Solving Planning and Scheduling Problems w/ Quantum Annealers: Status and Challenges

Davide Venturelli

Quantum AI Laboratory (QuAIL),
Research Institute for Advanced Computer Science (RIACS)
Universities Space Research Association (USRA)
davide.venturelli@nasa.gov

Collaborators:

E. Rieffel, B. O'Gorman, Z. Wang, J. Frank, M. Do, B. Pokharel (NASA)
D. Marchand (1Qbit)
I. Trummer (Cornell Univ.)
T. Tran (Univ. Toronto)
T. Stollenwerk (DLR)

Literature

- *A Hybrid Quantum-Classical Approach to Solving Scheduling Problems* . Tony T. Tran, Minh Do, Eleanor Rieffel, Jeremy Frank, Zhihui Wang, Bryan O'Gorman, Davide Venturelli and Chris Beck. In Symposium on Combinatorial Search (SoCS-16), 2016.

- *Explorations of Quantum-Classical Approaches to Scheduling a Mars Lander Activity Problem* . Tony T. Tran, Zhihui Wang, Minh Do, Eleanor G. Rieffel, Jeremy Frank, Bryan O'Gorman, Davide Venturelli, and J. Christopher Beck. In AAAI-16 Workshop on Planning for Hybrid Systems.

- *Job Shop Scheduling Solver based on Quantum Annealing.* Davide Venturelli, Dominic Marchand, Galo Rojo. In ICAPS-16 workshop Constraint Satisfaction Techniques for Planning and Scheduling (COPLAS-16)

- *A case study in programming a quantum annealer for hard operational planning problems* Eleanor Rieffel, Davide Venturelli, Bryan O'Gorman, Minh B. Do, Elicia Pristay, Vadim Smelyanskiy. Quantum Inf Process (2015) 14: 1

Upcoming on the arXiv

- T. Tran, DV et al. (2016)
- B. Pokharel, E. Rieffel, DV et. al (2016)
- I. Trummer, DV et. al (2016)

Bonus: Quantum Computing Background

Universal Quantum Computing (Gate Model)

- ~30 years of theoretical research
- ~20 years of experimental research

+ Quadratic speedup in database search (Grover search)

+ Exponential speedup in cryptanalysis (Shor's factoring)

+ **Killer app: Quantum Simulations**

- Around 10 qubits working across technologies

- ~1M physical qubits required for real world applications

- 15+ years before fully integrated system

Bonus: Quantum Computing Background

Universal Quantum Computing (Gate Model)

- ~30 years of theoretical research
- ~20 years of experimental research

+ Quadratic speedup in database search (Grover search)

+ Exponential speedup in cryptanalysis (Shor’s factoring)

+ **Killer app: Quantum Simulations**

- Around 10 qubits working across technologies

- ~1M physical qubits required for real world applications

- 15+ years before fully integrated system

Quantum Optimization (Annealing)

- ~15 years of theoretical research
- ~7-8 years experiments

+ **General approach for all combinatorial optimization problems**

+ Other groups are creating machines (Google, MIT Lincoln Lab.)

+ 1000+ qubit processors available

+ ~10K physical qubits required for useful problems

- Speedup and effect of noise/temperature largely unknown

Bonus: Quantum Computing Background

Universal Quantum Computing (Gate Model)

- ~30 years of theoretical research
- ~20 years of experimental research

+ Quadratic speedup in database search (Grover search)

+ Exponential speedup in cryptanalysis (Shor's factoring)

+ **Killer app: Quantum Simulations**

- Around 10 qubits working across technologies

- ~1M physical qubits required for real world applications

- 15+ years before fully integrated system

Quantum Optimization (Annealing)

- ~15 years of theoretical research
- ~7-8 years experiments

+ **General approach for all combinatorial optimization problems**

+ Other groups are creating machines (Google, MIT Lincoln Lab.)

+ 1000+ qubit processors available

+ ~10K physical qubits required for useful problems

- Speedup and effect of noise/temperature largely unknown

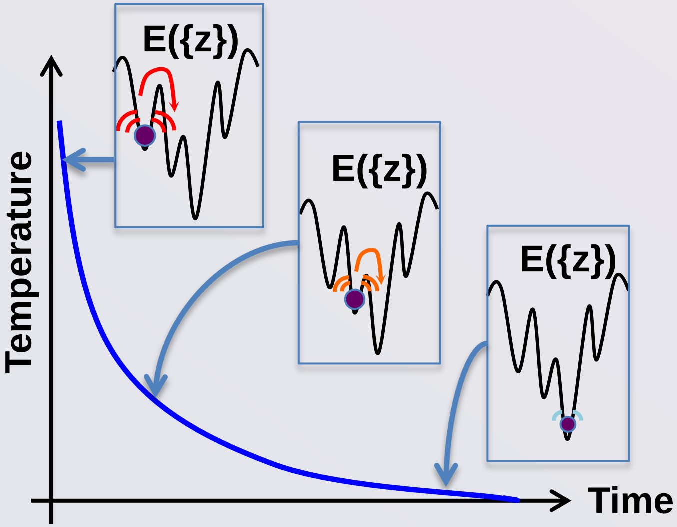
$$\boxed{A\ 2^{N} \Rightarrow B\ e^{N^{C}}}$$

Best hope!

Intro: Simulated VS Quantum Annealing

Simulated Annealing

(Kirkpatrick et al., 1983)

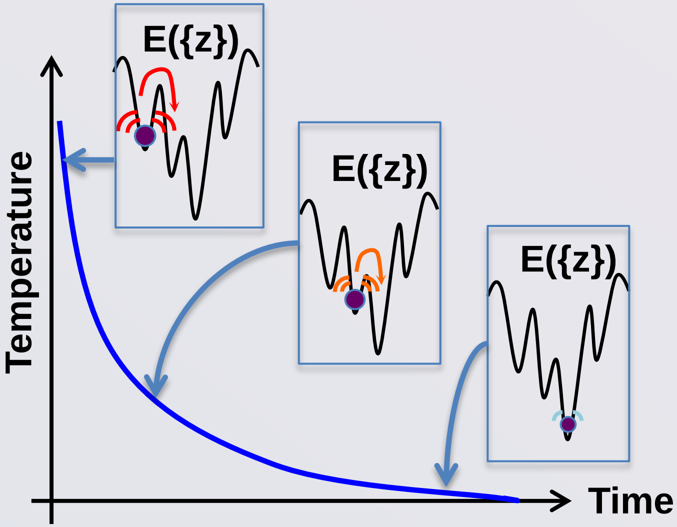


Bit flips activated by temperature

Quantum Annealing in a nutshell: D-Wave 2X

Simulated Annealing

(Kirkpatrick et al., 1983)



Bit flips activated by temperature

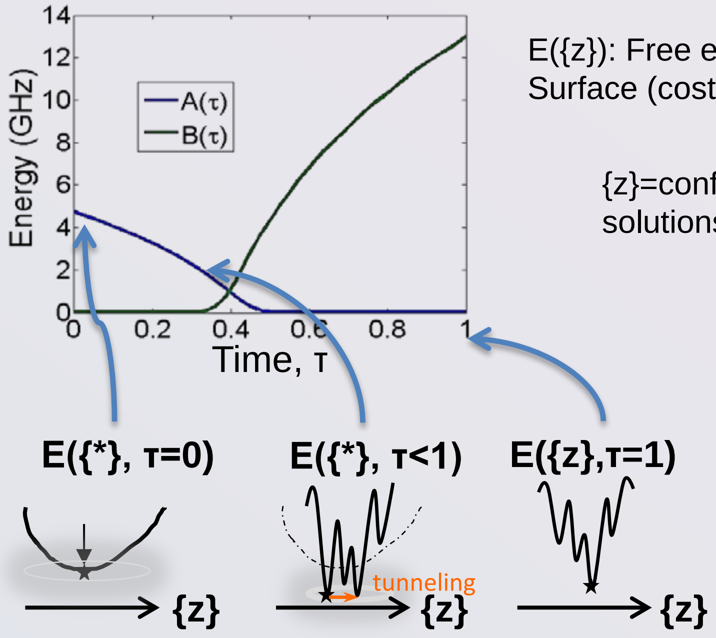
3 Key differences:

1) Superposition of bit-strings (tunneling)
2) Energy landscape changes over time
3) Equilibration and Adiabatic Theorem

Quantum Annealing in a nutshell: D-Wave 2X

Quantum Annealing

(Finnila et al. 1994, Kadawaki&Nishimori 1998, Farhi et.al. 2001)



E({z}): Free energy
Surface (cost funct.)

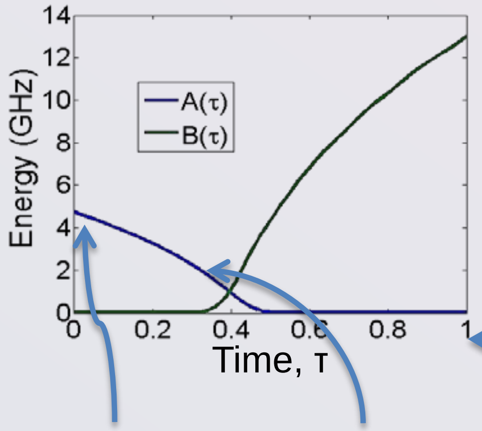
{z}=configurations in
solutions space

Bit flips activated by tunneling

Quantum Annealing in a nutshell: D-Wave 2X

Quantum Annealing

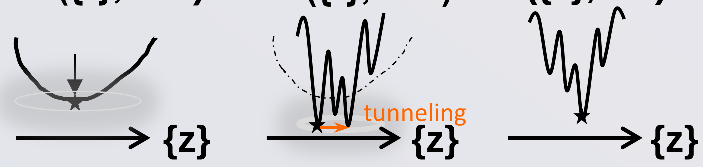
(Finnila et al. 1994, Kadawaki&Nishimori 1998, Farhi et.al. 2001)



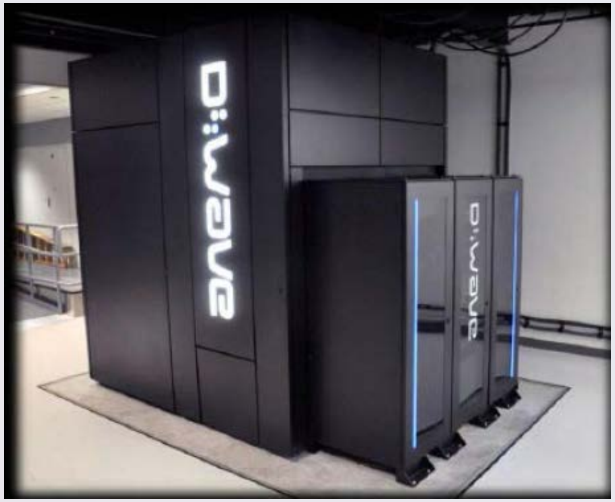
E({z}): Free energy Surface (cost funct.)

{z}=configurations in solutions space

E({*}, τ=0) **E({*}, τ<1)** **E({z},τ=1)**



Bit flips activated by tunneling

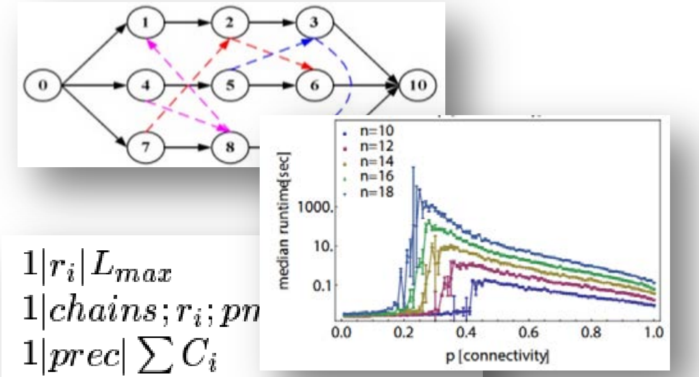


- ❑ Not adiabatic:
  - ❑ “Strong” noise
  - ❑ “High” temperature (12mK)
- ❑ Only a single annealing protocol
  - ❑ “Slow” speed (5μs)

Theory VS Real World

Paradigmatic Theory of Scheduling Problems

- Truly random ensembles
- Known mappings and “phase transitions”
- Solid classical algorithmics and literature
- “Easy” parametrization



$1|r_i|L_{max}$

$1|chains; r_i; pn$

$1|prec|\sum C_i$

$1|r_i|\sum C_i$

$1|chains; p_i = 1; r_i|\sum w_i C_i$

$1|prec; p_i = 1|\sum w_i C_i$

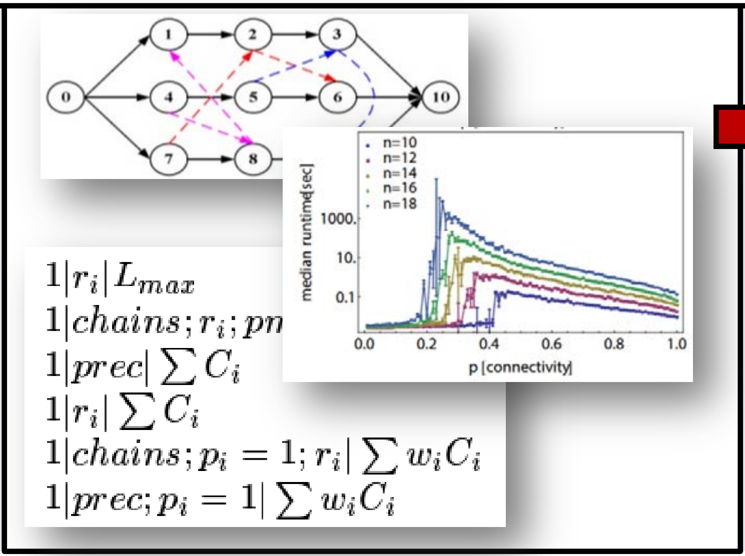
Theory VS Real World

Paradigmatic Theory of Scheduling Problems

- Truly random ensembles
- Known mappings and “phase transitions”
- Solid classical algorithmics and literature
- “Easy” parametrization

Real world scheduling problems

- Correlated, not random
- Hardness is very much instance dependent
- Classical approaches are ad-hoc heuristics
- Can feature convoluted structure



$1|r_i|L_{max}$

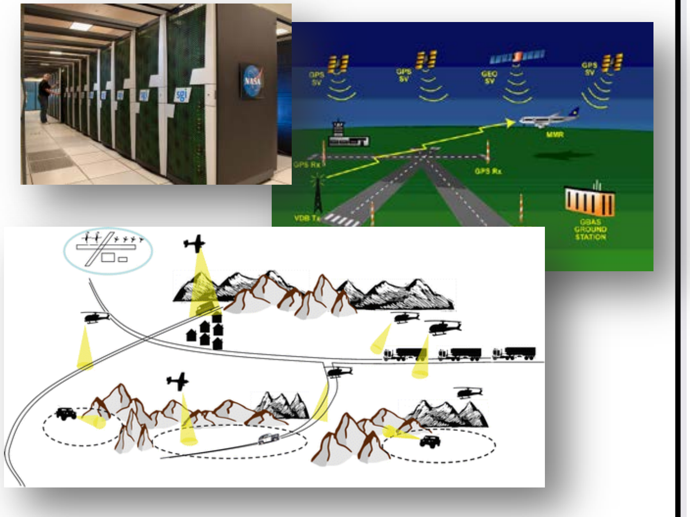
$1|chains;r_i;pm$

$1|prec|\sum C_i$

$1|r_i|\sum C_i$

$1|chains;p_i=1;r_i|\sum w_iC_i$

$1|prec;p_i=1|\sum w_iC_i$



The basics of Scheduling

Machine Environment

Shared Resources with finite capacities:
Regions of Space, Regions of time, Shared Equipment..

Job Characteristics

Processing times, ordering, Batching, due dates, validity windows …

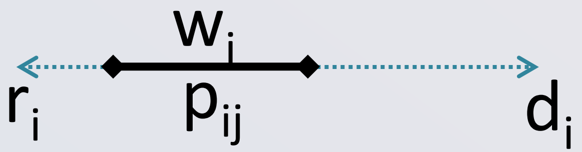
Objective Function

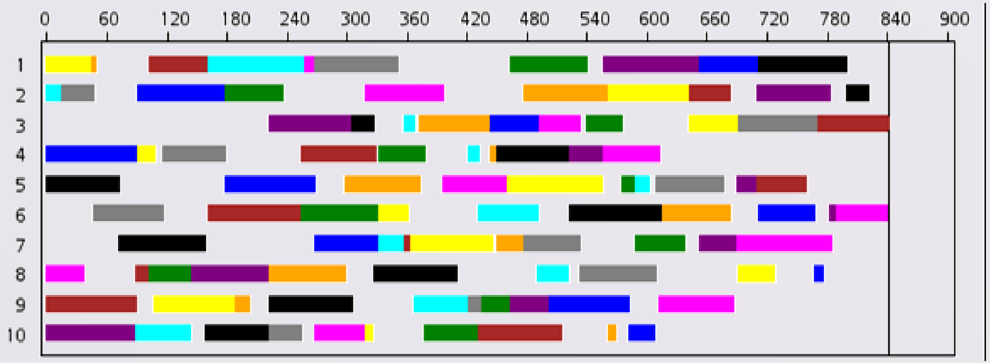
Metric that determines best solutions:
Minimize total time, Maximize total priority, Maximize total utilizations

Example: $R10 \mid p_{ij}=[0,\dots,\tau], r_j, d_j \mid \sum_i w_i U_i$

Example of notation for Alternative Resource Scheduling

Graphical Representation of a schedule and of a problem:

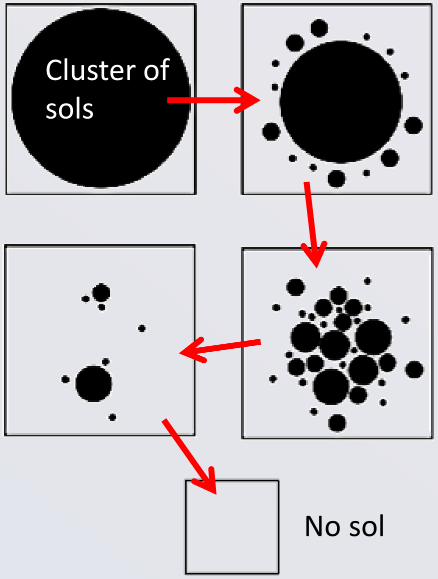




Scheduling Benchmarks

Phase Transitions

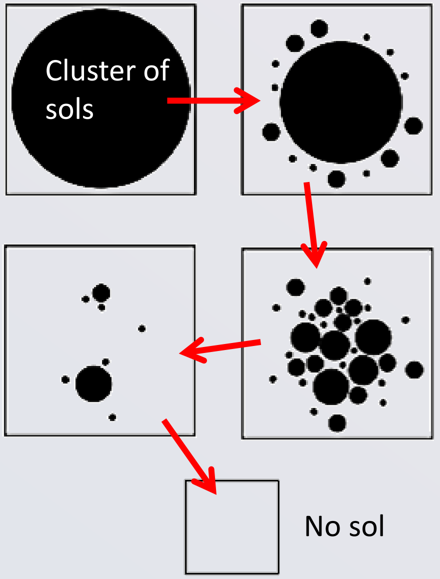
What parameters make instances truly hard?



Scheduling Benchmarks

Phase Transitions

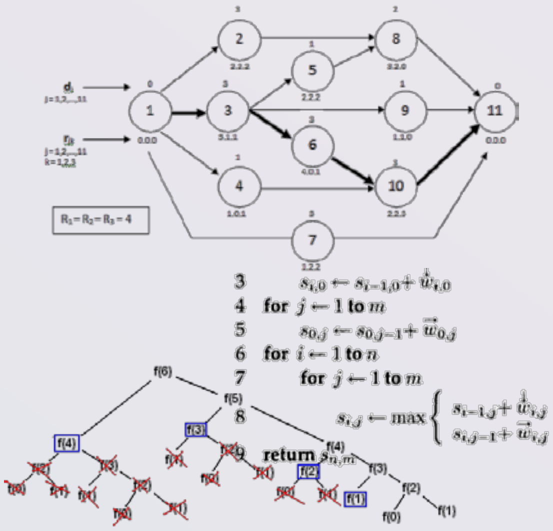
What parameters make instances truly hard?



Tailored Algorithms

What is the best possible known way to solve these hard instances?

Figure 1. Graphic Example of a Project with Resource Constraints

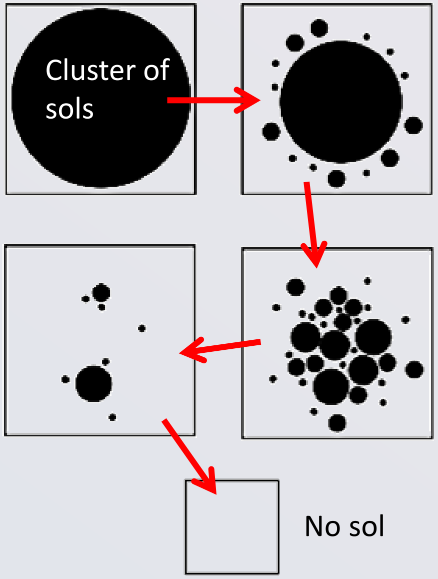


```
        $s_{i,0} \leftarrow s_{i-1,0} + \overset{\downarrow}{w}_{i,0}$
for $j \leftarrow 1$ to $m$
        $s_{0,j} \leftarrow s_{0,j-1} + \vec{w}_{0,j}$
for $i \leftarrow 1$ to $n$
        for $j \leftarrow 1$ to $m$
                $s_{i,j} \leftarrow \max \begin{cases} s_{i-1,j} + \overset{\downarrow}{w}_{i,j} \\ s_{i,j-1} + \vec{w}_{i,j} \end{cases}$
return $s_{n,m}$
```

Scheduling Benchmarks

Phase Transitions

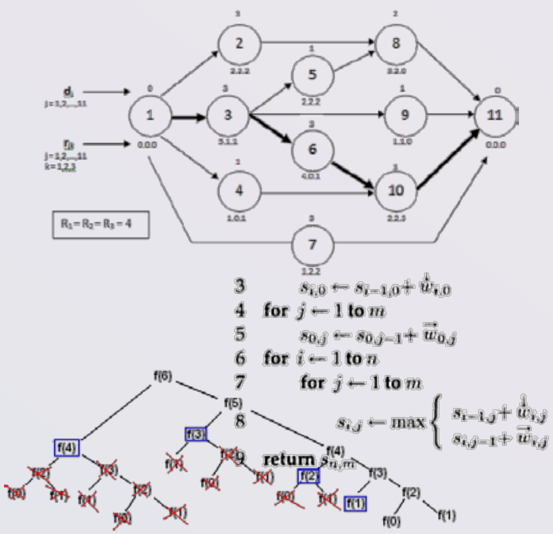
What parameters make instances truly hard?



Tailored Algorithms

What is the best possible known way to solve these hard instances?

Figure 1. Graphic Example of a Project with Resource Constraints



```
        $s_{i,0} \leftarrow s_{i-1,0} + \overset{\downarrow}{w}_{i,0}$
for $j \leftarrow 1$ to $m$
        $s_{0,j} \leftarrow s_{0,j-1} + \vec{w}_{0,j}$
for $i \leftarrow 1$ to $n$
        for $j \leftarrow 1$ to $m$
                $s_{i,j} \leftarrow \max \begin{cases} s_{i-1,j} + \overset{\downarrow}{w}_{i,j} \\ s_{i,j-1} + \vec{w}_{i,j} \end{cases}$
return $s_{n,m}$
```

Commercial Solvers

What is the current way to solve these instances?

Gurobi Optimization

IBM ILOG CPLEX

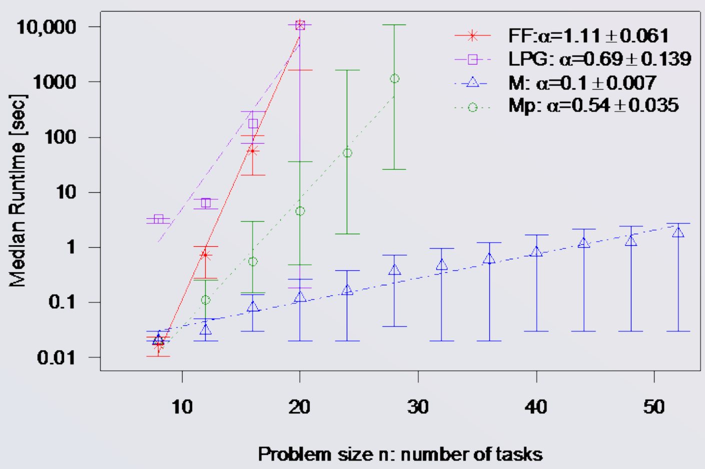
SCIP

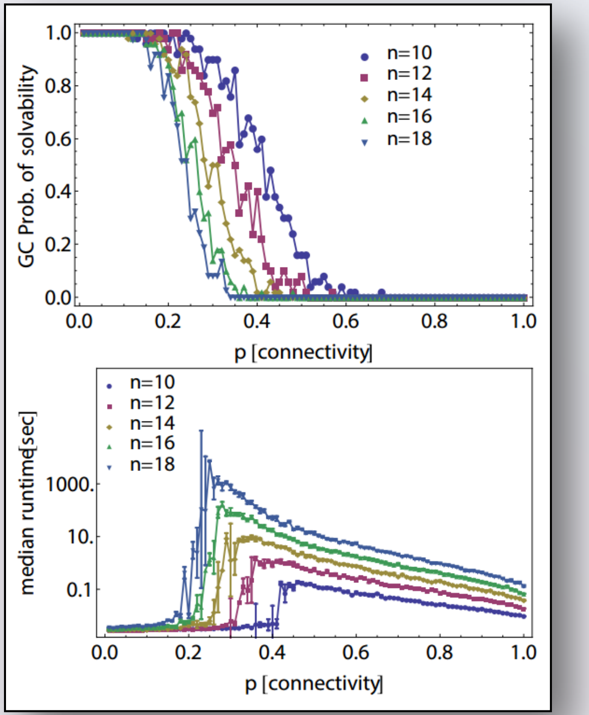
Solving Constraint Integer Programs

Phase Transitions in Combinatorial Problems

- Parametrize an ensemble of instances
- Find an “easy-hard-easy” pattern
- Check for exponential scaling in N

Planner Comparison: All Scheduling Problems





(Rieffel, Venturelli, Do, Hen, Frank 2013)

See Taillard Instances, standard benchmarks, found in OR library

Commercial solvers

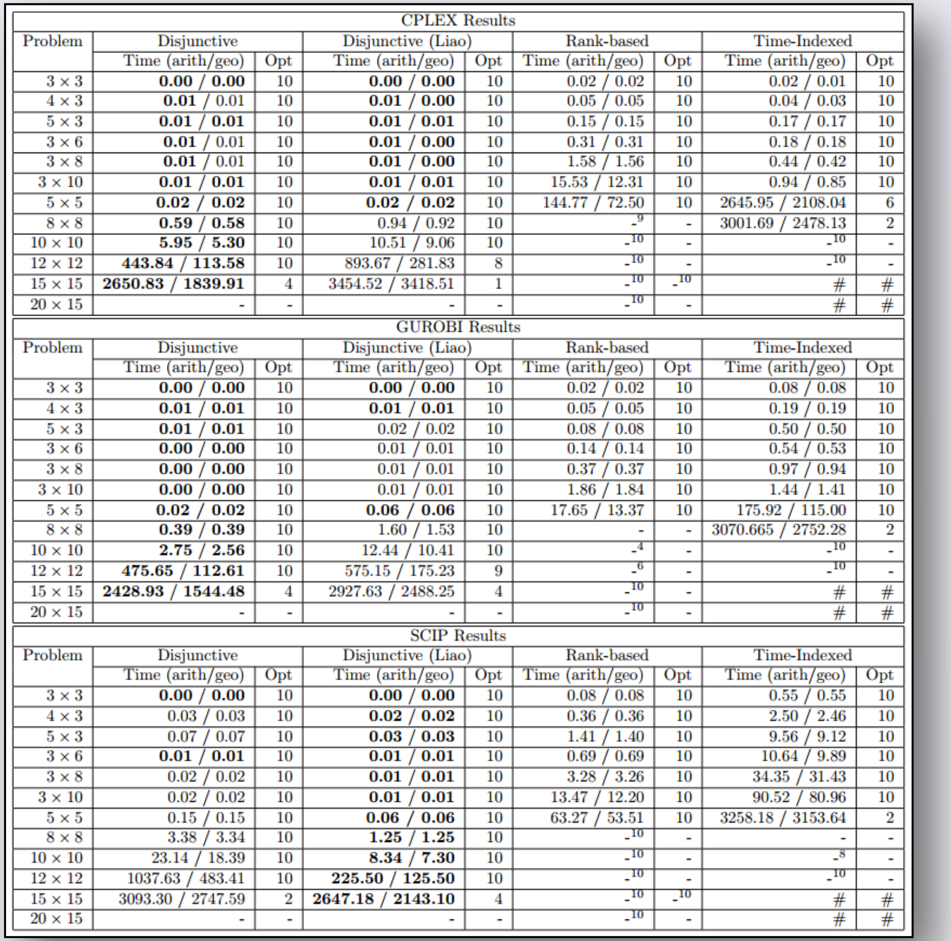
W. Ku and J. Beck, technical report, Univ. of Toronto (2014).

Commercial Solvers needs to be properly tuned to take advantage of parallelism and most recent features.

Dash, S. (2013). A note on QUBO instances defined on Chimera graphs.arXiv preprint arXiv:1306.1202.

(D-Wave was benchmarked ≈20x faster than what it was possible)

Other example: for diagnostics we used HyDE… Programs of Xerox PARC



CPLEX Results

Problem	Disjunctive		Disjunctive (Liao)		Rank-based		Time-Indexed	
	Time (arith/geo)	Opt	Time (arith/geo)	Opt	Time (arith/geo)	Opt	Time (arith/geo)	Opt
3 × 3	**0.00 / 0.00**	10	**0.00 / 0.00**	10	0.02 / 0.02	10	0.02 / 0.01	10
4 × 3	**0.01** / 0.01	10	**0.01 / 0.00**	10	0.05 / 0.05	10	0.04 / 0.03	10
5 × 3	**0.01 / 0.01**	10	**0.01 / 0.01**	10	0.15 / 0.15	10	0.17 / 0.17	10
3 × 6	**0.01** / 0.01	10	**0.01 / 0.00**	10	0.31 / 0.31	10	0.18 / 0.18	10
3 × 8	**0.01** / 0.01	10	**0.01 / 0.00**	10	1.58 / 1.56	10	0.44 / 0.42	10
3 × 10	**0.01 / 0.01**	10	**0.01 / 0.01**	10	15.53 / 12.31	10	0.94 / 0.85	10
5 × 5	**0.02 / 0.02**	10	**0.02 / 0.02**	10	144.77 / 72.50	10	2645.95 / 2108.04	6
8 × 8	**0.59 / 0.58**	10	0.94 / 0.92	10	-^{9}	-	3001.69 / 2478.13	2
10 × 10	**5.95 / 5.30**	10	10.51 / 9.06	10	-^{10}	-	-^{10}	-
12 × 12	**443.84 / 113.58**	10	893.67 / 281.83	8	-^{10}	-	-^{10}	-
15 × 15	**2650.83 / 1839.91**	4	3454.52 / 3418.51	1	-^{10}	-^{10}	#	#
20 × 15	-	-	-	-	-^{10}	-	#	#

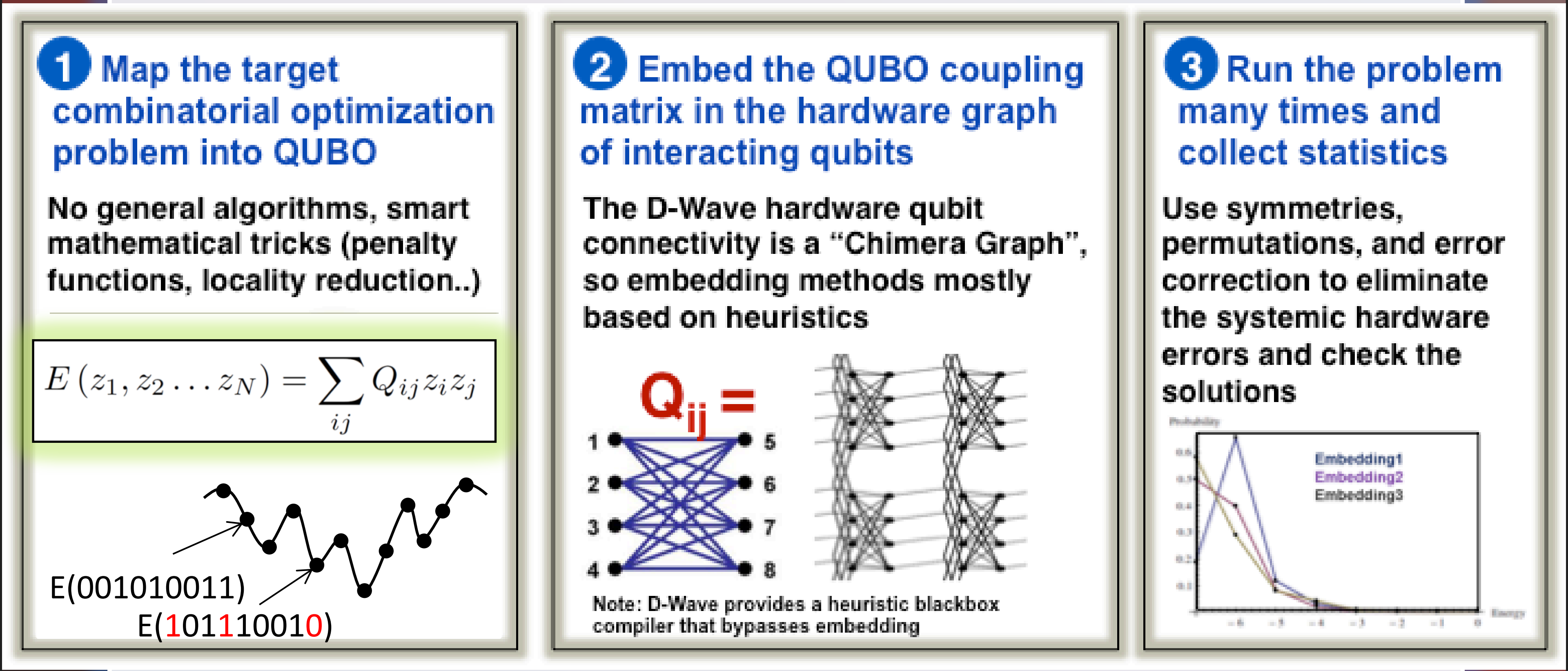
GUROBI Results

Problem	Disjunctive		Disjunctive (Liao)		Rank-based		Time-Indexed	
	Time (arith/geo)	Opt	Time (arith/geo)	Opt	Time (arith/geo)	Opt	Time (arith/geo)	Opt
3 × 3	**0.00 / 0.00**	10	**0.00 / 0.00**	10	0.02 / 0.02	10	0.08 / 0.08	10
4 × 3	**0.01 / 0.01**	10	**0.01 / 0.01**	10	0.05 / 0.05	10	0.19 / 0.19	10
5 × 3	**0.01 / 0.01**	10	0.02 / 0.02	10	0.08 / 0.08	10	0.50 / 0.50	10
3 × 6	**0.00 / 0.00**	10	0.01 / 0.01	10	0.14 / 0.14	10	0.54 / 0.53	10
3 × 8	**0.00 / 0.00**	10	0.01 / 0.01	10	0.37 / 0.37	10	0.97 / 0.94	10
3 × 10	**0.00 / 0.00**	10	0.01 / 0.01	10	1.86 / 1.84	10	1.44 / 1.41	10
5 × 5	**0.02 / 0.02**	10	**0.06 / 0.06**	10	17.65 / 13.37	10	175.92 / 115.00	10
8 × 8	**0.39 / 0.39**	10	1.60 / 1.53	10	-	-	3070.665 / 2752.28	2
10 × 10	**2.75 / 2.56**	10	12.44 / 10.41	10	-^{4}	-	-^{10}	-
12 × 12	**475.65 / 112.61**	10	575.15 / 175.23	9	-^{6}	-	-^{10}	-
15 × 15	**2428.93 / 1544.48**	4	2927.63 / 2488.25	4	-^{10}	-	#	#
20 × 15	-	-	-	-	-^{10}	-	#	#

SCIP Results

Problem	Disjunctive		Disjunctive (Liao)		Rank-based		Time-Indexed	
	Time (arith/geo)	Opt	Time (arith/geo)	Opt	Time (arith/geo)	Opt	Time (arith/geo)	Opt
3 × 3	**0.00 / 0.00**	10	**0.00 / 0.00**	10	0.08 / 0.08	10	0.55 / 0.55	10
4 × 3	0.03 / 0.03	10	**0.02 / 0.02**	10	0.36 / 0.36	10	2.50 / 2.46	10
5 × 3	0.07 / 0.07	10	**0.03 / 0.03**	10	1.41 / 1.40	10	9.56 / 9.12	10
3 × 6	**0.01 / 0.01**	10	**0.01 / 0.01**	10	0.69 / 0.69	10	10.64 / 9.89	10
3 × 8	0.02 / 0.02	10	**0.01 / 0.01**	10	3.28 / 3.26	10	34.35 / 31.43	10
3 × 10	0.02 / 0.02	10	**0.01 / 0.01**	10	13.47 / 12.20	10	90.52 / 80.96	10
5 × 5	0.15 / 0.15	10	**0.06 / 0.06**	10	63.27 / 53.51	10	3258.18 / 3153.64	2
8 × 8	3.38 / 3.34	10	**1.25 / 1.25**	10	-^{10}	-	-	-
10 × 10	23.14 / 18.39	10	**8.34 / 7.30**	10	-^{10}	-	-^{8}	-
12 × 12	1037.63 / 483.41	10	**225.50 / 125.50**	10	-^{10}	-	-^{10}	-
15 × 15	3093.30 / 2747.59	2	**2647.18 / 2143.10**	4	-^{10}	-^{10}	#	#
20 × 15	-	-	-	-	-^{10}	-	#	#

Programming Steps



1 Map the target combinatorial optimization problem into QUBO

No general algorithms, smart mathematical tricks (penalty functions, locality reduction..)

$$E(z_1, z_2 \dots z_N) = \sum_{ij} Q_{ij} z_i z_j$$

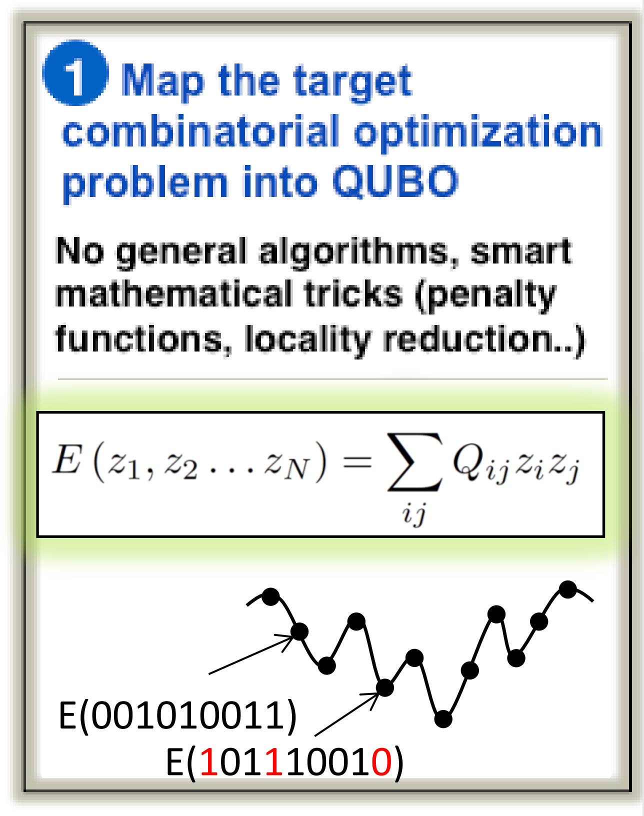
2 Embed the QUBO coupling matrix in the hardware graph of interacting qubits

The D-Wave hardware qubit connectivity is a "Chimera Graph", so embedding methods mostly based on heuristics

Note: D-Wave provides a heuristic blackbox compiler that bypasses embedding

3 Run the problem many times and collect statistics

Use symmetries, permutations, and error correction to eliminate the systemic hardware errors and check the solutions



1 Map the target combinatorial optimization problem into QUBO

No general algorithms, smart mathematical tricks (penalty functions, locality reduction..)

$$E\left(z_1, z_2 \dots z_N\right) = \sum_{ij} Q_{ij} z_i z_j$$

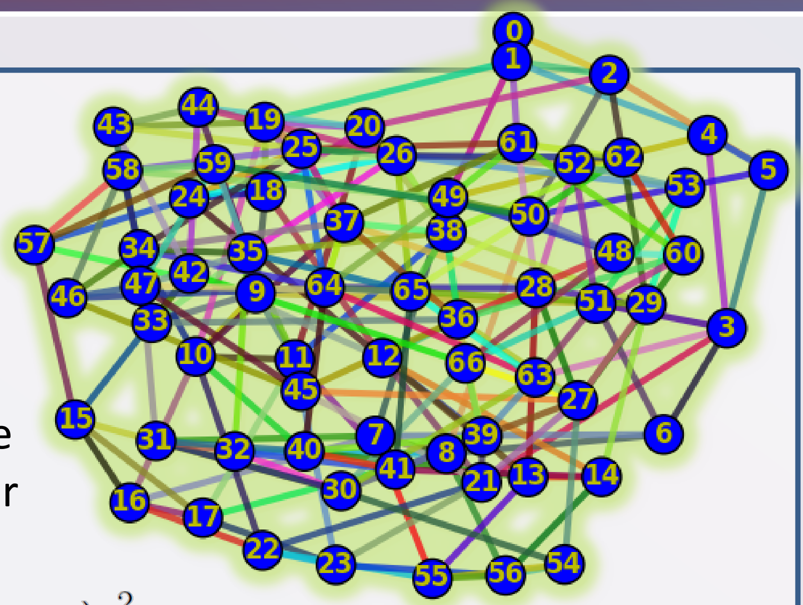
E(001010011)

E(101110010)

Pre-processing, QUBO mapping, decomposition

- Coloring
- Single Machine Scheduling → Complete Tree search
- Multiple Machines → LBBD
- Job-Shop → Decision/ Opt decomp
- Other scheduling
- Other hybrid approaches

Example 0: Graph Coloring



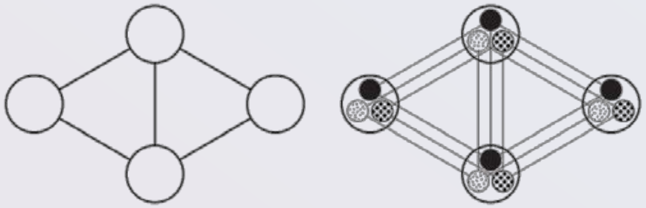
Only one Color per node:

$$\left(1-\sum_{c} x_{ic}\right)^{2}$$

Edge constraints (an edge cannot connect the same colors):

$$\sum_{(i,j)\in E(i)} \sum_{c} x_{ic}x_{jc}$$

Mapping the problem takes only **3N**



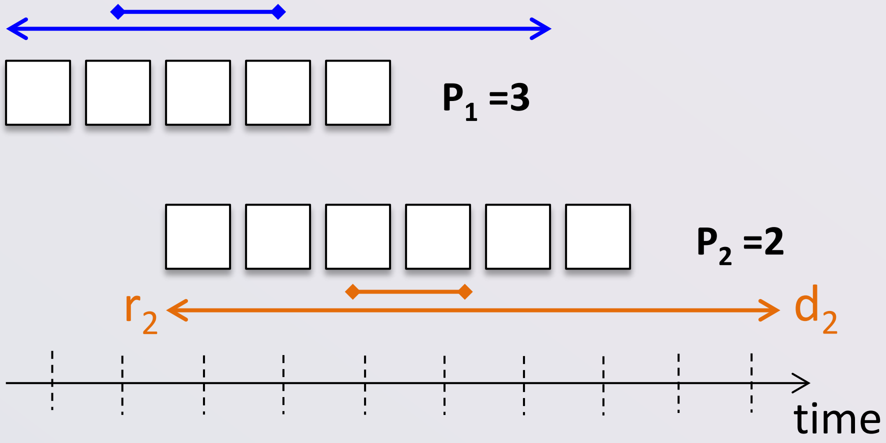
From beigel Eppstein (2000)

HARD CONSTRAINT

SOFT CONSTRAINT

Single-Machine Scheduling

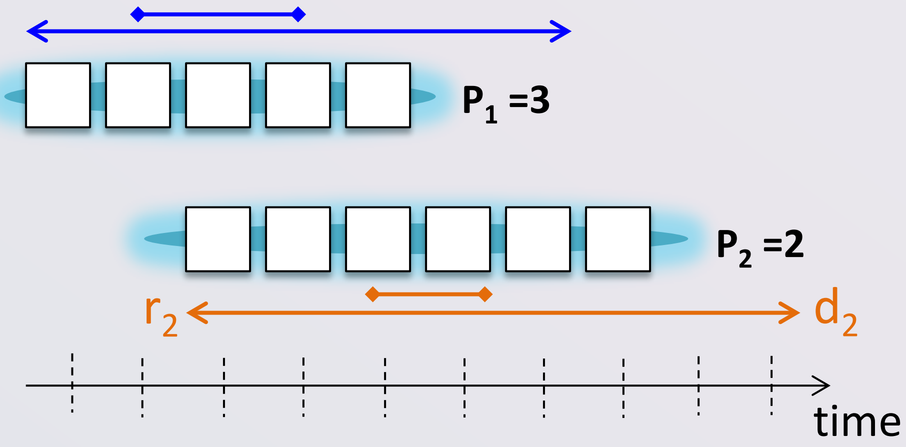
Time-Indexed Formulation: X_{it}=1 if job executed at time t or =0 otherwise



Only the starting points are represented by a bit.

Single-Machine Scheduling

Time-Indexed Formulation: X_{it}=1 if job executed at time t or =0 otherwise



Only the starting points are represented by a bit.

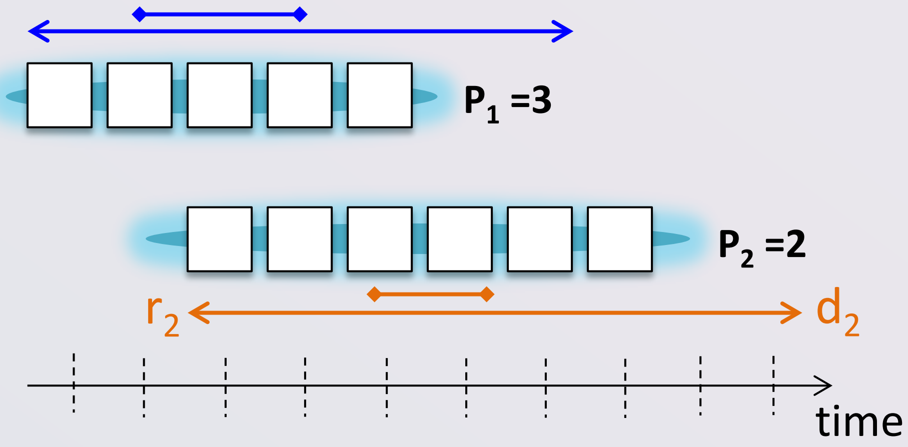
This generates fully-connected cliques: see embedding talk.

Jobs needs to be scheduled only once:

$$\Delta H_a = \Sigma_i (\Sigma_t x_{it} - 1)^2$$

Single-Machine Scheduling

Time-Indexed Formulation: X_{it}=1 if job executed at time t or =0 otherwise



Only the starting points are represented by a bit.

This generates fully-connected cliques: see embedding talk.

Jobs needs to be scheduled only once:

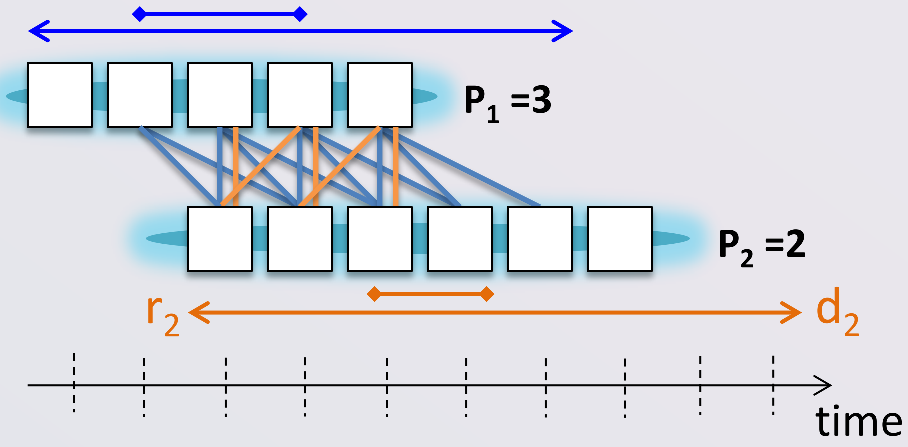
$$\Delta H_a = \Sigma_i \left(\Sigma_t x_{it} - 1\right)^2$$

$x_{it} = ½(s_{it}+1)$

$$\Delta H_a = \Sigma_i ½\Sigma_{tt'} s_{it}\, s_{it'} \;+ \ldots$$

Single-Machine Scheduling

Time-Indexed Formulation: X_{it}=1 if job executed at time t or =0 otherwise



Specific Job-dependent “setup times” can be trivially added the same way.

Jobs needs avoid conflict, considering the processing times:

$$\Delta H_b = ½ \Sigma_{it} \Sigma_{j\neq i} (\Sigma_\tau s_{it} s_{j(t+\tau)}) + \ldots$$

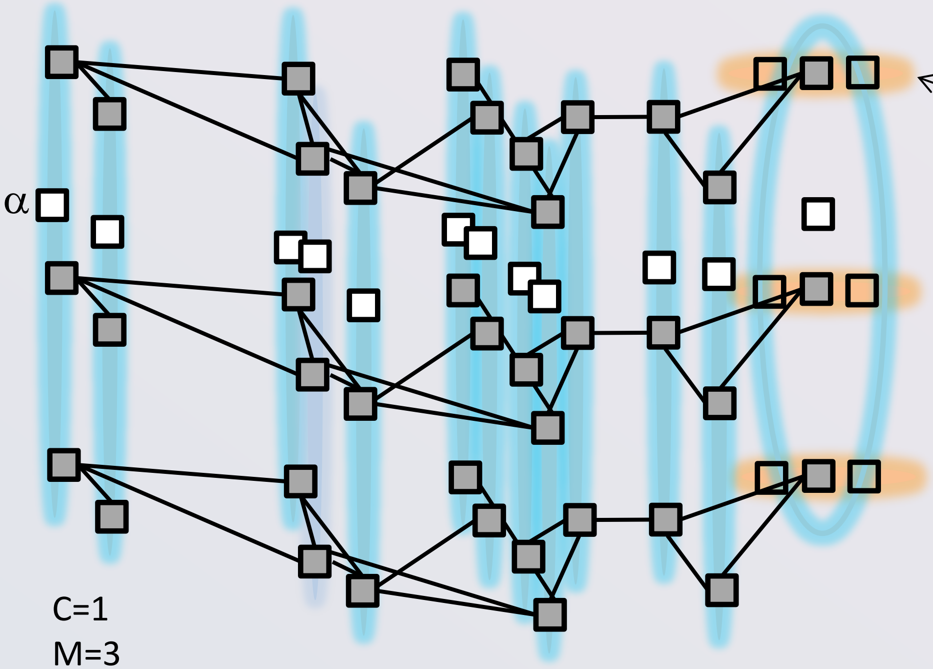
Single-Machine Scheduling

$X_{it} \longrightarrow X_{itm}$ (introducing the machine index)

$$\Delta H_d = -\sum_{it} \left(\sum_m x_{itm} + \alpha - C\right)^2$$

α is a “slack” variable.

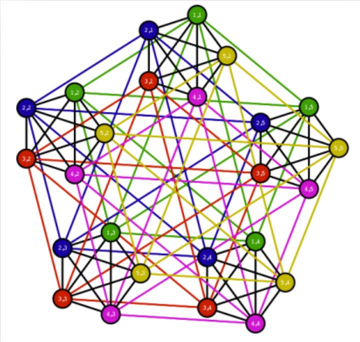
$\Delta H_d > 0$ if $\sum_m x_{itm} > C$



Intersecting Cliques

C=1

M=3



Resource Requirement Scaling

Naturally quadratic fomulation:

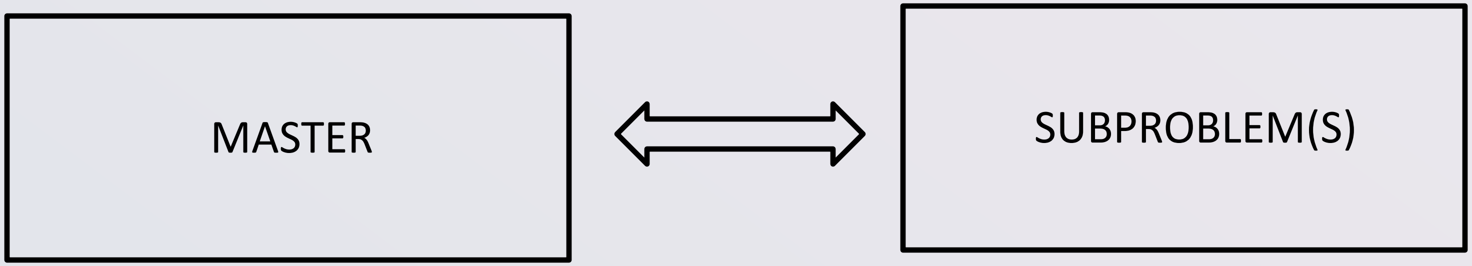
- Typically required **N*M*L** qubits, with **L=[d_i-r_i]** before pre-processing.
- **N*M** cliques of **L** size, intersecting **N*L** cliques of size ≈**M**
- Each δτ overlap of **R** tasks also generates cliques of size ≈**R**δτ
- Reset times just add connections *(consider all N(N-1)/2 pairs)*
- Capacities introduce ancilla slack qubits and possible precision requirements.

Resource Requirement Scaling

Naturally quadratic fomulation:

- Typically required **N*M*L** qubits, with **$L=[d_i-r_i]$** before pre-processing.
- **N*M** cliques of **L** size, intersecting **N*L** cliques of size $\approx$**M**
- Each $\delta\tau$ overlap of **R** tasks also generates cliques of size $\approx$**R**$\delta\tau$
- Reset times just add connections *(consider all N(N-1)/2 pairs)*
- Capacities introduce ancilla slack qubits and possible precision requirements.

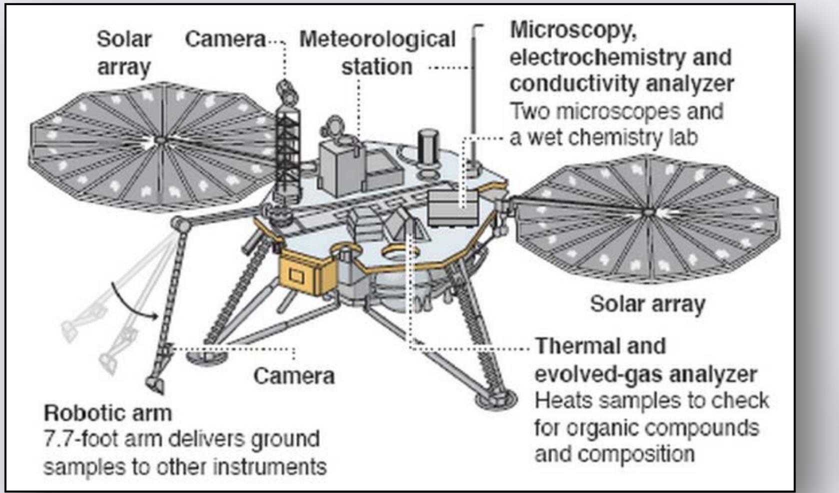
Pre-processing and decompose



MASTER

SUBPROBLEM(S)

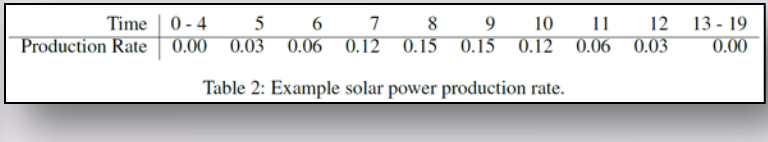
Example 1: Mars Lander Scheduling


Solar array
Camera
Meteorological station
Microscopy, electrochemistry and conductivity analyzer
Two microscopes and a wet chemistry lab
Solar array
Thermal and evolved-gas analyzer
Heats samples to check for organic compounds and composition
Camera
Robotic arm
7.7-foot arm delivers ground samples to other instruments

(Tran, Wang, Do, Rieffel, Frank, O'Gorman, Venturelli, Beck 2015)

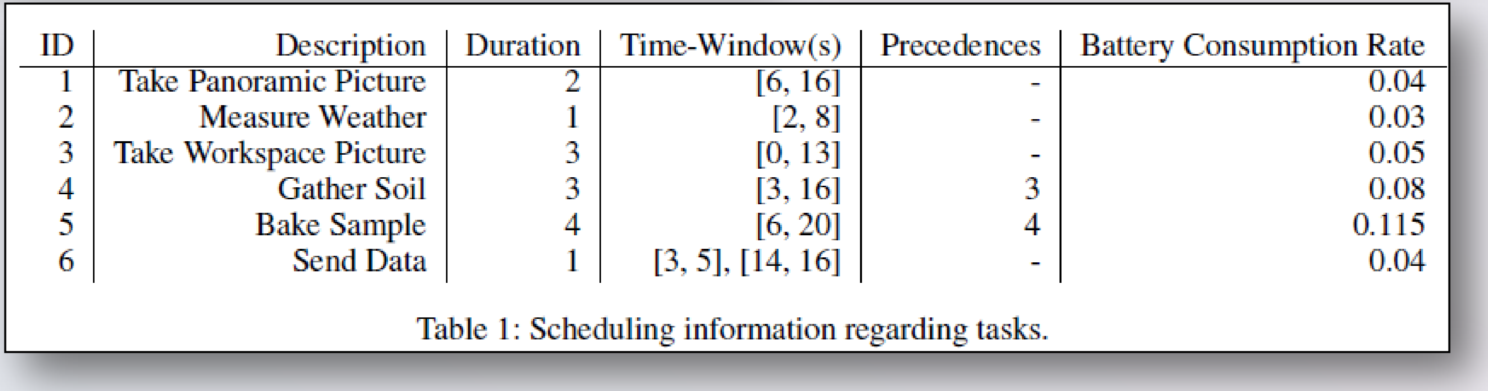
Instances

- Different initial battery levels
- Different battery capacity
- Different martian weather



Time	0 - 4	5	6	7	8	9	10	11	12	13 - 19
Production Rate	0.00	0.03	0.06	0.12	0.15	0.15	0.12	0.06	0.03	0.00

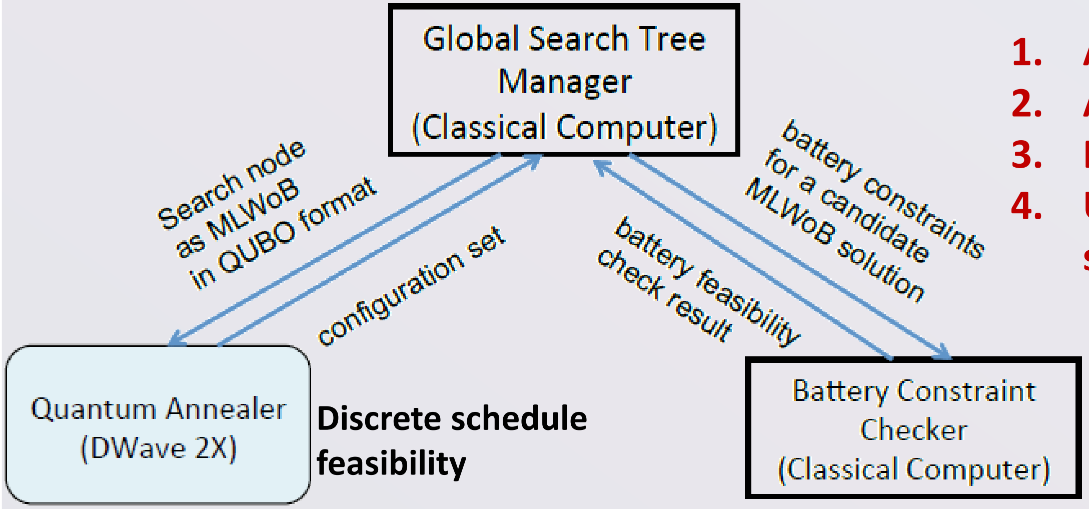
Table 2: Example solar power production rate.



ID	Description	Duration	Time-Window(s)	Precedences	Battery Consumption Rate
1	Take Panoramic Picture	2	[6, 16]	-	0.04
2	Measure Weather	1	[2, 8]	-	0.03
3	Take Workspace Picture	3	[0, 13]	-	0.05
4	Gather Soil	3	[3, 16]	3	0.08
5	Bake Sample	4	[6, 20]	4	0.115
6	Send Data	1	[3, 5], [14, 16]	-	0.04

Table 1: Scheduling information regarding tasks.

Decomposing the battery constraint



Global Search Tree Manager (Classical Computer)

Search node as MLWoB in QUBO format

configuration set

battery constraints for a candidate MLWoB solution

battery feasibility check result

Quantum Annealer (DWave 2X)

Discrete schedule feasibility

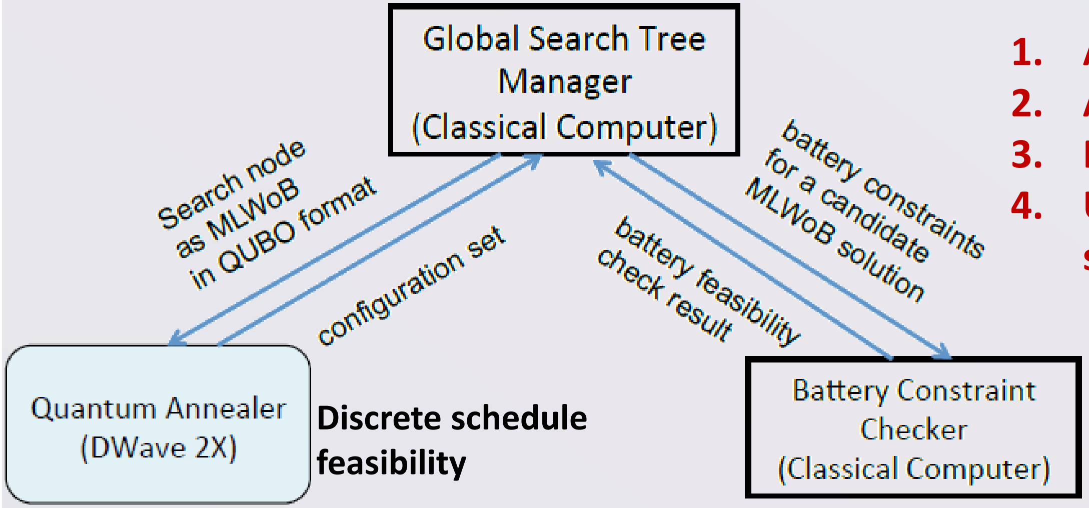
Battery Constraint Checker (Classical Computer)

Continuous Constraint

1. **Allows continuous constraints**
2. **Allows complex battery models**
3. **Reduces number of variables**
4. **Use the annealer in a complete search**

N_{bits}=52
N_{qubits}=764

Decomposing the battery constraint



Global Search Tree Manager (Classical Computer)

Search node as MLWoB in QUBO format

configuration set

battery constraints for a candidate MLWoB solution

battery feasibility check result

Quantum Annealer (DWave 2X)

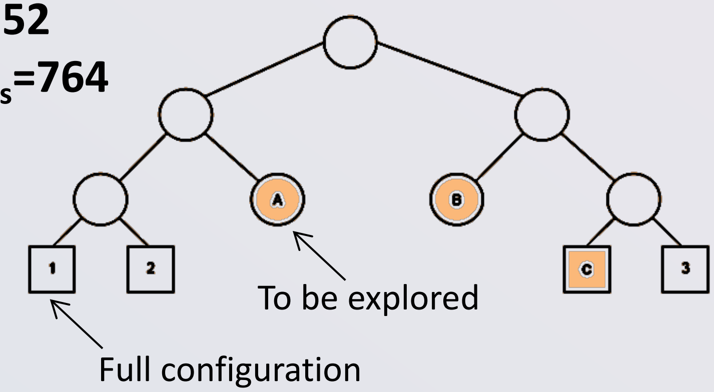
Discrete schedule feasibility

Battery Constraint Checker (Classical Computer)

Continuous Constraint

1. **Allows continuous constraints**
2. **Allows complex battery models**
3. **Reduces number of variables**
4. **Use the annealer in a complete search**

N_{bits}=52
N_{qubits}=764

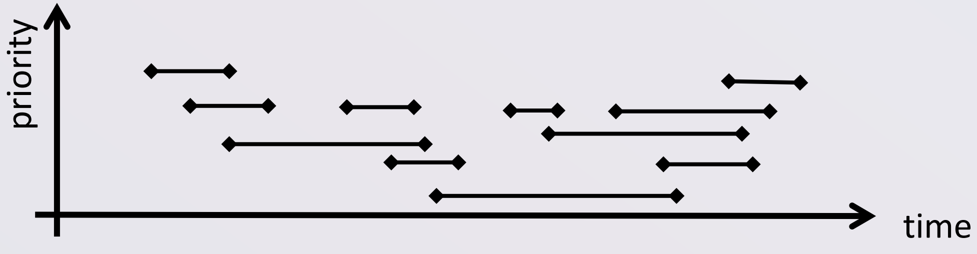


To be explored

Full configuration

1. Run Annealer K times
2. Check Battery constraint on E=0 solutions
3. Build a search tree and identify the non-returned solutions (ordering)
4. Prune and explore the tree branches with dedicated annealing runs

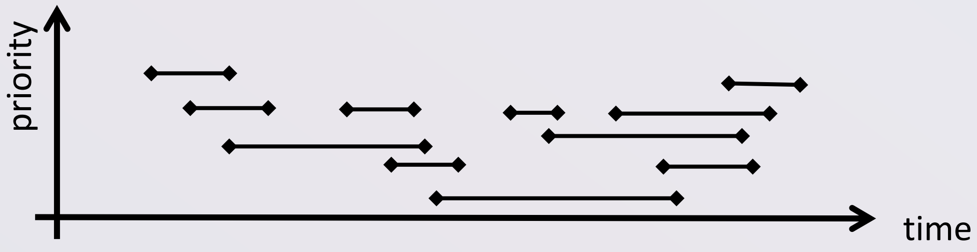
Example 2: Alternative Resource Scheduling



- M□ 1 Machines
- N □ 1 Jobs
- Overlapping windows $[r_j, d_j]$
- Machine-dependent processing times p_{mj}
- Machine-dependent execution cost c_{mj}

How to distribute the N jobs among the M machines to minimize the cost?

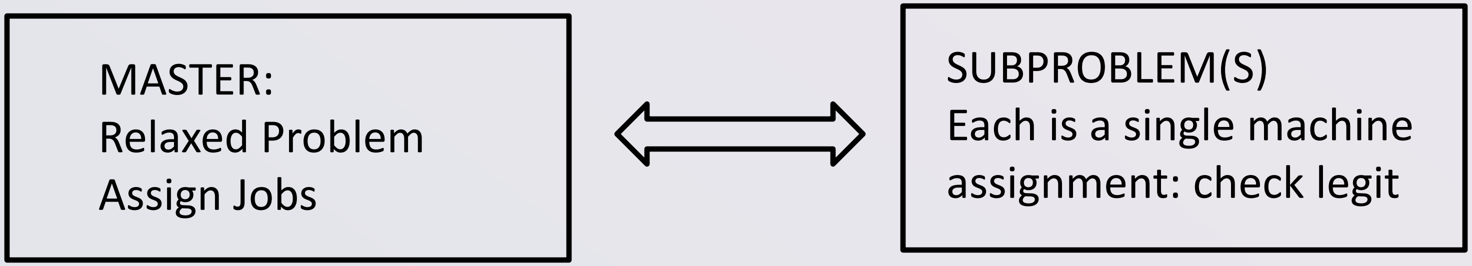
Example 2: Alternative Resource Scheduling



- M□ 1 Machines
- N □ 1 Jobs
- Overlapping windows $[r_j, d_j]$
- Machine-dependent processing times p_{mj}
- Machine-dependent execution cost c_{mj}

How to distribute the N jobs among the M machines to minimize the cost?

Pre-processing and decompose



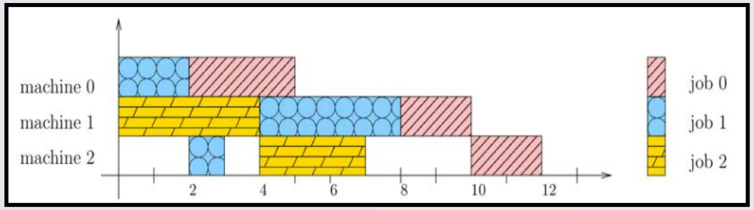
MASTER:
Relaxed Problem
Assign Jobs

SUBPROBLEM(S)
Each is a single machine assignment: check legit

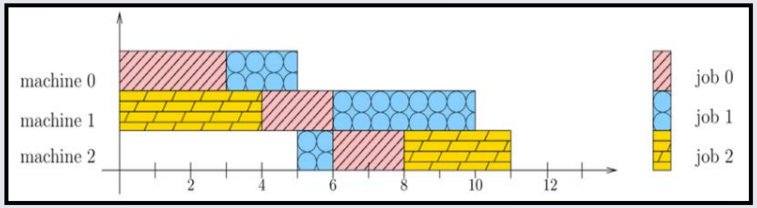
Example 3: Job-shop Scheduling

	1st operation	2nd operation	3rd operation
JOB 0	Machine 0 for 3t	Machine 1 for 2t	Machine 2 for 2t
JOB 1	Machine 0 for 2t	Machine 2 for 1t	Machine 1 for 4t
JOB 2	Machine 1 for 3t	Machine 2 for 3t	Machine 0 for 0t

Feasible schedule with makespan 12



Feasible schedule with makespan 11

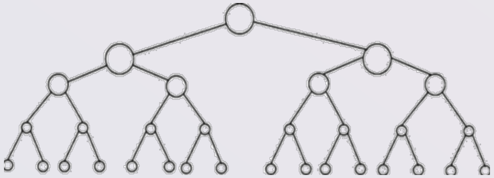


Aeronautics applications

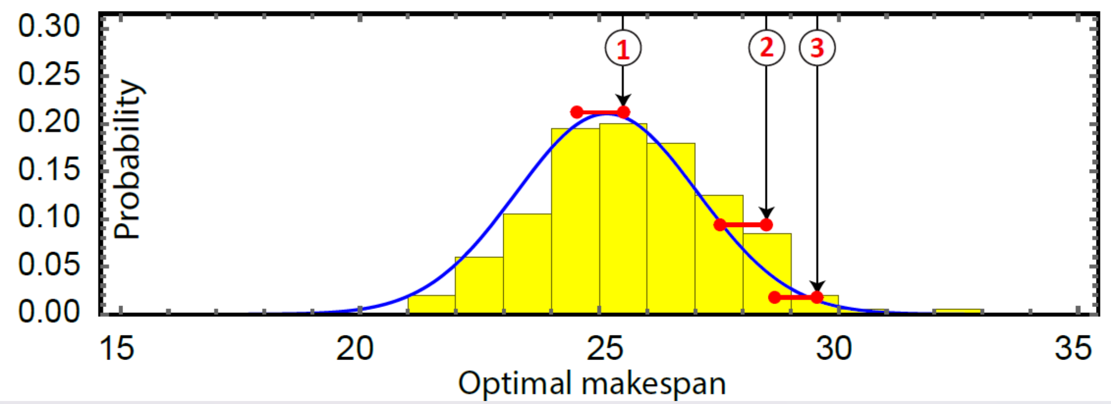
Computing applications

Resource allocation of assets

JSP as a CSP + Binary Search



Log$_2$(T) calls in the worst case

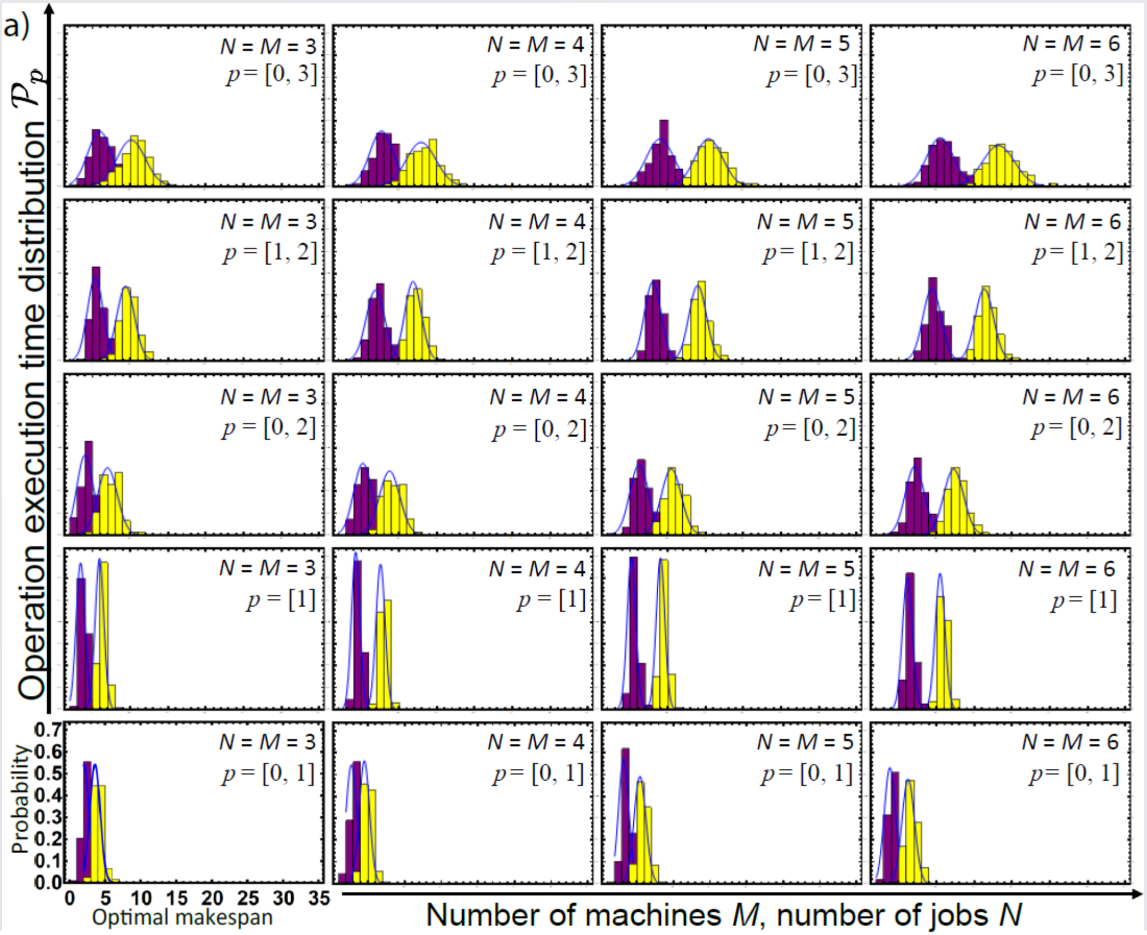


For N=M=2<p>=50

Knowing the distribution I need less than 5 calls on average, instead of ≈20

$$\operatorname{erf}\left(\frac{T_{\max}+\frac{1}{2}-\langle T\rangle}{\sigma\sqrt{2}}\right)+\operatorname{erf}\left(\frac{T_{\min}+\frac{1}{2}-\langle T\rangle}{\sigma\sqrt{2}}\right)=$$
$$\operatorname{erf}\left(\frac{T+\frac{1}{2}-\langle T\rangle}{\sigma\sqrt{2}}\right)+\operatorname{erf}\left(\frac{T-\max(1,\,K)+\frac{1}{2}-\langle T\rangle}{\sigma\sqrt{2}}\right),$$

Benchmarking: ensemble pre-characterization



JSP: QUBO mapping

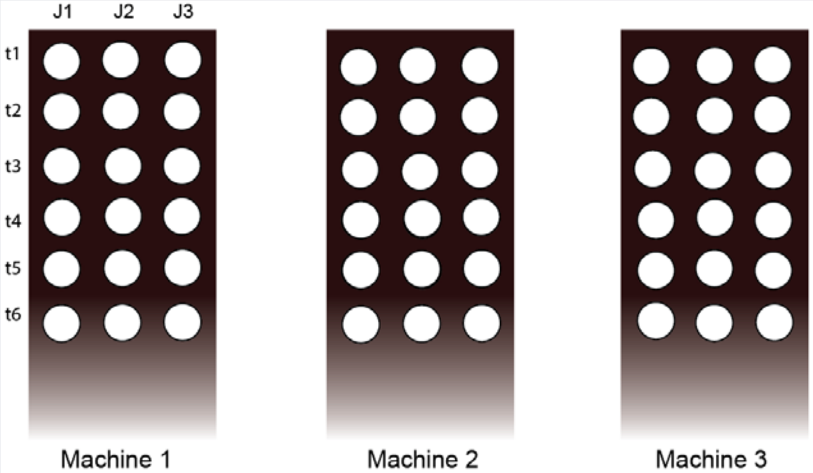
	1st operation	2nd operation	3rd operation
JOB 0	Machine 0 for 3t	Machine 1 for 2t	Machine 2 for 2t
JOB 1	Machine 0 for 2t	Machine 2 for 1t	Machine 1 for 4t
JOB 2	Machine 1 for 3t	Machine 2 for 3t	Machine 0 for 0t

$$E(x_1,\ldots,x_N)=\sum_{i\leq j}^{N}Q_{ij}x_ix_j$$

JSP: QUBO mapping

	1st operation	2nd operation	3rd operation
JOB 0	Machine 0 for 3t	Machine 1 for 2t	Machine 2 for 2t
JOB 1	Machine 0 for 2t	Machine 2 for 1t	Machine 1 for 4t
JOB 2	Machine 1 for 3t	Machine 2 for 3t	Machine 0 for 0t

$$E(x_1,\dots,x_N)=\sum_{i\leq j}^{N}Q_{ij}x_ix_j$$



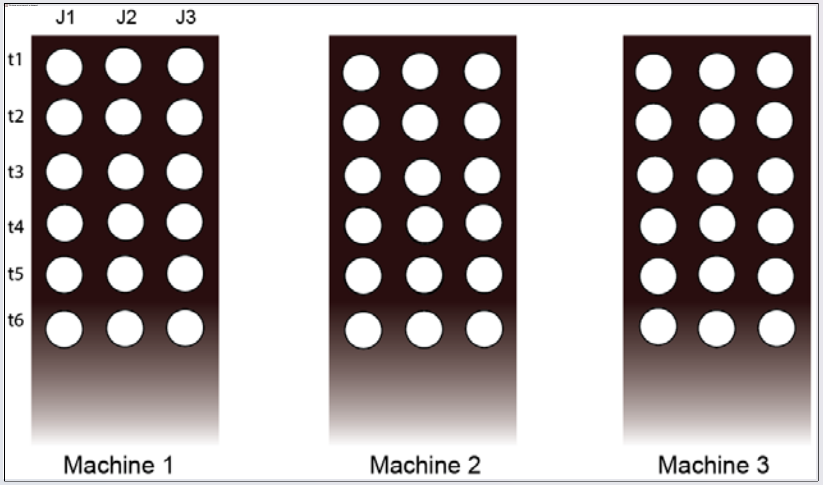
$x_{nmt} = 1$ If job n is executing on machine m at time t

$x_{nmt} = 0$ otherwise

JSP: QUBO mapping

	1st operation	2nd operation	3rd operation
JOB 0	Machine 0 for 3t	Machine 1 for 2t	Machine 2 for 2t
JOB 1	Machine 0 for 2t	Machine 2 for 1t	Machine 1 for 4t
JOB 2	Machine 1 for 3t	Machine 2 for 3t	Machine 0 for 0t

$$E(x_1,\dots,x_N)=\sum_{i\le j}^{N} Q_{ij}x_i x_j$$



$x_{nmt} = 1$ If job n is executing on machine m at time t

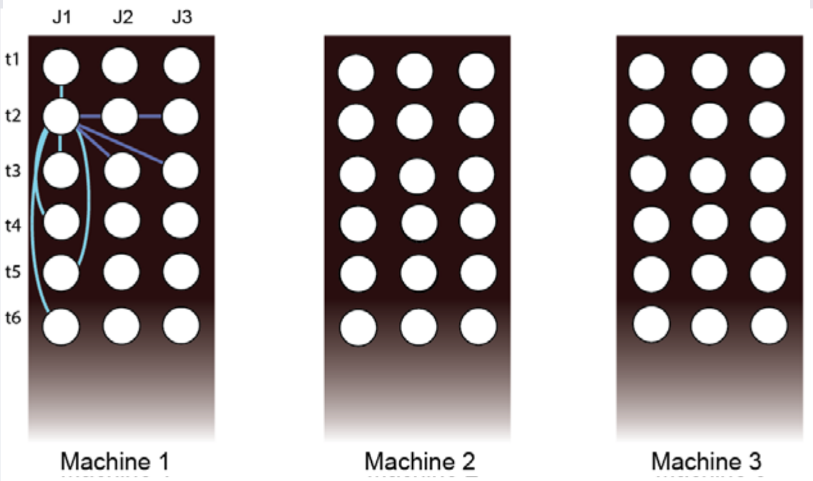
$x_{nmt} = 0$ otherwise

$$\sum_{n,m}\left(\sum_{t} x_{mnt} - 1\right)^2$$

JSP: QUBO mapping

	1st operation	2nd operation	3rd operation
JOB 0	Machine 0 for 3t	Machine 1 for 2t	Machine 2 for 2t
JOB 1	Machine 0 for 2t	Machine 2 for 1t	Machine 1 for 4t
JOB 2	Machine 1 for 3t	Machine 2 for 3t	Machine 0 for 0t

$$E(x_1,\dots,x_N) = \sum_{i\leq j}^{N} Q_{ij} x_i x_j$$



$x_{nmt} = 1$ If job n is executing on machine m at time t

$x_{nmt} = 0$ otherwise

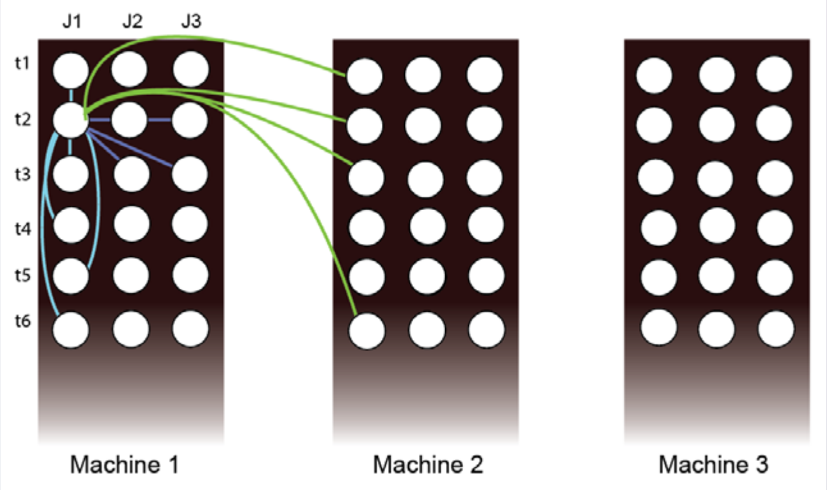
$$\sum_{n,m}\left(\sum_{t} x_{mnt} - 1\right)^2$$

$$\sum_{m,n}\left(\sum_{\bar{n}\neq n,\tau} x_{mnt} x_{m\bar{n}(t+\tau)}\right)$$

JSP: QUBO mapping

	1st operation	2nd operation	3rd operation
JOB 0	Machine 0 for 3t	Machine 1 for 2t	Machine 2 for 2t
JOB 1	Machine 0 for 2t	Machine 2 for 1t	Machine 1 for 4t
JOB 2	Machine 1 for 3t	Machine 2 for 3t	Machine 0 for 0t

$$E(x_1,\dots,x_N)=\sum_{i\le j}^{N} Q_{ij}x_i x_j$$



$x_{nmt} = 1$ If job n is executing on machine m at time t

$x_{nmt} = 0$ otherwise

$$\sum_{n,m}\left(\sum_{t} x_{mnt} - 1\right)^2$$

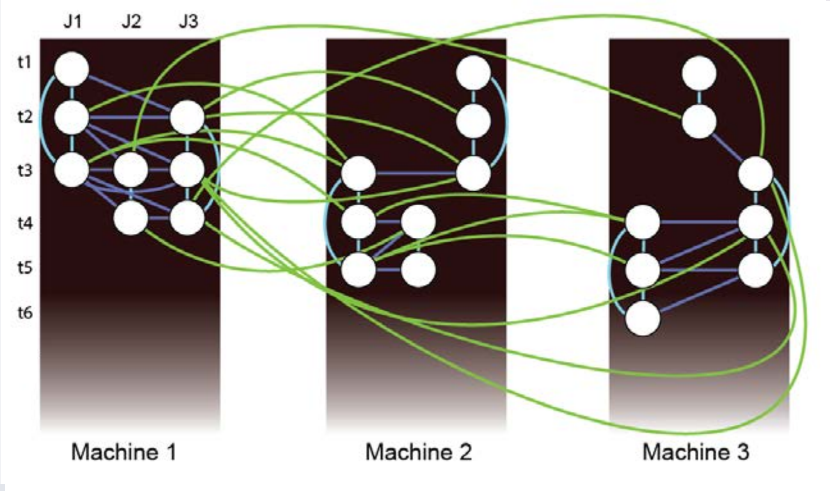
$$\sum_{m,n}\left(\sum_{\bar{n}\neq n,\tau} x_{mnt}x_{m\bar{n}(t+\tau)}\right)$$

$$\sum_{(m,n,t),(\bar{m},\bar{n},\bar{t})\in R_m} x_{mnt}x_{\bar{m}\bar{n}\bar{t}}$$

JSP: QUBO mapping

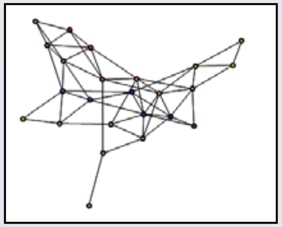
$$E(x_1,\ldots,x_N) = \sum_{i \le j}^{N} Q_{ij} x_i x_j$$

Simple execution time bounds computation



N M T bits required

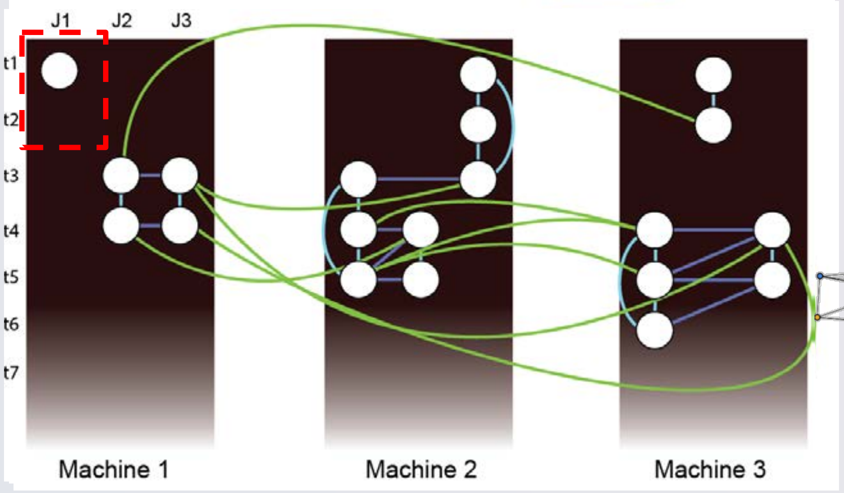
- N M (M <p> -1) bits



JSP: QUBO mapping

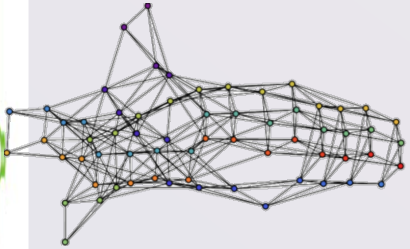
$$E(x_1,\ldots,x_N)=\sum_{i\le j}^{N} Q_{ij}x_i x_j$$

More advanced pre-processing (EdgeFinding, TaskInterval...)

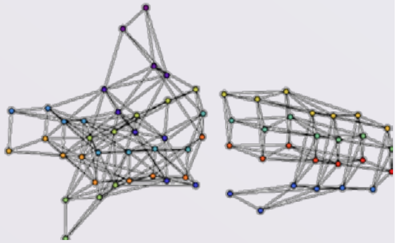


O(N²M²T logN) complexity

(Carlier and Pinson 1990)



N=4 M=4
T=7



2 parallel runs.
Easier embedding.

Wrapping up: hybrid approaches

PRE-PROCESSING

e.g. evaluating trivial simplifications where the job execution choices are obvious

- ❑ Polynomial algorithms of “shaving” and “pruning”
- ❑ Attempts to solve in polynomial time to eliminate easy instances

DECOMPOSITION SEARCHING

e.g. turning an optimization problem into a series of decision calls

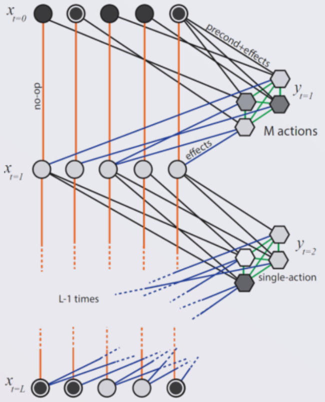
- ❑ Decomposing the problem in smaller sub-problems
- ❑ Explore the tree: exploration vs exploitation tradeoff

Use statistical information due to the pre-characterization of instance ensemble
Perhaps exploit the “unique sampling” capabilities of the annealer?

Other Scheduling Problems

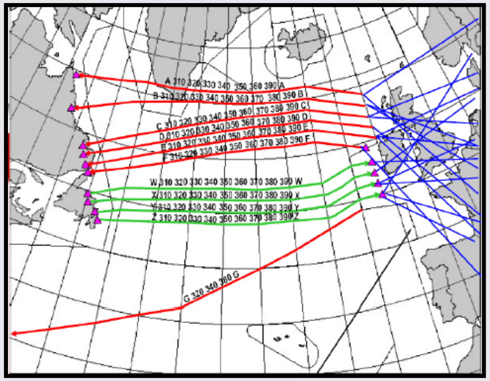
Not discussed..

- Planning (Rieffel et al.)
- Runway Landing Sequencing (Z.Wang et al.)
- Lagrangian Dual (Ronagh et al.)
- Database Query Optimization (Trummer et al.)
- Iterative Variable fixing heuristics (Karimi et al.)

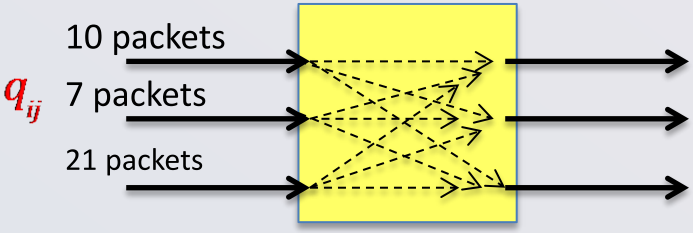


Air-Traffic-Management

See Stollenwerk!



Packet-Switching, Advisory Problems, Asset Allocation…

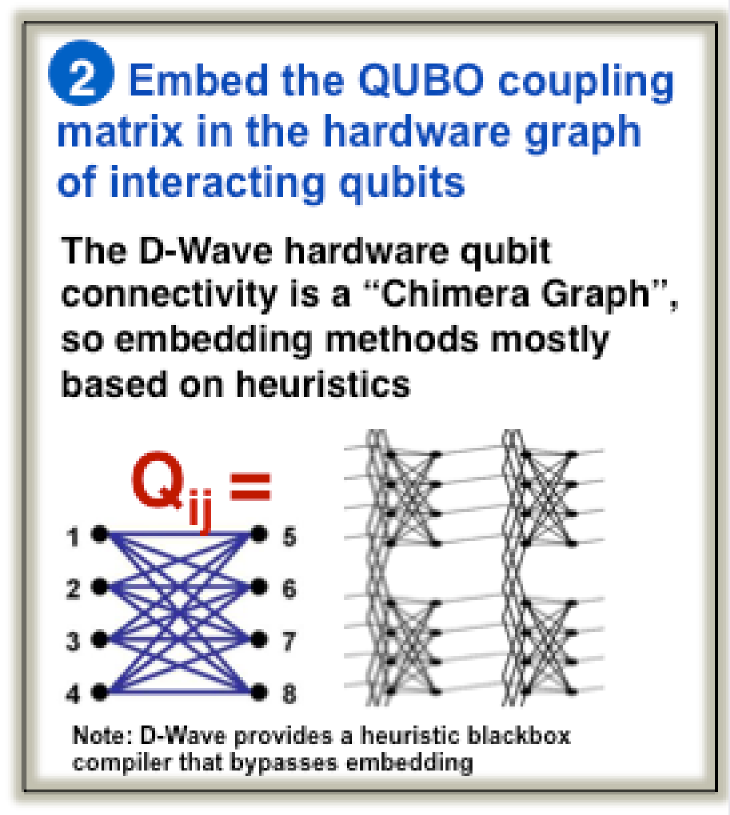


q_{ij}

10 packets

7 packets

21 packets



2 Embed the QUBO coupling matrix in the hardware graph of interacting qubits

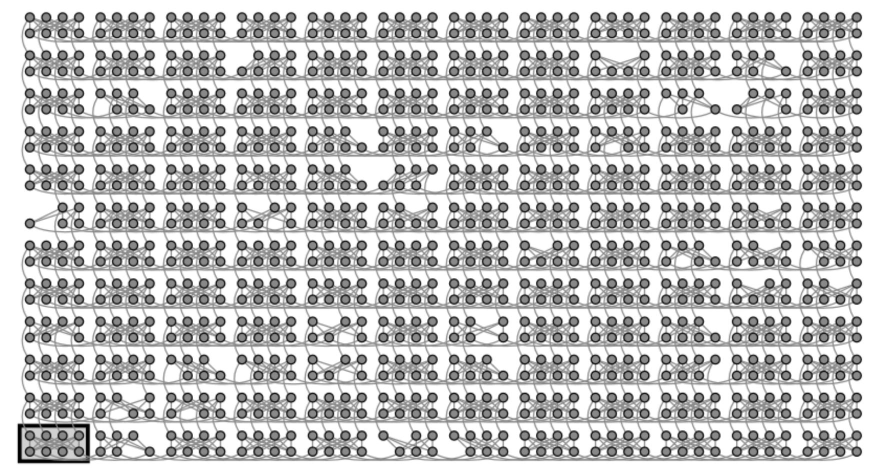
The D-Wave hardware qubit connectivity is a “Chimera Graph”, so embedding methods mostly based on heuristics

Q_{ij} =
1
2
3
4
5
6
7
8

Note: D-Wave provides a heuristic blackbox compiler that bypasses embedding

Embedding: optimizing Compilation

Minor embedding: QUBOs → Chimera



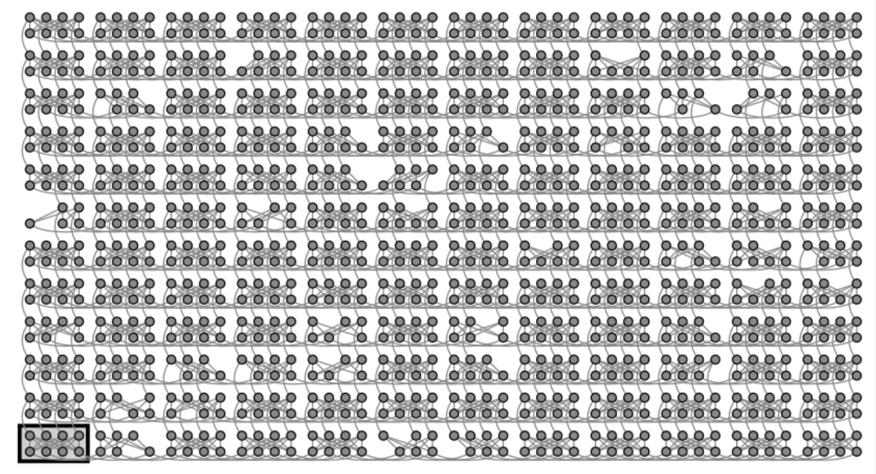
$$E(\mathbf{s}) = \sum_{i} h_i s_i + \sum_{i,j} J_{i,j} s_i s_j$$

$S_i = \pm 1$

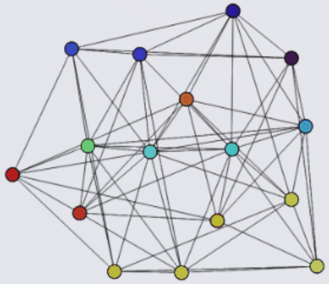
$h_i \approx [-1, 1]$ ≈10 values

$J_i \approx [-1, 1]$ ≈10 values

Minor embedding: QUBOs → Chimera



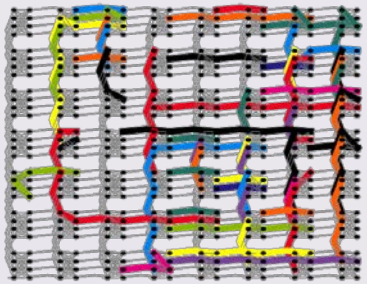
(n$_P$ logical bits)



$$\mathcal{E}(i) : \{1, \ldots, n_L\} \to 2^{\{1,\ldots,n_P\}}$$

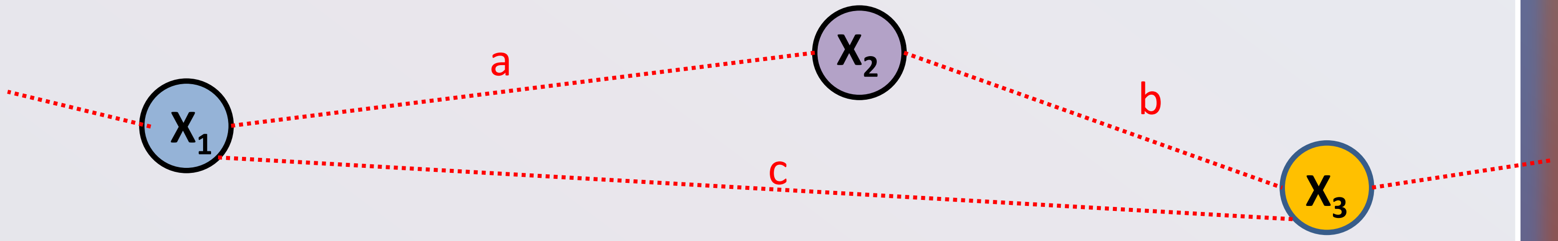
Assign “colors” to connected sets of qubits

(n$_H$ hardware qubits)



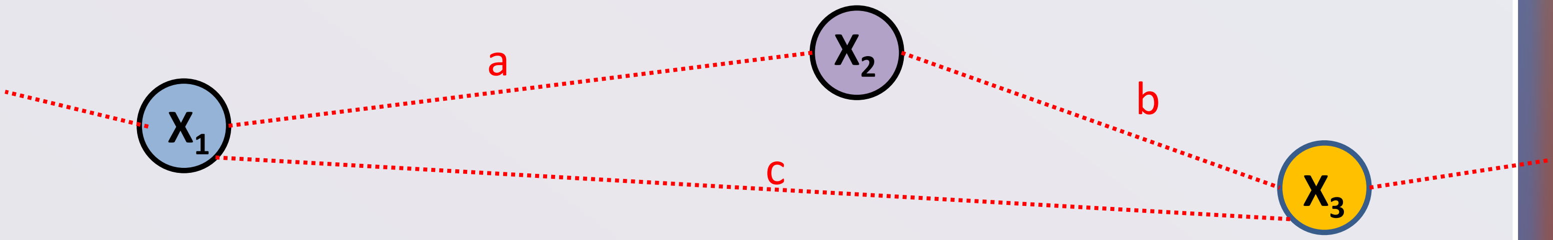
Embedding: Parameter setting

$a X_1 X_2 + b X_2 X_3 + c X_1 X_3$ **QUBO FORMULA**

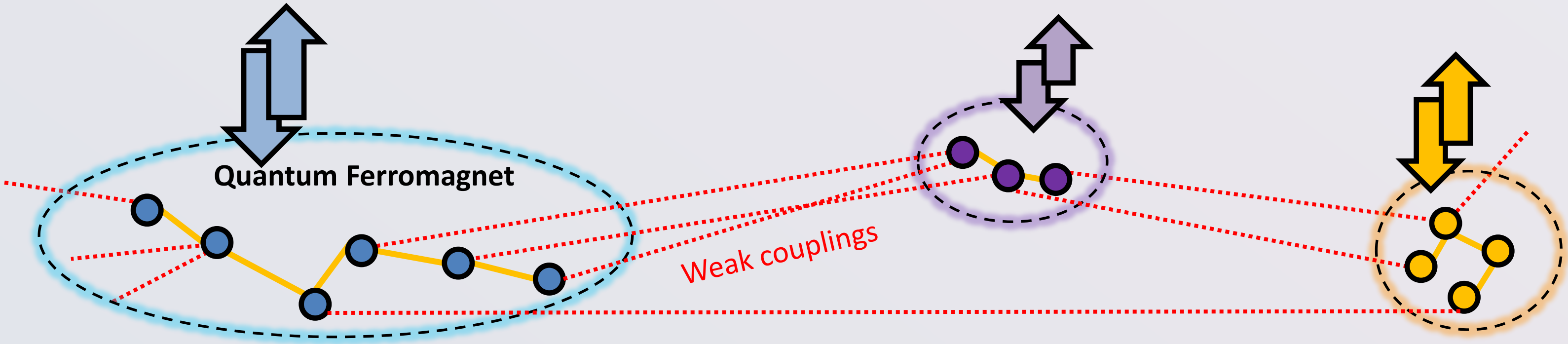


Embedding: Parameter setting

$a\,X_1 X_2 + b\,X_2 X_3 + c\,X_1 X_3$ QUBO FORMULA



AFTER EMBEDDING



Embedding: Parameter setting

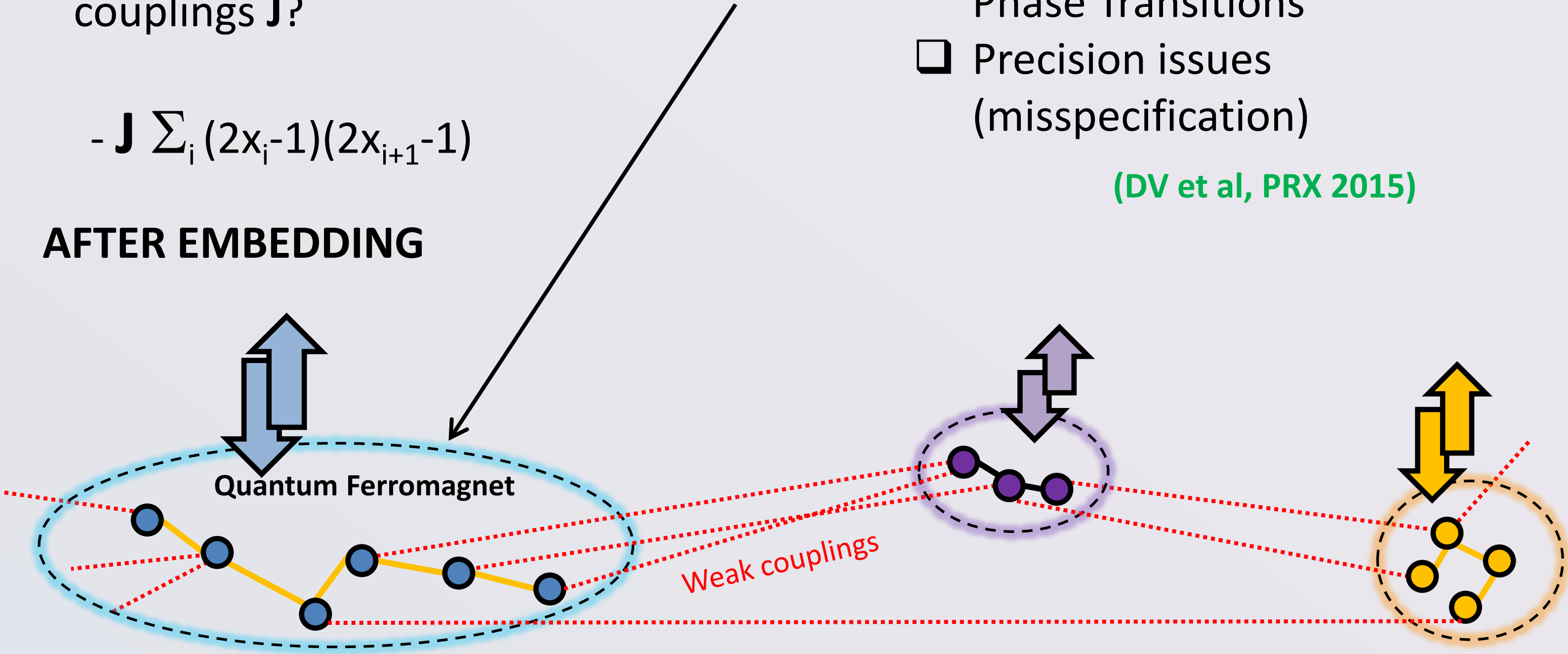
What is a good value of the internal couplings **J**?

$- \mathbf{J} \sum_i (2x_i-1)(2x_{i+1}-1)$

AFTER EMBEDDING

- ☐ Classical Energy landscape more rugged
- ☐ Emergence of Quantum Phase Transitions
- ☐ Precision issues (misspecification)

(DV et al, PRX 2015)



Embedded H

$$\sum_{ij} J_{ij}[\sigma_i^z][\sigma_j^z] \qquad \sum_i h_i[\sigma_i^z] - \Gamma_i \sum_i [\sigma_i^x]$$

Embedded H

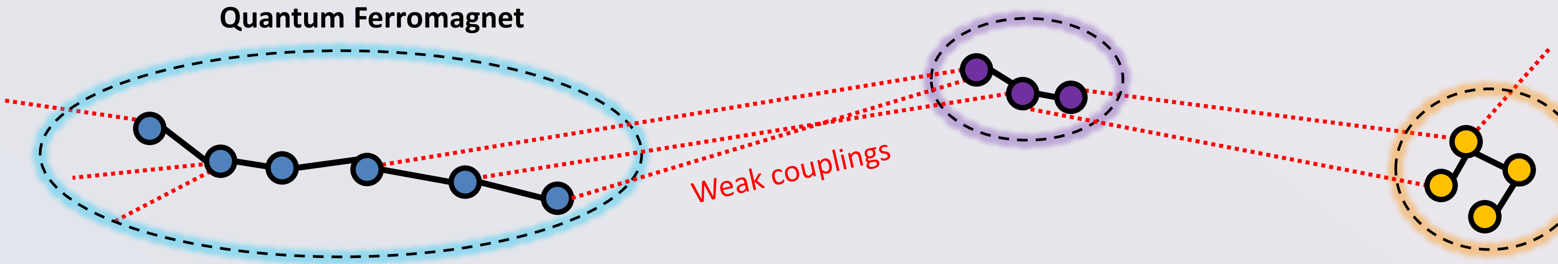
$$\begin{tikzcd}
\sum_{ij} J_{ij}[\sigma_i^z][\sigma_j^z] \arrow[d] \arrow[ddr] & & \sum_i h_i[\sigma_i^z] - \Gamma_i \sum_i [\sigma_i^x] \arrow[d] \\
\sum_{ij} J_{ij}[\sigma^z_{\alpha_i k_j}][\sigma^z_{\alpha_j k_i}] & & \sum_i \frac{h_i}{M_i}[\sum_k \sigma^z_{\alpha_i k}] - \Gamma_i \sum_i [\sum_k \sigma^x_{\alpha_i k}] \\
& \sum_{\alpha_i} \left(\sum_{k \in \alpha_i} J_F(\sigma^z_{\alpha_i k} \sigma^z_{\alpha_i k+1}) \right) &
\end{tikzcd}$$

Embedded H

$$\boxed{\sum_{ij} J_{ij}[\sigma_i^z][\sigma_j^z] \qquad \sum_i h_i[\sigma_i^z] - \Gamma_i \sum_i [\sigma_i^x]}$$

$$\sum_{ij} J_{ij}[\sigma^z_{\alpha_i k_j}][\sigma^z_{\alpha_j k_i}] \qquad \sum_i \frac{h_i}{M_i}[\sum_k \sigma^z_{\alpha_i k}] - \Gamma_i \sum_i [\sum_k \sigma^x_{\alpha_i k}]$$

$$\sum_{\alpha_i} \left(\sum_{k \in \alpha_i} J_F(\sigma^z_{\alpha_i k} \sigma^z_{\alpha_i k+1}) \right)$$

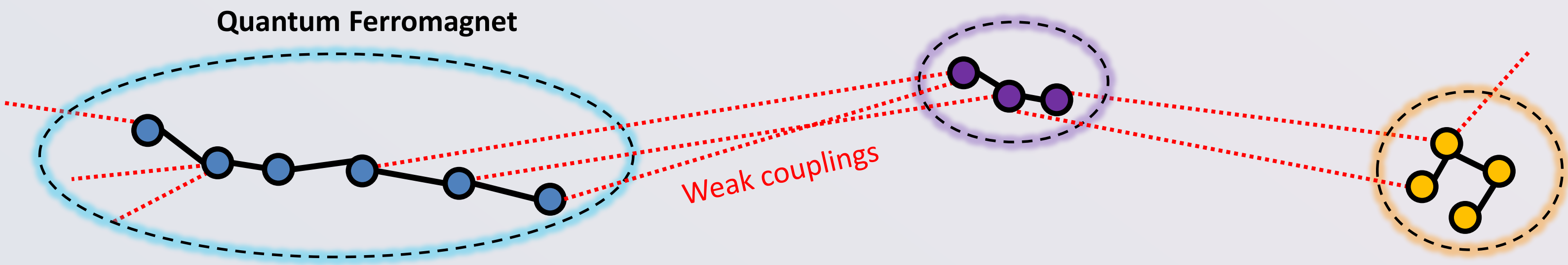


Embedded H: precision

$$\boxed{\sum_{ij} J_{ij}[\sigma_i^z][\sigma_j^z] \qquad \sum_i h_i[\sigma_i^z] - \Gamma_i \sum_i [\sigma_i^x]}$$

$$\sum_{ij} \underset{\overline{|J_F|}}{J_{ij}}[\sigma^z_{\alpha_i k_j}][\sigma^z_{\alpha_j k_i}] \qquad \sum_i \frac{h_i}{M_i}[\sum_k \sigma^z_{\alpha_i k}] - \Gamma_i \sum_i [\sum_k \sigma^x_{\alpha_i k}]$$

$$\sum_{\alpha_i}\left(\sum_{k\in\alpha_i} -1\ (\sigma^z_{\alpha_i k}\sigma^z_{\alpha_i k+1})\right)$$



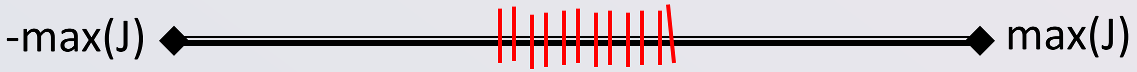
Embedded H: precision

$$\sum_{ij} J_{ij}[\sigma_i^z][\sigma_j^z] \qquad \sum_i h_i[\sigma_i^z] - \Gamma_i \sum_i [\sigma_i^x]$$

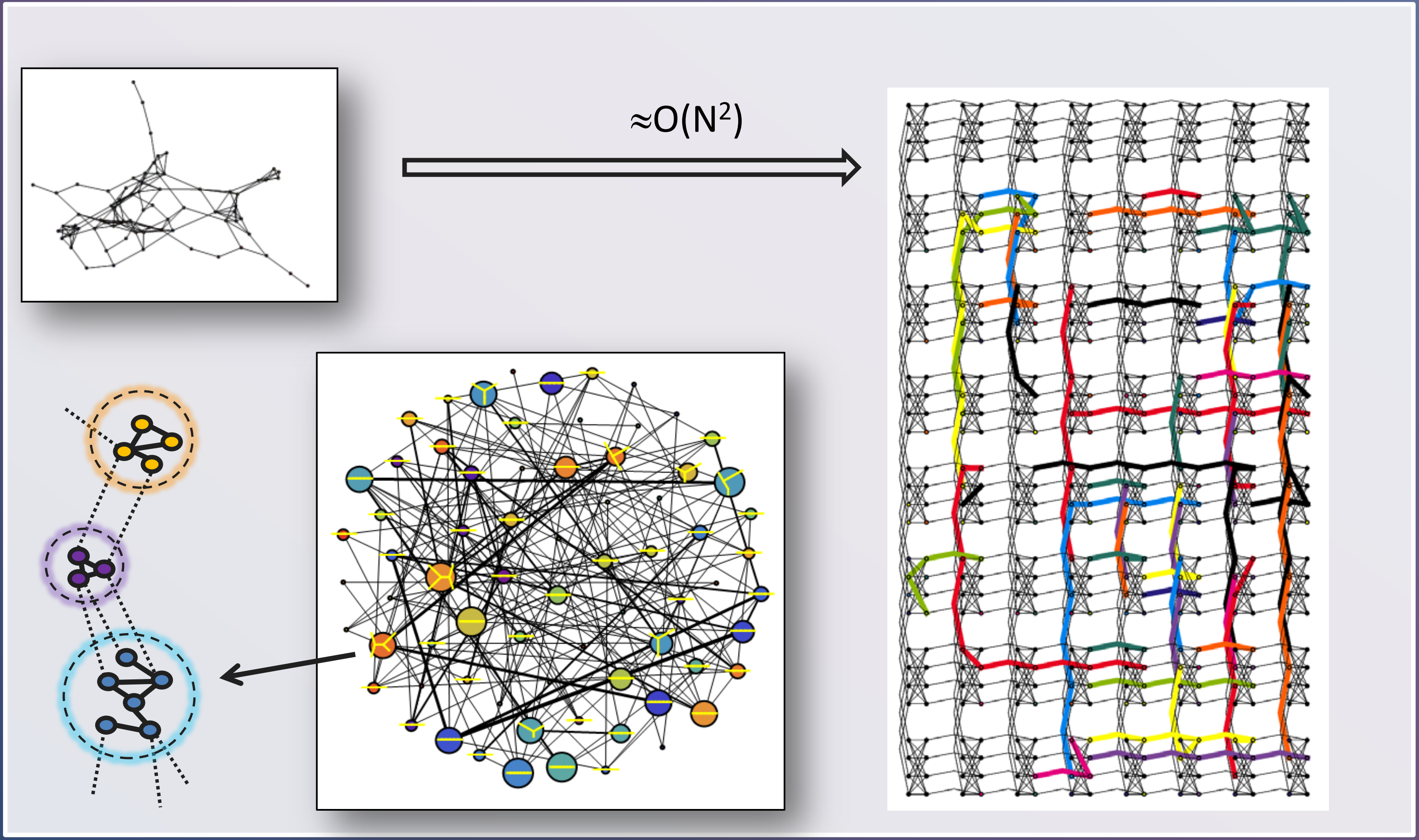
$$\sum_{ij} J_{ij}[\sigma^z_{\alpha_i k_j}][\sigma^z_{\alpha_j k_i}] \qquad \sum_i \frac{h_i}{M_i}[\sum_k \sigma^z_{\alpha_i k}] - \Gamma_i \sum_i [\sum_k \sigma^x_{\alpha_i k}]$$

$\overline{|J_F|}$

$$\sum_{\alpha_i}\left(\sum_{k\in\alpha_i} \mathbf{-1}\,(\sigma^z_{\alpha_i k}\sigma^z_{\alpha_i k+1})\right)$$



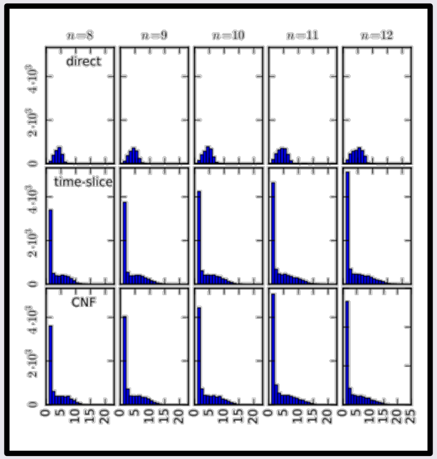
Embedding of one JSP instance

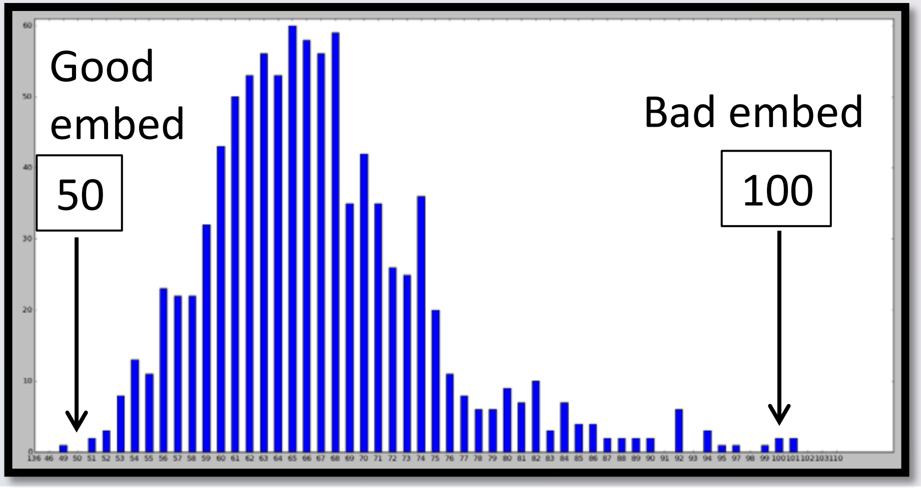


Topological aspect of embedding

D-Wave Heuristics (Cai et al.)

O’Gorman, B., Rieffel, E. G., Do, M., Venturelli, D., & Frank, J. “Compiling planning into quantum optimization problems: a comparative study.” Constraint Satisfaction Techniques for Planning and Scheduling Problems (COPLAS-15) (2015)





Good embed

50

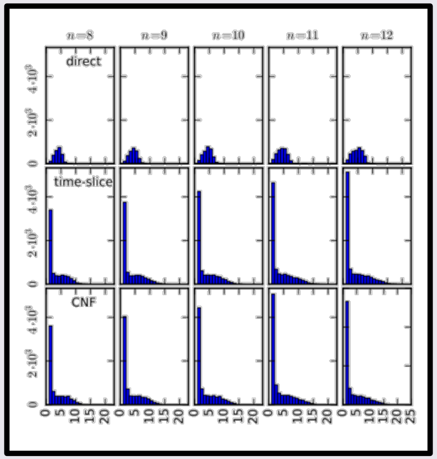
Bad embed

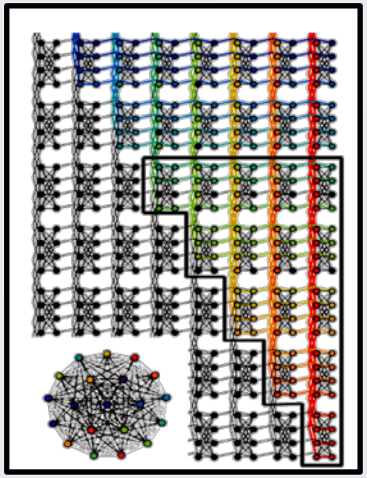
100

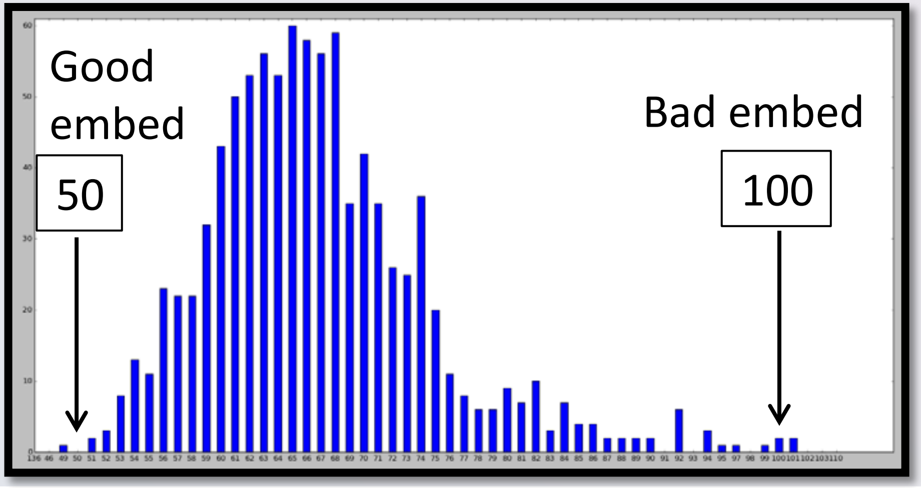
Topological aspect of embedding

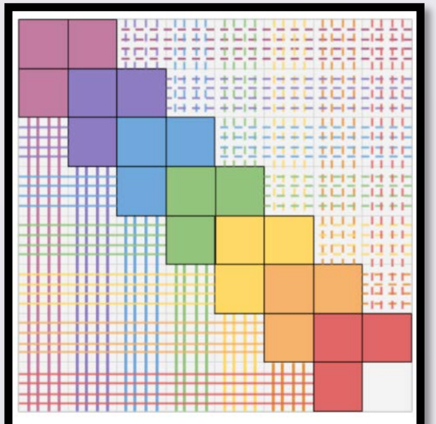
D-Wave Heuristics (Cai et al.)

O’Gorman, B., Rieffel, E. G., Do, M., Venturelli, D., & Frank, J. “Compiling planning into quantum optimization problems: a comparative study.” Constraint Satisfaction Techniques for Planning and Scheduling Problems (COPLAS-15) (2015)









(b) Embedding for $K_8\square K_7$

Embedding bottleneck

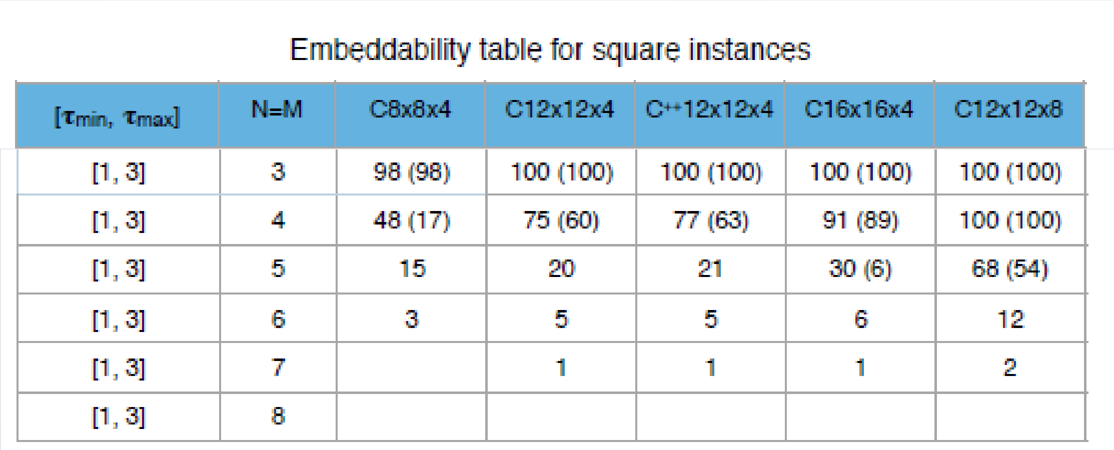
$\theta N M[T - \theta M\langle p\rangle + 1]$ **Logical Qubits**

Current heuristics

Previous D-Wave
50% of 4x4

Current D-Wave
20% of 5x5

Next D-Wave (?)
10% of 6x6



Embeddability table for square instances

$[\tau_{min}, \tau_{max}]$	N=M	C8x8x4	C12x12x4	C^{++}12x12x4	C16x16x4	C12x12x8
[1, 3]	3	98 (98)	100 (100)	100 (100)	100 (100)	100 (100)
[1, 3]	4	48 (17)	75 (60)	77 (63)	91 (89)	100 (100)
[1, 3]	5	15	20	21	30 (6)	68 (54)
[1, 3]	6	3	5	5	6	12
[1, 3]	7		1	1	1	2
[1, 3]	8					

Heuristic embedding not scalable…

Size	Time	Best method
5x5 τ=[1,20]	0.015 seconds	Scip
10x10 τ=[1,20]	2.75 seconds	Gurobi
15x15 τ=[1,20]	2430 seconds	Cplex (40%)

Need ≈ 6000 logical qubits for intractability.

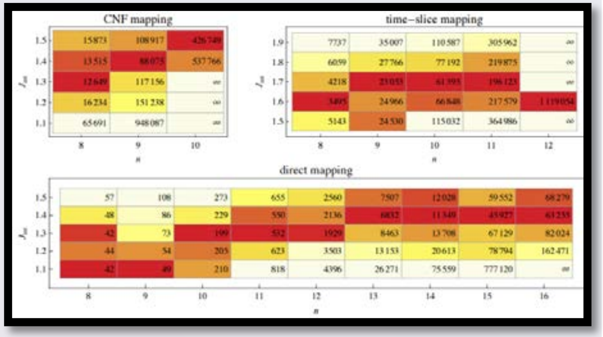
$\xrightarrow{\qquad\qquad\qquad}$ **≈ 1 M physical**

Ku, W.-Y. & Beck J.C., *Computers & Operations Research*, 73, 165-173, 2016.

Empirical approaches to parameter setting

Constant J_F

Rieffel, E., Venturelli, D., O'Gorman, B., Do, M. B., Prystay, E. M., & Smelyanskiy (2015)



CNF mapping

J_{ss} \ n	8	9	10
1.5	15 873	108 917	426 749
1.4	13 515	88 075	537 766
1.3	12 649	117 156	∞
1.2	16 234	151 238	∞
1.1	65 691	948 087	∞

time–slice mapping

J_{ss} \ n	8	9	10	11	12
1.9	7737	35 007	110 587	305 962	∞
1.8	6059	27 766	77 192	219 875	∞
1.7	4218	23 053	61 393	196 123	∞
1.6	3405	24 966	66 848	217 579	1 119 054
1.5	5143	24 530	115 032	364 986	∞

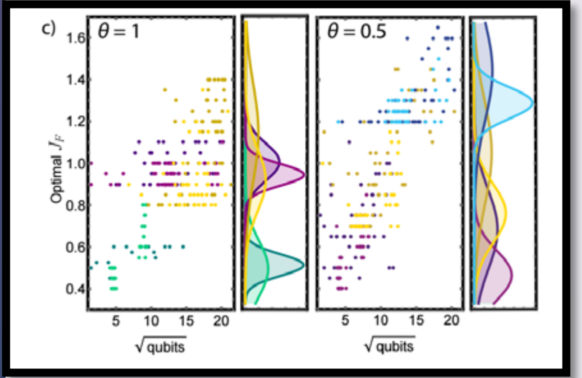
direct mapping

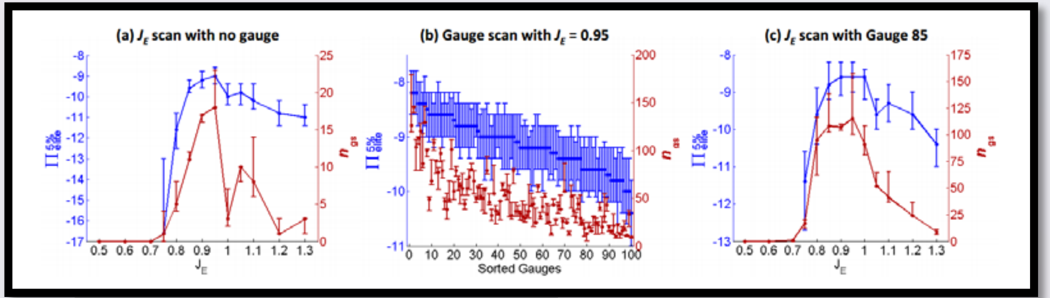
J_{ss} \ n	8	9	10	11	12	13	14	15	16
1.5	57	108	273	655	2560	7507	12 028	59 552	68 279
1.4	48	86	229	550	2136	6832	11 349	45 927	63 235
1.3	42	73	199	532	1929	8463	13 708	67 129	82 024
1.2	44	54	205	623	3503	13 153	20 613	78 794	162 471
1.1	42	49	210	818	4396	26 271	75 559	777 120	∞

Constant J_F, based on statistics

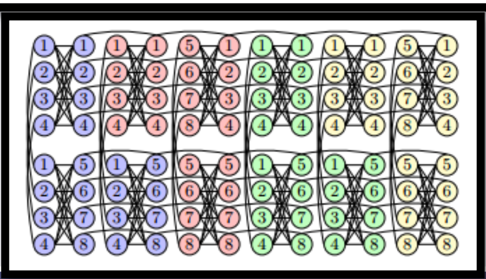
Venturelli, Davide, Dominic JJ Marchand, and Galo Rojo (2016)

Empirically adaptive





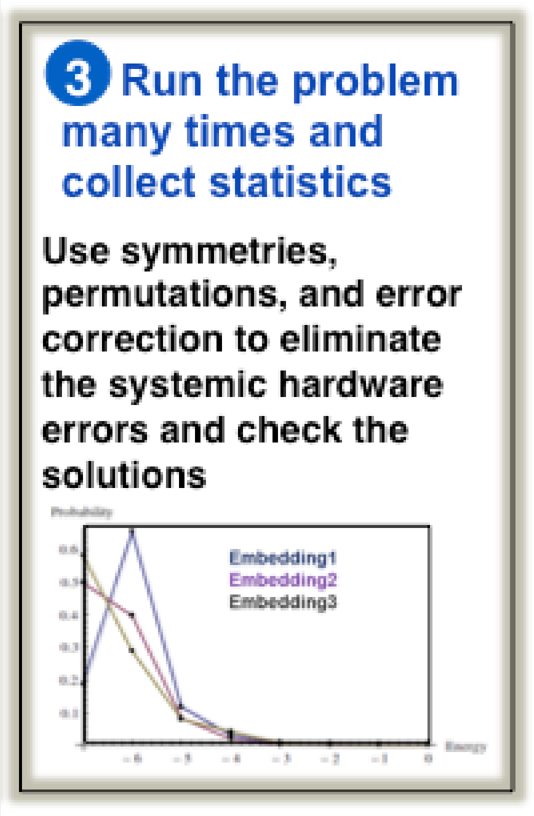
Trummer, I., & Koch, C. Multiple Query (2016)



Perdomo-Ortiz, Alejandro, et al. (2015).

Inspired by Classical Reasoning

Running and Analyzing



3 Run the problem many times and collect statistics

Use symmetries, permutations, and error correction to eliminate the systemic hardware errors and check the solutions

- Probability to find the ground state after 1 annealing run (20µs):

  P_{GS}

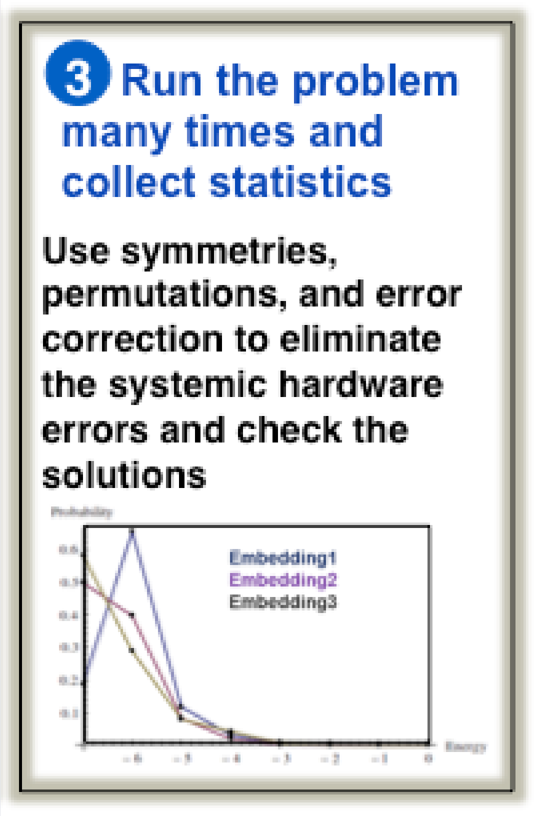
- Probability to find the ground state after R repetitions:

  $P^{X} = 1\text{-}(1\text{-}P_{GS})^{R}$

- **Expected number of repetitions to solve with 99% prob:**

  $R^{99} = \log(0.01)/\log(1\text{-}P_{GS})$

Running and Analyzing



3 Run the problem many times and collect statistics

Use symmetries, permutations, and error correction to eliminate the systemic hardware errors and check the solutions

Probability
0.6
0.5
0.4
0.3
0.2
0.1
Embedding1
Embedding2
Embedding3
-6
-5
-4
-3
-2
-1
0
Energy

- Probability to find the ground state after 1 annealing run (20μs):

  P_{GS}

- Probability to find the ground state after R repetitions:

  $P^{X} = 1-(1-P_{GS})^{R}$

- **Expected number of repetitions to solve with 99% prob:**

  $R^{99} = \log(0.01)/\log(1-P_{GS})$

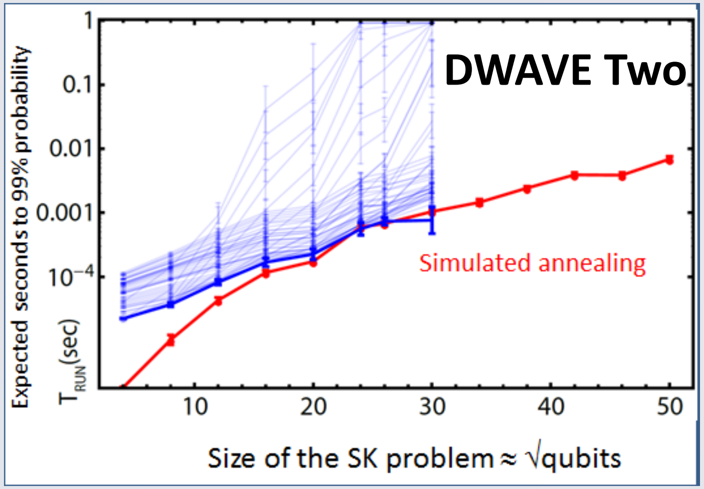
But Also:

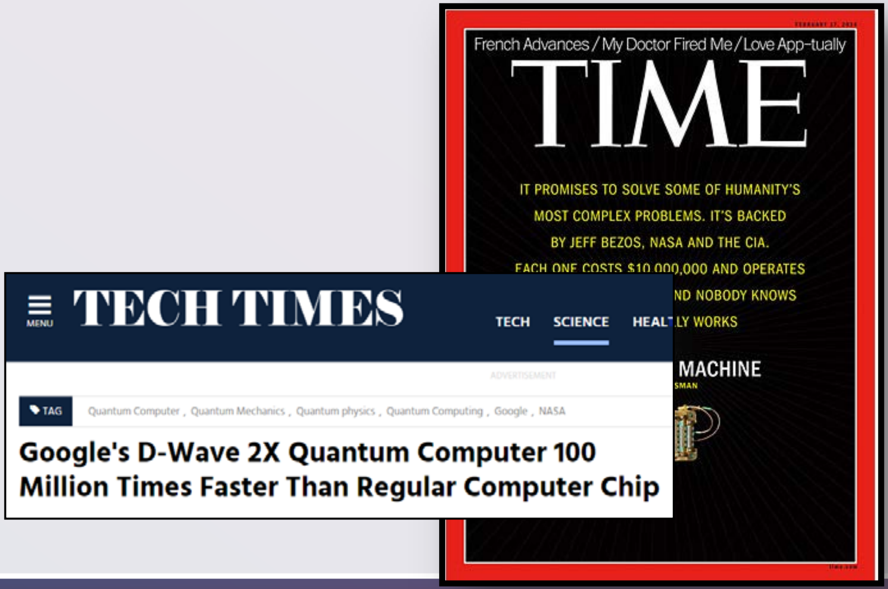
- # different solutions found at equal time.
- Best approximate solution found at equal time.

Hype and Reality

- NO SPEEDUP PROVEN NOT EVEN IN THEORY. SCALING ≈ SQA / PIQMC
- “PREFACTOR” 10^8 SPEEDUP AGAINST SIMULATED ANNEALING ON CRAFTED INSTANCES DESIGNED AGAINST S.A.
- SOME EARLY EVIDENCE OF UNIQUE SAMPLING (MACHINE LEARNING, ETC.)
- AT MOST “COMPETITIVE” WITH 1-CORE ON NATIVE/EMBEDDED PROBLEMS*
- THE SCALING IS DIFFICULT TO OBSERVE FOR SMALL N

DV et al. PRX (2015)




FEBRUARY 17, 2014
French Advances / My Doctor Fired Me / Love App-tually
TIME
IT PROMISES TO SOLVE SOME OF HUMANITY'S
MOST COMPLEX PROBLEMS. IT'S BACKED
BY JEFF BEZOS, NASA AND THE CIA.
EACH ONE COSTS $10,000,000 AND OPERATES
ND NOBODY KNOWS
LY WORKS
MACHINE
SMAN
≡ MENU
TECH TIMES
TECH
SCIENCE
HEAL
ADVERTISEMENT
TAG
Quantum Computer , Quantum Mechanics , Quantum physics , Quantum Computing , Google , NASA

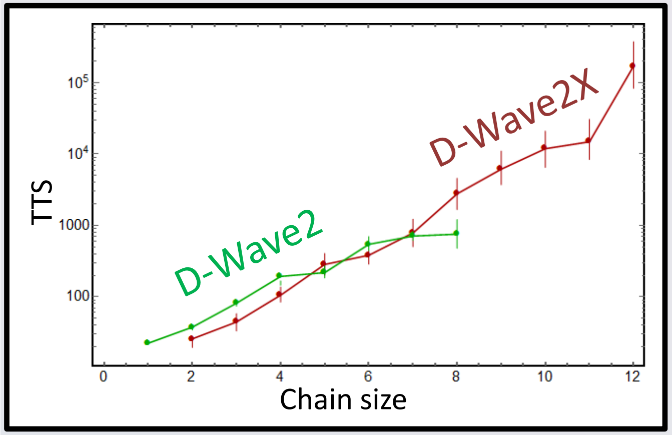
Google's D-Wave 2X Quantum Computer 100 Million Times Faster Than Regular Computer Chip

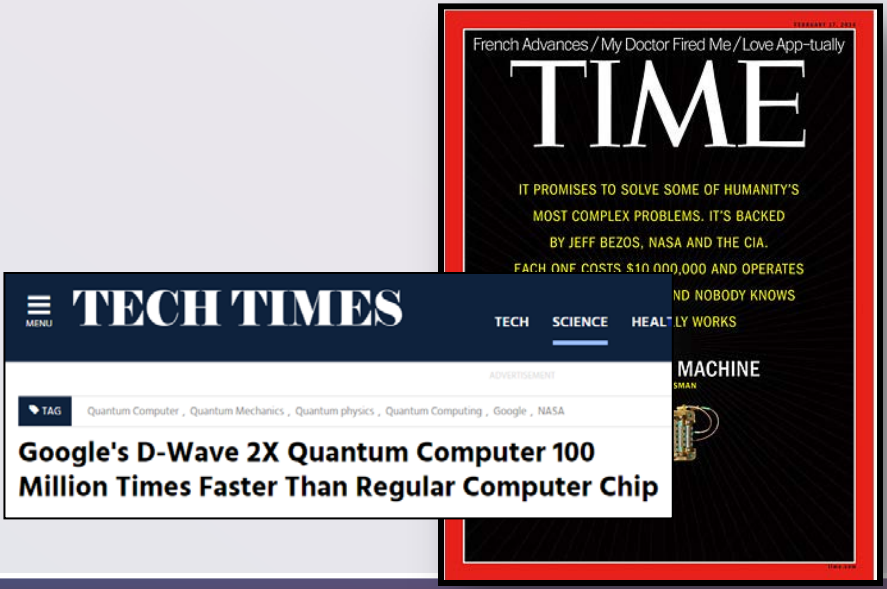
Hype and Reality

- NO SPEEDUP PROVEN NOT EVEN IN THEORY. SCALING ≈ SQA / PIQMC
- "PREFACTOR" 10^8 SPEEDUP AGAINST SIMULATED ANNEALING ON CRAFTED INSTANCES DESIGNED AGAINST S.A.
- SOME EARLY EVIDENCE OF UNIQUE SAMPLING (MACHINE LEARNING, ETC.)
- AT MOST "COMPETITIVE" WITH 1-CORE ON NATIVE/EMBEDDED PROBLEMS*
- THE SCALING IS DIFFICULT TO OBSERVE FOR SMALL N

DV et al. PRX (2015)

DWAVE2X With 20 μs



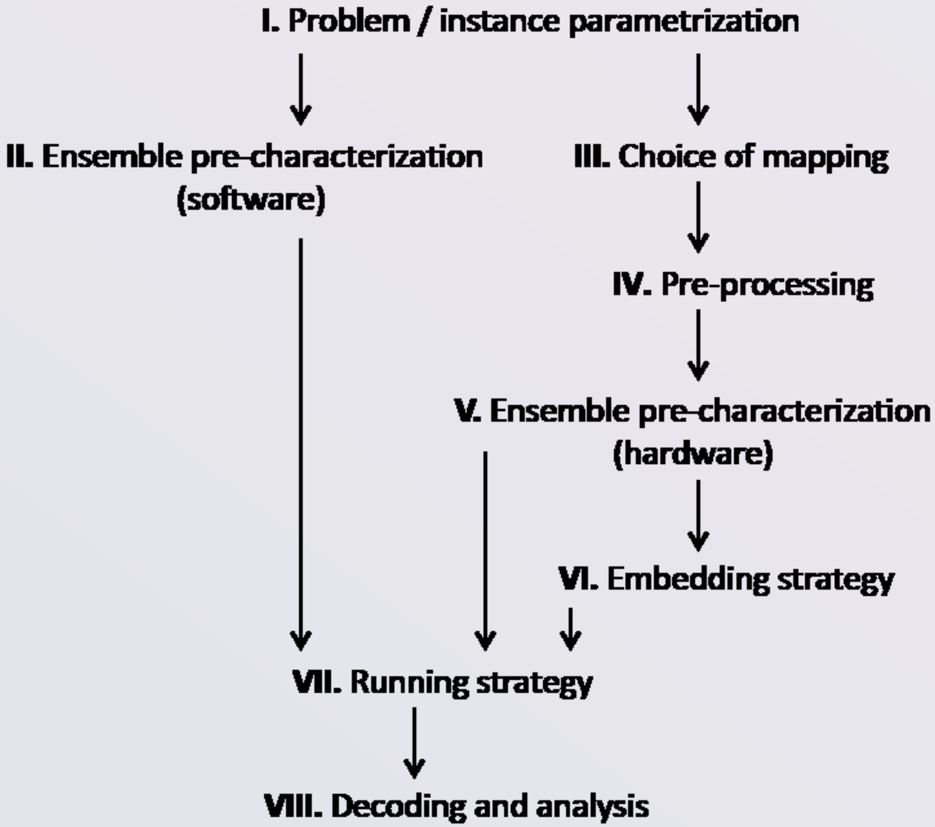


IT PROMISES TO SOLVE SOME OF HUMANITY'S MOST COMPLEX PROBLEMS. IT'S BACKED BY JEFF BEZOS, NASA AND THE CIA. EACH ONE COSTS $10,000,000 AND OPERATES
ND NOBODY KNOWS
LY WORKS

Quantum Computer , Quantum Mechanics , Quantum physics , Quantum Computing , Google , NASA

Google's D-Wave 2X Quantum Computer 100 Million Times Faster Than Regular Computer Chip

Example of running strategy (JSP)



I. Problem / instance parametrization

II. Ensemble pre-characterization (software)

III. Choice of mapping

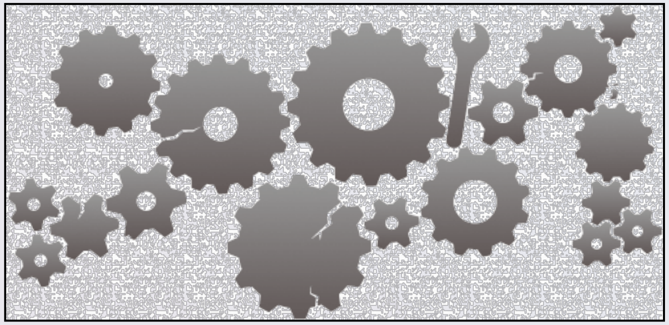
IV. Pre-processing

V. Ensemble pre-characterization (hardware)

VI. Embedding strategy

VII. Running strategy

VIII. Decoding and analysis

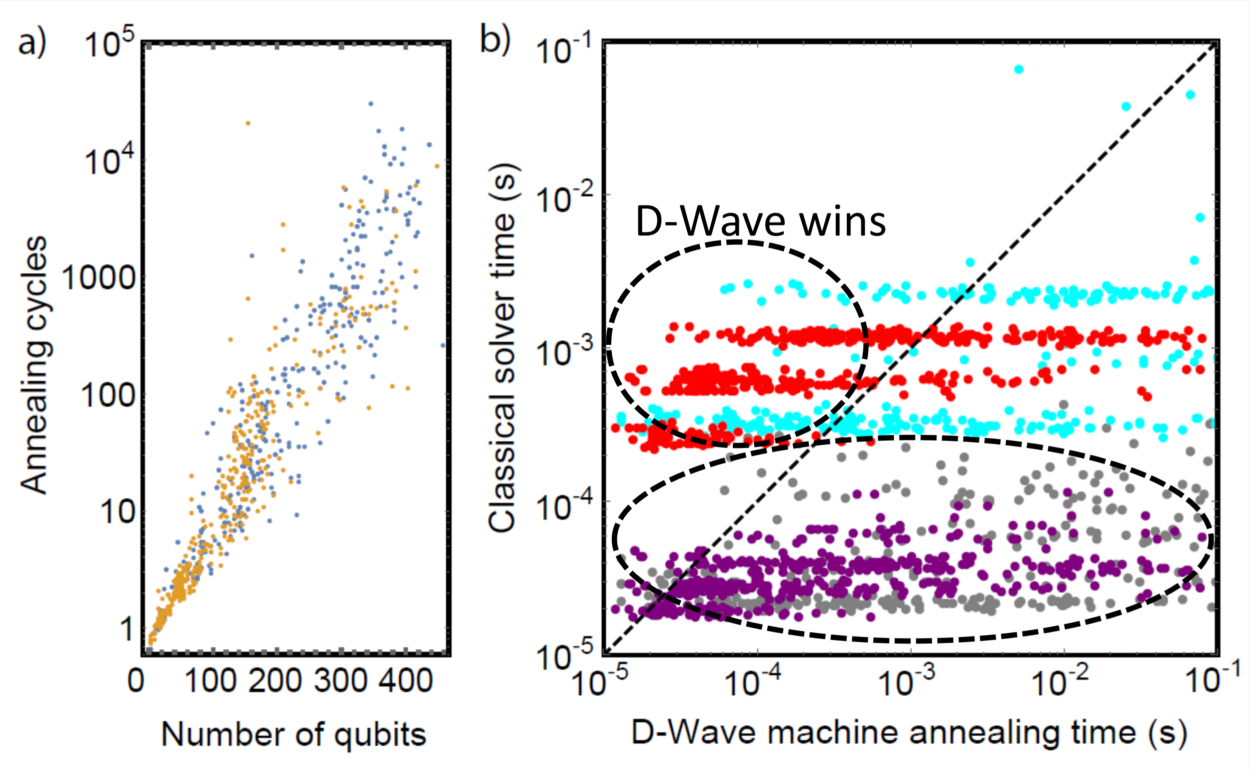


But also:

Performance tuning
(Perdomo-ortiz et al. 2015)
Error suppression
(Pudenz et al. 2014)
(Rieffel et al. 2015)

D-Wave Two Results

Time to solve at 99% probability $R^{99} = \log(0.01)/\log(1-P_{GS})$



Decision Solver

Martin, Shmoys '96

Full opt B&B

Brucker '94

Improvements and outlooks

SHORT TERM (2016-2017)

- ❑ BETTER EMBEDDING TECHNIQUES
  - ❑ NEW WORKS ON SEMI-DETERMINISTIC MILP EMBEDDINGS
  - ❑ PARAMETER SETTING CAN BE IMPROVED (x10 perfomance)
- ❑ HYBRID APPROACHES
  - ❑ RELAXATIONS, DECOMPOSITIONS
- ❑ APPROXIMATE SOLUTIONS?

⇓

Speed can be improved by 50-100x

Improvements and conservative outlooks

SHORT TERM (2016-2017)

- BETTER EMBEDDING TECHNIQUES
  - NEW WORKS ON SEMI-DETERMINISTIC MILP EMBEDDINGS
  - PARAMETER SETTING CAN BE IMPROVED (x10 perfomance)
- HYBRID APPROACHES
  - RELAXATIONS, DECOMPOSITIONS
- APPROXIMATE SOLUTIONS?

MEDIUM TERM (2018-2020)

- BETTER ARCHITECTURE, N□5000
- MORE COMPLEX SCHEDULE
  - INCREASED QUANTUMNESS
  - INTERPLAY WITH DISSIPATION

↓ Speed can be improved by 50-100x

Improvements and conservative outlooks

SHORT TERM (2016-2017)

- ❑ BETTER EMBEDDING TECHNIQUES
  - ❑ NEW WORKS ON SEMI-DETERMINISTIC MILP EMBEDDINGS
  - ❑ PARAMETER SETTING CAN BE IMPROVED (x10 perfomance)
- ❑ HYBRID APPROACHES
  - ❑ RELAXATIONS, DECOMPOSITIONS
- ❑ APPROXIMATE SOLUTIONS?

Speed can be improved by 50-100x

MEDIUM TERM (2018-2020)

- ❑ BETTER ARCHITECTURE, N□5000
- ❑ MORE COMPLEX SCHEDULE
  - ❑ INCREASED QUANTUMNESS
  - ❑ INTERPLAY WITH DISSIPATION

hope

Problems that take 10min could be solved in milliseconds?

Research Opportunity on D-Wave 2X

- Oak Ridge National Laboratory (USA)
- Scuola Normale Superiore di Pisa (ITALY)
- Swiss Fed. Inst. Tech Lausanne (SWITZERLAND)
- Mississippi State University (USA)
- University of British Columbia (CANADA)
- Technológico de Monterrey (MEXICO)
- University of California, San Diego (USA)
- University of Southern California (USA)
- University of Verona (ITALY)
- University of Oxford (UK)
- TATA Consulting Services (India)
- Fiat Physica (USA)
- 1-Qbit (CANADA)
- QC-Ware (USA)
- QX-Branch (USA)
- Lockheed Martin (USA)
- Carnegie Mellon University (USA)
- Cornell University (USA)

1097 Qubits
5 μs min anneal time
24/7 support

USRA

http://www.usra.edu/quantum/rfp

(5 pages proposal, training)

davide.venturelli@nasa.gov