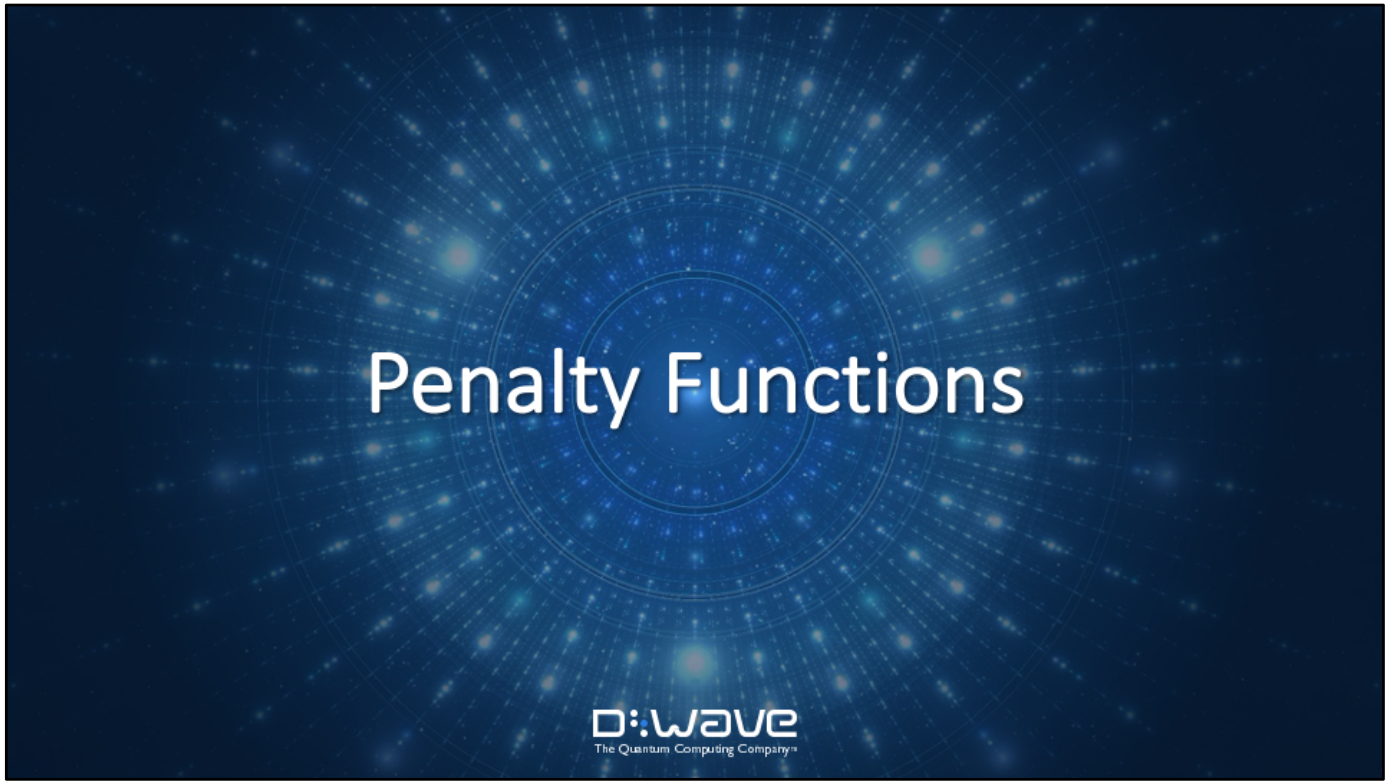


- We'll now take a look at penalty functions.
- Penalty functions are what we use to map constraints that are equalities or inequalities into a QUBO or Ising form.

Penalty Functions

Consider the problem:

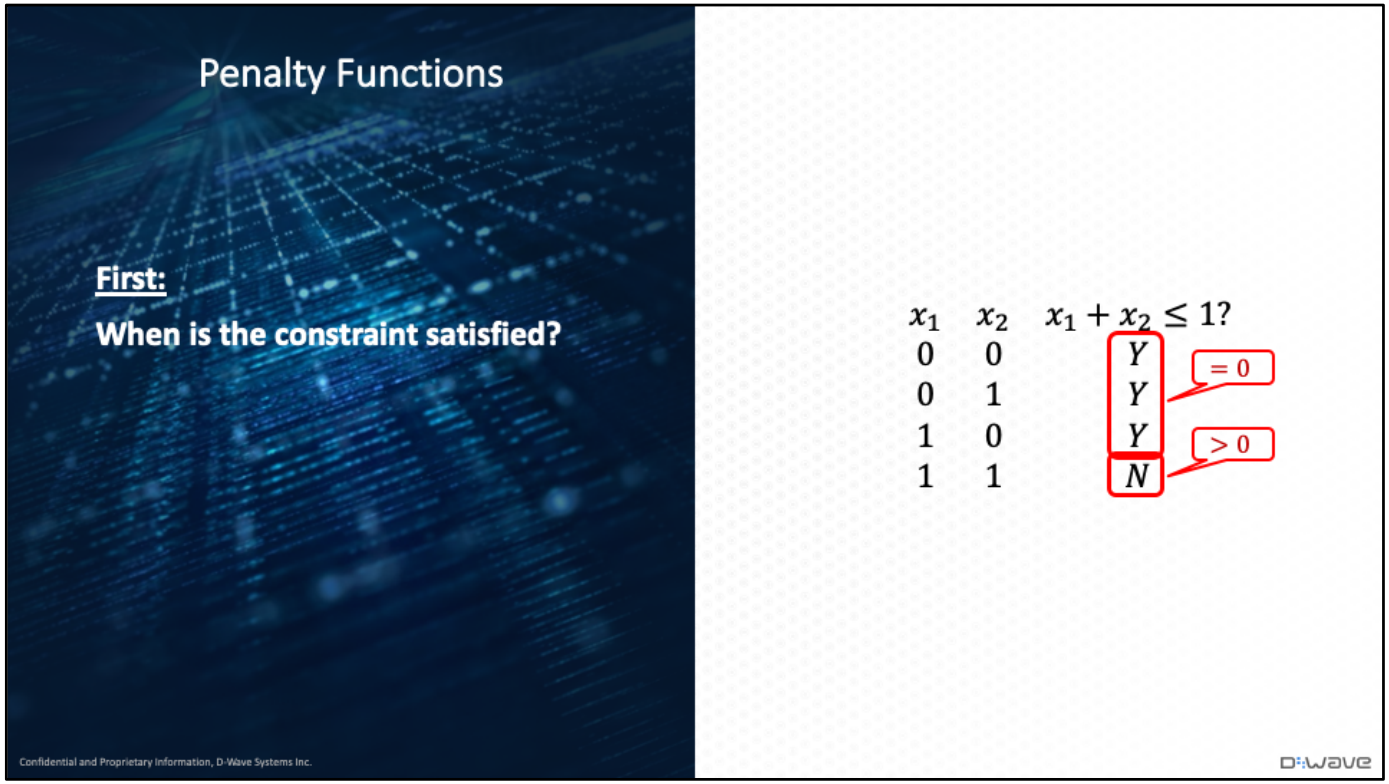
$$\begin{aligned} \min \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & x_1 + x_2 \leq 1 \end{aligned}$$

What is a penalty function for the constraint?

x_1	x_2	$x_1 + x_2 \leq 1$?
0	0	Y
0	1	Y
1	0	Y
1	1	N

Confidential and Proprietary Information, D-Wave Systems Inc.

- We'll start with a basic problem.
- Suppose we have some objective function, f(x), that we're trying to minimize, and we also have this constraint: $x_1 + x_2 \leq 1$.
- How can we transform this constraint to fit into a QUBO model?
- We need the constraint to be rewritten as an expression that is true at the smallest value and false at all larger values.
- We can start with a truth table like we did in our earlier modules.



- First, we look at which combinations of our binary variables satisfy our constraint, and which combinations don't satisfy it.
- Next, we replace the "Y"'s with a small number, let's pick 0 to make it easy, and the "N"'s with any larger number.

Penalty Functions

First:

When is the constraint satisfied?

x_1	x_2	$P(\boldsymbol{x}) = x_1 x_2$
0	0	0
0	1	0
1	0	0
1	1	1

Confidential and Proprietary Information, D-Wave Systems Inc.

- What mathematical expression produces this final column in our table?
- We can go through and solve our system of equations as we did earlier and we'll find that our expression or penalty function is simply x1 times x2.

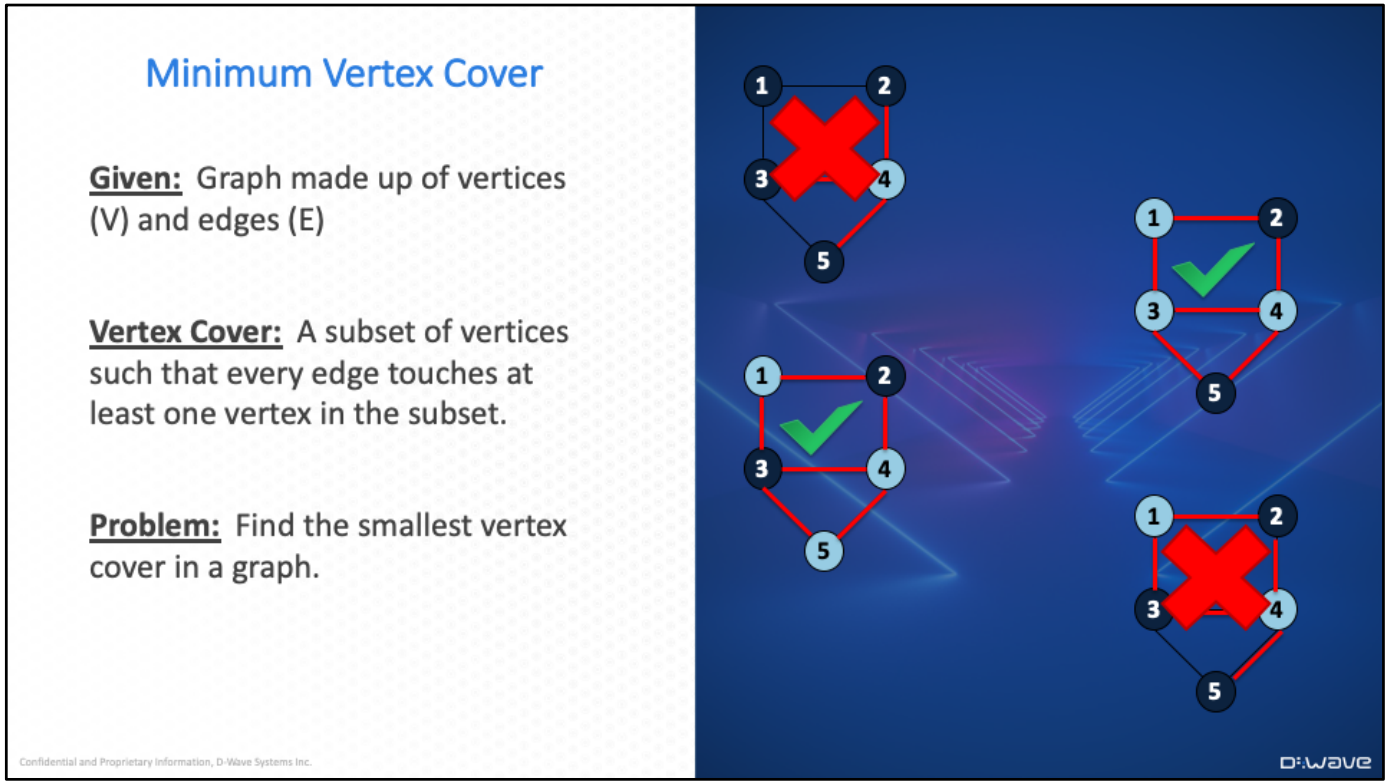
Some known penalty functions

Constraint	Penalty Function
$x + y \leq 1$	$x \cdot y$
$x + y \geq 1$	$1 - x - y + x \cdot y$
$x + y = 1$	$1 - x - y + 2 \cdot x \cdot y$
$x \leq y$	$x - x \cdot y$
$x + y + z \leq 1$	$x \cdot y + x \cdot z + y \cdot z$
$x = y$	$x + y - 2 \cdot x \cdot y$

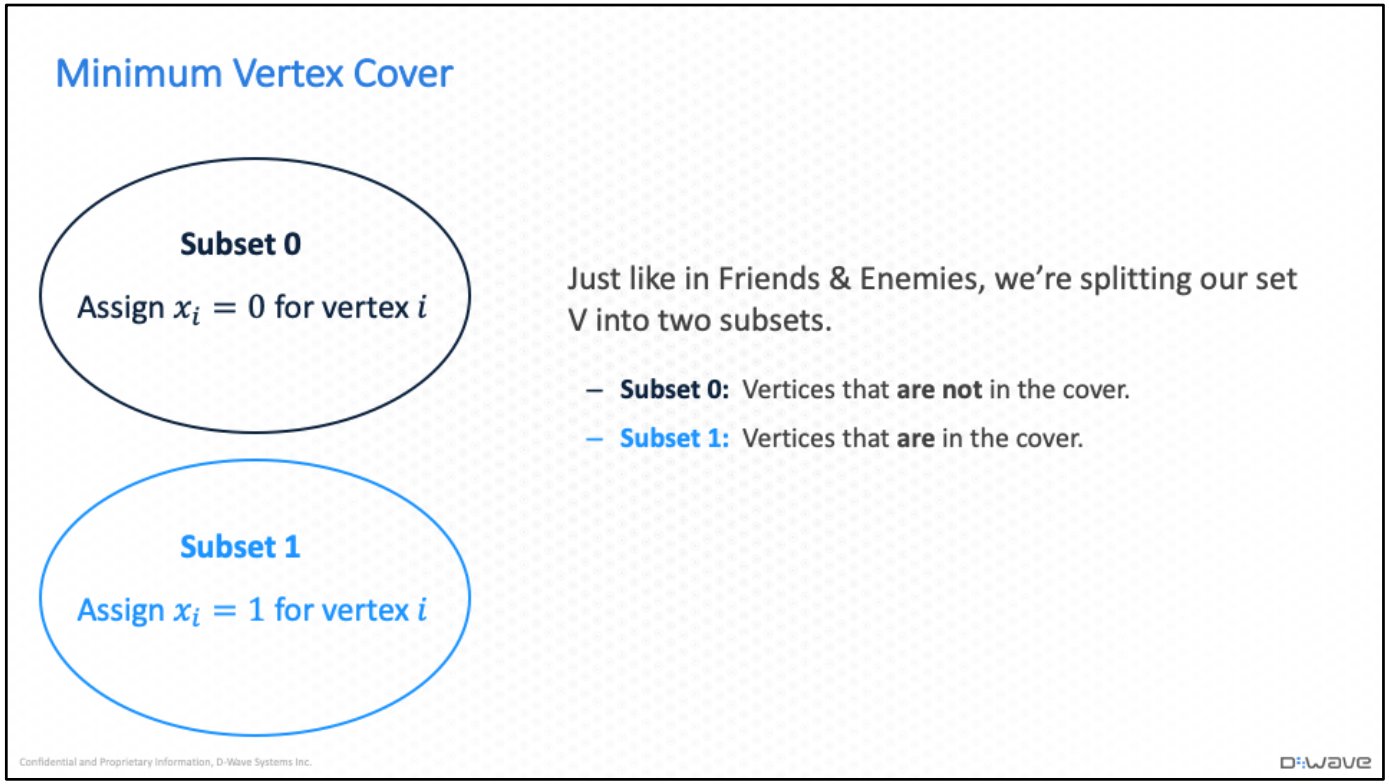
Confidential and Proprietary Information, D-Wave Systems Inc.

- We often see the same types of constraints over and over again in different problems, so it's useful to have a reference table of some basic penalty functions.
- Here are six different penalty functions to keep handy so that you don't need to go through the derivation every time you come across them.

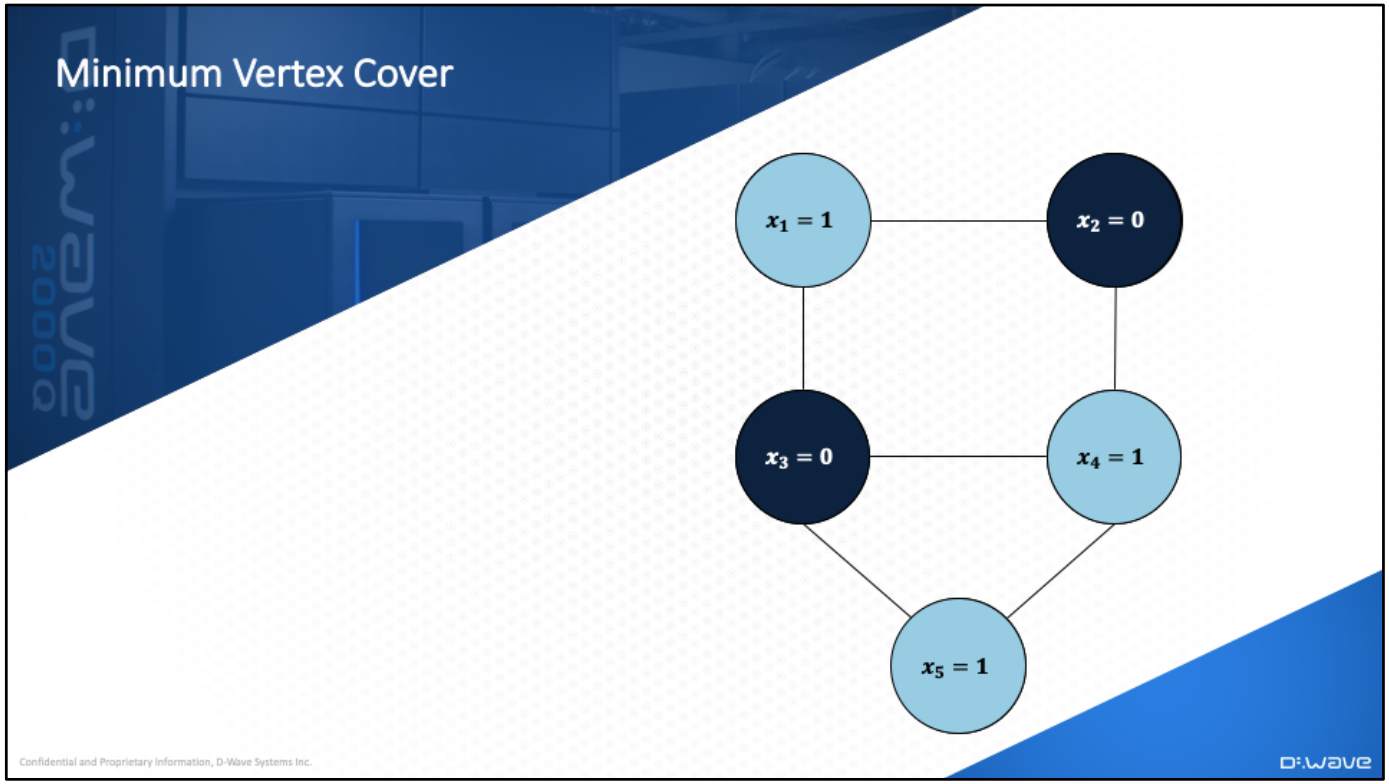
- Let's see how we can use these in an example.
- We'll take a look at the vertex cover problem.



- Here's the set up for this problem.
- We start with a graph that's made up of vertices (or nodes) and edges.
- We want to pick a subset of the nodes so that every edge in the graph touches at least one vertex in the subset.
- The goal is to find the smallest subset that does this – the smallest vertex cover.
- Let's look at some examples.
- This first graph is picking the subset that contains only vertex 4.
- We see all of the edges touching the subset highlighted red.
- Notice that there are edges in the graph that are NOT red, so this is not a valid vertex cover.
- Next let's look at the subset containing vertices 1 and 4.
- Again, not all of the edges are highlighted red, so this is NOT a valid vertex cover.
- What about the subset containing vertices 1, 4, and 5?
- Now we see all edges are highlighted red, so this is a valid vertex cover.
- Last but not least, let's look at the subset containing vertices 1, 3, and 4.
- Again, all edges are highlighted red, so this is a valid vertex cover.



- We start by choosing our binary variables.
- Just like we've seen in a few earlier examples, we are splitting our set of vertices into two subsets.
- One subset is the "special" vertices – the ones we are picking for a vertex cover.
- The other subset is the leftover vertices.
- We can use our binary variables to label these subsets: each special vertex in the cover takes value 1, and the vertices that are not in the cover take value 0.



- For example, in the graph we were looking at earlier, we saw that a valid vertex cover was the subset containing vertices 1, 4, and 5.
- Using our binary variables, we would assign x1, x4, and x5 a value of 1 and the rest a value of 0.

Minimum Vertex Cover

Objective: **Minimize subset size**

Constraint: **Every edge is incident to the subset**

Confidential and Proprietary Information, D-Wave Systems Inc.

- The objective of our problem is to minimize the cover size, and our constraint is that every edge need to touch the cover.

Minimum Vertex Cover

Objective: $\min \sum_{i \in V} x_i$

Constraint: $x_i + x_j \geq 1 \text{ for all } (i,j) \in E$

We need to use one of our penalty functions!

Confidential and Proprietary Information, D-Wave Systems Inc.

- Rewriting these with our binary variables, our objective becomes a summation of the binary variables to count how many are assigned to our cover.
- The constraint makes sure that at least one endpoint of every edge is in our cover or has a value of 1.

Minimum Vertex Cover

Constraint: $x_i + x_j \geq 1$ for all $(i, j) \in E$

Constraint	Penalty Function
$x + y \leq 1$	$x \cdot y$
$x + y \geq 1$	$1 - x - y + x \cdot y$
$x + y = 1$	$1 - x - y + 2 \cdot x \cdot y$
$x \leq y$	$x - x \cdot y$
$x + y + z \leq 1$	$x \cdot y + x \cdot z + y \cdot z$
$x = y$	$x + y - 2 \cdot x \cdot y$

Confidential and Proprietary Information, D-Wave Systems Inc.

- Referring back to our penalty function table, we see a penalty function for this type of constraint.
- We can plug in this penalty function for each edge in our graph.

Minimum Vertex Cover

Constraint:

$$x_i + x_j \geq 1 \text{ for all } (i,j) \in E$$

becomes

$$\sum_{(i,j)\in E} (1 - x_i - x_j + x_i \cdot x_j)$$

Confidential and Proprietary Information, D-Wave Systems Inc.

- Here our constraint has been rewritten.
- We see the penalty function written for an arbitrary individual edge between vertices i and j, and we sum over all possible edges in the graph.

Minimum Vertex Cover

QUBO:

$$\min \sum_{i \in V} x_i + \gamma \sum_{(i,j) \in E} (1 - x_i - x_j + x_i \cdot x_j)$$

Simplified:

Degree of vertex i

$$\min \sum_{i \in V} (1 - \gamma \cdot d(i)) x_i + \sum_{(i,j) \in E} \gamma \cdot x_i \cdot x_j$$

Confidential and Proprietary Information, D-Wave Systems Inc.

D-Wave

- So our QUBO can be built by adding our objective and constraint together with a Lagrange parameter.
- To write a program we need to find the QUBO matrix, so we can multiply this out and simplify.
- The math equation gets a bit uglier, but we can simplify it using some notation from graph theory.
- D(i) means the degree of vertex i, or how many edges are touching vertex i.

Doing the math…

$$\min \sum_{i \in V} x_i + \gamma \sum_{(i,j) \in E} (1 - x_i - x_j + x_i \cdot x_j)$$

$$= \min \sum_{i \in V} x_i + \sum_{(i,j) \in E} (\gamma - \gamma x_i - \gamma x_j + \gamma x_i \cdot x_j)$$

$$= \min \sum_{i \in V} x_i + \sum_{(i,j) \in E} \gamma + \sum_{(i,j) \in E} (-\gamma x_i - \gamma x_j) + \sum_{(i,j) \in E} \gamma x_i x_j$$

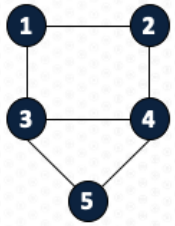
$$= \min \sum_{i \in V} x_i + |E|\gamma + \sum_{i \in V} -\gamma d(i) x_i + \sum_{(i,j) \in E} \gamma x_i x_j$$

Confidential and Proprietary Information, D-Wave Systems Inc.

- For those of you that are interested in how this formula was simplified, here's a few more steps.
- First, we can move gamma inside the second summation.
- Next, let's look at these linear terms inside the second summation – we want to eventually combine them with the linear terms in our first summation.
- We can break apart our summation into separate summations for constants, linear terms, and quadratic terms.
- In the linear terms' summation we have $-\gamma x_i - \gamma\, x_j$.
- For one individual vertex i we have a coefficient of $-\gamma$.
- Every edge that has vertex i as an endpoint will give us a linear term x_i with coefficient $-\gamma$.
- We can count how many edges have i as an endpoint using the degree notation I mentioned earlier, remember that D(i) is the number of edges that have i as an endpoint.
- That means that we can rewrite this term as a sum over the vertices (instead of edges) where each variable x_i has coefficient $-\gamma \cdot d(i)$.
- Last but not least, remember that we won't put any constant terms in our matrix – they don't get sent over to the QPU – so we can disregard the $|E| \cdot \gamma$ term.

Minimum Vertex Cover

Back to our example graph…



$$\min \sum_{i \in V} \left(1 - \gamma \cdot d(i)\right) x_i + \sum_{(i,j) \in E} \gamma \cdot x_i \cdot x_j$$

$$Q = \begin{pmatrix} 1-2\gamma & \gamma & \gamma & 0 & 0 \\ 0 & 1-2\gamma & 0 & \gamma & 0 \\ 0 & 0 & 1-3\gamma & \gamma & \gamma \\ 0 & 0 & 0 & 1-3\gamma & \gamma \\ 0 & 0 & 0 & 0 & 1-2\gamma \end{pmatrix}$$

Confidential and Proprietary Information, D-Wave Systems Inc.

- When we look at this QUBO for our example graph we can build up our matrix as shown.
- We see the linear coefficients, $1 - \gamma \cdot d(i)$, along the diagonal of the matrix.
- The off-diagonal entries are our quadratic coefficients, which is gamma whenever there is an edge.

- For your assignment, we have some problems for you to work on.
- As usual, you'll submit your answers over on Gradescope.

Knapsack Problem

We have five projects, each with some profit. We want to pick three projects to work on.

Which project(s) should we do in order to maximize profit?

Project	Profit
P1	20
P2	18
P3	22
P4	26
P5	21

Confidential and Proprietary Information, D-Wave Systems Inc.

- Here's the first question – it's a nice warm-up question before we get into more penalty functions.
- You have five projects to work on, and each project has some amount of profit associated with it.
- The problem is that you only have time to work on three projects.
- Which projects should you work on in order to maximize profit?
- Think about how you would write this problem out as an objective and constraints.
- Don't worry about getting this down to a single QUBO equation, just think about individual expressions for the objective and any constraints.

Knapsack Problem

Suppose that we also need to choose at least one project from {P1, P2} and at most one project from {P1, P4, P5}.

Which project(s) should we do?

Project	Profit
P1	20
P2	18
P3	22
P4	26
P5	21

Confidential and Proprietary Information, D-Wave Systems Inc.

- Next question.
- We start with the same set up as before, but now we have some extra requirements.
- You have to work on at least one project out of P1 and P2, and at most one project out of P1, P4, and P5.
- Now which projects should you work on?
- Again, think about how you would write this out as an objective and constraints, and see if any of the penalty functions from our table come in handy for this.
- When you're ready, go on over to Gradescope and complete the assignment. Good luck!