Test-Driven Performance

Lenny Maiorani - Sr. Software Engineer

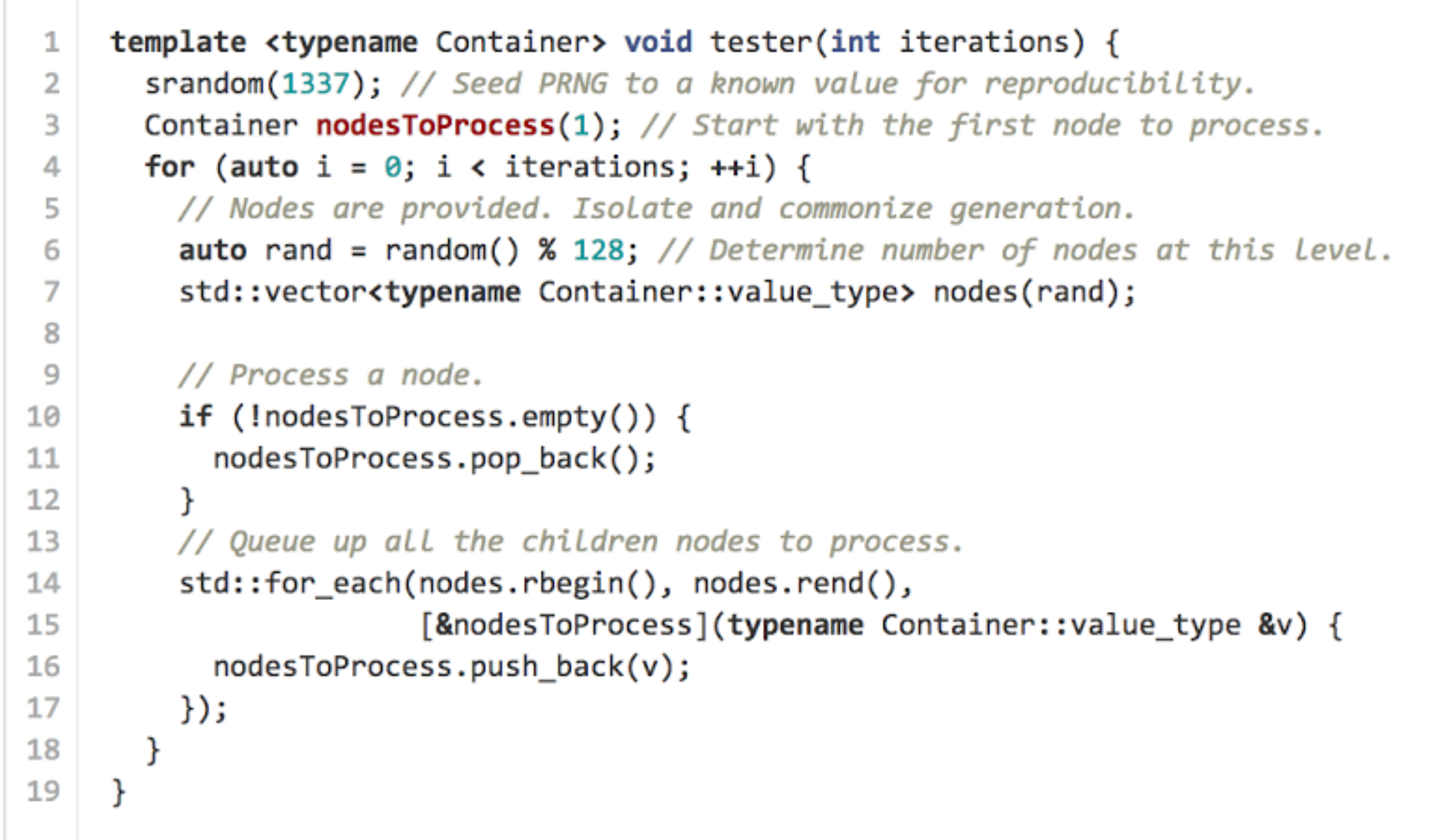
F5 Networks, LineRate Systems team

Boulder, Colorado

How do you choose a container?

- Do you have a favorite?
- Pseudo-random container generator?
- Best guess?
- std::vector because Chandler says so

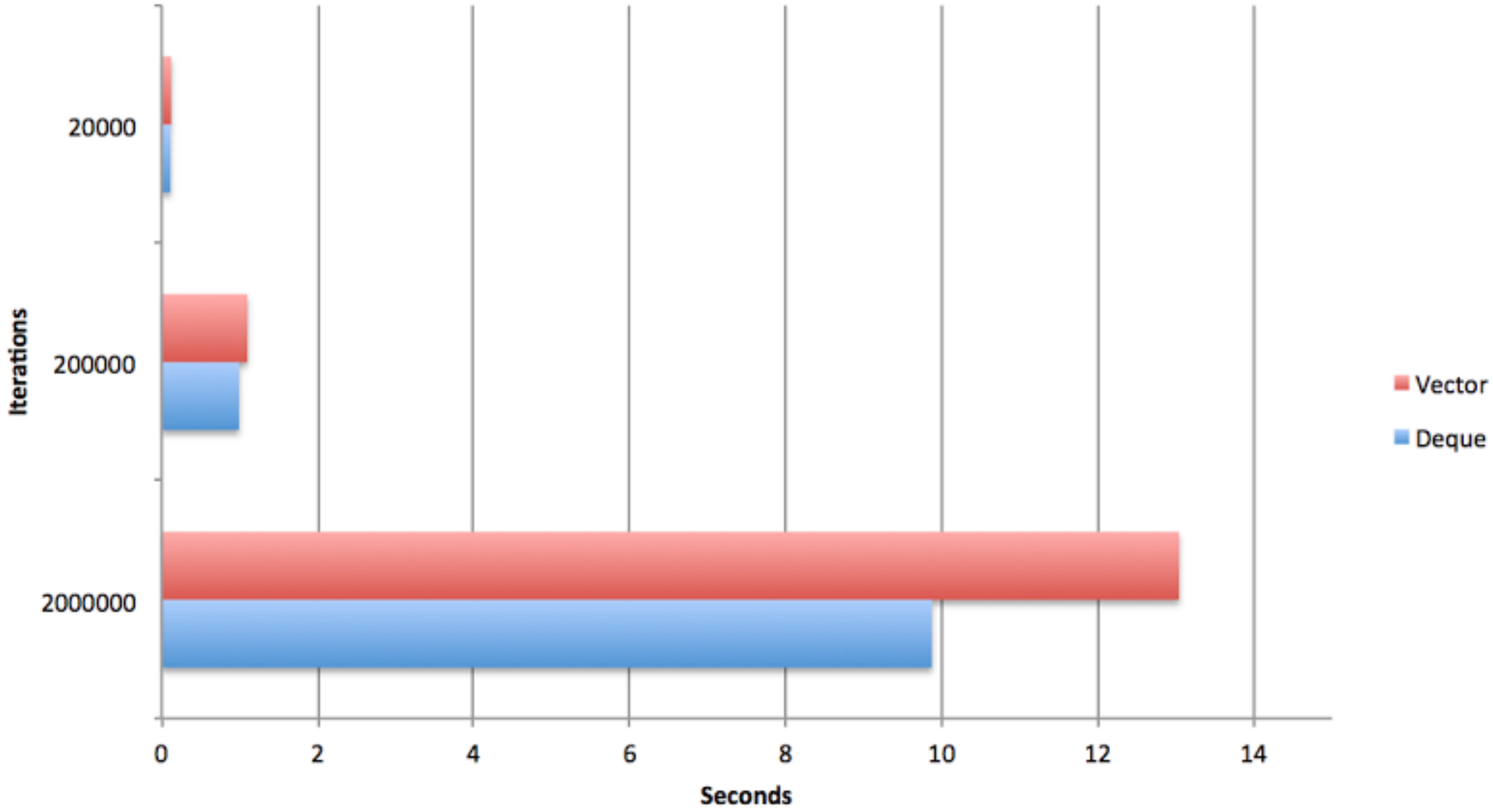
Know your access pattern.



```cpp
template <typename Container> void tester(int iterations) {
  srandom(1337); // Seed PRNG to a known value for reproducibility.
  Container nodesToProcess(1); // Start with the first node to process.
  for (auto i = 0; i < iterations; ++i) {
    // Nodes are provided. Isolate and commonize generation.
    auto rand = random() % 128; // Determine number of nodes at this level.
    std::vector<typename Container::value_type> nodes(rand);

    // Process a node.
    if (!nodesToProcess.empty()) {
      nodesToProcess.pop_back();
    }
    // Queue up all the children nodes to process.
    std::for_each(nodes.rbegin(), nodes.rend(),
                  [&nodesToProcess](typename Container::value_type &v) {
      nodesToProcess.push_back(v);
    });
  }
}
```

Collect results.


Iterations
20000
200000
2000000
Seconds
0
2
4
6
8
10
12
14
Vector
Deque

std::deque wins!

Note: Don't consider deque a "faster" data structure. It is under this special case.

Go forth and test.

https://github.com/ldm5180/cppcon14-tdp