Week 3

Advanced part

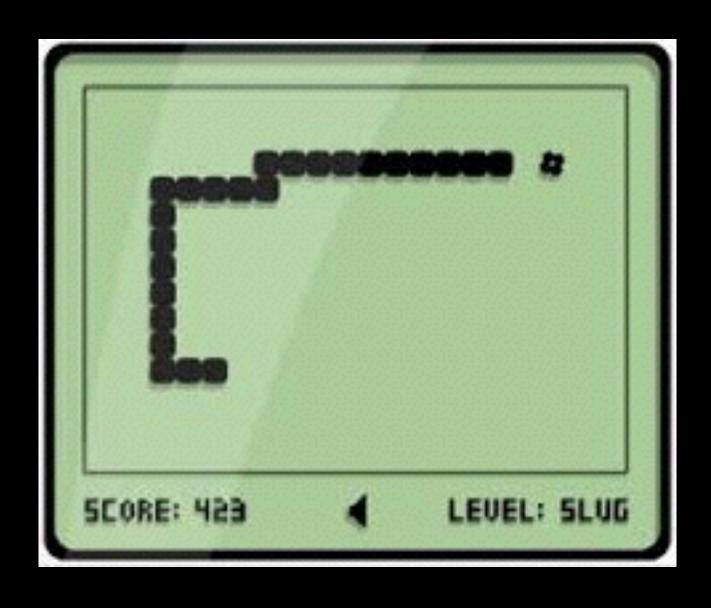
Project

- 1. Each student one project.
- 2. Each student one facilitator.
- 3. Scope must be modest.
- 4. Deadline on Friday 4th of Nov.

Your project plan

I. See the PDF on the course repository for instructions.

Example snake game



Example snake game

Snake:

attr: position

attr: length

move(self)

World:

attr: food pos

attr: size

new food(self)

Input:

attr: current_key

attr: last_key

Display:

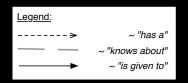
attr: last_update

draw(self)

15 minutes break



Composition



Status

Prints the status of the project on the terminal.

Project

(Subclass of Aggregate)

- Json path
- Many samples

SLURM Job

ProjectRunner

Will run stuff on the

login node or on the SLURM system

Id, status.

- Assemblies
- etc.

Parameters (JSON)

- * Must tell the pipeline where the original fastq.gz files are
- * Contains optional project and
- sample metadata
- * Contains parameters for running the algorithm including:
- kmer length
- contig length threshold
- etc

Report Auto generated PDF

Assembly
Calls Assembly
- km Calls Assembly

- leng - kme Calls Ray
- leng - kmer size
- length cutoff

Merged assembly

Calls Newbler - length cutoff

SLURM Job

SampleRunner

Will run stuff on the

login node or on the

SLURM system

Report

Auto generated PDF

ld, status

Forward reads
FASTQ

Reverse reads

PairedFASTQ

Will give you pairs of reads

Sample

- Fwd path - Rev path - Mapper Sayr - Report FASTQ

Forward reads

FASTQ

Reverse reads

PairedFASTQ

Will give you pairs of reads

Cleaner Makes a clean

Report
Auto generated PDF

SAM file

BAM file

Coverage dict

Linkage dict

keys are contigs

lorem

lorem

lorem

Mapper
Calls Bowtie
- Mean coverage
- Coverage fraction

FASTA file all contigs

generated

Linkage

Takes care of computing the mate pair information

Stats
- L50 and other stats

Taxonomy

Runs Phylophlan
Returns a tree
- Coverage of a Coverage of

Calls concoct

Coverage dataframe

Composition

Live demo

Databases



Databases

- I You can query data in a database (ask it questions).
- 2 You can look up data from a database relatively rapidly (index).
- 3 You can relate data from two different tables together using JOINs.
- 4 You can create meaningful reports from data in a database.
- 5 Your data has a built-in structure to it.
- 6 Information of a given type is always stored only once.
- 7 Databases are ACID.
- 8 Databases are fault-tolerant.
- 9 Databases can handle very large data sets.
- 10 Databases are concurrent
- II Databases scale well.

When to use files

- You have unstructured data in reasonable amounts that the file system can handle
 - You don't care about structure, relationships
- You don't care about scalability or reliability (although these can be done, depending on the file system)
- You don't want or can't deal with the overhead a database will add
- You are dealing with structured binary data that belongs in the file system, for example: images, PDFs, documents, etc.

Technology

- I SQLite 3
- 2 MySQL.
- 3 postgresql
- 4 MongoDB.

••••

Live demo

Exercise

- I. Get the file "test.db" from the repo.
- 2. Make a script that connects to that file (It's an SQLite 3 database file).
- 3. Get a list of all the tables in that file.
- 4. Print that list on the screen.

Solution

```
oldsymbol{1} # Use the sqlite3 library #
 2 import sqlite3
 3 # Absolute path #
 4 input_path = 'test.db'
5 # Open the database #
 6 connection = sqlite3.connect(input_path)
 7 cursor = connection.cursor()
8 # Make an SQL statement #
  query = "SELECT name FROM sqlite_master WHERE type='table'"
10 # Execute it #
11 cursor.execute(query)
12 # Fetch the results #
13 tables = cursor.fetchall()
14 # Display #
15 for t in tables: print t[0]
16
```

Exercise

- I. Get the file "lulu_mix_16.csv".
- 2. Make a script that writes the contents of that file to a database.
- 3. You should make one table called "songs" that has three columns.

<u>Solution</u>

```
1 # Use the sqlite3 library #
2 import sqlite3
3 # Load the whole text file in memory #
4 input_path = "lulu_mix_16.csv"
5 features = [l.strip('\n').split(',') for l in open(input_path) if not
   l.startswith('#')]
6 # Create database #
  output_path = "lulu_mix_16.db"
  connection = sqlite3.connect(output_path)
g connection.text_factory = str
10 cursor = connection.cursor()
11 # Create table #
12 columns = ('title', 'artist', 'duration')
13 cursor.execute("CREATE TABLE 'songs' " + str(columns))
14 # Insert everything #
15 cursor.executemany("INSERT INTO 'songs' VALUES (?,?,?)", features)
16 # Save changes #
17 connection.commit()
18 connection.close()
```

15 minutes break

