Taxonomic Classification DIAMOND+MEGAN & CLARK

Tutorial 6

ECEST480/680 Statistical Analysis of Genomes

Anna Lu, Keyur Shah

Diamond

Double Index AlignMent Of Next-generation sequencing Data

DIAMOND is a sequence aligner for protein and translated DNA searches

Drop In replacement of the NCBI BLAST tool

Suitable for protein-protein search as well as DNA-protein search

Upto 20000 times faster than BLAST

Installing Diamond

On a Linux based system like Proteus it is very easy to install Diamond

wget http://github.com/bbuchfink/diamond/releases/download/v0.8.35/diamond-linux64.tar.gz

tar xzf diamond-linux64.tar.gz

DIAMOND

Get nr.faa, a FASTA format protein database

```
wget ftp://ftp.ncbi.nih.gov/blast/db/FASTA/nr.gz
```

Build reference database

```
./diamond makedb --in nr.faa -d nr
```

Outputs a diamond database file `nr.dmnd`

Run Alignment (tabular .m8 is default, we used -f 100 .daa for MEGAN)

```
./diamond blastx -d nr -q reads.fna -o matches.m8
```

```
./diamond blastx -d nr -q data.fa -o matches.daa -f 100
```

MEGAN

METaGenomic ANalyzer

Tool for studying the taxonomic content of a set of DNA reads

Main application is to parse and analyze the result of an alignment of a set of reads against a reference database, using tools like DIAMOND to compare against NCBI-NT or NCBI-NR

Based on Lowest Common Ancestor (LCA) algorithm.

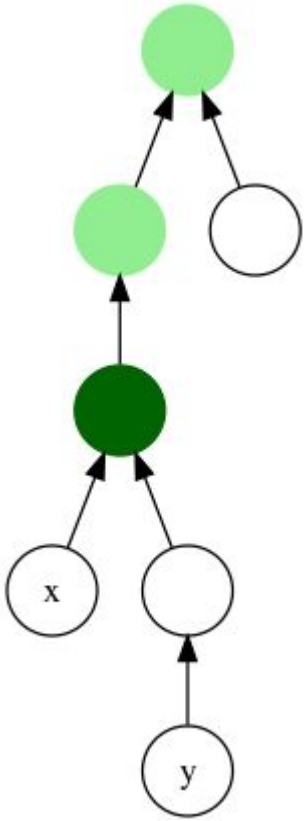
Installing MEGAN

The latest version of MEGAN, MEGAN 6 can be downloaded from the following website: https://ab.inf.uni-tuebingen.de/software/megan6/download

Last Common Ancestor Algorithm

Determines distance between pairs of nodes in a tree

Assigns reads to taxa such that the taxonomic level of the assigned taxon reflects the level of conservation of the sequence



A tree, lowest common ancestors in dark green

Qwertyus - Own work CC

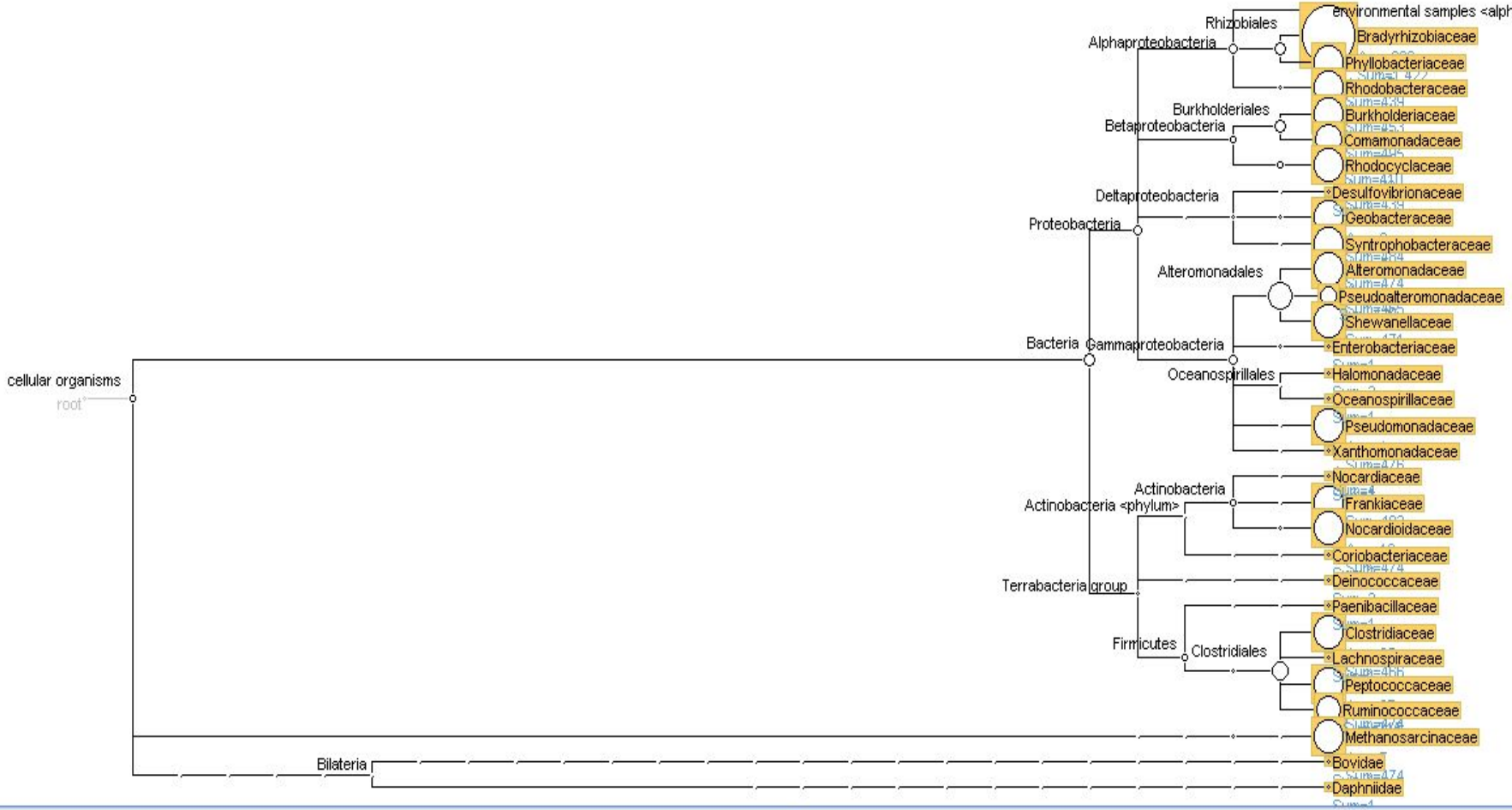
Implementation steps

Build the reference Diamond database

Run alignment to generate a diamond alignment archive file

Meganize the .daa file using MEGAN

Open the meganized file to view the taxonomic classification


cellular organisms
root
Bacteria
Proteobacteria
Alphaproteobacteria
Rhizobiales
Bradyrhizobiaceae
Betaproteobacteria
Burkholderiales
Burkholderiaceae
Deltaproteobacteria
Desulfovibrionaceae
Geobacteraceae
Gammaproteobacteria
Alteromonadales
Alteromonadaceae
Pseudoalteromonadaceae
Shewanellaceae
Enterobacteriaceae
Oceanospirillales
Halomonadaceae
Oceanospirillaceae
Pseudomonadaceae
Xanthomonadaceae
Actinobacteria <phylum>
Actinobacteria
Nocardiaceae
Frankiaceae
Nocardioidaceae
Coriobacteriaceae
Terrabacteria group
Deinococcaceae
Firmicutes
Paenibacillaceae
Clostridiales
Clostridiaceae
Lachnospiraceae
Peptococcaceae
Ruminococcaceae
Methanosarcinaceae
Bilateria
Bovidae
Daphniidae

CLARK Overview

(Supervised) CLAssifier based on Reduced K-mers

Taxonomic Classification of DNA/RNA {objects} to {target} reference sequences

Agenda:

- How to use CLARK
- What is a k-mer?
- k-mer assembly using De Brujin Graph

CLARK parameters

Command line tool for Linux and Mac only

Reference dataset: bacteria, virus, human, custom

Taxonomic ranks: species (default), genus, family, order, class, phylum

-k, K-mer size (default, k = 31)*

* See Appendix for k-mer length selection

Running CLARK

Select reference database and taxonomic rank

```
./set_targets.sh DIR_DB bacteria --species
```

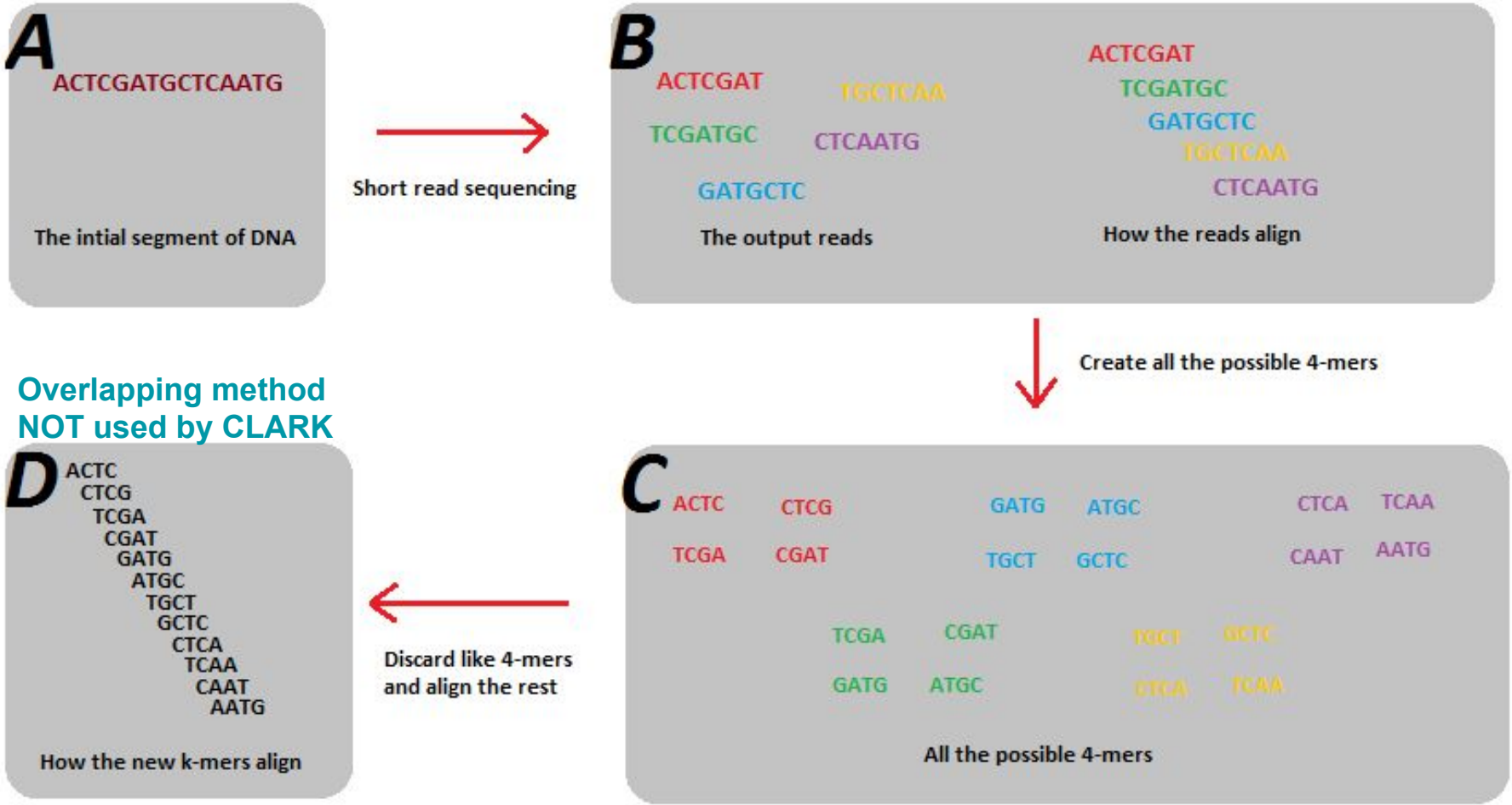
Run the classification, -O <objects> -R <results in csv format>

```
./classify_metagenome.sh -O ../data.fa -R ../result
```

Analysis of Results: Estimate abundance (counts and proportions)

```
./estimate_abundance.sh -F resultA.csv -D DIR_DB
```

What is a k-mer?



K-mer Algorithm

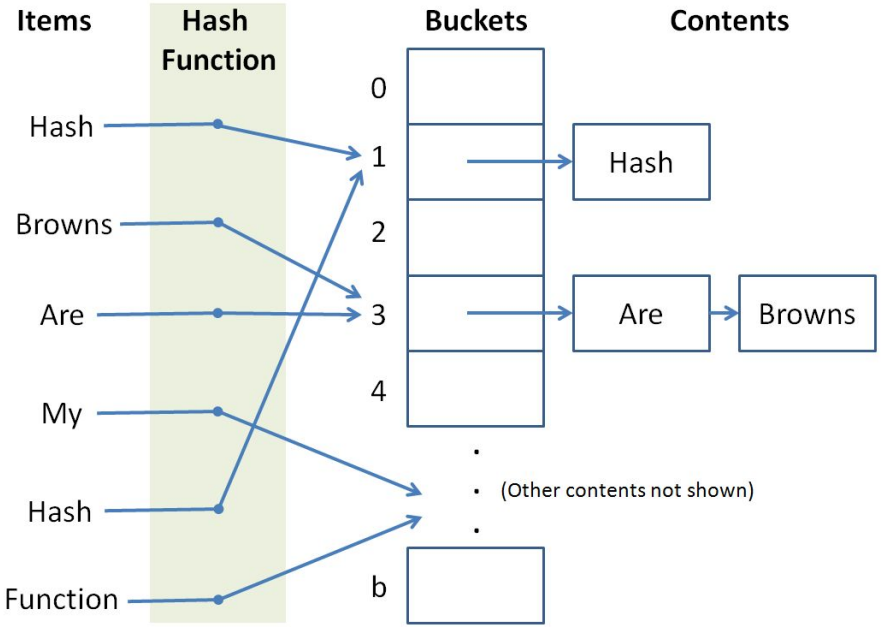
```
Function kmer(String, k):
    N = length(String)
    for i =1 until N-k+1
        Print String(i, k + i)
    end
End function
```

Outputs: all possible sequence substrings of length k

Inputs:

- DNA/RNA sequence (String)
- k (Integer length)

CLARK Discriminant k-mer classification



Forney Andrew, UCLA Computer Science

k=31 k-mer hashed to 4 bytes

Iterative matches between {target} and {objects} stored in hash table buckets

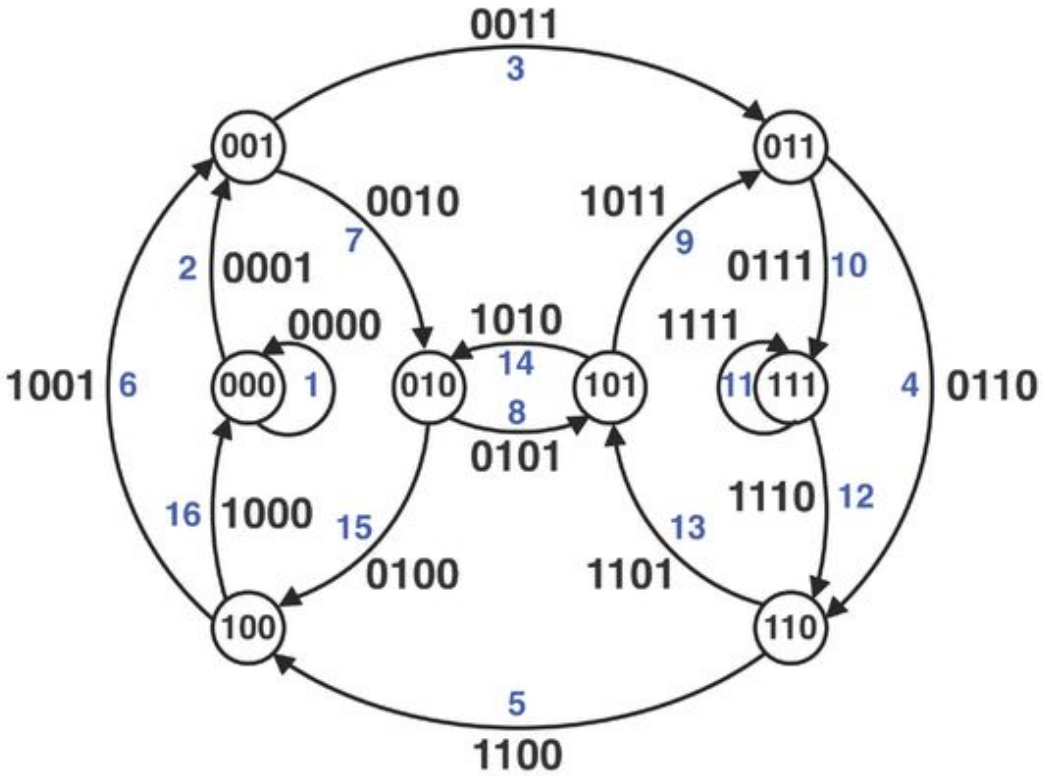
Hash function, where l = k-mer

$$l = \sum_{i=1}^{k} a[i]4^{i-1}$$

`a[i] = 0,1,2,3 for A,C,G,T or U`

Indexing linear time O(1)

De Brujin Graph (k=4) of Cyclic Superstring 0000 1100 1011 1101



Graph theory applied to solving k-mers

Eulerian cycle problem (every edge visited)

ATCG instead of 1 and 0

Better than overlapping method for repeated read

Compeau P. et. al. Nature 2011. http://www.nature.com/nbt/journal/v29/n11/full/nbt.2023.html

simHC.20.500 Comparison Dataset

3 metagenomic microbial sets

HiSeq, MiSeq, simBA-5

500 extracted out of 10000 reads per set

Equal sequence abundance per genome

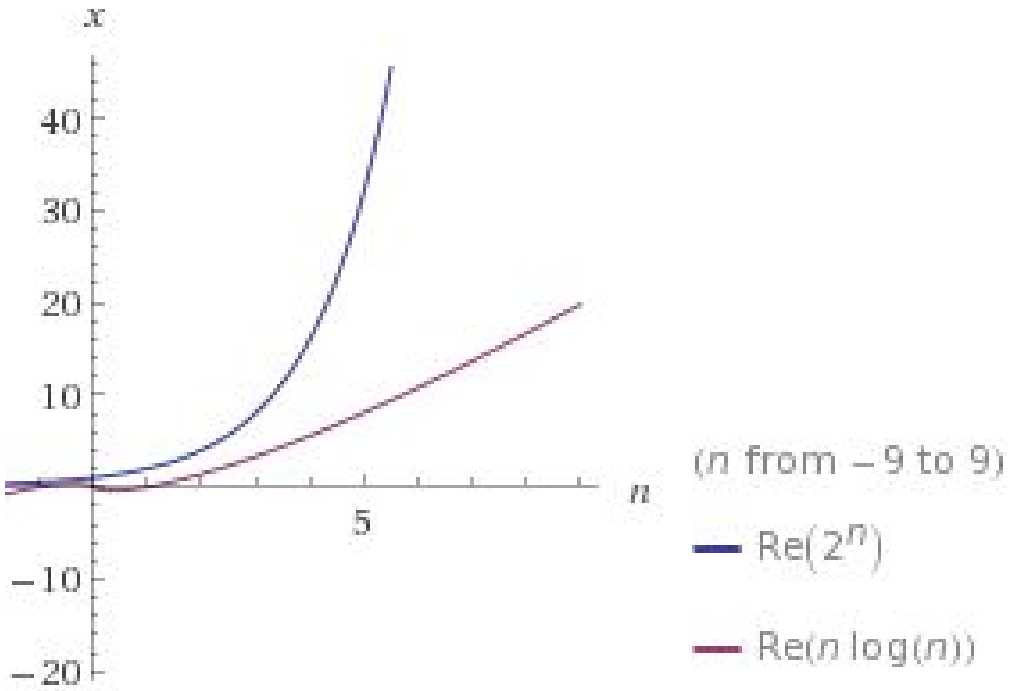
20 arbitrary genomes

IMG Taxon ID	Genome
640753002	*Alkaliphilus metalliredigens* QYMF
640427103	*Bradyrhizobium sp.* BTAi1
637000047	*Burkholderia cepacia* AMMD
637000160	*Chelativorans sp.* BNC1
640069309	*Clostridium thermocellum* ATCC 27405
637000088	*Dechloromonas aromatica* RCB
643348537	*Desulfitobacterium hafniense* DCB-2
637000116	*Frankia sp.* CcI3
637000119	*Geobacter metallireducens* GS-15
639633037	*Marinobacter aquaeolei* VT8
637000162	*Methanosarcina barkeri Fusaro*, DSM 804
637000192	*Nitrobacter hamburgensis* X14
639633046	*Nocardioides sp.* JS614
637000208	*Polaromonas sp.* JS666
637000216	*Pseudoalteromonas atlantica* T6c
637000221	*Pseudomonas fluorescens* Pf0-1
640069327	*Rhodobacter sphaeroides* 2.4.1, ATCC BAA-808
637000260	*Shewanella sp.* MR 7
639633063	*Syntrophobacter fumaroxidans* MPOB

Table S4: Genomes used in the "simHC.20.500" dataset (JGI database).

Classification Comparison Results

DIAMOND			CLARK		
Linux, Mac, Windows		Operating Systems	Linux, Mac OS X		Operating Systems
25.7 GB		Database size	42 GB		Database size
7.16 min	2.118 GB	Build Database	5 Hrs	148.5 GB	Build Database + Classification
45 min	1,880,528 GB	Alignment	1,422,389 GB		
6417000000 reads/min		Speed	31160000 reads/min		Speed
O(n log n)		Least Common Ancestor	O(2^n)		Discriminant k-mer, De Brujin



Generated in WolframAlpha

Appendix: K-mer Length Choice

Low k-mer size

Fewer edges, decreases sequence storage space

Increases chance of overlap

More vertices per k-mer, path ambiguities

Reduced information (entropy)

Cannot resolve microsatellites or repeats

High k-mer size

More memory needed to store sequence, more edges

Better alignments due to fewer paths and vertices per k-mer

Possible disjoints due to no vertices, paths. Less likely to overlap.

More information gain

Better resolution for microsatellites or repeats

Appendix: CLARK k-mer length selection

RECOMMENDATIONS

High sensitivity range k = [19 - 23]

Recommend k = 20-21 for sensitivity

Recommend k = 26-32 for precision and speed

Recommend k=31 for default mode

TRADEOFFS

Higher k-mer length, higher RAM usage

References

DIAMOND (2015), Protein Alignment

DIAMOND github

MEGAN (2007), Analysis of Metagenomic Data

CLARK (2015), Classification of metagenomic and genomic sequences using discriminative k-mers

Compeau (2011), How to apply De Brujin Graphs to Genome Assembly

Blog about De Brujin Graphs in Bioinformatics