Dataset

Source: en.wikipedia.org

Data set

The data set has been generated using an ChIP-seq experiment performed on the Drosophila melanogaster "Oregon-R S2" strain. The sequencing data have been produced using the Illumina tehcnologie using single-end sequencing. The data are available here:

https://owncloud.ulb.ac.be/index.php/s/ceP7Vcu039z2Aw7

The reads and the dm6 version of the Drosophila melanogaster reference genome are provided:

Full data set

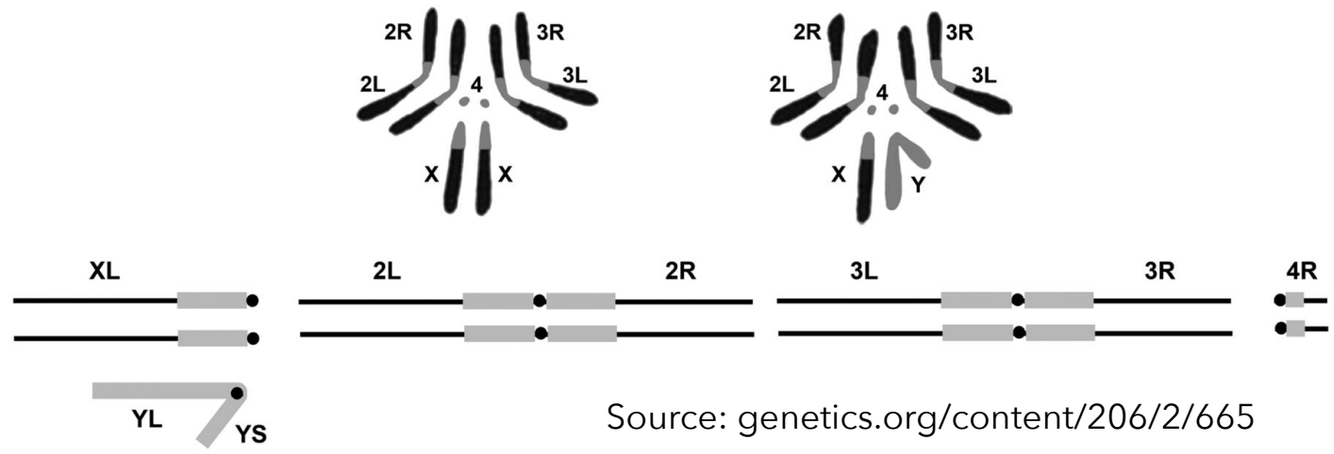
- all_reads.fastq.gz, all the ChIP-seq reads
- dm6.fa.gz, the full D melanogaster reference genome

To facilitate the development of the software, reduced data files, resulting from down sampling the full data set are also provided:

Mini data set

- dm6_chr2L.fa.gz, only chromosome chr2L extracted from the dm6.fa.gz genome
- 10k_reads.fastq.gz, only 10,000 reads extracted from all_reads.fastq.gz

D melanogaster Genome



Source: genetics.org/content/206/2/665

fastq (.fastq or .fq) format: raw reads with quality
fasta (.fa or .fasta) format: sequences (here the genome)

Genome fa format

```
>chr2L
Cgacaatgcacgacagaggaagcagaacagatatttagattgcctctcat
tttctctcccatattatagggagaaatatgatcgcgtatgcgagagtagt
gccaacatattgtgctctttgatttttggcaacccaaaatggtggcgga
tgaaCGAGATGATAATATATTCAAGTTGCCGCTAATCAGAAATAAATTCA
TTGCAACGTTAAATACAGCACAATATATGATCGCGTATGCGAGAGTAGTG
CCAACATATTGTGCTAATGAGTGCCTCTCGTTCTCTGTCTTATATTACCG
CAAACCCAAAAAgacaatacacgacagagagagagagcagcggagatatt
tagattgcctattaaatatgatcgcgtatgcgagagtagtgccaacatat
tgtgctctCTATATAATGACTGCCTCTCATTCTGTCTTATTTTACCGCAA
ACCCAAatcgacaatgcacgacagaggaagcagaacagatatttagattg
cctctcattttctctcccatattatagggagaaatatgatcgcgtatgcg
agagtagtgccaacatattgtgctctttgatttttggcaacccaaaatg
gtggcggatgaaCGAGATGATAATATATTCAAGTTGCCGCTAATCAGAAA
TAAATTCATTGCAACGTTAAATACAGCACAATATATGATCGCGTATGCGA
GAGTAGTGCCAACATATTGTGCTAATGAGTGCCTCTCGTTCTCTGTCTTA
TATTACCGCAAACCCAAAAAgacaatacacgacagagagagagagcagcg
gagatatttagattgcctattaaatatgatcgcgtatgcgagagtagtgc
caacatattgtgctctCTATATAATGACTGCCTCTCATTCTGTCTTATTT
TACCGCAAACCCAAatcgacaatgcacgacagaggaagcagaacagatat
ttagattgcctctcattttctctcccatattatagggagaaatatgatcg
cgtatgcgagagtagtgccaacatattgtgctctttgatttttggcaac
ccaaaatggtggcggatgaaCGAGATGATAATATATTCAAGTTGCCGCTA
ATCAGAAATAAATTCATTGCAACGTTAAATACAGCACAATATATGATCGC
GTATGCGAGAGTAGTGCCAACATATTGTGCTAATGAGTGCCTCTCGTTCT
CTGTCTTATATTACCGCAAACCCAAAAAgacaatacacgacagagagaga
```

Reads fastq format

```
@SRR034724.1 HWI-EAS279_chipseq14:4:1:403:244 length=36
GTTTAAATTTTCTGTTTATTCTTTTTTTTTTTTTTC
+SRR034724.1 HWI-EAS279_chipseq14:4:1:403:244 length=36
;;;;;;5;;;;;;5;;;%;;3;;;;;;,;;;;*4($
@SRR034724.2 HWI-EAS279_chipseq14:4:1:778:1793 length=36
GGATGGTAAGTTTCAAAATGTGCAAAAATCGGAAGG
+SRR034724.2 HWI-EAS279_chipseq14:4:1:778:1793 length=36
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;5;
```

Work to be done

- The description of the mapping software algorithm and its pseudo-code (generic code).
- A summary of the alignment output when applied to D melanogaster data set (e.g. percentage of reads mapped).
- Extract of the implementation code with comments (e.g. python, R or C code).

A PDF version of the report named ‘assignment2_lastname_firstname.pdf’ should be emailed to matthieu.dc.defrance@ulb.ac.be before the deadline (May 8, 2021).

Evaluation

- Explanations and choice of the algorithm
- Clarity of the implementation (code)
- Results and statistics
- Critical insights

Alignment to genome

Step 1: index the genome

- Suffix array
- Hash Table
- ….

Step 2: map the reads using the index

Cython

C variable and type definitions

The **cdef** statement is used to declare C variables, either local or module-level:

```
cdef int i, j, k
cdef float f, g[42], *h
```

and C **struct**, **union** or **enum** types:

```
cdef struct Grail:
    int age
    float volume

cdef union Food:
    char *spam
    float *eggs

cdef enum CheeseType:
    cheddar, edam,
    camembert

cdef enum CheeseState:
    hard = 1
    soft = 2
    runny = 3
```

Suffix array


```
ATATA$          [5, 3, 1, 4, 2]
12345$
```

```
np.asarray([5, 3, 1, 4, 2], dtype = np.int32)

np.int32(1).nbytes
4 bytes
```

```
180M genome size
G = np.zeros(int(180e6), dtype = np.int8)
S = np.zeros(int(180e6), dtype = np.int32)

sys.getsizeof(G) / (2 ** 20)
172Mb

sys.getsizeof(S) / (2 ** 20)
687Mb
```