Ashar Nadeem - an3056
Matthew Goodman - mdg2198

TSK - TSP on K-Means

Introduction

K-Means clustering is an NP-hard problem where nodes are partitioned into clusters with the mean of the cluster serving as its prototype. Traveling Salesman is an NP-hard problem where the shortest path between nodes is found under the constraint that each node is visited at least once. These problems can be combined together to create an algorithm that creates efficient paths within a list of inputs by clustering them together by proximity and then finding the optimal path within each cluster.

Proposal

The standard brute force sequential implementation of a TSP algorithm is as follows:

Input: $city\_list$, $cost\_matrix$
$optimal\_cost \leftarrow \infty$
$optimal\_path \leftarrow null$
$city\_list \leftarrow list\ of\ cities\ excluding\ city_1$
while $next\_permutation(city\_list)$ **do**
  $temp\_cost \leftarrow get\_path\_cost(city\_list, cost\_matrix)$
  {Cost of travelling cities in order of cities in $city\_list$}
  if $temp\_cost < optimal\_cost$ **then**
    $optimal\_cost \leftarrow temp\_cost$
    $optimal\_path \leftarrow city\_list$ {with $city_1$ appended at start & end}
  end if
end while
Output: $optimal\_path$, $optimal\_cost$

A serial implementation of the proposed algorithm takes a list of coordinates c, and an integer number of paths n, and applying k-means clustering to create an n number of clusters. This groups together coordinates via proximity and creates smaller chunks of nodes on which optimal paths can be found. Traveling salesmen is then run on every cluster, creating n number of optimal paths.

The parallel implementation applies the standard sequential TSP on smaller subsets of the clusters c to generate start and end points with values representing the lowest found cost for the sub-paths. Higher cost subpaths are pruned away. Subpaths are then connected to find possible traversals to form a completed path with the shortest possible distance.

Outcome

The intended outcome is to implement a pathsfinding algorithm using a parallel, clustered implementation of TSP that performs faster than the standard sequential implementation, which runs in exponential time.