VICTORIA GOLIBER
SENIOR TECHNICAL ANALYST
vgoliber@dwavesys.com
<table>
<thead>
<tr>
<th>Time</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:30 - 10:45</td>
<td>Constrained Quadratic Models / Hybrid Solvers</td>
</tr>
<tr>
<td></td>
<td>Break</td>
</tr>
<tr>
<td>11:00-12:15</td>
<td>Binary Quadratic Models / QPU Solvers</td>
</tr>
<tr>
<td></td>
<td>Break</td>
</tr>
<tr>
<td>12:30-13:00</td>
<td>Use cases and wrap up</td>
</tr>
<tr>
<td>13:00-13:30</td>
<td>Q&amp;A</td>
</tr>
</tbody>
</table>
Constrained Quadratic Models
POWERFUL HYBRID SOLVERS

CONstrained QUADRATIC MODEL SOLVER
- Up to 500,000 variables and 100,000 constraints
- Provide separate objective and constraints
- Inequality & equality constraints

BIrinary QUADRATIC MODEL SOLVER
- Up to 1,000,000 binary variables
- Full flexibility in problem formulation

DiSCRETE QUADRATIC MODEL SOLVER
- Enables optimization with option selection: e.g., Choose one of 11, 19, 29
- Accepts up to 5,000 discrete variables
CQM WITH CONTINUOUS VARIABLES

✓ EXPANDS HYBRID SOLVER PORTFOLIO
✓ ALLOWS FOR MORE NATIVE REPRESENTATIONS
✓ UNLOCKS LARGER APPLICATION PROBLEMS

FEATURES:
• Binary, integer and real/continuous variables
• Linear and quadratic terms
• Up to 100,000 constraints
• Inequality & equality constraints
3D BIN PACKING

GOAL: Determine the optimal placement and orientation of boxes within containers

CHALLENGES:

• Which container to put each box in?
• Where to place each box in the container?
• What orientation for each box?

This is a challenging, NP-hard problem.
TECHNICAL DESCRIPTION

Objective:

• Stack boxes from the floor upwards
  • Minimize average height
  • Minimize total height

• Minimize number of bins used

Constraints:

• Choose an orientation for each box

• Put each box in a container
  • Assign to one from a list
  • Ensure dimensions are within boundaries
  • Ensure boxes don't overlap

• Place boxes into containers in order
Quantum Annealing

Classical solutions are sequential
• Traverse landscape to find low points
• Must climb out of any local minima to find global solution
• Energy and time intensive
• Certain problems intractable

Quantum annealing is simultaneous
• Can tunnel through the hills
• Entanglement and superposition accelerate discovery of deep valleys
• Fast and energy efficient
• Programmable, no quantum expertise needed, large scale systems deployed today
QUADRATIC MODELS

\[
\text{Obj}(a_i, b_{ij}; q_i) = \sum_i a_i q_i + \sum_{i<j} b_{ij} q_i q_j
\]

Control bias on qubits

Control bias on couplers
CQM DEVELOPMENT PROCESS

1. Write out the objective and constraints in your problem domain

2. Define the binary and/or integer variables for your problem

3. Convert the objective and constraints to math statements with binary and/or integer variables

4. Build the CQM model in Ocean from the individual objectives and constraints
The knapsack problem is a well-known optimization problem. It is encountered, for example, in packing shipping containers. A shipping container has a weight capacity which it can hold. Given a collection of items to be shipped, where each item has a value and a weight, the problem is to select the optimal items to pack in the shipping container.

This optimization problem can be defined as an objective with a constraint:

- **Objective**: Maximize freight value (sum of values of the selected items)
- **Constraint**: Total freight weight (sum of weights of the selected items) must be less than or equal to the container's capacity
**KNAPSACK – FORMULATION**

**Variables**
Binary variable for each item

**Objective**
Maximize the value of the selected items

**Constraints**
Weight of all the selected items cannot exceed the knapsack's capacity

A maximum of 2 items can fit in the knapsack

\[
\sum_{i=0}^{N-1} v_i x_i \\
\sum_{i=0}^{N-1} w_i x_i \leq W \\
\sum_{i=0}^{N-1} x_i \leq 2
\]
from dimod import ConstrainedQuadraticModel, Binary, quicksum
from dwave.system import LeapHybridCQMSampler

values = [34, 25, 78, 21, 64]
weights = [3, 5, 9, 4, 2]
W = 10
n = len(values)

# Create the binary variables
x = [Binary(i) for i in range(n)]

# Construct the CQM

cqm = ConstrainedQuadraticModel()

# Add the objective

cqm.set_objective(quicksum(-values[i]*x[i] for i in range(n)))

# Add the two constraints

cqm.add_constraint(quicksum(weights[i]*x[i] for i in range(n)) <= W, label='max weight')
cqm.add_constraint(quicksum(x[i] for i in range(n)) <= 2, label='max items')

# Submit to the CQM sampler
sampler = LeapHybridCQMSampler()
sampleset = sampler.sample_cqm(cqm)

print("\nFull sample set:")
print(sampleset)
## KNAPSACK – SAMPLE SET

<table>
<thead>
<tr>
<th>Variable values</th>
<th>Energy of the objective function</th>
<th>Constraint satisfaction array</th>
<th>Solution feasibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4</td>
<td>energy  num_oc.</td>
<td>is_sat. is_fea. object.</td>
<td></td>
</tr>
<tr>
<td>1 0.0 0.0 1.0 0.0 1.0</td>
<td>-142.0 3</td>
<td>arra... False -142.0</td>
<td></td>
</tr>
<tr>
<td>2 1.0 0.0 0.0 0.0 1.0</td>
<td>-98.0 36</td>
<td>arra... True -98.0</td>
<td></td>
</tr>
<tr>
<td>0 0.0 0.0 1.0 0.0 0.0</td>
<td>-78.0 13</td>
<td>arra... True -78.0</td>
<td></td>
</tr>
<tr>
<td>3 0.0 0.0 0.0 0.0 1.0</td>
<td>-64.0 1</td>
<td>arra... True -64.0</td>
<td></td>
</tr>
</tbody>
</table>

['INTEGER', 4 rows, 53 samples, 5 variables]
# KNAPSACK – SAMPLE SET

<p>| | | | | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>energy</td>
<td>num_oc.</td>
<td>is_sat.</td>
<td>is_fea.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>1.0</td>
<td>-142.0</td>
<td>16 arra...</td>
<td>False</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>-98.0</td>
<td>21 arra...</td>
<td>True</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td>-64.0</td>
<td>1 arra...</td>
<td>True</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

['INTEGER', 3 rows, 38 samples, 5 variables]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>items</td>
<td>Total weight</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
<td>142</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>98</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>64</td>
</tr>
</tbody>
</table>
feasible_sampleset = sampleset.filter(lambda row: row.is_feasible)
Questions?
Binary Quadratic Models
POWERFUL QUANTUM COMPUTERS BUILT FOR BUSINESS

D-WAVE advantage

SUPPORTS HYBRID APPLICATIONS OF REAL-WORLD SIZE
Up to 1 million variables

ANNEALING QUANTUM PROCESSOR DESIGN
5,000+ qubits

ONGOING INCREASES IN COHERENCE & CONNECTIVITY

NOW SYSTEMS LOCATED IN US, EUROPE & CANADA
D-Wave 2000Q QPU
A Chimera Unit Cell

The D-Wave QPU is built from tiny loops of metal, each of which is one qubit.
The Chimera Graph

The underlying architecture of the QPU looks like a graph.

In the 2000Q D-Wave System the native QPU topology is a *Chimera graph*. 
The Pegasus Graph

In the Advantage system the native QPU topology is a Pegasus graph.
Quadratic Models

\[ \text{Obj}(a_i, b_{ij}; q_i) = \sum_i a_i q_i + \sum_{i<j} b_{ij} q_i q_j \]

- Control bias on qubits
- Control bias on couplers
We start with a problem in the real world (problem domain)

A Binary Quadratic Model (BQM) is a minimization function that incorporates:

- **Constraints:** The rules that we must follow
- **Objective:** What we want to minimize

**General form:**

\[ BQM = \min \text{ (objective)} + \gamma \text{ (constraints)} \]
BQM Development Process

1. Write out your objective and constraints in your problem domain

2. Convert your objective and constraints to math statements with binary variables

3. Modify your objective and constraints for the BQM form
   - Objective is a minimization function
   - Constraints are either:
     - Linear/quadratic equalities/inequalities, or
     - Satisfied at their minimum values

4. Combine your objective and constraints
Program Structure

- Establish sampler
- Define QM
- Send info to sampler
- Evaluate response
Defining a BQM

Initialize:  
\[
bqm = \text{BinaryQuadraticModel('BINARY')} \]

To add \(x + 5y + 8xy\) to our BQM:

Terms:
\[
bqm.\text{add_variables_from}([('x', 1), ('y', 5)])
bqm.\text{add_interactions_from}([('x', 'y', 8)])
\]

Symbolic:
\[
x = \text{dimod.Binary('x')}
y = \text{dimod.Binary('y')}
bqm = x + 5*y + 8*x*y
\]
Sending our problem to the sampler

BQM: \[
\text{sampleset} = \text{Sampler.sample}(\ bqm) \]

Additional parameters are available depending on what sampler you have decided to use.
What does a sampleset look like?

<table>
<thead>
<tr>
<th>w</th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>energy</th>
<th>num_oc.</th>
<th>chain_</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.0</td>
<td>18</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0.0</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.0</td>
<td>31</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.0</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.0</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0.0</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.0</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0.0</td>
<td>22</td>
</tr>
</tbody>
</table>

['BINARY', 8 rows, 100 samples, 4 variables]
Example

Suppose we have this list of projects we might work on, and each project has the associated profit.

How should we formulate a BQM to maximize profit?

What would the expected solution be?

How would we write a program to get this solution?

<table>
<thead>
<tr>
<th>Project</th>
<th>Profit</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>20</td>
</tr>
<tr>
<td>P2</td>
<td>18</td>
</tr>
<tr>
<td>P3</td>
<td>22</td>
</tr>
<tr>
<td>P4</td>
<td>26</td>
</tr>
<tr>
<td>P5</td>
<td>21</td>
</tr>
</tbody>
</table>
Now suppose that we:
1. Want to maximize profit, and
2. Must choose either P1 or P2, but not both.

How would we formulate this constraint?

How can we add this to our program?

<table>
<thead>
<tr>
<th>Project</th>
<th>Profit</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>20</td>
</tr>
<tr>
<td>P2</td>
<td>18</td>
</tr>
<tr>
<td>P3</td>
<td>22</td>
</tr>
<tr>
<td>P4</td>
<td>26</td>
</tr>
<tr>
<td>P5</td>
<td>21</td>
</tr>
</tbody>
</table>
Equality Constraints

Constraint:
\[(\sum a_i x_i) + k = 0\]

1. Build a BQM object

2. Use `bqm.add_linear_equality_constraint(...)` to add the constraint
   Parameters:
   • Terms: \[\{(\text{var}1, \text{coeff}1), (\text{var}2, \text{coeff}2), \ldots\}\]
   • Constant: \(k\) in above equation
   • Lagrange_multiplier: weight for constraint

Example:
Choose 2 numbers that sum to 5 from the set \(A = \{1, 2, 3, 4\}\).

```python
from dimod import BinaryQuadraticModel
bqm = BinaryQuadraticModel('BINARY')
bqm.add_linear_equality_constraint([(1,1), (2,1), (3,1), (4,1)],
        constant = -2,
        lagrange_multiplier = 1)
bqm.add_linear_equality_constraint([(1,1), (2,2), (3,3), (4,4)],
        constant = -5,
        lagrange_multiplier = 1)
```
Questions?
Industry Use Cases
APPLICATIONS ACROSS KEY VERTICALS

**LOGISTICS**
- Shipping container logistics
- Employee scheduling
- Farm to market food delivery
- Last mile vehicle routing

**PHARMA**
- Protein folding
- Clinical trials
- Drug discovery

**FINANCE**
- Portfolio risk reduction and return optimization
- Marketing campaign optimization
- Fraud detection
REDUCE WASTE IN THE AUTOMOTIVE SUPPLY CHAIN

Volkswagen

“By continuing to research and develop these types of algorithms, we hope to have a significant impact on Volkswagen’s core business throughout multiple units. This application has immediate, real-world implications for production and logistics.”

—VOLKSWAGEN QUANTUM COMPUTING RESEARCHER SHEIR YARKONI

80% REDUCTION IN WASTE
A QUANTUM SOLUTION FOR PREDICTING PROFITABILITY

These results are exciting because they really show a commercially valuable application of quantum computing today.

—SAM MUGEL, CTO, MULTIVERSE

Maximum value at lowest risk
With $10^{382}$ possible portfolios, the hybrid quantum/classical system made short work of it.

TIME TO SOLVE
D-Wave: 171 seconds
Tensor Networks: 1 day
Other classical solvers: No solution found
BUSINESS-CRITICAL GROCERY TASKS IN MINUTES INSTEAD OF HOURS

“What Advantage gives us, is the ability to seamlessly integrate quantum into our business problems. We've been able to decrease the amount of time to get a result from 25 hours down to seconds.”

—ANDREW DONAHER, VP DIGITAL & ANALYTICS, SAVE-ON-FOODS
LOGISTICS OPTIMIZATION AT PORT OF LA WITH QUANTUM COMPUTING

SAVANTX

D-Wave’s quantum system is used as part of the SavantX HONE optimization engine at the Port of Los Angeles. The goal is to expedite delivery of containers out of the terminal while increasing the amount of cargo that can be handled.

“With HONE and D-Wave, each huge crane handled 60% more cargo per day, while the turnaround time for trucks was reduced by 12%.”

— SAVANTX TEAM

60%
MORE CARGO HANDLED EACH DAY PER CRANE
OPTIMIZING THE RENEWABLE ELECTRIC GRID

E.ON

Explored how D-Wave’s quantum hybrid technology can more efficiently manage modern power grids, which have an increasingly diverse and decentralized set of generating facilities.

“The IEEE 118-bus test case shows time performance speed-up. The increment in performance would enable real time planning and operations of electrical grids.”

—E.ON TEAM
A QUANTUM SOLUTION FOR INVESTMENT APPLICATIONS

“What normally took the bank several hours of compute time was reduced to just minutes via quantum computing technology – an up to 90% decrease in compute time over the traditional solution.”

—CAIXA BANK TEAM

The D-Wave hybrid solver significantly decreased compute time to minimize the capital needed for hedging operations, optimized investment portfolios, and increased a bond portfolio internal rate of return (IRR).
Questions?