo G_{k-1} criando G_k;

7 Identifique caminho i-j em CT_{k-1};

8 Mapeie subárvores T_a^b em CT_{k-1};

9 Gere grafo contraído de G_k;

10 Calcule, com algoritmo de Gusfield, árvore de cortes dos nós do caminho i-j no grafo contraído;

11 Substitua caminho i-j pela árvore de cortes em CT_{k-1}, gerando CT_k;

12 **finalize para**

13 **retorne** CT;

finalize A1;

Figura 4.1: Pseudocódigo do algoritmo A1.

A fim de ilustrar com maior clareza o algoritmo, segue um exemplo de sua aplicação. Seja G a rede da Figura 4.2.

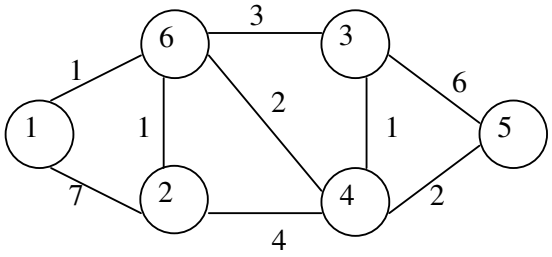

Figura 4.2: Rede G na qual será aplicado o algoritmo A1.

A Figura 4.3 nos mostra o resultado do primeiro passo do algoritmo: a seleção de uma árvore geradora (*spanning tree*) T de G, nomeada G_0, e que, também, é a sua própria árvore de cortes CT_0.

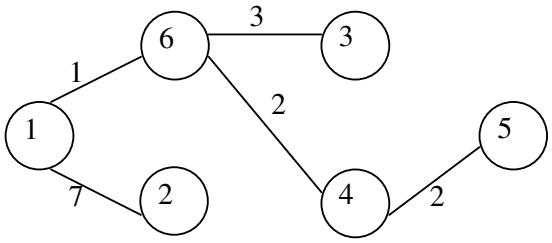

Figura 4.3: Árvore geradora T de G, nomeada G_0, e que, também, é a sua própria árvore de cortes CT_0.

Selecionada a árvore geradora, o conjunto E' será composto pelas arestas $E' = \{(2,6), (2,4), (3,4), (3,5)\}$. As mesmas serão adicionadas aos grafos na ordem em que se encontram no conjunto.

Ao adicionarmos a aresta (2,6), linha 6 do pseudocódigo, o algoritmo identifica o caminho *i-j* (linha 7) em CT_0 como sendo formado pelo nós 2, 1 e 6. Duas subárvores são então mapeadas: a composta pelo nó 3 e a composta pelos nós 4 e 5 (linha 8). Esta última, por ter mais de um nó, será contraída para o cálculo do algoritmo de fluxo máximo. G_1 e G_1 contraído estão ilustrados nas Figuras 4.4 e 4.5, respectivamente.

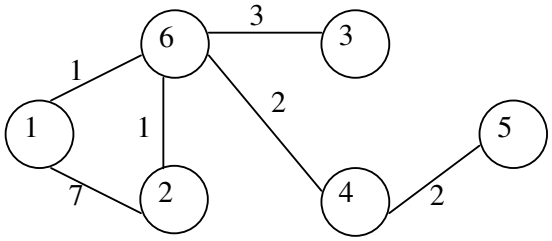

Figura 4.4: Grafo G_1.

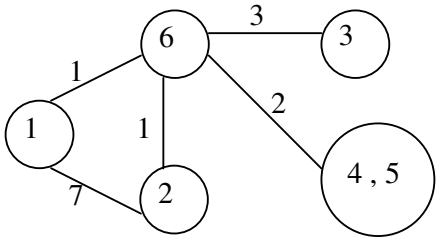

Figura 4.5: G_1 contraído.

Em G_1 contraído serão aplicados dois algoritmos de fluxo máximo, estes necessários para construir, por meio da rotina de Gusfield, a árvore de cortes que substituirá o caminho 2-1-6 em CT_0. Os nós origem e destino dos algoritmos de fluxo máximo sairão do conjunto $V = \{1, 2, 6\}$. A Figura 4.6 nos mostra o resultado.

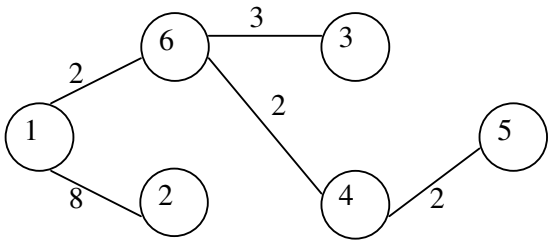

Figura 4.6: CT_1.

Observe que CT_1 é a árvore de cortes de G_1. Termina aqui a execução da primeira iteração do laço **para**.

Pela ordem, a próxima aresta a ser adicionada é (2,4). Com suas extremidades, o caminho *i-j* em CT_1 será 2-1-6-4. Neste caso não haverá contrações possíveis, pois as duas subárvores existentes possuem apenas um nó cada: subárvore nó 3 e subárvore nó 5. Portanto, os fluxos máximos serão computados no próprio G_2. As Figuras 4.7 e 4.8 ilustram os grafos desta segunda iteração.

A terceira aresta incluída é (3,4), transformando G_2 em G_3. G_3, com somente uma aresta a menos, é quase igual ao grafo original *G*. Nesta iteração: caminho *i-j* em CT_2 é 3-6-2-4; nenhuma subárvore a ser contraída, logo, não há G_3 contraído; três algoritmos de fluxo máximo computados em G_3 para construção da árvore de cortes entre os nós do caminho *i-j*; substituição do caminho *i-j* pela árvore de cortes calculada por Gusfield. Figuras 4.9 e 4.10 demonstram G_3 e CT_3 respectivamente.

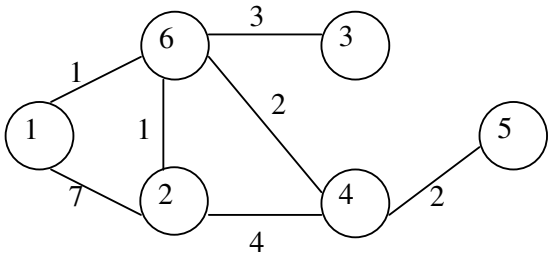

Figura 4.7: G_2.

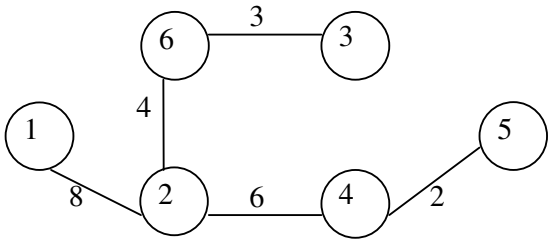

Figura 4.8: CT_2.

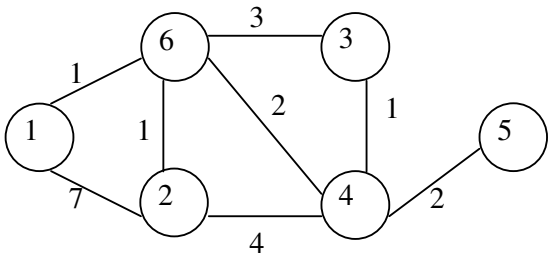

Figura 4.9: G_3.

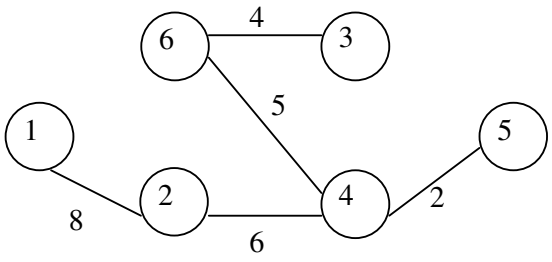

Figura 4.10: CT_3.

A aresta (3,5) é a quarta e última a ser adicionada. Com ela, G_3 se torna G_4, ou simplesmente G, o grafo original. As Figuras 4.11, 4.12 e 4.13 ilustram os procedimentos executados nesta iteração. Observa-se que uma contração é possível e, então, os algoritmos de fluxo máximo são computados em G_4 contraído. *CT* é a solução final do problema.

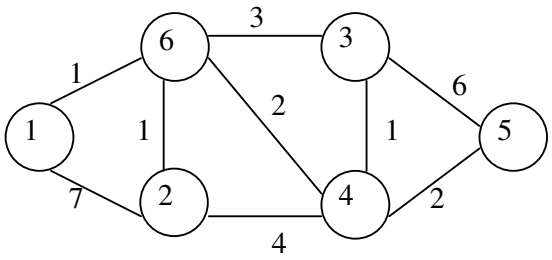

Figura 4.11: G_4 ou G.

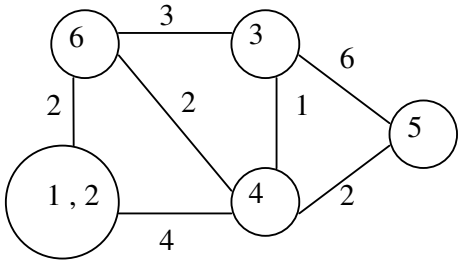

Figura 4.12: G_4 contraído.

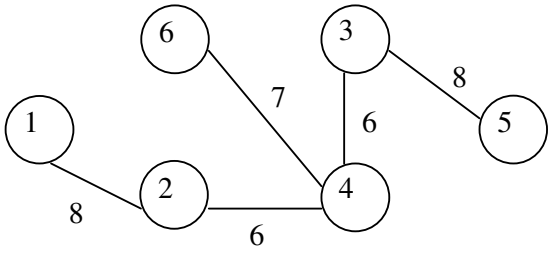

Figura 4.13: CT_4 ou CT. Resultado Final.

4.1
Algoritmo A2

O algoritmo A2 é igual ao anterior A1, a não ser por uma diferença: enquanto que em A1, a cada iteração do laço **para,** é adicionada somente uma aresta, em A2 é possível adicionar várias arestas na mesma iteração.

O teorema 5 demonstra que esta modificação é válida, não alterando o resultado final do algoritmo. Ressalta-se, apenas, a necessidade de todas as extremidades das arestas adicionadas em uma mesma iteração estarem contidas no caminho *i-j*. Desta forma, o algoritmo procederá primeiramente, adicionando uma aresta ao grafo; em seguida, identificando o caminho *i-j*, e por último, procurando novas arestas a adicionar que satisfaçam o requisito acima. A Figura 4.14 é um pseudocódigo do algoritmo A2. A modificação encontra-se na linha 9. Observe que foi necessário incluir o tomador de decisão **se** na linha 6 do algoritmo para não haver duplicidade de arestas nos grafos.

procedimento A2 (G);

1 $T \leftarrow$ crie uma árvore geradora (*spanning tree*) do grafo G;
2 $G_0 = T$;
3 $CT_0 = G_0$;
4 $E' \leftarrow$ conjunto de arestas que estão em G mas que não estão em G_0;
5 **para** $e_k \in E'$ onde $k = 1, ..., m-n+1$ **faça**
6 **se** e_k não está incluído no grafo G_{k-1}
7 Adicione e_k ao grafo G_{k-1} criando G_k;
8 Identifique caminho i-j em CT_{k-1};
9 Adicione e_w caso suas extremidades estejam contidas
no caminho i-j, para $w = k + 1, k + 2, ..., m-n+1$;
10 Mapeie subárvores T_a^b em CT_{k-1};
11 Gere grafo contraído de G_k;
12 Calcule, com algoritmo de Gusfield, árvore de cortes dos nós
do caminho i-j no grafo contraído;
13 Substitua caminho i-j pela árvore de cortes em CT_{k-1} gerando CT_k;
14 **finalize se**
15 **finalize para**
16 **retorne** CT;
finalize A2;

Figura 4.14: Pseudocódigo do algoritmo A2.