D-Wave
The Quantum Computing Company™

Embedding:

The first abstraction layer: virtual vs. physical qubits

LANL / D-Wave Quantum Programming

June 9, 2016

D-Wave Systems Inc.

Denny Dahl

Three-qubit relations

- We've implemented several two-qubit relations (i.e., AND, OR, IMPLIES) via QUBOs
- A two-qubit relation has four final states…
- …and a two-qubit QUBO has three parameters
- That's enough to handle **all** two-qubit relations
- What about three-qubit relations?
- There are eight final states…
- …and a three-qubit QUBO has six parameters
- So, we're anticipating some **trouble**.

First three-qubit objective: $q_1 \vee q_2 = q_3$

q_1	q_2	q_3	$O(a_1, a_2, b_{12}; q_1, q_2)$
0	0	0	0
0	0	1	a_3
0	1	0	a_2
0	1	1	$a_2 + a_3 + b_{23}$
1	0	0	a_1
1	0	1	$a_1 + a_3 + b_{13}$
1	1	0	$a_1 + a_2 + b_{12}$
1	1	1	$a_1 + a_2 + a_3 + b_{12} + b_{13} + b_{23}$

$$a_1 = 1$$
$$a_2 = 1$$
$$a_3 = 1$$
$$b_{12} = 1$$
$$b_{13} = -2$$
$$b_{23} = -2$$

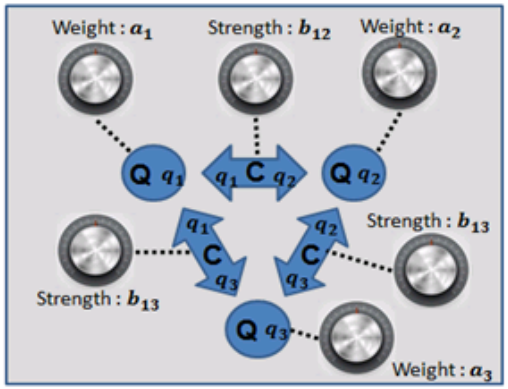
Transfer the a_i and b_{ij} values to the Three Qubits tab and verify that the valid states are correct.

Three qubit spectrum

q_1	q_2	q_3	$q_1 + q_2 + q_3 + q_1q_2 - 2q_1q_3 - 2q_1q_3$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	3
1	1	1	0

Quantum Apprentice: : $q_1 V q_2 = q_3$

Programming Model: Three qubits



$$O(a_i, b_{ij}; q_i) = a_1q_1 + a_2q_2 + a_3q_3 + b_{12}q_1q_2 + b_{13}q_1q_3 + b_{23}q_2q_3$$

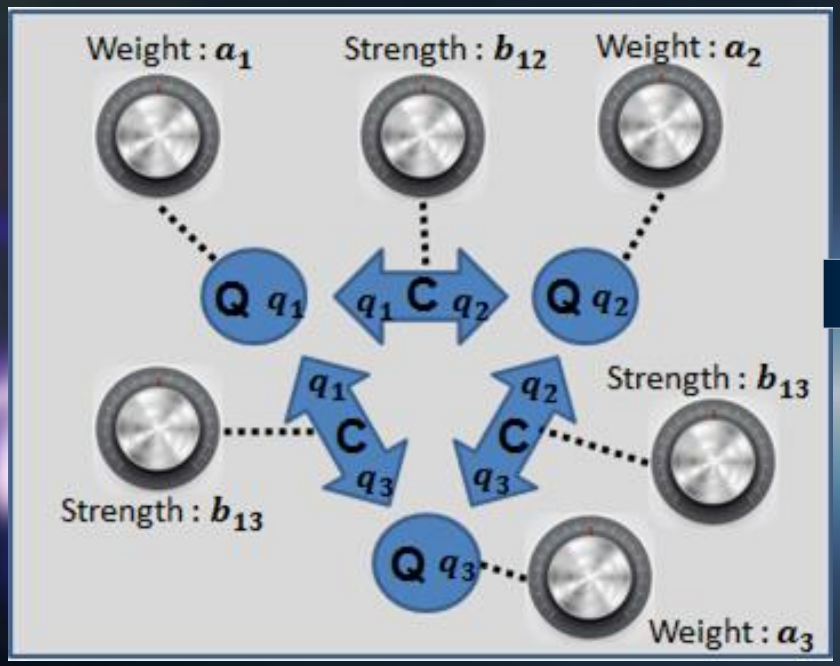
QMI (Quantum Machine Instruction) :

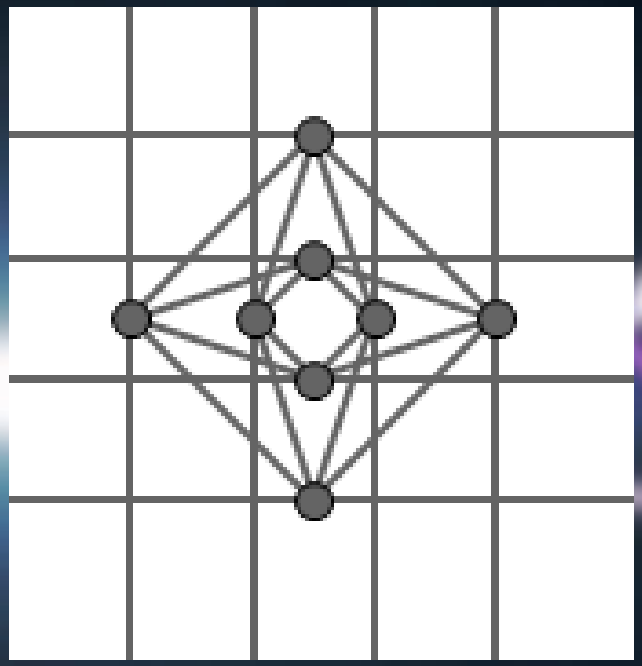
a_1	a_2	a_3	b_{12}	b_{13}	b_{23}

q_1	q_2	q_3	Objective
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	3
1	1	1	0

a_1	a_2	a_3	b_{12}	b_{13}	b_{23}
1	1	1	1	-2	-2

Attempt to map to unit cell

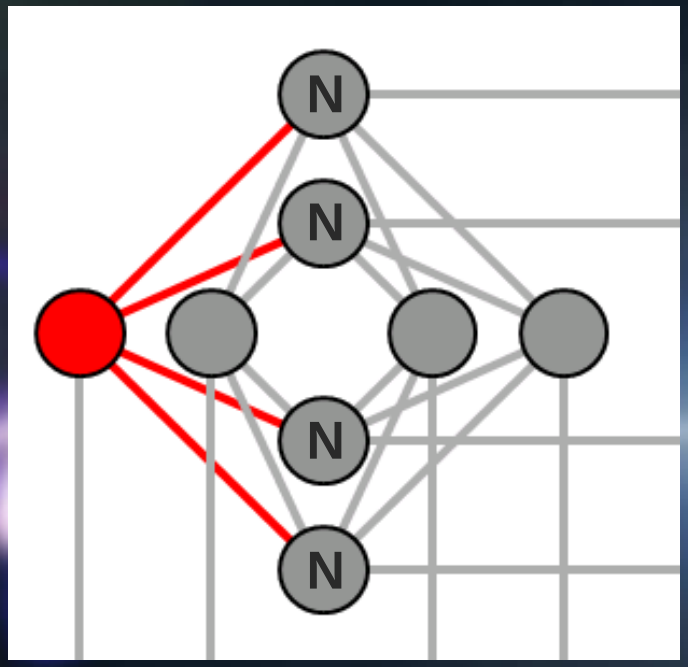




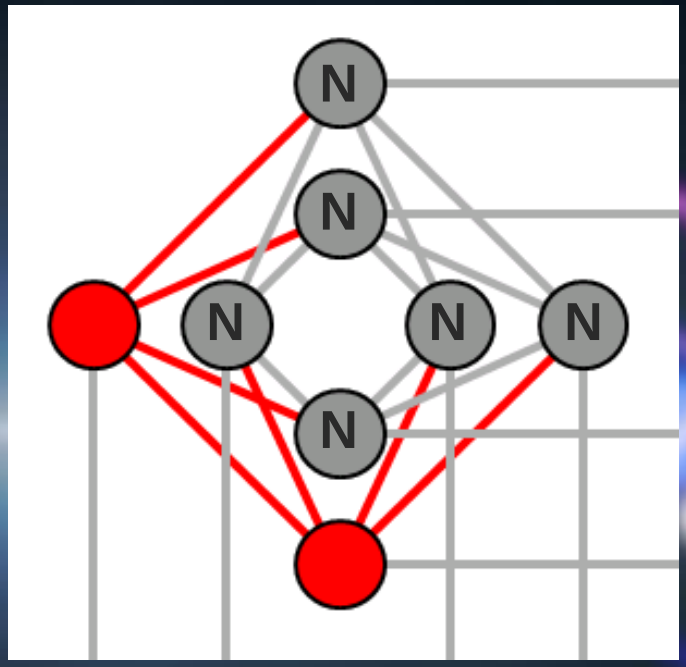
Impossible!

Idea: Qubit Chains

4 neighbors



6 neighbors



Two qubits are better than one

How do we make a qubit chain?

equal

q_1	q_2	$O(a_1, a_2, b_{12}; q_1, q_2)$	$q_1 = q_2$
0	0	0	0
0	1	a_2	1
1	0	a_1	1
1	1	$a_1 + a_2 + b_{12}$	0

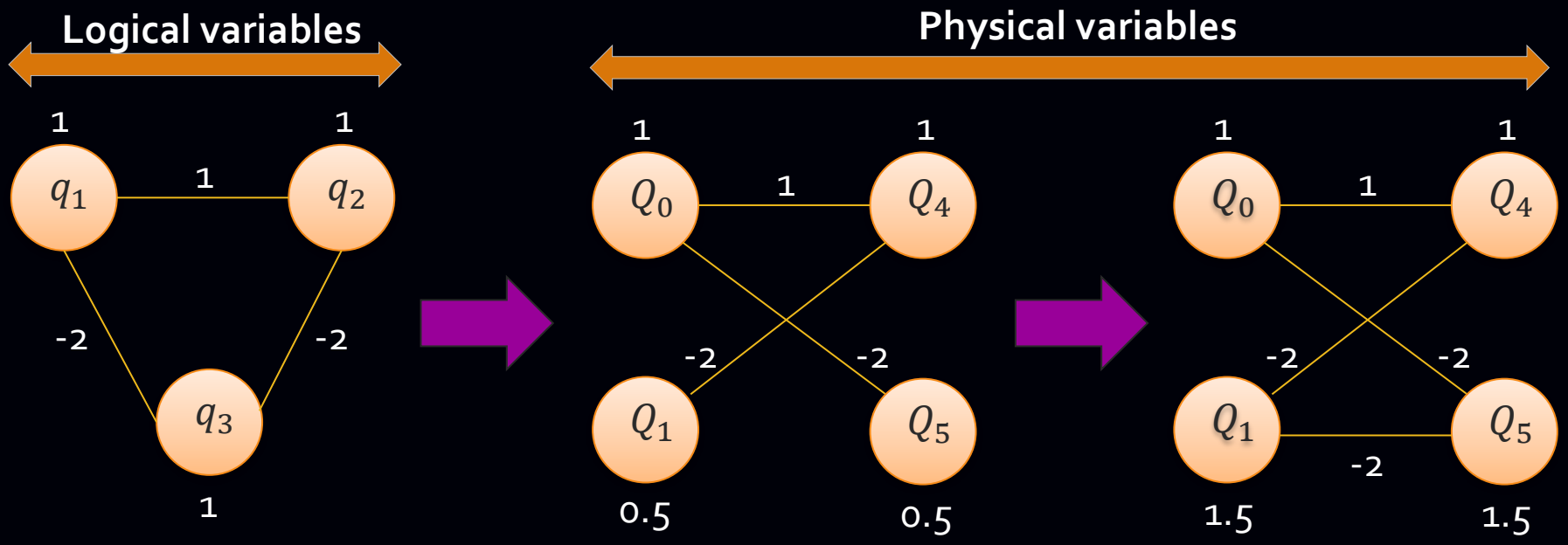
$a_1 = 1$
$a_2 = 1$
$b_{12} = -2$

This two-variable QUBO causes q_1 and q_2 to be the same.
If the qubits differ, we'll pay an *objective penalty.*

Embedding rules

1. A logical qubit can be mapped to N physical qubits as long as the physical qubits form a connected set. We call this a **chain**.
2. For each physical coupler connecting two physical qubits in the same **chain**, include QUBO terms to cause the two physical qubits to align.
3. When a logical qubit is mapped to an N-qubit **chain**, divide the weight for the logical qubit by N and apply that weight to each qubit in the **chain**.
4. Split the coupling strength between two logical qubits over all available physical couplers connecting the **chains** corresponding to the logical qubits.

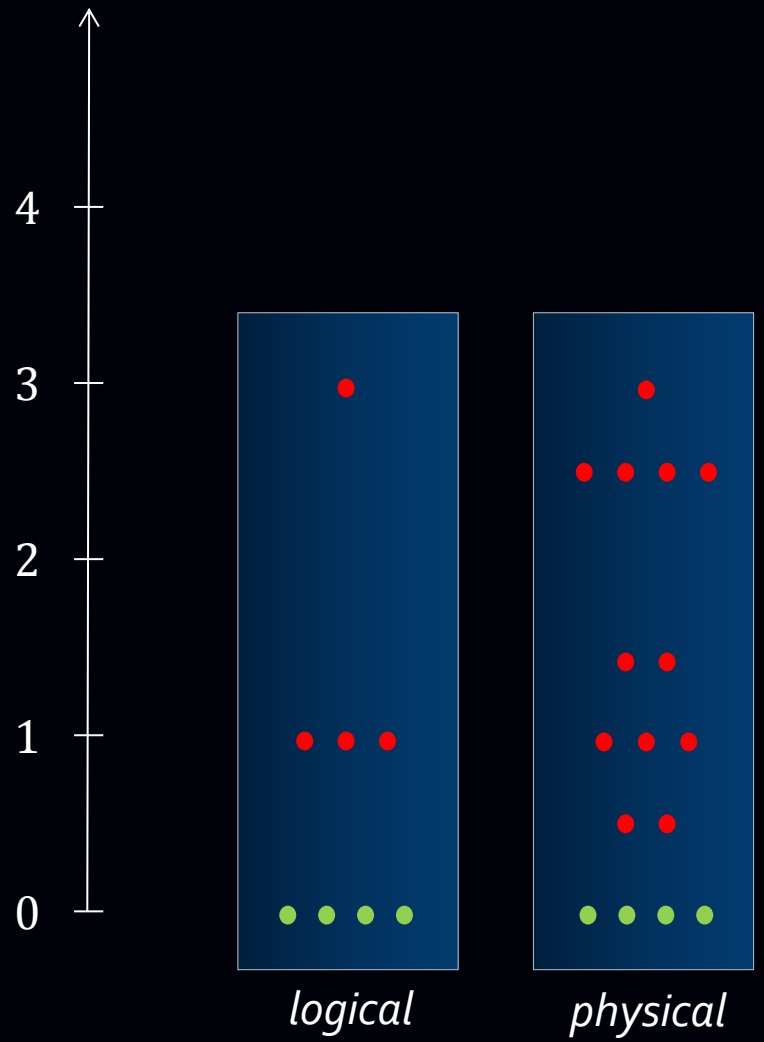
Embedding rules at work:



$$q_1 \rightarrow Q_0 \qquad q_2 \rightarrow Q_4 \qquad q_3 \rightarrow [Q_1\ Q_5]$$

Logical to physical embedding

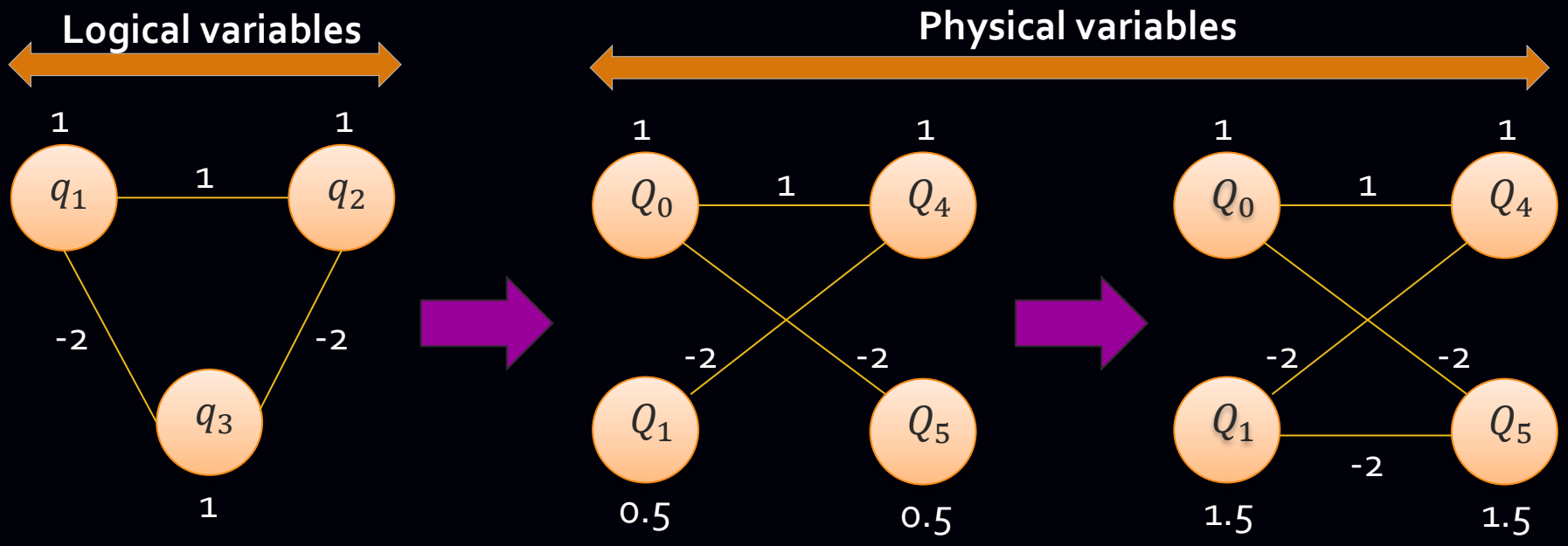
Why do the embedding rules work?



The embedding rules keep the lowest part of the spectrum intact.

Both the logical and physical spectra have the same four valid states at the bottom of the objective well.

OR via Quantum Apprentice



1. Transfer the logical problem to Quantum Apprentice's *Three Qubits* tab and note the spectrum.
2. Transfer the physical problem to Quantum Apprentice's *Four Qubits* tab and note the spectrum.

OR via Qubist

Transfer the problem from the **Four Qubits** tab in Quantum Apprentice to the **Solvers Visualizer** page in Qubist:

1. Before entering the problem data, use **Configure** to:
   a. Select the **DW2X_SYS4** solver on the **Configurations** tab
   b. Set the **Problem Type** to **QUBO** on the **Configurations** tab
   c. Set the **Number of Reads** to 1000 on the **Parameters** tab
   d. Click **View Graph Data**
2. Choose your scaling parameter for weights and strengths so that you use the entire dynamic range available in Qubist
3. Adjust the weights and strengths on the **Graph** and **Data** tabs
4. Click **Submit Problem to DW2X_SYS4**
5. On the **Solutions** tab, look at:
   a. Solutions Data
   b. Reads by Energy Chart
   c. Timing Information

Results of a 1000-sample run

99.8% of 1000 answers are valid

Solution	Energy	Occurrences
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...] Graph Show	-0.62	260
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...] Graph Show	-0.62	327
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...] Graph Show	-0.62	134
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...] Graph Show	-0.62	277
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...] Graph Show	-0.37	2

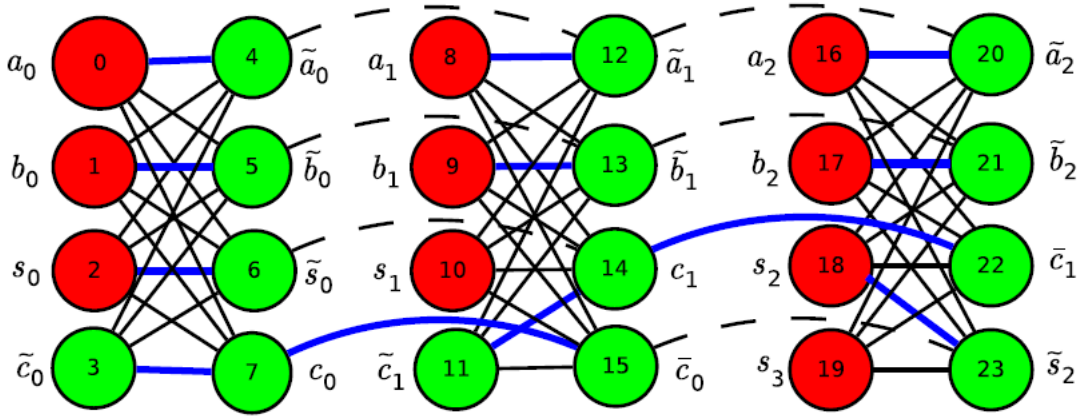
Showing 1 to 5 of 5 entries Previous 1 Next

Your mileage may differ

dw

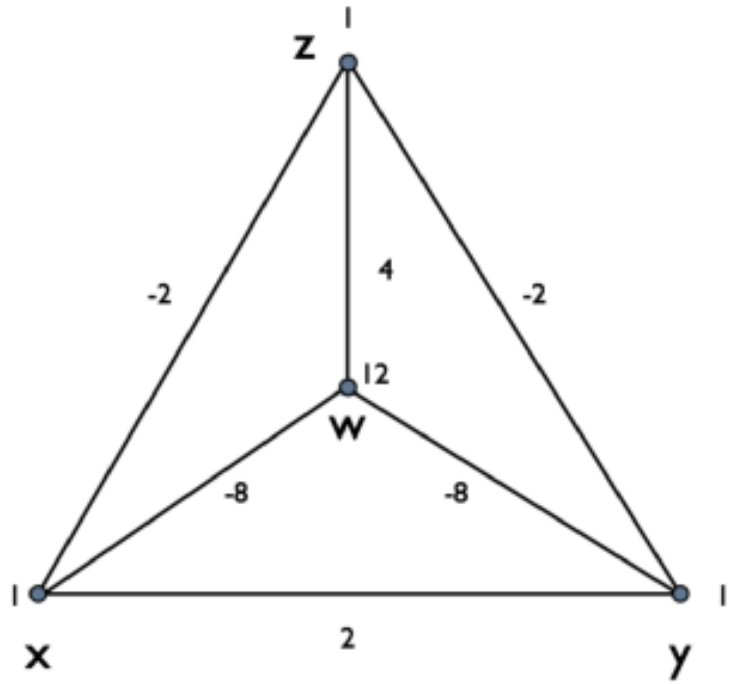
- Interactive command
- Provides similar functionality to C API via command line
- Gets connections and solvers from simulator or quantum hardware
- Displays geometry, qubit and coupler connectivity
- Embeds QUBOs into Quantum Machine Instructions
- Executes Quantum Machine Instructions
- Validates solutions
- Manages multiple workspaces – one per geometry

Example: three bit adder



A three bit adder adds two three-bit numbers in the range $[0 - 7]$ and returns a four bit result in the range $[0 - 14]$.

Half adder QUBO



c	s	y	x	objective
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	4
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	12
1	0	0	1	5
1	0	1	0	5
1	0	1	1	0
1	1	0	0	17
1	1	0	1	8
1	1	1	0	8
1	1	1	1	1

Where are we going with this?

- Constraint satisfaction problems (CSPs) can be formulated as a number of independent clauses, each of which must be satisfied (CNF = conjunctive normal form)
- Each clause typically involves a small number of boolean variables
- Karp showed that clauses with 3 terms are "hard enough"
- We've looked at three clauses with two terms and a single clause with three terms
- If we can handle 3-term clauses, we can embed general CSPs

Can we implement all 3-qubit relations?

- Not directly!
  - the Chimera topology does not contain 3-cycles
  - There are not enough parameters in a 3-qubit objective to implement all 3-qubit relations

- Conclusion:
  - We're going to need tools to mimic qubit connectivity not implemented in hardware
  - We're going to need to add ancillary qubits to provide enough parameters to implement more complex N-qubit relations

