第二代高通量测序组装综述

孙钊
zixiaojindao@gmail.com

2013年1月18日

摘要

第二代高通量测序产生了大规模的短序列。大规模输入，指数级的序列重叠，高概率的序列重复使得早期基于“overlap-layout-consensus”的方法不再适用。新的基于de bruijn图的方法在短序列组装中取得了成功。本文以一个纯计算机专业的学生的角度阐述了第二代高通量测序方法，序列组装的挑战以及如何进行序列组装的问题。在文末描述了我自己在序列组装领域做出的一些努力。

1 开始之前

首先我想解释一下我写这篇文章的动机。这篇文章的主题是关于基因工程的计算机辅助过程–序列组装。序列组装实际上是生物信息学的范畴。作为一个纯计算机背景的学生，我在2010年的夏天开始在这个领域进行研究。在读了大量的相关领域论文，与国内外的生物科研人员充分交流，尝试了各种各样的生物信息学中的工具之后，我没有取得任何创新性的成果，但是获得了非常宝贵的领域经验，并希望与那些对该领域一无所知又徘徊在领域门口的计算机的学生分享这些经验。本文我讲逐一回答“什么是基因测序”,“什么是序列组装以及它的挑战”, “如何组装序列”, “如何评价组装软件”这些问题，以及在最后列举我个人在这个领域做出的一些努力。本文中，我不会涉及太多算法的细节，我主要的目的是给读者一些关于基因序列组转中“做什么”和“如何做”的直觉印象，并且，我会提供大量的相关文章，材料和链接来帮助你构建一个完整的领域框架。

2 什么是基因测序

在基因工程和生物化学领域，测序意味着确定无支链生物聚合物的一级结构，例如DNA, RNA和蛋白质。本文将除去蛋白质而只关注DNA和RNA的序列分析。DNA测序就是确定一段给定DNA链的核苷酸序列的过程。具体而言，DNA是由4种分别用‘A’, ‘G’, ‘C’, ‘T’来表示的核苷酸组成的序列，也就是包含‘A’，‘G’，‘C’，‘T’的一个字符串。但是，生物上应用最广泛的鸟枪法测序，随机的把原始的DNA链条打断成碎片DNA，然后对这些碎片DNA进行测序。这些DNA碎片测出来的DNA序列称为一个read。为了提高read的覆盖度和质量，大量的由PCR扩增技术通过细菌作为模板的DNA复制链被同时测序图1显示了鸟枪法测序的过程，同时你会发现，仅碎片DNA的两端被测序了而不是全部。这种只测两端的现象是由生物测序的方法决定的，但是通过加入特别的方法，这一对的read的距离可以被估计出来。这时，这两个read被称作是pair-end 的read，它们之间的距离叫做insert length。

DNA分子是由两条互补链组成的双螺旋结构。根据碱基配对的原则–‘A’配对‘T’，‘G’配对‘C’，可由DNA分子中的一条链推导出互补的那条链。一些用3’和5’表示的特殊序列用来表示两条链的方向。为了简化说明起见，如果一条链定义为3’到5’方向，则其互补链为5’到3’方向。例1显示了一个双链的碎片DNA，其中红色部分为pair-end的read。注意到pair-end的两条read是分别位于两条互补链上的，但他们测序的方向都是从3’ 到5’。因此，例1中的两条read对应的序列字符串为‘AGCTAA’ and ‘GCCAA’。

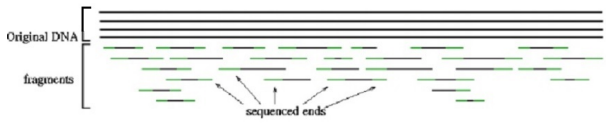

图 1:

3'→5'
AGCTAATGCTATCTTGGC
TCGATTACGATAGAACCG
5'→3'
例1

测序方法和平台是随着时间的发展而发展的。第一代测序平台的代表是Sanger[13], 它测序出来的read平均长度大约为800bp（一般为500-600bp）。最近，新的测序技术已经出现[9]。已经可以商业应用的第二代测序平台有454测序仪[10]，Illumina测序仪[2]和SOLID测序仪(www.appliedbiosystems.com)。和传统的Sanger测序平台相比，这些技术节省了大量的花费并且输出的吞吐量很高。但是，第二代测序技术产生的read序列要比传统的Sanger 序列短很多，一般来说454可以达到400-500bp，Illumina为50bp，而SOLID为35bp。正是因为他们长度太短，和早起的测序项目相比，他们必须提高read的覆盖度输出而输出大规模的read序列。
测序芯片把输出的read序列保存在用FASTA或者FASTQ格式的文件中。FASTA格式的文件中，每一条read序列由一行描述行开始，然后紧跟着read序列字符串的行。描述行一般都由符号“>”开始，read序列建议每行不超过80个字符。图2显示了一个FASTA格式的序列文件。FASTQ文件使用4行来表示一个read序列，第一行一般由‘@’符号开始表示一个序列标识符。第二行是原始的read序列字符串。第三行由‘+’ 号开始一行新的描述标识符。第四行用ascii编码质量数值，其中每个字符的ascii数字对应了第二行同样位置的序列的质量。图3显示了一个最简单的FASTQ文件。如果想了解更多关于质量打分行的细节，可以参考维基百科上的对FASTQ 格式的介绍。
最后需要一提的是RNA测序。由于RNA是由DNA转录获得，它所有信息都已经存储在细胞的DNA中。通常对RNA的测序方法首先采取逆转录的反应，将RNA逆转录为cDNA，然后对cDNA采用DNA的测序方法即可。
总之第二代测序平台对DNA或RNA进行测序，输出了一个用FASTA或FASTQ文件格式的序列文件。

3 什么是序列组装

测序阶段输出的仅仅是一堆碎片DNA的序列，而我们需要的是原始的完整DNA的核苷酸序列。序列组装就要要对齐和合并这些碎片DNA序列为原始序列。直觉上，需要利用read序列之间的重叠区域来把他们粘成一个更长的序列。下面列出的一些挑战使得序列组装成为一个十分复杂的任务：

```
>AB000263 |acc=AB000263|descr=Homo sapiens mRNA for prepro cortistatin like peptide, complete
cds.||len=368
ACAAGATGCCATTGTCCCCCGGCCTCCTGCTGCTGCTGCTCTCCGGGGCCACGGCCACCGCTGCCCTGCC
CCTGGAGGGTGGCCCCACCGGCCGAGACAGCGAGCATATGCAGGAAGCGGCAGGAATAAGGAAAAGCAGC
CTCCTGACTTTCCTCGCTTGGTGGTTTGAGTGGACCTCCCAGGCCAGTGCCGGGCCCCTCATAGGAGAGG
AAGCTCGGGAGGTGGCCAGGCGGCAGGAAGGCGCACCCCCCCAGCAATCCGCGCGCCGGGACAGAATGCC
CTGCAGGAACTTCTTCTGGAAGACCTTCTCCTCCTGCAAATAAAACCTCACCCATGAATGCTCACGCAAG
TTTAATTACAGACCTGAA
```

图 2:

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+
!''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65
```

图 3:

- 一个细胞中含有多条染色体，继而含有多条DNA链。这些DNA链同时被测序，不同链上的序列可能会有重叠。
- DNA是双链结构，一条链上的序列可能和它的互补链序列重叠。
- 随机打断序列的过程并不是一个标准的泊松随机过程，可能会有一部分DNA区域没有被read覆盖到。
- DNA序列中可能会有重复出现的序列，这些序列会被错误的当做重叠区域而合并。
- 测序错误使得read序列出现插入，删除，突变的字符。
- 第二代高通量测序输出较短的read序列，是的read之间重叠的概率增大。

在上面的这些挑战中，序列重复的问题是最具挑战行的。图4A展示了一个简单的重复序列问题，其中组装软件错误的把两段重复出现的A序列合并成为了一段序列使得最终得到两个连续段而不是一个图4B。
序列组装的基本策略有两种，一种是基于参考序列的，另外一种是全新组装。

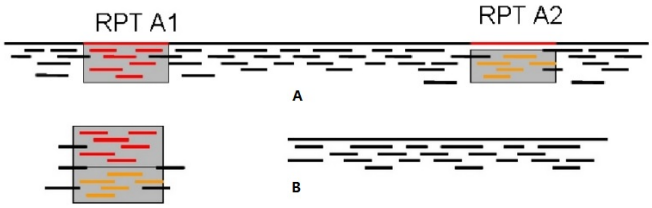

图 4:

基于参考序列的方法一般使用对齐工具来把read序列对齐到参考序列上，然后

组装出一个和参考序列相似但不完全相同的序列。全新组装策略不使用参考序列，因此对于没有参考序列的物种特别有用。本文仅仅包含对全新组装方法的介绍，并在后续的章节中阐述该方法的细节。
理想情况下，组装程序应该对于每一条测序的染色体输出一个连续段。但实际上，因为上述挑战的存在，组装程序的输出是多个被打断的连续段。总而言之，组装软件输入一个FASTA/FASTQ的read文件，然后输出一个连续段的集合。

4 如何组装序列

第一代测序的序列组装一般是基于“Overlap-consensus-layout”[11]方法。它把每个read看做是一个独立的顶点，如果两个顶点代表的read之间有重叠则给它们之间连一条边。虽然“overlap-consensus-layout”方法是非常直接和有效的，特别是在read很长的情况下。但是对于第二代测序产生的高质量大规模的read序列而言，这种方法用于计算任意一对read之间的重叠部分的代价将是非常大的。历史上仅有一个基于该方法并用来组装短序列的软件–EDENA[6]。随后，Idury and Waterman提出了一个新的用图的方法来组装短序列，并且这个思想最终发展成为至今为止用来组转短序列的最为流行的方法–De Bruijn图。这个序列的De Bruijn图是以一堆kmer作为顶点，如果其中一个kmer的后k-1个字符串和另外一个kmer的前k-1个字符串相同，那它们之间就连一条边。kmer是用一个固定长度为k 的滑动窗口来切read序列而来的。kmer和de bruijn图的例子可以在图5和图6中找到。序列de bruijn图有以下性质：

- 长度为n的read产生n-k+1个kmer。
- 每一个read唯一对应了de bruijn图中的一条路径。
- 每一个顶点通过在最前面或者最后面添加‘A’，‘G’，‘C’，‘T’得到它可能的前后各4个邻居。

根据性质2和性质3，de bruijn图的顶点规模的上界为O(n-k+1)，边的上界为O(8(n-k+1)),其中n为目标DNA链的长度。顺便提一下，人类基因组的总长度为3Gbp，它构成的de bruijn图可以存储在一个单独的128G内存的服务器上或者集群系统上。性质3减少了计算任意一对read之间重叠的代价，使得de bruijn图的方法可行。基于这种方法的组装软件有Velvet [16]，ABYSS [15]，ALLPATHS [3]和SOAPDenovo [8]。ABYSS和ALLPATHS声称他们组装出来的人类的基因组序列。表1展示了这些组装软件的一个简明的对比说明。软件链接可以在表2中找到。
我个人阅读过Velvet和ABYSS的源代码，并实际运行过这两个软件，在这里提供它们的一些细节部分。Velvet是由两个模块软件组成的，一个velveth，一个是velvetg，‘h’的意思是哈希，‘g’的意思是图。Velvet设计了一个新颖的构造de bruijn图的算法并且声称这种方法要比de bruijn图定义的构造方法在单机上更省内存，效率更高。velveth首先顺序的读取read文件，对于每一个read将其拆分为一堆kmer。对于每一个kmer，用一个哈希表来记录它第一次（同时记录它的互补kmer）出现时所在的read的ID（互补kmer的read ID为负）和偏移量。为了保证每个kmer的互补kmer不是自己，k必须取值为奇数。这第一次扫描使

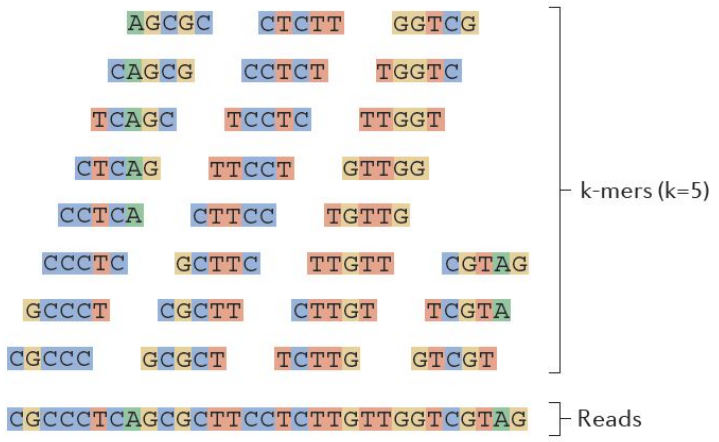

图 5:

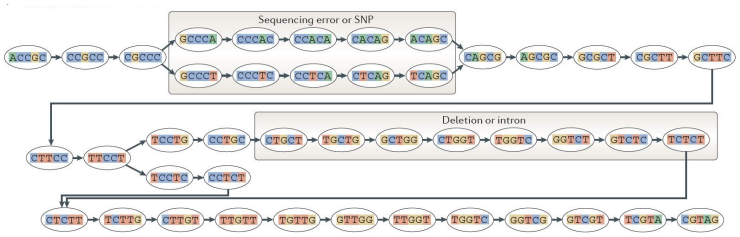

图 6:

得所有的read信息变为了kmer与它前面出现的read之间重叠的信息，这个新的表示结果叫做roadmap，并用一个名为ROADMAP的文件保存。velvetg读取这个ROADMAP文件，计算每个read与它后面的read之间的重叠来产生一个具有相反信息的结构。对于每一个read，一旦它中间有序列与其他read重叠，我们就把它分离出来，那些没有被打断的连续序列就对应一个顶点。根据read之间的重叠关系就可以一个顶点接着顶点地找到它们的邻居。velvetg的最终输出也就是Velvet的最终输出保存为一个名为config.fa文件。ABYSS按照de bruijn图的定义构造图。并行版本（MPI版本）的ABYSS通过一个定义好的哈希函数来确定顶点所属的机器。ABYSS程序由一个用makefile语法写的接口文件“abyss-pe”组织。abyss-pe根据当前的中间文件自动计算出模块的调用依赖，并执行一条最短的可以输出最终结果的调用。ABYSS的最终输出保存为名为*-contigs.fa，*-scanffolds.fa，*-unitigs.fa和一个结果的统计信息文件*-stats。

构造完de bruijn图之后，需要进行的修正图的过程。修正图的思想是通过删除图中的分支路径来把de bruijn图简化为一条长路径。分支路径有两种类型，一种是tips，一种bubbles。直观上讲tips是末端顶点，bubbles是两条共用末端顶点的路径组陈过的“气泡”。一旦一个顶点A唯一指向了一个顶点B，而顶点B也

表 1:

软件名称	并行性	大规模处理	内存消耗
Velvet	OpenMP	No	High
ABYSS	OpenMP+MPI	Yes	Low
ALLPATH	?	Yes	?
SOAPDenovo	?	Yes	Low

表 2:

软件名称	链接
Velvet	http://www.ebi.ac.uk/ zerbino/velvet/
ABYSS	http://www.bcgsc.ca/platform/bioinfo/software/abyss/
ALLPATH	http://www.broadinstitute.org/science/programs/genome-biology/crd
SOAPDenovo	http://soap.genomics.org.cn/soapdenovo.html

只有一个邻居A指向它，那么这两个顶点就可以合并为一个更长的顶点。Fig .7显示了tips和bubbles的样例。需要注意的是重复序列也可以在de bruijn图中产生类似“气泡”的回路，而且我们不知道这些重复的序列在原始序列中重复出现了多少次。在修正图的最后一步，如果输入是pair-end序列的话，我们可以估计回路两端顶点之间距离进而在这两个顶点中找出一条可行的路径。
最后需要提的是RNA序列组装。RNA序列组装要比DNA序列组装更为复杂，主要原因如下：

- 一些转录序列具有低表达水平，而另外一些具有高表达水平。
- 出现测序错误的高表达水平的序列可能比测序正确的低表达水平的序列多很多。
- 不同的转录组序列可能从DNA的同一段序列编码，因此有可能被错误的合并为同一个转录序列。

直接使用DNA组装软件去组装RNA序列只会得到更坏的结果。Velvet和ABYSS已经把它们组装DNA的软件扩展为相应的组装RNA的软件，分别为Velvet_Oases [14]和Trans-ABYSS [12]。最近，一个专门为RNA组装设计的软件Trinity [4]被开发出来。Zhao等人[17]对上述三个软件的性能，结果和资源消耗做了全面的比较，得出了Trinity组装出来的结果要比Velvet_Oasese和Trans-ABYSS好，但是时间代价和内存代价非常大。随后一组专门做HPC的研究人员[5]对Trinity进行了全面的分析和调优，大大的降低了Trinity的时间和资源代价，使得Trinity成为组装RNA的首选软件。

5 如何评测组装软件

为了评测现有的组装软件或者我们自己设计的组装软件，第一件需要做的

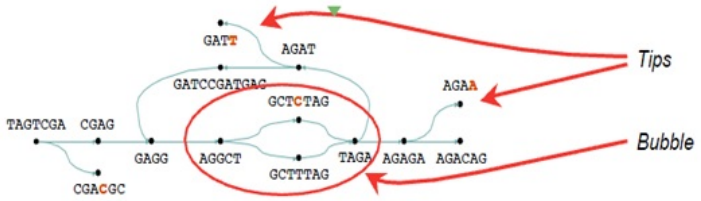

图 7:

事情就是准备read序列数据。通常有两种数据集可以用来评测，一种是人工合成的模拟数据，一种是真实测序数据。dwgsim [1]是基于参考序列的一个全基因组模拟测序软件。参考序列可以在ftp://ftp.ncbi.nlm.nih.gov/refseq/, ftp://ftp.ncbi.nlm.nih.gov/genomes/或者NCBI's nucleotide数据库(http://www.ncbi.nlm.nih.gov/nucleotide)中找到。关于refseq的介绍你可以参考http://www.ncbi.nlm.nih.gov/books/NBK50679/。NCBI的SRA数据库[2]提供了大量的真实测序数据。用户可以通过以SRP（研究），SRS（样本），SRX（实验），SRR（实例）开头的访问码来检索这个数据库。同一个样本的多个SRR数据可以用在一起来组装序列。顺便提一下，从SRA数据库下载的数据是以sra格式保存的，用户需要使用NCBI SRA Toolkit中的fastq-dump来抽取出FASTA/FASTQ格式的文件。如果数据是pair-end类型，需要使用fastq-dump的'–split-3' 或者'–split-files'参数。评价测序结果的度量有N50长度，运行时间，资源消耗，95%对齐率等。给定一个连续段的集合（也就是组装软件的输出），N50长度被定义为L，对于所有长度大于L的连续段，它们长度的总和恰好超过所有连续段总和的50%。N50长度越长就越说明结果中长连续段多，一般也就认为结果越好。在linux上，记录软件运行的时间，内存代价可以使用'/usr/bin/time -v'命令来记录。95% 对齐率定义为连续段可以以95%的相似度对齐到参考序列上的数量除以总的连续段的数量。BLAST [1]和BLAT [7]是两个最著名的序列对齐工具。由于BLAT效率更高，我一般选择BLAT作为对齐工具。BLAT输出的是一张用PSL格式存储的大表格，其中每一行代表了一个查询序列对齐到参考序列上的信息。对于PSL格式和表格中每一列含义不熟悉的用户可以参考PSL格式。每一个查询序列可能有多个对齐位置，也就是在PSL文件中有连续多行对齐信息，BLAT 软件没有提供像在web BLAT[3]上的相似度和得分两列，需要用户自己计算。计算的方法可以参考http://www.genoscope.cns.fr/blat-server/cgi-bin/vitis/webBlat，然后使用最佳匹配来计算95%对齐率。

6 个人贡献

我个人对序列组装的贡献列举在表3中。

[1]http://sourceforge.net/apps/mediawiki/dnaa/index.php?title=Whole_Genome_Simulation
[2]http://www.ncbi.nlm.nih.gov/sra
[3]http://www.genoscope.cns.fr/blat-server/cgi-bin/vitis/webBlat

表 3:

贡献	链接
Velvet and Oases for windows	https://github.com/zixiaojindao/velvet.git
ABYSS for windows	https://github.com/zixiaojindao/ABYSS.git
MemoryUsageMonitor for windows	https://github.com/zixiaojindao/MemoryUsageMonitor.git
blat-statistics for windows	https://github.com/zixiaojindao/blat-statistics.git
Sequence Assembly Information	https://github.com/zixiaojindao/RNA-sequence-info.git

7 总结

从一个程序员的角度看，序列组装就是通过序列之间的重叠把碎片序列的字符串粘贴成为较长原始字符串的过程。但是实际上，因为各种生物相关的挑战存在，使得组装任务变得非常复杂。大部分现有的基于de bruijn图的方法取得了较为满意结果。但是组装软件的时间和资源代价在组装大规模序列时依然十分可观。希望新的计算机算法和生物技术的发展可以在未来帮助尽快解决这一问题。

参考文献

[1] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, D.J. Lipman, et al. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[2] D.R. Bentley. Whole-genome re-sequencing. *Current opinion in genetics & development*, 16(6):545–552, 2006.

[3] J. Butler, I. MacCallum, M. Kleber, I.A. Shlyakhter, M.K. Belmonte, E.S. Lander, C. Nusbaum, and D.B. Jaffe. Allpaths: De novo assembly of whole-genome shotgun microreads. *Genome research*, 18(5):810–820, 2008.

[4] M.G. Grabherr, B.J. Haas, M. Yassour, J.Z. Levin, D.A. Thompson, I. Amit, X. Adiconis, L. Fan, R. Raychowdhury, Q. Zeng, et al. Full-length transcriptome assembly from rna-seq data without a reference genome. *Nature biotechnology*, 29(7):644–652, 2011.

[5] R. Henschel, P.M. Nista, M. Lieber, B.J. Haas, L.S. Wu, and R.D. LeDuc. Trinity rna-seq assembler performance optimization. In *Proceedings of the 1st Conference of the Extreme Science and Engineering Discovery Environment: Bridging from the eXtreme to the campus and beyond*, page 45. ACM, 2012.

[6] D. Hernandez, P. François, L. Farinelli, M. Østerås, and J. Schrenzel. De novo bacterial genome sequencing: millions of very short reads assembled on a desktop computer. *Genome research*, 18(5):802–809, 2008.

[7] W.J. Kent. Blat-the blast-like alignment tool. *Genome research*, 12(4):656–664, 2002.

[8] R. Li, H. Zhu, J. Ruan, W. Qian, X. Fang, Z. Shi, Y. Li, S. Li, G. Shan, K. Kristiansen, et al. De novo assembly of human genomes with massively parallel short read sequencing. *Genome research*, 20(2):265–272, 2010.

[9] E.R. Mardis et al. The impact of next-generation sequencing technology on genetics. *Trends in genetics*, 24(3):133, 2008.

[10] M. Margulies, M. Egholm, W.E. Altman, S. Attiya, J.S. Bader, L.A. Bemben, J. Berka, M.S. Braverman, Y.J. Chen, Z. Chen, et al. Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, 437(7057):376–380, 2005.

[11] E.W. Myers. Toward simplifying and accurately formulating fragment assembly. *Journal of Computational Biology*, 2(2):275–290, 1995.

[12] G. Robertson, J. Schein, R. Chiu, R. Corbett, M. Field, S.D. Jackman, K. Mungall, S. Lee, H.M. Okada, J.Q. Qian, et al. De novo assembly and analysis of rna-seq data. *Nature methods*, 7(11):909–912, 2010.

[13] F. Sanger, GM Air, BG Barrell, NL Brown, AR Coulson, JC Fiddes, P-M Slocombe, and M. Smith. Nucleotide sequence of bacteriophage (d x174 dna. 1977.

[14] M.H. Schulz, D.R. Zerbino, M. Vingron, and E. Birney. Oases: robust de novo rna-seq assembly across the dynamic range of expression levels. *Bioinformatics*, 28(8):1086–1092, 2012.

[15] J.T. Simpson, K. Wong, S.D. Jackman, J.E. Schein, S.J.M. Jones, and İ. Birol. Abyss: a parallel assembler for short read sequence data. *Genome research*, 19(6):1117–1123, 2009.

[16] D.R. Zerbino and E. Birney. Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome research*, 18(5):821–829, 2008.

[17] Q.Y. Zhao, Y. Wang, Y.M. Kong, D. Luo, X. Li, and P. Hao. Optimizing de novo transcriptome assembly from short-read rna-seq data: a comparative study. *BMC bioinformatics*, 12(Suppl 14):S2, 2011.