# Northern Michigan University (Marquette Co, MI)

CS470-01-26W: Artificial Intelligence (Andrew A. Poe) Name: _____
Quiz 5 Wednesday 11 March 2026 9:00 A.M. EST

Time: 15 minutes

The Traveling Salesman Problem is a well-known NP-Hard problem. You have a list of *n* cities, and an $n \times n$ lookup table containing the travel distances between all pairs of cities. (And it's symmetric; the distance from A to B is always the same as the distance from B to A.) There is always a path between any two cities with a measurable distance.

The idea is that a traveling salesman needs to travel to each city, visiting each city exactly once, returning to their starting point. And the salesman wants to choose an itinerary that minimizes the total distance traveled. This problem is very hard; essentially, the only known way to find the shortest path involves trying every possible ordering of cities, an algorithm too slow to be practical.

We can think of a "parent" in a genetic algorithm for this problem as being a valid permutation of the cities. For example, if there are six cities, one member of the population might be [1,6,5,3,2,4]. Another might be [1,5,6,2,3,4]. It's easy to compare two members of the population to work out which is better (compute the total distance from 1 to 6 to 5 to 3 to 2 to 4 to 1, for example).

What would be a good way to combine two members of the population (parents) to make a child? The child has to be a valid member of the population, which is to say, it must be a valid permutation of the cities (each city appears once and only once). The child should have attributes of both parents and there should be a random element involved in the computation so that if the same two parents have another child, the two children would likely be different from each other.

**There are many ways to do it. I would choose n/2 random positions in one parent (suppose I picked positions 2, 4, and 5, and suppose the entries they contained were 1, 4, and 3. I would then locate 1, 4, and 3 in the second parent and make sure that they were in that order. So, if the second parent was 1,2,3,4,5,6, I would change that to 1,2,4,3,5,6 so that 1,4,3 is a subsequence. This so altered second parent would be the child.**

There also needs to be a chance that the child will experience a (slight) mutation. What would be a reasonable way to introduce a mutation to a child, again understanding that the child, after the mutation, still has to be a valid permutation of the cities?

**If a mutation is to be done, just finding two random positions and exchanging their values would work just fine.**