1 March 2015

Jakub Ciecierski
Bartlomiej Dybisz
Lukasz Wojcik

Universal Computational Cluster

Start up documentation

1 Methodology and technology

We choose Agile development to provide a regular and iterative work flow.

For technology we have chosen .NET C#, a powerful tool which we would like to explore even further.

2 Schedule of tasks

- **Stage 1: Communication**

  Deadline 31.03
  Progress meetings (within the team) 16.03, 23.03

  - **Stage 1.1: Internal Deadline: 16.03**

    1. **Communication Server. (Lukasz)**
       - Registering: computational node, task manager.
       - Updating states of nodes (keeping alive / removal).

       Time Required: 8h
       Actual Time Spent:

    2. **Computational node, Task manager, Client. (Karol)**
       - Scheme of classes.
       - Client input mechanism.

       Time Required: 8h
       Actual Time Spent:

    3. **Communication protocol. (Jakub)**
       - Appropriate API.
       - Classes of messages and problems .

       Time Required: 8h
       Actual Time Spent:

    4. **Network. (Bartek)**
       - TCP/IP connection.
       - Listening to given port.
       - Establishing connection.

       Time Required: 8h
       Actual Time Spent:

    5. **Testing. (All)**
       Time Required: 2h
       Actual Time Spent:

– Stage 1.2: Internal Deadline: 23.03

1. **Communication Server. (Lukasz)**
   - Queueing of Messages.
   - Receiving problems from CC.
   - Sending problems to TM, CN.
   - Receiving solutions from TM, CN .

   Time Required: 12h
   Actual Time Spent:

2. **Computational node, Task manager, Client. (Karol)**
   - Registering to server.
   - Sending problems/solutions to server.

   Time Required: 8h
   Actual Time Spent:

3. **Communication protocol. (Jakub)**
   - Handling messages. Creating Appropriate classes

   Time Required: 8h
   Actual Time Spent:

4. **Network. (Bartek)**
   - Failures, Closure

   Time Required: 8h
   Actual Time Spent:

5. **Testing. (All)**
   Time Required: 2h
   Actual Time Spent:

– **Stage 1.3: Internal Deadline: 30.03**

1. **Communication Server. (Lukasz)**
   - Any additional tasks

   Time Required:
   Actual Time Spent:

2. **Computational node, Task manager, Client. (Karol)**
   - Any additional tasks

   Time Required:
   Actual Time Spent:

3. **Communication protocol. (Jakub)**
   - Any additional tasks

   Time Required:
   Actual Time Spent:

4. **Network. (Bartek)**
   - Any additional tasks

   Time Required:
   Actual Time Spent:

5. **Testing. (All)**
   Time Required: 2h
   Actual Time Spent:

Update 31.03 Communication stage completed.

– All problems have been fixed

- **Stage 2: Algorithm**

  Deadline 28.04
  Progress meetings (within the team) 12.04, 25.03

  Update 02.06 Algorithm stage completed, although not in time.

  - A team member has been kicked out due to inactivity.
  - Missed a deadline since the team got smaller.

- **Stage 3: Testing All functionalities**

  Deadline 02.06
  Progress meetings (within the team)

  Update 02.06 Final stage completed.

- Backup servers completed.

3 Algorithm

The algorithm uses cluster analysis to predict number of vehicles that should be used in the optimal solution. In other words, the problem of Multiple TSP is split into few single TSP problems. The basic flow of the algorithm looks as follows

- Divide Combine location coordinates (e.g. (x,y) 2D) and time(z) into 3d (x,y,z) array Compute Cluster evaluation e.g. prediction strength. Each of k clusters is assumed to be one different vehicle. If there are more clusters than vehicles, take $k = num\_vehicles$
- Solve Now mTSP has been partitioned into k TSP problems. K TSP's are computed.
- Merge Combine costs

3.1 Divide

Algorithms used in the divide part.

3.1.1 K-means

Most common algorithm for constructing clusters

1. Randomly select k centers

2. Group points to their closest centroids
3. Update centroids by computing mean
4. Repeat 2. and 3. untill convergence

3.1.2 Prediction Strength

1. Split training data set into: Learning and Test sets.
2. Compute k-means for both sets.
3. Computes the co-membership of each Test cluster in respect to Learnings clusters. **Definition: co-membership** Let data be a set of n elements, then co-membership is an n by n matrix with (i,j)th element equal to 1 if i and j fall into the same cluster from the clusters set, 0 otherwise. The clusters and data can come from different samples (of the same population)
4. Compute strength of each Test cluster and find the minimum value.

3.2 Solve

Travelling Salesman Problem with Triangle Inequality.

1. Extract locations points from provided benchmark.
2. Convert [1]'s points into undirect graph with wages based on distances.
3. Calculate Prim's Algorithm with root set to a car's starting position.
4. Recursively traverse minimum spanning tree from [3] in PRE-ORDER fashion.
5. Create hamiltonian cycle by connecting first and last element from [4]'s sequence.
6. Pack [5] and its length (in terms of graph's wages).
7. Return [6].

3.3 Merge

The final solution is a union of partial solutions.