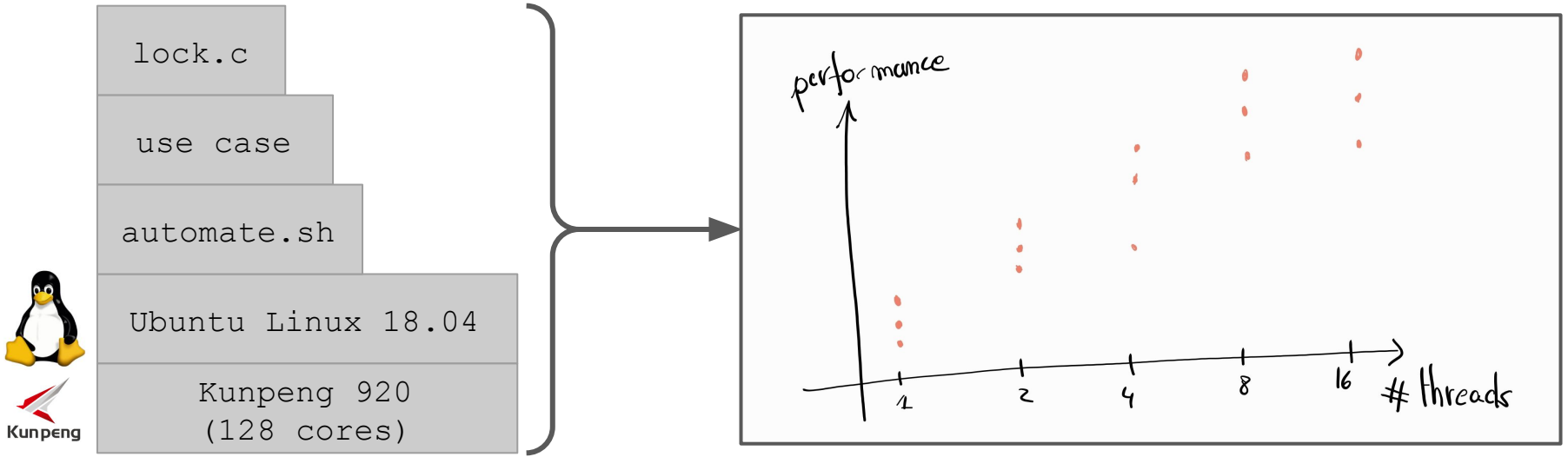
Evaluate All the Things with `benchkit`

Antonio Paolillo

29th of January, 2025 – FOSDEM BOF Track C

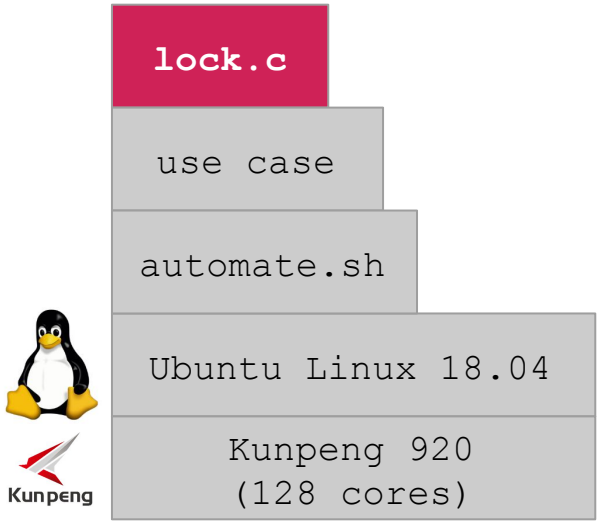
Multicore & Concurrency:
Algorithms, Performance, Correctness

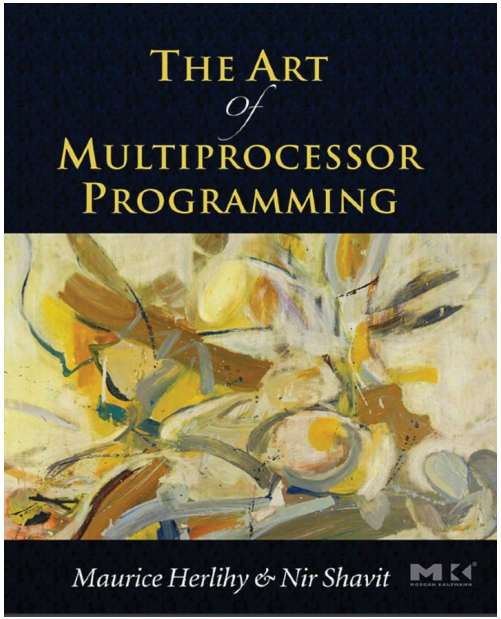
Example use case: evaluating locks

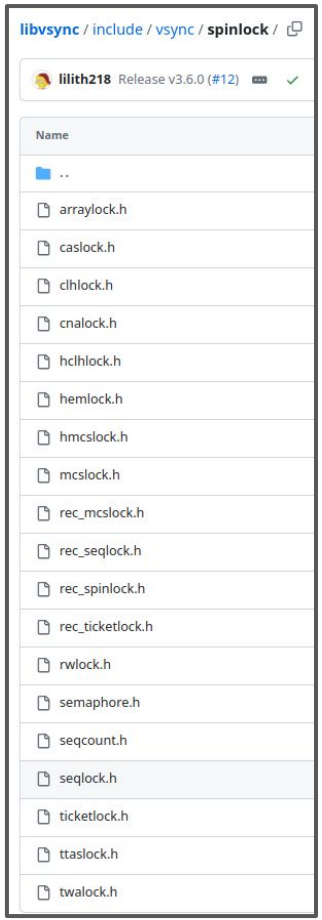


Locks implementation are plenty

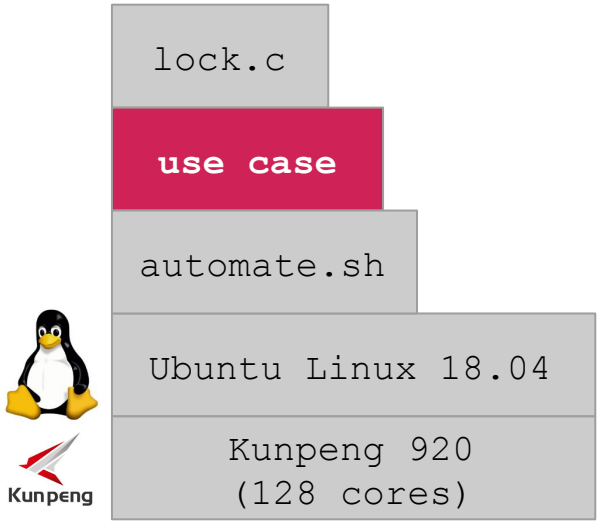
e.g. [libvsync]



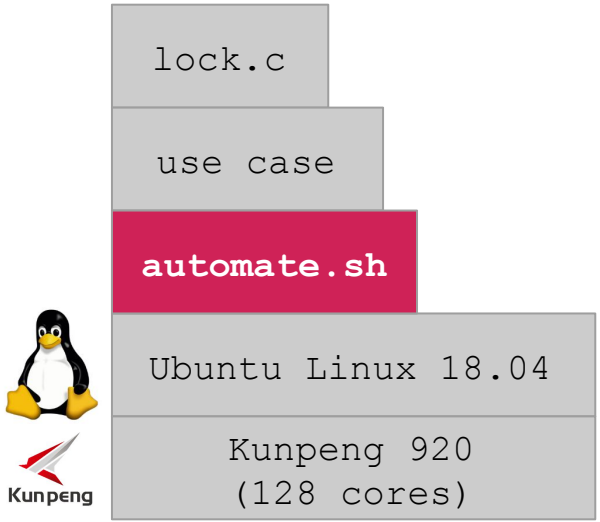




Benchmarking locks in many use cases

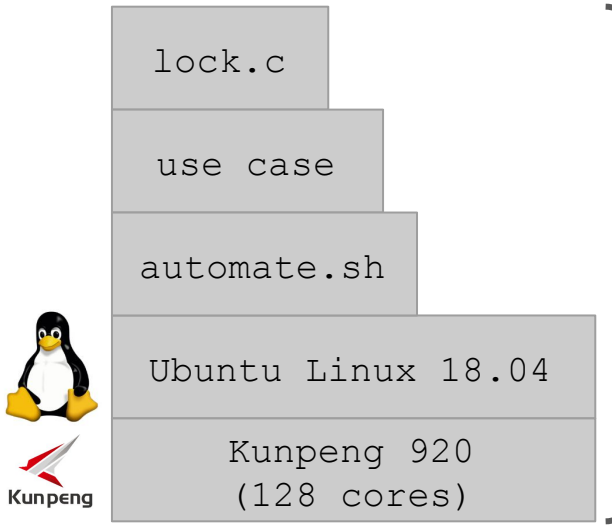


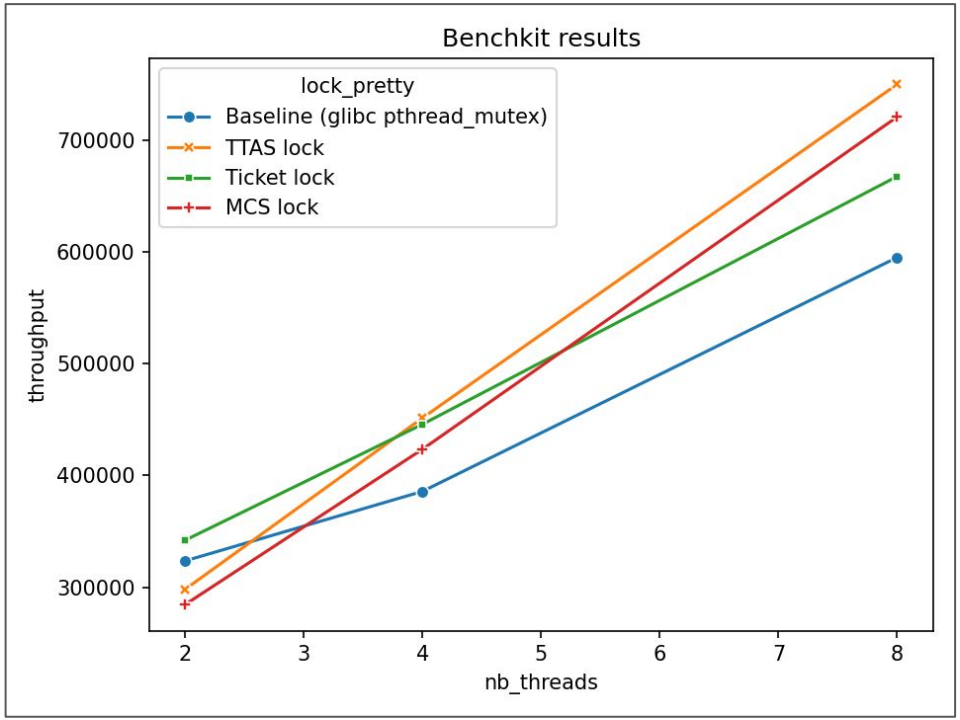
Writing shell script to evaluate locks?



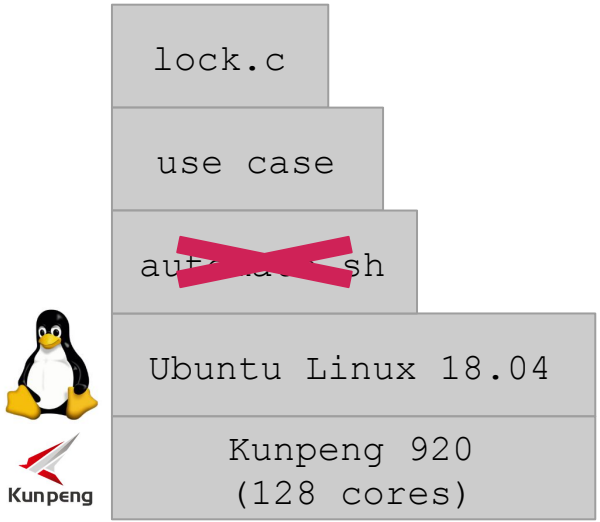

```
nb_threads="1 2 4 8"
bench_name=readrandom
locks="spin mcs ticket clh"
lses="on off"
echo "bench_name;lock;lse;nb_thread;microsop" | tee /tmp/results.csv
for lse in ${lses}
do
 for lock in ${locks}
 do
   make -C ../tilt ${lock} LSE=${lse}
   for nb_thread in ${nb_threads}
   do
     taskset --cpu-list 0 env LD_PRELOAD="../tilt/${lock}.so" ./db_bench
--benchmarks=${bench_name} --threads=${nb_thread} \
         > /tmp/leveldb_results.txt
     results=$(cat /tmp/leveldb_results.txt | tail -n 1 | grep -o "[0-9.]\+ micros/op" |
cut -d ' ' -f 1)
     echo "${bench_name};${nb_thread};${results}" | tee -a /tmp/results.csv
   done
 done
done
```

Getting some results…



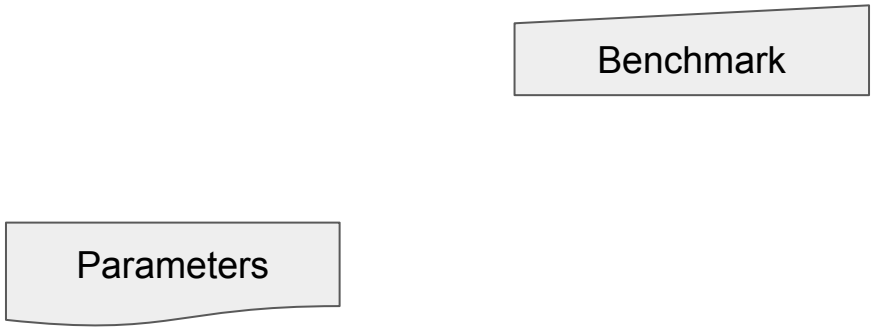


Shell scripts are getting complex quickly

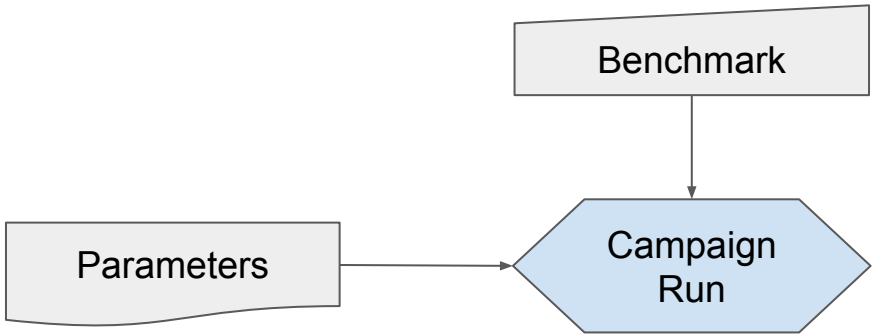



```
nb_threads="1 2 4 8"
bench_name=readra
locks="spin mcs
lses="on off"
echo "bench
for lse i
do
 for lock in $
 do
   make -C ../tilt ${lo
   for nb_thread in ${nb_t
   do
     taskset --cp
--benchmarks=
     resul
cut -d ' ' -
     echo "${b
   done
 done
done
```

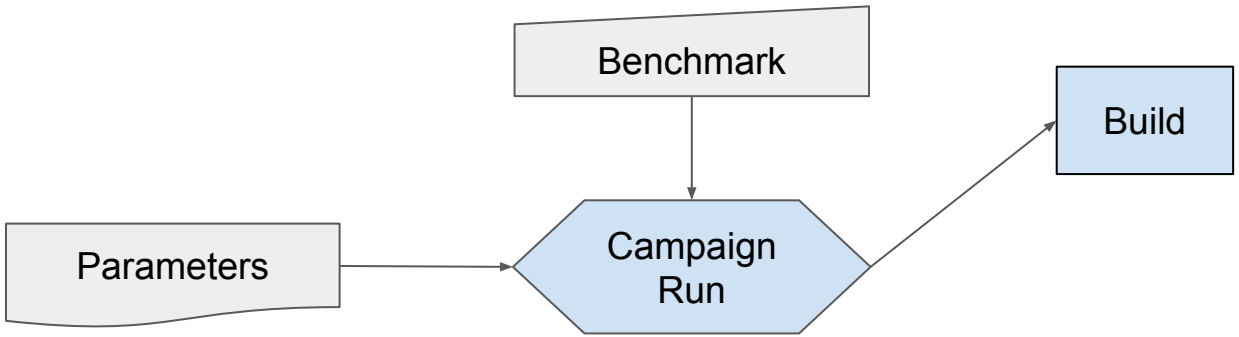
Benchkit flow



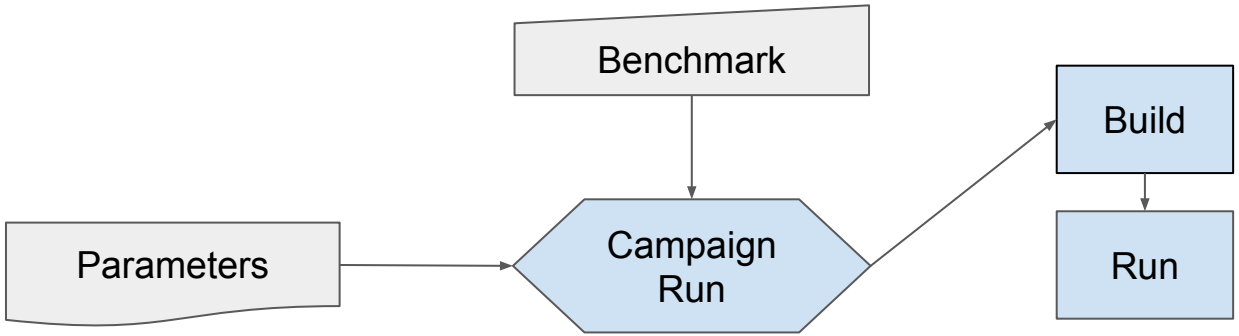
Benchkit flow



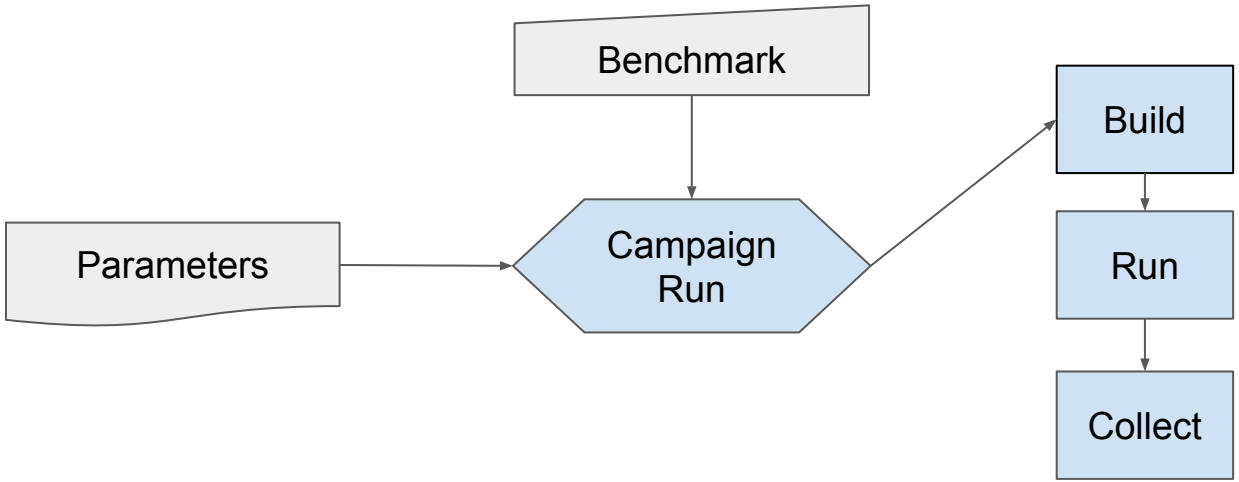
Benchkit flow



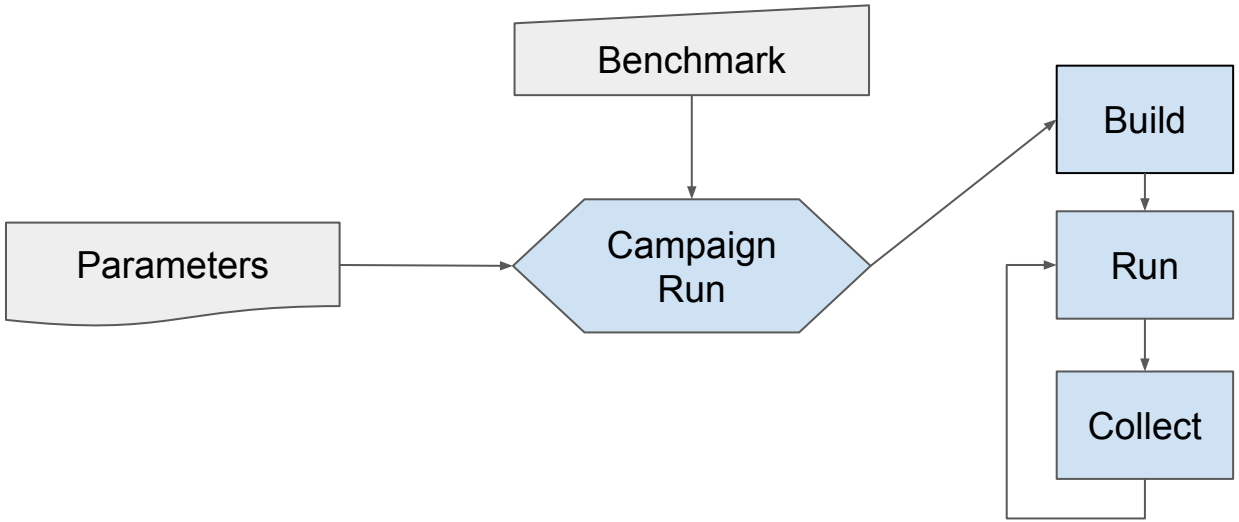
Benchkit flow



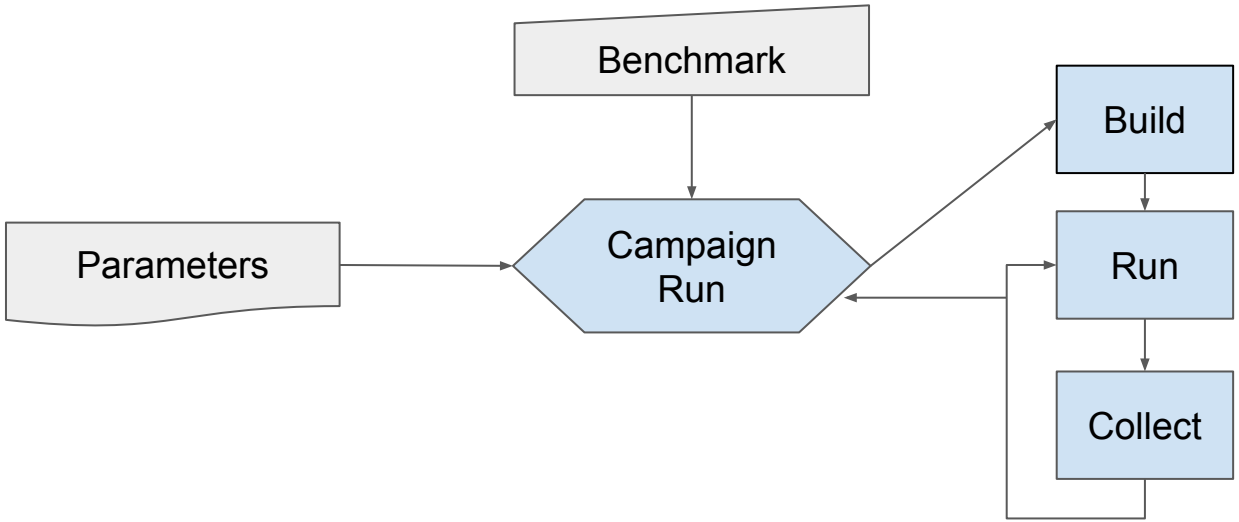
Benchkit flow



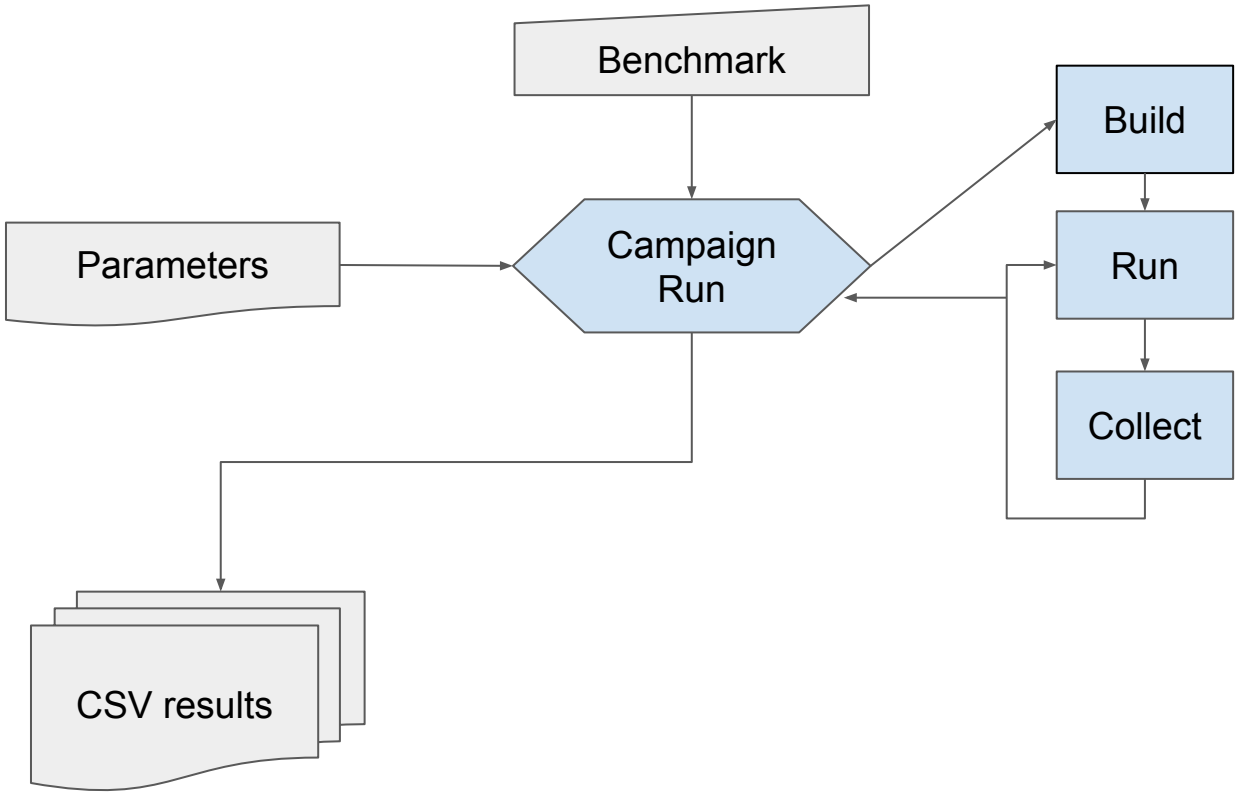
Benchkit flow



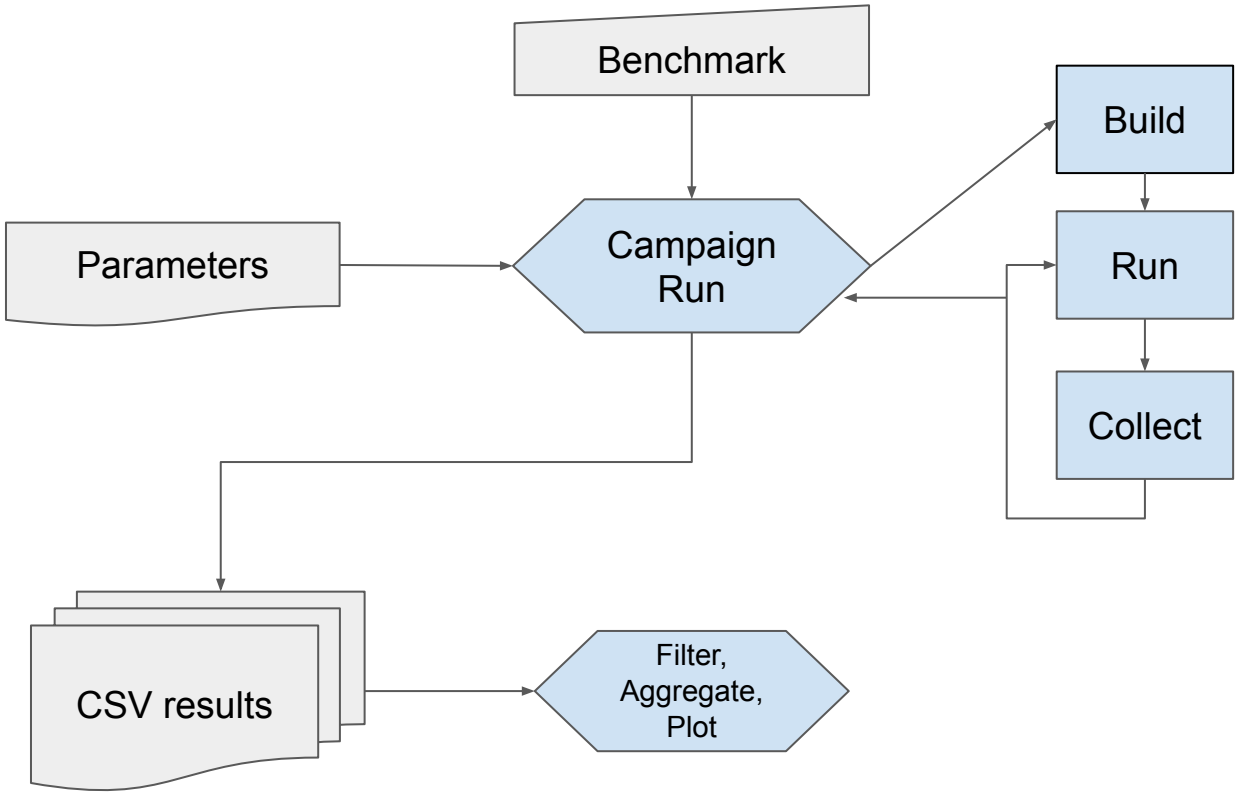
Benchkit flow



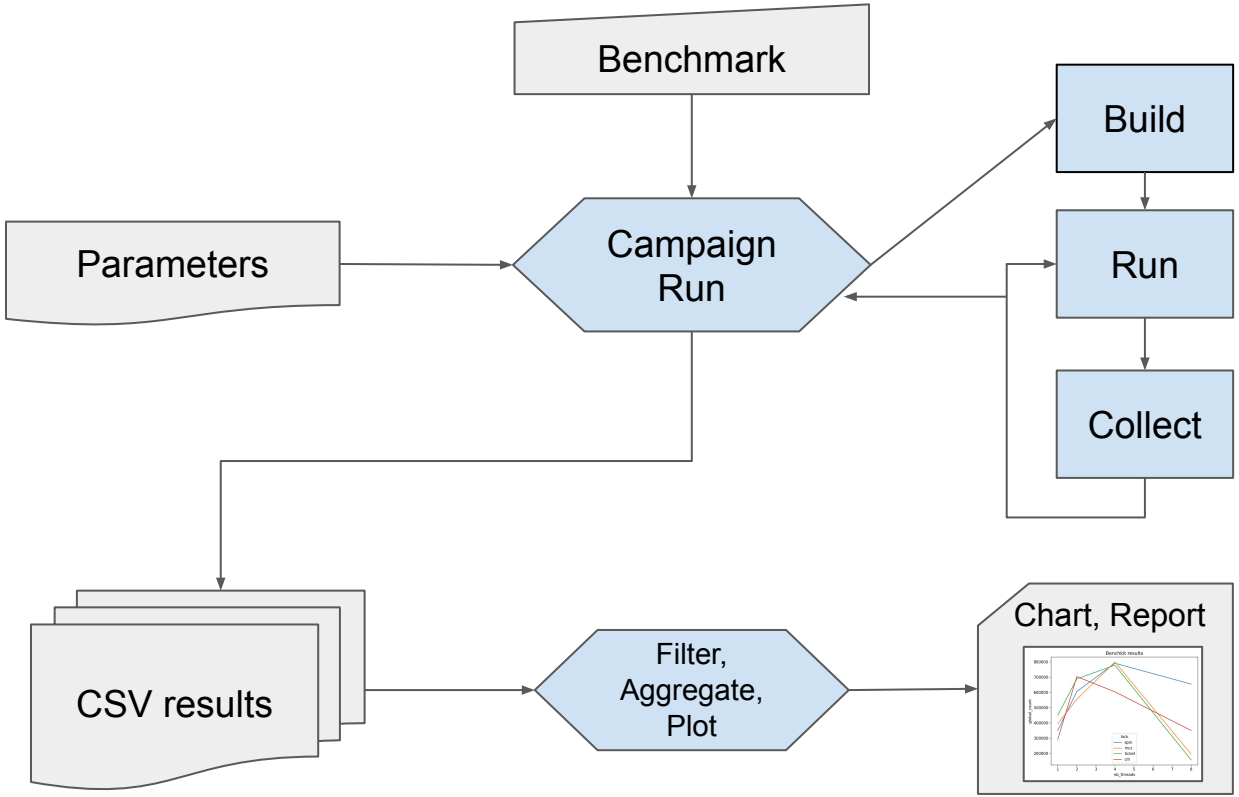
Benchkit flow



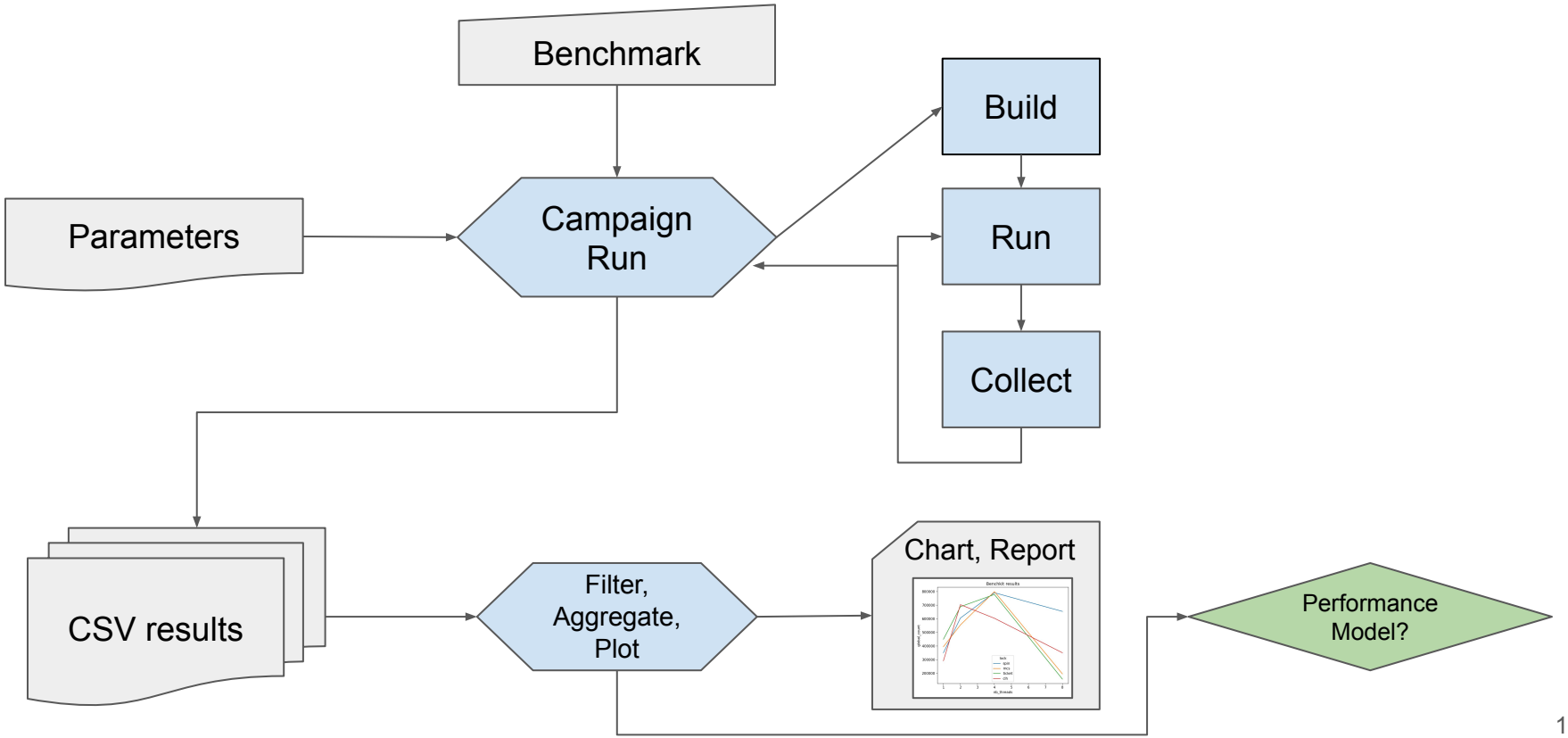
Benchkit flow



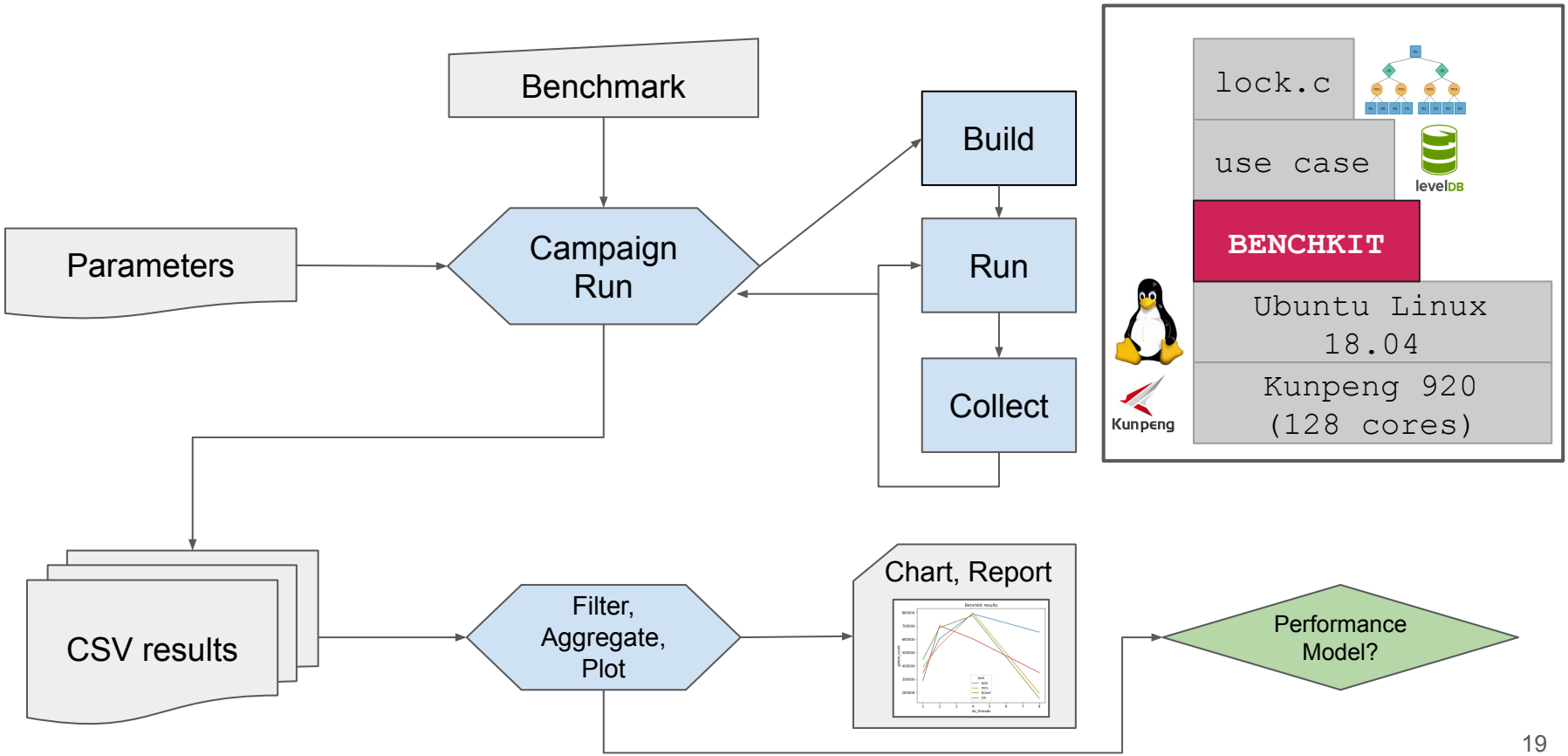
Benchkit flow



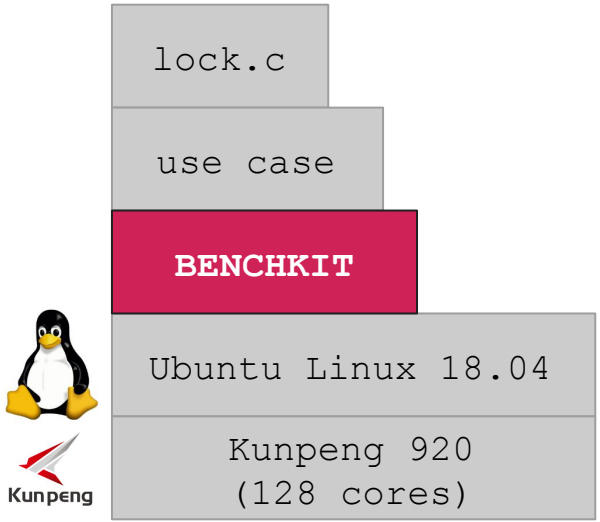
Benchkit flow



Benchkit flow



`benchkit` provides an alternative

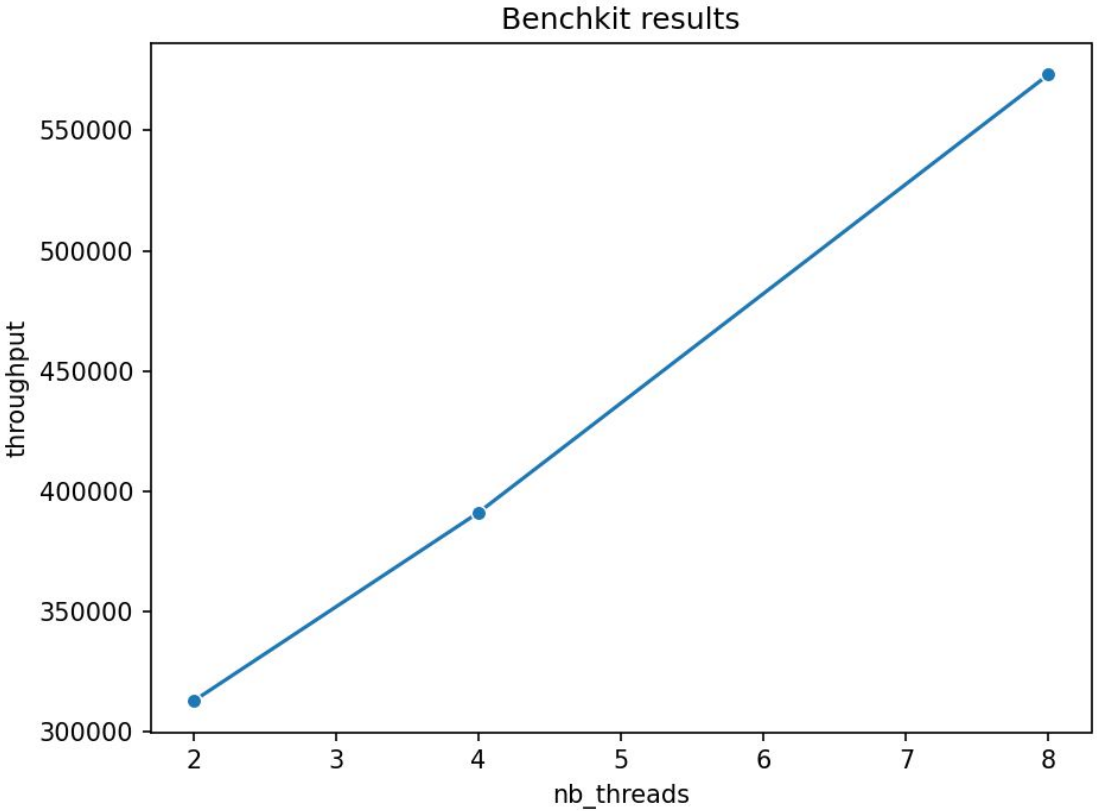



```
nb_threads="1 2 4 8"
bench_name=readrandom
locks="spin mcs ticket clh"
lses="on off"
echo "bench_name;lock;lse;r
for lse in ${lses}
do
 for lock in ${locks}
 do
   make -C ../tilt ${lock}
   for nb_thread in ${nb_th
   do
     taskset --cpu-list 0 e
--threads=${nb_thread} \
         > /tmp/leveldb_res
     results=$(cat /tmp/lev
     echo "${bench_name};${
   done
 done
done

ipython3 < EOF
df = pd.read_csv('/tmp/resu
ax = sns.lineplot(data=df,
                  x='nb_thr
                  y='micros
                  hue='benc
plt.show()
EOF
```

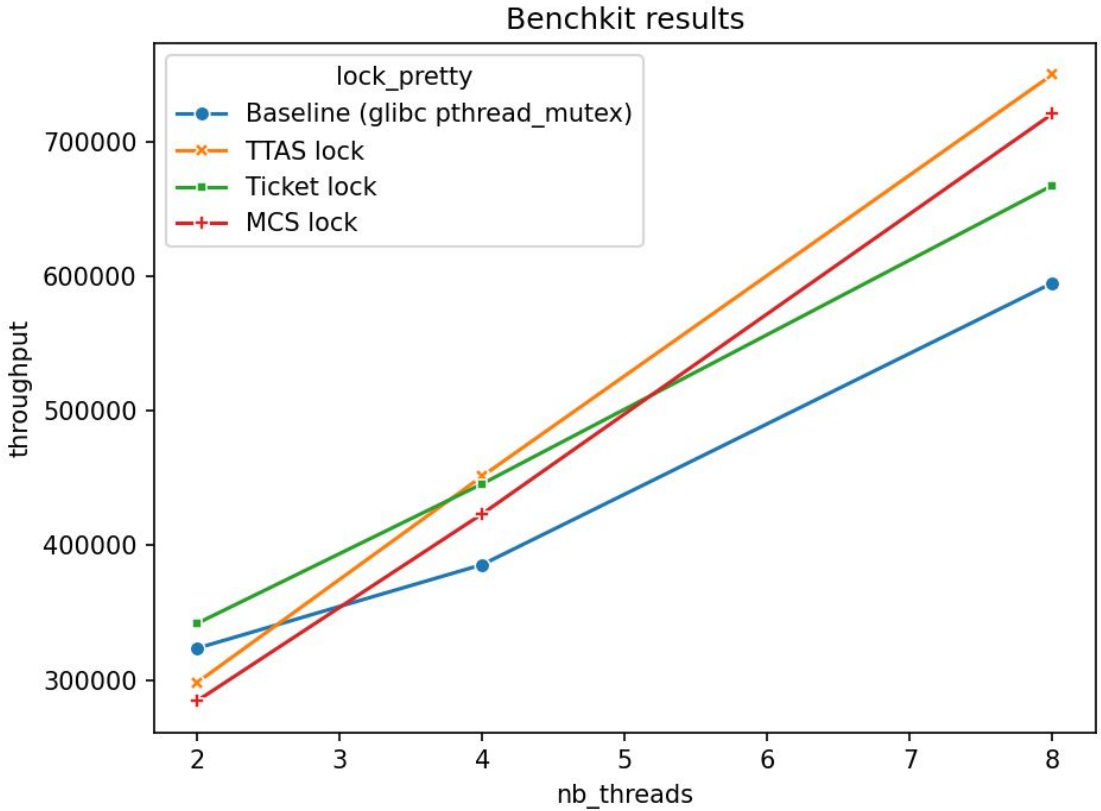
```
if __name__ == '__main__':
    campaign = leveldb_campaign(
        nb_runs=5,
        bench_name=['readrandom', 'fillseq', 'fillrandom'],
        locks=['spin', 'mcs', 'ticket', 'clh'],
        cpu_order=('desc',),
        use_lse=[False, True],
        atomics=('a64', 'blt'),
        nb_threads=[1, 2, 4, 8],
        shared_libs=[get_assignlib(),
                     get_fledgedtilt()],
        command_wrappers=[TasksetWrap()],
    )

    campaign.run()
    campaign.generate_graph(plot_name='lineplot',
                            x='nb_threads',
                            y='global_count',
                            hue='lock')
```

demo

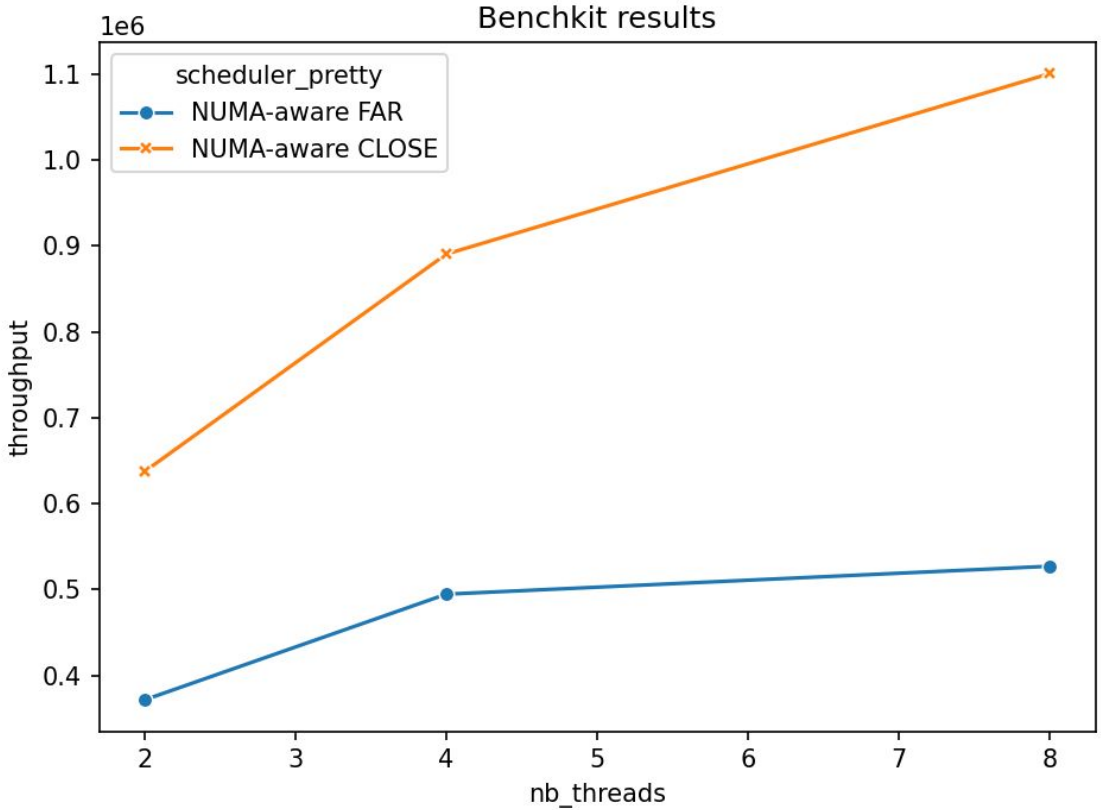
00 - LevelDB


Benchkit results
throughput
550000
500000
450000
400000
350000
300000
2
3
4
5
6
7
8
nb_threads

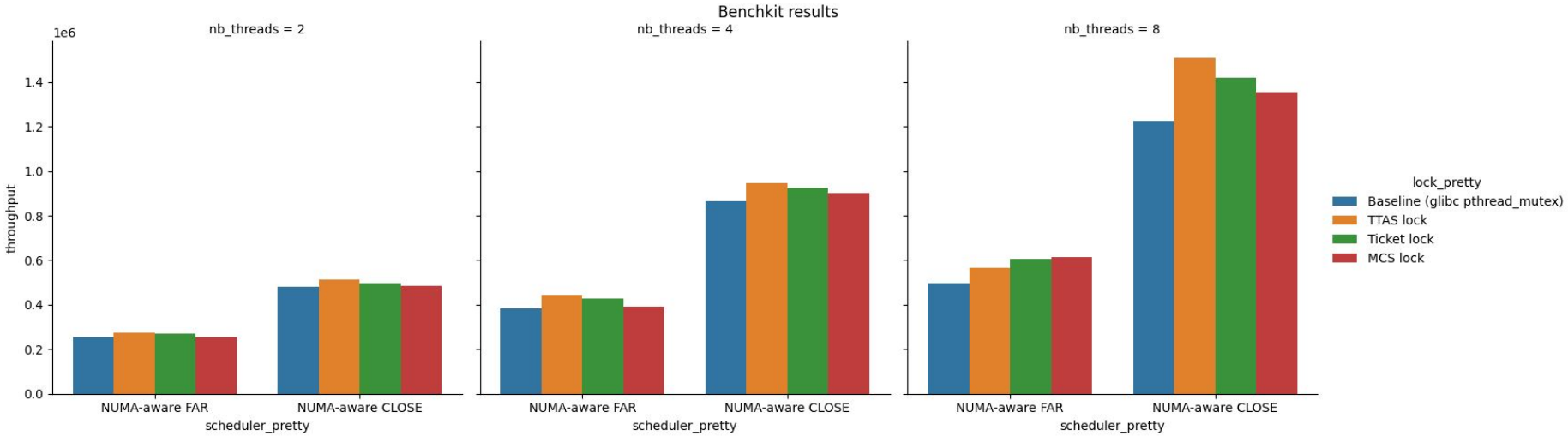
01 - LevelDB - various locks



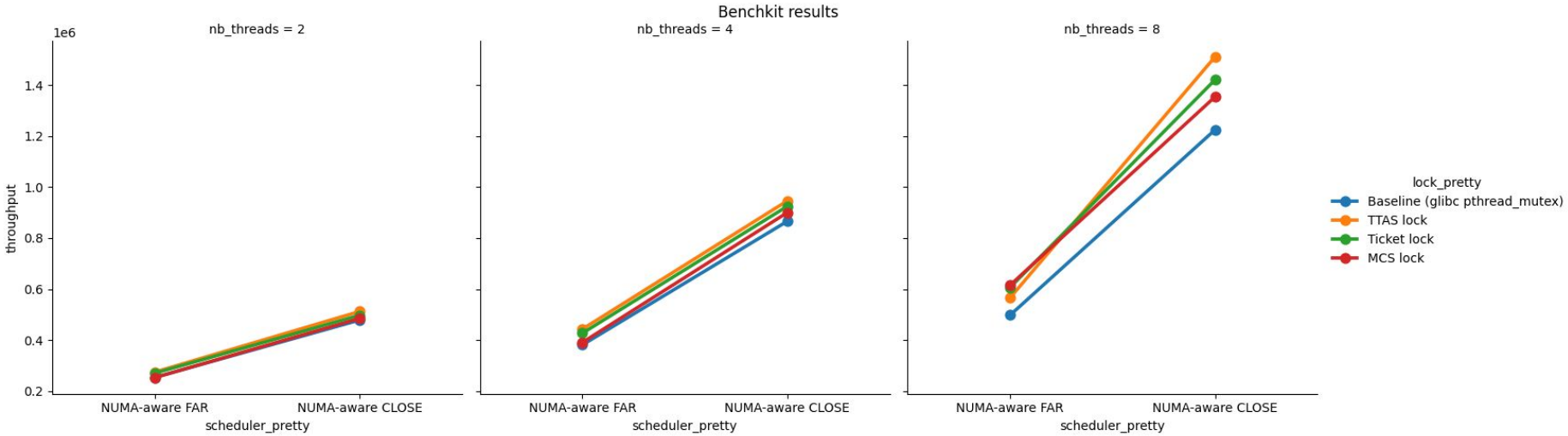
02 - LevelDB - various schedulers



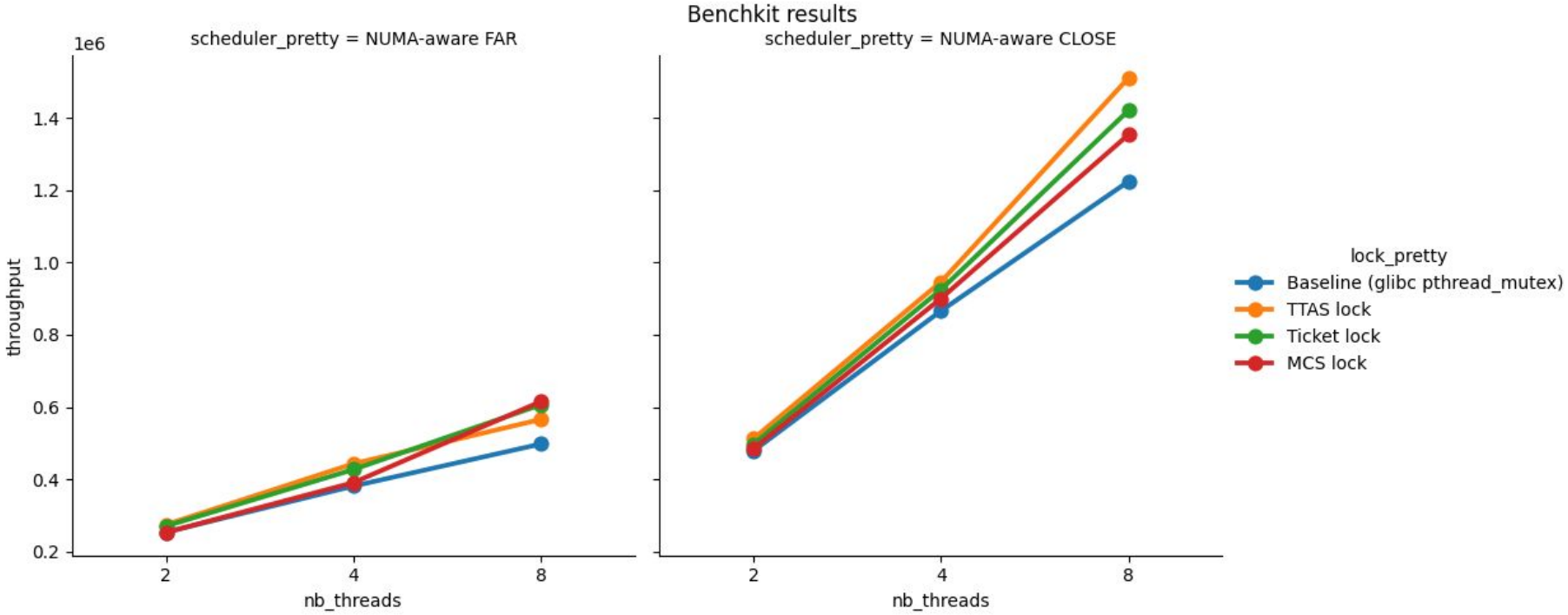
03a - LevelDB - various schedulers & locks



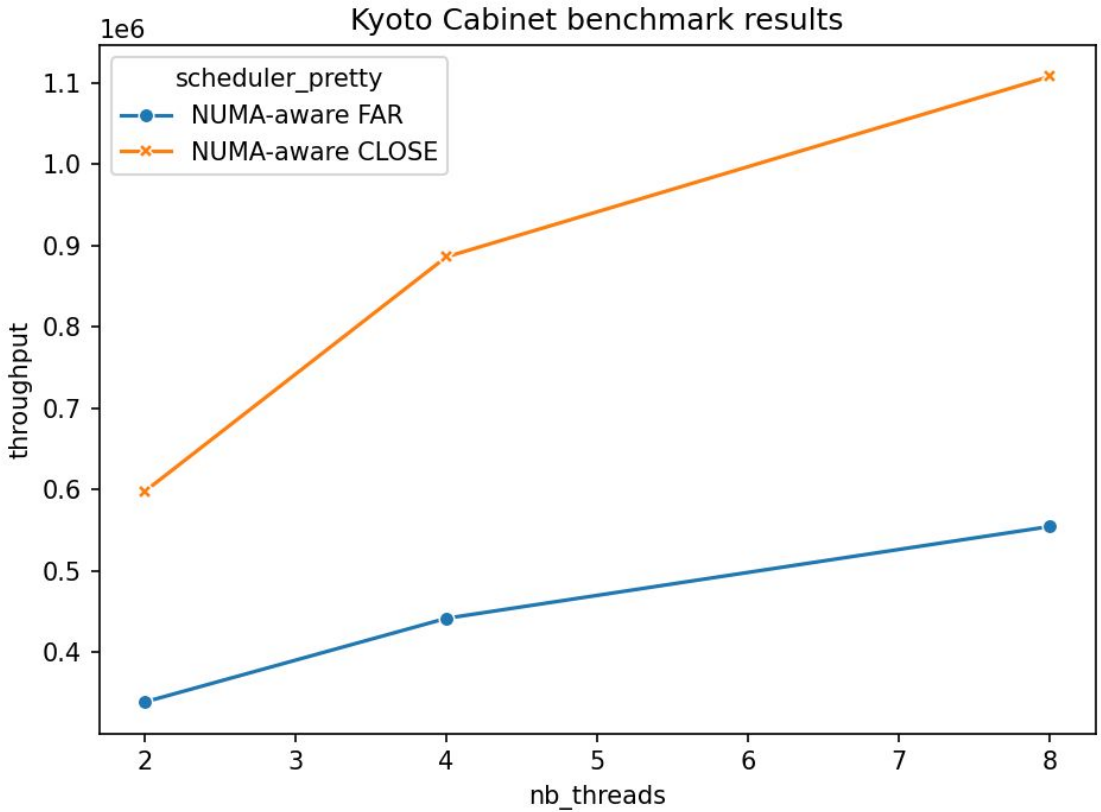
03b - LevelDB - various schedulers & locks



03c - LevelDB - various schedulers & locks



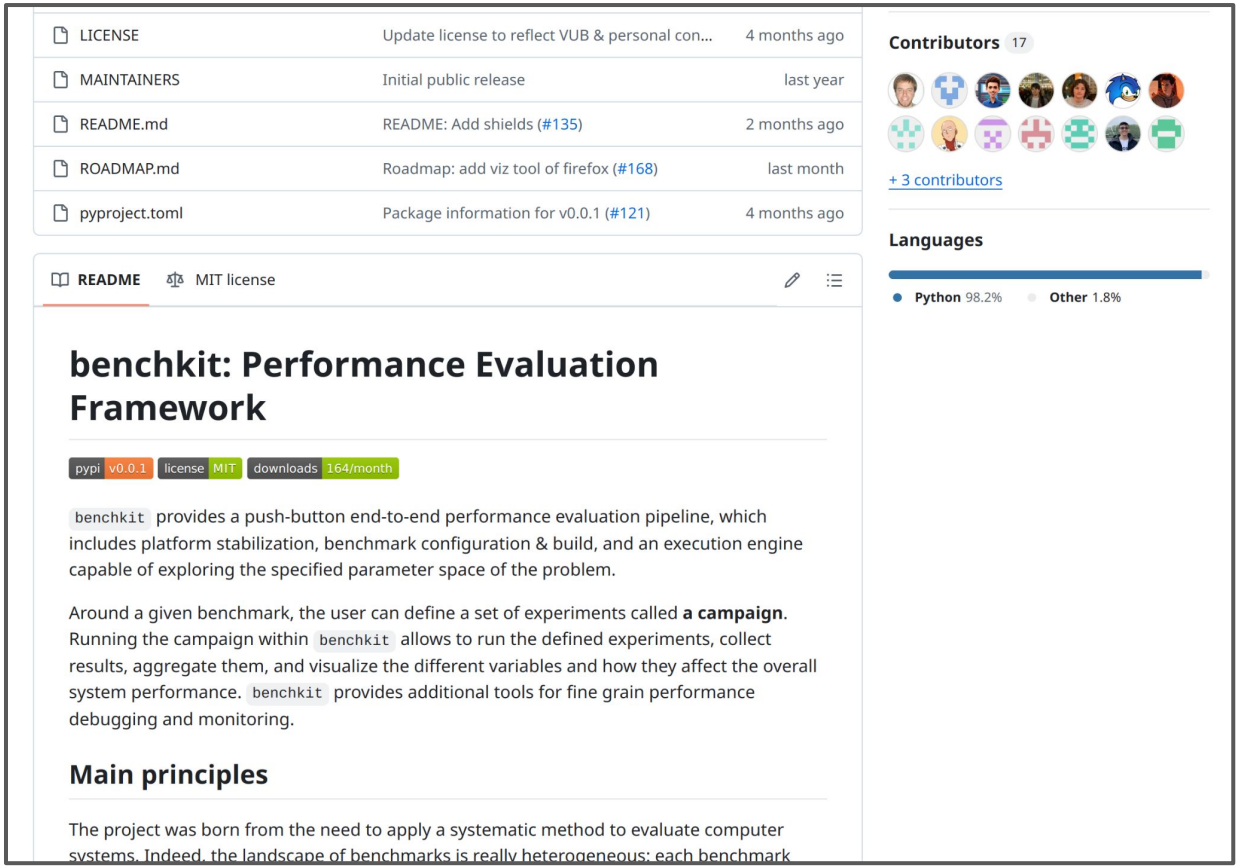
04 - Kyoto Cabinet - various schedulers



Try it! → https://github.com/open-s4c/benchkit

🌐 On GitHub

⚖️ **MIT** license



LICENSE	Update license to reflect VUB & personal con...	4 months ago
MAINTAINERS	Initial public release	last year
README.md	README: Add shields (#135)	2 months ago
ROADMAP.md	Roadmap: add viz tool of firefox (#168)	last month
pyproject.toml	Package information for v0.0.1 (#121)	4 months ago

README MIT license

benchkit: Performance Evaluation Framework

pypi v0.0.1 | license MIT | downloads 164/month

`benchkit` provides a push-button end-to-end performance evaluation pipeline, which includes platform stabilization, benchmark configuration & build, and an execution engine capable of exploring the specified parameter space of the problem.

Around a given benchmark, the user can define a set of experiments called **a campaign**. Running the campaign within `benchkit` allows to run the defined experiments, collect results, aggregate them, and visualize the different variables and how they affect the overall system performance. `benchkit` provides additional tools for fine grain performance debugging and monitoring.

Main principles

The project was born from the need to apply a systematic method to evaluate computer systems. Indeed, the landscape of benchmarks is really heterogeneous: each benchmark

Contributors 17

+ 3 contributors

Languages

Python 98.2% Other 1.8%

Interested in *Multicore & Concurrency*?

Contact us!

db7@sdf.org
hernanl.leon@huawei.com
antonio.paolillo@vub.be