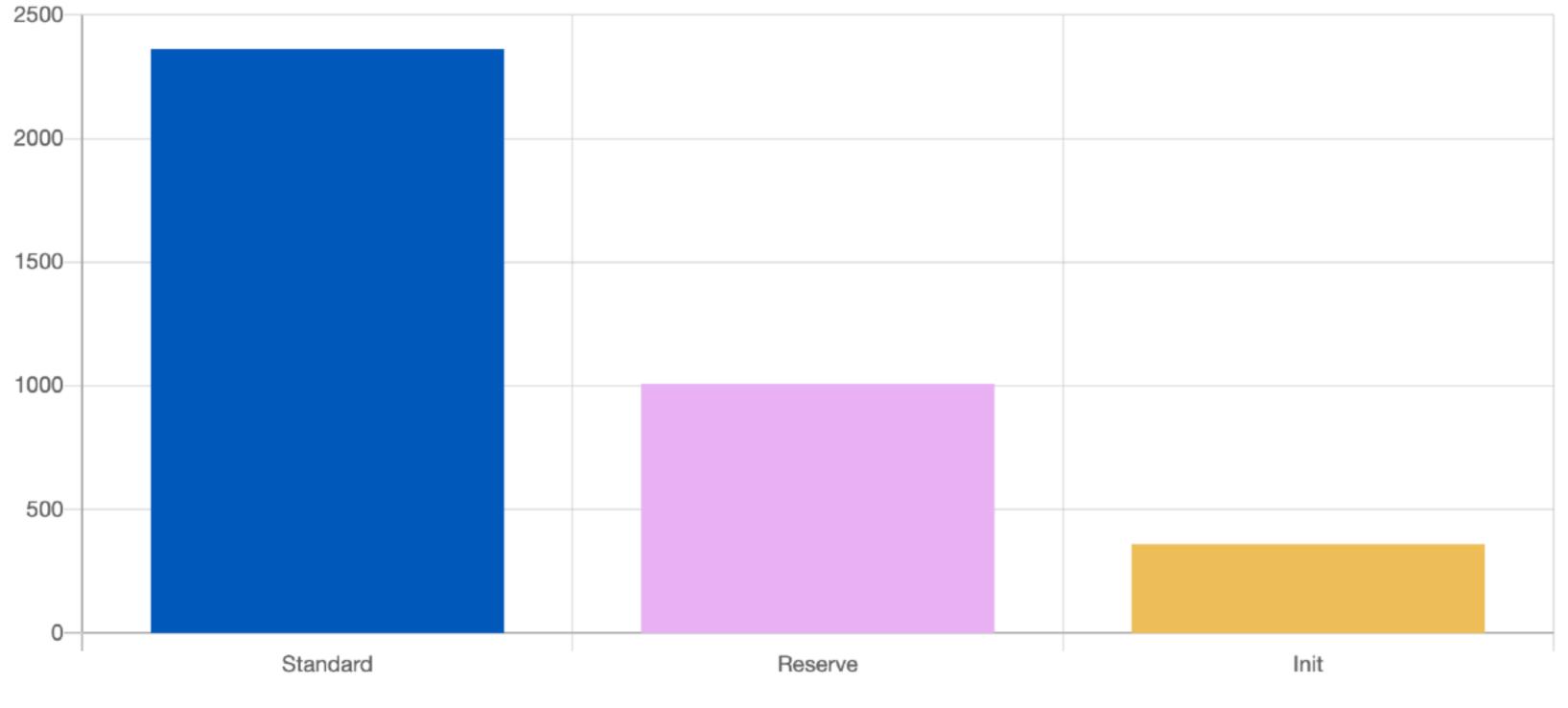
https://quick-bench.com/q/9uTY-mY6yFS1gIE7hIDBFjU5uNw



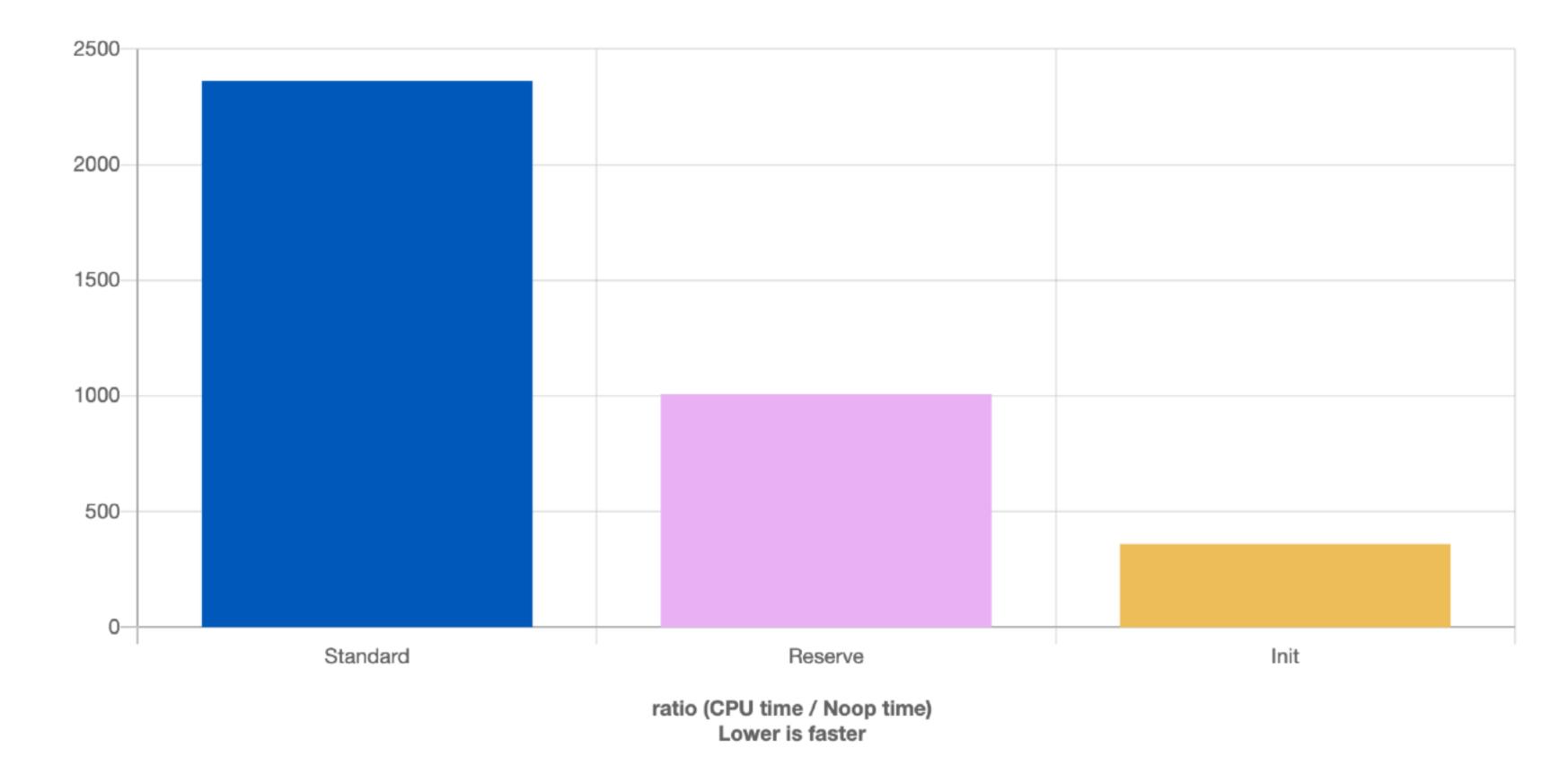
ratio (CPU time / Noop time) Lower is faster

```
2
 3
      static void Standard(benchmark::State& state)
 4
      {
 5
6
7
        for (auto _ : state)
        {
          std::vector<double> vec;
 8
 9
          for (int i = 0; i < 1'000; ++i)
10
            vec.push_back (0.0);
11
        }
      }
12
13
      BENCHMARK(<mark>Standard</mark>);
14
15
      static void Reserve(benchmark::State& state)
      {
16
17
        for (auto _ : state)
        {
18
          std::vector<double> vec;
19
          vec.reserve (1'000);
20
21
22
          for (int i = 0; i < 1'000; ++i)
23
            vec.push_back (0.0);
24
        }
25
      }
26
      BENCHMARK(Reserve);
27
28
      static void Init(benchmark::State& state)
      {
29
30
        for (auto _ : state)
31
        {
32
             std::vector<double> vec (1'000, 0.0);
33
            benchmark::DoNotOptimize (vec);
        }
34
35
36
      BENCHMARK(Init);
```

1

#include <vector>

```
#include <vector>
         std::vector<double> vec;
         for (int i = 0; i < 1'000; ++i)
           vec.push_back (0.0);
12
15
16
17
18
         std::vector<double> vec;
19
         vec.reserve (1'000);
20
21
         for (int i = 0; i < 1'000; ++i)
22
           vec.push_back (0.0);
23
24
25
26
27
28
30
31
           std::vector<double> vec (1'000, 0.0);
32
33
34
```



## Techniques for Optimisation

2. Identifying work that can be combined