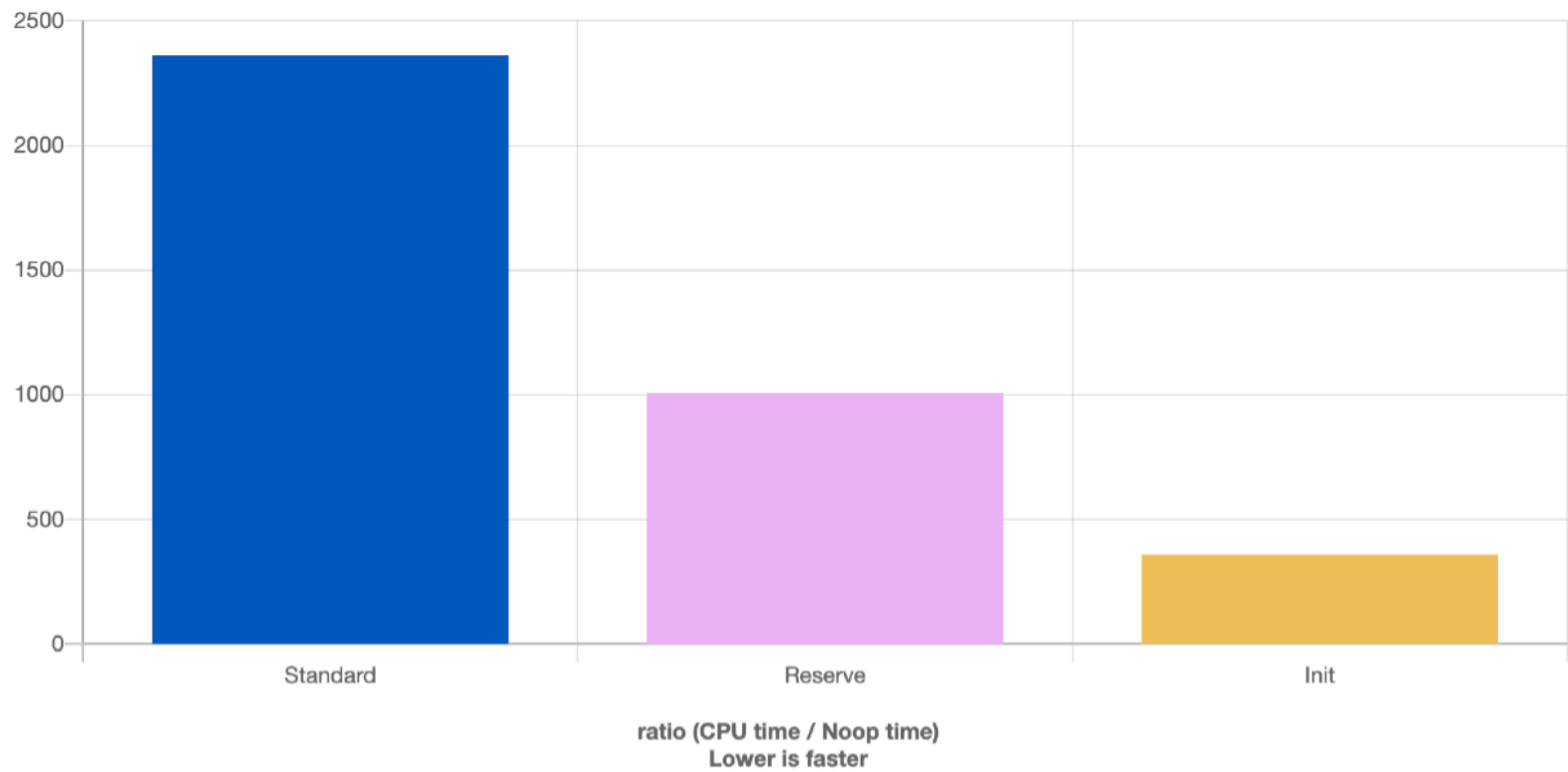


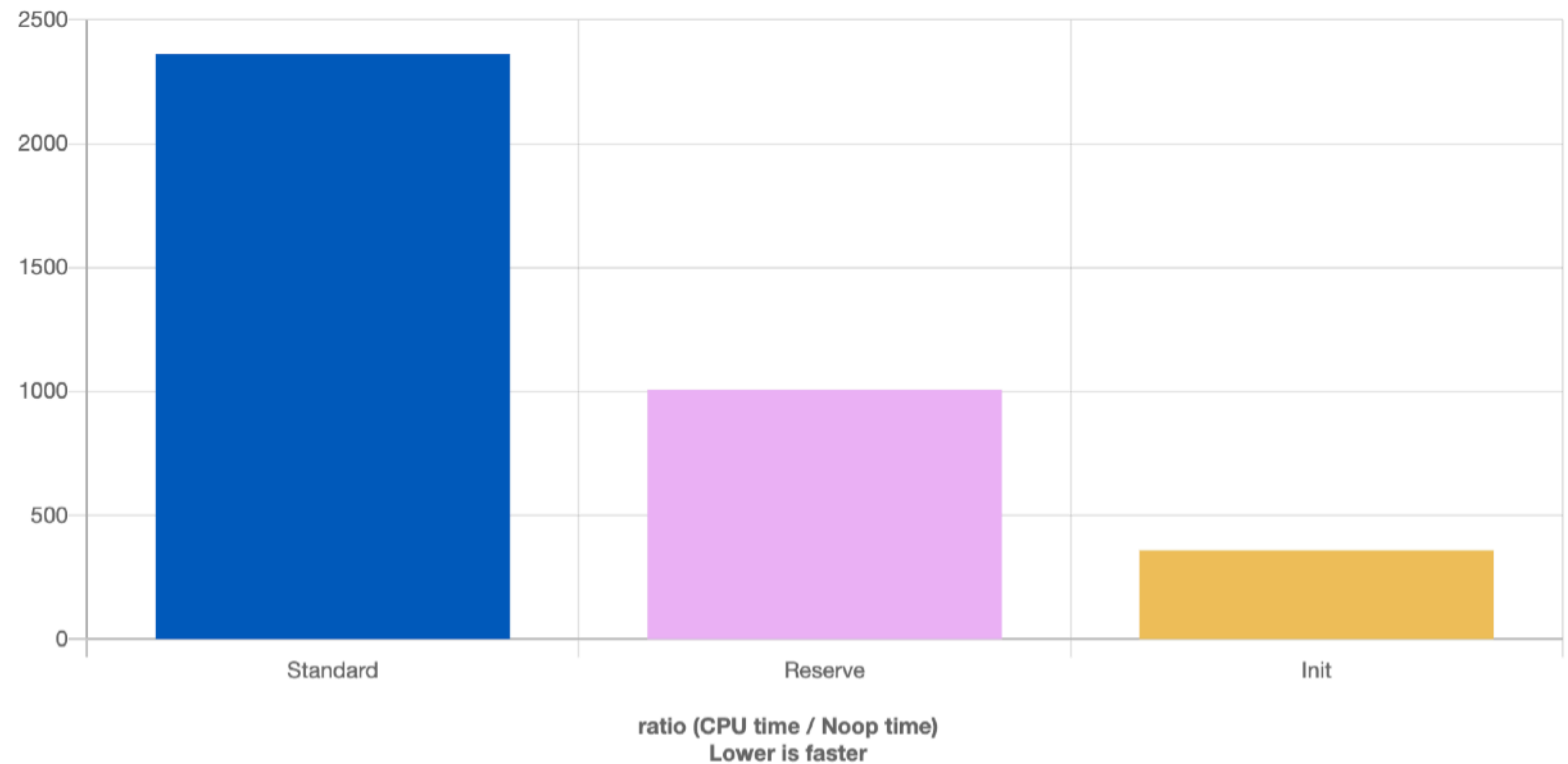
<https://quick-bench.com/q/9uTY-nY6yFS1gIE7hIDBFjU5uNw>



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



Techniques for Optimisation

2. Identifying work that can be combined