Experimentos em rearranjo de genomas com a operação *Single-Cut-or-Join*

Este exemplar corresponde à redação da Dissertação apresentada para a Banca Examinadora antes da defesa da Dissertação.

Campinas, 08 de fevereiro de 2012.

João Meidanis (Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Instituto de Computação
Universidade Estadual de Campinas

Experimentos em rearranjo de genomas com a operação *Single-Cut-or-Join*

Priscila do Nascimento Biller[1]

Fevereiro de 2012

Banca Examinadora:

- João Meidanis (Orientador)
- Marília Dias Vieira Braga
  Grupo de Pesquisa em Bioinformática – INMETRO
- Zanoni Dias
  Instituto de Computação – UNICAMP
- João Carlos Setubal (Suplente)
  Instituto de Química – USP
- Guilherme Pimentel Telles (Suplente)
  Instituto de Computação – UNICAMP

[1]Suporte financeiro de: Bolsa do CNPq (processo147990/2010-6) 2010–2012.

Resumo

Os rearranjos são eventos evolutivos que alteram de diferentes formas a ordem de grandes segmentos do genoma. Explicar a história evolutiva de um conjunto de espécies com rearranjos pode ser visto como um problema de otimização computacional, chamado de Problema de Rearranjo de Múltiplos Genomas. Este problema consiste em encontrar uma árvore que relaciona o conjunto de genomas recebido, minimizando a soma dos pesos das arestas, sendo o peso de uma aresta o número de rearranjos que explica a evolução entre os genomas dos vértices incidentes. A qualidade da inferência e a complexidade do problema dependem da operação de rearranjo utilizada, que define formalmente como os genomas podem ser modificados.

Recentemente, uma nova operação de rearranjo foi proposta, o *Single-Cut-or-Join* (SCJ), que traz como grande vantagem a simplificação de muitos problemas, que sob outras operações são NP-difíceis. Apesar da teoria do SCJ ser bem construída, havia dúvidas sobre sua relevância biológica. Neste trabalho contribuímos com o entendimento desta operação, realizando um extenso estudo que aplica o SCJ sob diferentes condições evolutivas, com dados reais e simulados, analisando dois aspectos da reconstrução evolucionária: a estrutura da árvore e o genoma (ordem dos genes) das espécies ancestrais. Na primeira análise, descobrimos que o SCJ é capaz de recuperar entre 60% e 80% da estrutura da árvore. Em relação à segunda questão, dada a estrutura da árvore, a reconstrução dos genomas ancestrais varia conforme a distância da espécie ancestral para as espécies conhecidas. No caso de espécies ancestrais mais próximas às folhas, cerca de 85% da ordem dos genes foi coberta enquanto, em espécies mais distantes, aproximadamente 50% da ordem dos genes foi coberta, usando conjuntos de genomas de 64 espécies. Em relação ao tempo, os métodos, que implementamos em Java, podem encontrar a topologia de 64 genomas com 2000 genes cada em cerca de 10,7 minutos e reconstruir seus genomas ancestrais em 0,05 minutos, ambos em um computador desktop padrão.

Abstract

Rearrangements are evolutionary events that modify in different ways the order of large segments in genomes. To explain the evolutionary history of a set of species with rearrangements can be seen as an computational optimization problem, called Multiple Genome Rearrangement Problem. This problem consists in finding a tree which relates the set of genomes received, minimizing the sum of edge weights, where the weight of an edge is the number of rearrangements that explains the evolution between the genomes of incident vertices. The quality of the inference and complexity of the problem depend on the rearrangement operation used, which formally defines how the genomes can be modified.

Recently, a new rearrangement operation was proposed, Single-Cut-or-Join (SCJ), which brings a significant advantage in simplifying many problems that are NP-hard under other operations. Although the SCJ theory is well constructed, there were doubts about its biological relevance. In this work we contribute to the understanding of this operation, performing an extensive study that applies the SCJ under different evolutionary conditions, with real and simulated data, analyzing two aspects of evolutionary reconstruction: the tree structure and the genome (gene order) of the ancestral species. In the first analysis, we found out that SCJ can recover between 60% to 80% of the tree structure. Regarding the second question, given a tree structure, the reconstruction of ancestral genomes varies according to the distance from ancestral species to the known species. In the case of ancestral species close to the leaves, about 85% of the gene order can be recovered while, in more distant species, about 50% of gene order are recovered, using genome sets of 64 species. As far as time is concerned, the methods we implemented can find a topology for 64 genomes with 2000 genes each in about 10.7 minutes, and reconstruct the ancestral genomes in about 0.05 minutes, both on a typical desktop computer.

Agradecimentos

Gostaria de agradecer ao meu orientador, João Meidanis, pela oportunidade de poder trabalhar com ele e pelo tempo dedicado. Agradeço-o também pelas interessantes conversas e pelo modo diferente como ensina, que me motivam a aprender cada vez mais. Agradeço ao Pedro Feijão por generosamente ter feito parte deste trabalho, enriquecendo as reuniões com seu conhecimento sobre a área e sobre outras coisas legais.

Agradeço ao pessoal da secretaria, por cuidarem da parte burocrática e, mesmo assim, serem tão gentis. Nunca tinha visto algo parecido. Agradeço também as agências financiadoras: CNPq, CAPES e FAPESP, por financiarem este trabalho e ao de tantos outros, incentivando a pesquisa. Agradeço ao CENAPAD e ao LOCo por tornarem possível a execução dos experimentos e também a Scylla, por permitir que as minhas reuniões ocorressem.

Agradeço aos amigos do LOCo por terem me acolhido no laboratório, pelas discussões sobre temas aleatórios e pelas horas durante e, principalmente, após o almoço. Saio desta etapa uma pessoa mais curiosa e, talvez, mais excêntrica, por tantas pessoas profundamente curiosas e interessantes que conheci.

Agradeço ao Pedro Hokama por tudo que passamos e por cuidar de mim em Campinas. Agradeço ao meu irmão Guinho e aos meus pais, Rodolfo e Sandra, pelo enorme apoio em todas as etapas e por tudo que me ensinaram. Amo vocês!

Finalmente, agradeço a Deus, não só aqui, mas todos os dias da minha vida, por tudo que sou e por todas estas pessoas maravilhosas que pude conhecer.

Sumário

Resumo **iii**

Abstract **iv**

Agradecimentos **v**

1 Introdução **1**
1.1 Organização do Texto 3

2 Evolução em genomas **5**
2.1 DNA 5
2.2 Genes, cromossomos e genomas 7
2.3 Eventos evolutivos 9
2.3.1 Mutações pontuais 9
2.3.2 Rearranjos de Genes 9
2.4 Distância evolutiva 12
2.5 Árvore filogenética 13
2.6 Conclusões 14

3 Matemática da evolução **15**
3.1 Hipóteses sobre a evolução 15
3.1.1 Matriz Métrica 16
3.1.2 Parcimônia 18
3.1.3 Máxima Verossimilhança 19
3.2 Representações 20
3.2.1 Genoma 21
3.2.2 Árvore Filogenética 22
3.2.3 Operação de Rearranjo 22
3.3 Conclusões 23

4 Problemas de Rearranjo de Genomas **24**
4.1 Problema da Distância . . . 24
4.2 Problema da Mediana . . . 25
4.3 Problema de Rearranjo de Múltiplos Genomas . . . 25
4.3.1 Problema Pequeno da Parcimônia (PPP) . . . 26
4.3.2 Problema Grande da Parcimônia (PGP) . . . 26
4.4 Conclusões . . . 26

5 Operações de rearranjo **27**
5.1 Estado da Arte . . . 27
5.1.1 *Breakpoints* . . . 27
5.1.2 Hannenhalli-Pevzner (HP) . . . 28
5.1.3 *Double-Cut-and-Join* (DCJ) . . . 29
5.1.4 *Single-Cut-or-Join* (SCJ) . . . 30
5.1.5 Outras Operações . . . 31
5.2 A operação de rearranjo *Single-Cut-or-Join* (SCJ) . . . 33
5.2.1 Operações . . . 33
5.2.2 Eventos biológicos . . . 34
5.2.3 Problemas de Rearranjo sob o modelo SCJ . . . 35
5.3 Conclusões . . . 39

6 O Experimento **40**
6.1 Objetivo . . . 42
6.2 Dados utilizados . . . 42
6.2.1 Simulações . . . 42
6.2.2 Dados reais . . . 47
6.3 Métodos . . . 51
6.3.1 Problema Pequeno da Parcimônia . . . 51
6.3.2 Problema Grande da Parcimônia . . . 54
6.4 Métricas . . . 56
6.4.1 Acurácia da topologia . . . 56
6.4.2 Acurácia dos ancestrais . . . 57
6.4.3 Eficiência dos métodos . . . 59
6.5 Conclusões . . . 59

7 Resultados **60**
7.1 Acurácia da Topologia . . . 60
7.1.1 Dados simulados . . . 60
7.1.2 Dados reais: Campanulaceae . . . 64

7.1.3 Dados reais: Protostômios 68
7.2 Acurácia dos Ancestrais 69
7.2.1 Adjacências em comum 70
7.2.2 Adjacências Falso-Positivas 72
7.3 Eficiência dos métodos 74

8 Conclusões **75**
8.1 Trabalhos Futuros 76

A SCJ suite – Manual do Usuário **79**
A.1 Estrutura 79
A.2 Requisitos do Sistema 80
A.3 Arquivos de Entrada 80
A.3.1 Genomas – Formato de Permutações 80
A.3.2 Árvore – Formato Newick 82
A.4 Módulo *CompareTrees* 83
A.4.1 Fluxo do Sistema 83
A.4.2 Parâmetros 84
A.4.3 Cenários de Uso: Entradas e Saídas 87

Bibliografia **94**

Lista de Tabelas

2.1 Características dos eventos de rearranjo. 12

3.1 Princípios utilizados pelos métodos de inferência filogenética. 16

4.1 Algumas variantes do Problema de Rearranjo em Genomas. 24

5.1 Características das operações de rearranjo propostas. 32
5.2 Comparação entre as operações de rearranjo HP, DCJ e SCJ. 35

6.1 Características avaliadas. 40
6.2 Métodos implementados. 41
6.3 Valores dos parâmetros da simulação (outros estudos). 45
6.4 Valores dos parâmetros da simulação (nosso estudo). 46
6.5 Onde encontrar a análise do impacto dos parâmetros na acurácia dos métodos. 47
6.6 Medidas, métricas e indicadores usados para avaliar as características. . . . 57
6.7 Indicadores da acurácia dos ancestrais. 59

A.1 Descrição dos parâmetros do módulo *CompareTrees*. 86
A.2 Cenários de uso do módulo *CompareTrees*. 87

Lista de Figuras

2.1 Molécula de DNA. 6
2.2 Adjacências dos genes. 7
2.3 Genoma. 8
2.4 Efeitos das mutações pontuais no genoma. 10
2.5 Rearranjos de genes. 11
2.6 Árvore filogenética. 13

3.1 Método baseado em distância. 17
3.2 Método baseado na parcimônia. 19
3.3 Método baseado na máxima verossimilhança. 20
3.4 Representação do Genoma. 22

4.1 Problemas da Distância e da Ordenação. 25
4.2 Problema da Mediana. 25

5.1 Efeito da operação *cut* no genoma. 33
5.2 Efeito da operação *join* no genoma. 34
5.3 Problema da Mediana por SCJ. 36
5.4 Redução do Problema da Árvore de Steiner para o Problema Grande da Parcimônia por SCJ. 39

6.1 Diagrama dos experimentos. 42
6.2 Parâmetros da simulação. 44
6.3 Conjunto de dados da família Campanulaceae. 48
6.4 Conjunto de dados do grupo Protostômio. 50
6.5 Árvore de pesquisa do *branch-and-bound*. 55
6.6 Métricas para avaliar ancestrais. 57

7.1 Influência dos parâmetros na topologia inferida. 61
7.2 Campanulaceae — Topologias ótimas do SCJ. 65
7.3 Comparação das topologias de Campanulaceae. 67

7.4 Comparação das topologias dos Protostômios. 69
7.5 Influência dos parâmetros na reconstrução de genomas ancestrais (adjacências em comum). 71
7.6 Influência dos parâmetros na reconstrução de genomas ancestrais (adjacências falso-positivas). 73

8.1 Percentagem de reconstrução relacionada à altura do nó. 76

A.1 Formato Newick. 83
A.2 Módulo *CompareTrees* — Etapas. 84
A.3 Módulo *CompareTrees* — Parâmetros. 85

Capítulo 1

Introdução

Todos os organismos vivos estão sujeitos a mudanças de suas características, que podem ou não ser passadas para seus descendentes. A *evolução* está relacionada às modificações que são transmitidas para a próxima geração e que, ao longo do tempo, podem ser incorporadas à espécie. Existem muitos questionamentos interessantes em torno deste tema: ainda não compreendemos totalmente como e porque estas mudanças ocorrem e como isso pode nos relacionar a outras espécies.

Ao estudar a evolução de uma espécie, tentamos inferir as várias espécies de que ela descende, estabelecendo hipóteses sobre o seu relacionamento com outras espécies vivas, as quais chamamos de *filogenia*. A filogenia é útil não somente aos pesquisadores de evolução, mas também em outras importantes áreas, como a da saúde humana, pois permite compreender como complexos mecanismos morfológicos e químicos foram incorporados ao longo do tempo e quais papéis desempenham em nosso organismo.

Um exemplo disso é como os métodos filogenéticos têm sido críticos no estudo de patógenos, como no estudo do vírus HIV (do inglês, *Human Immunodeficiency Virus*), que dribla o sistema imunológico dos seres humanos diversificando-se rapidamente devido a sua alta taxa evolutiva. Outro exemplo é a aplicação da filogenia no desenvolvimento de vacinas, identificando a evolução que ocorreu no passado e o potencial do patógeno alvo evoluir no futuro.

Anteriormente, a filogenia era inferida através de dados geográficos, comportamentais, morfológicos ou metabólicos. Mas, desde o avanço e barateamento das tecnologias de sequenciamento do genoma, que ganhou destaque por volta do ano 2000, temos à disposição, em quantidade abundante, outros tipos de dados, os dados moleculares. Agora, é possível estudar a evolução de uma forma mais precisa, visto que esta ocorre nas moléculas, e com um grande volume de dados acessíveis.

Mesmo ao trabalhar somente com dados moleculares, vemos que a evolução ocorre em diferentes níveis: desde a troca de uma simples base em uma sequência imensa de bases

do DNA, até a troca de grandes segmentos, como os genes, causando alterações mais profundas. As mudanças de uma única base, chamadas de mutações pontuais, são mais frequentes, inclusive entre indivíduos da mesma espécie. Já as mudanças que alteram a ordem dos genes, rearranjando-os de uma nova forma, por trazerem mudanças mais radicais, são mais raras e podem ser utilizadas para estudar relacionamentos evolutivos mais antigos como, por exemplo, entre organismos de espécies diferentes.

Independentemente de trabalharmos com mutações pontuais ou rearranjos, estes eventos que modificam a composição genética de uma população de organismos ao longo do tempo recebem o nome de *eventos evolutivos*, e denominamos de *distância evolutiva* o número de eventos evolutivos que transformam um genoma em outro. Os rearranjos são eventos evolutivos que podem alterar de diferentes formas a ordem dos genes, invertendo um bloco de genes, trocando dois blocos de genes no mesmo cromossomo ou em cromossomos diferentes, criando uma ou mais cópias de um gene, entre tantos outros. Atribuímos a cada tipo de rearranjo uma frequência em relação aos outros, mas não há um consenso sobre estas frequências, pois provavelmente variam com a espécie e outros fatores.

Somada à grande variedade de rearranjos, ainda não há um modelo bem definido para a evolução dos genes. E, nessa pesquisa em desenvolvimento, a matemática e a computação tem formado uma base essencial para os estudos evolucionários, não somente com ferramentas que automatizam o trabalho de análise, mas também indicando limitações e dando orientações aos modelos propostos. Da mesma forma, problemas inspirados na pesquisa em evolução e genética têm trazido avanços na área de combinatória, geometria e teoria de probabilidade.

Um destes problemas, de explicar a história evolutiva de um conjunto de espécies através de rearranjos, pode ser visto como um problema de otimização computacional, chamado de Problema de Rearranjo de Múltiplos Genomas (em inglês, *Multiple Rearrangement Problem*). Este problema tem por objetivo explicar, com o menor número de rearranjos, a evolução do conjunto de genomas recebido na entrada, gerando uma filogenia representada através de uma árvore, conforme será explicado no Capítulo 4. Um fator impactante, tanto na qualidade dos resultados obtidos como na complexidade do problema, é a operação de rearranjo que será utilizada.

Atualmente existem diversas propostas para computar a distância de rearranjo. Devido à complexidade em incluir a grande variedade de eventos evolutivos existentes, inicialmente as operações de rearranjo incluíam apenas um tipo de evento e, gradualmente, foram sendo incluídos outros tipos de eventos, tornando as operações de rearranjo cada vez mais realísticas. Apesar dos esforços, ainda hoje alguns eventos evolutivos importantes não foram incluídos.

Em 2009, Feijão e Meidanis [30] propuseram uma operação de rearranjo chamada de *Single-Cut-or-Join* (SCJ), que serve como uma espécie de operação básica, cuja com-

posição permite obter todos os eventos de rearranjo clássicos e traz como grande vantagem a simplificação de muitos problemas, que sob outras operações são NP-difíceis e sob a perspectiva do SCJ tornam-se polinomiais. Entretanto, o SCJ não utiliza a frequência relativa entre os eventos de rearranjo que é comumente adotada, e sua capacidade prática de reconstruir histórias evolutivas ainda não é totalmente compreendida.

Neste trabalho contribuímos com o entendimento desta operação, realizando um extenso estudo que aplica o SCJ sob diferentes condições evolutivas, com dados reais e simulados. Representando a filogenia através de uma árvore, analisamos dois aspectos da reconstrução evolucionária: (1) quão bem o SCJ reconstrói a estrutura da árvore?; e (2) quão bem o SCJ reconstrói o genoma (ordem dos genes) das espécies ancestrais? Para a primeira pergunta, descobrimos que o SCJ é capaz de inferir corretamente de 60% a 80% da estrutura da árvore. Em relação à segunda questão, dada à estrutura da árvore, a reconstrução dos genomas ancestrais varia conforme a distância da espécie ancestral para as espécies conhecidas. No caso de espécies ancestrais mais próximas, cerca de 85% da ordem dos genes foi coberta enquanto, em espécies mais distantes, aproximadamente 50% da ordem dos genes foi coberta, usando conjuntos de genomas de 64 espécies. Em relação ao tempo, os métodos, que implementamos em Java, podem encontrar a topologia de 64 genomas com 2000 genes cada em cerca de 10,7 minutos e reconstruir seus genomas ancestrais em 0,05 minutos, ambos em um computador desktop padrão.

A princípio, o SCJ foi proposto com o objetivo de auxiliar o cálculo de outros modelos de rearranjo considerados, na teoria, mais realísticos. Entretanto, como principais contribuições, conseguimos detectar a acurácia deste modelo em variados cenários que, somadas a sua eficiência, indicam o alto potencial de reconstrução filogenética da operação SCJ.

1.1 Organização do Texto

Esta dissertação de Mestrado está organizada como segue.

O **Capítulo 2** dá uma visão geral de como ocorre a evolução nos genomas sob o ponto de vista biológico, introduzindo os conceitos utilizados ao longo desta dissertação.

O **Capítulo 3** discorre sobre os princípios que os métodos de inferência filogenética se baseiam, além de abordar as definições que permitem o estudo matemático e computacional do problema.

Após a contextualização, o **Capítulo 4** define de modo mais formal os problemas estudados.

O **Capítulo 5** faz uma revisão bibliográfica com as principais operações de rearranjo propostas, e como a complexidade dos problemas é modificada ao usar diferentes modelos. O capítulo também detalha a operação de rearranjo *Single-Cut-or-Join*, o foco do nosso estudo, especificando os eventos permitidos e como os problemas são vistos sob este

modelo.

O **Capítulo 6** detalha o experimento, apresentando os tipos de dados utilizados como entrada, os métodos usados na resolução dos problemas e as métricas para avaliação dos resultados.

No **Capítulo 7** expomos os resultados dos experimentos e nossa análise.

O **Capítulo 8** resume as conclusões que obtivemos a partir dos resultados apresentados e como o trabalho pode ser estendido futuramente.

Ao final estão as referências bibliográficas e também um apêndice com maiores detalhes do projeto implementado.

Capítulo 2

Evolução em genomas

Como os problemas estudados são derivados de uma perspectiva biológica, neste capítulo trataremos de conceitos vindos de diversas áreas da Biologia, tais como Biologia Molecular, Evolução e Genética.

Nas Seções 2.1 e 2.2 definimos a composição de um genoma, que é a informação recebida como entrada dos problemas. A Seção 2.3 mostra os mecanismos pelos quais evoluem os genomas.

A seguir, apresentamos na Seção 2.4 e na Seção 2.5 formas usualmente utilizadas para representar a evolução de um conjunto de genomas.

2.1 DNA

Moléculas enormes de DNA são responsáveis por codificar toda a informação hereditária e passá-la de geração para geração (exceto em alguns vírus, como os retrovírus, em que a informação é passada através do RNA). São compostas de unidades menores, os nucleotídeos, que podem ser de quatro tipos, diferenciando-se somente na base nitrogenada que os constitui: as purinas adenina (A) e guanina (G), e as pirimidinas citosina (C) e timina (T).

Estruturalmente, uma molécula de DNA consiste de duas grandes cadeias complementares de nucleotídeos, formando uma dupla hélice. As duas cadeias são mantidas juntas por pontes de hidrogênio, entre pares de bases específicos: adenina (A) pareia com timina (T), formando duas pontes de hidrogênio, enquanto guanina (G) pareia com citosina (C), formando três pontes de hidrogênio. Note que todo par de bases consiste de uma purina (A ou G) e uma pirimidina (T ou C). Este padrão é conhecido como *pareamento de bases complementares*.

Cada fita possui uma direção de leitura, ou seja, as enzimas que participam do processo de replicação do DNA e da transcrição para o RNA podem vincular-se e percorrer a fita

em apenas uma direção. Curiosamente, as duas fitas do DNA possuem direções de leitura opostas, e por isso dizemos que são *antiparalelas*. Esta orientação antiparalela é necessária para que as duas fitas possam se encaixar no espaço tridimensional.

A Figura 2.1 apresenta uma molécula de DNA, com suas fitas complementares e antiparalelas dispostas em formato de dupla hélice.

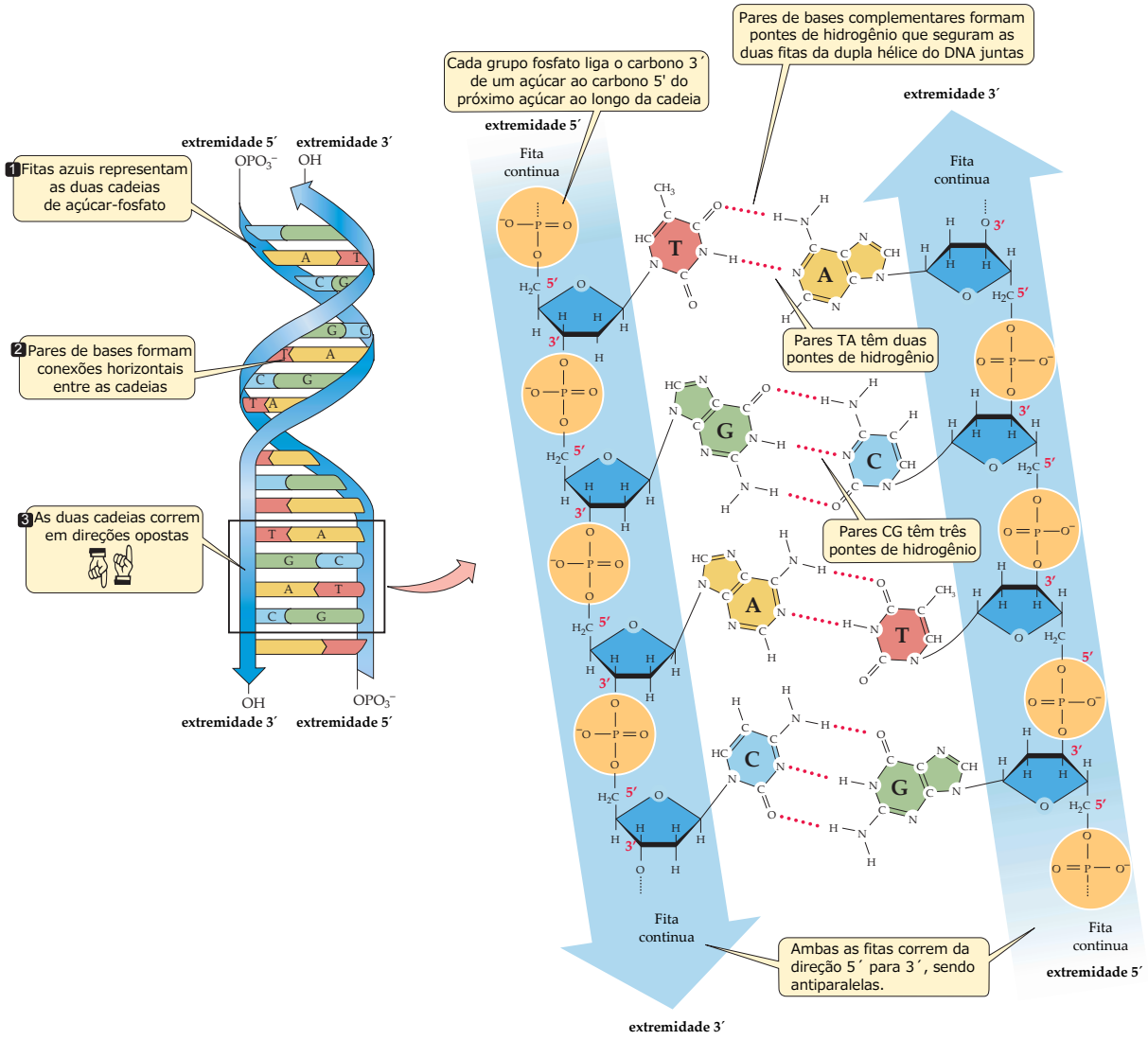

Figura 2.1: Molécula de DNA. Fonte: Purves et al. [60, pg. 218–219].

2.2 Genes, cromossomos e genomas

Um *gene* é uma sequência orientada do DNA que contém todas as informações para produzir uma determinada proteína ou o RNA, que controlam, ou ajudam a controlar, uma característica. Na biologia, as extremidades de um gene são comumente chamadas de 5' e 3', que determinam, respectivamente, o começo e o fim do gene.

Dentro de cada gene, somente uma das duas fitas de DNA, chamada de fita molde, é transcrita. A outra fita, nomeada de fita complementar, não participa do processo de transcrição. Mesmo genes que pertencem à mesma molécula de DNA podem utilizar fitas diferentes durante a transcrição. Dessa forma, a fita que é complementar de um gene pode ser a fita molde de outro. Considerando a orientação de dois genes consecutivos, suas extremidades podem estar conectadas de quatro formas possíveis, exemplificadas na Figura 2.2 através dos genes B e C. Nesta figura, o sinal "$-$" indica a orientação relativa do gene em uma das fitas de DNA.

As extremidades terminais do cromossomo são chamadas de *telômeros*, como no caso da extremidade final do gene D na Figura 2.2.

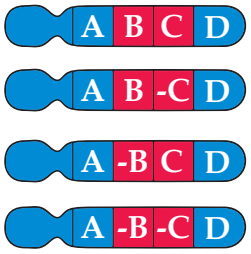

Figura 2.2: Adjacências.[1]

Um *cromossomo* é uma molécula de DNA, que pode ser linear ou circular, associada com proteínas. Estas proteínas tem a função de dobrar a longa fita de DNA, embalando-a, de modo que fique mais compacta.

Uma possível forma de representar o cromossomo é listando seus genes. A lista de genes pode ser obtida tomando um telômero e, a partir deste, ir percorrendo os genes ao longo do componente. No caso de cromossomos lineares, o procedimento é interrompido ao encontrar outro telômero. No caso de cromossomos circulares, inicialmente é tomado um gene arbitrário e o procedimento é interrompido quando o gene inicial aparece pela segunda vez na lista.

Muitas bactérias e outros procariotos carregam seus genes em uma única molécula de DNA, que é frequentemente circular. Por outro lado, os genes em plantas, animais, fungos e outros eucariotos, são particionados em vários cromossomos, que costumam ser lineares.

Um *genoma* é o conjunto de todos os cromossomos de um organismo. Um genoma é dito *unicromossomal* quando contém apenas um cromossomo, caso contrário é chamado de *multicromossomal*.

Como foi observado anteriormente, em alguns vírus o genoma é constituído de moléculas de RNA. Outro detalhe é que, em organismos eucariotos, além do genoma

[1] As figuras que representam genomas, neste trabalho, são baseadas na ilustração da pg. 253 do livro de Purves et al. [60].

nuclear, existem algumas organelas, como mitocôndrias e, em plantas, cloroplastos, que possuem seu próprio genoma, com a função de codificar as proteínas que utilizam.

O genoma mitocondrial, apesar de ter a mesma composição química em relação ao DNA nuclear, possui características peculiares, sendo constituído de apenas uma molécula de DNA circular. Geralmente é haploide, sendo herdado exclusivamente da mãe. O conteúdo gênico é bastante conservado, e a ordem em que esses genes se encontram organizados no genoma também costuma ser conservada, sendo bastante utilizado em estudos de rearranjo.

Da mesma forma que o genoma mitocondrial, o genoma do cloroplasto também é composto por uma molécula de DNA circular e a ocorrência de rearranjos durante a evolução também é rara, sendo usado para demarcar grupos maiores. Em nosso estudo utilizamos conjuntos de genomas reais, tanto mitocondriais como de cloroplastos, apresentados na Seção 6.2.2.

Existem diversas formas de representar o genoma para fins computacionais, e na Seção 3.2.1 detalhamos a representação adotada em nosso trabalho. A Figura 2.3 exemplifica o genoma de um organismo eucarioto, abrangendo os conceitos aqui apresentados.

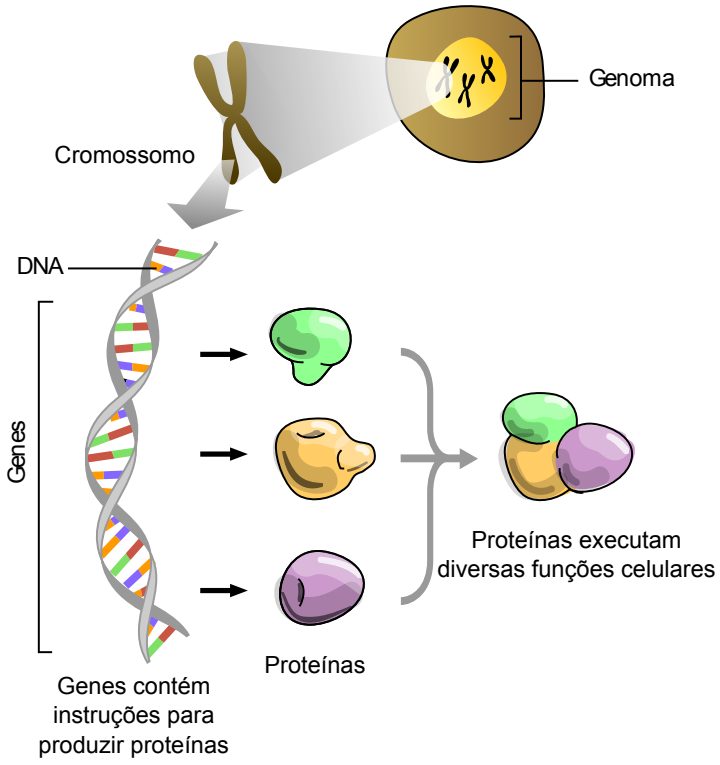

Figura 2.3: Genoma. Fonte: site "The Science Creative Quarterly" [61], arte de Jiang Long.

2.3 Eventos evolutivos

O material genético é suscetível a mutações, que causam mudanças permanentes na informação que ele codifica. Apesar dos mecanismos que mantém o DNA serem notavelmente precisos, podem ocorrer erros durante a replicação, recombinação ou reparo do DNA. Estes erros causam desde simples mudanças na sequência linear de pares de bases, como a substituição de um par de bases por outro, até rearranjos em larga escala, tais como reversões, deleções, duplicações e translocações de DNA de um cromossomo para outro.

Além destas falhas, existem vários elementos móveis no DNA, chamados de *transposons*, que são capazes de ser inseridos em novos locais, no mesmo cromossomo ou em cromossomos diferentes, espalhando-se no genoma e alterando sua estrutura.

As mutações podem ocorrer em qualquer tipo de célula mas, quando ocorrem em células germinativas, podem ser herdadas pelos descendentes do indivíduo. No nível molecular podemos dividir as mutações em duas categorias, mutações pontuais e mutações cromossomais (rearranjos de genes), que serão explicadas adiante.

2.3.1 Mutações pontuais

Mutações pontuais são mudanças em um único nucleotídeo, resultantes da inclusão ou remoção de um nucleotídeo, ou mesmo a substituição de uma base por outra no DNA. Podem ser causadas por erros durante a replicação do cromossomo ou por fatores ambientais, como agentes químicos e radiação.

Se tomarmos os genomas de quaisquer dois humanos, vemos que estes diferem um do outro devido às substituições de nucleotídeos (polimorfismos de um único nucleotídeo, ou SNPs) e também devido à herança de perdas e ganhos de DNA. Compreender estas diferenças traz melhorias à medicina e à nossa compreensão da biologia humana.

A Figura 2.4 ilustra os efeitos das mutações pontuais no genoma.

2.3.2 Rearranjos de Genes

No processo de quebra e união das moléculas de DNA podem ocorrer interrupções grosseiras na sequência de nucleotídeos, alterando a quantidade ou estrutura dos cromossomos. Essas alterações podem ocorrer de diversas formas, tais como deleções, duplicações, inversões, transposições e translocações, trazendo profundas alterações no genoma, muitas vezes resultando em deficiências graves ou até mesmo letais.

Abaixo descrevemos os eventos que aparecem frequentemente em problemas de rearranjo e, na Figura 2.5, suas respectivas ilustrações.

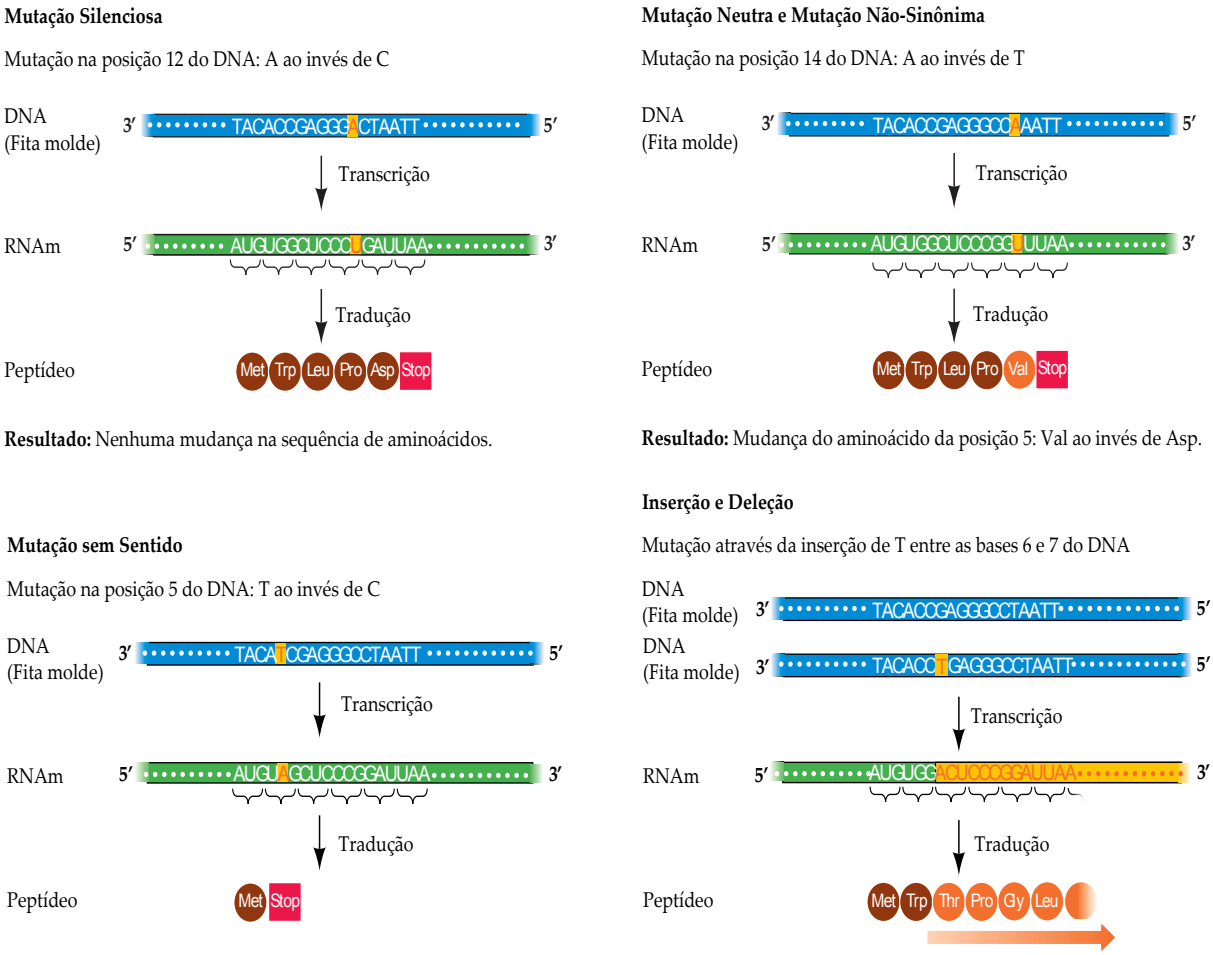

Figura 2.4: Efeitos das mutações pontuais no genoma. Fonte: Purves et al. [60, pg. 251–252].

A *inversão* é uma das operações de rearranjo mais observadas nos genomas, onde os segmentos das fitas de DNA são reinseridos no mesmo local, mas em ordem linear invertida, sendo lidos no sentido oposto.

A *transposição* ocorre quando um segmento do DNA passa a ocupar uma nova posição na mesma molécula, e geralmente é causada por transposons. Ao ser tratada computacionalmente, o movimento da transposição pode ser visto de outro modo, como a troca de dois segmentos de DNA adjacentes.

A *translocação*, junto com a inversão, é um dos eventos de rearranjo mais comuns na evolução dos mamíferos. Nele há troca de material genético entre cromossomos diferentes, ocorrendo quando uma molécula de DNA se quebra e um pedaço de seu segmento se transfere para outra molécula. Frequentemente a translocação é recíproca, com os dois

cromossomos trocando segmentos entre si. Quando a translocação é modelada, ela é tratada como a troca das pontas de dois cromossomos diferentes, ou seja, para a troca de segmentos internos é necessário que ocorra mais que um evento evolutivo. Algumas doenças humanas são causadas por translocações, como a leucemia mielóide crônica [29].

As operações de fissão e fusão alteram o número de cromossomos, aumentando-o ou diminuindo-o, sem que haja variação na quantidade de DNA. A *fusão* ocorre quando dois cromossomos fundem-se em um único cromossomo, enquanto a *fissão* quebra um cromossomo em dois (o inverso da fusão).

A *deleção* remove um segmento do DNA, e geralmente traz severas consequências devido à perda dos genes do segmento. Pode ocorrer durante a quebra da molécula de DNA em dois pontos seguida da junção destes pontos, deixando de fora o segmento entre eles. Existem deleções nos seres humanos bem conhecidas por causarem doenças.

A *inserção* é o processo onde um segmento de DNA é incluído ao genoma, podendo interromper a estrutura e função de um gene.

As *duplicações* podem ser produzidas ao mesmo tempo que as deleções, no momento em que dois cromossomos homólogos se quebram em locais diferentes e se reconectam de forma incorreta. Nesse caso, em uma das duas moléculas envolvidas nesse mecanismo faltaria um segmento do DNA (deleção), enquanto em outro haveria duas cópias (uma duplicação) do segmento que foi excluído do primeiro. Note que, apesar do termo *duplicação* ser utilizado, este evento abrange a criação de duas ou mais cópias de um segmento do genoma, que pode ser desde

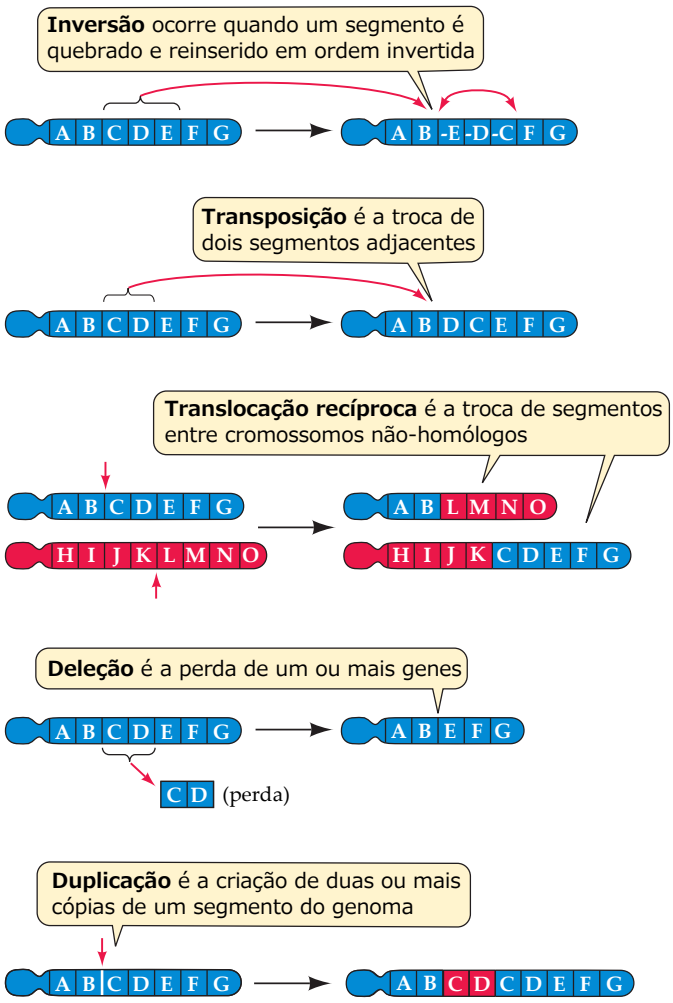

Figura 2.5: Rearranjos. As letras A, B, ..., G representam trechos do genoma como, por exemplo, genes. Fonte: Purves et al. [60, pg. 253].

um gene até um cromossomo inteiro.

A Tabela 2.1 sumariza as principais características de cada evento.

Tabela 2.1: Características dos eventos de rearranjo.

Eventos	Modificações no Genoma			Número de cromossomos afetados
	Ordem dos genes	Sentido da leitura	Qtde. de genes	
Inversão	Sim	Sim	Não	1
Transposição	Sim	Não	Não	1
Translocação	Sim	Depende	Não	2
Fissão	Sim	Não	Não	2
Fusão	Sim	Depende	Não	2
Deleção	Sim	Não	Sim	1
Inserção	Sim	Não	Sim	1
Duplicação	Sim	Não	Sim	1

2.4 Distância evolutiva

As distâncias evolutivas são métricas da genética molecular que medem as diferenças genéticas entre um par de espécies, sendo definidas como o número de eventos evolutivos necessários para transformar um genoma em outro.

Estimar a distância evolutiva entre duas espécies é útil durante a inferência da evolução de um conjunto de espécies, pois dão uma ideia de quais espécies estão mais próximas. Muitos estimadores têm sido propostos utilizando diferentes modelos de evolução, considerando desde um único tipo de rearranjo, uma combinação de dois ou mais tipos, ou mutações pontuais, por exemplo. No Capítulo 5 falaremos mais a respeito de alguns desses estimadores.

Como as distâncias evolutivas são estimadas a partir das espécies atuais, o número de eventos evolutivos obtido costuma ser inferior ao número real de eventos, pois durante a evolução podem ocorrer eventos que são "escondidos" devido à evolução paralela ou convergente das espécies.

2.5 Árvore filogenética

Denominamos filogenia uma hipótese proposta por um sistemata para descrever a história evolutiva de um grupo de organismos a partir de um ancestral comum. A árvore filogenética é uma forma de representar essa história, mostrando os relacionamentos evolucionários entre as várias espécies que acredita-se ter um ancestral comum. É uma representação bastante utilizada, pois o processo de geração de novas espécies a partir de variação genética costuma ser visto como um processo ramificado, sendo causado, por exemplo, por especiação.

A árvore filogenética é composta por nós e conexões entre pares de nós. Cada nó representa uma unidade taxonômica, que pode ser desde uma espécie até algo mais abrangente, como um reino inteiro. Os nós internos da árvore filogenética representam os ancestrais, que na maior parte das vezes são hipotéticos, e os nós extremos (folhas) representam, em geral, espécies conhecidas.

A conexão entre um par de nós (aresta) representa o relacionamento evolucionário entre os dois táxons associados a estes nós. Dependendo da forma como a árvore é inferida, suas arestas podem ter um valor numérico associado, representando uma estimativa da distância evolutiva que separa os dois táxons conectados.

A Figura 2.6 ilustra um exemplo de árvore filogenética, construída a partir de um conjunto de características observadas atualmente. Nas arestas onde ocorrem um ou mais eventos evolutivos existe uma marcação indicando que uma nova característica foi incorporada e todos os descendentes abaixo dela possuem a característica.

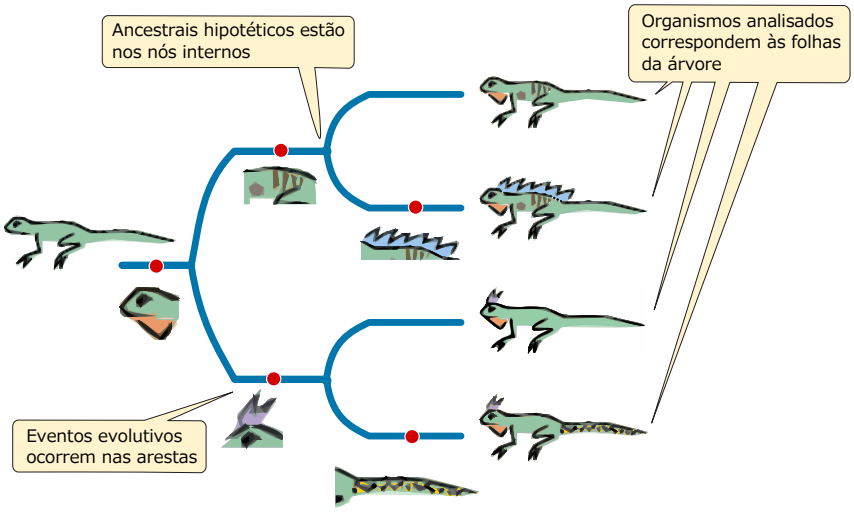

Figura 2.6: Árvore filogenética representando a possível evolução de quatro organismos.[2]

[2]As ilustrações dos lagartos e de suas características, usadas neste trabalho, foram retiradas do software PhyloStrat [42].

2.6 Conclusões

Este capítulo contextualizou os problemas sob o ponto de vista biológico, definindo o que são genomas, de que formas evoluem e como podemos representar a evolução. Os Capítulos 3 a 5 mostram como esses conceitos são representados para fins computacionais.

A evolução dos genomas foi vista de um modo geral e já podemos notar que trata-se de um processo bastante complexo. Para que os problemas possam ser definidos, são necessários alguns princípios que simplifiquem o modo como ela é vista. O Capítulo 3 complementa este capítulo na tarefa de contextualização dos problemas, apresentando os princípios que são utilizados.

Capítulo 3

Matemática da evolução

Embora os conceitos de genomas, rearranjos e evolução vistos no Capítulo 2 sejam inerentemente biológicos, eles requerem definições matemáticas precisas para os propósitos computacionais. Existem muitas formas de representá-los, e a Seção 3.2 contém as representações usadas neste trabalho.

Tão importante quanto as definições, certas hipóteses sobre como ocorre a evolução são necessárias para reduzir a complexidade deste processo, tornando-o tratável. As principais hipóteses adotadas, que constituem os princípios nos quais os métodos de inferência filogenética se baseiam, são detalhadas na Seção 3.1.

3.1 Hipóteses sobre a evolução

Como a evolução é um processo muito complexo e multifacetado, a inferência filogenética é sempre baseada em certos princípios evolutivos que permitem a simplificação do problema. Podemos classificar um método de reconstrução filogenética de acordo com os princípios em que ele se baseia e, nesta seção, apresentamos três princípios comumente usados. A Tabela 3.1 resume as principais características de cada um deles. Se, para um determinado conjunto de espécies, os princípios evolutivos nos quais o método se baseia falham, a estimativa da árvore filogenética será inconsistente, convergindo para a árvore incorreta.

Tabela 3.1: Princípios utilizados pelos métodos de inferência filogenética. Entre parênteses estão os métodos utilizados em nosso estudo.

Princípio	Aspectos Inferidos		Métodos
	Ancestrais	Topologia	
Matriz métrica	Não	Sim	(NJ) [63], UPGMA [51]
Parcimônia	Sim	Sim	(Fitch) [36], Sankoff [64]
Máxima Verossimilhança	Sim	Sim	Felsenstein [33]

3.1.1 Matriz Métrica

Os métodos baseados neste princípio, também conhecidos como métodos baseados em distância, assumem que, dado um conjunto de organismos L, ao construirmos uma matriz a partir da distância evolutiva entre cada par de organismos do conjunto, esta matriz possuirá uma *estrutura métrica* (ou próxima disso). Uma função $d : S \times S \rightarrow \mathbb{R}^+$ é *métrica* se, para quaisquer elementos x e y pertencentes ao conjunto S, satisfazer as seguintes condições:

1. $d(x, y) \geq 0$, para todo $x, y \in S$, com igualdade se, e somente se, $x = y$ (positividade);
2. $d(x, y) = d(y, x)$, para todo $x, y \in S$ (simetria);
3. $d(x, z) \leq d(x, y) + d(y, z)$, para todo $x, y, z \in S$ (desigualdade triangular);

No caso de árvores de genomas, note que:

1. O conjunto L dos organismos conhecidos, dado como entrada no problema, corresponde às folhas da árvore T.
2. Como explicado na Seção 2.5, nas arestas da árvore T ocorrem os eventos evolutivos e cada aresta da árvore T possui um valor numérico associado, correspondente à distância evolutiva.
3. Dados dois nós quaisquer da árvore T, existe somente um caminho possível entre eles.

Definindo $d_T : L \times L \rightarrow \mathbb{R}^+$ como uma função que associa a cada par de genomas (x, y) pertencente à L o valor correspondente à soma das distâncias evolutivas do único caminho que existe entre x e y na árvore T, podemos provar que d_T é métrica.

Baseado no fato de que a árvore filogenética de um conjunto de genomas tem estrutura métrica, escolhemos um modelo estocástico de evolução para estimar $d_T(x, y)$ a partir das diferenças observadas atualmente entre os organismos x e y. Os resultados irão variar conforme o modelo de evolução escolhido, que assume certas premissas que podem não corresponder aos genomas reais.

Independente do modelo escolhido, o princípio básico da abordagem baseada em distância é computar a estimativa da distância $d_T(x, y)$ entre todo par de genomas do conjunto dado, construindo uma matriz de distância. Como d_T é métrica, é esperado que a matriz de distância construída a partir do estimador também seja. Quando a matriz de distância possui *estrutura métrica*, chamamos a matriz de *aditiva*.

A partir desta matriz métrica, o método encontra uma árvore T onde seja possível atribuir valores às arestas de modo que as distâncias entre os organismos nas folhas corresponda à matriz de distância fornecida.

Apesar da árvore filogenética real possuir estrutura métrica, na prática, a suposição de que a matriz de distâncias do conjunto de organismos também será métrica nem sempre se aplica. Como em grande parte das vezes o conjunto analisado é composto somente por espécies existentes atualmente, podem existir os eventos escondidos mencionados na Seção 2.4 que fazem com que a distância estimada não necessariamente corresponda à distância real. Nestes casos existem métodos que tentam encontrar a árvore mais próxima da matriz de distância, e o critério do que é mais próximo varia entre os métodos. A Figura 3.1 exemplifica a abordagem descrita.

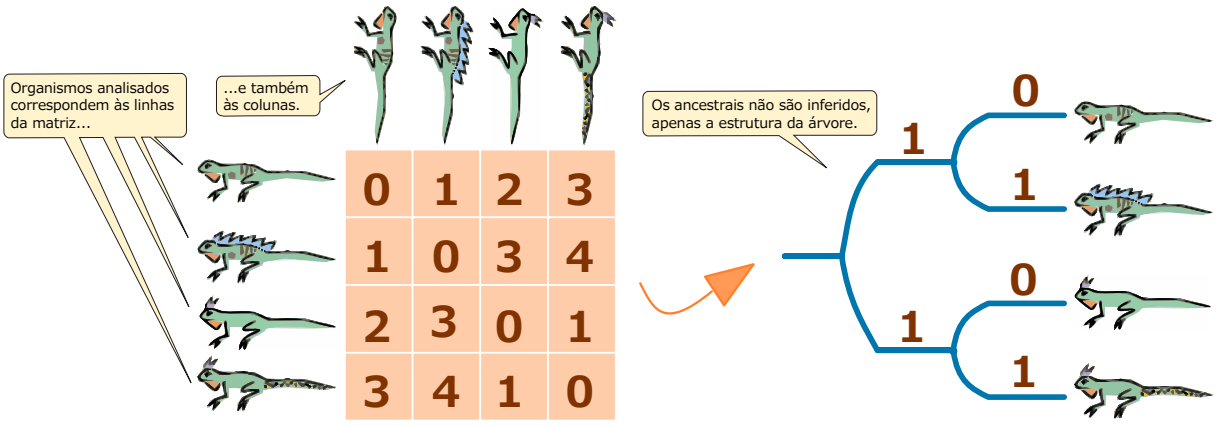

Figura 3.1: Distância: Árvore filogenética é construída a partir da matriz de distâncias. Método encontra a árvore que melhor representa essa matriz.

Note que, por serem baseados em uma matriz de distância, estes métodos não são capazes de computar os genomas dos nós ancestrais, fornecendo como resultado apenas a

estrutura (topologia) da árvore que melhor representa a matriz. Apesar disso, os métodos são largamente utilizados por apresentarem uma acurácia razoável e alta velocidade computacional. Nesta categoria destacam-se os métodos *Neighbor-Join* [63], proposto por Saitou e Nei em 1987, e suas versões aperfeiçoadas, como o BioNJ [38]. Outro método bastante usado é o UPGMA [51] (do inglês, *Unweighted Pair-Group Method with Arithmetic mean*), que supõe que a distância de todas as folhas até a raiz é a mesma (árvore ultramétrica). No nosso trabalho implementamos o método *Neighbor-Join*, conforme explicaremos na Seção 6.3.

3.1.2 Parcimônia

O princípio de parcimônia, muito empregado em diversas áreas, afirma que, se existirem várias explicações para uma dada observação, devemos adotar a mais simples. Muitos biólogos aceitam a hipótese de que a natureza é parcimoniosa, sendo econômica durante a evolução dos organismos. Sua aplicação aos métodos de inferência filogenética significa minimizar o número de mudanças evolucionárias necessárias para produzir a diferença observada no conjunto de organismos analisados.

Os dados de entrada usados na análise da máxima parcimônia estão na forma de *características*. Uma característica é composta de um número de estados, que são a manifestação da característica em uma espécie particular. Basicamente, os métodos baseados em parcimônia realizam os seguintes passos:

1. Dado um conjunto de organismos L, escolhemos um conjunto de características C e, para cada característica pertencente à C, os estados possíveis;

2. Construímos uma *matriz de características*, onde as linhas correspondem aos organismos dados e as colunas correspondem às características selecionadas. Para cada posição (i, j) da matriz, definimos o estado da característica j no organismos i;

3. A partir da matriz de características, inferimos uma árvore T, onde os vértices de T representam os estados da característica e as arestas representam a possível evolução entre os estados. A árvore resultante deve explicar a evolução com o menor número de trocas de estado.

Além da topologia, note que é possível inferir os ancestrais a partir dos estados das características. Por exemplo, na Figura 3.2 as características selecionadas para o conjunto de lagartos são: barbela, crista nas costas, crista na cabeça, manchas no corpo e rabo pintado. Os estados possíveis são presença ou ausência da característica analisada. As características presentes ou ausentes nos ancestrais dão uma noção de como eles seriam.

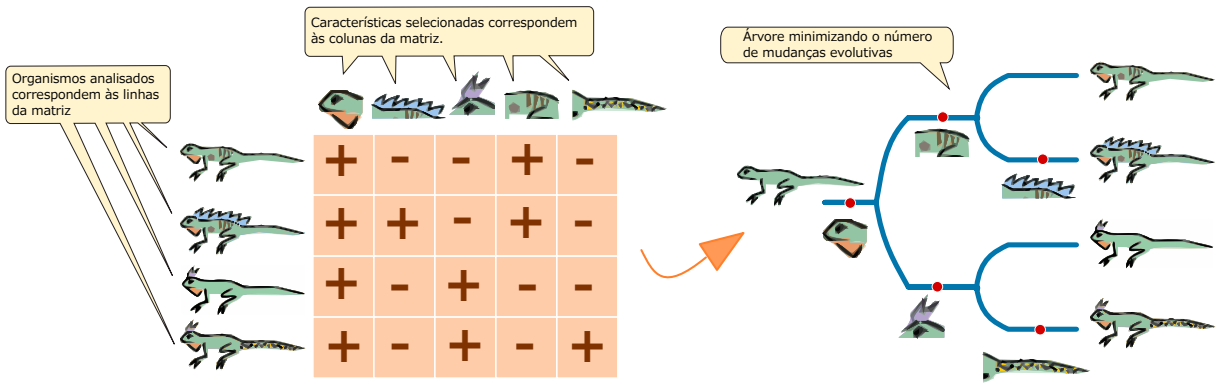

Figura 3.2: Parcimônia: Árvore filogenética é construída a partir da matriz de características. Método minimiza o número de mudanças evolutivas.

Como ocorre com os métodos baseados em distância, aqueles que supõem a parcimônia na evolução de um conjunto de espécies também estão sujeitos aos problemas causados por evolução paralela ou convergente. Outro problema é que este princípio não define como proceder no caso de existir mais que uma árvore parcimoniosa. Na prática, costuma-se calcular a árvore consenso das árvores parcimoniosas, que mantém apenas as ramificações comuns à todas árvores. Os grupos que diferem formam nós com mais que dois ramos, e são considerados "não resolvidos", já que durante o evento de especiação costumam ser geradas apenas duas espécies filhas.

Nesta categoria enquadra-se o método proposto por Fitch [36], que foi bastante utilizado em nosso trabalho e será explicado na Seção 6.3.

3.1.3 Máxima Verossimilhança

A máxima verossimilhança é uma estimativa muito utilizada em métodos de inferência estatística. Os métodos baseados em verossimilhança supõem que a árvore que explica a evolução de um conjunto de organismos é aquela que, dado um modelo de evolução específico, maximiza a probabilidade dos dados observados.

A verossimilhança inicia-se com um modelo de como os dados surgem. Esse modelo dá uma probabilidade $P[D|\Theta]$ de observar os dados a partir dos parâmetros do modelo, denotados por Θ. Os parâmetros podem incluir a topologia da árvore, os pesos das arestas da árvore e assim por diante. A idéia é escolher os valores dos parâmetros que maximizem a probabilidade dos dados observados. Dessa forma, definimos a função de verossimilhança $L(\Theta) = P[D|\Theta]$, que mostra o quão provável é observar os dados a partir de um conjunto de valores para os parâmetros de Θ. Uma alta verossimilhança indica uma boa escolha dos valores. A estimativa da máxima verossimilhança é o valor de Θ que maximiza $L(\Theta)$. Em

nosso contexto, procuramos pela estimativa da máxima verossimilhança de uma árvore filogenética.

Esta abordagem tem sido bastante usada para inferir a árvore filogenética a partir de sequências moleculares, já que estas possuem poucos estados — 4 para nucleotídeos, 20 para aminoácidos — e seus modelos de evolução são bem estudados por biólogos e bioquímicos. No exemplo da Figura 3.3, temos um modelo de evolução que define qual a probabilidade de ocorrer a substituição de um nucleotídeo. Dadas duas hipóteses de como aconteceu a evolução, a escolhida seria a mais provável entre as duas.

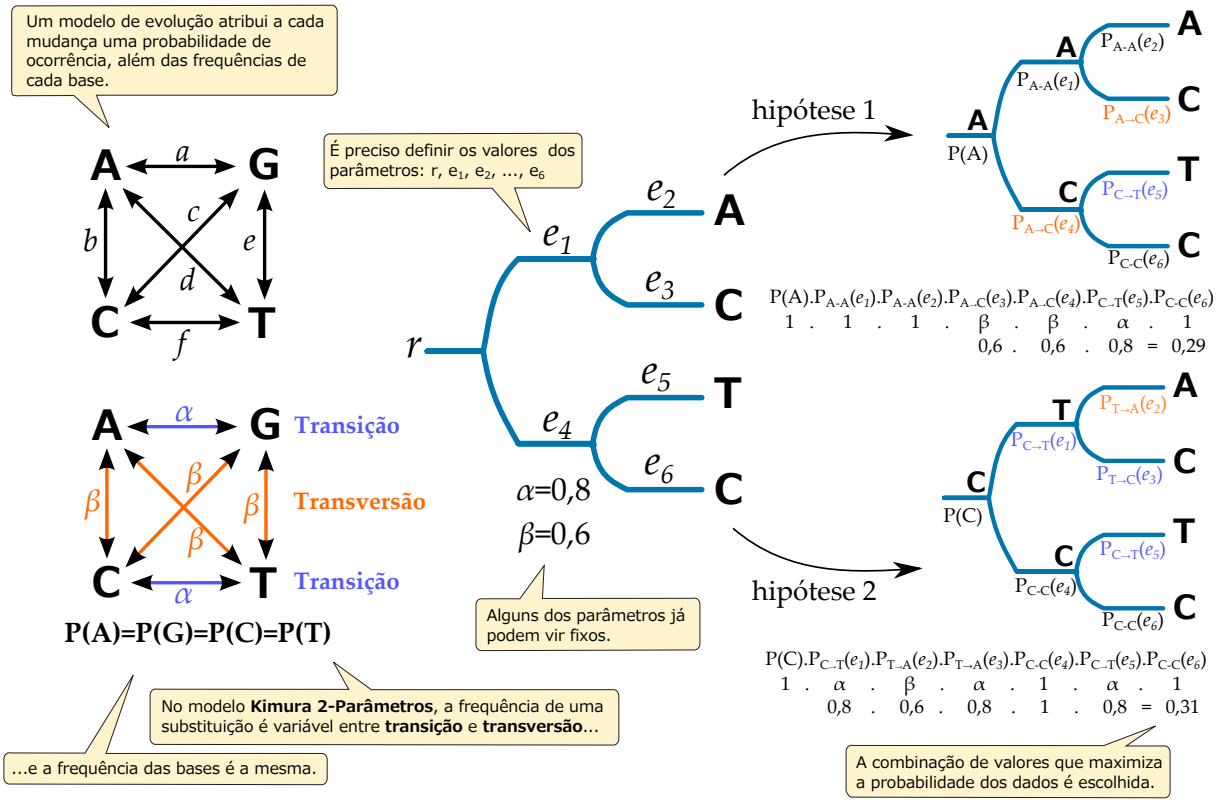

Figura 3.3: Máxima Verossimilhança: A árvore filogenética é construída a partir de um modelo de evolução e de combinações dos valores dos parâmetros. O método encontra a árvore que maximiza a probabilidade dos dados observados.

3.2 Representações

O Capítulo 1 mencionou vários tipos de dados que são usados na representação dos organismos durante a inferência filogenética. Já no Capítulo 2 vimos que a evolução nos

genomas ocorre em diferentes níveis.

Neste trabalho, os organismos serão representados por seus respectivos genomas, através da ordem dos seus genes. Os eventos evolutivos que modificam a ordem dos genes são os rearranjos, explicados na Subseção 2.3.2, que são apropriados para o estudo de relações evolutivas mais antigas. Os rearranjos são modelados a partir de *operações de rearranjo*.

Dependendo da quantidade de genomas recebidos na entrada, também precisamos utilizar árvores filogenéticas para representar a hipótese inferida pelo método.

Nas subseções a seguir definimos a representação dessas informações.

3.2.1 Genoma

Nos primeiros esforços para definir formalmente um genoma, Watterson et al. [75] propuseram representar as posições relativas dos genes como uma permutação com sinais, onde a cada gene é atribuído um número único, que é encontrado exatamente uma vez em cada genoma. Entretanto, havia a desvantagem de não ser possível aplicar rearranjos que modificassem a quantidade de genes no genoma, como duplicações e deleções, por exemplo.

Recentemente, outras abordagens baseadas em conjuntos e grafos foram propostas para tratar o genoma de um modo mais abrangente. Neste trabalho adotaremos a representação proposta por Bergeron [10], em 2006.

Mantendo as definições da Seção 2.2, para cada gene a existem duas extremidades: uma extremidade *tail* a_t e uma extremidade *head* a_h, que representam as extremidades inicial (5') e final (3') do gene, respectivamente. Dado um conjunto de genes $\mathcal{G}$, o conjunto de extremidades correspondente é definido por:

$$\mathcal{E}(\mathcal{G}) = \{a_t : a \in \mathcal{G}\} \cup \{a_h : a \in \mathcal{G}\}.$$

Uma *adjacência* é um par não ordenado de extremidades, que representa a ligação entre dois genes consecutivos em uma certa orientação no cromossomo. Dada uma adjacência entre dois genes consecutivos a e b, dependendo de suas orientações (vide Figura 2.2), a adjacência pode se apresentar de quatro formas:

$$a_h b_t,\ a_h b_h,\ a_t b_t,\ a_t b_h.$$

Uma extremidade que não é adjacente a qualquer outra extremidade define um *telômero*, representada por um único elemento a_h ou a_t. Note que os pares $a_h b_t$ e $b_t a_h$ representam a mesma adjacência.

Um *genoma* é definido por um par $(\mathcal{G}, \Pi)$, onde $\mathcal{G}$ é o conjunto de genes e Π é um

conjunto de adjacências disjuntas de $\mathcal{E}(\mathcal{G})$. Apesar de não existir um conjunto que represente os telômeros, eles são determinados unicamente através do conjunto de genes $\mathcal{G}$ e do conjunto de adjacências Π.

Duas adjacências são ditas *conflitantes* quando elas compartilham pelo menos uma extremidade em comum. Dessa forma, dado um conjunto de genes, um genoma pode ser caracterizado como um conjunto de adjacências mutualmente não conflitantes.

Na Figura 3.4 temos um genoma multicromossomal, constituído por dois cromossomos lineares e um cromossomo circular, que é representado da seguinte forma:

$$\mathcal{G} = \{a,\ b,\ c,\ d,\ e,\ f,\ g\},$$
$$\Pi = \{a_h c_t,\ c_h d_h,\ b_h e_t,\ b_t e_h,\ f_h g_t\}.$$

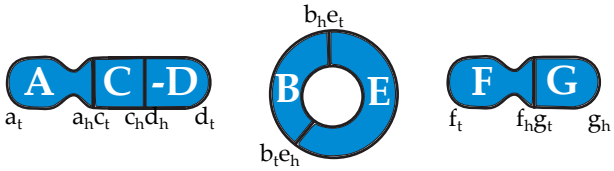

Figura 3.4: Representação do Genoma.

3.2.2 Árvore Filogenética

Dado um conjunto de genomas $G_1, G_2, \ldots, G_n$, definimos uma árvore filogenética como uma árvore $T = (V, E)$ onde os nós com grau 1 (folhas) são associados ao genomas de $G_1, G_2, \ldots, G_n$.

Os nós internos possuem grau pelo menos 3. Quando todos os nós internos possuem grau 3, a árvore é binária, e dizemos que é *completamente resolvida*. Nós internos com grau maior que 3 representam incertezas na ordem de especiação.

A árvore filogenética pode ou não ter uma raiz, um nó interno de grau 2 que representa o ancestral comum a todos. No caso de árvores filogenéticas enraizadas, definimos como *topologia* a árvore filogenética sem raiz associada (obtida a partir de T através da supressão da raiz).

3.2.3 Operação de Rearranjo

Uma *operação de rearranjo*, também chamada de modelo de rearranjo, pode ser vista como uma função ρ que relaciona dois genomas, ou seja, a aplicação da operação ρ sob um genoma π resulta em um novo genoma σ.

As operações de rearranjo especificam os tipos de rearranjo permitidos, que podem ser desde um único evento até uma combinação de eventos. Quando a operação de rearranjo combina dois ou mais eventos, é dado um peso a cada um deles de acordo com a probabilidade de ocorrência, onde um peso menor implica uma probabilidade maior de ocorrência.

Cada modelo de rearranjo possui uma definição precisa de suas operações. No Capítulo 5 apresentamos uma revisão bibliográfica desses modelos e também definimos formalmente as operações que compõem o modelo *Single-Cut-or-Join*, que é o foco da nossa pesquisa.

3.3 Conclusões

Este capítulo detalhou os tipos de dados utilizados, com sua respectiva representação, e também apresentou os princípios comumente usados pelos métodos de inferência filogenética. Um desses princípios, a parcimônia (Seção 3.1.2), é o critério usado na resolução dos problemas estudados, que são definidos no Capítulo 4.

Capítulo 4

Problemas de Rearranjo de Genomas

Na Seção 3.1.2 vimos que a filogenia construída com o método da parcimônia procura explicar a evolução com o menor número de trocas de estado. As trocas de estado ocorrem conforme a operação de rearranjo ρ escolhida.

Este problema pode ser definido como um problema de otimização combinatória, que possui diversas variantes. Nas seções a seguir apresentamos variações do problema conforme o número de genomas recebidos na entrada (vide Tabela 4.1).

Tabela 4.1: Algumas variantes do Problema de Rearranjo em Genomas.

Problema de Rearranjo	Dados de entrada
Distância	2 genomas
Mediana	3 genomas
Parcimônia, variante Pequena	n genomas, com árvore filogenética dada
Parcimônia, variante Grande	n genomas, sem árvore filogenética dada

4.1 Problema da Distância

O problema da distância entre dois genomas é definido como segue:

Dados dois genomas G_1 e G_2, encontre o número mínimo de eventos, definidos de acordo com a operação de rearranjo ρ, que transformam G_1 em G_2.

Note que neste problema estamos interessados apenas no valor da distância, que designamos como $d(G_1, G_2)$. Um outro problema, chamado de Problema da Ordenação (*Sorting*), consiste em encontrar, além da distância, a sequência de eventos que separam dois genomas.

Na Figura 4.1 temos um exemplo extraído do artigo de Hannenhalli e Pevzner [41], onde eles propuseram uma operação de rearranjo que modela inversões. Neste caso, a distância evolutiva entre os dois genomas é 3, e a ordenação deles é a sequência de inversões realizada. Note que os dois genomas utilizados, do repolho e do nabo, são os genomas mitocondriais que, conforme explicado na Seção 2.2, possuem a ordem dos genes altamente conservada.

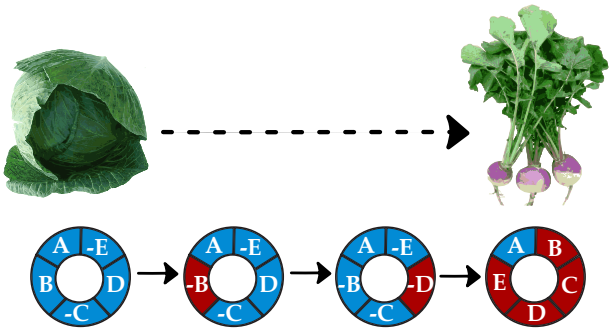

Figura 4.1: Transformando repolho em nabo através de inversões.

4.2 Problema da Mediana

O Problema da Mediana, no contexto de rearranjos de genomas, consiste em, dados três genomas G_1, G_2 e G_3, encontrar um genoma G_M que minimize a seguinte soma:

$$d(G_1, G_M) + d(G_2, G_M) + d(G_3, G_M).$$

Dependendo da operação de rearranjo ρ este problema é NP-Difícil, conforme veremos no Capítulo 5. A Figura 4.2 exemplifica o problema.

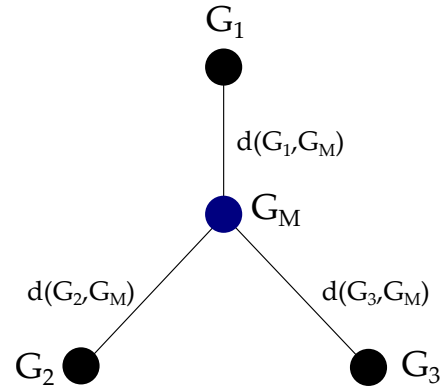

Figura 4.2: Problema da Mediana.

4.3 Problema de Rearranjo de Múltiplos Genomas

Quando o problema recebe mais que três espécies, ele pode ser decomposto em dois aspectos que devem ser resolvidos:

1. Encontrar a forma ótima (ou topologia) da árvore;

2. Otimizar a reconstrução do ancestral de cada nó interno.

Estes aspectos dão origem aos dois problemas tratados nas próximas subseções.

4.3.1 Problema Pequeno da Parcimônia (PPP)

Dada a topologia da árvore T, um conjunto de genomas $G_1, G_2, \ldots, G_n$ — que estão nas folhas de T — e uma operação de rearranjo ρ, o objetivo do Problema Pequeno da Parcimônia é construir um conjunto de genomas e associá-los aos nós internos de T, minimizando o custo total da árvore T.

O custo de T é definido como o número de trocas de estado que explicam os dados, isto é, o número de passos evolucionários, tais como perda, ganho ou modificação de uma característica. O custo é calculado através da soma do peso de cada aresta de T, sendo o peso de uma aresta o número de eventos evolutivos que explica a evolução entre os genomas dos vértices incidentes. Os eventos evolutivos permitidos, assim como a complexidade do problema, dependem da operação de rearranjo ρ aplicada.

4.3.2 Problema Grande da Parcimônia (PGP)

Dado um conjunto de genomas $G_1, G_2, \ldots, G_n$ e uma operação de rearranjo ρ, o Problema Grande da Parcimônia consiste em reconstruir uma árvore filogenética T, tal que o custo total de T seja mínimo.

O problema tem como referência apenas as características das espécies atuais, que serão as folhas da árvore, o que torna o problema muito mais complexo pois, além de atribuir os estados das características dos nós internos, também é necessário definir a topologia da árvore.

4.4 Conclusões

Este capítulo definiu os problemas de rerranjo de genomas aqui estudados. Note que a definição destes problemas é independente do que é um "genoma", e de quais eventos são permitidos, o que torna os problemas de rearranjo bastante diversificados. Vários modelos de rearranjo tem sido propostos, e no Capítulo 5 fazemos uma revisão bibliográfica deles.

Capítulo 5

Operações de rearranjo

No Capítulo 4 vimos que existem diversas variantes do problema de rearranjo em genomas e definimos aquelas que foram estudadas. Independente da variante, a operação de rearranjo impacta tanto na complexidade do problema quanto na qualidade dos resultados.

Neste capítulo fazemos uma revisão bibliográfica das principais operações de rearranjo propostas e, na Seção 5.2, especificamos a operação *Single-Cut-or-Join*, objeto de estudo deste trabalho.

5.1 Estado da Arte

Embora muitas operações de rearranjo tenham sido propostas, somente um conjunto reduzido delas vem sendo estudadas em problemas de rearranjo que envolvem mais que dois genomas. Na primeira parte dessa revisão abordamos as operações que vem recebido atenção nesses problemas. Na segunda parte da revisão apresentamos uma listagem de outras operações propostas que contribuíram para o avanço da área.

5.1.1 *Breakpoints*

Nos primeiros estudos computacionais em rearranjos de genomas, Watterson et al. [75] definiram o problema de transformar um genoma em outro através de inversões, representando os genomas como permutações, sem considerar o sentido de leitura dos genes. Durante o cálculo de um limitante inferior para o problema, introduziram o conceito de *breakpoints*, uma adjacência presente em somente um dos genomas. Dessa forma, a medida de *breakpoints* foi a primeira tentativa de computar as diferenças entre genomas considerando a ordem dos genes e, as permutações, a primeira forma de definir um genoma. A distância de *breakpoints* é facilmente calculada em tempo linear no tamanho do genoma (número de genes). Note que os *breakpoints* eram vistos mais como uma medida

do que uma operação de rearranjo, até o surgimento do SCJ, já que não era possível definir uma sequência de operações que transformasse um genoma em outro por *breakpoints*, por exemplo.

Quinze anos mais tarde, Sankoff e Blanchette [66] iniciaram o estudo do Problema da Mediana por *Breakpoints*. Neste trabalho eles também propuseram uma redução do problema para o Problema do Caixeiro Viajante (TSP), que vem sendo estudado extensivamente e, apesar de ser NP-difícil, possui bons métodos para resolvê-lo. No ano seguinte, Pe'er e Shamir [58] e Bryant [18] independentemente provaram que trata-se de um problema NP-Completo, considerando os genomas definidos como permutações.

Usando a redução para o problema do TSP, Sankoff e Blanchette [13] descreveram o primeiro método para resolver, de forma heurística, os Problemas Pequeno e Grande da Parcimônia por *breakpoints*. A heurística itera pela árvore resolvendo o problema da mediana entre os nós. Mais tarde, a técnica foi implementada recebendo o nome *BPAnalysis* [67]. Cosner et al. [23, 22] fizeram algumas melhorias e aplicaram o modelo a dados reais.

Desde então, a distância de *breakpoints* passou a ser bastante estudada com permutações. Outras propostas foram feitas no sentido de melhorar a eficiência do BPAnalysis, como o software GRAPPA, desenvolvido por Moret et al. [53] e, recentemente, uma alternativa à redução para o TSP [39]. Somente em 2009, Tannier, Zheng e Sankoff [73], utilizaram uma outra representação de genoma, diferente de permutações. Representando os genomas da forma proposta por Bergeron [10] (a qual também utilizamos, conforme mencionado na Seção 3.2.1), foi possível modelar genomas com mais de um cromossomo. Ao estudar o problema da mediana utilizando essa representação, eles provaram que, para o caso de genomas com vários cromossomos contendo somente genomas lineares, o problema é NP-difícil. Um outro resultado interessante foi a complexidade no caso de genomas com vários cromossomos, contendo cromossomos lineares e circulares (ou somente circulares), onde o problema é polinomial, permanecendo em aberto nas variantes pequena e grande da parcimônia. A versão polinomial do problema foi implementada mais tarde por Adam e Sankoff [2], substituindo a redução da mediana para o TSP na heurística que resolve o problema da parcimônia. Em todos os casos, os genomas analisados devem possuir os mesmos genes, sem repetições.

5.1.2 Hannenhalli-Pevzner (HP)

O problema introduzido por Watterson et al. [75], de transformar um genoma em outro através de inversões, desconsiderando o sentido dos genes, foi o primeiro problema combinatorial da área e o mais estudado, dando origem a teoria do grafo de *breakpoints* de Bafna e Pevzner [7], que é uma forma de analisar as permutações dos genomas. Neste

trabalho, Bafna e Pevzner também estudaram uma outra versão do problema, com maior relevância biológica: será que é possível resolver o problema das inversões, considerando o sentido dos genes, em tempo polinomial?

Surpreendentemente, eles mostraram que neste segundo caso o problema torna-se polinomial. Aproveitando o grafo de *breakpoints*, Hannenhalli e Pevzner [40], em 1995, provaram o *teorema da dualidade* e obtiveram o primeiro algoritmo de tempo polinomial para resolver o problema de inversões com sinal — em contraste a esse problema, Caprara provou que o problema original, sugerido por Watterson et al., é NP-difícil [20]. O modelo de Hannenhalli e Pevzner também foi o primeiro a tratar eventos em genomas multicromossomais, modelando as operações de translocação, fissão e fusão.

O modelo de Hannenhalli e Pevzner ficou conhecido como modelo HP, e sua teoria foi muito estudada nos anos seguintes. Em 2001, Bader et al. [5] propuseram o primeiro algoritmo de tempo linear para computar a distância de inversão, que inicialmente era $O(n^4)$. Nesse mesmo ano, El-Mabrouk [28] estendeu a teoria HP para tratar inserções e deleções. Algumas correções também foram feitas na teoria HP: primeiramente por Tesler [74], em 2002; depois, Ozery-Flato e Shamir [56] encontraram um contra-exemplo para a demonstração do teorema da dualidade, revisando a demonstração do teorema e os algoritmos correspondentes; e, em 2007, outra correção foi apresentada por Jean e Nikolski [44].

Os problemas de rearranjo com mais de dois genomas, até então bastante estudados com *breakpoints*, começaram a ser estudados com a teoria HP em 2001, com a abordagem exata de Siepel [70]. No ano seguinte, Bourque e Pevzner [16] propuseram uma heurística para resolver o problema da mediana e da parcimônia, além de implementarem o software MGR, aplicando-o a conjuntos de dados reais. Em 2003, Caprara [21] provou que o problema da mediana por reversões com sinais é NP-difícil, no caso de genomas unicromossomais representados como permutações. A complexidade dos problemas da mediana e parcimônia, para o caso de genomas multicromossomais, permanece em aberto [73].

5.1.3 *Double-Cut-and-Join* (DCJ)

O modelo HP é considerado um modelo biologicamente realístico, por modelar operações comumente observadas. Ademais, sua distância pode ser calculada em tempo linear. Entretanto, o modelo HP tem uma teoria bastante complicada, que inclui etapas de pré-processamento, como a inserção de *caps* nos cromossomos e a concatenação, além de envolver sete parâmetros, cada um representando uma propriedade combinatorial do problema. Em contraste ao complicado modelo HP, Yancopoulos et al. [79] introduziram uma nova operação chamada *Double-Cut-and-Join* (DCJ), que usa uma representação mais geral dos genomas, considerando tanto genomas lineares como circulares, unicromossomais

e multicromossomais. Além das inversões, translocações, fusões e fissões já modeladas por HP, o DCJ também modela transposições e troca de blocos, um tipo de transposição mais geral. Apesar de menos realístico, devido a inserção das novas operações que não são tão observadas como inversões e translocações, o problema da distância e da ordenação são computados de forma simples e eficiente.

A princípio o modelo DCJ utilizava o grafo de *breakpoints* e a teoria HP para representar os genomas. Entretanto, em 2006, Bergeron et al. [10] simplificaram a forma como os genomas são representados, introduzindo uma nova estrutura chamada de *grafo de adjacências*, reduzindo a complexidade do problema de ordenação de genomas por DCJ, que passou a ser linear. Nesta nova representação, não há uma relação direta entre operações DCJ e eventos biológicos, como ocorria no algoritmo original, onde havia fases para realizar operações DCJ relacionadas à cada tipo de evento (translocações, reversões, etc.).

O modelo DCJ e a nova estrutura de *grafo de adjacências* passaram a ser bastante estudados devido à sua clareza e simplicidade. Yancopoulos e Friedberg [80] estenderam o modelo DCJ a fim de tratar inserções e deleções e, em 2010, Braga et al. [17] propuseram um algoritmo para computar a distância DCJ com estas novas operações, em tempo linear.

O problema da mediana por DCJ foi provado ser NP-difícil tanto para os casos onde os genomas são unicromossomais (permutações) [21] como para o caso onde os genomas são multicromossomais com cromossomos circulares e lineares, que é o cenário mais geral, e também para cromossomos somente circulares. Os últimos dois resultados foram provados em 2009 por Tannier et al. [73], permanecendo em aberto a variante do problema da mediana por DCJ onde os genomas são multicromossomais e lineares.

Apesar disso, existem muitas propostas para computar a mediana e a árvore filogenética por DCJ. Em 2008, Adam e Sankoff [1] propuseram um algoritmo para resolver o problema de reconstrução filogenética inspirado nos algoritmos mencionados anteriormente, que resolvem sucessivas medianas, e aplicaram o método a dados reais. No mesmo ano, Xu et al. [76, 77] propuseram abordagens exatas para computar a mediana eficientemente.

5.1.4 *Single-Cut-or-Join* (SCJ)

A proposta do modelo DCJ generalizou e descomplicou a teoria do modelo HP. Mas seria possível propor um modelo ainda mais simplificado? Em busca de um modelo de rearranjo que fosse o mais simples possível, Feijão e Meidanis [30], em 2009, propuseram um modelo minimalista chamado *Single-Cut-or-Join* (SCJ). O modelo, baseado na medida de *breakpoints*, é composto de duas operações, explicadas na Seção 5.2, e trata de todos os eventos clássicos, tais como inversão, translocação, transposição, fissão e fusão. Entre-

tanto, ao definirmos o peso de cada evento como o número de SCJs que um evento precisa para ser realizado, vemos que a distribuição dos pesos não segue os valores comumente adotados, além de não existir um mapeamento direto para eventos biológicos observados com frequência, como inversões. Por estes motivos, este seria um modelo menos realístico que DCJ e HP, por exemplo.

A simplicidade do modelo trouxe resultados interessantes: Feijão e Meidanis [31] provaram que a complexidade do Problema da Mediana é linear no tamanho do genoma, enquanto o Problema Pequeno da Parcimônia é linear em relação ao número de genomas e seus respectivos tamanhos. Como vimos anteriormente, a maioria dos problemas com mais de dois genomas torna-se NP-difícil sob diferentes modelos, exceto em alguns casos bastante específicos. Outros problemas de rearranjos também tornaram-se polinomiais com o modelo SCJ, embora não sejam abordados em nosso estudo.

Nesse mesmo trabalho, também comprovou-se que a variante grande da parcimônia é NP-difícil. Apesar disso, é possível desenvolver abordagens exatas e heurísticas eficientes aproveitando o fato da variante pequena ser polinomial.

A princípio o modelo SCJ seria utilizado apenas como uma aproximação para modelos mais realísticos. Apesar da teoria do SCJ ser bem construída, ainda existe uma lacuna sobre a relevância biológica da operação. Enquanto as outras operações foram extensivamente aplicadas a dados reais e simulações, por ser uma operação recente, ainda não há estudos deste tipo para a operação SCJ. Este trabalho visa preencher esta lacuna, fornecendo um estudo experimental consistente, analisando a operação SCJ sob diferentes cenários evolutivos e verificando a qualidade de seus resultados na reconstrução de histórias evolutivas.

5.1.5 Outras Operações

Além das operações de rearranjo mencionadas acima, recentemente outras operações, como a transposição [81, 82], começaram a ser estudadas em problemas de rearranjo de múltiplos genomas. Recentemente, em 2010 e 2011, respectivamente, foi comprovado que tanto o problema da distância [19] como o problema da mediana [6] por transposição em permutações é NP-difícil.

Apesar de não serem consideradas em nosso estudo, existe uma quantidade abundante de operações de rearranjo propostas, a maioria sendo estudada em problemas que envolvem dois genomas, como distância e ordenação. A Tabela 5.1 sumariza os principais modelos propostos, relacionando-os com os eventos biológicos tratados e sua complexidade nos problema da distância, a variante mais simples dos problemas de rearranjo, e no problema da ordenação, onde é necessário fornecer a sequência dos eventos.

Note que o evento de duplicação não é tratado em nenhum dos modelos, devido a

Tabela 5.1: Operações de rearranjo propostas, eventos modelados e complexidade dos problemas de distância e ordenação. Abreviações utilizadas: Inv., Inversão; Trl., Translocação; Trp., Transposição; Fus., Fusão; Fis., Fissão; Dist., Distância; Ord., Ordenação. A complexidade é dada em relação ao tamanho do genoma, sendo n o número de genes.

Operação	Genoma	Ano	Eventos					Problemas	
			Inv.	Trl.	Trp.	Fus.	Fis.	Dist.	Ord.
Inversão [7]	Permutação com sinais	1993	•					$O(n)$ [5]	$O(n\sqrt{n \log n})$ [72]
Transposição [8]	Permutação com sinais	1995			•			NP [19]	Aberto
Translocação [45]	Permutação com sinais	1995		•				$O(n)$ [48]	$O(n^{\frac{3}{2}}\sqrt{\log n})$ [57]
HP [40]	Grafo de *breakpoints*	1995	•	•		•	•	$O(n)$ [11]	$O(n^4)$ [40]
Combinação [8]	Permutação com sinais	1995	•		•			Aberto [49]	Aberto [49]
Combinação [27]	Formalismo Algébrico	2001			•	•	•	$O(n)$ [27]	$O(n^2)$ [27]
Double-Cut-and-Join [79]	Grafo de Adjacências	2005	•	•	•	•	•	$O(n)$ [79]	$O(n)$ [10]
Single-Cut-OR-Join [30]	Grafo de Adjacências	2009	•	•	•	•	•	$O(n)$ [30]	$O(n)$ [30]
Single-Cut-AND-Join [9]	Grafo de Adjacências	2010	•	•	•	•	•	$O(n)$ [9]	$O(n)$ [9]

dificuldade combinatorial em modelá-lo. Entretanto, várias abordagens tem sido propostas [26, 25, 65] com o objetivo de aplicar um pré-tratamento aos genomas de forma que estes fiquem com o mesmo conteúdo gênico, sem duplicações. As duplicações também dão origem a outros problemas de rearranjo, como o *genome halving problem*, onde o genoma inteiro é duplicado. Estes problemas tem sido bastante estudados [4, 52], inclusive com o modelo SCJ [30]. Apesar de bastante interessantes, problemas que envolvem duplicações, perda e ganho de genes não serão abordados em nosso trabalho.

5.2 A operação de rearranjo *Single-Cut-or-Join* (SCJ)

Nesta seção descrevemos a operação *Single-Cut-or-Join*, definindo formalmente as operações que a compõem, os eventos modelados e a complexidade dos problemas de rearranjo ao usarem o SCJ como operação evolutiva.

5.2.1 Operações

O SCJ é baseado nas duas operações mais básicas existentes: o *cut*, uma operação que quebra uma adjacência em dois telômeros (extremidades), e o *join*, que é a operação contrária, emparelhando dois telômeros em uma adjacência. Qualquer *cut* ou *join* aplicado a um genoma será chamado de uma operação *Single-Cut-or-Join* (SCJ). Como estamos representando o genoma como um conjunto de adjacências (vide Seção 3.2.1), um *cut* pode também ser visto como a remoção de uma adjacência do conjunto, enquanto o *join* é a inclusão de uma adjacência não conflitante.

Cut

Dado o genoma $G_1 = (\mathcal{G}, \Pi_1)$, sendo $\mathcal{G}$ o conjunto de genes e Π_1 o conjunto de adjacências, e uma adjacência $e_i e_j$ pertencente a Π_1, a operação *cut* define um novo genoma G_2 da seguinte forma:

$$G_2 = cut(G_1, e_i e_j) = (\mathcal{G}, \Pi_1 \setminus \{e_i e_j\}).$$

A Figura 5.1 exemplifica o efeito da operação *cut* em um genoma.

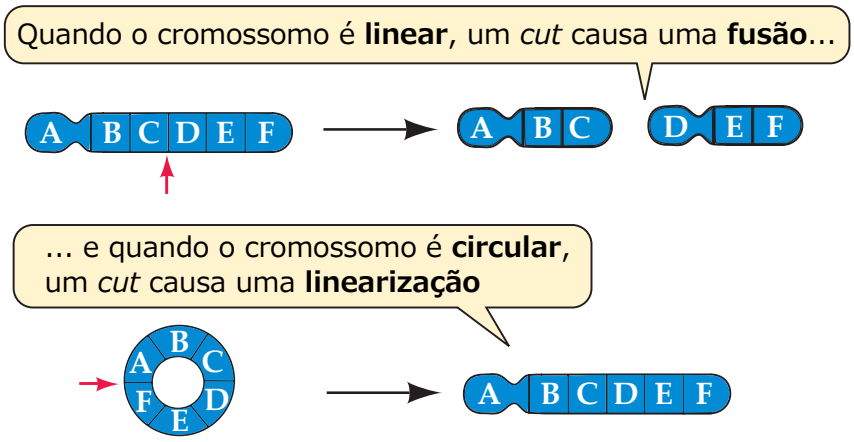

Figura 5.1: Efeito do *cut* em cromossomos lineares e circulares.

Join

Seja $G_1 = (\mathcal{G}, \Pi_1)$ um genoma representado pelo seu conjunto de genes $\mathcal{G}$ e seu conjunto de adjacências Π_1. Dadas duas extremidades e_i, e_j, que devem estar livres no genoma G_1, e o conjunto de extremidades $\mathcal{E}(\mathcal{G})$, a operação *join* define um novo genoma G_2 da seguinte forma:

$$G_2 = join(G_1, e_i e_j) = (\mathcal{G}, \Pi_1 \cup \{e_i e_j\}).$$

A Figura 5.2 exemplifica o efeito da operação *join* em um genoma.

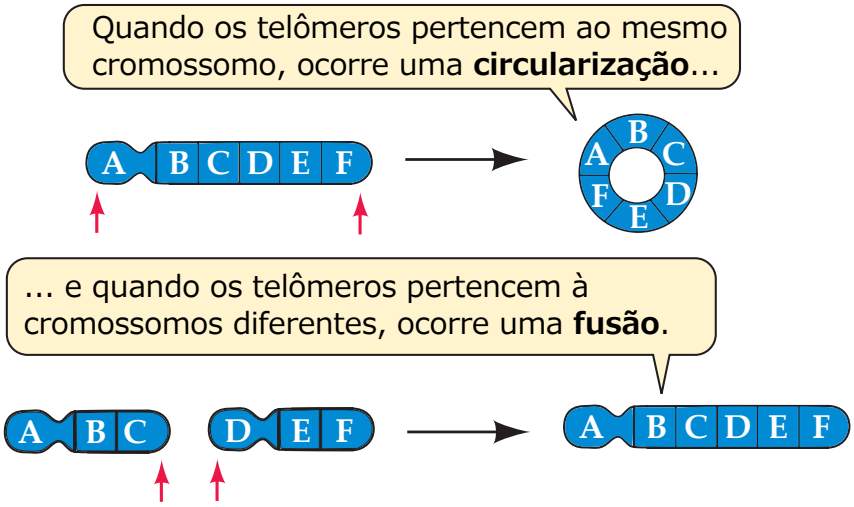

Figura 5.2: Efeito do *join* quando os telômeros pertencem ao mesmo cromossomo e quando pertencem a cromossomos distintos.

5.2.2 Eventos biológicos

A operação SCJ modela todos os eventos biológicos clássicos: inversão, transposição, translocação, fissão e fusão. O peso de cada evento corresponde ao número de operações SCJ necessárias para executá-lo. A Tabela 5.2 compara os pesos atribuídos nos modelos SCJ, DCJ e HP, definindo como peso o número de operações necessárias para modelar o evento.

Como vimos na Seção 3.2.3, quanto menor o peso atribuído a uma operação, maior a probabilidade de ocorrência na natureza. No trabalho de Blanchette et al. [14], foram analisadas as frequências com que ocorriam os eventos em diferentes contextos. Em todos os cenários analisados, a transposição foi menos observada que os demais eventos, e eles concluíram que ela deveria ter um peso muito maior que as demais operações, sugerindo que o peso da transposição fosse duas vezes maior que o peso da inversão. Além disso, nos

Tabela 5.2: Comparação entre as operações de rearranjo HP, DCJ e SCJ, com relação aos pesos dados à cada evento biológico. Abreviações utilizadas: Inv., Inversão; Inv. Suf., Inversão de Sufixo/Prefixo; Trp., Transposição; Trc. Blc., Troca de Blocos; Trl., Translocação; Fus., Fusão; Fis., Fissão.

Operação	Eventos						
	Inv.	Inv. Suf.	Trp.	Trc. Blc.	Trl.	Fus.	Fis.
HP [40]	1	1			1	1	1
DCJ [79]	1	1	2	2	1	1	1
SCJ [30]	4	2	6	8	4	1	1

pesos comumente adotados para os eventos, não há distinção entre inversões e inversões prefixo/sufixo. Dessa forma, os pesos adotados pelo SCJ não seguem a convenção. Note também que o SCJ não modela eventos de perda e ganho de genes, nem duplicações.

5.2.3 Problemas de Rearranjo sob o modelo SCJ

A seguir definimos a complexidade dos problemas de rearranjo ao utilizar a operação SCJ. A representação utilizada para os genomas é definida na Seção 3.2.1.

Problema da Distância

A distância SCJ, denotada por d_{SCJ}, é definida como o menor número de operações *single-cut-or-join* que transformam um genoma em outro. A distância SCJ é facilmente computada, como veremos abaixo.

Lema 1 ([30]) *Considere dois genomas $G_1 = (\mathcal{G}, \Pi)$ e $G_2 = (\mathcal{G}, \Sigma)$, onde $\mathcal{G}$ é o conjunto de genes e Π e Σ são o conjunto de adjacências dos genomas G_1 e G_2, respectivamente. Seja $\Gamma = \Pi - \Sigma$ e $\Lambda = \Sigma - \Pi$. Então, Γ e Λ podem ser computados em tempo linear, e definem o conjunto mínimo de operações SCJ que transformam Π em Σ, onde adjacências em Γ definem cuts e adjacências em Λ definem joins. Consequentemente, $d_{SCJ}(G_1, G_2) = |\Pi - \Sigma| + |\Sigma - \Pi|$.*

Além da definição utilizando operações em conjuntos, também é possível definir a distância SCJ baseado no grafo de adjacências proposto por Bergeron [10]. Esse resultado é demonstrado no trabalho de Feijão e Meidanis [30], entretanto não utilizaremos essa outra alternativa em nosso trabalho.

Problema da Mediana

Dados três genomas G_1, G_2 e G_3, a mediana G_M é o genoma que minimiza a distância SCJ entre ele e os demais genomas:

$$d_{SCJ}(G_1, G_M) + d_{SCJ}(G_2, G_M) + d_{SCJ}(G_3, G_M).$$

Sejam Π_1, Π_2 e Π_3 o conjunto de adjacências dos genomas G_1, G_2 e G_3, respectivamente. Dada uma adjacência xy pertencente ao conjunto $\Pi_1 \cup \Pi_2 \cup \Pi_3$, existem três possíveis casos:

- *Caso 1:* xy pertence aos três genomas. Nesse caso, não incluir xy em G_M implicaria em operações SCJ em todos os genomas. Portanto, é mais vantajoso que G_M tenha xy.
- *Caso 2:* xy pertence a dois genomas. Se xy não for incluído em G_M, são necessárias operações SCJ em dois dos três genomas. Portanto, é mais vantajoso que G_M tenha xy.
- *Caso 3:* xy pertence a somente um genoma. Caso xy seja incluído em G_M, as operações SCJ serão realizadas em dois dos três genomas. Portanto, é mais vantajoso que G_M não tenha xy.

Portanto, o conjunto de adjacências da mediana por SCJ é definido como:

$$\Gamma = (\Pi_1 \cap \Pi_2) \cup (\Pi_1 \cap \Pi_3) \cup (\Pi_2 \cap \Pi_3).$$

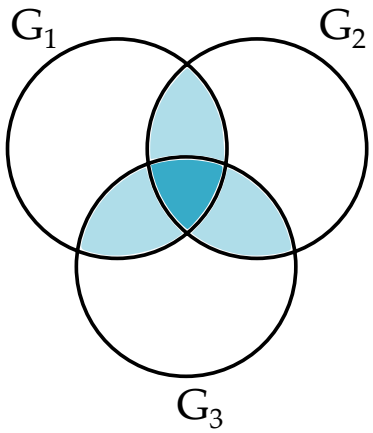

Figura 5.3: Mediana por SCJ. As partes realçadas representam a mediana. Cada conjunto corresponde ao conjunto de adjacências de um genoma. A mediana é composta por todas as adjacências que estão em pelo menos dois dos três genomas.

A mediana engloba os casos 1 e 2 descritos acima, conforme podemos ver na Figura 5.3. Note que não há adjacências conflitantes em Γ. Nenhuma adjacência do conjunto $\Pi_1 \cap \Pi_2$ conflita com o conjunto $\Pi_1 \cap \Pi_3$, pois ambos são subconjuntos de Π_1, e Π_1 não possui adjacências conflitantes. Da mesma forma, $\Pi_1 \cap \Pi_2$ não conflita com $\Pi_2 \cap \Pi_3$ pois Π_2 não possui adjacências conflitantes. O mesmo raciocínio aplica-se aos demais casos.

Nesta seção apresentamos apenas uma idéia da estratégia utilizada para o cálculo da mediana. Uma prova mais formal e generalizada pode ser encontrada no trabalho de Feijão e Meidanis [30].

Problema Pequeno da Parcimônia

Dada uma árvore T, em que cada folha corresponde a um genoma definido sob o mesmo conjunto de genes $\mathcal{G}$, o Problema Pequeno da Parcimônia consiste em encontrar um genoma ancestral $G_v = (\mathcal{G}, \Gamma_v)$ para cada nó interno v de T, tal que o tamanho da árvore T (a soma do peso de cada aresta, definido como a distância SCJ entre os genomas dos vértices incidentes) é minimizado. Formalmente, denotando o custo da árvore como c_T, desejamos encontrar um mapeamento M tal que:

$$c_T = \min_M \sum_{uv \in E(T)} d_{SCJ}(G_u, G_v), \tag{5.1}$$

onde $E(T)$ é o conjunto de arestas de T e M é o mapeamento de v para G_v.

No modelo SCJ é possível resolver este problema em tempo polinomial aplicando o algoritmo de Fitch [36] a cada adjacência, conforme explicaremos a seguir.

Dada a topologia de uma árvore e, para cada folha, um estado possível da característica, o algoritmo de Fitch determina os estados dos nós internos de modo que minimize o número de trocas evolucionárias (princípio da Parcimônia, vide Seção 3.1.2). Neste caso, as características são as adjacências dos genomas e os estados possíveis são a presença ou ausência da adjacência. O algoritmo é executado em duas etapas: uma primeira, das folhas para a raiz, e uma segunda, da raiz para as folhas.

Na primeira parte do algoritmo, é atribuído a cada nó interno um conjunto de possíveis estados, dependendo do conjunto de estados dos seus filhos. Esse conjunto é denotado por $B(d, v)$, onde d é uma adjacência e v é um nó da árvore. O conjunto $B(d, v)$ pode assumir os seguintes valores: $\{0\}$, quando ambos filhos não possuem a adjacência; $\{1\}$ quando ambos filhos possuem; $\{0, 1\}$, quando um filho possui a adjacência, mas outro não. No caso das folhas, é possível atribuir apenas $\{0\}$ ou $\{1\}$.

Na segunda parte do algoritmo, todos os nós já possuem seu conjunto $B(d, v)$ definido. O estado final escolhido, denotado por $F(d, v)$, será o próprio conjunto $B(d, v)$, se este possuir somente um elemento, ou o $F(d, u)$ de seu nó pai u, caso $B(d, v)$ possua dois ele-

mentos. Se a raiz r possuir $B(d, r) = \{0, 1\}$, basta escolher um estado arbitrário no algoritmo original. Entretanto, essa estratégia funcionaria somente se todas as características fossem independentes. Neste caso, como as características equivalem às adjacências, elas não são independentes, pois caso uma adjacência xy esteja presente em um genoma, todas as outras adjacências que utilizarem as extremidades x ou y devem estar ausentes, pois são adjacências conflitantes.

Um passo importante para evitar a ocorrência de adjacências conflitantes é atribuir à raiz $F(d, r) = \{0\}$ quando $B(d, r) = \{0, 1\}$. No trabalho de Feijão e Meidanis [31], este fato é utilizado para demonstrar que não é possível ocorrer o caso em que, dadas duas adjacências conflitantes d e e, $F(d, v) = \{1\}$ e $F(e, v) = \{1\}$.

Dessa forma, o genoma do nó v é composto por todas as adjacências em que $F(d, v) = \{1\}$, sendo um genoma válido (sem adjacências conflitantes). O todo é minimizado através da minimização de suas partes e, como cada adjacência minimiza as trocas evolucionárias pelo algoritmo de Fitch, os genomas compostos por elas também minimizam o custo da árvore, conforme demonstrado por Feijão e Meidanis [31].

Problema Grande da Parcimônia

Dado um conjunto de genomas $G_1, G_2, ..., G_n$, definidos sob o mesmo conjunto de genes $\mathcal{G}$, encontre uma árvore T onde as folhas possuem uma correspondência de um-para-um com os genomas $G_1, G_2, ..., G_n$, e encontre um genoma ancestral $G_v = (\mathcal{G}, \Gamma_v)$ para cada nó interno v de T, tal que o tamanho da árvore T (a soma do peso de cada aresta, definido como a distância SCJ entre os genomas dos vértices incidentes) é minimizado.

Este problema é NP-difícil [31], pois é possível fazer uma redução polinomial do Problema da Árvore de Steiner para ele, conforme explicaremos a seguir.

O Problema da Árvore de Steiner consiste em, dado um grafo $H = (V, E)$, uma métrica que define o peso das arestas $s : E \to N$ e um subconjunto de vértices $S \subseteq V$, encontrar uma árvore de Steiner, isto é, uma sub-árvore de H que inclua todos os vértices de S, tal que a soma dos pesos das arestas seja minimizada. Consideramos o caso do Problema de Steiner no grafo hipercubo Q_n, com $S \subseteq V(Q_n)$. Todos os vértices de Q_n são representados como vetores binários de tamanho n, onde cada índice do vetor corresponde a uma dimensão do hipercubo, e dois vértices são conectados por uma aresta de peso 1 quando eles discordam em exatamente uma coordenada. Dessa forma, a distância $d(x, y)$ entre dois vértices x e y de Q_n é o número de coordenadas em que x e y diferem.

Dado um vetor $v = (v_1, v_2, ..., v_n)$ de tamanho n, considere o conjunto de genes $\mathcal{G} = \{g_1, g_2, ..., g_n\}$. Codificamos v como um genoma $G_v = (\mathcal{G}, \Pi_v)$ com as seguintes adjacências:

$$\Pi_v = \{g_h^i g_t^{i+1} : 1 \leq i \leq n \text{ e } v_i = 1\},$$

onde a soma $i+1$ torna-se 1 no caso de $n+1$. A Figura 5.4 mostra como os vértices de um grafo hipercubo são representados através de genomas no caso de $n=3$.

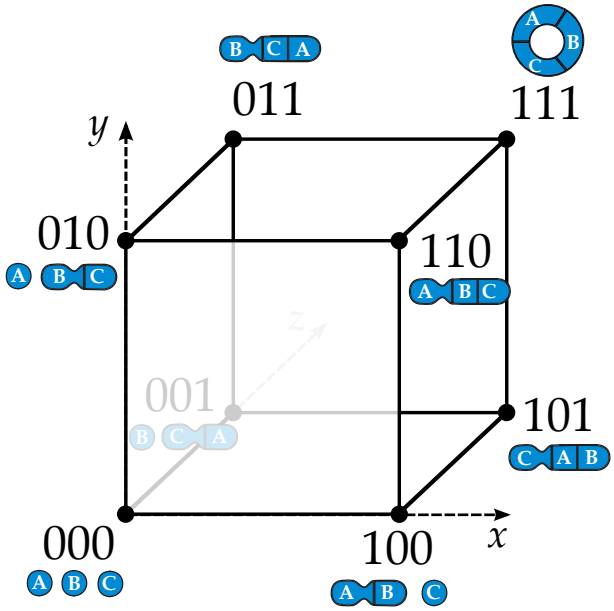

Figura 5.4: Representação dos vetores como genomas.

Dado um conjunto de vetores binários de tamanho n, criamos uma instância para o problema grande da parcimônia sob SCJ com o genoma G_v de cada vetor. Existe uma solução para este problema contendo somente adjacências presentes em pelo menos um dos genomas, pois qualquer outra adjacência pode ser seguramente removida sem aumentar a distância total. Tal solução pode ser representada através de vetores binários de tamanho n, onde cada genoma $G_i = (\mathcal{G}, \Pi_i)$ de um nó interno i terá um vetor único v tal que $\Pi_v = \Pi_i$.

5.3 Conclusões

Nesta seção, fizemos uma revisão bibliográfica das operações de rearranjo, dando maior ênfase àquelas que vêm sendo estudadas em problemas de rearranjo com mais que dois genomas. Também definimos a operação SCJ e como os problemas de rearranjo estudados neste trabalho são vistos sob esse modelo. No Capítulo 6 descrevemos os algoritmos implementados para resolvê-los e como os resultados inferidos foram analisados.

Capítulo 6

O Experimento

No Capítulo 5 vimos que, enquanto outras operações foram extensivamente estudadas com genomas reais e simulados, a operação SCJ ainda não havia sido vista em um contexto prático. Neste trabalho estamos interessados em avaliar experimentalmente a qualidade das árvores filogenéticas obtidas usando o SCJ como modelo de rearranjo. Para verificar quão boa uma árvore filogenética é, examinaremos separadamente os vários atributos que a constituem, tais como topologia, custo da árvore (número de eventos evolucionários nas arestas) e genomas ancestrais (genomas dos nós internos da árvore). As características mencionadas foram analisadas com as métricas explicadas na Seção 6.4. A Tabela 6.1 mostra brevemente quais métricas foram utilizadas para avaliar cada característica.

Tabela 6.1: Características avaliadas.

Critério	Característica	Métrica	Obtido a partir de
Acurácia	Topologia	Distância Robinson-Foulds (RF)	PGP
	Ancestrais	Adjacências em comum	PPP
		Adjacências falso-positivas	
		Adjacências falso-negativas	
Eficiência		Tempo (ms)	PGP / PPP

Para computar as métricas da Tabela 6.1, aplicamos o modelo SCJ em dois problemas de rearranjo de múltiplos genomas. O primeiro é o Problema Grande da Parcimônia (PGP), onde a entrada consiste de n genomas e tentamos encontrar a "melhor" árvore (custo mínimo) com os n genomas da entrada como folhas. Neste caso, estamos interessados em comparar as topologias das árvores obtidas com SCJ. Note que não é possível comparar os ancestrais com os resultados deste problema pois, como a topologia inferida

pode ser diferente, não há necessariamente uma correspondência entre os nós internos de soluções distintas. O outro problema considerado é o Problema Pequeno da Parcimônia (PPP), onde os genomas existentes e a topologia são dados, e precisamos encontrar os ancestrais que minimizem o custo da árvore. Neste caso, analisamos várias métricas relacionadas aos ancestrais inferidos.

Na Seção 5.2.3 explicamos que o PGP é NP-difícil com o SCJ. Para resolvê-lo, implementamos dois métodos: um exato, sendo uma abordagem *branch-and-bound*, e um método heurístico que adiciona um genoma por vez, ao qual chamamos de "Inclusão Passo-a-Passo". Para o PPP, implementamos uma adaptação do algoritmo de Fitch, sugerida por Feijão e Meidanis [31], que é executada em tempo polinomial em relação ao número de genomas e ao número de genes desses genomas (vide Seção 5.2.3). Esta abordagem é muito rápida e fornece uma solução ótima. Estes métodos são descritos em mais detalhes na Seção 6.3.

Além da acurácia para inferir cada uma das características, outro aspecto em que estamos interessados é a eficiência do SCJ, medida pelo tempo computacional demandado em cada experimento.

Depois de definir as métricas e métodos, definimos os dados de entrada que foram usados em cada método, mostrados na Tabela 6.2. Nestes experimentos usamos dados reais e simulados, dependendo da característica avaliada. Os dados reais foram usados para avaliar a topologia, através da comparação com as topologias inferidas por outros métodos da literatura. Escolhemos dois conjuntos de genomas bastante estudados: Campanulaceae, que é difícil por ser altamente rearranjado, e Protostômios, composto de 66 genomas, que é desafiador devido ao seu tamanho. Dados reais não foram usados na comparação de ancestrais, porque estes não são conhecidos nos conjuntos que selecionamos. Na verdade, não temos conhecimento de dados reais nos quais são conhecidos os ancestrais.

Tabela 6.2: Métodos implementados.

Problema	Método	Entrada
PPP	Fitch Adaptado (OPT)	Dados simulados
PGP	*Branch-and-bound* (OPT)	Dados simulados e reais
	Inclusão Passo-a-Passo (HEU)	Dados simulados e reais

Todos os métodos também foram testados usando dados simulados. Nas simulações, variamos os valores de diferentes parâmetros, tais como número de genomas da entrada, tamanho dos genomas, número de eventos de rearranjo em cada aresta da árvore e frequência de cada tipo de evento de rearranjo, avaliando a influência de cada parâmetro na qualidade da árvore inferida. A Seção 6.2.1 possui uma descrição mais detalhada de como

as simulações foram conduzidas. A Figura 6.1 mostra um diagrama que sumariza os experimentos.

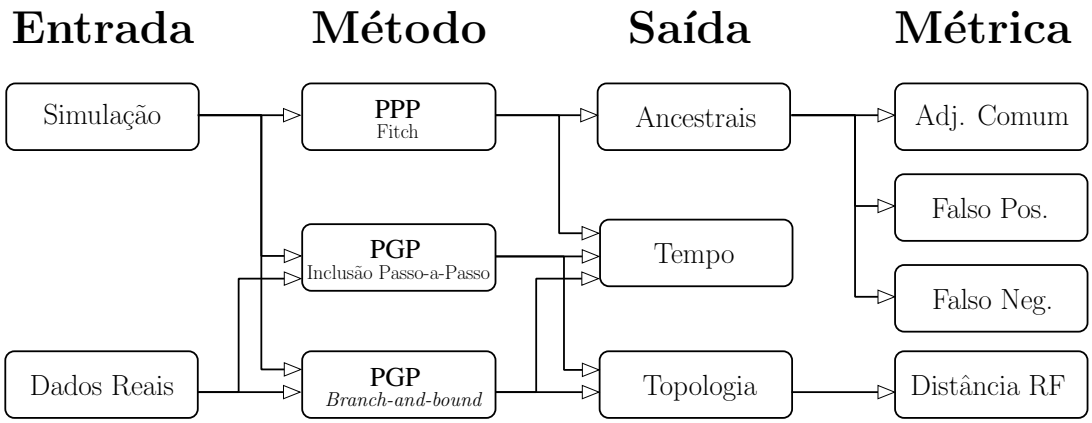

Figura 6.1: Diagrama dos experimentos.

6.1 Objetivo

O objetivo deste trabalho é verificar a capacidade da operação *Single-Cut-or-Join* no que diz respeito à reconstrução evolucionária, avaliando dois aspectos: (1) quão bem o SCJ reconstrói topologias evolucionárias?, e (2) quão bem o SCJ reconstrói genomas ancestrais? Respondemos a essas perguntas de duas formas:

- Através de simulações, avaliamos em que medida a operação SCJ é capaz de reconstruir histórias evolutivas a partir dos genomas dados;
- Através de aplicação a dados reais, avaliamos como os resultados de reconstrução de histórias evolutivas por SCJ comparam-se às árvores mais aceitas pela comunidade científica.

6.2 Dados utilizados

Nesta seção descrevemos como foram obtidos os dados de entrada dos métodos e quais aspectos podemos analisar através deles.

6.2.1 Simulações

Simulações computacionais são bastante úteis para verificar o quão boa é a inferência do SCJ em diferentes condições evolucionárias, já que é possível explorar exaustivamente o

impacto de diferentes parâmetros. Neste trabalho, simulamos a evolução por rearranjos para um conjunto de espécies, representadas por seus respectivos genomas. Com estas simulações, avaliamos como os seguintes parâmetros influenciam os resultados:

- *Número de eventos por aresta:* denota o número esperado de eventos evolucionários que ocorrem em uma aresta da árvore. Os valores são obtidos a partir de uma distribuição uniforme do conjunto $\{1, 2, .., e_{max}\}$, sendo e_{max} o número máximo de eventos. O valor de e_{max} é determinado em função do tamanho do genoma, como um percentual do número de genes. Por exemplo, no caso de $e_{max} = 0,2n$, se o genoma possuir 1000 genes, então $e_{max} = 200$. O percentual do número de genes é o valor variável deste parâmetro, assumindo valores entre 0 e 1;

- *Tamanho da árvore:* denota o número de folhas da árvore filogenética. O número de árvores possíveis é exponencial no número de folhas, aumentando a dificuldade em inferir a árvore;

- *Tamanho do genoma:* consiste de duas partes: o número de genes e o número de cromossomos. Escolhemos os valores para um ancestral hipotético da raiz da árvore, e propagamos os valores para os demais nós.

- *Distribuição de rearranjo:* define a frequência de cada tipo de evento de rearranjo, que permanece a mesma durante a evolução simulada. Apesar de existirem diversos tipos de eventos, consideramos somente inversões, transposições e translocações. Cada tipo de evento recebe uma probabilidade entre 0 e 1, e a soma das probabilidades é igual à 1.

Como não é possível testar todas as combinações de valores dos parâmetros, selecionamos um valor padrão para cada parâmetro, e estudamos variações em volta deste ponto. A Figura 6.2 exemplifica a estratégia adotada para o caso de três parâmetros.

A escolha dos valores de cada parâmetro corresponde a valores já utilizados na literatura, procurando refletir, sempre que possível, o que é observado em dados reais. A Tabela 6.3 apresenta alguns estudos com evolução simulada e os respectivos valores adotados. Através da tabela podemos observar variações nos valores de alguns parâmetros, principalmente no tamanho do genoma e da árvore. Entretanto, parece haver um consenso sobre a distribuição de rearranjo, com a frequência de inversões sendo predominante em relação aos demais tipos de eventos, por ser mais observada em dados reais. Concordando com estes trabalhos, atribuímos 90% de inversões e 10% de translocações como valor padrão para a distribuição de rearranjo.

Já a escolha do número de genes e de cromossomos reflete os tamanhos observados em genomas reais. Em nosso caso, os valores utilizados para o número de genes equivalem a

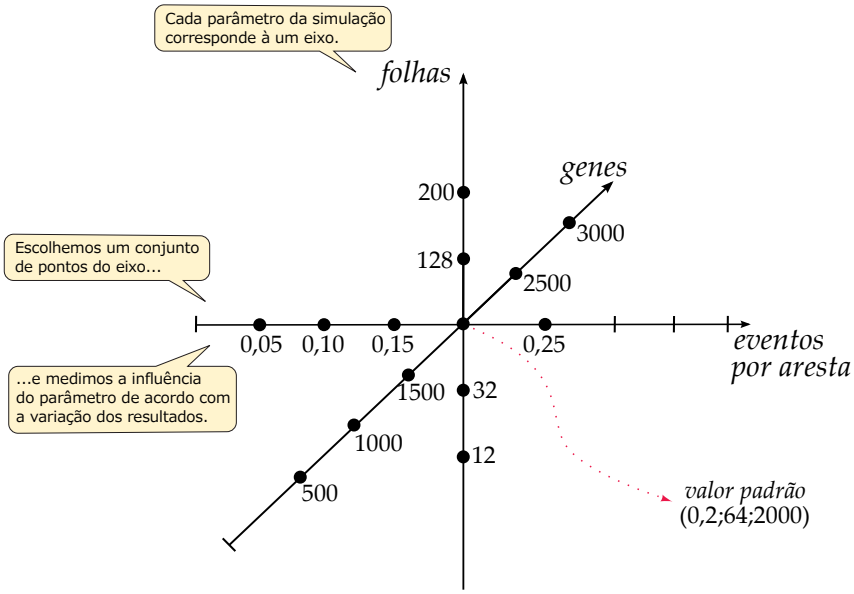

Figura 6.2: Parâmetros da simulação.

genomas de bactérias como, por exemplo, a bactéria associada a diversas infecções respiratórias (*Streptococcus pneumoniae*), tais como pneumonia e meningite, e a bactéria associada a infecções sexualmente transmissíveis (*Mycoplasma genitalium*). Não foi possível representar genomas mais complexos devido ao tempo computacional necessário para resolver os problemas com genomas maiores que 3000 genes. Entretanto, se observarmos a relação de genes por cromossomo, nosso experimento também se estende a organismos mais complexos, como o Panda (*Ailuropoda melanoleuca*), que possui 25000 genes e 21 cromossomos.

O parâmetro que define o número de eventos por aresta é bastante variável e cada trabalho determina uma forma diferente para seu cálculo. Nesse caso, definimos uma faixa de valores que contivesse os valores usados em outros estudos. O tamanho da árvore (número de folhas) é dependente, principalmente, da escalabilidade dos métodos em teste. Em nosso experimento, conseguimos realizar testes com árvores de até 200 folhas.

Os valores padrão são mostrados na Tabela 6.4. Note que, em alguns casos, o valor padrão depende do conjunto de dados.

Nos experimentos nós variamos apenas um parâmetro e fixamos os demais parâmetros com seus valores padrão — por exemplo, para analisar o número de genes (vide Figura 6.2), usamos todos os pontos destacados no eixo correspondente: 500, 1500, 2000, 2500 e 3000 genes. Para uma dada combinação, geramos 200 árvores simuladas, conforme explicaremos a seguir.

Tabela 6.3: Valores dos parâmetros (outros estudos). Abreviações utilizadas: CARs, *Contiguous Ancestral Regions*; DCJ, *Double-Cut-and-Join*; Inv., Inversões; op., operação; SoRT[2], *Sorting genomes and reconstructing phylogenetic trees by Reversals, generalized Transpositions and Translocations*; GASTS, *Generalized Adequate Subtree Tree Scoring*.

Estudo	Genoma		Árvore	Eventos por	Rearranjo			Sim. [3]
	Genes	Crom.	(folhas)	Aresta (e)	Inv.	Trp.	Trl.	
CARs [50]	6000	25	6	NI[1]	90	-	10^2	50
Experimento c/ DCJ e Inv. [69]	100	1	20	$r = \{2, .., 32\}$ $e = \{\frac{r}{2}, .., \frac{3r}{2}\}$	100	-	-	100
Métrica p/op. de rearranjo [2]	307	24	7	NI[1]	90	-	10	3
	603	23	5					
Métrica p/filogenias [68]	$\{100, 200\}$	1	$\{20, 40\}$	$r = \{4, 8, .., 32\}$ $e = \{\frac{r}{2}, \frac{r}{2}+1, .., \frac{3r}{2}\}$	100	-	-	100
SoRT[2] [43]	200	1	$\{10, 14, .., 46\}$	$e = \{1, .., 5\}$	1	0	1	100
					2	1	2	
					1	1	1	
GASTS [78]	2000	20	80	$e = \{31, .., 125\}$	100	0	0	10
					80	20	0	

[1] NI: valores não informados, apenas o modo como é calculado.
[2] Translocação recíproca: 5%; Fusão: 3,75%; Fissão: 1,25%.
[3] Número de simulações para cada combinação de parâmetros.

O primeiro passo da simulação é criar a topologia de uma árvore binária com o número de folhas especificado. Para isso, é necessário definir um modelo para as árvores, ou seja, uma distribuição de probabilidade que associa a cada topologia uma probabilidade de ocorrência, para todas as topologias com um dado número de folhas. A geração da topologia segue o modelo *beta-splitting*, proposto por Aldous [3]. Nesse modelo, é possível modificar a distribuição de probabilidade das topologias usando o parâmetro β, que varia de -2 a ∞. O parâmetro β ajusta a probabilidade de gerar árvores com um certo grau de balanceamento, com um β maior correspondendo a um maior balanceamento.

O modelo *beta-splitting* engloba outros modelos conhecidos, como o modelo de Yule [83] e o modelo PDA (do inglês, *Proportional to Different Arrangements*). No modelo de Yule, correspondente a $\beta = 0$, o tamanho médio da menor subárvore filha é $\frac{1}{4}n$, onde n é o tamanho da árvore pai. Já no modelo PDA, obtido com $\beta = -1.5$, todas as topologias possuem chances iguais de serem escolhidas, gerando árvores mais desbalanceadas do que no modelo de Yule. Entretanto, árvores filogenéticas reais costumam ser mais balanceadas

Tabela 6.4: Valores dos parâmetros (nosso estudo).

Parâmetro		Faixa de Valores	Valor Padrão
Árvore	Número de eventos por aresta	$\{0,05;\ 0,1;\ 0,15;\ 0,2;\ 0,25\}$	0,2
	Folhas	$\{12; 32; 64; 128; 200\}$	$\{64; 12\}$[1]
Genoma	Genes	$\{500; 1000; 1500; 2000; 2500; 3000\}$	2000
	Cromossomos	$\{1; 5; 10; 15; 20\}$	5
Distribuição de Rearranjo (Inversão, Transposição,Translocação)		{(0,2; 0,0; 0,8), (0,2; 0,1; 0,7), (0,4; 0,0; 0,6), (0,4; 0,1; 0,5), (0,6; 0,0; 0,4), (0,6; 0,1; 0,3), (0,8; 0,0; 0,2), (0,8; 0,1; 0,1), (0,9; 0,0; 0,1), (1,0; 0,0; 0,0), (0,0; 1,0; 0,0), (0,0; 0,0; 1,0)}	(0,9; 0,0; 0,1)

[1] O valor padrão do número de folhas depende do método: 64 genomas para o método de Fitch e para a heurística de Inclusão Passo-a-Passo, e somente 12 genomas para o *Branch-and-bound*, devido à quantidade proibitiva de tempo computacional necessária para conjuntos de dados maiores.

do que o modelo PDA e menos balanceadas do que o modelo de Yule. Aldous notou que, em árvores filogenéticas maiores, existe uma forte tendência do tamanho médio da menor subárvore filha ser muito menor do que $\frac{1}{4}n$, implicando um maior desbalanceamento. Por esta razão, usamos $\beta = -1$, o valor obtido empiricamente por Aldous que melhor representa o balanceamento observado em árvores filogenéticas reais.

Após definir a topologia, o ancestral hipotético da raiz é criado, com o número especificado de genes e cromossomos, sem duplicações. A partir da raiz da árvore, em direção às folhas, o simulador visita cada aresta, evoluindo os genomas com inversões, translocações e transposições, baseado na distribuição de rearranjo e no número de eventos por aresta definidos na entrada, até que todas as folhas sejam alcançadas.

Ao final do processo temos um conjunto de genomas com sua história evolucionária conhecida, usados para avaliar os ancestrais e a topologia da árvore reconstruída. A Tabela 6.5 mostra onde estes dados serão usados em nossa análise.

Tabela 6.5: Onde encontrar a análise do impacto dos parâmetros na acurácia dos métodos. Referências entre parênteses indicam que o gráfico correspondente não é mostrado, mas possui comportamento similar ao parentizado. Abreviações usadas nesta tabela: Crom., Cromossomos; Ev., Eventos; Distr. R., Distribuição de Rearranjo; Falso-pos., Falso-positivos

Resultado	Parâmetro				
	Genes	Crom.	Folhas	Ev. por Aresta	Distr. R.
Topologia (ótimo)	Fig.7.1a	(Fig.7.1a)	—	Fig.7.1e	Fig.7.1c
Topologia (heurística)	Fig.7.1b	(Fig.7.1b)	Fig.7.1g	Fig.7.1f	Fig.7.1d
Ancestrais (adjacências)	Fig.7.5a	Fig.7.5b	Fig.7.5e	Fig.7.5d	Fig.7.5c
Ancestrais (falso-pos.)	Fig.7.6a	Fig.7.6b	Fig.7.6e	Fig.7.6d	Fig.7.6c

6.2.2 Dados reais

Campanulaceae

Para comparar a operação SCJ com outras operações, aplicamos o SCJ ao conjunto de genomas da família Campanulaceae, composta por plantas com flores. Este conjunto foi criado por Cosner et al. [23] como um caso de teste para o método MPBE (do inglês, *Maximum Parsimony on Binary Encodings*). Este é um conjunto bastante estudado, consistindo de 13 genomas de cloroplasto, representativos de alguns dos gêneros da família, todos com um cromossomo circular contendo 105 marcadores. Note que, apesar de mencionarmos anteriormente que os genomas de cloroplasto são altamente conservados, os genomas do cloroplasto da família Campanulaceae são uma exceção, possuindo uma taxa de rearranjo maior em relação a outras famílias.

O conjunto criado por Cosner et al. contém genomas de 12 espécies representativas da família Campanulaceae, apresentadas na Figura 6.3, e também o *outgroup* Tobacco. Chamamos de *outgroup* o organismo que é relacionado aos demais, mas não pertence à mesma linhagem destes, com um ancestral comum mais distante do que o ancestral comum das espécies consideradas. O *outgroup* é utilizado como um marcador da raiz da árvore filogenética e, neste caso, o Tobacco foi escolhido como *outgroup* por ser um bom representante do genoma ancestral das angiospermas. A árvore inferida por MPBE é mostrada na Figura 7.3a.

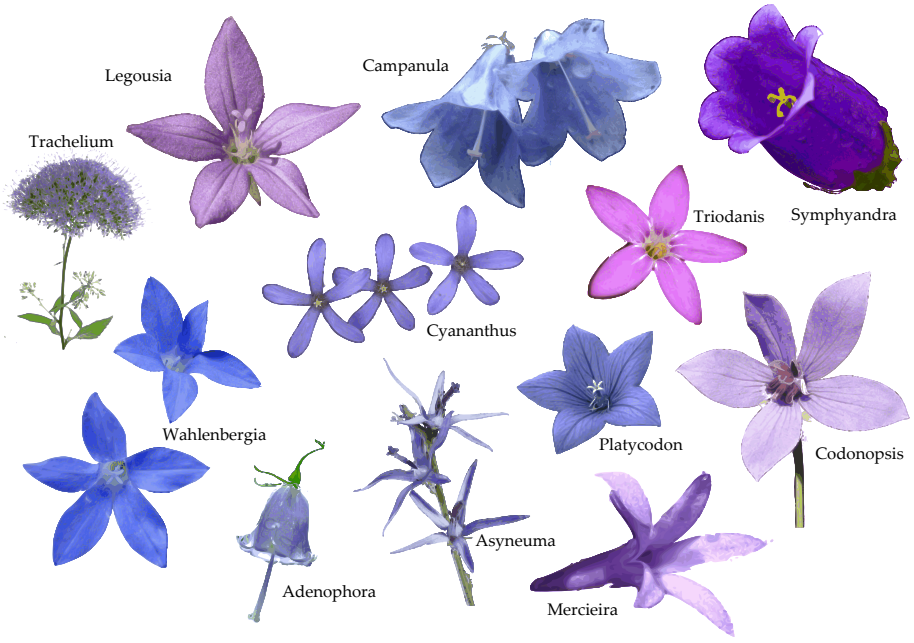

Figura 6.3: Conjunto de dados da família Campanulaceae.

O MPBE foi proposto como uma heurística para resolver o Problema de Rearranjos de Múltiplos Genomas por *breakpoints*. Além de comparar o SCJ com o MPBE, também utilizamos a topologia proposta por Bourque e Pevzner [16], muito usada em estudos de rearranjo. Eles encontraram uma árvore filogenética com 65 inversões usando a abordagem MGR (do inglês, *Multiple Genome Rearrangements*) e apresentada na Figura 7.3b. Mais tarde, a mesma topologia foi usada por Adam e Sankoff [1] para resolver o PPP sob o modelo DCJ, obtendo uma árvore com 64 DCJs. Em 2010, Kovac et al. [46] também usaram essa árvore para resolver o PPP sob DCJ, mas penalizando múltiplos cromossomos. Eles obtiveram várias topologias com 59 DCJs, onde todos os ancestrais possuem um único cromossomo.

Outra topologia considerada em nossa comparação foi apresentada por Xu e Moret [78], em 2011. Eles propuseram um método chamado GASTS (do inglês, *Generalized Adequate Subtree Tree Scoring*) para resolver o PPP, e encontraram 294 árvores com custo igual a 63 DCJs. Essas topologias são diferentes da topologia obtida por MGR. A partir delas, retraímos as arestas internas com custo 0, onde nenhuma operação DCJ foi feita, e construímos uma árvore consenso com o software PHYLIP 3.69 [32] (módulo CONSENSE). A árvore consenso do método GASTS é mostrada na Figura 7.3d.

A última árvore comparada foi obtida por Cosner et al. [24], usando 18 espécies de Campanulaceae e o *outgroup* Tobacco. Excluímos dessa árvore as 6 espécies que não estavam contidas em nosso conjunto. Cosner et al. conduziram um extenso estudo sobre a filogenia de Campanulaceae, usando dados de entrada complementares: a sequência

molecular do gene rbcL (gene que evolui lentamente, exclusivo das plantas, presente no DNA de seus cloroplastos) e da região ITS (região não codificante do DNA ribossômico que evolui rapidamente, apropriada para diferenciar espécies relacionadas ou variedades da mesma espécie), além de três matrizes de características baseadas na ordem dos genes. As três matrizes representam diferentes interpretações do conjunto de *breakpoints*: a primeira contém somente *breakpoints*; a segunda matriz modifica a primeira, substituindo os *breakpoints* por eventos evolucionários quando possível; e a terceira matriz modifica o conjunto de *breakpoints* da matriz 2, levando em conta hipóteses para eventos de rearranjo que ocorreram durante a evolução dos três principais clados. As matrizes 2 e 3 foram analisadas de duas formas: com e sem pesos para as características. No caso com peso, um maior peso foi dado as características que representam eventos. Cada um dos 7 conjuntos descritos foi usado como entrada de um método *branch-and-bound*, a fim de minimizar o número de trocas. Todas as árvores resultantes são apresentadas em [24], mas escolhemos somente a árvore obtida a partir da Matriz 2 com pesos (Figura 7.3c) pois, na nossa opinião, melhor representa os resultados obtidos no trabalho deles.

A Figura 7.3e mostra a árvore obtida a partir da minimização do número de operações SCJ. Usando um método *branch-and-bound*, encontramos todas as árvores ótimas (4 ao total). A árvore apresentada é a árvore consenso computada pela ferramenta PHYLIP.

Protostômios

Além do conjunto Campanulaceae, também usamos um conjunto de genomas maior, composto pelo DNA mitocondrial de 66 Protostômios, com 36 genes cada, apresentado na Figura 6.4. Os Protostômios são animais que, no desenvolvimento embrionário, formam primeiro a boca e somente depois o ânus. A origem embrionária da boca e do ânus são importantes características, constituindo a base morfológica para classificar os animais em dois grupos (Protostômios e Deuterostômios).

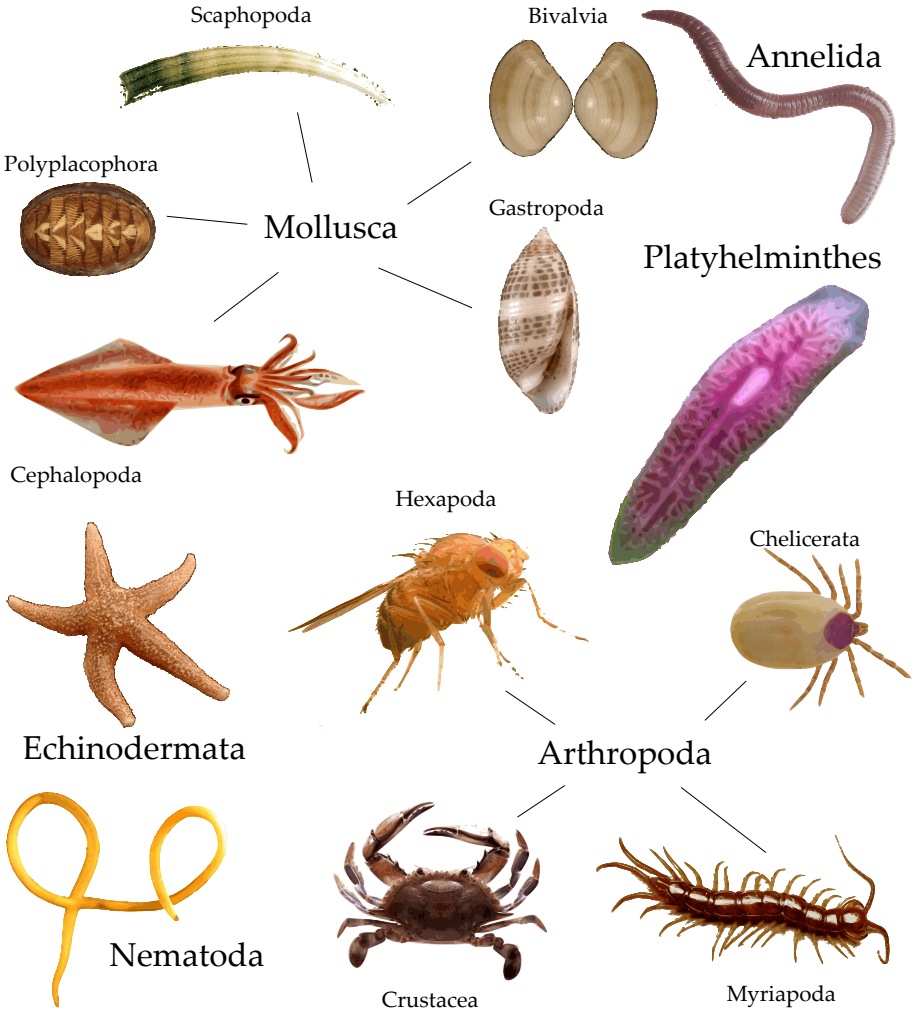

Figura 6.4: Conjunto de dados do grupo Protostômio.

O conjunto usado em nosso trabalho foi publicado por Fritzch et al. [37], como caso de teste para sua abordagem baseada em alinhamento de sequências. Após alinhar os genomas, eles obtiveram a árvore a partir de heurísticas que usam a parcimônia, tais como Inclusão Passo-a-Passo e Troca de Ramos. A árvore resultante é mostrada na Figura 7.4b. Eles usaram 112 genomas com 37 genes, mas o conjunto possui genomas duplicados, genes duplicados e indels (inserções e deleções de genes). Para obter um conjunto de genomas com o mesmo conteúdo gênico, sem duplicações, aplicamos um tratamento semelhante ao usado por Bernt et al. [12], que obtiveram um subconjunto menor com 62 genomas e 36 genes. A partir do conjunto apresentado por Fritzch, tratamos os genomas com conteúdo diferente de genes usando os seguintes passos:

1. Removemos todos os genomas duplicados, restando 78 genomas;

2. A partir dos 78 genomas, removemos todos os genomas com genes duplicados (6 ao total);

3. Removemos o gene ATP8. Antes da remoção, existiam 18 genomas com conteúdo diferente de genes, e somente 6 após a remoção;

4. Removemos os 6 genomas que permaneceram com conteúdo gênico diferente.

Após o tratamento, executamos a heurística de Inclusão Passo-a-Passo 100 vezes, armazenando a árvore com o menor número de operações SCJ, mostrada na Figura 7.4c. Nós também usamos em nossa comparação a árvore taxonômica do NCBI (do inglês, *National Center for Biotechnology Information*), mostrada na Figura 7.4a. Este grupo tem sido estudado com muitos outros métodos. A árvore filogenética dos Protostômios baseada em rearranjos foi primeiro apresentada por Blanchette et al. [15], mas usando um conjunto de dados diferente.

6.3 Métodos

6.3.1 Problema Pequeno da Parcimônia

Para resolver o PPP, implementamos o algoritmo polinomial proposto por Feijão e Meidanis [31]. Este método é análogo ao algoritmo de Fitch [36], onde cada adjacência d é uma característica, e os possíveis estados são presença, denotado por "1", ou ausência, denotado por "0", de uma característica específica. Como explicado na Seção 5.2.3, este algoritmo é composto por duas etapas. A primeira etapa, que ocorre das folhas em direção à raiz, é mostrada no Algoritmo 3. Após obter os conjuntos $B(d, v)$ de cada nó v da árvore T, o algoritmo executa a segunda etapa, da raiz para as folhas, definindo o estado da característica no nó v. A segunda etapa é apresentada no Algoritmo 4. Estas duas etapas compõem o algoritmo de Fitch, mostrado no Algoritmo 2, que ao final minimiza o número de trocas evolucionárias da adjacência d na árvore T. Note que a Linha 5 do Algoritmo 4 assegura que o genoma da raiz r não terá a adjacência d no caso de $B(d, r) = \{0, 1\}$. Ao repetir esse procedimento para cada adjacência d inferimos os genomas dos nós ancestrais da árvore T que minimizam seu custo (Algoritmo 1).

Algoritmo 1: Problema da Pequena Parcimônia por SCJ

Entrada: (1) Genomas $G_1, G_2, \ldots, G_n$. (2) Árvore T com raiz r onde os genomas $G_1, G_2, \ldots, G_n$ correspondem às folhas.

Saída: Árvore filogenética T com os genomas ancestrais definidos, tal que o custo de T é minimizado.

```
1  início
2      para todo nó v de T faça
3          se v é folha então
4              Adj ← Adj ∪ G_v
5          senão
6              inicializar G_v como um genoma sem adjacências
7          fim se
8      fim para todo
9      para cada adjacência d de Adj faça
10         T ← Fitch(T, d)
11     fim para cada
12     devolva a árvore T
13 fim
```

Algoritmo 2: Fitch

Entrada: (1) Árvore T com raiz r, onde os genomas $G_1, G_2, \ldots, G_n$ correspondem às folhas e já estão definidos. (2) Adjacência d.

Saída: Árvore filogenética T minimizando as trocas de estado da adjacência d.

```
1  início
2      Bottom-Up(r, d)
3      Top-Down(r, d)
4      devolva a árvore T
5  fim
```

Algoritmo 3: Bottom-Up

Entrada: (1) Nó v pertencente à árvore filogenética T. Os genomas $G_1, G_2, \ldots, G_n$ correspondem às folhas de T. (2) Adjacência d.

Resultado: A chamada Bottom-Up(d, v) calcula recursivamente o conjunto $B(d, v)$ para o nó v e seus descendentes.

```
1  início
2      se o nó v é folha então
3          se G_v contém a adjacência d então  B(d, v) ← {1}
4          senão  B(d, v) ← {0}
5      senão
6          Bottom-Up(e_v, d)                    /* e_v é o filho esquerdo de v */
7          Bottom-Up(d_v, d)                    /* d_v é o filho direito de v */
8          se  B(d, e_v) ∩ B(d, d_v) ≠ ∅ então
9              B(d, v) ← B(d, e_v) ∩ B(d, d_v)
10         senão
11             B(d, v) ← B(d, e_v) ∪ B(d, d_v)
12         fim se
13     fim se
14     armazenar B(d, v)
15 fim
```

Algoritmo 4: Top-Down

Entrada: (1) Nó v pertencente à árvore filogenética T. Cada nó u de T possui um genoma G_u e um conjunto $B(d, u)$. (2) Adjacência d.

Resultado: A chamada Top-Down(d, v) calcula recursivamente o conjunto $F(d, v)$ para o nó v e seus descendentes.

```
1  início
2      se B(d, v) ≠ {0, 1} então
3          F(d, v) ← B(d, v)
4      senão
5          se v é raiz então  F(d, v) ← {0}
6          senão  F(d, v) ← F(d, p_v) ;          /* p_v é o nó pai de v */
7      fim se
8      se F(d, v) = {1} então
9          incluir a adjacência d ao genoma G_v
10     fim se
11     armazenar F(d, v)
12     Top-Down(e_v, d)                          /* e_v é o filho esquerdo de v */
13     Top-Down(d_v, d)                          /* d_v é o filho direito de v */
14 fim
```

A vantagem do SCJ, em relação às outras operações, é que o PPP é fácil, e resolvido em tempo polinomial com o algoritmo apresentado, enquanto em outros métodos — como DCJ e Hannenhalli-Pevzner (HP) — é NP-difícil mesmo no caso em que somente três genomas são considerados (o caso especial do PPP chamado de Problema da Mediana) [73].

6.3.2 Problema Grande da Parcimônia

Para resolver o PGP, nós usaremos dois métodos: um exato, com uma abordagem *branch-and-bound*, e um método heurístico guloso, chamado de Inclusão Passo-a-Passo.

A inclusão passo-a-passo é similar a outras heurísticas usadas anteriormente para este problema [35, pp. 216]. Uma ideia do algoritmo é mostrada no Algoritmo 5. O algoritmo inicia resolvendo o Problema da Mediana com três genomas de entrada e, em cada passo, um genoma arbitrário ainda não incluído na árvore é adicionado, resolvendo o Problema da Mediana para cada aresta. O genoma é incluído na árvore na aresta que implica no menor custo da árvore. A heurística termina quando todos os genomas tiverem sido incluídos. Nós repetimos a heurística um mínimo de 100 vezes e selecionamos a melhor árvore (que possui o menor número de operações SCJ).

Algoritmo 5: Heurística de Inclusão Passo-a-Passo

Entrada: Genomas $G_1, G_2, \ldots, G_n$.
Saída: Árvore filogenética de custo mínimo com as folhas $G_1, G_2, \ldots, G_n$.

```
1  início
2      resolver o problema da mediana para G_1, G_2, G_3, denotando por T a árvore
       resultante
3      para l ← 4 até n faça
4          para cada aresta {u, v} em T com os genomas G_u, G_v faça
5              computar a mediana G_M^uv de G_u, G_v, G_l
6              C(u, v) ← d(G_u, G_M^uv) + d(G_v, G_M^uv) + d(G_l, G_M^uv) − d(G_u, G_v)
7          fim para cada
8          C(u_0, v_0) ← min{C(u, v) | {u, v} ∈ E(T)}
9          remover a aresta {u_0, v_0} de T
10         incluir os vértices G_M^uv, G_l à T
11         incluir as arestas {G_M^uv, u_0}, {G_M^uv, v_0}, {G_M^uv, G_l} em T
12     fim para
13     devolva a árvore T
14 fim
```

Quando o tamanho do problema permite a busca por uma solução exata (tipicamente 12 genomas ou menos), nós também usamos um algoritmo *branch-and-bound*. Este tipo

de abordagem tem sido usada há muito tempo na reconstrução filogenética [34]. Como na heurística, nós construímos uma árvore inicialmente resolvendo o Problema da Mediana com três genomas de entrada e extensivamente construímos o espaço completo de todas as árvores filogenéticas possíveis. A árvore de pesquisa do algoritmo *branch-and-bound*, para o caso de cinco genomas, é mostrada na Figura 6.5. O número de topologias possíveis cresce rapidamente conforme o número de genomas nas folhas aumenta. Note que a heurística explicada anteriormente explora este espaço de forma gulosa, ramificando, em cada nível da árvore de pesquisa, somente o nó que possui a solução parcial de menor custo.

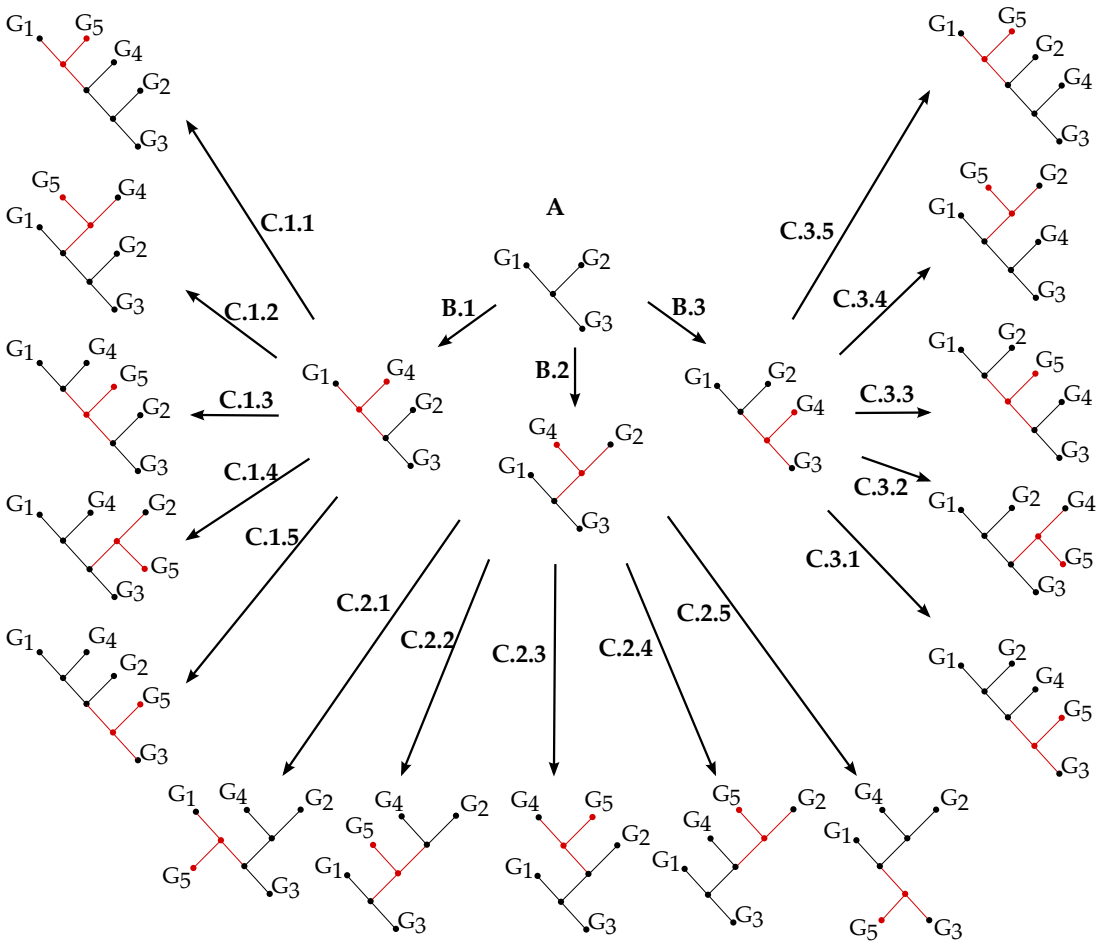

Figura 6.5: Árvore de pesquisa do *branch-and-bound*.

Na árvore de pesquisa, o algoritmo executa as operações *branching* e *bounding* como explicaremos a seguir. A operação de *branching* constrói um filho de um nó baseado em sua solução parcial $P = (V^P, E^P)$ e na lista L de genomas não incluídos em P. Um genoma arbitrário l pertencente a L e uma aresta (i, j) de E^P são escolhidas, e o algoritmo

computa a mediana M dos genomas i, j, e l, definindo uma nova árvore pela remoção da aresta (i, j) e incluindo em seu lugar as arestas (i, M), (M, j), e (l, M). Os filhos de um nó da árvore de pesquisa são obtidos aplicando o passo anterior a cada aresta (i, j) de E^P, gerando todas as possíveis árvores que podem ser obtidas a partir da inclusão de l em uma solução parcial P.

A operação de *bounding* inicialmente computa um limitante superior usando a heurística *Neighbor-Join* [63], e melhora este limitante usando a melhor árvore encontrada. Para o limitante inferior de um nó, o algoritmo usa sua solução parcial $P = (V^P, E^P)$ e a lista L dos genomas não incluídos em P, construindo uma nova árvore para cada genoma como descrito na operação de *branching*. A partir destas árvores, o algoritmo encontra o menor custo que cada genoma de L adiciona à árvore, e usa como limitante inferior o maior deles, que representa o menor custo para incluir o genoma mais distante. Note que o cálculo do custo da árvore é muito rápido, porque o SCJ possui um algoritmo de tempo polinomial para esta tarefa.

Apesar de custoso, encontrar as árvores ótimas é útil, por exemplo, quando queremos conhecer se o SCJ infere uma topologia realística.

6.4 Métricas

A *métrica* é uma forma quantitativa de avaliar uma determinada característica. O cálculo da métrica utiliza uma ou mais *medidas* obtidas durante o experimento. Após computar os valores das métricas, utilizamos *indicadores*, que dão contexto a uma métrica ou a uma combinação de métricas, servindo como base para comparação.

Usando as métricas descritas a seguir, avaliamos três características, que constituem os principais aspectos da reconstrução filogenética: a acurácia da topologia, a acurácia dos ancestrais e a eficiência dos métodos. A *acurácia* relaciona os resultados inferidos aos resultados esperados, definindo o quanto estes aproximam-se. A *eficiência* está relacionada ao esforço computacional necessário para obter os resultados. A Tabela 6.6 mostra a relação entre as medidas, as métricas e os indicadores usados nos experimentos para avaliar as características mencionadas.

6.4.1 Acurácia da topologia

A acurácia da topologia reconstruída representa o quanto a topologia inferida aproxima-se da topologia real. Uma forma quantitativa de medir esta característica é através da distância Robinson-Foulds [62], também conhecida como distância Split. Esta métrica tem sido bastante utilizada para comparar a topologia de duas árvores definidas sobre o mesmo conjunto de folhas.

Tabela 6.6: Características avaliadas, indicadores que avaliam as características, métricas que compõem os indicadores e medidas usadas no cálculo das métricas.

Característica	Indicadores	Métricas	Medidas
Acurácia da Topologia	Muito, Razoavelmente e Pouco acurado	Distância RF	*Splits*
Acurácia dos Ancestrais	Muito, Razoavelmente e Pouco acurado	Percentual de Reconstrução e Falso-Positivos	Adjacências
Eficiência do Método	—	Tempo de processamento	Tempo de processamento

Dada uma árvore, a remoção de uma aresta particiona as folhas em dois subconjuntos disjuntos, formando o que é chamado de *split*. A distância Split é a diferença em *splits* entre a árvore original e a reconstruída, dividido pelo número total de splits. O valor da distância Split varia entre 0 e 1, onde 0 representa o cenário de "melhor caso", onde todos os splits são iguais. Nós usamos o software TOPD [59] para computar a distância Split.

Os indicadores adotados foram propostos por Moret et al. [54]. A topologia reconstruída é considerada muito acurada se a distância RF for abaixo de 0,05 e, pouco acurada, se a distância RF estiver acima de 0,25. No estudo de Nakhleh et al. [55] também foram usados indicadores semelhantes, variando apenas o caso de árvores pouco acuradas, em que foram consideradas as distâncias RF acima de 0,20, ao invés de 0,25.

6.4.2 Acurácia dos ancestrais

Nós comparamos os ancestrais considerando as adjacências de seus genes. Dados dois genomas, o original e o inferido, classificamos suas adjacências em três possíveis classes (vide Figura 6.6):

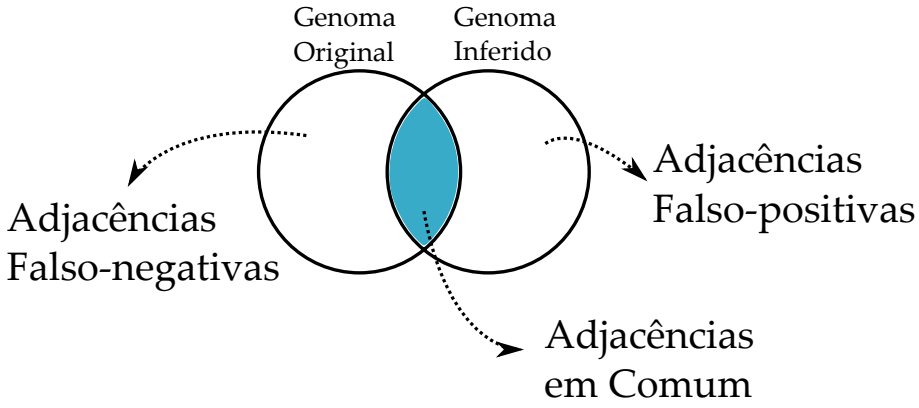

Figura 6.6: Métricas para avaliar ancestrais.

- *Adjacências falso-negativas:* adjacências presentes no genoma original, mas não no genomas inferido;
- *Adjacências em comum:* presentes em ambos genomas, elas são usadas para calcular a percentagem de reconstrução do genoma original;
- *Adjacências falso-positivas:* adjacências incorretas, presentes somente no genoma inferido.

A percentagem de reconstrução é definida como o número de adjacências em comum dividido pelo número total de adjacências do genoma original. Esta métrica é relacionada à métrica CARs (do inglês, *Contiguous Ancestral Regions*) [50], mas não é idêntica. Os CARs são diferentes porque eles também consideram o relacionamento entre as adjacências, tentando modelar blocos ortólogos. Baseado nas adjacências ancestrais preditas, os genes são conectados em CARs se o relacionamento de seu predecessor e sucessor é consistente com o genoma original. Outro modo de analisar a percentagem de reconstrução é obter a percentagem do genoma original coberto pelos CARs.

Sejam A e B os conjuntos de adjacências dos genomas original e inferido, respectivamente. A métrica de falso-positivos é definida como $\frac{|B-A|}{|B|}$ e, similarmente, a métrica de falso-negativos é determinada por $\frac{|A-B|}{|A|}$. O conjunto de falso-negativos é o complemento do conjunto de adjacências em comum no genoma original e, por esta razão, os gráficos de falso-negativos não são mostrados em nossa análise.

Note que é inviável mostrar as métricas para todos os nós. Para ajudar a compreender a comparação de resultados dos genomas ancestrais, decidimos correlacioná-los com a posição do nó na árvore. Isso parece particularmente apropriado para o percentual de reconstrução, pois toda informação conhecida está nas folhas e, portanto, o esforço de reconstrução necessário é maior conforme nos movemos em direção à raiz. Consideramos várias alternativas para uma métrica posicional. As seguintes medidas foram consideradas: altura (caminho mais longo de um nó para uma folha), profundidade (caminho mais longo da raiz para o nó), distância média para a folha e distância mínima para a folha. Nós terminamos escolhendo a altura, porque os desvios padrão das outras medidas foram significantemente maiores.

Para avaliar as métricas de reconstrução de ancestrais, propusemos uma composição dos resultados das métricas de falso-positivos e percentagem de reconstrução. A definição dos indicadores, a partir da combinação dos valores das métricas, é apresentada na Tabela 6.7. As faixas de valores estabelecidas em cada uma das métricas são inspiradas na proposta de Moret et al. [54], explicada na subseção anterior. Em relação aos falso-positivos, o resultado é muito acurado se, no máximo, 5% das adjacências do ancestral inferido são falso-positivas e, pouco acurado, se mais de 25% das adjacências são

falso-positivas. Complementar a este indicador, também verificamos a percentagem do ancestral coberto, sendo 95% ou mais uma alta cobertura e, abaixo de 75%, uma baixa cobertura.

Tabela 6.7: Indicadores da acurácia dos ancestrais.

Percentagem de Reconstrução	Falso-Positivos		
	Menor que 5%	Entre 5% e 25%	Maior que 25%
Menor que 75%	Pouco acurado	Pouco acurado	Pouco acurado
Entre 75% e 95%	Razoavelmente acurado	Razoavelmente acurado	Pouco acurado
Maior que 95%	Muito acurado	Razoavelmente acurado	Pouco acurado

6.4.3 Eficiência dos métodos

A eficiência é avaliada tomando como medida os tempos de processamento em todos os testes executados. Estes tempos serão analisados na Seção 7.3. Neste caso não foi possível estabelecer indicadores de eficiência para os métodos, já que o tempo de processamento é influenciado por vários fatores, como o ambiente computacional usado e a implementação do método.

6.5 Conclusões

Neste capítulo detalhamos todas as etapas do procedimento executado para avaliar a operação SCJ. Descrevemos a seleção dos dados de entrada, os métodos implementados e as métricas usadas para mensurar a qualidade dos resultados. Os resultados obtidos com este experimento são descritos no Capítulo 7.

Capítulo 7

Resultados

Neste capítulo discutiremos os resultados obtidos a partir do experimento descrito no Capítulo 6. A Seção 7.1 analisa a acurácia do SCJ em inferir topologias, tanto com dados simulados como com dados reais; a Seção 7.2 verifica a acurácia de outro atributo da árvore, os ancestrais, usando dados simulados; por fim, na Seção 7.3 falamos sobre a eficiência dos métodos nos experimentos mencionados anteriormente.

7.1 Acurácia da Topologia

Nesta seção apresentamos os resultados da acurácia em inferir a topologia em diferentes cenários evolutivos, usando como medidas quantitativas as métricas descritas na Seção 6.4. Começamos pela discussão da influência dos parâmetros da simulação nos resultados. Também verificamos a acurácia dos métodos: como a abordagem heurística aproxima-se do resultado ótimo? E como o resultado ótimo aproxima-se da árvore original? Para os dados onde nós não conhecemos o ótimo (todos aqueles com mais que 12 espécies), comparamos a heurística com a árvore original diretamente. No final desta seção, consideramos os resultados em dados reais, comparando a topologia inferida com as topologias propostas em outros estudos.

Abaixo seguem somente os resultados do Problema Grande da Parcimônia (PGP). O Problema Pequeno da Parcimônia (PPP) não é considerado pois a topologia é dada neste caso. Usamos os métodos descritos na Seção 6.3 (*branch-and-bound* e heurística de Inclusão Passo-a-Passo) para resolver o problema.

7.1.1 Dados simulados

Nesta seção, usamos como entrada os conjuntos de genomas simulados descritos na Seção 6.2.1.

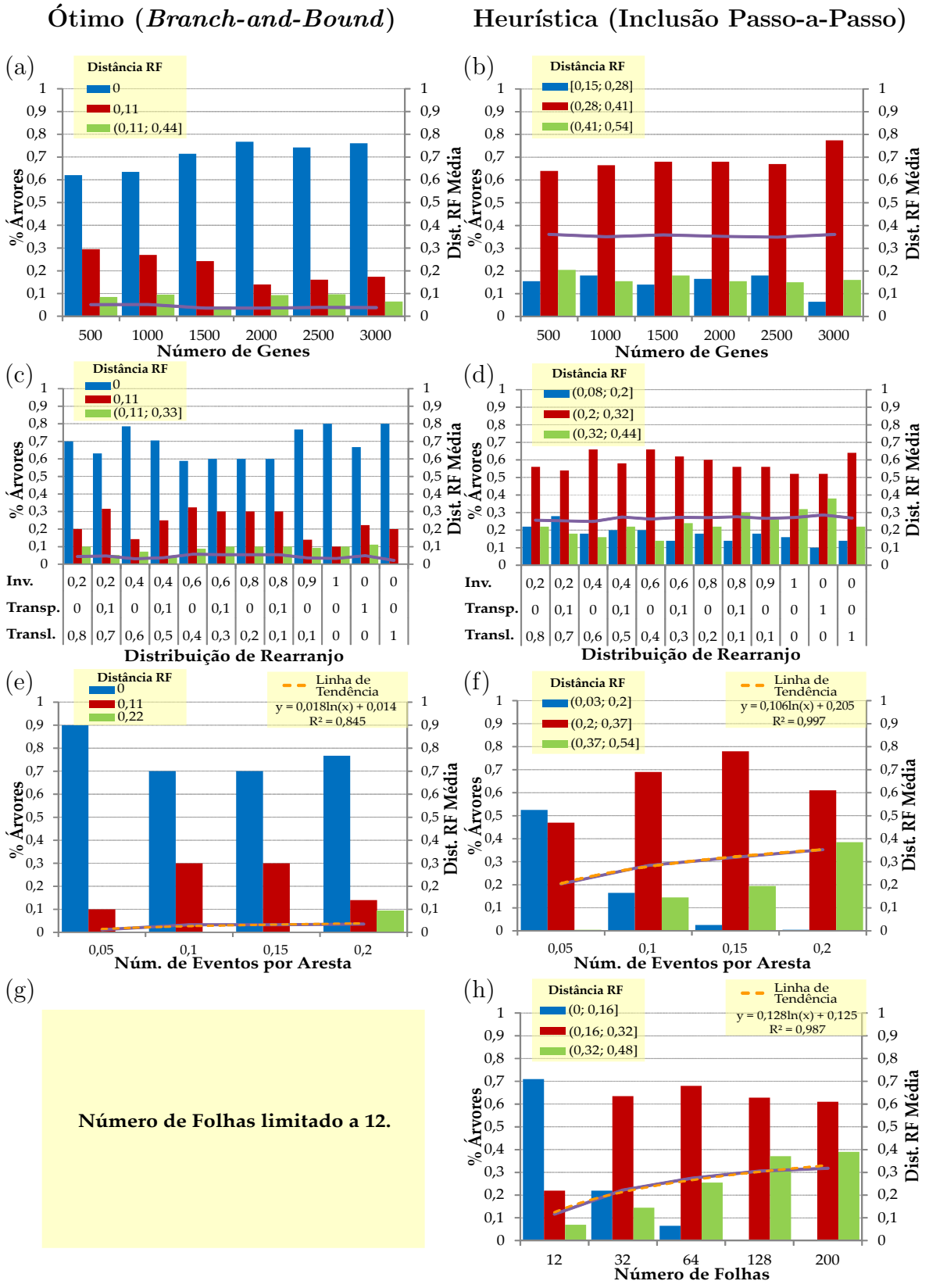

Figura 7.1: Influência dos parâmetros na topologia inferida, para os métodos ótimo e heurístico (PGP). Ao analisar um parâmetro, todos os outros mantêm seus valores padrão (vide Tabela 6.4).

Os gráficos da Figura 7.1 mostram como o desvio da topologia inferida (eixo y) varia quando os valores de um parâmetro na entrada do problema são modificados (eixo x). Cada ponto do gráfico tem um histograma com a distribuição da distância RF, onde uma barra representa a frequência que certa faixa de valores da distância RF foi observada (veja a legenda de cada gráfico).

Este tipo de gráfico foi o mais adequado, entre os testados, para exibir a variação dos resultados. Os gráficos *boxplot*, que usam como medida de tendência a mediana, não foram representativos devido ao fato de, em alguns dos pontos, mais de 50% dos valores possuírem distância RF igual a 0 (quando a árvore inferida é exatamente igual à árvore original). Já os gráficos que usam o desvio padrão como medida de dispersão não foram representativos por apresentarem um desvio padrão muito alto, devido ao fato da distribuição dos valores, em alguns pontos, não ser uma distribuição normal. Nos gráficos apresentados, a linha que exibe a média apenas sumariza a variação, que pode ser vista em mais detalhes a partir dos histogramas, que mostram como a distribuição dos dados é modificada ao variar os pontos.

Podemos notar que, nas Figuras 7.1a e 7.1b, a média da distância RF não é significativamente afetada pelo número de genes, pois a distância RF permanece próxima à uma constante em ambos os métodos (exato e heurístico). Em termos da distribuição, a operação SCJ leva à topologia da árvore original em mais de 60% das simulações (distância RF igual à zero). Nos histogramas da Figura 7.1a, está claro que a inferência do método exato tem uma menor probabilidade de erros, e que esta probabilidade decresce à medida que o número de genes aumenta. Um comportamento similar é visto quando o número de cromossomos aumenta (dados não mostrados).

Com a heurística, os casos de teste envolvendo árvores maiores (64 folhas) apresentaram resultados localizados na fronteira dos indicadores razoavelmente acurado e pouco acurado. A heurística obteve melhores resultados em alguns cenários específicos, como árvores pequenas (abaixo de 32 folhas) ou com um menor número de eventos de rearranjo por aresta (abaixo de 5% do tamanho do genoma), onde a distância RF permaneceu abaixo de 0,2, sendo considerada razoavelmente acurada. Entretanto, na maioria dos casos, a média da distância RF está entre 0,30 e 0,35, consideravelmente maior que no método exato (somente 0,05). Note que o método exato foi testado apenas em árvores com 12 folhas.

Como a qualidade da topologia inferida varia quando usamos diferentes distribuições de rearranjo? Os gráficos das Figuras 7.1c e 7.1d mostram um resultado interessante: quando a frequência de cada evento aumenta ou diminui em relação aos outros eventos, a acurácia da inferência, assim como nos resultados anteriores, não é significativamente afetada.

Provavelmente isto ocorre pois a mesma distribuição de rearranjo é usada em todas

as arestas da árvore, sendo um termo comum. Mais especificamente, seja $n_{A,B}$ o número de eventos de rearranjo entre dois genomas A e B, e seja $F = [f_{inv}, f_{trp}, f_{trl}]$ o vetor com as frequências relativas de inversões, transposições e translocações. O número esperado de eventos de cada tipo entre A e B é $n_{A,B} * [f_{inv}, f_{trp}, f_{trl}]$.

Se o número de operações SCJ necessárias em cada evento é $S = [s_{inv}, s_{trp}, s_{trl}]$, então a distância SCJ esperada entre A e B será

$$d_{A,B} = n_{A,B}(f_{inv}s_{inv} + f_{trp}s_{trp} + f_{trl}s_{trl}).$$

Isto é, $d_{A,B}$ é aproximadamente proporcional à $n_{A,B}$. Estas contas rápidas parecem implicar que o *número* de eventos por aresta é muito mais importante que os tipos de evento utilizados (distribuição de rearranjo). Note que esta ideia aplica-se a outros tipos de operação de rearranjo em problemas de múltiplos genomas, apesar dos estudos experimentais adotarem uma frequência de inversões predominante. Isso é útil pois, como as frequências de rearranjo variam conforme o contexto, é possível determinar a acurácia da operação em diferentes contextos a partir de testes sob um contexto específico, desde que o número de eventos seja parecido.

Para avaliar o impacto do número de eventos por aresta, nós realizamos experimentos cujos resultados são mostrados nas Figuras 7.1e e 7.1f. Na Figura 7.1f, que apresenta os resultados da heurística para 64 folhas, observamos um decrescimento da acurácia quando há um número maior de eventos. Um resultado semelhante é observado ao variar o número de folhas (Figura 7.1h). O número de folhas não é variado no método exato devido ao alto tempo computacional necessário para computar árvores com mais de 12 folhas.

Resumindo as análises realizadas, parece que o tamanho da árvore, determinado pelo número de folhas, e o número de eventos por aresta são os únicos parâmetros que impactam a inferência por SCJ: ambos parâmetros aumentam a distância RF. Ajustamos um modelo linear em x (o parâmetro) e $\log y$, onde y é a distância RF média, e encontramos boas correlações em todos os casos (dados das Figuras 7.1e, 7.1f e 7.1h).

Com os gráficos apresentados anteriormente, vimos que a inferência por SCJ possui um comportamento muito acurado no método exato, com distância RF por volta de 0,05, mesmo usando um alto número de eventos evolucionários (até 20% do número de genes em cada aresta). A heurística possui resultados menos acurados, mas é capaz de resolver instâncias muito maiores.

7.1.2 Dados reais: Campanulaceae

Em conjuntos de dados reais, executamos os métodos SCJ, exato e heurístico, que resolvem o PGP nos genomas do cloroplasto de Campanulaceae, comparando quantitativamente e visualmente o resultado obtido com as árvores apresentadas por Cosner et al. [24], e também com as árvores inferidas por MGR, MPBE e GASTS. A Figura 7.3 mostra as cinco topologias usadas nesta comparação, desenhadas com a ferramente online iTOL (do inglês, *Interactive Tree Of Life*) [47].

Note que não é possível aplicar diretamente a distância RF nesta análise, pois as árvores comparadas precisam ser binárias e, dentre as árvores usadas, somente a árvore MGR é completamente resolvida. Dessa forma, dadas duas árvores, sendo pelo menos uma delas não resolvida, antes de aplicarmos a distância RF, resolvemos seus relacionamentos de forma que minimize a diferença entre os *splits* das duas árvores.

Complementando a análise quantitativa, a comparação visual das árvores, neste caso, permite avaliar a qualidade dos relacionamentos evolutivos considerando a literatura. Por exemplo, quando os relacionamentos evolutivos de uma dada espécie divergem em dois métodos, é considerado correto o que é mais próximo da hipótese mais aceita atualmente. A análise visual também permite identificar padrões entre os relacionamentos que podem ser úteis em estudos de sistemática das espécies.

Encontramos todas as árvores ótimas (4 ao total, todas com o custo de 150 SCJs) sob o modelo SCJ. As quatro árvores são muito semelhantes, exceto por dois relacionamentos evolucionários que divergiram:

1. Relacionamento entre Platycodon e o ancestral de Codonopsis e Cyananthus: em duas árvores, Platycodon está mais distante de Codonopsis e Cyananthus, como mostrado na Figura 7.2a enquanto, nas outras duas, o relacionamento é igual ao da árvore apresentada na Figura 7.2b.

2. Relacionamento entre Symphyandra, Trachelium, e o ancestral de Campanula e Adenophora: em duas árvores, o ancestral de Campanula e Adenophora está mais próximo de Symphyandra do que de Trachelium (Figura 7.2a), enquanto em outras duas, a situação oposta aparece, como na topologia da Figura 7.2c.

As quatro árvores ótimas são apresentadas na Figura 7.2. A árvore SCJ da Figura 7.3e, usada na comparação, é a árvore consenso destas quatro árvores.

Para a comparação quantitativa, calculamos a distância RF entre a árvore consenso do SCJ e as árvores dos outros quatro métodos, resolvendo os nós de grau maior que 2 de forma a minimizar a diferença entre os *splits* das duas árvores. Em todos os casos, a distância RF obtida é igual a 0, o que mostra que a árvore inferida por SCJ concorda com as demais árvores.

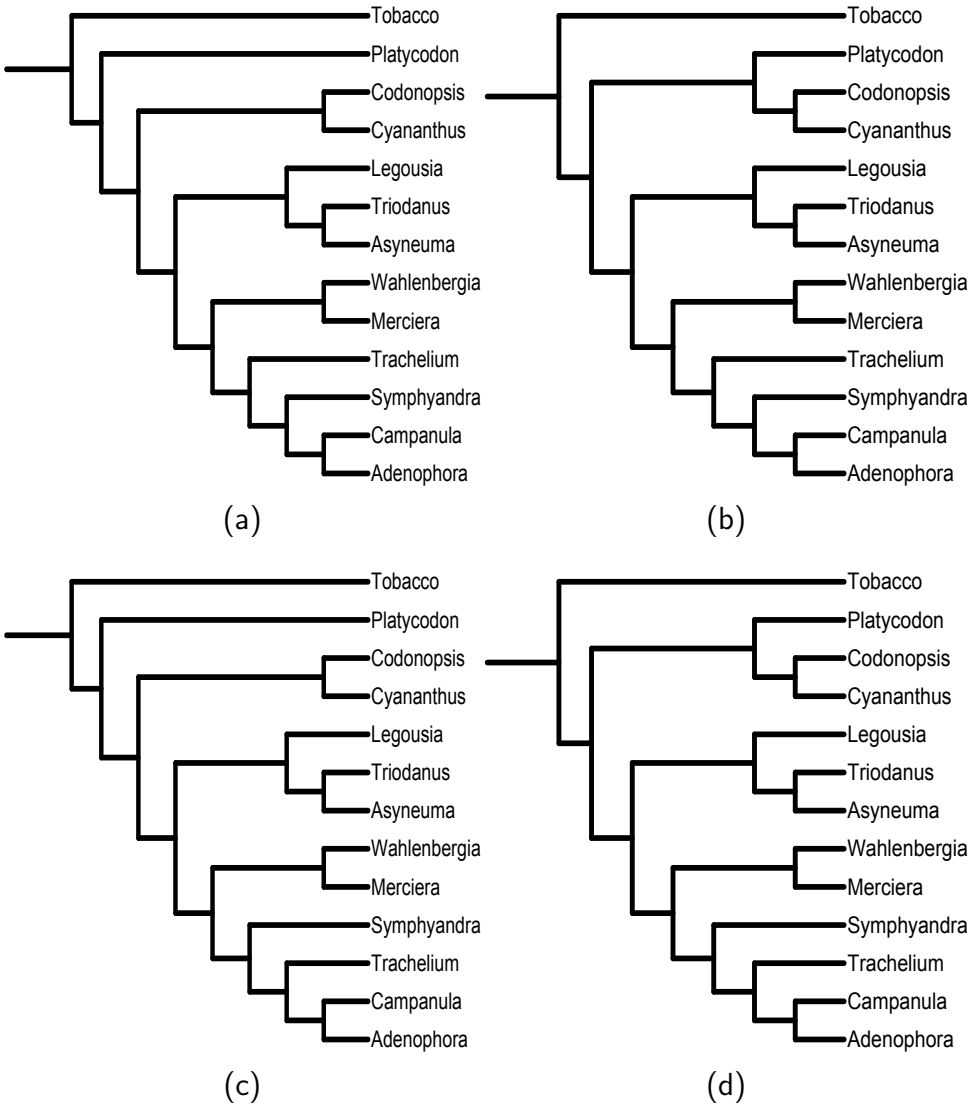

Figura 7.2: Topologias ótimas de Campanulaceae, com o custo de 150 SCJs. As árvores diferem em apenas dois relacionamentos evolutivos: entre Platycodon e o ancestral de Codonopsis e Cyananthus; e entre Symphyandra, Trachelium, e o ancestral de Campanula e Adenophora.

Ao compararmos visualmente as árvores mencionadas, que não são completamente resolvidas em pontos diferentes, é possível identificar alguns relacionamentos evolutivos em comum, listados a seguir e bem representados na árvore da Figura 7.3c:

- **Grupos pequenos:**
  - **Grupo 1:** Campanula e Adenophora são gêneros irmãos em todos os resultados;
  - **Grupo 2:** Merciera e Wahlenbergia são gêneros irmãos em todos os resultados;
  - **Grupo 3:** Legousia, Asyneuma, e Triodanis formam uma subárvore em todos os resultados;
- **Grupos médios:**
  - **Grupo 4:** Platycodon, Codonopsis e Cyananthus estão sempre agrupados juntos, o que corrobora com as evidências deles formarem um clado basal dentro da família, apresentadas por Cosner et al. [24];
  - **Grupo 5:** formado pelos grupos 1, 2 e 3, e também pelos gêneros Trachelium e Symphyandra;
- **Grupo grande:**
  - **Grupo 6:** formado pelos grupos 4 e 5, representa a família Campanulaceae.

Os resultados aqui apresentados mostram que o SCJ é capaz de fornecer árvores reconstruídas de alta qualidade a partir de dados reais.

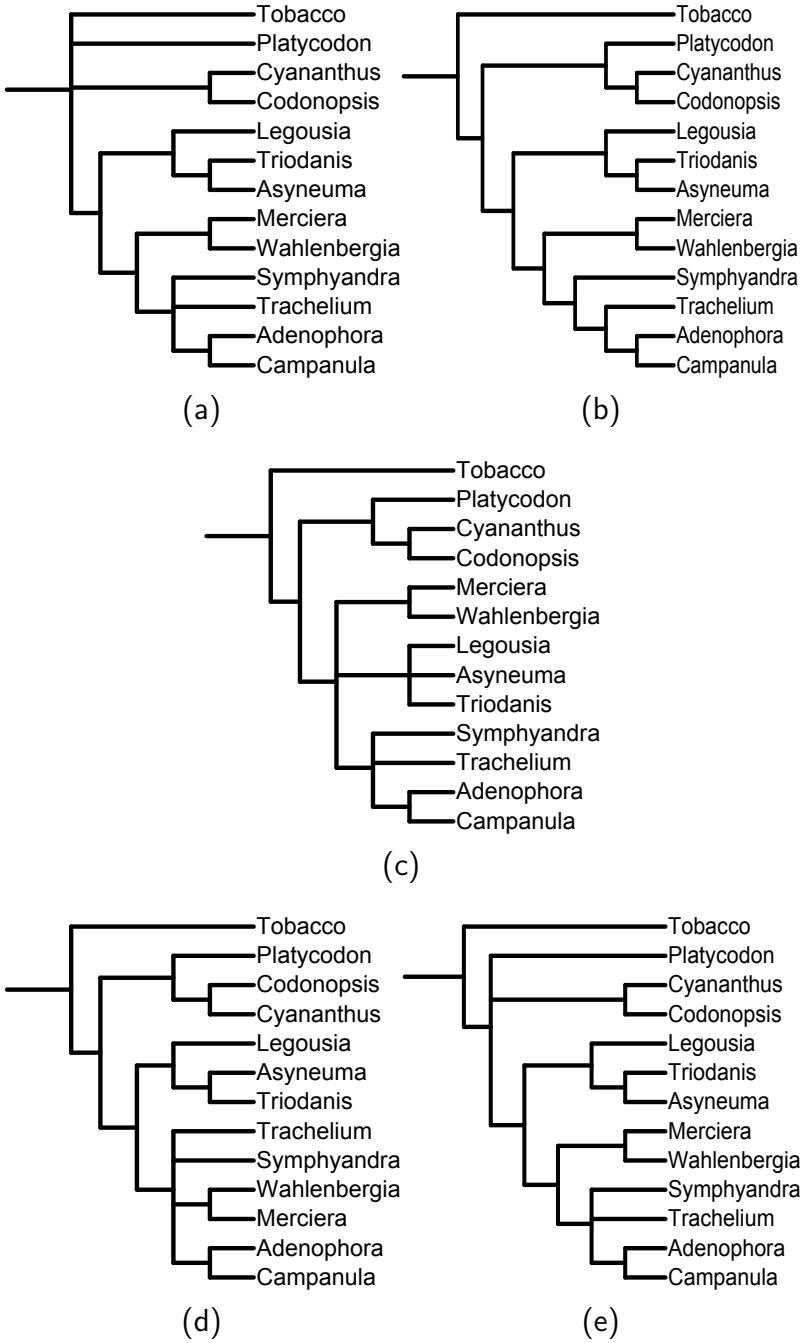

Figura 7.3: Topologia Campanulaceae reconstruída por diversos métodos: (a) MPBE [23]; (b) MGR [16]; (c) Cosner et al. [24]; (d) GASTS [78]; (e) SCJ. Todos os métodos, exceto o de Cosner et al., usaram como entrada os genomas do cloroplasto de 12 espécies de Campanulaceae e do *outgroup* Tobacco. Cosner et al. usaram um conjunto de dados maior, baseado na ordem dos genes e na sequência molecular, composto de 18 espécies de Campanulaceae e do *outgroup* Tobacco. Tanto SCJ como o MPBE são medidas baseadas em *breakpoints*, diferente do método MGR, que é baseado em inversões. O GASTS resolve o PPP minimizando o número de DCJs. A análise de Cosner et al. foi baseada na parcimônia, usando *breakpoints*, eventos definidos por *breakpoints* e as sequências moleculares da região ITS e do gene rbcL. Em geral, a topologia obtida por SCJ concorda com as demais, revelando algumas tendências, como descrito no texto principal.

7.1.3 Dados reais: Protostômios

No caso do conjunto de genomas dos Protostômios, executamos o método heurístico 100 vezes, armazenando as árvores com o menor número total de operações SCJ (3 ao total, cada uma com 915 SCJs). Então, a árvore consenso foi computada por PHYLIP, e o resultado é apresentado na Figura 7.4c. Comparamo-lo com a topologia do NCBI (Figura 7.4a) e com os resultados de Fritzsch et al. [37] (Figura 7.4b).

Nossa abordagem classificou corretamente as espécies dentro dos filos Arthropoda, Nematoda, Echinodermata, Annelida, e Plathelminthes. Mollusca apresentou um problema, que é justificado pela observação de que alguns Mollusca possuem altas frequências de rearranjo de genes nos genomas mitocondriais, comparado a outros metazoários [37], tornando a inferência da topologia baseada somente na ordem dos genes uma tarefa difícil. Nossa abordagem foi, entretanto, capaz de classificar mais espécies dentros dos filos, pois a árvore apresentada por Fritzsch et al. não resolveu completamente o filo Nematoda (veja Figura 7.4b).

Usando a ordem dos genes ou sequências moleculares, nem Fritzch et al. nem SCJ classificaram as espécies dos subfilos de Arthropoda (Myriapoda, Chelicerata, Crustacea e Hexapoda), discordando com as classificações baseadas em características morfológicas (dados não mostrados). A taxonomia de Arthropoda não é completamente resolvida, devido à diversidade e tamanho deste filo.

Embora o método SCJ tenha resolvido corretamente as espécies dentro dos filos, os relacionamentos evolucionários entre estes filos estão um pouco diferentes dos obtidos por Fritzch et al. Comparando estas árvores com a árvore taxonômica do NCBI, reconhecemos três grupos bem definidos (veja Figura 7.4a):

- **Grupo 1:** Echinodermata;
- **Grupo 2:** Platyhelminthes e Nematoda;
- **Grupo 3:** Arthropoda, Annelida e Mollusca.

A árvore obtida por SCJ concorda nos Grupos 1 e 3, mas não no Grupo 2. As espécies do filo Mollusca aparecem em várias partes da árvore. Na árvore de Fritzch et al., os filos Annelida e Nematoda são monofiléticos, e os Platyhelminthes estão mais próximos do subfilo Gastropoda (Mollusca). Nós também observamos que nossa árvore está mais próxima da árvore taxonômica do NCBI do que a árvore de Fritzch et al.

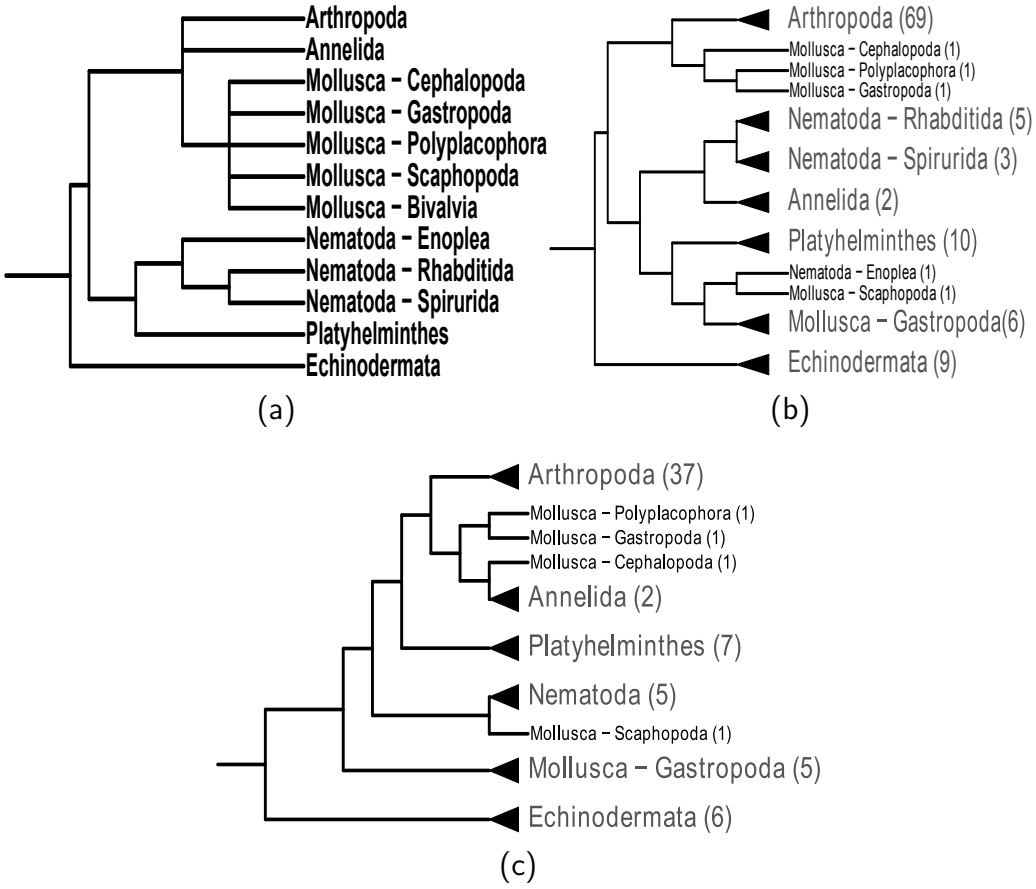

Figura 7.4: Topologia dos Protostômios, reconstruída por diversos métodos: (a) NCBI; (b) Fritzsch et al. [37]; (c) SCJ. Note que o método SCJ agrupa corretamente todos os filos, exceto o filo Mollusca, que é altamente rearranjado e também foi um problema na inferência de Fritzsch et al. A árvore obtida por SCJ está mais próxima da árvore obtida a partir da taxonomia do NCBI, com os relacionamentos mais próximos entre Annelida e Arthropoda.

7.2 Acurácia dos Ancestrais

Anteriormente analisamos como a topologia inferida desvia dos resultados esperados. Nesta seção nosso foco é a avaliação dos genomas ancestrais inferidos a partir de dados simulados. Para garantir que os ancestrais inferidos tenham uma contrapartida na árvore original, as topologias das árvores inferidas e original devem ser iguais. Portanto estudamos somente o Problema Pequeno da Parcimônia, onde a topologia é dada. No Problema Grande da Parcimônia não existe garantia de que a topologia inferida será a mesma da árvore original.

7.2.1 Adjacências em comum

Para mensurar a qualidade da reconstrução, começamos esta análise com uma medida simples, que é a quantidade relativa de adjacências entre genes consecutivos que foi coberta pelo método SCJ. Abaixo, chamamos esta quantidade de *percentagem de reconstrução*.

Os gráficos da Figura 7.5 mostram a correlação entre a percentagem de reconstrução do genoma original (eixo y) e o parâmetro analisado (eixo x). Cada ponto do gráfico possui um histograma, que relaciona a altura do nó da árvore inferida com a percentagem de reconstrução. Uma barra do histograma corresponde à percentagem de reconstrução média considerando somente os nós que estão em uma certa faixa de alturas, especificada na legenda de cada gráfico.

Observamos nestes gráficos uma percentagem de reconstrução menor nos nós que tem altura maior na árvore (mais próximos à raiz, mais distantes das folhas). A variação do tamanho do genoma (número de genes e cromossomos), mostrado nas Figuras 7.5a e 7.5b, exibe um comportamento similar à variação da distribuição de rearranjo, explicada a seguir. Na Figura 7.5c podemos ver que, independentemente da variação da distribuição de rearranjo, o decrescimento na percentagem de reconstrução é similar em todos os histogramas, novamente mostrando que a distribuição de rearranjo tende a impactar de forma não significativa a acurácia dos métodos SCJ.

Em todos os histogramas desta seção, os nós são agrupados em três barras, de acordo com a sua altura. Dessa forma, a quantidade de nós considerada para o cálculo de cada barra diminui à medida que as alturas tornam-se maiores: se a árvore é balanceada, o número de nós de uma altura i é aproximadamente a soma de todos os nós com uma altura maior; se a árvore é desbalanceada, a raiz alcançará uma altura maior. Por exemplo, na Figura 7.5a podemos notar que a altura da árvore chega a 25, enquanto uma árvore balanceada com 64 folhas possui, no máximo, altura 6. Com a diminuição da quantidade de nós, há um aumento da dispersão dos valores em relação ao valor médio representado pelas barras.

Como na análise de topologia, o parâmetro que afeta a inferência mais fortemente é o número de eventos na aresta. Na Figura 7.5d, os histogramas que representam um maior número de eventos possuem um queda mais acentuada. Note que, nos experimentos que rearranjam até 20% do tamanho do genoma em cada aresta, todos os genomas tem pelo menos metade do genoma reconstruído, independente da altura, o que é um resultado muito positivo.

O gráfico da Figura 7.5e representa diferentes números de folhas. Como as árvores com mais folhas tem um número maior de nós, elas podem alcançar alturas maiores e, por essa razão, a altura máxima dos histogramas varia em cada ponto. O número de folhas afeta como a percentagem de reconstrução decresce, como no caso da Figura 7.5c.

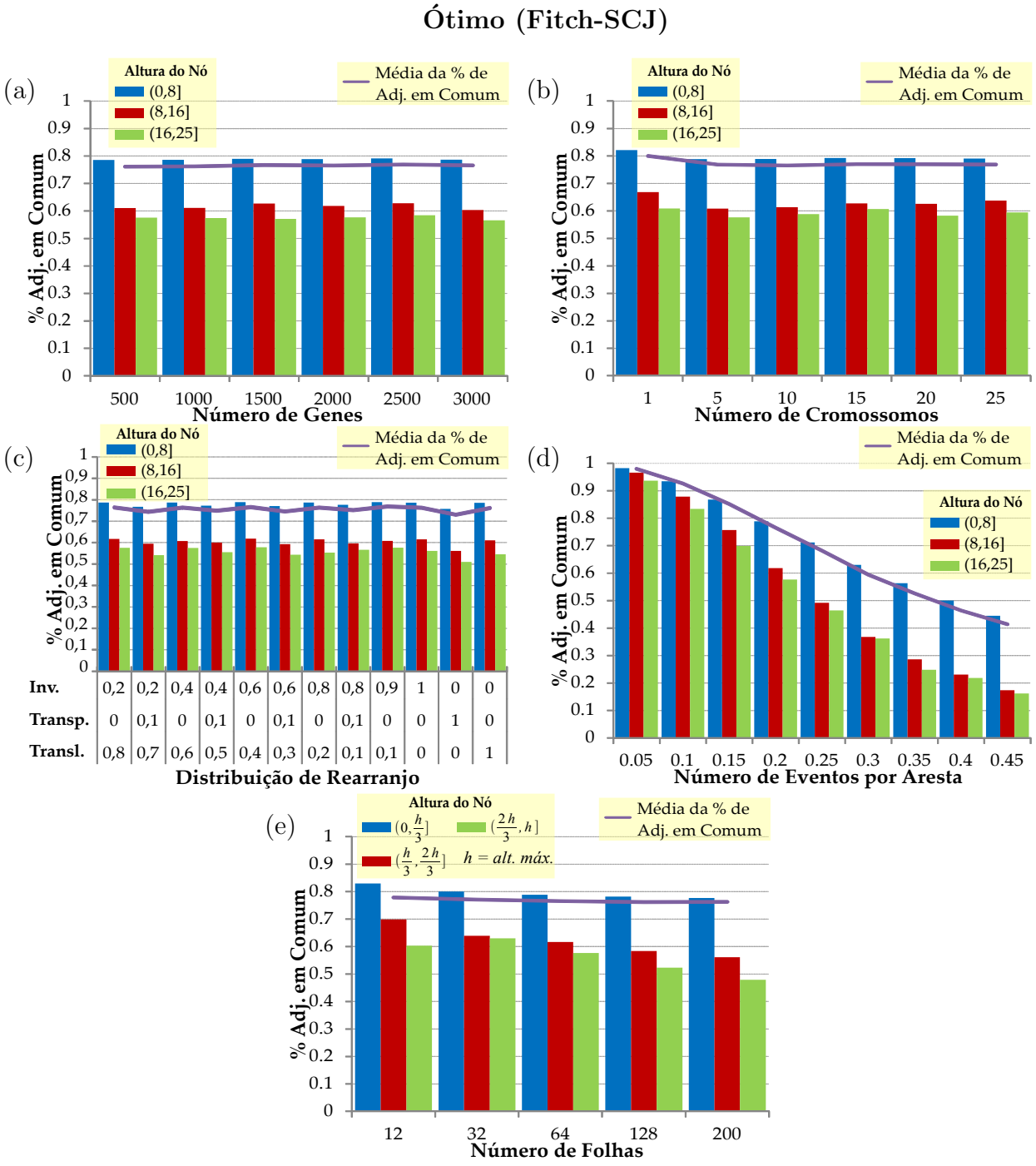

Figura 7.5: Influência dos parâmetros na reconstrução de genomas ancestrais, considerando somente a percentagem de adjacências corretamente reconstruídas com respeito ao número total de adjacências no genoma original. Em todos os casos, a percentagem do genoma reconstruído decresce quando a altura do nó aumenta. O número de nós analisados também decresce, e ocorre uma maior dispersão dos dados.

Usando os indicadores descritos na Seção 6.4.2, observamos que os nós com altura entre 1 e $\frac{h}{3}$, sendo h a altura máxima de um nó, possuem uma cobertura razoável, de aproximadamente 77%, enquanto os demais nós possuem uma baixa cobertura. Como esta faixa de valores corresponde a mais de 85% dos nós ancestrais analisados, podemos concluir que, de modo geral, os nós ancestrais são cobertos razoavelmente.

7.2.2 Adjacências Falso-Positivas

Considerando todas as adjacências inferidas, a percentagem de reconstrução define quantas adjacências são corretas, presentes em ambos os genomas. Agora vamos olhar as adjacências inferidas que não estão presentes no genoma real, as quais chamamos de *falso-positivas*.

A Figura 7.6a mostra a correlação entre o número de falso-positivos e o número de genes. Estes histogramas classificam o número de genomas ancestrais de acordo com o número de adjacências incorretas. Notamos que 90% ou mais dos genomas ancestrais têm todas as suas adjacências reconstruídas corretamente, mesmo quando incrementamos o número de genes. Este mesmo comportamento é observado ao variar o número de folhas (Figura 7.6e), indicando que não existe influência significativa destes parâmetros no número de falso-positivos.

Nós também identificamos um segundo tipo de comportamento, mostrado nas Figuras 7.6b e 7.6d: quando o número de cromossomos ou o número de eventos por aresta aumenta, o número de falso-positivos também cresce. Genomas ancestrais sem falso-positivos predominam em todos os histogramas. No caso do número de eventos, nós notamos que este comportamento ocorre devido a um aumento da probabilidade do genoma original não ser a solução mais parcimoniosa, isto é, quando mais eventos evolucionários ocorrem, uma dada região do genoma pode ser similar em espécies de diferentes ancestrais (como homoplasias). Portanto, é possível explicar a evolução com um número menor de eventos, se os ancestrais comuns destes ramos são também similares.

Existe também um terceiro comportamento observado na Figura 7.6c: ao usar diferentes distribuições de rearranjo, encontramos uma variação no número de falso-positivos, que estão distribuídos de diversos modos. Quando existe somente um tipo de evento durante a evolução, o número de falso-positivos é significantemente maior do que em cenários evolutivos com rearranjos mais diversificados.

Em todos os casos, o número de falso-positivos é baixíssimo, indicando uma inferência muito acurada.

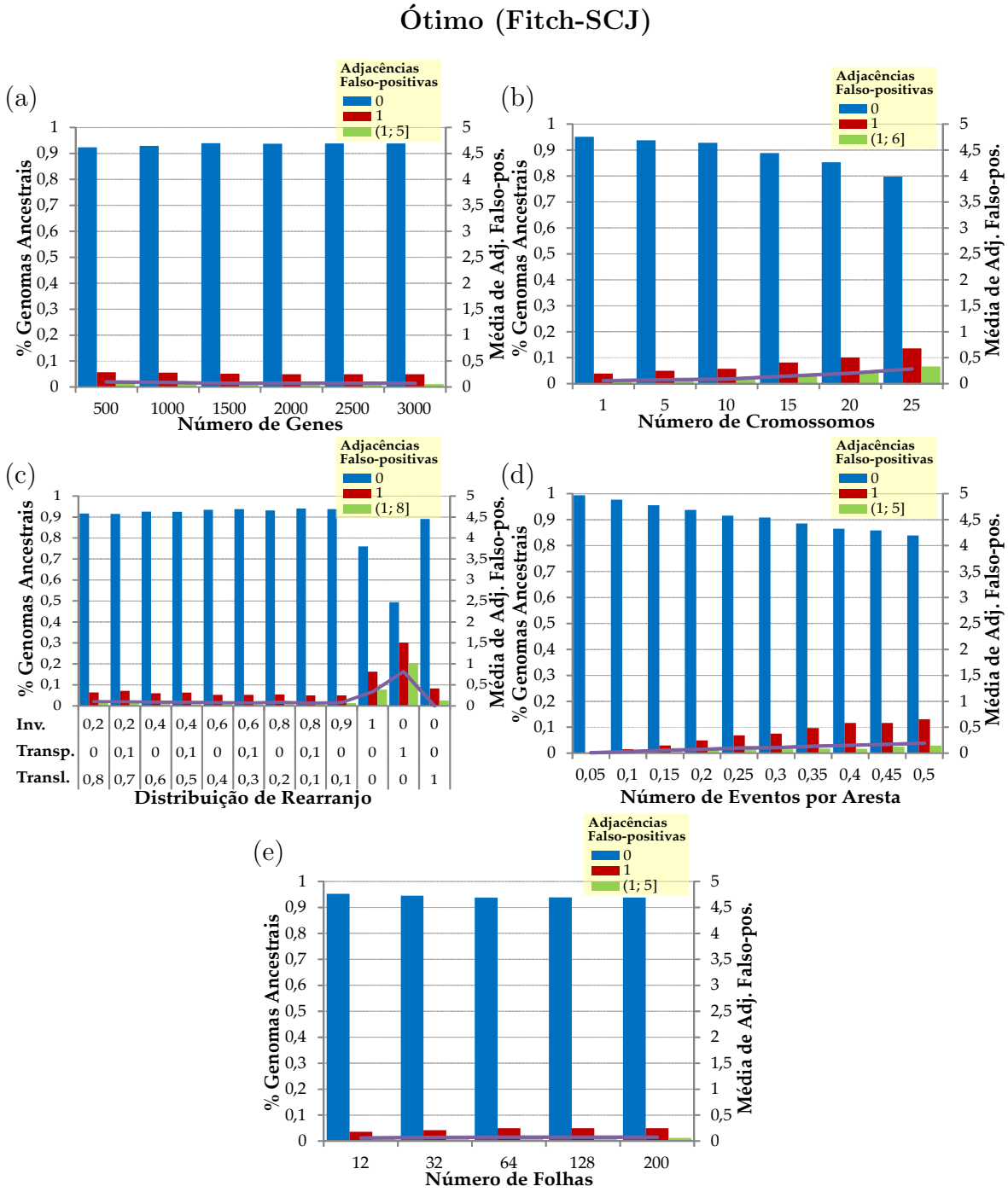

Figura 7.6: Influência dos parâmetros na reconstrução de genomas ancestrais, considerando somente adjacências falso-positivas. A métrica de falso-positivos é definida como o número de adjacências reconstruídas não presentes no genoma original dividido pelo número total de adjacências reconstruídas, variando entre 0 e 1. Como os resultados de falso-positivos foram muito baixos em todos os casos testados, mostramos a quantidade de falso-positivos em valores absolutos para melhor visualizarmos a variação dos valores. Os parâmetros analisados são mostrados no eixo x. O eixo y à esquerda mede os histogramas de distribuição dos falso-positivos e, à direita, a média de adjacências falso-positivas. Em quase todos os casos, o SCJ infere corretamente as adjacências, mantendo a quantidade de falso-positivos próxima a zero.

7.3 Eficiência dos métodos

Em relação ao tempo de execução, distribuímos os experimentos em diferentes ambientes, de acordo com o esforço computacional necessário para resolvê-los.

Dada uma topologia, o tempo que o SCJ leva para reconstruir os ancestrais na árvore é aproximadamente 3,3 segundos. Os experimentos foram realizados em um processador Intel Core i5 2.67GHz, com 6GB de memória RAM. Nós implementamos uma versão sequencial deste algoritmo.

Em ambos os métodos para resolver o PGP, usamos paralelização *multithreading*: a heurística (inclusão passo-a-passo) é repetida 100 vezes, distribuída em três threads concorrentes; o algoritmo *branch-and-bound* também foi executado concorrentemente em três threads, compartilhando os limitantes entre elas. Executamos a heurística no mesmo computador descrito acima, e o SCJ reconstruiu a topologia dos 66 protostômios em 3,9 segundos, enquanto a topologia de 64 genomas (dados simulados) foi obtida em 10,7 minutos, usando computadores com 4GB de RAM e processadores Intel Core 2 Quad 2.40GHz. Note a grande diferença de tempo entre eles, devido ao tamanho dos genomas de entrada: os dados simulados tem genomas com 2000 genes como tamanho padrão, enquanto os genomas dos protostômios tem somente 36 genes. Nós observamos que, em ambos os casos, o número de genomas e o número de genes impacta no tempo de processamento da heurística.

Os experimentos para resolver o PGP com o método exato gastaram um tempo significativamente maior, devido à complexidade do problema para o SCJ. Como o conjunto de dados de Campanulaceae é pequeno (13 genomas), é possível encontrar árvores ótimas em cerca de 6,9 horas, usando um computador com 4GB de memória RAM e processador Intel Core 2 Quad 2.40GHz.

Para a reconstrução da topologia com 12 genomas (dados simulados), usamos dois ambientes diferentes: um com 128GB de RAM e quatro processadores IBM Power7 com 3.55GHz de clock; e outro com 8GB RAM e processador Intel Xeon 2.93GHz. Estes experimentos levaram desde algumas horas até 5 dias para completarem, com um tempo médio de 1,2 dias. Uma escolha mais cuidadosa dos limitantes superior e inferior pode aumentar a eficiência do método. Note novamente a diferença de tempo causada pelo tamanho dos genomas, que impacta altamente na complexidade do cálculo da mediana e de outras operações básicas frequentemente usadas.

Capítulo 8

Conclusões

O *Single-Cut-or-Join* (SCJ) foi proposto por Feijão e Meidanis com o objetivo de ser um modelo de rearranjo bastante simples, usando os dois eventos mais básicos que alteram a ordem dos genes nos genomas: o corte e a junção das extremidades dos genes. Baseado na ideia de *breakpoints*, podendo ser considerado até como uma definição formal destes, o SCJ não possui relação direta com eventos biológicos frequentemente observados, como inversões e translocações, e, por essa razão, havia dúvidas sobre sua relevância biológica e aplicabilidade em cenários reais.

Respondendo a estas dúvidas, neste trabalho, realizamos experimentos que verificam a capacidade do SCJ em reconstruir histórias evolutivas, em dois aspectos: (1) quão bem o SCJ reconstrói topologias evolutivas?, e (2) quão bem o SCJ reconstrói genomas ancestrais?

Em relação à reconstrução de topologias, usando um método heurístico, o SCJ é capaz de recuperar de 60% a 90% da topologia, mensurado através da distância RF entre a árvore original e reconstruída, usando como entrada dados simulados. Este valor foi obtido com testes envolvendo diversas topologias aleatórias, tomadas a partir de uma distribuição que se aproxima ao que é encontrado na vida real, e variando diversos parâmetros, incluindo quantidade de inversões, translocações e transposições em cada aresta, o número dos genomas recebidos como entrada (até 200 genomas) e também o número de genes dos genomas (até 2000 genes). Nós não temos conhecimento de outros experimentos com conjuntos de dados tão grandes quanto estes, além daqueles que usam métodos escaláveis, como o método de Cobertura por Discos [71], que por sua vez também podem ser aplicados ao SCJ, aumentando ainda mais a sua escalabilidade. Foi possível resolver estes grandes conjuntos de dados com SCJ devido aos seus algoritmos extremamente rápidos.

O algoritmo exato para a topologia (Problema Grande da Parcimônia por SCJ) também foi implementado e testado. Neste caso, até 95% da topologia foi reconstruída. Entretanto, este algoritmo pode ser usado somente em instâncias menores, até 13 geno-

mas, devido ao longo tempo que leva.

Em dados reais, a capacidade do SCJ em reconstruir topologias também foi notável. Para o conjunto de dados Campanulaceae, o SCJ foi capaz de reconstruir a topologia aceita através do método exato. No conjunto de dados de Protostômios, o SCJ foi capaz de reconstruir diversos clados importantes, como Arthropoda, Nematoda, Echinodermata, Annelida e Platyhelminthes. Mollusca foi um problema, mas este clado é conhecido por ter problemas especialmente difíceis. A reconstrução alcançada pelo SCJ foi melhor do que em outros métodos relatados na literatura.

Com respeito à reconstrução dos genomas ancestrais para uma dada topologia, o sucesso do SCJ depende de quão longe da raiz o ancestral está. No caso dos nós com altura 1, a percentagem de reconstrução média é de 85%. Sendo h a altura máxima do nó, que representa a altura da raiz, para os nós mais próximos das folhas, cuja altura varia entre 1 e $\frac{h}{3}$, cerca de 77% das adjacências podem ser recuperadas. Esta percentagem decresce à medida que nos movemos em direção à raiz, mas, mesmo na raiz, cerca de 50% das adjacências podem ser reconstruídas (vide Figura 8.1). Nossos resultados corroboram o fato de que o SCJ leva a reconstruções de genomas bastante conservadoras, rendendo muito poucos falso-positivos nas adjacências de genes dos ancestrais, à custa de uma quantidade relativamente maior de falso-negativos. Quando o número de eventos por aresta corresponde a 5% do tamanho do genoma, acima de 90% de todos os genomas ancestrais foi coberto.

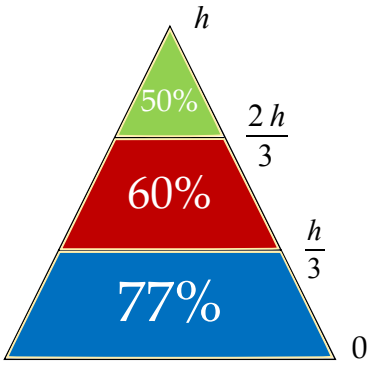

Figura 8.1: Percentagem de reconstrução relacionada à altura do nó.

Nós usamos esta característica do SCJ para reconstruir grupos de genes muito conservados dos artrópodes e de outras classes, que podem ser interessantes também em outros estudos.

As principais contribuições deste trabalho vêm dos resultados obtidos com este experimento, que atestam a aplicabilidade do modelo SCJ, que se mostrou acurado durante a inferência dos principais aspectos que compõem a árvore (ancestrais e topologia).

8.1 Trabalhos Futuros

Diversas extensões deste trabalho podem ser realizadas. A maioria delas envolve o projeto que implementa diversos algoritmos que utilizam a operação de rearranjo SCJ, iniciado por Feijão durante seu doutorado e prosseguido durante este mestrado. Mais detalhes deste projeto serão apresentados no Material Suplementar. A seguir apresentamos algumas

ideias que podem ser executadas em um trabalho futuro.

- *Estender o estudo experimental a outras operações, como DCJ e HP.*
  O SCJ foi comparado a outros métodos através de dados reais. Para realizar essa comparação com dados simulados, é preciso refatorar o projeto dando prioridade ao requisito de manutenibilidade, de forma que outros modelos de rearranjo possam ser facilmente incorporados.

- *Tonar o simulador mais realístico.*
  Ao simular a evolução de um conjunto de genomas, consideramos alguns parâmetros, tais como tamanho do genoma, tamanho da árvore e quantidade de cada tipo de evento nas arestas. Além destes, outros parâmetros podem tornar a simulação mais realística, como o tamanho da área rearranjada e a posição do cromossomo onde ocorre o rearranjo, por exemplo.

- *Tratar eventos de perda, ganho e duplicações de genes.*
  Neste estudo não consideramos eventos que alteram o conteúdo de genes do genoma, como inserções, deleções e duplicações. Ao aplicar o SCJ em conjuntos de dados reais, tivemos que tratar de forma manual os genomas com conteúdo gênico diferente. Um modo mais eficiente de tratar estes eventos seria integrar o modelo SCJ a um pré-tratamento, como o proposto por Côgo [25], que posteriormente foi aperfeiçoado por Zupo [26]. Implementar um pré-tratamento também permitiria que estes eventos fossem incorporados ao simulador.

- *Propor uma heurística mais acurada, com resultados mais próximos ao ótimo.*
  Ao testar a heurística de Inclusão Passo-a-Passo em conjuntos com 12 genomas, notamos que esta dificilmente alcançava a topologia ótima. Como existe uma grande variedade de heurísticas propostas para inferir árvores filogenéticas, tais como as populares heurísticas de Troca de Ramos (*Branch Swapping*) e Decomposição da Estrela (*Star decomposition*), além de muitas outras que usam diversas técnicas de inteligência artificial, podemos adaptá-las ao modelo SCJ e verificar se há uma melhoria na acurácia.

- *Propor um método exato mais eficiente, permitindo que este seja escalável a um conjunto maior de genomas.*
  Apesar do Problema Grande da Parcimônia ser NP-Difícil com o modelo SCJ, é possível tornar a resolução deste problema mais eficiente de duas formas possíveis:

  – Propondo limitantes melhores para o método *branch-and-bound*, principalmente o limitante inferior. Durante o experimento, computamos em que nível da árvore de pesquisa era realizada a operação de *poda*, ou seja, quando uma

subárvore da árvore de pesquisa é eliminada pelo fato de seu melhor caso (limitante inferior), não melhorar a solução atual. Em grande parte das vezes, esta operação é realizada apenas no penúltimo ou antepenúltimo nível da árvore. Propor um limitante inferior que permita realizar a poda de um ramo em níveis anteriores reduziria consideravelmente o esforço computacional necessário para obter a topologia ótima.

  - Otimizando o código. Mais especificamente, pensamos em duas formas possíveis. A primeira seria investir na paralelização do código, de forma que aproveite melhor os ambientes computacionais utilizados, como o ambiente do CENAPAD-SP, que é bastante poderoso. Uma possível ideia seria distribuir o processamento dos métodos utilizando algum padrão para passagem de mensagens, como o MPI (*Message Passing Interface*). A segunda forma seria através da otimização de rotinas muito usadas, como a que computa a mediana por SCJ. Como vimos na análise de eficiência, o número de genes tem alto impacto nos tempos obtidos, devido à complexidade de métodos muito usados dependerem deste argumento. Melhorar estas rotinas permitiria que o SCJ fosse aplicado a genomas maiores que 3000 genes, viabilizando o tratamento de organismos mais complexos.

- *Implementar uma interface para o projeto SCJ.*
  Como os resultados da inferência por SCJ obtidos neste experimento foram bastante positivos, seria interessante disponibilizar uma ferramenta de fácil uso para que outros pesquisadores também possam usar os métodos SCJ e as outras funcionalidades do projeto, como o simulador. Para isso é necessário refatorar o projeto, dando ênfase à sua usabilidade, já que atualmente a interação com o sistema é feita por linha de comando, através de vários parâmetros.

Apêndice A

SCJ suite – Manual do Usuário

Os experimentos descritos neste trabalho utilizaram o projeto *"SCJ suite"* em sua execução. Este projeto implementa, na linguagem Java, diversos algoritmos que resolvem problemas de rearranjos de genomas com a operação SCJ. Além dos algoritmos descritos na Seção 6.3, o projeto engloba outros algoritmos relacionados a problemas de rearranjos e também scripts que auxiliam na obtenção de métricas e visualização dos resultados.

Nas seções seguintes, explicaremos onde encontrar e como utilizar o projeto *SCJ suite*, detalhando um de seus módulos, chamado de *CompareTrees*, que foi o módulo usado neste experimento.

A.1 Estrutura

Os arquivos que compõem a versão executável do projeto, compactados no arquivo "SCJ-suite.zip", estão organizados da seguinte forma:

- *scj.jar:* executável do projeto;
- *data:* esta pasta contém todos os dados usados no experimento. As topologias das árvores simuladas estão na subpasta "simulatedTrees" (topologias geradas com o modelo *beta-splitting*, explicado na Seção 6.2.1); e os conjuntos de genomas reais estão na subpasta "input";
- *graficos:* pasta onde os resultados dos métodos ficam armazenados;
- *scriptsPerl:* nesta pasta estão os scripts auxiliares, usados no cálculo de métricas.

A versão executável e o código-fonte do projeto são disponibilizados no endereço:

`http://www.ic.unicamp.br/~meidanis/PUB/Mestrado/2010-Biller.`

Após realizar o download, basta descompactar os arquivos, preferencialmente em um diretório que não contenha espaços no nome nem em seu caminho completo.

A.2 Requisitos do Sistema

O projeto foi testado nos sistemas Windows e Linux. Para ser executado, são necessários os seguintes softwares:

- *Perl*, para execução dos scripts auxiliares;
- *Java*, para execução do projeto principal. O sistema requer Java 5, ou uma versão superior;

A.3 Arquivos de Entrada

A.3.1 Genomas – Formato de Permutações

Quando os genomas não são criados pelo projeto, mas recebidos por um arquivo de entrada, o projeto considera m genomas *unicromossomais*, lineares ou circulares, com n genes. Os genes são representados pelos números $1, 2, \ldots, n$ ou pelos nomes $nome_1, nome_2, \ldots, nome_n$. As duas orientações do gene i são representadas por $+i$ e $-i$. Um genoma é representado por uma permutação com sinais dos números $1, 2, \ldots, n$ (ou dos nomes, se for o caso). O projeto é capaz de criar internamente genomas multicromossomais, mas não os recebe por arquivos atualmente. Não é possível utilizar arquivos de entrada que contenham conjuntos de genomas multicromossomais ou com conteúdo gênico variável.

Um conjunto de genomas é armazenado em um arquivo texto com o seguinte formato:

- A primeira linha do arquivo tem o formato "m n [s]", onde m é o número de genomas, n é o número de genes e $[s]$ é a indicação de que os genes são representados por texto ao invés de número. Os dois primeiros argumentos são obrigatórios e o terceiro argumento é opcional, e todos são separados por um espaço. A ausência do terceiro argumento indica que os genes são representados por $1, 2, \ldots, n$; sua presença indica que os genes são representados por $nome_1, nome_2, \ldots, nome_n$;
- Se o argumento $[s]$ está presente na primeira linha, a segunda linha deve conter os nomes dos genes: "$nome_1$ $nome_2$ ... $nome_n$". Os nomes dos genes são compostos por letras e números e devem ser separados por um espaço;

- As demais linhas contém os genomas das espécies. Cada genoma é representado em três linhas:
  - Na primeira linha deve estar o nome do genoma, no formato "*>nome_do_genoma*", como ocorre no formato FASTA, onde a inclusão de um genoma se inicia pelo seu nome. Note que o nome do genoma só é reconhecido quando o símbolo ">" o precede. O nome do genoma é composto por letras, números, hífen ou *underline*;
  - Na segunda linha deve estar a orientação do genoma, que pode ser "linear" ou "circular";
  - Na terceira linha devem estar a ordem e a orientação dos genes no genoma. Os genes são separados por um espaço, e cada gene é representado por "*sentido_na_fita rótulo_do_gene*", onde o argumento *sentido_na_fita* pode ser "+" ou "−" e o argumento *rótulo_do_gene* pode ser um número ou o nome do gene, no caso da opção [*s*] estar habilitada na primeira linha. Caso sejam usados os nomes dos genes, são válidos apenas os nomes especificados na segunda linha do arquivo, sensíveis a maiúsculas e minúsculas.

Seguindo o formato explicado, um exemplo de conjunto de genomas válido seria:

```
3 20
>rco
linear
+1 +2 +3 +4 +5 +6 +7 +8 +9 +10 +11 +13 -12 +14 +15 +16 +17 +18 +19 +20
>raf
linear
+1 +2 +3 +4 +5 +6 +7 +8 +9 +10 +11 +12 -13 +14 +15 +16 +17 +18 +19 +20
>rma
linear
+1 +2 +3 +4 +5 +6 +7 +8 +9 +10 +11 +12 +13 +14 +15 +16 +17 +18 -19 +20
```

Neste caso a primeira linha indica que existem 3 genomas (*rco*, *raf* e *rma*) com 20 genes cada. A omissão do argumento [*s*] indica que os genes são representados por números. Todos os genomas são lineares neste caso. No caso dos genes representados de forma literal, um exemplo válido seria:

```
4 18 s
K D atp8 A R S1 E F nad5 H nad4L T P S2 nad1 L1 rrnL Q
>Nematoda_Trichinella_spiralis
circular
+E +nad1 +K -F -nad5 -H -R -nad4L +T -P +S1 +rrnL +Q +D +atp8 +S2 +L1 +A
>Arthropoda_Anopheles_quadrimaculatus
circular
+K +D +atp8 +R +A -S1 +E -F -nad5 -H -nad4L +T -P +S2 -nad1 -L1 -rrnL -Q
>Arthropoda_Apis_mellifera
circular
+D +K +atp8 -R -F -nad5 -H -nad4L +T -P +S2 -nad1 -L1 -rrnL +E +S1 +Q +A
>Hemichordata_Balanoglossus_carnosus
circular
-S2 +D +K +atp8 +R +nad4L +H +S1 +nad5 +E +T -P +F +rrnL +L1 +nad1 -Q -A
```

Na primeira linha do arquivo o formato literal foi indicado através da opção $[s]$ e, na segunda linha, os nomes dos genes foram especificados. Para exemplos adicionais, verifique os arquivos da pasta "data/input".

A.3.2 Árvore – Formato Newick

O formato Newick é largamente utilizado por diversos softwares de reconstrução filogenética para representar árvores em um arquivo texto. Neste projeto usamos uma simplificação do formato Newick, isto é, as árvores representadas pelo formato explicado a seguir são árvores Newick válidas, mas nem todas as árvores representadas pelo formato Newick seriam consideradas válidas no projeto.

Em nosso formato, apenas as folhas, que representam as espécies conhecidas, são rotuladas. O rótulo pode ser o nome da espécie ou qualquer texto composto por letras, números, hífens ou underlines. O nome não pode conter espaços: utilize hífen ou underline no lugar de um espaço.

No projeto são consideradas apenas árvores binárias. Dado um nó pai, seus filhos esquerdo e direito são representados com o seguinte formato:

$$(filho_direito : peso, filho_esquerdo : peso).$$

Se o filho é uma folha, basta inserir o rótulo da folha. Se o filho é um nó interno, esta estrutura se repete dentro do argumento *filho_direito* (ou *filho_esquerdo*). As arestas da

árvore possuem pesos, definidos por números naturais que indicam o número de eventos que separam o genoma filho do genoma pai, considerando qualquer tipo de distância.

A representação não pode conter espaços ou tabulações. Cada linha contém a representação de uma árvore e, ao final da linha, deve ser colocado um ponto-e-vírgula para indicar o fim da representação da árvore.

Um exemplo do formato explicado é apresentado a seguir, com a respectiva árvore que está sendo representada (Figura A.1). Note que todas as arestas devem possuir um custo associado e apenas as folhas são rotuladas. Para mais exemplos, consulte os arquivos que estão na pasta "data/simulatedTrees".

```
((((A:3,B:3):1,C:4):1,D:5):1,E:6);
```

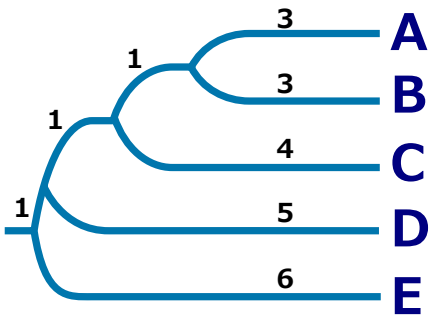

Figura A.1: Exemplo de árvore representada através do formato Newick.

A.4 Módulo *CompareTrees*

O projeto *SCJ suite* é composto por vários módulos. Nestes experimentos usamos o módulo *CompareTrees*, que tem por objetivo comparar duas árvores, a original e a inferida. Este módulo executa todo o processo, desde a preparação dos dados até o cálculo das métricas, conforme será explicado a seguir.

A.4.1 Fluxo do Sistema

Baseado nas informações passadas pelo usuário, o sistema produz os dados de entrada ou recebe-os por arquivos, passa os dados aos métodos, obtém a resposta, calcula as métricas que comparam o resultado original e inferido, e armazena os resultados da inferência, bem como as métricas calculadas, em um arquivo de saída. Neste módulo o usuário precisa definir três informações:

1. Qual problema será resolvido;
2. Qual método será usado para resolver o problema;
3. Quais serão os dados de entrada usados.

Na primeira etapa, o usuário pode escolher se deseja resolver o Problema Pequeno ou o Problema Grande da Parcimônia. Dependendo do problema escolhido, existe um subconjunto de métodos disponíveis para resolvê-lo. Atualmente, é possível resolver o

PPP usando somente a adaptação do método de Fitch, enquanto o PGP pode ser resolvido com o método heurístico (inclusão passo-a-passo) ou exato (*branch-and-bound*). A última etapa define os dados, que podem ser ou árvores simuladas ou conjuntos de genomas reais. Em cada etapa é possível escolher somente uma das opções disponíveis. Atualmente não é possível o usuário definir as métricas e o formato de saída, que por enquanto são fixos. A Figura A.2 sumariza as etapas que o usuário precisa definir e as respectivas escolhas que podem ser tomadas. Ao definir uma escolha para cada etapa, o usuário define um *cenário de uso*, ou seja, um dos caminhos possíveis do diagrama apresentado na figura.

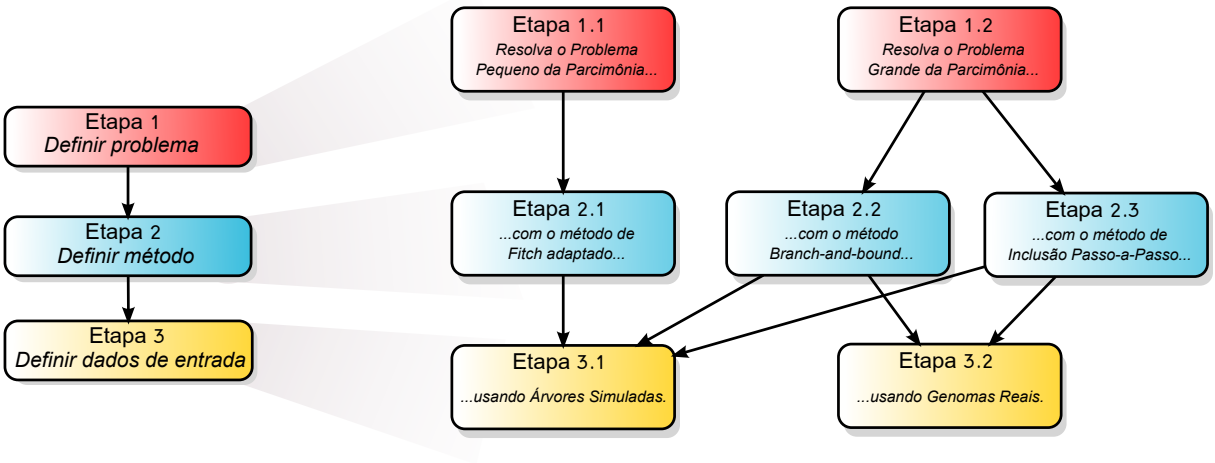

Figura A.2: Módulo *CompareTrees* — Etapas. Em cada etapa, o usuário precisa selecionar apenas uma das opções disponíveis. O conjunto de escolhas do usuário compõe um *cenário de uso*.

A.4.2 Parâmetros

A interação entre o usuário e o sistema ocorre através de parâmetros. Anteriormente explicamos sobre as decisões que o usuário precisa tomar para definir um *cenário de uso*. Cada escolha é caracterizada por um conjunto de parâmetros e valores específicos, apresentados no diagrama da Figura A.3. Todos os parâmetros são descritos com mais detalhes na Tabela A.1, que também define o domínio de valores válidos e formas alternativas para representá-lo.

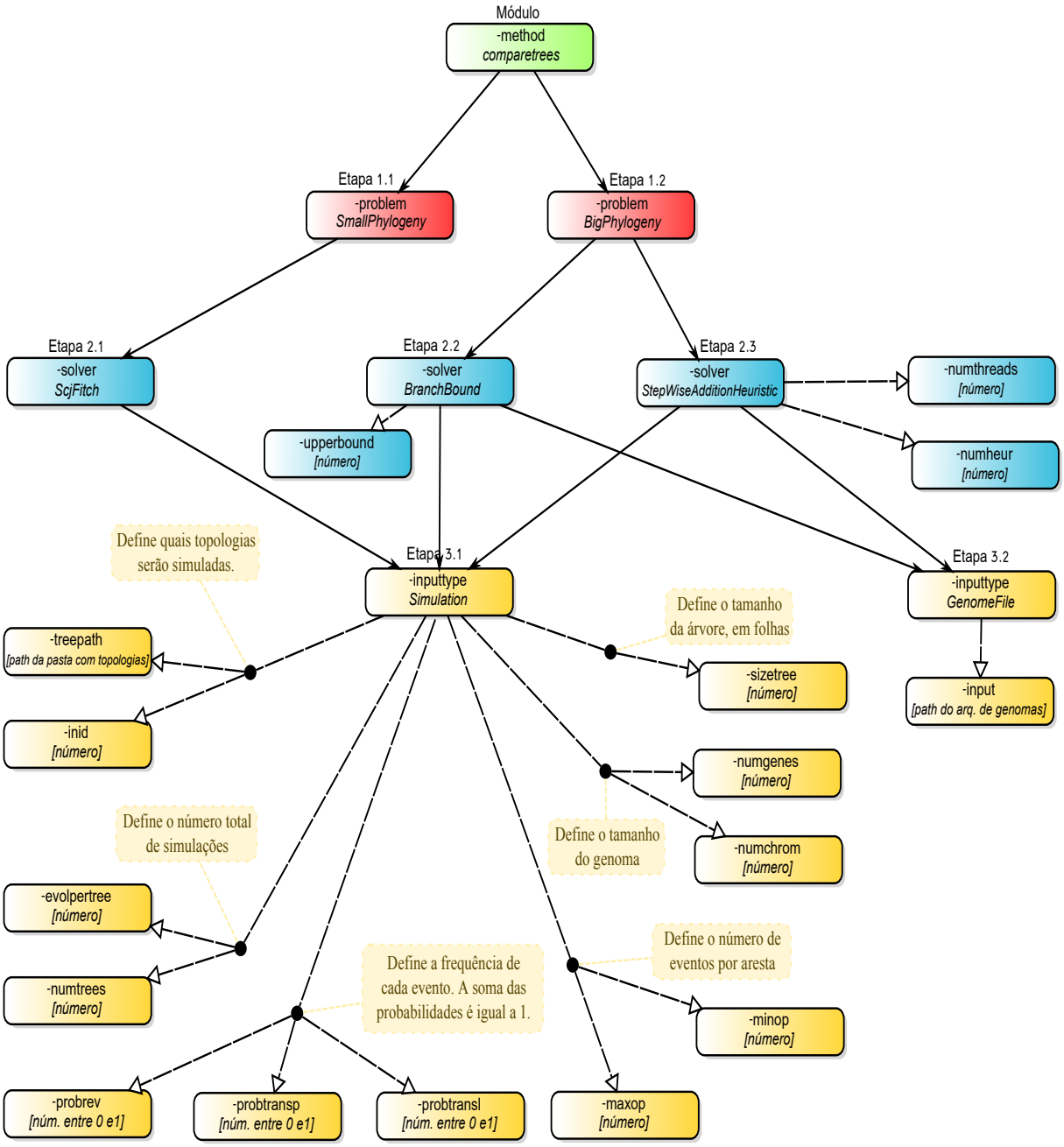

Figura A.3: Módulo *CompareTrees* – Parâmetros. Neste diagrama, cada bloco corresponde a um parâmetro: o texto em negrito indica o nome do parâmetro e, o texto em itálico, seu valor. O relacionamento entre os parâmetros é representado através de setas. As setas cheias indicam um fluxo possível, correspondendo às etapas definidas na Figura A.2. As setas pontilhadas indicam a dependência entre os parâmetros, isto é, todos os parâmetros que um determinado parâmetro aponta, com setas pontilhadas, devem estar presentes no comando com os respectivos valores indicados.

Tabela A.1: Descrição dos parâmetros do módulo *CompareTrees*. O domínio de tipos não numéricos, como o tipo Texto, é sensível a maiúsculas e minúsculas. $\mathbb{N}^*$ denota todos os números naturais, exceto zero.

Parâmetro	Forma abrev.	Descrição	Domínio	Etapas
-evolpertree	-e	Número de evoluções simuladas executadas em cada árvore.	$\mathbb{N}^*$	3.1
-inid		Identificador da primeira topologia que será simulada.	$\mathbb{N}^*$	3.1
-input	-i	Caminho completo para o arquivo com o conjunto de genomas.	Texto	3.2
-inputtype		Tipo dos dados de entrada.	{Simulation, GenomeFile}	3.1, 3.2
-maxop		Número máximo de eventos evolutivos em cada aresta da árvore.	$n \geq [minop]$	3.1
-method	-m	Módulo que será executado.	{comparetrees}	—
-minop		Número mínimo de eventos evolutivos em cada aresta da árvore.	$\mathbb{N}^*$	3.1
-numchrom	-c	Número de cromossomos do genoma ancestral.	$\mathbb{N}^*$	3.1
-numgenes	-g	Número de genes do genoma ancestral.	$n \geq [numchrom]$	3.1
-numheur	-h	Número de vezes que a heurística será executada para cada árvore.	$\mathbb{N}^*$	2.3
-numthreads		Número máximo de instâncias da heurística que podem ser executadas em paralelo.	$\mathbb{N}^*$	2.3
-numtrees	-t	Quantidade de topologias que serão simuladas.	$\mathbb{N}^*$	3.1
-problem		Problema que será resolvido.	{SmallPhylogeny, BigPhylogeny}	1.1, 1.2
-probrev	-prv	Frequência de inversões durante a evolução de uma aresta.	$0.0 \leq n \leq 1.0$	3.1
-probtransl	-ptl	Frequência de translocações durante a evolução de uma aresta.	$0.0 \leq n \leq 1.0$	3.1
-probtransp	-ptp	Frequência de transposições durante a evolução de uma aresta.	$0.0 \leq n \leq 1.0$	3.1
-sizetree	-s	Número de folhas da árvore.	$\mathbb{N}^*$	3.1
-solver		Método que resolverá o problema.	{StepWiseAdditionHeuristic, BranchBound, ScjFitch}	2.1, 2.2, 2.3
-treepath		Caminho até a pasta onde estão as topologias que serão usadas nas simulações.	Texto	3.1
-upperbound	-u,-ub	Limitante superior inicial para o *branch-and-bound*.	$\mathbb{N}^*$	2.2

Alguns detalhes devem ser observados, dependendo do tipo de dado utilizado. Caso os dados de entrada sejam genomas reais, é necessário especificar o caminho completo do arquivo que contém o conjunto de genomas (por exemplo, "C:\scj\data\input\campanulaceae.txt").

Estes genomas devem estar representados conforme explicado na Seção A.3.1. No caso dos dados de entrada serem dados simulados, é necessário especificar o caminho completo até a pasta que contém as topologias das árvores (por exemplo, "C:\scj\data\simulatedTrees\size32"). Em ambos os casos, se o sistema operacional for o Windows, é necessário delimitar o caminho com aspas.

A pasta que armazena as topologias das árvores deve conter um conjunto de arquivos, onde cada arquivo especifica uma única topologia, conforme descrito na Seção A.3.2. Os arquivos devem ter o nome com o seguinte padrão:

`tree[id_da_árvore].nwk`

O argumento *id_da_árvore* é um número utilizado como identificador da árvore. Na pasta especificada, o identificador deve começar com o valor 0 ("tree0.nwk") e ser incrementado em 1 a cada arquivo. O parâmetro *-inid* define a partir de qual topologia ocorrerá a evolução simulada. Note que todas as topologias com identificadores entre *inid* e $inid + numtrees - 1$ devem existir na pasta, caso contrário ocorrerão erros durante a simulação.

A.4.3 Cenários de Uso: Entradas e Saídas

Nesta seção exemplificamos o comando Java e a resposta do sistema em cada cenário de uso. Nos exemplos, vamos supor que o sistema operacional é o Windows e o arquivo "SCJ-suite.zip" foi descompactado na pasta "C:\". Os exemplos correspondem aos cenários usados nos experimentos, listados na Tabela A.2.

Tabela A.2: Cenários de uso do módulo *CompareTrees*. Abreviações utilizadas: PPP, Problema Pequeno da Parcimônia; PGP, Problema Grande da Parcimônia; B&B, *branch-and-bound*; IPP, Inclusão Passo-a-Passo; Sim., Simulados.

Cenário	Problema		Método			Dados	
	PPP	PGP	Fitch	B&B	IPP	Sim.	Reais
1	•		•			•	
2		•		•		•	
3		•		•			•
4		•			•	•	
5		•			•		•

O comando Java executado em cada cenário apresenta uma sintaxe similar, variando apenas os conjuntos de parâmetros, escolhidos conforme as decisões do usuário. A sintaxe básica do comando é a seguinte:

```
java -jar scj.jar -method comparetrees <parâmetros do problema>
                  <parâmetros do método> <parâmetros dos dados de entrada>
```

A saída do sistema, constituída dos resultados inferidos e das métricas, é armazenada em um arquivo de log, localizado na pasta "graficos". O nome do arquivo de log, independente do cenário, é o seguinte:

```
problem[valor]-method[valor]-input[valor]-[parâmetros variáveis e seus valores].txt
```

Caso já exista um arquivo com o mesmo nome, é incluído o sufixo "-try$n$.txt" ao nome do arquivo, tal que n seja um número ainda não usado na composição de nomes de arquivos semelhantes. Assim como o comando Java, o arquivo de log possui uma estrutura básica:

1. No começo do arquivo de log estão os valores dos parâmetros de entrada;
2. Para cada conjunto de dados de entrada, são informados:
   (a) Dados originais e métricas aplicadas a estes dados;
   (b) Dados inferidos e métricas aplicadas a estes dados;
   (c) Resumo das métricas obtidas com os resultados inferidos e tempos computacionais.

Cenário 1

Problema: PPP; Método: Fitch; Dados simulados.

Comando Java

```
java -jar scj.jar -method comparetrees -problem SmallPhylogeny -solver ScjFitch
                  -inputtype Simulation -sizetree 5 -evolpertree 1 -numtrees 1
                  -numgenes 5 -numchrom 2 -minop 1 -maxop 1 -inid 0 -probrev 0.9
                  -probtranspo 0.0 -probtransl 0.1 -treepath "C:\scj\data\simulatedTrees"
```

Arquivo de log obtido

```
1  MIN REV  GEN MAX TRP TRL  CHR FOL
2  1   90.0 5   1   0.0 10.0 2   5
3
4  SIMULATION(1/1)        Tue Jan 31 22:54:48 GMT-03:00 2012
5
6  ID  NUM_REV NUM_TRANSL  NUM_TRANSP  BRANCHLEN_SCJ(REAL) BRANCHLEN_SCJ(OPT)  TREE
7  0   7.0     0.0         0.0         14.0                4.0                 (E,(D,(C,(B,A))));
8
9  HEIGHT NODE            %RECONS  FALSO_POS  FALSO_NEG  PARENT          SCJ_DIST_TO_PARENT  GENOME_RECONSTRUCTED     GENOME_ORIGINAL
10 0      A               1        0          0          internal3       0                   [[ 1 -2 ][ -3 4 ][ 5 ]]  [[ 1 -2 ][ -3 4 ][ 5 ]]
11 0      B               1        0          0          internal3       0                   [[ 1 -2 ][ -3 4 ][ 5 ]]  [[ 1 -2 ][ -3 4 ][ 5 ]]
12 0      C               1        0          0          internal2       0                   [[ 1 -2 ][ -3 4 ][ 5 ]]  [[ 1 -2 ][ -3 4 ][ 5 ]]
13 0      D               1        0          0          internal1       2                   [[ 1 2 ][ -3 -4 ][ 5 ]]  [[ 1 2 ][ -3 -4 ][ 5 ]]
14 0      E               1        0          0          Evolution root  0                   [[ 1 2 ][ -3 4 ][ 5 ]]   [[ 1 2 ][ -3 4 ][ 5 ]]
15 1      internal3       0.5      1          1          internal2       0                   [[ 1 -2 ][ -3 4 ][ 5 ]]  [[ -1 -2 ][ -3 4 ][ 5 ]]
16 2      internal2       1        0          0          internal1       2                   [[ 1 -2 ][ -3 4 ][ 5 ]]  [[ 1 -2 ][ -3 4 ][ 5 ]]
17 3      internal1       1        0          0          Evolution root  0                   [[ 1 2 ][ -3 4 ][ 5 ]]   [[ 1 2 ][ -3 4 ][ 5 ]]
18 4      Evolution root  0.5      1          1          -               0                   [[ 1 2 ][ -3 4 ][ 5 ]]   [[ 1 2 ][ 3 4 ][ 5 ]]
19
20 TEMPOS(MS)
21 SIM    INF     SPLIT
22 12     1       0
```

As Linhas 1–2 do arquivo de log apresentam os valores dos parâmetros do comando Java. A Linha 4 apresenta o tipo de dado usado, o andamento da execução e a data/hora em que uma execução específica iniciou. As Linhas 6–7 apresentam o identificador da topologia simulada (ID), o número de eventos evolutivos de cada tipo (NUM_REV, NUM_TRANSP e NUM_TRANSL) e o custo em SCJs da árvore, considerando os genomas ancestrais originais (BRANCHLEN_SCJ(REAL)) e os genomas ancestrais inferidos pelo método de Fitch (BRANCHLEN_SCJ(OPT)). As Linhas 9–18 apresentam as métricas de ancestrais para cada nó da árvore, além do genoma original (GENOME_ORIGINAL) e reconstruído (GENOME_RECONSTRUCTED).

Cenário 2

Problema: PGP; Método: B&B; Dados simulados.

Comando Java

```
java -jar scj.jar -method comparetrees -problem BigPhylogeny -solver BranchBound
             -upperbound 1000 -inputtype Simulation -sizetree 5 -evolpertree 1 -inid 0
             -numtrees 1 -numgenes 5 -numchrom 2 -minop 1 -maxop 1 -probrev 0.9
             -probtransp 0.0 -probtransl 0.1 -treepath "C:\scj\data\simulatedTrees"
```

Arquivo de log obtido

```
1  MIN  REV   GEN  MAX  TRP  TRL   CHR  FOL
2  1    90.0  5    1    0.0  10.0  2    5
3
4  SIMULATION(1/1)        Tue Jan 31 23:14:20 GMT-03:00 2012
5
6  ID  NUM_REV NUM_TRANSL  NUM_TRANSP  BRANCHLEN_SCJ(REAL) BRANCHLEN_SCJ(OPT) TREE
7  0   7.0     0.0         0.0         14.0                4.0                 (E,(D,(C,(B,A))));
8
9  #ID BRANCH  SPLIT   INFER(MS)   SPLIT(MS)   TREE
10 1   4.0     0.5     13          723         (E,((C,D),(B,A)));
11 2   4.0     0.5     13          723         (E,(((B,A),D),C));
12 3   4.0     0       13          723         (E,(((B,A),C),D));
13
14 SPLIT_INFER
15 BEST   WORST   AVG
16 0      0.5     0.33
17
18 BRANCH_LENGTH_INFER
19 BEST   WORST   AVG
20 4      4       4
21
22 TEMPOS(MS)
23 SIM    INF     SPLIT
24 12     39      2170
```

As Linhas 1–2 do arquivo de log apresentam os valores dos parâmetros do comando Java. A Linha 4 apresenta o tipo de dado usado, o andamento da execução e a data/hora em que uma execução específica iniciou. As Linhas 6–7 apresentam o identificador da topologia simulada (ID), o número de eventos evolutivos de cada tipo (NUM_REV, NUM_TRANSP e NUM_TRANSL) e o custo em SCJs da árvore, considerando os genomas ancestrais originais (BRANCHLEN_SCJ(REAL)) e os genomas ancestrais inferidos pelo método de Fitch (BRANCHLEN_SCJ(OPT)). As Linhas 9–12 apresentam as métricas de topologia (SPLIT) para cada árvore ótima obtida, ou seja, as árvores que minimizam o número de SCJs (BRANCH). Estas linhas também informam, no final, a topologia reconstruída (TREE). As Linhas 14–20 sumarizam os valores das métricas, apresentando o melhor e o pior valor obtido, e também uma média de todos os valores.

Cenário 3

Problema: PGP; Método: B&B; Dados reais.

Comando Java

```
java -jar scj.jar -method comparetrees -problem BigPhylogeny -solver BranchBound
                  -upperbound 1000 -inputtype GenomeFile
                  -treepath "C:\scj\data\input\campanulaceae-exemplo.txt"
```

Arquivo de log obtido

```
1  ARQ
2  campanulaceae-exemplo
3
4  GENOMEFILE(1/1)        Wed Feb 01 00:20:07 GMT-03:00 2012
5
6  ARQ
7  CAMPANULACEAE-EXEMPLO
8
9  #ID  BRANCH  INFER(MS)  TREE
10 1    70.0    127        (Tobacco,((Codonopsis,(Campanula,Trachelium)),Platycodon));
11
12 BRANCH_LENGTH_INFER
13 BEST   WORST   AVG
14 70     70      70
15
16 TEMPOS(MS)
17 SIM    INF
18 35     127
```

As Linha 1–2 do arquivo de log apresentam o nome do arquivo com o conjunto de genomas, informado através do parâmetro "-treepath" do comando Java. A Linha 4 apresenta o tipo de dado usado, o andamento da execução e a data/hora em que uma execução específica iniciou. As Linhas 6–7 apresentam somente o nome do conjunto de genomas (ARQ), já que a topologia não é conhecida neste caso. As Linhas 9–10 apresentam a árvore ótima obtida, ou seja, aquela que minimiza o número de SCJs (BRANCH). Ao final da Linha 9, a topologia reconstruída é impressa no formato Newick (TREE). O tempo computacional necessário para simular (SIM) e para inferir a árvore (INF) é apresentado nas Linhas 16–18.

Cenário 4

Problema: PGP; Método: Inclusão Passo-a-Passo; Dados simulados.

Comando Java

```
java -jar scj.jar -method comparetrees -problem BigPhylogeny
             -solver StepWiseAdditionHeuristic -numheur 3 -numthreads 3
             -inputtype Simulation -sizetree 5 -evolpertree 1 -inid 0
             -numtrees 1 -numgenes 5 -numchrom 2 -minop 1 -maxop 1 -probrev 0.9
             -probtransp 0.0 -probtransl 0.1 -treepath "C:\scj\data\simulatedTrees"
```

Arquivo de log obtido

```
1  MIN  REV    GEN  MAX  TRP  TRL   CHR  FOL
2  1    90.0   5    1    0.0  10.0  2    5
3
4  SIMULATION(1/1)        Tue Jan 31 23:02:38 GMT-03:00 2012
5
6  ID  NUM_REV  NUM_TRANSL NUM_TRANSP  BRANCHLEN_SCJ(REAL) BRANCHLEN_SCJ(OPT)  TREE
7  0   7.0      0.0        0.0         14.0                4.0                 (E,(D,(C,(B,A))));
8
9  #ID BRANCH  SPLIT   INFER(MS)  SPLIT(MS)   TREE
10 1   4.0     0.5     3          816         ((B,(D,(E,C))),A);
11 2   4.0     0       3          816         ((D,((A,B),C)),E);
12 3   4.0     0.5     3          816         ((B,((E,C),D)),A);
13
14 SPLIT_INFER
15 BEST   WORST   AVG
16 0      0.5     0.33
17
18 BRANCH_LENGTH_INFER
19 BEST   WORST   AVG
20 4      4       4
21
22 TEMPOS(MS)
23 SIM    INF     SPLIT
24 17     10      2450
```

As Linhas 1–2 do arquivo de log apresentam os valores dos parâmetros do comando Java. A Linha 4 apresenta o tipo de dado usado, o andamento da execução e a data/hora em que uma execução específica iniciou. As Linhas 6–7 apresentam o identificador da topologia simulada (ID), o número de eventos evolutivos de cada tipo (NUM_REV, NUM_TRANSP e NUM_TRANSL) e o custo em SCJs da árvore, considerando os genomas ancestrais originais (BRANCHLEN_SCJ(REAL)) e os genomas ancestrais inferidos pelo método de Fitch (BRANCHLEN_SCJ(OPT)). As Linhas 9–12 apresentam as métricas de topologia (SPLIT) e a topologia reconstruída (TREE), para todas as árvores obtidas durante a execução da heurística, repetida conforme o número de vezes informado no parâmetro "-numheur". Como o método é heurístico, estas árvores podem não ter o custo (BRANCH) mínimo.

Cenário 5

Problema: PGP; Método: Inclusão Passo-a-Passo; Dados reais.

Comando Java

```
java -jar scj.jar -method comparetrees -problem BigPhylogeny
          -solver StepWiseAdditionHeuristic -numheur 5 -numthreads 3
          -inputtype GenomeFile -input "C:\scj\data\input\campanulaceae-exemplo.txt"
```

Arquivo de log obtido

```
1  ARQ
2  campanulaceae-exemplo
3
4  GENOMEFILE(1/1)        Wed Feb 01 00:16:11 GMT-03:00 2012
5
6  ARQ
7  CAMPANULACEAE-EXEMPLO
8
9  #ID  BRANCH  INFER(MS)  TREE
10 1    70.0    6          ((Campanula,(Codonopsis,(Platycodon,Tobacco))),Trachelium);
11 2    70.0    6          ((Platycodon,(Codonopsis,(Trachelium,Campanula))),Tobacco);
12 3    70.0    6          ((Tobacco,((Trachelium,Campanula),Codonopsis)),Platycodon);
13 4    70.0    6          (((Trachelium,Campanula),(Platycodon,Tobacco)),Codonopsis);
14 5    70.0    6          ((Tobacco,(Codonopsis,(Trachelium,Campanula))),Platycodon);
15
16
17 BRANCH_LENGTH_INFER
18 BEST   WORST   AVG
19 70     70      70
20
21 TEMPOS(MS)
22 SIM    INF
23 36     30
```

As Linha 1–2 do arquivo de log apresentam o nome do arquivo com o conjunto de genomas, informado através do parâmetro "-treepath" do comando Java. A Linha 4 apresenta o tipo de dado usado, o andamento da execução e a data/hora em que uma execução específica iniciou. As Linhas 6–7 apresentam somente o nome do conjunto de genomas (ARQ), já que a topologia não é conhecida neste caso. As Linhas 9–14 apresentam o tamanho (BRANCH) e a topologia reconstruída (TREE), para todas as árvores obtidas durante a execução da heurística, repetida conforme o número de vezes informado no parâmetro "-numheur". O tempo computacional necessário para simular (SIM) e para inferir as árvores (INF) é apresentado nas Linhas 21–23.

Referências Bibliográficas

[1] Zaky Adam and David Sankoff. The ABCs of MGR with DCJ. *Evolutionary Bioinformatics Online*, 4:69–74, 2008.

[2] Zaky Adam and David Sankoff. A statistically fair comparison of ancestral genome reconstructions, based on breakpoint and rearrangement distances. *Journal of Computational Biology*, 17(9):1299–1314, 2010.

[3] David Aldous. Stochastic models and descriptive statistics for phylogenetic trees, from Yule to today. *Statistical Science*, 16(1):23–34, 2001.

[4] Max Alekseyev and Pavel Pevzner. Colored de Bruijn graphs and the genome halving problem. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(1):98–107, 2006.

[5] David Bader, Bernard Moret, and Mi Yan. A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. *Journal of Computational Biology*, 8(5):483–491, 2001.

[6] Martin Bader. The transposition median problem is NP-complete. *Theoretical Computer Science*, 412:1099–1110, 2011.

[7] Vineet Bafna and Pavel Pevzner. Genome rearrangements and sorting by reversals. In *Proceedings of the 34th annual IEEE symposium on Foundations of Computer Science*, FOCS'1993, pages 148–157, 1993.

[8] Vineet Bafna and Pavel Pevzner. Sorting permutations by transpositions. In *Proceedings of the 6th annual ACM-SIAM symposium on Discrete algorithms*, SODA'95, pages 614–623, 1995.

[9] Anne Bergeron, Paul Medvedev, and Jens Stoye. Rearrangement models and Single-Cut operations. *Journal of Computational Biology*, 17(9):1213–1225, 2010.

[10] Anne Bergeron, Julia Mixtacki, and Jens Stoye. A unifying view of genome rearrangements. In *Proceedings of the 6th International Workshop on Algorithms in Bioinformatics*, WABI'2006, pages 163–173, 2006.

[11] Anne Bergeron, Julia Mixtacki, and Jens Stoye. HP distance via double cut and join distance. In *Combinatorial Pattern Matching*, volume 5029/2008 of *Lecture Notes in Computer Science*, pages 56–68. Springer Berlin / Heidelberg, 2008.

[12] Matthias Bernt, Daniel Merkle, and Martin Middendorf. Using median sets for inferring phylogenetic trees. *Bioinformatics*, 23(2):e129–e135, 2007.

[13] Mathieu Blanchette, Guillaume Bourque, and David Sankoff. Breakpoint phylogenies. In *Proceedings of the 8th Genome Informatics Workshop*, GIW 1997, pages 25–34, 1997.

[14] Mathieu Blanchette, Takashi Kunisawa, and David Sankoff. Parametric genome rearrangement. *Gene*, 172(1):GC11–GC17, 1996.

[15] Mathieu Blanchette, Takashi Kunisawa, and David Sankoff. Gene order breakpoint evidence in animal mitochondrial phylogeny. *Journal of Molecular Evolution*, 49(2):193–203, 1999.

[16] Guillaume Bourque and Pavel Pevzner. Genome-scale evolution: reconstructing gene orders in the ancestral species. *Genome Research*, 12(1):26–36, 2002.

[17] Marília Braga, Eyla Willing, and Jens Stoye. Genomic Distance with DCJ and Indels. In *Algorithms in Bioinformatics*, volume 6293 of *Lecture Notes in Computer Science*, pages 90–101. Springer Berlin / Heidelberg, 2010.

[18] David Bryant. The complexity of the breakpoint median problem. Technical Report CRM-2579, Centre de recherches mathematiques, Université de Montréal, 1998.

[19] Laurent Bulteau, Guillaume Fertin, and Irena Rusu. Sorting by transpositions is difficult. In *Proceedings of the 38th International colloquium on Automata languages and programming*, ICALP 2010, pages 654–665, 2010.

[20] Alberto Caprara. Sorting by reversals is difficult. In *Proceedings of the 1st annual International conference on Computational molecular biology*, RECOMB'97, pages 75–83, 1997.

[21] Alberto Caprara. The reversal median problem. *INFORMS Journal on Computing*, 15(1):93–113, 2003.

[22] Mary Cosner, Robert Jansen, Bernard Moret, Linda Raubeson, Li Wang, Tandy Warnow, and Stacia Wyman. A new fast heuristic for computing the breakpoint phylogeny and experimental phylogenetic analyses of real and synthetic data. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*, ISMB 2000, pages 104–115, 2000.

[23] Mary Cosner, Robert Jansen, Bernard Moret, Linda Raubeson, Li Wang, Tandy Warnow, and Stacia Wyman. An empirical comparison of phylogenetic methods on chloroplast gene order data in Campanulaceae. In *Proceedings of the Conference on Gene Order Dynamics, Comparative Maps, and Multigene Families*, DCAF'2000, pages 99–121, 2000.

[24] Mary Cosner, Linda Raubeson, and Robert Jansen. Chloroplast DNA rearrangements in Campanulaceae: phylogenetic utility of highly rearranged genomes. *BMC Evolutionary Biology*, 4:27, 2004.

[25] Patrícia Pilisson Côgo. Comparação de genomas completos de espécies da família Vibrionacea empregando rearranjo de genomas. Master's thesis, Instituto de Computação, Universidade Estadual de Campinas, 2008.

[26] Karina Zupo de Oliveira. Construção de filogenias baseadas em genomas completos. Master's thesis, Instituto de Computação, Universidade Estadual de Campinas, 2010.

[27] Zanoni Dias and João Meidanis. Genome rearrangements distance by fusion, fission, and transposition is easy. In *Proceedings of the 8th International symposium on String processing and information retrieval*, SPIRE 2001, pages 250–253, 2001.

[28] Nadia El-Mabrouk. Sorting signed permutations by reversals and insertions/deletions of contiguous segments. *Journal of Discrete Algorithms*, 1(1):105–122, 2001.

[29] Stefan Faderl, Moshe Talpaz, Zeev Estrov, and Hagop Kantarjian. Chronic myelogenous leukemia: biology and therapy. *Annals of Internal Medicine*, 131(3):207–219, 1999.

[30] Pedro Feijão and João Meidanis. SCJ: a variant of breakpoint distance for which Sorting, Genome Median and Genome Halving Problems are easy. In *Proceedings of the 9th International Workshop on Algorithms in Bioinformatics*, WABI'2009, pages 85–96, 2009.

[31] Pedro Feijão and João Meidanis. SCJ: a breakpoint-like distance that simplifies several rearrangement problems. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8:1318–1329, 2011.

[32] Joseph Felsenstein. PHYLIP (Phylogeny Inference Package) version 3.69. `http://evolution.genetics.washington.edu/phylip/`. (acessado em 06 de fevereiro de 2012).

[33] Joseph Felsenstein. Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of Molecular Evolution*, 17(6):368–376, 1981.

[34] Joseph Felsenstein. *Inferring Phylogenies.* Sinauer Associates, 2004.

[35] Guillaume Fertin, Anthony Labarre, Irena Rusu, Eric Tannier, and Stéphane Vialette. *Combinatorics of Genome Rearrangements.* MIT Press, 2009.

[36] Walter Fitch. Toward defining the course of evolution: minimum change for a specific tree topology. *Systematic Zoology*, 20(4):406–416, 1971.

[37] Guido Fritzsch, Martin Schlegel, and Peter Stadler. Alignments of mitochondrial genome arrangements: applications to metazoan phylogeny. *Journal of Theoretical Biology*, 240(4):511–520, 2006.

[38] Olivier Gascuel. BIONJ: an improved version of the NJ algorithm based on a simple model of sequence data. *Molecular Biology and Evolution*, 14(7):685–695, 1997.

[39] Maryam Haghighi and Sylvia Boyd. A fast method for large-scale multichromosomal breakpoint median problems. In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, BCB 2011, pages 1–3, 2011.

[40] Sridhar Hannenhalli and Pavel Pevzner. Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals. In *Proceedings of the 27th annual ACM symposium on Theory of computing*, STOC'95, pages 178–189, 1995.

[41] Sridhar Hannenhalli and Pavel Pevzner. Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals. *Journal of the ACM*, 46(1):1–27, 1999. Uma versão preliminar deste artigo apareceu em 1995 no STOC'95.

[42] Jon Herron. PhyloStrat — Evolution Simulation Software for Evolutionary Analysis. `http://faculty.washington.edu/herronjc/SoftwareFolder/PhyloStrat.html`. (acessado em 06 de fevereiro de 2012).

[43] Yen-Lin Huang, Chen-Cheng Huang, Chuan Yi Tang, and Chin Lung Lu. SoRT2: a tool for sorting genomes and reconstructing phylogenetic trees by reversals, generalized transpositions and translocations. *Nucleic Acids Research*, 38(Web-Server-Issue):221–227, 2010.

[44] Géraldine Jean and Macha Nikolski. Genome rearrangements: a correct algorithm for optimal capping. *Information Processing Letters*, 104(1):14–20, 2007.

[45] John Kececioglu and R. Ravi. Of mice and men: Algorithms for evolutionary distances between genomes with translocation. In *Proceedings of the 6th ACM-SIAM symposium on Discrete Algorithm*, SODA'95, pages 604–613, 1995.

[46] Jakub Kovác, Brona Brejová, and Tomás Vinar. A new approach to the Small Phylogeny Problem. Technical report, ArXiv e-prints, 2010.

[47] Ivica Letunic and Peer Bork. Interactive Tree Of Life (iTOL): an online tool for phylogenetic tree display and annotation. *Bioinformatics*, 23(1):127–128, 2007.

[48] Guojun Li, Xingqin Qi, Xiaoli Wang, and Binhai Zhu. A linear-time algorithm for computing translocation distance between signed genomes. In *Proceedings of the 15th annual symposium on Combinatorial Pattern Matching*, CPM 2004, pages 323–332, 2004.

[49] Zimao Li, Lusheng Wang, and Kaizhong Zhang. Algorithmic approaches for genome rearrangement: a review. *IEEE Transactions on Systems, Man, and Cybernetics — Part C: Applications and Reviews*, 36(5):636–648, 2006.

[50] Jian Ma, Louxin Zhang, Bernard Suh, Brian Raney, Richard Burhans, William Kent, Mathieu Blanchette, David Haussler, and Webb Miller. Reconstructing contiguous regions of an ancestral genome. *Genome Research*, 16(12):1557–1565, 2006.

[51] Charles Michener and Robert Sokal. A quantitative approach to a problem in classification. *Evolution*, 11(2):490–499, 1957.

[52] Julia Mixtacki. Genome Halving under DCJ Revisited. In *Proceedings of the 14th annual international conference on Computing and combinatorics*, COCOON'08, pages 276–286, 2008.

[53] Bernard Moret, Li-San Wang, Tandy Warnow, and Stacia Wyman. New approaches for reconstructing phylogenies from gene order data. *Bioinformatics*, 17(Suppl. 1):S165–S173, 2001.

[54] Bernard M. E. Moret, Usman Roshan, and Tandy Warnow. Sequence-length requirements for phylogenetic methods. In *Proceedings of the Second International Workshop on Algorithms in Bioinformatics*, WABI '02, pages 343–356, London, UK, UK, 2002. Springer-Verlag.

[55] Luay Nakhleh, Bernard Moret, Usman Roshan, Katherine John, Jerry Sun, and Tandy Warnow. The accuracy of fast phylogenetic methods for large datasets. In *Proceedings of the 7th Pacific Symposium on Biocomputing*, PSB'02, pages 211–222, 2002.

[56] Michal Ozery-Flato and Ron Shamir. Two notes on genome rearrangement. *Journal of Bioinformatics and Computational Biology*, 1(1):71–94, 2003.

[57] Michal Ozery-Flato and Ron Shamir. An $o(n^{\frac{3}{2}}\sqrt{\log(n)})$ algorithm for sorting by reciprocal translocations. *Journal of Discrete Algorithms*, 9(4):344–357, 2011.

[58] Itsik Pe'er and Ron Shamir. The median problems for breakpoints are NP-complete. *Electronic Colloquium on Computational Complexity*, 5(71), 1998.

[59] Pere Puigbò, Santiago Garcia-Vallvé, and James McInerney. TOPD/FMTS: a new software to compare phylogenetic trees. *Bioinformatics*, 23(12):1556–1558, 2007.

[60] William Purves, David Sadava, Gordon Orians, and Craig Heller. *Life: The Science of Biology*. Sinauer Associates and W. H. Freeman, 7 edition, 2003.

[61] The Science Creative Quarterly. Banco de imagens. `http://www.scq.ubc.ca/image-bank/`. (acessado em 06 de fevereiro de 2012).

[62] David Robinson and Leslie Foulds. Comparison of phylogenetic trees. *Mathematical Biosciences*, 53:131–147, 1981.

[63] Naruya Saitou and Masatoshi Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425, 1987.

[64] David Sankoff. Minimal Mutation Trees of Sequences. *SIAM Journal on Applied Mathematics*, 28(1):35–42, 1975.

[65] David Sankoff. Genome rearrangement with gene families. *Bioinformatics*, 15(11):909–917, 1999.

[66] David Sankoff and Mathieu Blanchette. The median problem for breakpoints in comparative genomics. In *Proceedings of the 3rd Annual International Conference on Computing and Combinatorics*, COCOON'97, pages 251–263, 1997.

[67] David Sankoff and Mathieu Blanchette. Multiple genome rearrangement and breakpoint phylogeny. *Journal of Computational Biology*, 5(3):555–570, 1998.

[68] Jian Shi, William Arndt, Fei Hu, and Jijun Tang. Isolating — a new resampling method for gene order data. In *Proceedings of the 2011 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, CIBCB'11, pages 135–140, 2011.

[69] Jian Shi and Jijun Tang. An experimental evaluation of corrected inversion and DCJ distance metric through simulations. In *Proceedings of the 4th International Conference on Bioinformatics and Biomedical Engineering*, iCBBE 2010, pages 1–4, 2010.

[70] Adam Siepel. Exact algorithms for the reversal median problem. Master's thesis, Cornell University, 2001.

[71] Jijun Tang and Bernard Moret. Scaling up accurate phylogenetic reconstruction from gene-order data. *Bioinformatics*, 19(Suppl. 1):i305–i312, 2003.

[72] Eric Tannier and Marie-France Sagot. Sorting by reversals in subquadratic time. In *Combinatorial Pattern Matching*, volume 3109 of *Lecture Notes in Computer Science*, pages 1–13. Springer Berlin / Heidelberg, 2004.

[73] Eric Tannier, Chunfang Zheng, and David Sankoff. Multichromosomal median and halving problems under different genomic distances. *BMC Bioinformatics*, 10(1):120, 2009.

[74] Glenn Tesler. Efficient algorithms for multichromosomal genome rearrangements. *Journal of Computer and System Sciences*, 65(3):587–609, 2002.

[75] Geoffrey Waterson, Warren Ewens, Tom Hall, and A. Morgan. The chromosome inversion problem. *Journal of Theoretical Biology*, 99(1):1–7, 1982.

[76] Andrew Xu. A fast and exact algorithm for the median of three problem — a graph decomposition approach. In *Proceedings of the 6th International workshop on Comparative Genomics*, RECOMB-CG 2008, pages 184–197, 2008.

[77] Andrew Xu and David Sankoff. Decompositions of multiple breakpoint graphs and rapid exact solutions to the Median Problem. In *Proceedings of the 8th international workshop on Algorithms in Bioinformatics*, WABI'08, pages 25–37. 2008.

[78] Wei Xu and Bernard Moret. GASTS: Parsimony scoring under rearrangements. In *Proceedings of the 11th international workshop on Algorithms in Bioinformatics*, WABI'11, pages 351–363, 2011.

[79] Sophia Yancopoulos, Oliver Attie, and Richard Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21(16):3340–3346, 2005.

[80] Sophia Yancopoulos and Richard Friedberg. DCJ path formulation for genome transformations which include insertions, deletions, and duplications. *Journal of Computational Biology*, 16(10):1311–1338, 2009.

[81] Feng Yue, Meng Zhang, and Jijun Tang. A heuristic for phylogenetic reconstruction using transposition. In *Proceedings of the 7th IEEE International Conference on Bioinformatics and Bioengineering*, BIBE 2007, pages 802–808, 2007.

[82] Feng Yue, Meng Zhang, and Jijun Tang. Phylogenetic reconstruction from transpositions. *BMC genomics*, 9(Suppl. 2):S15, 2008.

[83] Udny Yule. A mathematical theory of evolution, based on the conclusions of Dr. J. C. Willis, F.R.S. *Philosophical Transactions of the Royal Society of London. Series B, Containing Papers of a Biological Character*, 213(402–410):21–87, 1924.