读书笔记·《STL 源码剖析》

目录

Ⅰ. 概述

之前在查阅 deque 的相关资料时，发现 deque 居然支持随机访问，顿生看一遍 STL 源代码的念头。刚好，又看到这本书的读书笔记，其中写到：“我花了 6 个小时的时间看完了这本书……”，于是便决定买一本以看一看长期以来作为黑盒程序使用的 STL 到底是怎么实现的。

Ⅱ. 容器相关

1. deque

① deque 虽然不使用连续的空间，但因为其使用的是中控器+大小相同的缓存区的存储模式，所以其仍然可以做

到常数级别的随机访问。但访问速度劣于 vector

② 控制器的大小默认是 8，以后使用同 vector 类似的倍增机制进行扩展。其缓存区的大小可以指定，一般的默认大小是？[1]。

2. 关联式容器

① set/map 那一大家子的底层都是 RB-Tree，但和 SGI-STL 的 Heap 一样，不提供公共接口

② 基于 Hash_Table 的一大家子，在 C98 中没有得到支持，但从 C11 开始就可以使用了

Ⅲ. 杂项

1. nth_element()

① 其底层调用了 partition()，而 partition()的时间复杂度已是 O(n),所以 nth_element()的复杂度下界已是 O(n)

② 仔细分析一下源代码，可以发现：平均情况下，其复杂度为 $O(n\cdot\sum_{i=1}^{\lfloor\log_2 n\rfloor}\frac{i}{2^{i-1}})$。其准确值我暂不能推出，但强行模拟可以得到：其趋近于 $O(4n)$。由此看来 nth_element()的复杂度在平均情况下，可以视为是 O(n)的

[1] 这一点不是很确定。网络上没有查找到相关资料，本书也没有提及。

2. reverse_iterator

其底层由 iterator 实现，所以速度理论上会略慢于 iterator

3. merge()

① 这货实际上就是归并排序

② STL 还提供很多我们不常用的实用函数，比如 unique() 等

Ⅳ. 小结

通过这本书，确实对于 STL 有了更深入的了解。虽然学到的实用的知识不多，但考虑到阅读此书所花费的时间并不长[2]，如有空闲还是可以阅读一下此书的。毕竟一直使用一个黑盒程序，心里也不放心啊！

[2] 我大概花了 5 个小时的样子