Cost of Concurrency in Hybrid Transactional Memory

Elsevier[1]

Radarweg 29, Amsterdam

Elsevier Inc[a,b], Global Customer Service[b,]*

[a]1600 John F Kennedy Boulevard, Philadelphia
[b]360 Park Avenue South, New York

Abstract

State-of-the-art *software transactional memory (STM)* implementations achieve good performance by carefully avoiding the overhead of *incremental validation* (i.e., re-reading previously read data items to avoid inconsistency) while still providing *progressiveness* (allowing transactional aborts only due to *data conflicts*). Hardware transactional memory (HTM) implementations promise even better performance, but offer no progress guarantees. Thus, they must be combined with STMs, leading to *hybrid* TMs (HyTMs) in which hardware transactions must be *instrumented* (i.e., access metadata) to detect contention with software transactions.

We show that, unlike in progressive STMs, software transactions in progressive HyTMs cannot avoid incremental validation. In fact, this result holds even if hardware transactions can *read* metadata *non-speculatively*. We then present *opaque* HyTM algorithms providing *progressiveness for a subset of transactions* that are optimal in terms of hardware instrumentation. We explore the concurrency vs. hardware instrumentation vs. software validation trade-offs for these algorithms. Our experiments with Intel and IBM POWER8 HTMs seem to suggest that (i) the *cost of concurrency* also exists in practice, (ii) it is important to implement HyTMs that provide progressiveness for a maximal set

*Corresponding author
Email address: support@elsevier.com (Global Customer Service)
URL: www.elsevier.com (Elsevier Inc)
[1]Since 1880.

of transactions without incurring high hardware instrumentation overhead or using global contending bottlenecks and (iii) there is no easy way to derive more efficient HyTMs by taking advantage of non-speculative accesses within hardware.

Keywords: `elsarticle.cls`, LaTeX, Elsevier, template

2010 MSC: 00-01, 99-00

References