A Speedup Exact Trajectory Simplification by applying Quantum Minimum Searching Algorithm

1st Ratthamontree Burimas
School of Information, Computer and Communication Technology
Sirindhorn International Institute of Technology
Pathum Thani, Thailand
bank23232525@gmail.com

2nd Teerayut Horanont
School of Information, Computer and Communication Technology
Sirindhorn International Institute of Technology
Pathum Thani, Thailand
teerayut@siit.tu.ac.th

***Abstract*—Trajectory data, locational data of moving objects, is logged extremely frequent from Global Positioning System (GPS) devices. For further applications i.e., visualizing and analzing, Tha data is needed to be simplified. Trajectory simplification distorts the data since it removes some unnecessary points out. An occured distortion is an error measured by local integral square synchronous Euclidean distance (LSSED). The LSSED collects distance of the removed points to the corresponding generated point from the simplified data. In short, it says how much the simplified data is shaped differently from the original one. Comparing all simplified solutions is introduced as the only way to obtain the lowest error solution. However, in classical computing, it consumes too much time due to an effort of trying to remove and retain all points. Hence, lower consuming rate algorithms are introduced to solve the problem, but still achieving only approximate solution. With the advent of quantum computing, many quantum algorithms are designed for speeding up classical jobs i.e., searching and factorizing. By exploiting quantum specialities, their speed outperform classical algorithms with the same solution. This paper proposes trajectory simplification algorithms by using quantum algorithms. It works by comparing all simplified solutions as the classics, besides the job is done by the algorithm for finding the location of the minimum value in an unsorted database, Quantum Maximum or Minimum Searching Algorithm. Database mapping part is replaced with the computation of LSSED. It results that this integration allows to find the exact lowest-error solution with speedup.**

***Index Terms*—trajectory data, trajectory simplification, quantum algorithm, local integral square synchronous Euclidean distance, Quantum Maximum or Minimum Searching Algorithm**

I. Introduction

Data is being kept frequently everyday in every field of technologies. Location is the data that is usually collected for many purposes, so that trajectory data becomes one of the essential data that is in use in most of fields.

Trajectory data is a sequence of point dataset which mostly contain 3 attributes those are lattitude, longitude and timestamp. Each point in trajectory data is usually produced from Global Positioning System (GPS) devices. The frequency of that process is depended on their criteria and capacity of the devices. Nowadays, those GPS devices are capable to log each point in every 1 to 5 seconds. Unaviodably, the data can easily consume too much capacity unnecessarily. So, the compression is needed to make the data applicable, and one way to achieve is trajectory simplification.

Trajectory simplification is one of the way to compress trajectory data. As the trajectory data is a sequence of points, the trajectory simplification is the process of removing some points which can affect a distortion to the data as less as possible. The distortion created is considered as an error. Local integral square synchronous Euclidean distance (LSSED) is a decent error criteria to be used for measuring a quality of the simplification because it collects distance between points at the same timestamp, one is the removed points and one is the estimated points from the simplified data.

The simplification in this paper is divided into 2 types. Fixed-# is a simplification that can be determined an amount of points to be retained and returns the simplified trajectory data that contains the desired amount of points with lowest LSSED and Fixed-LSSED is a simplification that can be determined a threshold of LSSED to be produced and returns the simplified trajectory data that contains lowest number of points with LSSED that does not exceed the desired threshold.

Both types can be considered as a combinatorial problem. The computation time of finding the finest solution is $O(2^N)$ as computing all possible solutions and finding the most satisfactory one to the requirement. That is called "exact solution". To consume lower computation time with the result close to the finest one, we go for "approximate solution"

This paper proposes algorithms for approximate solution of both Fixed-# and Fixed-LSSED simplifications. By exploiting Quantum Minimum Searching Algorithm (QMSA) [1], we are able to find the best solution by looking for the solution that can minimize an error metric. This allows us to find the finest solution while comsuming lower computation time by bringing specialities of quantum computing.

II. Background

A. Trajectory Data

Trajectory data is a locational data of a single object based on sequence of time. Normally, trajectory data contains trajectory points and each point has 3 attributes which are latitude, longitude and timestamp.

$$P = [p_1 ... p_n], \tag{1}$$

where P is a trajectory data contains trajectory points p_is.

$$p_i = (\text{latitude}, \text{longitude}, \text{timestamp}) \tag{2}$$

Fig. 1 shows a trajectory data contains 10 trajectory points $p_1 = (3, 1, 1)$ $p_2 = (3, 2, 2)$ $p_3 = (3, 3, 3)$ $p_4 = (2, 4, 4)$ $p_5 = (1, 5, 5)$ $p_6 = (1, 6, 6)$ $p_7 = (1, 7, 7)$ $p_8 = (1, 8, 8)$ $p_9 = (1, 9, 9)$ $p_{10} = (1, 10, 10)$.

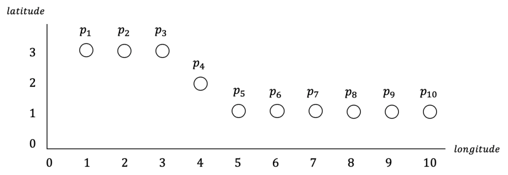

Fig. 1. An example of a trajectory data.

B. Trajectory Simplification

Trajectory simplification is a commonly used method for compressing trajectory data. By discarding some trajectory points, the distortion is produced unavoidably but also be kept as low as possible in each criteria. There are many algorithms proposed already in classical computing i.e., Top-down Time-ratio (TDTR) algorithm [7] that is a decent one. The algorithm is processed repetitively by selecting the furthest one point from the a single line from first and last point until a distance reaching specific threshold. The TDTR consumes time complexity of $O(N^2)$.

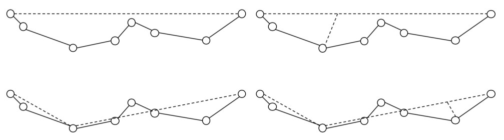

Fig. 2. An example of performing TDTR.

C. Error Metric

As mentioned, the distortion is unavoidable when performing simplification. A position error measurement in trajectory is represented with local integral square synchronous Euclidean distance (LSSED) that can be calcualted by synchronous Euclidean distance (SED).

1) Synchronous Euclidean Distance: Synchronous Euclidean distance is a measurement of the distortion discarded point created. The SED is computed at discarded point only and is computable if there are points with previous timestamp and following timestamp existing. Those 2 points are to be computed to estimatedly find the point in between with the timestamp matched to the discarded point. The distance the found point and the discarded point is called SED.

Fig. 3 shows that p_i is discarded, so that we do not know exactly where is the location at timestamp t_i. But we can find the point with the corresponding timestamp by the previous point p_s and following point p_e. Then we found p'_i that is believed to be where p_i is. The distance between them represented as frequent-dashed line is SED. The coordinate of the point on the retained line with timestamp corresponding to the discarded point can be computed by

$$p_i \text{ is located at } (lat_i, lon_i, ts_i) \tag{3}$$

,

$$p'_i \text{ is located at } (lat'_i, lon'_i, ts_i) \tag{4}$$

,

$$p_s \text{ is located at } (lat_s, lon_s, ts_s) \tag{5}$$

and

$$p_e \text{ is located at } (lat_e, lon_e, ts_e), \tag{6}$$

where

$$lat'_i = lat_s + \frac{ts_i - ts_s}{ts_e - ts_s} \cdot (lat_e - lat_s) \tag{7}$$

and

$$lon'_i = lon_s + \frac{ts_i - ts_s}{ts_e - ts_s} \cdot (lon_e - lon_s) \tag{8}$$

SED is the distance from p_i to p'_i that is generated from p_s and p_e, and can be computed by

$$SED(p_s, p_i, p_e) = dist(p_i, p'_i) = \sqrt{(lat_i - lat'_i)^2 + (lon_i - lon'_i)^2} \tag{9}$$

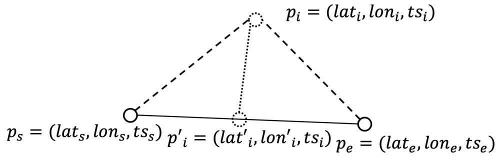
$p_i = (lat_i, lon_i, ts_i)$
$p_s = (lat_s, lon_s, ts_s)$
$p'_i = (lat'_i, lon'_i, ts_i)$
$p_e = (lat_e, lon_e, ts_e)$

Fig. 3. An example of SED calculation.

2) LSSED: Local integral square synchronous Euclidean distance is a summation of all squared SED along the trajectory data T to the trajectory data T'.

$$LSSED(T, T') = \sum_{i=2}^{N-1} SED(p_{s,i}, p_i, p_{e,i})^2, \text{ if } p_i \text{ is not in } T' \tag{10}$$

where $p_{s,i}$ is the previous closest retained point to p_i in T' and $p_{e,i}$ is the following closest retained point to p_i in T'. Note that if p_i is a retained point as it is in both T and T', $SED(p_{s,i}, p_i, p_{e,i})$ equals to 0 and the SED at p_1 and p_N are ignored since they are fixed-retained.

Fig. 4 shows that the dashed-line points are discarded, so that SED can be computed from each of them. The solid line drawn through the retained point is used to find points with timestamp corresponding to those discard point. By summing up all squared SED in frequent-dashed line, it returns LSSED value. Apparently, our goals are to find T' that result the lowest LSSED for a criteria of Fixed-# and T' that result LSSED not exceeding a threshold for a criteria of Fixed-LSSED.

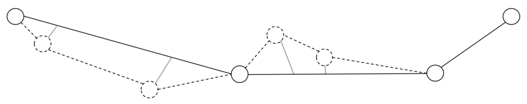

Fig. 4. An example of LSSED calculation.

D. LSSEDSTATE function

LSSEDSTATE function is a special function proposed in this paper to be used in simplifications. LSSEDSTATE is a summation of squared SED as LSSED but LSSEDSTATE takes binary state as an input instead of simplified trajectory T'.

$$LSSEDSTATE(T, y) = \sum_{i=2}^{N-1} SED(p_{s,i}, p_i, p_{e,i})^2, \text{ if } y_i = 0 \tag{11}$$

where y is a binary state with length N. $y_i = 0$ represents trajectory point at index i is discarded and $y_i = 1$ represents trajectory point at index i is retained. p'_i is computed by the closest previous retained point p_s and the closest following retained point p_e as in Eq. 4. $SED(p_{s,i}, p_i, p_{e,i})$ is only computable where $y_i = 0$ which means p_i is a discarded point. So, $SED(p_{s,i}, p_i, p_{e,i})$ is equal to 0 if $y_i = 1$ since p_i and p'_i are the same point. Note that y_1 and y_N are ignored because they are always equal to 1 (fixed-set).

E. Quantum Minimum Searching Algorithm

Quantum Minimum Searching Algorithm (QMSA), originally called Quantum Maximum or Minimum Searching Algorithm in [1], is an algorithm for finding minimum or maximum value in an unsorted database. In this paper, we are only focusing on finding the minimum part as we defined our goals are to minimize LSSED and an amount of retained points.

Classically, finding minimum value in an unsorted database with length N unavoidably costs complexity of $O(N)$ because we need to explore all data before deciding which one is the minimum amongst them. But QMSA, by exploiting quantum computing specialities, can do the same thing with complexity of approximately $O(\frac{\pi}{2}\sqrt{\frac{N}{M}})$ where M is the number of the minimum value.

Originally, The QMSA is used to find minimum value in an unsorted database by processing a set of bits with quantum state, qubits. Let z be the set of qubits that can be computed in quatum manners. z represented index of value in database. Apparently, the length of the database ought to be a power of two. The QMSA adjust z to be very close to the exact answer, then measure quantum state of z. An output of measuring qubits z is a normal binary bits with the same length. Let x be a set of those bits. x is believed to be the index of the minimum value in the database.

In short, The QMSA can find index of value that is a minimum amongst others under a certain condition i.e., mapping in a database. In this paper, the QMSA is applied to find solution of trajectory simplification that can returns the most satisfying result under both Fixed-# and Fixed-LSSED criterias.

Recall that x is a set of bits and the QMSA is to find x' that can results the smallest $LSSEDSTATE(T, x')$ amongst others x.

The QMSA performs by followings:
1.Randomly select index d_0 with some x.
2.Declare d_1 with ∞.
3.Select variable c with some integer.
4.Mark all binary states those are indices of values less than or equal to a value at index d_0 to be 1 and other to be 0, then store to an oracle function F.
5.Create quantum initial state z with the same length as x.
6.Apply Grover-Long algorithm to map F to the initial state and store output to z'.
7.Measure those z' and reassign to d_1.
8.If a value at index d_1 is greater than at d_0, and go back to Step 5.
9.If a value at index d_1 is smaller than at d_0, and go back to Step 4, then loop with c iteration.
11.If d_1 is equal to d_0, return d_0 as a considerably most satisfying x (x'), and break the iteration.
12.Reassign d_0 with d_1, and go back to Step 4

F. Grover-Long algorithm

Grover-Long algorithm [2] is a very well-known algorithm in a quantum computing field. It takes quantum state and an oracle function as inputs, and map the oracle function to the quantum state. the quantum state, initially has the same possibility of all binary outputs, is bended to have the highest measurable possibility at the binaries that is marked in the oracle function. Then, return a the processed quantum state as an output.

There are a lot of detail about the algorithm and most of them are described in quantum aspect. In short, after looking at a mapping of marked or unmarked for each state, this algorithm takes quantum initial states which has the same measurable possibility at all binary outputs and adjust possibilities of the unmarked states in the oracle function to be measured at 0% and the marked states in the oracle function to be measured at 100%.

III. Proposed Method

There are 2 main requirements in all types of compression which are compression ratio and error rate. Apparently, In our case, compression rate is fixed in Fixed-# and minimized in Fixed-LSSED and error metric is LSSED of a simplified trajectory.

A. State the problem

Taking an input as a trajectory data T that contains N trajectory points,

$$T = [t_1...t_n] \tag{12}$$

The output of the process is T' which contains M points, where $M \leq N$.

$$T' = [t'_1...t'_M] \tag{13}$$

The T' is considered as one with desired amount M of retained points resulting lowest LSSED as possible in Fixed-# or having LSSED lower than desired threhold with lowest amount of points as possible in Fixed-LSSED.

B. Formulate the problem

Consider trajectory data T with size N. Trajectory points in T at index 1 and N are obviously unremovable because SED as they are removed is not computable. So, those 2 are fixed as retained and only points in the middle from index 2 to $N-1$ will be further decided to be retained or discarded, so that we mark points at index 1 and N as set and let x be a serie of N digit binaries to represent selection of each point from index 1 to N, where bit 1 means that point is retained and bit 0 means that point is discarded. Let x_i represents status of point in T at index i.

$$x = x_1x_2...x_N \tag{14}$$

If $x_i = 1$ (set x_i), point in T at index i is retained. If $x_i = 0$ (unset x_i) , point in T at index i is discarded. Note that $x_1 = 1$ and $x_N = 1$. T' can be restated as

$$T' = [t'_1...t'_m], \tag{15}$$

where

$$t'_1 = t_1, t'_m = t_n \tag{16}$$

and

$$\text{for } k = 2 \text{ to } N-1 : \text{if } x_k = 1, T' \leftarrow T' \cup t_k \tag{17}$$

The T''s LSSED can be computed by :

$$T'_{LSSED} = LSSED(T, T') \tag{18}$$

where $LSSED$ is a function measuring error trajectory data T' to a trajectory data T by summing up squared SED created from the points T' discarded.

C. Solve the problem

Let T_1 contains N trajectory points. That causes x to contain N binaries.

First, we cast x:

$$x = x_1x_2...x_N \tag{19}$$

,

$$x_1, x_N = 1 \tag{20}$$

and

$$x_2...x_{N-1} \in \{0,1\} \tag{21}$$

Which means that if $x_i = 1$, point in T_1 at index i is retained, and if $x_i = 0$, point in T_1 at index i is discarded. Note that $x_1 = 1$ and $x_N = 1$.

D. Solve the problem: Quantum Part

The QSMA [1] play a crucial role from this point. Since we know that $x_1, x_N = 1$, so that we can economize cost of 2 qubits by taking only $x_2...x_{N-1}$ to be processed.

Let z be a set of qubits representing $x_2...x_{N-1}$

$$z = z_1z_2...z_{N-2}, \tag{22}$$

where z_i is for x_{i+1}.

At this stage, we can talk about our 2 goals according to 2 simplifications this paper proposes and the QSMA is used to find the most satisfying x' for each.

a) Fixed-#: Let w be a desired number of retained points. Recall that 2 x_is are fixed-set bit, so that $w \geq 2$. The QSMA is to find x' that can result lowest LSSED where number of set bits in x' is w.

The QSMA performs to achieve Fixed-# by followings:
1.Randomly select d_0 with some x state that satisfies w set bits i.e., $d_{0,2}...d_{0,2+(w-2)-1}$ to be set and $d_{0,2+(w-2)}...d_{0,n-1}$ to be unset. $d_{0,1}$ and $d_{0,n}$ are fixed-set so we leave 2 set bits to them.
2.Declare d_1 with ∞.
3.Select variable c with some positive integer.
4.Create an oracle function F that takes x as input and returns 1 if that $LSSEDSTATE(T, x)$ is less than or equal to $LSSEDSTATE(T, d_0)$ and satisfy w set bits, and 0 otherwise.
Since 2 bits of x are fixed, the possible state of xs is 2^{N-2}.
5.Perform the function in Step 4 and store it to F.
6.Initialize quantum state z with $N-2$ qubits which can represent value of all xs.
7.Apply Grover-Long algorithm to map the oracle function F to the quantum initial state z. This bends z to be measured at a state marked as 1 by the oracle funtion F.
8.Measure quantum state z and map back to x' will result $x'_2...x'_{N-1}$.
9.Assemble that $x'_2...x'_{N-1}$ with the fix-set x_1,x_N to create $x' = [x_1, x'_2...x'_{N-1}, x_N]$ and store to d_1 .
10.If $LSSEDSTATE(T, d_1)$ is greater than $LSSEDSTATE(T, d_0)$, it means this d_1 is a mistake and we go back to Step 5.

11.If $LSSEDSTATE(T, d_1)$ is less than $LSSEDSTATE(T, d_0)$, it means this d_1 is a progress, and we assign d_1 to d_0 and go back to Step 4 for c iterations.
12.If $LSSEDSTATE(T, d_1)$ is equal to $LSSEDSTATE(T, d_0)$, it means this d_1 is considerably the one that has minimum LSSED. We break the iteration and return d_0 as an output.

The pseudocode of the algorithm is shown in Algor. 1.

Algorithm 1: Fixed-#

```
Result: $x'$
Let $w$ be a desired number of retained points;
Let $T$ be a trajectoty data with length $N$;
Let $d_0$ be an empty binary bits with length $N$;
Set the first and last bits of $d_0$ with 1;
Set $d_{0,2}...d_{0,2+(w-2)-1}$ bits with 1;
Set $d_{0,2+(w-2)}...d_{0,n-1}$ bits with 0;
Let $c$ be some positive integer;
Let $d_1 = \infty$;
for $i = 1 \to c$ do
    Create an oracle function $F$ that return 1 for all
      solutions that have $w$ set bits and result LSSED
      less than $d_0$ does, 0 otherwise;
    while $LSSEDSTATE(T, d_1) >$
      $LSSEDSTATE(T, d_0)$ or $d_1 = \infty$ do
        Initialize quantum state $z$ with length $N - 2$
          for $d_{0,2}...d_{0,N-1}$;
        Apply Grover-Long algoritm to map $F$ to the
          quantum initial state $z$;
        Measue the quantum state $z$ and store to
          $x'_2...x'_{N-1}$;
        Assemble $x'_2...x'_{N-1}$ with $x'_1 = 1$ and $x'_N = 1$
          to create $x'$ and store to $d_1$;
    end
    if $LSSEDSTATE(T, d_1) <$
      $LSSEDSTATE(T, d_0)$ then
        Reset $i$ to 1;
    end
    else if $LSSEDSTATE(T, d_1) =$
      $LSSEDSTATE(T, d_0)$ then
        Return $d_0$;
    Set $d_0$ with $d_1$;
    Set $d_1$ with $\infty$;
end
Return $d_0$;
```

b) Fixed-LSSED: Let b be a LSSED threshold. The QSMA is to find x that can result LSSED not exceeding b where number of set x_i is minimized.

The QSMA performs to achieve Fixed-LSSED by followings:
1.Randomly select d_0 with some x state that satisfies a LSSED threshold b i.e., all $d_{0,i}$ to be set bits which will make $LSSEDSTATE(T, d_0) = 0$ which surely satisfies b.
2.Declare d_1 with 0.
3.Select variable c with some positive integer.
4.Create an oracle function F that takes x as input and returns 1 if that $LSSEDSTATE(T, x)$ is less than or equal to b and that x contains set bits less than or equal to d_0, and 0 otherwise.
Since 2 bits of x are fixed, the possible state of xs is 2^{N-2}.
5.Perform the function in Step 4 and store it to F.
6.Initialize quantum state z with $N - 2$ qubits which can represent value of all xs.
7.Apply Grover-Long algorithm to map the oracle function F to the quantum initial state z. This bends z to be measured at a state marked as 1 by the oracle funtion F.
8.Measure quantum state z and map back to x' will result $x'_2...x'_{N-1}$.
9.Assemble that $x'_2...x'_{N-1}$ with the fix-set x_1,x_N to create $x' = [x_1, x'_2...x'_{N-1}, x_N]$ and store to d_1 .
10.If d_1 contains set bit more than d_0 does, it means this d_1 is a mistake and we go back to Step 5.
11.If d_1 contains set bit less than d_0 does, it means this d_1 is a progress and we assign d_1 to d_0 and go back to Step 4 for c iterations.
12.If d_1 contains set bits as equal to d_0 does, it means this d_1 is considerably the one that has minimum set bit with LSSED not exceeding b. We break the loop and return d_0 as an output.

The pseudocode of the algorithm is shown in Algor. 2.

E. Obtain the answer

After performing those algorithm according to 2 simplications, we obtain d_0 that is considerably the finest solution x' that can satisfy Fixed-# or Fixed-LSSED the most amongst other x's.

$$x' = x_1 x'_2...x'_{N-1} x_N \tag{23}$$

IV. Evaluation

An evalution is achieved with quantum simulator provided by Huawei's HiQ Simulator written in Python 3.5. Code is available on Github[1]. An environment under the evaluation is Ubuntu debian 18.04 Operating system, 16-core CPU, 32-GB of ram computer. Trajectory data is from Microsoft's GeoLife GPS Trajectories, collected in (Microsoft Research Asia) Geolife project by 182 users in a period of over three years (from April 2007 to August 2012).

As parameters stated in the previous section, we demostrate trajectory simplification with 2 criterias for each simplification.

a) Fixed-# Evaluation: Let $w = 5$ and T be a trajectory data of length 10. QMSA is to find solution with minimum LSSED and keeping 5 points retained from 10 points. So, $x = [x_1...x_{10}]$, where $x_1 = 1$, $x_{10} = 1$ and $x_2...x_9$ is optimized by QMSA. Let $c = 5$ and 10. The result is shown in [Table. I] and [Table. II] respectively, where "#iteration" means a number of iterations, "#satisfying condition" means a number of time that

[1]https://github.com/rtmtree/quantum_trajectory_simplification.git

```
Algorithm 2: Fixed-LSSED
  Result: $x'$
  Let $b$ be an LSSED threshold;
  Let $T$ be a trajectoty data with length $N$;
  Let $d_0$ be an empty binary bits with length $N$;
  Set the all bits of $d_0$ with 1;
  Let $c$ be some positive integer;
  Let $d_1 = 0$;
  for $i = 1 \rightarrow c$ do
      Perform an oracle to mark all solutions that have
        LSSED not exceeding $b$ and contain set bits less
        than or equal to $d_0$ does, then store to array $F$;
      while (set bits of $d_1$ > set bits of $d_0$) or $d_1 = 0$ do
          Initialize quantum state $z$ with length $N - 2$
            for $d_{0,2}...d_{0,n-1}$;
          Map $F$ to that quantum initial state;
          Apply Grover-Long algoritm on the quantum
            state;
          Measue the quantum state $z$ and store to
            $x'_2...x'_{n-1}$;
          Construct $x'$ with set bit at index 1 and $n$ and
            store to $d_1$;
      end
      if set bits of $d_1$ < set bits of $d_0$ then
          Reset $i$ to 1;
      end
      else if set bits of $d_1$ = set bits of $d_0$ then
          Return $d_0$;
      Set $d_0$ with $d_1$;
      Set $d_1$ with 0;
  end
  Return $d_0$;
```

TABLE I
FIXED-# EVALUATION $c = 5$

Trajectory Data	#iteration	#satisfying condition	#finest solution
Geolife000	100	100	100
Geolife001	100	100	0
Geolife002	100	100	100
Geolife003	100	100	100

x' contains w retained points and "#finest solution" means a number of time that x' contains w retained points and result $LSSEDSTATE(T, x')$ lowest amongst other possible xs (an exact solution).

b) Fixed-LSSED Evaluation: Let $b = 0.00125$ and T be a trajectory data of length 10. QMSA is to find solution with LSSED not exceeding the threshold b while minimizing a number of retained points as low as possible. So, $x = [x_1...x_{10}]$, where $x_1 = 1$, $x_{10} = 1$ and $x_2...x_9$ is optimized by QMSA. Let $c = 5$ and 10. The result is shown in [Table. III] and [Table. IV] respectively, where "#iteration" means a number of iterations, "#satisfying condition" means a number of time that x' results $LSSEDSTATE(T, x')$ not exceeding b and "#finest solution" means a number of time that x' result $LSSEDSTATE(T, x')$ not exceeding b and contains lowest retained points amongst others possible xs (an exact solution).

TABLE II
FIXED-# EVALUATION $c = 10$

Trajectory Data	#iteration	#satisfying condition	#finest solution
Geolife000	100	100	100
Geolife001	100	100	100
Geolife002	100	100	100
Geolife003	100	100	100

TABLE III
FIXED-LSSED EVALUATION $c = 5$

Trajectory Data	#iteration	#satisfying condition	#finest solution
Geolife000	100	100	0
Geolife001	100	100	100
Geolife002	100	100	100
Geolife003	100	100	0

TABLE IV
FIXED-LSSED EVALUATION $c = 10$

Trajectory Data	#iteration	#satisfying condition	#finest solution
Geolife000	100	100	0
Geolife001	100	100	100
Geolife002	100	100	100
Geolife003	100	100	100

CONCLUSION AND FUTURE DEVELOPMENT

This paper proposes 2 algorithms for 2 simplification types, Fixed-# and Fixed-LSSED. Classically, to achievement the best satisfying result for both types, the time complexity of $O(2^{N-2})$ is needed, where N is a number of trajectory point. By exploiting quantum mechanics from the QSMA, we can do the same job by comsuming only the time complexity of $O(\frac{\pi}{2}\sqrt{\frac{2^{N-2}}{M}} \times c)$, where M is a number of finest solution and c is an adjustable parameter.

Due to the current lack of quantum property, we can not provide a high number of qubits to represent solutions for trajectory data with high number of points. When qubits are more available, this can simply apply to a bigger trajectory data. The 100-iteration evaluation result shows that the solutions satisfy conditions almost 100% and is improvable by increasing a value of c.

Since the lack of qubits does not affect the algorithms but shorten the length to represent the solutions only, this can apply for a bigger trajectory data if the solutions can enough map to the answer i.e., to fix size of desired retained points and map only those satisfying answer to smaller quantum qubit length. This can utilize more power for the algorithms. Apparently, The QSMA is not usable only for searching anymore, but can also be for other kinds of combinatory if it is adjusted appropriately. we suggest you to try exploring other possibilities by exploiting the QSMA or our algorithms.

Acknowledgment

We thank Sirindhorn International Institute of Technology for providing technical environment and supportive information.

References

[1] Y. Chen, S. Wei, X. Gao, C. Wang, J. Wu and H. Guo, “An Optimized Quantum Maximum or Minimum Searching Algorithm and its Circuits,” arXiv:1908.07943 [quant-ph] 21 Aug 2019.

[2] G. L. Long, “Grover algorithm with zero theoretical failure rate,” arXiv:quant-ph/0106071, 13 Jun 2001.

[3] A. Ahuja and S. Kapoor, “A Quantum Algorithm for finding the Maximum,” arXiv:quant-ph/9911082, 18 Nov 1999.

[4] C. Dürr and P. Høyer, “A quantum algorithm for finding the minimum,” arXiv:quant-ph/9911082, 18 Nov 1999.

[5] M. Chen, M. Xu and P. Fränti, “A Fast Multiresolution Polygonal Approximation Algorithm for GPS Trajectory Simplification,” IEEE Transactions on image processing, col. 21, No. 5, May 2012.

[6] D. Zhang, M. Ding, D. Yang, Y. Liu, J. Fan, H. T. Shen, “Trajectory Simplification: An Experimental Study and Quality Analysis,” Proceedings of the VLDB Endowment, vol. 11, No. 9, Rio de Janeiro, Brazil, August 2018.

[7] N. Meratnia and R. A. de By, “Spatiotemporal compression techniques for moving point objects,” In EDBT, pages 765–782, 2004.