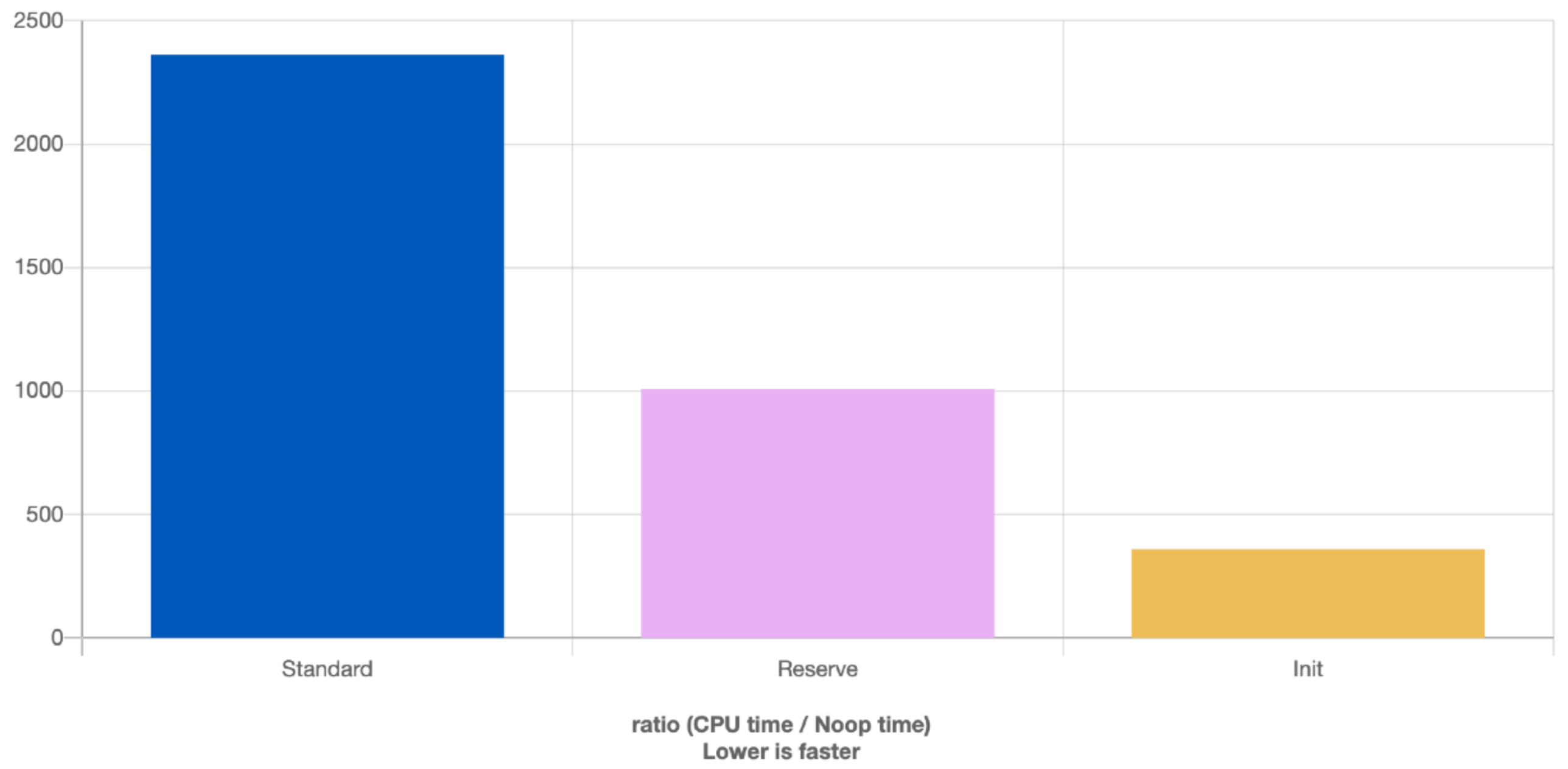
https://quick-bench.com/q/9uTY-mY6yFS1gIE7hIDBFjU5uNw


2500
2000
1500
1000
500
0
Standard
Reserve
Init
ratio (CPU time / Noop time)
Lower is faster

```cpp
#include <vector>

static void Standard(benchmark::State& state)
{
  for (auto _ : state)
  {
    std::vector<double> vec;

    for (int i = 0; i < 1'000; ++i)
      vec.push_back (0.0);
  }
}
BENCHMARK(Standard);

static void Reserve(benchmark::State& state)
{
  for (auto _ : state)
  {
    std::vector<double> vec;
    vec.reserve (1'000);

    for (int i = 0; i < 1'000; ++i)
      vec.push_back (0.0);
  }
}
BENCHMARK(Reserve);

static void Init(benchmark::State& state)
{
  for (auto _ : state)
  {
    std::vector<double> vec (1'000, 0.0);
    benchmark::DoNotOptimize (vec);
  }
}
BENCHMARK(Init);
```

32

```cpp
#include <vector>
```

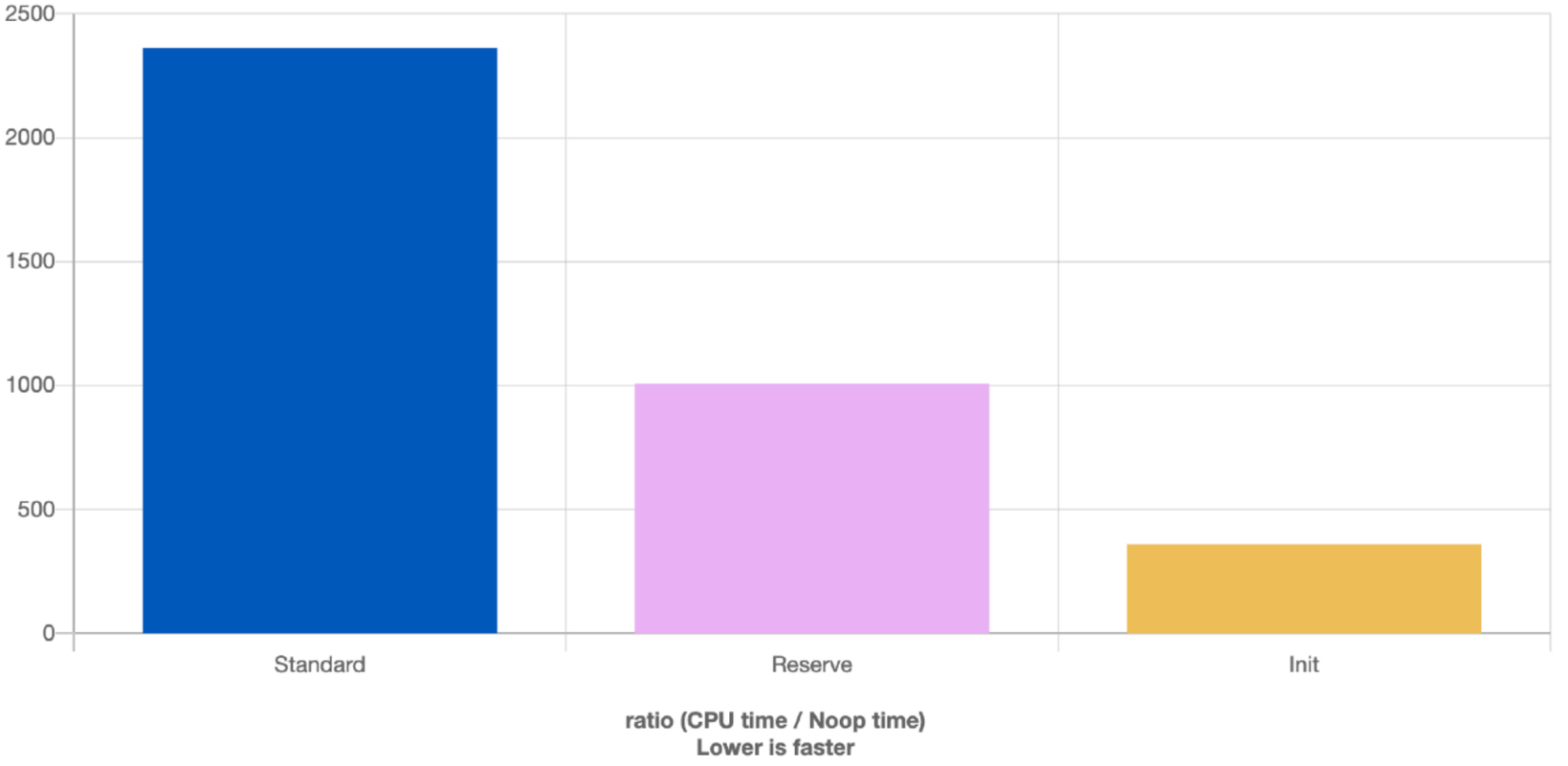
```cpp
std::vector<double> vec;

for (int i = 0; i < 1'000; ++i)
  vec.push_back (0.0);
```

```cpp
std::vector<double> vec;
vec.reserve (1'000);

for (int i = 0; i < 1'000; ++i)
  vec.push_back (0.0);
```

```cpp
std::vector<double> vec (1'000, 0.0);
```



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

Techniques for Optimisation

2. Identifying work that can be combined