

# A GREEDY QUANTUM ROUTE-GENERATION ALGORITHM

Jordan Makansi<sup>1</sup>, David Bernal Neira<sup>2</sup>  
Independent Consultant<sup>1</sup>, Purdue University<sup>2</sup>

## Background and Problem Formulation

In this work, we study the Fleet Size Vehicle Routing Problem with Time Windows (FSVRPTW), which determines the minimum number of vehicles required to service all customers within their respective time windows.

We present an annealing-based algorithm for generating routes under time window constraints that is suitable for the NISQ era [1] by its robustness to noise, convergence to a feasible solution, and competitive computation time and solution quality.

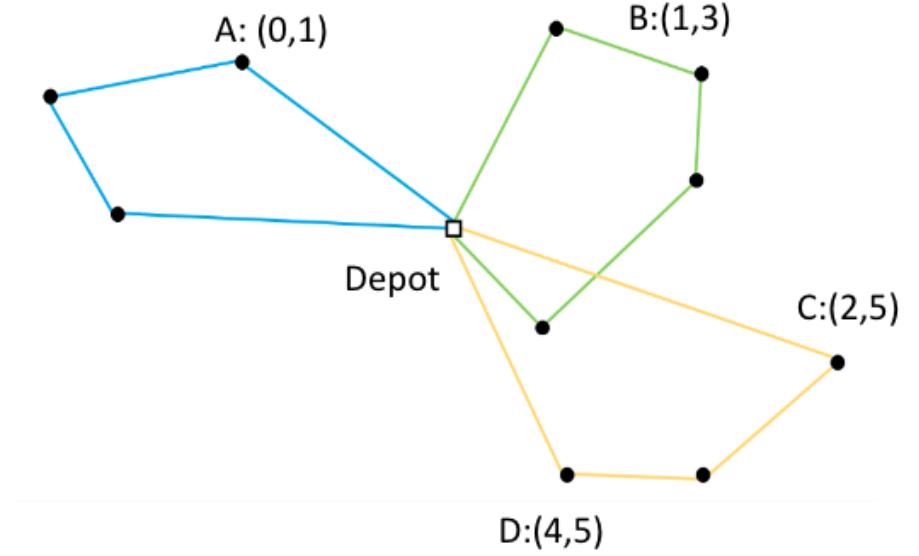


Table 1: Example Timetable and Distance Matrix

i	Timetable		Distance Matrix					
	$e_i$	$l_i$	$q_i$	i	0	1	2	N
0	0.0	$\infty$	0.0	0	0	1	3	0
1	1.0	1.15	1.0	1	1	0	2	1
2	3.5	3.75	1.0	2	3	2	0	3
N	0.0	$\infty$	0.0	N	0	1	3	0

Figure 1: Routing example, where time windows are only shown for customers A, B, C, D

## Methodology

We create binary variables  $x_{i,s,j,t}$  representing "an agent leaves customer  $i$  at time  $s$ , and subsequently leaves customer  $j$  at time  $t$ ". We duplicate the depot: 0 and  $N$ . The set  $T$  is the set of time discretizations, and the set  $W$  is the set of customers, which excludes the depots. For any variable  $x_{i,s,j,t}$ , we can compute  $b_{ij} = \max\{e_j, s + d_{ij}\}$  where  $d_{ij}$  is the distance between customers  $i$  and  $j$ .  $b_{ij}$  is the earliest possible time we can start servicing customer  $j$ , having left customer  $i$  at time  $s$ .

$$\min_{x_{i,s,j,t}} \sum_{j \in W} \sum_t x_{0,0,j,t} \quad (1a)$$

$$\text{s.t. } \sum_{i \in \{0,W\}, s, t} x_{i,s,j,t} = 1, \forall j \in W \quad (1b)$$

$$\sum_{j \in \{0,W\}, j \neq i, t} x_{j,t,i,s} = \sum_{j \in \{W,N\}, j \neq i, t} x_{i,s,j,t}, \forall i \in W, \forall s \quad (1c)$$

By making a few observations, we can prune many of the variables. For example, variables are pruned if they are time-infeasible. The full list of constraints can be found in the ArXiv version of the paper. Any set of active variables  $X^l$  can be represented as a DAG: A node is created for each unique tuple  $(i, s)$ , where  $i \in \{0, W, N\}$  and  $s \in T$ . A directed arc is from  $(i, s)$  to  $(j, t)$  for each variable.

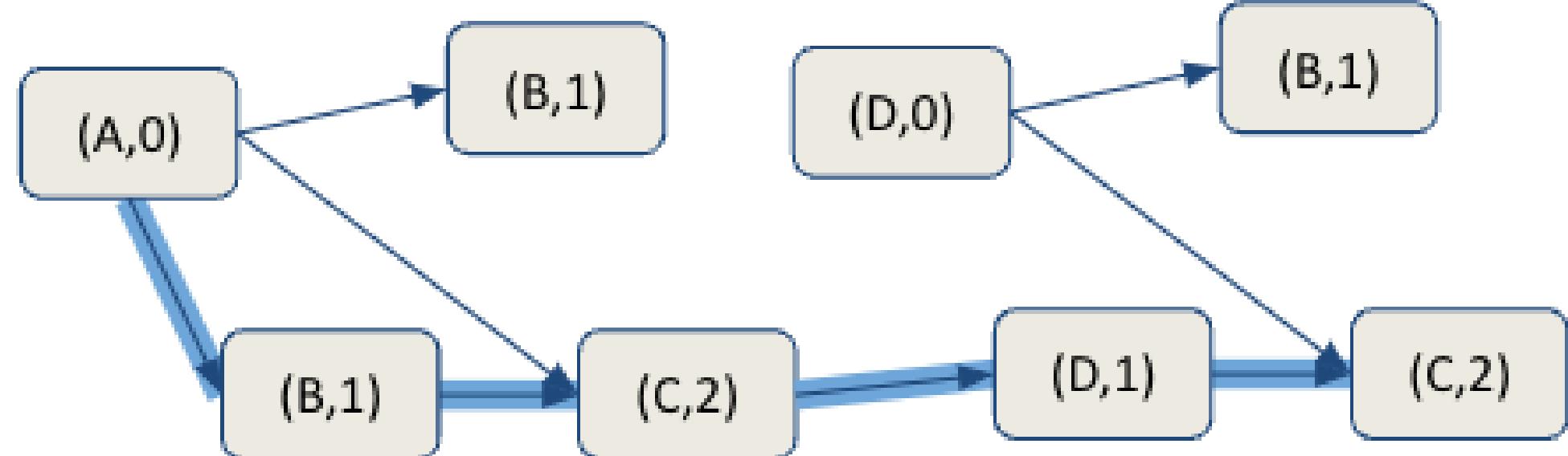


Figure 2: Initial set of variables  $X^0$  represented as a DAG, continued from example in Fig. 1

Our algorithm for solving problem (1) is shown in Figure 3. Any subset of the variables in our formulation can be represented as a directed acyclic graph (DAG). By combining this with the samples obtained from the annealer, we can quickly converge to a feasible solution. The annealer recommends a set of variables to be used in constructing a DAG, and a classical routine is used to find feasible sub-paths within the DAG. The variables corresponding to customers along each of the paths are pruned.

## A Greedy Quantum Route-Generation Algorithm

The algorithm terminates when there are no more active variables or when the problem is small enough to be solved exactly.

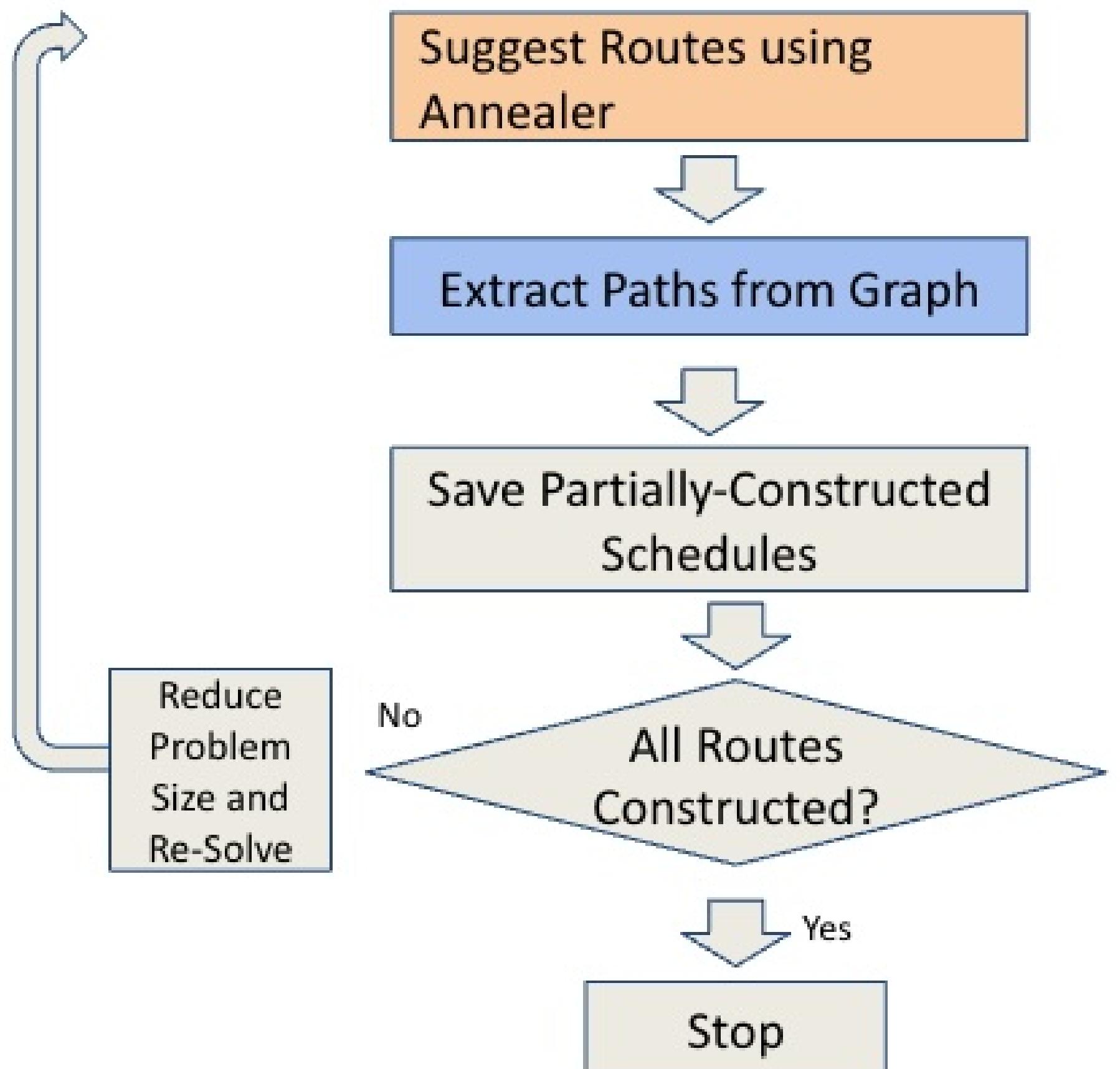


Figure 3: Greedy Quantum Route-Generation Algorithm

To evaluate the performance of our proposed algorithm for practical-sized problems, we used Solomon's benchmark instances [2]. We used instance types  $R101/R201$  that indicate narrow/wide time windows resp. Only  $R101$  results are shown ( $R201$  results are similar and can be found in the ArXiv version).

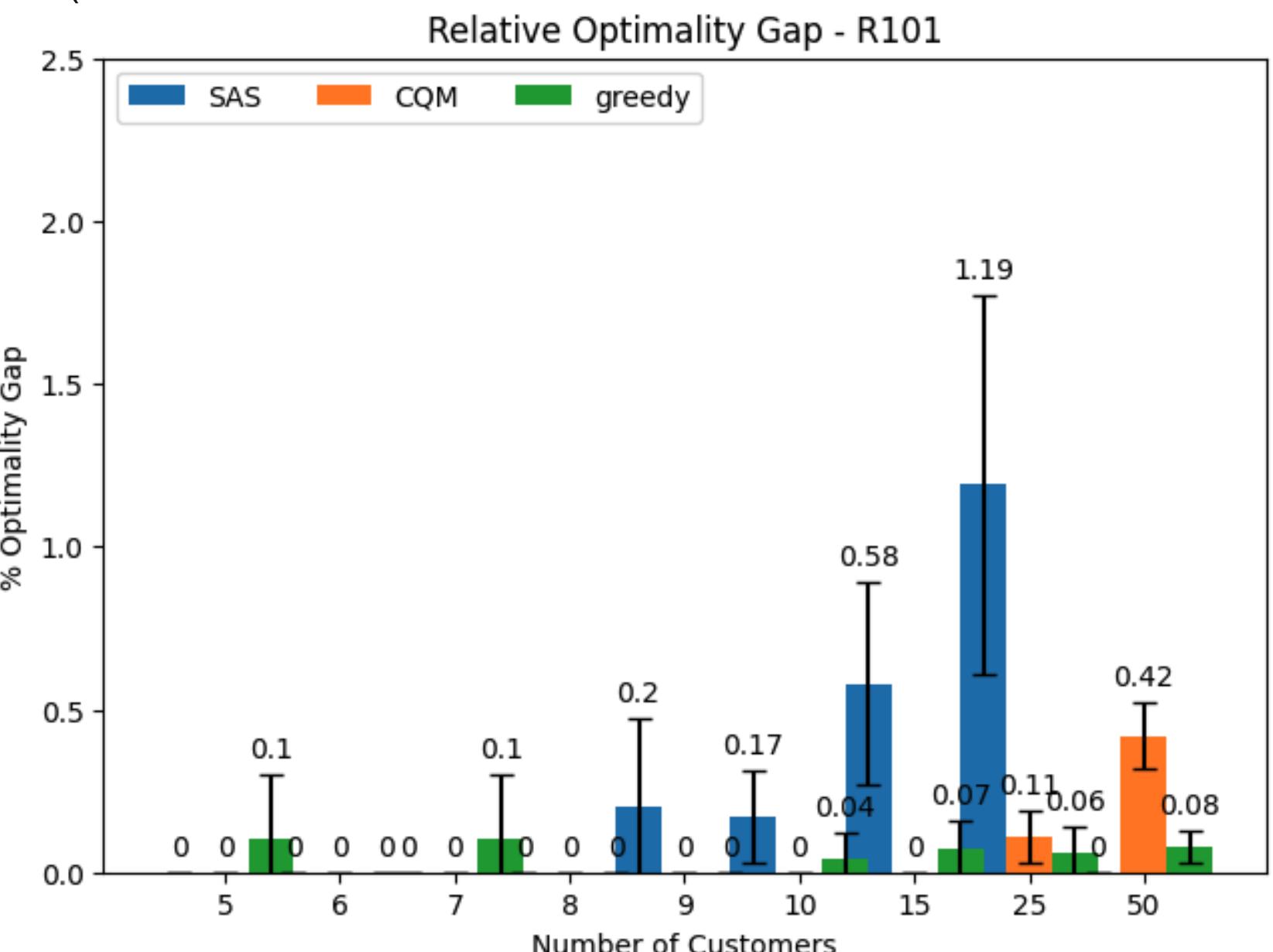


Figure 4: Average relative optimality gap for SAS, CQM, and the proposed greedy algorithm, averaged over 10 examples with standard deviation.

We benchmark with the SimulatedAnnealingSampler (SAS) and LeapHybridCQMSampler (CQM) provided by D-Wave. In our greedy algorithm, we used CQM for step "Suggest Routes using Annealer", with the default parameters.

We can see that our greedy approach enjoys a small relative optimality gap as compared to SAS and CQM, in particular for large problem sizes  $N > 15$ . For  $N = 25$ , SAS returned feasible solutions for only 2 of the 10 random examples and for  $N = 50$  it returned no feasible solutions. Statistics are aggregated only over feasible solutions. To evaluate its robustness to noise, our algorithm was also tested on DWave Advantage 4.1 after tuning the parameters of the DWaveSampler and trying an annealing pause.

## Computational Results

Even after enhancing the DWaveSampler with these features, it is unable to find feasible solutions in any of the 10 examples for problem sizes greater than  $N=5$  customers. The total time of the greedy algorithm and the average relative optimality gap are labeled  $t_{\text{greedy}}$  and  $\alpha$ , respectively, Table 2.

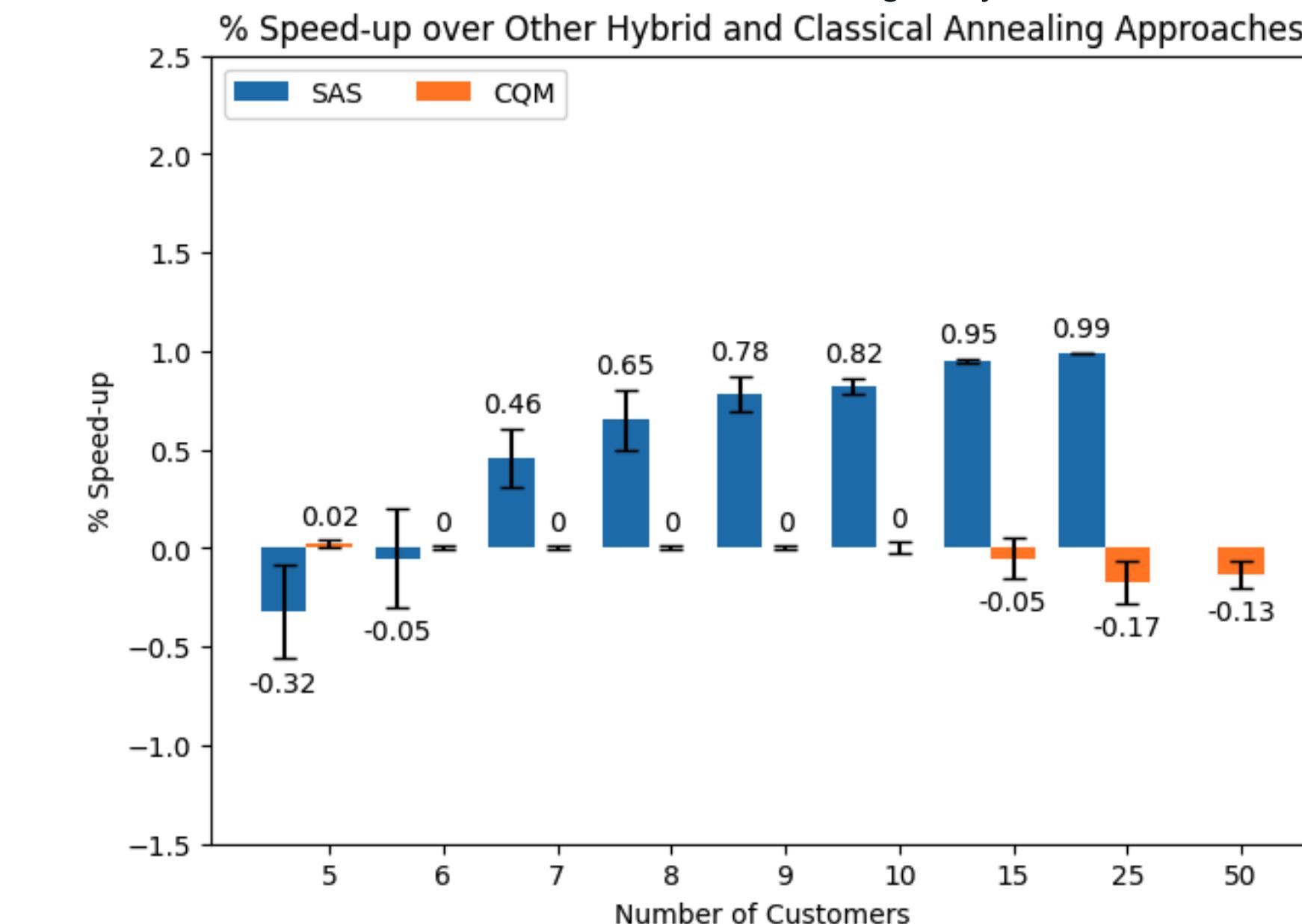


Figure 5: Average relative time difference of greedy compared to SAS and CQM, averaged over 10 examples. Relative time difference is computed as  $\frac{t_A - t_{\text{greedy}}}{t_A}$  where  $t_A$  is the time taken by the annealing method, and  $t_{\text{greedy}}$  is the time taken by our proposed greedy algorithm.

To show the scaling of this formulation on the binary variables, the average number of logical qubits and physical qubits are also reported. When the number of customers exceeds 7, the rate at which the number of logical qubits increases more slowly. This can be explained by the fact that only one discretization value is needed within each time window.

Table 2: QPU Results

N	DWaveSampler Valid	% DWaveSampler with Pause % Valid	$t_{\text{greedy}}$	$\alpha$	Avg.# logical	Avg.# physical
5	20	10	0.08	10	44	415
6	0	0	0.074	17	84	601
7	0	0	0.08	33	109	890
8	0	0	0.09	0	121	955
9	0	0	0.09	0	127	1021

## Conclusions and Future Work

This work presents the first effective algorithm for generating routes using an annealing-based approach. We demonstrated its efficiency by solving the FSVRPTW. We show that it converges to a feasible solution and benchmark it against state-of-the-art classical and hybrid annealing-based approaches using D-Wave. We showed that it enjoys a smaller optimality gap than other annealing-based approaches, at a competitive computation time, even for practical-sized problems of 50 customers. It also shows to be noise-robust when executed on a QPU, taking advantage of the entire sample set returned by the quantum annealer. Natural directions for future work include determining the dependence of the proposed algorithm's performance on the dataset, and the effect of the control parameters.

## Acknowledgements and References

<sup>1</sup>This work was done while the author was at USC under the support of the PWICE Fellowship.

- [1] John Preskill. "Quantum Computing in the NISQ era and beyond". In: *Quantum* 2 (Aug. 2018), p. 79. ISSN: 2521-327X. DOI: 10.22331/q-2018-08-06-79. URL: <http://dx.doi.org/10.22331/q-2018-08-06-79>.
- [2] Marius M Solomon. "Algorithms for the vehicle routing and scheduling problems with time window constraints". In: *Operations research* 35.2 (1987), pp. 254–265.