Quantum-inspired algorithm for vehicle sharing problem

Whei Yeap SUEN
Singapore Management University, wysuen@smu.edu.sg

Chun Yat LEE
Singapore Management University, chunyat.lee.2019@mitb.smu.edu.sg

Hoong Chuin LAU
Singapore Management University, hclau@smu.edu.sg

Quantum-inspired algorithm for Vehicle Sharing Problem

Whei Yeap Suen*, Chun Yat Lee†, Hoong Chuin Lau‡
School of Computing and Information Systems
Singapore Management University
Singapore
*wysuen@smu.edu.sg, †chunyat.lee.2019@mitb.smu.edu.sg, ‡hclau@smu.edu.sg

Abstract—**Recent hardware developments in quantum technologies have inspired a myriad of special-purpose hardware devices tasked to solve optimization problems. In this paper, we explore the application of Fujitsu's quantum-inspired CMOS-based Digital Annealer (DA) in solving constrained routing problems arising in transportation and logistics. More precisely in this paper, we study the vehicle sharing problem and show that the DA as a QUBO solver can potentially fill the gap between two common methods: exact solvers like Cplex and heuristics. We benchmark the scalability and quality of solutions obtained by DA with Cplex and with a greedy heuristic. Our results show that the DA is a general QUBO solver that is more robust than heuristics, and more scalable than Cplex. Our methodology and framework which focus on QUBO problems is also applicable to other quantum-inspired and fully quantum devices that are undergoing development.**

I. Introduction

Optimization problems are extremely prevalent in our physical world, with applications to real-world planning and scheduling problems. Presently, most widely used methods to solve them are exact solvers like Cplex/Gurobi, or handcrafted heuristic algorithms. Both have their respective strengths and weaknesses. Exact solvers like Cplex are able to obtain optimal solutions of optimization problems. However, the efficiency and quality of solutions obtained are dependent on characteristics of the problem, e.g. model size, types of constraints, and scaling of the model. These affect the intrinsic solvability of the model, and could take years or decades to solve optimally even on the fastest imaginable computers today. In contrast, heuristic algorithms offer quick solutions which are easy to understand and implement. They are practical, serving as fast and feasible short-term solutions to planning and scheduling problems. However, in general, heuristic algorithms are unable to deliver provably optimal solutions.

Recently, developments in quantum technologies have inspired a myriad of special-purpose hardware devices tasked to solve optimization problems. These devices range from physics-inspired classical machines [1], [2], to classical-quantum hybrid [3], to quantum hardware [4], [5]. These hardware developments have brought forth new possibilities in solving various optimization problems, such as factory optimization, drug discovery, banking and financial services. However, the commercial prospects of these new hardware and techniques largely depend on whether they can overcome the shortcomings of current hardware technologies.

Alongside this trend, our interest lies in the viability of the Fujitsu's quantum inspired Digital Annealer (DA) [6] in solving real-world transportation and logistics problems at its current stage. DA is a CMOS hardware specially designed to mimic the behaviour of quantum annealing devices used to solve optimization problems such as constrained clustering [7]. While many problems that have demonstrated success have relatively simple constraints, it is interesting to investigate the application the same technology to solve complex constrained routing problems. In this paper, we study the Vehicle Sharing Problem (VSP), where the goal is to minimize total traveling time of a fleet of vehicles transporting passengers from their starting positions to their destinations. The VSP and its variations play an important role in urban mobility in the form of taxi services, carpooling services, emergency personnel dispatch, and etc. These activities widely dictate the social and economic efficiencies in modern societies. Common VSP variants are the dial-a-ride problem (DARP) [8], taxi services [9], workplace car pooling [10], and etc. These problems are NP-hard in general, and a wide range of different methods of solving such problems have been developed, ranging from exact methods [11]–[13] to various heuristics algorithms [14]–[18].

Relatively, the study of using quantum computing to solve complex VSP is much less prevalent in the literature. Quantum computing techniques ranging from quantum inspired algorithms executed on classical computers to quantum algorithms implemented on quantum hardware have been studied to solve routing problems. In [19], a path integral Monte Carlo (PIMC) quantum annealing algorithm was used to solve the traveling salesman problem (TSP). Similarly, [20] uses the same technique to solve the capacitated vehicle routing problem (CVRP). There are also classical-quantum hybrid algorithms, such as one proposed in [21] to study the capacitated vehicle routing problem (CVRP). In [22], the authors explored the applicability of quantum annealing in solving selected space planning problem. In our manuscript, we study the VSP using the Fujitsu Digital Annealer (DA) [6]. Our contributions are as follows:

- We formulate single and multi destination static VSP as mixed integer programs (MIP).

- We also formulate their quadratic unconstrained binary optimization (QUBO) representations as inputs to the DA.
- Our experimental results highlights the scalability and robustness of DA compared to Cplex and a nearest neighbour heuristics.

II. Methodology

Quantum annealers like D-Wave solve optimization problems by searching for the ground state of an Ising Hamiltonian corresponding to the objective function of the problem (e.g. [23]). Similarly with the quantum-inspired Fujitsu DA, we formulate the problem Hamiltonian in the form of a QUBO:

$$E(q_i) = \sum_{i=1}^{n} a_i q_i + \sum_{i,j} a_{i,j} q_i q_j \tag{1}$$

where $q_i \in \{0,1\}$ is the binary decision variable of the problem, a_i and $a_{i,j}$ are coefficients that influence the shape of the energy landscape of the problem. The DA currently solves problems of up to 8192 binary variables. Details of the underlying hardware and algorithm in the DA have been published in [6]. To date, the DA has been applied to solve various optimization problems [7], [24]–[27]. These problems are somewhat "simplistic" in that they do not involve sequences and schedules. By studying the VSP, we wish to explore the effectiveness of QUBO models and DA in dealing with these complex constraints.

A. Single Destination VSP (SDVSP)

The VSP aims to minimize total distance traveled by a fleet of vehicles when transporting individuals from their initial locations to their destinations. The SDVSP can be formulated as a modification of the multi TSP where multiple vehicles are deployed to transport multiple passengers to a **single** destination.

Consider a graph $G = (V, E)$, where V is the set of $N+1$ nodes, and E is the set of edges. Associated with each edge $(i,j) \in E$ is a cost (or distance) d_{ij}. We let node 0 to be the common destination of all passengers, and there are K vehicles. The decision variable is defined as:

$$x_{ij} = \begin{cases} 1 & \text{if edge } (i,j) \text{ is included in a route} \\ 0 & \text{otherwise} \end{cases}$$

We also define u_i to be the position of node i on a route, and p to be the maximum number of nodes that can be visited by any vehicle. The MIP is formulated as:

$$\min \sum_{\substack{i\in[1,N] \\ j\in[0,N] \\ i\neq j}} d_{ij} x_{ij} \tag{2}$$

$$\text{s.t.} \sum_{i} x_{i0} = K \tag{3}$$

$$\sum_{i} x_{ij} = 1 \quad \forall j \tag{4}$$

$$\sum_{j} x_{ij} = 1 \quad \forall i \tag{5}$$

$$u_i - u_j + p \cdot x_{ij} \leq p - 1 \quad 1 \leq i \neq j \leq N \tag{6}$$

The objective function (2) minimizes the total distance traveled by all vehicles. Constraint (3) ensures that all K vehicles arrives at the single common destination. Constraint (4) and (5) ensure that each node is only visited and traveled from once. Constraint (6) eliminates sub-tours within each vehicle's route. Using the MIP, we can construct the QUBO formulation that describes the Hamiltonian of the SDVSP by defining penalty terms that serve the same purposes as the constraints in the MIP. We define

$$x^k_{i,l} = \begin{cases} 1 & \text{if } i \text{ is the } l\text{-th node visited by vehicle } k \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The QUBO can then be formulated as:

$$\begin{aligned} H_1 = & \sum_{\substack{i,j\in[1,N] \\ l\in[0,L-1] \\ k\in\mathbb{K}}} d_{i,j} x^k_{i,l} x^k_{j,l+1} + \sum_{\substack{i\in[1,N] \\ k\in\mathbb{K}}} d_{i,0} x^k_{i,3} x^k_{0,4} \\ & \sum_{\substack{i,j\in[1,N] \\ l\in[0,L-1] \\ k\in\mathbb{K}}} c_{i,j} x^k_{i,l} x^k_{j,l+1} + A_1 \sum_{k} \left(\sum_{\substack{i\in[1,N] \\ l\in[0,L-1]}} x^k_{i,l} - 4 \right)^2 \\ & + A_2 \sum_{k,l} \left(\sum_{i\in[1,N]} x^k_{i,l} - 1 \right)^2 + A_3 \sum_{i} \left(\sum_{l,k} x^k_{i,l} - 1 \right)^2 \end{aligned} \tag{8}$$

where $d_{i,j}$ denotes the distance between passenger nodes i and j. Together, the first two terms form the objective function. The third term serves as an additional distance term to heavily discourage the model from assigning passengers that are far apart from each other to the same vehicle. We define

$$c_{i,j} = \begin{cases} M & \text{if} \quad d_{i,j} > r \cdot d_{\max} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where $d_{\max}$ is the maximum inter-node distance in the problem, $0 \leq r \leq 1$ is the coefficient that controls the distance threshold that activates the term, and M is a very large constant. The subsequent terms in (8) are the penalty terms of the QUBO. The fourth term in (8) ensures that each vehicle picks up the maximum number of passengers possible before dropping them off. The fifth term ensures that each position in a tour of each vehicle can only be occupied by a single city and the sixth term ensures that each city can only be assigned to one vehicle and one position in a tour. Together, the last two terms serve as sub-tour elimination penalties. The coefficients A_i's are hyper-parameters that controls the strength of each penalty and shape the energy landscape of the problem.

B. Multi Destination VSP (MDVSP)

The MDVSP generalizes the SDVSP into multiple destinations, and the number of destinations is less than or equals to the number of passengers. This implies that more than one passenger can have the same destination. To preserve a high level of robustness, there are no restriction that vehicles must only pick up passengers going to the same destination. Moreover, vehicles do not need to pick up all the passengers before traveling to their respective destinations, i.e. vehicles can drop off passengers that are along the way the next passenger pick-up.

Using the same graph notation of $G = (V, E)$, where V is the set of nodes and E is the set of edges, we let $v = 1, 2, \ldots, N-1, N$ denote vertices that represent the N number of passengers, and the N number of vertices representing destination locations be $v = N+1, N+2, \ldots, 2N-1, 2N$. While there may be fewer than N number of unique destinations, we still define there to be as many destination vertices as passenger vertices such that each passenger i's corresponding destination vertex is given by $i + N$. Additionally, we define the vertex $V = 0$ to be a dummy depot node such that any travel distance to and from the dummy node is equal to 0. For each vehicle k, let the maximum route length be $L = 8$ and the maximum passenger capacity be $C = 4$. The total number of vehicles required is then given by $K = \lceil \frac{N}{C} \rceil$. The primary decision variable is a 4-index variable defined as follows:

$$x_{ij}^{kl} = \begin{cases} 1 & \text{if edge } (i,j) \text{ is the } l\text{-th edge traveled by vehicle } k \\ 0 & \text{otherwise} \end{cases}$$

The MDVSP MIP model is formulated as follows:

$$\min \sum_{\substack{i,j \in [0,2N] \\ l \in [0,L-1] \\ k \in [0,K-1]}} d_{ij} x_{ij}^{kl} \tag{10}$$

The objective function (10) minimizes the total distance travelled by all vehicles from the first passenger picked up till the last destination visited, excluding the distance travelled from the last destination to the first passenger node the vehicle departs from. This is because in our formulation, each vehicle completes a route that ends at the first node it departs from.

$$\sum_{j,k,l} x_{ij}^{kl} \quad = \quad 1 \qquad \forall i \in [1, 2N] \tag{11}$$

$$\sum_{i,k,l} x_{ij}^{kl} \quad = \quad 1 \qquad \forall j \in [1, 2N] \tag{12}$$

$$\sum_{i,j} x_{ij}^{kl} \quad \leq \quad 1 \qquad \forall k, l \tag{13}$$

Constraints (11) and (12) ensure that each passenger and destination node is visited exactly once, while constraint (13) ensures that each edge position of a vehicle k's tour can only be assigned to one edge (i, j), similar to constraints (4) and (5) in the SDVSP MIP formulation.

$$\sum_{h,l} x_{i+N,h}^{kl} = \sum_{h,l} x_{i,h}^{kl} \qquad \forall i \in [1, N], k \tag{14}$$

Constraint (14) ensures that each passenger i picked up by a vehicle k will be dropped off at destination $i+N$ by the same vehicle k.

$$\begin{gathered} \sum_{h} x_{ih}^{kl} + \sum_{h} x_{i+N,h}^{kl'} \leq 1 \\ \forall i \in [1, N], l, l' \in [0, L-1], l' \leq l, k \end{gathered} \tag{15}$$

$$\begin{gathered} \sum_{h} x_{ih}^{kl} + \sum_{h} x_{i+N,h}^{kl'} \leq 1 \\ \forall i \in [1, N], l \in [0, L-C-1], \\ l' \in [l+C, L-1], l' \leq l, k \end{gathered} \tag{16}$$

Constraints (15) ensures that the destination $i + N$ is visited only after passenger i has been picked up. More specifically, it ensures that the edge position l of vehicle k departing from passenger node i cannot be greater than or after the route position $l^{'}$ of vehicle k departing from the destination node $i + N$ to any other node h. Constraints (16) further ensures that all passengers will be dropped off within $C = 4$ route segments. This has a dual effect of 1) imposing a soft limit on each passenger's journey duration and 2) preventing the vehicle capacity from being exceeded at any point in the journey, since any passenger picked up will be dropped off within the next 4 route segments regardless of how many of the route segments were pick ups or drop offs.

$$\sum_{i} x_{ij}^{kl} \geq \sum_{h} x_{j,h}^{k,l+1} \qquad \forall j \in [1, 2N], k, l \tag{17}$$

Constraint (17) ensures that the route continuity of each vehicle k is not violated.

$$\sum_{l \in [1,L-1],j} x_{0j}^{kl} \quad = \quad 0 \qquad \forall k \tag{18}$$

$$\sum_{l \in [1,L-1],j} x_{i0}^{kl} \quad = \quad 1 \qquad \forall k \tag{19}$$

Constraint (18) ensures that each vehicle k may only leave from the dummy depot node as the first edge traveled, while constraint (19) ensures that each vehicle k may only enter the dummy depot node exactly once, but not as the first edge traveled. Together, this has the effect of constraining each vehicle to only depart from the dummy node exactly once at the start of the route, and to enter the dummy node exactly once at the end of the route.

Interestingly, even though the MIP model of the MDVSP is considerably more complex than its SDVSP counterpart, generalizing the QUBO formulation of the SDVSP to MDVSP is relatively simpler. This is because in a QUBO formulation, we can use quadratic functions of the decision variable when formulating penalty terms. We also let $v = 0, 1, \ldots, N-1$ denote vertices that represent passengers starting locations, and $v = N, N+1, \ldots, 2N-1$ represent their respective destinations (slightly different from the MIP labeling). Similar to the MIP formulation, the destination of a passenger starting at $v = i$ is $v = i + N$. Once again, $\mathbb{K}$ is the set of all vehicles, each with a maximum capacity of 4 passengers, and a maximum $L = 8$ nodes traveled. The total vehicles required is $K = \lceil \frac{N}{4} \rceil$. Using the same decision variables as Equation (7), the problem is formulated as follows:

$$H_2 = \sum_{\substack{i,j\in[0,2N-1]\\ l\in[0,L-1]\\ k\in\mathbb{K}}} d_{i,j} x^k_{i,l} x^k_{j,l+1} + \sum_{\substack{i,i'\in[0,2N-1]\\ l\in[0,L-1]\\ k\in\mathbb{K}}} c_{i,j} x^k_{i,l} x^k_{j,l+1}$$
$$+B_1 \sum_k \left(\sum_{\substack{i\in[0,N-1]\\ l\in[0,L-s]\\ l'\in[l,L]}} x^k_{i,l} x^k_{i+N,l'} \right)^2 + B_2 \sum_{k,l} \left(\sum_{i\in[1,N]} x^k_{i,l} - 1 \right)^2$$
$$+B_3 \sum_i \left(\sum_{l,k} x^k_{i,l} - 1 \right)^2 + B_4 \sum_k \left(\sum_{\substack{i\in[0,N-1]\\ l'<l\in[0,L-1]}} x^k_{i,l} x^k_{i+N,l'} \right)^2$$
$$+B_5 \sum_{k,i} \left(\sum_{l'\in[0,L-1]} x^k_{i+N,l'} - \sum_{l\in[0,L-1]} x^k_{i,l} \right)^2 \tag{20}$$

The objective function (first term) of (20) calculates the total distance traveled by all vehicles, and same additional distance term (Section II-A) is introduced in the second term to penalize assignments of passengers separated far away to the same vehicle. The third term ensures that the passengers picked up is dropped within the next s nodes visited. Fourth term ensures that each route slot of each vehicle can only be occupied by a single city and fifth ensures that each city can only be assigned to one vehicle and one route slot. Similar to the SDVSP, they form sub-tour elimination penalties. The sixth term ensures that destination nodes must only be visited after corresponding passenger nodes. Lastly, the final term ensures that passengers that are picked up are dropped off at corresponding destinations. Similarly, B_i's control the strengths of each penalty term.

In the QUBO formulations presented, each vehicle has to travel L nodes. This is to avoid formulating inequality constraints. Encoding inequality constraints as a penalty term in the QUBO requires the introduction of slack variables, which increases the number of decision variables required. This results in a more complex energy landscape of the problem and lower overall quality of solutions. However, fixing L as a constant mandates all vehicles to travel L nodes. This restricts input instances of the model to those with $N = LK$. Thus, when $N \neq LK$, we let $N_{\max} = \lceil \frac{N}{L} \rceil \cdot K$, and introduce $N_{\max} - N$ fictitious passenger nodes and their destinations. These nodes share the same coordinate as one of the destination nodes in the instance. For simplicity, we set this to the first destination node in the list. This allows us to avoid inequality constraints, and maintains the flexibility of solving input instances of all sizes. In theory, introducing dummy nodes sharing the same coordinates as one of the destination nodes will not increase distance traveled, as they add 0 distance traveled to the vehicle that services the passenger traveling to that destination. In practice, this may not be true, but it is a small trade-off between robustness and average accuracy.

C. *Nearest-neighbour (NN) heuristics*

Aside from Cplex, we also compare the performance of DA with a simple heuristic algorithm. To this end, we implemented a simple nearest neighbour heuristic algorithm, described in Algorithm 1.

Algorithm 1: Nearest neighbour heuristics

```
Identify $K$ number of starting points, by using a farthest
  neighbour algorithm, each one representing a starting point
  of each vehicle.
while # of passengers $<$ vehicle capacity do
    Pick up the passenger nearest to the current passenger. #
      of passengers $+= 1$
end
while # of passengers $> 0$ do
    Go to the nearest destinations of any of the passengers
      currently on-board. # of passengers $-= 1$
end
```

III. Experimental Results

Our experimental results are presented as follows. We first present the overall performance of the DA against Cplex for both the SDVSP and MDVSP by comparing quality of solutions obtained by DA to those obtained by Cplex. Not all instances presented in this paper were solved by Cplex to optimality, so we compare the solutions obtained over the same amount of runtime. For the SDVSP instances, we have runtime limits of 1500, 3000, 4000, and 5000 seconds for instances of size $N = 40, 60, 70,$ and 80 respectively. Similarly, for the MDVSP instances, runtime limits are set to 3600, 7200, and 7200 seconds for instances of size $N = 20, 30,$ and 40 respectively. DA runtime is defined as the total annealing time, which excludes the QUBO construction at the beginning. The largest instances that we experimented on are based on the hardware limit of the DA. On the other hand, the MIP presented are implemented on Cplex version 12.10 on a CPU which has **2 $\times$ Intel® Xeon® Silver 4110 CPU 2.10 GHz, 8 Cores, 16 Logical Processors**.

Subsequently, we present results from using of our algorithms to solve instances generated based on Singapore's population distribution and custom clustered instances. We also note that the parameters A_i's and B_i's in our QUBO formulation are tuned using the hyper-parameter optimization algorithm Optuna [28].

A. *Solution Quality Frequency Distribution*

The results presented in this section utilizes instances generated from the $r101$ and $rc101$ Solomon instances [29]. For the SDVSP, we used the depot in the original Solomon instances as the common destination for all passengers, and we sampled the N passenger nodes randomly from the total ${}^{100}C_N$ possible instances. As for the MDVSP, the N passenger nodes are sampled in the same fashion. Additionally, we sample M destination nodes that are farthest away from each other to obtain a realistic distribution of destinations. We also let M to be a random number between 5 to 10. Passengers are then assigned to destinations randomly. For each N, we generated a total of 30 samples, and compared the performance difference of DA against both Cplex and the NN heuristics, where the performance difference is defined as $\Delta(s) = 100 \times \frac{\text{obj}_{\text{DA}} - \text{obj}_s}{\text{obj}_s}$, where obj_{DA} and obj_s are objective values obtained by DA and solver s respectively. Solver s can be either Cplex or NN-heuristics.

Figure 1 illustrates the frequency distribution of $\Delta(\text{Cplex})$ and $\Delta(\text{NN})$. First, we observe that the NN-heuristics produces better solutions than DA and Cplex. This highlights the strength of a simple NN-heuristics in solving simple problems. We also observe that Cplex produces higher quality solutions than DA. However, we note that the performance gap between the two becomes closer at $N = 80$, suggesting that Cplex does not scale as well at large instances. Also interesting to note that at $N = 80$, there are two instances where DA outperforms Cplex in solving the same problem. The instance with $\Delta(\text{Cplex}) = 100$ implies that Cplex was unable to obtain a feasible solution within the time limit. This difference in scalability between DA and Cplex is also present when we apply the same study to the MDVSP.

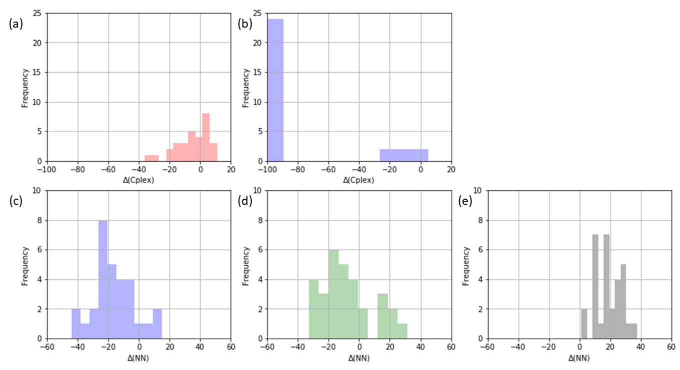

Fig. 2: Frequency distribution of $\Delta(\text{Cplex})$ for MDVSP instances for: (a) $N = 20$, (b) $N = 30$, $\Delta(\text{NN})$ for (c) $N = 20$, (d) $N = 30$, and (e) $N = 40$.

Figure 2 shows the frequency distribution of $\Delta(\text{Cplex})$ and $\Delta(\text{NN})$ for MDVSP instances of different sizes. Interestingly, the relative performance of DA in contrast to Cplex and the NN-heuristics is much better than in the SDVSP. At $N = 20$, the frequency distribution shows that the average $\Delta(\text{Cplex})$ is close to 0, i.e. DA and Cplex are almost equal in terms of solving the $N = 20$ MDVSP instances. For larger instances, Cplex struggles to obtain feasible solutions within the time limit. At $N = 30$, Cplex solved only 5 out of 30 instances within an hour. Instances with $\Delta(\text{Cplex}) = -100$ indicates that Cplex was unable to obtain feasible solution within time limit. Cplex did not return any feasible solutions within 2 hours for instances with $N = 40$ and beyond. Similar behaviour is also observed in [8] which compares performances of different algorithms in solving the DARP. There, it was also observed that Cplex was unable to obtain solution for larger instances.

Also from Figure 2, when $N \in \{20, 30\}$, DA produces higher quality of solution than the NN-heuristics overall. However, at $N = 40$, the NN-heuristics performs better than DA, with a smaller margin compared to the SDVSP case (Figure 1). When N is small, nodes are further from each other. Thus, an algorithm that is designed to pick up/drop off the nearest neighbour all the the time may not be efficient. However, as N increases, nodes are more densely packed, picking up/dropping off the nearest neighbour becomes a better strategy. Like all heuristics designed with specific instructions, the NN-heuristics is not very robust when the characteristics of the problem instances change.

Instance	t_{DA}	obj_{DA}	$\text{obj}_{\text{Cplex}}$	t_{Cplex}	$\Delta(\text{Cplex})$	obj_{NN}	$\Delta(\text{NN})$
sgu_20_1	679	275	203	7200	35.47	246	11.79
sgu_20_2	808	269	249	7200	8.03	218	23.39
sgu_30_1	3325	499		7200		354	40.96
sgu_30_2	2909	515		7200		360	43.06
sgu_32_1	2000	358		7200		375	-4.53
sgu_32_2	2048	356		7200		385	-7.53
sgu_40_1	4066	464	Infeasible	7200	Infeasible	476	-2.52
sgu_40_2	6679	782		7200		436	79.36
sgu_44_1	7847	795		7200		415	91.57
sgu_44_2	7834	845		7200		555	52.25
clustered_16	1000	715	632	600	13.13	915	-21.86
clustered_20	1713	845	857	1800	-1.40	1077	-21.54
clustered_24	2471	2386		2500		2982	-19.99
clustered_32	2557	3321	Infeasible	2600	Infeasible	3947	-15.86
clustered_40	2596	4464		2600		4927	-9.40

TABLE I: Comparison of performance between DA, Cplex, and NN-heuristics for specific instances.

B. *Digital Annealer performance in custom instances*

Our results so far showed that DA scales better than Cplex as the size and complexity of the benchmark problem increases. However, a NN-heuristics outperforms both DA and Cplex in many of the instances studied so far. In this section, we present results obtained by studying two other types of instances. The first type of instances are generated from Singapore's population and industrial data (labeled *sgu_*). The passenger-destination pairs in these instances are for the purposes of 1. transporting security personnel from their homes to camps during an emergency recall, 2. the transporting construction workers between their dormitories and assigned worksites. The clustered instances (labeled *clustered_*) are synthetic instances designed to mimic clusters of residential areas with target destinations outside of them. The goal behind these custom instances is to provide insights on instances with different types of node distributions.

Table I shows despite the Fujitsu DA being a new technology, there area areas in which it can perform well. We observe that the DA is better at solving larger instances of the MDVSP, as Cplex was unable to obtain feasible solution in the same amount of time. We also observe that in some of the instances presented, the DA was able to obtain higher quality solution than the NN-heuristics. This reinforces our findings that despite the NN-heuristics being able to obtain much better solutions for many of the SDVSP and MDVSP

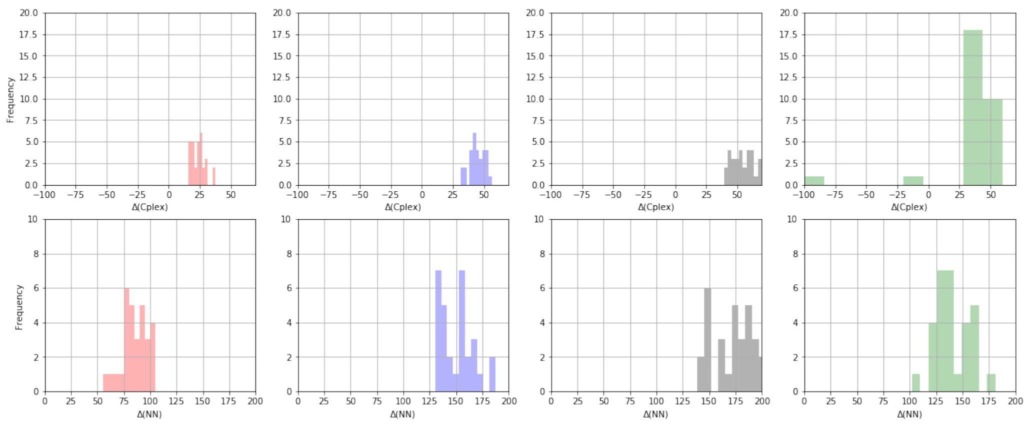
Fig. 1: Frequency distribution of Δ(Cplex) *(top)* and Δ(NN) *(bottom)* for SDVSP instances of different sizes. From left to right: $N = 40, 60, 70, 80$.

instances presented in the previous section, the algorithm is not very robust. This is because like all heuristics, the NN-heuristics are designed for specific purposes, which may not work when the problem changes drastically. In contrast to that, the DA has a more generalized framework, much like Cplex. As long as the problem Hamiltonian is formulated in the form of QUBO (which can be systematically constructed), DA is able to solve it to the best of its capability.

IV. Compatibility with Quantum Hardware

Our results from using the DA to solve a VSP has highlighted the strengths and weaknesses of the hardware. Studying quantum-inspired hardware such as the DA provides us with various insights on the applications of quantum computing in solving constrained optimization problems. Firstly, since the DA is a QUBO solver, the methodology and framework presented here is applicable to quantum platforms such as D-Wave (quantum annealing) and IBM Qiskit (QAOA and VQE). The QUBO formulation of an optimization problem as input for all these devices will be similar [30].

At the current stage of hardware development, the main advantage of a quantum-inspired classical QUBO solver such as the DA is that it can solve problems that require all-to-all connections between decision variables, such as the VSP, without any embedding or enconding algorithms. Furthermore, classical simulations of quantum algorithms such as QAOA are also very limited in terms of the number of qubits that can be simulated. As a result, the DA will be able to handle such problems the require many more decision variables compared to a quantum solver. This gives us an avenue to study complex optimization problems beyond what is implementable in the current quantum hardware, but will eventually be useful as more powerful quantum hardware are developed.

V. Conclusion and Future Work

We explored the applicability and performance of the Fujitsu DA in solving complex optimization problems involving complex routing constraints. Despite the shortcomings discussed, DA's performance lies somewhere between exact solvers and heuristics algorithms. In short, the DA scales better than Cplex. As QUBO formulations of problems allow for quadratic constraints, we can better solve more complex problems in contrast to an MIP formulation. On the other extreme, against a NN-heuristics, the DA is relatively more robust in terms of the range of problem instances it can solve. Certainly, one can make the case that a much more sophisticated heuristics algorithm would have fared better in Section III-B. Nonetheless, construction of such algorithms requires substantial handcrafting in terms of algorithm design, which is often non-generalizable. QUBO formulations can be systematically constructed for wide range of problems.

Nonetheless, our work has also shed light on some weaknesses of DA. The biggest limitation that is the hardware. DA currently can solve problems up to only 8192 binary variables, which limits the complexity and size of problems we can explore. However, it is still a very new device, and is constantly undergoing development in this regard. As the hardware improves, we hope to be able to study full-scale industrial problems with the DA.

Another area where the DA has room to improve is the quality of solution. We observed that despite having better scalability than Cplex, the solutions obtained by DA for small size instances are not as good as Cplex and the NN-heuristics. To this end, our future work focuses on improving the performance of the DA, through methods like problem parameter and engine parameter tuning, or some DA-heuristics hybrid algorithms. Each of these topics leads to different directions in future research.

VI. Aknowledgements

We would like to thank Fujitsu Laboratories Ltd. for providing access to the Fujitsu Digital Annealer for the duration of this project. This research is funded by the National Research Foundation Singapore under its Corp Lab @ University scheme and Fujitsu Limited as part of the A*STAR-Fujitsu-SMU Urban Computing and Engineering Centre of Excellence.

References

[1] M. Yamaoka, C. Yoshimura, M. Hayashi, T. Okuyama, H. Aoki, and H. Mizuno, "A 20k-spin ising chip to solve combinatorial optimization problems with cmos annealing," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 1, pp. 303–309, 2015.

[2] T. Inagaki, Y. Haribara, K. Igarashi, T. Sonobe, S. Tamate, T. Honjo, A. Marandi, P. L. McMahon, T. Umeki, K. Enbutsu *et al.*, "A coherent ising machine for 2000-node optimization problems," *Science*, vol. 354, no. 6312, pp. 603–606, 2016.

[3] A. Cross, "The ibm q experience and qiskit open-source quantum computing software," *APS*, vol. 2018, pp. L58–003, 2018.

[4] M. W. Johnson, M. H. Amin, S. Gildert, T. Lanting, F. Hamze, N. Dickson, R. Harris, A. J. Berkley, J. Johansson, P. Bunyk *et al.*, "Quantum annealing with manufactured spins," *Nature*, vol. 473, no. 7346, pp. 194–198, 2011.

[5] E. Gibney, "D-wave upgrade: How scientists are using the world's most controversial quantum computer," *Nature News*, vol. 541, no. 7638, p. 447, 2017.

[6] M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura, and H. G. Katzgraber, "Physics-inspired optimization for quadratic unconstrained problems using a digital annealer," *Frontiers in Physics*, vol. 7, p. 48, 2019.

[7] E. Cohen, A. Senderovich, and J. C. Beck, "An ising framework for constrained clustering on special purpose hardware," in *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*. Springer, 2020, pp. 130–147.

[8] H. Hosni, J. Naoum-Sawaya, and H. Artail, "The shared-taxi problem: Formulation and solution methods," *Transportation Research Part B: Methodological*, vol. 70, pp. 303–318, 2014.

[9] C. Ma, R. He, and W. Zhang, "Path optimization of taxi carpooling," *PLoS One*, vol. 13, no. 8, p. e0203221, 2018.

[10] R. Baldacci, V. Maniezzo, and A. Mingozzi, "An exact method for the car pooling problem based on lagrangean column generation," *Operations Research*, vol. 52, no. 3, pp. 422–439, 2004.

[11] H. N. Psaraftis, "An exact algorithm for the single vehicle many-to-many dial-a-ride problem with time windows," *Transportation science*, vol. 17, no. 3, pp. 351–357, 1983.

[12] J.-F. Cordeau, "A branch-and-cut algorithm for the dial-a-ride problem," *Operations Research*, vol. 54, no. 3, pp. 573–586, 2006.

[13] S. Ropke, J.-F. Cordeau, and G. Laporte, "Models and branch-and-cut algorithms for pickup and delivery problems with time windows," *Networks: An International Journal*, vol. 49, no. 4, pp. 258–272, 2007.

[14] L. Coslovich, R. Pesenti, and W. Ukovich, "A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem," *European Journal of Operational Research*, vol. 175, no. 3, pp. 1605–1615, 2006.

[15] R. M. Jorgensen, J. Larsen, and K. B. Bergvinsdottir, "Solving the dial-a-ride problem using genetic algorithms," *Journal of the operational research society*, vol. 58, no. 10, pp. 1321–1331, 2007.

[16] S. N. Parragh, K. F. Doerner, and R. F. Hartl, "Variable neighborhood search for the dial-a-ride problem," *Computers & Operations Research*, vol. 37, no. 6, pp. 1129–1138, 2010.

[17] M. Schilde, K. F. Doerner, and R. F. Hartl, "Metaheuristics for the dynamic stochastic dial-a-ride problem with expected return transports," *Computers & operations research*, vol. 38, no. 12, pp. 1719–1730, 2011.

[18] D. Kirchler and R. W. Calvo, "A granular tabu search algorithm for the dial-a-ride problem," *Transportation Research Part B: Methodological*, vol. 56, pp. 120–135, 2013.

[19] R. Martoňák, G. E. Santoro, and E. Tosatti, "Quantum annealing of the traveling-salesman problem," *Physical Review E*, vol. 70, no. 5, p. 057701, 2004.

[20] A. Crispin and A. Syrichas, "Quantum annealing algorithm for vehicle scheduling," in *2013 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2013, pp. 3523–3528.

[21] S. Feld, C. Roch, T. Gabor, C. Seidel, F. Neukart, I. Galter, W. Mauerer, and C. Linnhoff-Popien, "A hybrid solution method for the capacitated vehicle routing problem using a quantum annealer," *Frontiers in ICT*, vol. 6, p. 13, 2019.

[22] T. Stollenwerk and A. Basermann, "Experiences with scheduling problems on adiabatic quantum computers," in *Proceedings of the 1st International Workshop on Post-Moore Era Supercomputing (PMES)*. Future Technologies Group Technical Report FTGTR-2016-11, 2016, pp. 45–46.

[23] D. Venturelli, D. Marchand, and G. Rojo, "Job shop scheduling solver based on quantum annealing," in *Proc. of ICAPS-16 Workshop on Constraint Satisfaction Techniques for Planning and Scheduling (COPLAS)*, 2016, pp. 25–34.

[24] S. Hong, P. Miasnikof, R. Kwon, and Y. Lawryshyn, "Market graph clustering via qubo and digital annealing," *arXiv preprint arXiv:2006.10716*, 2020.

[25] O. A. Malik, H. Ushijima-Mwesigwa, A. Roy, A. Mandal, and I. Ghosh, "Binary matrix factorization on special purpose hardware," *arXiv preprint arXiv:2010.08693*, 2020.

[26] H. Asaoka and K. Kudo, "Image analysis based on nonnegative/binary matrix factorization," *Journal of the Physical Society of Japan*, vol. 89, no. 8, p. 085001, 2020.

[27] M. Javad-Kalbasi, K. Dabiri, S. Valaee, and A. Sheikholeslami, "Digitally annealed solution for the vertex cover problem with application in cyber security," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2642–2646.

[28] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2623–2631.

[29] S. Marius, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations Research*, vol. 35, no. 2, pp. 254–265, 1987.

[30] F. Glover, G. Kochenberger, and Y. Du, "Quantum bridge analytics i: a tutorial on formulating and using qubo models," *4OR*, vol. 17, no. 4, pp. 335–371, 2019.