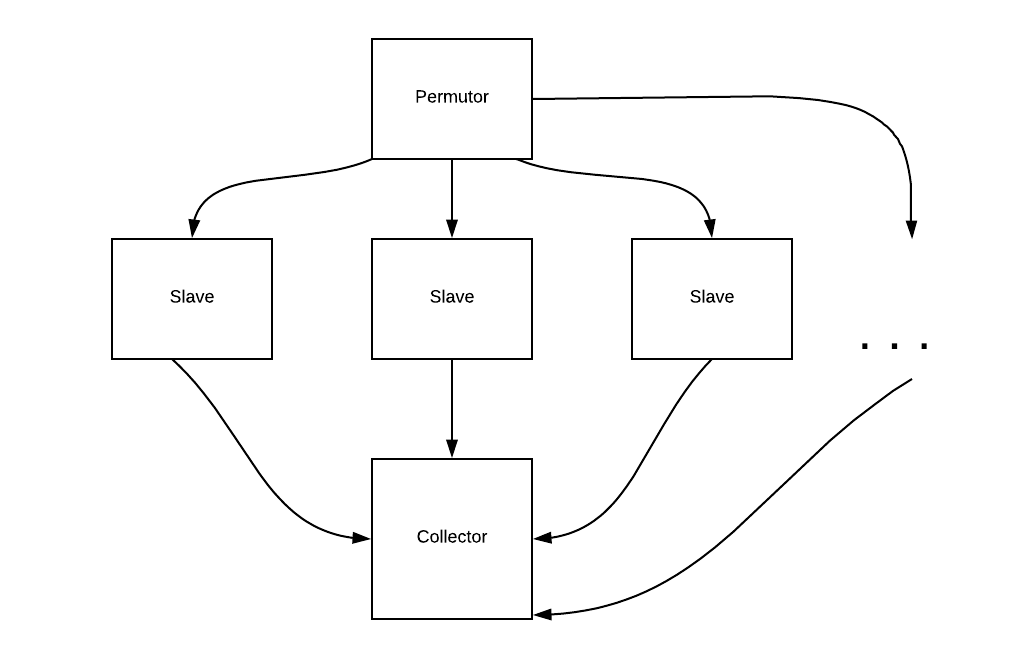
Dustin Cook

CSC 410 final

12/10/2018

Program Description

This solution uses MPI / openMP to solve the N Queens problem in parallel. There are three different types of threads for this application.

* Permutor – The Permutor rotates a tuple and sends the rotations to a slave.
* Collector – Waits for a slave to find all solutions for a given rotation.
* Slave – After receiving a rotation from Permutor, finds all solutions in lexicographical order up to but not including the next rotation of the given tuple.

Compile:

make parallel

Usage:

mpirun -np <threads> -hostfile hosts ./nqueens <n> <print> <omp threads>

* threads – 3 is required to run, its recommended to have 2 + n.
* n – nxn board
* print – 1 to print valid solitions 0 to just print number of solutions.
* omp threads – number of inner omp threads to use, 1 or 6 recommended. (Does not appear to give significant performance boost), used in slave and permutor.

Algorithms and Libraries Uses

* std::next\_permutation
* std::rotate
* std::vector
* MPI
* OMP
* Python for testing
* Make for building

Functions and Structure

There are three major functions: permutor, collector, and slave. Immediately a thread determines what kind of thread it is and calls its appropriate callback function. OMP is in the program, but I have found that it gives no real benefit.

Testing

test.py

Data

Analysis

Resources