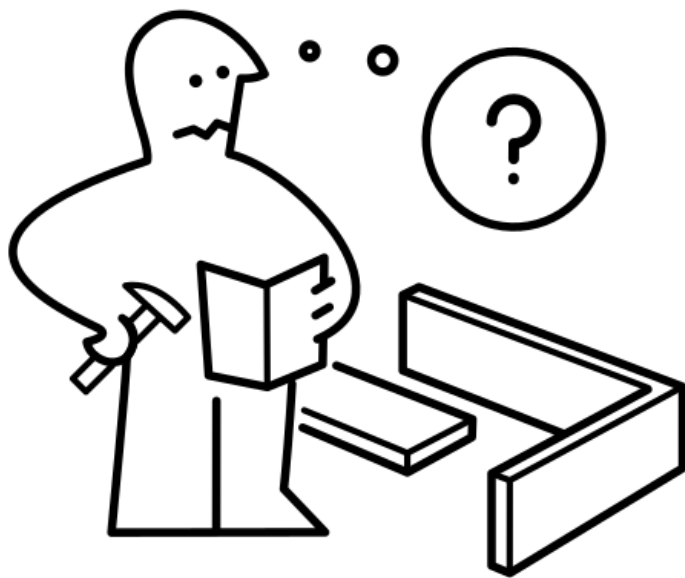
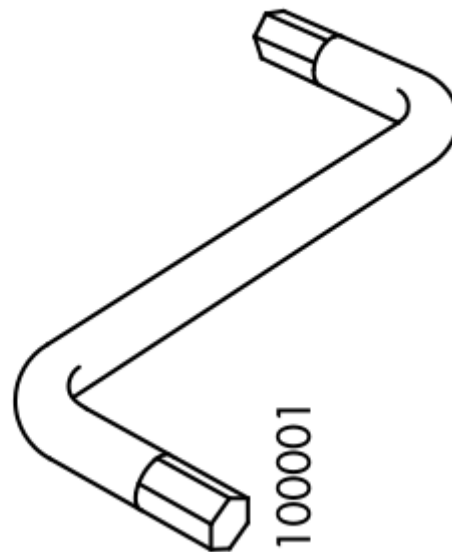


How poor can a stock perform?

Markus Bilz, 1705042



Greatest, likely loss?



Value at Risk
Simulation

Value at Risk using a Monte Carlo simulation

```
repeat n times -> first kernel
  repeat t times
    generate normal distributed number
    update interim price
    save end price to path array
extract the 0th rank -> second kernel
print results
```

1st kernel

(generating random prices)

specification

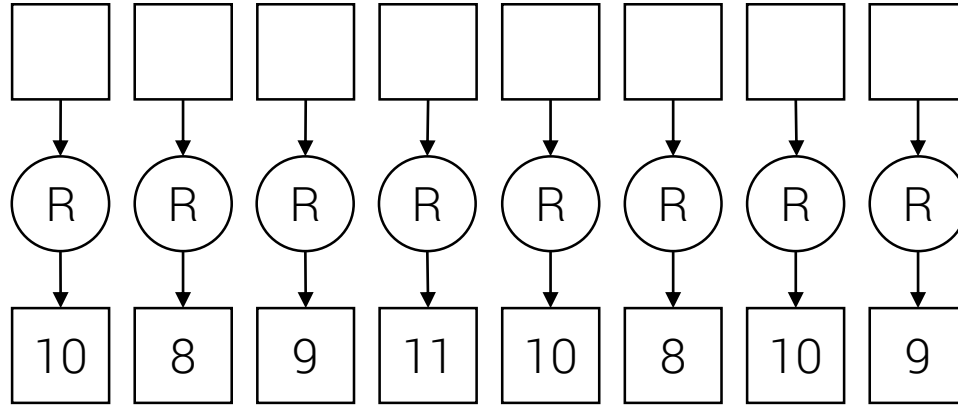
trivial problem / no interaction between threads

transform algorithm

All threads run the same code / no thread divergence

...

algorithm



own drawing inspired by Ade Miller cppcon 2014

2nd kernel

(extracting the minimal price)

specification

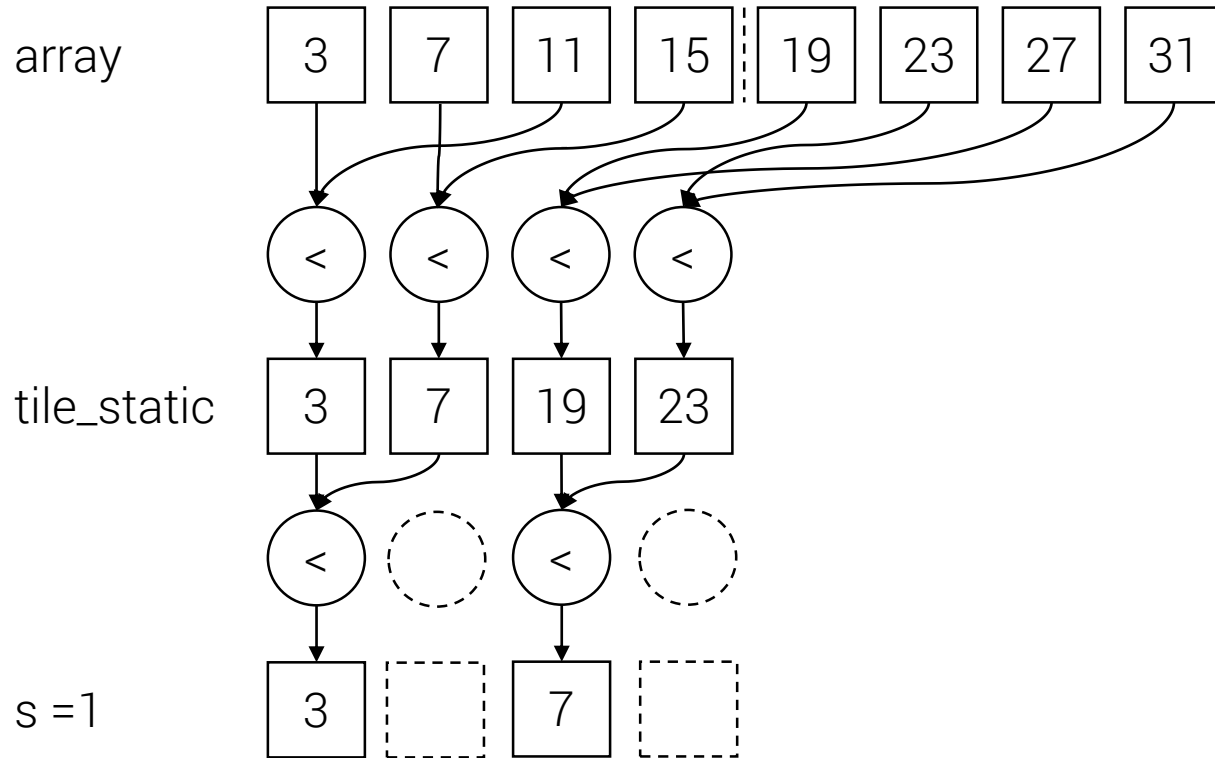
non trivial problem / dependence between threads

reduce algorithm

potential bank conflicts / idle threads

...

algorithm



own drawing inspired by Ade Miller cppcon 2014

performance evaluation

gpu specification

gpu specification

NVIDIA GeForce 940MX

misc

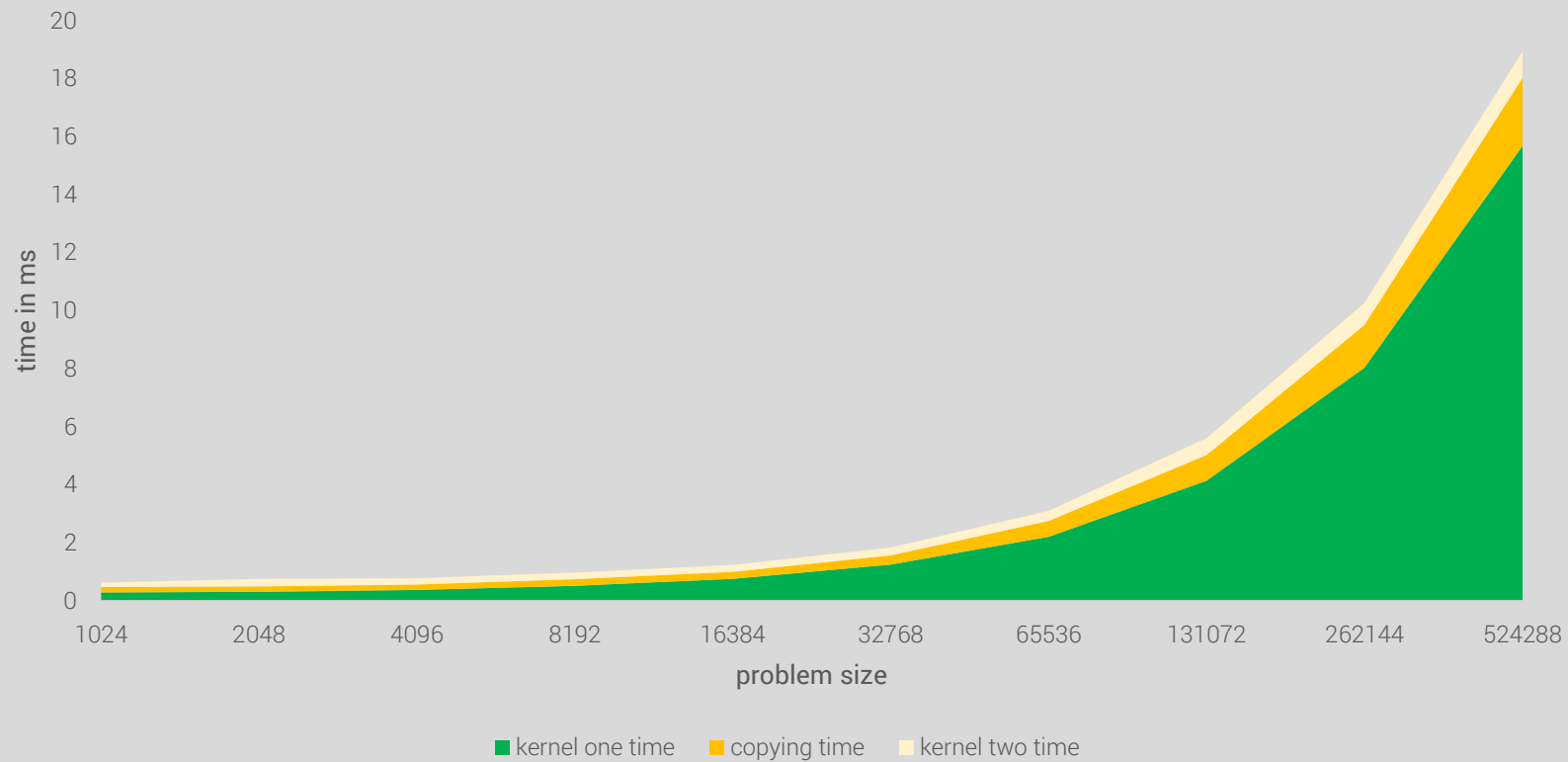
using wakeup call to prevent JIT and lazy initialization

using `std::chrono::steady_clock`

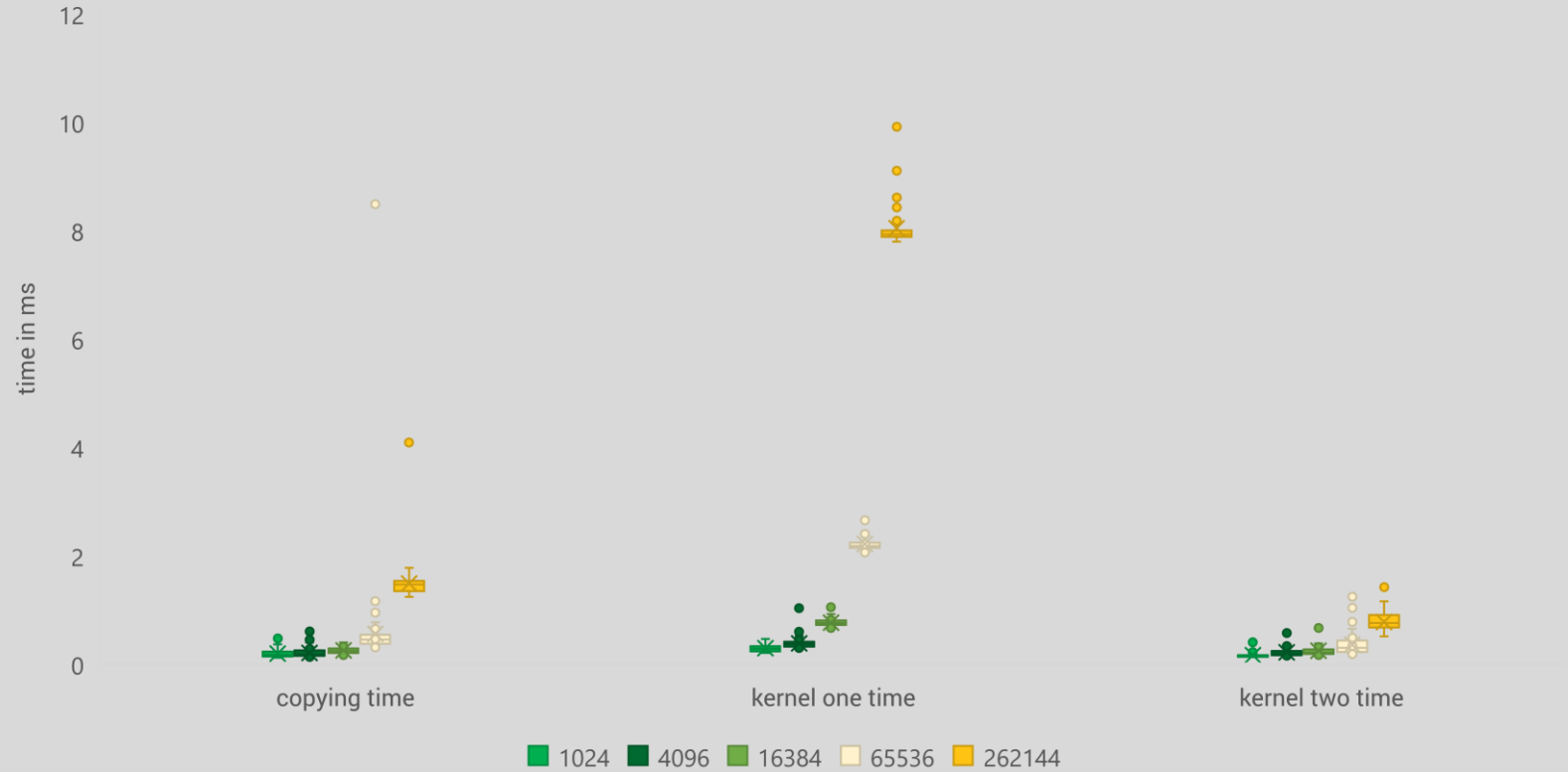
using `tile_size` with multiple of two

using templated functions for changing `tile_size`

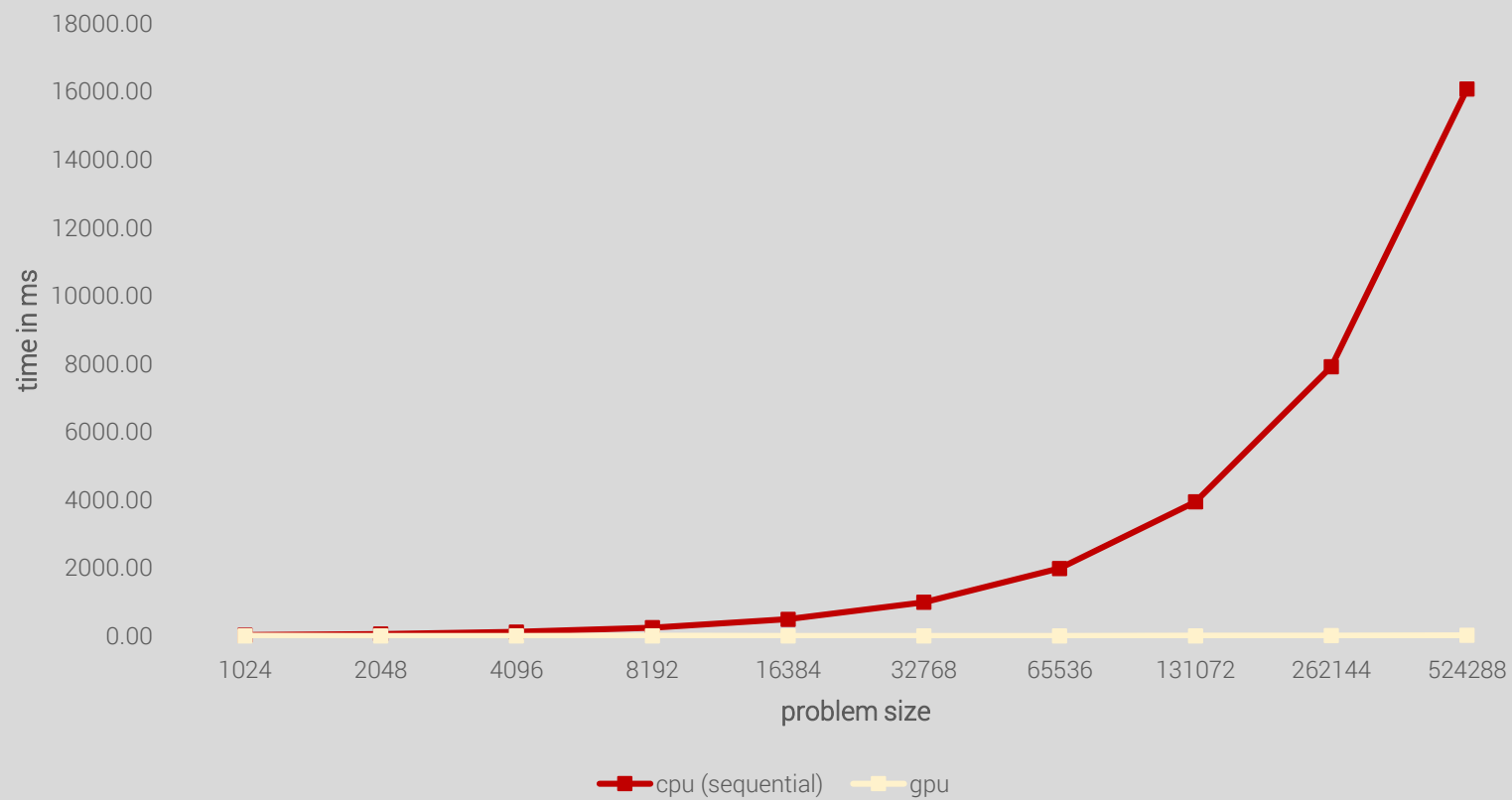
`accelerator_view::wait()`



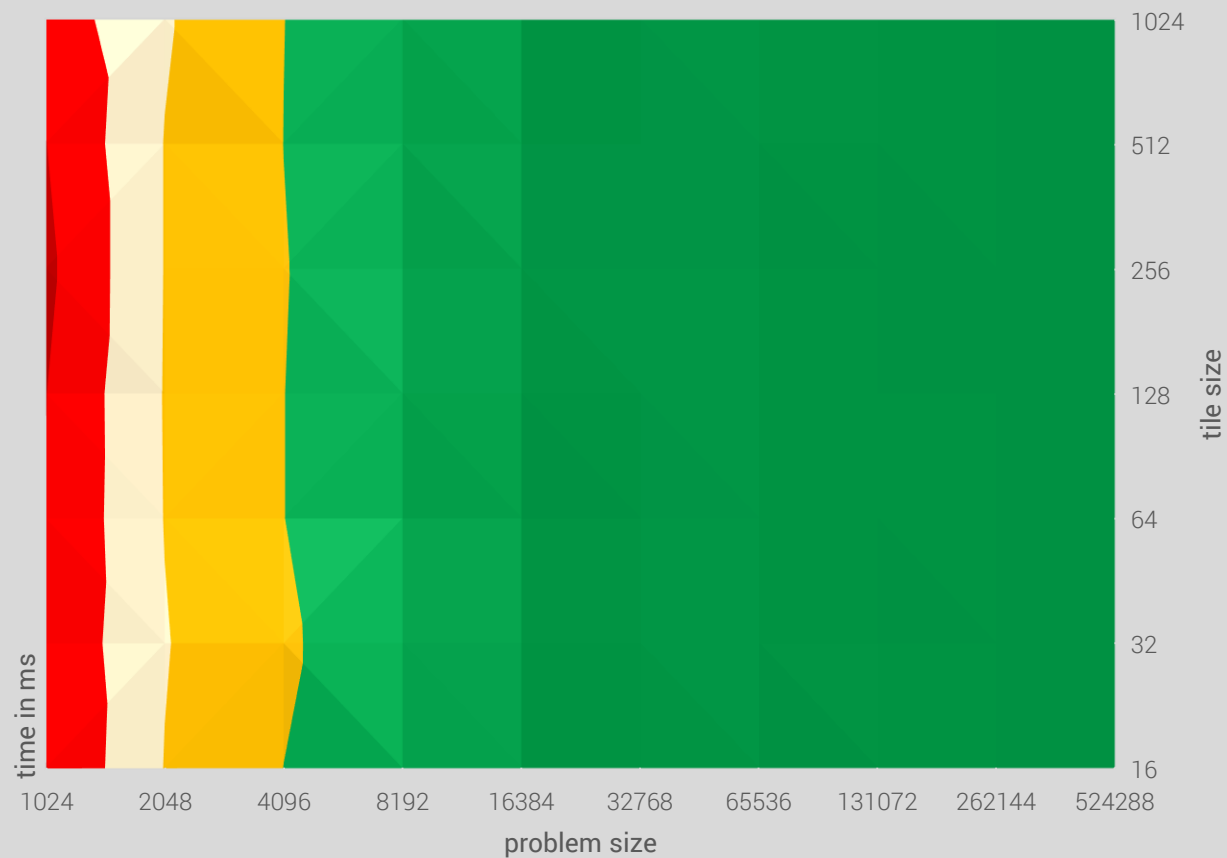
(avg. of 100 runs, ts = 1024)



(100 runs, ts = 1024)



(avg. of 3 runs)



0-0.00025 0.00025-0.0005 0.0005-0.00075 0.00075-0.001 0.001-0.00125

(avg. of 100 runs)

Slides shared under cc-by 4.0