

Determinando distâncias e consensos entre árvores

Thiago Lopes Nascimento

Universidade Federal Fluminense

Departamento de Ciência da Computação - IC - Niterói - RJ

thiago.nascimento@id.uff.br

Luís Felipe Ignácio Cunha

Universidade Federal Fluminense

Departamento de Ciência da Computação - IC - Niterói - RJ

lfignacio@ic.uff.br

RESUMO

A heterogeneidade mutacional dos tumores de uma pessoa pode ser descrita como uma árvore. Portanto, a compreensão das semelhanças das árvores é um desafio para prever as semelhanças na evolução do câncer de diferentes pacientes. Neste trabalho é apresentada uma abordagem para determinar distâncias entre árvores, associando o seu formato Newick e operações de SWAP em strings. Também consideramos como entrada um conjunto de árvores e trabalhamos com os problemas de MEDIANA e CLOSEST, que são formas de relacionar todas as entradas como os problemas de minimizar a soma e minimizar o máximo, respectivamente, onde provamos ser NP-completa a determinação da solução do último problema. Além disso, desenvolvemos dois algoritmos para obter limites superiores para as soluções de MEDIANA e CLOSEST, além de apresentarmos alguns experimentos.

PALAVRAS CHAVE. Algoritmos em grafos, Strings, Distância entre árvores.

Teoria e Algoritmo em Grafos e Otimização Combinatória

ABSTRACT

The mutational heterogeneity of tumors of a person can be described as a tree. Therefore, understanding tree similarities is a challenge in order to predict similarities of the evolution of cancer of different patients. In this work it is presented an approach to determine distances between trees, by associating their Newick format and swap operations on strings. We also consider set of trees as input and deal with MEDIAN and CLOSEST problems, which are forms of relate all input together as the min-sum and min-max problems, resp., where we set the decision for the last problem as NP-complete. Moreover, we develop two algorithms to yield upper bounds on the median and closest solutions and some experiments are presented.

KEYWORDS. graph algorithms, strings, distance between trees.

Theory and Algorithm graphs and Combinatorial Optimization

1. Introdução

Problemas de consenso considerando permutações Como proposto por Watterson et al. [1982], um problema de *Rearranjo de genoma* é interpretado como a transformação de uma permutação em outra através do menor número de operações, dependendo dos possíveis rearranjos permitidos. Este modelo considera os seguintes premissas: a ordem dos genes em cada genoma é conhecida; todos os genomas que comparamos compartilham de um mesmo conjunto de genes; todos os genomas contêm uma única cópia de cada gene; e todos os genomas são constituídos por um único cromossoma. Assim, os genomas podem ser modelados como permutações, uma vez que cada gene é codificado como um número inteiro. Se uma entrada tiver mais de dois genomas, existem diversas abordagens para encontrar genomas ancestrais Bader [2011]; Caprara [2003]; Cunha et al. [2020]; Cunha e Protti [2019]; Haghghi e Sankoff [2012]; Pe'er e Shamir [1998]. Um problema relevante é o problema da MEDIANA, em que, para uma métrica M , o objetivo é encontrar um genoma solução que minimize a soma das distâncias entre a solução e todos os genomas de entrada.

Haghghi e Sankoff [2012] observaram que, no que diz respeito à métrica de breakpoint, a tendência para medianas é cair sobre ou perto de um dos genomas de entrada, o que não contém informação útil para a reconstrução filogenética. Também conjecturaram o mesmo comportamento relativamente a outras métricas. Assim, uma abordagem alternativa é considerar o problema CLOSEST para uma métrica fixa M , que visa encontrar um genoma que minimize a distância máxima a qualquer genoma na entrada, o que pode ser visto como encontrar um genoma no centro de todos os outros, ou seja, um genoma correspondente ao raio do conjunto de entrada.

Lancot et al. [2003] estudaram o problema do CLOSEST considerando cadeias de caracteres relativamente à distância de Hamming, e concluíram que este problema é NP-difícil mesmo para strings binárias. Popov [2007] estudou o problema de CLOSEST considerando permutações relativas à operação de swap, e mostrou que é NP-difícil. Cunha et al. [2020] mostrou que o problema CLOSEST é NP-difícil para várias distâncias de rearranjo do genoma bem conhecidas como a de breakpoint e a block-interchange.

Apresentamos duas abordagens para calcular a distância entre árvores. A primeira, apresentada em Subseção 3.1, é quando a altura das árvores da entrada é restrita, especificamente limitada por 2, uma vez que a distância entre árvores se torna equivalente à distância de swap entre permutações. A segunda, apresentada em Subseção 3.2, é desenvolvida considerando árvores de entrada genéricas e definindo a QUASE v -ÁRVORE MÁXIMA COMUM conseguimos desenvolver a distância entre árvores em tempo polinomial. Além disso, em Seção 4 consideramos um problema mais genérico em que a entrada é constituída por mais de duas árvores e o nosso objetivo é determinar as soluções das árvores minimização-máxima (problema de CLOSEST) e minimização-soma (problema da MEDIANA). Desenvolvemos algoritmos para determinar limites superiores para as soluções. Na mesma seção é possível observar resultados práticos.

2. Formato de Newick e distância entre árvores

Cada árvore enraizada está associada a uma permutação através da sua representação entre brackets. As folhas são representadas como inteiros e a hierarquia da árvore corresponde aos pares de brackets utilizados para envolver os elementos. Se dois elementos estiverem envolvidos pelo mesmo brackets, qualquer ordenação entre eles está associada à mesma árvore. Portanto, a distância entre as árvores neste caso é igual a zero.

Normalmente, estas árvores têm comprimentos de borda por razões práticas. O formato padrão para representar árvores filogenéticas é o *formato de Newick*. A descrição Newick de uma árvore filogenética é uma sequência de parênteses anotados com valores. Pode também incluir comprimentos de ramos ou valores de bootstrap. Esta descrição é obtida percorrendo a árvore

filogenética em pós-ordem e seguindo regras simples que permitem realizar o parser de uma cadeia de Newick na árvore filogenética correspondente e vice-versa. Este método foi descrito por Cardona et al. [2008]. É importante notar que todas as arestas têm o mesmo comprimento com base na nossa motivação. As árvores e as suas sequências em formato Newick analisadas neste estudo são apresentadas em Figura 1.

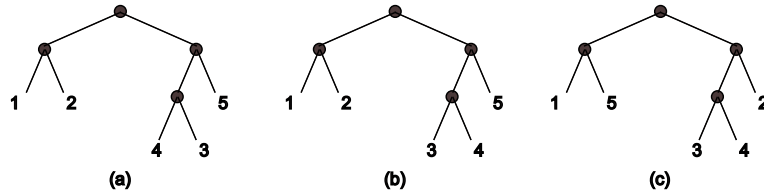


Figura 1: Árvores enraizadas com seus formatos de Newick. Strings de tamanho 5. (a) Árvore associada a $((1, 2)((4, 3)5))$, que é igual a árvore descrita em (b) associada à $((1, 2)((3, 4)5))$. Ambas as árvores são distintas da árvore descrita em (c) associada à $((1, 5)((4, 3)2))$.

DISTÂNCIA ENTRE ÁRVORES

Instância: Duas árvores, T_p and T_q , com seus formatos de Newick sendo p and q , de mesmo tamanho, e um inteiro k .

Pergunta: $d(T_p, T_q) \leq k$?

Para estabelecer a distância entre árvores, deve-se tomar os seguintes passos.

Observação 1. *Pode-se assumir que todo par de brackets contem ao menos dois elementos.*

Como consequência direta da construção das árvores baseada no formato de Newick, tem-se Proposição 1.

Proposição 1. *O número de nós internos de uma árvore é igual ao número de pares de brackets.*

Para determinar a distância entre T_p e T_q , é necessário encontrar o menor número de operações sobre uma árvore de elementos através de T_p para transformar em T_q . Uma *operação sobre uma árvore* T_p é definida como mover uma subárvore do seu nível para um nível acima ou abaixo em T_p . Essa subárvore é um nó interno com todos os seus descendentes ou um único nó folha de T_p .

3. Distância entre árvores

A seguir, apresentamos duas abordagens para determinar a distância entre duas árvores, sendo a primeira baseada na altura das árvores e a segunda no problema que definimos como MAIOR SUBÁRVORE COMUM.

3.1. Distância baseada na altura

A altura de uma árvore, denotada por $h(T)$, é definida como o número de arestas no caminho mais longo entre a raiz e uma folha. Para determinar a distância entre árvores, consideramos os casos em que $h(T)$ é limitado por valores fixos. Especificamente, relacionamos esta métrica com a distância de swap entre permutações quando $h(T) = 2$. Cunha et al. [2020].

Proposição 2. *Para cada par de árvores T_p e T_q associados com as sequências de Newick p e q de mesmo tamanho, se ambos $h(T_p)$ e $h(T_q)$ serem iguais, então $d(T_p, T_q) = 0$.*

Demonstração. Como a altura de cada árvore é igual a 1, pela Observação 1 que existe um único par de brackets das sequências correspondentes. Assim, todos os elementos pertencem ao mesmo brackets, e então, qualquer ordem desses inteiros associa-se a uma única árvore, considerando a construção das árvores. $\square$

Como consequência da Proposição 2, cada par de permutações $\pi, \sigma \in S_n$ satisfaz $d(T_\pi, T_\sigma) = 0$, onde S_n é o grupo simétrico de permutações de comprimento n .

Permutações também podem ser representadas por cada elemento seguido da sua imagem. Por exemplo, dado um conjunto $\{1, 2, 3\}$, a sequência $(1\ 2\ 3)$ mapeia 1 para 2, 2 para 3 e 3 para 1, correspondendo à permutação $[2\ 3\ 1]$. Esta representação não é única; por exemplo, $(2\ 3\ 1)$ e $(3\ 1\ 2)$ são equivalentes. Permutações são compostas por um ou mais ciclos algébricos, em que cada *ciclo algébrico de uma permutação* π é uma representação de um domínio i , seguido da sua imagem $\pi(i)$, seguido de tomar a imagem de $\pi(i)$ como elemento seguinte, ou seja, $\pi(\pi(i))$, e assim por diante. Continuamos este processo até chegarmos a um elemento repetido, definindo assim a permutação. Denotamos por $c(\pi)$ o número de ciclos algébricos de π . Por exemplo, dado $\pi = [3\ 2\ 4\ 1] = (1\ 3\ 4)(2)$, temos $c(\pi) = 2$. De modo geral, quando comparamos duas permutações distintas π e σ podemos também calcular o ciclo algébrico da mesma forma. Um *ciclo algébrico de uma permutação* π em relação a uma permutação σ é uma representação de um domínio $\sigma(i)$, seguido da sua imagem em π , continuando até chegarmos a um elemento repetido. Por exemplo, dado $\sigma = [2\ 4\ 1\ 3]$ e $\pi = [3\ 2\ 4\ 1]$, temos o seguinte ciclo algébrico $c(\pi, \sigma) = (2\ 3\ 1\ 4)$. Logo, $c(\pi) = c(\pi, \iota)$, onde ι é a *permutação identidade* $\iota = [1\ 2\ \dots\ n]$.

Uma troca de elementos que envolva os elementos a e b de tal forma que a e b estejam no mesmo ciclo é uma troca que *quebra* o ciclo em dois, enquanto que se a e b pertencerem a ciclos diferentes, a troca destes elementos *unes* os dois ciclos. Fertin et al. [2009] Assim, ao considerar a métrica de *swap*, a distância *swap* entre uma permutação π e uma permutação σ é determinada da seguinte forma: $d_{\text{swap}}(\pi, \sigma) = n - c(\pi, \sigma)$, onde $c(\pi, \sigma)$ é o número de ciclos algébricos de π em relação a σ , e pode ser calculado em tempo linear.

Lema 1. Dada uma árvore T_p de Newick a sequência p de tamanho n , $h(T_p) \leq 2$ se e só se p for um ciclo algébrico de uma permutação de comprimento n .

Demonstração. Por um lado, com base na hierarquia da árvore enraizada T_p , o número de filhos que a raiz tem é o número de pares de brackets dentro do par mais externo associado ao nó raiz. Além disso, cada filho da raiz tem um único par de parênteses associado a ele. Portanto, se $h(T_p) \leq 2$, então todas as folhas com o mesmo pai correspondem a um ciclo algébrico. Por outro lado, cada ciclo algébrico está associado a uma subárvore que é composta por um único nó interno e todos os outros são folhas correspondentes aos elementos do ciclo. Logo, todos os ciclos são filhos de um mesmo nó interno, e então $h(T_p) \leq 2$. $\square$

Seja T_p uma árvore com $h(T_p) = 2$, denotamos por π_p a permutação cujo ciclo algébrico é p .

Teorema 2. Se $h(T_p) = h(T_q) = 2$, então $d(T_p, T_q) = d_{\text{swap}}(\pi_p, \pi_q)$.

Demonstração. Visto que as alturas de T_p e T_q são iguais a 2, qualquer operação numa árvore tem o efeito de quebrar um par de brackets ou de unir dois brackets, assim como o efeito de qualquer operação de troca nos ciclos algébricos correspondentes. Além disso, como dois nós podem ser filhos do mesmo nó com qualquer rótulo, é sempre possível aplicar uma troca viável na ordem correspondente entre p e q . $\square$

3.2. Distância entre árvores de acordo com as subárvores comuns

Dada uma sequência de Newick p e a sua árvore T_p , temos uma subárvore de T_p como B_i^p , que se chama um *conjunto de brackets*. Todos esses conjuntos são elementos de $\mathcal{B}^p$. Por conveniência, listamos os elementos de um conjunto de brackets de cima para baixo de uma dada árvore T_p . Por exemplo, $p = ((1, 2)((4, 3)5))$ tem $\mathcal{B}^p = \{B_1^p, B_2^p, B_3^p, B_4^p\}$, onde $B_1^p = \{B_2^p, B_3^p\}$, $B_2^p = \{1, 2\}$, $B_3^p = \{B_4^p, 5\}$, $B_4^p = \{4, 3\}$.

Observação 2. Para quaisquer dois conjuntos de brackets B_i^p e B_j^p que satisfaçam $B_i^p = \{B_j^p\}$, o conjunto de brackets B_j^p pode ser mantido em p e B_i^p removido de p , o que implica remover um par de brackets de p .

Árvores isomorfas Dadas duas sequências p e q , se existem conjuntos de bracket em que $B_i^p = B_j^q$, então esses brackets correspondem à mesma sub-árvore de T_p e T_q . Portanto, dizemos que $d(B_i^p, B_j^q) = 0$ uma vez que essas subárvores de T_p e T_q são iguais.

Teorema 3. Seja f uma função em que $f : \mathcal{B}^p \rightarrow \mathcal{B}^q$. A função f é uma bijeção em que $f(B_i^p) = B_j^q$ se e só somente se $d(T_p, T_q) = 0$.

Demonstração. Se existe uma bijeção $f : \mathcal{B}^p \rightarrow \mathcal{B}^q$ onde $f(B_i^p) = B_j^q$, então cada subárvore de p está relacionada com uma subárvore igual de q com os mesmos elementos, então para cada B_i^p existe B_j^q onde $d(B_i^p, B_j^q) = 0$. Isto implica que $d(T_p, T_q) = 0$. Inversamente, se $d(T_p, T_q) = 0$ implica que cada nó interno i de p tem um nó interno correspondente j de q , onde $B_i^p = B_j^q$. Portanto, esta correspondência é uma bijeção onde $f(B_i^p) = B_j^q$. $\square$

Teorema 4. Dado T_p e T_q , pode ser decidido em tempo polinomial se $d(T_p, T_q) = 0$.

Demonstração. Como consequência do Teorema 3, se $d(T_p, T_q) = 0$, então $T_p \approx T_q$, i.e. as árvores correspondentes são isomorfas. Se este for o caso, devemos analisar se cada conjunto de brackets de p é igual a um conjunto de brackets de q . Como o isomorfismo de árvores é determinado em tempo linear Campbell e Radford [1991], e encontrar dois conjuntos iguais também pode ser determinado em tempo linear, isto conclui a prova. $\square$

Dadas duas árvores isomorfas T_p e T_q , em que p e q têm o mesmo tamanho, para determinar $d(T_p, T_q)$, precisamos de encontrar uma função f que mapeie $\mathcal{B}^p$ para $\mathcal{B}^q$ de tal forma que se $B_i^p \neq B_j^q$ e $f(B_i^p) = B_j^q$, então temos de contar as remoções (ou inserções) de elementos num conjunto de brackets considerando os caminhos desses elementos através de T_p . Assim, para determinar a distância entre T_p e T_q , a função f deve ser a que minimiza os tamanhos dos caminhos dos elementos através de T_p para transformar em T_q .

Uma operação sobre uma árvore T_p pode ser vista como o deslocamento de uma subárvore do seu nível para um nível acima ou abaixo em T_p , onde essa subárvore pode ser um nó interno com todos os seus descendentes ou um único nó folha de T_p . Assim, a distância entre T_p e T_q , dadas T_p e T_q árvores de duas sequências do mesmo comprimento, é:

$$d(T_p, T_q) = \min_{f: \mathcal{B}^p \rightarrow \mathcal{B}^q} \left\{ \sum_{i,j} d(f(B_i^p), B_j^q) \right\}, \quad (1)$$

onde B_i^p está associado a B_j^q , e $d(f(B_i^p), B_j^q)$ é o número de operações sobre T_p relativamente a B_i^p e B_j^q .

Dado, como instância, $p = ((1(2(3, 4)))(5, 6))$ e $q = ((1(2(5, 4)))(3, 6))$. O conjunto de bracket de p é $\mathcal{B}^p = \{B_1^p, B_2^p, B_3^p, B_4^p, B_5^p\}$, para $B_1^p = \{B_2^p, B_3^p\}$, $B_2^p = \{1, B_4^p\}$, $B_3^p = \{5, 6\}$,

$B_4^p = \{2, B_5^p\}$, $B_5^p = \{3, 4\}$. O conjunto de bracket de q é $\mathcal{B}^q = \{B_1^q, B_2^q, B_3^q, B_4^q, B_5^q\}$, para $B_1^q = \{B_2^q, B_3^q\}$, $B_2^q = \{1, B_4^q\}$, $B_3^q = \{3, 6\}$, $B_4^q = \{2, B_5^q\}$, $B_5^q = \{5, 4\}$. Assim, vamos considerar $f(B_i^p) = B_i^q$, para $i = 1, \dots, 5$. Visto que $B_3^p \neq B_3^q$ e $B_5^p \neq B_5^q$, é necessário para criar caminhos levando os elementos 3 de B_5^p até B_3^p e levando os elementos de 5 de B_3^p para B_5^p . Esses caminhos são simulados em T_p primeiro passando o elemento 3 de B_5^p para B_4^p , e então de B_4^p para B_2^p , e então de B_2^p para B_1^p , e então de B_1^p para B_3^p , obtendo então $((1(2(4)))(3, 5, 6))$. Aplicando mais 4 operações para mover o elemento 5 de B_3^p para B_5^p finalmente obtemos T_q . Portanto, $d(T_p, T_q) \leq 8$.

Considerando duas árvores isomorfas, depois de mapear os conjuntos de brackets, descrevemos agora como mover elementos para ramos distintos de uma árvore, ilustrado em Figura 2. Dadas T_p e T_q árvores isomorfas, suponha que a raiz de T_p contém num ramo B_1 os elementos a e b em que b está mais distante que a da raiz, e noutro ramo B_2 contém os elementos c e d , em que d está mais distante que c da raiz. Além disso, a e b precisam de se mover de B_1 para B_2 , enquanto c e d precisam de se mover de B_2 para B_1 . Movemos b para estar ao mesmo nível que a , e depois movemos ambos os elementos juntos para B_2 . Da mesma forma, movemos d para estar ao mesmo nível que c , e depois movemos ambos os elementos juntos para B_1 . Ao aplicar esses movimentos recursivamente, determinamos a distância entre duas árvores pela quantidade desses movimentos. Figura 3 mostra como mover elementos de um ramo para outro. O processo é iniciado de forma bottom-up, com cada camada definida em termos do nó folha mais distante da raiz da árvore. Todos os elementos de k ($v_1, v_2, \dots, v_{k-1}, v_k$) devem ser movidos de B_1 para B_2 .

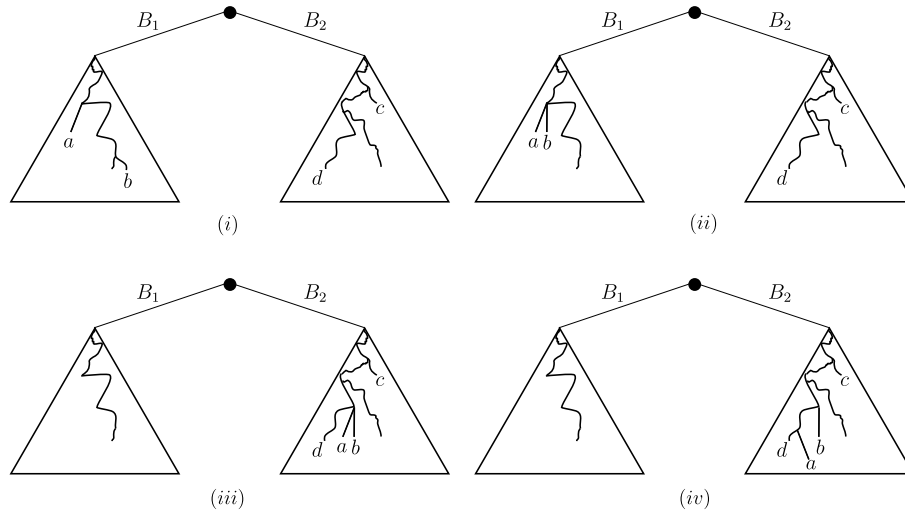


Figura 2: Ilustração de (i) para (iv) dos movimentos associados com os elementos a e b de B_1 para B_2 . Os movimentos de c e d funcionam de forma similar.

Árvores genéricas De forma genérica, dadas duas árvores T_p e T_q , para determinar $d(T_p, T_q)$, determinamos primeiro uma variação da SUBÁRVORE COMUM MÁXIMA (MCS - maximum common subtree) entre T_p e T_q , chamada $MCS(T_p, T_q)$. O MCS é uma restrição do bem conhecido e geral LARGEST COMMON SUBGRAPH, um problema NP-completo Garey e Johnson [1979], mas o MCS é resolvível em tempo polinomial uma vez que os grafos de entrada são árvores Lozano e Valiente [2004].

Quando é dado um nó v numa árvore T , a subárvore enraizada em v com todos os seus nós descendentes é designada por v -árvore. É importante destacar que uma v -árvore é uma subárvore

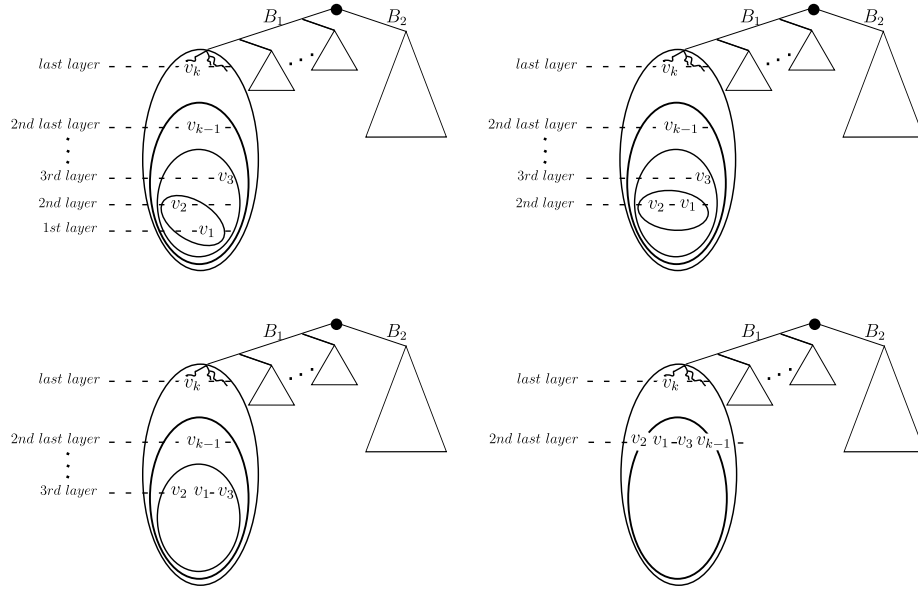


Figura 3: Ilustração de alguns movimentos associados com as camadas, onde em um processo bottom-up combinamos as folhas de v_1 para v_k e movemos todas juntas para B_1 e B_2 .

de T , mas nem todas as subárvores de T com raiz em v são v -árvores. Isto porque uma subárvore pode não incluir todos os nós num caminho entre a raiz v e uma folha descendente de v em T .

Definição 1. Uma quase v -árvore de um nó $v \in T$ consiste de:

1. Considere-se um nó interno $v \in T$ com k filhos $v_1, v_2, \dots, v_k$. Assim, obtenha a sua árvore v , e entre as árvores v_i , para $i = 1, 2, \dots, k$, remova j delas, para $0 \leq j \leq k$. Se $j = 0$, então nenhum filho foi removido;
2. A árvore resultante é designada por $H_v^{v_1, \dots, v_j}$ de T , se $v_1, \dots, v_j$ forem os j filhos de v removidos da árvore v de T , para $j \in \{1, 2, \dots, k\}$. Adicionalmente, $H_v^{v_0}$ indica que nenhum filho foi removido da árvore de v , i.e. $H_v^{v_0}$ é igual à árvore de v .

$H_v^{v_1, \dots, v_j}$ é uma quase v -árvore de v .

O nosso objetivo é determinar a QUASE v -ÁRVORE MÁXIMA COMUM, que é uma restrição do problema MCS.

QUASE v -ÁRVORE MÁXIMA COMUM (MCAT - MAXIMUM COMMON ALMOST v -TREE)

Instância: Duas árvores, T_p e T_q , com formato Newick associadas p e q , de mesmo tamanho.

Objetivo: Obter $\text{MCAT}(T_p, T_q)$, que é uma quase v -árvore de T_p que é isomorfa a uma quase v' -árvore de T_q com o número máximo de folhas. A árvore de v e a árvore de v' devem conter o mesmo conjunto de folhas.

Visto que cada x -árvore tem folhas rotuladas, é importante notar que a posição das folhas da v -árvore pode não ser a mesma que a da árvore v' , mesmo sabendo que elas têm os mesmos conjuntos de folhas.

Depois de obter uma solução para o problema MCAT, determinamos a distância entre a v -árvore e a v' -árvore determinando $d(T_p[H_v^{v_1, \dots, v_j}], T_q[H_{v'}^{v'_1, \dots, v'_j}])$, que pode ser obtida a partir de Equação 1. Em seguida, é necessário verificar se os nós v ou v' foram previamente rotulados. A presença de um rótulo indica que já ocorreu uma contração com um dos dois como ponto de partida. No caso em que o nó v tem um rótulo e v' não, temos de mover todos os nós de $H_{v'}^{v'_1, \dots, v'_j}$ em T_q para que o nó em T_q com o mesmo rótulo que v seja a raiz deles. Se o contrário ocorrer, o mesmo processo deve ocorrer, movemos todos os nós de $H_v^{v_1, \dots, v_j}$ para T_p para ter o nó em T_p com o mesmo rótulo que v' como sua raiz. No caso em que ambos v e v' tenham rótulos diferentes, fazemos este movimento com base naquele que requer o menor número de passos. Note que o valor de $d(T_p[H_v^{v_1, \dots, v_j}], T_q[H_{v'}^{v'_1, \dots, v'_j}])$ deve ser aumentado em uma unidade para cada passo para mover todos os nós.

Finalmente, é seguro reduzir as árvores de entrada por contrações de $H_v^{v_1, \dots, v_j}$ em T_p e $H_{v'}^{v'_1, \dots, v'_j}$ em T_q , que é uma substituição destas árvores como um nó, denotado por $T_p/H_v^{v_1, \dots, v_j}$ e $T_q/H_{v'}^{v'_1, \dots, v'_j}$, respectivamente. Adicionalmente, é também necessário rotular v e v' com algum valor não utilizado anteriormente, para que tenhamos uma indicação de que a mesma contração começou a partir deles. Figura 4 ilustra um exemplo.

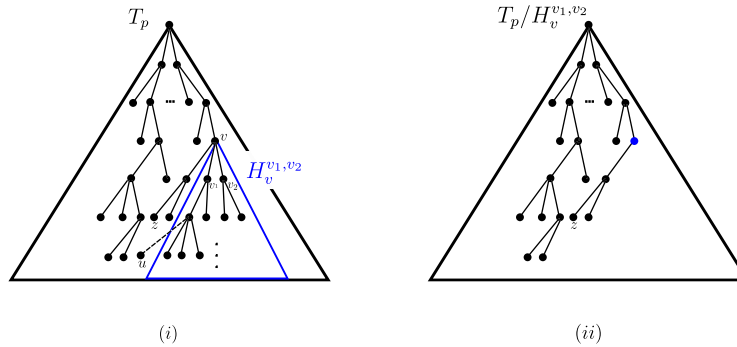


Figura 4: (i) A árvore solução para MCAT é mostrada em azul, com a sua raiz sendo o nó v e v_1 e v_2 como seus filhos. Note que não existe nenhum nó u em T que não esteja na solução e que o seu caminho para chegar a v passe por um descendente de v . (ii) O nó a azul é o nó representativo de $H_v^{v_1, \dots, v_j}$ após a sua contração. Além disso, cada descendente de v que não faz parte da solução do MCAT mantém o mesmo que em (i), bem como o nó z .

Note que não há nenhum nó folha em $T_p - H_v^{v_0, v_1, \dots, v_j}$ que deva passar por qualquer nó em $H_v^{v_0, v_1, \dots, v_j}$ para ser corrigido em relação a T_q , porque todos os descendentes dos filhos de v que pertencem a $H_v^{v_0, v_1, \dots, v_j}$ também pertencem a $H_{v'}^{v'_0, v'_1, \dots, v'_j}$. Assim, a operação de contração preserva a distância entre as árvores, exceto para as operações internas em $H_v^{v_0, v_1, \dots, v_j}$, que devem ser transformadas em $H_{v'}^{v'_0, v'_1, \dots, v'_j}$, porque as folhas únicas que têm de ser corrigidas e passam por $H_v^{v_0, v_1, \dots, v_j}$ passam necessariamente de forma única por v , que é representado pelo nó identificado após a contração.

Agora, vamos descrever como obter a solução do MCAT (em um processo top-down):

1. Para cada nó v de T_p , obtenha o conjunto S com alguns nós que são filhos de v entre $v_1, v_2, \dots, v_k$, se for possível, de tal modo que esses $|S|$ irmãos escolhidos correspondam a $|S|$ nós irmãos em T_q , chamados S' . de tal modo que existe uma bijeção de S em S' criando uma correspondência para cada v_i -árvore a uma v'_i -árvore, onde $v_i \in S$ e $v'_i \in T_q$. Note-se



Figura 5: Caso base da recursão considerando T_p em (i) e T_q em (ii). Após contrair H , existe um nó interno em (i) e três nós internos em (ii), e então 2 operações devem ser aplicadas.

que cada v_i -árvore pode ter uma quantidade diferente de folhas, cada uma com um rótulo diferente, e que uma v_i -árvore específica em T_p não precisa de ter as mesmas folhas que a sua árvore isomórfica correspondente em T_q . No entanto, todas as folhas em S têm de existir no conjunto correspondente em T_q .

2. A solução é aquela que tem o número máximo de folhas.

Visto que para cada nó de T_p temos que comparar a sua v -árvore com todos os nós de T_q , o tempo de execução é $O(n^2)$, onde T_p e T_q contêm $O(n)$ nós. A estratégia completa para calcular a distância entre duas árvores é descrita em Teorema 5.

Teorema 5. Dado T_p e T_q , $d(T_p, T_q)$ pode ser determinado em tempo polinomial.

Demonstração. Determinamos a distância entre T_p e T_q obtendo H que é a solução de $\text{MCAT}(T_p, T_q)$. Assim:

$$d(T_p, T_q) = d(T_p[H], T_q[H]) + d(T_p/H, T_q/H) + |n(T_p) - n(T_q)|, \quad (2)$$

onde $n(T_p)$ (análogo a $n(T_q)$) é o número de nós em T_p (em T_q), G/H_v é o grafo resultante após a contração de H_v . Como o conjunto de folhas de $T_p[H]$ é igual ao conjunto de folhas de $T_q[H]$, então Equação 2 utiliza Equação 1 para calcular $d(T_p[H], T_q[H])$, e depois calculamos recursivamente a distância entre T_p/H e T_q/H . Em cada chamada do problema MCAT, uma sub-árvore solução H é uma quase v -árvore, comum em ambas as árvores, portanto tendo o mesmo número de folhas e nós internos. Portanto, como consequência do problema MCAT, quando a chamada recursiva atinge um caso base, existem $|n(T_p) - n(T_q)|$ nós internos contidos em T_p (em T_q) que não estão contidos em T_q (em T_p). Para cada um desses nós, é necessário realizar uma operação extra, removendo um nó interno de cada um deles (como ilustrado em Figura 5). Visto que a solução do MCAT pode ser determinado em tempo quadrático e as recursões utilizam uma solução do MCAT no máximo $n(T_p)$ vezes, e com isso podemos computar a distância entre as árvores da entrada em tempo cúbico, no pior caso. $\square$

Note que se $H = T_p$, então $T_p \approx T_q$, e Equação 2 implica exatamente em Equação 1.

Tempos de execução . Se estivermos lidando com árvores que têm altura igual a 2, então a distância pode ser calculada analisando a distância de swap entre as permutações correspondentes. O código apresenta complexidade de tempo linear uma vez que a distância de swap também é calculada em tempo linear. Caso contrário, calculamos as distâncias por entre as árvores correspondentes usando a fórmula recursiva obtida na prova de Teorema 5, que pode ser efetuada em tempo de execução $O(n^3)$.

4. Experimentos e trabalhos futuros

Em relação ao problema de CLOSEST, por meio de Teorema 6, mostramos que ele é NP-completo para alturas maiores ou igual a 2, entretanto, ainda não existe uma solução para o problema da MEDIANA em relação ao seu tempo de execução, sendo considerado ainda um problema aberto.

Teorema 6. *O problema de decisão para o problema de CLOSEST de um conjunto de árvores é NP-completo, mesmo que a altura das árvores de entrada seja igual a 2.*

Demonstração. Popov [Popov, 2007] mostrou que SWAP-CLOSEST PROBLEM é NP-completo. Uma operação de permutação pode ser vista como uma operação numa árvore de altura 2, uma vez que cada permutação ou quebra ou funde ciclos. Assim, uma permutação é uma solução do SWAP-CLOSEST PROBLEM se e somente se a sua representação de ciclos for uma solução do problema CLOSEST no formato de Newick, como consequência de Teorema 2. $\square$

Inicialmente, para calcular as distâncias entre árvores, consideramos como entradas duas cadeias de caracteres no formato Newick para representar duas árvores. Para estudar os problemas CLOSEST e MEDIANA, desenvolvemos dois algoritmos, que descrevemos de seguida, para analisar mais do que duas árvores de entrada.

Generalizando o Quase v -árvore Máxima Comum Para o primeiro algoritmo, a abordagem consiste em generalizar o problema QUASE v -ÁRVORE MÁXIMA COMUM definido em Subseção 3.2 considerando agora um conjunto de $k \geq 2$ árvores de entrada. A ideia principal é derivar o MCAT do conjunto de k árvores de entrada. Isto é feito contraindo a solução MCAT H em cada árvore de entrada e construindo simultaneamente o conjunto H , parte da árvore de solução T^* . Enquanto a solução ainda não está completa, este passo anterior continua com as restantes árvores.

Uma sub-árvore H' que obtemos num passo MCAT é estabelecida numa posição de T^* da seguinte forma. Seja v o nó ancestral mais frequente de H' entre as árvores de entrada: Se v já estiver em T^* , então H' é definido como sendo um descendente de v em T^* ; Se v ainda não foi incluído em T^* , adicionamos H' a um conjunto auxiliar que será incluído em T^* assim que v for colocado nesse conjunto.

Finalmente, para cada subárvore de solução H' de um passo MCAT, definimos os valores das suas folhas na árvore de solução T^* para estarem na mesma ordem que a ordem mais frequente entre as subárvores de entrada H' .

Agora iremos argumentar que o nó ancestral mais frequente de H' entre as árvores de entrada pertence à árvore de solução T^* e, portanto, esse nó v pode ser considerado o nó ancestral de H' em T^* (como descrito acima). Suponha que H' é uma solução de MCAT entre todas as árvores de entrada. Além disso, assume-se que o nó ancestral mais frequente de H' é v , mas que v não pertence a T^* . Uma vez que v aparece nas árvores de entrada, pode concluir-se que v é ancestral de uma folha que não pertence a H' . Esta folha é denotada por i , com $i \in \{1, \dots, n\}$. Uma vez que i tem de pertencer a T^* , pois é uma folha, tem de haver um passo de aplicação de um MCAT sobre as restantes árvores, depois de obter H' , que inclua i numa solução.

Após a conclusão, obtivemos T^* que é a árvore candidata a solução para os problemas CLOSEST e MEDIANA. Isto é conseguido tomando a distância máxima entre T^* e todas as árvores de entrada e a soma das distâncias entre T^* e as outras árvores, respetivamente.

Árvore filogenética Uma árvore filogenética é uma representação gráfica, sob a forma de uma árvore, das relações evolutivas entre entidades biológicas, geralmente sequências ou espécies [Kapli et al., 2020]. Cada folha representa uma espécie distinta e cada nó interno desta árvore representa

o antepassado mais comum de todos os descendentes a partir desse ponto. Assim, o nó raiz corresponde ao antepassado mais comum de todas as folhas das espécies na árvore.

A partir deste conceito, o segundo algoritmo recebe como entrada k árvores, onde k é um número arbitrário, e tenta replicar a mesma ideia da árvore filogenética. Entre todos os pares de árvores, selecione aquele que é o par mais próximo entre todas. Sejam k_1 e k_2 esse par. Depois de selecionar ambas as árvores, substitua-a por outra árvore j que pertença a um caminho mais curto entre k_1 e k_2 , e $d(j, k_1) = d(j, k_2)$. O passo seguinte consiste em considerar a árvore j em vez de k_1 e k_2 , e selecionar o par mais próximo entre as árvores $k - 1$ restantes. Este processo faz com que o número de entradas diminua um de cada vez, e a sua repetição $k - 1$ vezes resulta numa árvore singular, que é a árvore candidata a solução para os problemas CLOSEST e MEDIANA.

Experimentos Os experimentos foram feitos em um computador com 32 GB RAM e 66 GB HDD. Os testes aconteceram em uma máquina virtual tendo 4 GiB RAM e 8 Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz. As implementações para cada algoritmo descrito anteriormente foram feitos para que possamos calcular o tempo de execução de ambos além de comparar as diferentes soluções encontradas para os problemas de CLOSEST e MEDIANA.

Para efetuar os cálculos, geramos 100 instâncias distintas, sendo que cada uma delas continha 3 strings que descreviam as árvores com base no formato de Newick. Essas 100 instâncias foram criadas para 2 tamanhos diferentes, onde esse tamanho é determinado pelo número de folhas. Os valores selecionados foram 10 e 20. A altura foi limitada pelo número de folhas, a altura máxima é de no máximo $\log n$, sendo n o número de folhas. O primeiro algoritmo se mostrou mais rápido para a solução dos problemas de CLOSEST e MEDIANA, como se pode observar pelas tabelas.

Tabela 1 mostra os resultados para o Quase v -árvore Máxima Comum e o Tabela 2 os resultados para o algoritmo baseado na árvore filogenética. A primeira coluna de ambas as tabelas mostra o número de folhas para o teste; a segunda é o tempo médio para computar todas as 100 instâncias; a terceira é a média das distâncias entre a árvore encontrada pelo algoritmo e o limite inferior dado pela consequência da desigualdade triangular e diz que a mediana f para as permutações π_1, π_2, π_3 satisfaz $f \geq \frac{d_M(\pi_1, \pi_2) + d_M(\pi_1, \pi_3) + d_M(\pi_2, \pi_3)}{2}$; a última coluna mostra a diferença entre a distância máxima da árvore encontrada pelo algoritmo e as árvores de entrada e a distância máxima entre todas as entradas.

Como trabalho futuro pretendemos focar na seguinte abordagem construtiva. Dado um conjunto de árvores imperfeitas, queremos analisar se o consenso representa uma opção verdadeira melhor. Esse procedimento pode ser feito gerando dados do tipo célula única e construindo o consenso de um conjunto de árvores por outros métodos conhecidos e Métodos de Monte Carlo via Cadeias de Markov (MCMC) Jahn et al. [2016].

	Time (s)	Mediana	Closest
10	0.04832	1.04	1.19
20	94.4141	1.23	1.30

Tabela 1: Resultados médios para o algoritmo da Quase v -árvore Máxima Comum generalizada.

	Time (s)	Mediana	Closest
10	0.3392	3.54	2.31
20	472.0339	4.26	3.28

Tabela 2: Resultado médio para o algoritmo baseado em árvores filogenéticas.

Referências

- Bader, M. (2011). The transposition median problem is NP-complete. *Theor. Comput. Sci.*, 412 (12-14):1099–1110.
- Campbell, D. M. e Radford, D. (1991). Tree isomorphism algorithms: Speed vs. clarity. *Mathematics Magazine*, 64(4):252–261.
- Caprara, A. (2003). The reversal median problem. *INFORMS J. Comput.*, 15(1):93–113.
- Cardona, G., Rosselló, F., e Valiente, G. (2008). Extended newick: it is time for a standard representation of phylogenetic networks. *BMC bioinformatics*, 9(1):1–8.
- Cunha, L. F. I., Feijão, P., dos Santos, V. F., Kowada, L. A. B., e de Figueiredo, C. M. (2020). On the computational complexity of closest genome problems. *Discrete Applied Mathematics*, 274: 26–34.
- Cunha, L. F. I. e Protti, F. (2019). Genome rearrangements on multigenomic models: Applications of graph convexity problems. *J. Comput. Biol.*, 26(11):1214–1222.
- Fertin, G., Labarre, A., Rusu, I., Vialette, S., e Tannier, E. (2009). *Combinatorics of Genome Rearrangements*. MIT press.
- Garey, M. R. e Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-completeness*.
- Haghighi, M. e Sankoff, D. (2012). Medians seek the corners, and other conjectures. In *BMC bioinformatics*, volume 13, p. 1–7. Springer.
- Jahn, K., Kuipers, J., e Beerenwinkel, N. (2016). Tree inference for single-cell data. *Genome Biology*, 17(1):86.
- Kapli, P., Yang, Z., e Telford, M. J. (2020). Phylogenetic tree building in the genomic age. *Nature Reviews Genetics*, 21(7):428–444.
- Lanctot, J. K., Li, M., Ma, B., Wang, S., e Zhang, L. (2003). Distinguishing string selection problems. *Inf. Comput.*, 185(1):41–55.
- Lozano, A. e Valiente, G. (2004). On the maximum common embedded subtree problem for ordered trees. *String Algorithmics*, p. 155–170.
- Pe’er, I. e Shamir, R. (1998). The median problems for breakpoints are NP-complete. In *Elec. Colloq. on Comput. Complexity*, volume 71.
- Popov, V. Y. (2007). Multiple genome rearrangement by swaps and by element duplications. *Theor. Comput. Sci.*, 385(1-3):115–126.
- Watterson, G. A., Ewens, W. J., Hall, T. E., e Morgan, A. (1982). The chromosome inversion problem. *J. Theor. Biol.*, 99(1):1–7.