SOAPdenovo 软件

1 使用程序及参数：

SOAPdenovo 可以一步跑完，也可以分成四步单独跑

一步跑完的脚本:

```
./ SOAPdenovo all -s lib.cfg -K 29 -D 1 -o ant >>ass.log
```

四步单独跑的脚本:

```
 ./ SOAPdenovo pregraph -s lib.cfg -d 1  -K 29 -o ant >pregraph.log
./ SOAPdenovo contig -g ant -D 1 -M 3 >contig.log
./ SOAPdenovo map -s lib23.cfg -g ant >map.log
./ SOAPdenovo scaff -g ant -F >scaff.log
```

2 参数说明

-s STR 配置文件

-o STR 输出文件的文件名前缀

-g STR 输入文件的文件名前缀

-K INT 输入的 K-mer 值大小，默认值 23，取值范围 13-63

-p INT 程序运行时设定的线程数，默认值 8

-R 利用 read 鉴别短的重复序列，默认值不进行此操作

-d INT 去除频数不大于该值的 k-mer，默认值为 0

-D INT 去除频数不大于该值的由 k-mer 连接的边，默认值为 1，即该边上每个点的频数都小于等于 1 时才去除

-M INT 连接 contig 时合并相似序列的等级，默认值为 1，最大值 3。

-F 利用 read 对 scaffold 中的 gap 进行填补，默认不执行

-u 构建 scaffold 前不屏蔽高覆盖度的 contig，这里高频率覆盖度指平均 contig 覆盖深度的 2 倍。默认屏蔽

-G INT 估计 gap 的大小和实际补 gap 的大小的差异，默认值为 50bp。

-L 用于构建 scaffold 的 contig 的最短长度，默认为：Kmer 参数值 ×2

3 使用方法及示例

1）示例

```
SOAPdenovo_Release1.0/SOAPdenovo all -s Data/HCB.lib -K 25 -d   -o test
```

2) 输入文件

configFile (配置文件内容如下，非程序生成，需要软件使用者自己配置)

```
#maximal read length （read 的最大长度）
```

以“#”开头的行是注释内容
max_rd_len=50
#该值一般设置的比实际 read 读长稍微短一些，截去测序最后的部分，具体长度看测序质量
[LIB]
#文库信息以此开头
avg_ins=200
#文库平均插入长度，一般取插入片段分布图中给出的文库大小
reverse_seq=0
#序列是否需要被反转，目前的测序技术，插入片段大于等于 2k 的采用了环化，所以对于插入长度大于等于 2k 文库，序列需要反转，reverse_seq＝1，小片段设为 0
asm_flags=3
#该文库中的 read 序列在组装的哪些过程（contig/scaff/fill）中用到
设为 1：只用于构建 contig；
设为 2：只用于构建 scaffold；
设为 3：同时用于构建 contig 和 scaffold；
设为 4：只用于补洞
短插入片段(<2K)的设为 3，同时用于构建 contig 和 scaffold，长插入片段(>=2k)设为 2，不用于构建 contig，只用于构建 scaffold，454single 长 reads 只用于补洞。
rank=1
#rank 该值取整数，决定了 reads 用于构建 scaffold 的次序，值越低，数据越优先用于构建 scaffold。设置了同样 rank 的文库数据会同时用于组装 scaffold。一般将短插入片段设为 1；2k 设为 2；5k 设为 3；10k 设为 4；当某个档的数据量较大时，也可以将其分为多个档，同样，当某档数据量不足够时，可以将多个档的数据合在一起构建 scaffold。这里说的数据量够与不够是从该档的测序覆盖度和物理覆盖度两个方面来考虑的。
pair_num_cutoff=3
#可选参数，pair_num_cutoff 该参数规定了连接两个 contig 或者是 pre-scaffold 的可信连接的阈值，即，当连接数大于该值，连接才算有效。短插入片段(<2k)默认值为 3，长插入长度序列默认值为 5
map_len=32
#map_len 该参数规定了在 map 过程中 reads 和 contig 的比对长度必须达到该值(比对不容 mismacth 和 gap)，该比对才能作为一个可信的比对。可选参数，短插入片段(<2k)一般设置为 32，长插入片段设置为 35，默认值是 K＋2。
q1=/path/**LIBNAMEA**/fastq_read_1.fq
#read 1 的 fastq 格式的序列文件，“/path/**LIBNAMEA**/fastq_read_1.fq”为 read 的存储路径
q2=/path/**LIBNAMEA**/fastq_read_2.fq
#read 2 的 fastq 格式的序列文件，与 read1 对应的 read2 文件紧接在 read1 之后）
f1=/path/**LIBNAMEA**/fasta_read_1.fa
#read 1 的 fasta 格式的序列文件
f2=/path/**LIBNAMEA**/fasta_read_2.fa
#read 2 的 fasta 格式的序列文件
q=/path/**LIBNAMEA**/fastq_read_single.fq
#单向测序得到的 fastq 格式的序列文件
f=/path/**LIBNAMEA**/fasta_read_single.fa

#单向测序得到的 fasta 格式的序列文件
p=/path/**LIBNAMEA**/pairs_in_one_file.fa
#双向测序得到的一个 fasta 格式的序列文件

4 输出文件及说明

SOAPdenovo 分四部分别对应的输出文件：
pregraph 生成 7 个文件 *.kmerFreq *.edge *.preArc *.markOnEdge *.path *.vertex *.preGraphBasic
contig 生成 4 个文件 *.contig *.ContigIndex *.updated.edge *.Arc
map 生成 3 个文件 *.readOnContig *.peGrads *.readInGap
scaff 生成 6 个文件 *.newContigIndex *.links *.scaf *.scaf_gap *.scafSeq *.gapSeq

*.contig：contig 序列文件，fasta 格式
*.scafSeq：fasta 格式的 scaffold 序列文件，contig 之间的 gap 用 N 填充
#对于得到的*.scafSeq 文件还需要用 GapCloser 去合并其中的 gap，最后的 contig 文件则是对补洞之后的 scaffold 文件通过打断 N 区的方法得到。
以上两个文件是组装结果中最主要的输出。
*.scaf：包括 scaffold 中 contig 的详细信息。
在 scaffold 行中包括 scaffold 名字、contig 长度和该 scaffold 长度。
在 contig 行包括 contig 名字、contig 在 scaffold 上的起始位置、正反链、长度和 contig 间的链接信息
*.links：contig 间的 pair-end 连接信息
*.readOnContig：reads 在 contig 上的位置
*.peGrads： 主要可以通过调整本文件中的参数来显示构建 scaffold 所用到的插入片段库的个数，总共要到的 read 数，最长的 read 的长度，每个库对应的哪些 reads，rank 设置，pair_num_cutoff 设置。例如：

```
grads&num: 10    522083934        70
323      104577616        1        3
334      180770522        1        3
345      226070520        1        3
486      361955834        2        3
2200     392088076        3        5
2290     422272580        3        5
2400     445522690        3        5
4870     475666064        4        5
9000     511030930        5        8
9110     522083934        5        5
```

#文件中共分成 4 列。组装的配置文件中有 n 个文库，该文件则有 n+1 行，且按照文库大小顺序排列。
第 1 行中，第二三四列分别是 所用文库，reads 总数和组装中用到的最长的 reads 长度。

第 2 行中，四列分别是文库大小，文库中的 reads 数目，该文库 reads 用到的 rank 等级和该文库中 reads 用到的 pair_num_cutoff。
第 3～n+1 行，四列分别是文库大小，文库中的 reads 数目加上前面的文库中的 reads 总数，该文库 reads 用到的 rank 等级和该文库中 reads 用到的 pair_num_cutoff。
如果配置文件中没有设置 pair_num_cutoff，即使用默认参数，则最后一列显示为 0。

对于 SOAPdenovo 的每个步骤都有日志文件输出，要保存好日志文件，日志文件中包含有很多有用的信息。

SOAPdenovo 日志输出说明

pregraph.log:

其中有很多的统计信息，包括构建 debruijn-graph 时用到多少 reads 数，构图中生成了多少 uniq 的 kmer 以及设置-d 参数后去除了多少 kmer。
在 pregraph 中，可选参数有 -R -K -d
5467781332 nodes allocated, 70662750348 kmer in reads, 70662750348 kmer processed
3283081670 kmer removed
Kmer 数是取决于所设 k 值大小以及数据量，nodes 数即特异性的 kmer 数目，当 nodes 数目过高（一般和基因组大小差不多大小），可能是数据的错误率比较高，也可能是存在杂合。若 nodes 数目偏小，并且 kmer 数目很多，则基因组本身可能存在一定的重复度。对于 k 值的选取，当数据量充足时（>=40X），植物基因组一般采用大 kmer 会有比较好的效果，而对于动物基因组，k 值一般多取 27 和 29 则足够。kmer removed 表示的 -d 参数所去除的低频的 kmer。

contig.log:

contig 中，可选参数 -R -D -M，注，-R 参数的选定，必须 pregraph 和 contig 中同时选择才有效。
16430183 pairs found, 2334584 pairs of paths compared, 1674493 pairs merged
从 merged 的数量可以作为估计杂合以及测序错误的程度
sum up 1932549703bp, with average length 1170
the longest is 36165bp, contig N50 is 2871 bp,contig N90 is 553 bp
相关的统计信息，一般情况下，植物基因组的 contig N90 都在 200bp 左右，若 N90 高于 200bp，则该基因组的 scaffold 构建都不会有太大的问题。动物基因组，scaffold 构建很少有出问题的。

map.log:

Output 415219610 out of 1956217742 (21.2)% reads in gaps
1661094582 out of 1956217742 (84.9)% reads mapped to contigs
一般情况下，reads in gap 的比例和 map to contig 的比例总和大于 1。可能是因为 reads map 到多个地方都被算在其中的原因。当 map to contig 的比例很高（80%左右时），但是组装效果并不很好，可能是重复序列比较多。reads in gap 比例较高（大于 40%），是因为基因组组装的较碎，gap 区域较多。
map_len 默认值=K+5，当默认值大于设置的 map_len 时，以默认值为准，当默认值小于 map_len 值时，设置的 map_len 为准。

scaff.log:

average contig coverage is 23, 5832270 contig masked

构建 scaffold 是对高频覆盖的 contig 进行屏蔽（即频率高于 average contig coverage 的两倍的 contig 不用于构建 scaffold），从这里可以看出组装的基因组一定的重复情况。
estimated PE size 162, by 40034765 pairs
on contigs longer than 173, 38257479 pairs found,SD=8, insert_size estimated: 163
173 是配置文件中该文库的 insertsize，163 是根据 reads map 到 contig 上的距离的估计值，8 是这个分布的标准偏差。一般考虑 比对上去的 pair 数目和 SD 值。若 pair 对数很多且 SD 值很小（小片段文库数据不超过三位数，大片段文库数据部超过 500），那我们一般可以将配置文件中的文库插入片段的值改对短插入片段文库（<1k）的大小估计值，一般是比较准确的，下次组装以及补洞时应根据这个值对原来配置文件中的 insertsize 信息做修正。对于大片段文库（>=2K），因为是把 reads map 到 contig 上，若最长 contig 较短时，可能找不到成 pair 比对上去的 reads，这时，无法估计文库大小，需要自己将大片段一级一级的 map 到前一级的组装结果上，然后再分析大片段文库的插入片段大小。注，需要调整 insertsize 信息时，只需要修改* .peGrads 文件中的第一列，然后删除*.links 文件，重新跑 scaff 这一步即可。即构建 scaffold 时，主要是根据*.links 文件的信息进行连接。

Cutoff for number of pairs to make a reliable connection: 3
1124104 weak connects removed (there were 4773564 active cnnects))
Cutoff for number 是在配置文件中设的 pair_num_cutoff 值，weak connects 是低于这个值被认定为无效的连接数，active connects 是满足 cutoff 的连接数，根据这个数值可对 pair_num_cutoff 做调整

Picked 25241 subgraphs,4 have conflicting connections
conflicting connections 是表示构建 scaffold 时的矛盾数，矛盾数比较高（>100）时，可根据前面的有效连接数，适当提高 pair_num_cutoff 值，即提高 scaffold 连接要求的最少关系数

182483 scaffolds&singleton sum up 1990259817bp, with average length 10906
the longest is 6561520bp,scaffold N50 is 836795 bp, scaffold N90 is 157667 bp
scaffold 统计信息，将是根据 rank 分梯度的统计

Done with 13301 scaffolds, 2161915 gaps finished, 2527441 gaps overall
-F 参数补洞的统计信息。

5 参数调整

#一般组装时需要调整的参数，主要分两种：
一种是针对脚本中的参数改动：如调整 -K -R -d -D -M
-K 值一般与基因组的特性和数据量相关，目前用到的 SOAPdenovo 软件主要有两个版本，grape1123 和 grape63mer,其中 grape1123 是最新版的组装软件,K 值范围 13-31,grape63mer 是可以使用大 kmer 的组装版本，K 值范围 13-63。
经验：植物基因组的组装采用大 kmer 效果会比较好（要求短片段 reads 长度 75bp），动物基因组很少有用到大 kmer 后有明显改进效果的，且动物基因组的组装 K 值一般设置为 27 和 29 较多。

-R 参数，对于动物基因组，R 参数一般不设置，植物基因组由于较多的 repeat 区，则设置 R 参数后，效果更好。注意，设置-R 时，一般使用-M 的默认值。（熊猫基因组组装时得出的结论）
-M 参数，0-3,默认值 1。一般杂合率为千分之几就设为几。熊猫基因组组装时-M 2 。
-d 参数，对于没有纠错，没有处理的质量又较差的原始数据，kmer 的频数为 1 的很多的数据的组装，一般设置为-d 1 则足够。对于处理过，或者是测序质量较好的数据，可以不用设置。数据量很多时，也可以以-d 参数去除部分质量稍差的数据。
-D 参数，默认为 1，一般不用另行设置。
第二种，从 map 这一过程去调节参数。可以调整配置文件的 map_len 的值和调整文件*.peGrads。
当文库插入片段分布图中文库大小与实验给出的文库大小差异很大时，调整*.peGrads 文件中的插入片段大小。
根据每一档数据的数据量去调整文库的 rank 等级。当该文库的数据量很多或者是在构建 scaffold 的过程中的冲突数很多时，可是适当的调大第四列 的 pair_num_cutoff，把条件设置的更严一些。

6 内存估计

SOAPdenovo 的四个步骤消耗的内存是不一样的，其中第一步消耗的内存最多，使用没有纠错的的 reads，(K<=31)第一步消耗的内存在基因组大小的 80－100 倍左右，纠过错则在 40－50 倍左右，第二步相对消耗的内存会少很多，第三步消耗的内存是仅次于第一步的，在第一步的一半左右，第四步消耗的内存也会比较少。对于 CPU 的使用，默认是 8 个，如果申请内存时申请一个计算节点的所有内存测将 CPU 就设置为该计算节点的 CPU 个数充分利用计算资源，如果仅申请一个节点的部分内存则根据实际情况考虑。对于大 kemr(K>31)其内存使用是(k<=31)的 1.5 倍左右，有时甚至更多，要充分估计内存的使用，在第一次运行的时候考虑不能太保守。

7 常见错误

1）配置文件中 read 存储路径错误
只输出日志文件。
pregraph.log 中的错误信息：“Cannot open /path/**LIBNAMEA**/fastq_read_1.fq. Now exit to system...”
2）-g 后所跟参数与 pregraph（第一步） -o 后所跟参数名不一致
contig map scaff 这三个步骤都只是输出日志文件。
contig.log 中的错误信息：“Cannot open *.preGraphBasic. Now exit to system...”
map.log 中的错误信息：“Cannot open *.contig. Now exit to system...”
scaff.log 中的错误信息：“Cannot open *.preGraphBasic. Now exit to system...”
3）从 map 开始重新跑时，需要删除*.links 文件，否则会生成 core 文件，程序退出。