计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(6):100−104 [doi: 10.15888/j.cnki.csa.006918]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

面向国产平台的程序并发性能分析技术①

王立杰, 胡先浪, 张大方

(江苏自动化研究所 电子设备事业部, 连云港 222006)

摘　要: 随着国产处理器和国产操作系统的逐步推广应用, 越来越多的开发人员在国产平台下开发多线程程序. 目前国产平台普遍采用的 Qt Creator 工具中缺乏可视化的并发性能分析工具, 使得优化由于多线程同步/互斥和资源竞争带来的性能问题变得特别困难. 设计一种 Qt Creator 下的并发性能分析方案, 通过实时监控程序并发事件, 采集程序运行过程中的并发性能数据, 分析程序并发性能瓶颈和死锁原因, 并以插件形式进行多视图数据显示. 通过实验表明, 该并发性能分析方案可以方便、快捷地辅助用户开发多线程并发程序, 提高软件开发效率.

关键词: 国产平台; 多线程; 并发; Qt Creator 插件; 死锁检测

引用格式: 王立杰,胡先浪,张大方.面向国产平台的程序并发性能分析技术.计算机系统应用,2019,28(6):100–104. http://www.c-s-a.org.cn/1003-3254/6918.html

Program Concurrency Performance Analysis Technology for Domestic Platforms

WANG Li-Jie, HU Xian-Lang, ZHANG Da-Fang

(Department of Electronic Equipment, Jiangsu Automation Institute, Lianyungang 222006, China)

Abstract: With the gradual promotion and application of domestic processors and domestic operating systems, more and more developers are developing multi-threaded programs under domestic platforms. The lack of visual concurrent performance analysis tools in the Qt Creator tools commonly used in domestic platforms makes it extremely difficult to optimize performance problems due to multi-thread synchronization/mutual exclusion and resource competition. This study designs a concurrent performance analysis scheme under Qt Creator through real-time monitoring of concurrent events, collects concurrent performance data during program running, analyzes performance concurrency bottlenecks and deadlock causes, and displays multi-view data in plug-in form. Experiments show that the concurrent performance analysis program can easily and quickly assist users to develop multi-threaded concurrent programs and improve software development efficiency.

Key words: domestic platform; multithreading; concurrent; Qt Creator plug-in; deadlock detection

随着龙芯 3A3000、申威 421 和飞腾 1500A 等国产多核处理器的投入使用以及中标麒麟、深度、银河麒麟等国产操作系统的快速推广应用, 国产处理器与国产操作系统任意两两组合构成了目前国产平台多样化的生态格局[1]. 目前在国产平台上开发并发程序经常采用多线程技术来实现, 每个线程利用单个 CPU 核的计算能力, 多个线程就可以更好地利用整个 CPU 的多核处理能力来实现复杂的功能[2]. 但是, 线程需要通过信号量、互斥锁、临界区等并发资源来实现相互之间的同步. 多线程同步机制选择必须谨慎, 因为不恰当的任务同步可能导致一段时间内一个任务运行而其他任务阻塞的情况, 使得程序性能下降很多. 多个线程在使用共享资源时, 就会发生竞争. 如果程序设计的不好, 则程序会发生死锁[3,4].

① 收稿时间: 2018-12-04; 修改时间: 2018-12-25; 采用时间: 2019-01-03; csa 在线出版时间: 2019-05-25

当前, 在国产平台中使用图形化集成开发环境开发多线程并发程序时, 由于缺少良好的并发性能分析工具, 使得用户难以分析程序中的并发资源使用情况, 以及判断程序中可能存在的死锁问题, 从而影响了应用程序的开发. 开发面向国产平台的并发性能分析插件, 一方面能够让用户了解程序动态运行过程和并发资源的使用情况, 对自己的程序进行优化; 另一方面, 可以解决程序中因为并发资源争用而导致的死锁问题.

本文提出的 Qt Creator 下并发性能分析插件可以实时监控程序不同线程的锁、信号量及同步操作, 记录相关行为信息, 同时采集程序执行过程中的上下文切换、cache 命中率等反映程序执行性能的数据. 插件提供了性能数据分析和死锁检测的可视化界面, 便于上层开发理解和分析.

1 相关工作

英特尔开发的 Vtune Amplifier[5]工具支持对并发应用程序的性能进行分析, 使用线程分析工具可以获取处理器计算能力使用效率, 以及在线程运行时定位由于线程同步时的资源争用导致的处理器效率低的问题, 工具使用有效 CPU 利用率指标作为线程效率的主要衡量指标. Vtune 功能比较完善, 但使用费用较高, 仅支持 Intel 的处理器.

微软开发的 Visual Studio 可以通过性能资源管理器分析并发程序中的信号量、临界区等共享资源的同步与互斥时间开销, 包括等待时间和使用时间, 可以查看争用最激烈的资源和代码段等信息, 进一步优化并发程序性能, 目前仅支持 x86 平台[6].

Helgrind 是一个 Linux 平台下的一个开源并发性能分析工具[7], 用于检测使用 POSIX pthreads 线程原语的 C/C ++程序中的同步错误. Helgrind 可以检测错误使用 POSIX threads API、加锁顺序导致的死锁和数据争用三类错误. Helgrind 是基于命令行的工具, 没有友好的人机界面, 同时使用该工具会引入较大的系统开销.

目前国产平台下大量采用 Qt Creator 作为程序开发工具, 在 Qt Creator 下缺乏一种低开销的可视化并发性能分析插件来辅助应用开发.

2 Qt Creator 插件技术

Qt Creator 是一种跨平台的集成开发环境[8], 它采用微内核架构, 所有的功能都是通过插件实现, 其结构如图 1 所示.

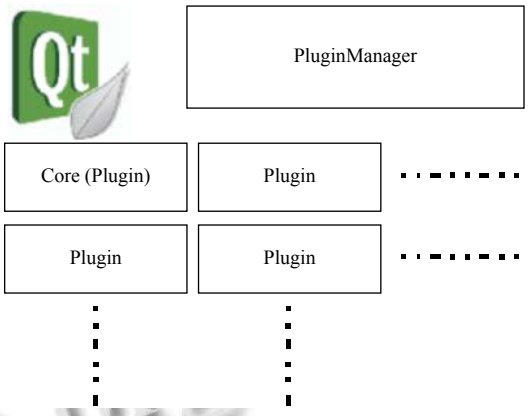

图 1 Qt Creator 架构示意图

Qt Creator 最主要的功能是在名为 Core 的插件中实现的. Core 插件暴露了大量核心对象如主窗口本身、视图、导航面板、大纲、文件系统、菜单等, 这些对象构成了基础的集成开发环境. Core 插件也提供了大量与环境进行交互的接口, 比如 IEditor、IDocument、IOptionsPage、IEditorToolBar 等. 插件管理器管理所有的插件, 包括读取插件描述文件, 解决插件依赖, 加载插件, 按正确的顺序初始化插件等. 插件管理器提供插件合作机制以支持插件相互依赖, 共同工作.

3 面向国产平台的并发性能分析

在以国产处理器与国产操作系统为核心的国产平台上开发面临最大的问题就是生态链不完善. 为了弥补国产平台上缺少并发性能分析工具的问题, 我们设计了面向国产平台的并发性能分析方案, 该方案分为并发事件监控、性能数据采集和分析插件界面三个模块, 其整体构成如图 2 所示. 并发事件监控模块主要通过接管操作系统并发相关接口 API, 实时捕获并发事件. 性能数据采集与分析模块获取并发事件发生时的线程、上下文和时间戳等信息, 并根据采集到的数据进行死锁分析. 分析插件界面主要包含多视图并发性能数据展示界面、人机友好的数据视图切换和可定位至源代码的死锁检测界面等.

3.1 并发事件监控

目前在国产 Linux 平台上, 常用于线程同步的并发操作主要有 POSIX 信号量和 pthreads[9], 需要监控的详细事件如表 1 所示.

表 1 监听事件列表

监听类别	监听事件
基本线程操作	创建, 阻塞, 销毁等
互斥锁	创建, 销毁, 加锁, 释放等
条件变量	创建, 删除, 等待等
信号量	创建, 等待, 释放, 取值, 删除等

1) 信号量 (semaphore) 是一种用于提供不同进程间或一个给定进程的不同线程间同步手段的原语. 信号量的使用主要是用来保护共享资源, 使得资源在一个时刻只有一个进程 (线程) 所拥有. 信号量的值为正的时候, 说明它空闲. 所测试的线程可以锁定而使用它. 若为 0, 说明它被占用, 测试的线程要进入睡眠队列中, 等待被唤醒.

2) pthreads 库除了提供了一套符合 POSIX 标准的线程创建、运行、销毁的实现机制外, 还提供了符合 POSIX 标准的线程同步库. 在 pthreads 库中, 主要提供了互斥体 (mutex) 与条件变量 (conditional variable) 给线程进行同步. 互斥体可以被不同的线程用来同步对一个共享资源的访问, 例如不允许对一个共享资源写入的同时进行读取. 而条件变量则起到了补充性的作用, 即可以用来通知其它的线程某个共享资源的状态已经发生了变化.

为了不修改系统库和应用程序, 本文使用了 Linux 共享库的 LD_PRELOAD 机制接管 POSIX 并发相关接口, 在用户进程载入时首先载入性能分析库. 在性能分析库中定义了与 POSIX 并发接口完全相同的接口, 在接口实现里首先调用真正的 POSIX 接口, 然后再执行监控模块, 在监控模块里收集并发事件信息, 从而实现了监控过程的非侵入.

3.2 性能数据采集与分析

在对程序进行并发性能分析时, 只是监控并发事件的发生还不能有效帮助用户开发多线程并发程序. 用户需要了解发生并发事件时的所有关键信息, 包括事件发生的时间、所属线程和发生并发事件时的上下文信息, 以便精确定位发生性能瓶颈甚至死锁的具体原因. 在 3.1 节中定义的监控模块中利用系统函数可以方便的采集到时戳和所属线程信息. 为了获取发生并发事件时的上下文信息, 本文使用了第三方库 libunwind[10], 该工具定义了可移植的、高效的 C 编程接口来获取程序的调用栈. 获取上下文信息基本流程图如图 3 所示.

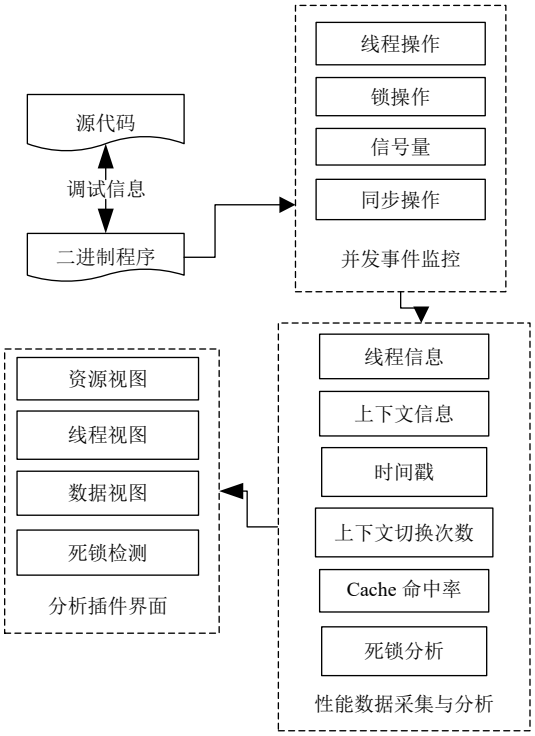

图 2 面向国产平台的并发性能分析插件总体框架

利用二进制文件中的调试信息将调用栈中的符号映射为源代码中的行号, 从而能精确定位发生并发事件在源代码中的位置, 进而为性能优化提供依据.

除了并发事件以外, 程序执行时的某些系统参数在一定程度上也反映了程序执行的并发性能, 如 Cache 命中情况, 上下文切换次数等. 为了更全面、低开销的获取程序并发性能态势, 本文借助硬件性能计数器 PMU 获取性能数据. 由于国产处理器会根据自身体系结构特征定义不同的硬件事件组合, 不同处理器的硬件性能计数器参数均不相同[11,12]. 考虑到事件监测和性能分析的需求, 不同处理器的事件集合往往在功能上会有交集. 通过将国产处理器共有事件特性抽象出来, 设计统一的 PMU 访问接口, 将不同处理器中存在功能共性的事件抽象成接口专用的事件集并同统一命名, 简化 PMU 访问过程, 其基本架构如图 4 所示.

基于上述采集到的并发性能数据, 根据死锁形成的条件对程序是否存在死锁进行分析. 死锁指的是在并行程序中, 多个并行任务形成相互等待的状态, 导致这些任务均无法继续执行. 死锁发生的条件被称为 Coffman 条件, 它们包括以下 4 条[13]:

(1) 互斥. 即必然有某个共享资源只能被单个任务使用, 而不能被多个任务并发访问;

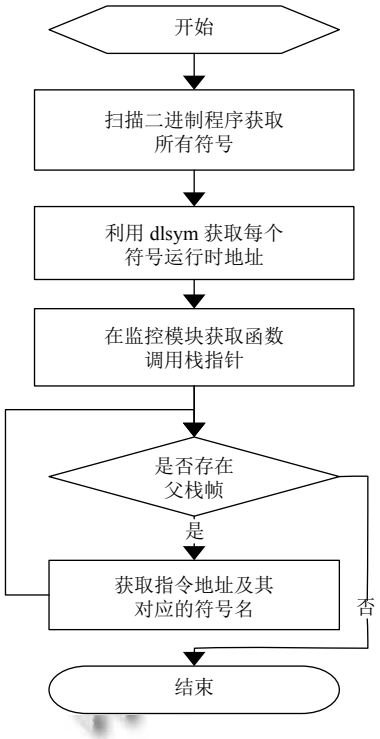

图 3 获取上下文信息流程

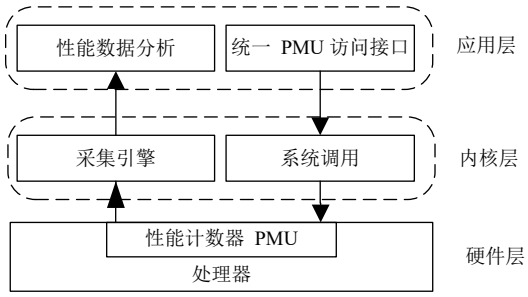

图 4 面向国产平台的 PMU 统一访问架构

(2) 任务处在持有一个共享资源, 同时又要求拥有其他共享资源的状态;

(3) 共享资源无法被抢占 (preempt), 只能由拥有者主动放弃独占才能被释放;

(4) 循环等待. 即一组任务每个都持有一个共享资源, 同时在等待其他任务已经持有的共享资源.

因此, 对于死锁的检测, 主要也是针对上述 4 个条件进行. 针对第一个条件, 在上述同步对象中, pthread 互斥体与 POSIX 信号量均产生了互斥, 因此满足第一个条件. 而一旦持有相应的同步对象, 也只有等待持有者主动放弃才可以释放, 因此也满足第三个条件. 所以只需要检测第二个条件与第四个条件即可.

根据采集到的性能数据, 如果发现某个线程 (假设其为 t1) 在持有某个同步对象 (假设其为 s1), 同时又在调用同步原语等待其它同步对象 (假设其为 s2) 时, 即可得知第二个条件被满足. 在此基础上, 若发现 s2 已经被其它线程持有 (假设其为 t2), 而 t2 也在等待其它的同步对象时, 以此类推, 若最后发现 tn 在等待 s1, 则形成了循环等待, 满足第四个条件. 一旦发现上述第二个与第四个条件被满足, 插件即可判定形成了死锁.

3.3 分析插件界面

在 Qt Creator 菜单中增加“concurrency analyze”菜单, 并添加“start”、“stop”和“detect deadlock”子菜单, 在数据输出面板中增加概览视图、时间线视图和死锁检测视图.

1) 在概览视图中, 显示每个线程的生命周期、阻塞时间以及阻塞时间占比, 在时间轴上显示每个线程等待和占用共享资源的情况, 从资源角度显示每一个等待和占用该资源的线程. 在性能数据窗口能实时显示多线程上下文切换和资源争用等数据.

2) 在时间线视图中, 按时间顺序显示每个并发事件的详细信息, 通过每个事件可以查看详细函数调用栈.

3) 在死锁检测视图中, 死锁检测报告能显示所有线程循环等待信息, 通过每条等待信息可以查看详细函数调用栈.

4 系统实现

本文在国产龙芯和申威平台上实现了上述并发性能分析方案, 其中龙芯运行环境配置如表 2 所示, 申威运行环境配置如表 3 所示.

表 2 龙芯运行环境配置

运行环境	参数
主机	龙芯 3A3000 台式机
操作系统	中标麒麟 5.0 龙芯版
集成开发环境	Qt Creator4.0.1

表 3 申威运行环境配置

运行环境	参数
主机	申威 421 台式机
操作系统	深度操作系统 15.5 申威版
集成开发环境	Qt Creator4.0.1

面向国产平台的并发性能分析插件如图 5 到图 7 所示. 在图 5 中用户打开待分析工程, 点击图中 A 区“concurrency analyze”的“start”菜单, 即可开始采集性能数据. 点击“stop”后进入概览视图. 在概览视图 (图 5 中 B 区) 中能查看并发性能数据, 包括线程、并发资

源、上下文切换等数据. 通过概览视图可以辅助用户初步定位程序并发性能瓶颈.

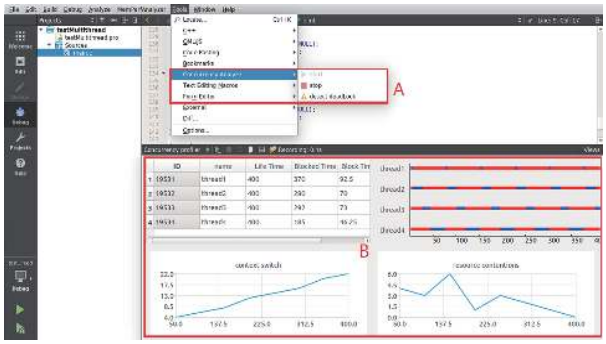

图 5 插件概览视图

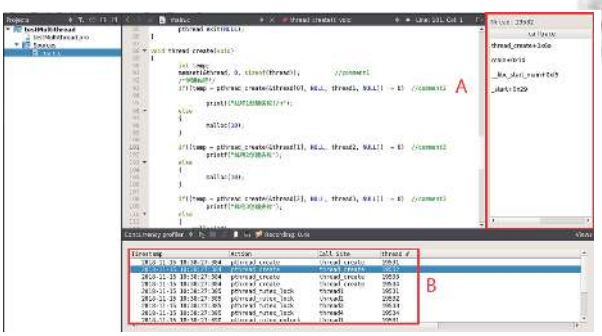

图 6 插件时间线视图

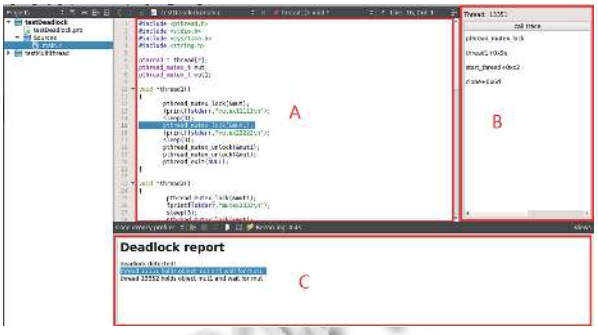

图 7 插件死锁检测视图

通过标签切换到时间线视图, 如图 6 所示. 在图 6 中 B 区按时间顺序详细显示并发事件类型、调用者和所属线程, 任意点击并发数据, 在图 6 中 A 区显示详细的函数调用栈信息. 通过时间线视图可以详细定位并发性能低效的位置.

打开死锁测试用例, 点击“detect deadlock”即可进入死锁检测视图, 如图 7 所示. 在图 7 中 C 区给出死锁检测报告, 如果程序存在死锁, 检测报告会给出循环等待时每个线程等待的位置, 任意点击一处数据, 在图 7 中 B 区会显示等待时刻的函数调用栈. 通过死锁检测视图可以精确定位死锁位置.

5 结语

本文阐述了 Qt Creator 下的插件开发机制, 分析了目前国产平台下多线程程序开发遇到的瓶颈, 设计了低开销的性能数据采集后端和人机友好的并发性能分析插件界面, 实现了对 Qt Creator 开发工具的功能扩展, 大大提高国产平台下多线程程序开发效率, 同时通过辅助优化并发程序性能, 有助于弥补当前国产平台性能较低的问题. 本文设计的插件进一步丰富完善了国产平台开发生态链, 对国产平台进一步推广应用起到一定促进作用.

利用 Qt Creator 的插件机制扩展其并发性能分析功能可以有效辅助用户开发多线程程序, 但仍然存在一些不足, 如目前只能适用于国产龙芯和申威处理器平台, 对于国产飞腾平台还不支持, 性能数据视图设计比较简单, 这些问题需要在下一步工作中继续改进.

参考文献

1 张晓清, 龚波, 田丽韫, 等. 国产自主可控应用性能优化研究. 软件, 2015, 36(2): 5–9. [doi: 10.3969/j.issn.1000-386x.2015.02.002]
2 温莎莎, 刘轶, 刘弢, 等. PPAT: 一种 Pthread 并行程序线程性能分析工具. 计算机应用与软件, 2012, 29(11): 43–47.
3 郑龙. 多线程锁同步运行时特征分析与调优机制研究[博士学位论文]. 武汉: 华中科技大学, 2016.
4 贾非. 多线程并发系统瓶颈分析及优化研究[硕士学位论文]. 北京: 北京信息科技大学, 2014.
5 Intel VTune amplifier. https://software.intel.com/en-us/vtune.
6 利用 Visual Studio 2010 中的 Concurrency Visualizer 优化性能. https://msdn.microsoft.com/zh-cn/magazine/ee33602.
7 Valgrindusermanual.http://valgrind.org/docs/manual/manual.html.
8 Qt Creator manual. https://doc.qt.io/qtcreator/index.html.
9 周丽, 焦程波, 兰巨龙. Linux 系统下多线程与多进程性能分析. 微计算机信息, 2005, 21(9-3): 118–120, 149.
10 The libunwind project. https://www.nongnu.org/libunwind/.
11 闫洁, 徐恒阳, 安虹, 等. Pview: 一种基于 PMU 的支持并行程序性能分析的新方法. 计算机科学, 2011, 38(2): 288–292. [doi: 10.3969/j.issn.1002-137X.2011.02.070]
12 彭林, 方建滨, 杜琦, 等. 飞腾 1500A 处理器性能分析工具 Likwid 研究. 计算机工程与科学, 2018, 40(7): 1147–1154. [doi: 10.3969/j.issn.1007-130X.2018.07.001]
13 黄理. 基于 Petri 网的多线程死锁检测研究[硕士学位论文]. 合肥: 中国科学技术大学, 2015.