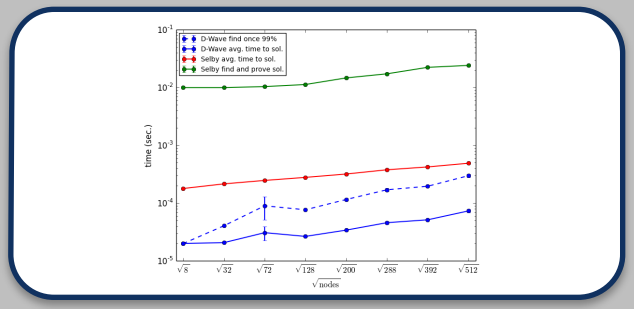
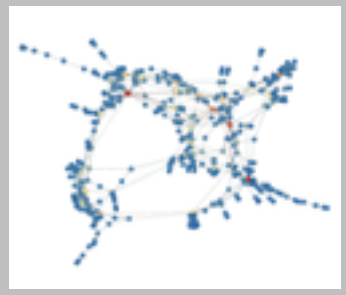
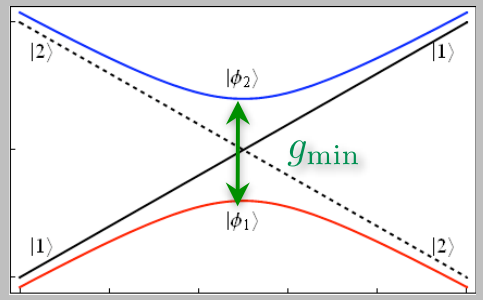
Exceptional service in the national interest

Sandia National Laboratories







Unearthing planted solutions in quantum-annealing-inspired Ising problems

Ojas Parekh

Robert Carr

Sandia National Labs

U.S. DEPARTMENT OF ENERGY NNSA

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

A question?

Sandia National Laboratories

Which is harder (pick a reasonable definition of “hard”)?

Finding an optimal solution 99% of the time

Finding a solution within 99% of optimal all the time

Which is more practically relevant?

A question?

Sandia National Laboratories

Which is harder (pick a reasonable definition of “hard”)?

Finding an optimal solution 99% of the time

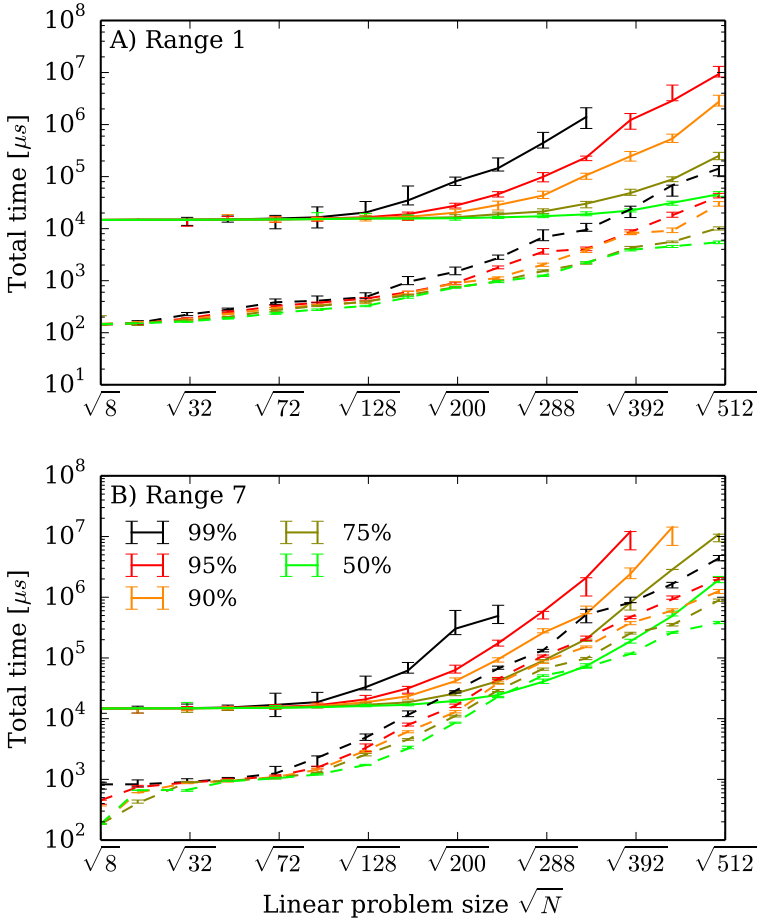
“Average”-case analysis, rather than worst-case

Finding a solution within 99% of optimal all the time

Approximation algorithm, or approximation scheme

How should we measure success?

Optimal 99% of the time



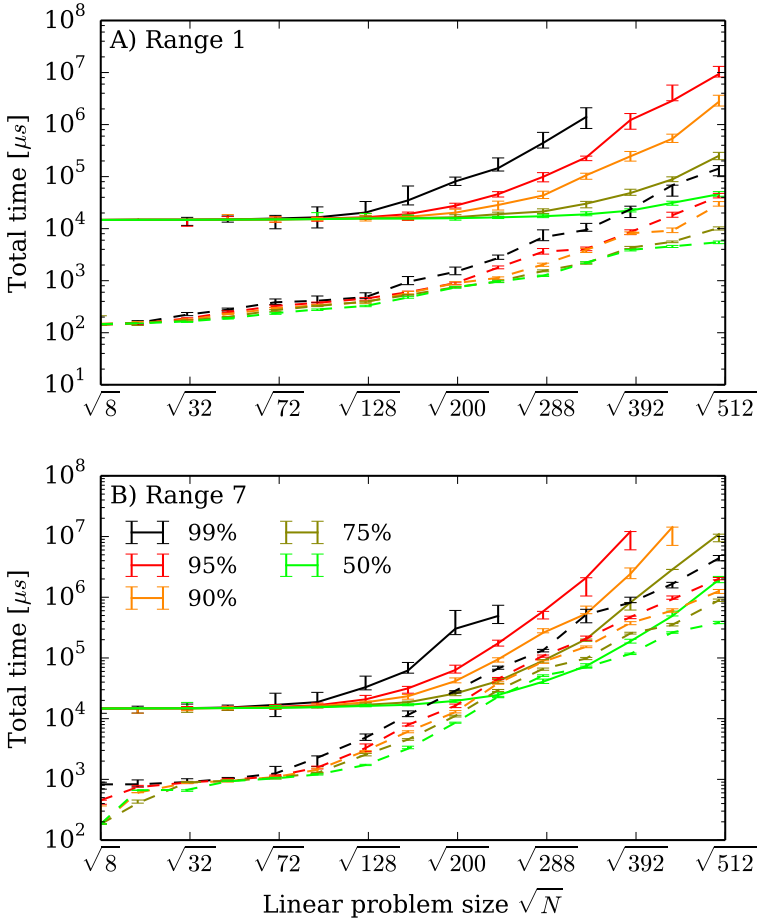
Instances: D-Wave 2 Chimera graph with randomly chosen coupler weights

"Exponential" scaling as $c^{\sqrt{n}}$ for both D-Wave 2 and simulated annealing

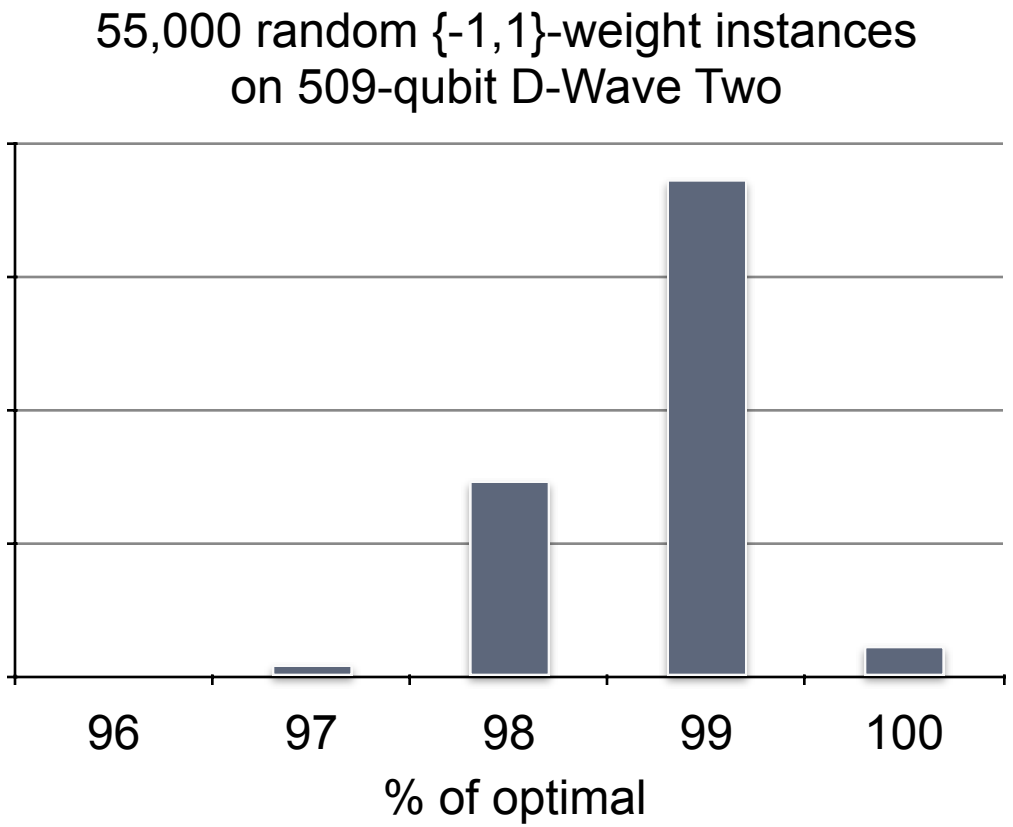
Left: Ronnow *et al.*, *Defining and detecting quantum speedup*, arXiv:1401.2910v1 (2014)

How should we measure success?

Optimal 99% of the time



Within 99% of optimal all of the time



Was always within 96% of optimal!

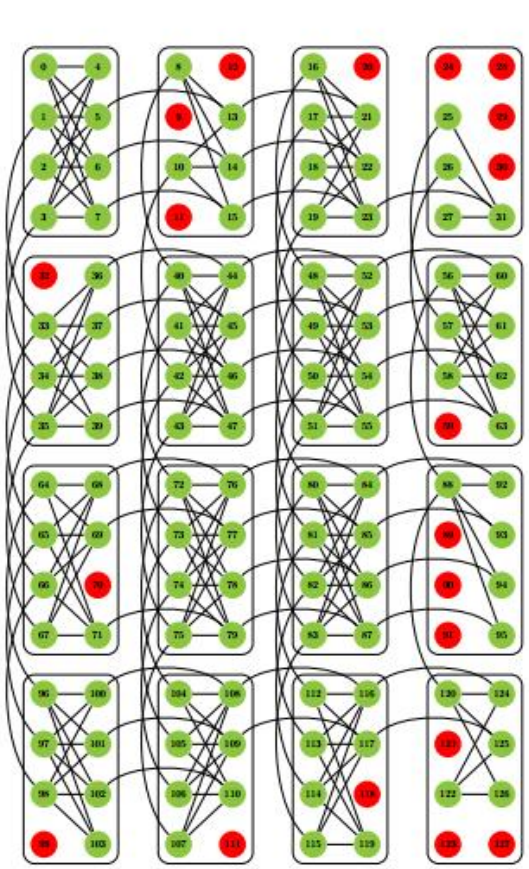
Left: Ronnow *et al.*, *Defining and detecting quantum speedup*, arXiv:1401.2910v1 (2014)

We ask

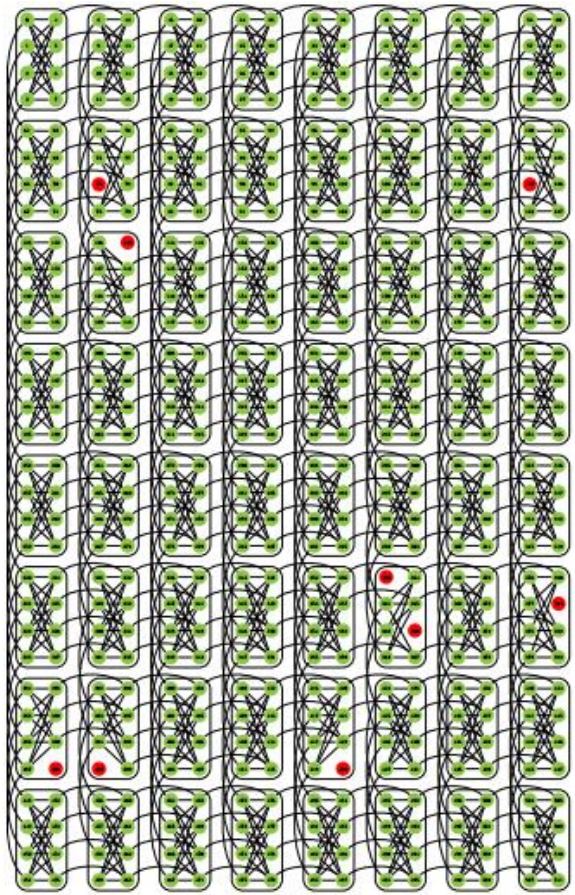
- What is an appropriate measure of success?
- What classical algorithm(s) should be used for comparison?
- How should one select appropriate benchmark instances?

Comparison of Rainier and Vesuvius chips

Rainier
108/128
spins



D-Wave One

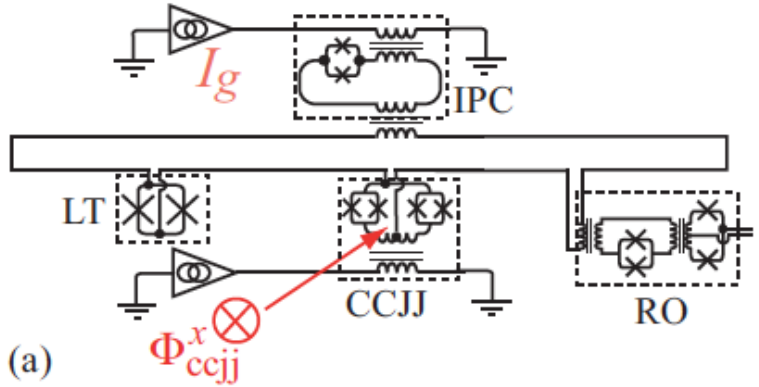


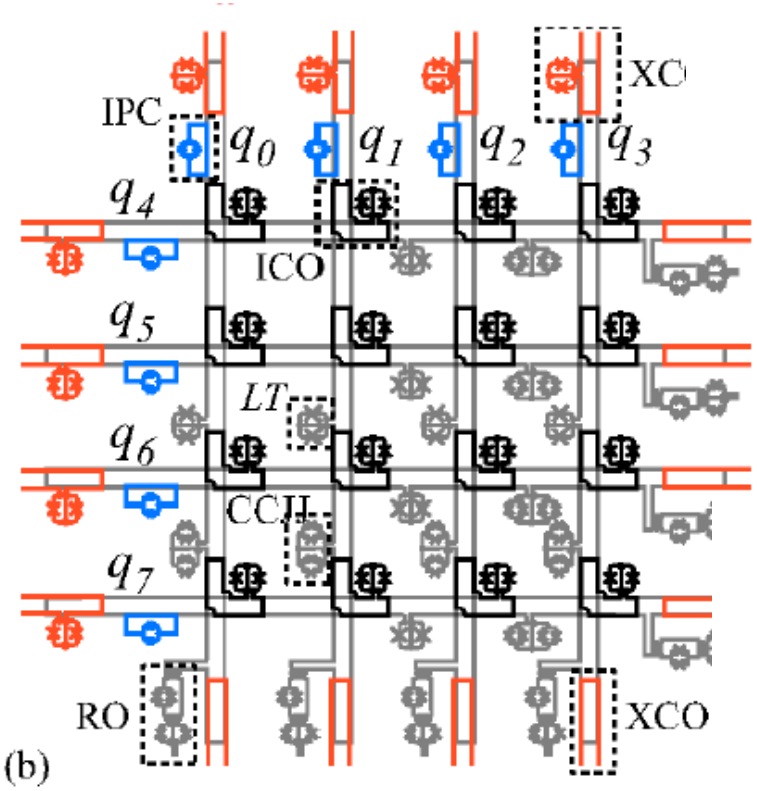
D-Wave Two

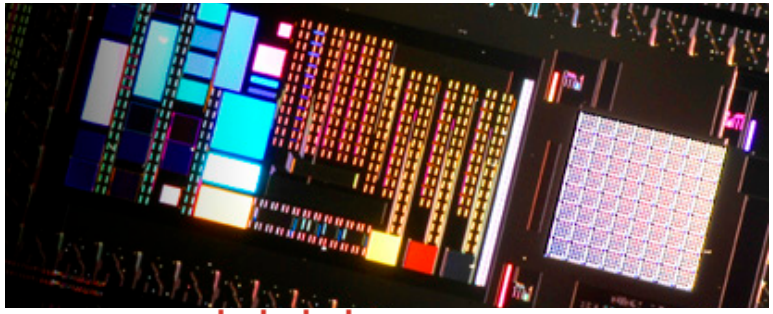
Vesuvius
506/512
spins

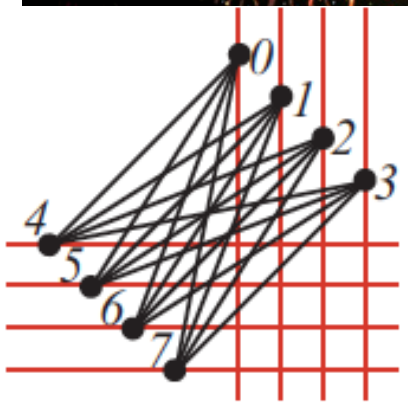
Images from D-Wave Systems: http://www.dwavesys.com .

Eight qubit cell architecture









$$\begin{pmatrix} 0 & 0 & 0 & 0 & J_{0,4} & J_{0,5} & J_{0,6} & J_{0,7} \\ 0 & 0 & 0 & 0 & J_{1,4} & J_{1,5} & J_{1,6} & J_{1,7} \\ 0 & 0 & 0 & 0 & J_{2,4} & J_{2,5} & J_{2,6} & J_{2,7} \\ 0 & 0 & 0 & 0 & J_{3,4} & J_{3,5} & J_{3,6} & J_{3,7} \\ J_{4,0} & J_{4,1} & J_{4,2} & J_{4,3} & 0 & 0 & 0 & 0 \\ J_{5,0} & J_{5,1} & J_{5,2} & J_{5,3} & 0 & 0 & 0 & 0 \\ J_{6,0} & J_{6,1} & J_{6,2} & J_{6,3} & 0 & 0 & 0 & 0 \\ J_{7,0} & J_{7,1} & J_{7,2} & J_{7,3} & 0 & 0 & 0 & 0 \end{pmatrix}$$

R. Harris *et al.*, Phys. Rev. B 82, 024511 (2010)

QUBO
(Quadratic Unconstrained Binary Optimization)

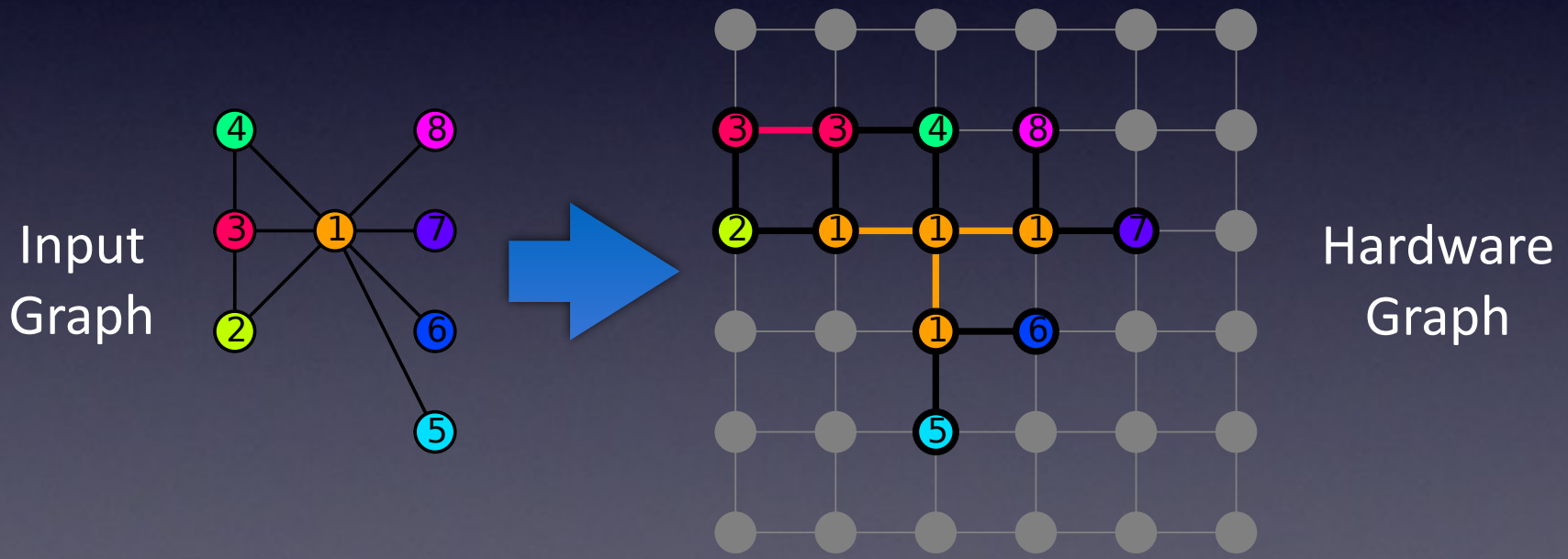
$$\text{QUBO: } f(x) = \min_{x \in \{0,1\}^n} b^T x + x^T C x$$

- Well-suited for discrete optimization applications
- Variables correspond to qubits
  More qubits = richer problem modeling
- Matrix C corresponds to a graph
- AQC architectural constraints and hardware dictate edges and weight ranges

Sandia National Laboratories

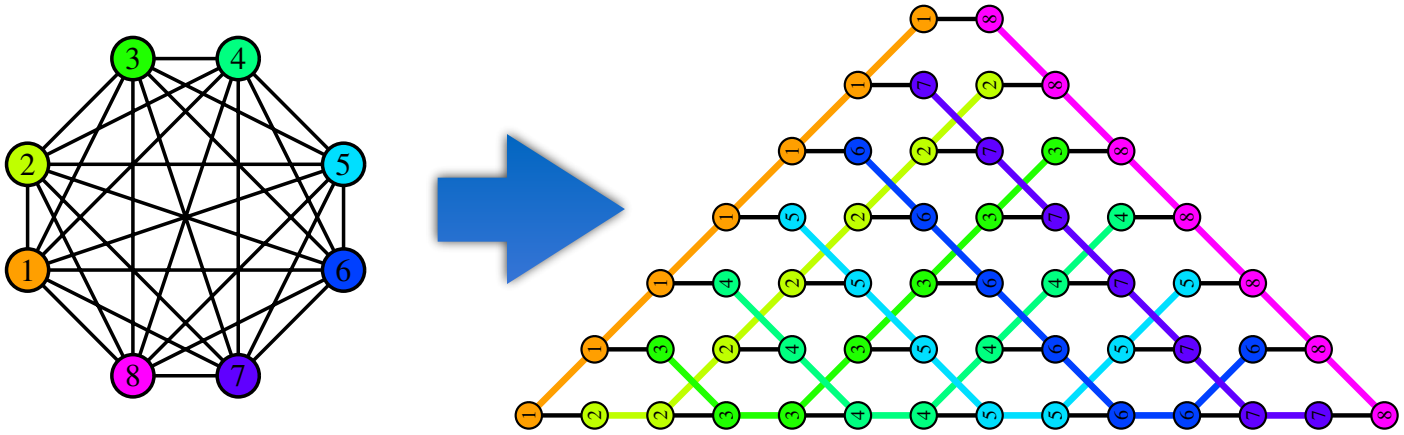
Problem embedding

Goal: preserve $f(x) = \min_{x \in \{0,1\}^n} b^T x + x^T C x$



Approaches to problem embedding

- Embedding is NP-hard, also space is $O(n^n)$ vs $O(2^n)$
- Even harder when optimizing # qubits
- Choi: worst case $O(n^2)$ qubits for *n* vars
  - Requires (linearly) large coupler weights



Complexity of QUBO on Chimera

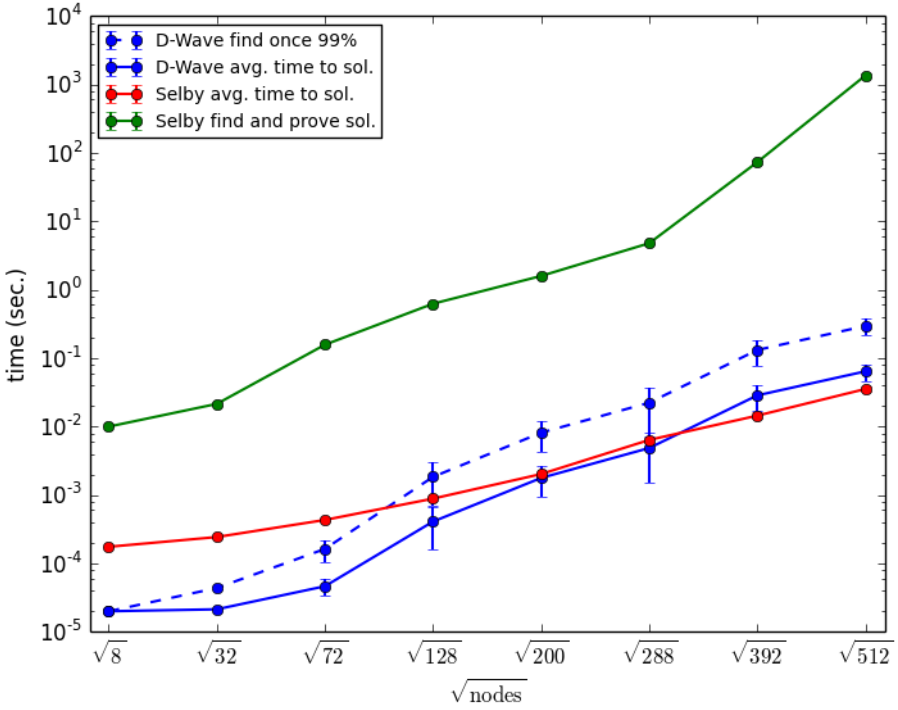
Sandia National Laboratories

- (Decision version) NP-complete even with no linear term and {-1, 0, 1} weights [Barahona, 1982]
- We observe also NP-complete with {-1,1} weights
  - Instances used in D-Wave benchmarking studies
- Tree-width (path-width) is $\Theta(\sqrt{n})$, yielding $O(2^{\sqrt{n}})$ algorithm
  - "Subexponential" exact algorithm even though NP-hard
- Approximation complexity?
  - Polynomial-time approximation scheme (PTAS) [Saket, 2013, arXiv:1306.6943]
  - PTAS's are rarely efficient; theory vs practice?
  - Efficient approx algorithm for say, getting within 90%?

Limits of reducing to Chimera

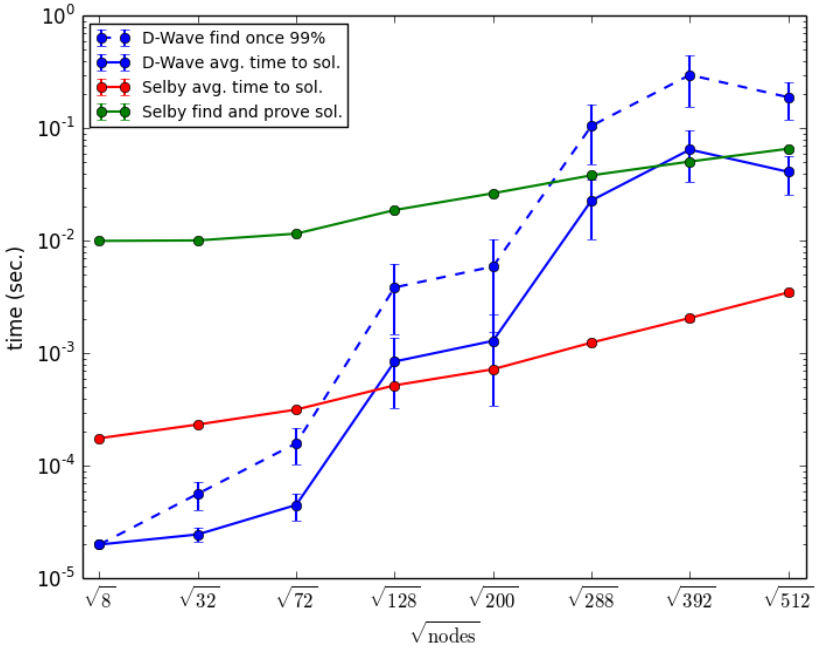
- Can we do better than a quadratic blowup in qubits?
- Probably not, due to the Exponential Time Hypothesis
  - Exact algorithms for problems like Max-Cut on general graphs are conjectured to require $O(2^n)$ time
  - But we have a $O(2^{\sqrt{n}})$ time algorithm for Chimera Ising
  - So in some sense quadratic factor is artifact of Chimera
- Weights make this worse: Choi embedding assumes (linearly) large weights
- Reduction better than $O(n^2)$ for Max-Cut on bounded-degree graphs would improve best-known classical algorithm
  - Applies to **any** reduction, not just minor embeddings

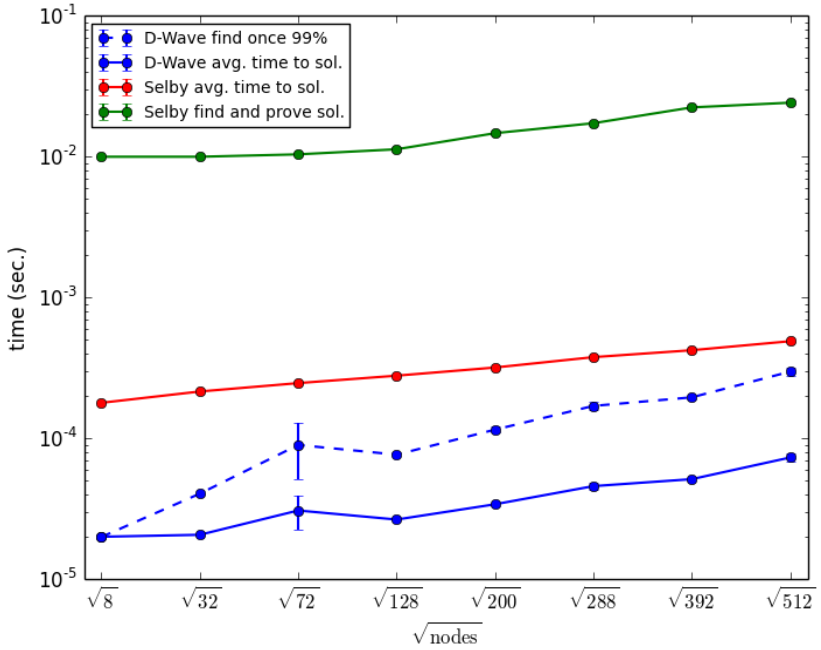
Example: hard problem

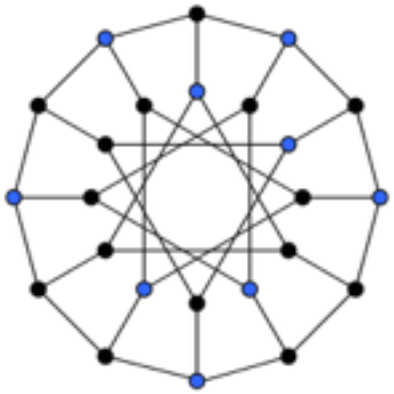


- Random {-1,1} Chimera
- Empirically scales as $c^{\sqrt{n}}$?
- NP-hard (worst-case)

Example: “easy” problem

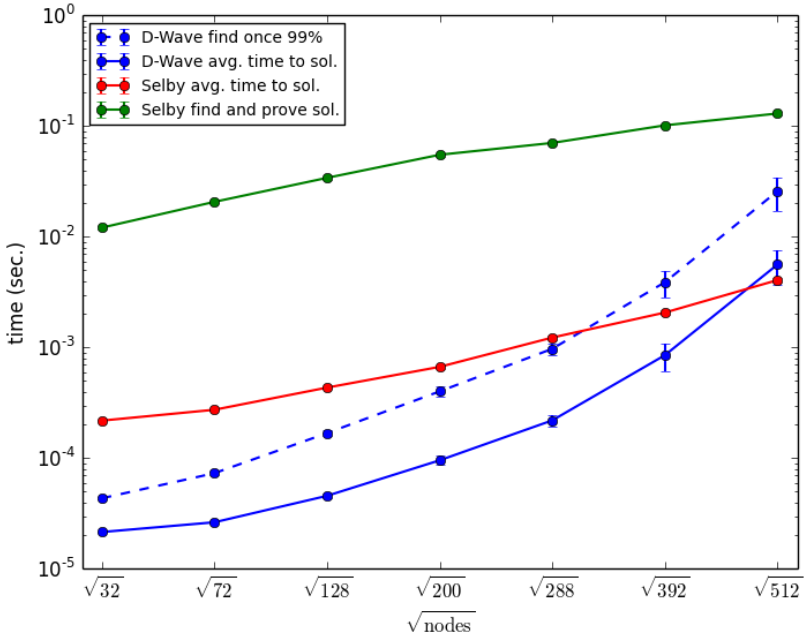


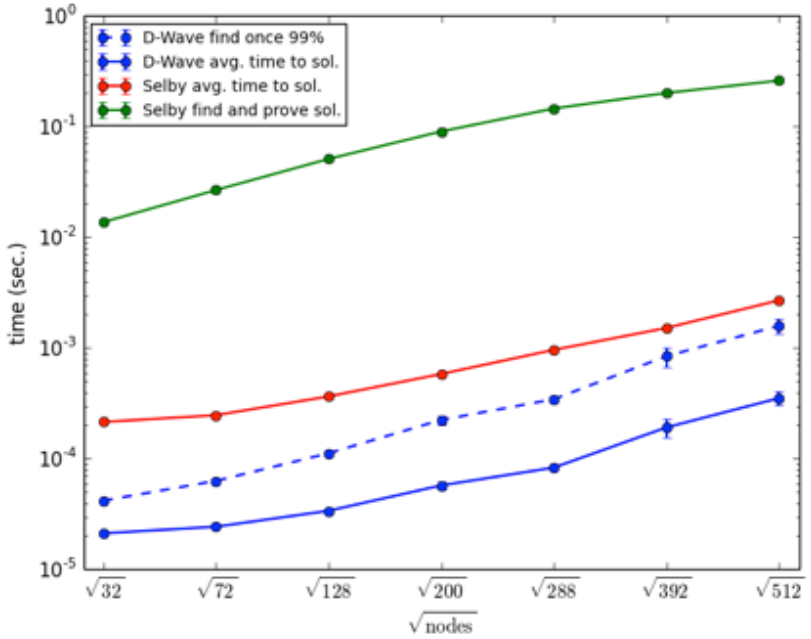




- Independent set problems
- Random {0,1} (Left) or {-1,1} (Right) on Chimera
- Linear term for i = (weighted degree of i) - 2
- Chimera is bipartite, so at least {0,1} case is in P!

Example: open problem



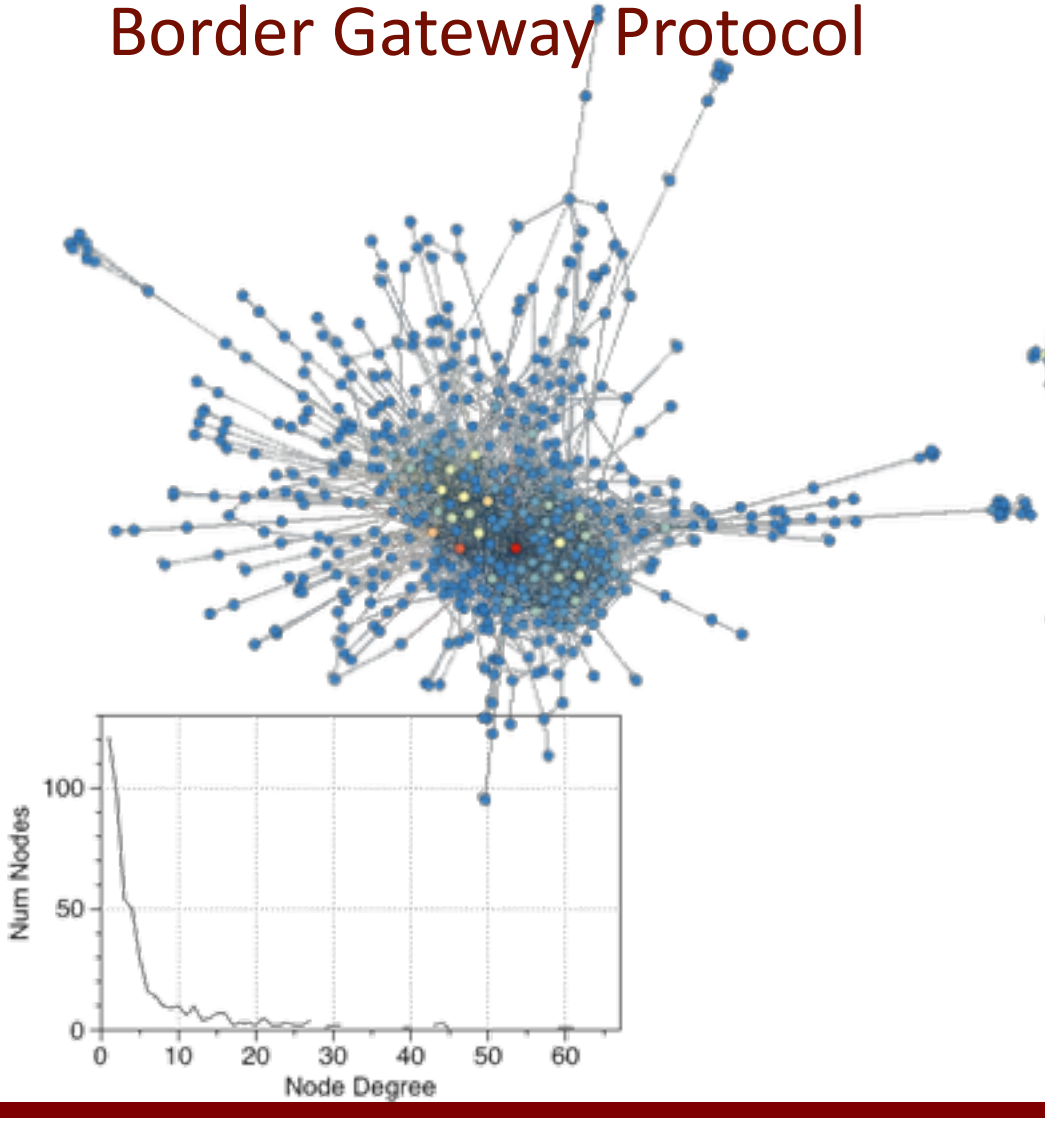


- Recent planted solution instances of Hen et al.
- One of few known instances where D-Wave performs best
- We give poly-time algorithm for optimal solution *value*

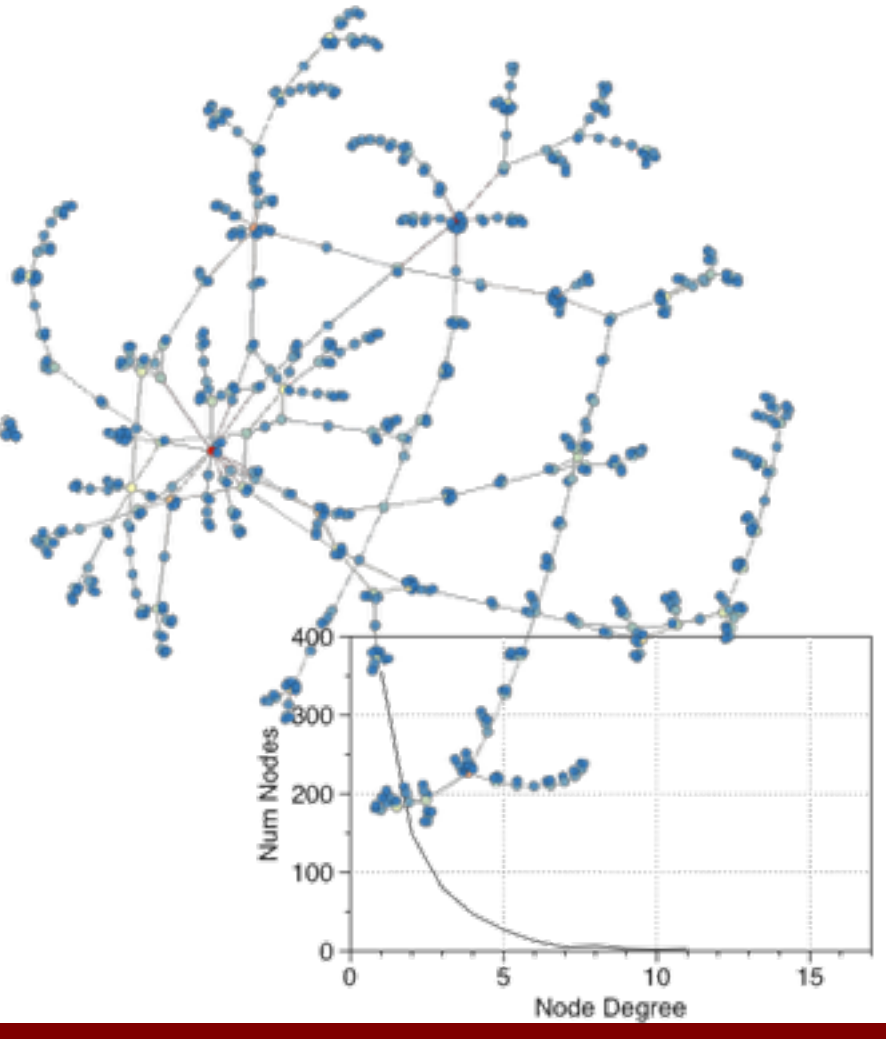
Digression: real-world complex networks

Sandia National Laboratories

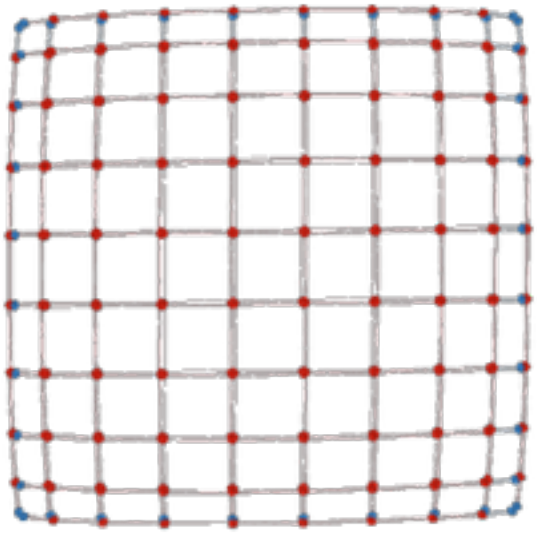
Border Gateway Protocol

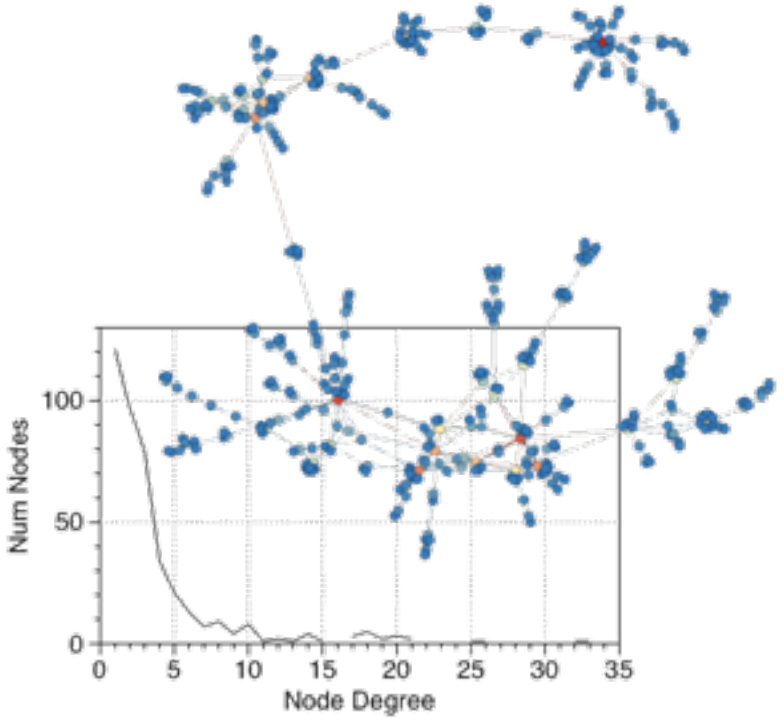


Twitter



Synthetic complex networks on a Chimera





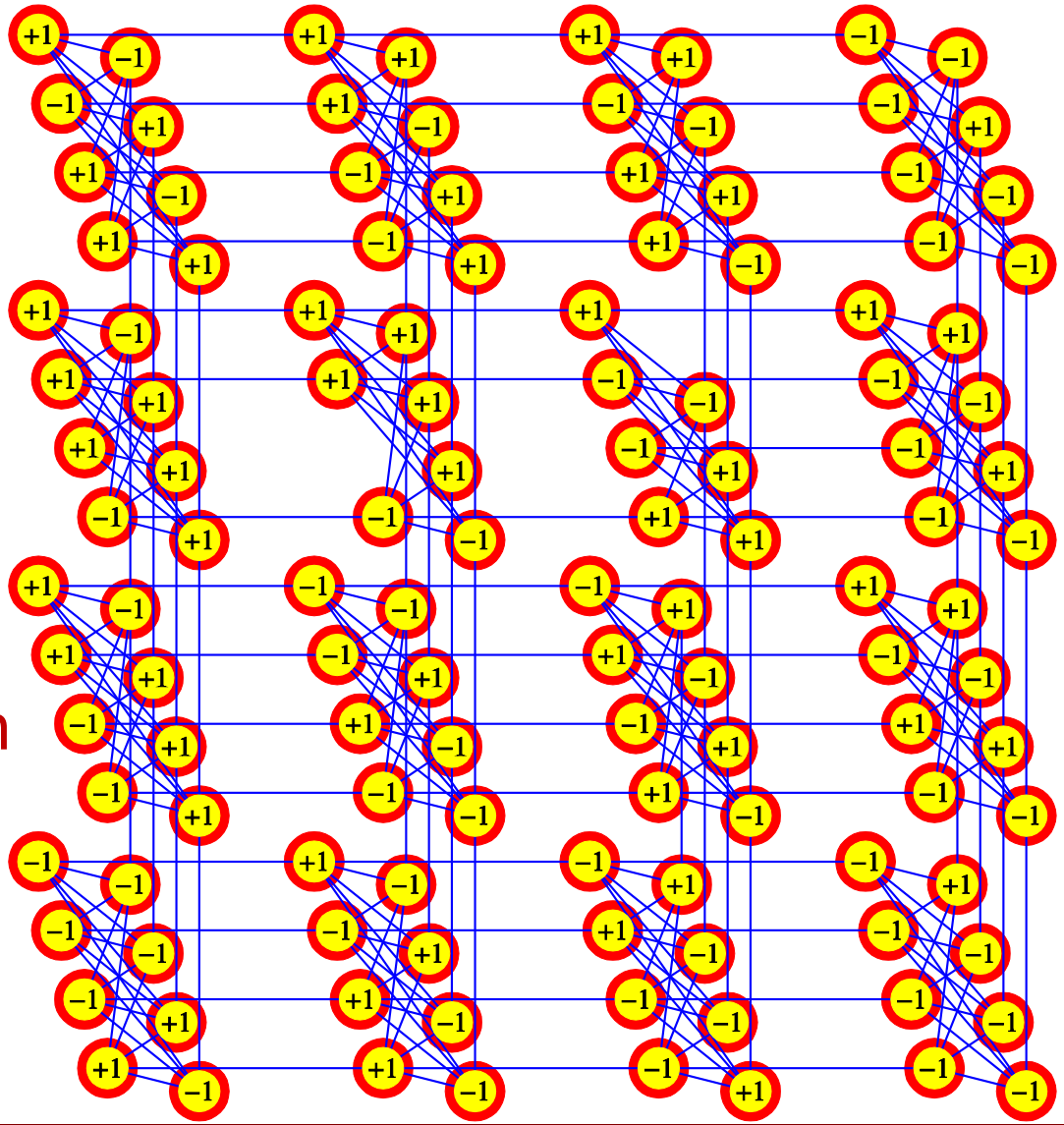
- (Minor-)Embedding arbitrary graph within Chimera is hard
- Instead, cheat and alter Chimera graph to produce minor with “real-world” properties
- Our algorithm tends to give minors with ~50% logical nodes of underlying physical Chimera

D-Wave-specific problems

- to construct a problem with a planted solution on the D-Wave chip, of the form:

$$H = \sum_{\langle ij \rangle} J_{ij} S_i S_j + \sum_i h_i S_i$$

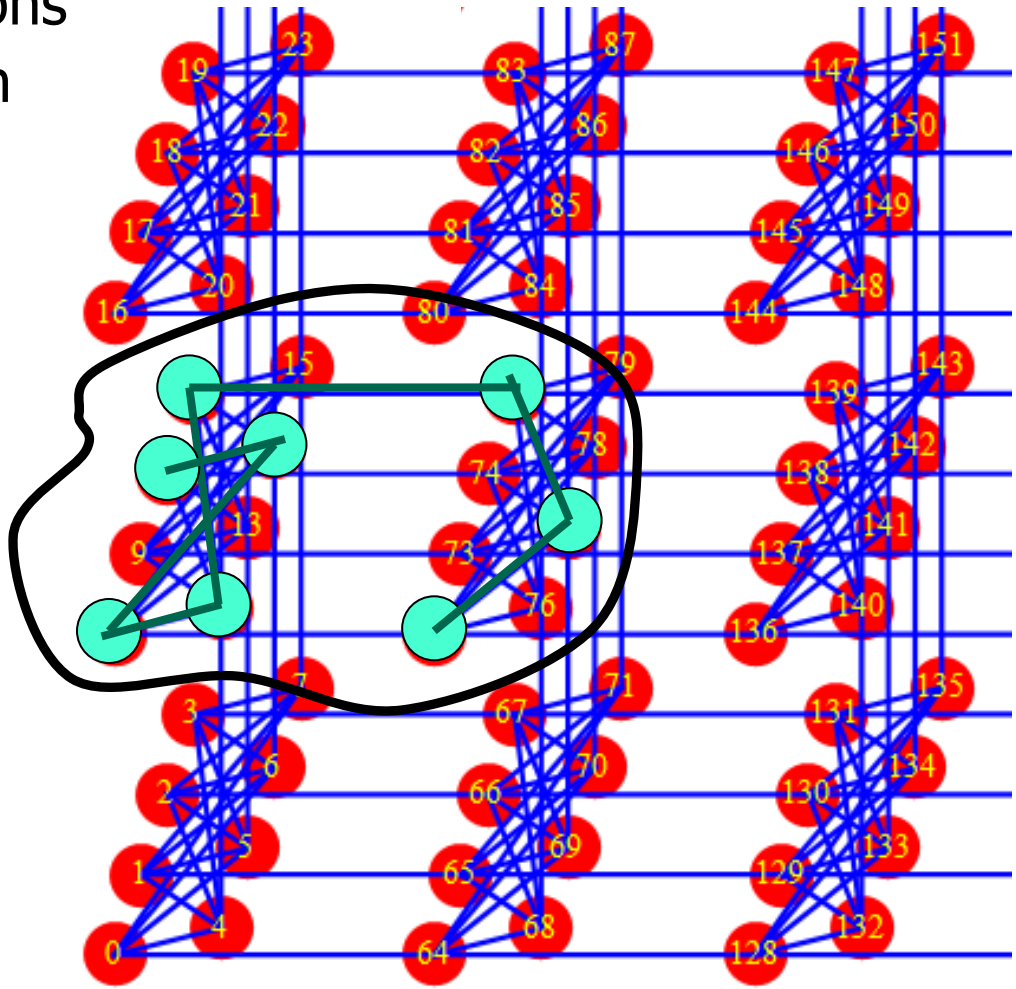
- we first generate a random bit configuration on the Chimera graph.
- this configuration will be our planted solution.



D-Wave-specific problems

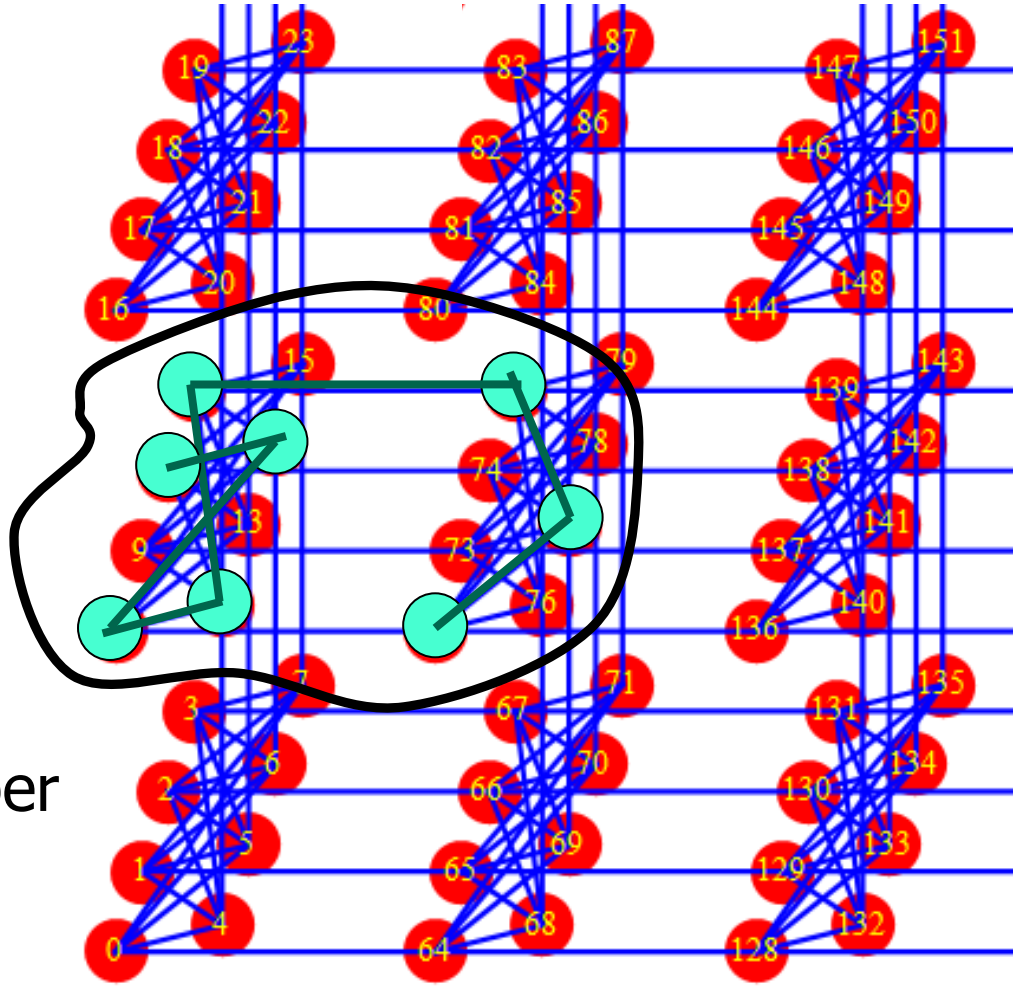
- as a next step, we generate "clauses", or, local cost functions, H_i , defined on small sub-graphs (whose size will not scale with problem size).
- clauses will be constructed in such a way that they are minimized by the planted solution.
- after constructing M such clauses, these will be added up to form one big cost function:

$$H = \sum_{i=1}^{M} H_i$$



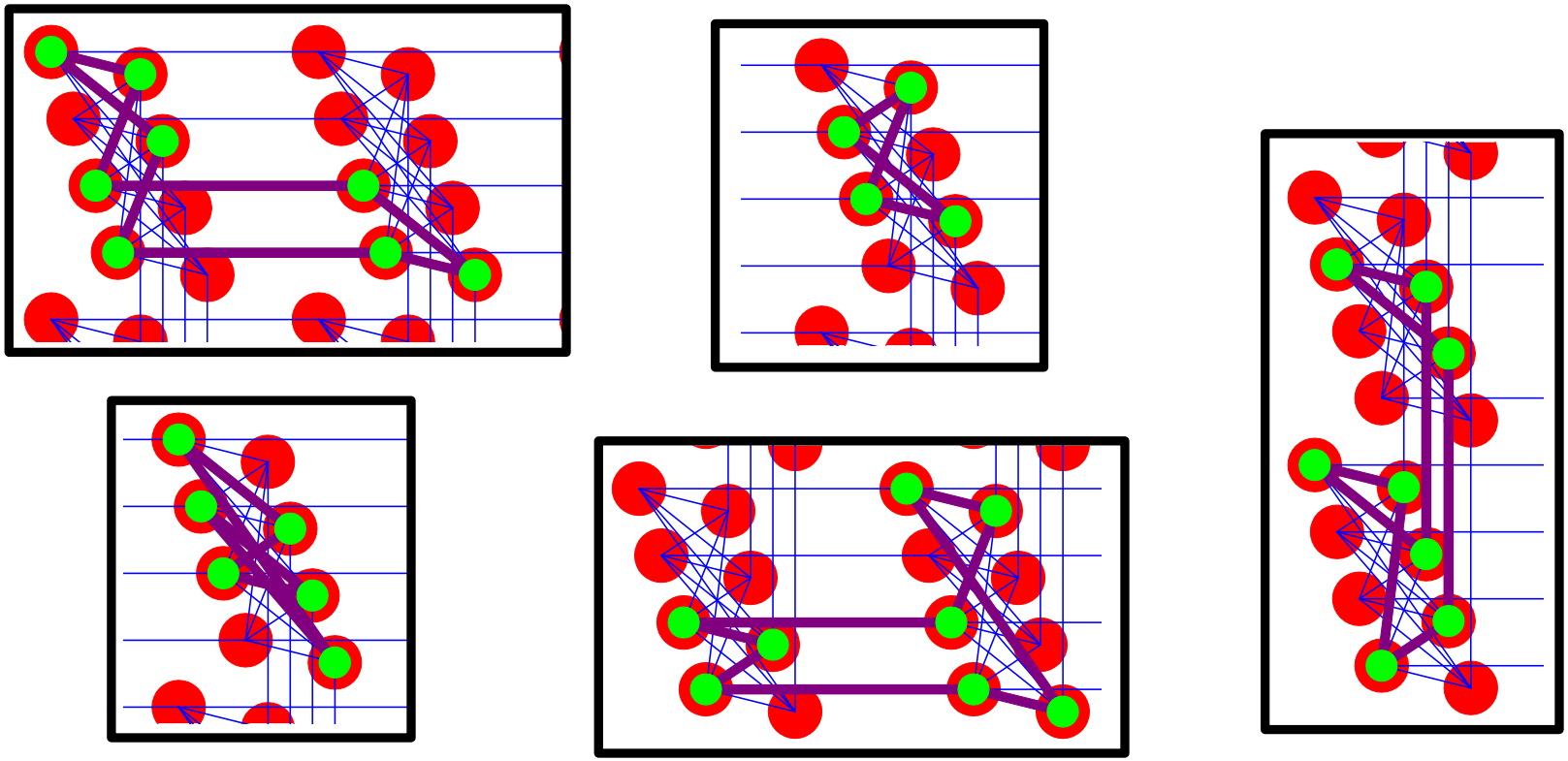
D-Wave-specific problems

- clauses will generally overlap.
- final Hamiltonian is "frustration free", i.e., the planted solution is a global solution of each clause, and of the total Hamiltonian as well.
- global solution will not necessarily be the only solution (depends on number of clauses).



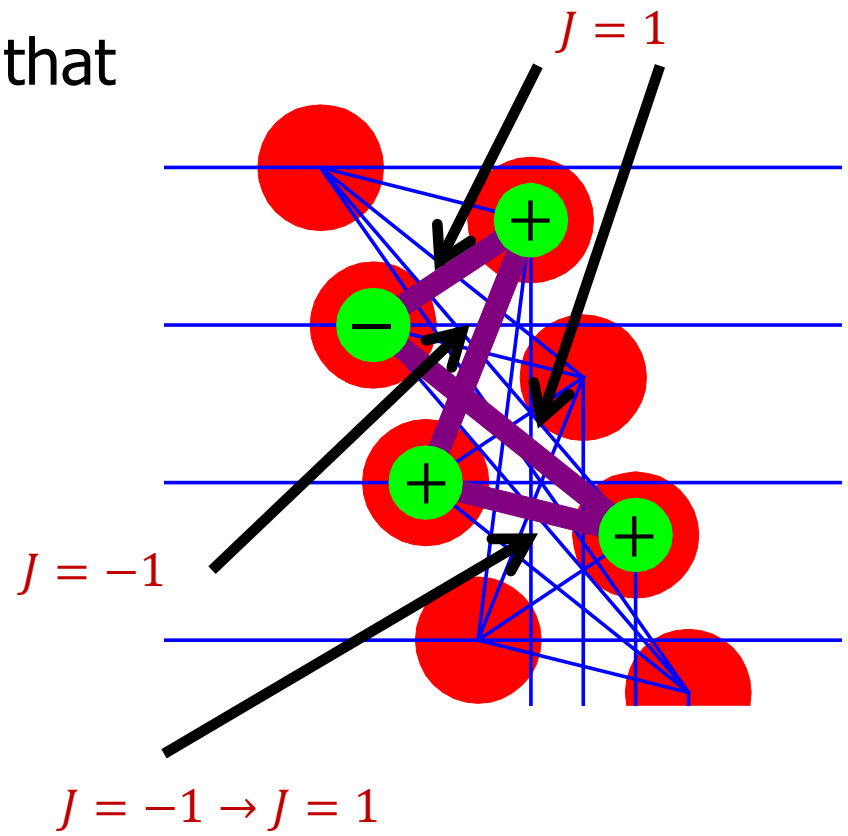
Frustrated loops instances

- "frustrated loops" are generated by random walks on the chimera with a randomly selected qubit as the point of origin.
- when random walker reaches a qubit that has already been visited, a loop is formed (tail is thrown away). examples:



Frustrated loops instances

- next, we generate Ising problems on the loops. how can we generate a frustrated problem?
- we assign couplings $J \in \{-1, +1\}$ that respect the planted solution.
- to achieve frustration, we flip one of the J's at random.
- this way, the planted solution will remain a ground state of the loop, but it will be a frustrated ground state.
- planted solution will be one of several configurations minimizing the loop.

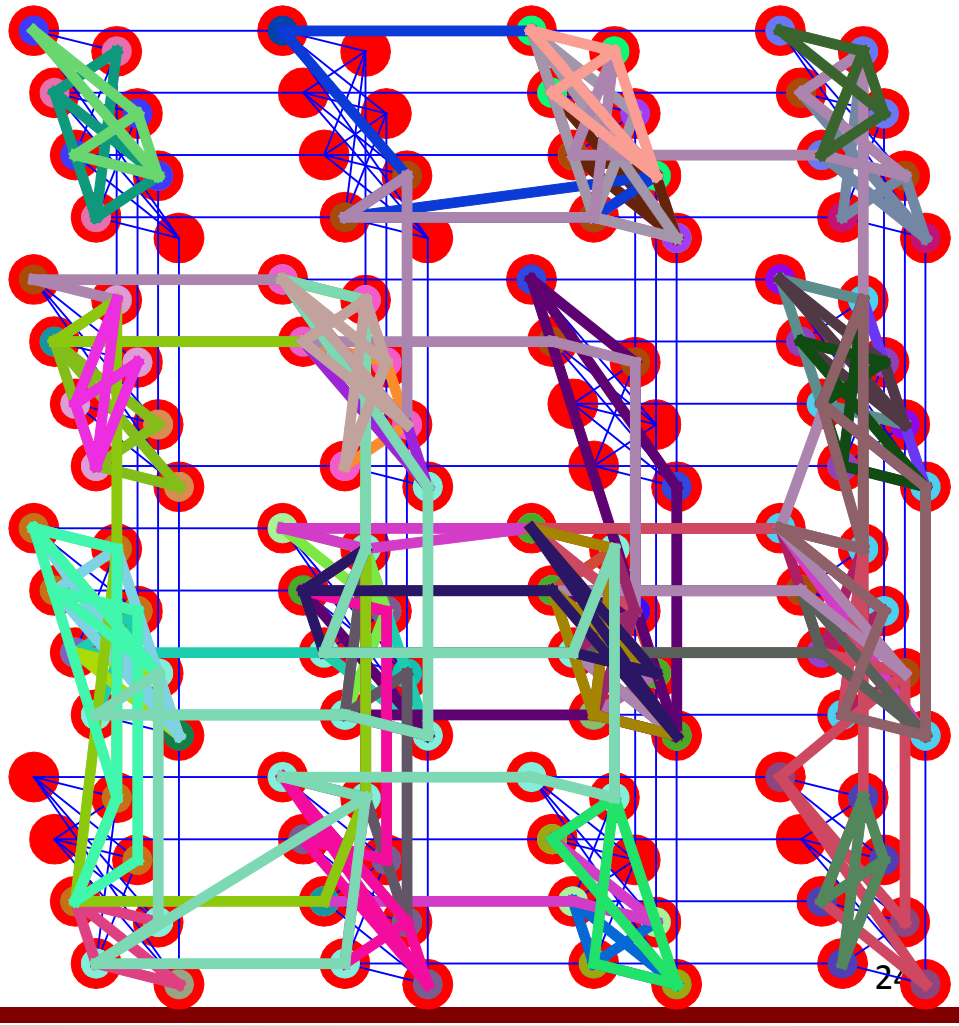


Frustrated loops instances

- after constructing $M = N_{loops}$ such loop Hamiltonians H_i , we add them all up to form one big cost function:

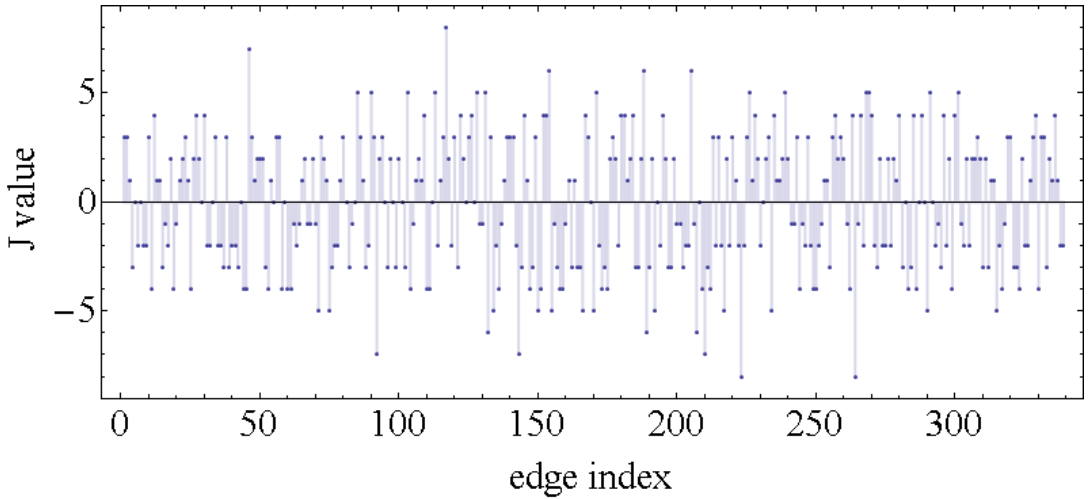
$$H = \sum_{i=1}^{N_{loops}} H_i$$

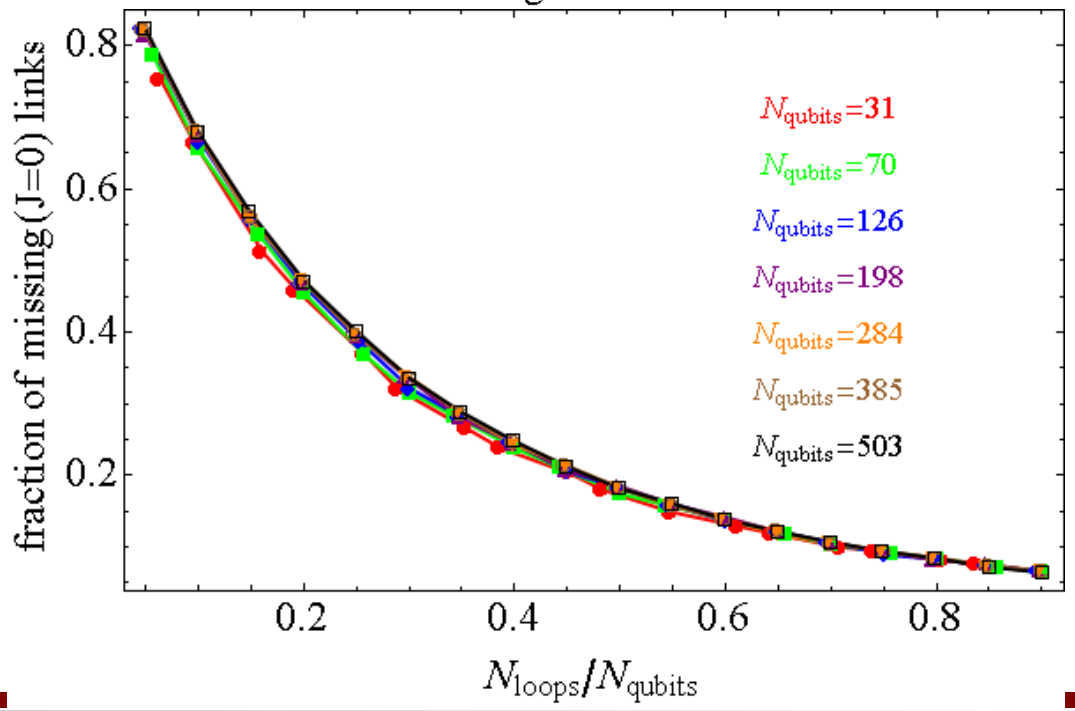
- loops will generally overlap.
- final J's should fall within precision bounds (this will not be strictly enforced).



Instance properties

- distribution of J values for $N = 126$ qubit Chimera with $M = 101$ loops.
- looking at the final J's, it would be virtually impossible to reconstruct the loops.
- fraction of $J = 0$ couplings as a function of loop-to-qubit ratio:





A linear-programming approach

Sandia National Laboratories

- We can obtain a linear program that always produces the optimal solution *value* as well as a generating set of cycles (using metric polytope relaxation [Barahona 83, Barahona Mahjoub 86])
- Exponential number of constraints, some for each cycle; however, ellipsoid gives us poly-time solution
- Fractional extreme point solutions exist, thus we cannot always hope to directly find an optimal solution
- Above renders self-reduction approach difficult

Conclusion

- Partial resolution of status of Hen et al. planted solutions with polynomial-time algorithm to find optimal solution *value*
- Understanding complexity of benchmarking instances is critical
- Need for math-programming approaches in D-Wave benchmarking