O Problema da Árvore Geradora Mínima (AGM)

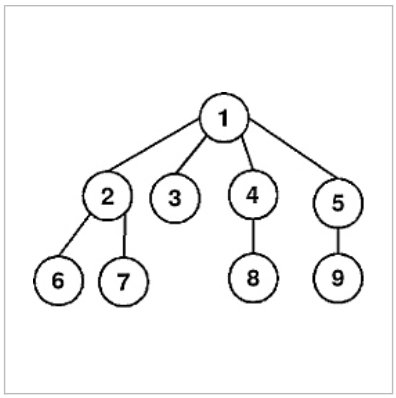
fold

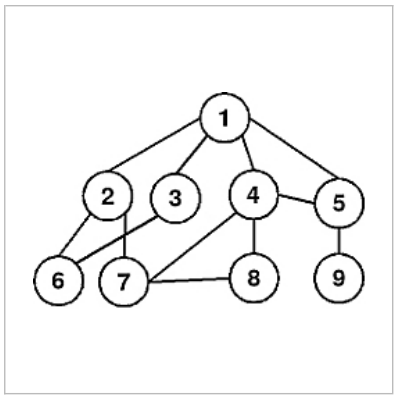
Table of Contents

Vamos tentar entender o problema dissecando seu nome: vamos ver o que são árvores, depois o que são árvores geradoras e finalmente o que são árvores geradoras mínimas.

Árvore?

Na teoria dos grafos, uma *árvore* nada mais é do que um tipo especial de grafo: árvores são grafos em que não existem ciclos. Vejamos os exemplos abaixo:





Perceba como o grafo da esquerda não possui nenhum ciclo, logo **é** uma árvore. Já o grafo da direita possui ciclos, como, por exemplo, o formado entre os nós 1, 4 e 5. Esse grafo **não é** uma árvore.

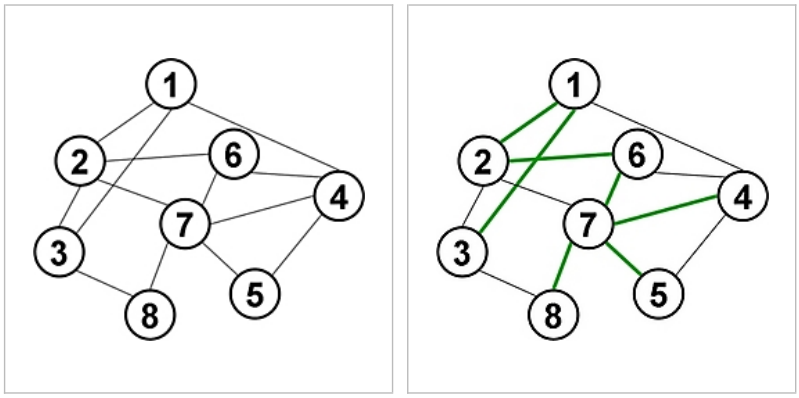
Apesar de parecer estranho, árvores são modelos matemáticos muito utilizados no dia a dia. Dois exemplos são as árvores genealógicas, que mostram toda a estrutura de uma família, em que os nós representam pessoas e as arestas representam a relação "é pai/mãe de" e as árvores de hierarquia, que representam a hierarquia de uma determinada organização, em que os nós representam funcionários e as arestas representam a relação "é chefe de".

E como assim *geradora*?

Uma árvore é dita *geradora* se ela interliga (direta ou indiretamente) todos os nós do grafo, ou seja, se todos os nós do grafo fazem parte da árvore - não ficou nenhum nó "isolado". Perceba que, por exemplo, os nós 1 e 8 da árvore acima não estão diretamente ligados (não existe aresta entre eles), mas mesmo assim ambos fazem parte da árvore. Se não existisse a aresta que liga 4 e 8, o nó 8 ficaria "sozinho", isolado do resto da estrutura. Os nós {1,2,3,4,5,6,7,9} continuariam a formar uma árvore, mas essa árvore não seria mais geradora, pois não inclui todos os nós do grafo - o nó 8 ficou de fora.

Uma propriedade interessante das árvores geradoras é que elas são as menores estruturas que conectam todos os nós do grafo. Se temos um grafo com várias arestas (imagine que temos um grafo em que cada nó possui arestas para todos os outros) e queremos escolher somente algumas dessas arestas de maneira que o grafo continue **conexo** (ou seja, todos os nós estejam conectados entre si, direta ou indiretamente), acabaríamos escolhendo as arestas de forma a escolher uma árvore geradora. Mas porquê? Pense comigo: queremos escolher o conjunto de arestas de forma que todos os nós continuem ligados, e queremos que esse conjunto seja o menor possível. Bom, se queremos escolher o menor número de arestas necessárias, vamos escolher somente arestas suficientes para que o grafo fique conectado - não é interessante escolher nenhuma aresta a mais. Dessa maneira, não vamos nunca colocar arestas que formem ciclos, afinal essas arestas que formam ciclos estão ligando dois nós que já estavam conectados! Vamos então, no fim das contas, escolher as arestas e formar uma estrutura que ligue todos os nós e que seja acíclica. Então, vamos escolher (obrigatoriamente) uma árvore geradora. Vamos deixar isto

mais claro com um exemplo:

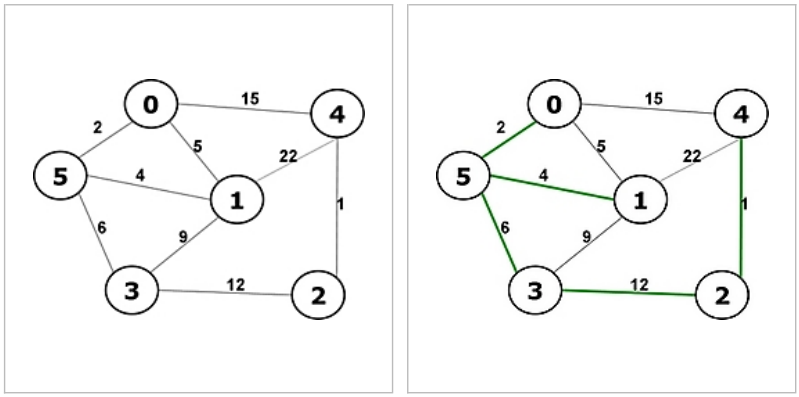


A imagem da esquerda apresenta um grafo qualquer com 8 nós e 13 arestas. Vamos escolher somente algumas dessas arestas (o menor número possível), de forma que o grafo fique conectado. Na imagem da direita, as arestas escolhidas estão destacadas. Não precisamos escolher nenhuma aresta a mais, já que todos os nós já estão conectados. Também não podemos escolher nenhuma aresta a menos, pois estaríamos desconectando os nós. Perceba como escolhemos justamente uma árvore geradora - um grafo sem ciclos (árvore) que conecta todos os nós (geradora).

Mas espere um pouco, porque foram escolhidas justamente estas arestas? Dava para ter formado uma árvore geradora com outros conjuntos de arestas! É verdade, estas arestas que estão marcadas foram escolhidas arbitrariamente. Poderia ter sido escolhido um outro conjunto de arestas completamente diferente que tivesse a mesma propriedade: formasse uma árvore geradora. Poderíamos, por exemplo, ter escolhido a aresta 3-8 e ter retirado a aresta 7-8, formando outra árvore geradora. Então, só **revisando** o que foi visto até agora: árvores são grafos que não contém ciclos; quando as árvores incluem todos os nós do grafo são chamadas de árvores geradoras; as árvores geradoras são interessantes pois são a menor estrutura que conecta todos os nós da árvore.

Tá bom, e esse negócio de *mínima*?

Chegamos agora à última definição: quando uma árvore geradora é dita **mínima**? Bem, imagine que temos um grafo, mas agora este grafo possui um peso associado as arestas (lembre-se que esse peso pode representar qualquer valor em um problema real, como custo, fluxo, confiabilidade, etc). O problema de escolher uma árvore geradora continua semelhante: para o mesmo grafo, podemos escolher vários conjuntos de arestas que formam árvores geradoras. Entretanto, agora vamos levar em consideração o peso das arestas - queremos escolher, entre as árvores geradoras possíveis, aquela que possua o menor *peso total*, onde o peso total é dado pela soma dos pesos das arestas da árvore. Vejamos um exemplo:



Na imagem da esquerda temos um grafo qualquer com 6 nós e 9 arestas. Cada aresta possui um peso associado a ela. Como escolher uma árvore geradora que tenha soma de pesos mínima? Já na imagem da direita, as arestas escolhidas estão destacadas, formando uma árvore geradora de peso 1+2+4+6+12 = 25. Perceba que a árvore geradora escolhida possui o menor peso (onde *peso* é a soma dos pesos das arestas) dentre todas as possíveis árvores geradoras.

Árvore geradora mínima

Finalmente, podemos enunciar o problema: o **problema da árvore geradora mínima** consiste em encontrar, dado um grafo com arestas ponderadas, uma estrutura de conexão (árvore) em que todos os nós (geradora) se conectem (direta ou indiretamente) uns aos outros. Essa estrutura deve possuir o menor peso possível, onde o peso é dado pela soma dos pesos das arestas escolhidas (mínima).

Algoritmos para o problema da AGM

Dá para perceber que o problema de achar uma árvore geradora *mínima* em um grafo qualquer não é tão simples quanto achar somente uma árvore geradora. Temos de escolher as arestas com cuidado para que, no final, a soma de seus pesos seja a menor possível. Como vamos escolher essas arestas? Fica claro que não podemos escolher quaisquer arestas - temos de tomar cuidado para formarmos uma árvore geradora. Então vamos propor outra solução: vamos formar todas as árvores geradoras possíveis e, entre todas essas, escolher aquela que tem a menor soma dos pesos (ou seja, aquela que é mínima)!

Essa abordagem não vai funcionar muito bem. Há muito tempo atrás (quando ainda nem existiam computadores e, obviamente, nem a ciência

da computação), um matemático chamado Arthur Caley provou que um grafo com, digamos, N nós possui N^{N-2} árvores geradoras diferentes.[1] Imagine só o tamanho desse número: se tivermos um grafo com 4 nós, teremos 16 árvores geradoras diferentes (mas somente 1 delas é a mínima). Se tivermos um grafo com 10 nós, teremos 100000000 - 100 milhões de árvores geradoras diferentes (e, ainda assim, somente 1 delas é a mínima!). Esse negócio de ir testando todas as árvores geradoras então claramente não vai dar certo: estamos buscando uma agulha em um palheiro cada vez maior (e que fica **gigantesco** rapidamente)!

Algoritmo de Kruskal

Mas tudo bem, para o problema da AGM isso não vai ser um problema muito grande. Alguns matemáticos já conseguiram comprovar que o problema pode ser resolvido de maneira muito mais fácil: basta irmos escolhendo as menores arestas, uma a uma e com o cuidado de não formar ciclos, que eventualmente teremos uma árvore geradora mínima para qualquer grafo! Essa estratégia para resolver o problema é conhecida como o **Algoritmo de Kruskal** e pode ser resumida como:

> Dado um grafo formado pelo conjunto de nós N e o conjunto de arestas E, faça, até que tenhamos formado uma árvore geradora:
>
> 1. Escolher, do conjunto de arestas E, aquela que possua o menor peso.
> 2. Se a inclusão desta aresta na solução não formar um ciclo, incluimos a aresta.
> 3. Caso contrário, descartamos a aresta e a retiramos de E.

Algoritmo de Prim

Outro algoritmo famoso para o problema é o **Algoritmo de Prim**, que funciona de maneira bastante semelhante ao anterior. No algoritmo de Kruskal, começamos com várias árvores separadas (no começo, cada nó é uma árvore) e vamos juntando essas árvores através das arestas de menor valor até formarmos uma árvore geradora. No algoritmo de Prim, iniciamos com uma árvore formada por um único nó (qualquer nó do grafo) e vamos adicionando à árvore, a cada passo, o nó que estiver mais próximo dela. Poderíamos resumir o algoritmo assim:

> Dado um grafo formado pelo conjunto de nós N e o conjunto de arestas E, escolha um nó qualquer do grafo e coloque na árvore T. Repita os seguintes passos até que T seja uma árvore geradora:
>
> 1. Escolher o nó que está mais *próximo* da árvore T, ou seja, o nó que está ligado a um nó de T pela aresta de menor valor.
> 2. Adicionar este nó a T. (podemos parar de adicionar quando tivermos adicionado todos os nós do grafo).

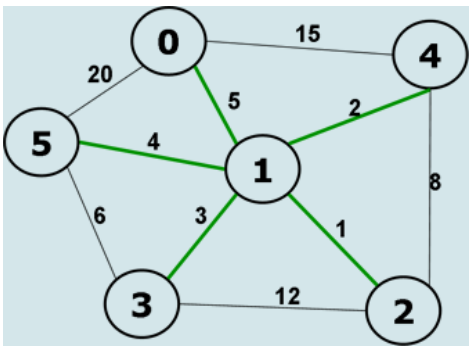
Para uma discussão aprofundada de vários algoritmos para o problema da AGM, veja esse estudo de cinco diferentes algoritmos para o problema.

Conclusões e o problema da AGM-MO

Esses são dois algoritmos clássicos da Ciência da Computação, são simples o suficiente para que todos possam compreendê-los e resolvem de maneira fácil e rápida um problema que parecia ser assustadoramente grande. Perceba que ambos os algoritmos, a cada passo, escolhem para incluir na solução a aresta ou nó que, naquele momento, parece a melhor possível (a cada passo, no algoritmo de Kruskal, escolhemos a menor aresta que não forma ciclo e no algoritmo de Prim escolhemos o nó que está ligado a árvore pela menor aresta). Essa característica de, a cada passo, escolher a opção que parece a melhor possível (sem se preocupar com o futuro, com as possíveis consequências dessa escolha), faz com que esses algoritmos sejam chamados de **algoritmos gulosos** (ou míopes).

Certo, então até agora eu falei um pouco sobre o que são grafos, depois falei de um problema muito famoso no campo da teoria dos grafos e expliquei dois algoritmos clássicos para resolver esse problema. Acho que deu para reparar que o problema da AGM por si só não é mais de muito interesse para a pesquisa científica: já existem vários algoritmos que resolvem o problema fácil e rapidamente, para problemas de tamanho até na casa dos bilhões de nós!

Mas o problema da AGM por si só não é muito aplicável no chamado "mundo real" (na indústria, nas empresas, em modelos reais): ele é muito "solto", possui muito poucas restrições. Imagine que você vai construir uma rede de distribuição de água e no seu modelo de grafos os nós vão representar bombas de distribuição, enquanto as arestas vão representar canos correndo pela cidade (cujo peso será equivalente ao custo de instalação do cano). Vamos querer criar então um sistema de conexão em que de qualquer bomba eu consiga enviar água para qualquer outra, e queremos minimizar o custo de instalar os canos pela cidade (ou seja, vamos buscar a AGM). Será que seria interessante, para esse sistema, uma solução assim:



Todo o fluxo de água do seu sistema está passando pelo nó 1, que muito provavelmente ficará sobrecarregado e não conseguirá redistribuir

toda a água que recebe. Nesse caso, dizemos que o nó 1 possui um grau (número de arestas da árvore que se ligam a ele) muito grande, e vamos buscar uma solução em que o maior grau seja o menor possível. Essa é uma possível variação do problema da AGM, chamada *árvore geradora mínima restrita em grau*. Vamos ainda buscar a árvore geradora mínima, mas com uma restrição a mais: o nó que possuir maior grau no grafo não pode ter um grau maior que um determinado valor fixo. Esse é um problema mais real (pois normalmente árvores que centralizam tudo em um só nó, ou seja, possuem um nó de grau muito grande, não são soluções interessantes) e **muito** mais complicado de resolver. Os algoritmos de Kruskal e Prim descritos anteriormente não funcionam mais para este problema! Para ser mais preciso, não existem mais algoritmos simples (gulosos, por exemplo), para resolver este problema: para resolvê-lo vamos precisar percorrer aquele gigantesco universo de ***todas as possíveis árvores geradoras***.

Outro grande problema reside no próprio modelo de grafos que estamos trabalhando: até agora estivemos associando a cada aresta um peso. Mas porque somente um único peso? Será que, em problemas reais, o relacionamento entre os nós é tão simples que pode ser representado por somente um único valor? E se, em nosso sistema de distribuição de água, quisermos levar em conta não só o custo de ligar duas bombas, mas também a quantidade de água que conseguiremos bombear? E se quisermos pensar na confiabilidade dos canos que irão ligar as bombas? Fica claro que, se quisermos uma solução realmente boa (e, consequentemente, um sistema realmente bem-feito), não podemos levar em conta somente o custo, temos de levar em conta vários outros fatores na hora de construir o sistema! Vamos então associar a cada aresta não um, mas vários valores, cada um representando uma característica diferente que deve ser levada em conta (como custo, confiabilidade, fluxo, etc). O problema de encontrar uma árvore geradora mínima em um grafo no qual as arestas possuem vários valores é chamado de **problema da árvore geradora mínima multi-objetivo (AGM-MO)**.

Referências recomendadas

Meu estudo sobre o problema da AGM não foi muito prolongado: os algoritmos mais famosos são bastante conhecidos e as pesquisas na área atualmente buscam algoritmos cada vez mais rápidos para o problema (já existem algoritmos de ordem de tempo praticamente linear). Para minha pesquisa, os algoritmos de Kruskal e Prim são rápidos e simples o suficiente. Entretando, recomendo esse artigo para quem estiver interessado em um resumo detalhado dos algoritmos estado-da-arte para o problema:

Minimum-weight spanning tree algorithms a survey and empirical study, por Cüneyt F. Bazlamaçci and Khalil S. Hindi.

BibTeX

```
@article{380267,
author = {C{\"u}neyt F. Bazlama{\c{c}}ci and Khalil S. Hindi},
title = {Minimum-weight spanning tree algorithms a survey and empirical study},
journal = {Comput. Oper. Res.},
volume = {28},
number = {8},
year = {2001},
issn = {0305-0548},
pages = {767—785},
doi = {http://dx.doi.org/10.1016/S0305-0548(00)00007-1},
publisher = {Elsevier Science Ltd.},
address = {Oxford, UK, UK},
}
```

Footnotes

1. Na verdade, a fórmula de Caley funciona para grafos completos (em que todos os nós se ligam uns aos outros). Para uma fórmula mais geral, procure o teorema de Kirchhoff.