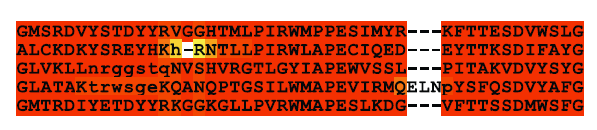
Multiple sequence alignment（MSA，多序列比对）问题总结

1. **多序列比对问题及衡量指标**
   1. **什么是多序列比对**

输入是多个相似的DNA或蛋白质序列，通过在每个序列的不同位置插入尽量少的空格，使得多个序列具有相同的长度，叠放在一起具有最大的相似性。如下图所示



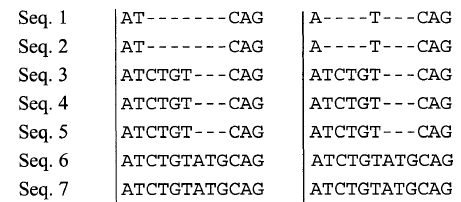
* 1. **多序列比对的衡量指标**

1. sum of pairs

把多序列比对的结果，每一对（pair）都拿出来计算相似度，然后求和/平均值。

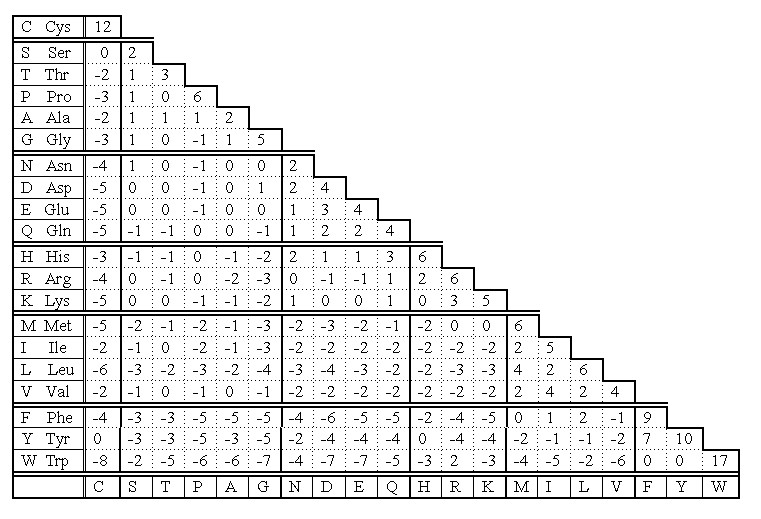
1. **仿射空隙罚分（Affine Gap Penalty）*重要***

对于DNA/蛋白质序列，通常不能用编辑距离来当作相似程度。认为：连续的一片空格（空隙gap，空格是space）好于间断的多个空格。对于下图，显然第一种方案好于第二种方案，但是如果算编辑距离的话，二者是相同的。



一般DNA通用的规则是：match得1分，mismatch得-3分，出现一个gap得-5分，gap中每增加一个space得-2分。（例子：插入了一个空格-得-5分，连续插入了2个空格得-7分，如果不连续的连个空格，就是-10分）

蛋白质就比较复杂了，不同的字母比对得分不一样，需要查表（PAM250或者BLOSUM62），下图就是PAM250表



1. 准确率

在评价蛋白质的多序列比对时候，有时也会用准确率，给定的benchmark数据有标准比对好的结果，看你的结果与标准相差多少；但在DNA的多序列比对中几乎不用，因为后面我准备做DNA的多序列比对，所以这里没有过多关注。

* 1. Benchmark dataset

1. BAlibase

多数相关论文都用这个数据集。现在打不开。以前我看过，多是蛋白质数据，而且都是小数据，所以我一般不用。

1. Large scale dataset

我在我之前的论文Bioinformatics2015上给出了2组数据， 另外RECOMB2014的论文open了一些数据，但多数是自己生成的模拟数据(<http://www.cs.utexas.edu/users/phylo/datasets/alignment.html>),包括：

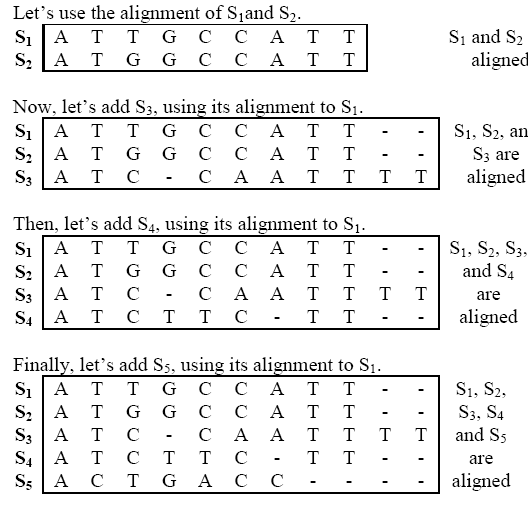
1. RNA模拟数据（含二级结构，NEXUS格式），相似度很高<http://kim.bio.upenn.edu/software/csd.shtml>
2. Liu Kevin Science论文的数据，相似度很低<http://www.cs.utexas.edu/~phylo/sate/public/sate_journal.html>
3. 模拟noncoding数据，好像差异还比较大，有fasta格式<http://bioinfolab.unl.edu/~cstrope/iSG/benchmark/index.html>
4. 各种RNA数据，序列短、多，差异性大<http://www.rna.ccbb.utexas.edu/DAT/3C/Alignment/>
5. <http://www.zmuc.dk/public/phylogeny/TNT/More/Supp_Data_Set.tgz>

Ref：Goloboff, P. A., S. A. Catalano, J. M. Mirande, C. A. Szumik, J. S. Arias, M. Källersjö, J. S. Farris. 2009. Phylogenetic analysis of 73 060 taxa corroborates major eukaryotic groups. Cladistics 25:211-230.

1. **目前多序列比对的研究现状**
   1. **传统思路**

将多序列比对转化成双序列比对是该问题的主要思路。双序列可以用动态规划来解决（即使是仿射空隙罚分也可以用稍麻烦的动态规划来解决）。

1. **星比对**：选择一个中心序列（center star sequence），将中心序列与其他的所有序列分别作双序列比对（pairwise alignment），然后检查中心序列都在哪些位置插入了多少个空格，合并后修正其他序列，得到最终结果。如下图所示，对于S1，S2，…，S5，选S1当中心序列，分别与S2-S5进行比对



1. **树比对**：树比对类似于自底向上的层次聚类，先算出一颗指导树(guide tree)(无根树，每个叶子结点代表一个输入的序列)。然后按照树的结构依次做双序列比对，最后得到最终的多序列比对结果。
   1. **最新方法**

目前多序列比对的研究多集中于蛋白质的多序列比对，或整体多序列比对策略，几乎没人去关注string algorithm。下面介绍研究比较多的几个策略。

1. 指导树构建方法

如果对于大数据，基本就是先采样，构造出一颗骨架树(backbone)，然后把剩余的（并行的）插入骨架树，并比对。

Ref: Mirarab S, Nguyen N, Warnow T. PASTA: ultra-large multiple sequence alignment[C]//Research in Computational Molecular Biology (RECOMB). Springer International Publishing, 2014: 177-191.

还有，构建指导树的时候，可以估算两两序列之间的距离，不用动态规划比对，而是近似的快速估算距离，也有一些方法

Ref：Lassmann T, Sonnhammer E L L. Kalign–an accurate and fast multiple sequence alignment algorithm[J]. BMC bioinformatics, 2005, 6(1): 298.

1. FFT变换

蛋白质字母比较的时候要查表，比较麻烦。有人用一个二维向量代替每一个字母，字母比较变成了二维向量之间的距离；kmer（就是k-gram）之间的比较转换成向量的点乘，利用FFT变换把O(k2)降低到O(klogk)，具体如何变换我也没太看明白。

Ref : Katoh K, Toh H. Recent developments in the MAFFT multiple sequence alignment program[J]. Briefings in bioinformatics, 2008, 9(4): 286-298.

1. 基于模板的比对

也就是说比对的时候会有一些先验知识，比如哪一部分要和哪一部分对齐，或者是已经比对齐了若干序列，现在又来了一堆新的序列。这些内容不是我关注的，所以没有仔细看。

Ref：Kemena C, Notredame C. Upcoming challenges for multiple sequence alignment methods in the high-throughput era[J]. Bioinformatics, 2009, 25(19): 2455-2465.

* 1. **现有软件**

|  |  |  |  |
| --- | --- | --- | --- |
| PASTA | <https://github.com/smirarab/pasta> | RECOMB, 2014 | 效果还可以，即能处理DNA，也能处理蛋白质 |
| HAlign | http://datamining.xmu.edu.cn/software/halign/ | Bioinformatics, 2015 | 我自己开发的软件，只能处理DNA，优点是并行，能处理大数据 |
| MAFFT | http://mafft.cbrc.jp/alignment/software/ | MBE, 2013; NAR, 2002, 2005, 2013; Bioinformatics,2007, 2010, 2012; BIB 2008; BMC Bioinformatics 2008; | 是一个日本团队持续开发了很多年的软件，支持并行、GPU等，效果还不错 |
| ClustalW | http://msa.sbc.su.se/cgi-bin/msa.cgi | MSB 2011 | 主流软件，可以很好的可视化比对结果，但是处理不了大规模数据 |
| KAlign | http://msa.sbc.su.se/cgi-bin/msa.cgi | NAR, 2005, 2006, 2009; BMC Bioinformatics 2005; | 号称处理大数据，但我的实验效果一般 |
| Muscle | http://www.drive5.com/muscle/ | NAR 2004; BMC Bioinformatics 2004 | 早期的软件，我没太尝试，主要是为了处理蛋白质数据设计的 |

1. **我的想法**

现在的MSA软件，在进行双序列比对时，还都采取的是完整的动态规划，以保证蛋白质的比对效果，尤其是可以处理仿射空隙罚分。目前有很多字符串算法的技巧，比如后缀树（或更高级的一些技巧），仅仅被用在了Mapping这个生物信息学问题中，没有用在MSA上。我想借鉴一些Mapping上面的技巧，用于快速的DNA多序列比对（暂时还没想用于蛋白质）。

1. Kmer(等同于n-gram)

对一条序列（相当于模板T，Text）做kmer索引，其他的序列弄成overlap或不overlap的kmer，把这些kmer向做了索引的T上面map，从而找出其他序列与T较长的完全匹配，进而去比对那些剩余的区间。整个思路类似于BLAST。

优点：kmer的索引可以以文件形式存到硬盘上，从而可以处理大规模数据；其他序列和T的map及对其都可以分布式执行，没有依赖关系，适合并行化。

1. Space seed

原来的kmer相当于连续的k个字母完全match，变种的kmer相当于可以不连续，按照一定的规则（pattern）就可以。比如说3mer，如果两个字符串，每隔2个字母就有1个字母不一样，虽然有66.67%的相似度，但是却没有一个公共的3mer。如果按照1101这种变种的3mer（中间那位0不作要求，匹不匹配都行），可以从理论上证明比111这种传统的3mer敏感。这种构造可以试图找出那些连续的一大片、不完全match的区域。

1. BWT变换(Burrows-Wheeler transform), FM-index

最近，与kmer相比，FM-index被认为是更有效的索引方式。

FM-index的优点是节省内存，是一种类似于后缀Trie树的索引结构。我这里没想好，如果对一个序列做索引，其他序列都和它比对（星比对），就不需要FM-index，直接后缀树就好，比较简单；如果要对所有的序列一起做索引，比如用特殊符号$把所有的序列连起来，这时后缀树就太耗内存了，需要用FM-index。

1. **下一步计划**

我计划先用后缀Trie树做一个简单的多序列比对软件，看一下实验效果，然后再考虑是否用FM-index，或是GPU或Hadoop进行加速。还没有更深入的想法。