

# AQUA@home

A white paper

December 9 2008

Dr. Geordie Rose  
Founder and Chief Technology Officer  
D-Wave Systems Inc.  
[rose@dwavesys.com](mailto:rose@dwavesys.com)



# D:wave

D-Wave is developing a new class of high-performance computing system designed to solve complex optimization problems, with an initial emphasis on machine learning applications.

D-Wave systems are architected around an innovative processor that uses a computational model known as adiabatic quantum computing (AQC). These processors exploit quantum effects to solve optimization problems in a new way. They are fabricated using superconducting metals instead of semiconductors and are operated at ultra-low temperatures in a magnetic vacuum.

[www.dwavesys.com](http://www.dwavesys.com)



## INTRODUCTION

An **algorithm** is a prescription for solving a problem.<sup>1</sup> An example is provided in the Wikipedia entry on algorithms, where the problem to be solved is getting a broken lamp to work (see Figure 1). In this case, a sequence of steps is followed, with the problem-solving tactics at each step depicted in a flowchart.

Usually it is straightforward to evaluate how well a proposed algorithm works. One way to do this is by counting how many steps it takes to solve a problem of a given size.

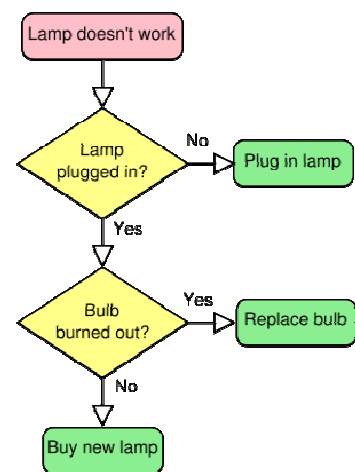
Some algorithms are better than others. In the lamp example in Figure 1, a perfectly valid algorithm could require a trip to Siberia between checking that the lamp was plugged in and checking if the bulb was burned out. The addition of the Siberia leg doesn't affect the outcome of the algorithm—the lamp still gets fixed. This modified lamp fixing algorithm would still generally be viewed as inferior to the original.

Sometimes a newly discovered prescription for solving a problem is clearly superior to all the others that have preceded it. When this happens, the incumbent approaches are often abandoned and everyone who shares the problem in question gets an immediate and sometimes spectacular performance boost.<sup>2</sup> In the lamp scenario, imagine everyone used the “visit Siberia” version, until some brilliant computer scientist pointed out you could omit this step and still get your lamp fixed. Most would immediately switch to the better approach.

Sometimes it is very difficult to tell whether a particular algorithm could be made better. Over the past two decades a lot of progress has been made in understanding one type of limitation to how good algorithms can be. It turns out that the laws of physics set limits on how efficiently problems can be solved. Just as physics places limits on how fast something can travel (the speed of light), it also places fundamental limits on how few steps (or bits of memory) are required in order to solve a given problem. These limits are independent of how smart a coder is or what types of hardware are available.

Conceptually, the reason that the laws of physics matter for computation is that computers are built out of stuff, which is of course subject to the laws of physics. The information which is stored and processed in a computer is subject to these rules, and therefore inherits all of the constraints that these rules imply.

**Figure 1.** From Wikipedia's entry on algorithms. This particular algorithm is designed to solve the problem of turning a broken lamp into a working lamp.



<sup>1</sup> An interesting article about algorithms: <http://www.crmbuyer.com/story/33488.html>

<sup>2</sup> A famous example of this involves algorithms for factoring; see <http://dwave.wordpress.com/2007/08/27/algorithms-vs-hardware-the-throwdown/> for references.



## QUANTUM INFORMATION AND ALGORITHMS

Progress in understanding the relationship between physics and computer science has led to the development of a new category of scientific investigation, where the ultimate limits of computation are studied. This category is called **quantum information science**. The basic premise is that the fundamental laws of physics—which are not necessarily the ones relevant to today’s processors and computing systems—set the ultimate limits on how good algorithms can be. Thirty years ago this category of science did not exist. Today, you would be hard-pressed to find a single university without faculty studying the fascinating open problems in this field.

One of the most tantalizing discoveries made in the course of these investigations is that there are algorithms that are allowed by nature but that cannot be run on the types of computing systems we have available today.<sup>3</sup> These **quantum algorithms** require operations that are allowed by quantum mechanics but not by classical physics.<sup>4</sup> Running them requires special hardware systems (usually called **quantum computers**) designed to be able to perform the “quantum operations” required by the quantum algorithm. Some of these algorithms are a lot better than the best known classical algorithms. Some of them would be game changers for the problem they are designed to solve.

A well known example is Shor’s Algorithm, which is a quantum algorithm for factoring.<sup>5</sup> Factoring is a hard problem used as the basis for certain cryptosystems; being able to factor large products of prime numbers quickly allows you to break certain commonly used codes. Shor’s Algorithm reduces the number of steps required for factoring by an enormous amount. If a modern PC could run Shor’s Algorithm it could break the code protecting your credit card-based online purchase in about one second, whereas using the best known classical factoring algorithms this same feat would take the world’s largest supercomputer more than a day.<sup>6</sup>

## ADIABATIC QUANTUM ALGORITHMS

One important class of quantum algorithms are the **adiabatic quantum algorithms**.<sup>7</sup> Initially discovered in 2000 by a group at MIT, these algorithms solve an important class of hard computational problems in a novel way.<sup>8</sup> These algorithms have generated a large amount of controversy in the computer science and physics community because there is **no way to easily analyze how long it takes them to solve the problems they are designed for**.

---

<sup>3</sup> The Quantum Algorithms Zoo <http://www.its.caltech.edu/~sjordan/zoo.html> lists all known algorithms of this sort.

<sup>4</sup> Quantum mechanics is the framework upon which all modern understanding of matter and energy is built. [http://en.wikipedia.org/wiki/Quantum\\_physics](http://en.wikipedia.org/wiki/Quantum_physics) has references to good introductions to quantum mechanics.

<sup>5</sup> [http://en.wikipedia.org/wiki/Shor%27s\\_algorithm](http://en.wikipedia.org/wiki/Shor%27s_algorithm)

<sup>6</sup> [http://en.wikipedia.org/wiki/Key\\_size](http://en.wikipedia.org/wiki/Key_size)

<sup>7</sup> The hardware required to run these algorithms is typically referred to as an adiabatic quantum computer, see for example <http://arxiv.org/abs/quant-ph/0211152>

<sup>8</sup> <http://www.sciencemag.org/cgi/content/abstract/292/5516/472>



Some theoretical computer scientists believe that they provide no advantage whatsoever, whereas many studying these problem solving approaches believe that there may be exponential reductions in the time it takes them to solve certain problems over the best known classical algorithms.<sup>9</sup>

Resolving the question of how long it takes for an adiabatic quantum algorithm to compute the answer to a problem is a very important scientific question related to some of the biggest open problems in computer science. In addition to being an important academic problem, this issue is of extreme practical importance to the effort at D-Wave, which is focused around building hardware that runs these algorithms.

## USING QUANTUM MONTE CARLO TO CALCULATE RUNTIMES

Recently a technique has been introduced for calculating the time it takes an adiabatic quantum algorithm to solve relatively small problems by **simulating the quantum computer running the algorithm**.<sup>10</sup>

While this approach unfortunately can't tell us how these algorithms perform on large problems, it is sufficient to predict the expected performance of adiabatic quantum computers of up to about 128 qubits.

This simulation approach is based on a technique called Quantum Monte Carlo (QMC). The approach is embarrassingly parallel and can efficiently make use of large numbers of independent processors.

D-Wave and Dr. Peter Young<sup>11</sup> (a co-author on the QMC paper referenced previously and a professor at UCSC) have built a distributed QMC platform to calculate the runtime of an adiabatic quantum algorithm which will be run in experimental hardware at D-Wave. The project is called AQUA (Adiabatic QUantum Algorithms), with the distributed version called AQUA@home.

The distributed computing technology is based on BOINC, the same technology that enables SETI@home and a host of other large-scale distributed scientific computations.<sup>12</sup>

AQUA@home is running on a server at <http://aqua.dwavesys.com/> that is set up to accept volunteer cycles from individuals who wish to contribute computing resources to the project.

---

<sup>9</sup> One of the world leading organizations in studying this problem is at MIT; see [http://www.rle.mit.edu/xqit/research\\_01.htm](http://www.rle.mit.edu/xqit/research_01.htm)

<sup>10</sup> <http://arxiv.org/abs/0803.3971>

<sup>11</sup> <http://physics.ucsc.edu/~peter/>

<sup>12</sup> <http://boinc.berkeley.edu/>



## THE HARDWARE BEING SIMULATED

The processors that are being simulated in the AQUA@home project are currently being fabricated at D-Wave, and are code-named Rainier. They are formed out of lithographically defined metal circuits. The metal used is niobium, which becomes a superconductor at very low temperatures. Superconductors have many exotic properties that make them particularly well suited to certain types of high performance processor design. An extensive overview of superconducting processor technology can be found in this reference.<sup>13</sup>

The processor being simulated contains 128 qubits. This makes it small enough to study using the QMC technique used in AQUA@home. Extensive descriptions of the hardware system and processor can be found at <http://dwave.wordpress.com>.



**Figure 2.** The 28-qubit processor used in the D-Wave / Google Supercomputing 2007 image matching demonstration.

## THE PROBLEM THE ADIABATIC QUANTUM ALGORITHM SOLVES

The Rainier processor is specifically designed to solve a particular hard problem called **quadratic unconstrained binary optimization**, or QUBO.<sup>14</sup> Solving a QUBO problem requires finding a set of  $N$  binary variables  $\{x_i\}$  that minimize an objective function of the form

$$E(x_1, \dots, x_N) = \sum_{i \leq j}^N Q_{ij} x_i x_j$$

where  $\mathbf{Q}$  is a real valued upper triangular matrix. Any particular matrix  $\mathbf{Q}$  is called a **problem instance**. Generally the matrix  $\mathbf{Q}$  arises from an application that requires the solution of a QUBO. One example of such an application is machine learning, a focus area for D-Wave's algorithms development team.<sup>15</sup>

<sup>13</sup> <http://www.nitrd.gov/pubs/nsa/sta.pdf>

<sup>14</sup> <http://www.springerlink.com/content/n4372840250660v3/>

<sup>15</sup> <http://arxiv.org/abs/0811.0416>



## WHAT IS AQUA@home CALCULATING?

AQUA@home is calculating the runtime of a set of problem instances of the type shown above for problem sizes ranging from 8 to 128 variables. We generate 50 problem instances for each of the following sizes: 8, 16, 32, 48, 72, 96 and 128 variables, for a total of 350 problem instances. We use AQUA@home to calculate the runtime for each instance. This is the desired output of the computation.

This information can be used for two important purposes. The first is that it tells us how quickly the hardware being simulated can be operated and still be expected to solve the problem in question. The second is that it tells us how the runtimes are scaling with increasing problem size. Knowing how a problem scales provides us with vital information about whether that problem is a good fit to the adiabatic approach. For example if the runtimes are exponentially increasing with problem size, that tells us that the problem instances being studied are not a good fit to the hardware.

## WHAT WILL HAPPEN TO THE DATA GENERATED BY AQUA@home?

Data generated by AQUA@home will be made public in the form of graphs of median runtime vs. problem size for all hardware types, quantum algorithms and problem instance classes run on AQUA@home as the data becomes available. In addition we will publish full descriptions of the science behind the computations in scientific papers on the preprint server arxiv.org, and certain of these results will be submitted for peer review in leading scientific journals, such as Physical Review Letters.

## WHERE CAN I LEARN MORE ABOUT D-WAVE AND THE SCIENCE OF AQUA@home?

An overview of the science of AQUA@home:

<http://arxiv.org/abs/0803.3971>

An introduction to adiabatic quantum algorithms:

<http://arxiv.org/abs/quant-ph/0104129>

An introduction to superconducting adiabatic quantum computers:

<http://arxiv.org/abs/quant-ph/0403090>

Information on D-Wave:

<http://dwave.wordpress.com/>