O problema da conectividade dinâmica em grafos

Gabriel de Russo e Carmo

Instituto de Matemática e Estatística da Universidade de São Paulo

O problema

No problema da conectividade dinâmica em grafos, estamos interessados em manter um grafo que está sujeito a atualizações e consultas. Uma atualização é uma adição ou remoção de aresta, enquanto uma consulta deseja saber se dois vértices estão conectados.
Concretamente, desejamos uma estrutura que permite as seguintes operações:

- GrafoDinâmico(n): devolve um grafo G com n vértices e nenhuma aresta.
- Conectado(G, v, w): devolve Sim se v, w estão conectados em G e Não caso contrário.
- Adicione(G, vw): adiciona a aresta vw em G. Supõe que vw não está em G.
- Remova(G, vw): remove a aresta vw de G. Supõe que vw está em G.

Objetivos

1. Entender o problema e apresentar uma solução de forma detalhada. Grande parte dos artigos que tratam do problema são de alto nível e deixam de lado detalhes importantes da implementação.
2. Implementar a solução numa linguagem de programação moderna.
3. Testar seu desempenho na prática.

Florestas geradoras

Uma **floresta geradora** de um grafo é um subgrafo que contém uma árvore geradora de cada um de seus componentes. Equivalentemente, uma floresta geradora é um subgrafo acíclico maximal. A figura 1 ilustra o conceito. Arestas pontilhadas são arestas do grafo que não estão na floresta.

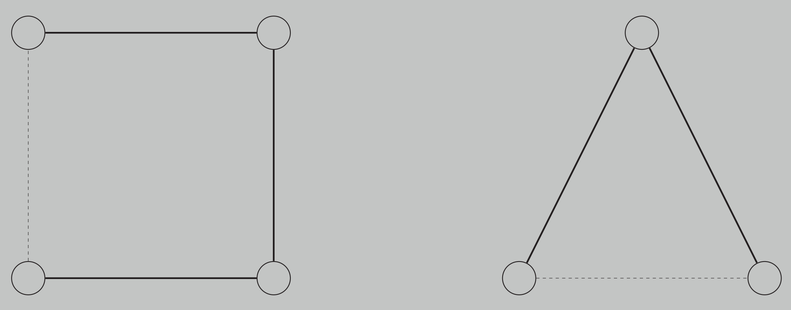

Figura 1: Exemplo de floresta geradora.

Estrutura de HDT

Para resolver o problema da conectividade dinâmica num grafo G, mantemos diversas florestas geradoras de G [1]. Intuitivamente, desejamos uma floresta geradora F de G para fazer consultas de conectividade. Chamamos as arestas que estão numa floresta de **arestas de árvore**. Todas as outras arestas são chamadas de **arestas reservas**. Ao se remover uma aresta de árvore, a maior parte do esforço está na procura de uma **arestas substituta**, isto é, uma aresta reserva que pode substituir a aresta de árvore recém-removida nas floresta.

Implementação

A estrutura foi implementada em C++ e seu código está disponível em https://github.com/gabrielrussoc/mac499.
O diagrama da figura 2 ilustra a representação da estrutura.

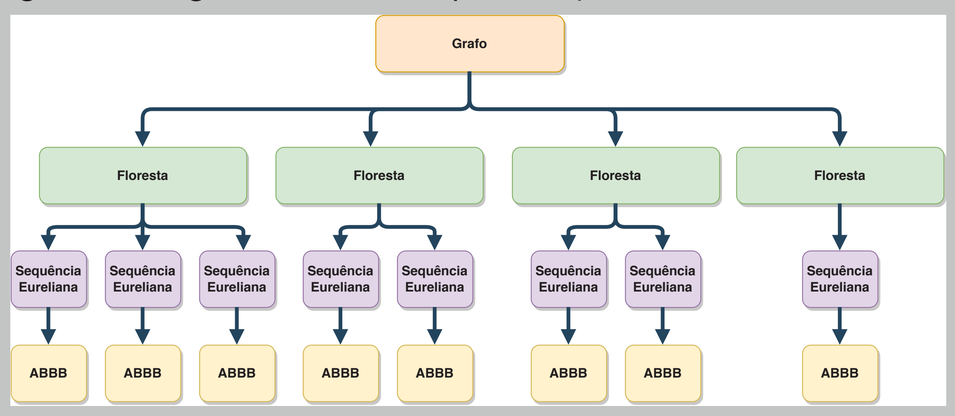

Figura 2: Diagrama da representação da estrutura

Florestas dinâmicas

Para resolver o problema da conectividade dinâmica restrito à florestas, usamos o fato de que cada componente da floresta é uma árvore.
Definimos a **sequência eureliana** [2] de uma árvore T enraizada num vértice r por uma trilha eureliana de T começando em r.
Uma possível sequência eureliana da árvore da figura 3 é
`AA AB BB BE EE EB BA AC CC CF FF FC CG GG GC CA AD DD DA`.

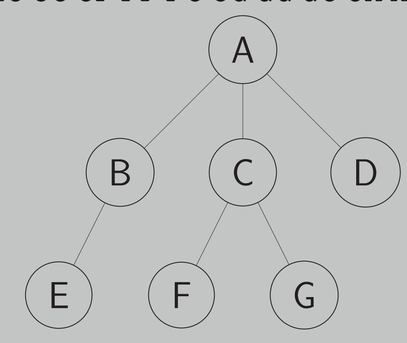

Figura 3: Árvore enraizada no vértice A

Tanto adicionar uma aresta que liga duas árvores disjuntas quanto remover uma aresta de uma árvore se reduz a um número constante de concatenações e fatiamentos de suas sequências eurelianas.

Sequências

Representamos sequências por árvores de busca binária implícitas. Essas árvores tem como chave as posições dos elementos em ordem. Uma possível árvore da sequência `B R A S I L` está ilustrada na figura 4.

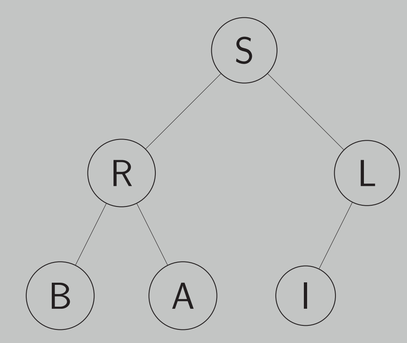

Figura 4: Árvore de busca binária da sequência `B R A S I L`.

Usando árvores balanceadas, como por exemplo árvores rubros-negras, é possível concatenar e fatiar sequências em tempo $O(\lg n)$ [3].

Desempenho

A eficiência da estrutura foi testada na prática. A figura 5 mostra o desempenho médio das operações Adicione e Conectado.

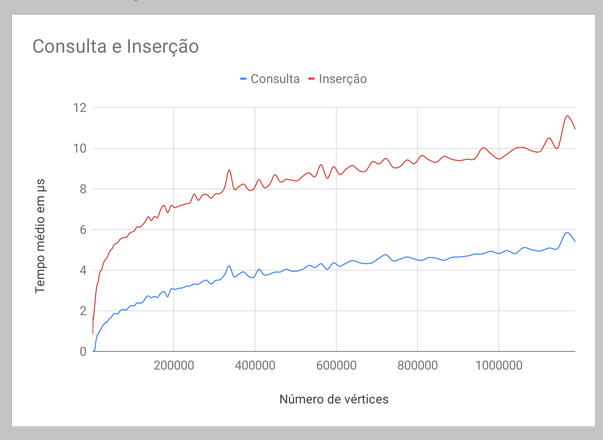

Figura 5: Desempenho médio das operações Adicione e Conectado.

Referências

[1] Jacob Holm, Kristian de Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *J. ACM*, 48(4):723–760, July 2001.

[2] Monika Rauch Henzinger and Valerie King. Randomized dynamic graph algorithms with polylogarithmic time per operation. In *Proceedings of the Twenty-seventh Annual ACM Symposium on Theory of Computing*, STOC '95, pages 519–527, New York, NY, USA, 1995. ACM.

[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.

Veja mais

A monografia e outras informações estão disponível em https://linux.ime.usp.br/~gabrielrc/mac0499/.