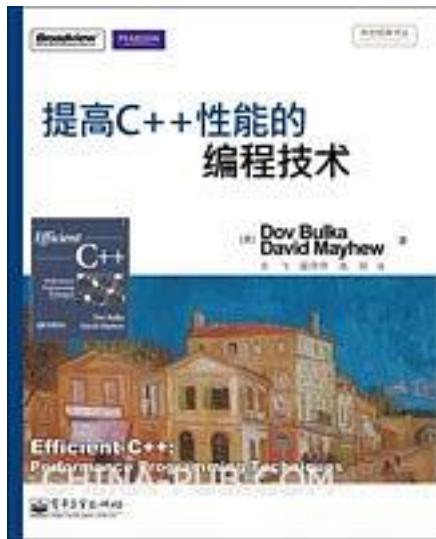


# 提高C++性能的编程技术



[提高C++性能的编程技术 下载链接1](#)

著者:(美)Dov Bulka (多夫.布尔卡) David Mayhew (大卫.梅休)

出版者:电子工业出版社

出版时间:2011-3-28

装帧:

isbn:9787121129377

很多程序员及软件设计师都认为,用c++开发意味着放弃程序性能提升的可能。在很多人眼里,使用c++来开发那些效率至上的应用无疑将导致一场空前的浩劫。因此,在许多性能敏感型领域,诸如网络协议、操作系统内核、移动设备驱动等等,c++都常常处于被冷落的境地。

而本书正是对这种错误观念的最有力回击。《提高c++性能的编程技术》揭示了c++开发高效应用的潜力,向广大读者展示了大量实用的c++面向对象编程技术。通过改善普遍藏匿于设计编码过程暗处的缺陷,这些技术无一不为c++的性能提升带来最为强劲的动力。

《提高c++性能的编程技术》详细讨论了临时对象、内存管理、继承、虚函数、内联、引用计数以及stl等一切有可能提升c++效率的细节内容。最终,该书将c++性能提升的各种终极利器,完美地呈现在广大读者的面前!无论你是相关领域的从业人员,还是c++程序设计爱好者,或者是渴望突破编程瓶颈、大幅提升自我修为的程序设计爱好者,

本书都必将使你获益良多。

作者介绍:

Dov

Bulka在软件开发以及向市场交付大型软件产品方面拥有超过15年的实战经验。他曾是IBM DominoGo

Web服务器的性能设计师，一些曾出现在Internet上的最大型网站使用了这种服务器，其中包括1996年亚特兰大奥运会的网站。Dov Bulka在杜克大学获得了计算机科学博士学位。

David Mayhew是StarBridge

Technologies, Inc.的首席设计师。他主要从事互连构造、对等处理和PCI总线研发等方面的工作，他曾就职于IBM的网络软件部。David Mayhew在弗吉尼亚理工大学获得了计算机科学博士学位。

目录: 导读 1

第1章 跟踪实例 10

1.1 初步跟踪的实现 12

1.2 要点 18

第2章 构造函数和析构函数 20

2.1 继承 20

2.2 复合 32

2.3 缓式构造 34

2.4 冗余构造 37

2.5 要点 41

第3章 虚函数 43

3.1 虚函数的构造 43

3.2 模板和继承 46

3.3 要点 51

第4章 返回值优化 52

4.1 按值返回机制 52

4.2 返回值优化 54

4.3 计算性构造函数 57

4.4 要点 58

第5章 临时对象 59

5.1 对象定义 59

5.2 类型不匹配 60

5.3 按值传递 63

5.4 按值返回 64

5.6 使用op=()消除临时对象 66

5.7 要点 67

第6章 单线程内存池 69

6.1 版本0: 全局函数new()和delete() 70

6.2 版本1: 专用rational内存管理器 71

6.3 版本2: 固定大小对象的内存池 76

6.4 版本3: 单线程可变大小内存管理器 80

6.5 要点 87

第7章 多线程内存池 88

7.1 版本4: 实现 88

7.2 版本5: 快速锁定 91

7.3 要点 95

第8章 内联基础 96

8.1 什么是内联? 96

8.2 方法调用的代价 100

8.3 因何内联? 105

8.4 内联详述 105

8.5 虚方法的内联 107

8.6 通过内联提升性能 108

8.7 要点 109

第9章 内联——站在性能的角度 110

9.1 调用间优化 110

9.2 何时避免内联? 115

9.3 开发阶段及编译期的内联考虑 118

9.4 基于配置的内联 119

9.5 内联规则 123

9.6 要点 125

第10章 内联技巧 126

10.1 条件内联 126

10.2 选择性内联 127

10.3 递归内联 129

10.4 对静态局部变量进行内联 134

10.5 与体系结构有关的注意事项: 多寄存器集 136

10.6 要点 137

第11章 标准模板库 138

11.1 渐近复杂度 138

11.2 插入 139

11.3 删除 146

11.4 遍历 149

11.5 查找 150

11.6 函数对象 152

11.7 比stl更好? 154

11.8 要点 157

第12章 引用计数 158

12.1 实现细节 160

12.2 已存在的类 172

12.3 并发引用计数 175

12.4 要点 179

第13章 编码优化 180

13.1 缓存 182

13.2 预先计算 183

13.3 降低灵活性 184

13.4 80-20法则: 加快常用路径的速度 185

13.5 延迟计算 189

13.6 无用计算 191

13.7 系统体系结构 192

13.8 内存管理 193

13.9 库和系统调用 194

13.10 编译器优化 197

13.11 要点 198

第14章 设计优化 200

14.1 设计灵活性 200

14.2 缓存 204

14.3 高效的数据结构 208

14.4 延迟计算 208

14.5 getpeername() 209

14.6 无用计算 212  
14.7 失效代码 213  
14.8 要点 214  
第15章 可扩展性 215  
15.1 对称多处理器架构 217  
15.2 amdahl定律 218  
15.3 多线程和同步 220  
15.4 将任务分解为多个子任务 221  
15.5 缓存共享数据 222  
15.6 无共享 224  
15.7 部分共享 226  
15.8 锁粒度 228  
15.9 伪共享 230  
15.10 惊群现象 231  
15.11 读/写锁 233  
15.12 要点 234  
第16章 系统体系结构相关话题 235  
16.1 存储器层级 235  
16.2 寄存器：存储器之王 237  
16.3 磁盘和内存结构 241  
16.4 缓存效应 244  
16.5 缓存抖动 246  
16.6 避免跳转 247  
16.7 使用简单计算代替小分支 248  
16.8 线程化的影响 249  
16.9 上下文切换 251  
16.10 内核交叉 254  
16.11 线程化选择 255  
16.12 要点 257  
参考文献 258  
索引 260  
· · · · · (收起)

[提高C++性能的编程技术](#) [下载链接1](#)

标签

C++

性能优化

C/C++

编程

程序设计

计算机

Programming

计算机科学

## 评论

C++的性能优化书，不过内容有点老了

---

见鬼 英文版是1999年的了 很老了

---

料比较少

---

这本关于 C++ 程序优化的专著 Efficient C++ (1999.11)，出版时间较早，适用于 C++98。虽然本书的主要优化原则仍具有指导意义，但某些平台环境和操作细节已过时。可将本书作为学习其它最新 C++ 程序优化资料时的交叉参考。

---

书还不错，就是有些小贵，而且纸张太搓了。看了这本书想起之前看深入理解计算机系统的时候，发现现在写代码的一些优化手段都在集中在设计优化，对高速缓存之类的考虑太少了

---

这本书放在手边，闲暇的时候随便找一章翻翻，受益无穷，做Web的人，后端的性能至上。

通俗易懂，来源实践，篇幅也不长，好书。

-----  
关于容器部分的性能提升很有感触。

-----  
太浅显了

-----  
罗列了一些性能优化的基本方法，比较有价值的是末尾的伪共享、惊群、缓存一致性的描述，更深描述需要去阅读《多处理器编程的艺术》《现代体系结构上的unix系统》《深入理解并行编程》

-----  
后边的逼格太高，降不住啊

-----  
Harry

-----  
前面几章通俗易懂，很多都是写代码起手就要注意的事项，倒数的四章算是带入门，仅就介绍而已，但是对新人有很大参考价值

-----  
才看完1到2章就知道这本书完全过时了，C++  
98什么年代的标准，什么年代的编译器。慢了赶紧换，那时候RVO TCO  
估计实现还不成熟。

-----  
有点过时，但是总结得不错

-----  
浅显易懂....

正如灵活性、可重用性与性能的对立，不能指望有奇技淫巧给C++带来C的性能。了解语言各功能底层大致的实现，开发中的局部优化基本保证。对算法、操作系统、硬件环境以及软件使用场景特点的了解使得性能优化变得更加灵活。为图快看了中文版，翻译质量不佳，有时需英文版对照。两颗星给中文版。

整体不错，有点冗余部分

对程序优化很有教益，浅显易懂

看过《(more) effective c++》和《inside the c++ object model》的话这本书看起来还是挺轻松的，有些重复。内联那几章算是新知识，受益匪浅。

[提高C++性能的编程技术 下载链接1](#)

## 书评

第一感觉：这本书是好书，但是并不适合初级C++程序员阅读  
本书坚持着这样的一个观点：C++并非就无法达到C那样的高性能，只要经过一定的技巧优化，C++也能够在对程序性能要求苛刻的如WEB传输等应用中表现出等同甚至超越C语言的性能。不过书中并没有一味的强调性能至上，如在STL...

如果你比较关心代码的性能，这无疑是一本好书。  
书中讲了一些代码优化的细节，包括构造与析构的成本、内联、多线程等。比较实用。  
翻译有点烂，很多都是照字面意思翻译，太不专业了。所以推荐看英文版的。英文版看起来比许多其他的英文技术书籍要轻松些。

这本书提供了C++性能优化的一些可以实践的技巧。特别是前面几章内存，比如构造函数，返回值优化，虚函数，临时对象，内存池还有内联。最好的实践的方法是先把书读一遍，再把例子写一遍，最后把自己的写过的代码进行一次优化，看下性能是否有所提高。

---

书是好书，翻译的一般。有些东西可以马上用到实践中，有些东西却是更偏理论些。如果能多增加一些相应的实践中的实例，那效果就更好了。

书中有很多都关注了编程中的细节，这点相当不错。性能不仅要从大处着眼，对于细节也同样不能忽视。

---

在图书馆看到这本书，看了下目录感觉还可以，但翻了一个小时候看完了前面第二章，发现了一些问题：

首先就是翻译了，这个不多说了，反正就那样，大家都懂的，有的句子不知道要讲什么意思。然后发觉了书中两个明显的错误，这个不知道是原作者的错误（估计不...

---

C++作为一种面向对象的语言，其工作效率在诸多面向对象语言之中是屈指可数的，但是，如果您觉着编写的C++程序并没有感受到高效带来的执行性能，而且在多次检查重构代码之后依然感到疑惑，那么，请您阅读这本书，它从C++底层开始，为C++的各种特性对程序性能的影响做出了深入浅...

---

我看过了 我看...

---

[提高C++性能的编程技术\\_下载链接1](#)