OnCall Operator Scheduling for Satellites with Grover's Algorithm

Antonius Scherer[2], Tobias Guggemos[1], Sophia Grundner-Culemann[1], Nikolas Pomplun[2], Sven Prüfer[2], and Andreas Spörl[2]

[1] *MNM-Team, Ludwig-Maximilian Universität (LMU) München*, Munich, Germany
{guggemos, grundner-culemann}@nm.ifi.lmu.de
[2] *German Space Operation Center, German Aerospace Center*, Wessling, Germany
{antonius.scherer, nikolas.pomplun, sven.pruefer, andreas.spoerl}@dlr.de

Abstract. The application of quantum algorithms on some problems in NP promises a significant reduction of time complexity. This work uses Grover's Algorithm, designed to search an unstructured database with quadratic speedup, to find valid a solution for an instance of the on-call operator scheduling problem at the German Space Operation Center. We explore new approaches in encoding the problem and construct the Grover oracle automatically from the given constraints and independent of the problem size. Our solution is not designed for currently available quantum chips but aims to scale with their growth in the next years.

Keywords: Grover's Algorithm · On-Call Scheduling · Satellites.

1 Introduction

Grover's algorithm [15] is one of the best-known algorithms offering significant speed-up for computational problems on a quantum computer. It features a quadratic speed-up for every problem which can be described as a search in an unsorted database. Hence, it allows significant improvements for problems where finding solutions requires searching through a large part of the input space. The algorithm consists of three major parts (see Fig. 1):

State preparation creates a superposition of the search space.
Grover Iteration is repeated $^1/_k \cdot \sqrt{N}$ times to solve a search problem for an input space with N elements and $k \geq 1$ valid solutions. It has two parts:
- **Oracle** A search function $f(x)$ is applied on the input state, which outputs 1 for the searched values $\hat{x}$.
- **Diffusion** The amplitude (and therefore the measurement probability) of the searched input states is increased.

Measurement outputs a binary string that represents $\hat{x}$ with high probability.

Satisfiability (SAT) problems are problems where a solution for a set of *variables* has to be found satisfying a given set of *constraints*. Although there is a variety of classical SAT-solvers that allow finding solutions under a large number of constraints, the general satisfaction problem for an arbitrary number of constraints

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

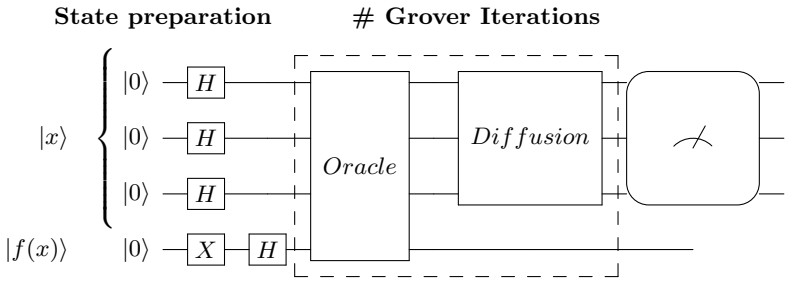


Fig. 1: Major steps of Grover's algorithm.

is believed to be NP-hard [5,8]. Grover's algorithm certainly does not change such problems' complexity, however, once we see scalable quantum computers beyond the NISQ-era, using them may offer significant improvements.

The German Space Operation Center (GSOC) hosts multiple applications where a SAT-solver is used to find a solution for a given problem. One of them is "Spacecraft On-Call Scheduling" (SOCS), where the operators' time-table for various spacecraft missions needs to be determined. Currently, it deals with 24 hour shifts for ~ 50 operators and 20 positions (missions) over a period of 180 days.

Scope of this work

This work aims to solve the aforementioned problem with a quantum computer using Grover's algorithm. However, a quantum device that is able to leverage the potentials of Grover's algorithm is far beyond the horizon. Hence, we present a method for encoding the variables of the SOCS problem into a quantum state which can be used for Grover's algorithm once such a device is available. Additionally, we develop an algorithm for representing constraints as a circuit, which is then used as the *Oracle*. To ensure correctness, we evaluate the approach with *Qiskit* for a reduced problem size that fits the currently available IBMq simulator.

2 Spacecraft On-Call Scheduling

The GSOC as a part of the German Aerospace Center (DLR) operates a variety of spacecraft missions. As depicted in Fig. 2, some of those missions require the constant presence of one or more *operators* per subsystem – called *position* – and *time period*. The operators are organized through an on-call spacecraft operator scheduling which is conducted multiple times per year to incorporate updated unavailability times or new personnel. A valid schedule has to allocate approximately 50 operators on 20 positions over a period of 180 days and may need to be updated if changes in the assumptions arise later on. On-call shifts are scheduled in whole days and every operator can cover a certain subset of positions

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

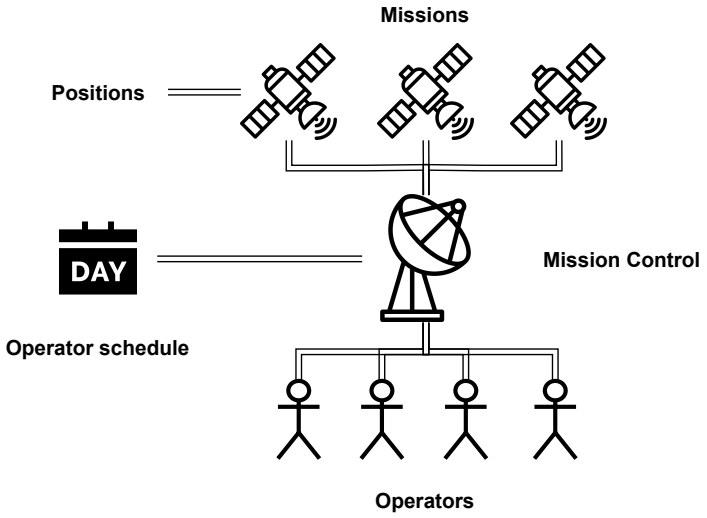


Fig. 2: Overview of the use case "Spacecraft On-Call Scheduling"

depending on their training and responsibilities. A valid schedule depends on various inputs from the spacecraft missions as well as the operators themselves and must satisfy the following constraints:

(i) Operators can only work on some given positions.
(ii) Per tuple of day and position at least one operator needs to be assigned.
(iii) An operator can be assigned to at most one position a day.
(iv) Operators can specify days in advance when they are unavailable.
(v) A partial on-call schedule may be supplied and needs to be obeyed.[3]
(vi) Operators can work at most two out of any three consecutive weeks.
(vii) Operators can work at most 35 days out of any 105 consecutive days.
(viii) Operators shall work preferably whole weeks.
(ix) All operators shall work a similar amount of days.

Notice that **(viii)** and **(ix)** are not constraints but rather optimization goals. As implementing all these constraints efficiently in full generality is rather complicated and is certainly not feasible on todays quantum computers, we adapt the constraints **(ii)**, **(iii)** and **(vi)** and restrict ourselves to them. This means that we consider the following constraints in this paper:

A Per tuple of day and position exactly one operator needs to be assigned.
B Out of three consecutive days, an operator is only allowed to work two.
C An operator can be assigned to at most one position a day.

[3] This is needed e. g. for updating an on-call schedule during the year when operators have updated their vacations, for example. In this case only the future will be replanned, but the past may influence the applied constraints in the future.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

Currently, the full problem at GSOC is solved with a heuristic search algorithm using backtracking based on the Plato library, see [20]. As creating personnel schedules routinely requires manual intervention due to e. g. late changes, the automated algorithm is supported by a graphical tool allowing the on-call planner to easily modify the schedule and recognize conflicts. This tool is called Program for Interactive Timeline Analysis (PINTA), an overview can be found in [21] and a screenshot in Fig. 3. Notice that both Plato and PINTA are tools developed at GSOC for spacecraft mission planning purposes, i. e. they are commonly used for planning onboard activities of spacecraft[26] as well as related onground activities[16].

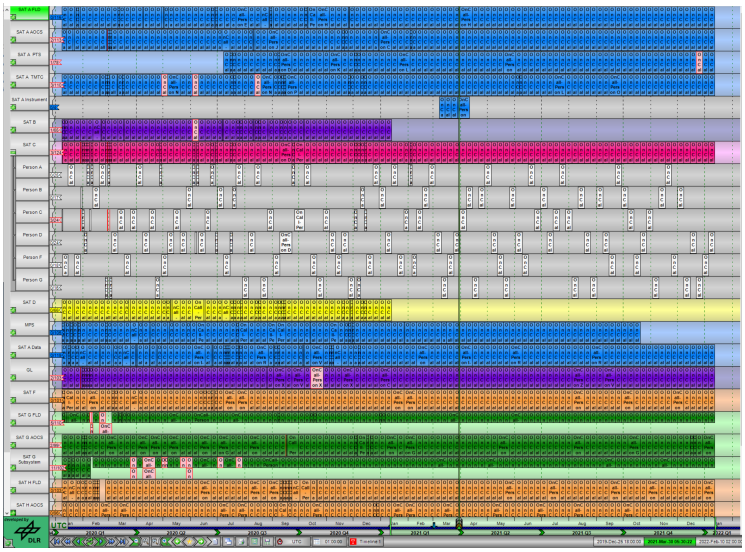


Fig. 3: A screenshot of an on-call schedule from PINTA

3 Related Work

Spacecraft On-Call Scheduling is closely related to the nurse scheduling problem, an especially hard version of the personnel scheduling problem [8]. Linear programming [1,19], simulated annealing [2,11], and tabu search [4,3] are classical approaches for calculating solutions. An overview of the complexity of various personnel scheduling variants and classical solvers is provided in [5].

It is well known that Grover's algorithm [15] can be used to solve such optimization problems with a single solution. It can further be extended to work without knowledge of the exact or multiple solutions [7] or for highly structured combinatorial search problems [17]. How to exploit the structure of NP-complete problems to perform a nested quantum search was shown in [9]. This

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

is quadratically faster than classical nested search and exponentially faster than unstructured quantum search. Iterative application of Grover's algorithm allows searching the optimum of an objective function [13], and can be used to solve constrained polynomial binary optimization problems. Due to their complexity and ubiquity, NP problems are also the focus of various other quantum algorithms. Especially for combinatorial optimization problems, the Quantum Approximate Optimization Algorithm (QAOA) [12] and the Variational Quantum Eigensolver (VQE) [22] play an important role for NISQ era devices [23]. For example, the graph coloring problem, which is similar to how we encode the scheduling problem, and the Traveling Salesman Problem have been approached with space efficient QAOA methods [24,14]. There are also approaches using quantum annealing for the nurse scheduling problem [18].

4 Encoding of Variables and Constraints

This section presents a method for encoding the SOCS problem's variables as qubits on which we apply Grover's algorithm. In contrast to a naive approach, we reduce the number of qubits by a factor of ~ 10. Additionally, we present a way to encode the three constraints into an oracle function.

4.1 Variables

As a first step, we encode the variables into Grover's input register. After the Grover iterations, the input register is measured, which results in a binary representation of a valid schedule. A naive approach represents all combinations of a day d, position p and operator o in the input state as depicted in Fig. 4a:

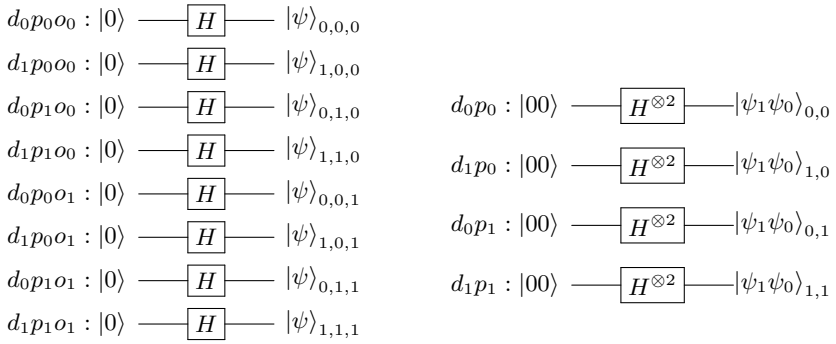
$$|\psi\rangle_{d,p,o} = \begin{cases} |0\rangle\,, & \text{operator } o \textbf{ is not} \text{ assigned to position } p \text{ at day } d \\ |1\rangle\,, & \text{operator } o \textbf{ is} \text{ assigned to position } p \text{ at day } d \end{cases}$$

This requires a quantum register with $d \cdot p \cdot o$ qubits to encode the search space. For the use case presented in Section 2, the problem volume amounts to $o \cdot p \cdot d = 50 \cdot 20 \cdot 180 = 180,000$ qubits, each of which is a binary representation of whether an operator is scheduled for a certain day and position (so-called *time-position*) or not.

Encoding values in binary can help to reduce the contribution of one variable from linear to logarithmic. As the variable *days* has the biggest impact on the number of qubits in our case with a value of 180, one may try to apply this to days, i. e. counting the days starting from one and assigning them this number in a binary representation. This way one could encode allocations of a day-position-operator combination by assigning binary encoded day values to position–operator combinations. This would reduce the number of qubits to $o \cdot p \cdot \log_2 d$ – in our use case $7,492$ qubits when encoded, *e.g.* for 4 days, as

$$|\psi_1\psi_0\rangle_{p,o} = \begin{cases} |00\rangle\,, & \text{operator } o \textbf{ is} \text{ assigned to position } p \text{ at day } \textbf{0} \\ \cdots & \\ |11\rangle\,, & \text{operator } o \textbf{ is} \text{ assigned to position } p \text{ at day } \textbf{3} \end{cases} \tag{1}$$

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2



(a) Naive, 1-hot approach to encoding (8 qubits, 2 operators)

(b) Improved approach to encoding (8 qubits, 4 operators)

Fig. 4: Input state for four time-positions and (a) two resp. (b) four operators.

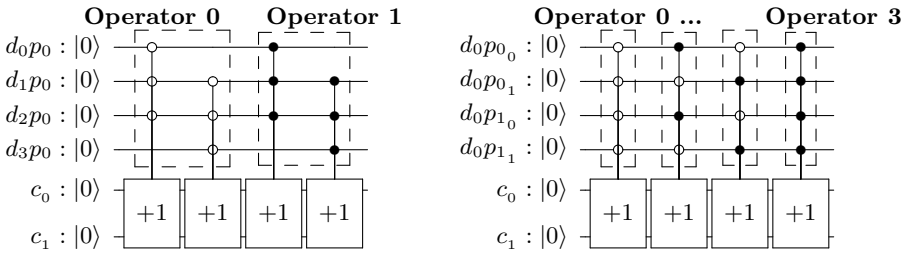
and correspondingly for larger numbers of days (powers of 2, for simplicity).

Notice, however, that one needs to be careful with such efficient encodings as they reduce the size of the representable state space and may in fact rule out valid solutions. The encoding in Eq. 1 forces every operator to work every position exactly once within the given number of days. It is easy to construct examples in which *all* solutions to the problem are removed via this encoding, e. g. two operators, one position and three days. As the number of days is larger than the available operators, one operator would need to work twice, which cannot be represented in this encoding. This problem cannot be easily fixed as there are 2^d possible subsets of all available days when an operator may work at a particular position, thus parametrizing all of them eliminates the logarithmic advantage that one gained. Notice also that parametrizing sets of scheduled days makes expressing day-wise constraints rather cumbersome.

However, in our case, constraint **A** actually says that for every day–position combination, *exactly* one operator needs to be scheduled. Therefore, encoding the *operators* instead of *days* in the above binary fashion is actually possible and only reduces the state space in a way that only invalid solutions are removed. Although the number of required qubits is not decreased as much, the circuit size can be decreased drastically, since we implement constraint **A** directly in the encoding, see Section 4.2 for details. In Fig. 4b it can be seen that with the same available amount of qubits, e. g. eight, twice as many operators, i. e. four operators instead of two, can be assigned to four time-positions. The required number of qubits for the full problem is $d \cdot p \cdot \log_2 o$ – in our case $20,318$ – and the assignment looks as follows for four operators:

$$|\psi_1\psi_0\rangle_{d,p} = \begin{cases} |00\rangle\,, & \text{operator } \mathbf{0} \text{ } \mathbf{is} \text{ assigned to position } p \text{ at day } d \\ \cdots & \\ |11\rangle\,, & \text{operator } \mathbf{3} \text{ } \mathbf{is} \text{ assigned to position } p \text{ at day } d \end{cases} \tag{2}$$

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2



(a) Constraint B for four days, two operators and one position; no operator is allowed to work 3 days in a row.

(b) Constraint C for four Operators, one day and two positions; no operator works the same day at two different positions.

Fig. 5: Exemplary circuits Constraint **B** and **C** that increment the counter $|c_1c_0\rangle$ if one of the constraints is violated.

Notice that this encoding still has some pecularities. Imagine, for example, that the total number of operators is not a power of two. In that case there will be states that schedule a non-existent operator and thus do not solve the initial problem although they may satisfy all constraints. There are various ways how to deal with this, e. g. by preparing initial states such that the corresponding amplitudes are allways zero or by considering them as operators that have an outage during every timeslot. For the current paper we will restrict to numbers of operators which are powers of two.

4.2 Constraints

A schedule is valid if none of the constraints defined in Section 2 are violated. Hence, we encode *conflicts* in the oracle function and sum up the number of conflicts for a state. Only if the amount of conflicts is zero, a state is considered a valid schedule and the Oracle function returns one.

For counting the conflicts, we use a controllable *Increment*-gate – depicted as $\boxed{+1}$ in Fig. 5. The constraints themselves are encoded as follows:

Constraint A: Assigning multiple operators to the same position at one day is impossible by construction: Since every *time-position* is encoded in an individual set of $\log_2 o$ qubits, this satisfies the "One operator per time-position" constraint automatically.

Constraint B: For the constraint that an operator is only allowed to work two days in a row, we check this constraint for gliding windows of length three days for any position and any operator. The global conflict counter register is incremented each time an operator is assigned to a position for all three consecutive days during a gliding window. Fig. 5a depicts the resulting circuit with an example of four days, two operators and one position, i. e. two gliding windows.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

Every state where $(d_0p_0 = d_1p_0 = d_2p_0) \vee (d_1p_0 = d_2p_0 = d_3p_0)$, shall increment the global conflict counter register. $\boxed{+1}$ gates are activated for all configurations of the control qubits that represent invalid *time-positions*.

This results in $p \cdot o \cdot (d-2)$ incrementors, however one can see that at most $p \cdot (d-2)$ can be activated at the same time as it is not possible to have two operators both working the same position three out of three given days due to constraint **A**.

Constraint C: Similarly, the constraint that an operator can only be assigned to at most one position per day is implemented. The idea here is that for any day and any operator we can verify that the operator is not scheduled for two positions. The global counter register is thus incremented each time an operator is assigned to more than one position per day. Fig. 5b depicts the resulting circuit with an example of one day, four operators and two positions. Each of the four operators $0-3$ is represented by one configuration of the control-qubits activating the $\boxed{+1}$ gate, which is activated if $d_0p_{00} = d_0p_{10} \wedge d_0p_{01} = d_0p_{11}$. Notice that this results in $d \cdot o \cdot \binom{p}{2}$ incrementors as we need to check on every day that each pair of distinct positions is not occupied by the same operator. However, again due to constraint **A**, for every time-position there is exactly one operator scheduled, meaning that at most $d \cdot \binom{p}{2}$ incrementers can register a constraint violation at the same time.

We can use these calculations to estimate the number of required counter qubits to avoid overflows of the counter register. Notice that such an overflow would effectively mean that the Grover oracle would mark states with particular numbers of constraint violations as valid and amplify their state correspondingly. This would mean that our amplified end results could actually be invalid if the counter register is not large enough. Combining **B** and **C**, we see that we have at most $p(d-2)+d\binom{p}{2}$ constraint violations and, since a count of zero constraint violations needs to be represented, we thus need at least $\left\lceil \log_2 \left(p(d-2) + d\binom{p}{2} + 1\right)\right\rceil$ qubits to represent this number in binary. However, depending on the constraints and the parameters it might be possible that this estimate is too crude and that it is not possible that all of these constraint violations can be achieved at the same time, so a smaller number may be sufficient.

5 Evaluation

We evaluate the solutions with Qiskit's QASM simulator provided by IBM, which simulates a noiseless quantum computer with up to 32 qubits. Due to the limitation of available qubits and a maximum simulation time, only reduced problem sizes are feasible. Table 1 summarizes our simulation of six different configurations of operators, positions and days. There are five valid (**Case I-V**) and one invalid (**Case X**) configurations, all of which are problems with $\log_2 o \cdot p \cdot d \leq 30$.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

Table 1: Simulation results for different problem sizes used for the evaluation.

Case	I	II	III	IV	V	X
Operators	4	4	4	4	8	4
Postions	2	2	2	2	2	3
Days	3	4	5	6	3	3
Used Qubits	15	21	25	29	21	23
Used Counter Qubits	2 (3)	4 (4)	4 (4)	4 (4)	2 (3)	4 (4)
Percentage Solutions	0.223	0.107	0.048	0.022	0.587	0
Grover Iterations	1	2	3	5	2	-
Success Rate	0.99	0.99	0.99	0.96	0.88	0

As an example, Case I reads as follows: Given 4 operators, 2 positions and 3 days, the number of used qubits[4] to run the algorithm is 15. The minimal number of counter qubits was empirically evaluated and maybe less than the upper bounds given in Section 4.2. The latter is given in paranthesis, e. g. for Case I $\lceil \log_2 \left(2 \cdot (3-2) + 3\binom{2}{2} + 1\right) \rceil$ is 3.

Further, the percentage of valid solutions in the overall search space is given. With an input size of 12 qubits, the number of possible schedules is $2^{12} = 4096$ of which 912 (= 22.27%) are valid schedules. The number of Grover iterations to reach the first maximum of amplitude amplification can be approximated with $\left\lfloor \frac{\pi}{4}\sqrt{\frac{1}{s}} \right\rfloor$[6], where s is the percentage of valid solutions as given in Table 1.[5] The success rates are calculated by running the algorithm 8000 times and counting measured results that are valid schedules (see e. g. , Fig. 6).

Notice that all but one of the shown cases have 2 positions, whereas one can easily see that Case X with 3 positions has no valid solution at all. The quantum algorithm verifies this true negative result, by amplifying non of the input states.

Case I to V in Table 1 show that with an increasing number of days, the number of required qubits increases. This is mostly due to constraint **B** which introduces a greater amount of conflicts if the number of days is bigger than three. The effect is also clearly visible in Fig. 7, where the success probability (y-axis) of the configuration is plotted with respect to the size of the counter register. Configurations where constraint **B** comes into play require a larger counter register to achieve a high success probability.

The correctness of the implementation is further validated by plotting success rates against a larger number of Grover iteration. The expected sinusoidal-squared behavior is clearly visible in Fig. 8

[4] The number of used qubits is given by the sum of counts of problem-encoding qubits, necessary global conflict counter register qubits and one Grover phase-flip qubit, so e. g. $\log_2 4 \cdot 2 \cdot 3 + 2 + 1 = 15$.

[5] Except for Case V, where the approximations are not valid, but the exact formula reproduces the numbers given in the table.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

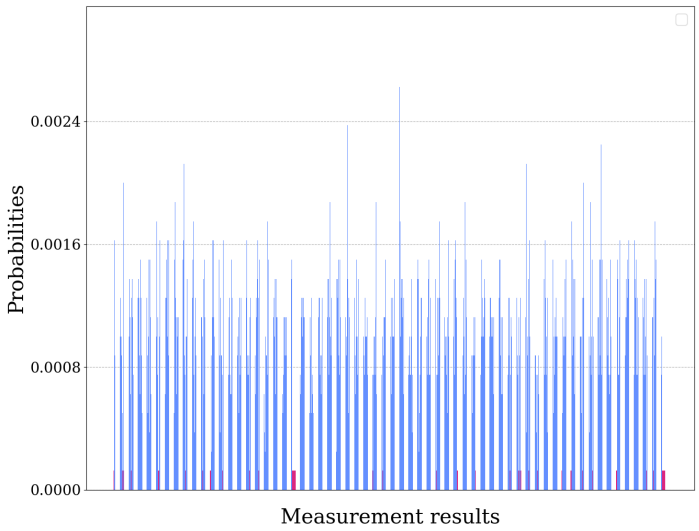


Fig. 6: Statistic for 8000 shots resulting (in)valid schedules in (red)blue for Case I.

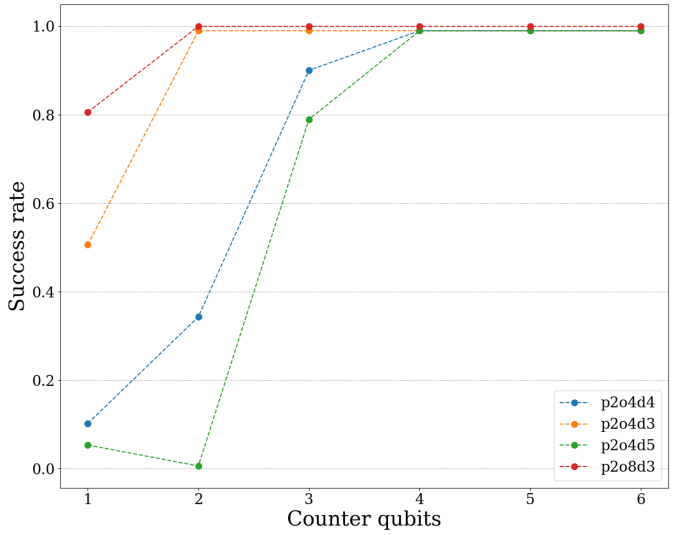


Fig. 7: Heuristical evaluation of minimal number of required counter qubits.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

Unfortunately, we were not able to test the overall circuit on IBM's real quantum devices, since the circuit depth exceeded their limits. To receive a comparable result, we introduced an error rate on the used gates which imitates those of current IBM quantum hardware to the smallest configuration. The error decreases the success probability to $\sim 33\%$, which is only slightly higher than the random distribution of $\sim 22\%$. Therefore, states with correct solutions are no longer distinguishable from a random distribution in this case. This indicates that current error rates are too high to deliver results with our approach even if we would have enough qubits to fit our problem size.

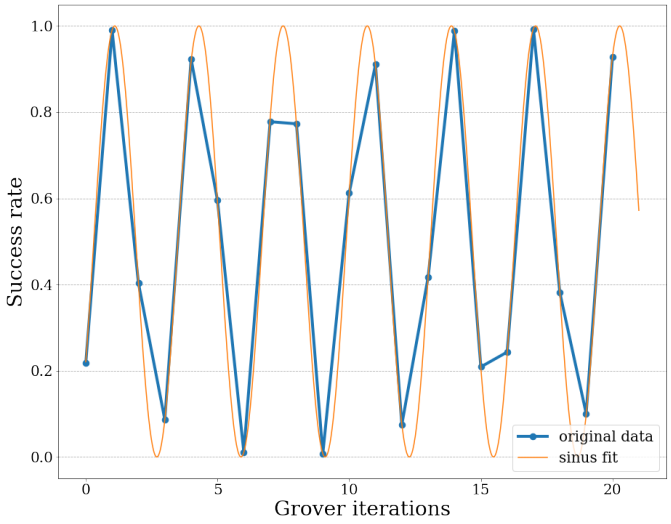


Fig. 8: Fit for expected $\sin^2$-behavior of success rate as a function of Grover iterations in Case I. Fitting parameters: $p = 1.002 \cdot \sin^2(\pi(0.313r + 1.155)) + 0.000$

6 Conclusion and Future Work

This paper successfully presents how to find a valid schedule for the "on-call spacecraft operator scheduling"-problem at the German Space Operating Center by using a Grover's quantum search algorithm. The presented encoding of variables and constraints is transferable to similar nurse-scheduling problems. Our algorithm creates a Qiskit circuit for *Grover's Oracle* with three variables of arbitrary size and complies with three selected constraints[6]. Due to the absence

[6] Including qubit initialisation and measurement operations.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

of a sufficiently large and stable quantum computer, we validated our circuits with affordable small input sizes on a quantum simulator capable of processing up to 32 qubits, by various means.

Future Work

What we showed are just the first steps towards solving a classical problem on a real quantum computer. To execute the necessary real quantum operations of the algorithms on a quantum computer, these need an improvement in qubit stability and quality of their control – or the availability of fully error-corrected qubits.

Quantum computers are currently way too small for real life problems. But the problem can be divided in parts, *e.g.* in fragments of days or weeks. This approach allows to run the quantum algorithm as a quantum subroutine within a classical framework. The latter composes the partial results from the quantum processing unit. Remark that this quantum-classical-hybrid approach asks to implement the constraint **(v)** from Section 2 as a constraint for the quantum algorithm.

While the availability of necessarily large quantum computers is pending, we will also focus on minimizing the number of qubits necessary and on reducing the circuit's width and depth with better optimizations, for example by improving conflict counting and by applying the ZX-calculus language [10,25].

References

1. Abernathy, W.J., Baloff, N., Hershey, J.C., Wandel, S.: A three-stage manpower planning and scheduling model—a service-sector example. Operations Research **21**(3), 693–711 (1973)
2. Akbari, M., Zandieh, M., Dorri, B.: Scheduling part-time and mixed-skilled workers to maximize employee satisfaction. The International Journal of Advanced Manufacturing Technology **64**(5-8), 1017–1027 (2013)
3. Bai, R., Burke, E.K., Kendall, G., Li, J., McCollum, B.: A hybrid evolutionary approach to the nurse rostering problem. IEEE Transactions on Evolutionary Computation **14**(4), 580–590 (2010)
4. Bard, J.F., Wan, L.: The task assignment problem for unrestricted movement between workstation groups. Journal of Scheduling **9**(4), 315–341 (2006)
5. Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., De Boeck, L.: Personnel scheduling: A literature review. European journal of operational research **226**(3), 367–385 (2013)
6. Boyer, M., Brassard, G., Høyer, P., Tapp, A.: Tight bounds on quantum searching. Fortschritte der Physik **46**(4-5), 493–505 (1998). https://doi.org/10.1002/(SICI)1521-3978(199806)46:4/5¡493::AID-PROP493¿3.0.CO;2-P
7. Brassard, G., Hoyer, P., Mosca, M., Tapp, A.: Quantum amplitude amplification and estimation. Contemporary Mathematics **305**, 53–74 (2002)
8. Burke, E.K., De Causmaecker, P., Berghe, G.V., Van Landeghem, H.: The state of the art of nurse rostering. Journal of scheduling **7**(6), 441–499 (2004)

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2

9. Cerf, N.J., Grover, L.K., Williams, C.P.: Nested quantum search and np-complete problems. arXiv preprint quant-ph/9806078 (1998)
10. Coecke, B., Horsman, D., Kissinger, A., Wang, Q.: Kindergarden quantum mechanics graduates (...or how I learned to stop gluing LEGO together and love the zx-calculus). CoRR **abs/2102.10984** (2021), `https://arxiv.org/abs/2102.10984.pdf`
11. Cordeau, J.F., Laporte, G., Pasin, F., Ropke, S.: Scheduling technicians and tasks in a telecommunications company. Journal of Scheduling **13**(4), 393–409 (2010)
12. Farhi, E., Goldstone, J., Gutmann, S.: A quantum approximate optimization algorithm. arXiv preprint arXiv:1411.4028 (2014)
13. Gilliam, A., Woerner, S., Gonciulea, C.: Grover adaptive search for constrained polynomial binary optimization. arXiv preprint arXiv:1912.04088 (2019)
14. Glos, A., Krawiec, A., Zimborás, Z.: Space-efficient binary optimization for variational computing (2020)
15. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Miller, G.L. (ed.) Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 212–219. ACM, New York, NY (1996)
16. Hartung, J., Nibler, R., Peat, C., Spörl, A.K., Wörle, M.T., Lenzen, C.: GSOC SoE-Editor 2.0 - A Generic Sequence of Events Tool. https://doi.org/10.2514/6.2016-2568
17. Hogg, T.: Highly structured searches with quantum computers. Physical review letters **80**(11), 2473 (1998)
18. Ikeda, K., Nakamura, Y., Humble, T.S.: Application of quantum annealing to nurse scheduling problem. Scientific reports **9**(1), 1–10 (2019)
19. Jaumard, B., Semet, F., Vovor, T.: A generalized linear programming model for nurse scheduling. European journal of operational research **107**(1), 1–18 (1998)
20. Lenzen, C., Wörle, M.T., Mrowka, F., Spörl, A., Klaehn, R.: The algorithm assembly set of plato. In: 12th International Conference on Space Operations (SpaceOps 2012) (2012), `https://elib.dlr.de/75694/`, ePoster
21. Nibler, R., Mrowka, F., Wörle, M.T., Hartung, J., Lenzen, C., Peat, C.: Pinta and timonweb – (more than) generic user interfaces for various planning problems. In: IWPSS 2017 - 10th International Workshop on Planning and Scheduling for Space (Juli 2017), `https://elib.dlr.de/114095/`
22. Peruzzo, A., et al.: A variational eigenvalue solver on a quantum processor. eprint. arXiv preprint arXiv:1304.3061 (2013)
23. Sanders, Y.R., Berry, D.W., Costa, P.C., Tessler, L.W., Wiebe, N., Gidney, C., Neven, H., Babbush, R.: Compilation of fault-tolerant quantum heuristics for combinatorial optimization. PRX Quantum **1**(2), 020312 (2020)
24. Tabi, Z., El-Safty, K.H., Kallus, Z., Hága, P., Kozsik, T., Glos, A., Zimborás, Z.: Quantum optimization for the graph coloring problem with space-efficient embedding. In: 2020 IEEE International Conference on Quantum Computing and Engineering (QCE). pp. 56–62 (2020). https://doi.org/10.1109/QCE49297.2020.00018
25. van de Wetering, J.: ZX-calculus for the working quantum computer scientist. arXiv preprint arXiv:2012.13966 (2020), `https://arxiv.org/pdf/2012.13966.pdf`
26. Wörle, M.T., Spörl, A., Hartung, J., Lenzen, C., Mrowka, F.: The Mission Planning System for the Firebird Spacecraft Constellation. In: Proceedings of the 14th International Conference on Space Operations. Daejeon, Republic of Korea (2016)

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77980-1_2