­­­

A Quantum Optimization Algorithm to Efficiently Route Public Transportation in Cities

Ashwin Kaliyaperumal

Nikola Tesla STEM High School

­­Abstract

Cars are one of the biggest polluters of greenhouse gases and to reduce their impact on the environment, there needs to be a nation-wide shift towards the use of public transportation - specifically the use of buses. Designing new bus routes can help buses arrive more frequently on time, especially during rush hour, which is an incentive for people to use buses. The Urban Transit Routing Problem (UTRP) is an NP-hard optimization problem, solved with heuristic algorithms that output approximate solutions, that focuses on constructing bus routes from an existing road network. The purpose of this project is to use quantum computing to solve the UTRP by minimizing a road network’s average travel time (ATT), a measure of the average travel time for all passengers to travel between all pairs of bus stops. My engineering goal was to use quantum computing, which has a theoretical advantage over heuristics in solving NP-hard problems, to produce routes with a lower ATT than heuristics for Mandl’s Swiss benchmark and existing bus routes in downtown Seattle. My algorithm achieved an ATT of 10.11 minutes for the benchmark, which is an improvement over current UTRP solvers. Additionally, my algorithm decreased the ATT in a section of downtown Seattle by 53%. This research supported my engineering goals by showing the feasibility of using quantum computers to develop routes with a lower ATT than both classical heuristic methods for a benchmark and existing bus routes in downtown Seattle.

1. Introduction
   1. Rationale

With global temperatures on the rise from greenhouse gas (GHG) emissions, cars, which are one of the biggest sources of anthropogenic GHG emissions, need to be replaced by less polluting means of transport. According to the U.S. Department of Transportation, cars were the second largest polluter of carbon dioxide in 2010 (US Department of Transportation, 2010). An increase in the ridership of public transportation, would help reduce these emissions. Public transportation ridership in the U.S., however, is extremely low. According to the United States Census Bureau, in 2019, only about 5% of people took public transportation. Buses were the most used form of public transportation, with about 2.3% of people in the U.S, or about 46.3% of public transportation users, taking the bus. (US Census Bureau, 2019).

Other forms of non car-centric transportation are already being used frequently around the world. The Dutch have become famous for their increased use of bikes thanks to convenient bike parking, an abundance of safe bike lanes, and a traffic system that considers both bikers and drivers in the Netherlands. Europe and China have expanded their high-speed rail networks over the past twenty years. This has happened at such a quick rate in Europe, in fact, many short distance airlines have had to shut down out of being outcompeted by the new high-speed trains. Both of these examples show the feasibility of switching to more renewable modes of transportation, for both short and long-distance trips.

In the U.S, improvements in bus reliability and scope will most likely increase public transportation the most. Adding new bus routes doesn't require the expensive costs that building new rail networks does, and most of the infrastructure for buses is already built. The existing road network built to serve cars can be used to run buses. Increasing ridership will help buses get more funding to expand their routes and the number of stops they service, reducing the number of cars on the road as more people choose to take the bus.

One reason for a lack of ridership is that buses are regularly delayed from their schedules and are stuck in traffic. According to the Seattle Times, buses from King County Metro arrive on time about 77% of the time and are usually late during rush hour, when people need it most (Gutman, 2018).

A solution to this problem could be the use of a quantum optimization algorithm that would help efficiently route buses, increasing the percentage of buses that arrive on time as well as minimizing the amount of time the buses spend stuck in traffic.

This problem has been introduced and developed as the urban transit routing problem (UTRP), a variation on the more common vehicle routing problem. The UTRP starts with a graph of bus stops, which are connected with edges that each have their own travel time. Each pair of bus stops has their own travel demand that indicates the amount of people that want to travel from a specific bus stop to another. UTRP solutions try to minimize the overall time it takes to get from one bus stop to another throughout the entire set of bus stops, as well as decrease the total number of transfers between stops. It also considers the travel demand and transfer time between routes.

Some classical algorithms that have been used to solve this optimization problem have used metaheuristic algorithms, like genetic algorithms, particle swarm optimization, and simulated annealing.

However, quantum computers are better suited towards solving optimization problems than modern, classical computers. Instead of using bits to store information, which can only be in a binary state, quantum computers use qubits that can be in both states simultaneously. This stems from the quantum mechanical property of superposition and allows for the computational power of a quantum computer to increase exponentially with the number of qubits it has. For example, a quantum computer with fifty qubits matches the processing power of a modern computer with 250 bits. Quantum computers also use other quantum mechanical properties, like interference and entanglement, which allow for additional speedups when computing tasks in parallel and further reduces the computational time.

Modern quantum computers, however, are limited in their size and accuracy. These noisy-intermediate scale quantum computers usually don’t work well with large programs and certain qubit operations. The solution to this is to use quantum annealing computers, like the ones developed by D-Wave. These solve optimization problems, formatted in the Quadratic Unconstrained Binary Optimization (QUBO) model, through a heuristic approach called quantum annealing. These computers currently support problems with over 1000 qubits.

Formatting the UTRP into a problem that a quantum computer can optimize will allow for more efficient solutions compared to classical algorithms.

In this paper, I focus on answering two research questions:

1. Can a quantum computing approach to the UTRP produce with a lower average travel time for the Mandl’s Swiss Road network benchmark?
2. Can a quantum computing algorithm to solve the UTRP produce routes with a lower average travel time than existing bus routes in cities?

where the average travel time is defined as the average travel time over the entire road network in minutes per transit user.

1.2. UTRP Definition

The UTRP attempts to design routes (in this case, bus routes) with a known set of nodes (bus stops) and a road network that connects the nodes. The graph with the connected road network with the nodes is also called the transit graph. The UTRP is also constrained with a travel time matrix and a demand cost matrix. The travel time matrix notes the transit time for each pair of nodes that are directly connected. The demand cost matrix describes the number of people that travel from any pair of nodes. Both matrices are symmetric, so that both the travel time and demand between any two nodes are the same in both directions of travel.

The UTRP generally has two parts to it: the operator cost and the passenger cost. In this paper, I only focused on optimizing the passenger cost. The passenger cost, for an initial transit graph, is the average travel time for all sets of stops and for all passengers, with a specific route set. This is defined mathematically by Buba et al. with the formula:

where n is the total number of nodes, i and j are the node numbers, is the demand between nodes i and j, and is the minimum total travel time between nodes i and j (Buba et al. 2016). The minimum travel time for a certain route set is calculated by using the Floyd-Warshall algorithm (Floyd, 1962) and the travel time matrix.

Solutions to the UTRP consist of a route set with a certain number of routes. Each route is a path, with a start and end node, that contains a set number of nodes. For example, in the Mandl’s Swiss Road Network benchmark, I will be optimizing for four routes with a maximum of eight nodes each. Transfers between routes are allowed, defined to be a switching of routes in a route set and are only allowed if both routes share a common node to transfer from. Each route and the overall route set are subject to some additional constraints as well: 1) all nodes in the original graph must be included, 2) each route must connect to any other route (this constraint guarantees that every node will be able to reach any other node with a certain number of transfers), and 3) each route is free of any repeated nodes.

2. Formulation of the UTRP Objective Function and Conversion to a Quantum Problem

The quantum annealing (QA) algorithm, to solve the UTRP, attempts to optimize and find the candidate routes that, together, meet the highest amount of demand. There are four steps to the algorithm:

1. Generate candidate route sets from the original transit graph.
2. Create a QUBO matrix using the candidate route sets and the UTRP cost function.
3. Optimize the QUBO matrix using D-Wave’s hybrid quantum-classical samplers and get the optimized route sets.
4. If there are any nodes missing from the original transit graph in the optimized route sets, repeat from step 2 and add a weigh factor to the QUBO matrix in step 2.

2.1. Generating Candidate Route Sets

The candidate route sets are generated using a Breadth First Search algorithm to find all possible routes for each pair of nodes. Potential routes that contain exactly the maximum number of nodes per route are added randomly to the list of candidate routes chosen to include about 10% of all routes that had exactly the maximum number of nodes, however, this percentage can be tweaked to allow a greater or smaller sample set of candidate route sets.

2.2. The UTRP Objective Function

The UTRP objective function for the QA attempts to both choose a set number of routes and to make sure the overall route set meets as much demand as possible. To ensure this, the objective function is split up into two parts: a cost function and a constraint function. The objective function is written in terms of binary variables, as it will be easier to convert the function into a form the quantum annealer can solve. This will be gone over in further depth in a later section.

The cost function is mathematically described as:

where *v* and *w* are nodes on the transit graph, is the binary variable that describes if a certain candidate route is in the optimized route set (*i* is the route index and *j* is the candidate route index), is the set of candidate routes that don’t contain the nodes v and w, is the travel time between nodes *v* and *w* on the transit graph, and is the demand between nodes *v* and *w*. This cost function penalizes routes that do not meet the demand of a certain pair of nodes. The overall objective function applies this cost for all pairs of nodes.

The constraint function ensures that, for each route index *i,* there is only one candidate route *j* that gets picked. As formulated by Neukart et al., this is described mathematically as:

where, again, *i* is the route index, *j* is the candidate route index, and is the binary variable that describes if a certain candidate route is in the optimized route set. is a scaling factor that guarantees that the constraint is always met (Neukart et al. 2017). It is found by using the value of the maximum cost of any two routes. This ensures that the constraint is always greater than the cost function and only returns *i* number of routes.

Putting the cost and constraint functions together yields the objective function, described mathematically as:

where the first term is the cost function, applied over every pair of nodes in the transit graph, and the second term is the one-route constraint.

2.3. Mapping of the UTRP Objective Function as a QUBO Problem.

The objective function is mapped as a QUBO problem. Mathematically, a QUBO problem is defined as:

where *x* is a vector of binary variables of length *n* and Q is an *n­*-by-*n* array that encodes the problem. The goal of optimizing a QUBO problem is to find a vector of binary variables of length *n* that minimizes the cost of the QUBO objective function.

Quantum annealers can turn a QUBO problem into an Ising Hamiltonian that encodes the problem as an energy minimization problem. This allows both the quantum annealer to natively solve the QUBO and allows for the formatting of an optimization problem as a QUBO problem for the QA to solve.

As stated previously, the UTRP objective function for the QA is written in terms of binary variables. This allows for the reformatting of the UTRP objective function in the format of the QUBO objective function. As a reminder, the vector of binary variables *x* is a vector of size *i \* j*, where *i* is the number of routes and *j* is the number of candidate routes. Each route uses the same list of candidate routes.

To convert the UTRP objective function into the QUBO objective function, all that is needed is to find the parameters for the Q matrix. This can be done explicitly by adding up the cost of the cost function and constraint function separately for , or, for clarity, , where *a* and *b* are the row index and column index respectively. Both a and b are equal to the length of *x* and correspond to a route number – candidate route pairing.

To explicitly add the cost function to the Q matrix, I use the following procedure:

1. If the candidate route doesn’t contain both nodes *v* and *w* and *a* equals *b*, add the travel time and demand for nodes *v* and *w* to
2. If the candidate route doesn’t contain both nodes *v* and *w* and *a* does not equal *b*, add the travel time and demand for nodes *v* and *w* to
3. Repeat steps 3 – 4 for each index *a* and *b*
4. Repeat steps 3 – 5 for each node *v* and *w*

A similar approach is taken to explicitly add the constraint to the Q matrix, which is described in the following procedure:

1. If a equals *b*, subtract to
2. If *a* does not equal *b*, add to
3. Repeat steps 1 – 2 for each index *a* and *b*

After calculating out a Q matrix for a candidate route set, optimizing it only requires inputting the matrix into D-waves hybrid quantum classical solver. In this paper, I used the Leap Hybrid Sampler from D-Wave (D-Wave Systems 2017).

3. Comparing the QA UTRP Solver with Classical UTRP Solvers

It isn’t possible to directly compare different classical UTRP solvers, as well as the QA UTRP solver, together because objective functions and constraints vary between solvers. To solve this problem, a standardized benchmark, Mandl’s Swiss Road Network, as well as different performance parameters have been created in previous research on the UTRP.

Mandl’s Swiss Road Network is comprised of three standardized parts: the road network graph, the travel time matrix for each pair of connected nodes, and the demand cost matrix for each pair of nodes. There are five main performance parameters:

1. - the percentage of demand satisfied without any transfers,
2. - the percentage of demand satisfied with a single transfer,
3. - the percentage of demand satisfied with two transfers,
4. - the percentage of demand unsatisfied, and
5. Average Travel Time (ATT) - the average travel time over the entire road network in minutes per transit user.

Diagram

Description automatically generated

Figure 1: A graph of Mandl’s Swiss Road Network benchmark with travel times for each edge labeled

To compare the QA UTRP solver to prior solutions to the UTRP, I obtained the best solution from Mandl’s Swiss Road Network using the QA UTRP solver and calculated the performance parameters for the solution route. I compared my solver’s best solutions to the different classical UTRP approaches. A summary of the performance parameters are in Table 1.

Ideal ATT, according to Mumford, is 10.01. The Hybrid QA Solver was 0.2% off of the ideal. Compared to the best overall ATT, found by the Differential Evolution Solver, who was at 3.5% off of the ideal ATT for Mandl’s Swiss Road Network.

My algorithm was able to decrease ATT from 10.36 minutes from the current best performing classical heuristic, Differential Evolution, to 10.11 minutes. To put this in perspective, in terms of total travel time (TTT), my algorithm decreased TTT from about 161,000 minutes to 157,000 minutes. This is a decrease of 4,000 minutes of travel time for all passengers in Mandl Swiss’s Road Network.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameters | Initial Heuristic (Fan 2008) | Particle Swarm Optimization (Kechagiopoulos et al. 2014) | Differential Evolution (Buba et al 2016) | QC Solver |
| d0 (%) | 69.94 | 91.84 | 94.28 | 91.65 |
| d1 (%) | 29.93 | 7.64 | 5.72 | 8.34 |
| d2 (%) | 0.13 | 0.52 | 0.00 | 0.00 |
| dun (%) | 0.00 | 0.00 | 0.00 | 0.00 |
| ATT (min) | 12.90 | 10.64 | 10.36 | 10.11 |

Table 1: Comparison of my algorithm (QC Solver) with other classical heuristics

4. Applying the QA UTRP Solver to Optimize Down Seattle Bus Routes

To test the feasibility of UTRP solutions and to compare a real-world application of a quantum annealing solution, I used our algorithm to generate an optimized bus route set for bus stops in downtown Seattle.

4.1. Creating and Formatting Data for Downtown Seattle

I planned on optimizing bus routes that passed through bus stops located in a 1250-meter radius from the center of downtown Seattle, located at the coordinate point (47.608013, -122.335167). I will refer to this area as the “downtown Seattle area” from now on. All of these routes are run by King County Metro, which lists their stops and their route data on Open Transit Data (Open Transit Data).

To locate all of the bus stops in the downtown Seattle area, I created a bounding box defined with four latitude-longitude points and filtered out all latitude-longitude points. I did this by approximating the change in distance from the original coordinate point using the Earth’s radius. I got the bounding box coordinates (i.e., the range of coordinates that I used to define the downtown Seattle area.

To match the problem format of the UTRP, I needed to create an undirected graph of with the downtown Seattle bus routes and bus stops, to create the travel time and demand cost matrix.

There are two steps to create the undirected graph of downtown Seattle, 1) creating an OSMnx graph (Boeing 2017) a certain radius around a center point, which I did earlier, and 2) adding in all of the bus stops as singular points.

To calculate the travel time matrix, I used OSMnx get\_travel\_time() method to calculate the travel time between all pairs of bus stops that were connected by the road network. They were indexed in order of their bus ID number described in the stops.txt file from OpenTransitData.

To calculate the demand cost matrix, I imported Annual Average Weekday Traffic (AAWDT) data from SeattleGeoData (Seattle GeoData) and included it in the attribute data for each street on the OSMnx graph. To approximate the AAWDT for each pair of nodes, I then calculated the shortest path between each node and added up the AAWDT for all of the road segments on that graph.

To import the existing routes, I used the data from trips.txt to approximate the bus routes already taken. This provided a map of downtown Seattle as shown below, where the purple lines are the existing bus routes in downtown Seattle.

Graphical user interface

Description automatically generated

Figure 2: Existing route set in downtown Seattle

A picture containing graphical user interface

Description automatically generated

Figure 3: Optimized route set created by quantum algorithm

4.2. Results

I calculated the ATT for both the existing route set and the optimized route set that my quantum algorithm created, with the results shown in the table below.

|  |  |
| --- | --- |
| Parameters | ATT (min) |
| Existing downtown Seattle bus routes | 7.08 |
| Optimized routes using QC algorithm | 3.33 |

Table 2: Comparison of my algorithm's optimized routes with the existing routes in downtown Seattle. My algorithm was able to decrease average travel time by 53%.

The optimized route set is shown in the figure below, with the lines in blue showing all of the roads covered by the optimized r­­­oute set.

My routes were able to decrease the average travel time by 53%, from a travel time of 7.08 minutes to 3.33 minutes.

However, my algorithm did omit some bus stops on the outer edge of the area I tried to optimize. This shows that my algorithm, along with optimizing the travel time, “chose” certain bus stops to not include in the final route set. This finding shows that the quantum algorithm is actively deciding which bus stops are worth including in the final route set. This can help urban planners to make decisions if certain bus stops should be discontinued.

5. Discussion and Future Applications

The quantum algorithm is still limited in size in both the number of candidate routes and the number of bus stops it can include. This is due to both classical memory constraints (the computer I tested on couldn’t hold an array of more than about 500 candidate routes by 50 bus routes of Q matrix data) and quantum restraints (D-Wave only has a limited number of qubits to perform computations with). This means that the algorithm is only able to optimize routes for small cities or sections of big cities. However, classical computers can be upgrades to contain more memory and there is a trend of the size of quantum computers growing quickly, so this limited capability could be fixed in a few years.

My algorithm also only focuses on optimizing existing bus routes using average travel time and demand information; it doesn’t consider the number of buses on each line (frequency setting) or the existing bus schedule (timetable setting). My algorithm also doesn’t develop new bus stops, which could limit the minimization of average travel time. I would like to improve my algorithm and the UTRP formulation to use demand data about areas, instead of specific bus stops, to create optimized bus routes. In other words, the demand data would contain information about hotspots in the entire city, instead of focusing on specific bus stops, which would allow for the creation of a more informed placement of bus stops.

References

Boeing, G. (2017). Osmnx: new methods for acquiring, constructing, analyzing, and visualizing complex street networks. Comput. Environ. Urban Syst. 65, 126–139. doi:10.1016/j.compenvurbsys.2017.05.004

Buba, A.T. and Lee, S. (2016) Differential Evolution for Urban Transit Routing Problem. Journal of Computer and Communications, 4, 11-25. <http://dx.doi.org/10.4236/jcc.2016.414002>

Commuting by public transportation in the United States: 2019 - Census.gov. (n.d.). Retrieved December 15, 2022, from <https://www.census.gov/content/dam/Census/library/publications/2021/acs/acs-48.pdf>

D-Wave Systems. (2017). D-Wave Initiates Open Quantum Software Environment. Available at: https://www.dwavesys.com/press-releases/d-wave- initiates-open-quantum-software-environment

Fan, Lang, and Mumford, Christine. A metaheuristic approach to the urban transit routing problem. Journal of Heuristics, Vol 16 (3) 2010 pp 353-372. (Also first published online in 2008). A metaheuristic approach to the urban transit routing problem | SpringerLink

Floyd, R.W., 1962. Algorithm 97: Shortest path. Communications of the ACM 5-6, 345.

Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. 2019. Quantum supremacy using a programmable superconducting processor. Nature 574, 7779 (2019), 505–510.

Gutman, D. (2018, February 12). Where's the bus? here's how Metro Transit decides whether your bus is on Time. The Seattle Times. Retrieved December 14, 2022, from <https://www.seattletimes.com/seattle-news/transportation/your-metro-bus-is-on-time-more-often-than-you-may-realize-heres-why/>

Kechagiopoulos, P.N. and Beligiannis, G.N. (2014) Solving the Urban Transit Routing Problem Using a Particle Swarm Optimization Based Algorithm. Applied Soft Computing, 21, 654-676. <http://dx.doi.org/10.1016/j.asoc.2014.04.005>

Mumford, C.L. (2013) New Heuristic and Evolutionary Operators for the Multi-Objective Urban Transit Routing Problem. 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 20-23 June 2013, 939-946. <http://dx.doi.org/10.1109/CEC.2013.6557668>

Neukart F, Compostella G, Seidel C, von Dollen D, Yarkoni S and Parney B (2017) Traffic Flow Optimization Using a Quantum Annealer. Front. ICT 4:29. doi: 10.3389/fict.2017.00029

Open Transit Data (OTD). Open Transit Data (OTD) | Developer resources | Sound Transit. (n.d.). Retrieved December 14, 2022, from <https://www.soundtransit.org/help-contacts/business-information/open-transit-data-otd>

Public transportation’s role in responding to climate change. (n.d.). Retrieved December 15, 2022, from <https://www.transit.dot.gov/sites/fta.dot.gov/files/docs/PublicTransportationsRoleInRespondingToClimateChange2010.pdf>

Seattle geodata. Seattle GeoData. (n.d.). Retrieved April 30, 2023, from <https://data-seattlecitygis.opendata.arcgis.com/>