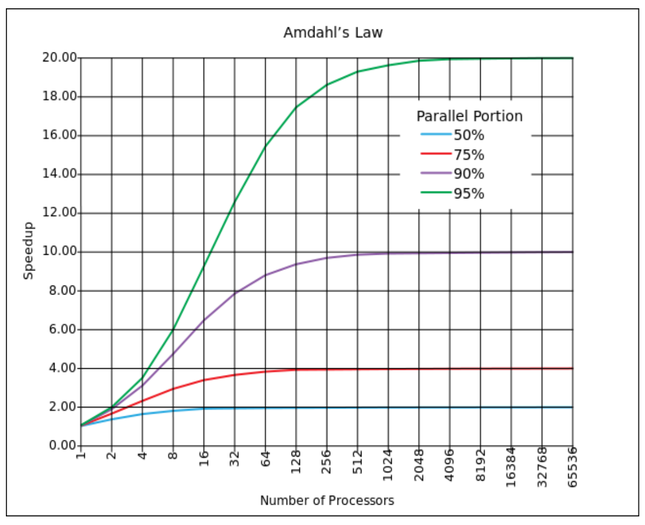
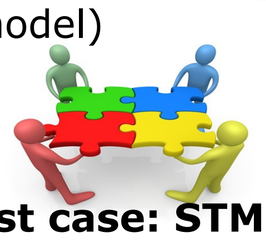
Enhancing the Performance of Assisted Execution Runtime Systems

Gokcen Kestor, Roberto Gioiosa, Osman Unsal, Adrian Cristal, Mateo Valero

Motivation

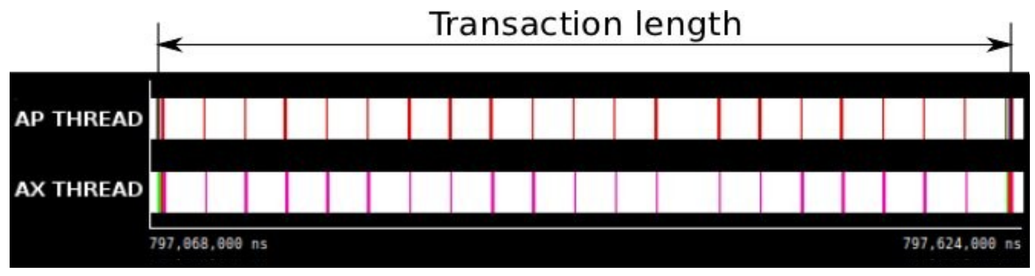
• Amdahl's Laws limits the maximum theoretical speedup achieved by a parallel application
• Extra computing elements can be used to increase single thread performance (Assisted-Execution model)





Test case: STM^2

• Parallel software TM
• Uses additional auxiliary threads (***AxTs***) to offload TM operations
• Application threads (***ATs***) perform computation, TM operations handled by *AxTs*

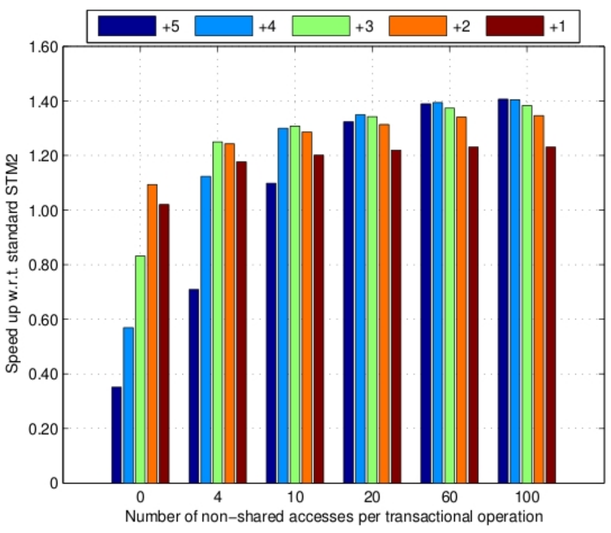


In this scenario,
• *ATs* issue transactional operations at a low rate, thus *AxTs* are frequently idle
• *AxTs "wastes"* hardware resources without doing any process

How to increase the utilization of system?
How to use available resources more effectively?

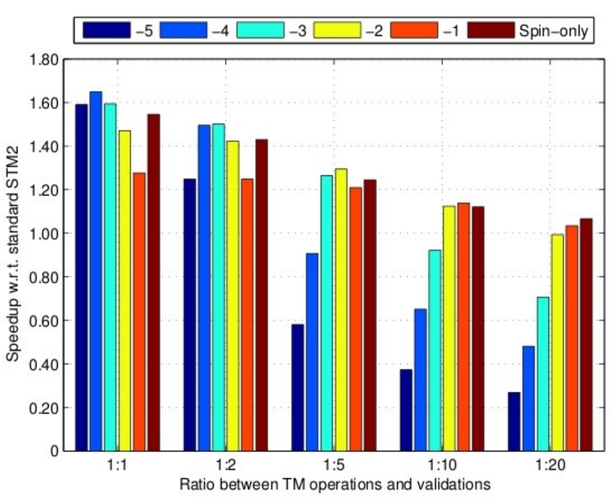
Integrated Fine-Grained Resource Partitioning

We explore static techniques applied at compile time to maximize the effective use of hardware resources through IBM POWER7 hardware thread prioritization:
1) Embarrassingly parallel phases: Auxiliary threads are **idle**
2) Load imbalance inside transactions: Applications/Auxiliary threads are **overloaded**



Overloaded *ATs*

• If #local accesses is limited, reducing *AxTs* priority reverses the imbalance
• If not, an overall performance improvement of 35%



Overloaded *AxTs*

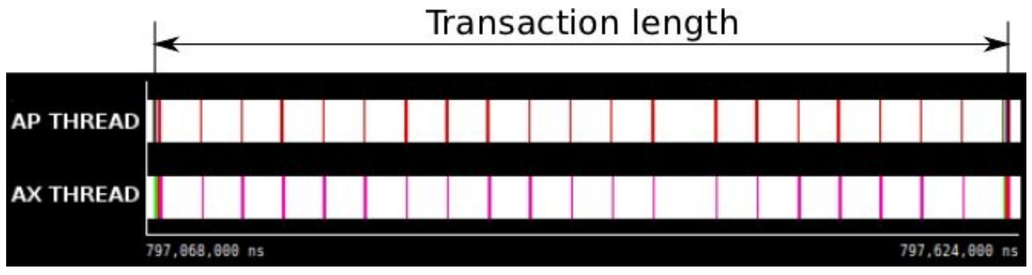
• If the validation is frequent, better to increase *AxT* priority for txs
• If not, increasing *AxT* priority for txs is dangerous
• Reducing *AT* priority at commit phase is safe

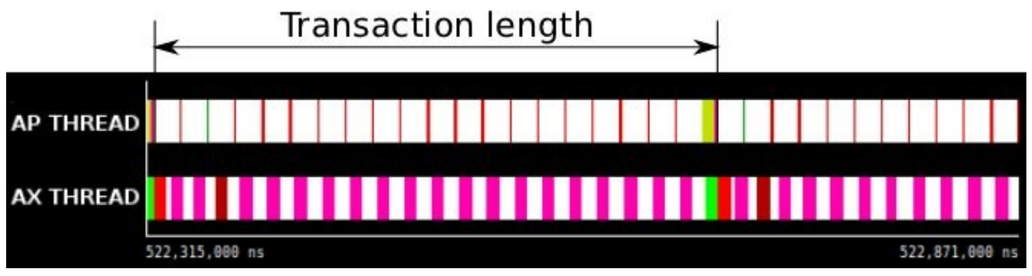
Load imbalance inside transactions

Happens when one or more tasks in a parallel application have more work to perform than the others.

• **Overloaded Application Threads:**
  • ***AxT* is idle,** spinning on the communication channel for incoming messages
  • *AxT priority* can be reduced without performance loss

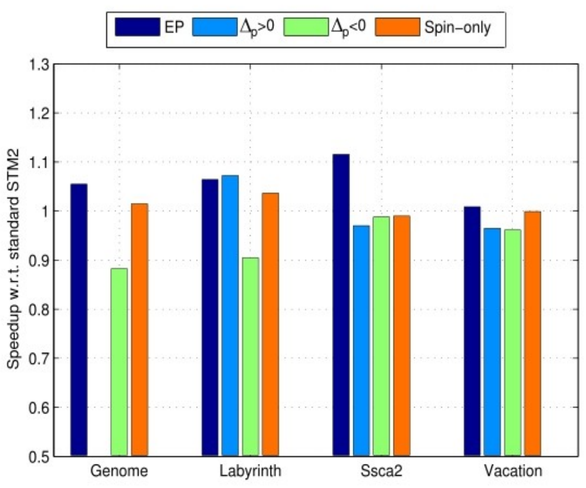
• **Overloaded Auxiliary Threads:**
  • ***ATs* is idle,** waiting for its corresponding AxT at commit phase
  • *AxT priority* can be increased to reduce AT waiting time





Increase the priority of *ATs*, thus improve the performance 27%

STAMP Applications



• Real applications are more complex
• Some static solutions may fail to improve performance
• However, several applications benefit from fine-grained resource allocation

Conclusions and Future Work

Fine-grained resource allocation is crucial for assisted execution systems
• Improves the performance
• Increases the utilization of system
• Uses available resources more effectively

However, static fine-grained resource allocation is not trivial at all:
• Prioritize *AT* or *AxT*, depending on the transaction's structure
• Need to set the right priority value of difference between *AT* and *AxT*

Need a fully automated approach to dynamically Partitioning the hardware resources between *AT* and *AxT*

G. Kestor, R. Gioiosa, T. Harris, A. Cristal, O. Unsal, M. Valero, and I. Hur. STM^2: A parallel STM for high performance simultaneous multi-threading systems. In The 20th IEEE International Conference on Parallel Architectures and Compilation Techniques (PACT), Oct 2011.

Barcelona Supercomputing Center
Centro Nacional de Supercomputación
BSC Microsoft Research Centre
Microsoft Research