Evaluation of a simple, scalable, parallel best-first search strategy.


Summary: Large-scale, parallel clusters composed of commodity processors are increasingly available, enabling the use of vast processing capabilities and distributed RAM to solve hard search problems. We investigate hash-distributed A* (HDA*), a simple approach to parallel best-first search that asynchronously distributes and schedules work among processors based on a hash function of the search state. We use this approach to parallelize the A* algorithm in an optimal sequential version of the fast downward planner, as well as a 24-puzzle solver. The scaling behavior of HDA* is evaluated experimentally on a shared memory, multicore machine with 8 cores, a cluster of commodity machines using up to 64 cores, and large-scale high-performance clusters, using up to 2400 processors. We show that this approach scales well, allowing the effective utilization of large amounts of distributed memory to optimally solve problems which require terabytes of RAM. We also compare HDA* to transposition-table driven scheduling (TDS), a hash-based parallelization of IDA*, and show that, in planning, HDA* significantly outperforms TDS. A simple hybrid which combines HDA* and TDS to exploit strengths of both algorithms is proposed and evaluated.