Conexidade Dinâmica

Projeto de Pesquisa para Iniciação Científica

Orientadora: Cristina Gomes Fernandes
Aluno: Daniel Angelo Esteves Lawand

Este projeto de pesquisa acompanha a requisição de bolsa de
Iniciação Científica para o aluno Daniel Angelo Esteves Lawand.

1 Introdução

O objetivo deste projeto é o estudo e implementação de estruturas de dados e algoritmos para problemas de conexidade em grafos num contexto onde o grafo pode sofrer modificações [3].

O problema central será o de manter informações sobre as componentes conexas do grafo, ou mais exatamente, o problema de responder eficientemente consultas do tipo:

- Quantas componentes têm o grafo?
- Os vértices u e v estão numa mesma componente do grafo?

Num contexto estático, em que o grafo não sofre alterações, há um algoritmo linear que determina as componentes conexas do grafo e responde de maneira ótima consultas como estas.

No entanto, se o grafo em questão está sendo alterado, podendo ganhar ou perder arestas, o problema torna-se mais desafiador. Os tipos de modificações permitidas determinam a dificuldade do problema. Podemos considerar as três seguintes possibilidades:

- **conexidade incremental:** arestas podem ser acrescentadas ao grafo;
- **conexidade decremental:** arestas podem ser removidas do grafo;
- **conexidade totalmente dinâmica:** arestas podem ser acrescidas ou removidas do grafo.

O problema de conexidade incremental é resolvido de maneira bastante eficiente por meio de uma estrutura de dados conhecida como *union-find* [9], que provê resposta às consultas em tempo aproximadamente constante.

O problema de conexidade decremental foi resolvido por Even e Shiloach [4]. O método usa uma tabela onde se mantem o identificador da componente de cada vértice do grafo. A grande questão é como atualizar essa tabela quando da remoção de uma aresta.

Se o grafo em questão é uma floresta, ou seja, não tem circuitos, o problema é mais simples. A ideia é que a remoção de uma aresta sempre quebrará uma componente em duas, e atualizaremos os identificadores do menor dos dois pedaços resultantes. Ao remover por exemplo a aresta uv, devemos determinar se o pedaço em que u ficou é menor ou maior que o pedaço em que v ficou. Isso pode ser feito por exemplo usando-se duas buscas, uma a partir de u e uma a partir de v, que são executadas paralelamente (ou de maneira intercalada) e, ao chegarmos ao final de uma delas, abortamos a segunda, pois determinamos já qual é o menor dos pedaços. O tempo gasto desta maneira será proporcional ao tamanho do menor dos pedaços, implicando que o tempo para a atualização, amortizado pelo número de consultas é $O(\lg n)$, onde n é o número de vértices do grafo.

Para grafos arbitrários, é necessário determinar se a aresta uv removida quebra ou não uma componente em duas. A ideia de novo é executar dois processos em paralelo (ou intercaladamente): um para determinar se u e v estão ainda na mesma componente após a remoção de uv, e o outro para decidir se uma componente foi quebrada em duas. Esse segundo processo é semelhante ao que foi usado no caso em que o grafo é uma floresta. Já o procedimento envolvido no primeiro processo é um pouco mais sofisticado, e utiliza uma estrutura de dados construída a partir de uma BFS do grafo original, que vai sendo atualizada a medida que o grafo vai sofrendo remoções de arestas.

Para conexidade totalmente dinâmica em florestas, podemos usar uma coleção das chamadas *link-cut trees* [7] ou das chamadas *Euler tour trees* [5] para representar a floresta. Estas estruturas implementam uma rotina chamada FindRoot(x), semelhante à rotina FindSet(x) do *union-find*, que devolve um representante da componente em que o vértice x se encontra. Com isso, podemos responder às consultas facilmente, pois dois vértices x e y estão na mesma componente da floresta se e somente se FindRoot(x) = FindRoot(y). O tempo amortizado de atualização e consulta é $O(\lg n)$.

Para o caso geral da conexidade totalmente dinâmica, podemos representar um grafo arbitrário por uma floresta geradora maximal do grafo. Se chamarmos uma tal floresta de F, podemos usar por exemplo Euler tour trees para armazená-la. Com isso, inserções e consultas podem ser implementa-

das diretamente usando as operações correspondentes das Euler tour trees. Por outro lado, as remoções são mais desafiadoras. Em especial, se a aresta removida fizer parte da floresta F, isso pode requerer encontrar, de forma eficiente, uma outra aresta do grafo para ser adicionada a F. Esta operação é implementada por meio de uma estrutura de dados mais complexa, que planejamos estudar também, como parte deste projeto [2].

2 Justificativa

O estudo de estruturas de dados e algoritmos mais sofisticados ou com análises mais complexas proporciona uma oportunidade de aprofundamento de conhecimentos importantes adquiridos durante um bom curso de graduação em Ciência da Computação.

Problemas dinâmicos em grafos dependem de estruturas de dados não triviais, para serem resolvidos de maneira eficiente. Várias destas estruturas não são abordadas nos cursos obrigatórios de uma graduação em Ciência da Computação.

Redes como a internet são extremamente dinâmicas, e ampliaram dramaticamente o interesse no estudo de problemas dinâmicos em grafos. O estudo da conexidade de tais redes é uma das primeiras etapas no entendimento da dinâmica da rede, por isso eles têm um papel central nessa área.

O tema escolhido para este projeto serve como motivação para o estudo de tópicos centrais de áreas importantes da Ciência da Computação, como projeto e análise de algoritmos e estruturas de dados.

O Daniel é um aluno do terceiro ano do Bacharelado em Ciência da Computação (BCC) do IME-USP. Embora suas notas no primeiro ano não sejam tão boas, levando-o a uma média 7,0 nas disciplinas que cursou, o Daniel não obteve nenhuma reprovação, e melhorou significativamente seu aproveitamento no segundo ano, obtendo notas todas acima de 7,0. O seu interesse em fazer iniciação científica demonstra esse amadurecimento e certamente vai consolidar ainda mais essa melhora do seu aproveitamento no curso.

As disciplinas do BCC que já foram concluídas pelo Daniel são suficientes para levar adiante este projeto. Dentre as disciplinas que ele cursou, destacamos MAC0121 Estruturas de Dados e Algoritmos I e MAC0323 Estruturas de Dados e Algoritmos II, que dão uma base essencial para o desenvolvimento desse projeto.

3 Objetivos

Neste projeto, o plano é estudar e implementar estruturas de dados e algoritmos usados no contexto de conexidade dinâmica de grafos. Além do funcionamento destas estruturas e algoritmos, serão estudados também suas análises de correção e seu consumo assintótico de tempo. Ou seja, serão abordados também conceitos e técnicas de análise de algoritmos.

O Daniel fez um rápido estudo preliminar do problema da conexidade dinâmica em grafos, que deve ser aprofundado durante os próximos meses. Ele também já implementou uma primeira estrutura de dados, baseando-se no livro de Sedgewick e Wayne [6], que será usada como base da implementação das chamadas *splay trees* [8]. Estas por sua vez são usadas na implementação das *link-cut trees*. No presente momento, o Daniel está estudando as splay trees, e deve começar a implementá-las nas próximas semanas.

O objetivo dessa iniciação científica é o aprendizado de várias estruturas de dados que possuem aplicações não apenas em conexidade dinâmica, mas em diversas outras áreas da Ciência da Computação. Vale destacar que as estruturas de dados que serão abordadas são não triviais e, sendo este um projeto de iniciação científica, é possível que apenas parte do material mencionado seja completamente estudado e implementado.

Como subproduto da iniciação científica, o Daniel deve também produzir um texto com tudo o que foi estudado.

4 Plano de trabalho e cronograma

O Daniel começou recentemente a trabalhar nesse projeto. Numa primeira fase, ele estudou a seção 1.1 do capítulo *Dynamic Graphs* no livro de Demetrescu et al. [3] sobre os conceitos básicos de grafos dinâmicos e as suas possíveis estruturas de dados para a implementação. Após isto, estudou o funcionamento das *link-cut trees* [1], implementou uma Árvore Binária de Busca e está estudando as *splay trees*.

A nossa intenção é inicialmente estudar e implementar os algoritmos e as estruturas de dados que são base para as *link-cut trees*. Tendo passado pelos conceitos fundamentais, inicia-se a implementação de tal estrutura, junto a isso se faz a análise de desempenho do algoritmo medindo seu consumo assintótico de tempo. Ao término desta primeira etapa, o Daniel irá elaborar um texto descrevendo o que foi necessário para a implementação das *link-cut*

trees e expor os resultados obtidos das análises.

Ao passo que escreve o texto referente à primeira etapa, ele seguirá para o próximo tópico, que é o estudo e implementação das *euler tour trees*. Primeiramente, ele irá estudar o funcionamento da estrutura de dados e verficar se esta faz uso de outras estruturas, caso positivo, ele irá estudar e implementar as estruturas de dados base para as *euler tour trees*. Ao término deste estudo, o Daniel iniciará a implementação e análise das *euler tour trees*.

Ao fim desta segunda etapa, o Daniel escreverá um texto descrevendo como foi a implementação das *euler tour trees* e os resultados obtidos das análises. A partir disso, ele irá escrever um relatório intermediário contendo as informações dos textos já escritos e irá comparar o comportamento de cada estrutura de dados.

É preciso adicionar mais algum tópico de estudo?.

5 Material e métodos

Há muito material sobre o grafos dinâmicos na literatura. Primeiramente, utilizamos o livro [3] para ter uma familiaridade com os conceitos de grafos dinâmicos. Utilizaremos as seguintes notas de aula [1, 2] e os seguintes os artigos [7, 5] para nos aprofundarmos nas *link-cut trees* e nas *euler tour trees*. E utilizaremos o livro [6] e o artigo [8] para estudar as estruturas de dados que são base para implementação das estruturas de grafos dinâmicos.

O método que usaremos para conduzir essa iniciação científica é tradicional. O aluno estudará cuidadosamente os diversos resultados, e irá implementando parte do que for estudado, e teremos reuniões a cada duas semanas para discutir os assuntos estudados e as implementações. Ao mesmo tempo que estuda e implementa parte dos tópicos, o aluno escreverá um texto, o que possibilitará uma melhor avaliação de quão bem o material estudado está sendo absorvido.

6 Forma e análise dos resultados

Durante todo o período de estudo, além das implementações, o aluno estará preparando um texto, que, ao final do trabalho, conterá tudo que foi estudado na iniciação científica. Este é o principal objeto que pode ser usado na análise do trabalho que estará sendo desenvolvido. Fora isso, evidentemente

esperamos que o aluno mantenha o bom desempenho (ou até melhore) no BCC.

Referências

[1] E. Demaine, J. Holmgren (scriber), J. Jian (scriber), M. Stepanenko (scriber), and M. Ishaque (scriber). Link-cut trees problems. Lecture Notes in Advanced Data Structures, 2012.

[2] E. Demaine and K. Lai (scriber). Dynamic graph problems. Lecture Notes in Advanced Data Structures, 2007.

[3] C. Demetrescu, I. Finocchi, and G. F. Italiano. *Handbook of Data Structures and Applications*, chapter Dynamic Graphs. Chapman and Hall/CRC, 2004.

[4] S. Even and Y. Shiloach. An on-line edge-deletion problem. *Journal of the ACM*, 28(1):1–4, 1981.

[5] M. R. Henzinger and V. King. Randomized dynamic graph algorithms with polylogarithmic time per operation. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing (STOC)*, page 519, 1995.

[6] R. Sedgewick and K. Wayne. *Algorithms*. Addison-Wesley, 4 edition, 2011.

[7] D. D. Sleator and R. E. Tarjan. A data structure for dynamic trees. In *Proceedings of the Thirteenth Annual ACM Symposium on Theory of Computing (STOC)*, page 114, 1983.

[8] D. D. Sleator and R. E. Tarjan. Self-adjusting binary search trees. *Journal of the ACM*, 32(3):652–686, 1985.

[9] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *Journal of the ACM*, 22(2):215–225, 1975.