Broadview®
www.broadview.com.cn

IBM China Development Laboratories Series
IBM中国开发中心作品系列

C++应用程序性能优化

冯宏华 徐莹 程远 汪磊 等编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
HTTP://WWW.PHEI.COM.CN

IBM China Development Laboratories Series

IBM中国开发中心作品系列

C++应用程序性能优化

冯宏华 徐莹 程远 汪磊 左力 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书主要针对的是C++程序的性能优化，深入介绍C++程序性能优化的方法和实例。

全书由4个篇组成，第1篇介绍C++语言的对象模型，该篇是优化C++程序的基础；第2篇主要针对如何优化C++程序的内存使用；第3篇介绍如何优化程序的启动性能；第4篇介绍了三类性能优化工具，即内存分析工具、性能分析工具和I/O检测工具，它们是测量程序性能的利器。

本书适用于有一定C++程序开发经验的开发人员，也可以作为高校相关专业师生的参考书。

图书在版编目（CIP）数据

C++应用程序性能优化 / 冯宏华等编著. —北京：电子工业出版社，2007.3
（IBM中国开发中心作品系列）
ISBN 978-7-121-03831-0

I. C… II. 冯… III. C语言—程序设计 IV. TP312

中国版本图书馆CIP数据核字（2007）第012912号

责任编辑：孙学瑛
印　　刷：北京智力达印刷有限公司
装　　订：北京中新伟业印刷有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　邮编100036
开　　本：787×980　1/16　印张：22　字数：404千字
印　　次：2007年3月第1次印刷
印　　数：5000册　定价：49.80元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系电话：（010）68279077；邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：（010）88258888。

序

致力于软件业务在中国的长期发展，IBM 公司于 1999 年在中国投资成立了中国开发中心（IBM China Development Lab，CDL）。在为 IBM 全球客户提供满足需求的软件、硬件产品，以及技术和解决方案的同时，作为 IBM 全球软件资源在中国的窗口，CDL 把全球先进技术引入中国，为中国软件产业与世界的交流搭建桥梁。

八年来，CDL 不断吸引全球卓越的科技和管理人才，以及中国各大学府的顶尖学生、优秀工程师加入，并积极致力于同本土合作伙伴共同建设团队，研发队伍从八年前的 100 多位软件工程师增至今天的 3000 多位，年平均成长速度超过 50%。

CDL 高度重视员工培养，除各种短期或长期的培训，员工还被派往 IBM 在美国或其他地区的实验室，与世界各地同仁一起工作，在工作中学习先进的技术和管理方式。CDL 更是将 IBM 全球实验室中拥有丰富经验的技术和管理人才请到中国工作，向中国员工传授经验。庞大的资金注入，人性化的管理方式，以及对人才方面的巨大投资，对人力资本的极度重视，使中国开发中心得以与 IBM 全球实验室共同成长，最终拥有一支经验丰富，训练有素的团队。

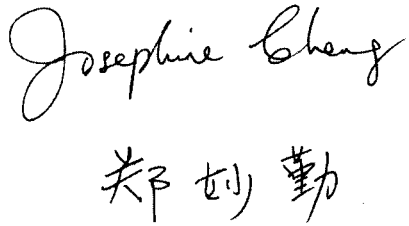
目前，CDL 与全世界同步发展多项领域产品，正在为包括 Information Management、WebSphere、Lotus、Tivoli、Rational 在内的所有 IBM 软件核心产品的研究和开发做出卓越贡献，并在 SOA、数据库、WebSphere 产品系列、普及运算、Lotus Workplace Client 技术及 Linux 系统方面取得非凡的成就，被视为 IBM 全球产品的开发重心之一。

在 CDL 高速发展的同时，为将信息产业的最新技术尽快地转化为对中国用户有价值的解决方案，帮助用户更有成效地开展业务，增强竞争优势，我们恪守为中国软件业与世界交流搭建桥梁的承诺，希望将 IBM 全球公司几十年的技术积淀和我们的心得与大家共同分享，于是，我们选择了实力非凡、专业创新的电子工业出版社博文视点公司作为合作伙伴，推出这一由 IBM 中国开发中心（CDL）的架构师、资深软件工程师们编写的系列丛书，范围涵盖了从开发实践、测试方法、项目实践、最新技术标准和发展趋势探讨，到先进解决方案构建、面向服务的架构的提供等诸多方面。

我希望这套丛书能把我们一线专家宝贵的经验，以及我们的见解呈现给读者，并希望

无论是企业 IT 经理、程序设计和开发人员、软件工程师、软件架构师，还是在校学生，或者是对计算机领域有兴趣的人员，都能从中获取知识或者得到启发。

在同业界分享经验和世界最新技术及趋势的同时，我们希望能为推动中国软件产业的加速发展贡献微薄之力。IBM 中国开发中心将一如既往地同业界同仁一起，共铸中国信息产业的辉煌明天！



IBM 院士、总经理　IBM 中国开发中心

2007 年 2 月

作者介绍

冯宏华，清华大学计算机科学与技术系硕士。IBM 中国开发中心高级软件工程师。2003 年 12 月加入 IBM 中国开发中心，主要从事 IBM 产品的开发、性能优化等工作。兴趣包括 C/C++应用程序性能调优，Windows 应用程序开发，Web 应用程序开发等。

徐莹，山东大学计算机科学与技术系硕士。2003 年 4 月加入 IBM 中国开发中心，现任 IBM 中国开发中心开发经理，一直从事 IBM 软件产品在多个操作系统平台上的开发工作。曾参与 IBM 产品在 Windows 和 Linux 平台上的性能优化工作，对 C/C++编程语言和跨平台的大型软件系统的开发有较丰富的经验。

程远，北京大学计算机科学与技术系硕士。IBM 中国开发中心高级软件工程师。2003 年加入 IBM 中国开发中心，主要从事 IBM Productivity Tools 产品的开发、性能优化等工作。兴趣包括 C/C++编程语言，软件性能工程，Windows/Linux 平台性能测试优化工具等。

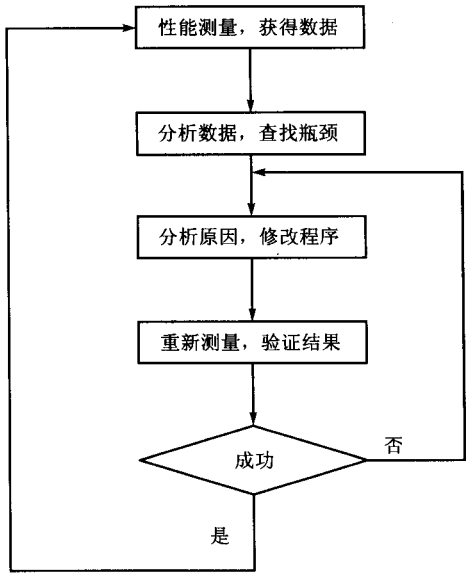
汪磊，北京航空航天大学计算机科学与技术系硕士，目前是 IBM 中国软件开发中心高级软件工程师。从 2002 年 12 月加入 IBM 中国开发中心至今一直从事旨在提高企业生产效率的应用软件开发。兴趣包括 C\C++应用程序的性能调优，Java 应用程序的性能调优。

前　言

在计算机发展的早期阶段，硬件资源相对而言是非常昂贵的。不论是CPU时间，还是内存，都给编程人员设置了很大的限制。因此，当时程序对运行性能和内存空间占用的要求是非常严格的。很多开发人员为了减少1%的CPU运行时间或者减少几十个，甚至几个字节而努力。随着计算机技术的快速发展，硬件资源变得相对便宜。因此有的观点认为在开发软件时，软件的性能优化将不再重要，硬件将解决性能问题。但事实上，这种观点是相对片面的。的确，硬件的发展解决了部分软件的性能问题。但随着硬件计算能力的提高，人们对软件功能的要求也越来越高。当今的软件功能越来越复杂，给用户的界面和操作体验也越来越智能和友好，这些需求带来的软件性能上的要求是硬件不能完全解决的。很多实际的项目证明，如果在开发软件时不重视性能的优化，最终虽然实现了功能上的要求，但软件的运行效率低下，也不能给用户带来很好的效益。因此，软件的性能优化是计算机软件开发过程中需要一直关注的重要因素。

程序性能优化的过程

从开发过程的角度来看，程序的性能优化流程一般如下图所示。



性能优化的第 1 步是测量，尤其是规模较大，并且比较复杂的软件系统，测量性能数据是进行性能优化的基础。有了真实的数据，才可以进行第 2 步，即分析数据，从而找到系统真正的瓶颈所在。毫无疑问，优化应该是针对系统的性能瓶颈进行，而找到性能瓶颈应该是建立在真实性能数据的基础上，而不能是主观臆断。现在有很多工具可以辅助进行性能测量和数据分析，本书也会介绍一些这方面工具的使用方法和实践经验。

进行性能优化的核心在于第 3 步，即分析原因，修改程序，这也将是本书重点介绍的部分。程序的性能包括很多方面，常见的有程序的启动速度，运行速度及运行时占用的内存等。而影响这些性能的因素也很多，大致可以分为如下两类。

- 软件编程设计因素：如算法和数据结构的选择，编程语言的使用等。
- 软件系统结构因素：如动态库/静态库的组织、外部数据的存储及网络环境等。

软件编程设计因素可以看做是程序的内在本质，一般来说，也是对软件性能影响较大的因素。只有对编程语言、算法和数据结构有深入的了解，才能分析出原因，并且找出解决性能问题的方法。本书将针对 C++语言，深入介绍 C++程序性能优化的方法和实例。

软件系统的结构因素可以看做是程序的外在形式，它们一般和操作系统紧密相关。尤其是现在的软件系统，由于功能复杂，大都采用组件形式，以最大限度地提高可复用性。因此，一般会包含一些动态库/静态库，这些库的组织也会影响到软件系统的性能。本书将针对 Windows 和 Linux 介绍动态库和静态库的基本知识及其对软件系统性能的影响。需要指出的是，上面这个优化的过程需要在软件开发的整个过程中不断地迭代进行。而且开始得越早，出现的性能问题越容易解决。

本书的结构

本书主要针对 C++程序的性能优化，由 4 篇组成。第 1 篇介绍 C++语言的对象模型，与性能有关的语言特性及一些数据结构的性能，本篇是 C++程序优化的基础。

第 2 篇主要介绍 C++程序内存使用的优化。内存无疑是影响程序性能的重要因素，很多程序由于没有采用正确的方法分配和使用内存，不仅占用内存较多，而且运行效率不高。在本篇中将结合一些操作系统的内存管理机制介绍如何在特定的平台下进行内存优化。此外，还将深入介绍 C++语言管理动态内存的机制和方法，并介绍一个常用的内存管理方法，

即内存池的实现。

第 3 篇介绍程序启动性能的优化。程序的启动性能不仅受到软件编程设计因素的影响，也会受到系统结构因素的影响，尤其是动态库的影响。本篇将首先介绍动态库的基本知识，然后介绍一些程序启动性能优化的具体方法。

“工欲善其事，必先利其器”，好的工具会大大提高程序性能优化的效率。第 4 篇将介绍 3 类性能工具，即内存分析工具、性能分析工具和 I/O 检测工具，它们是性能测量和分析的利器。

本书适用于有一定 C++开发经验的开发人员，也可以作为高等院校相关专业师生的参考书。

致　谢

本书是集体创作的结晶，在此感谢大家出色的协作精神。本书的写作也占用了大家大量的休息、娱乐，以及和家人在一起的时间，所以在此感谢作者们家人的理解和支持。同时，在成书的过程中与许多人的关怀、鼓励和支持密不可分，其中包括CDL总裁Josephine，律师Andrew，Director Dennis，资深经理Debbi 和Cindy，研发经理阎小兵和扈晓炜等，在此表示感谢。最后，特别感谢出版社的有关领导、协调人员及编辑，没有他们的支持和参与，本书的出版是不可能的。

由于时间仓促、水平有限，书中难免有许多不妥、甚至错误之处。在此敬请读者不吝指出，我们将愿意与读者共同探讨，并不胜感激。

目　录

第 1 篇　C++程序优化基础

第 1 章　C++对象模型 …… 3

1.1　基本概念 …… 4

1.1.1　程序使用内存区 …… 4

1.1.2　全局/静态存储区及常量数据区 …… 7

1.1.3　堆和栈 …… 9

1.1.4　C++中的对象 …… 10

1.2　对象的生命周期 …… 11

1.3　C++对象的内存布局 …… 16

1.3.1　简单对象 …… 17

1.3.2　单继承 …… 20

1.3.3　多继承 …… 23

1.4　构造与析构 …… 33

1.5　本章小结 …… 35

第 2 章　C++语言特性的性能分析 …… 37

2.1　构造函数与析构函数 …… 39

2.2　继承与虚拟函数 …… 51

2.3　临时对象 …… 61

2.4　内联函数 …… 77

2.5　本章小结 …… 86

第 3 章　常用数据结构的性能分析 …… 87

3.1　常用数据结构性能分析 …… 88

3.1.1　遍历 …… 93

3.1.2　插入 …… 95

3.1.3 删除 …… 98
3.1.4 排序 …… 101
3.1.5 查找 …… 105
3.2 动态数组的实现及分析 …… 107
3.2.1 动态数组简介 …… 107
3.2.2 动态数组实践及分析 …… 109
3.3 本章小结 …… 116

第 2 篇 内存使用优化

第 4 章 操作系统的内存管理 …… 119
4.1 Windows 内存管理 …… 120
4.1.1 使用虚拟内存 …… 121
4.1.2 访问虚拟内存时的处理流程 …… 123
4.1.3 虚拟地址到物理地址的映射 …… 126
4.1.4 虚拟内存空间使用状态记录 …… 128
4.1.5 进程工作集 …… 130
4.1.6 Win32 内存相关 API …… 132
4.2 Linux 内存管理机制 …… 142
4.2.1 进程的内存布局 …… 143
4.2.2 物理内存管理 …… 145
4.2.3 虚拟内存管理 …… 146
4.2.4 虚拟地址映射为物理地址 …… 147
4.3 本章小结 …… 148

第 5 章 动态内存管理 …… 149
5.1 operator new/delete …… 150
5.2 自定义全局 operator new/delete …… 155
5.3 自定义类 operator new/delete …… 160
5.4 避免内存泄漏 …… 163
5.5 智能指针 …… 169

5.6 本章小结 …… 181

第 6 章 内存池 …… 183

6.1 自定义内存池性能优化的原理 …… 184
6.1.1 默认内存管理函数的不足 …… 184
6.1.2 内存池的定义和分类 …… 184
6.1.3 内存池工作原理示例 …… 185
6.2 一个内存池的实现实例 …… 186
6.2.1 内部构造 …… 187
6.2.2 总体机制 …… 188
6.2.3 细节剖析 …… 191
6.2.4 使用方法 …… 202
6.2.5 性能比较 …… 202
6.3 本章小结 …… 203

第 3 篇 应用程序启动性能优化

第 7 章 动态链接与动态库 …… 207

7.1 链接技术的发展 …… 208
7.1.1 编译、链接和加载 …… 208
7.1.2 静态链接与静态链接库 …… 211
7.1.3 动态链接与动态库 …… 218
7.2 Windows DLL，Dynamic Linked Library …… 219
7.2.1 DLL 基础 …… 219
7.2.2 DLL 如何工作 …… 224
7.2.3 关于 DLL 的杂项 …… 232
7.3 Linux DSO …… 233
7.3.1 DSO 与 ELF …… 234
7.3.2 DSO 如何工作 …… 241
7.3.3 构建与使用 DSO …… 248
7.4 本章小结 …… 260

第 8 章　程序启动过程 ····· 261
8.1　Win32 程序启动过程 ····· 262
8.2　Linux 程序启动过程 ····· 266
8.3　影响程序启动性能的因素 ····· 267
8.3.1　源代码因素 ····· 268
8.3.2　动态链接库因素 ····· 269
8.3.3　配置文件/资源文件因素 ····· 276
8.3.4　其他因素 ····· 277
8.4　本章小结 ····· 279
第 9 章　程序启动性能优化 ····· 281
9.1　优化程序启动性能的步骤 ····· 282
9.2　测试程序启动性能的方法 ····· 283
9.3　优化可执行文件和库文件 ····· 286
9.3.1　减少动态链接库的数量 ····· 286
9.3.2　减小动态链接库尺寸 ····· 288
9.3.3　优化可执行文件和库文件中的代码布局 ····· 288
9.4　优化源代码 ····· 290
9.4.1　优化启动时读取的配置文件及帮助文件 ····· 291
9.4.2　预读频繁访问的文件 ····· 291
9.4.3　清除产生 exception 的代码 ····· 293
9.4.4　PreLoad ····· 294
9.4.5　延迟初始化 ····· 294
9.4.6　多线程化启动 ····· 295
9.5　本章小结 ····· 295

第 4 篇　性能工具

第 10 章　内存分析工具 IBM Rational Purify ····· 299
10.1　Rational Purify 工作原理 ····· 300
10.2　Rational Purify 使用指南 ····· 303

10.3 Rational Purify 实例分析 ····· 308
10.4 本章小结 ····· 312

第 11 章 性能分析工具 IBM Rational Quantify ····· 313

11.1 Rational Quantify 工作原理 ····· 314
11.2 Rational Quantify 使用指南 ····· 316
11.3 Rational Quantify 实例分析 ····· 319
11.4 本章小结 ····· 324

第 12 章 实时 IO 监测工具 FileMon ····· 325

12.1 FileMon 的工作原理 ····· 326
12.2 FileMon 使用指南 ····· 328
12.3 使用 FileMon 解决问题 ····· 331
12.4 本章小结 ····· 334

参考文献 ····· 335

第 1 篇　C++程序优化基础

如果说 C++开发人员对于程序性能的追求，丝毫不弱于沙漠中迷途路人对绿洲的渴望，相信很多人会表示赞同（尤其是那些几近偏执的完美主义者）。从使用者的角度来看，软件的可用性始终都是第一位的。即使程序的功能再强大，缓慢的执行速度或是庞大的资源消耗都会让人望而生畏。不幸的是，开发高性能的 C++程序并不是一件简单的工作，起码不会比在沙漠中找水更容易（从事性能提升的 C++工作者也许有相似体会）。它需要开发人员有坚实的 C++编程基础、深厚的数据结构知识，以及宽阔的知识面。当然这有些危言耸听，不过起码说明了本书存在的必要性。

如果你希望开发出高效的 C++程序，需要做的第 1 件事就是熟悉所使用的这门语言及其相关的各种基本数据结构，这正是本篇的目的（当然，如果你已经是一个熟练的 C++开发人员，那么可以直接跳过本篇）。深入了解 C++对象的内存布局、分配及释放方式，以及生命周期，能够帮助你使用正确的编程方法优化生成的代码的大小，并且提高内存使用效率。例如，在什么情况下禁止在堆中产生某种对象，而在什么情况下确保一个对象在堆中产生，以及这样的方法对程序性能提升有何帮助。

本篇详细介绍影响 C++程序性能的主要特性，构造/析构函数关乎对象的资源是否被正确地分配和释放；继承和虚函数是 C++的核心之一，而继承同样也是影响 C++程序性能的一个重要因素；临时对象是比较容易被人忽视的性能影响因素，但其确确实实非常重要；inline 函数是把双刃剑，适当使用能够降低函数的调用开销，滥用则有可能适得其反。

另一方面，本篇还从各种常用操作的角度（遍历、插入、删除、排序及查找），对几种主要数据结构（数组、链表、哈希表及二叉树）的时间及空间复杂度进行详细分析。同时，读者可以通过一个详细的程序实例分析动态数组，即在大型软件系统中很常用的一种数据结构。

总之，通过阅读本篇，读者将熟悉在何种应用环境下使用最佳数据结构，并且深入理解各种 C++语言的对象模型和性能特性。

第 1 章　C++对象模型

对象模型是面向对象程序设计语言的一个重要方面，它会直接影响面向对象语言编写程序的运行机制及对内存的使用机制，因此了解对象模型是进行程序优化的基础。本章将首先介绍一般意义上程序中的数据在内存中的分布，以及程序使用的不同种类的内存等基本概念。然后介绍 C++语言中对象的生命周期，以及 C++对象的内存布局。只有深入了解了 C++对象模型，才能避免程序开发中一些不易察觉的内存错误。从而改善程序的性能，提高程序的质量。