



LOBACHEVSKY
UNIVERSITY

Modern C++

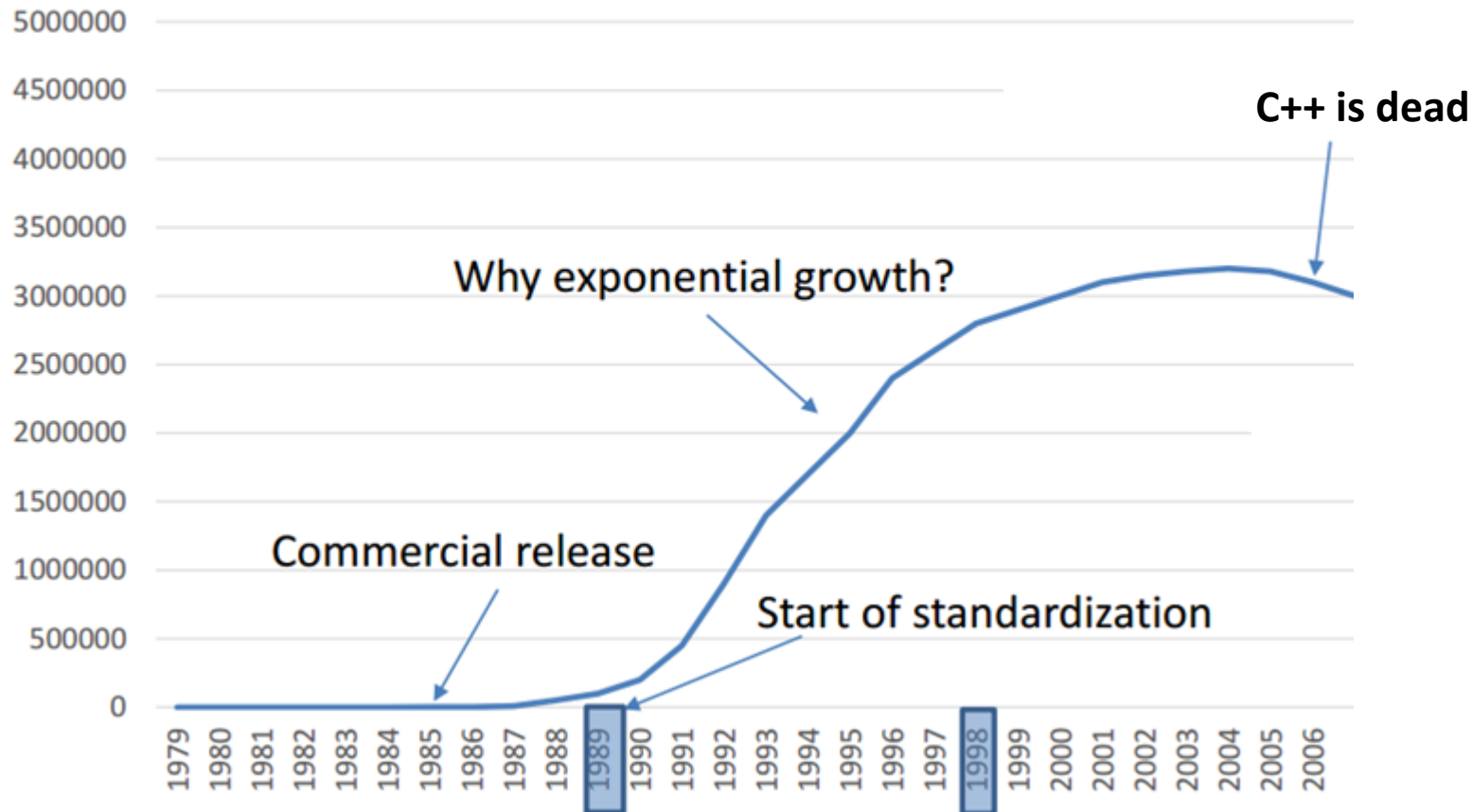
1. Binary search

27.10.2017

Sidnev A.A.

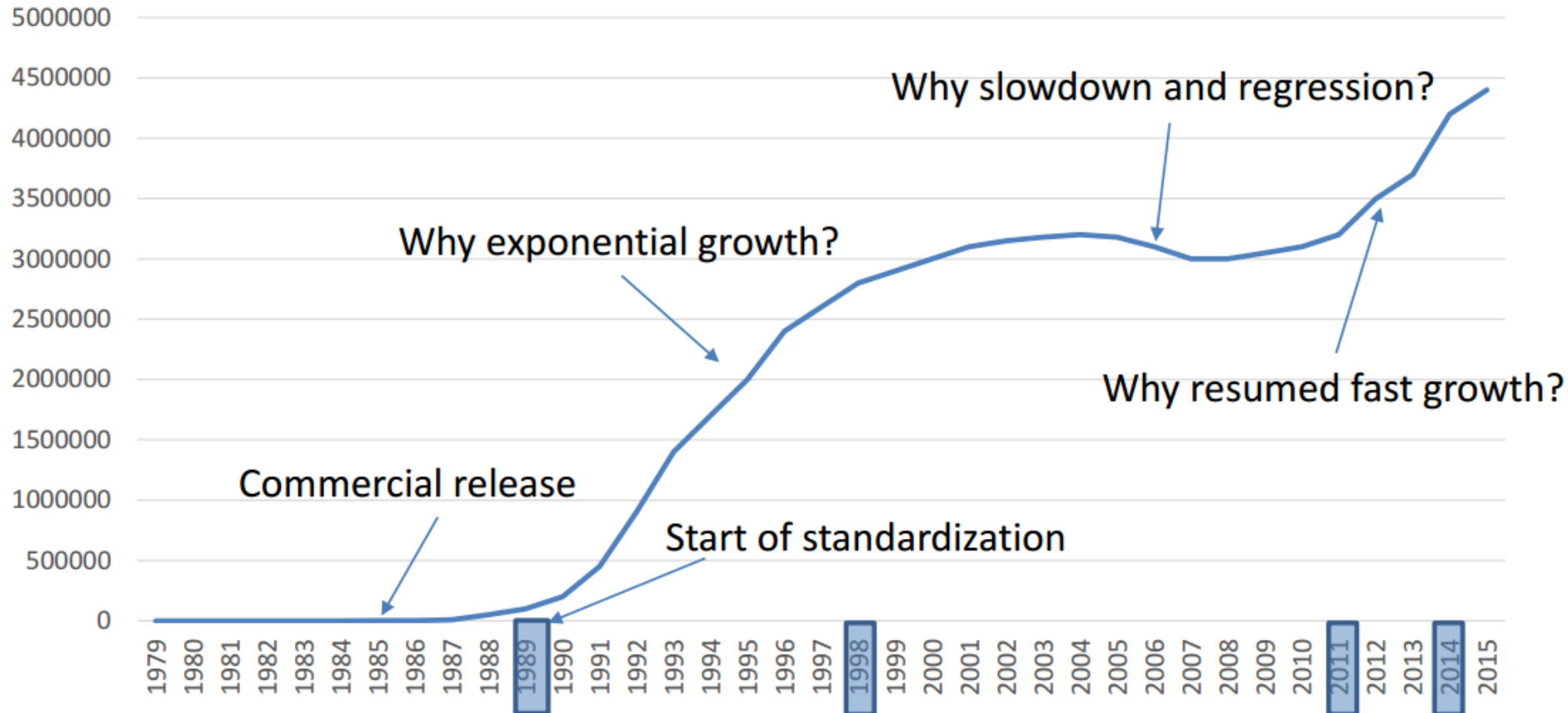
C++ users

#C++ users (approximate, with interpolation)



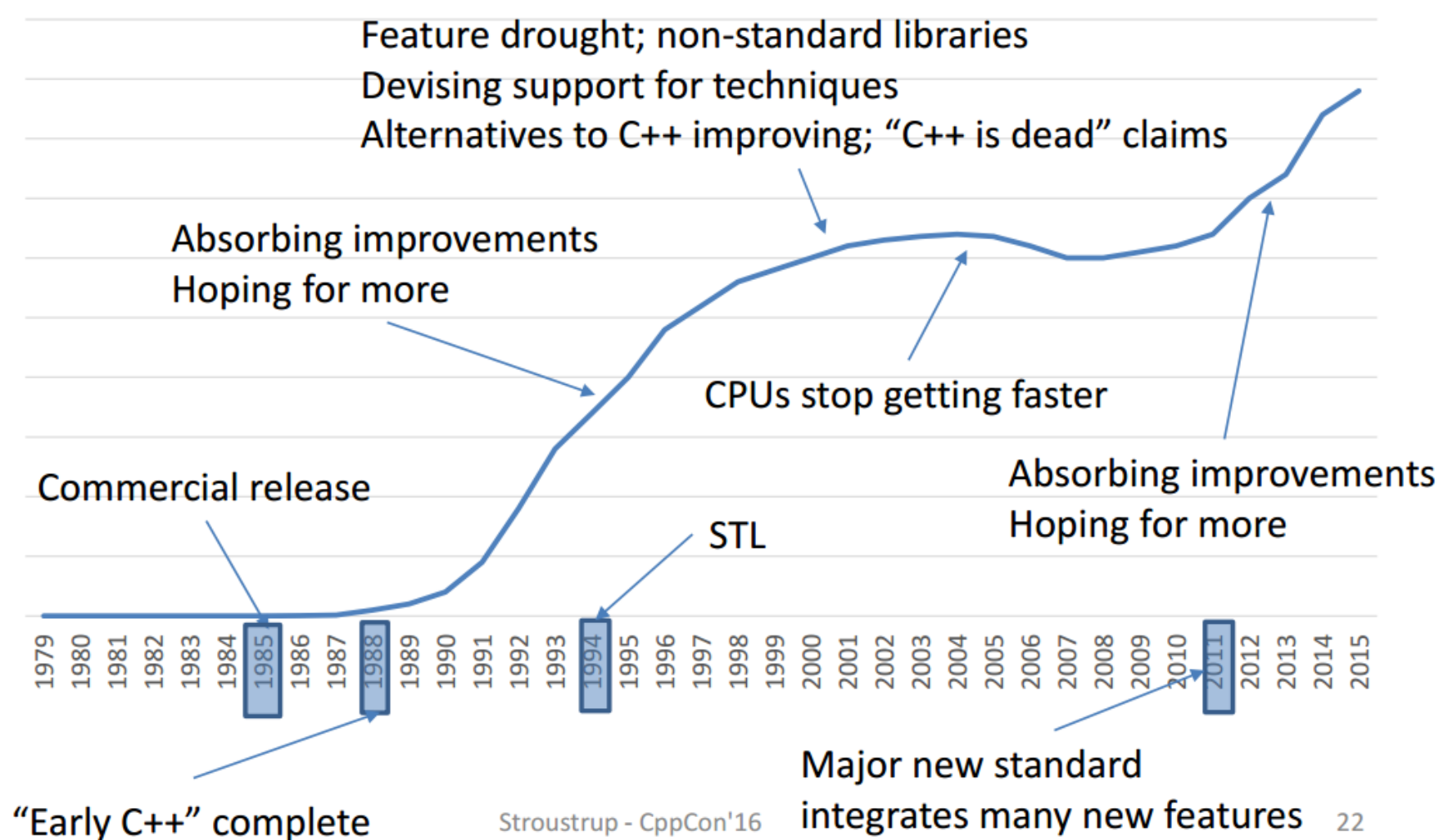
C++ users

#C++ users (approximate, with interpolation)

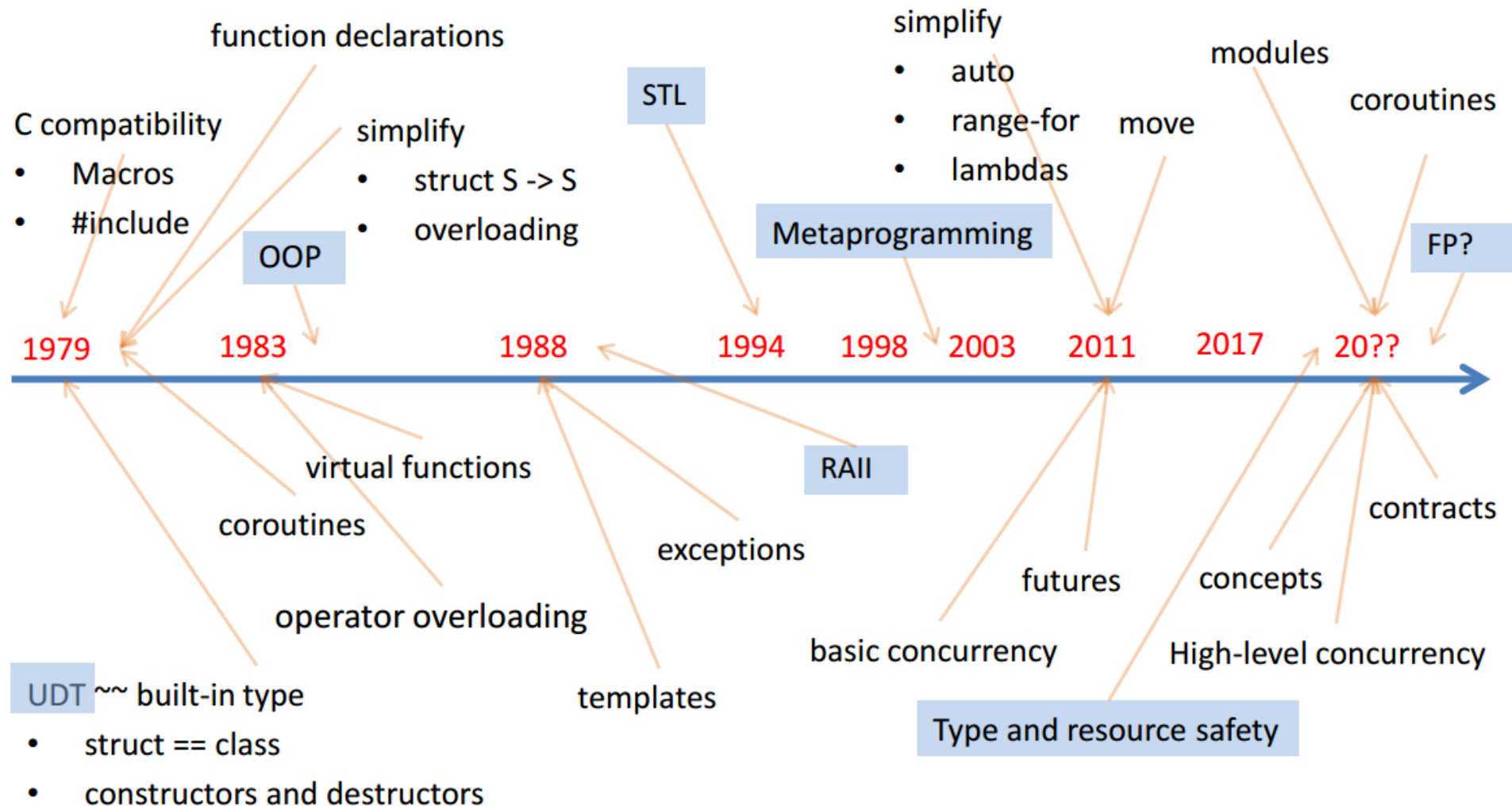


C++ users (cont.)

#C++ users (approximate, with interpolation)



C++ evolution



Compilers

- GCC 5 (6.1, 7)
- Clang 3.4 (3.9)
- MSVC 19 (Visual Studio 2015)

Compiler support

References

- Bjarne Stroustrup. The C++ Programming Language (4th Edition). 2013.
- Scott Meyers. Effective Modern C++, 2014.
 - The book on effective use of the features new in “modern” C++ (i.e., C++11 and C++14). A complement to Scott’s existing books, 42 all-new guidelines address smart pointers, move semantics, lambda expressions, the concurrency API, moving from C++98 to modern C++, and much more.
- Бьёрн Страуструп. Язык программирования C++ (3-е издание).
- CppCon. <https://cppcon.org/>
- Reference. <http://en.cppreference.com>

Binary search

Task: Find position for element in sorted array

Examples:

[1] , 0 -> 0

[1, 2, 3] , 2 -> 1

[1, 2, 4] , 3 -> 2

[1, 2, 4] , 5 -> 3

std::binary_search

```
template<class ForwardIt, class T>  
bool binary_search(ForwardIt first,  
                  ForwardIt last,  
                  const T& value);
```

```
template<class ForwardIt, class T, class Compare>  
bool binary_search(ForwardIt first,  
                  ForwardIt last,  
                  const T& value,  
                  Compare comp);
```

Binary search

Task: Find position for element in sorted array

```
int find(const vector<int>& v, const int& target) {  
    return lower_bound(v.begin(), v.end(), val) - v.begin();  
}
```

lower/upper bound

```
std::vector<int> data =  
    { 1, 1, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 6 };  
// 0  1  2  3  4  5  6  7  8  9 10 11 12  
// ???  
auto lower = std::lower_bound(data.begin(),  
                               data.end(), 4);  
  
// ???  
auto upper = std::upper_bound(data.begin(),  
                              data.end(), 4);
```

lower/upper bound

```
std::vector<int> data =  
    { 1, 1, 2, 3, 3, 3, 3, 4, 4, 4, 5, 5, 6 };  
// 0  1  2  3  4  5  6  7  8  9 10 11 12  
//                                     ^  
auto lower = std::lower_bound(data.begin(),  
                               data.end(), 4);  
  
// 0  1  2  3  4  5  6  7  8  9 10 11 12  
//                                     ^  
auto upper = std::upper_bound(data.begin(),  
                              data.end(), 4);
```

Homework: Leetcode

[69. Sqrt\(x\)](#)

[441. Arranging Coins](#)

Sqrt(x)

Submission Details

Runtime: 6 ms

Status: Accepted

Submitted: 9 months ago

Accepted Solutions Runtime Distribution

