Title: Non-preemptive scheduling of real-time software transactional memory

Authors: António Barros and Luís Miguel Pinho

Abstract: Recent processor architectures for embedded applications, containing multiple heterogeneous cores and non-coherent caches rely on parallel execution of concurrent software to increase computing power. However, providing predictable synchronisation mechanisms that do not impair the effective parallelisation of tasks if a major challenge.

We propose that a STM contention manager for real-time systems should sequence transactions by their order of arrival (FIFO): the principle behind it is that any transaction should depend only on the presence of older transactions to successfully commit, and if the number of older transactions is limited, so is the time to commit.

However, a preemptive scheduling policy can have adverse effects on the predictability of the progress of transactions.

If a preempted transaction can be aborted be a contending transaction executing with higher priority in the same core in order to avoid deadlock, then the FIFO principle is subverted, and its effects are cascaded to other contending transactions that are executing in other cores, waiting for the preempted transaction to commit before themselves are able to commit.

In another scenario, should a transaction abort for an earlier arrived transaction preempted in another core, and discard work that is acceptable to commit? Again, if the executing transaction overtakes the preempted transaction, then the FIFO principle is subverted, and, in the worst case, the number of aborts can be unlimited.

Therefore, we propose that transactional code should be non-preemptible: this approach makes the transaction attempt immune to conflicts in the same core, and the outcome of the attempt is purely dependent on the transactions executing in other cores at the moment of commit.
This solution is similar to lock-based priority boosting techniques, in which the priority of a job is raised to a non-preemptible value during the execution of a critical section.
The principle resembles FMLP (FIFO ordering, non-preemptive execution of critical sections) but an STM approach can achieve finer granularity and, thus, higher parallelisation of concurrent transactions.

We consider two limited-preemption approaches to transactions: non-preemptive until commit (NPUC) in which the job is non-preemptible since the transaction starts its first attempt until it is finally able to commit,
and non-preemptive during attempt (NPDA), in which the scheduler is able to preempt the job between attempts. NPUC is the most deterministic approach, because the time required to commit is solely dependant on the transactions that are executing at the time the transaction starts. NPDA is able to improve the

responsiveness of higher-priority tasks, since preemptions can occur before the transaction commit, but the time to commit is not easily bounded; however, the success of each attempt is solely dependant on the set of earlier arrived transactions that are still attempting to commit.

Finally, we also consider that a transaction can overtake an earlier arrived contender that is zombie at the time of commit, as an optimisation that does not subvert the principle of FIFO. The reasoning is that overtaking a soon-to-be failed transaction does not increase the overhead of such transaction, and improves the progress of the system.