Quantum Annealing Programming Techniques for Discrete Optimization Problems

OUTLINE

- Primer about quantum annealing and its challenges
- General programming/compiling/running applications
- The example of Job-Shop-Scheduling (JSP)
- Programming (mapping) techniques
- Compiling (embedding) techniques
- Running (annealing) techniques
- Annealing for problems of ASCR interests

http://www.nas.nasa.gov/quantum

Davide Venturelli, Alejandro Perdomo-Ortiz, Eleanor G. Rieffel, Bryan O’Gorman (NASA)
JSP In collaboration with: Dominic Marchand, Galo Rojo (1QBit)
PRIMER ABOUT QUANTUM ANNEALING

- Bottleneck is typically the minimum gap, but it is difficult to locate or compute, not necessarily at QPT.
- No proof of exponential speedup for easy to conceive Hamiltonians, mapping to universality unpractical.
- Research directly on this topic has only 15 years of history, with less than 300 theory papers published.

PRIMER ABOUT QUANTUM ANNEALING DEVICES (e.g. D-Wave Two)

- They do not support arbitrary problem Hamiltonians: D-Wave has an Ising model on a "chimera lattice"
- They do not support arbitrary drivings: D-Wave has a stoquastic transverse field
- They do not operate as a closed system at T=0, D-Wave qubit decoherence is ≈5ns and operating temperature ≈15mK
- Hamiltonian parameters are specified within a given precision, and they fluctuates over the annealing time, D-Wave has ≈5% of precision
- They do not support arbitrary schedules, D-Wave has a min 20μs annealing quench interlaced with the problem Hamiltonian energy
- They cannot encode arbitrary energies: D-Wave machine maximum energy is 3.2 Ghz

(Knysh et al. 2015)
GENERAL SCHEME FOR RUNNING APPLICATION PROBLEMS

1. **Mapping the problem into a suitable classical binary form (QUBO)**
   - (Rieffel et al. 2014)

2. **Pre-processing with polynomial algorithms**

3. **Pre-characterization (classical and quantum)**
   - Expectations through pre-characterization

4. **Error suppression**

5. **Optimal compilation (graph-minor embedding)**
   - Machine tuning

6. **Running with the optimal (potentially adaptive) run-strategy**
   - Error corrections
   - Post-processing

7. **Analysis of results**

8. **Embedding libraries**
   - (Venturelli et al. 2014)

   - (Perdomo-Ortiz et al. 2015)
THE PROBLEM: A SIMPLE 3X3 EXAMPLE

<table>
<thead>
<tr>
<th>1st operation</th>
<th>2nd operation</th>
<th>3rd operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>JOB 0</td>
<td>Machine 0 for 3t</td>
<td>Machine 1 for 2t</td>
</tr>
<tr>
<td>JOB 1</td>
<td>Machine 0 for 2t</td>
<td>Machine 2 for 1t</td>
</tr>
<tr>
<td>JOB 2</td>
<td>Machine 1 for 3t</td>
<td>Machine 2 for 3t</td>
</tr>
</tbody>
</table>

Feasible schedule with makespan 12

Optimal schedule with makespan 11

CLASSICAL INTRACTABILITY

<table>
<thead>
<tr>
<th>Size</th>
<th>Time</th>
<th>Best method</th>
</tr>
</thead>
<tbody>
<tr>
<td>5x5</td>
<td>0.015 seconds</td>
<td>Scip</td>
</tr>
<tr>
<td>10x10</td>
<td>2.75 seconds</td>
<td>Gurobi</td>
</tr>
<tr>
<td>15x15</td>
<td>2430 seconds</td>
<td>Cplex (40%)</td>
</tr>
</tbody>
</table>

(Arief et al. 2014) (Beck et al. 2014)

AERONAUTICS APPLICATIONS

(Banavar et al. 2007)
MAPPING INTO QUBO: THE TIME-SLICE APPROACH

\( x_{nmt} = 1 \) If the job \( n \) is executing on machine \( m \) at time \( t \)

\( x_{nmt} = 0 \) otherwise

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

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

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

Note: scheduling problems naturally quadratic, this is not always the case.
\( x_{nmt} = 1 \) If the job \( n \) is executing on machine \( m \) at time \( t \\
\( x_{nmt} = 0 \) otherwise

\[
\text{N M T bits required} \\
= -NM(M<\tau> + 1)
\]

Polynomial pre-processing:
Trivial bounds on heads/tails of jobs.

Ad the end the reading of the corresponding non-zero bits will immediately determine the schedule.

Note: \( \approx6000 \) logical qubits for intractable \( N=15 \) problems
COMPILING: GRAPH-MINOR EMBEDDING CHALLENGES

TOPOLOGICAL ASSIGNMENT OF CONNECTED COMPONENTS

Fully-connected substructures appear frequently in mappings and scale with the problem size.

INCREMENT IN PROBLEM MISSPECIFICATION DUE TO CONTROL PRECISION ERRORS

Resolving multiple values of parameters (including embedding) will result into loss of precision and problem misspecification.

PHYSICAL CHANGE IN THE PROBLEM SPECTRUM RESULTS IN CHANGE IN PERFORMANCE

Ground state $\Psi$ of Ideal Hamiltonian

Ground state $\Psi$ of ensemble of spoiled Hamiltonian

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

$N_{HW}^* \approx \frac{N_L^2}{k - 2}$

(Venturelli et al 2014)

(King et al 2015)
COMPILING: OPTIMAL PARAMETER SETTING

Early math considerations on parameter settings: the stronger the better

paramagnetic phase: \[ \Delta E \approx |B(t)-A(t)| \]

ferromagnetic phase: \[ \Delta E \approx \left| \frac{A(t)}{B(t)} \right|^N \]

Basic Method: search for the single optimal ferromagnetic embedding strength

(Rieffel et al. 2014)
THE TIME-SLOT DISCRIMINATION METHOD FOR TIME-SLICED DECISION PROBLEMS (PRECISION LIMITED)

Log₂(T) calls
Optimal solutions: <α energy
Suboptimal solutions: zero energy

Resolving k-time slots, given a maximum precision of ε for a problem of N jobs, with minimum logical penalty α:

ε ≈ \frac{\alpha}{N^k}

TUNING HARDWARE IMPERFECTIONS CAN HAVE A STRONG IMPACT.

We can pre-select the best gauge by either Hamiltonian learning or with the use of “performance estimators”

(Perdomo-Ortiz in prep.)

OPTIMIZING ALL RUNNING PARAMETERS (NOTABLY ANNEALING TIME): NECESSARY FOR BENCHMARKING AND FOR EVALUATING SPEEDUP.

(Ronnow 2014)
(Venturelli et al 2014)
OTHER TECHNIQUES AND PROBLEMS OF ASCR INTEREST: MATERIALS DESIGN

Toy example of binary alloy: Mg$_{17}$Al$_{12}$

$2^{29}$ possible configurations!

Experimental annealing can get stuck on metastable

**Objective:** Finding new different stable structures

**Cluster expansion approach:** approximate the Hamiltonian using information from subset of the unit cells that are solved with Density Functional Theory. Minimize the resulting classical energy functional with Monte Carlo methods.

$$E(\sigma) = V_0 + \sum_{i,j} V_{ij} \sigma_i \sigma_j + \sum_{i,j,k} V_{ijk} \sigma_i \sigma_j \sigma_k + \cdots$$

- Typical cluster expansion contains around $\approx 15$-20 V parameters and **3-local or maximum 4-local terms**.
- Limit of solvability for DFT $\approx$50 atoms. But long range interactions generate large non-local hamiltonians non easily minimizable by MonteCarlo. **Quantum Annealing can help?**

(Sanchez et al. 1984)

$$H_X(q_i, q_j, \tilde{q}_n) = \delta(3\tilde{q}_n + q_i q_j - 2q_i \tilde{q}_n - 2q_j \tilde{q}_n).$$

In general, we add 1 more qubit every 3-body term

(Babbush et al. 2012)
GENERAL SCHEME FOR RUNNING APPLICATION PROBLEMS

Expectations through Pre-characterization (classical and quantum)

Mapping the problem into a suitable classical binary form (QUBO)

Pre-processing with polynomial algorithms

Analysis of results

Optimal compilation (graph-minor embedding)

Running with the optimal (potentially adaptive) run-strategy

Embedding libraries

Error suppression

Error corrections

Post-processing

Machine tuning

(Perdomo-Ortiz et al. 2015)

(Venturelli et al. 2014)

(Rieffel et al. 2014)
SUMMARY

- We need to identify problems that are:
  - SMALL (Mapping+Embedding Bottleneck can allow programming in near-future machine)
  - HARD (Ideally on the verge of a phase transition of solvability)
  - RESILIENT TO MISSPECIFICATION (The Analog control errors are a precision bottleneck)
  Where classical algorithms have hard time finding even an approximate solution

- To design an annealing architecture supporting programming of discrete optimization problems we need to take advantage of the programming strategies that are appearing in the problems of interest, that will be naturally fitting some connectivity of the hardware graph and some precision requirements of the problem.

- To prepare a machine to solve a class of problems we want to:
  - CALIBRATE it properly for the classes of instances
  - PRE-CHARACTERIZE the classes of problems to gain expectations useful for the embedding and running
  Prepare EMBEDDING LIBRARIES that could fit the class with pre-determined parameters.