

第1章 引言

1.1 研究背景

生物信息学（bioinformatics）是生物学与计算机科学以及应用数学等学科相互交叉而形成的一门新兴学科。它通过对生物学实验数据的获取、加工、存储、检索与分析，进而达到揭示数据所蕴含的生物学意义的目的。它涉及生物学、数学、计算机科学和工程学，依赖于计算机科学、工程学和应用数学的基础，依赖于生物实验和衍生数据的大量储存。生物信息学不只是一门为了建立、更新生物数据库及获取生物数据而联合使用多项计算机科学技术的应用性学科，也不仅仅是只限于生物信息学这一概念的理论性学科。事实上，它是一门理论概念与实践应用并重的学科。过去二十年，随着人类基因组工程的实施和深入，生物学数据获得前所未有的增加，数据的内容也从生理生化数据向遗传、结构、功能及其相互关系等数据发展。同一时期，计算机微处理器芯片、半导体存储器和系统软件也在按照指数方式增长。如何有效利用组合学、统计学等数学方法和现代计算机的强大计算能力，从这些生物学数据中提取有用的知识、发现重大的科学规律，成为生物学家、数学家、计算机学家们面临的巨大挑战，从而导致了计算生物学、生物信息学的产生和发展。

计算进化生物学是生物信息学的主要研究方向之一。进化生物学研究物种的起源和演化。引入信息学到进化生物学中，使得研究者能够：

- 通过度量DNA序列的改变研究众多生物体间的进化关系（超越了以前基于身体和生理特征观察的研究方法）
- 通过整个基因组的比对，研究更为复杂的进化论课题，如基因复制，基因水平转移等
- 为种群进化建立复杂的计算模型，以预测种群随时间的演化
- 保存大量物种的遗传信息

未来的研究工作包括重建现已相当复杂的进化树，并进行相关的研究。

1.2 研究意义

进化生物学在生物学中是一个非常重要的研究领域，其中进化树，又称系统发生树，是用来研究一群物种进化历史的标准模型，是进化生物学中不可缺少的重要工具。由于生物进化中可能产生的杂交、基因水平转移、基因重组等网状事件，导致一个物种的基因可能来源于多个祖先。多年来对生物进化历史的研究发现，杂交事件分别在植物、鸟类、鱼类的种群中均有发生。自然的杂交事件在哺乳动物，甚至是灵长类动物中都被发现过。研究显示，25%的植物和10%的动物，尤其是年轻的物种，都与杂交事件相关。

已有许多方法能够从一组物种中构建起他们对应的进化树，但由于进化过程中网络事件的存在，一组相同物种基于不同基因的分析可能产生不同的进化树，反之，通过对比这些进化树的相似性是帮助人们发现这些网络事件的重要手段。许多计算生物学家都对此问题非常感兴趣，他们常用的衡量标准有子树剪切再接距离（subtree prune and regraft distance, rSPR）和杂交数（hybridization number, HN）两种。Baroni等人证明了rSPR距离为进化过程中网状事件的数目给出了一个下限。^[2]而对这些衡量标准的计算可以很好地抽象为具体的数学问题，因此借助计算机来研究进化树具有十分重要的意义。

1.3 固定参数算法概述

对于NP完全问题（NP难问题），因为当问题的规模增长时，算法执行所需的时间会随着问题的规模呈指数级增长，所以寻找一种有效的精确算法被广泛认为是不可能的。然而许多问题可以得到一个时间复杂度仅随问题的某个固定参数呈指数增长的算法。当问题在 $O(f(k) \cdot n^c)$ 的时间内可解， $f(k)$ 是一个与 n 无关且 c 为常数时便属于固定参数可解类（fixed parameter tractable, FPT）。如果 k 远小于问题规模 n ，那么我们便可以获得该问题一个相对更有效的精确算法。例如，对于一个常见的NP问题——顶点覆盖问题，存在一个复杂度为 $O(kn + 1.274^k)$ 的固定参数算法^[2]，其中 n 是顶点总数， k 是顶点覆盖的大小。这就意味着顶点覆盖问题可以用该问题的解参数化。本文也是根据类似的思路利用固定参数算法来研究进化树的对比问题。

1.4 论文组织架构

本文采用了如下的结构：

第一章：引言。本章主要介绍本论文课题的研究背景，研究意义和主要的研究方法，最后对论文的章节进行了一个合理的逻辑安排。

第二章：问题建模和描述。本章先介绍本论文研究过程中所使用到的相关概念，包括进化树的定义和相关操作的定义，然后对所研究的问题给出了具体的形式化描述，最后简单介绍该问题的研究现状。

第三章：算法思想与设计。本章先介绍Whidden的固定参数算法，随后给出本文基于此算法的改进思路和具体方法和步骤。

第四章：数据结构设计及算法实现。本章给出改进后的固定参数算法的具体实现方法，包括数据结构设计以及具体的算法实现。

第五章：实验分析。本章通过随机生成的数据集和真实的数据集对算法进行实验分析，验证了其改进效果。

第六章：总结及展望。本章对本文研究的课题进行了归纳和总结，指明了改进和完善的方向。

第2章 问题建模和描述

2.1 基本定义

本节将给出在研究进化树结构的过程中所需要的一些重要概念的定义。

定义 2.1.1 有根二叉进化树（简称进化树）是一棵叶节点被集合 X 中的元素所标记的二叉树，满足除叶节点没有子节点外，其余节点都有且仅有两个子节点，记作 T 。将 T 的所有边的集合记为 E_T 。将 X 称作该进化树的标识集。

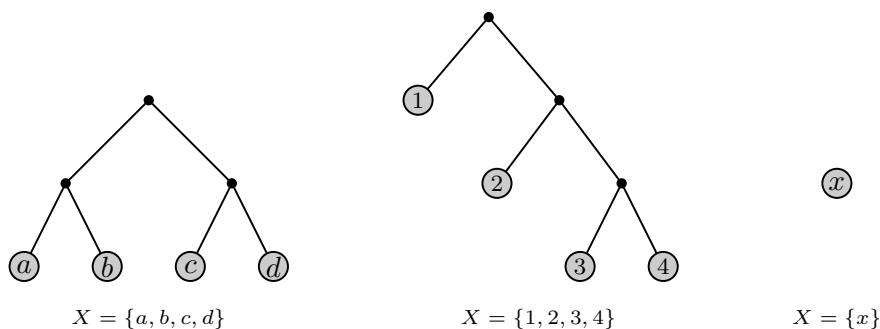


图 2-1 有根二叉进化树示例

定义 2.1.2 如果两棵进化树具有相同的标识集，并且同构，则认为这两棵进化树相等，记作 $T_1 = T_2$ 。

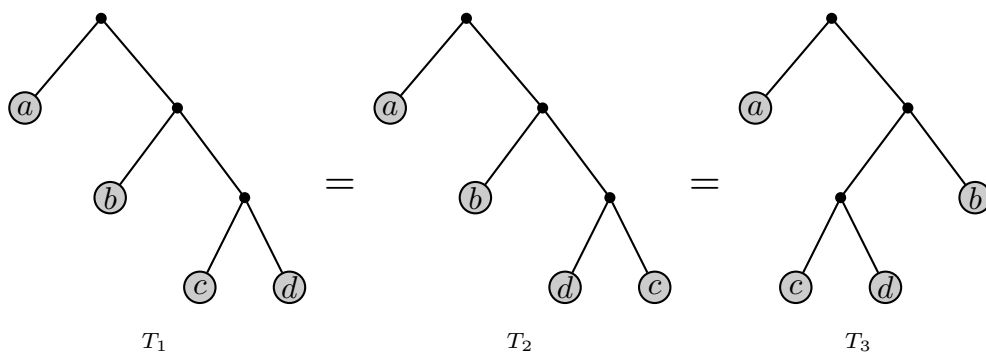


图 2-2 相等的3棵进化树

定义 2.1.3 将一棵二叉树删去所有只有一个子节点的内部节点以及所有未被标记的叶节点，使其变成一棵进化树的操作称为**收缩**。

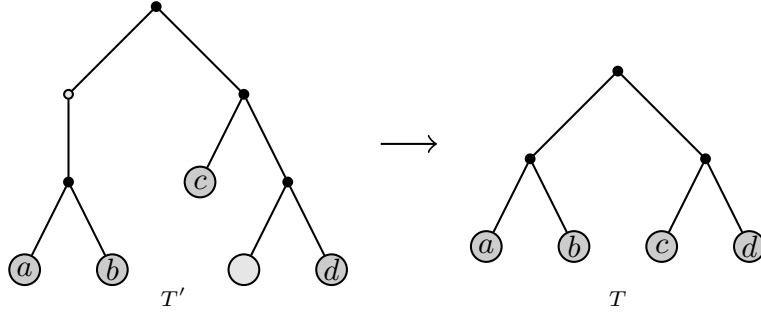


图 2-3 将 T' 收缩得到 T

定义 2.1.4 子树剪切再接 (rooted subtree prune and regraft, 简称rSPR) 对于一棵进化树 T ，剪去任意节点 x 的父边 e_x 得到一棵以 x 为根的子树 t_x ，将 t_x 嫁接到余下子树 $T - t_x$ 的一条边上，并对操作后的树进行一次**收缩**，获得一棵新的进化树，该过程称为一次**rSPR**操作。对于两棵进化树 T_1, T_2 ，将其中一棵树转化为另一棵所需的最少rSPR操作次数称为 T_1, T_2 的**rSPR**距离，记作 $d(T_1, T_2)$ 。

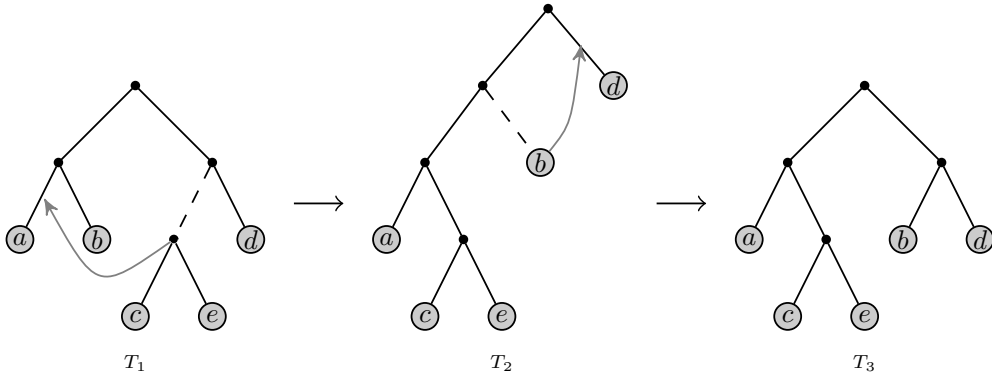


图 2-4 将 T_1 转化为 T_3 需要至少2次SPR操作， $d(T_1, T_3) = 2$

定义 2.1.5 进化树森林 (简称森林) 是由若干棵进化树组成的集合。记作 $F = \{t_1, t_2, \dots, t_n\}$ 。将一棵进化树 T 删去若干条边 E ，并对每棵独立的子树进行一次收缩操作得到的森林 F ，记作 $F = T - E$ 。同理，将一个森林 F 删去若干条边 E ，并对每棵独立的子树进行一次收缩操作得到的森林 F' ，记作 $F' = F - E$ 。

定义 2.1.6 最大一致森林 (maximum-agreement forest, 简称MAF) 对于两棵进化树 T_1, T_2 , 若存在 $E_1 \subset E_{T_1}$, $E_2 \subset E_{T_2}$, 使得 $F = T_1 - E_1$ 并且 $F = T_2 - E_2$, 我们称 F 为 T_1, T_2 的一致森林。将 T_1, T_2 的所有一致森林中, 包含最少进化树个数的森林称为最大一致森林, 其包含的进化树个数记为 $m(T_1, T_2)$ 。

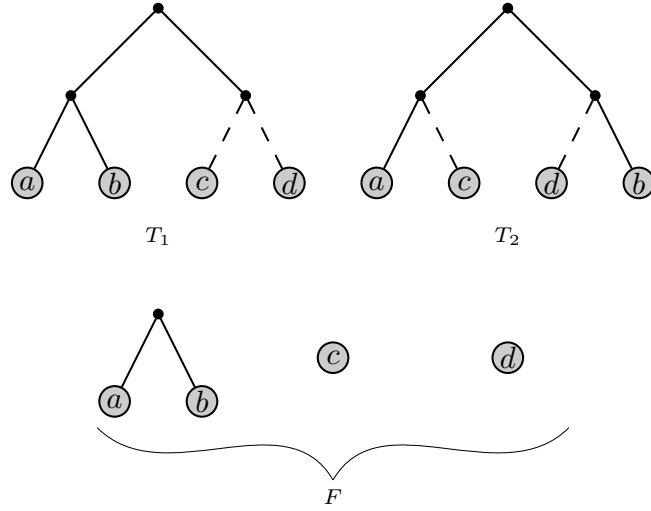


图 2-5 将 T_1 和 T_2 删去对应边后获得最大一致森林 F , $m(T_1, T_2) = 3$

Bordewich和Semple^[2]已经指出两棵进化树的rSPR距离与MAF的大小存在着等价关系, 他们证明了如下定理:

定理 2.1.1 对于两棵具有相同标识集 X 的进化树 T_1, T_2 , 存在

$$d(T_1, T_2) = m(T_1, T_2) - 1$$

2.2 问题描述

根据定理??我们可以将求解两棵进化树的rSPR距离转化为求解它们的最大一致森林的最优化问题, $d(T_1, T_2)$ 也可等价于将 T_1, T_2 转化为它们的MAF所需要删除的最少边数。而因为该问题是NP完全问题, 我们可以将其进一步转化为可以用固定参数算法求解的判定性问题, 问题的具体描述如下:

给定两个具有相同标识集 X 的进化树 T_1, T_2 , 以及一个参数 k 。

判断 $d(T_1, T_2) \leq k$ 是否成立?

2.3 相关研究

尽管rSPR距离在生物学上具有十分重要的意义，但计算它却被证明是NP难的。因此研究针对该问题的近似算法，参数算法以及一些启发式算法成为了计算rSPR距离的主要手段。在介绍本文提出的参数算法之前，先回顾之前针对此问题各种算法的研究进展。

2.3.1 近似算法

自从Jotun Hein等人在[?]中引入MAF的概念以来，MAF已经成为解决该问题最有效的工具，不少以此为基础的近似算法得以提出。Hein首先在[?]中提出了一个近似比为3的近似算法，但却被Rodrigues在[?]中证明在某些情况下其近似比不可能小于4。同时，Rodrigues也提出了一些修正，宣称修正后的算法的近似比为3。然而，Bonet在[?]中给出的反例证明了[?]和[?]中的算法近似比实际上均为5，并且可以在线性时间复杂度下实现。Bordewich在[?]中提出了一个正确的近似比为3的算法，然而是以时间复杂度增加到 $O(n^5)$ 为代价^①。在[?]中提出的第二个近似比为3的算法的时间复杂度为 $O(n^2)$ 。最终，线性时间复杂度下近似比为3的算法由Whidden在[?]中提出。

2.3.2 固定参数算法

在生物进化过程中网状事件发生的次数远远小于物种的数量，因此固定参数算法往往是计算rSPR距离精确值的最好途径。将两棵进化树的rSPR距离作为参数 k 的参数算法便是一个很有前景的研究方法。以此为基础，Bordewich在[?]中提出了一个时间复杂度为 $O(4^k \cdot k^5 + n^3)$ 参数算法。之后，Hallett, Allen, Bonet等人针对该问题的相关问题也有许多有意义的研究。^{[2][12][13]} Whidden在[?]中提出的算法可以说是该问题在参数算法上的一个突破，该算法的时间复杂度为 $O(2.42^k n)$ ，并且从实验结果可以看出其性能远优于先前的算法，有能力处理具有更多节点数更大rSPR距离的进化树对比问题。随后，Chen和Wang在[?]中对Whidden的算法中最差的情况进行了许多改进，最后获得了一个时间复杂度为 $O(2.34^k n)$ 的参数算法，但其改进方法分类较多，比较复杂，因此算法实现难度很大。本文也是基于相同的思想，但使用相对更简单的方法得到了更好的改进效果。

① 适当地改进数据结构可以将时间复杂度降低到 $O(n^4)$

2.3.3 启发式算法

许多关于SPR距离的启发式算法在近年也得以提出，并开发了相关的软件。其中，Hallett和Lagergen开发的程序LatTrans^[2]针对一些特殊的rSPR操作进行建模，只考虑进化树只可能在两种情况下相异，在这种特殊的条件下，它的时间复杂度可以减少到 $O(2^k n^2)$ 。Macleod开发的HorizStory^[2]可以计算多叉树的SPR距离，但只考虑SPR操作对象是只有一个叶节点的子树。SPRdist^[2]和TreeSAT^[2]是两个计算rSPR距离精确值的软件，他们分别把计算MAF的问题转化为整数线性规划问题（integer linear programming, ILP）和可满足性问题（satisfiability problem, SAT），然后利用求解对应问题的有效手段来获得rSPR距离的解。但根据实验结果，它们的性能均不能与Whidden所提出的算法相比。^[2]

第3章 算法思想与设计

本文的算法与[?]中的相似，都是基于Whidden的参数算法^[2]改进而来。因此，在这里有必要先简单介绍Whidden算法的内容，并给出简要的时间复杂度分析。之后，会详细介绍本文的改进思路，并给出详细的方法和步骤。

3.1 Whidden的参数算法^[2]

3.1.1 算法概述

Whidden的算法所解决的正是本文在第??节中所提出的问题。设 $MAF(T_1, F_2, k)$ 为针对此问题的判定函数， T_1 代表第一棵进化树， F_2 代表 T_2 删除某个边集后所得的森林， k 是一个非负整数。若 F_2 能够在删除至多 k 条边后变成 T_1, T_2 的最大一致森林，则 $MAF(T_1, F_2, k)$ 返回 $true$ ，否则返回 $false$ 。初始时， $F_2 = T_2$ 。求解rSPR距离，只需要从0开始枚举 k 的值，直到 $MAF(T_1, T_2, k)$ 返回 $true$ 。因为时间复杂度是关于 k 的指数，所以相对于计算 $MAF(T_1, T_2, k)$ ，计算rSPR距离只会在时间复杂度的常数上有所增加。算法采用递归的思想， MAF 函数的具体步骤如下：

1. 如果 $k < 0$ ，返回 $false$
2. 如果在 T_1 中存在一对兄弟叶节点 a, b ，并且它们在 F_2 中的对应节点也是兄弟叶节点，那么就合并 a, b ，然后把它们的在 T_1, F_2 中的父节点作为对应的叶节点。重复此步骤，直至没有节点可以合并。
3. 如果在 F_2 中存在只有一个节点的子树 x ，则将 x 从 T_1, F_2 中移除。重复此步骤，直至没有节点可以移除。此时如果产生新的可以合并的兄弟叶节点，则转至??，否则继续。
4. 如果 F_2 为空，返回 $true$
5. 任意选择 T_1 中的一对兄弟叶节点 a, b （注意到此时 a, b 在 F_2 中一定不是兄弟叶节点），分三种情况讨论（如图??）：

- (a) Case 1: 若 a, b 在 F_2 中属于不同的连通分量。可以证明^[2]， a 的父边 e_a 和 b 的父边 e_b 两条边中至少有一条需要删除。对于两种情况下修改

后的 $F'_2 = F_2 - \{e_a\}$ 和 $F'_2 = F_2 - \{e_b\}$ ，分别调用两次 $MAF(T_1, F'_2, k-1)$ 。只要有任意一次结果为 $true$ ，则返回 $true$ ，否则返回 $false$ 。

(b) Case 2: 若 a, b 在 F_2 中连通，并且 a, b 之间有且只有一个悬挂节点 p 。可以证明^[3]，一定存在最优解 $E \subset E_{T_2}$ ，使得 $T_2 - E$ 是 T_1, T_2 的MAF，并且 $e_p \in E$ 。因此，只需要直接修改 F_2 ，得到 $F'_2 = F_2 - \{e_p\}$ 。若 $MAF(T_1, F'_2, k-1)$ 结果为 $true$ ，则返回 $true$ ，否则返回 $false$ 。

(c) Case 3: 若 a, b 在 F_2 中连通，并且 a, b 之间有至少两个悬挂节点。设 a, b 之间所有悬挂节点的集合为 $P = \{p_1, p_2, \dots, p_m\}, m \geq 2$ 。可以证明^[3]，一定存在最优解 $E \subset E_{T_2}$ ，使得 $T_2 - E$ 是 T_1, T_2 的MAF，并且 $e_a \in E$ 或者 $e_b \in E$ 或者对于所有 $1 \leq i \leq m$ ，有 $e_{p_i} \in E$ 。因此，分别修改 F_2 得到 $F'_2 = F_2 - \{e_a\}$ 、 $F'_2 = F_2 - \{e_b\}$ 和 $F'_2 = F_2 - E_P$ （ E_P 为 P 中所有节点对应的父边的集合），然后分别调用两次 $MAF(T_1, F'_2, k-1)$ 和一次 $MAF(T_1, F'_2, k-m)$ 。只要有任意一次结果为 $true$ ，则返回 $true$ ，否则返回 $false$ 。

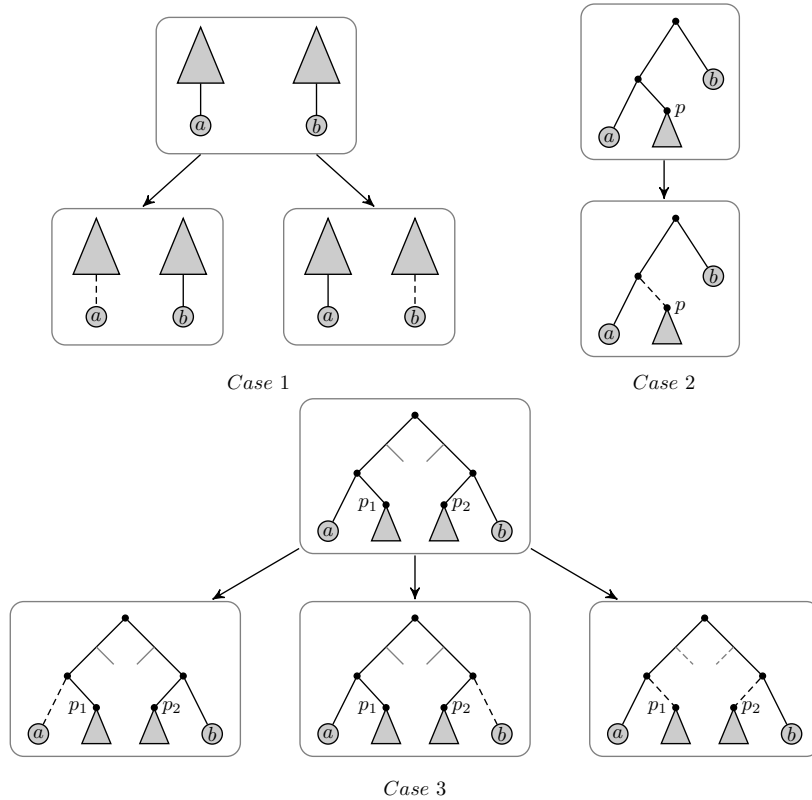


图 3-1 Whidden算法第??步的三种情况

3.1.2 时间复杂度分析

不难看出，适当地设计数据结构，Whidden算法中前四步可以在 $O(n)$ 的时间内完成。因此，我们关键需要分析搜索树的大小，而搜索树的大小与参数 k 有直接关系。设 $C(k)$ 为参数为 k 时最坏情况下搜索树的大小。根据第??步中的三种情况，我们可以得到下面三个不等式：

1. 若 a, b 在 F_2 中属于不同的连通分量。

$$C(k) \leq 2C(k-1)$$

2. 若 a, b 在 F_2 中连通，并且 a, b 之间有且只有一个悬挂节点。

$$C(k) \leq C(k-1)$$

3. 若 a, b 在 F_2 中连通，并且 a, b 之间有至少两个悬挂节点。

$$C(k) \leq 2C(k-1) + C(k-m), m \geq 2$$

可以看出， $C(k)$ 的最坏情况出现在第3种情况 $m=2$ 时，此时有

$$C(k) \leq 2C(k-1) + C(k-2)$$

设 $C(k) = \alpha^k$ ，带入可得

$$1 = 2\alpha^{-1} + \alpha^{-2}$$

解得 $\alpha = 1 + \sqrt{2} \approx 2.42$ ，因此Whidden参数算法复杂度为 $O(2.42^k n)$ 。Whidden的算法最大贡献在于发现了Case 2中当 a, b 直接只有1个悬挂节点便可以直接删除其父边的规律。而该算法的限制则在于Case 3，本文正是试图寻找方法改进Case 3的最坏情况，具体内容将在下一节讨论。

3.2 改进思路

显然，当叶节点个数 n 小于3时，两棵进化树的rSPR距离一定是0，因此在后面的讨论中，我们假设 $n \geq 3$ 。为了改进Whidden的算法，我们试图在Whidden所利用的两个兄弟叶节点的基础之上加入更多的节点，以此获取更多的信息用以改进算法。一种自然的想法是在 T_1 中寻找形如图??中(a)的结构。但(a)所表示的结构并非在所有进化树中都会出现，例如一棵叶节点个数为4的满二叉树。然而可以证明，若(a)中的结构不存在，一定存在形如图??中(b)所表示的结构。

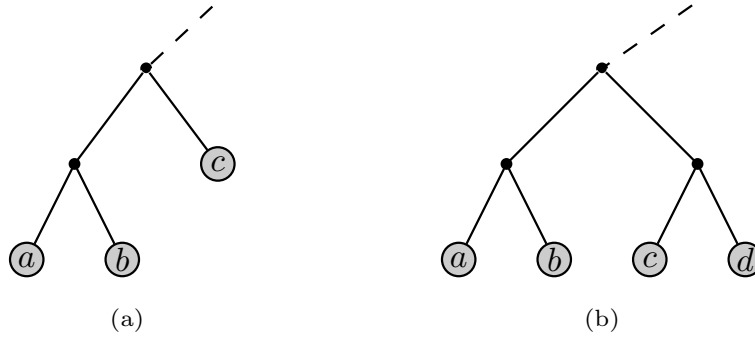


图 3-2 两种结构

引理 3.2.1 对于任意一棵叶节点数大于3的进化树 T ，存在下面至少一种情况：

- 存在 T 的三个叶节点 a, b, c ，满足 a, b 是兄弟节点，并且 a, b 的父节点与 c 是兄弟节点（图?? (a)）
- 存在 T 的四个叶节点 a, b, c, d ，满足 a, b 是兄弟节点， c, d 是兄弟节点，并且 a, b 的父节点与 c, d 的父节点是兄弟节点（图?? (b)）

证明：不妨设 a 是离 T 的根节点最远的叶节点，根据进化树的定义，一定存在 a 的兄弟节点 b 。（图?? (a)）设 p 为 a, b 的父节点的兄弟节点，因为 a 离根节点的距离最远，那么以 p 为根节点的子树 T_p 只有两种可能。

- p 为叶节点，如图?? (b)，此时出现引理??中第一种情况
- p 不为叶节点，则 p 一定有且只有两个叶节点，如图?? (c)，此时出现引理??中第二种情况

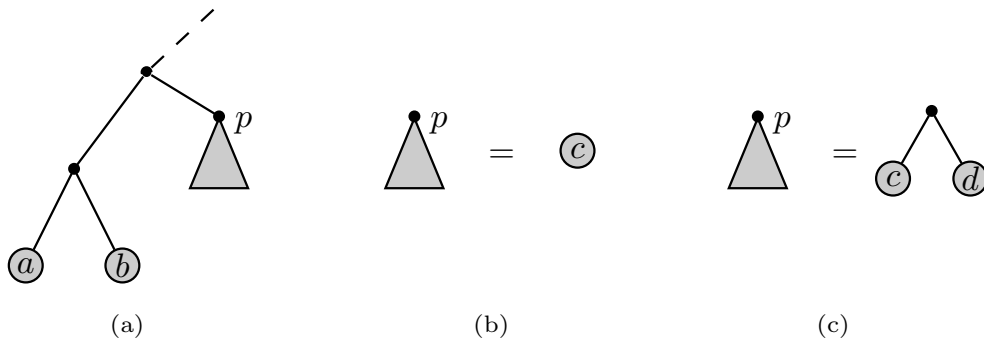


图 3-3 引理??证明

于是，基于引理??我们可以改进Whidden算法的第??步。

3.3 改进算法

本文改进算法的前四步与第??节中叙述的完全一致，主要针对第??步进行优化。为了方便叙述，我们定义 $\rho(a,b)$ 等于从节点 a 到节点 b 的路径上的边数减2，即节点 a,b 之间悬挂节点的个数。

在详细叙述本文的改进方法之前，我们再明确一下当 T_1, F_2 经过前四步的处理后所具有的两点性质：

1. T_1 中的任意一对兄弟叶节点 a,b 在 F_2 中一定不是兄弟节点。换句话说， a,b 在 F_2 中要么不连通，要么连通但 $\rho(a,b) \geq 1$ 。
2. T_1, F_2 中的任意一个叶节点 x 都不可能是某棵树的根节点。这意味着任意叶节点 x 的父边 e_x 一定存在。

改进算法的第五步首先是在 T_1 中寻找图?? (a)所表示的结构，这可以通过遍历 T_1 在最多 $O(n)$ 的时间内完成。接下来，我们根据子树 T_p 的两种情况分类讨论。

3.3.1 当 p 为叶节点

Case 1: 当 p 为叶节点时，我们得到 T_1 中的3个叶节点 a,b,c ，使其结构如图??。此时，根据 a,b,c 在 F_2 中的不同形态，我们采取不同的策略。

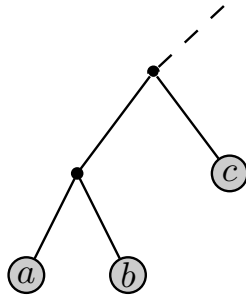


图 3-4 T_1 中的 a, b, c

Case 1.1: 若 a,b 在 F_2 中属于不同的连通分量，此时出现Whidden算法第??步中的情况??，用相同方法处理。

Case 1.2: 若 a,b 在 F_2 中属于相同的连通分量，但 $\rho(a,b) = 1$ 时，此时出现Whidden算法第??步中的情况??，也用相同方法处理。

Case 1.3: 若 a, b 在 F_2 中属于相同的连通分量, 且满足 $\rho(a, b) \geq 2$, 但 a, b 与 c 不属于同一连通分量 (如图??)。

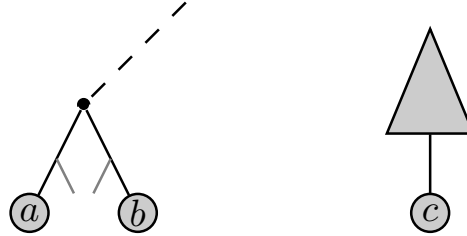


图 3-5 *Case 1.3*中的 a, b, c

3.3.2 当 p 不为叶节点