Intro to D-Wave Leap

Cloud-based Quantum Annealing (QA)

AQC algorithm for 3-SAT E. Farhi et al, Science **292**, 472 (2001)

a) Starting Hamiltonian:

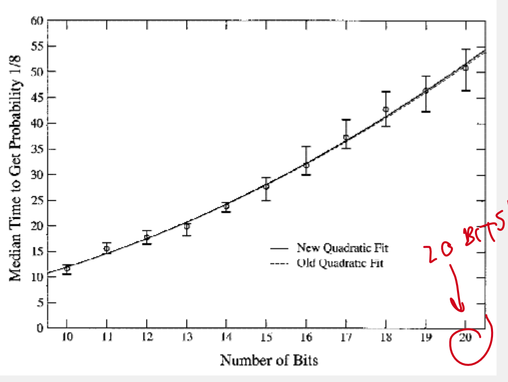
$$\mathcal{H}(t=0)=\mathcal{H}(0)=\omega\sum_{i=0}^{n-1}\left(\frac{I-X_i}{2}\right)$$

b) Final Hamiltonian:

$$\mathcal{H}(t=t_f)=\mathcal{H}(1)=\sum_{\text{all } C(i,j,k)}\left[\left(\frac{I-Z_i}{2}\right)+\left(\frac{I-Z_j}{2}\right)+\left(\frac{I-Z_k}{2}\right)-1\right]^2$$

c) Picture shows median t_f required to get the right answer with prob. 1/8 using exact integration of Schröedinger eqn for linear s(t):

$$\mathcal{H}(t)=\left(1-\frac{t}{t_f}\right)\mathcal{H}(0)+\frac{t}{t_f}\mathcal{H}(1)$$



3-SAT is a particular case of QUBO

- Let's expand the 3-SAT $\mathcal{H}(1)$:

$$\mathcal{H}(1)=\sum_{\text{all } C(i,j,k)}\left[\left(\frac{I-Z_i}{2}\right)+\left(\frac{I-Z_j}{2}\right)+\left(\frac{I-Z_k}{2}\right)-1\right]^2$$

$$=N_c-\sum_i B_i z_i+2\sum_{i<j}C_{i,j}\,z_i z_j$$

- This is a binary quadratic form. So 3-SAT is a particular case of

Quadratic **U**nconstrained **B**inary **O**ptimization (QUBO)

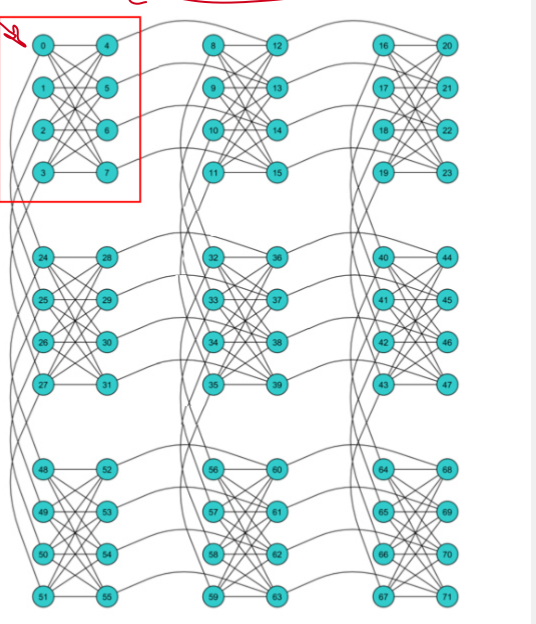
$$\mathcal{H}_{\text{QUBO}}(1)=-\sum_i b_i z_i+\sum_{i<j}c_{ij}\,z_i z_j$$

D-Wave Leap

- Visit cloud.dwavesys.com, sign up for an account, log in.
- A tour of the Dashboard. Note that you will get 1 minute of QPU for free. To get one additional minute per month all you have to do is to enter your GitHub username and repository link.
- There are many ways to get started. As a first reading, I suggest the "Getting Started with the System" chapter in the D-Wave System Documentation (This class is based on this document): https://docs.dwavesys.com/docs/latest/doc_getting_started.html
- You will find the Python and Jupyter notebooks for the class today here: https://github.com/rogdesousa/DWaveLeap

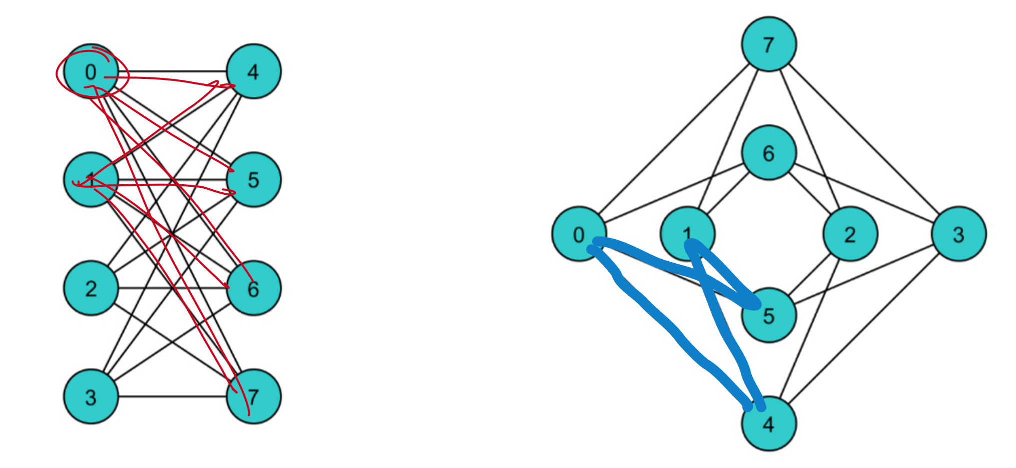
Implementing a simple QUBO with the Chimera graph

- The key issue of programming in the DW_2000Q_5 QPU is to handle the connectivity between qubits. The architecture is a Chimera graph.
- DW_2000Q_5 has 16X16 unit cells, each with 8 qubits. Total 16 x 16 x 8 = 2048 qubits but only 2030 are functional.



Unit cell

- Each unit cell can be rendered as either a column or a cross:



Implement simple QUBO with two qubits

- We can embed more general graphs into the Chimera graph by "chaining" two qubits together, so that they assume the same state.
- We start by doing this with two qubits, q0 and q4. We will force them to have as their lowest energy state either 00 or 11. The corresponding QUBO is

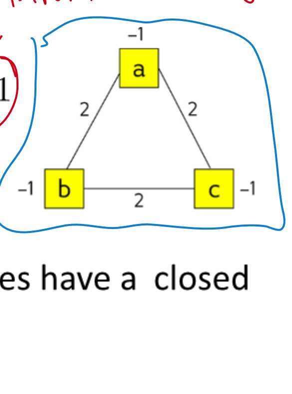
$$E(\boldsymbol{q})=(q_0-q_4)^2=q_0+q_4-2q_0q_4$$

- See implementation in 3SAT_Embedding_Example.ipynb

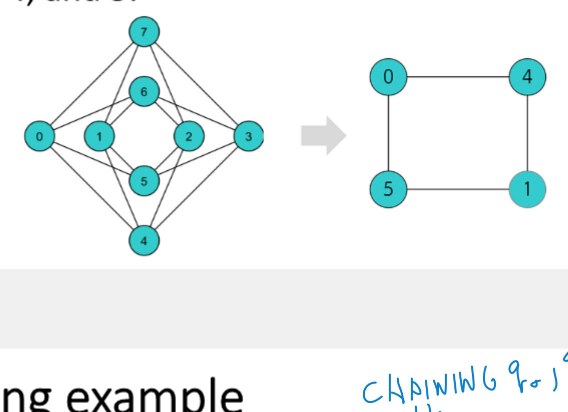
Minor-embedding: How to implement 3-qubit 3-SAT

- To implement the simplest 3-SAT we need the QUBO:

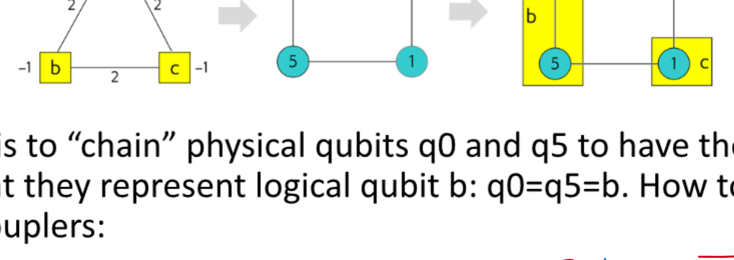
$$E(\boldsymbol{z})=(a+b+c-1)^2=-(a+b+c)+2(ab+ac+bc)+1$$



- Chimera does not have a closed 3-qubit loop. However, it does have a closed loop of 4 qubits: 0, 1, 4, and 5.



Minor-embedding example



- The strategy is to "chain" physical qubits q0 and q5 to have the same state (00 or 11). So that they represent logical qubit b: q0=q5=b. How to choose the biases and couplers:

Qubits	0	5	4	1
	b	b	a	c
	?	?	−1	−1
	−0.5	−0.5		
	1	1		

coupler strength	(0,4)	(5,0)	(4,1)	(1,5)
	2	?	2	2
		−3		

$$E(\boldsymbol{q}) = (q_0+q_5)-(q_4+q_1)+2(q_0q_4+q_4q_1+q_1q_5)-3q_0q_5 \quad \div 3$$

$$= \frac{1}{3}(q_0+q_5)-\frac{1}{3}(q_4+q_1)+\frac{2}{3}(q_0q_4+q_4q_1+q_1q_5)-q_0q_5$$

DWave's tool for embedding: *EmbeddingComposite*

- 5-qubit 3-SAT from previous class : $z_0z_1z_2z_3z_4$ — $C(0,1,2)$, $C(1,2,3)$, $C(0,3,4)$

$$E(\boldsymbol{z})=-\sum_i B_i z_i+2\sum_{i<j}C_{ij}z_iz_j$$

$$=-2(z_0+z_1+z_2+z_3)-z_4$$
$$+2(z_0z_1+z_0z_2+z_1z_3+z_2z_3+z_0z_3+z_0z_4+z_3z_4)+4z_1z_2$$

- See implementation in 3SAT_Embedding_Example.ipynb