**The Firefighter Problem- Problem brief**

**Set-up of the problem:** Fire starts at one or more vertices of a graph (variety of graphs you can play around with and see how different strategies work) and after each round, the neighbouring fires at node(s) on fire, the fire spreads to those nodes.

**Potential Aims**:

* Minimise the number of nodes that a fire spreads to
* Minimise the number of round the fire is live for
* Maximise the number of nodes that are saved from a given subset of nodes

**Solution to apply**: Before each time round, we can apply a number of defender nodes to nodes in the graph, which will mean that a fire cannot spread to those nodes (i.e: acting as a blocker to the fire spreading elsewhere). Each time round ,we apply a number of defenders until there are no more nodes that can catch fire.

**Terminology:**

* A node that doesn’t catch fire, by the end of the simulation is considered “saved”
* A node that has caught fire in a previous time period is considered “burnt”
* “Defender nodes” are what we allocate to nodes in the graph, once allocated in a certain time period, that node cannot catch fire and the fire cannot spread through any edges connected to that node

**Variation to The Firefighter Problem:**

* Firebreak (from Jessica Enright email to Ozgur Akgun, which Ozgur kindly forwarded to me): Rather than allocating a certain number of defender nodes every time period, we question:
  + Given a graph G = (V,E), a single node (v\_f, a single node where the fire starts), an integer s (number of nodes we want to save atleast) and an integer k (number of defender nodes we can allocate), does V contain a k-subset which separates at least t nodes from the v\_f.

**Things to note**:

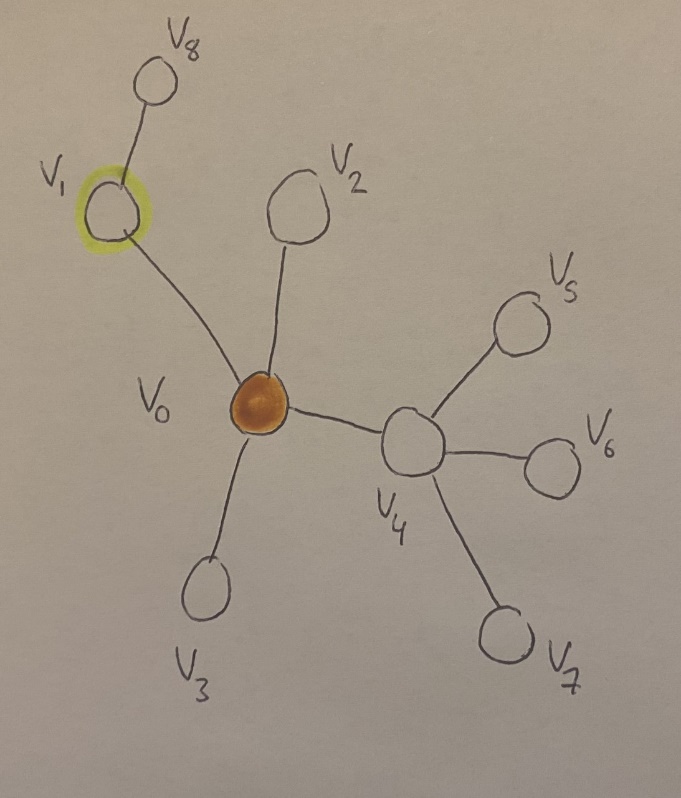
* Each time round, the **same** number of defender nodes can be allocated
* Once a vertex has caught fire, it remain “burnt” for the remainder of the simulation
* We also can not just aim to find a solution (sequence of application of defender nodes) to minimise (or maximise) something, we can also determine whether the instance has a particular property (e.g.: there exists a solution which means all vertices from a sub-graph of the instance of the problem class
* This problem is also considered for infinite graphs (but probably beyond of the scope of this dissertation, as it seems this is more of graph theory exercise than a practical computer science exercise)

Here the problem class depends on the aim, but even so, the parameters of problem classes will include at least (V, E, d):

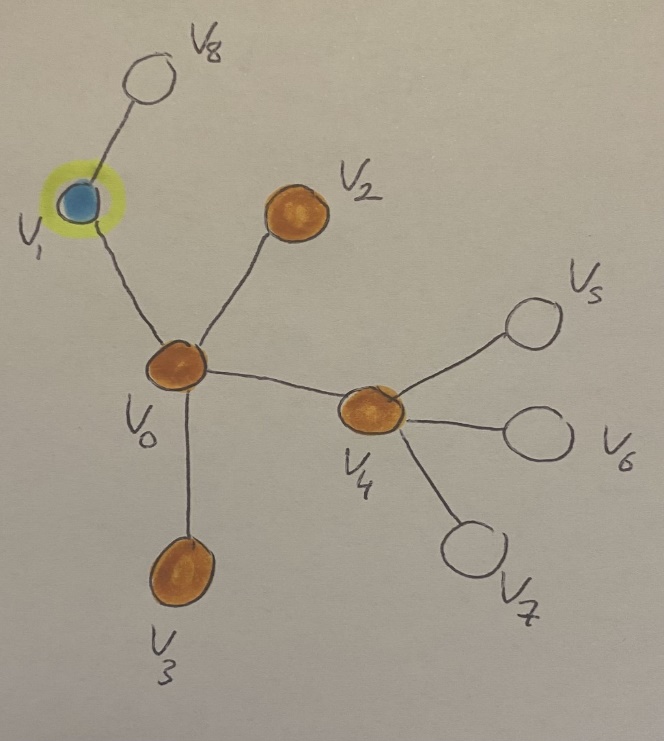
* Where V is the set of nodes of the graph
* E is the set of edges of the graph
* d is the number of defender nodes allocation you can apply to the graph
* S is the set of nodes, to which a fire starts (S is a subset of V)

What’s an interesting element of The Firefighter Problem is that two different objectives can be at conflict, here’s an example:

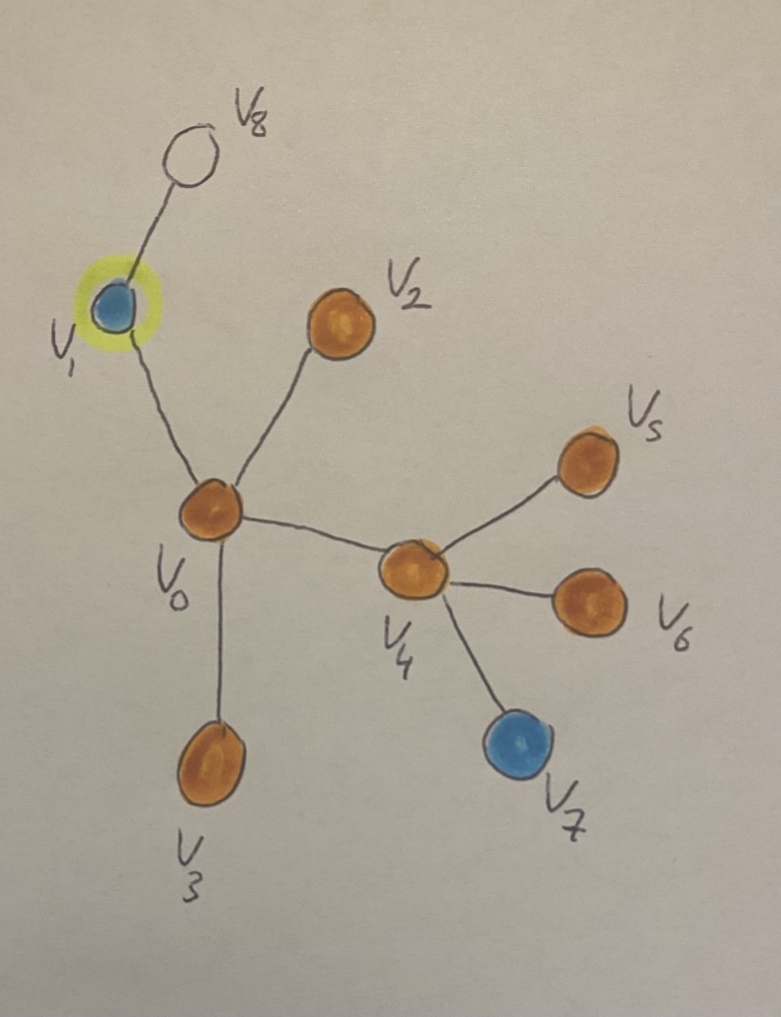
If we had the following graph and we had 1 node that we could choose to be defended at each round, and our aim was to ensure that the v\_1 node was saved at all costs:



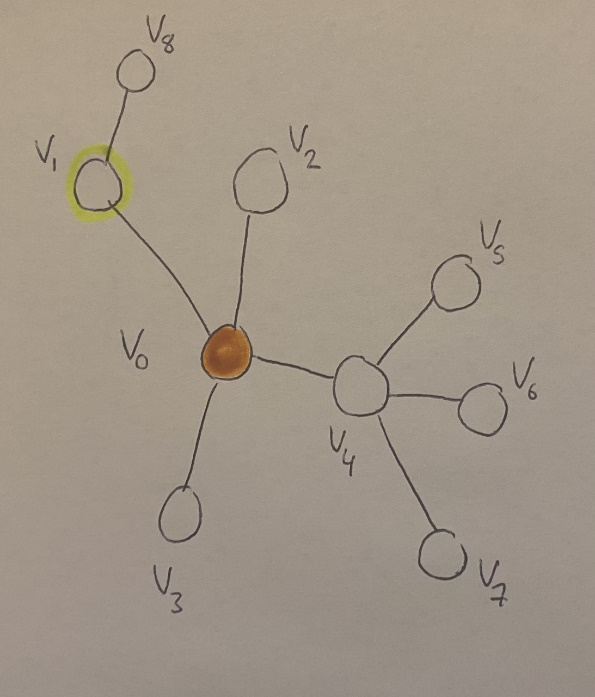
An obvious first step would be to place our single defender choice on node v\_1, however this leads to v\_4 catching fire, which we would want to avoid if we wanted to minimise the number of nodes that catch fire, since it’s connected to 3 other nodes:



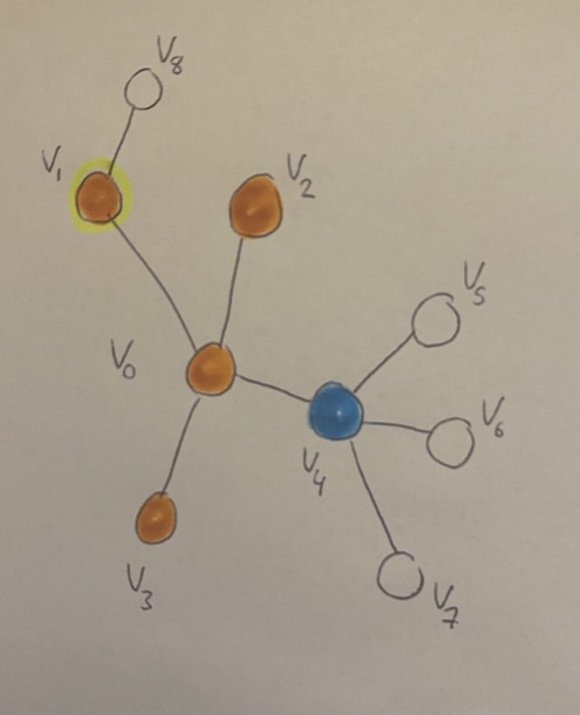
Choosing where to place our last defender node might as well be done at random, since each will lead to exactly 1 node being saved and we are one time period away from the simulation ending:



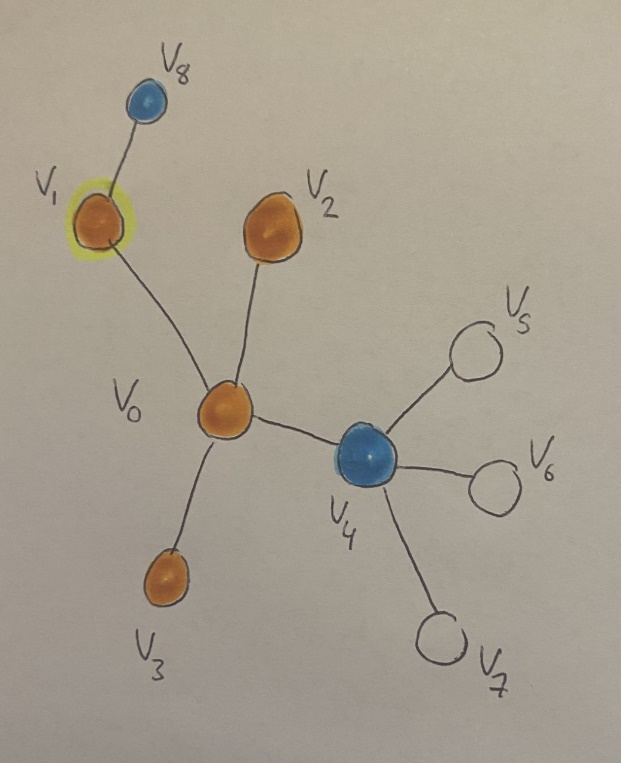
Instead, if we were to aim to minimise the number of nodes that are to catch fire, let’s start the simulation again:



Now instead, we can choose v\_4 to be defended, as we know, since it has degree 4, if it was to catch fire (given the number of nodes we have as defender nodes we can allocate), we would not be minimising the number of nodes that catch fire. However, by doing this we leave v\_1 to catch fire (in conflict with our first aim):



And we can conclude by choosing v\_8 to be defended:



As we can see, 4 nodes have caught fire (the minimal amount that will catch fire), however, we failed to save v\_1. On the flip side, in our first simulation, we saved v\_1 as a node, however 6 nodes in total have caught fire and we failed to minimise the number of nodes that caught fire.

Applications:

* This has an application, anywhere where a network is involved and whatever is representing the nodes, something is spreading (through the edges). So some obvious examples include:
  + Epidemiology -> spread of virus through a population
  + Social network -> spread of information (or dis-information)
  + Transport networks -> tracking individuals on the transport network, ensuring that possible routes they could of taken (in combination of where they might be) have been explored

Greedy algorithm:

One version of the greedy algorithm that exists (Hartnell, 2000 citied in Finbow, 2009), is (when the number of nodes that a fire starts out at is equal to 1) to place a defender node at each round, to one of the nodes connected to a node which is on fire that will lead to the biggest sub-graph (with that nodes and its ancestors) being saved. We know that placing a node as defender will lead to a sub-graph being “saved” since we are considering a tree (since, for a graph which is a tree; every path between each pair of nodes is distinct, so no other path for which the fire can spread to, exists). This can be extended if we had multiple defenders available in each round, as we can simply choose the next biggest sub-graph to “save”.

Papers which seem referenced but can’t find:

A.E. Scott, U. Stege and N. Zeh, Politician’s Firefighting. Manuscript, 2007.

[33] K. L. Ng and P. Raff, Fractional firefighting in the two dimensional grid. DIMACS Technical Report 2005–23, Rutgers University, New Brunswick, NJ, USA.

[15] S. Finbow, B. L. Hartnell, Q. Li and K. Schmeisser, On minimizing the effects

of fire or a virus on a network, J. Combin. Math. Combin. Comput. 33 (2000),

311–322.