Scientific Research Publishing

Journal of Quantum Information Science, 2021, 11, 112-123
https://www.scirp.org/journal/jqis
ISSN Online: 2162-576X
ISSN Print: 2162-5751

A New Path to Create Solutions for Quantum Annealing Problems

Jose Luis Hevia[1], Ezequiel Murina[1], Guido Peterssen[1], Mario Piattini[2]

[1]aQuantum: Alhambra IT, Madrid, Spain
[2]aQuantum/University of Castilla-La Mancha, Ciudad Real, Spain
Email: jluis.hevia@alhambrait.com, ezequiel.murina@alhambrait.com, guido.peterssen@alhambrait.com, mario.piattini@uclm.es

How to cite this paper: Hevia, J.L., Murina, E., Peterssen, G. and Piattini, M. (2021) A New Path to Create Solutions for Quantum Annealing Problems. *Journal of Quantum Information Science*, **11**, 112-123.
https://doi.org/10.4236/jqis.2021.113009

Received: June 9, 2021
Accepted: September 13, 2021
Published: September 16, 2021

Abstract

Quantum computing has already become a technology to be used by large companies in finance, distribution, health care, chemistry, etc. Among the different approaches, quantum annealing is one of the most promising in the short term. However, software development platforms do not offer user-friendly interfaces for the definition of annealing problems. In this paper we present a solution to this problem: QPath®'s Annealer Compositor that facilitates the definition and execution of annealing algorithms in either quantum annealing or digital annealing computers. An example based on a nurse work schedule is used for illustrating this special interface.

Keywords

Quantum Software, Quantum Annealing, QPath, Annealer Compositor

1. Introduction

Quantum technology includes several research areas such as true random number generation, quantum information, atomic quantum clocks, quantum sensors, quantum simulators, quantum cryptography and security, quantum communication networks and quantum Internet, among others. All those topics, but quantum computing, have attracted a great deal of interest, since there are countless interesting applications in several knowledge and business areas [1]; e.g., economics and finance services, chemistry, medicine and health, supply chain and logistics, energy, and agriculture. The success of these applications of quantum technologies requires something that we take as implicit, high-quality software, so its implementation and exploitation depend on the professional development of quantum software. So, there is a need for methodologies that enable

quantum software engineers to design and implement "killer" software applications, in which the computation speed afforded by quantum computers has a real-world impact [2].

Dozens of quantum programming languages have been proposed, most of them oriented to quantum circuits (like the "classical" assembler) [3], and several quantum software development environments [4]. As for the other major approach in the field of quantum computing, quantum annealing, there are hardly any proposals to help designers specify its applications. However, it is important to offer tools to specify and design quantum software applications [5], which is essential to achieve a real "Quantum Software Engineering" [6]. Especially in view of the serious problems of availability of personnel prepared to develop quantum software [7].

Much of the existing research in the field of quantum annealing has focused on the needs of the quantum physicist or the quantum information [8] but has not considered the needs of the quantum software developer. In this article, we introduce the annealing compositor module as an interesting proposal of the QuantumPath® (QPath®) technology [9]. QPath® is a quantum software development platform to support the design, implementation, and execution of quantum software applications.

The rest of the paper is organized as follows: Section 2 summarizes the most important concepts of quantum annealing; Section 3 provides an overview of the existing quantum annealing systems; Section 4 explains the new annealing definition interface of QPath®, and Section 5 shows an example. Finally, Section 6 presents the conclusions and future work.

2. Quantum Annealing

Quantum technology is currently preferentially used to solve combinatorial optimization problems such as, for example, portfolio management, scheduling challenges, optimization, etc. These problems are usually formulated as QUBO (Quadratic Unconstrained Binary Optimization) problems [10], which allows them to be solved very efficiently. These types of models are used with digital annealing and quantum annealing. Digital or simulated annealing is a metaheuristic search algorithm for global optimization problems; the general objective of this type of algorithm is to find a good approximation to the (global) optimal value of a function in a large search space. The name of "annealing" is because its similarity with the annealing process of steel and ceramics, a technique that consists of heating and then slowly cooling the material to vary its physical properties. In the case of quantum annealing, quantum tunneling effect decreases the computational sampling time compared to the thermal fluctuation effect of simulated annealing.

One of the first proposal of quantum annealing was from Kadowaki and Nishimori in 1998 [11], where the authors "introduce quantum fluctuations into the simulated annealing process of optimization problems, aiming at faster con-

vergence to the optimal state", they tested this idea in the transverse Ising model obeying the time-dependent Schrödinger equation.

3. Annealing Computers

Ising machine systems could be divided into quantum annealing (based on superconducting circuit, such as D-Wave systems) and digital annealing (based on digital circuits, such as Fujitsu Digital Annealer).

Fujitsu Digital Annealer [12] "solves large-scale complex combinatorial optimization problems in near real-time. Delivered as an end-to-end service, it can be deployed anywhere from cloud to on-premises". This type of systems allows to develop a "quantum-inspired" algorithm or solution; reducing the time required to solve combinatorial optimization problems. Fujitsu offers an end-to-end solution starting with their expert engineers understanding the business problem and formulating and converting these problems into QUBOs.

D-Wave Systems [13] represent the quantum approach, in which "a user maps a problem into a search for the 'lowest point in a vast landscape', corresponding to the best possible outcome. The quantum processing unit considers all the possibilities simultaneously to determine the lowest energy required to form those relationships. The solutions are values that correspond to the optimal configurations of qubits found, or the lowest points in the energy landscape. These values are returned to the user program over the network". D-Wave facilitates quantum application development using open-source Ocean software development kit, based on Python.

It should be noted that, in addition to D-Wave and Fujitsu, the annealing approach increasingly highlights the proposals of IARPA [14], NEC [15] and Qilimanjaro [16].

4. A new Annealing Definition Interface

QPath®, is a quantum software development and lifecycle application platform, based on a hybrid model for the construction of services that abstract quantum technology without having to worry directly about the manufacturers' platforms and their requirements. QPath® rigorously applies the principles of quantum computing and programming, the good practices in Quantum Software Engineering and Programming [5] and, of course, the requirements defined by manufacturers for their ecosystems. So that QPath guaranties truly agnostic quantum software development, the compatibility with the ecosystems of different quantum providers, and full and ongoing interoperability with them.

QPath® support quantum annealing algorithm design, deployment, and execution, providing a set of abstractions, tools, scalable executions, and unified results that simplifies the development stage with different supported platforms (see **Figure 1**). So, all this software technology is agnostic to a specific current and future quantum hardware.

Another of QPath®'s basic principle is to facilitate the quantum algorithm

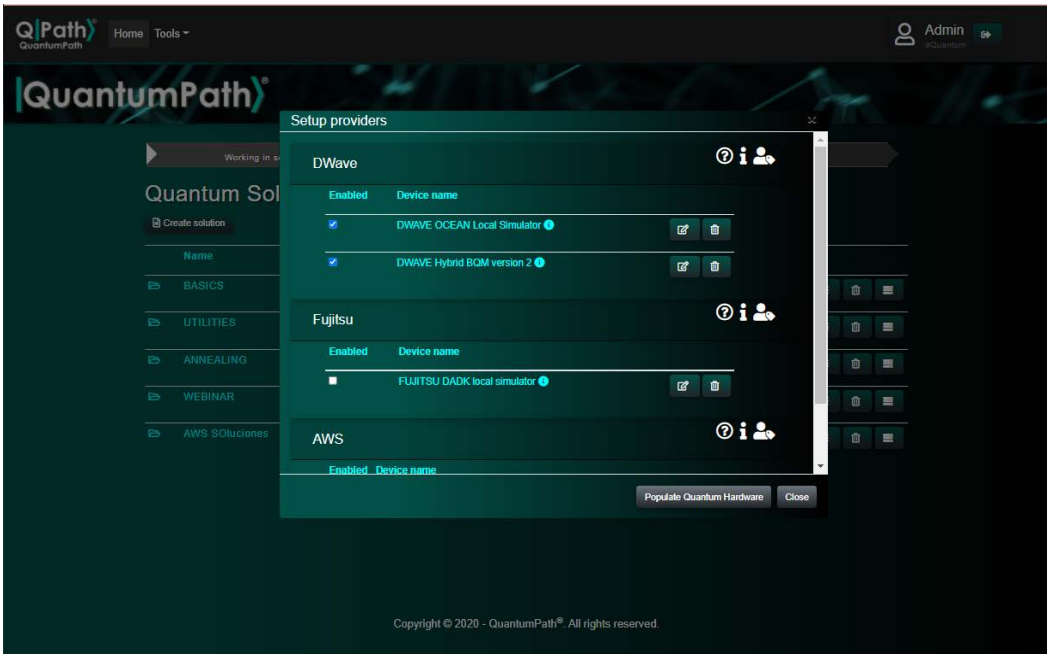

Figure 1. Different quantum platforms supported by QPath®.

design process by means of visual interfaces that guide the specialists (engineers, scientists…) from the creation of the quantum algorithm through its development, testing and implementation, to its deployment and reuse. Sometimes, especially in quantum annealing, these tasks are too arid or required a very specialized knowledge and experience. To facilitate the quantum designer task, QPath® offers special quantum annealer interface, as show in **Figure 2**.

5. Example of Quantum Annealing Definition

We will illustrate quantum annealer compositor features, using the problem of work schedule of nurses in a hospital. It is a rank 2 optimization problem, in which the goal is to find an optimal work assignment for a group of nurses, under a set of constraints:

- No nurse should work two consecutive days.
- There should be only one nurse working per shift.
- The nurses must work a similar number of shifts.

For this example, the number of nurses will be set to 3, the number of days 10 and the approximate number of days each nurse should work 3. This problem can be modeled with a combination of two sets of variables: nurses and days, with the actual number of variables in the problem being the Cartesian product of nurses x days: $Z_{\text{nurse,day}}$.

The Hamiltonians associated with each constraint are as follows:

- No nurse should work two consecutive days

$$\sum_n \sum_d a z_{n,d} z_{n,d+1}$$

(where a is a fixed coefficient of value 1).

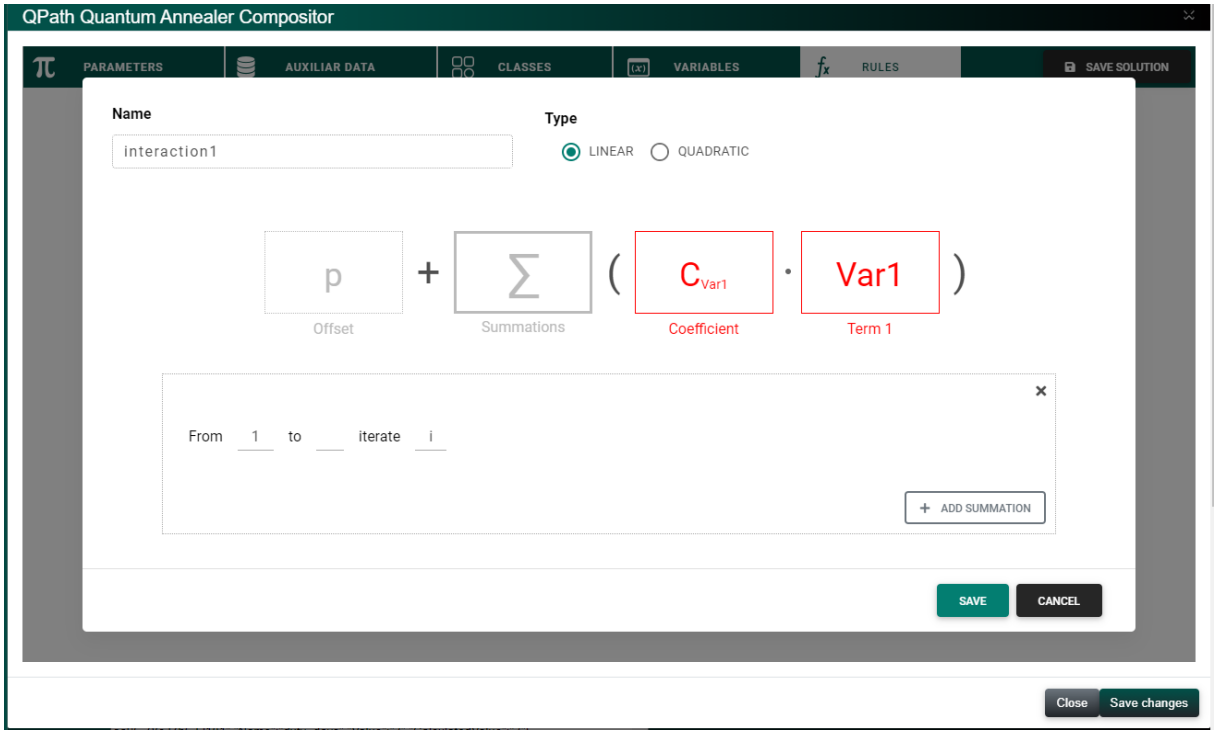

Figure 2. QPath®'s annealer compositor interface.

- There should be only one nurse working per shift

$$\text{days} - \sum_d \sum_n z_{n,d} + \sum_d \sum_n \sum_{m>n} z_{n,d} z_{m,d} \ .$$

- Nurses are required to work a similar number of shifts

$$(1 - 2\text{duty\_days}) \sum_n \sum_d z_{n,d} + 2 \sum_n \sum_d \sum_{m>d} z_{n,d} z_{n,m} + \text{days} * \text{duty\_days}^2 .$$

(where duty_days is the desired number of days each nurse is expected to work)

To implement the problem in QuantumPath, a new circuit of type "QuantumAnnealing" was created with the name "Nurses Calendar". We edit the circuit with the Annealer Composer, with which we shaped the problem by defining the elements shown in **Figure 3**.

The variable classes are defined as shown in **Figure 4**. Also, the three required rules are defined using the Quantum Annealer Compositor. **Figure 5** shows for example how to define rule 2, that specifies that "there should be only one nurse working per shift" (**Figure 6**).

Once all the parameters, variables and rules have been defined, we proceed to compilation and transpilation. The compilation of the solution, according to a specific syntax, produces a representation that formalize the problem:

```
PARAM(days, 10);
PARAM(duty_days, 3);
PARAM(nurses, 3);
CLASS(Nurses, n, nurses, "Enfermeras");
CLASS(Days, d, days, "Días del calendario");
RULE(Rule1, "No two consecutive days of work for each nurse");
```

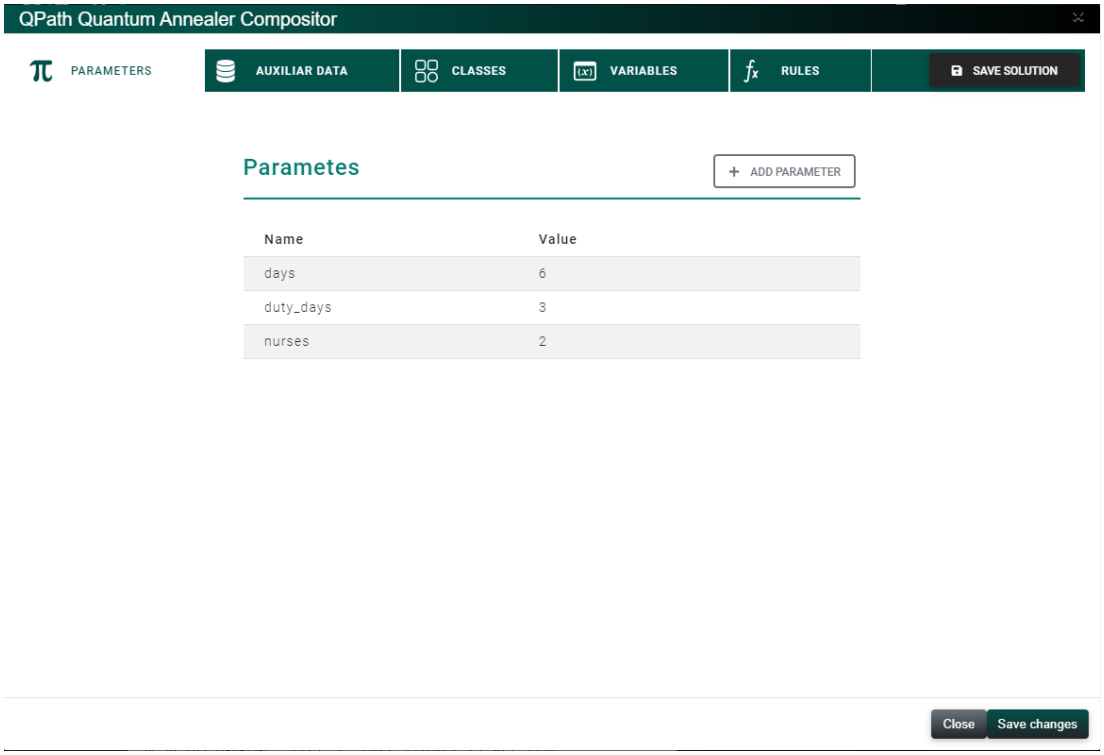

Figure 3. Parameters for quantum circuit to solve the nurse example.

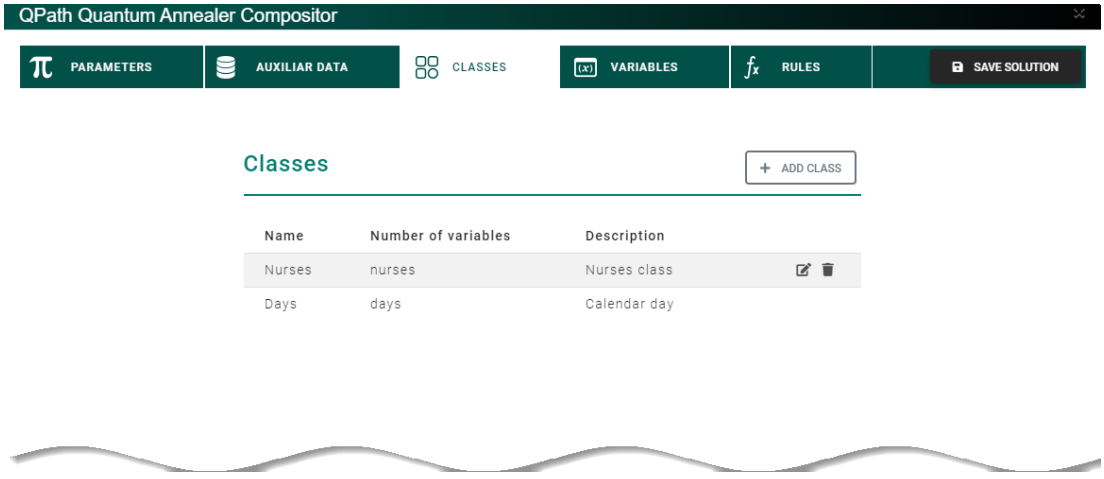

Figure 4. Variable classes to solve the nurse example.

```
INTERACTION(Rule1, Interaction1,
  QUADRATIC(
        {from 0 to Nurses pass 1 iterator n, from 0 to Days - 1 pass 1 iterator d},
        {Nurses[n], Days[d]},
        {Nurses[n], Days[d+1]},
        1,
        0
  )
);
```

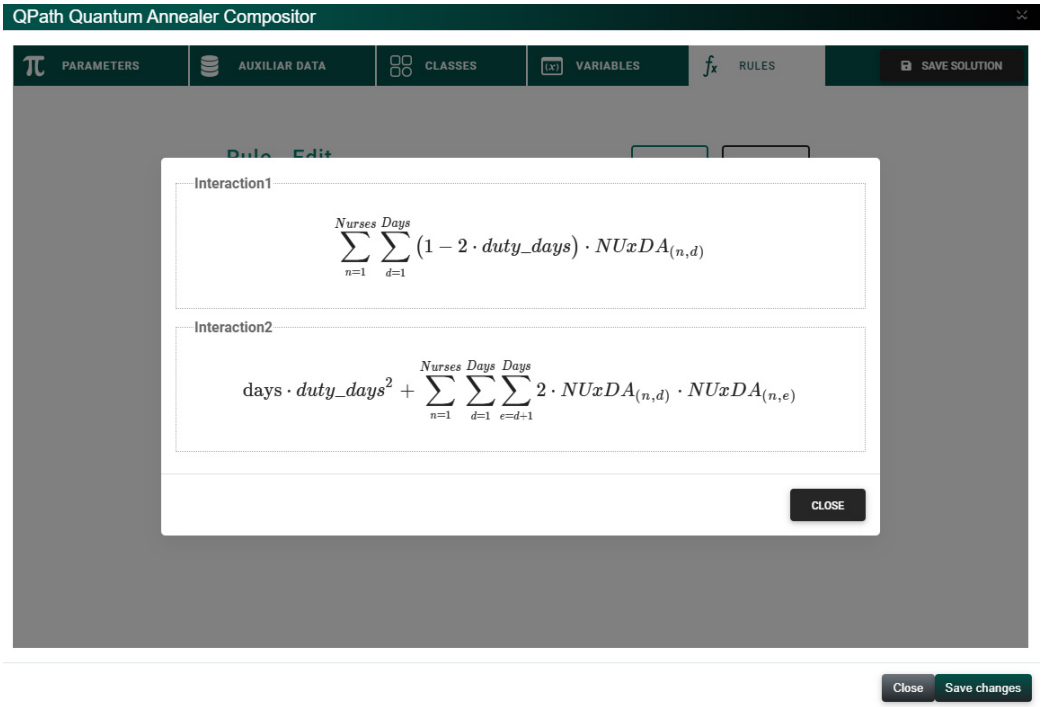

Figure 5. Interactions to solve the nurse example.

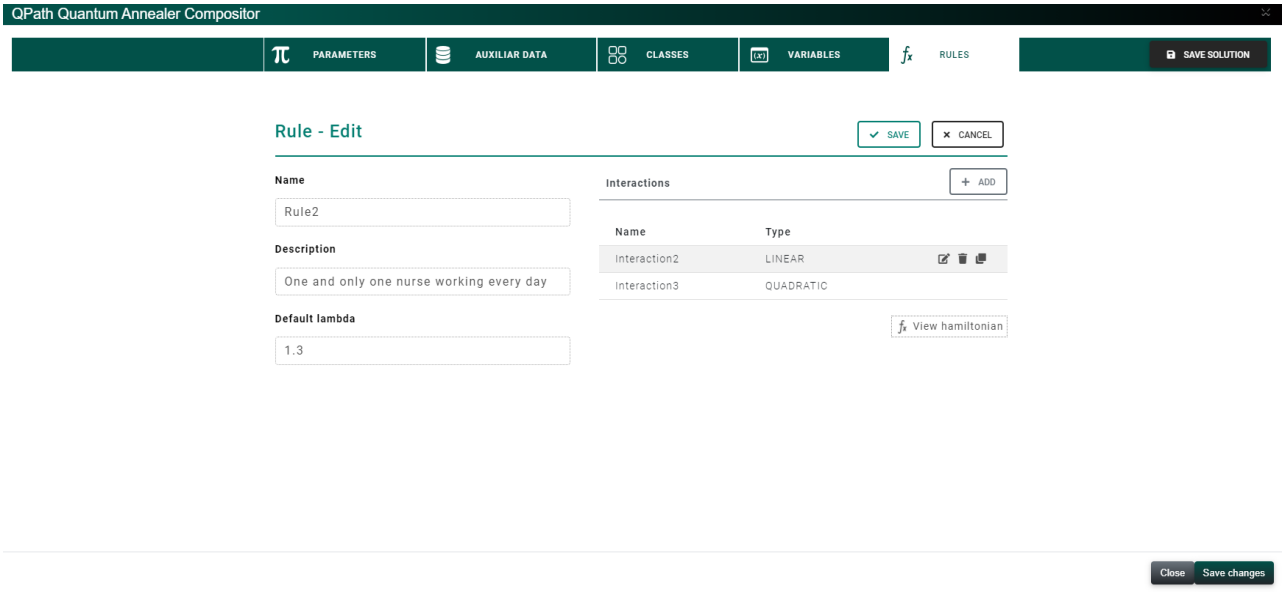

Figure 6. Rules to solve the nurse example.

```
LAMBDA(Rule1, 3.5);
RULE(Rule2, "One and only one nurse working every day");
INTERACTION(Rule2, Interaction2,
  LINEAR(
      {from 0 to Days pass 1 iterator d, from 0 to Nurses pass 1 iterator n},
      {Nurses[n], Days[d]},
      -1,
```

```
            days
        )
    );
    INTERACTION(Rule2, Interaction3,
        QUADRATIC(
            {from 0 to Days pass 1 iterator d, from 0 to Nurses pass 1 iterator n,
from n+1 to Nurses pass 1 iterator m},
            {Nurses[n], Days[d]},
            {Nurses[m], Days[d]},
            1,
            0
        )
    );
    LAMBDA(Rule2, 1.3);
    RULE(Rule3, "All nurses should have approximately even work schedules");
    INTERACTION(Rule3, Interaction1,
        LINEAR(
            {from 0 to Nurses pass 1 iterator n, from 0 to Days pass 1 iterator d},
            {Nurses[n], Days[d]},
            1-2*duty_days,
            0
        )
    );
    INTERACTION(Rule3, Interaction2,
        QUADRATIC(
            {from 0 to Nurses pass 1 iterator n, from 0 to Days pass 1 iterator d,
from d+1 to Days pass 1 iterator e},
            {Nurses[n], Days[d]},
            {Nurses[n], Days[e]},
            2,
            days*duty_days^2
        )
    );
    LAMBDA(Rule3, 0.3);
```

Transpilation of the above metalanguage to Python, for execution on the D-Wave machine will generate the necessary artifacts, modules, and code to be executed into the final hardware. As an example, and using python to demonstrate what occurs, the code needed to implement the rules could be something like this:

```
    ...
    for n1 in range (0, len(Nurses), 1):
        for d1 in range (0, len(Days), 1):
            for n2 in range (n1, len(Nurses), 1):
                for d2 in range (d1, len(Days), 1):
```

```
                    var1 = Nurses[n1]+ "_" +Days[d1]
                    var2 = Nurses[n2]+ "_" +Days[d2]
                    Q[(var1, var2)] = 0
##Rule1: No two consecutive days of work for each nurse
...
for n in range (int(0), int(len(Nurses)), 1):
  for d in range (int(0), int(len(Days) - 1), 1):
      var1 = Nurses[int(n)]+ "_" + Days[int(d)]
      var2 = Nurses[int(n)]+ "_" + Days[int(d+1)]
      if((var1, var2) in Q):
          Q[(var1, var2)] += lambda_Rule1*(1)
##Rule2: One and only one nurse working every day
...
for d in range (int(0), int(len(Days)), 1):
  for n in range (int(0), int(len(Nurses)), 1):
      var = Nurses[int(n)]+ "_" + Days[int(d)]
      Q[(var, var)] += lambda_Rule2*(-1)
for d in range (int(0), int(len(Days)), 1):
  for n in range (int(0), int(len(Nurses)), 1):
      for m in range (int(n+1), int(len(Nurses)), 1):
          var1 = Nurses[int(n)]+ "_" + Days[int(d)]
          var2 = Nurses[int(m)]+ "_" + Days[int(d)]
          if((var1, var2) in Q):
              Q[(var1, var2)] += lambda_Rule2*(1)
##Rule3: All nurses should have approximately even work schedules
...
for n in range (int(0), int(len(Nurses)), 1):
  for d in range (int(0), int(len(Days)), 1):
      var = Nurses[int(n)]+ "_" + Days[int(d)]
      Q[(var, var)] += lambda_Rule3*(1-2*duty_days)
...
for n in range (int(0), int(len(Nurses)), 1):
  for d in range (int(0), int(len(Days)), 1):
      for e in range (int(d+1), int(len(Days)), 1):
          var1 = Nurses[int(n)]+ "_" + Days[int(d)]
          var2 = Nurses[int(n)]+ "_" + Days[int(e)]
          if((var1, var2) in Q):
              Q[(var1, var2)] += lambda_Rule3*(2)
...
```

QPath® transparently creates all the assets that are needed to the execution, taking care about all the things needed to create complete modules that covers the business layers needs. We chose in the runtime dashboard the specific quantum machine or simulator that acts as an annealing solver. And in the QPath®'s execution dashboard we can see the results of this example (**Figure 7**).

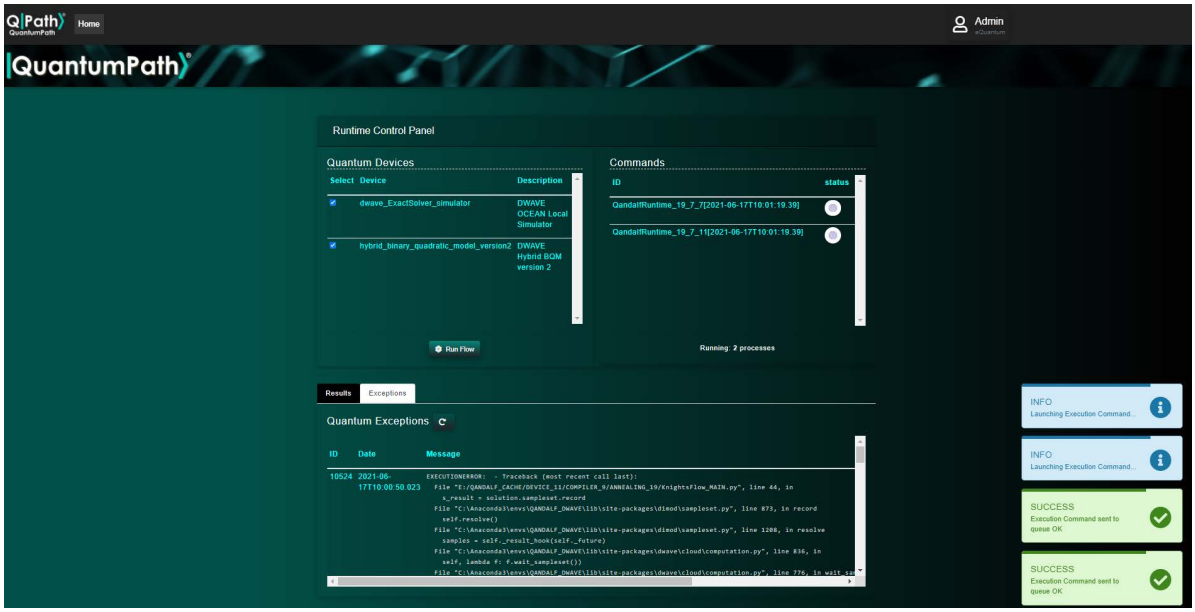

Figure 7. QPath®'s execution dashboard.

6. Conclusions and Future Work

Quantum computing offers us the possibility of initiating a new software engineering golden age [17], to do so, it is necessary to devise techniques and tools that support the software engineer's task of defining quantum applications.

The agnostic quality of QPath® makes it an accelerator of quantum software development because, through a hybrid model of building services that abstract quantum technology, allowing developers to be unconcerned about manufacturers' platforms and their necessary requirements. They can focus on the knowledge and leave the details to the QPath® platform. In fact, thanks to its architecture, QPath® can incorporate new target quantum computers and simulators and migrate transparently to the user all the code and applications developed.

In this way, to develop quality algorithms with QPath®, it is not necessary to manually program any line of code, in any programming language. The system takes care of this, so the time of the construction phase is reduced so drastically that its weight in the project life cycle is minimal. As a result, significant savings are recorded in project development times, with a corresponding positive impact on productivity and in production and maintenance costs.

In the same way, these characteristics of QPath® have a positive effect on the learning curve, with the corresponding positive effect on costs, eliminating the need to know the technological specificities of the different quantum providers.

As future work we are creating a set of Apps that support Software Engineering and Programming best practices adoption and allow to assure the quality of quantum development projects.

Acknowledgements

This work is part of "QHealth: Quantum Pharmacogenomics Applied to Aging",

2020 CDTI Missions Programme (Center for the Development of Industrial Technology of the Ministry of Science and Innovation of Spain)/FEDER and GEMA (SBPLY/17/180501/000293) funded by the Dirección General de Universidades, Investigación e Innovación—Consejería de Educación, Cultura y Deportes; Gobierno de Castilla-La Mancha. We would like to thank all the aQuantum members for their help and support.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] IDB (2019) Quantum Technologies. Digital Transformation, Social Impact, and Cross-Sector Disruption. Interamerican Development Bank. https://publications.iadb.org/publications/english/document/Quantum_Technologies_Digital_Transformation_Social_Impact_and_Crosssector_Disruption.pdf

[2] Mueck, L. (2017) Quantum Software. *Nature*, **549**, 171. https://doi.org/10.1038/549171a

[3] Piattini, M. (2021) Requirements for a Robust Quantum Software Development Environment. *Cutter Business Technology Journal*, **34**, 12-17.

[4] Hevia, J.L., Peterssen, G., Ebert, C. and Piattini, M. (2021) Quantum Software Development Toolkits. IEEE Software Sept/Oct.

[5] Piattini, M., Peterssen, G., Pérez-Castillo, R., Hevia, J.L., *et al.* (2020) The Talavera Manifesto for Quantum Software Engineering and Programming. QANSWER 2020 QuANtum SoftWare Engineering & Programming. *Proceedings of the* 1*st International Workshop on the QuANtum SoftWare Engineering & Programming,* Talavera de la Reina, Spain, 11-12 February 2020, 1-5. http://ceur-ws.org/Vol-2561/paper0.pdf

[6] Piattini, M., Serrano, M., Pérez-Castillo, R., Peterssen, G. and Hevia, J.L. (2021) Towards a Quantum Software Engineering. *IT Professional*, **23**, 62-66. https://doi.org/10.1109/MITP.2020.3019522

[7] Peterssen, G. (2020) Quantum Technology Impact: The Necessary Workforce for Developing Quantum Software. *Proceedings of the* 1*st International Workshop on the QuANtum SoftWare Engineering & Programming* (QANSWER), Talavera de la Reina, Spain, 11-12 February 2020, 6-22.

[8] Corcoles, A.D., Kandala, A., Javadi-Abhari, A., McClure, D.T., Cross, A.W., Temme, K., Nation, P.D., Steffen, M. and Gambetta, J.M. (2019) Challenges and Opportunities of Near-Term Quantum Computing Systems. *Proceedings of the IEEE*, **108**, 1338-1352. https://doi.org/10.1109/JPROC.2019.2954005

[9] QuantumPath (2021) https://www.quantumpath.es/

[10] Glover, F., Kochenberger, G. and Du, Y. (2019) A Tutorial on Formulating and Using QUBO Models. arXiv: 1811.11538. https://arxiv.org/abs/1811.11538

[11] Kadowaki, T. and Nishimori, H. (1998) Quantum Annealing in the Transverse Ising Model. *Physical Review E*, **58**, 5355. https://doi.org/10.1103/PhysRevE.58.5355

[12] Fujitsu (2021) Fujitsu Digital Annealer. https://www.fujitsu.com/global/services/business-services/digital-annealer/index.ht

ml

[13] D-Wave (2021) D-Wave System. https://www.dwavesys.com/

[14] IARPA (2021) Quantum Enhanced Optimization (QEO). https://www.iarpa.gov/index.php?option=com_content&view=article&id=564&Itemid=339

[15] NEC (2021) Using Quantum Computers to Solve Complex Social Issues. https://www.nec.com/en/global/quantum-computing/index.html

[16] Qilimanjaro (2021) https://www.qilimanjaro.tech/technology/

[17] Piattini, M., Peterssen, G. and Pérez-Castillo, R. (2020) Quantum Computing: A New Software Engineering Golden Age. *ACM SIGSOFT Software Engineering Notes*, **45**, 12-14. https://doi.org/10.1145/3402127.3402131