Speculative Parallel Threading (SPT)

- A promising multi-threading technique to speed up difficult-to-parallelized single-thread applications on multi-core

This research work

- Evaluated the use of transactional memory to support SPT
- Developed state-of-the-art compiler technologies to generate optimal SPT code
- Exploring runtime speculation and parallelization techniques

SPT speculatively parallelizes sequential code and executes the parallel codes speculatively using multiple threads. The supporting hardware or runtime software checks and validates the speculative execution results and initiates recovery when the check fails.
In this demo, we showed speculative parallel execution of a non-parallelizable loop extracted from gcc in the Spec2000Int benchmark on a 2-way dual-core SMP machine running Software Transactional Memory (STM), a pure software implementation of transactional memory. Despite the high runtime overhead in STM and the long inter-thread communication latency on the SMP machine, our SPT version runs almost twice as fast as the sequential version.

Programming Systems Lab, MTL