Performance des ordinateurs quantiques analogiques pour l'optimisation

Daniel Vert
Coordinateur du Hub Advanced Engineering & Computing
Systematic Paris Region

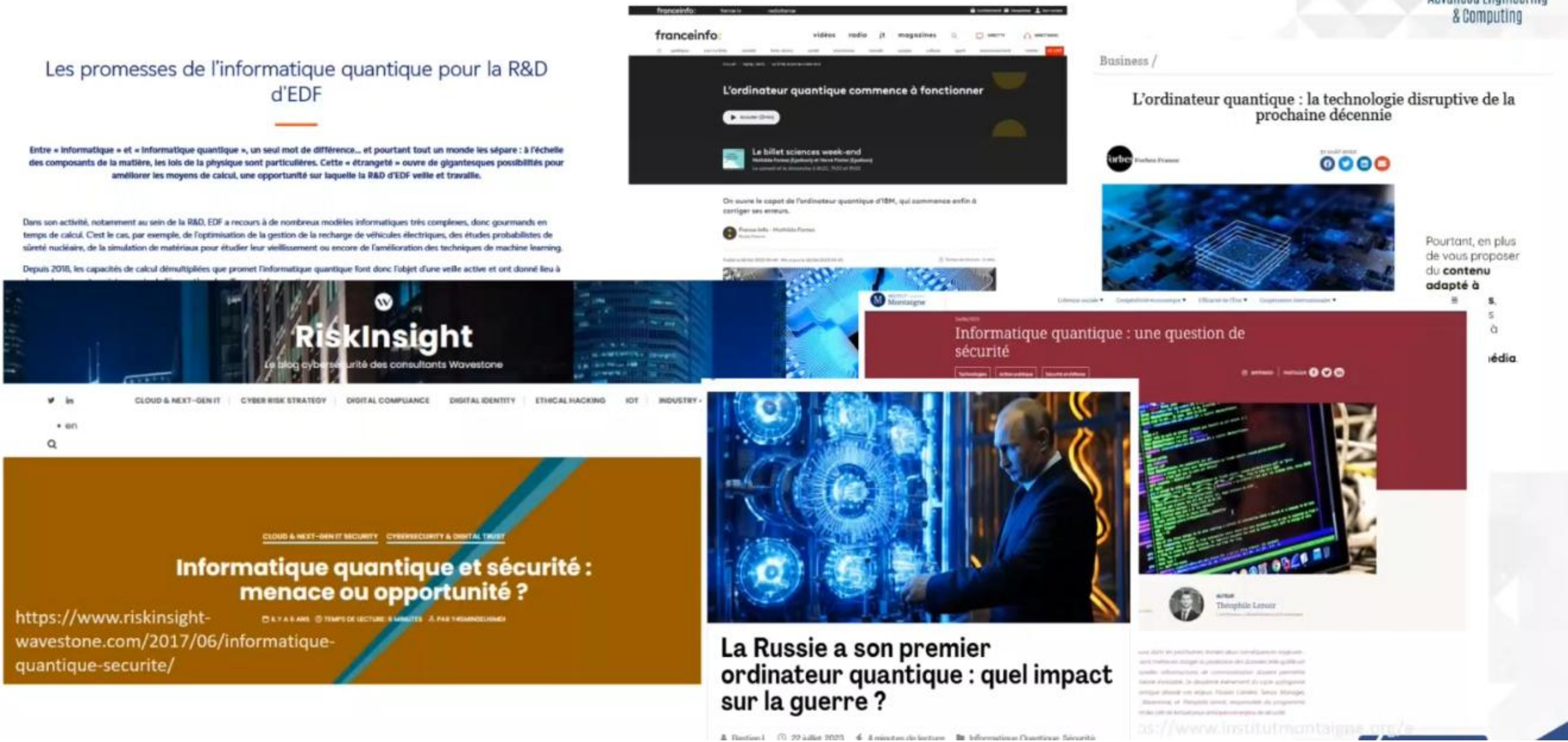
daniel.vert@systematic-paris-region.org

Daniel VERT

Systematic
Paris Region Deep Tech Ecosystem

Why quantum computing? A new digital wave? A hype ...

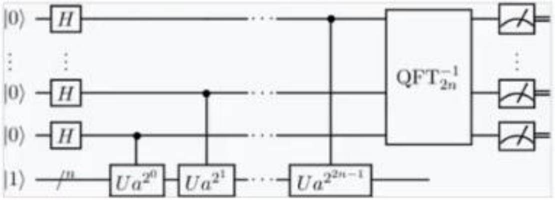
Advanced Engineering & Computing



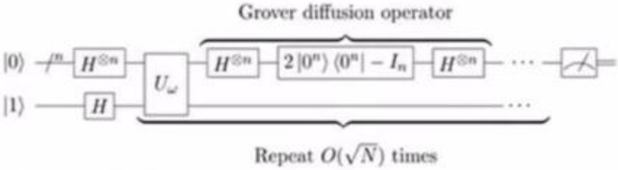
Quantum machines: 2 families

Universal quantum gates

- Google/IBM etc <= 127 qubtis
- Algo :
  - Shor (Shor, 1994) factorize an integer N in time ((log N)^3))



  - Grover (Grover, 1996) quadratic acceleration to find a value



  - QAOA (Farhi et al., 2014) Quantum Approximate Optimization Algorith

Quantum annealing

- **D-Wave**/qilimanjaro >= 5000 qubits
- Algo :
  - focus on a particular Hamiltonian → Ising model

$$\mathcal{H}_P = \sum_i h_i \sigma_i^z + \sum_{(ij)} J_{i,j} \sigma_i^z \sigma_j^z$$

What is quantum annealing?
How does it work?
How is it programmed?
What is its performance?

Machine quantique analogique

- It is a machine specialized in solving a difficult NP-optimization problem.
- Using an algorithm similar to a simulated → (meta)heuristic annealing to quickly solve complex problem classes (optimization problems, machine learning problems, or operations research problems) inspired by statistical physics.

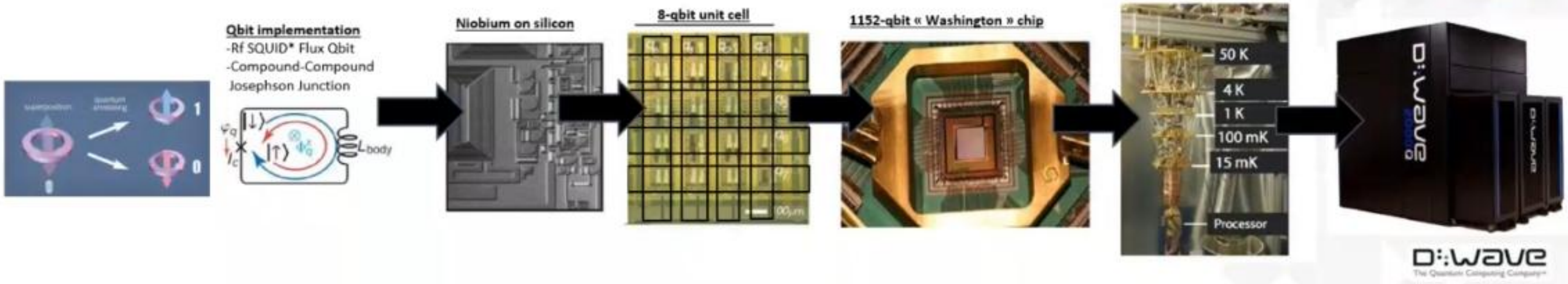
https://www.oezratty.net/wordpress/2018/comprendre-informatique-quantique-algorithmes-et-applications/

- D-Wave is thus a kind of optimization oracle for the typical machine problem (spin glasses) using a quantum phenomenon.

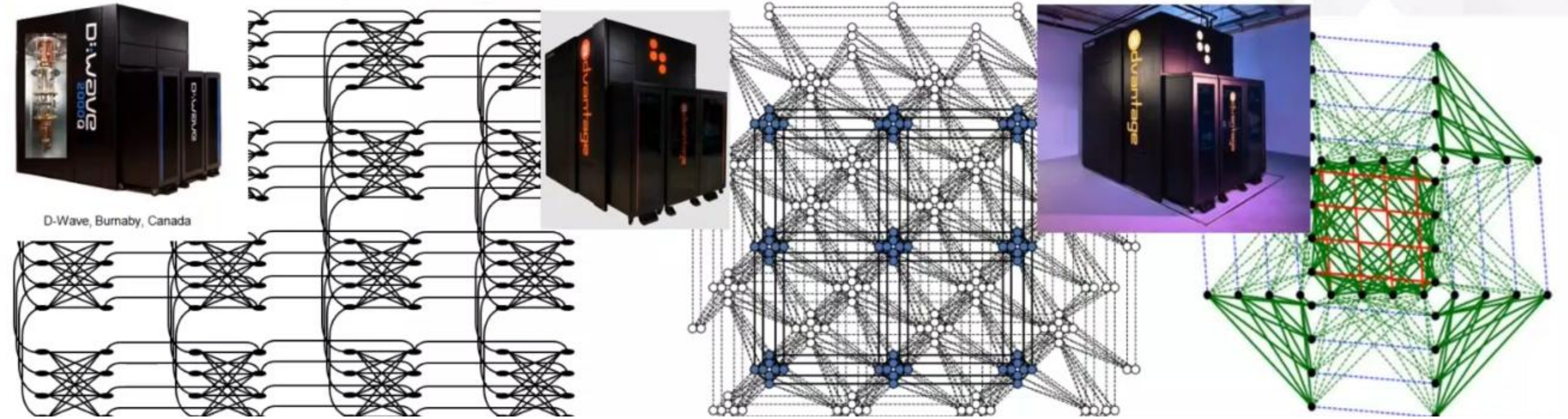
- **Question :**
  - Behavior / Performance on a problem known to be difficult to anneal (acceleration factor between QA and SA)

D-Wave QPU architecture: topologies

- **QPU :**



- D-Wave 2000Q/Advantage/Advantage2.0 → more 2000/5600/7000 qubits



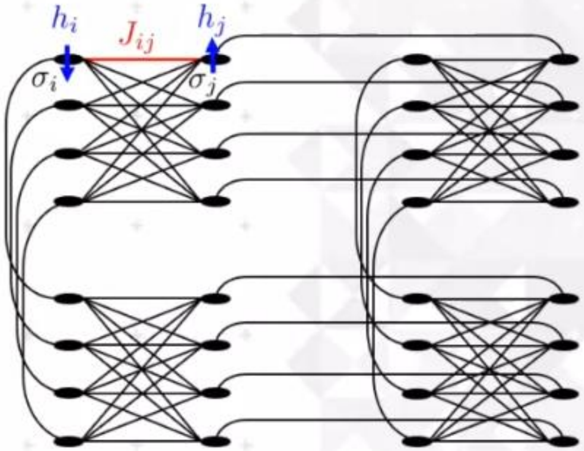
From the machine to the ising model

- D-Wave systems are based on a quantum annealing process whose aim is to minimize Ising's Hamiltonian :

$$\mathcal{H}_P = \sum_i h_i \sigma_i^z + \sum_{(ij)} J_{i,j} \sigma_i^z \sigma_j^z$$

h External field
J spin coupling interaction matrix
σ spin vector (or qubit) of values {-1, 1} → variable for which the energy of the system is minimized

- **Input :** The values of the **h**-weights (qubits) and **J**-weights (connections) for each qubit and mapped on the Chimera graph.
- Execute a quantitative/adiabatic annealing → The result should be the lowest energy state (lowest energy) of the HP problem.
- **Output** : The spin of each qubit is measured at the end of the process → This set of spins defines the lowest energy state found.



Chimera graph 2 x 2 → 32 qubits (**h**) / 80 coupling (**J**)

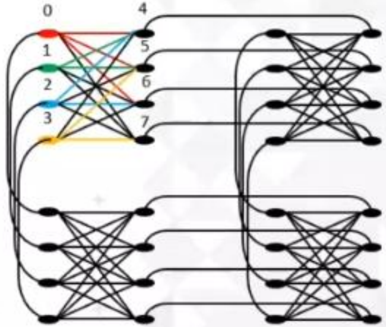
QUBO integration on D-Wave and limitation

Advanced Engineering & Computing

Integrating the QUBO matrix into the internal graph of physical qubits

- Directly integrable approach (fictitious example) :

	0	1	2	3	4	5	6	7
0	17	0	0	0	−5	2	−21	0
1	0	17	0	0	−16	−1	0	9
2	0	0	17	0	−1	0	4	0
3	0	0	0	17	0	16	0	8
4	0	0	0	0	17	0	0	0
5	0	0	0	0	0	17	0	0
6	0	0	0	0	0	0	17	0
7	0	0	0	0	0	0	0	17



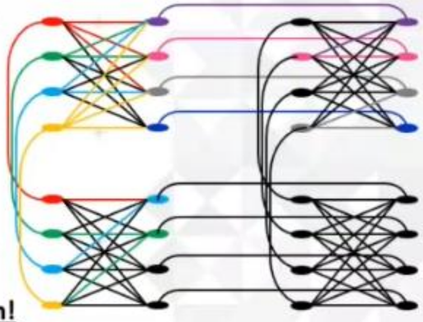
No problem for D-Wave!

- 1 variable → N qubits

Not enough connections between qubits (hi) → maximum 6!

- Variable 1,2,5,7 duplicated 3 times
- Variable 0,3,4,6 duplicated 2 times
- Total number of connections: 15
- Total number of qubits used: 20

	0	1	2	3	4	5	6	7
0	17	0	4	0	−5	2	−21	0
1	0	17	0	−2	−16	−1	0	9
2	0	0	17	0	−1	0	4	0
3	0	0	0	17	9	16	−1	8
4	0	0	0	0	17	−6	0	0
5	0	0	0	0	0	17	0	0
6	0	0	0	0	0	0	17	−12
7	0	0	0	0	0	0	0	17



Several physical qubits are ONE variable of the problem!

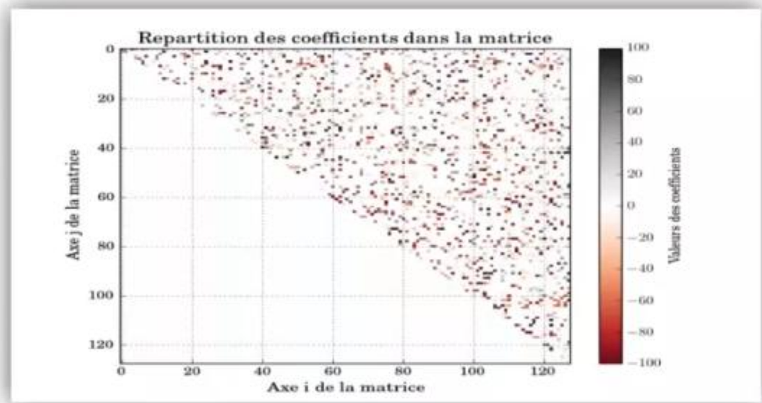
If the graph induced by the matrix's non-zero couplings is not isomorphic to Chimera's graph → it is necessary to duplicate the variable(s) on several physical qubits!

Limitation (next)

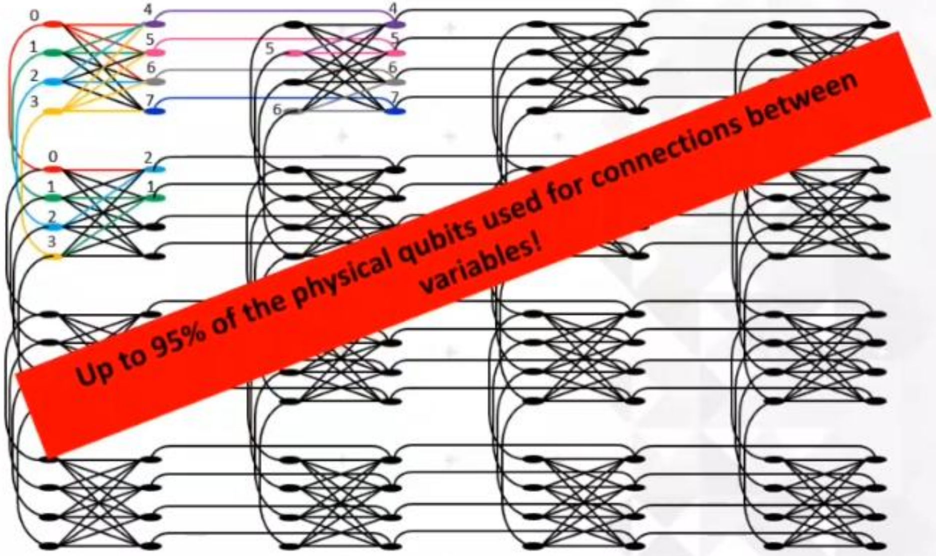
Integrating the QUBO matrix into the internal graph of physical qubits

1 variable → 1 qubit

Total number of connections: 384 edges!



128 h_i (qubits) + 8128 J_{ij} (terms)



Chimera Graph 4x4 → 128 qubits

The topology represents only 4% (worst case) of the total number of couplers needed to integrate the matrix.

Problem definition

- **Matching problem on a general graph :**

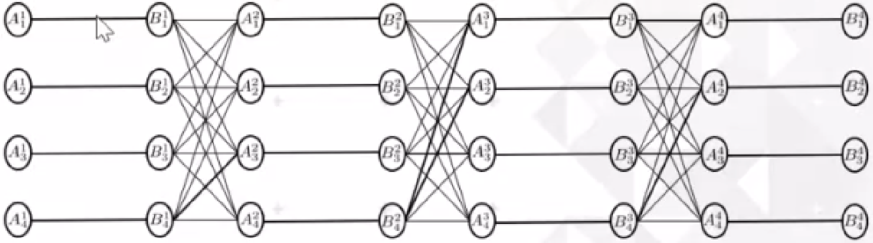
  Goal: select a set of edges that gives a **matching** in order to maximize the number of vertices covered

  Polynomial Problem

- **Maximum matching problem :**

[sasaki-1988] → Special cases to require an **exponential** number of iterations to reach an optimal solution

→ (trivial to solve algorithmically)
→ And even to the eye



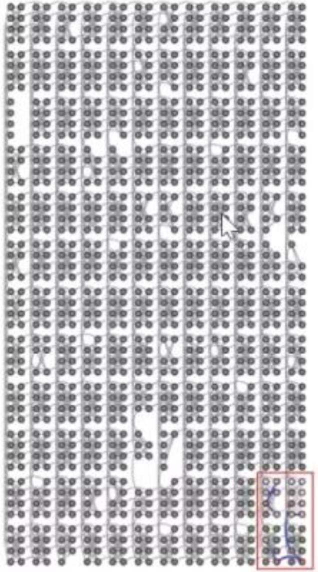
Vertices A not connected to each other and idem for vertices B

Number of vertices: $O(n^3)$
Number of edges: $O(n+1)^3$
Number of edges in an optimal solution: $O(n^2)$
→ It's hard for simulated annealing

Objective: Compare QA to SA

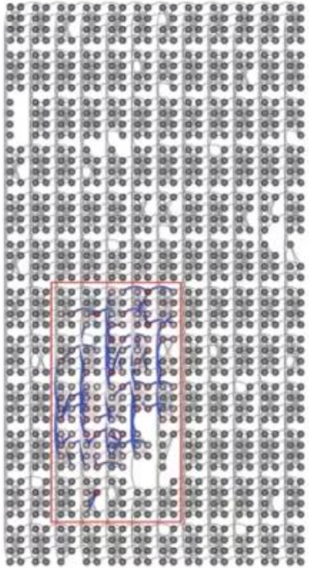
Instance on D-Wave

- G_1 -



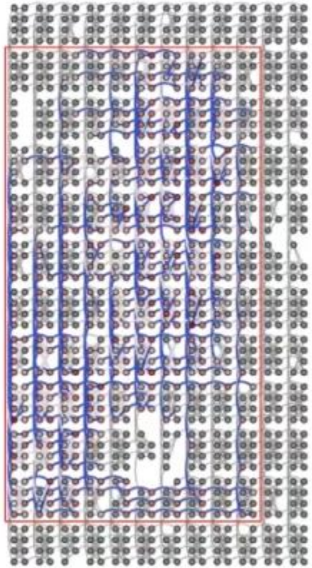
8 variables
16 qubits physiques → ~2%
Moy. dup. : 2.0
12 coefficients non 0

- G_2 -



27 variables
100 qubits physiques → ~10%
Moy. dup. : 3.7
72 coefficients non 0

- G_3 -

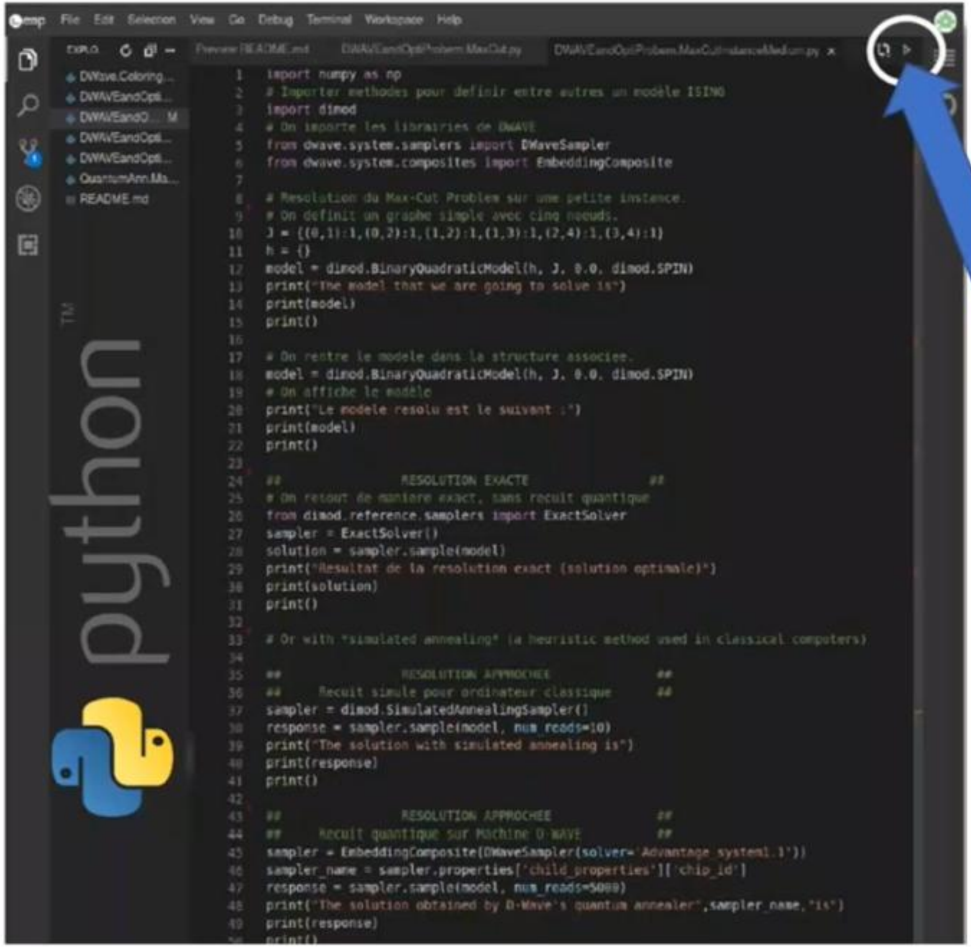


64 variables
431 qubits physiques → ~40%
Moy. dup. : 6.7
240 coefficients non 0

- G_4 -

125 variables
951 qubits physiques → ~87%
Moy. dup. : 7.6
600 coefficients non 0

Solve a problem on D-Wave machine ?



Direct resolution
online terminal displaying the result

easy no ?

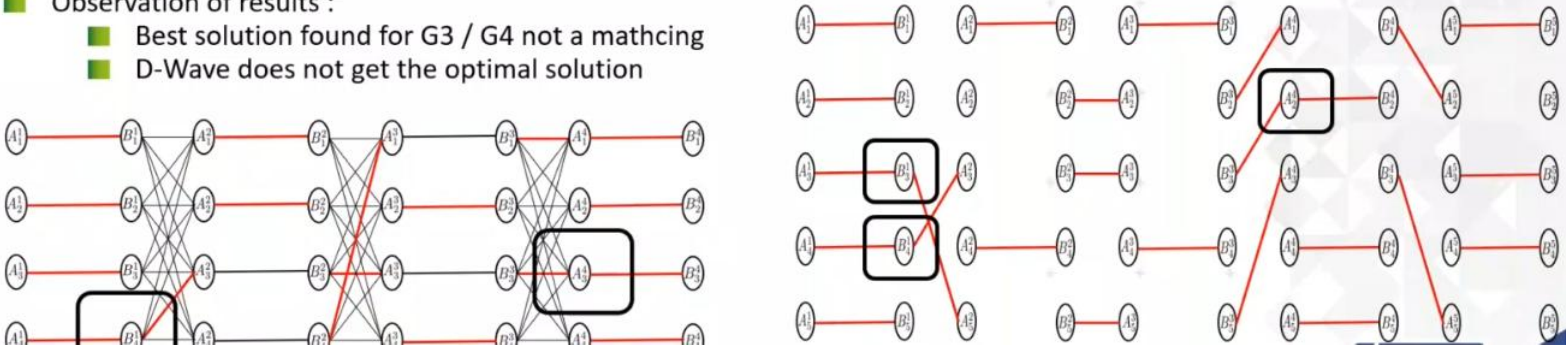
```
The solution obtained by D-Wave's quantum annealer Advantage_system1.1 is
   0  1  2  3  4 energy num_oc. chain_.
0 -1 +1 +1 -1 +1   -4.0    1398     0.0
1 +1 -1 -1 +1 -1   -4.0    1029     0.0
2 -1 +1 +1 -1 -1   -4.0    1487     0.0
3 +1 -1 -1 +1 +1   -4.0    1084     0.0
4 -1 +1 +1 +1 -1   -2.0       1     0.0
5 +1 -1 -1 -1 +1   -2.0       1     0.0
['SPIN', 6 rows, 5000 samples, 5 variables]
```

RESULTS ON G_1 / G_2 / G_3 / G_4

	opt.	sans vote				
		best	worst	mean	median	stdev
G_1	-68	-68	-9	-66.8	-68	4.6
G_2	-495	-495	-29	-398.2	-388	48.1
G_3	-2064	-1810	-505	-1454.8	-1548	157.7
G_4	-6275	-5527	-2507	-4609.9	-4675	346.5

- **Results :**
  - G1 : Optimal solution more 9600 times.
  - G2 : Optimal solution 662 times → 6% success rate
  - G3 : Difference with the lowest cost solution found is about 15% (-1810 for -2064)
  - G4 : Difference with the lowest cost solution found is about 10% (-5527 for -6275)
- the smallest instances are trivially solved but not the biggest ones

- Observation of results :
  - Best solution found for G3 / G4 not a mathcing
  - D-Wave does not get the optimal solution



Résults on Advantage

- **Advantage4.1 (Peg.)**
  - **5627 qubits**

	#var.	#qubits	#Opt.	Opt.	Best	Worst	Mean	Median	Stdev
G1	8	8	1000	-68	-68	-68	-68	-68	0
G2	27	49	80	-495	-495	-225	-389.2	-387	34.5
G3	64	174	0	-2064	-1937 (x3)	-916	-1565.5	-1553	137.8
G4	125	471	0	-6275	-5278	-2782	-4268.9	-4278	370.4
G5	216	1033	0	-15588	-12134	-6527	-9629.3	-9549	912.3
G6	343	2372	0	-33663	-23383	-8301	-16299.9	-16528	2286.4
G7	512	4186	0	-65600	-40019	-11361	-27278.3	-27734	4645.5

→

G3 → ~6%.
G4 → ~16%
G5 → ~22%
G6 → ~31%
G7 → ~39%

Results on Advantage .... Identical to D2000Q ?
From 6 connections to 15!

Question1 : need post-processing? If yes → Constraint(s): fast (keeps the quantum advantage) and gives a **valid** solution!
Question 2 : Is it possible to improve results?

Is it possible to improve results?

```
###################################################################################################
#################################### partie 2 : dwave_leap ########################################
###################################################################################################

import dwave_sapi2
from dwave_sapi2.remote import RemoteConnection
from dwave_sapi2.core import solve_ising, solve_qubo

url = 'https://cloud.dwavesys.com/sapi/'
token = 'DEV-32b473f753f4641ca3e87d4318f325295665204d'


# create a remote connection
conn = RemoteConnection(url, token)

# get the solver
solver = conn.get_solver('DW_2000Q_6')
#solver = conn.get_solver('Advantage_system4.1')

params = { "num_reads": 1000, "max_answers": 1000, "answer_mode": "histogram", "auto_scale": True, "annealing_time": 20}, "num_spin_reversal_trans

# solve QUBO problem with parameters
answer = solve_qubo(solver, QUBO, **params)
print"\n"
print("Energie:", answer["energies"])
print("Nb occurences:", answer["num_occurrences"])

i = 0
count = 0
while (answer["energies"][i]*(norma) <= -68) :

        count = count + answer["num_occurrences"][i]
        i = i + 1


print([energie*(norma) for energie in answer["energies"]])


print("Taux de réussite : " + str(float(count)/sum(answer["num_occurrences"])))

print("Energie moyenne : " + str(sum([answer["energies"][i] * answer["num_occurrences"][i] for i in range (len(answer["energies"]))]) / sum(answer["num_occurrences"])*(norma)))
```

Advanced Engineering

The examples below use the following setup:

```
>>> from dwave.system import DWaveSampler, LeapHybridSampler, LeapHybridCQMSampler
...
>>> qpu_advantage = DWaveSampler(solver={'topology__type': 'pegasus'})
>>> qpu_2000q = DWaveSampler(solver={'topology__type': 'chimera'})
>>> hybrid_bqm_sampler = LeapHybridSampler()
>>> hybrid_cqm_sampler = LeapHybridCQMSampler()
```

- anneal_offset_ranges
- anneal_offset_step
- anneal_offset_step_phi0
- annealing_time_range
- beta_range
- category
- chip_id
- couplers
- default_annealing_time
- default_beta
- default_programming_thermalization
- default_readout_thermalization
- extended_j_range
- h_gain_schedule_range
- h_range
- j_range
- max_anneal_schedule_points
- max_h_gain_schedule_points
- maximum_number_of_biases
- maximum_number_of_constraints
- maximum_number_of_quadratic_variables
- maximum_number_of_variables
- maximum_time_limit_hrs
- minimum_time_limit
- minimum_time_limit_s
- num_biases_multiplier
- num_constraints_multiplier
- num_qubits
- num_reads_range
- num_variables_multiplier
- parameters
- per_qubit_coupling_range
- problem_run_duration_range
- programming_thermalization_range

First observation

Solutions without topologie

		opt.	best	worst	mean	median	stdev
G_1	n	-68	-68	-68	-68	-68	0
	$n^{1.5}$	-68	-68	-68	-68	-68	0
	n^2	-68	-68	-68	-68	-68	0
G_2	n	-495	-495	-495	-495	-495	0
	$n^{1.5}$	-495	-495	-495	-495	-495	0
	n^2	-495	-495	-495	-495	-495	0
G_3	n	-2064	-2064	-1810	-2004.7	-2064	79.9
	$n^{1.5}$	-2064	-2064	-2064	-2064	-2064	0
	n^2	-2064	-2064	-2064	-2064	-2064	0
G_4	n	-6275	-6275	-5528	-5785.3	-5777	178.9
	$n^{1.5}$	-6275	-6275	-6026	-6241.8	-6275	86.1
	n^2	-6275	-6275	-6275	-6275	-6275	0

Paramètres Annealing : - Standard cooling time of the form T_{k+1}=0,95T_k
- Stop annealing at T<10^-3.
- Number of iterations of the Metropolis algorithm
→ For each k to T = cst and n: number of variables in QUBO.

For **n iterations** per temperature plate: Lower quality results → standard solutions.

For **n^2 iterations** per plateau: Results of much better quality, but with a much longer computing time.

- **Better** performance simulated annealing
- **Worst** solutions over 30 cycles: Almost always better than D-Wave over 10,000 cycles.
- Asymptotic regime of the exponential number of iterations of Sasaki & Hajek's theorem not reached with annealing (sol. Opt. reached for G4)

- Instances small enough to remain easy to anneal in the conventional way

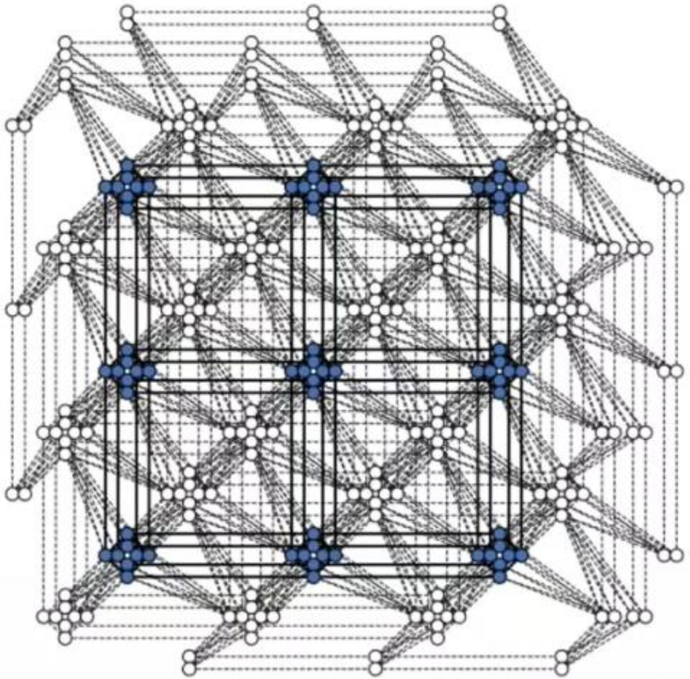
Unduplicated resolved instances for comparison with D-Wave

Second observation : topological bias

Advanced Engineering & Computing

Solutions with topologie (Chim. & Peg.)

		opt.	best	worst	mean	median
G_4 (Chim.)	n	-6275	-2213	3662	1453.9	1401.0
	$n^{1.5}$	-6275	-4526	-2654	-3585.6	-3699.8
	n^2	-6275	-5028	-4027	-4473.1	-4527.0
D-Wave		-6275	-5025	-3551	-4447.7	-4525
G_4 (Peg.)	n	-6275	-3930	-785	-2609.3	-2708.5
	$n^{1.5}$	-6275	-5028	-3580	-4305.5	-4281.0
	n^2	-6275	-5278	-4530	-5035.9	-5028.0



Topology of interconnections between 15 qubits

	#var.	#qubits (Chim.)	#qubits (Peg.)
G_1	8	16	8
G_2	27	100	46
G_3	64	431	164
G_4	125	958	513

- Results on D-Wave **competitive** with those of SA.
- Duplicate instances are **much more difficult** to resolve than non-duplicate instances on SA.
- The denser topology of Pegasus leads to **smaller duplicated** QUBOs and gives better (but not optimal) results.

Conclusion and perspectives ?

- **Difficult problem for simulated and quantum annealing**
  First benchmark on a D-Wave
  With topology → Similar results between D-Wave and SA.

  The constraints imposed by the graph show that SA and QA cannot solve the problem!

- **Low quality results**
  - The need to duplicate qubits strongly limits the size of accessible problems.
  - Necessity of post-treatments
    Duplication errors ☹
    not representing a valid solution

- **Sparse topology**
  - Need more connections between qubits → mapping denser and larger problems!

[ACM-CF] « ***On the limitations of the Chimera graph topology in using analog quantum computers*** » , in « *ACM International Conference on Computing Frontiers* » 2019, Alghero, Sardinia, Italy.

[ICCS] « **Revisiting old combinatorial beasts in the quantum age : quantum annealing versus maximal matching** », in « INTERNATIONAL CONFERENCE ON COMPUTATIONAL SCIENCE » (ICCS 2020 - Quantum Computing Workshop).

[ISVLSI] « **Operational Quantum Annealers are Cursed by Their Qubits InterconnectionTopologies** », in « 1st International Workshop on Quantum Computing : Circuits Systems Automation and Applications » (QC-CSAA) (ISVLSI2020 - Quantum Workshop).

[SN] « **Benchmarking quantum annealing against ”hard” instances of the bipartite matching problem** », in Special issue for Springer journals (SN Computer Science).