Quantum Annealing Applied to Optimization Problems in Radiation Medicine
Daryl P. Nazareth, PhD, DABR
Medical Physicist, Assistant Professor
Department of Radiation Medicine
We would like to thank D-Wave Systems for providing access to their hardware and for computational assistance. In particular, Bill Macready and Mani Ranjbar.

Jason Spaans
Tyler Paplham
Cancer is one of the leading causes of morbidity and mortality worldwide. In 2012 there were 14 million new cases of cancer. There were 8.2 million cancer-related deaths. Number of new cases expected to rise by 70% over next 2 decades (WHO).
Cancer in the US:
In 2016 there will be an estimated 1.69 million new cases of cancer.
There will be 596,000 cancer deaths.
39.6% of people will be diagnosed with cancer in their lifetimes.
Most common types: breast, lung, prostate, colon, bladder.
Cancer is treated using three methods:

Surgery
Cancer is treated using three methods:

Chemotherapy
Cancer is treated using three methods:

Radiation Therapy

This is the subject of our work
Radiation dose distribution

Absorbed dose measured in Gy (J/kg)
Calculated from well-known physics principles
Clinical calculations use FDA-approved software
CT scan determines electron density of each voxel of patient anatomy.

Allows dose calculation and anatomic structure identification (contouring).
Medical Linear Accelerator

Linear Accelerator Part 1
Linear Accelerator Part 2
Energy ~ 6 MeV
The PTV
### Beamlets

![Beamlets Diagram](image)

**x-jaw**

<table>
<thead>
<tr>
<th></th>
<th>$w_1$</th>
<th>$w_2$</th>
<th>$w_3$</th>
<th>$\ldots$</th>
</tr>
</thead>
<tbody>
<tr>
<td>y-jaw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**$w_{beam}$**

$$w_{beam} = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{pmatrix}$$
Beams

$w = \begin{pmatrix}
  w_{beam\,1} \\
  w_{beam\,2} \\
  w_{beam\,3} \\
  w_{beam\,4} \\
  w_{beam\,5}
\end{pmatrix}$
A Dose-Volume Histogram (DVH) is a graphical representation of the percentage of dose received by a portion of the volume.

For a given treatment plan, the PTV and each organ has an associated DVH.

Is critical to defining the objective function.
Approx. 15% of the PTV receives slightly less than the prescribed dose.

Approx. 85% of the PTV receives slightly more than the prescribed dose.
The lowest (min) dose any portion of the bladder receives is 4%.

40% of the total volume is receiving more than 50% of the prescription dose.

The highest (max) dose any portion of the bladder receives is 102%.
Fem. Heads DVH

*Dose distribution not symmetric
In this case, 25% of the total volume of the rectum receives more than 50% of the prescription dose.
The Objective Function

\[ F(\mathbf{w}) = \alpha (P_v - D_v(\mathbf{w}))^2 + \sum_i \sum_j \beta_i (\max[0, D_{ij}(\mathbf{w}) - C_{ij}])^2 \]

\( \mathbf{w} \) is a vector of beamlet weights or intensities
Minimizing \( F(\mathbf{w}) \) results in optimal IMRT treatment plan
The Target

\[ F(w) = \alpha(P_v - D_v(w))^2 + \sum \sum \beta_i (\max(0, D_{ij}(w) - C_{ij}))^2 \]

\( \alpha = \) Priority of target dose (How important is it that this dose is fully administered?)

\( P_v = \) Dose prescribed to a given volume, \( v \), of the target

\( D_v(w) = \) Dose actually received by volume \( v \) for weight vector \( w \)
In this case, if we want 95% of the volume to receive 80 Gy, $P_v = 80$
\[ \alpha \left( P_v - D_v(w) \right)^2 \]

This is the \( D_v(w) \), what the PTV is actually receiving.

This distance is the penalty created by \( (P_v - D_v(w))^2 \).

This is the \( P_v \) at 80 Gy.
In clinical terms, this ensures that the dose received by the target is as close as possible to the dose prescribed.
Organs at Risk (OARs)

\[ F(w) = \alpha \left( P_v - D_v(w) \right)^2 + \sum_i \sum_j \beta_i \left( \max[0, D_{ij}(w) - C_{ij}] \right)^2 \]

\[ \sum_i = \text{Sum over each OAR, eg. bladder} = 1 \]
\[ \sum_j = \text{Sum over multiple objectives for a given OAR} \]
\[ \beta_i = \text{Priority of OAR} \]
\[ C_{ij} = \text{Objective dose} \]
\[ D_{ij} = \text{Actual dose received by OAR} \]
Organs at Risk (OARs)

\[ \sum_i \sum_j \beta_i (\max[0, D_{ij} - C_{ij}])^2 \]

Again, let’s take a closer look
Organs at Risk (OARs)

\[
\sum_{i} \sum_{j} \beta_i (\max[0, D_{ij}(w) - C_{ij}])^2
\]

This is the \(D_{ij}(w)\)

\(C_{ij} = 40\text{ Gy at 40\% volume}\)
Organs at Risk (OARs)

\[ \sum_i \sum_j \beta_i (\max[0, D_{ij}(w) - C_{ij}])^2 \]

Since \( D_{ij}(w) > C_{ij} \), a positive penalty results.

Penalty

\( C_{ij} \)

\( D_{ij}(w) \)
There is no reward for $D_{ij}(w) < C_{ij}$ because there is negligible clinical benefit to administering less than the objective dose to the OAR.
The Process

Initial beamlet weight vector \( \mathbf{w} \) is chosen

Beamlet results in dose for each voxel

Beamlet dose matrices are added to create a beam dose matrix

DVHs are created and \( F(w) \) calculated

Beam dose matrices are added to create total dose matrix

\[
\mathbf{w}_n = \begin{pmatrix} d_1 \\ \vdots \\ d_n \end{pmatrix}
\]

\[
\mathbf{w}_n = \begin{pmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_n \end{pmatrix}
\]

\[
\mathbf{w}_{total} = \begin{pmatrix} \mathbf{w}_beam 1 \\ \vdots \\ \mathbf{w}_beam n \end{pmatrix}
\]
History of collaboration:

- Contacted D-Wave in 2009, put in touch with Bill
- Initially decided QA could not support IMRT optimization
- Visited lab in Burnaby in 2011 and revisited problem
- Worked remotely using Vesuvius chip and “Black Box” algorithm, 2012-2014
First application of quantum annealing to IMRT beamlet intensity optimization

Daryl P Nazareth and Jason D Spaans

Department of Radiation Medicine, Roswell Park Cancer Institute,
Buffalo NY 14263, US

E-mail: daryl.nazareth@roswellpark.org

Received 3 November 2014, revised 19 March 2015
Accepted for publication 2 April 2015
Published 1 May 2015

Abstract
Optimization methods are critical to radiation therapy. A new technology, quantum annealing (QA), employs novel hardware and software techniques to address various discrete optimization problems in many fields. We report on the first application of quantum annealing to the process of beamlet intensity optimization for IMRT.

We apply recently-developed hardware which natively exploits quantum mechanical effects for improved optimization. The new algorithm, called
Vesuvius chip supported ~ 512 qubits
Weight variables discretized to 7-digit binary variables
Therefore, 70 beamlet weights (non-negative, continuous) were included
Actual clinical case would require 600-1000 beamlet weights
Conventional simulated annealing (SA) features:

– Minimize function that is combo of original plus entropy
– Entropy is weighted by temp parameter $T$
– $T$ is slowly reduced from large values (search space exploration) to 0 (solution)
– Can attain global minimum if cooling slow enough (but exponentially long)
Three methods compared:

– Quantum annealing
– Simulated annealing
– Tabu search: popular heuristic used in combinatorial optimization

Methods were used to determine beamlet weights for two prostate bed cases
Each was run for $10^7$ function evaluations and compared for speed and score
<table>
<thead>
<tr>
<th>Patient</th>
<th>Method</th>
<th>Evals/sec/core</th>
<th>Final Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>QA</td>
<td>9.3</td>
<td>16.9</td>
</tr>
<tr>
<td>1</td>
<td>SA</td>
<td>9.6</td>
<td>6.7</td>
</tr>
<tr>
<td>1</td>
<td>Tabu</td>
<td>4.3</td>
<td>10.0</td>
</tr>
<tr>
<td>2</td>
<td>QA</td>
<td>15.4</td>
<td>70.7</td>
</tr>
<tr>
<td>2</td>
<td>SA</td>
<td>17.4</td>
<td>22.9</td>
</tr>
<tr>
<td>2</td>
<td>Tabu</td>
<td>6.3</td>
<td>120.0</td>
</tr>
</tbody>
</table>
DVHs

QA (solid) and SA (dashed) for Patient 1
QA (solid) and Tabu (dashed) for Patient 2
<table>
<thead>
<tr>
<th>Patient</th>
<th>Method</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>QA</td>
<td>1.00</td>
</tr>
<tr>
<td>1</td>
<td>SA</td>
<td>2.89</td>
</tr>
<tr>
<td>1</td>
<td>Tabu</td>
<td>3.23</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Patient</th>
<th>Method</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>QA</td>
<td>1.00</td>
</tr>
<tr>
<td>2</td>
<td>SA</td>
<td>2.67</td>
</tr>
<tr>
<td>2</td>
<td>Tabu</td>
<td>3.67</td>
</tr>
</tbody>
</table>
SA produced best score for both patients
QA was second, third
QA was fastest, by factors of 2.7 – 3.7
DVHs were compared and similar
Plans were not clinically viable due to small number of beamlets
Future Work – VMAT

VMAT
VMAT Treatment
Conclusions

This is first application of QA to IMRT optimization
Compared QA to SA and Tabu
Evaluated using clinical DVH-based objective functions
QA hardware will rapidly scale in size
Further research on application of QA to VMAT may offer promising returns
Thank You!
Thoughts and Experiences

Five years of Quantum Programming
Pre-history: Conversation with Geordie Rose

• Sometime in the fall of 2011, after flubbing my first phone interview with GR, I was granted a second chance.

• I remember two questions that he asked:

  What is a support vector machine?

  What are the odds that a book appearing on the New York Times best seller list in the next ten years will have been written by a machine?
Day 14: First things first

In Burnaby

Have downloaded emacs 23.3 onto the laptop.

Have downloaded the Matlab pack.

The URL for the machine is:

http://appsqa.internal.dwavesys.com

The Solver API is at:

http://appsqa.internal.dwavesys.com/sapi

The QP API (QP = Quantum Processor) is at:

http://appsqa.internal.dwavesys.com/qpapi
Day 15: Early software architecture

D-Wave Software Stack

- Client: MATLAB Software Solver
- Client: MATLAB Software Solver
- Client: MATLAB Software Solver
- Client: Python Software Solver
- Client: MATLAB Software Solver

Local solvers

- Implements SAPI & QPAPI
- Web services

Remote solvers

Software Solver
Software Solver
D-Wave 1 Rainier
D-Wave 1 Rainier
D-Wave 1 Rainier
Day 15: Chimera, circa 2012
Day 21 — 22: Hadamard matrices, manually

4x4 Hadamard: QUBO interactions

- Problem variables: 4 vars
- Ancillary variables: 4 vars
- Quadratic interactions: 16 terms, 6 terms
- Totals: Variables: 4x10 = 40
  Interactions: 18x16 + 10x6 = 348

Block structure of QUBO

Maximum number of interactions:
40 choose 2 = 780
Actual number of interactions: 348
Relative density: 348/780 = 45%
Day 23: Depression sets in
Day 24: BlackBox restores hope

Internal behavior of the code is unknown

- Sorting networks
- Ramsey theory
- Graph isomorphism
- Travelling salesman problem
Day 210 – 212: Learning from the Master

Conference on Uncertainty in Artificial Intelligence
Catalina Island, United States
August 15-17, 2012

uai2012

me  Bill
Day 250: Polytopes for Adiabatic QC

Chimeratope(L,M,N) – or – CH(L,M,N)

- L = half number of spins in the unit cell
- M = number of rows
- N = number of columns
## Day 270: Automorphism Groups of Chimera

<table>
<thead>
<tr>
<th>tiling</th>
<th>size</th>
<th>factorization</th>
<th>group</th>
</tr>
</thead>
<tbody>
<tr>
<td>1x1</td>
<td>1152</td>
<td>$2^7 \times 3^2$</td>
<td>$S(2) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>1x2</td>
<td>27648</td>
<td>$2^{10} \times 3^3$</td>
<td>$S(2) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>1x3</td>
<td>663552</td>
<td>$2^{13} \times 3^4$</td>
<td>$S(2) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>1x4</td>
<td>15925248</td>
<td>$2^{16} \times 3^5$</td>
<td>$S(2) \times S(4) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>2x1</td>
<td>27648</td>
<td>$2^{10} \times 3^3$</td>
<td>$S(2) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>2x2</td>
<td>2654208</td>
<td>$2^{15} \times 3^4$</td>
<td>$D(4) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>2x3</td>
<td>31850496</td>
<td>$2^{17} \times 3^5$</td>
<td>$S(2) \times S(2) \times S(4) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>2x4</td>
<td>764411904</td>
<td>$2^{20} \times 3^6$</td>
<td>$S(2) \times S(2) \times S(4) \times S(4) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>3x1</td>
<td>663552</td>
<td>$2^{13} \times 3^4$</td>
<td>$S(2) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>3x2</td>
<td>31850496</td>
<td>$2^{17} \times 3^5$</td>
<td>$S(2) \times S(2) \times S(4) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>3x3</td>
<td>1528823808</td>
<td>$2^{21} \times 3^6$</td>
<td>$D(4) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>3x4</td>
<td>1.6345885696e10</td>
<td>$2^{23} \times 3^7$</td>
<td>$S(2) \times S(2) \times S(4) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>4x1</td>
<td>15925248</td>
<td>$2^{16} \times 3^5$</td>
<td>$S(2) \times S(2) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>4x2</td>
<td>764411904</td>
<td>$2^{20} \times 3^6$</td>
<td>$S(2) \times S(2) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>4x3</td>
<td>1.6345885696e10</td>
<td>$2^{23} \times 3^7$</td>
<td>$S(2) \times S(2) \times S(4) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
<tr>
<td>4x4</td>
<td>8.8060251341e11</td>
<td>$2^{27} \times 3^8$</td>
<td>$D(4) \times S(4) \times S(4) \times S(4) \times S(4) \times S(4) \times S(4)$</td>
</tr>
</tbody>
</table>
Day 574: Conquering the Unit Cell

Mathieu Dutour Sikiric
Day 370: Vesuvius
Day 395: Big B
Day 455 − ∞: Training
Day 720: Washington
Day −8760: Archaeology of Map Coloring

Neural Network Algorithm for an NP-Complete Problem: Map and Graph Coloring

Edward Danning Dahl, L-298
Lawrence Livermore National Laboratory
P. O. Box 808
Livermore, CA 94550

1. Introduction

The four color conjecture states that any map drawn on a plane or sphere can be colored with four colors so that no two countries which share a border have the same color. The proof of this appealing conjecture took more than one hundred years, and the order of $10^{20}$ computer operations. Unfortunately, the proof of this conjecture is not constructive. In fact, there is no algorithm which is guaranteed to color an arbitrary planar map without essentially resorting to exhaustive search. The absence of such an algorithm extends to a class of problems which generalizes the task of coloring a planar map with four colors. In the general problem, the map is not necessarily planar, but may lie on a more complicated surface such as a torus with several holes. To complete the specification of the general problem, pick a positive integer $k$ which sets the number of colors available for using in coloring the map. The problem is to decide whether a $k$-coloring of the map exists: that is, an assignment of colors to regions on the map so that regions sharing a border receive different colors.

Since 1971 and the proof of Cook’s Theorem, the notion of NP-Completeness has been made precise. For a problem to be NP-Complete it is required that one be able to find the solution to the problem in an amount of time polynomial in the problem size, provided one uses the very special model of computation represented by a non-deterministic Turing Machine. Secondly, an NP-Complete problem is at least as hard as any problem which satisfies the first criteria. Deciding whether a non-planar map is $k$-colorable is an NP-Complete Problem.

John Hopfield has demonstrated that a Neural Network can provide a heuristic technique for solving the Traveling Salesman Problem (TSP). In one version of the Traveling Salesman Problem one is given some number of cities, the pairwise distances between all the cities, and some fixed length $L$. One must then decide whether there is some tour through the cities which visits each one once, and has total length less than the bound $L$. This problem is NP-Complete, and that equivalently difficult to the Map $k$-colorability problem.

It is natural to wonder whether a Neural Network solution exists for the Map $k$-colorability problem, given that both it and the TSP are equivalent hard and that a Neural Net solution exists for the latter problem. We demonstrate that a Neural Network solution does exist for the Map $k$-colorability problem. The connectivity of the neurons in the net follows simply from the connectivity of the regions in the map. The dynamics of the

---

Day 580: Map coloring on a quantum computer

Canada regions/shell script

<table>
<thead>
<tr>
<th># of colors</th>
<th>Needle</th>
<th>Haystack</th>
<th>N/H</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1728</td>
<td>$3^{13} = 1.6 \times 10^6$</td>
<td>0.0011</td>
</tr>
<tr>
<td>4</td>
<td>653184</td>
<td>$4^{13} = 6.7 \times 10^7$</td>
<td>0.0097</td>
</tr>
</tbody>
</table>
Day 610: Static decomposition

<table>
<thead>
<tr>
<th># of colors</th>
<th>Needle</th>
<th>Haystack</th>
<th>N/H</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0</td>
<td>$3^{49} = 2.4 \times 10^{23}$</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>25623183458304</td>
<td>$4^{49} = 3.2 \times 10^{29}$</td>
<td>$8 \times 10^{-17}$</td>
</tr>
</tbody>
</table>
Day 720: Dynamic decomposition / qbsolv

254 counties in Texas
Day 900: Dynamic decomposition / qbsolv

3108 US counties
Day 735: DEQO – the predecessor of ToQ

Deqo: A Direct Embedding Quantum Optimizer

E. D. Dahl, D-Wave Systems

January 24, 2014

Abstract

Deqo is a prototype compiler for the D-Wave System. It implements a simple language designed for constraint satisfaction problems (CSP) defined over both boolean and small integer variables. Deqo assumes its CSP has locality: in other words, it consists of a set of constraints each of which involves a small number of available variables. The compilation model proceeds by applying a sequence of transformations which move the problem representation closer to the native quantum machine instruction (QMI) of the D-Wave System. Each transformation is described in the context of a simple example. We conclude by discussing possible extensions to the compilation model.
Day 950: Map coloring made easy / ToQ

Snippet (28 of 596 LOC)

C

void setup_unit_cell(int row, int col)
{
    int i, j;
    if (cell_region[row][col] == UNDEF)
        return;
/* STEP 1: turn on one of C qubits */
for (i=0; i<C; ++i)
{
    weight[DW_QUBIT(row,col,'L',i)] += -0.5;
    weight[DW_QUBIT(row,col,'R',i)] += -0.5;
}
for (i=0; i<C; ++i)
    for (j=0; j<C; ++j)
    if (i != j)
        strength[DW_INTRACELL_COUPLER(row,col,i,j)] += 1;
/* STEP 2: create chains */
for (i=0; i<C; ++i)
{
    weight[DW_QUBIT(row,col,'L',i)] += 1;
    weight[DW_QUBIT(row,col,'R',i)] += 1;
    strength[DW_INTRACELL_COUPLER(row,col,i,i)] += -2;
}

ToQ

mbool: 1, 4, @AB
mbool: 1, 4, @BC
mbool: 1, 4, @MB
mbool: 1, 4, @NB
mbool: 1, 4, @NL
mbool: 1, 4, @NS
mbool: 1, 4, @NT
mbool: 1, 4, @NU
mbool: 1, 4, @ON
mbool: 1, 4, @QC
mbool: 1, 4, @SK
mbool: 1, 4, @YT

assert: @AB != @BC
assert: @AB != @NT
assert: @AB != @SK
assert: @BC != @NT
assert: @BC != @YT
assert: @MB != @NU
assert: @MB != @ON
assert: @MB != @SK
assert: @MB != @NS
assert: @MB != @QS
assert: @NL != @QC
assert: @NT != @NU
assert: @NT != @SK
assert: @NT != @SK
assert: @NT != @YT
assert: @NU != @QC

entire program

QMI:
weights
strengths

© 2016 D-Wave Systems Inc. All Rights Reserved
Day 1165: Sudoku

```
+---+---+---+
| 4 | 7 |   |
| 2 | 3 | 1 |
| 5 | 1 | 6 |
+---+---+---+
| 8 |   | 6 |
| 3 | 8 |   |
| 9 | 2 | 3 |
+---+---+---+
```

```
#!/bin/bash

sudoku=$1

if [[ -z $sudoku ]]; then
  echo "usage: driver.bash <sudoku-file>"
  exit 1
fi

echo "*************** Create QUBO for blank grid ***************"
s1.bash > s1.out

echo "*************** Identify fixed variables *******************"
s2.bash < $sudoku > s2.out

echo "*************** Compute reduced QUBO *********************"
s3.bash > s3.out

echo "*************** Identify free variables : Enumerate variables ***********"
s4.bash < s3.out > s4.out
```

```
Day $\pi \times 365$: The Virtuous Cycle

- USERS
- TOOLS
- ALGORITHMS
- PROBLEMS

thought
Day 1642: Virtual Full Yield

Submit Problem
## Any Day, All Day

<table>
<thead>
<tr>
<th>Problem</th>
<th>Technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hadamard matrices</td>
<td>Direct embedding</td>
</tr>
<tr>
<td>Ramsey lower bounds &amp; more</td>
<td>BlackBox (Qsage)</td>
</tr>
<tr>
<td>Travelling Salesman Problem</td>
<td>BlackBox, QUBO, Parallel Update</td>
</tr>
<tr>
<td>Quadratic Assignment Problem</td>
<td>BlackBox, QUBO</td>
</tr>
<tr>
<td>Cyclic Ordering</td>
<td>Blackbox / Sorting network</td>
</tr>
<tr>
<td>Graph Isomorphism</td>
<td>Blackbox / Sorting network</td>
</tr>
<tr>
<td>Map Coloring</td>
<td>Various</td>
</tr>
<tr>
<td>Hello World</td>
<td>SAPI</td>
</tr>
<tr>
<td>Sudoku</td>
<td>qbsolv</td>
</tr>
<tr>
<td>Factoring</td>
<td>qbsolv</td>
</tr>
</tbody>
</table>
Perspectives about what might work best

STONEBRAKER ALGORITHM

Code section 2.

Until (it works) {
    Come up with a new idea;
    Prototype it with the help of superb computer scientists;
    Persevere, fixing whatever problems come up; always remembering that it is never too late to throw everything away;
}

Likely copyright violation DOI:10.1145/2869958

The D-Wave Trinity

Optimization & Constraint Satisfaction Problems
Machine Learning
Sampling
Day 1721: Today

- software abstractions
- embedding
- decomposition
- constraint compilation
- post-processing
- basic physics
- scaling characterization
- machine learning
- hardware improvements
- error correction
- sampling