**东北大学**

**本科毕业设计（论文）开题报告**



**题 目 支持缓存与内存带宽动态划分的多核实时调度算法设计与实现**

**学 院 计算机科学与工程学院**

**专 业 计算机科学与技术**

**学生姓名 王岩**

**学 号 20215861 年级 2021级**

**指导教师 敖志广**

**教务处制表**

**二Ο二四 年 十二 月 二十五 日**

|  |  |
| --- | --- |
| **选**  **题**  **意**  **义** | 现代多核处理器通常采用共享缓存和内存带宽的架构，以提高资源利用率。然而，这种共享架构也带来了资源争用问题，影响实时任务的可预测性和执行性能。  尽管共享缓存可以帮助提高平均性能, 但由于复杂的核间共享缓存干扰问题, 也使最坏情况下的时序分析变得更具挑战性: 当不同核上同时运行的任务访问映射到相同的缓存集内存时, 它们可能会从缓存中逐出彼此的缓存内容, 从而导致难以预测的缓存不命中, 也就是抖动。  并且内存访问存在核心间依赖性，因为来自一个核心的内存访问也可能受到其他核心请求的影响，为了防止有类似“cache炸弹”的任务过度占用内存带宽，加入动态内存带宽分配也是很有必要的。  特别是，缓存与内存带宽具有密切的联动关系：处理器优先访问缓存，未命中时才访问内存。缓存分配大小直接影响命中率，从而决定对内存带宽的需求，二者需动态平衡以优化资源利用率并减少争用。这两者相结合也能使系统的实时性能更高，即更加精确地预测WCET。  因此，实现支持缓存与内存带宽动态划分技术的实时调度算法，是提升实时性能、优化资源利用率和保障任务实时性的关键挑战。  Meng Xu等人提出了一种使用抢占的固定优先级的全局作业级动态缓存分配的调度算法gFPca[1]; Heechul Yun等人提出了一个高效的内存带宽预留系统MemGuard[2]。本次毕设就是结合这两种算法，基于LITMUS-RT[3]实验平台设计并实现一个支持缓存和内存带宽动态划分的算法，从而进行一次优化资源利用率和保障任务实时性的尝试。 |
| **国**  **内**  **外**  **研**  **究**  **现**  **状**  **概**  **述** | 为了解决核间缓存干扰的问题，Guan等人提出了一个具有任务级动态缓存分配的非抢占固定优先级调度算法，称之为nFPca[4]。因为nFPca不允许抢占，因此可以简化可调度性分析。然而，由于调度是基于不可抢占的，这种做法虽然可以减少由于任务抢占产生的系统开销，但是也可能会由于无法获取资源导致较高优先级任务错过截止期。  基于此项研究中优先级倒转的问题，Meng Xu等人提出了全局抢占固定优先级(gFP)算法的一种缓存分配变体gFPca[1]，gFPca在作业开始或恢复时(即调度时)动态的给运行的任务分配缓存，并且它允许高优先级任务抢占低优先级任务的CPU和缓存资源。它还允许高优先级任务由于缓存资源不足而无法执行时执行低优先级任务，从而进一步提高缓存核CPU利用率。由于现存算法中通常不存在额外的恢复和抢占事件，gFPca的缓存开销计算非常具有挑战性。因此，Meng Xu等人开发了一种新的“开销”感知的分析方法[1]，Meng Xu等人的实验评估表明，他们提出的开销计算方法是高度准确的。此外，与缓存无关的gFP相比，gFPca大大提高了缓存密集型工作符在的可调度性, 在Meng Xu等人的实验中，它在大多数情况下都优于nFPca。  全局 EDF 调度是多核实时系统中非常经典的一种调度算法，但在实际的系统 应用中由于现在系统内部资源越来越复杂，单纯的全局EDF算法并未表现出较好的性能，这种调度算法不具备有关系统资源的管理的能力，甚至可能导致系统出现负面后果。林宇晗等人提出了gEDFca[5]，这是一种以缓存划分方式管理共享缓存的可抢占全局EDF调度算法。王书墨等人实现了该算法，并在ARM硬件平台上运行并调试了该算法[7]。  内存访问存在核心间依赖性，因为来自一个核心的内存访问也可能受到其他核心请求的影响；DRAM控制器通常采用调度算法来重新排序请求，以最大化整体DRAM吞吐量[6]。但由于内存访问时间的高方差，所有这些因素都会影响内存密集型实时应用程序的时间可预测性。Heechul Yun等人提出了、设计并实现了一种新的，高效的，细粒度的内存带宽预留系统，称之为MemGrard[2]。MemGuard将内存带宽分为两部分: 保证和尽力。保证带宽表示DRAM系统可以提供的最小服务速率，而额外的可用带宽是尽力而为的，系统是无法保证的。在满足每个核心的保证带宽后，通过利用尽力而为的带宽进一步提高了系统吞吐量。  Lanshun Nie等人提出了用于联合调度下并行实时任务的整体资源分配框架[8]，在他们提出的框架下，除了专用的核心之外，每个并行任务还分配有专用的缓存和内存带宽资源。他们还提出了一种整体资源分配算法，该算法很好地平衡了不同资源之间的分配，以实现良好的可调度性。Meng Xu等人提出了一种用于多核实时系统的整体缓存和内存带宽资源分配策略CaM [9]，CaM是为分区调度而设计的，其中任务被映射到内核上，共享缓存和内存带宽资源在内核之间进行分区，以减少并发访问造成的资源干扰。通过将具有相似特征（在资源需求方面）的任务分组到同一核心，它使每个核心上的任务能够充分利用分配的资源。  LITMUS-RT 这个开源的实时调度平台是北卡罗来纳州立大学建立了实时系统 研究的团队师生共同开发的。LITMUS-RT是Linux 内核的一种扩展补丁，提供多种基本的多核实时调度策略，还提供了开放的调度器插件接口以及配套的用户和追踪工具库，方便开发者在此平台上实现自定义的调度算法[10]。近年来有很多被提出被优化的调度算法都是基于 LITMUS-RT 平台实现的，但暂时 LITMUS-RT 平台还没有实现能够动态支持缓存与内存带宽动态划分技术的实时调度算法。 |
| **主**  **要**  **研**  **究**  **内**  **容** | 本次毕业设计预计将考虑以下的几个方向，分别为：调研支持缓存与内存带宽动态划分的多核实时调度领域最新进展；在Linux中设计并实现缓存和内存带宽划分技术，并验证任务隔离性；在Linux中实现支持多资源动态划分的实时调度算法；在ARM硬件平台上实现并验证该算法。  1. 调研支持缓存与内存带宽动态划分的多核实时调度领域最新进展: 系统梳理相关领域的研究成果，分析现有调度算法的优缺点及其在资源争用和任务可调度性方面的表现，为算法设计提供理论支持。  2. 在Linux中设计并实现缓存和内存带宽划分技术，并验证任务隔离性: 通过内核模块设计，开发支持缓存与内存带宽动态划分的技术，重点验证其在多核环境下对任务隔离性的保障效果以及动态划分技术的运行开销。  3. 在Linux中实现支持多资源动态划分的实时调度算法: 结合缓存和内存带宽动态划分技术，提出并实现一种能够动态适配任务多种资源需求的实时调度算法，保证任务的实时性。通过实际测试验证算法的有效性和运行开销。  4. 在ARM硬件平台上实现并验证该算法: 首先使用模拟器运行并调试该算法，接着在ARM硬件平台上运行并调试该算法，以验证该算法的可行性。 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **拟采用**  **的研究**  **思　路** | 根据毕设的选题与实际业务场景，我初步拟定的研究思路从下列问题作为导向进行切入：gFPca和MemGuard是如何实现的？怎么将这两个算法结合起来？ 以及 如何验证算法的有效性？  针对“gFPca和MemGuard是如何实现的”的问题，计划首先通过阅读论文理解其实现思路，并基于LITMUS-RT平台进行复现，然后在模拟器上进行 仿真验证。  针对“怎么将这两个算法结合起来”的问题，该问题的关键是缓存分配大小直接影响命中率，从而决定对内存带宽的需求，二者需动态平衡以优化资源利用率并减少争用。因为缓存分区和内存带宽都是处理器的属性，所以目前的想法是每个内核数据结构(core\_desc)要有一个holistic\_desc结构指针，来表示每个处理器的与任务相关的缓存分区和内存带宽信息，当发生任务调度时，将被抢占任务的holistic\_desc保存，并加载新任务的holistic\_desc。    如图所示，gFPca\_desc和MemGuard\_desc中各有一个指针指向holistic\_desc，并且这两个算法要使用的和任务相关的关键属性都存储在holistic\_desc中。所以每次任务调度替换holistic\_desc后，这两个算法的属性也会随之改变，这是一个将gFPca和MemGuard结合起来的一个思路。  简单来说，就是将任务相关的缓存分区和内存带宽信息(holistic\_desc)当作任务执行的上下文，跟随任务一起被调度。而gFPca和MemGuard这两个算法依赖holistic\_desc中的信息，这样做不仅结合了这两种算法，也使得代码的编写更加优雅。  针对“如何验证算法的有效性？”这个问题，目前的思路是自己写一个任务集，并对有无内存带宽分配这两种情况做一个对照实验，任务集的制作方法以及实时性能的计算方法参照Meng Xu 等人的gFPca[1]。 | | | |
| **工作**  **进度**  **安排** | **序号** | **起止日期** | **任务** | **提交的阶段成果** |
| **1** | 1月2日—1月23日  （寒假） | gFPca算法学习 | 论文汇报PPT，复现代码及实验报告 |
| **2** | 1月24日—2月13日  （寒假） | MemGuard算法学习 | 论文汇报PPT，复现代码及实验报告 |
| **3** | 2月14日—3月6日  （寒假，1-1周） | 动态缓存和内存带宽划分算法设计 | 算法设计 |
| **4** | 3月7日—4月3日  （寒假，2-5周） | 代码实现与调试 | 经过模拟器仿真调试的源程序 |
| **5** | 4月4日—4月24日  （6-8周） | 在ARM硬件平台验证算法。 | 实验报告 |
| **6** | 4月25日—5月8日  （9-10周） | 整理论文 | 论文全文 |
| **7** | 5月9日—5月29日  （11-13周） | 修改、审核论文 | 论文全文 |
| **8** | 5月30日—6月8日  （14-14周） | 准备答辩 | 各种答辩材料 |

|  |  |
| --- | --- |
| **参考**  **文献**  **目录** | 1. Meng Xu, Linh Thi Xuan Phan, Hyon-Young Choi, and Insup Lee, "Analysis and Implementation of Global Preemptive Fixed-Priority Scheduling with Dynamic Cache Allocation", IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2016) , 1-12. April 2016. 2. Heechul Yun , Gang Yao , Rodolfo Pellizzoni , Marco Caccamo , Lui Sha,et al. MemGuard: Memory Bandwidth Reservation System for Efficient Performance Isolation in Multi-core Platforms (2013) 3. LITMUS-RT: Linux Testbed for Multiprocessor Scheduling in Real-Time Systems: https://www.litmus-rt.org 4. Guan N, Stigge M, Yi W, et al. Cache-aware scheduling and analysis for multicores[C]//Proceedings of the seventh ACM international conference on Embedded software. 2009: 245-254. 5. 林宇晗,严健,王侃侃,邓庆绪.支持缓存划分的全局 EDF 实时系统调度策略[J]. 东北大学学报(自然科学版),2021,42(12):1673-1680. 6. K.J. Nesbit, N. Aggarwal, J. Laudon, and J.E. Smith. Fair queuing memory systems. In International Symposium on Microarchitecture (MICRO), 2006. 7. 王书墨,林宇晗,邓庆绪,et al.支持缓存划分的多核实时调度技术的研究与实现.东北大学硕士毕业论文, 2022年6月 8. Lanshun Nie, Chenghao Fan, Shuang Lin, Li Zhang, Yajuan Li, and Jing Li. 2022. Holistic Resource Allocation Under Federated Scheduling for Parallel Real-time Tasks. ACM Trans. Embed. Comput. Syst. 21, 1, Article 13 (January 2022), 29 pages. 9. Xu M, Phan L T X, Choi H Y, et al. Holistic resource allocation for multicore real time systems[C]//2019 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS). IEEE, 2019: 345-356. 10. Calandrino J M, Leontyev H, Block A, et al. Litmus^ rt: A testbed for empirically comparing real-time multiprocessor schedulers[C]//2006 27th IEEE International Real-Time Systems Symposium (RTSS'06). IEEE, 2006: 111-126. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **开 题 报 告 会 议 纪 要** | | | | | |
| **时 间** |  | **地点** |  | **主持人** |  |
| **开题**  **评议**  **小组** | **姓 名** | **职 务**  **（职 称）** | **姓 名** | **职 务**  **（职 称）** | |
|  |  |  |  | |
|  |  |  |  | |
|  |  |  |  | |
| **指**  **导**  **教**  **师**  **意**  **见** | **王岩同学的毕业论文选题结合前沿热点问题，具有先进性。对论文的研究意义和背景阐述清楚，确定的论文研究内容完整，时间安排合理，初步研究方案可行。希望严格按照进度安排按时完成毕业设计任务。**  **同意开题。**  **指导教师签名： 2024年12月25日** | | | | |
| **开**  **题**  **评**  **议**  **小**  **组**  **意**  **见** | **1.选题的意义: □ 有理论意义；□ 有实用价值；□ 有理论意义与实用价值；□ 意义不大。**  **2.选题的难度: □ 偏高；□ 适当；□ 偏低。**  **3.工作量的评估: □ 偏大；□ 适当；□ 偏小。**  **4.选题的可行性: □ 可行；□ 不可行。**  **5.学生表现出的综合能力和表达能力: □ 好；□ 较好；一般；□ 较差。**  **6.毕业设计（论文）形式意见：□ 适宜；□ 不适宜。**  **7.开题报告的总体评价: □ 优；□ 良；□ 中；□ 合格；□不合格。**  **8.是否同意论文选题报告：□ 同意；□ 否，需重做**  **9.综合评价：**  **评议小组组长签名： 年 月 日** | | | | |
| **备注：1、填写各栏目时可根据内容另加附页；2、开题评议小组教师不少于3人。** | | | | | |