The Quantum Computing Company™

Choosing good problems for quantum annealing

TECHNICAL REPORT

2020-09-06

Overview

Many complex optimization problems arise in research and industry. Solving these problems helps researchers make new scientific discoveries, build better materials, and synthesize new medicine; and helps businesses make better decisions, reduce costs, increase production, and design better products. *Quantum annealing* is a heuristic method for solving many of the optimization problems that appear in a range of disciplines. This document provides the background to help you choose the right problems for quantum annealing.

CONTACT

Corporate Headquarters
3033 Beta Ave
Burnaby, BC V5G 4M9
Canada
Tel. 604-630-1428

US Office
2650 E Bayshore Rd
Palo Alto, CA 94303

Email: info@dwavesys.com

www.dwavesys.com

14-1046A-A
D-Wave Technical Report Series

Notice and Disclaimer

Summary

Many complex optimization problems arise in research and industry. Solving these problems helps researchers make new scientific discoveries, build better materials, and synthesize new medicine; and helps businesses make better decisions, reduce costs, increase production, and design better products. *Quantum annealing* is a heuristic method for solving many of the optimization problems that appear in a range of disciplines. This document provides the background to help you choose the right problems for quantum annealing.

Contents

1 Introduction

Many complex optimization problems arise in research and industry. Solving these problems helps researchers make new scientific discoveries, build better materials, and synthesize new medicine; it helps businesses make better decisions, reduce costs, increase production, and design better products. *Quantum annealing* is a heuristic method for solving many of the optimization problems that appear in a range of disciplines.

As you begin to consider how quantum annealing may help solve the complex problems that you face in your research efforts or in your business, it is essential that you first understand the different types of complexity and optimization problems.

Every successful application requires a careful understanding of the requirements and parameters. The chain of questions that you consider for your application will guide you towards the mathematical formulation of your problem. These may be questions such as "what is the business problem that the application is trying to solve?" Or, "what is the research question?" Even at this stage, there are still many questions to address.

Does an application require a data processing pipeline or predictive analysis? Or does it need solving an optimization problem?

In this document, we will learn that understanding complexity, optimization problems, and variable and constraint types can help you choose the right problems for quantum annealing.

2 Complexity

Intuitively, *complexity* describes how difficult a problem is and what might make it challenging to solve.

There are different types of complexity. For example, suppose that you want to make a change to your company's website. Since users interact with your website regularly, you must consider how they will react to the change. If there are many users involved, they can be treated as a statistical population. The complexity of this problem is statistical inference and careful experimental design. In a different scenario, the complexity may be the fact that there is a human in the loop.

Alternatively, suppose your website contains a large number of records that must be searched efficiently. Or, you might want to build a predictive model using this large amount of data. In both examples, the problem is complicated because it involves big data.

Each example deals with different types of complexity. Some problems are complex because finding a good model is difficult, predicting the behavior of individuals is hard, sorting through a large amount of data is costly, or defining good experimental design is challenging.

As we begin to explore candidate problems for quantum annealing, we look for problems that consist of a hard optimization problem. Returning to an earlier example, after you sort through big data, and picked a predictive model, you may still need to optimize an objective. For example, you sort through billions of records, and you understand the interaction of each item with the entire organization. You have learned how each item impacts the overall costs for the organization. Now you need to optimize those costs subject to some constraints. For example, you may need to minimize costs while maintaining sales revenue.

Many optimization problems are *NP-hard*, which means no polynomial algorithm exist for solving them. As you add variables to these problems, the solve time scales exponentially. These types of problems are good candidates for quantum annealing. Let's dig deeper.

3 Optimization

Optimization problems arise in many areas of science and industry. For many companies (like retailers and distributors), improving business is equivalent to optimizing processes, cost, productivity, and quality.

Many optimization problems are not considered hard, in a mathematical sense, but only extensive, which means the optimization problem contains a huge number of entities and variables — even billions and trillions! On the other hand, some optimization problems are much more time-consuming to solve even though they involve a few hundred to a few thousand variables. These small but challenging optimization problems, such as personnel scheduling and the famous traveling salesperson problem, fall into the class of NP-hard problems.

Optimization problems that have discrete or binary variables are often better suited for quantum annealing than those with continuous variables (because conversion to discrete is required). When first learning about quantum annealing, it may be easier, to begin with, the former type of optimization problem. However, it is worth noting that there are also examples of continuous-valued problems that have seen success with quantum annealing. For example, QBoost is a boosting algorithm that assigns binary weights to a subset of weak classifiers [**6**].

3.1 Objectives

Sometimes an optimization problem is written as a mathematical expression (objective) that needs to be minimized (or maximized). This is an excellent starting point to enable using many optimization packages. For example, to run an optimization problem on D-Wave's quantum annealing hardware, we can take that mathematical expression and map it to an Ising Hamiltonian.

Objective functions can have different forms depending on the context and the type of problems.

Even simple linear problems with discrete variables can quickly become complicated if we have many competing constraints or interactions between many variables. For example, a constrained integer linear problem like many scheduling problems has a simple linear objective; however, the addition of constraints — such as contiguous shift, limited shift duration, and so on — make it a complex optimization problem. More on the constraints below.

The most suitable objective function for quantum annealing consists of a quadratic objective or one that includes only pairwise interactions of binary variables (QUBO) because that is the most similar objective to the Ising Hamiltonian of a quantum annealing processor. An example of a natural QUBO problem is the maximum-cut problem. This problem occurs in many applications involving social networks. For example, the classification of GitHub users into two categories, machine learning developers or web developers can be efficiently formulated as a QUBO problem.

3.2 Constrained problems

Many optimization problems are complex due to constraints. Imagine you want to minimize the operation cost of an organization. One method is to express the total cost as the sum of individual costs:

$$C = \sum_i x_i c_i \tag{1}$$

Where x_i is the number of units (integer), and c_i is the cost per unit of the item or process i.

Easy, right? The objective above is a linear function where all variables and unit costs are positive. The minimum is where all $x_i = 0$ and the total cost is zero. But wait! That's not a practical solution. To formulate an actual problem, we need to have constraints. Usually, these constraints add complexity.

There are many types of constraints that appear in optimization problems. For example, we might use a binary variable to express yes/no to the existence of a feature. We might also have many different possibilities, out of which we only want to choose one. In such a case, not only do we want the variable to behave as binary, but we also want a group of variables to respect the constraint that only one option is chosen. If x_i is a binary variable for feature i, then the following constraint ensures that the multiple variables in the set $\mathcal{C}$ are only equal to one, one at a time. For example, in graph coloring, we have a set of binary variables that correspond to different colors and we want each node/variable to choose one color only.

$$\sum_{i \in \mathcal{C}} x_i = 1 \tag{2}$$

This example demonstrates a linear constraint, which can be built into the optimization model using techniques shown in the "Learn to formulate problems" guide [**1**].

There are many different types of constraints that can be modeled as a quadratic optimization problem so that they are suitable for quantum annealing.

3.3 Unconstrained problems

There are other problems that, despite having no constraints, are complex.

In the following optimization problem, for example,

$$f = \sum_i a_i x_i + \sum_{i,j} w_{i,j} x_i x_j \quad (3)$$

the goal is to find a set of values $(x_1, x_2, ..., x_n)$ such that the function f is minimized (or maximized). If the variables x_i are integer or binary, this becomes a hard optimization problem.

While we can encode constraints as a set of linear and quadratic terms in the objective above, this is not always the most efficient approach.

One example of an unconstrained quadratic problem is the weighted maximum cut for the classification of nodes into two groups. For example, suppose that each variable x_i represents a user or an organization. An edge is defined between two nodes if there is a relationship between the two users or organizations. The weight of the edge represents the strength and type of relationship. Solving the optimization problem above will determine whether variables take the value of 1 (belonging to one group) or the value of -1 (belonging to the opposite group). One real-world example of this is the detection of the structural imbalance of the terrorist's network. Using QUBO formulation, we can assign the values of 0 and 1 to two groups of terrorists. For more information please see [**5**].

3.4 Variable types

Optimization and combinatorial problems with discrete and binary variables are hard to solve. These problems map to quadratic unconstrained binary optimization (QUBO) or other discrete optimization problems that are well suited for quantum annealing and hybrid solvers.

Integer variables require a logarithmic number of binary variables, and any integer problem can be represented efficiently as a binary problem.

Another binary representation of integer variables, which is called one-hot encoding, uses as many binary variables as there are integer values. For example, if ten categories are representing different digits, we can use ten binary variables x_i, where the value of x_i is equal to one, if the category i is chosen. This will require a constraint on the value of the variables. Despite the use of the constraint, problems with categorical variables are considered good candidates for discrete optimization with quantum annealing because they are NP-hard.

Optimization problems involving only continuous variables may be best handled by linear and quadratic programming packages that are available freely, or by other commercial optimization software.

4 Choosing the right problem

Putting all we learnt together, the best problems for getting started with quantum annealing are the ones in which:

- There is a complex optimization involved

- The variables are discrete
- The optimization is quadratic
  - The objective function is quadratic and / or
  - Constraints can be efficiently represented as a quadratic objective

5 Future

Despite the rules set out in this document, other types of optimization problems may also be suitable for quantum annealing. For example, quantum annealing can be used for machine learning applications [**4**], and material simulation [**2**, **3**].

Furthermore, there may be efficient hybrid algorithms for specific classes of problems that may not, at first glance, appear to be the best choice for quantum annealing. For such problems, a hybrid quantum-classical approach may be effective. For example, a class of constraint binary optimization problems can be efficiently solved by using a hybrid solver that works with the original constraints without explicitly relaxing them as a penalty objective.

References

[1] Learn to formulate problems, https://docs.dwavesys.com/docs/latest/doc_bqm_formulation.html (2020).

[2] Harris, R., et al. "Phase transitions in a programmable quantum spin glass simulator." Science 361.6398 (2018): 162-165.

[3] King, Andrew D., et al. "Observation of topological phenomena in a programmable lattice of 1,800 qubits." Nature 560.7719 (2018): 456-460.

[4] Amin, Mohammad H., et al. "Quantum boltzmann machine." Physical Review X 8.2 (2018): 021050.

[5] Mapping Militant Organizations, Stanford University, last modified February 28, 2016, http://web.stanford.edu/group/mappingmilitants/cgi-bin/.

[6] Neven, H., Denchev, V.S., Rose, G. and Macready, W.G., 2012. QBoost: Large Scale Classifier Training with Adiabatic Quantum Optimization. In ACML (pp. 333-348).