User guide

CLI

Demo

```
neroman --cluster mycluster localhost none
neroman --experiment ~/experiments/sleep
neroman --submit mycluster sleep2
neroman --monitor sleep2
```

Installing

~/.neronet/ - clusters.yaml - preferences.yaml - db.sqlite

clusters.yaml

```
clusters:
  triton:
    ssh_address: triton.aalto.fi
    type: slurm
    queue_max: 20
    special_attr1: foobar
  gpu1:
    ssh_address: gpu1
    type: unmanaged
  gpu2:
    ssh_address: gpu2
    type: unmanaged

groups:
  gpu: [gpu1, gpu2]
```

preferences.yaml

```
name: Pyry Takala
email: pyry.takala@gmail.com
```

db.sqlite

Defining cluster setup (S1)

Using the following command Neronet defines a new cluster into its database and tests whether it can access and operate with it.

```
Usage: neroman --cluster ID SSH_ADDRESS TYPE
Example: neroman --cluster triton triton.cs.hut.fi slurm
```

Neronet syncs the clusters.yaml based on input via CLI or GUI and notices direct file changes.

Defining experiments (s1)

```
Usage: neroman --experiment FOLDER
Example: neroman --experiment ~/experiments/lang_exp
```

Experiment folders must include a YAML config file named `neronet.yaml` of the following format:

```
# EXPERIMENT_ID: RUN_COMMAND PARAMETERS
lang_exp1: python3 lang_exp.py 1 2 3 data/1.txt
lang_exp2: python3 lang_exp.py 2 2 1 data/1.txt
lang_exp3: python3 lang_exp.py 3 2 1 data/2.txt

# EXPERIMENT_ID; RUN_COMMAND; PARAMETER1; PARAMETER2; PARAMETER3;
lang_exp1; python3 lang_exp.py; 1; 2; data/1.txt;
lang_exp2; python3 lang_exp.py; 2; 2; data/1.txt;
lang_exp3; python3 lang_exp.py; 1; 3; data/3.txt;
```

pakotettu (CSV yhteensopiva) yaml.

Another alternative pyexp config.yaml # pyexp financial_exp1 config.yaml financial_exp2 config.yaml financial_exp3 config.yaml speech_exp config.yaml lang_exp config.yaml lang_exp_x config.yaml code.py submissions submission1-201511161121 code.py run1 config.yaml stdout.log plot.png run2 config.yaml stdout.log plot.png run3 submission2-201511161243 code.py run1 config.yaml stdout.log plot.png lang_exp_y config.yaml code.py lang_exp_z config.yaml code.py

```
# pyexp/config.yaml
run_command_prefix: python3
main_code_file: main.py
```

```
# lang_exp/config.yaml
parent: pyexp
parameters:
  hyperparamx: [1, 3, 5, 1]
  hyperparamy: [4, 2, 5, 7]
  data_file: data/1.txt

# lang_exp_x/config.yaml
parent: lang_exp
parameters:
  hyperparamz: [2, 3, 1, 7]

# lang_exp_x/results/batch1/run1/config.yaml
run_command_prefix: python3
main_code_file: main.py
parameters:
  hyperparamx: 1
  hyperparamy: 4
  data_file: data/1.txt
  hyperparamz: 2

# lang_exp_x/results/batch1/run2/config.yaml
run_command_prefix: python3
main_code_file: main.py
parameters:
  hyperparamx: 2
  hyperparamy: 4
  data_file: data/1.txt
  hyperparamz: 2

# lang_exp_x/config.yaml
parent: lang_exp
parameters:
  hyperparamz: [2, 3, 1, 7]
```

Format

```
# lang_exp_x/results/batch1/run1/config.yaml
run_command_prefix: python3
main_code_file: main.py
parameters:
  hyperparamx: 1
  hyperparamy: 4
  data_file: data/1.txt
```

```
  hyperparamz: 2
parameter_format: '--hyperparamx %s{hyperparamx} %hyperparamy

parameter_format.format(**parameters)
parameter_names: 'attr1', 'attr2'
variables:
  hyperparamx: [1,2,3,4]
  hyperparamy: 2
```

neronet –csv-edit lang_exp –children neronet –list-experiments –details

The experiment IDs must be unique.

Submitting experiments to be run (S1)

```
Usage: neroman --submit CLUSTER_ID EXPERIMENT_ID
Example: neroman --submit triton lang_exp3
```

Tasks can be submitted also by logical arguments:

```
Usage: neroman --submit CLUSTER_ID ARGUMENT
Example: neroman --submit triton ~/experiments/lang_exp
Example: neroman --submit triton 'tmod>yesterday'
Example: neroman --submit triton 'params=*data/1.txt*
```

Checking status (S1)

The status command gives status information regarding configurations and any specified clusters and experiments.

```
Usage: neroman --status [ARGS]
```

ARGS can refer to experiment or cluster IDs, or be collection specifiers.

```
Example: neroman --status # Overall status information
Example: neroman --status lang_exp3 # experiment status
Example: neroman --status 'tsub>yesterday' # collection status
Example: neroman --status triton # cluster status
```

Monitor experiment progress

```
Usage: neroman --monitor EXPERIMENT_ID
Example: neroman --monitor lang_exp3
```

GUI