Technical Report

Supporting Real-Time Parallel Task Models with Work-Stealing

Cláudio Maia

Luís Nogueira

Luís Miguel Pinho

HURRAY-TR-120301

Version:

Date: 03-16-2012

Supporting Real-Time Parallel Task Models with Work-Stealing

Cláudio Maia, Luís Nogueira, Luís Miguel Pinho

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail: crrm@isep.ipp.pt

http://www.hurray.isep.ipp.pt

Abstract

Dynamic parallel scheduling using work-stealing has gained popularity in academia and industry for its good performance, ease of implementation and theoretical bounds on space and time. Cores treat their own double-ended queues (deques) as a stack, pushing and popping threads from the bottom, but treat the deque of another randomly selected busy core as a queue, stealing threads only from the top, whenever they are idle.

However, this standard approach cannot be directly applied to real-time systems, where the importance of parallelising tasks is increasing due to the limitations of multiprocessor scheduling theory regarding parallelism. Using one deque per core is obviously a source of priority inversion since high priority tasks may eventually be enqueued after lower priority tasks, possibly leading to deadline misses as in this case the lower priority tasks are the candidates when a stealing operation occurs.

Our proposal is to replace the single non-priority deque of work-stealing with ordered per-processor priority deques of ready threads. The scheduling algorithm starts with a single deque per-core, but unlike traditional work-stealing, the total number of deques in the system may now exceed the number of processors. Instead of stealing randomly, cores steal from the highest priority deque.

Supporting Real-Time Parallel Task Models with Work-Stealing

Cláudio Maia, Luís Nogueira, Luís Miguel Pinho
CISTER Research Centre
School of Engineering of the Polytechnic Institute of Porto
Porto, Portugal
{crrm, lmn, lmp}@isep.ipp.pt

Abstract

Dynamic parallel scheduling using work-stealing has gained popularity in academia and industry for its good performance, ease of implementation and theoretical bounds on space and time. Cores treat their own double-ended queues (deques) as a stack, pushing and popping threads from the bottom, but treat the deque of another randomly selected busy core as a queue, stealing threads only from the top, whenever they are idle.

However, this standard approach cannot be directly applied to real-time systems, where the importance of parallelising tasks is increasing due to the limitations of multiprocessor scheduling theory regarding parallelism. Using one deque per core is obviously a source of priority inversion since high priority tasks may eventually be enqueued after lower priority tasks, possibly leading to deadline misses as in this case the lower priority tasks are the candidates when a stealing operation occurs.

Our proposal is to replace the single non-priority deque of work-stealing with ordered per-processor priority deques of ready threads. The scheduling algorithm starts with a single deque per-core, but unlike traditional work-stealing, the total number of deques in the system may now exceed the number of processors. Instead of stealing randomly, cores steal from the highest priority deque.

Supporting Real-Time Parallel Task Models with Work-Stealing

Cláudio Maia, Luís Nogueira and Luís Miguel Pinho
{crrm, lmn, lmp}@isep.ipp.pt

Context

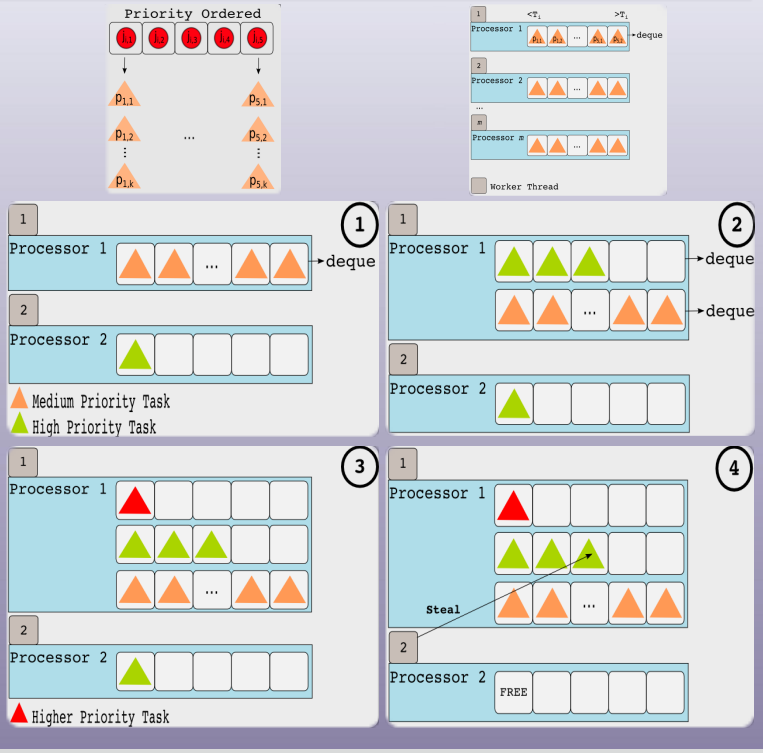
- Growing importance of **parallel task models in real-time applications** poses new challenges to real-time scheduling.
- Task-based parallelism enforced through compilers still **lacks the ability to handle highly complex source code**.
- The usefulness of existing **real-time scheduling approaches is limited by their restrictive parallel task models**.
- In contrast, **the more general parallel task model addressed in our work allows jobs to generate an arbitrary number of parallel threads** at different stages of their computations.

Work-stealing

- One of the **simplest, yet best-performing, dynamic load-balancing algorithms** for shared-memory architectures.
- Uses a "breadth-first theft, depth-first work" scheduling policy with **minimal overhead and good data locality**.
- The **challenge** is to impose a **priority-based scheduling policy** that positively **increases the speedup of parallel applications without jeopardising the schedulability of the system**.

Proposed Approach

- Jobs are **scheduled according to priority** and placed in a global submission queue, **parallel jobs inherit the timing properties of the parent job**.
- Instead of using one deque per core (as it is a source of priority inversion), our proposal is to replace the single non-priority deque of work-stealing with **ordered per-processor priority deques** of ready parallel jobs.
- The scheduling algorithm starts with a single deque per-core, but unlike traditional work-stealing, **the total number of deques in the system may now exceed the number of processors**.
- Instead of stealing randomly, **cores steal from the highest priority deque among all cores**.
- Timing properties of real-time tasks are assured through feasibility analysis.



Work in the scope of the CISTER Research Unit and RECOMP project, co-financed by:

http://www.cister.isep.ipp.pt
Email: cister-info@isep.ipp.pt
CISTER Research Unit ISEP/IPP Rua Dr. António Bernardino de Almeida 431 4200-072 PORTO Portugal