Software Transactional Memory (STM) systems have poor performance under high contention scenarios. Since many transactions compete for the same data, most of them are aborted, wasting processor runtime. Contention management policies are typically used to avoid that, but they are passive approaches as they wait for an abort to happen so they can take action. More proactive approaches have emerged, trying to predict when a transaction is likely to abort so its execution can be delayed. Such techniques are limited, as they do not replace the doomed transaction by another or, when they do, they rely on the operating system for that, having little or no control on which transaction should run. In this paper we propose LUTS, a Lightweight User-Level Transaction Scheduler, which is based on an execution context record mechanism. Unlike other techniques, LUTS provides the means for selecting another transaction to run in parallel, thus improving system throughput. Moreover, it avoids most of the issues caused by pseudo parallelism, as it only launches as many system-level threads as the number of available processor cores. We discuss LUTS design and present three conflict-avoidance heuristics built around LUTS scheduling capabilities. Experimental results, conducted with STMBench7 and STAMP benchmark suites, show LUTS efficiency when running high contention applications and how conflict-avoidance heuristics can improve STM performance even more. In fact, our transaction scheduling techniques are capable of improving program performance even in overloaded scenarios.