Application Note

Optimizing Store Locations Using QuEra's Quantum Computer

The Best Way To Quantum

Feb 2023

Jonathan Wurtz, QuEra Computing

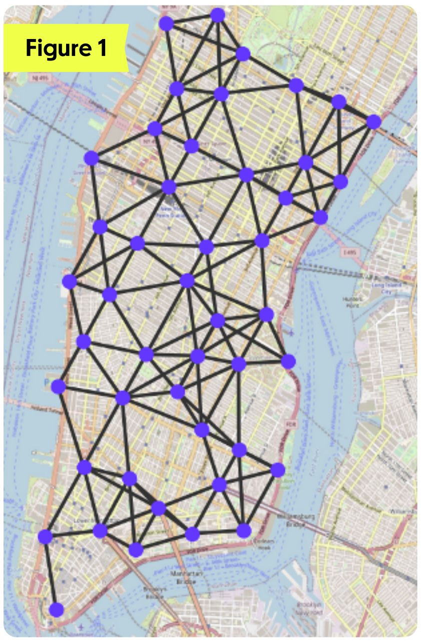
In this application note, we demonstrate how to leverage the power of QuEra's Aquila computer - operating in "analog quantum computing" mode - to optimize store placement in Manhattan. We will walk through the process of solving an example problem, followed by discussing what makes a good optimization problem for Aquila.

An example problem: Store placement in Manhattan

Suppose you want to break into a new market by establishing many new coffee shops in Manhattan. As an initial step, you scout out many viable shop locations, which are the blue dots shown in Fig. 1. However, it is not viable to put a shop at every location: some are too close together and may overlap in customer coverage. This is an independent set restriction: optimal solutions cannot have two coffee shops within a certain distance. Satisfying that restriction, you want to in addition maximize the number of shops. Edges between locations, shown in Fig. 1, encode the independent set restriction that no adjacent locations can both have stores simultaneously.

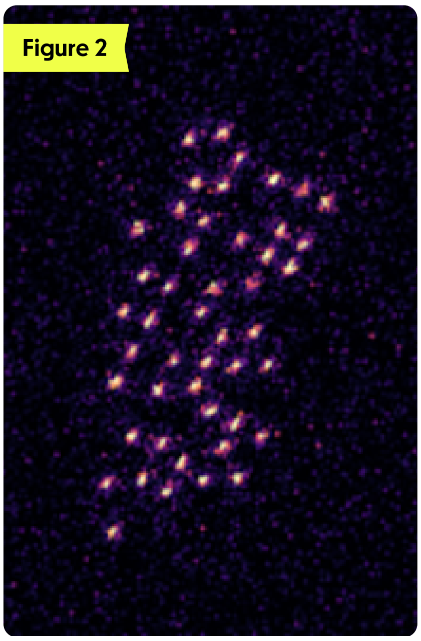
There are many variations on this problem; for example, certain locations may be more valuable or viable; locations may be placed incrementally; or there may be more advanced restrictions. These variations and more may reasonably be solved using Aquila.

Figure 1



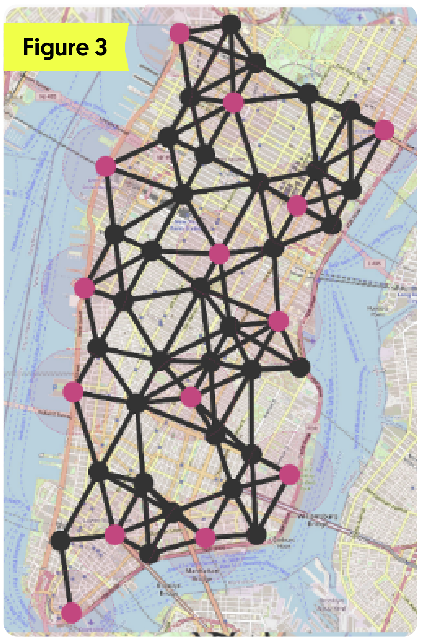
An example set of store locations. Each vertex is a potential location; each edge is an independent set restriction that no adjacent locations can both have stores simultaneously.

Figure 2



A photograph of individual atoms positioned in Aquila's atomic array encoding the store placement problem. In this example, each atom corresponds to a potential store location.

Figure 3



An example maximum independent set solution shown in red vertices.

Encoding the problem using QuEra's Aquila

In Aquila, arrays of up to 256 individual atoms (qubits) are arranged using laser tweezers, and the qubit is encoded into the electronic state of a valence electron. Quantum computations are implemented via precise optical laser pulses, which manipulate the electronic state. The user executes quantum programs on Aquila, such as optimization, by specifying the time dependence of the laser waveform, as well as choosing the positions of atoms arbitrarily on a two-dimensional space.

By choosing the positions of atoms, users encode a particular optimization problem onto the Aquila hardware. By choosing the waveforms of the lasers, users design a program such that Aquila prepares a particular quantum state. Upon measuring the state, the wavefunction collapses into a measurement of 0s and 1s, which may be used either directly or indirectly to solve the optimization problem. This process may be more effective than existing classical methods because of entanglement, superposition, and interference [1], though the exact mechanisms are still a subject of active research.

Solving the problem on Aquila

Computing the optimal solution on Aquila is simple. An example jupyter notebook, which you can run yourself, is here [4]. The quantum algorithm we implement is based on the quantum adiabatic algorithm, which evolves the quantum state to slowly "anneal" into the solution. The state is repeatedly prepared and measured on Aquila, which provides a set of bit string solutions, each encoding one possible solution.

The graph problem is directly encoded into atoms by configuring Aquila to place atoms in an analogous configuration, as shown in Fig. 2. Then, lasers excite the atoms, causing them to excite and spread entanglement throughout the system. By choosing the laser amplitude and phase as a function of time, the final state encodes the solution, with measurement occurring through the absence or presence of atoms in the array. For more details on the specifics of the quantum evolution, see the example notebook [4].

While QuEra's Aquila device is powerful, it is not perfect. For example, quantum fluctuations or decoherence may spuriously cause violations in the independent set constraint or provide solutions that are not maximal. For this reason, we perform a set of minimal classical post-processing to the optimization algorithm by adding a simple "greedy" subroutine to correct any independent set violations and add any missing vertices.

These simple "greedy" algorithms serve as a good classical baseline for classical-only performance: the all-zeros or all-ones bitstring is simple to compute classically and serves to compare quantum and classical performance. A comparison of classical vs. quantum performance is shown in Fig. 4. The relative performance of the quantum and classical algorithms depends on the particulars of the algorithm. For additional details on these classical post-processing steps, see the example notebook on the QuEra Computing GitHub page.

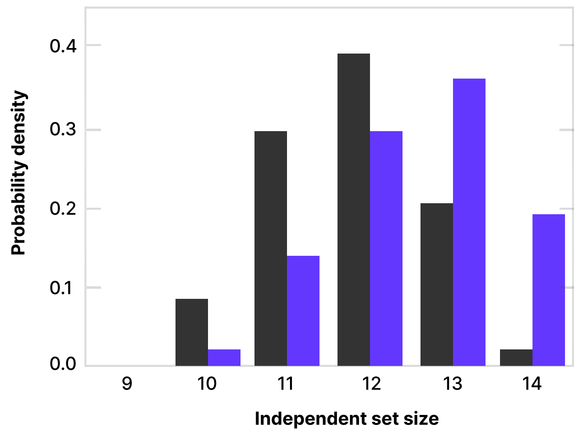


Figure 4

Performance of Aquila solving the MIS problem (purple), in comparison to a simple greedy heuristic classical optimizer (black). The program is nondeterministic and produces a different independent set each time. The quantum (classical) algorithm has an 18.8% (2.7%) chance of finding the MIS, and finds an average independent set size of 12.6 (11.8).

Results

Results of Aquila solving the store placement problem are shown in Fig. 3. By choosing a good quantum excitation protocol and taking advantage of the coherent quantum evolution of the state, we find that the quantum version of the algorithm performs better than a simple greedy heuristic classical algorithm. In this way, using Aquila lets us solve the optimization problem and maximize the number of locations to place stores. It is important to note that more sophisticated classical algorithms exist, and they currently outperform the quantum optimizer.

While this was a simple optimization problem, Aquila is capable of much more. Here, we solved a relatively "small" problem with "only" 47 nodes, which is on the edge of brute-force classical optimization, but Aquila can solve much larger problems with over 200 nodes if needed. Aquila can give solutions in as little as a single measurement; however in order to accumulate adequate statistics we take 100 measurements, which takes ~25 seconds.

What makes a good optimization problem for Aquila?

Not every optimization may be a good fit for Aquila's near-term hardware. While there are methods to encode a wide range of optimization problems onto neutral atom hardware [2], one should consider the overhead in qubit numbers. For example, arbitrary connectivity graph problems may be encoded on Aquila but may only be around tens of bits large. However, there are classes of optimization problems that do fit naturally onto Aquila and have real-world applicability and scale. These problems are two dimensional, such as geographic locations on a map. This is because Aquila's atoms can be arranged in arbitrary configurations in a 2d plane, so local interactions are naturally encoded. In particular, the maximum independent set (MIS) problem on 2d geometric graphs is perfectly encoded by the interaction of Aquila's atoms and, thus, is a good example problem.

Next steps

To try Aquila for yourself, head over to Amazon Braket, or contact QuEra at info@quera.com.
This example plus several others, which can be run using Amazon Braket, are also provided on our GitHub page.

About QuEra

Located in Boston, QuEra Computing is the best way to quantum. We make of advanced quantum computers based on neutral-atoms, pushing the boundaries of what is possible in the industry. Founded in 2018, QuEra is built on pioneering research recently conducted nearby at both Harvard University and MIT. We are building the industry's most scalable quantum computers to tackle useful but classically intractable problems for commercially relevant applications. Our signature machine, Aquila, is available now for general use over the Amazon Braket cloud.

References

1. S. Ebadi et. al. Quantum optimization of maximum independent set using Rydberg atom arrays. SCIENCE 376, 6598 (2022).
2. Nguyen et. al. Quantum optimization with arbitrary connectivity using Rydberg atom arrays. Nguyen et. al. PRX Quantum 4, 010316
3. Wurtz et. al. Industry applications of neutral-atom quantum computing solving independent set problems. arXiv:2205.08500
4. "Manhattan Coffee Shops" Jupyter notebook in QuEra's Github page.