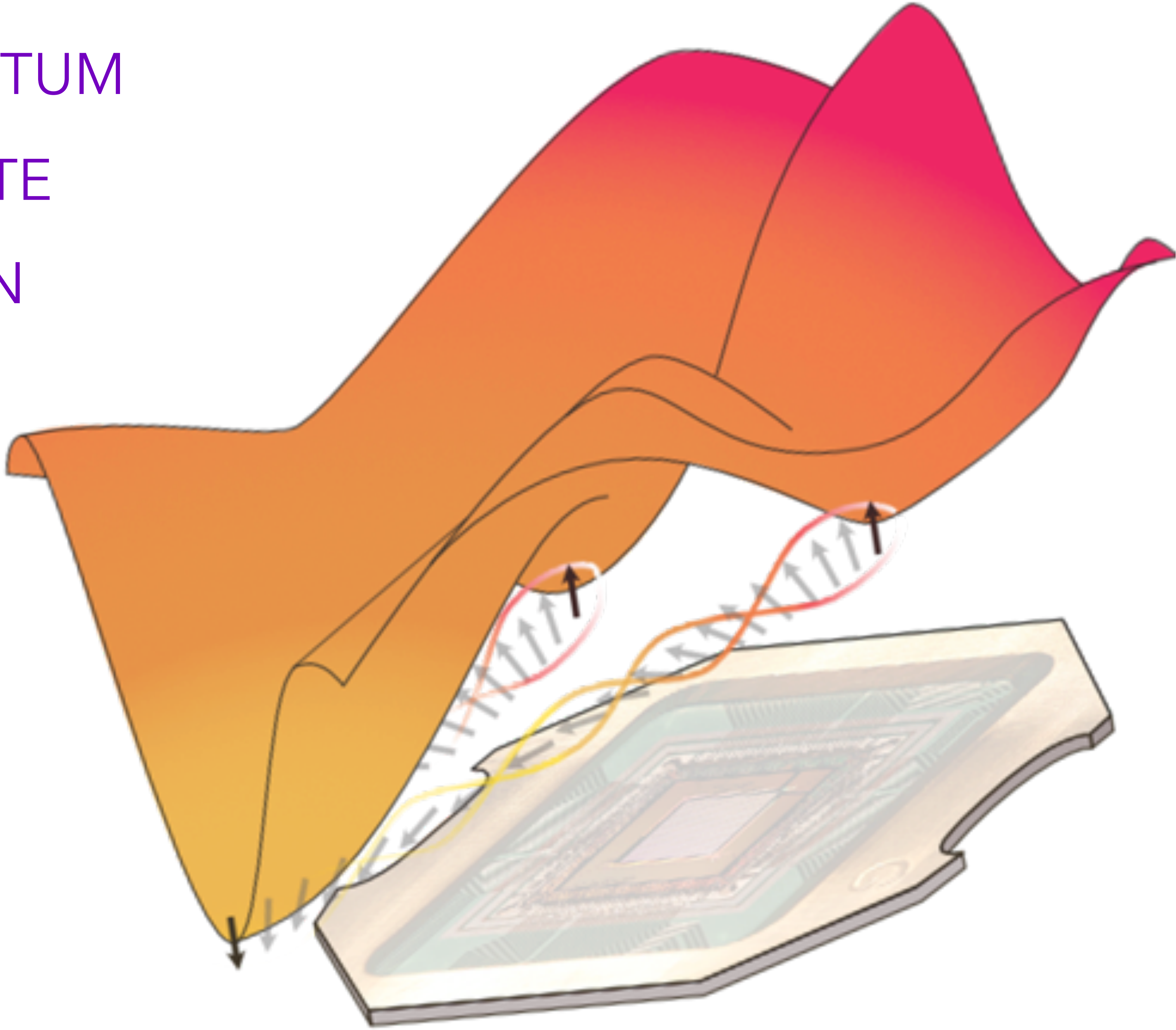
HANDS-ON D-WAVE: QUANTUM ANNEALING TO GENERATE STRUCTURAL MODELS IN MATERIALS

Ilaria Siloi
University of Southern California



WHAT?

Hardware

Connectivity

WHY?

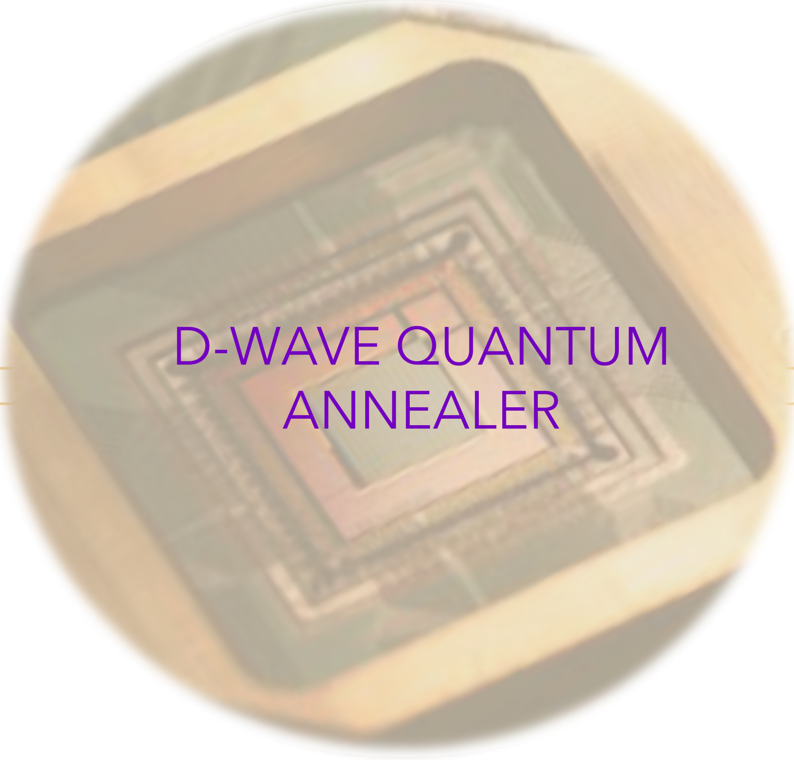
HOW?

Input problems

Algorithms

Output samples

Tools



D-WAVE QUANTUM ANNEALER

HANDS-ON

Routing problem

Graph Coloring problem

MATERIALS SCIENCE

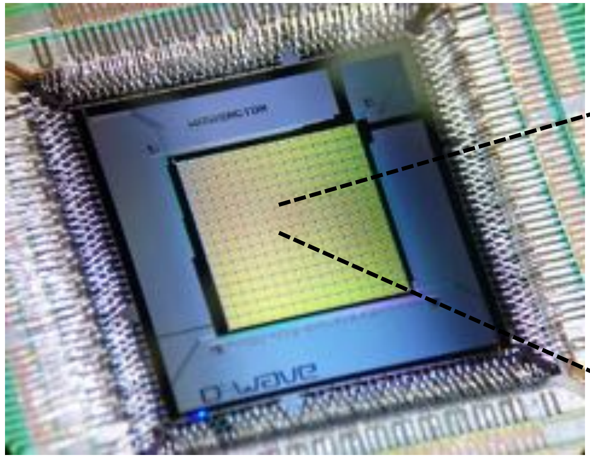
From combinatorics to structural model generation

WHAT?

D-Wave QPU

Connectivity

D-Wave 2000Q QPU



T<9.2 K superconducting loops

2048 flux qubits - 6016 couplers

128k Josephson junctions

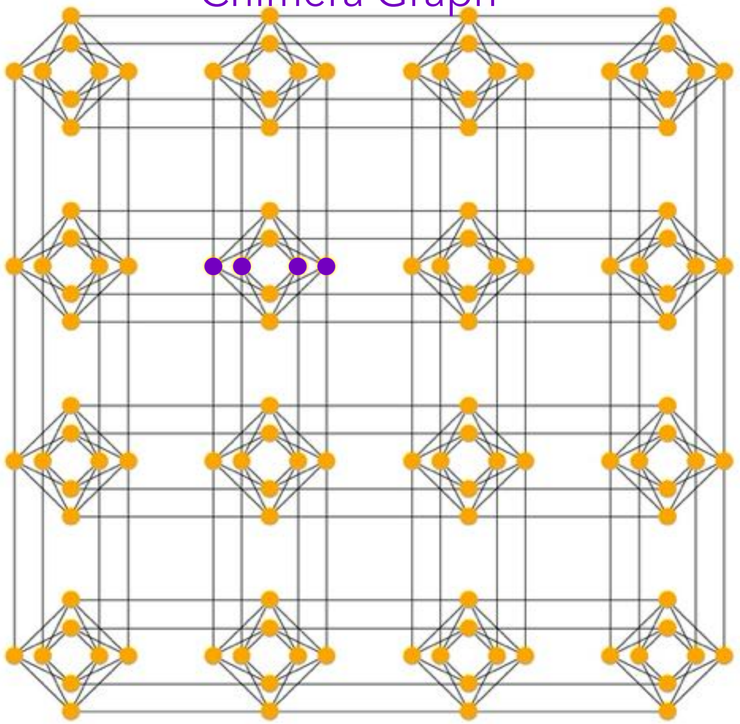
QPU operates @ 15mK

Unit are a bipartite graphs;

Qubits have 4 connections within the unit

Qubits have 2 connections within neighboring units

Chimera Graph



Applications

COMBINATORIAL OPTIMIZATION

Lowest energy solution

SAMPLING PROBLEMS

Low-energy samples (Machine Learning)

HOW?

Optimization problem as Ising/QUBO

INPUT

$H_{tar}(s_1, s_2, \dots, s_N) = \sum_i h_i s_i + \sum_{ij} J_{ij}\, s_i\, s_j$ with $s_i \in [-1,1]$

Ising Hamiltonian

$Q_{tar}(x_1, x_2, \dots, x_N) = \sum_i x_i Q_{ii} x_i + \sum_{ij} x_i Q_{ij}\, x_j$ with $x_i \in [0,1]$

Quadratic Unconstrained Binary Optimization

$$J_{ij} = \frac{Q_{ij}}{4};\; h_i = \frac{1}{2} Q_{ii} + \frac{1}{4} \sum_{i<j} Q_{ij}$$

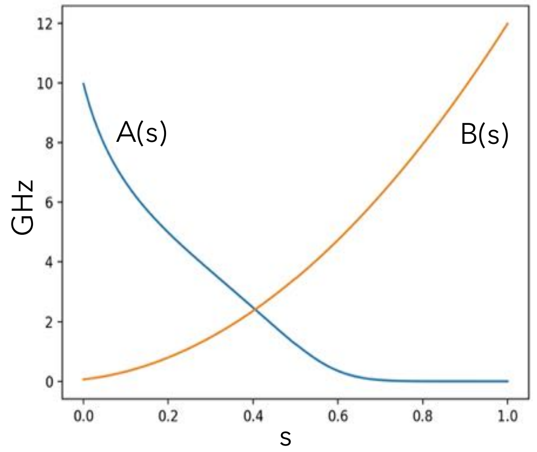
Algorithm/ Heuristic

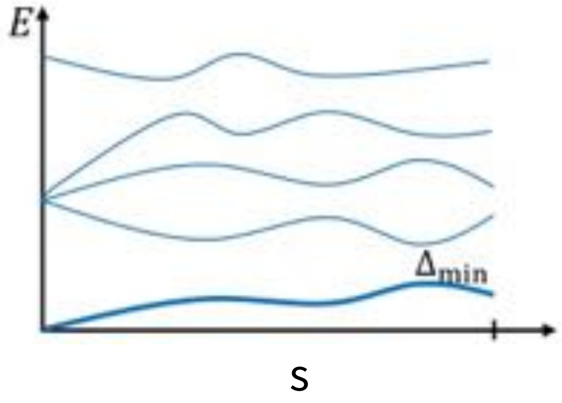
QUANTUM ANNEALING

$$H_{tot}(s) = A(s) H_0 + \mathrm{B}(s) H_{tar}$$

$$H_0 = \sum_{j \in V} h_j \sigma_j^x$$ initialization

$$H_{tar} = \sum_{(i,j)\in E} J_{ij} \sigma_i^z\, \sigma_j^z + \sum_{j \in V} h_j \sigma_j^z$$





Samples of bit strings

MEASUREMENT

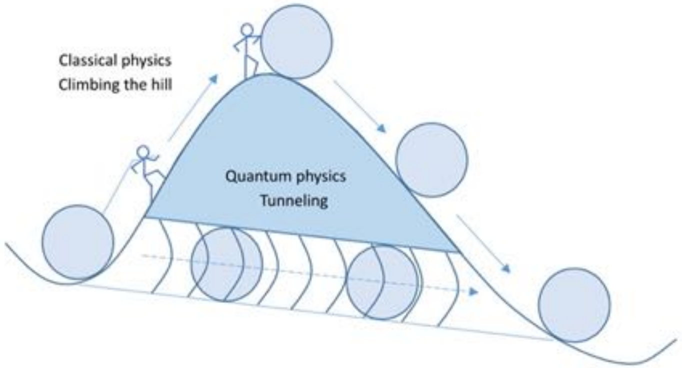
$\bigotimes_i \sigma_z^i$

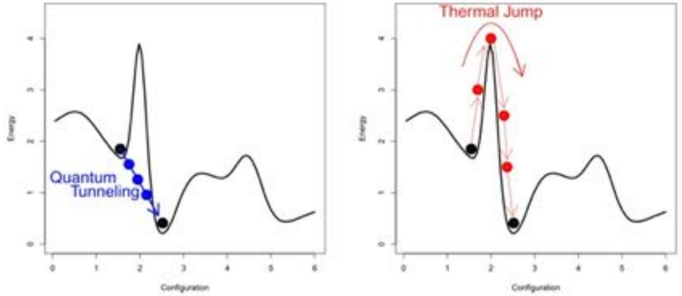
Outcome distribution

[1,0,1, … , 0,0,1]
[1,0,1, … , 0,0,0]
[1,0,1, … , 1,0,1]
…

WHY?

Thermal
VS
Quantum
fluctuations





Quantum Annealing of a Disordered Magnet

J. Brooke,[1] D. Bitko,[1] T. F. Rosenbaum,[1*] G. Aeppli[2]

Published: 11 October 2001

Tunable quantum tunnelling of magnetic domain walls

J. Brooke, T. F. Rosenbaum & G. Aeppli

Nature **413**, 610–613(2001) | Cite this article

Advantage

COMBINATORIAL OPTIMIZATION

?

SAMPLING PROBLEMS

?

Philipp Hauke *et al* 2020 *Rep. Prog. Phys.* **83** 054401
E.J.Crosson , D. Lidar, arXiv:2008.09913v1

HANDS-ON: ROUTING PROBLEM

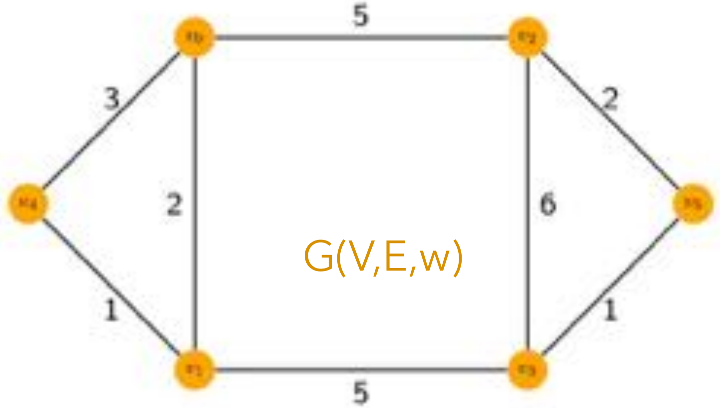
THE CHINESE POSTMAN PROBLEM (CPP)

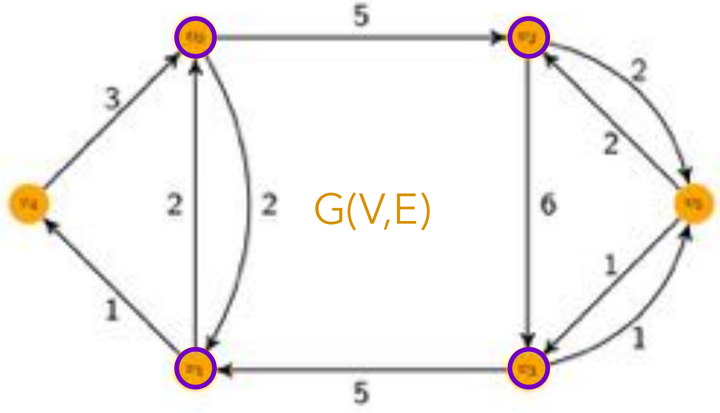
A Chinese postman has to cover his assigned route before returning to the post office. The problems involves finding the length of the shortest closed path traveling across all edges of the network at least once.

HANDS-ON: ROUTING PROBLEM

THE CHINESE POSTMAN PROBLEM (CPP)

A Chinese postman has to cover his assigned route before returning to the post office. The problems involves finding the length of the shortest closed path traveling across all edges of the network at least once.





1. Setting the undirected network G(V,E,w) → edges (streets), nodes (corners), weights (length)

2. The CPP admits solution if the network contains at least one cycle that crosses all the edges exactly once (eulerian cycle)

3a. Networks with even connectivity have a trivial solution (Eulerian cycle)

3b. Networks with odd connectivity double the edges between odd nodes to guarantee the existance of Eulerian cycle. Edges connecting odd-degree nodes can be crossed more than once (extra-path).

4. Choose the shortest extra paths between odd nodes in the new network.

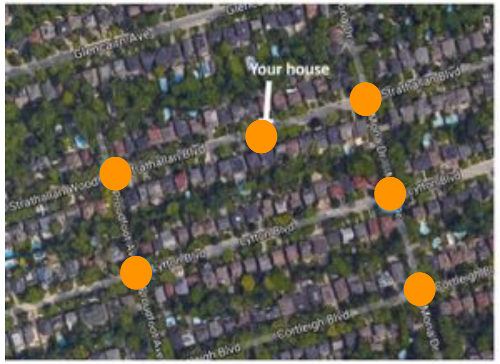
THE CPP SOLUTION IS EQUAL TO THE SUM OF ALL NETWORK WEIGHTS + EXTRA PATH

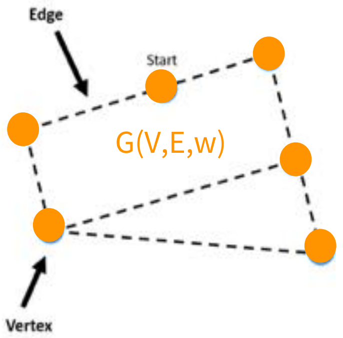
$$l(G) = \sum_{e \in E} w(e) + M_{min}$$

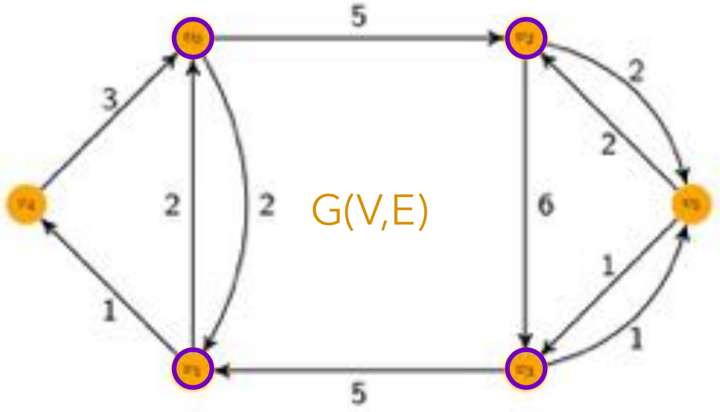
HANDS-ON: ROUTING PROBLEM

THE CHINESE POSTMAN PROBLEM (CPP)

A Chinese postman has to cover his assigned route before returning to the post office. The problems involves finding the "length" of the shortest closed path traveling across all edges of the network at least once.







THE CPP SOLUTION IS EQUAL TO THE SUM OF ALL NETWORK WEIGHTS + EXTRA PATH

$$l(G) = \sum_{e\in E} w(e) + M_{min} = \sum_{e\in E} w(e) + \min_{\alpha} m(\pi_\alpha)$$

The "length" of each extra path between pairs of odd degree nodes is determined by computing the minimum across all the possible paths.

$$m(\pi_1) = W(v_0, v_1) + W(v_2, v_3) = 2 + 3 = 5$$
$$m(\pi_2) = W(v_0, v_2) + W(v_1, v_3) = 5 + 5 = 10$$
$$m(\pi_3) = W(v_0, v_3) + W(v_1, v_2) = 7 + 7 = 14.$$

$$l(G) = 25 + 5 = 30$$

OPTIMIZATION PROBLEM → $\min_{\alpha} m(\pi_\alpha)$

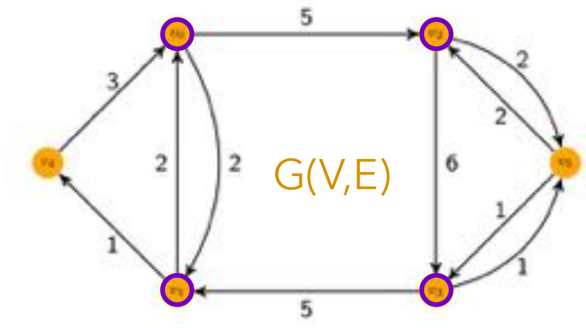
HANDS-ON: ROUTING PROBLEM as QUBO

Defining binary variables

Binaries x_{ij} are paths between the odd degree nodes i and j

$$x = (x_{01}, x_{02}, x_{03}, x_{10}, x_{12}, x_{13}\ , x_{20}, x_{21}\ , x_{23}, x_{30,} x_{31}, x_{32})$$

A bitstring x is a path across all odd degree nodes



Setting constraints

MAIN TERM $P_1(x)$ $P_2(x)$

$$Q(\boldsymbol{x}) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} W_{ij}x_{ij}^2 + p\sum_{i=0}^{n-1}\left(1 - \sum_{j=0}^{n-1}(x_{ij} + x_{ji})\right)^2 + p\sum_{i,j,k}^{n-1}(x_{ik}x_{jk} + x_{ki}x_{kj})$$

$W_{ij} = minimum\ path\ between\ i\ and\ j$

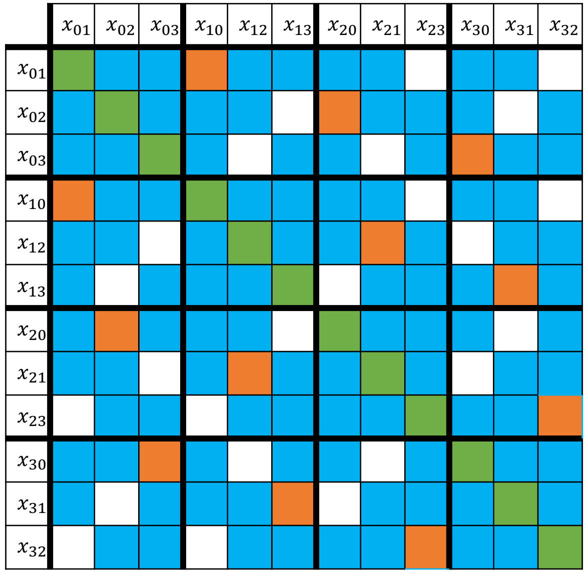
$p = constant$

QUBO

MAIN TERM   Sum over all the minimum paths across odd nodes

$P_1(x)$ Avoid double counting

$P_2(x)$ Each odd node appears only once (double counting makes the node odd again)

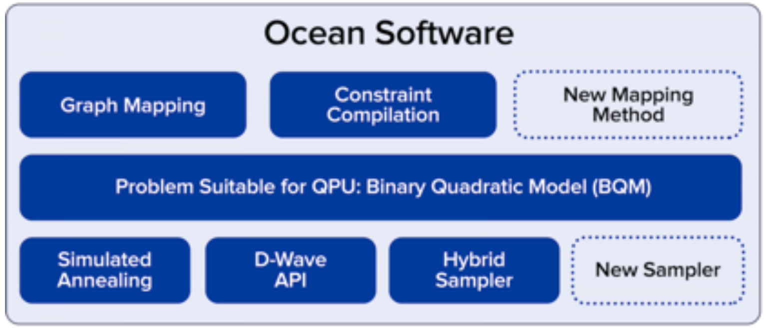


I.Siloi, V. Carnevali, B. Pokarel, M. Fornari, R. Di Felice,Quantum Machine Intelligence (2021) 3:3

HANDS-ON: ACCESSING D-Wave

Loading the problem(bqm)

```
bqm = dimod.BinaryQuadraticModel.from_qubo(Q)
```




```
pip install dwave-ocean-sdk
```

Solver

```
solver =
DWaveSampler(endpoint='https://cloud.dwavesys.c
om/sapi/',

token='xxxxxxxxxxxxxxxxxxxx',
        solver='DW_2000Q_6')

__, target_edgelist, target_adjacency =
solver.structure
```

Embedding on Chimera Graphs

```
sampler = FixedEmbeddingComposite(solver,embedding)

emb = find_embedding(Q, target_edgelist)
```

Setting annealing parameter

```
response = sampler.sample(bqm, num_reads=10000,
chain_strength=m, annealing_time=100)
```

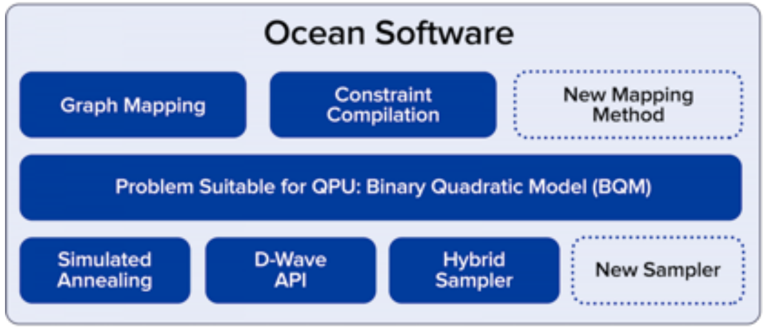
HANDS-ON: ACCESSING D-Wave

Loading the problem(bqm)

```
bqm = dimod.BinaryQuadraticModel.from_qubo(Q)
```

Solver

```
solver =
DWaveSampler(endpoint='https://cloud.dwavesys.c
om/sapi/',

token='xxxxxxxxxxxxxxxxxxxx',
        solver='DW_2000Q_6')

__, target_edgelist, target_adjacency =
solver.structure
```




```
pip install dwave-ocean-sdk
```

Embedding on Chimera Graphs

```
sampler = FixedEmbeddingComposite(solver,embedding)

emb = find_embedding(Q, target_edgelist)
```

Setting annealing parameter

```
response = sampler.sample(bqm, num_reads=10000,
chain_strength=m, annealing_time=100)
```

HANDS-ON: ACCESSING D-Wave

Loading the problem(bqm)

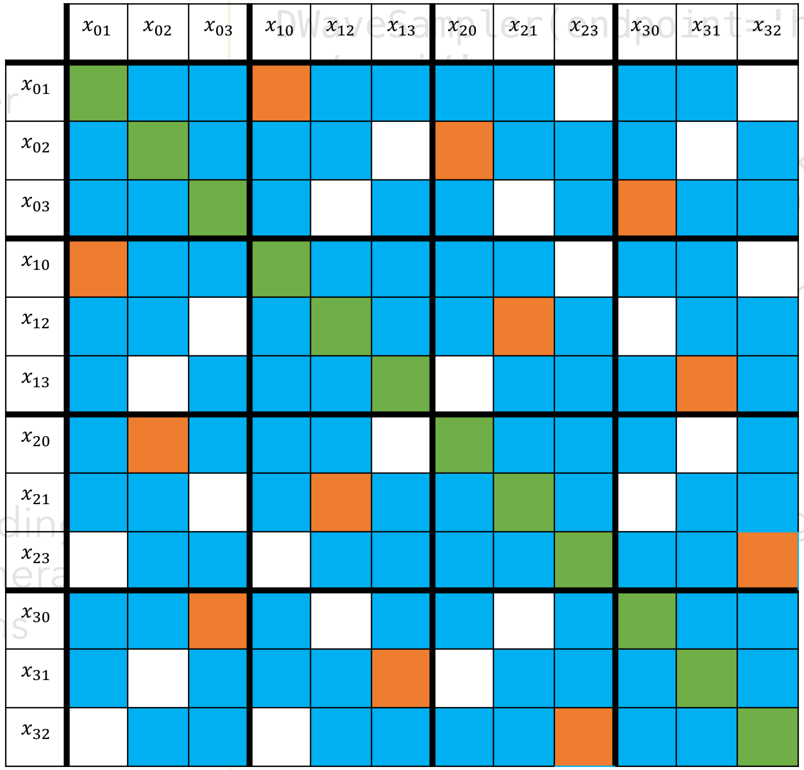
```
bqm = dimod.BinaryQuadraticModel.from_qubo(Q)
```

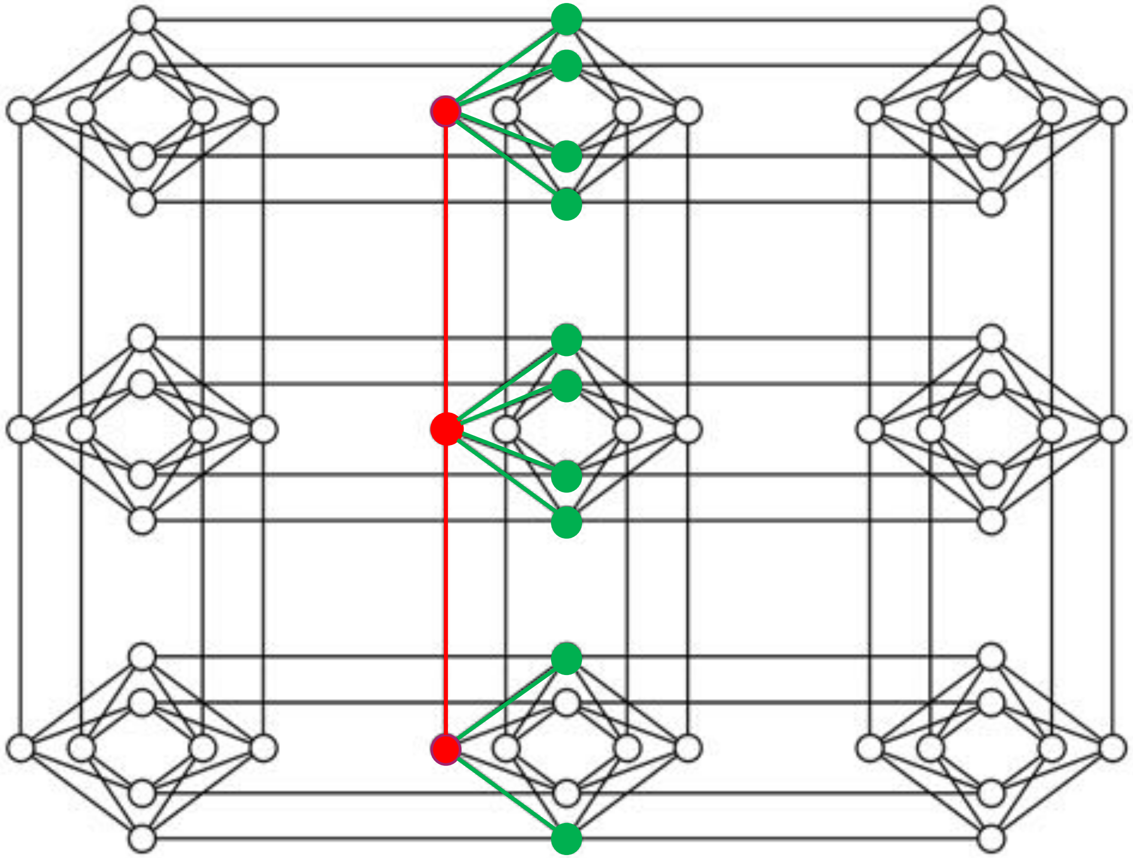
Ocean Software

Graph Mapping | Constraint Compilation | New Mapping Method

Solver

```
solver =
```





Embedding on Chimera Graphs

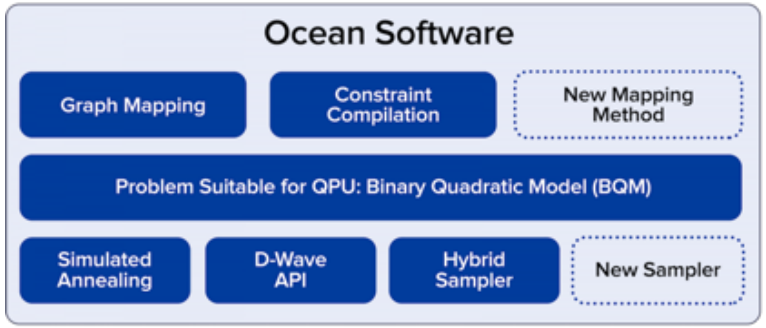
Setting annealing parameter

```
response = sampler.sample(bqm,
chain_strength=m, annealing_time=100)
```

HANDS-ON: ACCESSING D-Wave

Loading the problem(bqm)

```
bqm = dimod.BinaryQuadraticModel.from_qubo(Q)
```




```
pip install dwave-ocean-sdk
```

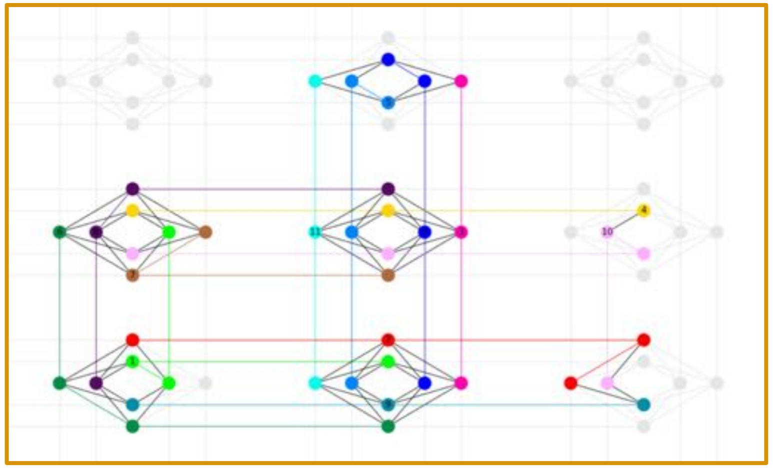
Solver

```
solver =
DWaveSampler(endpoint='https://cloud.dwavesys.c
om/sapi/',

token=‘xxxxxxxxxxxxxxxxxxxx’,
         solver='DW_2000Q_6’)

__, target_edgelist, target_adjacency =
solver.structure
```

N_{odd}	Logical Qubits	Q_{ij}	Physical Qubits	Embedding Max Chain
2	2	2	2	1
4	12	54	44	5
6	30	256	262	12
8	56	700	864	23

Embedding on Chimera Graphs

```
sampler = FixedEmbeddingComposite(solver,embedding)

emb = find_embedding(Q, target_edgelist)
```



Setting annealing parameter

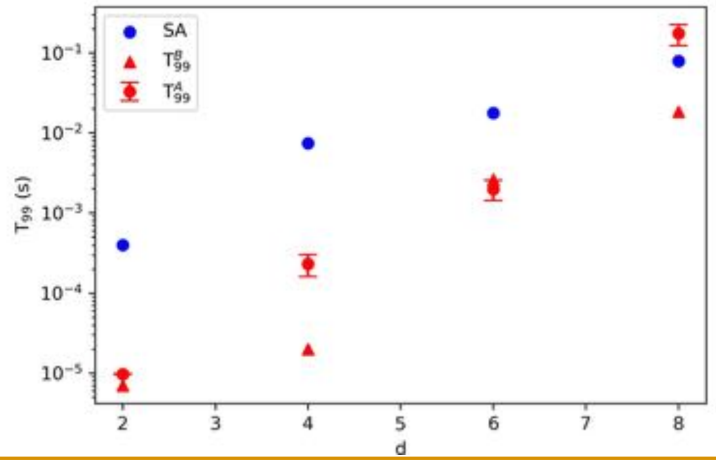
```
response = sampler.sample(bqm, num_reads=10000,
chain_strength=m, annealing_time=100)
```

HANDS-ON: PERFORMANCE

Metrics

$$P_{gs} = \frac{\#correct\ solutions}{\#total\ runs}$$

$$TTS = \frac{\log(1-0.99)}{\log(1-P_{gs})} \cdot T_{annealing}$$

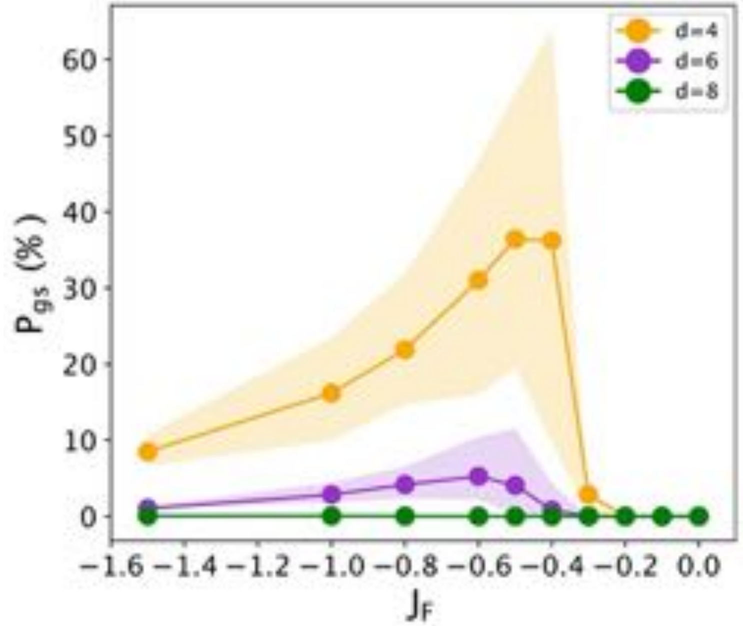


Intra-chain coupling in physical qubits

- J_F determines the ability of the chain to act as a single variable;
- J_F couplings should be strong enough to avoid chain-breaking without dominating the dynamics

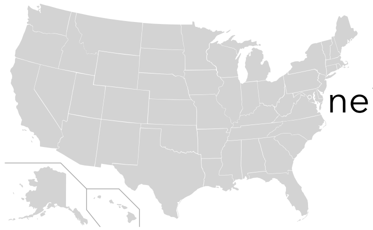
Embedding

d	Logical qubits	Number of Q_{ij} terms	Physical qubits	P_{gs} (%)
2	2	2	2	99.99
4	12	54	44	87.83
6	30	256	248	51.07
8	56	700	864	0.21



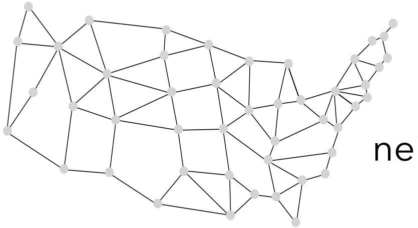
HANDS-ON: GRAPH COLORING PROBLEM

Problem



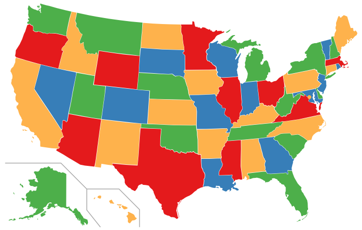
1 state → 1 color
neighboring states → different colors

Defining binary variables



state = node
neighboring state = edge

QUBO



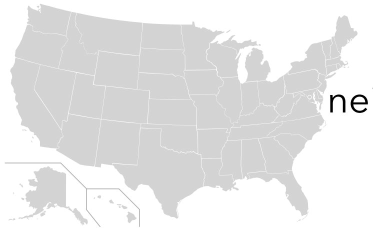
$$Q = P_1(x) + P_2(x)$$

$$P_1 = p_1 \sum_{i \in V} \left(1 - \sum_j x_{v,i}\right)^2$$ one color each node

$$P_2 = p_2 \sum_{i,k \in E} \sum_j x_{i,j}\, x_{k,j}$$ neighboring state, different color

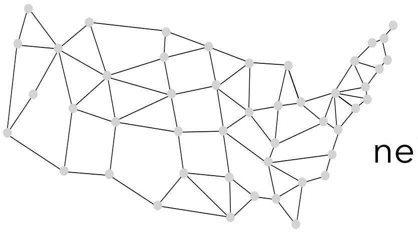
HANDS-ON: GRAPH COLORING PROBLEM

Problem



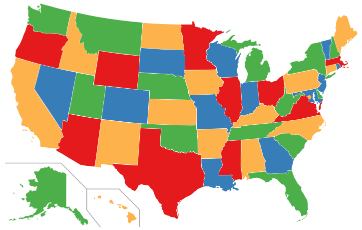
1 state → 1 color
neighboring states → different colors

Defining binary variables



state = node
neighboring state = edge

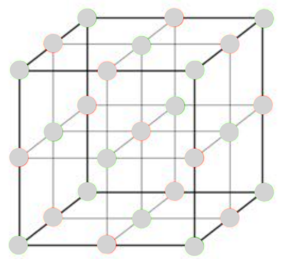
QUBO



$$Q = P_1(x) + P_2(x)$$

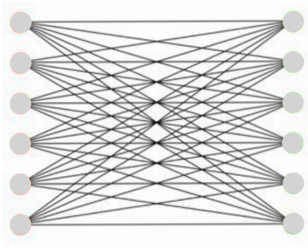
$$P_1 = p_1 \sum_{i \in V} \left(1 - \sum_j x_{v,i}\right)^2$$ one color each node

$$P_2 = p_2 \sum_{i,k \in E} \sum_j x_{i,j}\, x_{k,j}$$ neighboring state, different color



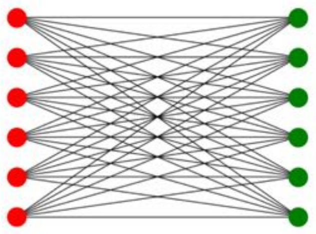
Arrange atoms in a crystal structure according to physical constraints

1 cation → bound to 6 anions
1 anion → bound to 6 cations



Colors as variables

- Oxidation states
- Vacancies
- Bond-length



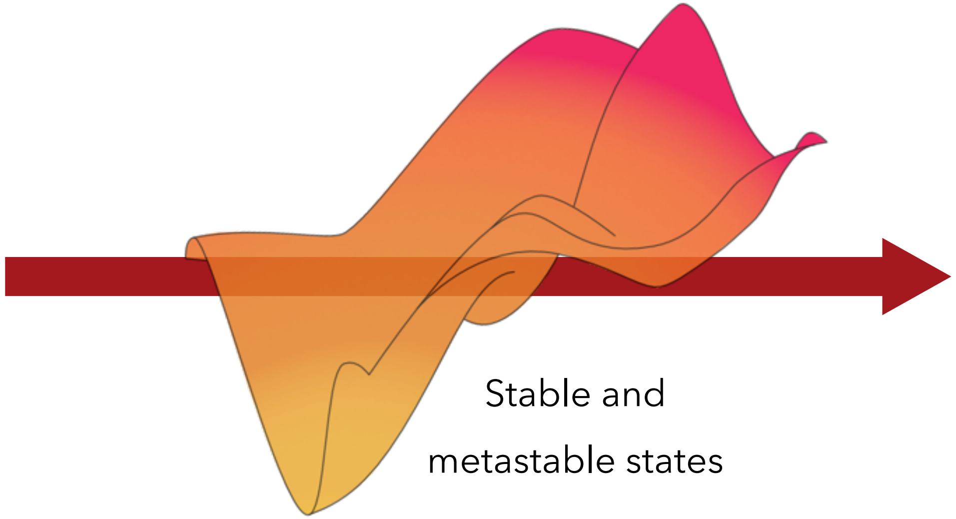
QUBO terms have physical meaning

$$Q(x) = Q_1(x) + Q_2(x) + \cdots + Q_n(x)$$

- Atomic coordination
- Anion-Cation attraction
- Composition
- …

Energy landscape exploration

Combinatorial complexity

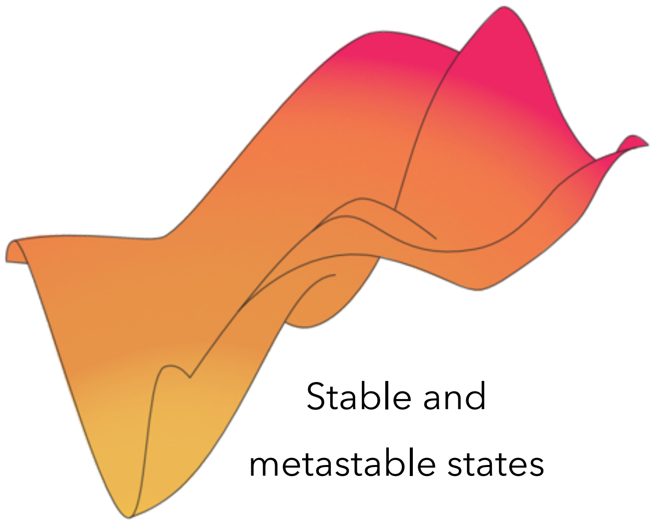


Ising Hamiltonian/QUBO

Optimization problem

Energy landscape exploration

Combinatorial complexity

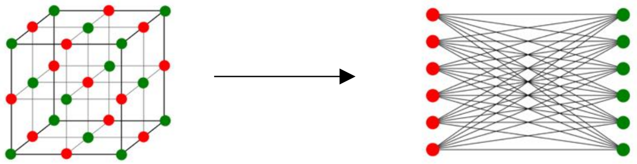


Ising Hamiltonian/QUBO

Optimization problem

METHOD

FROM CRYSTAL STRUCTURE TO NETWORK



atomic sites → nodes $\{v_i\}$

bonds → edges $\{e_i\}$

$G\ \{V, E\}$

ISING/QUBO DESIGN

$$Q(x) = p_1 P_1(x) + p_2 P_2(x) + \cdots$$

$Q(x) = 0$ Ground state

$Q(x) \neq 0$ Excited states

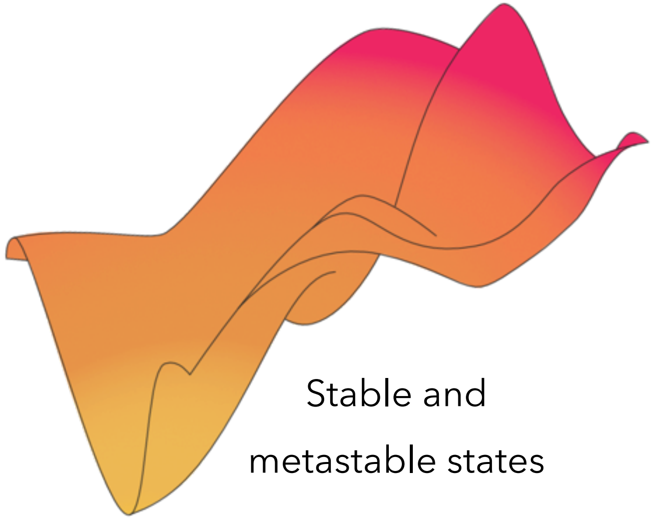
EXCITED STATE have a PHYSICAL MEANING

ALGORITHM

Quantum Annealing ⟷ D-Wave as structural model generator

Energy landscape exploration

Combinatorial complexity

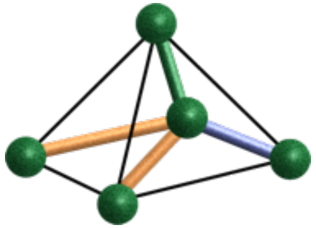


Stable and metastable states

Ising Hamiltonian/QUBO

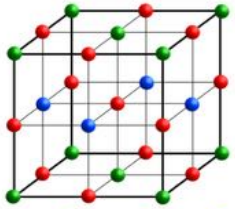
Optimization problem

DISORDERED SYSTEMS



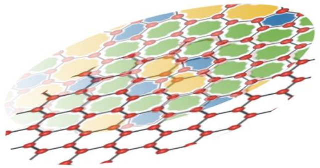
LIQUID SILICON

Competition between short and long range order



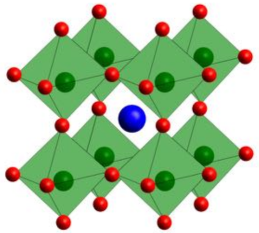
HIGH ENTROPY ALLOYS

Competition between energy and entropy



DEFECTED GRAPHENE

stability of multiple vacancies in graphene sheets

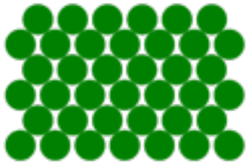


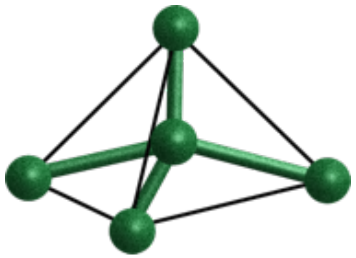
PEROVSKITES

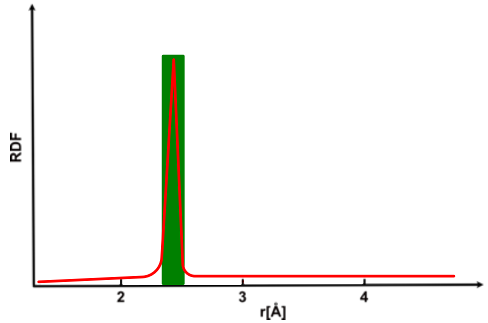
Competition between order and disorder

LIQUID SILICON

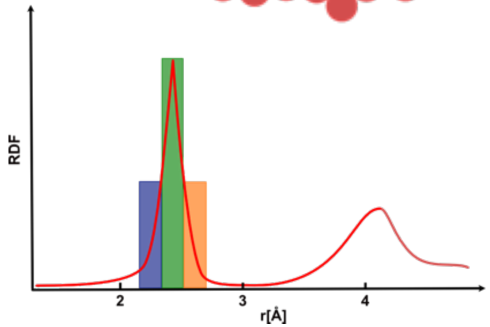
CRYSTALLINE

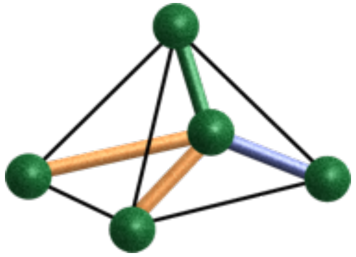






LIQUID





Radial Distribution Function

Bonds length distribution in the crystal

$$x = \{x_{i\alpha_n}\}$$

Silicon structural model as string of binaries: encoding #bonds and available bond lengths

GROUND STATE

$$Q(x) = 0$$

$$Q(x) = p\sum_i \left(1 - \sum_n x_{i\alpha_n}\right)^2 + \sum_{i,n} |\alpha_n - \alpha_c| x_{i\alpha_n}^2 + \sum_n \left(N_{\alpha_n} - \sum_i x_{i\alpha_n}\right)^2$$

How shrunk/ stretched the bonds are

How many bonds are not crystalline

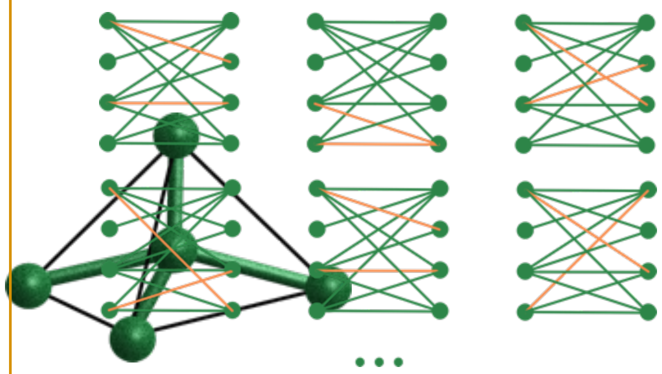
EXCITED STATES

$$Q(x) \neq 0$$

tuning p !

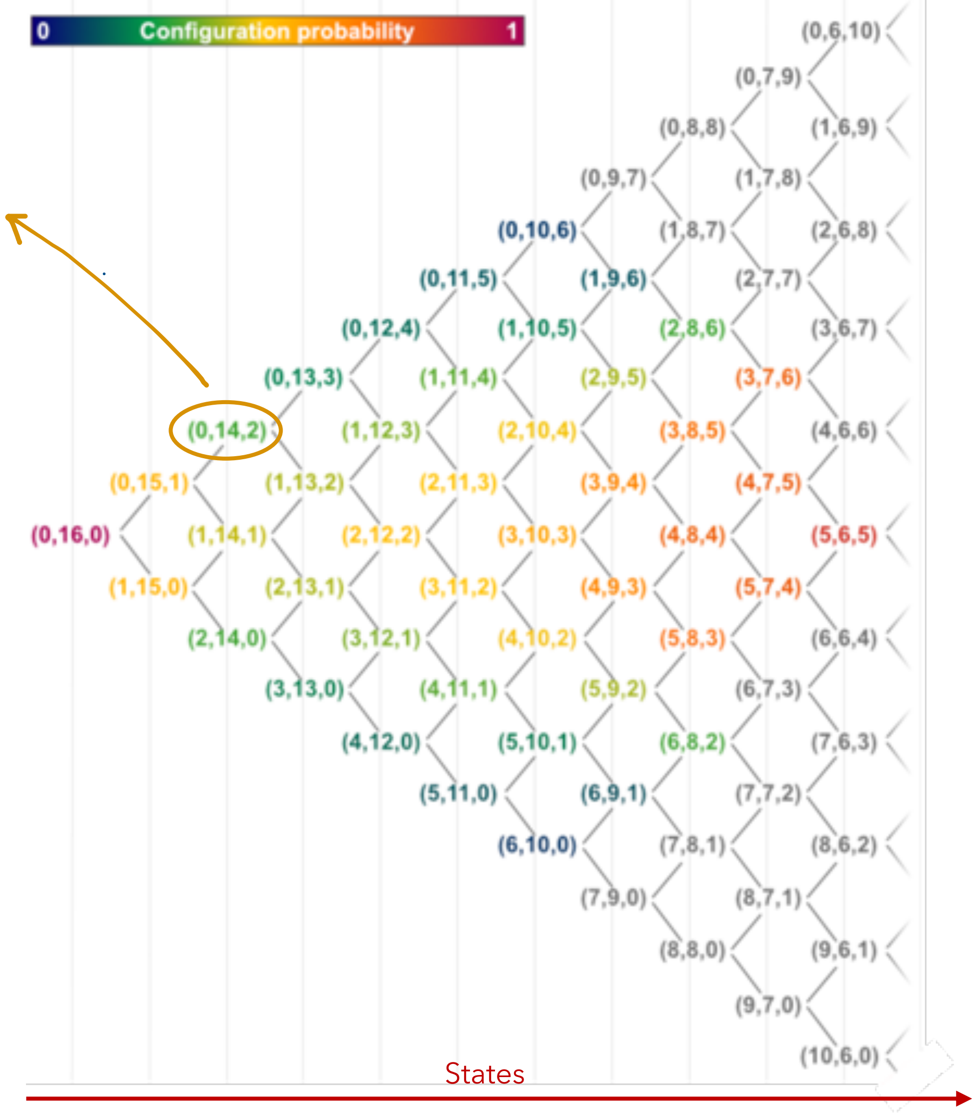
STRUCTURAL MODELS WITH QUANTUM ANNEALING

STRUCTURAL MODELS

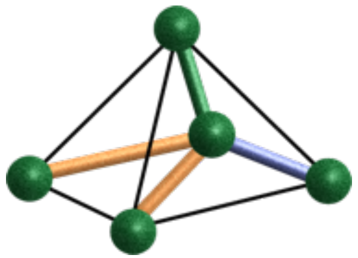


CRYSTALLINE

$$\#structures = \frac{16!}{14!\,2!} = 120$$



Non-crystalline bonds



LIQUID

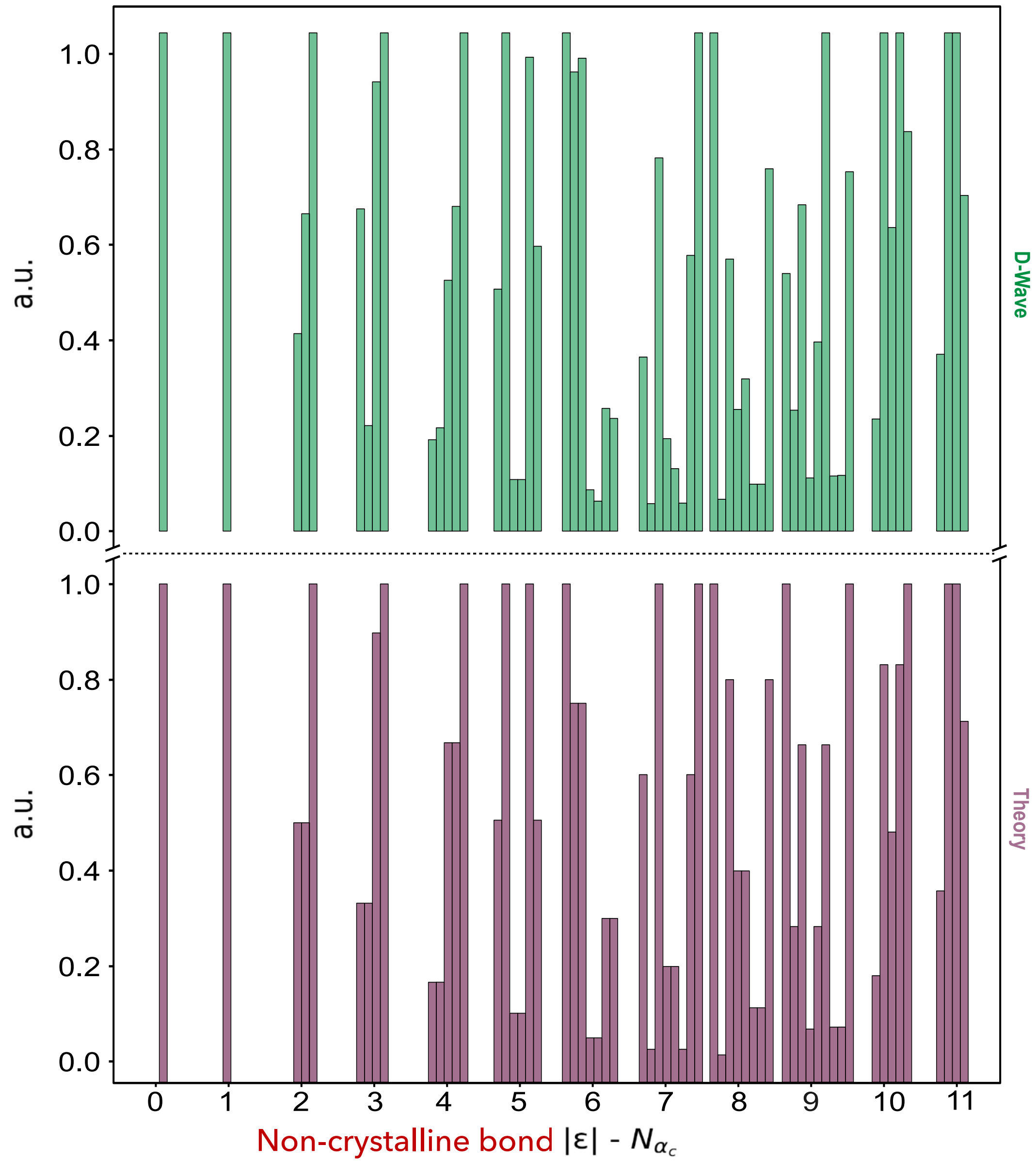
FILTERING EXCITATIONS

Binary strings that do not respect constraints are filtered out *a posteriori*

DETAILS

- Conventional cell with 8 atoms, 16 bonds
- 48 binaries, 408 couplings, 437 qubits

STRUCTURAL MODELS WITH QUANTUM ANNEALING



DW distribution of models close to theoretical one

Better resolution with larger statistic

General approach

CONCLUSIONS

D-WAVE

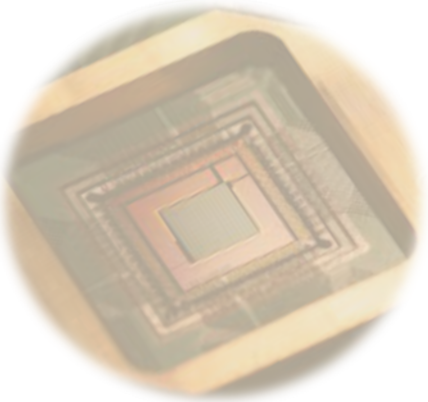
Alghoritm: Quantum annealing

Models: Ising & QUBO

HANDS-ON: EXCITED STATES

D-Wave as model generator for condensed matter problems

Liquid Silicon



HANDS-ON: GROUND STATE

Implementation of the Chinese postman problem on D-Wave

D-Wave perfomances: TTS, P_{GS}

STATISTICAL APPROACH

How to develop a QUBO where the excited states have a physical meaning