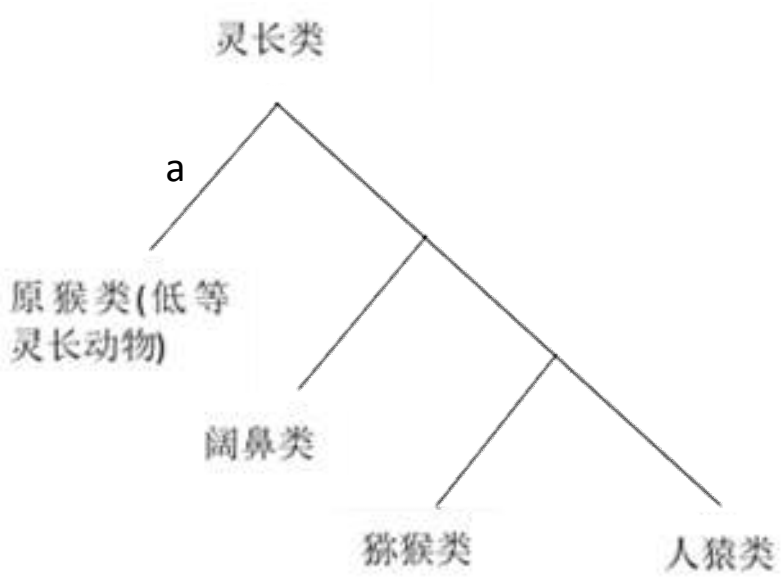
进化树的重建算法及其并行化

汇报人:张谋钊

进化树



分子序列：ATTCGCACG（举例）

每个位点可能状态：

A,C,T,G,U(RNA专用)

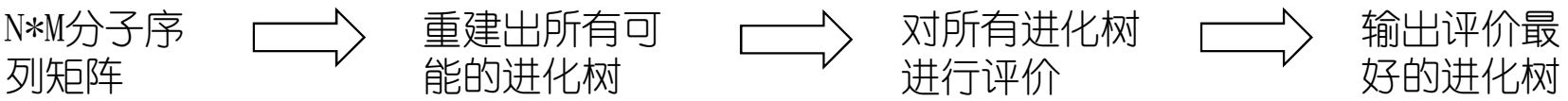
进化树包含信息：

1.分子序列之间的拓扑结构

2.分枝长度（进化距离）

进化树的重建方法

基于最优原则：



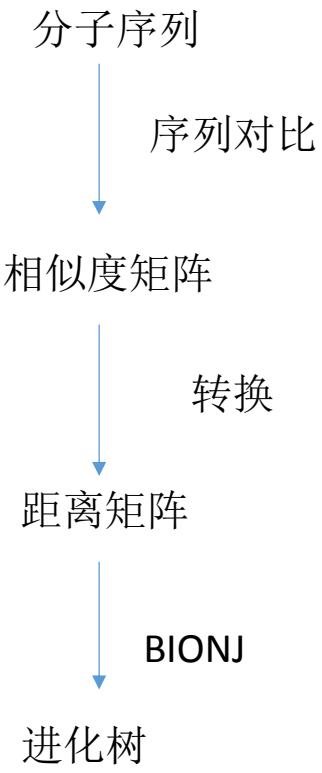
最大简约法

变异事件最少的树最接近真实，推测树内部节点最有可能的状态来计算进化过程中的最小替代数总和

最大似然法

对于由一组序列重建的进化树，其所有节点的所有的可能状态的似然值乘积最高的那一棵就是最接近真实的进化树。

进化树的重建方法--距离法



缺陷：当序列之间差异较大时，距离法中的距离矩阵就不能很明显反应序列之间的差异，往往会丢失重要的信息。距离矩阵是距离法最重要的信息量，所以距离法只有在满足一定条件下才能得到较好的解。

进化树的重建方法—总结

- 基于最优原则的**最大简约法**和**最大似然法**需要列出**所有**可能情况

$$|T| = \prod_{i=1}^{n} (2i-5) = \frac{(2n-5)!}{2^{n-3}(n-3)!}.$$

（属于NP难问题）

- 单独采取任何一种方法都无法得到最真实的进化树，基于的基本思想有一定片面性

将多目标进化算法应用于进化树重建

- 将最大简约法、最大似然法作为目标函数
- 距离法用于进化树重建(BIONJ)
- 多目标进化算法：NSGA2、MOEA/D、SPEA
- 采用NSGA2(Non-Dominated Sorting Genetic Algorithm-II)：
    - 快速非支配排序算法
    - 拥挤度和拥挤度比较算子

NSGA2重建进化树伪代码

Algorithm 2. NSGA-II Pseudocode

Input: *maxGenerations* (number of generations),
popSize (number of individuals in the population),
crossoverProb (crossover probability),
mutationProb (mutation probability),
dataset (input alignment).

Output: *ParetoFront* (non-dominated solutions found by the algorithm).

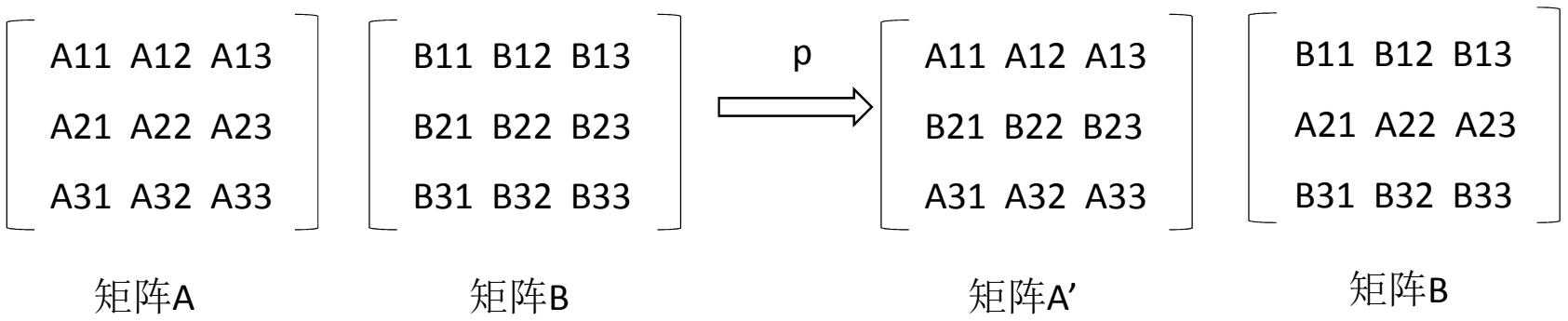
1: $C, Q, ParetoFront \leftarrow 0$
2: $P \leftarrow$ initializePopulation(*popSize*, *dataset*)
3: **for** $i = 1$ to *maxGenerations* **do**
4: /* *Population management* */
5: $C \leftarrow P \cup Q$
6: *Fronts* $\leftarrow$ fastNonDominatedSort(C)
7: $Fronts_j \leftarrow$ crowdingSorting($Fronts_j$) /*$\forall j$: *j=1 to|Fronts|* */
8: $P \leftarrow 0, j \leftarrow 1$
9: **while** $|P| + |Fronts_j| < popSize$ **do**
10: $P \leftarrow P \cup Fronts_j$
11: $j \leftarrow j + 1$
12: **end while**
13: $P \leftarrow P \cup Fronts_j$ [1:(*popSize*-$|P|$)]
14: /* *Generating the offspring population* */
15: **for** $j = 1$ to *popSize* **do**
16: $Q_j.M \leftarrow$ applyGeneticOperators(P, *crossoverProb*, *mutationProb*)
17: $Q_j.\tau \leftarrow$ buildPhylogeneticTree($Q_j.M$, *dataset*)
18: Q_j .*Scores* $\leftarrow$ evaluateTree($Q_j.\tau$, *dataset*)
19: **end for**
20: *ParetoFront* $\leftarrow$ updateParetoFront(P, Q, *ParetoFront*)
21: **end for**

遗传操作

基于距离矩阵

- Crossover：对于父矩阵A和B，A中每行都以0-1的随机数为概率进行与B的对应行交换，交换完的两个新矩阵作为子代矩阵
- 变异：从矩阵中随机选择元素进行服从gamma分布的变异

*举例：给定矩阵B，对矩阵A的第二行, 以p概率变成矩阵A’



BIONJ

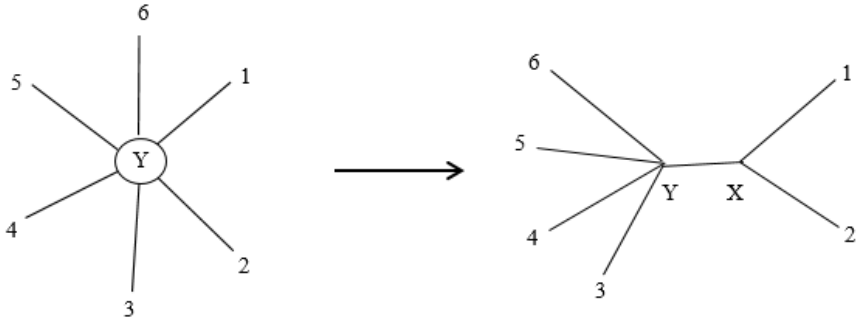
Minimun-evolution(ME)

⇩

Neighbor-joining(NJ)

⇩

BIONJ



- ME的思想：最小总枝长的树最优
- NJ:遵循ME思想，采用迭代聚合方法，生成最小总枝长树

NJ生成树的大概步骤：

1. 初始化：将所有节点与一个新建节点Y相连
2. 根据距离矩阵，从与Y相连的节点中选出两个合并后能够让总枝长最小的节点
3. 生成新节点x作为这对节点的父节点，并让X与Y相连
4. 缩小距离矩阵：计算X到其他节点的距离，将X加入矩阵来替代第2步产生的两个taxa（矩阵减少一维）
5. 重复2-4步骤，直到剩下三个节点，及距离矩阵的维度为3

进化树的变异操作

- NNI

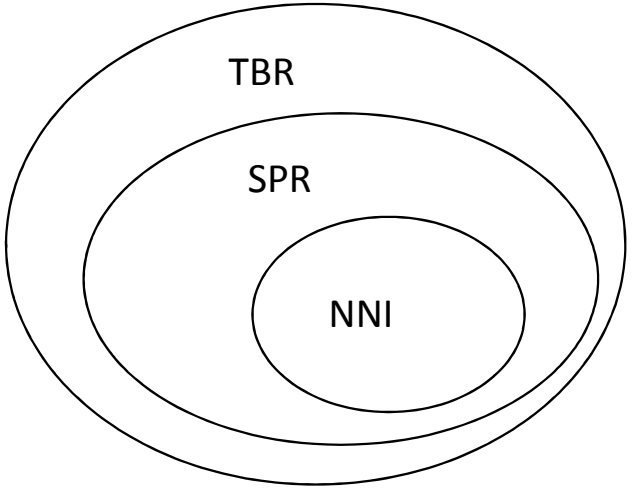
NNI (*Nearest Neighbor Interchange*) [37] consists of swapping two adjacent branches of the tree. We can say that NNI is a small size neighborhood for which a tree of n leaves has $(2n-6)$ neighbors [30].

- SPR

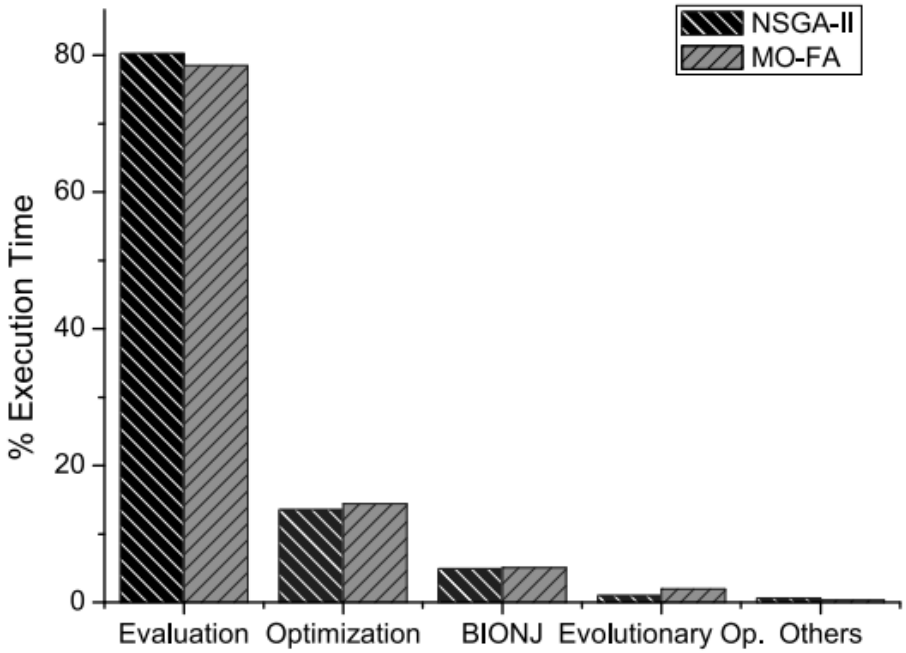
SPR (*Subtree Pruning Regrafting*) [35] is a strategy that cuts a branch and reinserts it elsewhere. There are $2(n-3)(2n-7)$ possible SPR rearrangements [2] for each tree that makes it a medium size neighborhood.

- TBR

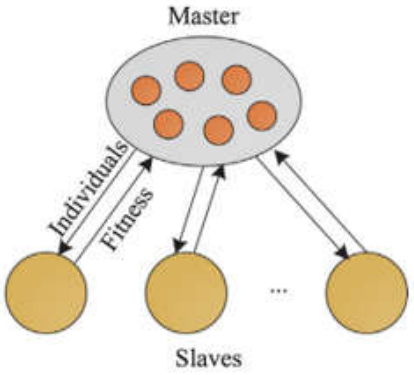
TBR (*Tree-Bisection-Reconnection*) [35] consists of breaking the tree in two subtrees that will be reconnected from one of their branches. From a given tree, the TBR neighborhood induces at most $(2n-3)(n-3)^2$ neighbor trees [2]. TBR is thus a much larger neighborhood than NNI and SPR.



并行化



- Master-slave (主从模型)



将耗资源的操作放在slave节点进行并行

并行化伪代码

主节点

Algorithm 5. NSGA-II Master Tasks

```
1: #pragma omp single
2: /* Population management */
3: C ← P ∪ Q
4: Fronts ← sortCombinedPopulation(C)
5: P ← updateParentPopulation (Fronts)
6: #pragma omp for schedule (scheduleType) /* Offspring
   generation loop */
7: for k = 1 to popSize do
8:    Q_k.M ← applyGeneticOperators(P,
      crossoverProb, mutationProb)
9: end for
```

从节点

```
1: #pragma omp for schedule (scheduleType)
2: for k = 1 to tasksNumber/nproc do
3:    Pop_k.τ ← buildPhylogeneticTree(Pop_k.M,
      dataset_idThread)
4:    Pop_k.Scores ← evaluateTree(Pop_k.τ, dataset_idThread)
5: end for
```

实验

实验平台：

- 多核smp集群架构，总共6个台机器，每台两个4核CPU组成

并行通讯工具：

- openMP（基于共享内存）
- MPI（基于消息传递）

通讯分为两类：

- 内部节点通讯
  处于同一机器内的不同CPU，采用openMP
- 外部节点通讯
  不同机器的不同CPU，采用MPI

实验数据：

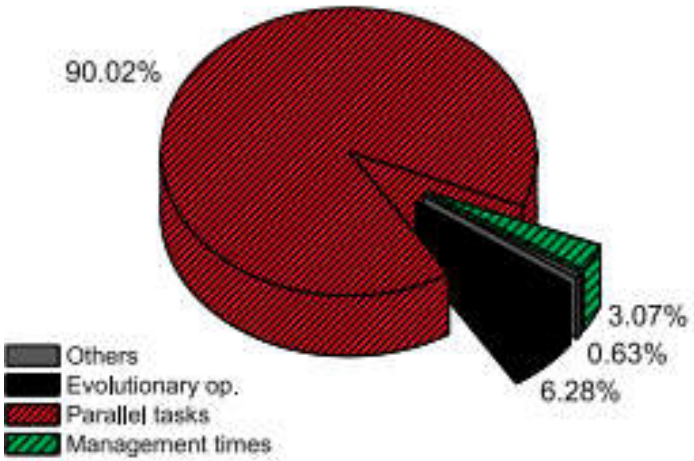
Phylogenetic Data Sets

Data set	Number of sequences	Sequence length	Description
rbcL_55	55	1,314	rbcL plastid gene [26]
mtDNA_186	186	16,608	Human mitochondrial DNA [29]
RDPII_218	218	4,182	Prokaryotic RNA [30]
S1482_346	346	897	Plant genus Tiquilia [31]
M2839_470	470	829	Fungi genus Basidiomycetes [32]
ZILLA_500	500	759	rbcL plastid gene [33]

实验

将以下操作加入并行，实验结果如左下：

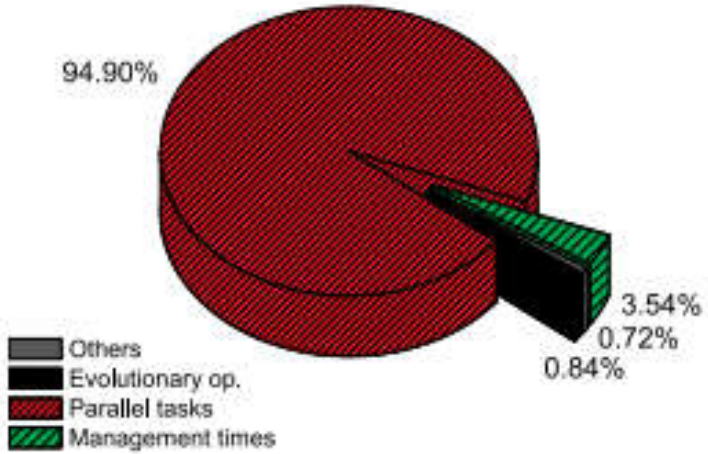
- Evaluation
- Optimization
- BIONJ



(d) OpenMP NSGA-II

将以下操作加入并行，实验结果如右下：

- Evaluation
- Optimization
- BIONJ
- Evolutionary



(d) OpenMP NSGA-II

实验结果

	8 cores		16 cores	
				rbcL_55
Algorithm	SU	Eff.(%)	SU	Eff.(%)
NSGA-II	**6.851**	**85.643**	**13.157**	**82.230**
MO-FA	6.722	84.024	12.759	79.748
				mtDNA_186
Algorithm	SU	Eff.(%)	SU	Eff.(%)
NSGA-II	**7.142**	**89.276**	**13.258**	**82.861**
MO-FA	6.923	86.539	12.907	80.668

Eff(efficiencies)=SU(speedup)/n

*n表示并行程度，即所使用节点数