Twister Tuning: Optimizing Insurance Claim Handling To Minimize Costs

Karl Kreuze, Kevin Chang, Gabriel Maravilla, Brian Choi, Nicholas Nguyen

April 13, 2025

Abstract

This document presents the results of a quantum algorithm designed to optimize insurance claim Handling to minimize costs. A classification of tornado severities are collected and converted to a binary encoding and then input to a quadratic unconstrained binary optimization (QUBO) problem. This conversion to a QUBO problem allows a quantum computer to efficiently solve the problem using a superposition of states to solve the problem without needing to exhaustively search through all possible solutions. With the increasing change in climate, the need for insurance companies to efficiently handle claims is becoming more important. Our derivations show that quantum computers will be very useful in the future for solving these types of problems.

Derivation of Cost Function

We are given the following variables as given by a tornado severity classification:

$$\begin{aligned} N_{ik} &= \# \text{ of tornadoes of severity } i \text{ in week } k \\ N_i &= \sum_{k=1}^{K} N_{ik} = \# \text{ of tornadoes of severity } i \\ M_{ij} &= \text{claims of severity } i \text{ servicable by skill } j \text{ per day} \\ W_i &= \text{weeks where 90\% of claims must be serviced} \end{aligned}$$

We are additionally given the following constraints:

$$\sum_{k=1}^{W_i} \sum^{j} 7X_{ijk} M_{ij} \geq \min(\sum_{k=1}^{W_i} N_{ik}, 0.9N_i), \quad \forall i$$

$$\sum_{k=1}^{4} \sum^{j} X_{ijk} M_{ij} \geq N_i, \quad \forall i$$

$$X_{ijk} \geq 0, \quad \forall i, j, k$$

where X_{ijk} is the number of workers servicing claim of severity i serviced by skill j in week k.

The top constraint says that 90% of claims must be serviced in W_i weeks. The second constraint says that all the claims must be serviced in 4 weeks.

Designing the Quantum Register

In the given spreadsheet, we have a thousand claims total, so we figured that 40 bits in a register, with 8 bits for each level of severity, would be enough to represent the number of workers servicing claims of severity i. Thus, we will put a 40 bit superposition state into the QUBO circuit. If we get an output of `0b100000001`, we may split up the bits into 5 groups of 8 bits, and convert each block into the corresponding skill level's number of workers. In this example, we would have one worker of skill level 1, and 1 worker of skill level 2.

Sum of Workers

Our cost function will start with the sum of workers of various skill levels.

$$J(x)_{x \in \{0,1\}^{40}} = \sum_{n=1}^{N} X_n$$

Next, we'll add penalties corresponding to the constraints.

$$J(x)_{x \in \{0,1\}^{40}} = \sum_{n=1}^{N} X_n + \lambda_0 \left(\sum^{j} \sum^{i} (X_j M_{ij} - N_i) \right)^2 + \lambda_1 \left(\sum^{j} \text{relu}(-X_j) \right)^2$$

The first penalty grows whenever we are not servicing all the claims. The second penalty grows whenever we have negative values in the register. The relu function is defined as:

$$\text{relu}(x) = \begin{cases} 0 & x < 0 \\ x & x \geq 0 \end{cases}$$

We'll make the penalty values very large, so that the cost function will be minimized when the constraints are satisfied.

Testing

Unfortunately, we were unable to test the QUBO circuit because of the release of Qisket 2 causing dependency issues with Nexus Labs

Additional constraints

We'd need to match workers to weeks to work, and match them to the claims they are servicing. This new cost function would include our given constraint above about servicing 90% of claims in W_i weeks.

Future Work

Additional constraints such as commute time, geography, and virtual versus in-person work would need their own cost functions to be evaluated. For example, a penalty could be added for a virtual worker servicing an in-person claim, and this could be represented by a function that checks the skill of the worker and compares it to the severity of the claim, and adds a penalty based on the given table.