Disjunctive Programming meets QUBO

Pedro Maciel Xavier,a,b,c Pedro Ripper,c,d Joshua Pulsipher,e Joaquim Dias Garcia,c Nelson Maculan,b David E. Bernal Neira a,f,g,\*

a Davidson School of Chemical Engineering, Purdue University,West Lafayette,IN, USA  
b Computer Science & Systems Engineering Program, COPPE, Federal University of Rio de Janeiro, RJ, Brazil  
c PSR Energy Consulting & Analytics, Rio de Janeiro, RJ, Brazil  
d PUC Department of Electrical Engineering, Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil  
e Department of Chemical Engineering, University of Waterloo, 200 University Ave. W., Waterloo, ON N2L 3G1, Canada  
f Quantum Artificial Intelligence Laboratory (QuAIL), NASA Ames Research Center, Moffett Field, CA, USA  
g USRA Research Institute for Advanced Computer Science, Mountain View, CA, USA

dbernaln@purdue.edu

Abstract

Optimization problems must often be reformulated from how they are modeled into a form solvers can tackle. This is the case for optimization problems over disjunctive sets that entail logic constraints, known as Generalized Disjunctive Programs (GDPs). GDPs are usually solved by reformulating them as Mixed-Integer Programs (MIPs), for which powerful solvers exist. Alternatively, physics-inspired methods have been proposed for Quadratic Unconstrained Binary Optimization (QUBO). MIPs can be approximated as QUBOs; for instance, GDPs can be solved by these physics-inspired solvers by using GDP-MIP-QUBO reformulations. We evaluate this approach empirically by solving resulting QUBOs from the well-known MIP reformulations of GDP, Big-M and Hull, and compare it to a proposed reformulation that directly encodes GDP as QUBO via indicator variables. Our results demonstrate an advantage in avoiding the intermediate MIP reformulations when obtaining QUBO problems from GDP.

**Keywords**: Disjunctive Programming, Quantum Computing, QUBO, Ising solver

* 1. Introduction

Optimization problems involving choosing among discrete alternatives are ubiquitous in decision-making processes, and many are relevant in process systems engineering (PSE). In many applications, these choices imply that certain constraints are active or not. A powerful framework for modeling these optimization problems is Generalized Disjunctive Programming (Grossmann and Trespalacios, 2013), where in addition to optimization objectives, , and constraints, , over continuous variables, , that usually appear in mathematical programming; Boolean variables, , can activate sets of algebraic constraints or disjuncts, , and be involved in logical constraints, , of which a common one is the disjunction or exclusive or over subsets of them, .

|  |  |
| --- | --- |
|  | (1) |

To solve GDP problems, one usually relies on the beautiful theory of Disjunctive Programming (Balas, 2018), which allows these problems to be reformulated into Mixed-Integer Programs (MIPs) to access Branch-and-Bound-based solvers. Over the past decade, MIP solvers have provided increasing support for indicator constraints. This feature presents a way to preserve the semantics of GDP models, making it easier for modeling and enabling solvers to leverage structure that is sometimes lost during MIP reformulation.

Much of the difficulty in solving GDP problems comes from the combinatorial aspect of disjunct selection. Novel solution heuristics have been devised in the past decades to address combinatorial optimization problems that will classically require tremendous computational effort. Most notably, the development of quantum optimization algorithms and other physics-inspired methods for Quadratic Unconstrained Binary Optimization (QUBO) has been an active research area that has had several advancements since the first theoretical speedup projections were made two decades ago. QUBO problems can be written as follows

|  |  |
| --- | --- |
|  | (2) |

Even though interesting optimization problems are often described as constrained MIPs, many QUBO reformulation techniques have been derived and implemented to reframe general models and feed them to these novel optimization architectures (Maciel Xavier et al. 2023). Most QUBO platforms will be able to reinterpret MIP and MINLP problems. Still, there's no support for GDP native inputs or a reformulation path that considers the specificities of disjunctive programs.

Considering GDP's modeling flexibility and industrial relevance and the exciting possibility of solving them using advanced solvers, this work aims to develop reformulation techniques to transform GDP problems into QUBO and evaluate their performance using simulated and quantum annealing. We implement two alternative GDP-MIP-QUBO reformulations using Big-M and Hull reformulations. We compare them to a newly proposed reformulation based on indicator variables and problem-aware considerations to enhance the final formulation's conditioning. We test these reformulations with examples, including a PSE application involving a choice of reactors in a chemical process (Iftakher et al., 2023).

* 1. Reformulation Methods

Reformulation techniques of more general optimization problems into QUBO have been developed in the past decade to develop applications for physics-inspired methods, better known as Ising solvers, given the Ising model of spins in a transverse field, a model equivalent to the QUBO shown in Eq. (2).

The QUBO.jl ecosystem implements a wide range of tools and novel and existing developments from literature. A brief overview of the relevant aspects of QUBO reformulation is presented in the following sections. Nevertheless, a more detailed discussion about the subject can be found in Maciel Xavier et al. (2023).

To access QUBO solvers, it becomes necessary to represent real- and integer-valued variables using only binary variables. Let be a variable from an optimization model. We call an encoding a function one built to represent using binary variables, even if approximately.

To enforce constraints while solving unconstrained problems, one can move them to the objective function through penalization. Each constraint of the form will be represented by a penalty function and a corresponding penalty factor .

|  |  |
| --- | --- |
|  | (3) |

By composing the capabilities of the DisjunctiveProgramming.jl (GDP to MIP) and QUBO.jl (MIP to QUBO) packages, one can establish the complete bridge between GDP and QUBO in terms of mathematical formulation and software development. Using the indicator reformulation of GDPs opens a more general reformulation path. Given that it is possible to penalize a constraint of the form as, one can encode as where is the binary variable that corresponds to . When information regarding variable bounds is made available within disjunctions , as in , we propose the following encoding,

|  |  |
| --- | --- |
|  | (4) |

Note that no extra penalization terms are produced for enforcing each constraint .

* 1. Results

Two GDP models and their reformulations were evaluated using both a classical (Neal Simulated Annealing) algorithm (D-Wave Systems Inc., 2022) and a quantum (D-Wave Quantum Annealing) solver (Finnila et al., 1994). For each problem reformulation, we consider: ∆, the largest magnitude among the QUBO coefficients; nvars, the number of binary variables; nqubits, the number of qubits after embedding the problem in the Quantum Processing Unit (QPU); TTT, the Time-to-target; and TTF, the Time-to-feasibility, both for Simulated and Quantum Annealing, SA and QA, respectively. We used the neal implementation for SA and executed it single-threaded in a Laptop with 2.80GHz processors and 16GB of RAM, where each problem variation was runned 1000 times, performing 4000 sweeps per run. For QA we used the D-Wave Advantage system to also obtain 1000 runs per problem, each requiring 200us of annealing time in the QPU.

We can interpret ∆ as a rough measure of the problem's conditioning, where greater values are related to harder instances. nvars and nqubits indicate the amount of resources required to run SA and QA, respectively. Depending on the interaction between variables, multiple qubits might be necessary to represent a single variable when embedding a QUBO problem in quantum hardware. The Time-to-target (TTT) and Time-to-feasibility (TTF) values are calculated using the time spent running the algorithm t and the success probability p = #success/#samples as inputs for

|  |  |
| --- | --- |
|  | (5) |

This quantity provides a performance metric for optimization heuristics that accounts for algorithmic effort and solution quality. The target is reached for TTT when getting a feasible solution within a 5% gap of the optimal value. All solutions feasible to the original problem are regarded as successful for TTF.

A screenshot of a computer

Description automatically generated

Figure 1: Solutions found via Simulated Annealing for (left) Big-M, (center) Hull, and (right) indicator reformulations of the Boxes Problem, Eq. (6).

* + 1. Boxes

Our first example of a disjunctive model is the optimization of a linear function of two variables over a feasible region composed of two separate boxes, as in Eq. (8).

|  |  |
| --- | --- |
|  | (6) |

Figure 1 shows how Simulated Annealing (SA) responds to the Big-M, Hull, and Indicator reformulations. As confirmed by Table 1, SA cannot find a feasible solution for the Big-M and Hull cases. In its last row is the interval-aware (IA) reformulation, highlighted in Figure 2, whose results from SA and Quantum Annealing (QA) indicate a clear advantage by yielding optimal solutions with SA and near-optimal with QA.

Table 1: Results for the Boxes Problem, Eq. (6)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Reformulation Method |  |  | TTTSA | TTFSA |  | TTTQA | TTFQA |
| Big-M |  |  |  |  |  |  |  |
| Hull |  |  |  |  |  |  |  |
| Indicator |  |  |  |  |  |  |  |
| Indicator (IA) |  |  |  |  |  |  |  |

As expected, we can see from Table 1 and Figure 1 that the Hull will require more variables than the Big-M approach and also yield a worser conditioning, which can also be seen by its TTT/TTF values for SA. The indicator method, if applied naively, will narrow the coefficient range but will require auxiliary variables to reduce the final expression’s high-degree terms to 2. An Interval-Aware (IA) technique that considers the disjunction of the variables’ domains at the moment in which they are encoded will be able to produce much better results. As depicted in Figure 2., this will happen because there’s no need to penalize such interval constraints, and their fulfilment arises from the values each variable’s encoding is able to take.

Moreover, in Figure 2 it is possible to see that both samplers will output most solutions being distributed within each box, instead of having them to be more spread throughout the whole domain while subject to penalization as in Figure 1. Figure 2 also shows that the optimal within each box is consistently found using SA, something that did not happen as often with the Big-M, Hull and Indicator methods in Figure 1.

A screenshot of a computer game

Description automatically generated

Figure 2: Solutions found via Simulated Annealing (left) and Quantum Annealing (right) for the problem-aware indicator reformulations of the Boxes Problem, Eq (6).

* + 1. Reactors Problem

A reactor choice GDP presented by (Iftakher et al., 2023) is given by Eq. (7),

|  |  |
| --- | --- |
|  | (7) |

The problem states the selection of one of two reactor designs under a global linear constraint. For this problem, leveraging knowledge from the problem and creating an interval-aware (IA), constraint-aware (CA) indicator reformulation was essential to produce a well-behaved reformulation, as shown in Table 2 and Figure 3. By using the global constraint to tighten each variable's bounds, it was possible to obtain good results for both SA and QA, as seen in Figure 3.

A screenshot of a graph

Description automatically generated

Figure 3: Solutions found via Simulated Annealing (left) and Quantum Annealing (right) for the constraint-aware indicator reformulation of the Reactors Problem, Eq. (7).

The Interval-Aware reformulation that provided the best setting in the previous example, will now interact catastrophically with the global constraint , producing high-order terms and, consequently, introducing many auxiliary variables. To address this issue, we use this same inequality combined with the knowledge from the disjuncts to derive bounds for and that will not only restrict their domain but enforce the global constraint as well. This method shows up in Table 2 as the Interval-Aware, Constraint-Aware (IA, CA) reformulation.

Table 2: Results for the Reactor Problem, Eq. (7)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Reformulation Method |  |  | TTTSA | TTFSA |  | TTTQA | TTFQA |
| Big-M |  |  |  |  |  |  |  |
| Hull |  |  |  |  |  |  |  |
| Indicator |  |  |  |  |  |  |  |
| Indicator (IA) |  |  |  |  |  |  |  |
| Indicator (IA, CA) |  |  |  |  |  |  |  |

Interesting to notice that despite using more variables than the Big-M and Hull variations, the (IA, CA) version presents a significant drop on the value of , followed by better TTT/TTF records.

* 1. Conclusion & Future Work

This work shows different reformulations of GDP problems into QUBO and their effect when solving via physics-inspired methods, namely simulated and quantum annealing. Our results indicate that the reformulations that require an intermediate MIP reformulation, either Big-M or Hull, lead to QUBO problems that become challenging for Ising solvers. Notably, the Hull reformulation performs the worst given the introduction of extra variables and constraints and that annealing-based methods do not use the continuous relaxation for solving the problem. Reformulating considering the disjunctive structure of the original GDP problems results in more amenable QUBO problems for the simulated and quantum annealing algorithms, highlighted by our proposed indicator reformulation. These results highlight the importance of considering the solver features when reformulating a problem. Namely, metrics such as the conditioning and number of variables/qubits after embedding are determining factors for the QUBO-based solvers, contrary to the tightness of the continuous formulation, for example. We see this work as the first step towards deriving QUBO reformulations of GDP problems to solve practical problems efficiently and envision structure-specific reformulations as the enabling technique for allowing physics-inspired (including quantum) methods to address GDP problems. The code for reproducing the experiments can be found at “https://github.com/pedromxavier/DisjunctiveToQUBO.jl”.

References

E. Balas, 2018. Disjunctive programming. Springer, New York, USA.

D-Wave Systems Inc., 2022. dwave-neal. https://https://github.com/dwavesystems/dwave-neal.

A. Finnila, M. Gomez, C. Sebenik, C. Stenson, J. Doll, 1994. Quantum annealing: A new method for minimizing multidimensional functions. Chemical Physics Letters 219 (5), 343–348.

I. E. Grossmann, F. Trespalacios, 2013. Systematic modeling of discrete-continuous optimization models through generalized disjunctive programming. AIChE Journal 59 (9), 3276–3295.

A. Iftakher, M.-K. Kazi, M. F. Hasana, 2023. Mixed-Integer Quadratic Optimization Using Quantum Computing For Process Applications. In: Foundations of Computer Aided Process Operations / Chemical Process Control (FOCAPO/CPC 2023).

S. Kirkpatrick, C. D. Gelatt Jr, M. P. Vecchi, 1983. Optimization by simulated annealing. Science 220 (4598), 671–680.

P. Maciel Xavier, P. Ripper, T. Andrade, J. Dias Garcia, N. Maculan, D. E. Bernal Neira, 2023. QUBO.jl: A Julia ecosystem for Quadratic Unconstrained Binary Optimization.

H. D. Perez, S. Joshi, I. E. Grossmann, 2023. Disjunctiveprogramming. jl: Generalized disjunctive programming models and algorithms for jump. arXiv preprint arXiv:2304.10492.