Improve Function Class based on Micro

1. Why to improve function class?

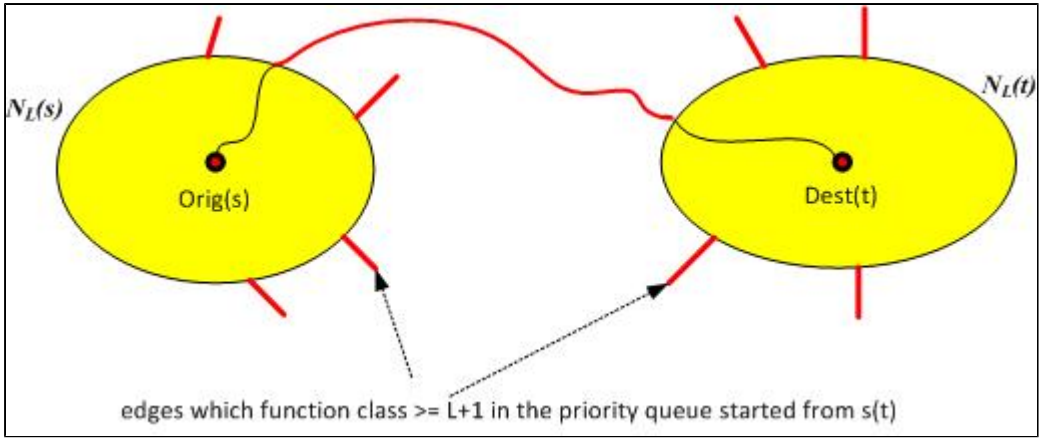
1. Micro's routing quality depends on function class quality a lot.
2. NA's function class quality is pretty good, but we can still find some poor routing quality cases due to wrong function class attribute.
3. CN and Brazil's function class quality is not as good as NA's, so in order to guarantee the route quality, improving the function class is necessary.
4. ...

2. The basic idea

Route quality depends on both route algorithm and data. The design assume route algorithm is fixed (including routing parameter is fixed), it try to improve route quality by improving function class attributes.

2.1. Micro route algorithm introduction

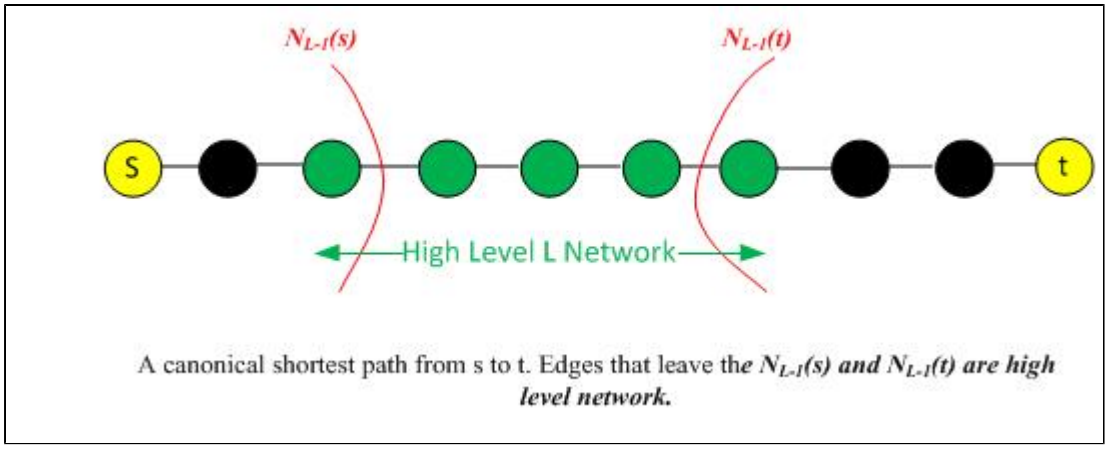
- Micro route algorithm
  It is bi-directional A* level promotion algorithm
- Level promotion strategy
  $\boldsymbol{N_L(s)}$ denote level L neighborhood of node s.
  For a given origin (destination) s, once $\boldsymbol{N_L(s)}$ has been explored, the search algorithm will be promoted to next level: L+1, which means only the roads which function class L+1 will be explored after promotion.
- $\boldsymbol{N_L(s)}$ definition
  During a DIJKSTRA/A* search from a given node s, all nodes are settled in a fixed order, $\boldsymbol{N_L(s)}$ is the relaxed edges set when the condition $C_L(s)$ H is first met in the search process.
  - H is the promote parameter, H is 40 for current algorithm
  - C_L (s) denotes the count of edges which function class L+1 in the priority queue started from s.
- Illustration



2.2. High level network

The basic idea comes from HH algorithm.

High level network, for a given parameter H, the high level L network $G_L = (V_L, E_L)$ of a graph G is defined by the set of E_L of edges: an edge(u,v)E belongs to E_L iff there are nodes s, t V such that the edge (u,v) appears in the canonical shortest path (s, …,u,v,…,t) from s to t with the property that v $N_{L-1}(s)$ and u $N_{L-1}(t)$.



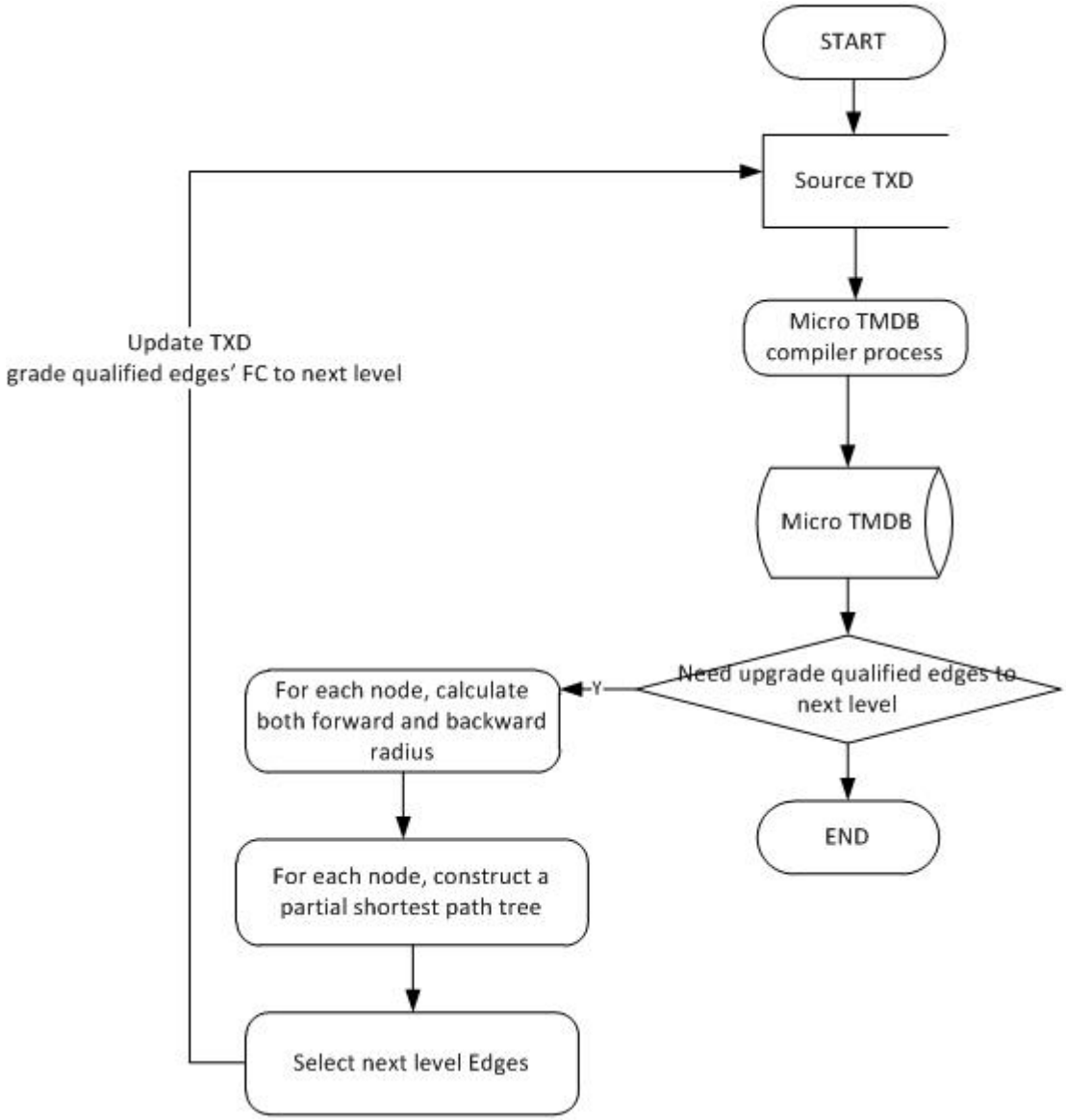
A canonical shortest path from s to t. Edges that leave the $N_{L-1}(s)$ and $N_{L-1}(t)$ are high level network.

2.3. Edges required to improve FC

The most challenge is how to find out the edges required for FC improving.

Edges required to improve FC = {e | (e E_L) and e's FC < L}

3. Preprocess for FC upgrading



4. The Design

4.1. Calculation of radius

4.1.1. API

*/** To calculate node N's radius*
***N** [in]: the input node for radius calculation*

***H** [in]: In micro, it is level promote parameter, the higher the H number, the larger the radius*

***Level** [in]: indicate which level's radius is supposed to calculate*

***nRadius** [out]: indicate forward and backward radius*

**/*

```
void CalculateRadius (Node N, int H, int level, int nRadius[2]);
```

4.1.2. Algorithm

TODO

4.2. Construction of a partial shortest path tree

4.2.1. API

*/** To calculate node N's radius*

***N** [in]: The input node for partial shortest path tree construction*

***nodeRadiusMap** [in]: node radius map*

Return Partial shortest path tree

**/*

```
PartialTree  ConstructPartialShortestPathTree (Node N,Map nodeRadiusMap);
```

4.2.2. Algorithm

TODO

4.3. Selection of the Level L Edges

4.3.1. API

*/** High level edge selection*

***tree**[in]: The input node's partial shortest path treen*

***nodeRadiusMap** [in]: node radius map*

Return Partial shortest path tree

**/*

```
vector SelectHighLevelEdges (PartialTree tree,Map nodeRadiusMap);
```

4.3.2. Algorithm

4.4. Distribution system (Map Reduce/Hadoop)

Distribute different tasks (node radius calculation, node partial shortest path tree construction to different machine, so the pre-process can be sped up.

5. Speeding up pre-process

1. Distribute the calculation tasks to multiple machine

Most time consumption tasks are:

a) For each node, calculate both forward and backward radius based on Micro API

b) For each node, construct a partial shortest path tree

These tasks can be easy to distribute to different machine to be finished independently.

1. Multiple-thread
2. Contraction of the road network

Reduce node number, TODO.