Vertex Cover

Input:

Vertices (V)

Edges (E)

Max vertex cover size (p)

Output:

True if the following conditions are met:

1) x <= p vertices are marked

2) The edges connected to these vertices form an equivalent set to E

False otherwise

Firing Problem

Inputs:

Number of Business-critical tasks (n)

Total number of employees (m)

Number of employees to fire (k)

Lists of which employees each task can be done by (Tasks)

Tasks.length == n

Output:

True if the following conditions can be met:

1) k employees get fired

2) For every task (t) in Tasks, there are 3 or more employees left over qualified for this task

False otherwise

Reduction

Let's start off by assuming that instead of the constraint that 3 or more employees left over must be qualified for the task,

only 1 employee must be qualified for the task. I will explain why this constraint can be manipulated later.

We will assign each Vertex given in the Vertex Cover problem to an employee at the company.

m = |V|

This also means that in the Vertex Cover problem, if we want to be left with p or less vertices,

this translates to the firing problem as being left with p employees.

k = |V| - p

We will then assign each Edge in the Vertex Cover problem to a task. A Vertex being attached to an Edge means

that an employee is qualified for that task.

n = |E|

Each task in the task list will then, naturally, be assigned to the employees that also correspond to each

individual vertex connected to the edge representing the task.

Now we'll go through the requirements of the two problems and confirm that they always return the same

result. The truth conditions for the two problems are the following.

Vertex Cover:

1) x <= p vertices are marked

2) The edges connected to these vertices are all the edges in E

(Modified) Firing Problem:

1) k employees get fired

2) For every task (t) in Tasks, there is at least 1 employee left who can perform this task

Given our transformation, we know that for the vertex cover to have k or less vertices marked, this means that our firing problem is left with p employees (we assume marking a vertex means they didn't get fired). Since the firing problem requires k employees to get fired, if we set k = |V| - p, we are left with p employees, which directly relates constraint (1) of Vertex Cover to constraint (1) of the Firing Problem.

As for constraint (2), we want to make sure that all the tasks can be performed by the employees left over (in order words, the marked vertices) if and only if all the edges are connected to these vertices in the Vertex Cover problem. This means that if we choose the Edges to represent a task assigned to each employee corresponding to a vertex in that edge, the Firing Problem will only return true if EVERY task has an employee who can do it, or in other words if EVERY edge has a marked vertex connected to it.

Our two constraints are therefore backwards compatible and will always return the same result given this transformation.

There is a problem we left for the end, which is that this logic only works with a modified version of our firing problem. This is only actually sound given the assumption that constraint (2) is modified to only require 1 employee for each task rather than 3. There is, however, a relatively straightforward way to amend this constraint, and that is by ensuring that there are always 2 employees left over who can do every task.

Therefore, our inputs will have to be amended to account for these two extra employees:

m = |V| + 2

k = |V| - p

n = |E|

And finally, our task lists will simply add employees |V|+1 and employees |V|+2 to every list. This will ensure that, since we're still firing the same number of people, these two employees may always remain and be 2 of the 3 required employees for each business-critical task.