Name: Elias Percy User-ID: cmcz82

Algorithm A: A* Search

Algorithm B: Ant Colony Optimisation

Description of enhancement of Algorithm A:

My first enhancement was a *solutions pool*, a heap of limited size that stores the best tours found throughout the search, so that we can attempt to enhance them further later. Regarding heuristics, I used nearest neighbour in my basic algorithm. I experimented with alternatives – generally, ones that produced better tours (getting closer to the Held-Karp lower bound[1]) also produced much larger runtimes, reducing the exploration of the search. I ultimately decided not to rely on the heuristic function to gain the best tours, rather as a tool to simply indicate which nodes are the best to take. One heuristic I developed, an alteration of nearest neighbour that works on each side of the subtours, I still implemented to some degree, but only to improve the tours on nodes selected as even this as the heuristic, despite attempts to optimise, resulted in huge iteration times for large city sets. Moreover, the power of randomness in finding good solutions to NP-Hard problems was a revelation to me, so I immediately incorporated this as an enhancement: by a given random constant, sometimes a random node is selected to expand (this entailed changing the data structure for the frontier to a heapq rather than PriorityQueue), and sometimes tours in the solutions pool are *mutated*. This allowed for a more explorative search. Additionally, I incorporated 2-opt local search to improve the tours collected in the solutions pool. A conscious decision was made to use 2-opt over 3-opt as after various experiments, the time complexity of 3-opt wasn't worth it since I was generating just-as-good tours by using 2-opt more frequently.

Description of enhancement of Algorithm B:

My initial enhancements were primarily influenced by those found in the ACO textbook[2]. First came nearest neighbour lists, which provided a general reduction in iteration time and a slight increase in tour quality (better paths more likely to be chosen). I experimented with all the major documented variations of ACO, including Elitism, Rank, Max-Min[3], and ACS. Whilst Max-Min at first provided the best tours, it was always neck and neck with Rank, until some of my further enhancements cemented a mixture of Rank and Elitism as the best enhancement out of the aforementioned. My Rank implementation alone provided significantly better tours than my basic implementation. After acute observation of the behaviour of the ant colony, the most apparent thing preventing the best possible tours was stagnation, so my enhancements are mainly derived from this fact. First, I implemented a stagnation detector, which essentially entailed an additional Ant operating to detect when stagnation occurred (i.e., no better tour was found after *n* iterations). Next came pheromone trail smoothing, which reinitialised the pheromone matrix once stagnation was detected to break out of the current minima. Then, inspired by the power of randomness, I conceived *diversity enforcement*: first a function calculates how diverse the ant colony is (based on its spread) after each iteration, and then, if not diverse enough, mutates ants in the ant colony at random to enforce a diverse ant colony (and increase the range of paths searched). Then I implemented a version of 2-opt local search: due to the toll this takes on runtime, I ensured it would only run scarcely – in particular, only when stagnation is *halfway*-reached (only applied to the uber_ant, *or* best so far ant), and when stagnation is reached (applied to the iteration best 6 ants as well as the uber_ant). The purpose of 2-opt is not so much to jump out of, or prevent, a local minima (enforced diversification and pheromone trail smoothing are for this) as it is to optimise this current stagnation – i.e., we convert the tour we stagnated on into the best tour it could possibly be, before smoothing the pheromones. By the way, it was observable that mutation occurred most frequently during short periods of stagnation, and it clearly, quite often, prevents longer periods of stagnation. Also, when I mention a "blend of elitism and rank", to be more specific: I allow for the iteration best ant to add additional pheromones above what corresponds to his rank (i.e., he is the elite ant) – this appeared to prevent the pheromone matrix from being too *uber_ant-centric*, and therefore the colony is more explorative, and exploration really is key. As a sidenote, it's worth mentioning that distinct parameters have large impacts on different city sets.

1. Nilsson, Christian. (2003). Heuristics for the Traveling Salesman Problem.
2. Dorigo and Stützle. (2004). MIT Press. Ant Colony Optimisation.
3. Stützle, Thomas & Hoos, Holger. (1999). MAX-MIN ant system. 16.