Batch: D-2 **Roll No.:** 16010122151

Experiment No. 10

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

TITLE: Report on the research paper.

Experiment: Analysis of Contemporary Memory Management Techniques

Resources:

- TechScience
- IJRITCC
- USENIX

Objective: Examine and review recent advancements, challenges, and methods in memory management within modern operating systems.

Title

Enhancements in Linux Kernel's Contiguous Memory Allocation Strategy

---

Introduction: The Linux kernel's Contiguous Memory Allocation (CMA) is key for efficient memory utilization. However, limitations in the current approach, including allocation failures and mapping overhead, affect performance. This study suggests a targeted modification to improve CMA.

Background: CMA is a distinct memory management structure providing an API for device drivers and reserving substantial memory at boot for uninterrupted allocations.

The model emphasizes memory continuity, assignable at boot through kernel commands or device configurations.

Proposed Solution: The proposed approach modifies CMA to reduce failures and latency by implementing a virtual remapping strategy, which delays physical-to-virtual mapping until allocation is required.

Implementation: A specialized device driver intercepts CMA allocation and deallocation, using a custom API to manage CMA's memory space without immediate mapping needs.

Experimental Setup: Tests were conducted on ARM-based hardware (BeagleBone Black, Raspberry Pi 3) with at least 600 MB reserved for CMA in 1 GB RAM setups.

Results: Improved performance, including reduced latency and increased system responsiveness, was observed for ARM-based devices, particularly Android.

Conclusion: The enhanced CMA model in Linux effectively addresses allocation issues through deferred mapping, boosting overall efficiency and response times.

EXTMEM: Application-Specific Memory Management Framework

Introduction: Efficient memory management is critical in handling the diversity of current applications and hardware. Traditional methods often lack adaptability for memory-intensive tasks. EXTMEM introduces a flexible framework supporting application-specific memory needs.

Background: Rising DRAM costs and demand have spurred interest in advanced memory designs, but new management strategies are needed to fully utilize these technologies.

Proposed Solution: EXTMEM is a customizable memory management substrate compatible with Linux, optimized for multithreaded environments, and capable of deploying isolated memory policies.

Implementation: EXTMEM uses upcalls to manage page faults within user space, enhancing adaptability and quality of service (QoS) with minimal kernel adjustments.

Experimental Setup: Microbenchmarks were used to assess memory throughput on a 16 GB allocation with RAM limited to 8 GB, simulating intensive workloads.

Results: EXTMEM consistently exceeded Linux in throughput, showing faster eviction of pages and improved scalability.

Application Study: EXTMEM demonstrated significant benefits in managing memory for graph-based applications.

Conclusion: As an adaptable framework, EXTMEM optimizes memory management, enhancing the performance of data-heavy applications on Linux.

M2ARP: Predictive Memory Management on Mobile Devices

Summary: M2ARP is a predictive system that estimates app relaunch likelihood to manage memory on smartphones. It uses an LSTM model to learn usage patterns, combining these predictions with a fallback LRU mechanism for efficient memory use.

Background: With growing memory demands, traditional LRU-based systems often fail to prioritize frequently accessed apps, degrading performance. M2ARP addresses this by forecasting relaunch likelihood.

Methodology: The system trains on app usage sequences with an LSTM model, estimating which apps are least likely to relaunch soon. An LRU fallback ensures memory management for apps with insufficient data.

Contributions: Key contributions include defining app relaunch distance and a comparison mechanism to optimize low-memory conditions.

Results: Testing showed reduced app load times on mobile devices, outperforming traditional LRU methods.

Future Work: Further improvement is anticipated through models generalizing patterns across multiple user datasets.

Conclusion: M2ARP's predictive approach to memory management offers a practical solution for mobile device performance, effectively balancing app usage and memory resources.