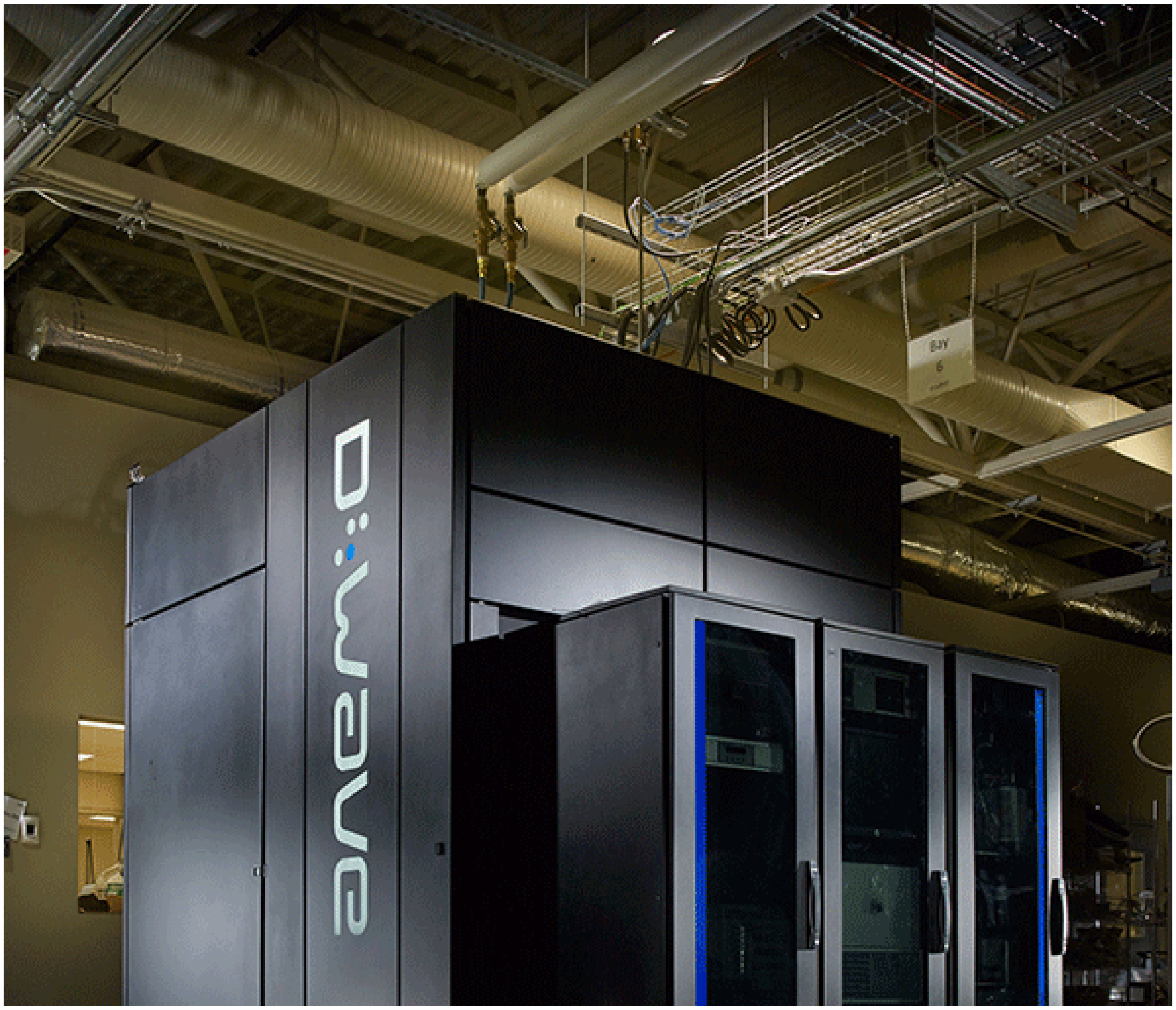
Quantum Computing 101

Machine Design
E. D. Dahl, Senior Research Scientist, D-Wave Systems
Thu, 2015-01-08 10:16

Here's a look at the first commercial quantum computer.



Quantum computing could change the face of computing over the coming decades, especially when it comes to quickly solving certain classes of problems such as optimization, code cracking (cryptography), and machine learning. Google and Lockheed Martin have already climbed aboard the bandwagon, each purchasing and now experimenting with a quantum computer from D-Wave Systems.

However, quantum computing and its reliance on quantum mechanics and esoteric phenomenon such as superposition and entanglement, however, make it difficult for most people to understand it. But then most people have no trouble using today's "conventional" computers despite knowing little about transistors, tunneling, or n- and p-type semiconductor materials and compilers.

So here's a look at the basics of quantum computing. Also check out *Programming a Quantum Computer*, which further explores and explains quantum computing.

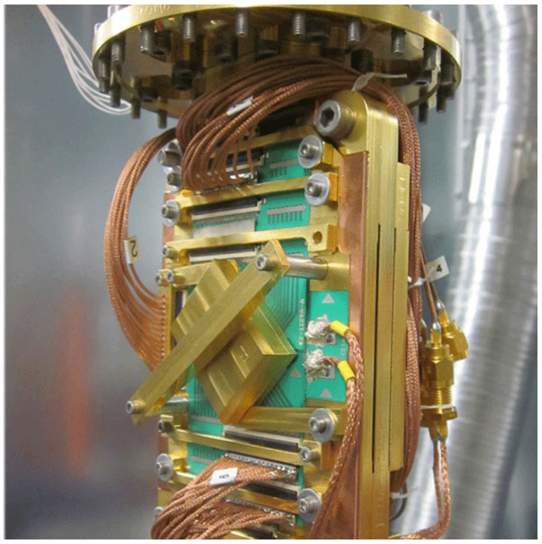
Related

A "Noiseless" Amp: One Step Closer to Quantum Computing

New Artificial Atom Could Be a Breakthrough in Quantum Computing

A Quantum Step Towards the Qubit PC

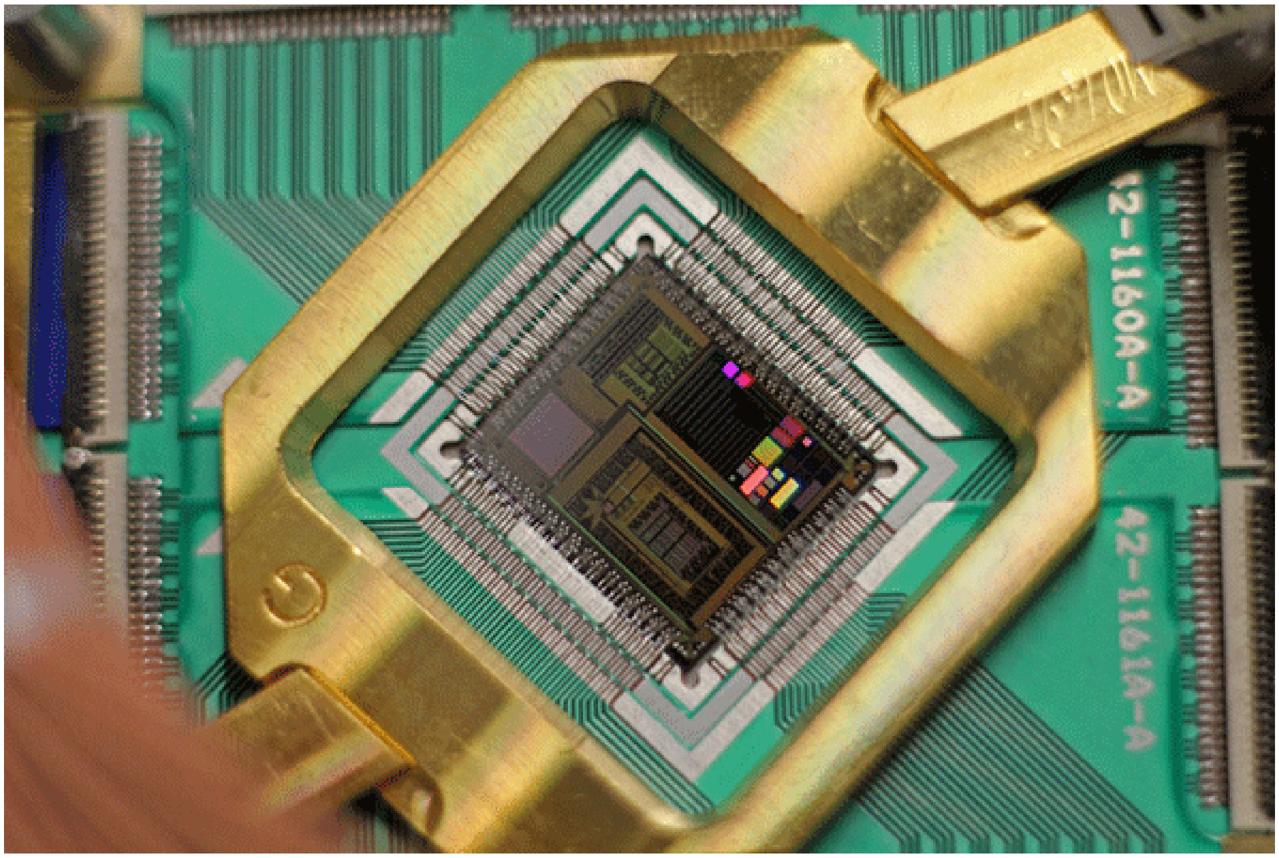
The theory



Quantum computers use an entirely different approach to problem solving than classical computers. A useful analogy is to think of a landscape with mountains and valleys. Solving an optimization problem can then be thought of as trying to find the lowest point on that landscape.

Every possible solution is mapped to a set of coordinates on the landscape, and the altitude of the landscape at those coordinates is the "energy'" or "cost" of the solution at that point. The goal is to find the lowest point on the map and get the associated coordinates, as this gives the lowest energy or best solution to the problem.

Classical computers can only "walk over this landscape." Quantum computers, on the other hand, can tunnel through it, letting them find the lowest point more quickly. In fact, they consider all possibilities simultaneously and determine the lowest-energy required to form those solutions.



Because quantum computers are probabilistic rather than deterministic, they return many good answers in a short amount of time—10,000 answers in one second on the D-Wave 2 computer. This not only gives users the best solution or single answer, but also a host of other alternatives to consider.

To program a quantum computer, users map problems into this search for the lowest point. Users interact with the computer by connecting to it over a network, as they often do with traditional computers. Problems are sent to a server interface that turns the optimization program into machine code to be programmed onto the chip. The system then executes one or more "quantum machine instructions (QMI)" to generate results.

Not quite ready for prime time

It will be quite a few years , more like decades, before everyone has a quantum-computing laptop or cell phone. The requirements for the support hardware are just too daunting.

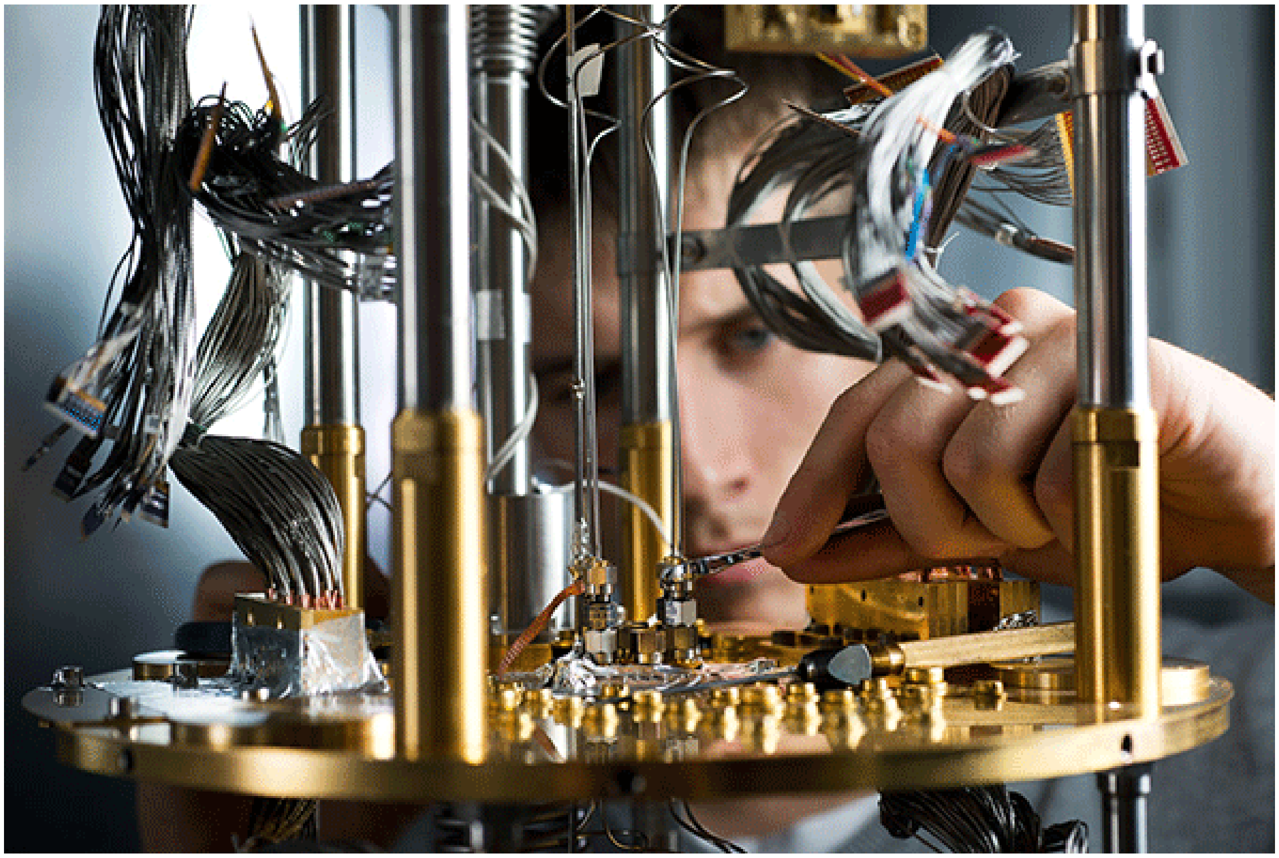
For example, D Wave's current commercial computer, the D-Wave Two, needs its qubits (or niobium wire loops) to be kept as cold as possible, with as little exposure to thermal noise or thermal vibrations as possible. This translates into –273°C, or 0.02° above absolute zero. To handle that task, it uses a dedicated closed-cycle dilution refrigerator. It consumes much of the 15.5 kW of power needed to run the 512-qubit machine.

Keeping the processors cold minimizes thermal noise—a must for inducing or setting the stage for quantum phenomenon. The processor is also kept in a near-vacuum to eliminate interference with stray molecules. Vacuum is said to be one ten-billionth that of atmospheric pressure.

The processor is also shielded from magnetic fields, a step to prevent the Earth's magnetic field from interfering with those in the niobium loops. D-Wave Two's shielding knocks the Earth's magnetic field down by a factor of 50,000. Engineers also had to construct a way to get 192 lines into the processor and one fiber-optic feed out of it without breaking the magnetic or pressure seals.

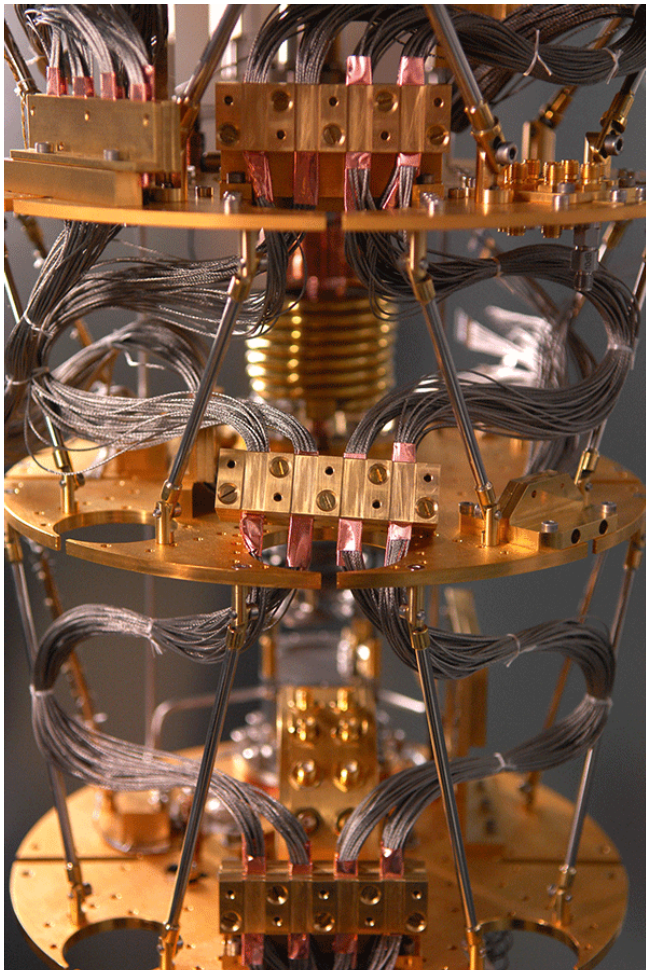
Quantum basics

Classical computers consist of registers and memories. The collective contents of these devices are referred to as the computer's state. Instructions for the computer act deterministically on this state, transforming the contents of a few registers or memory locations at a time. Quantum computers like the D-Wave, however, have no registers or memory locations and therefore do not possess anything analogous to state. Also, as noted before, instructions for quantum computers are not deterministic; instead, they return probabilistic results.



If quantum computers have no registers or memory locations, how do we learn anything from executing a QMI? The answer is that the computer returns samples from a distribution as a side effect of executing the QMI. To explain what these samples are and how this distribution is defined, we must introduce several entities that are key to the D-Wave programming model.

The first such entity is the *qubit*, which is simply a variable (q) that has a value from the set {0, 1}. Qubits hold information in quantum computers. Each qubit's behavior is governed by the laws of quantum mechanics and it lets them be in a "superposition" state—that is, both a 0 and a 1 at the same time until an outside event causes the qubit to "collapse" into either a 0 or a 1. This property is unlike anything in our everyday lives and completely nonintuitive. But it forms the basis upon which all quantum computers will be built and how they solve problems.



In the D-Wave computer, loops of niobium wire measuring 2 microns in diameter and 700 microns in circumference serve as a qubits. After the loops are supercooled, two currents can be generated that circle each loop in opposite directions. When the current travels in one direction, it is considered to be a 1; when it travels the opposite direction, it's a 0. When it travels both directions, it's a qubit.

The programming model used in D-Wave computers does not let programmers directly control the current flowing through these qubits. Instead, they influence the qubits and the computer responds to those influences.

Programmers influence qubits in two ways. First, each qubit has an associated weight which is part of each QMI and under programmer control. In D-wave programming language, qubit q_i *has a weight* a_i. And second, a coupler lets programmers control the influence one qubit exerts on another qubit. A coupler is always identified with exactly two qubits, and so the coupler connects the qubits.

Just as a weight is associated with each qubit, strength is associated with each coupler. If a coupler connects qubits q_i and q_j, the strength of that coupler is denoted by b_{ij}.

A programmer maps a problem to a QMI, whose objective is to return the minimal values (the optimal solutions). The programmer must provide two values: the "weights" of the qubits and "strengths" of the interaction between them. Up to about 500 weights and 1,500 strengths can be specified in the current D-Wave 512 qubit processor. Further, programmers specify the number of solutions they want the computer to return.

With definitions for qubits and weights, as well as couplers and strengths, programmers can compose an objective function that defines the distribution from which the samples (or solutions) will be selected:

$$\mathrm{O(a,b;q)} = \Sigma_{i=1}^{N} a_i\, q_i + \Sigma_{<i,j>} b_{ij} q_i\, q_j$$

This objective function takes the values of the weights (a_i), strengths (b_{ij}) and qubits (q_i) as its input. For each specification of its input, the function returns an objective value, or simply an objective.

The first summation in the function encompasses all qubits. Thus the limits range from 1 to N, where N is the number of qubits in the system. The second summation covers all couplers and is denoted by the pairs of qubits corresponding to a coupler by angle bracket notation $< i, j >$. It's important to note that each QMI consists of exactly the a_i and b_{ij} values that appear in the above objective function.

With a defined objective function, programmers can describe samples and the distribution from which they are drawn. Each sample is simply the collection of q_i values for the entire set of qubits in the problem. The distribution is (ideally) an equal weighting across all samples that give the minimum value of the objective function.

Programmers working with quantum computers encode the various possible solutions to an optimization problem in terms of the qubit variables. Then they translate constraints in the optimization problem into values of the weights a_i and strengths b_{ij} so that when the objective is minimized, the qubits satisfy the constraints.

Source URL: http://machinedesign.com/cad/quantum-computing-101