Project Proposal

ayb2121
vm2724

Summary

Parallelize A* search on the NP-Complete variation of the traveling salesman problem.

- NP-Complete TSP: given a weighted graph and a maximum length N, does a route with length <= N exist that visits a given list of nodes.
- A* search will not be performed on the nodes themselves, but on all the states of possible routes.
    - Vertices are routes
    - Edges are (legal) swaps between two cities on the route, where a swap is legal if the resulting route is legal (i.e. that path exists in the graph)

Serial Algorithm

A* search requires you to define a heuristic. A state object storing information like 1) current node, 2) visited nodes and 3) total distance traveled. The heuristic can be varied, but we will start with the total distance traveled, which the priority queue will be ordered on. This guarantees correctness - i.e. the first item off the queue that reaches the end is guaranteed to be the shortest path. With the NP-Complete variation of the problem, we can ‘early exit’ if no solution is found, making time comparisons easier.

Parallelization Plan & Anticipated Difficulties

Some ideas:

1. Explore frontier in parallel, each iteration pop x items off the queue and compute their children in parallel, then consolidate them back to the queue. Next iteration repeats.
2. Shard the priority queue up to some state depth (e.g. serial after 5 cities visited) and each branch gets its own queue.

Functional specific ideas:

- For coarse granularity while maintaining work sharing, have a spark explore N nodes before resynchronizing with the shared priority queue, prioritizing
- Implement strategies-compatible problem space?

Anticipated difficulty is correctness when states are explored in parallel. We will need to consolidate all the found solutions in each branch and then get the best one out of those.

Input Data

TSP gets expensive even with only 30-50 cities so the input data can be designed by hand, and IO will not be the bottleneck.