

QSPIN: A High Level Java API for Quantum Computing Experimentation¹

Tim Barth

NASA Ames Research Center
Moffett Field, California 94035 USA
(Timothy.J.Barth@nasa.gov)

¹Work supported by the NASA ARMD Transformational Tools and Technology (TTT) project. 🔍 🔍 🔍



Introduction and Outline

**QSPIN: A
High Level
Java API**

Tim Barth

Outline

Overview of
the D-Wave
Quantum
Annealer

Applications
Software

QSPIN API

Hybrid
Quantum-
Classical
QUBO
Solvers

QUBO
Model Error

Further Ap-
plications

Looking
Forward

- Overview of the D-Wave Quantum Annealer
- **QSPIN** Java API and benchmark applications
- Hybrid quantum-classical QUBO solvers
- Ising and QUBO model errors
- Looking forward

The behavior of the D-Wave quantum annealer ground state is closely approximated by Ising spin model objective minimization

Ising spin model

$$s^* = \operatorname{argmin}_s \sum_{i=1}^N h_i s_i + \sum_{i=1}^N \sum_{j=i+1}^N s_i J_{ij} s_j, \quad s_i \in \{-1, +1\}$$

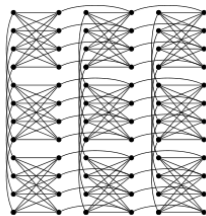
with $h_i \in [-2, 2]$ and $J_{ij} \in [-1, 1]$ which is mathematically equivalent to the Quadratic Unconstrained Binary Optimization function minimization

QUBO

$$x^* = \operatorname{argmin}_x \sum_{i=1}^N \sum_{j=i}^N x_i Q_{ij} x_j, \quad x_i \in \{0, 1\}$$

Ising and QUBO problems are NP-hard and thus, if $P \neq NP$, can not be solved in polynomial time.

The hardware graphs associated with J and Q are sparse but this does not change the NP-hardness of the reduced problems.

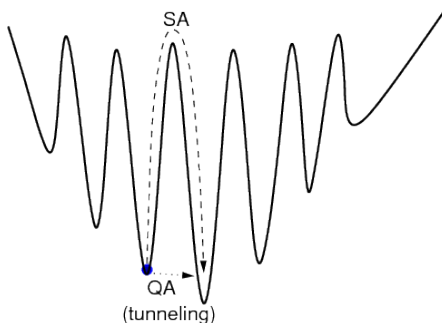


Adiabatic Quantum Optimization:

The annealing process starts from an initial transverse field Hamiltonian H_0 and slowly transitions in normalized time to the desired Ising spin Hamiltonian H_1 along the nondecreasing path $f(t) : [0, 1] \mapsto [0, 1]$ with effective Hamiltonian

$$H(t) = (1 - f(t))H_0 + f(t)H_1$$

The transverse field Hamiltonian H_0 plays a key role as it is responsible for quantum tunneling through tall narrow peaks in the energy landscape.





Applications Software Challenges

QSPIN: A
High Level
Java API

Tim Barth

Outline

Overview of
the D-Wave
Quantum
Annealer

Applications
Software

QSPIN API

Hybrid
Quantum-
Classical
QUBO
Solvers

QUBO
Model Error

Further Ap-
plications

Looking
Forward

Application programmers often encounter two challenges associated with the D-Wave quantum annealer

(1) Solving large scale Ising/QUBO problems on small scale quantum annealing hardware

Ising and QUBO problems with dense coupling graphs require an embedding onto the sparse D-Wave hardware chimera graph. This can be accomplished by the introduction of auxiliary qubits but it significantly decreases the size of problems that can be executed entirely on the D-Wave hardware.

(2) Coping with large Ising/QUBO coefficients and model error

The Ising and QUBO mathematical models are only approximately realized by the D-Wave hardware which can negatively impact optimization performance and complicate the selection of software parameters.

These issues are primary research areas for the **QSPIN** software project that are addressed via the use of hybrid quantum-classical algorithms.

QSPIN is a platform independent Java language API to the D-Wave 2000Q quantum annealer.

- ① **QSPIN-SAPI** : a low level Java native interface (JNI) to the D-Wave SAPI library.
- ② **QSPIN-CORE** : a pure Java language counterpart of the D-Wave SAPI storage classes and methods.
- ③ **QSPIN-LEVEL2** : high level Java objects and methods that greatly simplify use of the D-Wave system, provides hybrid quantum-classical solver algorithms and constrained QUBO representations, and tools to support ongoing hybrid quantum-classical algorithm research.
- ④ **QSPIN-DEMO** : a suite of 18 NP-hard and NP-complete applications (summarized in A. Lucas, 2013) written entirely in Java using **QSPIN**.

graph coloring Hamiltonian cycles clique covering binary linear programming minmax job sequencing set covering	graph MIS traveling salesman graph K -cliques boolean satisfiability number partitioning exact set covering	graph K -way partitioning vertex covering feedback vertex sets knapsack problem subset sum set packing
---	--	---

Remark: Traveling salesman problem using exact $\mathcal{O}(n^2 2^n)$ dynamic programming algorithm on exascale classical computer with $n = 100$ cities:

$$100^2 2^{100} ops \times \frac{sec}{10^{18} ops} \times \frac{year}{365 \times 24 \times 60^2 sec} \approx 402 \text{ million years}$$



Hybrid Quantum-Classical QUBO Solvers

QSPIN: A
High Level
Java API

Tim Barth

Outline

Overview of
the D-Wave
Quantum
Annealer

Applications
Software

QSPIN API

Hybrid
Quantum-
Classical
QUBO
Solvers

QUBO
Model Error

Further Ap-
plications

Looking
Forward

Hybrid quantum-classical QUBO solvers decompose a large QUBO problem into smaller QUBO subproblems that execute on the D-Wave quantum annealer.

D-Wave has supported the development of *QBSolve*² which uses tabu local search for a QUBO problem combined with D-Wave quantum annealing solves of smaller partitioned QUBO subproblems.

QSPIN provides a number of tools for research activities in hybrid quantum-classical solvers

- Suite of graph manipulation tools for graph contraction, restriction, prolongation, and visualization.
- A Java implementation of the *QBSolve* hybrid quantum-classical QUBO solver,
- A Java JNI to the METIS multi-level graph partitioner,
- D-Wave full graph solver with precomputed fixed embedding,
- Java-GPU accelerated simulated annealing solver.

²"Partitioning Optimization Problems for Hybrid Classical/Quantum Execution", Booth, Reinhardt, and Aidan Roy, 2017.



QBSolve Hybrid Quantum-Classical QUBO Solver

QSPIN: A
High Level
Java API

Tim Barth

Outline

Overview of
the D-Wave
Quantum
Annealer

Applications
Software

QSPIN API

Hybrid
Quantum-
Classical
QUBO
Solvers

QUBO
Model Error

Further Ap-
plications

Looking
Forward

QBSolve is 2-level QUBO solver. An key data structure is a ranked *impact vector* which measures the change in the objective function when a particular bit is flipped.

Algorithm QBSolve (Booth *et al.*, 2017):

- 1 The global QUBO problem is approximated by a local tabu search,
- 2 Subproblems are selected from the ranked impact vector,
- 3 QUBO subproblems are solved on the D-Wave quantum annealer with clamped external values,
- 4 Values of the global problem are updated,
- 5 If no progress is made after the subproblem update, the bits associated with the subproblem are randomized.
- 6 Go to (1) until no further progress is obtained.

Hybrid Quantum-Classical Example: K-way partitioning

QSPIN: A
High Level
Java API

Tim Barth

Outline

Overview of
the D-Wave
Quantum
Annealer

Applications
Software

QSPIN API

Hybrid
Quantum-
Classical
QUBO
Solvers

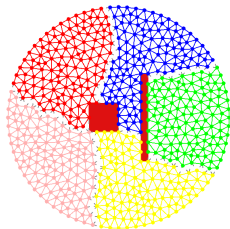
QUBO
Model Error

Further Ap-
plications

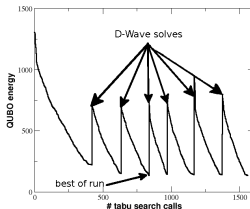
Looking
Forward

K-way Graph Partitioning

$$\begin{aligned}
 x^* = \operatorname{argmin}_x & \underbrace{\sum_{(v_i, v_j) \in E} \sum_{k=1}^K \sum_{\substack{l=1 \\ k \neq l}}^K x_{v_i, k} x_{v_j, l}}_{\text{minimize cut edges}} \\
 & + \delta_0 \sum_{i=1}^{|V|} \underbrace{\left(1 - \sum_{k=1}^K x_{v_i, k} \right)^2}_{\text{unique vertex partition index}} + \delta_1 \sum_{k=1}^K \underbrace{\left(\frac{|V|}{K} - \sum_{i=1}^{|V|} x_{v_i, k} \right)^2}_{\text{balance partition vertices}}
 \end{aligned}$$



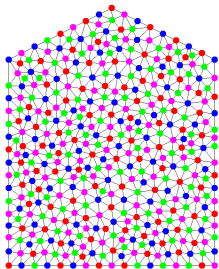
5-way graph partitioning (600 vertices)
110 cut edges, 118 cut edges (METIS)



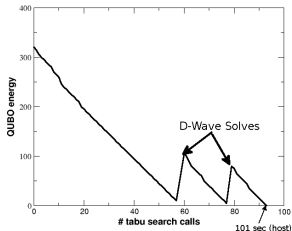
QBSolve performance
subproblem dim = 67

Graph 4-Coloring Problem

$$x^* = \underset{x}{\operatorname{argmin}} \delta_0 \underbrace{\sum_{i=1}^{|V|} \left(\sum_{k=1}^4 x_{v_i, k} - 1 \right)^2}_{\text{unique vertex color}} + \delta_1 \sum_{(v_i, v_j) \in E} \underbrace{\sum_{k=1}^4 x_{v_i, k} x_{v_j, k}}_{\text{distinct adjacent colors}}$$



4-colored planar graph (400 vertices)



QBSolve performance
subproblem dim = 67



Hybrid Quantum-Classical QUBO Solvers

QSPIN: A
High Level
Java API

Tim Barth

Outline

Overview of
the D-Wave
Quantum
Annealer

Applications
Software

QSPIN API

Hybrid
Quantum-
Classical
QUBO
Solvers

QUBO
Model Error

Further Ap-
plications

Looking
Forward

The development of hybrid quantum-classical QUBO solvers is just in its infancy. The topic is rapidly gaining interest in the technical community and several variants of *QBSolve* have already been suggested

- Alternative approaches to the selection of QUBO subproblems,
- Multi-level strategies for certain problems, e.g. graph partitioning,
- Hybrid simulated annealing-quantum annealing solvers.

In reality, the D-Wave annealer solves perturbed Ising and QUBO models

Ising spin model

$$s^{*,\delta} = \operatorname{argmin}_s \sum_{i=1}^N (h_i + \Delta h_i) s_i + \sum_{i=1}^N \sum_{j=i+1}^N s_i (J_{ij} + \Delta J_{ij}) s_j, \quad s_i \in \{-1, +1\},$$

QUBO

$$x^{*,\Delta} = \operatorname{argmin}_x \sum_{i=1}^N \sum_{j=i}^N x_i (Q_{ij} + \Delta Q_{ij}) x_j, \quad x_i \in \{0, 1\},$$

with Δ perturbations arising from

- weak 3-body qubit interactions and leakage of h biases,
- low frequency qubit flux noise,
- DAC quantization (perhaps as few as 6-8 bits),
- I/O system signal contamination,
- nonuniformity of qubits.

Unfortunately, these sources of model error can negatively impact optimization performance and complicate the determination of software parameters.

Model Error and Penalty Constraint Imposition

QSPIN: A
High Level
Java API

Tim Barth

Outline

Overview of
the D-Wave
Quantum
Annealer

Applications
Software

QSPIN API

Hybrid
Quantum-
Classical
QUBO
Solvers

QUBO
Model Error

Further Ap-
plications

Looking
Forward

Model problem

$$\begin{aligned} &\text{minimize} && (4x_1 - x_2 - 1)^2 \\ &\text{subject to} && x_1 + x_2 = 1 \end{aligned}$$

Penalty formulation, $\delta \in \mathbf{R}$

$$(x_1, x_2)^* = \underset{x}{\operatorname{argmin}} (4x_1 - x_2 - 1)^2 + \delta(x_1 + x_2 - 1)^2$$

Discrete optimization, $x_i \in \{0, 1\}$:

$$(x_1, x_2)^* = (1, 0) \quad \text{for } \delta > 3$$

Continuous optimization, $x_i \in \mathbf{R}$:

$$(x_1, x_2)^{*,\delta} = (\delta/(4 + \delta), 0), \quad (x_1, x_2)^* = \lim_{\delta \rightarrow \infty} (x_1, x_2)^{*,\delta} = (1, 0)$$

The D-Wave system automatically rescales QUBO problems, $Q_{ij} \in [-1, 1]$,

$$(x_1, x_2)^* = \underset{x}{\operatorname{argmin}} \frac{1}{\delta} (4x_1^2 + x_2^2) + (x_1 - 1)^2$$

Goldilocks predicament: Choosing δ too small (< 4) yields an incorrect minimum; choosing δ too large can potentially deteriorate the resolution of the objective function relative to model error.

Example: Traveling Salesman Problem

$$\begin{aligned}
 H(x) \equiv & \underbrace{\sum_{(v_i, v_j) \in E} \text{dist}(v_i, v_j) \sum_{k=1}^{|V|} (x_{v_i, k} x_{v_j, k+1} + x_{v_i, k+1} x_{v_j, k})}_{\text{minimize path distance}} + \underbrace{\delta_0 \sum_{i=1}^{|V|} \left(\sum_{k=1}^{|V|} x_{v_i, k} - 1 \right)^2}_{\text{vertex } v_i \text{ visited once}} \\
 & + \underbrace{\delta_1 \sum_{k=1}^{|V|} \left(\sum_{i=1}^{|V|} x_{v_i, k} - 1 \right)^2}_{\text{position } k \text{ occupied once}} + \underbrace{\delta_2 \sum_{(v_i, v_j) \notin E} \sum_{k=1}^{|V|} (x_{v_i, k} x_{v_j, k+1} + x_{v_i, k+1} x_{v_j, k})}_{\text{exclude path edges not in graph}} .
 \end{aligned}$$

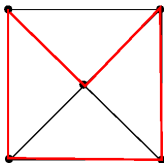
City sites

$\{(0, 0), (1, 0), (0, 1), (1, 1), (0.5, 0.51)\}$

3 tightly clustered (constraint satisfying) QUBO energies

$\{4.400, 4.412, 4.428\}$

D-Wave 2000Q sensitivity to penalty parameters (# reads = 10000, spin reversal = 10)



# successes	$\delta_0 = \delta_1 = \delta_2$
7	1
70	2
3280	4
250	6
65	10
43	100

Remarks:

- The recently installed D-Wave 2000Q has shown a significant improvement in model error.
- There is a technique for reducing large QUBO coefficients by spreading a large coefficient over auxiliary qubits. For example

$$\text{minimize } 4x_1^2$$

is mathematically equivalent to

$$\text{minimize } (x_1 + x_2)^2 \quad \text{subject to } x_1 - x_2 = 0$$

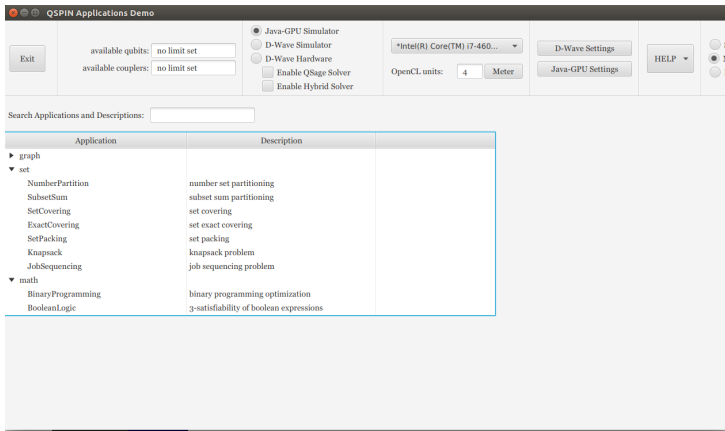
This technique is used heavily in the D-Wave graph embedding step.

- As an alternative, **QSPIN** includes a hybrid quantum-classical iterative Lagrange multiplier form of the QUBO problem

$$QUBO^{(n)}(x) = QUBO_0(x) + \lambda^{(n)} QUBO_1(x) + \delta^{(n)} QUBO_2(x)$$

where the coefficients $\lambda^{(n)}$ and $\delta^{(n)}$ are updated by the host application during $QUBO^{(n)}(x)$ iterations. By choosing $\lambda^{(n)}$ and $\delta^{(n)}$ correctly, one can reduce or eliminate the dependence on penalties (work in progress).

The **QSPIN-DEMO** provides a **suite** of NP-hard and NP-complete applications for the D-Wave system that maybe solved using the D-Wave QUBO solver for small problems and the hybrid *QBSolve* solver for large problems.



Application	Description
graph	
set	
NumberPartition	number set partitioning
SubsetSum	subset sum partitioning
SetCovering	set covering
ExactCovering	set exact covering
SetPacking	set packing
Knapsack	knapsack problem
JobSequencing	job sequencing problem
math	
BinaryProgramming	binary programming optimization
BooleanLogic	3-satisfiability of boolean expressions



Looking Forward

**QSPIN: A
High Level
Java API**

Tim Barth

Outline

Overview of
the D-Wave
Quantum
Annealer

Applications
Software

QSPIN API

Hybrid
Quantum-
Classical
QUBO
Solvers

QUBO
Model Error

Further Ap-
plications

**Looking
Forward**

The calculation of Ising spin ground states is a natural target problem for both quantum annealing and quantum logic computers. Consequently, algorithmic developments on the D-Wave system may translate to other quantum architectures.

Hybrid quantum-classical algorithms will undoubtedly play an important role in near term quantum computing devices constrained by relatively low qubit counts.

Most practical problems in discrete optimization contain constraints so advanced techniques for imposing them in quantum annealing and in general quantum models of computation will continue to be an active area of algorithm research.