Supplementary Material

Squeakr: An Exact and Approximate k-mer Counting System

Prashant Pandey 1,*, Michael A. Bender 1, Rob Johnson 1,2, and Rob Patro 1

1Department of Computer Science, Stony Brook University, Stony Brook, NY 11790, USA
2VMware Research, 3425 Hillview Ave, Palo Alto, CA 94304, USA

*To whom correspondence should be addressed.

Commands for counting benchmarks

Parameters to programs

These commands are for K=28. For k=55, the corresponding parameters were changed in each command.

KMC2

KMC2 was configured to count 28-mers for different datasets.

./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t16 @../../data/f_vesca.name.txt NA_f_vesca.res data
./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t8 @../../data/f_vesca.name.txt NA_f_vesca.res data
./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t16 @../../data/g_gallus.name.txt NA_g_gallus.res data
./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t8 @../../data/g_gallus.name.txt NA_g_gallus.res data
./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t16 @../../data/m_balbisiana.name.txt NA_m_balbisiana.res data
./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t8 @../../data/m_balbisiana.name.txt NA_m_balbisiana.res data
./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t16 @../../data/h_sapiens_1.name.txt NA_h_sapiens_1.res data
./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t8 @../../data/h_sapiens_1.name.txt NA_h_sapiens_1.res data
./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t16 @../../data/h_sapiens_2.name.txt NA_h_sapiens_2.res data
./kmc -k28 -ci0 -cs100000000 -cx1000000000 -r -t8 @../../data/h_sapiens_2.name.txt NA_h_sapiens_2.res data

Jellyfish

Jellyfish was configured to count 28-mers for different datasets.

./jellyfish count -m 28 -C -s 650M -t 16 -o jelly2 -g f_vesca.gen -G 11 --timing=jelly2time_f_vesca_16
./jellyfish count -m 28 -C -s 650M -t 8 -o jelly2 -g f_vesca.gen -G 11 --timing=jelly2time_f_vesca_16
Squeakr

Squeakr was configured to count 28-mers for different datasets. The k-mer size is specified in a header file (kmer.h). Please look at the github repo README for more information about the arguments.

```bash
./main 1 31 16 ../data/f_vesca/*
./main 1 31 8 ../data/f_vesca/*
./main 1 33 16 ../data/g_gallus/*
./main 1 33 8 ../data/g_gallus/*
./main 1 33 16 ../data/m_balbisiana/*
./main 1 33 8 ../data/m_balbisiana/*
./main 1 34 16 ../data/h_sapiens_1/*
./main 1 34 8 ../data/h_sapiens_1/*
./main 1 34 16 ../data/h_sapiens_2/*
./main 1 34 8 ../data/h_sapiens_2/*
```

Commands for de Bruijn graph traversal benchmarks

KMC2

We wrote a custom de Bruijn graph traversal benchmark using KMC2 API.

```bash
./kmc_de_bruijn NA_g_gallus.res
./kmc_de_bruijn NA_m_balbisiana.res
```

Squeakr

We wrote a custom de Bruijn graph traversal benchmark for Squeakr. The first argument is for the fastq file compression type and the second argument is the path to the directory where fastq files are stored. Please look at the github repo README for more information about the arguments.

```bash
./de-bruijn 1 ../data/g_gallus/*
./de-bruijn 1 ../data/m_balbisiana/*
```
Commands for inner-product query benchmarks

Mash
/usr/bin/time ./mash sketch -k 31 ../kmercounting/data/s_6_2.fastq.gz
/usr/bin/time ./mash sketch -k 31 ../kmercounting/data/s_6_1.fastq.gz
/usr/bin/time ./mash dist ../kmercounting/data/s_6_1.fastq.gz.msh ../kmercounting/data/s_6_2.fastq.gz.msh

Squeakr
The first two commands are for counting the k-mers from fastq files. The third command performs the inner-product between the two CQF representations.
./main 1 30 1 ../kmercounting/data/s_6_1.fastq.gz
./main 1 30 1 ../kmercounting/data/s_6_2.fastq.gz
./inner-prod ../kmercounting/data/s_6_1.fastq.gz.ser ../kmercounting/data/s_6_2.fastq.gz.ser 20 0

Commands for intersection benchmarks

KMC2
./kmc_tools intersect -t1 s_6_1.res -ci0 -cx1000000000 s_6_2.res -ci0 -cx1000000000 s_6_intersect.res -ci0 -cx1000000000

Squeakr
The first two arguments are the two CQF representations on which the intersection is performed.
The third argument is the size of the final CQF representation which written to disk.
The fourth argument is a flag to tell the program to perform intersection on the two representations.
The final argument is the number of threads.
./inner-prod ../kmercounting/data/s_6_1.fastq.gz.ser ../kmercounting/data/s_6_2.fastq.gz.ser 30 1 3