第1章 引言

1.1 研究背景

分子系统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，是分子系统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。

1.2 研究意义

1.3 近似算法

1.4 参数算法

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中的元素所标记的满二叉树，即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**。

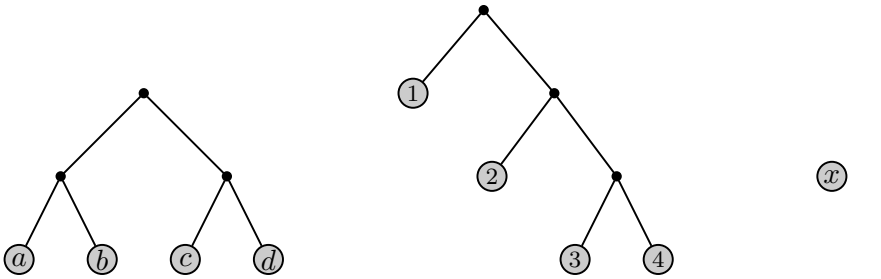

图 2–1 有根二叉进化树示例

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$T_1 = T_2$。

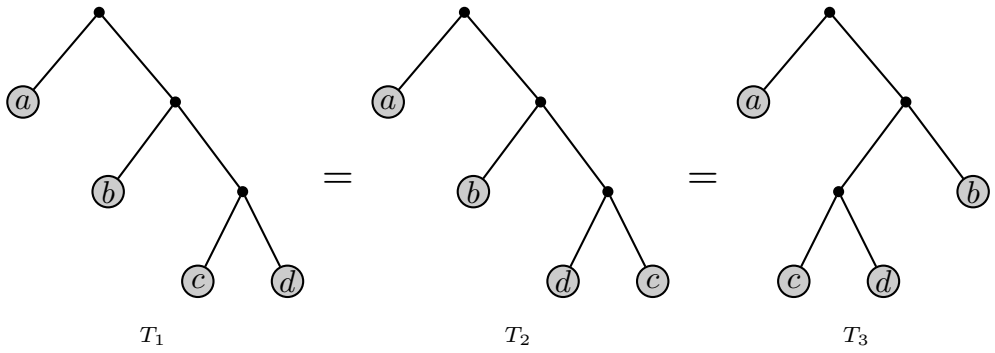

图 2–2 相等的3棵进化树

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**。

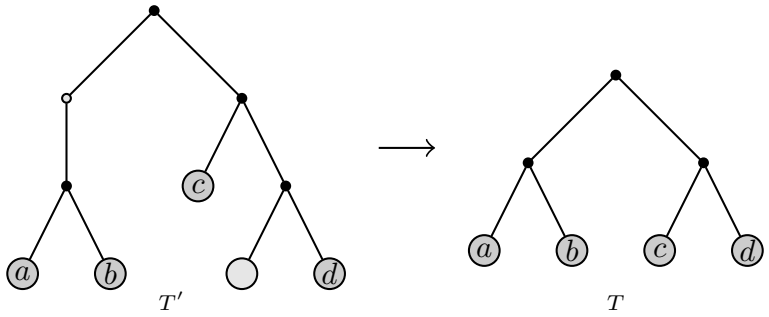

图 2–3 将T'收缩得到T

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 $d(T_1,T_2)$。

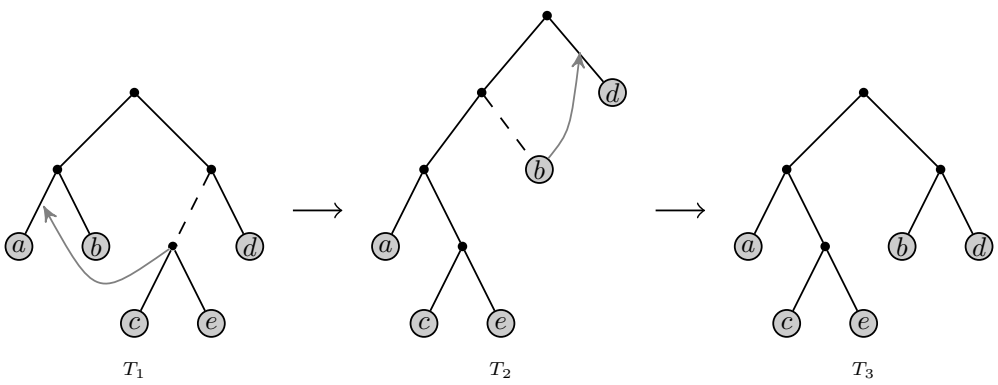

图 2–4 将T_1转化为T_3需要至少2次SPR操作，$d(T_1,T_3)=2$

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$F=\{t_1,t_2,...,t_n\}$。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$m(T_1, T_2)$。

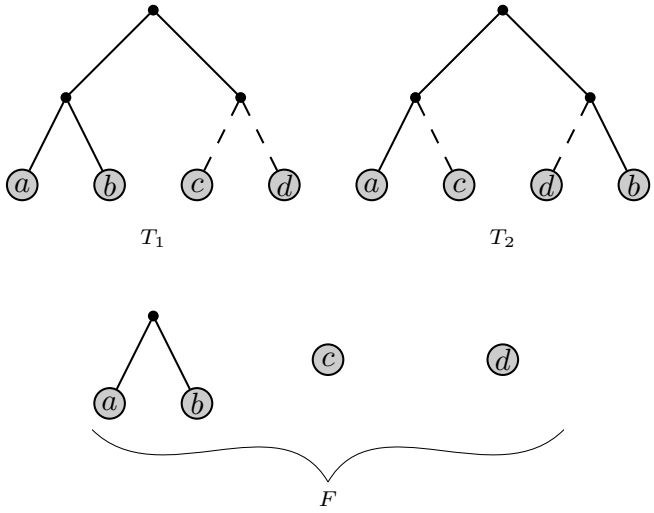

图 2–5 将T_1和T_2删去对应边后获得最大一致森林F，$m(T_1, T_2) = 3$

Bordewich和Semple[?]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，$d(T_1, T_2)$也可等价为将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 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。[?][?][?]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[?]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[?]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[?]和TreeSAT[?]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。[?]

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[?]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[?]

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[?]，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[?]，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$P = \{p_1, p_2, ..., p_m\}, m \geq 2$。可以证明[?]，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$false$。

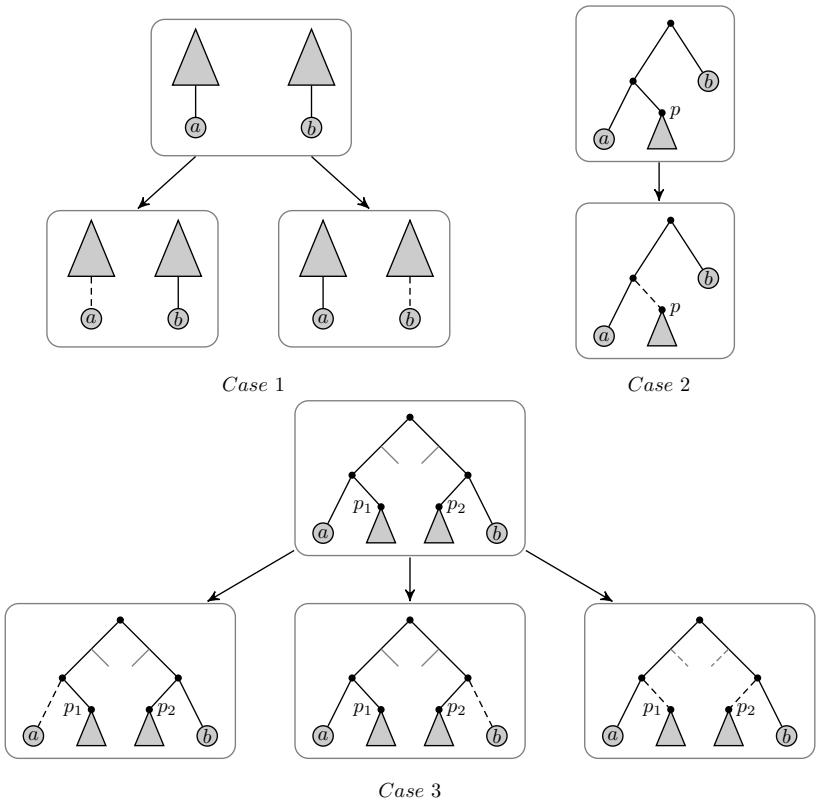

图 3–1 Whidden算法第**??**步的三种情况

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 详细步骤