IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 4 December 2024, accepted 19 December 2024, date of publication 2 January 2025, date of current version 8 January 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3525620

RESEARCH ARTICLE

D-Wave's Nonlinear-Program Hybrid Solver: Description and Performance Analysis

ENEKO OSABA AND PABLO MIRANDA-RODRIGUEZ
TECNALIA, Basque Research and Technology Alliance (BRTA), 48160 Derio, Spain

Corresponding author: Eneko Osaba (eneko.osaba@tecnalia.com)

This work was supported by the Basque Government through ELKARTEK Program under Grant KK-2024/00105 (KUBIT project) and Grant KK-2024/00024 (AURRERA project).

ABSTRACT The development of advanced quantum-classical algorithms is among the most prominent strategies in quantum computing. Numerous hybrid solvers have been introduced recently. Many of these methods are created ad hoc to address specific use cases. However, several well-established schemes are frequently utilized to address optimization problems. In this context, D-Wave launched the *Hybrid Solver Service* in 2020, offering a portfolio of methods designed to accelerate time-to-solution for users aiming to optimize performance and operational processes. Recently, a new technique has been added to this portfolio: the *Nonlinear-Program Hybrid Solver*. This paper describes this solver and evaluates its performance through a benchmark of 45 instances across three combinatorial optimization problems: the Traveling Salesman Problem, the Knapsack Problem, and the Maximum Cut Problem. To facilitate the use of this relatively unexplored solver, we provide details of the implementation used to solve these three optimization problems.

INDEX TERMS Quantum computing, hybrid quantum-classical computing, quantum annealing, D-Wave.

I. INTRODUCTION

The emergence of quantum technology is expected to have a significant impact on a number of industries. The field of Quantum Computing (QC), which leverages the principles of quantum mechanics to process information, is constantly making advances to connect quantum processing with practical use cases. Thus, QC has advanced significantly in recent years, mainly due to the rapid development of technology and advancements in its democratization, understanding it as the process of making QC accessible to a broader spectrum of individuals and communities [1], [2]. As a result, QC has facilitated the development of various proofs of concept across multiple sectors, including finance [3], energy [4], and logistics [5].

Historically, the idea of a quantum computer can be traced back to the works of Benioff [6] and Feynman [7], where they proposed a quantum mechanical implementation of a Turing machine and the idea of simulating a quantum system with another quantum system, respectively.

Currently, there are two types of real quantum devices that we can differentiate between: gate-based quantum computers and quantum annealers. A gate-based system, on the one hand, employs basic quantum circuit operations on qubits, which are akin to the classical operations on regular bits and may be combined in any order to create algorithms. This version is often referred to as *a universal quantum computer*. On the other hand, a quantum annealer is based on adiabatic computation, wherein an initial, easily prepared Hamiltonian is gradually and continuously evolved from its ground state to the ground state of a final, problem-specific Hamiltonian. If the evolution is slow enough, the adiabatic theorem guarantees that the system remains in the ground state during the whole computation. In quantum annealers, the adiabatic theorem is intentionally relaxed, allowing the system to evolve more rapidly than the adiabatic limit would dictate. As a result, transitions to higher energy states usually occur during the evolution; however, alternative methods to achieve adiabaticity have been proposed [8], [9]. Although this model of computation is also universal [10], D-Wave quantum annealer is based on an Ising Hamiltonian, restricting the types of problems that can be run on the system.

The associate editor coordinating the review of this manuscript and approving it for publication was Wei E. I. Sha.

This type of quantum annealer is, however, well suited to solve combinatorial optimization problems [11].

Despite all the progress made in the field, quantum computers are still in their infancy in contrast to classical computers, which have been developed for decades and are therefore highly advanced. Thus, quantum devices are currently unable to efficiently solve real-world problems mainly because of the small number of qubits and their unstable nature. Phenomena such as decoherence time, noise and information loss, in absence of error correction protocols, impact in the performance of the computation. Besides, there are other roadblocks such as quantum gate fidelity and gate noise. At this time, the number of qubits for both universal quantum computers and quantum annealers is around three orders of magnitude (thousands of qubits) [12], [13]. However, this number needs to be much greater in order for the technology to become truly useful for real industrial use cases. In any case, there are factors other than the qubit count that influence the practical capabilities of quantum computers, such as the connectivity topology or the previously mentioned operational fidelity and decoherence time.

As a result of this situation, recent advancements have emerged during the *noisy intermediate-scale quantum* (NISQ, [14]) era, a period characterized by quantum computers' limitations in efficiently handling problems, even those of small to medium size. As it turns out, both universal quantum devices and annealers suffer from these limitations.

As a consequence, the entire community is striving to come up with mechanisms to deal with the present limitations and capitalize on the promise that QC has to offer. The design of advanced hybrid algorithms, which combine the advantages of both computing paradigms, are among the most popular strategies [15]. Arguably, hybrid quantum computing represents the immediate future of this area. This is so because the adoption of quantum techniques to address real-world use cases is heavily reliant on hardware capabilities. In this regard, just as QC should not be thought of as a direct replacement for conventional computing, it would also be a mistake to view quantum-classical hybrid computing as merely a temporary fix to minimize the limitations of NISQ-era systems. As stated in works such as [16], hybrid algorithms will be influential well beyond the NISQ-era and even into full fault tolerance, with quantum computers enhancing the capabilities of already powerful classical processors by carrying out certain specialized tasks. The challenge here is in determining how to integrate classical and quantum computing to create a synergy that surpasses the performance of purely classical approaches.

Besides, hybrid algorithms have its own limitations. For example, since quantum and classical hardware are often physically separated and they need to share information, latency is introduced in the workflow [17]. Another roadblock is the possibility of barren plateaus in the optimization landscape, making it difficult for classical optimizers to converge [18], [19]. Also, when the problem at hand exceeds the computational capacity of the quantum processing unit (QPU), determining which subproblems to allocate to the quantum component of the algorithm is a non-trivial task [20], [21]. The choice of methodology is not always straightforward and often depends on the specific characteristics of the problem and the quantum hardware's limitations, for example its topology.

It is also crucial to note that the design and implementation of quantum algorithms could necessitate a high degree of subject matter expertise. This intricacy could be a hindrance for researchers who lack a sufficient background in disciplines like physics or quantum mechanics. In order to remove that barrier and make QC more accessible, various frameworks and programming languages are proposed, such as Silq [22], Eclipse Qrisp [23], or Qiskit [24]. The availability of such frameworks and languages helps to foster the building of a multidisciplinary community focused on QC and helps the field to progress toward new horizons [25].

In this article, we focus on the hybrid algorithms and frameworks proposed by the Canadian company D-Wave Systems. Anyway, it is worth noting that significant efforts have been made from various fronts in the design and implementation of both hybrid techniques and platforms. Probably the most well-known and widely studied hybrid resolution schemes are the Variational Quantum Algorithms (VQA, [26])), oriented towards gate-based quantum computing. The most representative examples of VQA are the Quantum Approximate Optimization Algorithm (QAOA, [27]) and the Variational Quantum Eigensolver (VQE, [28]). In this context, it is also interesting to highlight commercial approaches such as the so-called `HybridSolver` developed by the German company Quantagonia.[1] Regarding frameworks and platforms specifically created to facilitate the design, implementation, and execution of hybrid algorithms, notable examples include NVIDIA's CUDA-Q,[2] IBM Qiskit Runtime[3] and Google's Cirq.[4]

Focusing our attention on D-Wave, and as part of the strategy to bring QC to a wider audience, *D-Wave-Hybrid-Framework* was published in 2018, which, in the words of its creators, is "*a general, minimal Python framework to build hybrid asynchronous decomposition samplers for QUBO problems*".[5] This framework is appropriate for "*developing hybrid approaches to combining quantum and classical compute resources*".

Some further efforts were made by D-Wave to bring hybrid solvers to industry. Thus, in 2020, *D-Wave's Hybrid Solver Service* (HSS, [29]) was launched, which consists of a portfolio of hybrid solvers that leverage quantum and classical computation to tackle large and/or real-world optimization problems. HSS is tailored for researchers and

[1] https://www.quantagonia.com/hybridsolver
[2] https://developer.nvidia.com/cuda-q
[3] https://docs.quantum.ibm.com/api/qiskit-ibm-runtime/runtime_service
[4] https://quantumai.google/cirq
[5] https://github.com/dwavesystems/dwave-hybrid

TABLE 1. Brief survey on recent practical applications of HSS solvers, classified by the field of knowledge on which the research is focused.

Field	BQM-Hybrid	DQM-Hybrid	CQM-Hybrid
Logistics	[43], [44]		[47], [52]
Medicine	[31]–[33]	[56]	[48]
Energy	[35], [55]	[55]	[35], [51]
Industry	[36]–[39]	[39]	[38], [39], [46], [49], [50], [53]
Finance	[40]–[42]		[42], [54]
Others	[34], [45], [58]	[57]	

practitioners aiming to streamline the code development process. Consequently, all solvers in the portfolio are designed to enhance time-to-solution, aiding users in optimizing performance and operational workflows.

Until June 2024, the HSS included three solvers for solving three different problem types [30]: the binary quadratic model (BQM) solver, `BQM-Hybrid`, for problems defined using binary variables; the discrete quadratic model (DQM) technique, `DQM-Hybrid`, for problems defined on discrete values; and the constrained quadratic model (CQM) method, `CQM-Hybrid`, which can deal with problems defined on binary, integer, and even continuous variables. Recently, a new solver to be added to the portfolio has been unveiled: the *Nonlinear-Program Hybrid Solver*, or `NL-Hybrid`.

As we have summarized systematically in Table 1, much research has been carried out in recent years around `BQM-Hybrid`, `DQM-Hybrid`, and `CQM-Hybrid`. Among these methods, `BQM-Hybrid` is the one most frequently used by the scientific community. This is because it was the first to be implemented and its usage is similar to the D-Wave QPU. It is important to highlight that, precisely for this reason, the `BQM-Hybrid` is not well-suited for solving real-world problems with many constraints. Consequently, there are few studies where this method has been used to solve problems with realistic aspects [31], [32], [33], [34]. With all this, the `BQM-Hybrid` has been primarily used as a benchmarking algorithm [35], [36], [37], [38], [39], [40], [41], [42]. That is, as a method employed within an experimentation to measure the performance of other methods. In addition to this, `BQM-Hybrid` has also been used to solve academic problems, such as Tail Assignment Problem [43], Vehicle Routing Problem [44] and the Set Packing Problem [45].

Focusing on the `CQM-Hybrid`, we see how this advanced solver has a clear orientation towards solving real-world problems. The `CQM-Hybrid` has been used in a significant number of works to address complex problems with a high number of constraints, and it has been employed in different contexts: to solve a complete problem [35], [39], [46], [47], [48], [49], [50]; as part of a complex resolution pipeline where it is called a single time [51] or iteratively [52]; or to implement mechanisms to improve the performance of existing classical algorithms [53]. Furthermore, the `CQM-Hybrid` has been scarcely used as a benchmarking algorithm [42], [54].

Regarding the `DQM-Hybrid`, it is the HSS method that has received the least attention from the community. This is mainly because it is less flexible and powerful compared to `CQM-Hybrid`, leading researchers to clearly favor the latter. Thus, the `DQM-Hybrid` has been used in only a handful of papers, either to act as a benchmarking method [39] or to solve clustering and community detection problems [55], [56], [57].

Despite this abundant scientific activity, no work has yet been published on `NL-Hybrid`. Motivated by the lack of existing research, this paper focuses on describing the newly introduced `NL-Hybrid` solver. Furthermore, we will conduct an experiment to analyze the performance of this new solver in comparison with `BQM-Hybrid`, `CQM-Hybrid`, and D-Wave's QPU `Advantage_system6.4`. For these tests, we used a benchmark composed of 45 instances, equally distributed across three combinatorial optimization problems: the Traveling Salesman Problem (TSP, [59]), the Knapsack Problem (KP, [60]), and the Maximum Cut Problem (MCP, [61]). We have chosen these problems because:

- They have been extensively used for benchmarking purposes in QC-oriented research [5], [62], [63].
- They are appropriate for formulation and use in the solvers considered in this study. Furthermore, TSP and KP are well suited to be solved with `NL-Hybrid` newly introduced variables, while MCP can be easily formulated with traditional binary variables.
- Their complexity for being solved by QC-based methods has been previously demonstrated in the literature [64], [65], [66].

Finally, to facilitate the use of this still unexplored solver, we will share details of the implementation to solve these three optimization problems.

This research complements previous work on hybrid algorithms carried out by the authors of this study. In works such as [67] and [68], the authors presented ad hoc implemented hybrid algorithms to solve the TSP and its asymmetric variant. Furthermore, in [47], [52], [53], and [46], the authors used the `CQM-Hybrid` to tackle real-world problems in fields such as logistics and industry. Finally, the authors have also conducted research focused on benchmarking different hybrid algorithms, with representative works such as [69], [70], and [71]. This paper stands out from all these papers, being the first to work with and experiment on the newly unveiled `NL-Hybrid`.

The rest of this article is structured as follows. In Section II, we provide a brief background related to HSS and describe the main aspects of `NL-Hybrid`. Section III focuses on introducing the implementation details. Section IV details the experimentation carried out. The paper ends with Section V, which draws conclusions and outlines potential avenues for future research.

II. NONLINEAR-PROGRAM HYBRID SOLVER

We divide this section on the `NL-Hybrid` into two parts. First, in Section II-A, we give an overview of the method.

Then, in Section II-B, we take a look at its structure and workflow.

A. OVERVIEW OF NL-HYBRID

In recent years, the community has proposed a plethora of hybrid solvers. Many of these techniques are ad hoc implementations to tackle specific problems. Usually, researchers consider a number of factors when designing their hybrid solvers, such as *i)* the specifics of the problem to be solved; *ii)* the limitations and characteristics of the quantum device to be used; and/or *iii)* the knowledge and intuition of the developer. In many cases, the researcher's knowledge in fields such as artificial intelligence or optimization is crucial [72].

However, there are some well-established methods that the community routinely uses, such as the aforementioned QAOA and VQE, or QBSolv [20]. Another interesting and recognizable scheme is *Kerberos* [73], which is a concretization of the above-described *D-Wave Hybrid Framework*. More specifically, *Kerberos* is a reference hybrid workflow comprised of three methods iteratively running in parallel: two classical methods, Simulated Annealing and Tabu Search, and a quantum one that uses the QPU of D-Wave.

As mentioned beforehand, D-Wave's main motivation for creating HSS is to introduce a portfolio of hybrid techniques that lighten the method-implementation phase. In this way, HSS is made up of a set of four ready-to-use methods that target different categories of input and use cases. The last of the methods included in HSS is the `NL-Hybrid`, which is the one we will focus on in this article and which represents a breakthrough in the implementation of hybrid algorithms.

Firstly, `NL-Hybrid` stands out because it allows for the definition of variables in other additional formats than those considered in the methods previously included in HSS. More specifically, `NL-Hybrid` excels with decision variables that embody common logic, such as ordering permutations or subsets of options. For instance, in routing problems like the TSP, a permutation of variables indicates the sequence in which nodes are visited. Similarly, in the KP, the variables representing items to be stored can be categorized into two distinct groups: packed and unpacked.

In this way, in addition to allowing variables defined as binary and integer values, `NL-Hybrid` permits the definition of the following types of decision variables:

- `list(number_variables)`: The solver can use a `list` as the decision variable to optimize, this being an ordered permutation of size `number_variables` describing a possible itinerary.
- `set(number_variables)`: The decision variable can be a `set`, being this a subset of an array of size `number_variables`, representing possible items included in a knapsack.
- `disjoint_list(n_variables,n_lists)`: The solver can employ a `disjoint_list` as the decision variable, which divides a set of `n_variables` into `n_lists` disjoint ordered partitions, each representing a permutation of variables. This encoding is appropriate for complex logistic problems such as the Vehicle Routing Problem. There is a variant of this variable, called `disjoint_bit_sets`, where the order of the produced partitions is not semantically meaningful.

It is worth mentioning at this point that in the field of optimization, whether by means of classical or quantum systems, the performance of a solver is closely tied to its capacity to explore and exploit the whole *solution space*. That is, the larger the *search space*, the higher the probability of reaching better solutions, understanding *search space* as the region of the *solution space* that the algorithm can access. However, there is the downside that larger spaces are usually computationally expensive to explore and exploit. Therefore, employing decision variables that act as implicit constraints is an effective way to reduce both *search* and *solution spaces* and thus, the running time to find a solution. For instance, using `list(number_variables)` to represent a canonical TSP implicitly ensures that no nodes are visited more than once along the route and also that each node is visited. This results in the absence of infeasible solutions within the *solution space*, thereby ensuring that this space comprises solely feasible solutions to the problem. This characteristic is not observed with binary encoding. For this reason, the use of the aforementioned decision variables is a significant advantage for `NL-Hybrid`.

Finally, `NL-Hybrid` natively permits nonlinear (linear, quadratic, and higher order) inequality and equality constraints, expressed even arithmetically. This aspect represents a significant contribution compared to other well-established hybrid solvers. For comparison, `BQM-Hybrid` accepts linear soft constrains as penalty models, while `CQM-Hybrid` works with linear and quadratic constrains natively, although it is also possible to implement them as soft constraints.

To conclude this section, it is important to note that the existence of `NL-Hybrid` does not imply the complete deprecation of previously existing methods in HSS. Depending on the characteristics of the problem and the decision variables used, `NL-Hybrid` may not always be the most efficient algorithm. As will be demonstrated in later sections, `CQM-Hybrid` or `BQM-Hybrid` might be more suitable for problems primarily composed of binary variables.

B. STRUCTURE AND WORKFLOW OF NL-HYBRID

Being part of HSS, `NL-Hybrid` has the same structure as the other methods within the portfolio. This structure, which is depicted in Figure 1, is divided into three distinct phases:

A) First, the NL formulation of the problem is introduced as input into a classical front end. In this preliminary phase, the solver creates a predetermined number of equally structured branches.

B) Secondly, each created thread is executed in parallel on a set of Amazon Web Services (AWS) CPUs and/or GPUs. Each branch is composed of a Classical Heuristic Module (CM) and a Quantum Module (QM). The CM is in charge of exploring

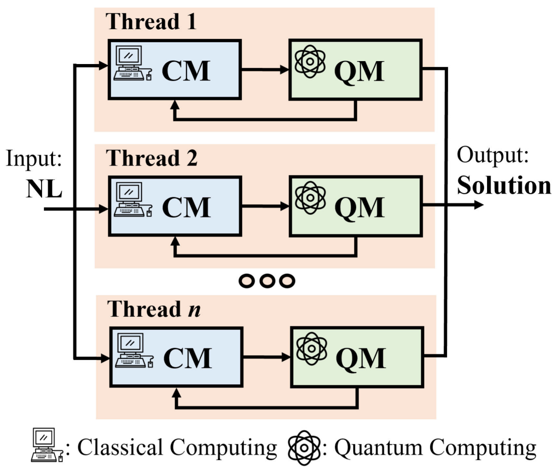

FIGURE 1. Structure of NL-Hybrid solver. CM = Classical Heuristic Module. QM = Quantum Module.

the problem-solution space using traditional heuristics. During this exploration, the CM formulates different quantum queries, which are executed by the QM, and which are partial representations of the problem that are accommodated to the QPU capacity. The solutions provided by the QPU are employed to guide the CM toward promising areas of the solution search space. Furthermore, QM can even improve the solutions found by the CM. `NL-Hybrid` resorts to the latest D-Wave quantum device to execute the quantum queries. At the time of this writing, the system used was the `Advantage_system6.4`, which is made up of 5616 qubits organized in a Pegasus topology.

C) Finally, after a predefined time limit T, all generated branches stop their execution and return their solution to the front end. Then, `NL-Hybrid` forwards the best solution found among all the threads. It should be noted that CM and QM communicate asynchronously, ensuring that latency in a particular branch does not hinder the overall progress of the `NL-Hybrid` solver.

Some of the benefits of using `NL-Hybrid` over ad hoc generated methods or other widely recognized solvers are:

- `NL-Hybrid` is built to manage low-level operational specifics, eliminating the need for users to have any expertise in properly parameterizing the QPU.
- `NL-Hybrid` accepts inputs that are much larger than those of other solvers focused on solving problems in QUBO format and even larger than those of the rest of the solvers within HSS. `NL-Hybrid` is intended to take advantage of the QPU's capability to quickly find promising solutions, expanding this property to a wider range of input types and sizes than would otherwise be feasible.
- `NL-Hybrid` provides a user-friendly use of quantum resources, allowing the user to model a problem in an intuitive way. This is an advantage in comparison to QUBO, which is the native formulation for QPUs, mainly because translating a problem to this binary formulation is often a challenging task [74]. In fact, inefficient translation can critically affect the performance of the solver.

Lastly, it should be noted that the `NL-Hybrid` solver is proprietary. Consequently, further technical details are not available to the general public. For additional information on this method, we refer interested readers to the D-Wave report on the HSS portfolio [29].

III. IMPLEMENTATION DETAILS

The experiments conducted in this study focus on three combinatorial optimization problems: TSP, KP, and MCP. Due to its incipient nature, there is little information regarding `NL-Hybrid`. Given this lack of documentation, we provide several key implementation details on how to tackle the above-mentioned problems by means of `NL-Hybrid`. This allows the reader to understand how intuitive the problem design is.

It should be noted that, while the implementation of MCP was done from scratch, those for TSP and KP are slight adaptations of the open-source code published by D-Wave.[6]

First, in order for the problem to be solved by `NL-Hybrid`, it must be defined using a special entity dedicated for this purpose, called `Model`. Once this model is initialized, the definition of a problem includes the following four steps:

- Defining the decision variables.
- Entering the information necessary to describe the problem as `constants` (if needed).
- Defining the problem constraints (if any).
- Formulating the objective function.

The rest of the section is divided into four subsections. The first three are devoted to each of the problems to be addressed, while the last is devoted to the execution of the `NL-Hybrid`.

A. TRAVELING SALESMAN PROBLEM

The TSP is a classical routing problem that can be represented as a complete graph $G = (V, A)$ of size N, where V illustrates the set of nodes and A the set of edges linking every pair of nodes in V. Furthermore, a matrix C of size NxN contains the costs c_{ij} associated with traveling from node i to node j.

The TSP is a problem that is particularly suitable for being solved by `NL-Hybrid`, since the decision variables can be defined using the above-described `list(number_variables)` type. Thus, `list(N)` represents an ordered permutation of the nodes in V. It is worth noting at this point that, thanks to permutation-based coding, it is not necessary to add any constraints to the model, such as those required when the TSP is defined using

[6]https://github.com/dwavesystems/dwave-optimization/blob/main/dwave/optimization/generators.py

the QUBO or CQM formulations. This is undoubtedly an advantage for the `NL-Hybrid`.

In the following Python code snippet, we show how to initialize the model and the decision variables. We also show how to introduce the information needed to describe the problem.

```
from dwave.optimization.model import Model

#Initializing the model entity
tsp_model = Model()

#Defining the variable as a list of size N
route = tsp_model.list(N)

#Entering the cost matrix as constant
cost_matrix = tsp_model.constant(C)
```

Regarding the objective function, which must be minimized, it can be mathematically formulated as follows, being **x** a `list` representing a feasible TSP `route`:

$$f(\mathbf{x}) = \sum_{i=1}^{N-1} cost\_matrix_{x_i,x_{i+1}} + cost\_matrix_{x_N,x_1} \quad (1)$$

Which is represented in code as shown in the following snippet:

```
route_cost = cost_matrix[route[:-1], route[1:]]
return_cost = cost_matrix[route[-1], route[0]]

# Sum the costs of the full route.
complete_cost = route_cost.sum()+return_cost.sum()

# The objective is introduced using model.minimize
tsp_model.minimize(complete_cost)
```

B. KNAPSACK PROBLEM

In summary, KP consists of a set I of N items, describing each one by its weight (w_i) and its profit (v_i), which must be packed into a knapsack with a maximum capacity C. The objective is to choose a subset of items to be stored that maximizes the profit obtained and does not exceed C.

Like the TSP, the KP is a problem that benefits from the features of `NL-Hybrid`, since we can use the `set(number_variables)` type to define the decision variables of the problem. Thus, `set(N)` represents a subset of I. We show the initialization process of the KP-related model in the following Python code snippet, where W and V are two sets that include the weights and benefits of all items, respectively.

```
from dwave.optimization.model import Model

#Initializing the model entity
kp_model = Model()

#Defining the variable as a set of size N
items = kp_model.set(N)

#Entering problem information as constants
capacity = kp_model.constant(C)
weights = kp_model.constant(W)
profits = kp_model.constant(V)
```

In contrast to MCP and TSP, in the case of the KP it is necessary to add a restriction, which ensures that the items placed in the backpack do not exceed C.

```
capacity_check = weights[items].sum() <= capacity

#Restrictions are introduced using
#model.add_constraint method
model.add_constraint(capacity_check)
```

Finally, the objective function, which must be maximized natively, can be mathematically formulated as follows, being **x** a feasible `set` of `items`, and M the size of this `set`:

$$f(\mathbf{x}) = \sum_{i=1}^{M} values_{x_i} \quad (2)$$

Which can be represented in code as follows:

```
# Sum the profits of introduced items.
sum_values = profits[items].sum()

# The objective is introduced using model.minimize
kp_model.minimize(-sum_values)
```

It is worth noting that `NL-Hybrid` method only allows for the minimization of the objective function. This is why problems natively designed to maximize an objective, such as the KP, are handled by adding a negative sign to the function, as seen in the code above. This situation also occurs with the MCP.

C. MAXIMUM CUT PROBLEM

Taking into account a directed graph G made up of N nodes and a weight matrix W of size NxN, the objective of the MCP is to divide N into two subsets such that the sum of the weights of the cut edges is maximum.

In order to formulate the MCP to be solved by `NL-Hybrid`, the set $X = x_1, \ldots, x_N$ of binary decision variables has been defined, where x_i is 0 if node i is part of the first subset and 1 otherwise.

We show the initialization process of the model in the following Python code snippet.

```
from dwave.optimization.model import Model

#Initializing the model entity
mcp_model = Model()

#Defining the N bynary decision variables
nodes = mcp_model.binary(N)

#Entering the weights into the model as constant
weights = mcp_model.constant(W)
```

Furthermore, the objective function, which must be maximized natively, can be mathematically formulated as follows, being **x** a feasible binary solution to the MCP:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \sum_{j=1}^{N} |x_i - x_j| * weights_{i,j}. \quad (3)$$

Which can be made explicit in code using the following this Python snippet:

```
obj = None

for i in range(N):
 for j in range(N):
  if i!=j:
   if obj is None:
    obj = abs(nodes[i]-nodes[j])*weights[i][j])

   else:
    obj = obj+(abs(nodes[i]-nodes[j])
    * weights[i][j]))

#The objective is introduced using model.minimize
mcp_model.minimize(-obj)
```

It should be noted that, because of the nature of the MCP, no problem constraints are needed.

D. EXECUTING THE PROBLEM

Once the problem has been modeled following the steps described in the previous subsections, it is ready to be submitted to `NL-Hybrid`. This process is carried out in the same way as the rest of the HSS portfolio, as can be seen in the following snippet:

```
from dwave.system import LeapHybridNLSampler

sapi_token = 'XXXXX'
dwave_url = 'https://cloud.dwavesys.com/sapi'

sampler = LeapHybridNLSampler(token=sapi_token,
                              endpoint=dwave_url)
sampler.sample(model)
```

Finally, the results-collecting process differs from the rest of the HSS solvers. In this case, `NL-Hybrid` deposits the complete set of outcomes in the *Model* itself. The results, as well as their energies, can be read as follows:

```
from dwave.system import LeapHybridNLSampler
from dwave.optimization.model import Model

for i in range(model.states.size()):
    solution = next(model.iter_decisions()).
                state(i).astype(int)
    solution_energy = model.objective.state(i)
```

The code above produce an outcome as we represent in the following code snippet, centered on a TSP example composed of 9 nodes.

```
solution {-} objective function
[7 8 6 4 2 3 0 1 5] - 2134.0
[1 5 7 8 6 4 2 3 0] - 2134.0
[1 0 3 2 4 6 8 7 5] - 2134.0
[4 6 8 7 5 1 0 3 2] - 2134.0
[3 2 4 6 8 7 5 1 0] - 2134.0
\ldots
```

IV. EXPERIMENTATION AND RESULTS

The main elements of the experiments carried out are described in this section. First, we detail the characteristics of the benchmark used in Section IV-A. Then, in Section IV-B, we describe the design choices adopted. Lastly, we show and analyze the results obtained in Section IV-C.

A. BENCHMARK DESCRIPTION

As mentioned, a benchmark composed of 45 instances has been used, equally distributed over three combinatorial optimization problems:

A) **TSP**, for which instances of sizes between 7 and 107 nodes have been used. Instances with a size equal to or less than 25 nodes have been obtained from the QC-oriented benchmark QOPTLib [71], while the rest have been obtained from the well-known TSPLib [75].

B) **KP**, for which each case is named `sX_Y`, where `X` is the number of items and `Y` is a suffix to distinguish the set of instances with the same `X`. All instances have been obtained from the KPLib benchmark,[7] described in [76].

C) **MCP**, for which each instance is coined `MC_X`, with `X` being the number of nodes that define the graph. Ten of the instances have been obtained from the QOPTLib mentioned above, while graphs of sizes 80, 90, 120, 140, and 170 have been generated ad hoc for this study.

In Table 2, we summarize the main characteristics of each instance considered. Finally, to improve the reproducibility of this study, all 45 instances are openly available in [77]. Furthermore, the newly created MCP instances have been generated randomly using a Python script, which is also available in the same repository.

B. EXPERIMENTAL SETTING

Using the benchmark described above, the main objective is to analyze the performance of the `NL-Hybrid`, and to compare the results obtained by this method with those obtained by three D-Wave-based counterparts: the QPU, `BQM-Hybrid` and `CQM-Hybrid`. Our main motivation for choosing these counterparts is to measure the main contribution that `NL-Hybrid` provides in relation to previously available methods. More specifically, the `BQM-Hybrid` has been employed in these tests because it is the most widely employed HSS solver and its input format is similar to that of the QPU. Additionally, the `CQM-Hybrid` has been used because it is the most advanced algorithm in the portfolio and is easy to use when modeling the problems addressed. Finally, the `DQM-Hybrid` has been set aside due to the little attention it has received in the literature and the complexity of its use compared to the other solvers within HSS.

Regarding the QPU, the `Advantage_system6.4` device has been used, which is the most recent from D-Wave at the time this work was written. This computer features 5,616 qubits and more than 35,000 couplers arranged in a Pegasus topology. Similar to hybrid solvers, the QPU has been accessed through the `D-Wave Leap` cloud service, and the common forward annealing process has been executed, which consists of the following main steps [78]: *i*) convert QUBO into a graph, *ii*) minor graph embedding, *iii*) QPU initialization, *iv*) annealing, *v*) readout, and *vi*) postprocessing.

[7] https://github.com/likr/kplib

TABLE 2. Summary of the TSP, KP and MCP instances used.

Traveling Salesman Problem Instance	Description	Knapsack Problem Instance	Description	Maximum Cut Problem Instance	Description
wi7	7-noded complete graph	**s50_1**	50 packages and a capacity of 11793	**MC_10**	10-noded graph with 37 edges
dj8	8-noded complete graph	**s50_2**	50 packages and a capacity of 135607	**MC_20**	30-noded graph with 148 edges
dj9	9-noded complete graph	**s100_1**	100 packages and a capacity of 238613	**MC_40**	40-noded graph with 570 edges
dj15	15-noded complete graph	**s100_2**	100 packages and a capacity of 233898	**MC_50**	50-noded graph with 1013 edges
dj22	22-noded complete graph	**s200_1**	200 packages and a capacity of 53104	**MC_60**	60-noded graph with 1450 edges
wi25	25-noded complete graph	**s200_2**	200 packages and a capacity of 51353	**MC_80**	80-noded graph with 2563 edges
wi29	29-noded complete graph	**s200_3**	200 packages and a capacity of 462929	**MC_90**	90-noded graph with 3200 edges
dj38	38-noded complete graph	**s500_1**	500 packages and a capacity of 128354	**MC_100**	100-noded graph with 3902 edges
eil51	51-noded complete graph	**s500_2**	500 packages and a capacity of 119918	**MC_120**	120-noded graph with 5696 edges
berlin52	52-noded complete graph	**s500_3**	500 packages and a capacity of 1236838	**MC_140**	140-noded graph with 7912 edges
st70	70-noded complete graph	**s1000_1**	1000 packages and a capacity of 2494539	**MC_150**	150-noded graph with 8822 edges
eil76	76-noded complete graph	**s1000_2**	1000 packages and a capacity of 243197	**MC_170**	170-noded graph with 11486 edges
rat99	99-noded complete graph	**s1000_3**	1000 packages and a capacity of 2429393	**MC_200**	200-noded graph with 15916 edges
eil101	101-noded complete graph	**s2000_1**	2000 packages and a capacity of 498452	**MC_220**	220-noded graph with 19423 edges
pr107	107-noded complete graph	**s2000_2**	2000 packages and a capacity of 497847	**MC_250**	250-noded graph with 24831 edges

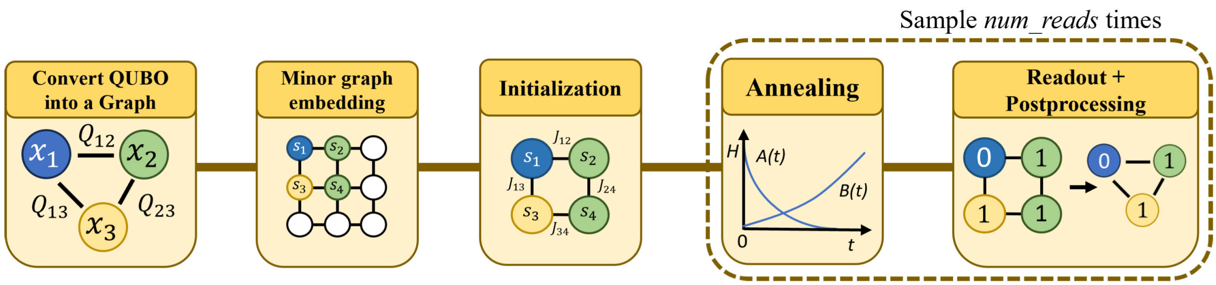

FIGURE 2. Graphical representation of the quantum annealing process followed by the D-Wave's QPU used in this work.

In Figure 2, we graphically represent this process to facilitate its understanding.

In relation to the parameterization, the default values have been used for all the solvers for the sake of fairness. For hybrid solver versions, v2.2, v1.12, and v1.1 have been used for `BQM-Hybrid`, `CQM-Hybrid`, and `NL-Hybrid`, respectively. For the QPU, the default parameters have also been used.

Finally, *Qiskit_Optimization*[8] v0.6.1 open libraries have been employed to aid in solving the three optimization problems using the QPU and `BQM-Hybrid`. These libraries have been utilized because they enable tasks such as data reading, solution decoding, result evaluation, and, most importantly, the automatic construction of the QUBOs. More specifically, to build the QUBO formulations, these classes relax the constraints using a penalty model, with the coefficient being automatically estimated. Finally, it is worth noting that *Qiskit_Optimization* does not present any incompatibility with any versions of Qiskit, such as the current v1.2.

The CQM implementations of the three problems have been developed ad hoc for this research. Finally, as mentioned above, the NL implementation of the MCP has been done ad hoc for this work, while the implementations of TSP and KP are slight adaptations of the open code published by D-Wave. To improve the reproducibility of this research, all codes are available from the corresponding author upon reasonable request.

C. RESULTS

In order to obtain representative results, all the outcomes presented in this section have been obtained after 10 independent runs per method and instance. Regarding TSP, we show in Figure 3 the average of the best solution reached by each solver. It should be noted that the metric used to represent the quality of the solutions is the approximation ratio. This metric has been applied to all the methods executed, and it is particularly suitable for measuring the distance between the obtained outcomes and known optimal solutions in terms of the objective function cost. Thus, the approximation ratio is calculated by using the cost of an obtained solution and the optimal value of the problem being solved. The optimal values for each instance have been obtained from [71] and [75].

In addition to the best solution found, each solver provides a set of solutions called *sample-set* for each execution. It is possible that there are repeated solutions in this *sample-set*. For this reason, the reliability of a method can be measured by analyzing the entire *sample-set*, with it being preferable that the quality of the solution set approaches the optimal solution to the problem. Thus, to represent the robustness of each

[8]https://qiskit.org/ecosystem/optimization/apidocs/qiskit_optimization.applications.html

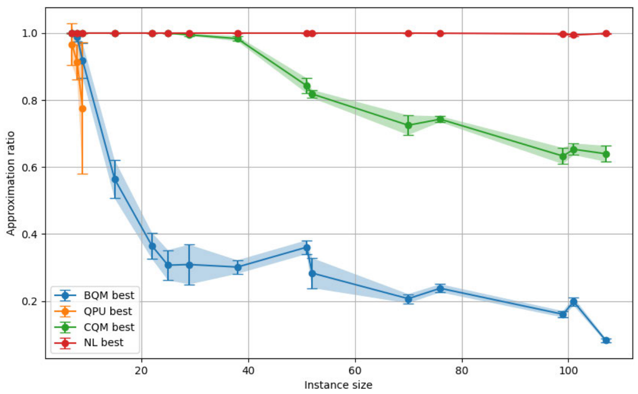

FIGURE 3. The average approximation ratio of the best solutions found by each method for the TSP instances.

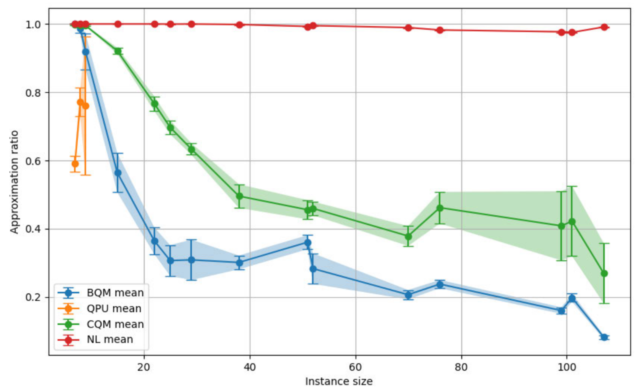

FIGURE 4. The average approximation ratio of the whole *sample-sets* found by each method for the TSP instances.

method, the averages of the complete *sample-sets* obtained by each technique are presented in Figure 4.

A simple glance at Figure 3 and Figure 4 is enough to detect the superiority of the NL-Hybrid solver over its competitors. To verify this improvement, two statistical tests have been performed with the results shown in these figures.

First, we applied Friedman's non-parametric test to determine if there are significant differences among NL-Hybrid, CQM-Hybrid, and BQM-Hybrid. It should be noted that QPU has been left out of all these tests as it does not provide outcomes for all datasets. The results of this test are presented in Table 3. Specifically, the average ranking value returned by the Friedman non-parametric test is presented for each of the compared algorithms. A lower rank indicates better performance. Furthermore, on the left-hand side of Table 3, we show the results referring to the best results obtained, while on the right-hand side, we show those concerning the complete *sample-set* mean.

The Friedman statistics obtained in these tests are 26.53 and 28.13. With a 99% confidence interval, the critical value in a χ^2 distribution with 2 degrees of freedom is 9.21. Since both statistics are greater than this critical value, we can conclude that there are significant differences among the results, with NL-Hybrid having the lowest rank.

TABLE 3. Average rankings obtained using the Friedman's test for the TSP experimentation.

Best solution found		Full *sample-set* average	
Algorithm	Average Ranking	Algorithm	Average Ranking
NL-Hybrid	1.3333	NL-Hybrid	1
CQM-Hybrid	1.8667	CQM-Hybrid	2.0667
BQM-Hybrid	2.8	BQM-Hybrid	2.9333

Following the results described above, we conducted Holm's post-hoc test to evaluate the statistical significance of NL-Hybrid's superior performance. The adjusted p-values from Holm's post-hoc procedure are presented in Table 4. Upon analyzing these results, and considering that all p-values are below 0.05, we can confidently conclude that NL-Hybrid significantly outperforms BQM-Hybrid and CQM-Hybrid with 95% confidence.

TABLE 4. Results obtained using the Holm's post-hoc procedure for the TSP. NL-Hybrid used as control algorithm.

Best solution found		Full *sample-set* average	
Algorithm	Adjusted *p*	Algorithm	Adjusted *p*
CQM-Hybrid	0.04461	CQM-Hybrid	0.003487
BQM-Hybrid	0.0000018	BQM-Hybrid	0.000027

Finally, as can be seen in Figure 3, the performance of NL-Hybrid in all the datasets considered is almost perfect. For this reason, and to get a glimpse of the limits of NL-Hybrid's performance, we have conducted additional experiments with TSPLib instances composed of between 100 and 783 nodes. We show the average of the best results and the average of the complete *sample-sets* in Figure 5, where it can be seen how NL-Hybrid obtains remarkable results (above 0.8 with respect to the optimum) in instances of up to 439 nodes. This performance is a substantial improvement over other hybrid methods in the literature.

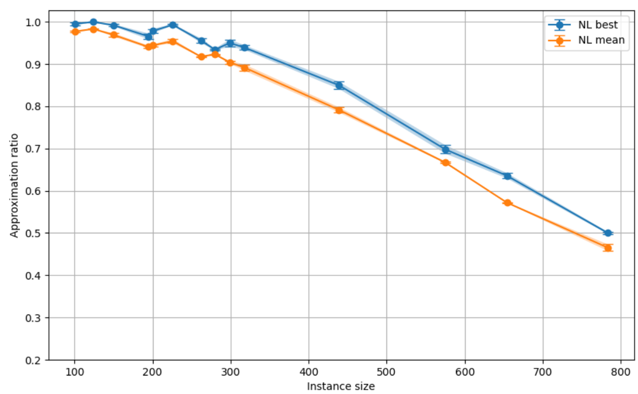

FIGURE 5. The average of best solutions and of the whole *sample-sets* found by the NL-Hybrid for big TSP instances.

Similar conclusions can be drawn if we focus our attention on the tests carried out with the KP. For the KP, we show in Figure 6 and Figure 7 the results obtained in terms of

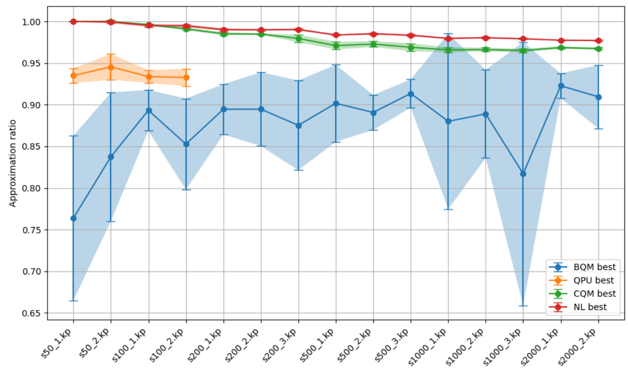

FIGURE 6. The average approximation ratio of the best solutions found by each method for the KP instances.

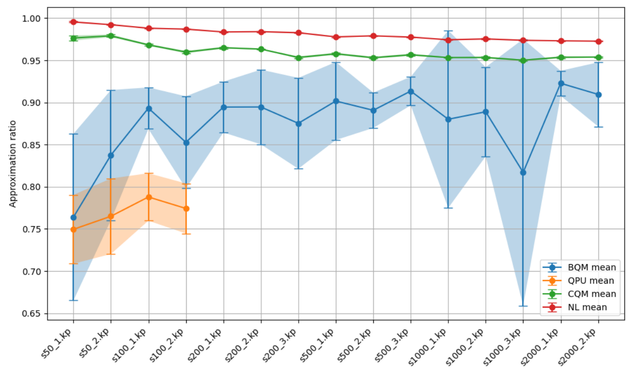

FIGURE 7. The average approximation ratio of the whole *sample-sets* found by each method for the KP instances.

TABLE 5. Average rankings obtained using the Friedman's test for the KP experimentation.

Best solution found		Full *sample-set* average	
Algorithm	Average Ranking	Algorithm	Average Ranking
NL-Hybrid	1.1333	NL-Hybrid	1
CQM-Hybrid	1.8667	CQM-Hybrid	2
BQM-Hybrid	3	BQM-Hybrid	3

the best solution found per run and the average of the entire *sample-sets*, respectively. In this case, the optimal values used as the baseline have been obtained by solving each instance through Google OR-Tools. As mentioned, NL-Hybrid proves to be superior to its competitors also for the KP. The outcomes obtained after the execution of both Friedman's and Holm's post hoc tests are depicted in Table 5 and Table 6, respectively. On the one hand, the Friedman statistics obtained are 26.53 and 30. Given that both statistics exceed the critical value, we can infer that there are significant differences between the results, with NL-Hybrid achieving the lowest rank. On the other hand, because all p-values are below 0.05, we can conclude that Holm's post-hoc test supports the conclusion that NL-Hybrid is significantly better than CQM-Hybrid and BQM-Hybrid with 95% confidence.

TABLE 6. Results obtained using the Holm's post-hoc procedure for the KP. NL-Hybrid used as control algorithm.

Best solution found		Full *sample-set* average	
Algorithm	Adjusted p	Algorithm	Adjusted p
CQM-Hybrid	0.04461	CQM-Hybrid	0.00617
BQM-Hybrid	0.000001	BQM-Hybrid	0

TABLE 7. Wilcoxon test results. Each cell contains a symbol per metric (best solution found and average of the *sample-set*).

	QPU	BQM-Hybrid	CQM-Hybrid
NL-Hybrid	▲ ▲	▽ ▽	▽ ▽

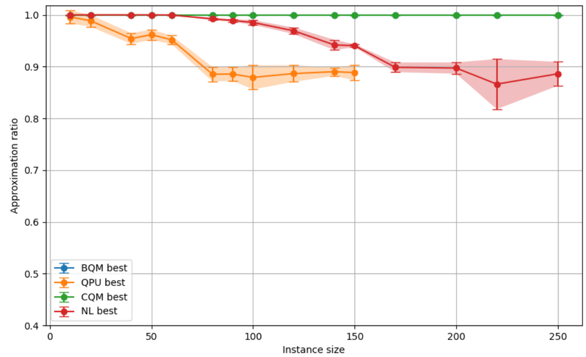

FIGURE 8. The average approximation ratio of the best solutions found by each method for the MCP instances.

Finally, Figures 8 and 9 present the results for MCP, where the conclusions differ significantly from the previous ones. A glance at the results shows that NL-Hybrid only outperforms the QPU, offering considerably lower performance compared to BQM-Hybrid and CQM-Hybrid, both of which demonstrate outstanding suitability for this problem. To assess if the differences between NL-Hybrid and the other algorithms are statistically significant, we used the Wilcoxon rank-sum test. The results are included in Table 7. Each cell displays the two metrics considered (best solution found and average of the sample set) using one of the following symbols: "▲" indicates that NL-Hybrid has produced better results than the algorithm in the column with 99% confidence, and "▽" denotes that the algorithm in the column is statistically superior to NL-Hybrid.

Thus, as can be observed in Table 7, NL-Hybrid significantly outperforms QPU in both metrics, but the superiority of BQM-Hybrid and CQM-Hybrid is also statistically significant for both metrics. However, this performance does not detract from the value of NL-Hybrid. Although these results seem counterintuitive compared to those of TSP and KP, they are consistent with what was expressed in Section II-A. That is, depending on the type of problem and the coding of its decision variables, NL-Hybrid may not always be the most suitable algorithm. Specifically, as shown through this experimentation and supported by D-Wave [79], for problems primarily involving binary variables or integers

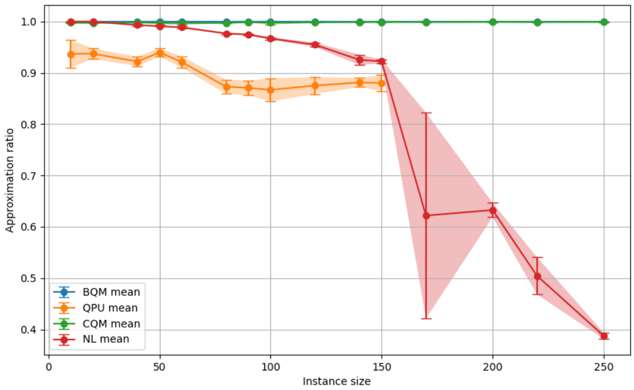

FIGURE 9. The average approximation ratio of the whole *sample-sets* found by each method for the MCP instances.

with low ranges, CQM−Hybrid or BQM−Hybrid are more appropriate. This situation opens up a wide range of future work to be carried out around the NL−Solver and its applicability to industrial problems, which we detail in the following section.

V. CONCLUSION AND FURTHER WORK

This paper focuses on the recently introduced NL−Hybrid solver, evaluating its performance through a benchmark of 45 instances across three combinatorial optimization problems. The results have been compared with three different counterparts. Additionally, to facilitate the use of this relatively unexplored solver, we have provided detailed implementation guidelines for solving the three optimization problems considered.

Given the results obtained, it is prudent to conclude that D-Wave's NL−Hybrid emerges as a promising alternative in the realm of hybrid classical-quantum algorithms. The NL−Hybrid has demonstrated superior performance compared to its competitors, particularly in constrained problems where the new decision variables are applicable. This algorithm proves to be a high-quality approach for such complex problems, efficiently solving large instances and achieving near-optimal solutions. It is likely that NL−Hybrid will become a flagship in the field, primarily due to the innovative types of variables it incorporates, which enable the efficient handling of highly complex problems that have posed significant challenges to existing hybrid solvers.

Furthermore, an important factor to consider when evaluating NL−Hybrid is its ease of use. As shown in Section III, the way variables, constraints, and the objective function are defined is very intuitive. This is not a trivial matter, as it helps to bring and expand the use of quantum computing to a larger group of researchers who may not be familiar with complex quantum concepts.

However, not everything that glitters is gold, as NL−Hybrid presents certain limitations that warrant more detailed exploration. Investigating these weaknesses will enable the scientific community to delineate the boundaries of this solver and determine the contexts in which it can perform optimally. Indeed, the NL−Hybrid has shown inferior performance compared to other hybrid alternatives in the MCP, where the decision variables used were of the binary type and the problem had no constrains. Although this is a limitation, it is not unexpected. In fact, unconstrained problems are ideally suited for the other methods, since there is no need for penalty models. This observation aligns with the statements previously made by D-Wave in [79].

Also, like any hybrid algorithm, there are further limitations that require further investigation. One of these limitations regards the overhead of asynchronous communications. Since quantum and classical computers are separate physical devices, each with its own computational interface and data transfer pipeline, the NL−Hybrid is constrained to repeatedly switch contexts and exchange intermediate data between devices. This latency prevents classical decisions from influencing the evolution of the quantum state before qubits undergo decoherence. Another aspect that requires further investigation is the scalability of the algorithm when dealing with increasingly larger problems.

With all this, the findings reported in this work have enabled us to identify a set of inspiring opportunities, paving the way for several future research directions. These are some of the most intriguing challenges to pursue:

- Analyze the performance of NL−Hybrid against other well-known optimization problems, such as Bin Packing, or the Job-shop Scheduling Problem. For this, it will be necessary to work on adapting the formulation of these problems to the new types of decision variables.
- Study the performance of NL−Hybrid compared to other commercial hybrid methods, not only in terms of quality, but also in terms of scalability.
- Considering the NL−Hybrid's ability to handle complex constraints, efforts will be made to solve problems composed of multiple constraints. The Rich-Vehicle Routing Problem or the Three-Dimensional Bin Packing Problem are suitable cases to advance in this line.
- Explore alternative problem formulations to improve the performance of the NL−Hybrid. An example of this line of research is the Maximum Cut problem, where a formulation that allows the use of the new types of decision variables will be studied.

REFERENCES

[1] Z. C. Seskir, S. Umbrello, C. Coenen, and P. E. Vermaas, "Democratization of quantum technologies," *Quantum Sci. Technol.*, vol. 8, no. 2, Apr. 2023, Art. no. 024005.
[2] R. Tringham and M. Lopez, "The democratization of technology," in *Proc. 7th Int. Conf. Virtual Syst. Multimedia*, 2001, pp. 271–279.
[3] D. Herman, C. Googin, X. Liu, Y. Sun, A. Galda, I. Safro, M. Pistoia, and Y. Alexeev, "Quantum computing for finance," *Nature Rev. Phys.*, vol. 5, no. 8, pp. 450–465, Jul. 2023.
[4] I. Joseph, Y. Shi, M. D. Porter, A. R. Castelli, V. I. Geyko, F. R. Graziani, S. B. Libby, and J. L. DuBois, "Quantum computing for fusion energy science applications," *Phys. Plasmas*, vol. 30, no. 1, pp. 1–11, Jan. 2023.
[5] E. Osaba, E. Villar-Rodriguez, and I. Oregi, "A systematic literature review of quantum computing for routing problems," *IEEE Access*, vol. 10, pp. 55805–55817, 2022.

[6] P. Benioff, "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines," *J. Stat. Phys.*, vol. 22, no. 5, pp. 563–591, May 1980.

[7] R. P. Feynman, "Simulating physics with computers," *Int. J. Theor. Phys.*, vol. 21, nos. 6–7, pp. 467–488, Jun. 1982.

[8] K. Takahashi, "Shortcuts to adiabaticity for quantum annealing," *Phys. Rev. A, Gen. Phys.*, vol. 95, no. 1, pp. 1–12, Jan. 2017.

[9] J. Ferreiro-Vélez, I. Iriarte-Zendoia, Y. Ban, and X. Chen, "Shortcuts for adiabatic and variational algorithms in molecular simulation," 2024, *arXiv:2407.20957*.

[10] A. Mizel, D. A. Lidar, and M. Mitchell, "Simple proof of equivalence between adiabatic quantum computation and the circuit model," *Phys. Rev. Lett.*, vol. 99, no. 7, Aug. 2007, Art. no. 070502.

[11] Z. Yang, M. Zolanvari, and R. Jain, "A survey of important issues in quantum computing and communications," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 2, pp. 1059–1094, 2nd Quart., 2023.

[12] D. Castelvecchi. (2023). *IBM Releases First-Ever 1,000-Qubit Quantum Chip*. Accessed: Aug. 8, 2024. [Online]. Available: https://www.nature.com/articles/d41586-023-03854-1

[13] C. McGeoch, P. Farre, and K. Boothby, "The D-Wave Advantage2 prototype," Wave Syst. Inc., Burnaby, BC, Canada, Tech. Rep. 14-1063A-A, Jun. 2022.

[14] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018.

[15] S. Endo, Z. Cai, S. C. Benjamin, and X. Yuan, "Hybrid quantum-classical algorithms and quantum error mitigation," *J. Phys. Soc. Jpn.*, vol. 90, no. 3, Mar. 2021, Art. no. 032001.

[16] A. Callison and N. Chancellor, "Hybrid quantum-classical algorithms in the noisy intermediate-scale quantum era and beyond," *Phys. Rev. A, Gen. Phys.*, vol. 106, no. 1, Jul. 2022, Art. no. 010101.

[17] T. Lubinski, C. Granade, A. Anderson, A. Geller, M. Roetteler, A. Petrenko, and B. Heim, "Advancing hybrid quantum–classical computation with real-time execution," *Frontiers Phys.*, vol. 10, pp. 1–15, Aug. 2022.

[18] M. Larocca, S. Thanasilp, S. Wang, K. Sharma, J. Biamonte, P. J. Coles, L. Cincio, J. R. McClean, Z. Holmes, and M. Cerezo, "A review of barren plateaus in variational quantum computing," 2024, *arXiv:2405.00781*.

[19] M. Ragone, B. N. Bakalov, F. Sauvage, A. F. Kemper, C. O. Marrero, M. Larocca, and M. Cerezo, "A lie algebraic theory of barren plateaus for deep parameterized quantum circuits," *Nature Commun.*, vol. 15, no. 1, p. 7172, Aug. 2024.

[20] M. Booth, S. P. Reinhardt, and A. Roy, "Partitioning optimization problems for hybrid classical/quantum execution," D-Wave Systems Inc., Burnaby, BC, Canada, Tech. Rep. 14-1006A-A, Jan. 2017.

[21] A. Verma and M. Lewis, "Penalty and partitioning techniques to improve performance of QUBO solvers," *Discrete Optim.*, vol. 44, May 2022, Art. no. 100594.

[22] B. Bichsel, M. Baader, T. Gehr, and M. Vechev, "Silq: A high-level quantum language with safe uncomputation and intuitive semantics," in *Proc. 41st ACM SIGPLAN Conf. Program. Lang. Design Implement.*, Jun. 2020, pp. 286–300.

[23] R. Seidel, S. Bock, R. Zander, M. Petrič, N. Steinmann, N. Tcholtchev, and M. Hauswirth, "Qrisp: A framework for compilable high-level programming of gate-based quantum computers," 2024, *arXiv:2406.14792*.

[24] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, "Quantum computing with qiskit," 2024, *arXiv:2405.08810*.

[25] E. Villar-Rodriguez, A. Gomez-Tejedor, and E. Osaba, "Hybrid classical-quantum computing: Are we forgetting the classical part in the binomial?" in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, Sep. 2023, pp. 264–265.

[26] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, Ł. Cincio, and P. J. Coles, "Variational quantum algorithms," *Nature Rev. Phys.*, vol. 3, no. 9, pp. 625–644, Aug. 2021.

[27] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," 2014, *arXiv:1411.4028*.

[28] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth, and J. Tennyson, "The variational quantum eigensolver: A review of methods and best practices," *Phys. Rep.*, vol. 986, pp. 1–128, Nov. 2022.

[29] D-Wave Developers, "D-wave hybrid solver service: An overview," Wave Syst. Inc., Burnaby, BC, Canada, Tech. Rep. 14-1039A-B, 2020.

[30] D-Wave Developers, "Measuring performance of the leap constrained quadratic model solver," Wave Syst. Inc., Burnaby, BC, Canada, Tech. Rep. 14-1065A-A, 2022.

[31] A. Nourbakhsh, M. Jadidi, and K. Shahriari, "Exploring quantum computing potentials in solving a combinatorial optimization problem to minimize exposure to COVID-19 during a city journey," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 2022, pp. 419–425, Jun. 2022.

[32] D. M. Fox, K. M. Branson, and R. C. Walker, "MRNA codon optimization with quantum computers," *PLoS ONE*, vol. 16, no. 10, Oct. 2021, Art. no. e0259101.

[33] D. M. Fox, C. M. MacDermaid, A. M. A. Schreij, M. Zwierzyna, and R. C. Walker, "RNA folding using quantum computers," *PLOS Comput. Biol.*, vol. 18, no. 4, Apr. 2022, Art. no. e1010032.

[34] A. Iturrospe, "Optimizing decision making for soccer line-up by a quantum annealer," 2021, *arXiv:2112.13668*.

[35] G. Colucci, S. V. D. Linde, and F. Phillipson, "Power network optimization: A quantum approach," *IEEE Access*, vol. 11, pp. 98926–98938, 2023.

[36] A. Makarov, M. M. Taddei, E. Osaba, G. Franceschetto, E. Villar-Rodriguez, and I. Oregi, "Optimization of image acquisition for earth observation satellites via quantum computing," in *Proc. Int. Conf. Intell. Data Eng. Automated Learn.* Cham, Switzerland: Springer, 2023, pp. 3–14.

[37] A. Makarov, C. Pérez-Herradón, G. Franceschetto, M. M. Taddei, E. Osaba, P. del Barrio Cabello, E. Villar-Rodriguez, and I. Oregi, "Quantum optimization methods for satellite mission planning," *IEEE Access*, vol. 12, pp. 71808–71820, 2024.

[38] T. Q. Dinh, S. H. Dau, E. Lagunas, and S. Chatzinotas, "Efficient Hamiltonian reduction for quantum annealing on SatCom beam placement problem," in *Proc. IEEE Int. Conf. Commun.*, May 2023, pp. 2668–2673.

[39] P. Schworm, X. Wu, M. Glatt, and J. C. Aurich, "Solving flexible job shop scheduling problems in manufacturing with quantum annealing," *Prod. Eng.*, vol. 17, no. 1, pp. 105–115, Feb. 2023.

[40] F. Phillipson and H. Bhatia, "Portfolio optimisation using the D-Wave quantum annealer," in *Proc. Int. Conf. Comput. Sci.* Cham, Switzerland: Springer, Jan. 2021, pp. 45–59.

[41] S. Mugel, C. Kuchkovsky, E. Sánchez, S. Fernández-Lorenzo, J. Luis-Hita, E. Lizaso, and R. Orús, "Dynamic portfolio optimization with real datasets using quantum processors and quantum-inspired tensor networks," *Phys. Rev. Res.*, vol. 4, no. 1, Jan. 2022, Art. no. 013006.

[42] S. Certo, A. D. Pham, and D. Beaulieu, "Comparing classical-quantum portfolio optimization with enhanced constraints," 2022, *arXiv:2203.04912*.

[43] L. Martins, A. Rocha, and A. Castro, "A QUBO model to the tail assignment problem," in *Proc. 13th Int. Conf. Agents Artif. Intell.*, 2021, pp. 899–906.

[44] T. D. Tambunan, A. B. Suksmono, I. J. M. Edward, and R. Mulyawan, "Quantum annealing for vehicle routing problem with weighted segment," *AIP Conf. Proc.*, vol. 2906, no. 1, pp. 1–7, 2023.

[45] M. Venere, G. Sorrentino, B. Branchini, D. Conficconi, E. Di Nitto, D. Sciuto, and M. D. Santambrogio, "On the design and characterization of set packing problem on quantum annealers," in *Proc. 20th Int. Conf. Smart Technol.*, Jul. 2023, pp. 695–700.

[46] S. V. Romero, E. Osaba, E. Villar-Rodriguez, I. Oregi, and Y. Ban, "Hybrid approach for solving real-world bin packing problem instances using quantum annealers," *Sci. Rep.*, vol. 13, no. 1, p. 11777, Jul. 2023.

[47] S. V. Romero, E. Osaba, E. Villar-Rodriguez, and A. Asla, "Solving logistic-oriented bin packing problems through a hybrid quantum-classical approach," in *Proc. IEEE 26th Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2023, pp. 2239–2245.

[48] B. Maurice Benson, V. M. Ingman, A. Agarwal, and S. Keinan, "A CQM-based approach to solving a combinatorial problem with applications in drug design," 2023, *arXiv:2303.15419*.

[49] M. A. Nau, A. Hans Vija, W. Gohn, M. P. Reymann, and A. K. Maier, "Hybrid adiabatic quantum computing for tomographic image reconstruction—Opportunities and limitations," 2022, *arXiv:2212.01312*.

[50] V. Nayak and V. Sahu, "Quantum approach to optimize aircraft cargo loading," in *Proc. Int. Conf. Trends Quantum Comput. Emerg. Bus. Technol. (TQCEBT)*, Oct. 2022, pp. 1–6.

[51] C. O'Meara, M. Fernández-Campoamor, G. Cortiana, and J. Bernabé-Moreno, ''Quantum software architecture blueprints for the cloud: Overview and application to peer-2-peer energy trading,'' in *Proc. IEEE Conf. Technol. Sustainability (SusTech)*, Apr. 2023, pp. 191–198.
[52] E. Osaba, E. Villar-Rodriguez, and A. Asla, ''Solving a real-world package delivery routing problem using quantum annealers,'' *Sci. Rep.*, vol. 14, no. 1, p. 24791, Oct. 2024.
[53] E. Osaba, E. Villar-Rodriguez, and A. Asla, ''Exploring utility in a real-world warehouse optimization problem: Formulation based on quantum annealers and preliminary results,'' 2024, *arXiv:2409.09706*.
[54] S. S. Tripathy, N. Koul, and H. Patel, ''Comparative study between quantum and classical methods: Few observations from portfolio optimization problem,'' in *Proc. Int. Conf. Trends Quantum Comput. Emerg. Bus. Technol. (TQCEBT)*, Oct. 2022, pp. 1–5.
[55] M. Fernández-Campoamor, C. O'Meara, G. Cortiana, V. Peric, and J. Bernabé-Moreno, ''Community detection in electrical grids using quantum annealing,'' 2021, *arXiv:2112.08300*.
[56] M. Wierzbiński, J. Falcó-Roget, and A. Crimi, ''Community detection in brain connectomes with hybrid quantum computing,'' *Sci. Rep.*, vol. 13, no. 1, p. 3446, Mar. 2023.
[57] S. Jhaveri, ''Quantum computing for software engineering: Code clone detection and beyond,'' Master's thesis, Univ. California, Irvine, CA, USA, 2023.
[58] E. Pasetto, M. Riedel, F. Melgani, K. Michielsen, and G. Cavallaro, ''Quantum SVR for chlorophyll concentration estimation in water with remote sensing,'' *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2022.
[59] S. Lin, ''Computer solutions of the traveling salesman problem,'' *Bell Syst. Tech. J.*, vol. 44, no. 10, pp. 2245–2269, Dec. 1965.
[60] S. Martello and P. Toth, ''Algorithms for knapsack problems,'' *North-Holland Math. Stud.*, vol. 132, pp. 213–257, Jan. 1987.
[61] F. Hadlock, ''Finding a maximum cut of a planar graph in polynomial time,'' *SIAM J. Comput.*, vol. 4, no. 3, pp. 221–225, Sep. 1975.
[62] W. Boüejko, A. Burduk, J. Pempera, M. Uchroński, and M. Wodecki, ''Optimal solving of a binary knapsack problem on a D-Wave quantum machine and its implementation in production systems,'' *Ann. Oper. Res.*, vol. 2024, pp. 1–16, May 2024.
[63] Z. Wang, S. Hadfield, Z. Jiang, and E. G. Rieffel, ''Quantum approximate optimization algorithm for MaxCut: A fermionic view,'' *Phys. Rev. A, Gen. Phys.*, vol. 97, no. 2, Feb. 2018, Art. no. 022304.
[64] W. V. Dam, K. Eldefrawy, N. Genise, and N. Parham, ''Quantum optimization heuristics with an application to knapsack problems,'' in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, Oct. 2021, pp. 160–170.
[65] G. Pecci, R. Wang, P. Torta, G. B. Mbeng, and G. Santoro, ''Beyond quantum annealing: Optimal control solutions to MaxCut problems,'' 2024, *arXiv:2405.08630*.
[66] W. Qian, R. A. M. Basili, M. M. Eshaghian-Wilner, A. Khokhar, G. Luecke, and J. P. Vary, ''Comparative study of variations in quantum approximate optimization algorithms for the traveling salesman problem,'' *Entropy*, vol. 25, no. 8, p. 1238, Aug. 2023.
[67] E. Osaba, E. Villar-Rodriguez, I. Oregi, and A. Moreno-Fernandez-de-Leceta, ''Hybrid quantum computing–Tabu search algorithm for partitioning problems: Preliminary study on the traveling salesman problem,'' in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2021, pp. 351–358.
[68] E. Osaba, E. Villar-Rodriguez, I. Oregi, and A. Moreno-Fernandez-De-Leceta, ''Focusing on the hybrid quantum computing–Tabu search algorithm: New results on the asymmetric salesman problem,'' in *Proc. Genetic Evol. Comput. Conf. Companion*, Jul. 2021, pp. 1476–1482.
[69] E. Osaba, M. Petrič, I. Oregi, R. Seidel, A. Ruiz, S. Bock, and M.-A. Kourtis, ''Eclipse qrisp QAOA: Description and preliminary comparison with qiskit counterparts,'' 2024, *arXiv:2405.20173*.
[70] E. Osaba, E. Villar-Rodriguez, A. Gomez-Tejedor, and I. Oregi, ''Hybrid quantum solvers in production: How to succeed in the NISQ era?'' 2024, *arXiv:2401.10302*.
[71] E. Osaba and E. Villar-Rodriguez, ''QOPTLib: A quantum computing oriented benchmark for combinatorial optimization problems,'' in *Benchmarks and Hybrid Algorithms in Optimization and Applications*. Cham, Switzerland: Springer, 2023, pp. 49–63.
[72] R.-E. Precup, E.-L. Hedrea, R.-C. Roman, E. M. Petriu, A.-I. Szedlak-Stinean, and C.-A. Bojan-Dragos, ''Experiment-based approach to teach optimization techniques,'' *IEEE Trans. Educ.*, vol. 64, no. 2, pp. 88–94, May 2021.
[73] G. Malviya, B. AkashNarayanan, and J. Seshadri, ''Logistics network optimization using quantum annealing,'' in *Proc. Int. Conf. Emerg. Trends Technol. Intell. Syst.* Cham, Switzerland: Springer, 2023, pp. 401–413.
[74] D. De Santis, S. Tirone, S. Marmi, and V. Giovannetti, ''Optimized QUBO formulation methods for quantum computing,'' 2024, *arXiv:2406.07681*.
[75] G. Reinelt, ''TSPLIB—A traveling salesman problem library,'' *ORSA J. Comput.*, vol. 3, no. 4, pp. 376–384, Nov. 1991.
[76] H. Kellerer, U. Pferschy, D. Pisinger, H. Kellerer, U. Pferschy, and D. Pisinger, ''Exact solution of the knapsack problem,'' in *Knapsack Problems*. Berlin, Germany: Springer, 2004, pp. 117–160.
[77] E. Osaba and P. Miranda-Rodriguez, 2024, ''Benchmark dataset and results for the D-Wave's nonlinear-program hybrid solver analysis,'' *Mendeley Data*, doi: 10.17632/fbj4ycdn98.1.
[78] S. Yarkoni, E. Raponi, T. Bäck, and S. Schmitt, ''Quantum annealing for industry applications: Introduction and review,'' *Rep. Prog. Phys.*, vol. 85, no. 10, Oct. 2022, Art. no. 104001.
[79] *Exploring Hybrid Quantum Solvers: Determining the Best Fit for Your Problem*. Accessed: Oct. 7, 2024. [Online]. Available: https://dwave.medium.com/exploring-hybrid-quantum-solvers-determining-the-best-fit-for-your-problem-09992daa497a

ENEKO OSABA received the Ph.D. degree in artificial intelligence, in 2015. He is currently with TECNALIA as a Principal Researcher in the DIGITAL/Next area. He has participated in more than 30 research projects. He has contributed to the development of more than 170 articles, including more than 30 Q1. He has performed several stays at universities of U.K., Italy, and Malta. He has served as a member of the program and/or organizing committee in more than 60 international conferences. He is a member of the editorial board of *Data in Brief* and *Journal of Advanced Transportation*. He has acted as a Guest Editor in journals, such as *Neurocomputing*, *Journal of Supercomputing*, *Swarm and Evolutionary Computation*, and *IEEE Intelligent Transportation Systems Magazine*. Finally, in 2022, he was recognized by the Basque Research and Technology Alliance as one of the most promising young researchers in Basque Country, Spain.

PABLO MIRANDA-RODRIGUEZ received the bachelor's degree in physics and the M.Sc. degree in mathematical physics, in 2019 and 2020, respectively, with a specialization in black hole thermodynamics, and the M.Sc. degree in quantum computing, in 2022, with a specialization in annealing methods for combinatorial optimization in finance. He is currently with TECNALIA as a Junior Researcher in the DIGITAL/Next area. Since then, he has worked in hybrid computing, especially using the quantum annealing paradigm on optimization problems. He has participated in several proof-of-concept projects in the field of quantum computing.