Project abstract

Deterministic highly-scalable concurrency control within database systems

Prior research on deterministic database systems has proposed pre-determining a serializable schedule of transactions prior to their execution. The pre-determined schedule is represented via an explicit dependency graph, which determines the order in which transactions must execute and effectively obviates the necessity for explicit locking.

Within this work we show that building a dependency graph by considering batches of transactions rather than single transactions independently from one another may lead to significant performance increases. We devise an algorithm for creating execution schedules from a set of transactions divided into batches based on an approximation of the MAX-packing problem. We subsequently investigate its performance through a series of simulations and show that our algorithm produces schedules that are much more performant than the baseline schedules created without batching. The performance increase is present throughout a series of scenarios which approximate real workloads. We further investigate the source of the increase in performance and show that the process of transaction batching leads to an increased utilization of shared locks within the system which results from transaction re-ordering within batches. The resulting schedule is highly parallelizable and may offer a new highly performant concurrency control mechanism.

The second semester of work on this project will be focused on replacing the simulation with a functional system to show that the overhead of the schedule creation mechanism does not render the approach impractical. Moreover, the created system will be compared with two phase locking to determine the scenarios in which currently investigated system is superior.