

Tutorial 3

Espaço de topologias

BIZ0433 - INFERÊNCIA FILOGENÉTICA: FILOSOFIA, MÉTODO E APLICAÇÕES.

Conteúdo

Objetivo	44
3.1 Requisitos de sistema	45
3.2 Conteúdo do diretório	45
3.2.1 Arquivos	45
3.3 Contextualização teórica	46
3.3.1 Topologias binárias e grafos	46
3.3.2 Enumeração	48
3.4 Explorando o espaço de topologias	49
3.4.1 Opção 1	49
3.4.2 Opção 2	54
3.4.3 Opção 3	54
3.5 Referências	56

Objetivo

Este tutorial apresenta os conceitos de grafos, enumeração e espaço de topologias. O objetivo do tutorial é apresentar os requisitos necessários para a compreensão da complexidade computacional associada às buscas de topologias de acordo com critérios de otimalidade. Durante o tutorial, você terá a oportunidade de executar uma série de exercícios cujos resultados gráficos lhe permitirão adquirir uma melhor compreensão dos conceitos em discussão. Os arquivos associados a este tutorial estão disponíveis em <http://lhe.ib.usp.br/cladistica>. Você pode baixá-los diretamente com o seguinte comando:

```
wget http://lhe.ib.usp.br/downloads/tutorial_03.zip
```

3.1 Requisitos de sistema

Este tutorial requer que seu computador possua os seguintes aplicativos:

- i. TNT: Esse programa [1] deve estar instalado em seu computador e executável pelo comando `$ tnt`.
- ii. R: O **R** é um ambiente livre de computação gráfica e estatística. Caso seu computador não possua o **R** instalado, você deverá fazê-lo pelo comando “`sudo apt-get install r-base-core`”. Adicionalmente você deverá instalar algumas bibliotecas que são necessárias para a execução deste tutorial: `scatterplot3d` e `vegan`. Para instalar essas bibliotecas você deverá abrir o **R** com o comando “`sudo R`” e executar o seguinte comando no *prompt* do **R**: “`install.packages("scatterplot3d")`”. O **R** solicitará que você escolha um repositório do qual ele deverá baixar o pacote solicitado. Você poderá escolher qualquer um dos países listados. Feita a escolha, o pacote começará a ser instalado.
A próxima biblioteca que deverá ser instalada é “`vegan`”. Use a mesma lógica acima.
- iii. Java VM: Um dos aplicativos utilizados nesse tutorial requer a instalação de **Java Virtual Machine**. O repositório do Ubuntu possui várias versões de Java disponível. Na imagem utilizada pelo curso foi instalado o pacote “`oracle-java7-installer`”.
- iv. YBYRÁ: O **YBYRÁ** [2] é um aplicativo desenvolvido por Denis J. Machado (Depto. Zoologia – IB/USP). O programa tem uma série de aplicações, tais como cálculo de distâncias topológicas, análise de sensibilidade, diagnose de clados, entre outras. Neste tutorial iremos usá-lo para o cálculo de distâncias entre topologias dentro do espaço de enumeração. Ao longo do curso, iremos usar esse mesmo programa para explorar outras aplicabilidades.

3.2 Conteúdo do diretório

3.2.1 ARQUIVOS

O diretório associado a este tutorial deverá ser baixado da página da disciplina. Nele você encontrará os seguintes itens:

```

-rw-rw-r-- 1 alan alan    1618079 Mar  2 13:02 03_tutorial.pdf
-rw-rw-r-- 1 alan alan      3359 Mar  2 13:02 06_enumeration.tre
-rw-rw-r-- 1 alan alan    35909 Mar  2 13:02 07_enumeration.tre
-rw-rw-r-- 1 alan alan   459269 Mar  2 13:02 08_enumeration.tre
-rw-rw-r-- 1 alan alan   6822899 Mar  2 13:02 09_enumeration.tre
-rw-rw-r-- 1 alan alan 115709579 Mar  2 13:02 10_enumeration.tre
-rw-rw-r-- 1 alan alan    2206 Mar  2 13:02 graph_data_dist.r

```

```

-rw-rw-r-- 1 alan alan      1514 Mar  2 13:02 graph_data.r
drwxrwxr-x 3 alan alan      4096 Mar  2 13:02 literatura
drwxrwxr-x 7 alan alan      4096 Mar  2 13:07 MSdist
-rw-rw-r-- 1 alan alan     10450 Mar  2 13:02 tree_space.pl
-rwxr--r-x 1 alan alan     51504 Mar  2 13:44 ybyra_sa.py

```

- i. *_enumeration.tre (linhas 2 a 6): Estes arquivos contém a enumeração das topologias para N táxons, variando de 6 a 10.
- ii. graph_data.r (linhas 7 e 8): São as rotinas em **R** para a produção dos gráficos gerados pelo *script* tree_space.pl.
- iii. Literatura: Diretório que contém alguns artigos associados com o tema desse tutorial.
- iv. MSdist: Diretório que contém o aplicativo MSdist (v. 0.5, documentação em [3]) que calcula distância entre topologias utilizando *Matching Cluster (MC) distance* (veja [4]).
- v. tree_space.pl: O *script* tree_space.pl gera matrizes e topologias, faz buscas em TNT, compila resultados e alimenta as rotinas de **R** para gerar os gráficos. Nos exercícios a seguir iremos usar as rotinas implementadas neste *script* para explorar algumas propriedades relacionadas com o espaço de topologias.

3.3 Contextualização teórica

3.3.1 TOPOLOGIAS BINÁRIAS E GRAFOS

As discussões levantadas por Mindell [5] e Morrison [6] com relação ao modelo de representação gráfica geralmente adotado em estudos filogenéticos é muito interessante. Os argumentos expressos por Morrison [6], principalmente, sugere que devemos refletir se o modelo que estamos adotando é apropriado para o estudo que estamos interessados. De qualquer forma, atualmente, topologias binárias são prevalentes em estudos filogenéticos e embora isso possa mudar no futuro, devemos explorar algumas de suas propriedades – mesmo porque o curso fará uso exclusivo desses diagramas em inferência filogenética.

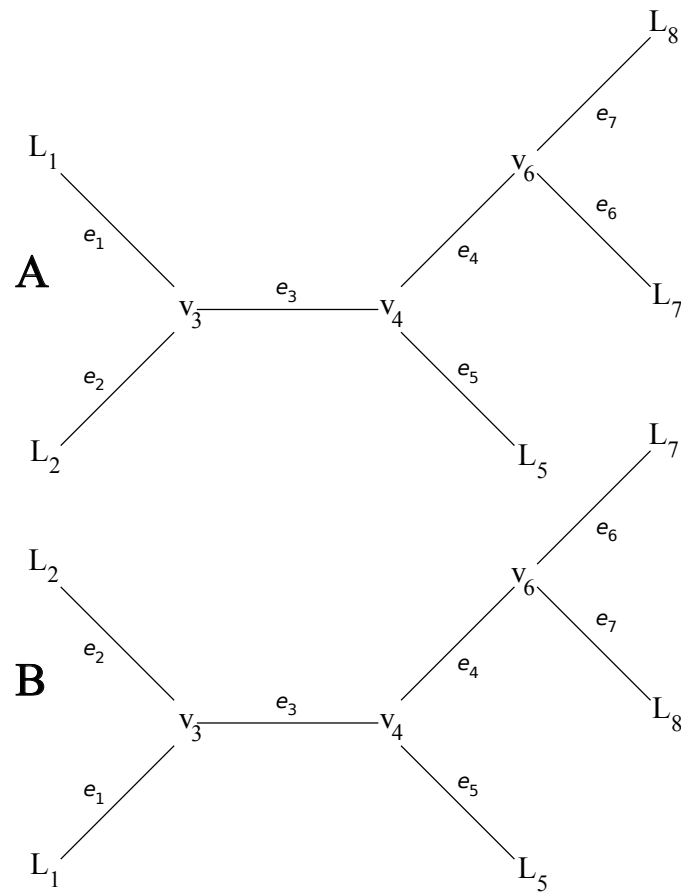


Figura 3.1: Dois grafos (A e B) com a mesma topologia expressa pelas relações entre vértices (*i.e.*, V), terminais (*i.e.*, L) e arestas (*i.e.*, e).

Grafos são objetos matemáticos que consistem de um par de conjuntos $(V/L, E)$ de vértices (nós, V/L) e arestas (ramos, E ; veja Figura 3.1). O grau de um vértice é o número de arestas conectados a ele. Desta forma, uma topologia binária $T = (V/L, E)$ é um grafo conectado sem ciclos em que os terminais (L) são vértices de grau 1 ao passo que os nós são vértices de grau 3.

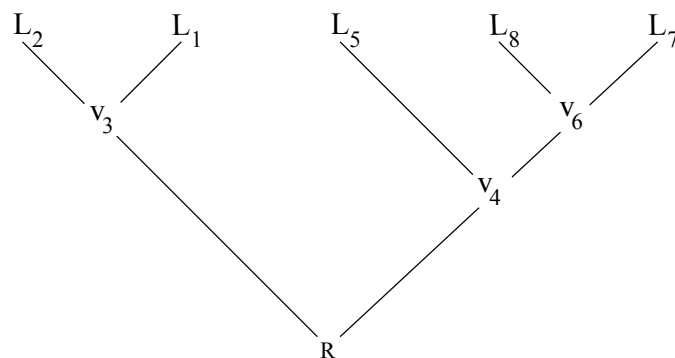


Figura 3.2: Grafo acíclico direcionado. Direcionamento é dado pela raiz (R), único vértice de graus 2.

O enraizamento de uma topologia binária (Figura 3.2) se dá pela inserção de um vértice de grau 2 (*i.e.*, R). Cosequentemente, diagramas enraizados possuem um vértice e uma aresta adicional. Diagramas binários não-enraizados possuem ainda algumas propriedades matemáticas que devem ser conhecidas. A primeira delas é que o número de vértices internos é igual a $|L| - 2$. A segunda

é que o número de arestas (*i.e.*, ramos) é igual a $2 * |L| - 3$. Finalmente, o número de topologias possíveis é uma função de $|L|$, ou seja, número de terminais. Portanto, o conjunto de topologias pode ser enumerado.

3.3.2 ENUMERAÇÃO

Em matemática e na ciência da computação, enumeração é a lista de todos os elementos de um conjunto. Em cladística, o termo é aplicado em buscas exatas no qual todas as topologias possíveis são avaliadas (ver Tutorial 4). A enumeração de topologias binárias não-enraizadas obedecem a seguinte fórmula:

$$\text{para } n \geq 3: \frac{(2n-4)!}{(n-2)!2^{n-2}}$$

O número de topologias enraizadas pode ser calculado multiplicando essa a fórmula acima pelo o número de ramos ($2n - 3$) ou incrementado n por 1 (veja Tabela 3.1).

Tabela 3.1: Número de topologias binárias em função do número de terminais (*i.e.*, n).

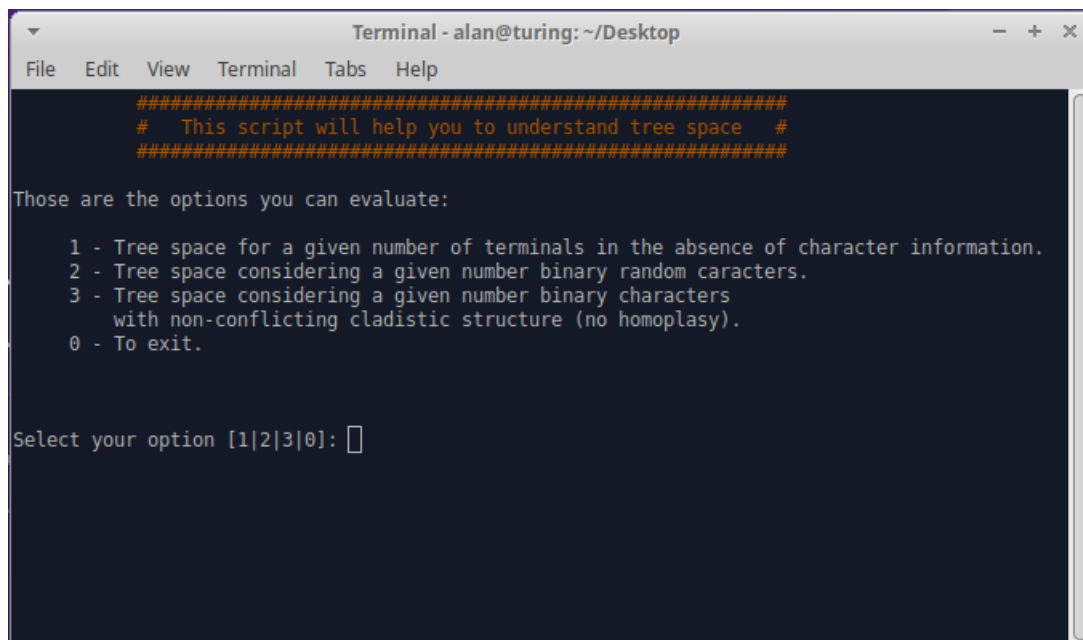
n	não-enraizada	enraizada
3	1	3
4	3	15
5	15	105
6	105	945
7	945	10395
8	10395	135135
9	135135	2027025
10	2027025	34459425
11	34459425	654729075
12	654729075	13749310575
13	13749310575	316234143225
14	316234143225	7905853580625
15	7905853580625	213458046676875

Como pode ser observado, o número de topologias incrementa exponencialmente à medida que adicionamos terminais. Neste tutorial, o número de diagramas binários é um dos parâmetros que define o espaço de topologias (*i.e.*, *tree space*). Observe que até este momento, estas topologias são simplesmente objetos matemáticos, cujas características são relevantes para análises filogenéticas. A transição entre estes objetos e o conceito de teste de hipóteses requer análise de mérito relativo destes objetos. O mérito relativo depende de como esses modelos explicam nossas observações. Desta, a transição entre objetos meramente matemáticos (“*tree-shaped-objects*”, senso [7]) e sua qualidade relativa para explicar (*i.e.*, distribuição de caracteres)

irá depender da implementação de critérios de otimalidade – o que será explicado no próximo tutorial.

3.4 Explorando o espaço de topologias

A seguir, vamos explorar alguns conceitos relacionados com o espaço de topologias. Iremos fazer uma série de execuções do *script* `tree_space.py` e é importante que ao executá-las, você tome nota dos resultados (uma opção é salva as figuras geradas), pois você deverá compará-los à medida em que vai respondendo os exercícios. Este *script* possui três rotinas básicas que serão explicadas no decorrer deste tutorial (Figura 3.3).



```

Terminal - alan@turing: ~/Desktop
File Edit View Terminal Tabs Help

#####
# This script will help you to understand tree space #
#####

Those are the options you can evaluate:

  1 - Tree space for a given number of terminals in the absence of character information.
  2 - Tree space considering a given number binary random characters.
  3 - Tree space considering a given number binary characters
      with non-conflicting cladistic structure (no homoplasy).
  0 - To exit.

Select your option [1|2|3|0]: 

```

Figura 3.3: Opções de `tree_space.py`. Veja texto abaixo para a descrição de cada uma dessas opções.

3.4.1 OPÇÃO 1

Iremos iniciar avaliando como o número de terminais afeta o espaço de topologias na ausência de informação provinda de caracteres cladísticos. Na opção 1, este *script* avalia todas as topologias possíveis (*i.e.*, grafos binários) para determinado número de terminais $\geq 4 \leq 10$ ¹. Se você selecionar a opção 1 e pedir para o *script* avaliar o espaço de topologias para 5 terminais você deverá obter dois gráficos representados da Figura 3.4.

¹não exceda o limite superior desta rotina, pois o resultado demorará muito.

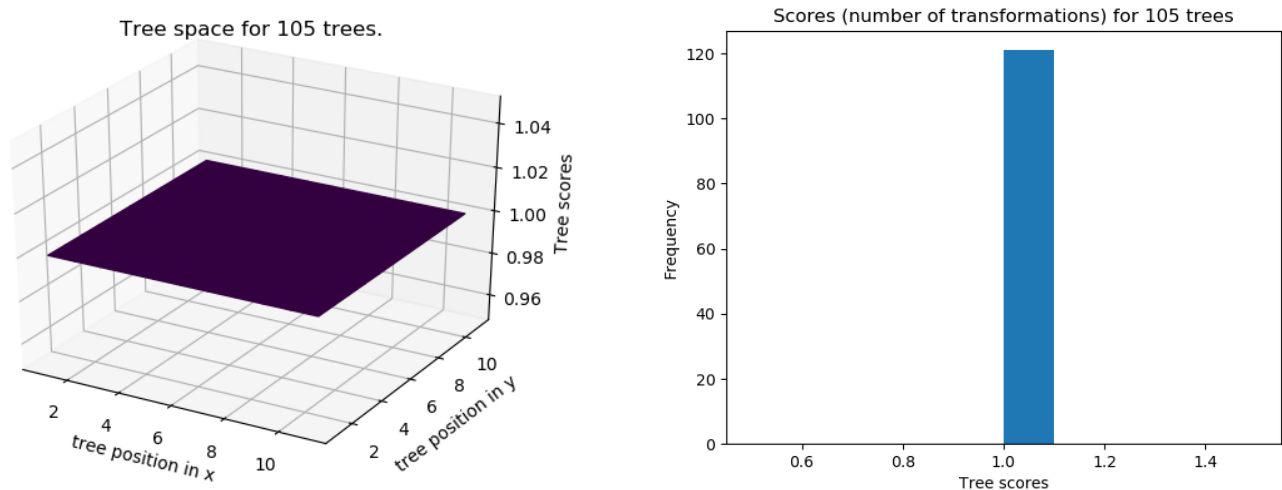


Figura 3.4: Gráficos associados à opção 1 considerando 6 terminais sem informação cladística. À direita você encontra o custo de todas as topologias e à direita a frequência de distribuição destes custos.

Nesta figura, à direita você encontra um gráfico no qual o eixo x e y representam as coordenadas de todas as topologias geradas e no eixo z seus respectivos custos (*Scores*)². Ao lado direito desta figura, há um histograma mostrando a distribuição dos custos das topologias encontradas.

Exercício 3.1

Considere estes gráficos como uma representação aproximada da complexidade do espaço de topologias. Faça algumas execuções deste *script* variando o número de terminais e responda:

- Há um único parâmetro deste espaço que é afetado pelo número de terminais. Qual seria esse parâmetro e de que forma o espaço de topologias está variando?

Como você pode observar, os custos de todas as topologias são idênticos. No entanto, considere a Figura 3.5.

²O “custo” refere-se ao número de transformações de estado de caráter para cada topologia.

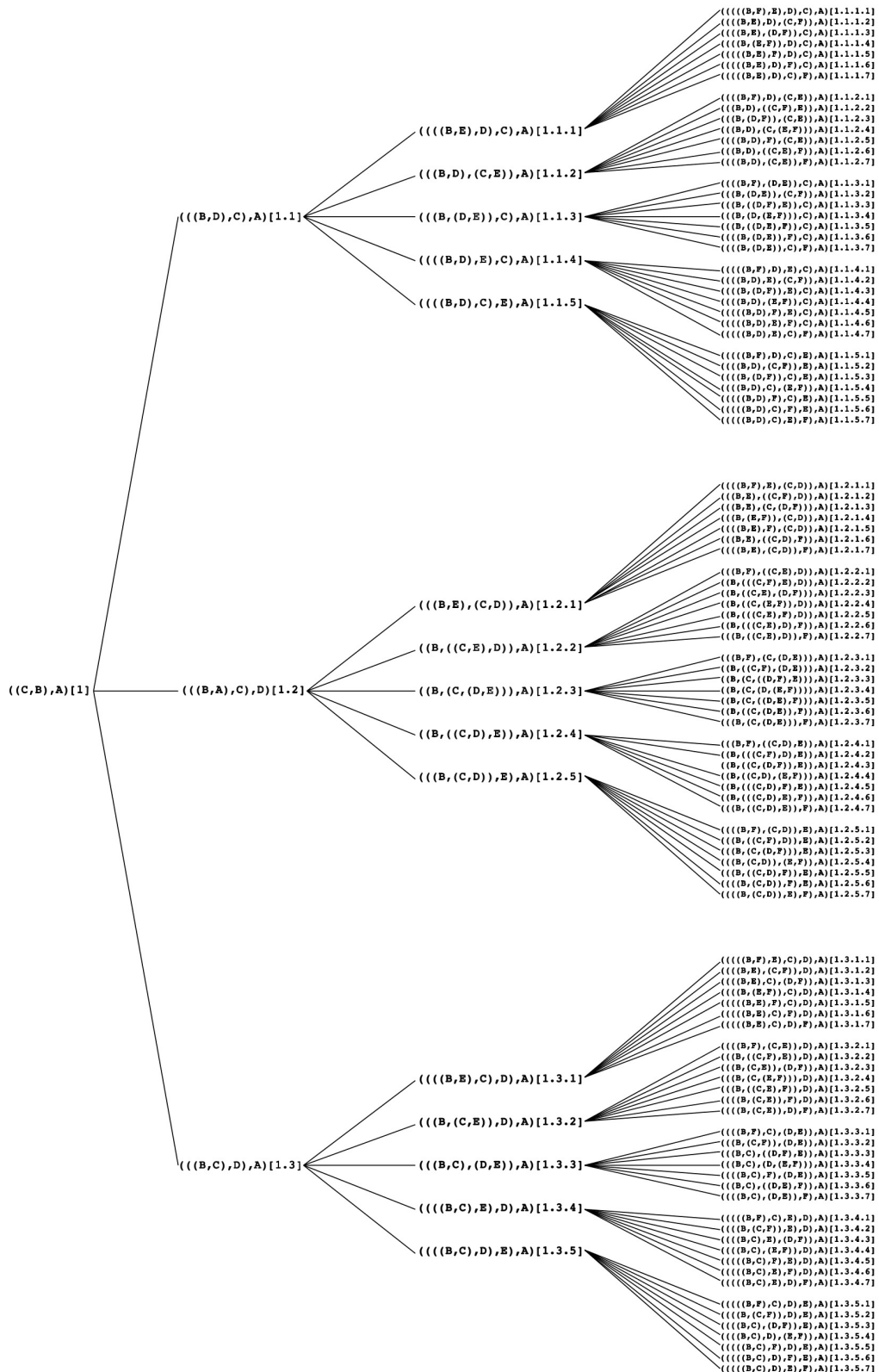


Figura 3.5: Enumeração das topologias para 6 terminais indexadas segundo o arquivo 06_enumeration.tre. Observe a lógica de indexação desta topologia pelos dígitos entre colchetes (*i.e.*, [1.3.5.1]).

Observe que essas topologias divergem à medida em que se afastam da topologia inicial (*i.e.*, [1]). É evidente que nosso primeiro exercício não detecta este componente do espaço de árvores. No entanto, isso seria possível avaliando a distância topológica de acordo com alguma outra métrica.

Robinson & Foulds [8] foram uns dos primeiros a propor uma métrica que avaliasse diferença entre árvores filogenéticas. Os detalhes do cálculo de distância são irrelevantes para os propósitos deste tutorial, mas o leitor mais curioso deve consultar o artigo de Robinson & Foulds [8] e Bogdanowicz & Giaro [4] para maiores detalhes (no diretório `literature` anexo a este tutorial). Via de regra, métricas que computam distâncias entre topologias consideram os passos necessários, de acordo com algum critério, para transformar uma topologia em outra. Neste tutorial iremos avaliar distâncias entre topologias utilizando o **YBYRÁ**, que usa a métrica proposta por Robinson & Foulds [8].

O uso do **YBYRÁ** para este propósito é relativamente simple. Suponha, por exemplo, que eu queira avaliar a distância entre a topologia `[1.1.1.1]` e as demais topologias enumeradas a partir de `[1.1.1]`. Os passos a seguir seriam:

- i. Extrair as topologias desta família (*i.e.*, `[1.1.1]`) do arquivo `06_enumeration.tre`. A forma mais simples de fazer isso seria utilizar a seguinte linha de comando:
`$ egrep '\[1\.1\.1\.*' 06_enumeration.tre > 1.1.1.n.tre`
- ii. Extrair a topologia `1.1.1.1.tre` de `1.1.1.n.tre` com a seguinte linha de comando:
`$ head -n 1 1.1.1.n.tre > 1.1.1.1.tre`
- iii. Criar o arquivo de configuração necessário para executar o **YBYRÁ**. Para computar as distâncias entre a topologia `1.1.1.1.tre` e as demais topologias em `1.1.1.n.tre`, o arquivo de configuração (*e.g.*, `conf.txt`) deve ter o seguinte conteúdo:

```
>id = CalcDist_6a
<begin files
    1.1.1.1.tre ;
    1.1.1.n.tre ;
end files >
>n = 1 [1.1.1.1.tre]
>opt = 3
>compare = 1
>root=A
>verbose
```

Neste arquivo de configuração, a linha 1 define a ID (identidade) da análise. As linhas 2 a 5 definem os arquivos que contém as topologias a serem comparadas. A linha 6 define a topologia de referência. As linhas 7 e 8 configuram o tipo de comparação que o programa irá executar (veja documentação do programa para maiores detalhes). A linha 9 informa o táxon de enraizamento – necessário para computar as distâncias. Finalmente, a linha 10 configura o programa para informar ao usuário as etapas que estão sendo executadas à medida em que o programa prossegue.

- iv. Executar o **YBYRÁ** da seguinte forma:

```
$ python ybyra_sa.py -f conf.txt
```

Após a execução do programa, você deverá obter, dentre várias linhas de saída, a seguinte informação:

```
The file MATRIX_calcDist_6a.txt was created.
```

Este arquivo contém os resultados que deseja e estão sumarizados na Tabela 3.2.

Tabela 3.2: Distâncias topológicas de acordo com Robinson & Foulds [8]. **SC**, Clados compartilhados (*i.e.*, comuns entre as duas topologias). **CC**, Clados combinados (*i.e.*, soma de todos os clados distintos encontrados em ambas topologias). **LD**, Distância local.

Tree No.	Tree No.	SC	CC	LD = $1 - (\frac{SC}{CC})$
1	2	5	5	0
1	3	2	8	0.75
1	4	3	7	0.57
1	5	4	6	0.33
1	6	4	6	0.33
1	7	3	7	0.57
1	8	2	8	0.75

Exercicio 3.2

Com base no exemplo acima, no qual foram calculadas as distâncias entre topologias responda:

- i. Como se comportam as distâncias entre a topologia 1.1.1.1.tre e aquelas enumeradas para as demais famílias (*i.e.*, 1.1.2 e 1.1.3)?

- ii. O tamanho do espaço de topologias influencia os resultados que você obteve no item anterior? (**OBS.:** considere que há topologias enumeradas para 7, 8, 9 2 10 terminais.)

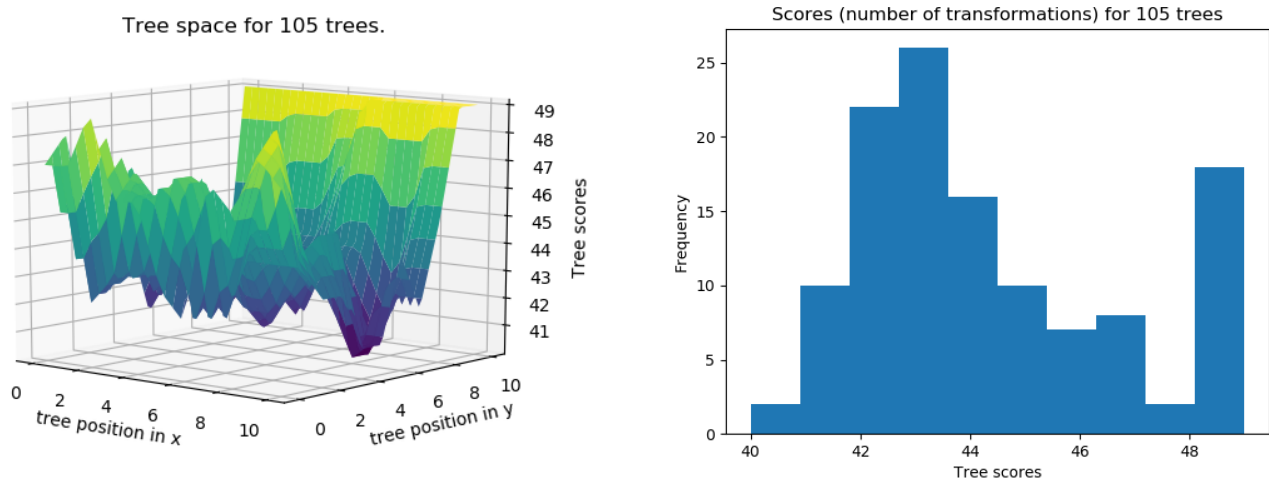


Figura 3.6: Gráficos associados à opção 2 considerando 6 terminais e 24 caracteres aleatórios. À direita você encontra o custo de todas as topologias e à direita a frequência de distribuição destes custos.

3.4.2 OPÇÃO 2

A opção 2 faz com que o *script* proceda da seguinte forma (Figura 3.6). Dado o número de terminais e caracteres, `tree_space.py` gera uma matriz aleatória com n caracteres binários cuja probabilidade de 0 ou 1 é igual a 0.50. Esta matriz de dados é submetida ao programa TNT para computar o custo de todas as topologias possíveis dado o número de terminais.

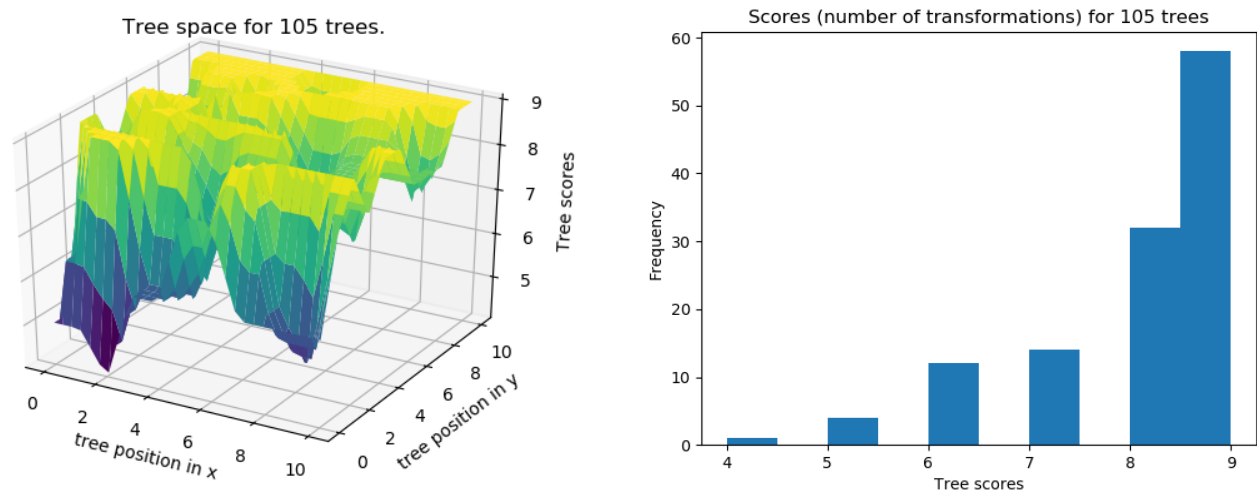


Figura 3.7: Gráficos associados à opção 2 considerando 6 terminais e uma matriz de representação (sem homoplasias). À direita você encontra o custo de todas as topologias e à direita a frequência de distribuição destes custos.

3.4.3 OPÇÃO 3

Esta rotina difere da anterior em alguns pontos importantes (Figura 3.7). Ao definir o número de terminais, `tree_space.py` gera uma topologia aleatória e uma matriz de representação. Por

esta razão, não há como definir o número de caracteres, pois estes são dependentes dos números de nós da topologia binária para os quais a matriz de representação dará suporte. Posteriormente, TNT avalia o custo desta matriz de representação em relação ao universo de enumeração de todas as topologias binárias.

Exercício 3.3

Você deverá fazer uma série de execuções das opções 2 e 3. Compare os resultados salvando as figuras se considerar necessário. Com base nos seus resultados, responda:

i. Como se comporta o espaço de topologias em ambas as rotinas?

ii. O comportamento dos valores de custo mínimo difere entre as duas rotinas?

iii. Qual é a observação mais relevante que você observa com relação aos histogramas produzidos por essas duas rotinas?

iv. Como você obteve algum resultado da opção 2 que resultou em uma única topologia com o menor custo? Você acha isso relevante para inferência filogenética?

Exercício 3.4

Abaixo, defina o que você entende por espaço de topologia e quais são os elementos que afetam sua conformação:

3.5 Referências

1. Goloboff, P.; Farris, J. & Nixon, K. 2008. TNT, a free program for phylogenetic analysis. *Cladistics* **24**(5): 774–786.
2. Machado, D. J. 2015. YBYRÁ, a fast and resourceful tool for sensitivity analysis. *BMC Bioinformatics Journal* **16**: 204.
3. Bogdanowicz, D. MSdist 0.5 – an application for computing the Matching Split distance between phylogenetic trees. <http://www.kaims.pl/~dambo/mcdist/>. Version 0.5. 2010.
4. Bogdanowicz, D. & Giaro, K. 2013. On a matching distance between rooted phylogenetic trees. *International Journal of Applied Mathematics and Computer Science* **23**: 669–684.

5. Mindell, D. P. 2013. The tree of life: metaphor, model, and heuristic device. *Systematic Biology* **62**(3): 479–489.
6. Morrison, D. A. 2014. Is the tree of life the best metaphor, model, or heuristic for phylogenetics? *Systematic Biology* **63**(4): 628–638.
7. Wheeler, W. C. 2012. Systematics: a course of lectures. Malaysia: Wiley-Backwell, 2012. 426.
8. Robinson, D. F. & Foulds, L. R. 1981. Comparison of phylogenetic trees. *Mathematical Biociences* **53**: 131–147.