An Exact Trajectory Simplification by applying Quantum Minimum Searching Algorithm  
\*Note: Sub-titles are not captured in Xplore and should not be used [[1]](#footnote-20)

Data grows every moment, a compression becomes vital to keep data as compact while maintaining error low. Trajectory data, a set of locational points of moving object, is logged from GPS devices at high frequency rate and to be processed further i.e., storing, vitualizing and analyzing. No one prefers to take a huge raw data as a burden in those jobs. So, trajectory data is needed to be compressed unavoidably. Trajectory simplification simply compresses trajectory data by removing some points out. By doing so, a simplified data is obviously distorted from the orginal one. The distortion can be measure by local integral square synchronous Euclidean distance (LSSED). the LSSED collects distance of the removed points to the corresponding points that the simplified data can generate. In short, it says how much the simplified data shape different from the original one.

The trajectory simplification become one of the hard combinatorial problem and is being challenged by many algorithms but those can only achieve approximate answers. To obtain an exact answer by demonstrating all simplified solutions and compare them, the complexity is as high as due to an effort of trying to remove and retain all points in length .

Today, one of state of the art in computation field is quantum computing. There many quantum algorithms that can do classical jobs i.e., searching, factorizing and etc. They can also find the same solutions while exploiting quantum specialities which makes them comsume lower complexity than classical algorithms. This paper proposes trajectory simplification algorithms that can result the exact answer with lower complexity by bringing quantum algorithms to be a main mechanism. It works with demonstrating all simplified solutions and compare them like the classics beside, the job is run by Quantum Maximum or Minimum Searching Algorithm (QUMMSA), the algorithm for finding the location of the minimum value in an unsorted database, we replace database mapping part with the computation of LSSED. And the result shows that this integration allow us to find the exact answer as the classics but comsume complexity of , where is a number of solutions with lowest value.

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

# Introduction

Data is being kept frequently everyday in every field of technologies. Location is the data that is usually collected for many purposes so, 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 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 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 process of removing some points out from the trajectory data will create some error. local integral square synchronous Euclidean distance (LSSED) is a decent error criteria to be used for measuring a quality of the simplification.

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 in mathamatics. The computation time of finding the finest solution is as compute all possible solutions and find the one with the most satisfactory 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) that is 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.

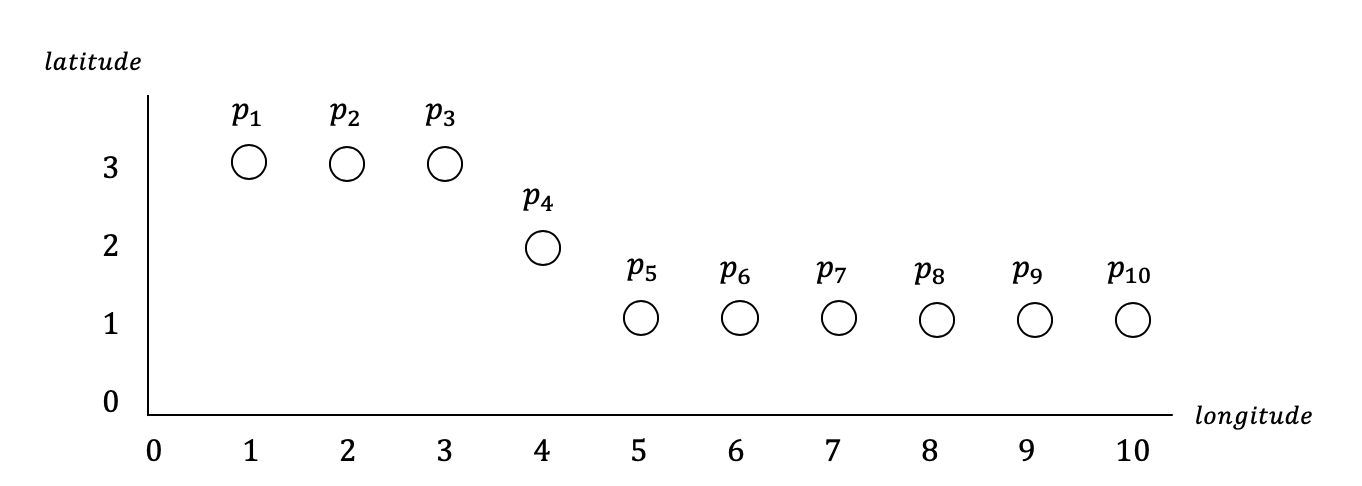
# Background

## 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.

where, is a trajectory data contains trajectory points

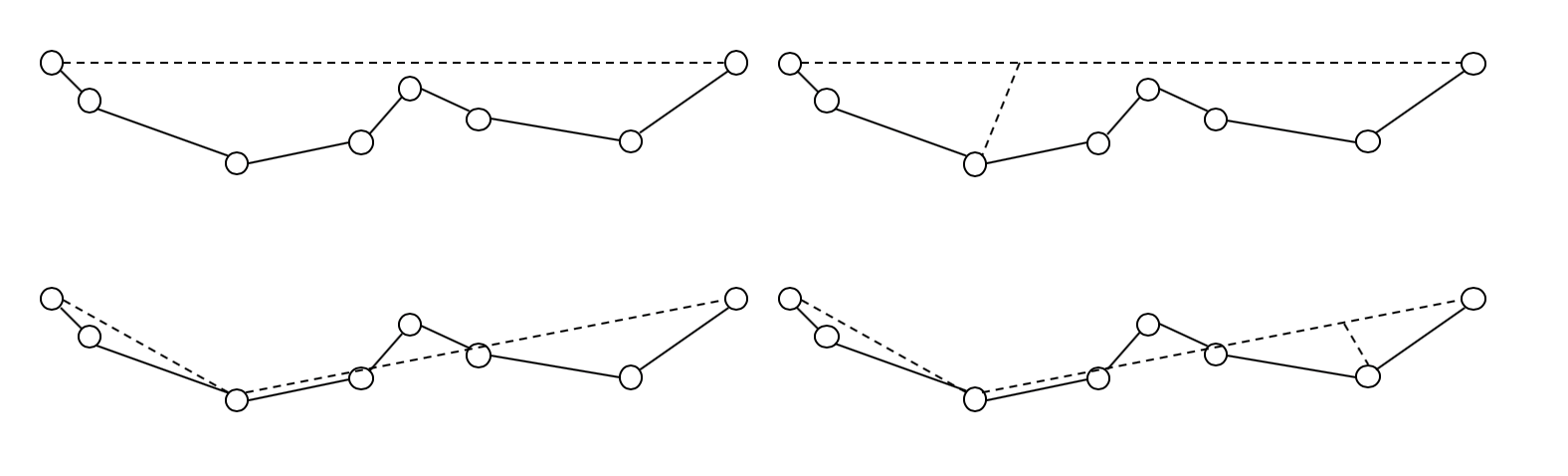
Fig. [1](#fig:trajectorydatafig) shows a trajectory data contains trajectory points  
 .



A example of a trajectory data.

## Trajectory Simplification

Trajectory simplification is a commonly used method for compressing trajectory data. By discarding some trajectory point, 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 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 reach specific threshold. The TDTR consumes of time complexity.



An xample of performing TDTR.

## Error Metric

As mentioned in Section 2B that distortion is unavoidable when performing simplification. The error measure ment in trajectory is represented with local integral square synchronous Euclidean distance (LSSED) that can be calcualted by synchronous Euclidean distance (SED).

### 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 points are to be computed to approximately 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](#fig:sed1) shows that is discarded so, we do not know exactly where is the location at timestamp . But we can find the point with the corresponding timestamp by the previous point and following point . Then we found that is believed to be where is but is not. 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,

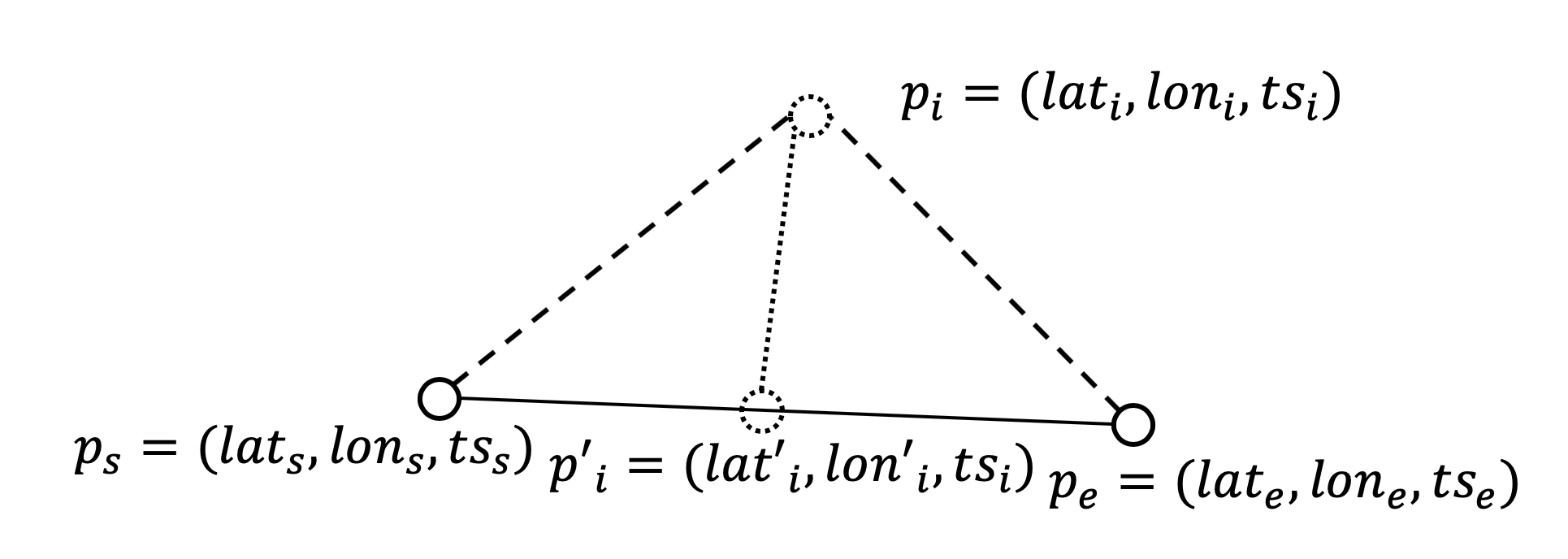
,

,

and

where

and SED is computed by distance from to generated from and is,



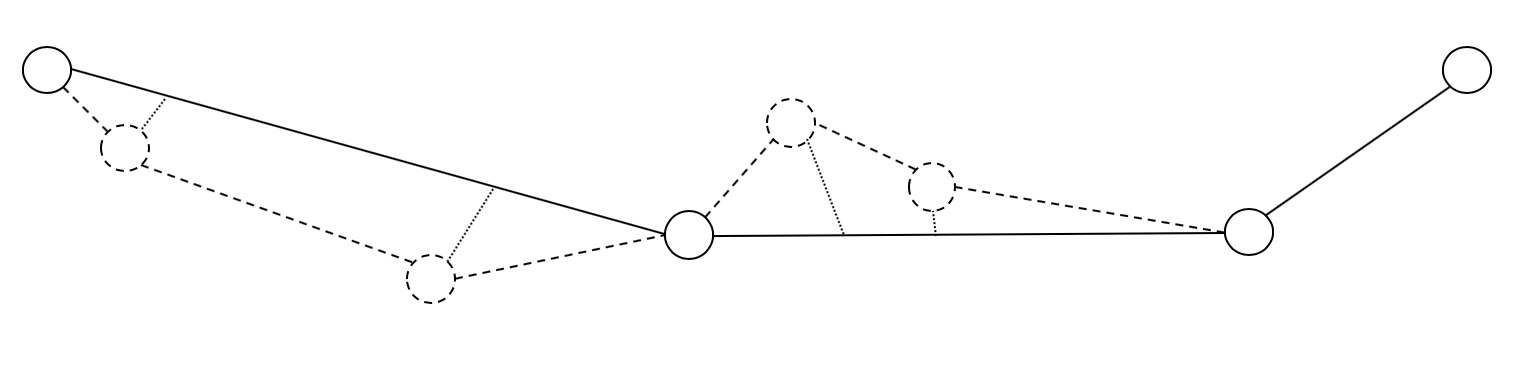
An example of SED calculation.

### LSSED

Local integral square synchronous Euclidean distance is a summation of all squared SED along the trajectory data to the trajectory data .

where is the previous closest retained point to in and is the following closest retained point to in . Note that if is a retained point as it is in both and , equals to and the SED at and do not need to compute since they are fixed-retained.

Fig. [4](#fig:LSSED1) shows that the dashed-line points are discarded so, 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 goal is to find that result the lowest LSSED with a criteria of Fixed-Ratio.



An example of LSSED calculation.

## LSSEDPAIR function

LSSEDPAIR function is a special function proposed in this paper to be used in SUMSED function. LSSEDPAIR is a summation of SED from one index to another like LSSED but LSSEDPAIR does blindly find all SEDs and sum them up.

where never equal to because LSSEDPAIR always see as all are discarded point.

## LSSEDSTATE function

LSSEDSTATE function is a special function proposed in this paper to be used in simplifications. LSSEDSTATE is a summation of SED like LSSED but LSSEDSTATE takes binary state as an input instead of simplified trajectory .

where is a binary state with length . represents trajectory point at index is discarded and represents trajectory point at index is retained. is computed by the closest previous retained point and the closest following retained point Eq. [[eq:pprime]](#eq:pprime). is only computable where which means is a discarded point. So is equal to if since and are the same point.

## Quantum Minimum Searching Algorithm

Quantum Minimum Searching Algorithm (QMSA), originally called Quantum Maximum or Minimum Searching Algorithm in , 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 goal is to minimize LSSED value.

Classically, finding minimum value in an unsorted database with length unavoidably costs complexity of 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 approximatedly where 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 qubits. Let be the set of qubits which are bits with quantum state that can be computed in quatum manners. The represented index of value in database. Apparently, the length of the database is needed to be a power of two. The QMSA adjust to be very close to the exact answer then, measure quantum state of . An output of measuring qubits is a normal binary bits with the same length. Let be a set of those bits. is believed to be the index of the minimum 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 condition.

Recall that is a set of binaries and the QMSA is to find that can results the smallest amongst others .

The QMSA performs by followings:  
1.Randomly select with some .  
2.Declare with .  
3.Select variable with an integer.  
4.Mark all that less than or equal to to be 1 with oracle function and store to a mapping set .  
5.Map all s to quantum initial state .  
6.Apply Grover-Long algorithm on the initial state and store output to .  
7.Measure those and reassign to .  
8.If is greater than , and go back to Step 5.  
9.If is smaller than , and go back to Step 4 loop with iteration.  
11.If is equal to , return as a considerably most satisfying .  
12.Reassign with , and go back to Step 4

## Grover-Long algorithm

Grover-Long algorithm is a very well-known algorithm in a quantum computing field that takes quantum state as input and processed to bend the one that is marked to have the highest measurable possibility. And 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 and adjust possibility of the unmarked state to be measured at and the marked state to be measured at .

# 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.

## State the problem

Taking an input as a trajectory data that contains trajectory points.

Each point has 3 attributes which are latitude, longitude and timestamp. The output of the process will be which contains points where .

The is considered as one with desired amount 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.

## Formulate the problem

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

if , point in at index is retained. if , point in at index is discarded. Note that and . can be restated as

where,

and

The ’s LSSED can be computed by :

where is a function measuring error trajectory data to a trajectory data by summing up squared SED created from the points discarded. Our goals are to minimize and to maintain to contain lowest number of trajectory points. In the other hand, the number of that equals needs to be as low as possible.

## Solve the problem

Due to Quantum constraint, this paper aims to simplify n-point trajectory to 2-point trajectory only. And each trajectory data maintain size to preserve to have size which can be represented by qubits.

Let contains trajectory points. That causes to contain binaries.

First, we cast :

,

and

Which means that if (Let it called set ) , point in at index is retained. if (Let it called unset ), point in at index is discarded. Note that and .

## Solve the problem: Quantum Part

The QSMA play a crucial role from this point. Since we know that so, we can economize cost of 2 qubits by taking only to be processed.

Let be a set of qubits representing

where is for

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

### Fixed-#

Let be a desired number of retained points. Recall that 2 s are fixed-set bit so, . The QSMA is to find that can result lowest LSSED where number of set is .

The QSMA performs to achieve Fixed-# by followings:  
1.Randomly select with some state that satisfies set bits i.e., to be set bit and is unset, and are fixed-set so we leave set bits to them.  
2.Declare with .  
3.Select variable with some positive integer.  
4.Design an oracle function that takes as input and returns a mapping set of all s to a marking binary. Mark with if that is less than and satisfy set bits, Unmark with 0 otherwise.  
Since bits of are fixed. The length of the output will be for all possible binary combination.  
5.Perform the function in Step 4 and store it to .  
6.Initialize quantum state with qubits which can represent value up to . Since there are fix-set bits, we can economize qubits.  
7.Map all s to the quantum initial states. The economized qubits create no effect because the also map with fixed-set bit.  
8.Apply Grover-Long algorithm on the quantum state. This returns one that is believed to be one of the state marked by the oracle funtion and can result lowest LSSED amongst others .  
9.Measure quantum state will result an unfinished .  
10.Construct that to the fix-set , and store to .  
11.If is greater than , it means this is a mistake and we go back to Step 5.  
12.If is less than , it means this is a good progress and we assign to and go back to Step 4 for iteration.  
13.If is equal to , it means this is considerably the one that has minimum LSSED. We break the loop and return as an output.  
The pseudocode of the algorithm is shown in Algor. [[algo:FixNum]](#algo:FixNum).

[algo:FixNum] Let be a desired number of retained points Let be a trajectoty data with length Let be an empty binary bits with length Set the first and last bits of with 1 Set bits with 1 Set bits with 0

Let be some positive integer Let

Return

### Fixed-LSSED

Let be a LSSED threshold. The QSMA is to find that can result LSSED not exceeding where number of set is minimized.

The QSMA performs to achieve Fixed-LSSED by followings:  
1.Randomly select with some state that satisfies set bits i.e., all to be set bits which will make which surely satisfies .  
2.Declare with .  
3.Select variable with some positive integer.  
4.Design an oracle function that takes as input and returns a mapping set of all s to a marking binary. Mark with if that is less than and that contains set bit less than or equal to , Unmark with 0 otherwise.  
Since bits of are fixed. The length of the output will be for all possible binary combination.  
5.Perform the function in Step 4 and store it to .  
6.Initialize quantum state with qubits which can represent value up to . Since there are fix-set bits, we can economize qubits.  
7.Map all s to the quantum initial states. The economized qubits create no effect because the also map with fixed-set bit.  
8.Apply Grover-Long algorithm on the quantum state. This returns one that is believed to be one of the state marked by the oracle funtion and contain set bit less than or equal to   
9.Measure quantum state will result an unfinished .  
10.Construct that to the fix-set , and store to .  
11.If contains set bit more than does, it means this is a mistake and we go back to Step 5.  
12.If contains set bit less than does, it means this is a good progress and we assign to and go back to Step 4 for iteration.  
13.If contains set bits that equals to does, it means this is considerably the one that has minimum set bit with LSSED not exceeding . We break the loop and return as an output.  
The pseudocode of the algorithm is shown in Algor. [[algo:FixSED]](#algo:FixSED).

[algo:FixSED] Let be an LSSED threshold Let be a trajectoty data with length Let be an empty binary bits with length Set the all bits of with 1 Let be some positive integer Let

Return

## Obtain the answer

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

# Evaluation

An evalution is achieved with quantum simulator provided by Huawei’s HiQ Simulator written in Python 3.5. Code is available on . 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.

#### Fixed-# Evaluation

Let and be a trajectory data of length . QMSA is to find solution with minimum LSSED and keeping points as retained out of points. So, where , and is optimized by QMSA. Let and . The result is shown in [Table. [1](#table:evalNum)] and [Table. [2](#table:evalNum2)] respectively, where “#iteration" means a number of iterations, “#satisfying condition" means a number of time that contains retained points and “#finest solution" means a number of time that contains retained points and result lowest amongst others possibilities.

Fixed-# Evaluation

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

Fixed-# Evaluation

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

#### Fixed-LSSED Evaluation

Let and be a trajectory data of length . QMSA is to find solution with LSSED not exceeding the threshold while minimizing a number of retained points as lowest. So, where , and is optimized by QMSA. Let and . The result is shown in [Table. [3](#table:evalSED)] and [Table. [4](#table:evalSED2)] respectively, where “#iteration" means a number of iterations, “#satisfying condition" means a number of time that results not exceeding and “#finest solution" means a number of time that result not exceeding and contains lowest retained points amongst others possibilities.

Fixed-LSSED Evaluation

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

Fixed-LSSED Evaluation

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

# 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 complexity of is needed where is a number of trajectory point. By exploiting quantum mechanics from the QSMA, we can do the same job by comsuming only the complexity of where is a number of finest solution.

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 and is the one that is possibly the best around .

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 of 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 may suggest to try exploring other possibilities by exploiting the QSMA or our algorithms.

# Acknowledgment

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ”. Instead, try “R. B. G. thanks”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

00 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. G. L. Long, “Grover algorithm with zero theoretical failure rate,” arXiv:quant-ph/0106071, 13 Jun 2001. A. Ahuja and S. Kapoor, “A Quantum Algorithm for finding the Maximum,” arXiv:quant-ph/9911082, 18 Nov 1999. C. Dürr and P. Høyer, “A quantum algorithm for finding the minimum,” arXiv:quant-ph/9911082, 18 Nov 1999.

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.

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

1. Identify applicable funding agency here. If none, delete this. [↑](#footnote-ref-20)