Approximation Algorithms for the Firefighter Problem: Cuts over Time and Submodularity

Elliot Anshelevich[1], Deeparnab Chakrabarty[2], Ameya Hate[1], and Chaitanya Swamy[2]

[1] Department of Computer Science, Rensselaer Polytechnic Institute
[2] Dept. of Combinatorics & Optimization, University of Waterloo

Abstract. We provide approximation algorithms for several variants of the FIREFIGHTER problem on general graphs. The Firefighter problem models the case where an infection or another diffusive process (such as an idea, a computer virus, or a fire) is spreading through a network, and our goal is to stop this infection by using targeted vaccinations. Specifically, we are allowed to vaccinate at most B nodes per time-step (for some budget B), with the goal of minimizing the effect of the infection. The difficulty of this problem comes from its temporal component, since we must choose nodes to vaccinate at every time-step while the infection is spreading through the network, leading to notions of "cuts over time".

We consider two versions of the Firefighter problem: a "non-spreading" model, where vaccinating a node means only that this node cannot be infected; and a "spreading" model where the vaccination itself is an infectious process, such as in the case where the infection is a harmful idea, and the vaccine to it is another infectious idea. We mainly look at two measures: the MAXSAVE measure in which we want to maximize the number of nodes which are not infected given a fixed budget B, and the MINBUDGET measure, in which we are given a set of nodes which we have to save and the goal is to minimize the budget. We give complexity and approximation results for these problems on both models.

1 Introduction

Faced with an epidemic that is spreading through a population, and a limited supply of vaccine (or simply a lack of time to administer it), it is necessary to decide whom to vaccinate. Questions about the spread of disease and epidemics in a social network have often been modeled using graph theory (e.g. [2, 11]), and correspond to fundamental graph-theoretic concepts [23]. Moreover, these graph theoretic principles can be applied to many diffusive network processes, including epidemics in computer networks, the spread of innovations and ideas, and viral marketing [24]. In this paper, we focus specifically on inhibiting the spread of an epidemic or an idea by using vaccination.

Questions about epidemic propagation have been studied in several fields (e.g., [4, 31]), although most of this research does not consider the structure of the corresponding network (partly because this structure is only recently becoming available). Instead, building on the work of [23] and others [2, 12], this paper assumes that we know the network's topology, and provides worst-case guarantees over all possible networks. The works mentioned above consider *prophylactic vaccinations,* where the goal is to vaccinate parts of the graph so that once the epidemic begins, the destruction caused by it will be limited. The problem we consider, however, considers the case where the infection has already begun, and we must attempt to minimize its effect.

Model and the Firefighter problem We model our network of agents as an undirected graph $G = (V, E)$ where vertices correspond to agents and an edge $e = (u, v)$ represents contact between u and v. The above is arguably a simplistic model, however, as we see below, even this model leads to many interesting questions and will be the main focus of our paper. In the appendix we describe how to make our model more general.

Such a model of spread of infection has been studied in the literature as the *Firefighter problem* [16, 21]. In this problem the infection/fire starts at a given node node s (or a set of nodes) at time $\tau = 0$. At every subsequent time step, the infection/fire deterministically spreads to all nodes that have an already infected neighbor. To stop the infection, we are allowed to vaccinate/defend at most B nodes per time-step, where B is a budget representing how much we are able to affect the network in a single time-step. A vaccinated node can no longer contract the infection, and therefore cannot pass it on to others. Once infected or vaccinated the vertex remains so for the rest of the time. The process comes to an end when the infection can no longer spread.

We consider two separate objectives in this paper. The first objective, which we call MaxSave, is to maximize the number of non-infected nodes in the end, when we are given a fixed budget B. The second objective, which we call MinBudget, is to minimize the budget B needed per time instant in order to save a given set of nodes, $T \subseteq V$.

It is not hard to see that in the end, the set of vaccinated notes form a vertex cut between the set of infected nodes and set of non-infected nodes. However, unlike previous works, such as [23], which examine the *static* problem of vaccinating a 'cut' before the infection has started spreading, we need to find the "best" *cut over time* (where best depends on the considered objective). This temporal nature of our optimization problem makes it significantly different and more challenging than the non-temporal versions.

In this paper, we also consider a different variant of this model, where the vaccination is also a process that spreads through the network. In this "spreading vaccination" model, if a node v is adjacent to a vaccinated node, then v itself becomes vaccinated during the next time-step (unless it is already infected). In the case of ideas propagating through a social network, this represents the fact that an antidote to a harmful idea is often another idea, which can be just as infectious. In disease propagation, this represents the fact that vaccines can be infectious as well, since they are often an attenuated version of the actual disease. We consider the above two objectives in this model as well.

Example 1. To gain some intuition about this problem, consider the example shown in Figure 1 using the non-spreading model of vaccination.

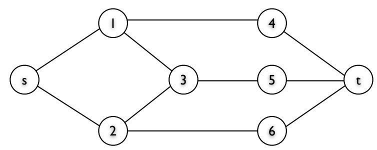

Fig. 1. This example shows that sometimes vaccinating nodes far away from the infection is the only way to save all the required nodes.

Consider the MinBudget objective for this example. The infection begins at node s, and the goal is to find the smallest number B of nodes that need to be vaccinated at every time step so that we can save the node t, which we assume cannot itself be vaccinated. If we were only allowed to cut nodes during the first time-step, this would be equivalent to the minimum s-t node-cut problem. The temporal nature of the problem, however, complicates matters: intuitively, the tradeoff is between vaccinating a small set of nodes close to the infection source early, or spreading out (over time) the vaccination of a larger set of nodes which are farther away from the source.

For instance, in the above example, a minimum s-t node-cut is $\{1, 2\}$, which requires $B = 2$. However, there *is* a solution to the above problem with $B = 1$, but the final set of vaccinated nodes does not form a minimum s-t node-cut. One such solution is to vaccinate vertices 4, 6, and 5 at time steps 1, 2, and 3 respectively, leading to the final set of vaccinated nodes being $\{4, 5, 6\}$ which is not a minimum cut. In fact, it is not hard to come up with examples where the optimal value of B is much smaller than the size of a minimum node s-t cut *and* the final set of vaccinated nodes is much larger than the size of a minimum node s-t cut (e.g., take a graph where s has k neighbours, each of which is connected to t via k long internally node-disjoint paths). Thus, this "cuts over time" problem is quite different from the classical min-cut problem, and in fact is known to be NP-hard (even when the graph is a tree!) [15]

Our Results In Section 3, we consider the model of spreading vaccinations. In general, our results show that this model is more tractable than the model with non-spreading vaccinations. For MaxSave we show that this problem reduces to maximizing a submodular function with a matroid constraint. Therefore a simple greedy algorithm provides a 2-approximation, and a recent result of [7] lets us prove a $(1 - 1/e)$ factor approximation. For MinBudget we give a $O(\log n)$ approximation algorithm, and show that this approximation ratio is tight, by showing a set-cover hardness.

The non-spreading model, on the other hand, does not yield itself to good approximation algorithms. In fact, we show in Section 4 that it is NP-hard to approximate MaxSave in general graphs by a factor of n^α, for any $\alpha < 1$. For MinBudget, we give a $O(\sqrt{n})$ factor approximation algorithm for general graphs based on a natural LP relaxation for the problem. We also show that the integrality gap of the LP is bounded by $\Omega(\log n)$. For *directed* layered graphs with ℓ layers, we give a $O(H_\ell) = O(\log \ell)$ approximation algorithm. The latter algorithm is combinatorial and requires just one max-flow computation.

Section 4.1 is devoted to vaccination strategies when the underlying graph is a tree. This special case has received a lot of attention [22,27], is computationally difficult [15], and is in fact a generalization of a complex scheduling problem (see the Appendix). For this special case we show that both the spreading and the non-spreading models are equivalent, so the stronger results from Section 3 hold for the non-spreading model as well. In addition, our algorithm for layered graphs also implies a $O(\log h)$ approximation algorithm for MinBudget on trees with height h. Note that this is stronger than the $O(\log n)$ algorithm we have for general graphs in the spreading model. Table 1 summarizes our results.

	Spreading	Non-Spreading	Trees
Max-Save	$(1-1/e)$ appx	n^α-hard for $\alpha < 1$	$(1-1/e)$ appx. Also by [27]
Min-Budget	$O(\log n)$ appx $\Omega(\log n)$-hard	General: $O(\sqrt{n})$-appx Directed ℓ-Layered Graph: $O(\log \ell)$-appx	$O(\log h)$-appx for height h trees.

Table 1. The summary of our approximation results. appx stands for approximation, hard stands for hard to approximate, unless $P = NP$.

Related Work Questions about epidemic propagation have been studied in several fields, (e.g., [4,31]), although most of this research models the epidemics as dynamic systems and ignores the effect of the network structure. Recently, a few groups [6,19,30,32] have considered the spread of viruses or ideas on Internet-like topologies, such as small-world networks [35] and preferential attachment models [5,26]. Several papers also study targeted vaccinations in this context [10,13], and show that they can be used to significantly reduce the effect of epidemics. These studies assume certain properties of the networks (based on where these networks arise from).

Several recent papers considered modeling vaccination by using graph cuts. For example, the work of Hayrapetyan et al. [23] and others [2,12] fully utilizes the social-network structure to “cut off” and contain various diffusive processes in a social network. As mentioned earlier, all this work is only concerned with vaccinating a set of nodes before the infection begins, however, and does not have the temporal component of the Firefighter problem. A lot more work has been done on maximizing the spread of an infection (instead of trying to stop it using vaccinations), by selecting the best nodes to immunize initially [11,24].

The *Firefighter problem* was first introduced by B. Hartnell [21], and there has been much work on this problem; see, e.g., [16] for a survey. However, much of the work has focused on special graph structures, such as grids [9,18,34], and that too usually with the MaxSave objective. The Firefighter problem is NP-complete even when the underlying graph is a tree [15], although [22] and [27] give approximation algorithms for this case, and [29] shows how to solve the problem in in polynomial time for special cases of trees.

The Firefighter problem with the MinBudget objective, has some structural similarity with the problem of length-bounded cuts [3], where the goal is to form a minimum s-t cut that destroys all paths of length at most L. However, as illustrated by the example above, the main difference here is that our paper deals with *dynamic cuts* i.e., cuts which arise over a period of time, while graph cuts are static in nature.

2 Formal description of our model

We are given a directed[3] graph $G = (V, E)$ and a source node s. All nodes in the graph can have one of three states: they can be *infected*, *vaccinated*, or *vulnerable*, that is neither vaccinated nor infected. At time $\tau = 0$, all nodes are vulnerable, except node s, which is infected. At each $\tau > 0$, any vulnerable vertex v which is connected to an infected node u, such that $(u, v) \in E$, gets infected at time $\tau+1$, unless it is vaccinated at time step τ. Infected and vaccinated nodes stay infected and vaccinated respectively. We call a node *saved* if it is either vaccinated or if all paths from any infected node to it contains at least one vaccinated node.

Definition 1. *A vaccination strategy is a set $\Psi \subseteq V \times J$ where V is the set of vertices of graph G and $J = \{1, 2, \ldots |V|\}$. The vertex v is vaccinated at time $\tau \in J$ by the vaccination strategy Ψ if $(v, \tau) \in \Psi$. A vaccination strategy Ψ is valid with respect to budget B, if the following two conditions are satisfied:*

i. if $(v, \tau) \in \Psi$ then v is not infected at time τ,
ii. let $\Psi_\tau = \{(v, \tau) \in \Psi\}$; then $|\Psi_\tau| \leq B \quad for \quad \tau = 1 \ldots |V|$.

The first condition implies we can only vaccinate vulnerable nodes, and the second condition requires us to obey the budget constraint.

In the *Non-spreading Vaccination Model*, vaccinating a vertex simply means that it can no longer be infected. In the *Spreading Vaccination Model*, however, the vaccination spreads to all its neighboring nodes which are still vulnerable, thereby vaccinating them. That is, at time step $\tau > 0$, if a node v is vaccinated and there is a vulnerable node u such that $(v, u) \in E$, then at time $\tau + 1$, the node v also gets vaccinated. Thus, the vaccination also spreads like the infection. Note that it could be a vulnerable node is adjacent to both an infected node and a vaccinated node. We will assume that the vaccine prevails over the infection, and in the subsequent time step, the vulnerable node is vaccinated, rather than being infected. This is actually a weak assumption as assuming otherwise doesn't change the quality of our results. In the spreading model, we will say that a node is vaccinated *directly* when it is vaccinated by the vaccination strategy, and it is

[3] We use a directed graph since it is more general – an undirected graph is just a directed graph with two arcs per edge.

vaccinated *indirectly* when it is vaccinated by the spread of the vaccine through the network.

The process stops when there are no vulnerable adjacent to a infected node, so the infection cannot spread any further. This must occur before time n, for n being the number of nodes.

Objectives The main two objectives we consider when developing a vaccination strategy are as follows.

MaxSave(G, B, s, T)
Instance: A rooted graph $(G(V, E), s)$, integer $B \geq 1$ and $T \subseteq V$
Objective: Find a valid vaccination strategy Ψ such that if s is the only infected node at time 0, then at the end of the above process the number of non-infected nodes that belong to T is maximized.

This problem is also referred to as the Firefighter problem in the literature when $T = V$.

MinBudget(G, s, T)
Instance: A rooted graph $(G(V, E), s)$, and $T \subseteq V$
Objective: Find a valid vaccination strategy Ψ with minimum possible budget B, such that if s is the only infected node at time 0, then at the end of the above process all nodes in T are saved.

In other words, in MaxSave we are interested in saving as many nodes of T as possible given a fixed budget, and in MinBudget we are interested in finding the minimum necessary budget to save all nodes in T.

3 Spreading Vaccination Model

We first show a few simple hardness results about this model, and then give approximation algorithms for both our objectives. Due to lack of space, all our proofs appear in the Appendix.

3.1 General Properties

We make certain useful observations about this model. Let $N(v, i)$ be the set of all the nodes that are a distance of at most i from v.

Lemma 1. *At time τ, all nodes in the neighborhood $N(s, \tau)$ will either be vaccinated or infected.*

Now, since all the nodes in the neighborhood $N(s, \tau)$ will be either infected or vaccinated by time τ, any optimal vaccination strategy would not vaccinate any node in this neighborhood at time $> \tau$. Since any valid strategy can vaccinate only B nodes at any time-step, it means that an optimal strategy would vaccinate at most $B \cdot \tau$ nodes directly in the neighborhood $N(s, \tau)$.

We define a set $\Gamma(v)$ for every node $v \in V$ by

$$\Gamma(v) = \{(u, \tau) | u \in V \text{ and } 0 < \tau \leq min(d(s, u), d(s, v) - d(u, v))\}$$

The tuple (v, τ) essentially represents the event of vaccinating node v at time τ. Using these definitions we state the following theorem.

Theorem 1. *A node $v \in V$ is vaccinated by the vaccination strategy Ψ* ***iff*** *$\Psi \cap \Gamma(v) \neq \emptyset$.*

This theorem tells us that vaccinating an element of $\Gamma(v)$ is exactly what is needed in order to save a node v, and this gives us insight into the structure of the problem.

3.2 Approximation for MaxSave

As we show in the Appendix, the MaxSave problem can be modeled as a problem of maximizing a submodular set function on a collection of sets that form a partition matroid. On the basis of this knowledge, techniques like the greedy algorithm [17] can be used to obtain a $\frac{1}{2}$ approximation for MaxSave, while the randomized algorithm of [7] can be used to obtain a $(1 - 1/e)$ approximation.

Theorem 2. *There is a randomized algorithm which gives with high probability a $(1 - 1/e)$ approximation for the* MaxSave *problem. Additionally, a simple greedy algorithm gives a $\frac{1}{2}$ approximation.*

The detailed proof is presented in Appendix C; here, we give its gist. A partition matroid consists of disjoint sets $E_1, \ldots, E_k$, and a set S is called independent if $S \cap E_i \leq \ell_i$, for some given numbers $\ell_1, \ldots, \ell_k$. We argue that one can actually (see Appendix C for details) consider vaccination strategies that satisfy only property (ii) in Definition 1. This set of strategies forms a partition matroid, since we can only choose at most B nodes at every time-step to vaccinate (so $\ell_i = B$ for all i). We next show that the function $f(\Psi)$ defined (suitably) as the number of nodes saved by using the (possibly invalid) vaccination strategy Ψ is submodular, by using Theorem 1, and if Ψ satisfies the budget-constraint then there is a valid vaccination strategy Ψ' such that $f(\Psi) = f(\Psi')$. We then use the results of [7,17] to obtain the desired approximations. See the Appendix for a more detailed definition of the concepts of Submodular Set Functions and Partition Matroids.

It should be noted here that the same $(1 - 1/e)$ approximation can also be obtained by applying a randomized rounding technique similar to [27] to a modified version of the MaxSave problem. We believe, however, that modeling the problem using partition matroids and submodular functions has great advantages. The algorithms produced in this way are combinatorial and far more efficient than randomized rounding, since they do not require solving a linear program. It also allows the use of a much simpler greedy algorithm as mentioned above, which has the advantage of being deterministic and efficient. Finally, the connection between submodular functions and MaxSave provides insight into the intrinsic structure of the problem, and allows for future results in the field of submodular functions to become applicable to MaxSave.

3.3 Approximation for MinBudget

Theorem 3. *The* MinBudget *problem is* $\log n$ *inapproximable by reduction from* Set Cover*.*

Consider an instance of MinBudget. First suppose that we know the size of the optimal budget B that is needed in order to save all nodes of T. Below we give an algorithm that saves all nodes in T using a budget of at most $B \log n$. To form a $\log n$ approximation algorithm without knowing B, we simply do a binary search on B, and run the algorithm below every time.

By slight adjustments to the proof of Theorem 2, we know that by running the greedy algorithm with budget B, we save at least half of the nodes in T. The greedy algorithm in this case chooses the nodes to vaccinate in each time-step one at a time, always picking the node that saves the most nodes of T. For this purpose the greedy algorithm needs to know exactly which nodes will be saved if a node u is vaccinated at time τ, which we can compute in poly-time. Once finished with the first time-step, the algorithm goes on to the second, and so on. The full algorithm for MinBudget is as follows.

Repeat $\log(n)$ times:

- vaccinate nodes in graph G using the greedy algorithm with budget B.
- Construct graph G_1 from G by removing all the vertices that were vaccinated directly and indirectly in the previous step. Let T_1 be the nodes of T that are in G_1.
- Set $G = G_1$ and $T = T_1$.

It is clear that the new graph G_1 will always contain the original source node s as it is never vaccinated by the greedy algorithm. By repeating the application of the greedy algorithm that vaccinates B nodes at every time-step $\log n$ times, we end up with an algorithm that vaccinates $B \log n$ nodes at every step. Let this be the *RepGreedy* algorithm.

Theorem 4. *The algorithm RepGreedy saves all nodes of* T *by vaccinating at most* $B \log n$ *per time-step.*

For the final $\log n$ approximation algorithm to MinBudget, do binary search on B, and run RepGreedy for every choice of B.

4 Non-Spreading Vaccination Model

The non-spreading model is considerably more difficult than the spreading model. One of the main reasons is that Lemma 1 (or any simple modification of it) is no longer true.

The MaxSave problem is NP-complete for bipartite graphs [29] and for cubic graphs (3-regular) [25]. The MaxSave problem is NP-complete even when restricted to trees with maximum degree three [15]. We prove the following about the inapproximability of MaxSave

Theorem 5. *The* MaxSave*(G(V,E),s,B,T) problem cannot be approximated in poly-time to the factor of n^α where $n = |V|$ and $\alpha < 1$, unless P=NP.*

We introduce an auxiliary problem, Save-t, which asks whether a specified node t can be saved by vaccinating one node (other than t) at a time. The NP-completeness of this problem follows from known NP-completeness proofs. We then give a gap introducing reduction from the Save-t problem to the MaxSave problem such that if there exists any n^α approximation for the MaxSave problem then we can solve the Save-t problem in polynomial time. The proof appears in Appendix D

In the remainder of the section, we focus on the MinBudget problem. Note that we need to save *all* the nodes in a set T with the minimum number of vaccinations required per time instant. To simplify notation, we consider the following equivalent problem: we add a new node t with edges from all nodes in T to t, and consider the problem of saving t with minimum budget *under the additional constraint that t itself cannot be vaccinated*. We call s the source and t the sink. Let $\mathcal{P}$ denote the collection of all s-t paths. We start with the following LP relaxation of the problem and its dual.

Minimize B (Primal)

$$\sum_{v \in V} x_v^\tau \leq B \quad \forall \tau = 1, \ldots, \ell \tag{1}$$

$$\sum_{i=1}^{k} \sum_{\tau=1}^{i} x_{v_i}^\tau \geq 1 \quad \forall (s, v_1, \cdots, v_k, t) \in \mathcal{P} \tag{2}$$

$$x_v^\tau \geq 0, \quad \forall v \in V, \ \forall \tau = 1, \ldots, \ell \tag{3}$$

Maximize $\sum_{P \in \mathcal{P}} f_P$ (Dual)

$$\sum_{\tau=1}^{\ell} z_\tau \leq 1 \tag{4}$$

$$\sum_{P \in \mathcal{P} : v \in P^{(\tau)}} f_P \leq z_\tau \quad \forall v \in V, \tau = 1, \ldots, \ell \tag{5}$$

$$z, f \geq 0 \tag{6}$$

The primal LP has a variable x_v^τ which indicates whether vertex v is vaccinated at time τ or not. $\ell \leq n$ is the length of the longest path from s to t; it is easy to see that we will not vaccinate any vertex after time ℓ. The first constraint bounds the number of vaccinations at every time instance. The second constraint says that for every path $(s, v_1, \cdots, v_k, t)$ to the sink t, one of the nodes, say v_i, must be vaccinated *by* time i. This is a necessary and sufficient condition for this path not to transmit the infection to t. In the dual, we have a flow for every s-t path P. We also have a variable z_τ which add up to 1. The second constraint in the dual is a bit subtle: it says, for every τ, the total flow through a vertex v via paths such that v lies at a distance τ or more from s on the path, is at most z_τ. In the LP, $P^{(\tau)}$ denotes the portion of the path from the τth vertex to t. That is if $P = (s, v_1, \ldots, v_k, t)$, then $P^{(\tau)} = (v_\tau, v_{\tau+1}, \ldots, t)$.

Although the primal LP above has exponentially many constraints, it can be solved in polynomial time since one can obtain the separation oracle in polynomial time. Strictly speaking the LP (Primal) may have an integrality gap of

$n = |V|$. However note that if OPT denotes the optimal value of (Primal), then in fact $\lceil OPT \rceil$ is a lower bound on the minimum budget, and by comparing the budget of our solution against this lower bound, we prove the following theorems. We only give proof sketches deferring the full proofs to the appendix.

Theorem 6. *In the non-spreading model, there is a $2\sqrt{n}$ approximation to the* MinBudget *problem in general graphs.*

(Proof Sketch) At a high level, the algorithm recognizes the set of vertices to be vaccinated by time i by looking at the fraction vaccinated by time i. If this fraction is larger than $1/\sqrt{n}$, then the node is vaccinated by day i. We can then show that in the remaining graph, infection can reach t only using paths of length longer than $\sqrt{n}$, and thus there is a cut of size $\sqrt{n}$ which separates s and t. Thus vaccinating this cut as well completes the algorithm. The analysis is slightly subtle and is deferred to the appendix.

An *s-t directed layered graph* with ℓ layers is one where (i) s has only outgoing edges, t has only incoming edges; (ii) all nodes except t can be partitioned into sets $L_0 := \{s\}, L_1, L_2, \ldots, L_\ell$ such that for every node $v \in L_{i+1}$ (so $v \neq t$) and every incoming edge (u, v) of v, we have $u \in L_i$.

Theorem 7. *If the network is a layered directed graph with ℓ layers, then there is a H_ℓ approximation to the* MinBudget *problem. Furthermore, there is an example which shows that the integrality gap of an ℓ layered network is at least $H_\ell = \Omega(\log n)$, where $H_r = 1 + 1/2 + \cdots + 1/r$.*

(Proof Sketch) The algorithm sets capacity $1/iH_\ell$ on each vertex of layer i, for all i, and simply computes a minimum s-t vertex cut. It then divides the cut into ℓ pieces, corresponding to the vertices vaccinated on day i. Using the dual of the LP, we can show that our solution is within H_ℓ of the LP optimum. The integrality gap example is a similar layered graph. We defer the details to the appendix.

4.1 Vaccination on Trees

We now briefly consider the special case when the underlying graph G is a tree rooted at s. We make the following observation which establishes the equivalence between the spreading model and non-spreading model on trees. For the spreading model on general graphs we defined a function $\Gamma(v)$ as a set of all tuples (u, τ) such that if u is vaccinated directly at time τ then the node v will be saved. For a tree, it is easy to observe that a node v will be saved if any of its ancestors is vaccinated directly before the infection reaches v. The fact that the vaccination spreads or not does not matter, since in both cases vaccinating a node saves the entire subtree and nothing else. The optimal strategy will be the same on a given tree irrespective of the vaccination model being spreading or non-spreading. This implies that all the positive results from Section 3 also hold for trees. Since the MinBudget problem on trees with height h yields an instance of MinBudget on an s-t directed graph with h layers, we immediately obtain the following Corollary of Theorem 7.

Corollary 1. *There is a* $O(\log h)$ *approximation for* MinBudget *on trees, where the set* T *is the set of leaves.*

Acknowledgments

The authors would like to thank David Kempe, Sanjeev Khanna, Jon Kleinberg and, Malik Magdon-Ismail for interesting and productive discussions on the Firefighter problem.

References

1. R. Ahuja, T. Magnanti, and J. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.
2. J. Aspnes, K. Chang, and A. Yamposlkiy. Inoculation strategies for victims of viruses and the sum-of-squares partition problem. In *Proc. 16th ACM Symp. on Discrete Algorithms (SODA)*, 2005.
3. G. Baier, T. Erlebach, A. Hall, E. Köhler, H. Schilling and M. Skutella M. Length-Bounded Cuts and Flows. In *Proc. 33rd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 679–690, 2006.
4. N. Bailey. *The Mathematical Theory of Infectious Diseases and its Applications*. Hafner Press, 1975.
5. A. Barabasi, R. Albert, and H. Jeong. Mean-field theory for scale-free random networks. *Physica A*, 272:173–187, 1999.
6. N. Berger, C. Borgs, J. Chayes, and A. Saberi. On the spread of viruses on the internet. In *Proc. 16th ACM Symp. on Discrete Algorithms*, 2005.
7. G. Calinescu, C. Chekuri, M. Pal, J. Vondrak. Maximizing a Monotone Submodular Function subject to a Matroid Constraint. *Integer Programming and Combinatorial Optimization, 12th International IPCO Conference*, Ithaca, NY, USA, 2007.
8. Stuart Crosby, Art Finbow, Bert Hartnell, Raina Moussi, Kate Patterson and Dania Wattar. Designing Fire Resistant Graphs. *Congr. Numerantium* 173, 207-222, 2005.
9. M. Develin, S.G. Hartke. Fire Containment in grids of dimension three or higher. *Discrete Applied Math*,155(17), 2257-2268, 2007.
10. Z. Dezső and A. Barabási. Halting viruses in scale-free networks. *Physical Review E*, 65, 2002.
11. P.A. Dreyer Jr. and F. S. Roberts. Irreversible k-threshold processes: Graph-theoretical threshold models of the spread of disease and of opinion. *Discrete Applied Mathematics*, 157(7), 1615-1627, 2009.
12. R. Engelberg, J. Könemann, S. Leonardi, and J. Naor. Cut problems in graphs with a budget constraint. In *Proc. 7th Latin American Theoretical Informatics Symposium*, 2006.
13. S. Eubank, V. Kumar, M. Marathe, A. Srinivasan, and N. Wang. Structural and algorithmic aspects of massive social networks. In *Proc. 15th ACM Symp. on Discrete Algorithms*, pages 711–720, 2004.
14. U. Feige A threshold of $\ln n$ for approximating set cover, *J. ACM* 45, 634-652, 1998.
15. S. Finbow, A. D. King, Gary MacGillivray, R. Rizzi. The Fire fighter problem on graphs of maximum degree three. *Discrete Mathematics 307* 2094-2105 2007.

16. Stephen Finbow, Gary MacGillivray The Firefighter Problem: A survey of results, directions and questions. manuscript, 2007.
17. M.L. Fisher, G.L. Nemhauser and L.A. Wolsey An analysis of approximations for maximizing submodular set functions - II. *Math. Prog. Study*, 8:7387, 1978.
18. P. Fogarty. *Catching Fire on Grids*, M.Sc. Thesis, Department of Mathematics, University of Vermont 2003.
19. A. Ganesh, L. Massoulie, and D. Towlsey. The effect of network topology on the spread of viruses. In *Proc. 24th IEEE INFOCOM Conference*, 2005.
20. G. Giakkoupis, A. Gionis, E. Terzi, P. Tsaparas. Models and algorithms for network vaccination. *Technical Report C-2005-75*, Department of Computer Science, University of Helsinki, 2005.
21. B.L. Hartnell. Firefighter! An application of domination. Presentation, *25th Manitoba Conference on Combinatorial Mathematics and Computing*, University of Manitoba in Winnipeg, Canada, 1995.
22. B. Hartnell, Q. Li. Firefighting on trees: How bad is the greedy algorithm? *Congr. Numer.* 145, 187-192 2000.
23. A. Hayrapetyan, D. Kempe, M. Pal and Z. Svitkina Unbalanced graph cuts. In *European Symposium on Algorithms (ESA)*, Mallorca, Spain 2005.
24. D. Kempe, J. Kleinberg and E Tardos. Influential Nodes in a Diffusion Model for Social Networks. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP)*, 2005.
25. Andrew King, Gary MacGillivray. The Firefighter Problem For Cubic Graphs. *Discrete Mathematics*, 307: 20942105, 2007.
26. R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the web graph. In *Proc. 41st IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 5765, 2000.
27. C. Leizhen, E. Verbin and L. Yang Firefighting on Trees: (1 - 1/ e)Approximation, Fixed Parameter Tractability and a Subexponential Algorithm, *ISAAC 2008*: 258269, 2008.
28. Cai Leizhen and Wang Weifan. The Surviving Rate of a Graph. to appear in *SIAM Journal on Discrete Mathematics*, 2009.
29. Gary MacGillivray, P. Wang. On The Firefighter Problem *JCMCC 47*, 83-96, 2003.
30. C. Moore and M. Newman. Epidemics and percolation in small-world networks. *Physical Review E*, 61:5678–5682, 2000.
31. M. Nowak and R. May. *Virus Dynamics: Mathematical Principles of Immunology and Virology.* Oxford University Press, 2000.
32. R. Pastor-Satorras and A. Vespignani. Epidemic dynamics in finite scale-free networks. *Physical Review E*, 65, 2002.
33. R. Raz and S. Safra. A sub-constant error-probability low-degree test, and sub-constant error-probability PCP characterization of NP. In *Proc. 29th Ann. ACM Symp. on Theory of Comp., ACM*, 475-484, 1997
34. P. Wang, S. Moeller Fire Control on graphs. *J. Combin. Math. Combin. Comput.* 41, 19-34, 2002.
35. D. Watts and S. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

A Open Problems

In this paper we have looked at two models. One where the rate of spread of the vaccination is the same as the rate of spread of the infection (*spreading model*) and the other where the rate of spread of vaccination is 0, i.e. it does not spread (*non-spreading model*). In reality the rate of spread of the vaccination may lie somewhere in the middle. The lemmas applied for the spreading model fail to hold for these intermediate cases. Hence the study of such models give rise many interesting open problems.

Another area of open problems is where the model of infection spread is more complex. For example, an infected individual might go through stages of incubation and initial symptomatic period causing a small stochastic delay before the person becomes contagious for his neighbors. We can also consider adding probabilities of transmission on the edges or probabilities on nodes indicating their susceptibility to the disease.

Spiltable Job Scheduling Although many graph problems become tractable when the graph is a tree, vaccination on trees is still complex. To give some intuition for why this is the case, we present a difficult scheduling problem that is equivalent to MinBudget on trees.

Consider a problem where there is a set of jobs, each with a deadline for completion. If a job is not completed by its deadline, then it splits into multiple jobs, each with a new deadline. We have a capacity of completing only a fixed number of jobs at every time-step. In this case what strategy should be applied so that all remaining jobs are completed by processing the fewest jobs at each step?

B Background on Submodular Functions and Partition Matroids

Submodular Set Function: Let E be a finite ground set. A real valued set function $f : 2^E \rightarrow \mathbb{R}$ is **normalized**, **nondecreasing/monotone** and **submodular** if it satisfies the following conditions, respectively:

1. $f(\emptyset) = 0$;
2. $f(A) \leq f(B)$ whenever $A \subseteq B \subseteq E$;

3a. $f(A) + f(B) \geq f(A \cup B) + f(A \cap B) \, \forall \, A, B \subseteq E$, or equivalently:
3b. $f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B) \, \forall \, A \subseteq B \subseteq E$ and $e \subseteq E \backslash B$.

Condition (2.) states that the function has to monotonically increasing on the sets. Submodular functions display a property of diminishing returns as shown by condition (3b.) which states that the differential gain obtained by adding an element to a set will always be no greater than the gain obtained by adding the same element to any of its subsets.

Partition Matroid: A set system $(E, \mathcal{F})$, where E is a finite set and $\mathcal{F}$ is a collection of subsets of E, is an **independence system** if it satisfies the following properties:

1. $\emptyset \in \mathcal{F}$;
2. If $X \subseteq Y \in \mathcal{F}$ then $X \in \mathcal{F}$.

Any set $X \in \mathcal{F}$ is called an **independent** set.

An independence system $(E, \mathcal{F})$ is a matroid if it satisfies the following additional property:

3. if $X, Y \in \mathcal{F}$ and $|X| > |Y|$, then there is an $x \in X \backslash Y$ such that $Y \cup \{x\} \in \mathcal{F}$.

A **partition** matroid is a special case of matroid where $(E, \mathcal{F})$ has the following structure:

4. $E = \bigcup_{i=1}^{k} E_i$ is a disjoint union of k sets, $l_1, \ldots, l_k$ are positive integers and,
5. $\mathcal{F} = \{F : F = \bigcup_{i=1}^{k} F_i \text{ where } F_i \subseteq E_i,\ |F_i| \leq l_i \text{ for } i = 1, \ldots k\}$.

In a partition matroid the initial set E of elements is actually a union of mutually exclusive sets E_i and while forming a set with elements of E, no more than a certain number of elements can be picked from each of the E_i sets. The constraint partitions the original set of elements and limits the number of elements that can be used from each partition, hence the name partition matroid.

C Proofs from Section 3

Proof of Lemma 1

Let $d(u, v)$ be the shortest distance between the nodes u and v in graph G, and let $path(u, v)$ be a shortest path between nodes u and v.

We prove this lemma by induction. The base case for time $\tau = 0$ is true since all nodes in neighborhood $N(s, 0)$, which contains only s, are infected. Let us assume that the statement of the lemma is true for all time-steps $\tau = \{1 \ldots k\}$ for some $k \geq 0$. Our assumption implies that all nodes in the neighborhood $N(s, k)$ are either infected or vaccinated at time $\tau = k$. Now consider the set of nodes $\Delta = \{v : v \in N(s, k) \backslash N(s, k)\}$. Each of these nodes have a neighbor in $N(s, k)$ which are either infected or vaccinated according to the hypothesis. According to the definition of the model, all nodes that are infected\vaccinated, infect\vaccinate their neighboring nodes in the next time-step. Hence all nodes in Δ will be either infected or vaccinated at time-step k. This proves the inductive hypothesis and hence statement of the lemma.

Proof of Theorem 1

Consider the claim that the node v is vaccinated if $\Psi \cap \Gamma(v) \neq \emptyset$. Let $(u, \tau_u) \in \Psi \cap \Gamma(v)$.

By Lemma 1 we know that v has to be vaccinated by time $\tau_v \leq d(s, v)$; before the infection reaches it. By the definition of set $\Gamma(v), \tau_u \leq d(s, u)$. It implies that u is vaccinated before infection from s reaches it and hence (u, τ_u) constitutes a valid vaccination strategy. Now we claim that for every node k on $path(u, v)$

the following condition holds true: $d(s,k) \geq \tau_u + d(u,k)$. Lets assume that this is not true, then it would imply the following:

$$\begin{aligned}
&d(s,k) < \tau_u + d(u,k) \\
&\Rightarrow d(s,k) + d(k,v) < \tau_u + d(u,k) + d(k,v) \\
&\Rightarrow d(s,v) < \tau_u + d(u,v) \\
&(\because d(s,v) \leq d(s,k) + d(k,v) \text{ and } d(u,k) + d(k,v) = d(u,v) : k \in path(u,v)) \\
&\Rightarrow \tau_u > d(s,v) - d(u,v)
\end{aligned}$$

But this is a contradiction since we chose τ_u such that $\tau_u \leq d(s,v) - d(u,v)$. Hence the inequality $d(s,k) \geq \tau_u + d(u,k)$ holds for $k \in path(u,v)$. Using this inequality it can be proven by induction that every $k \in path(u,v)$ will be vaccinated indirectly by the vaccination spread from u before the infection from s reaches it. Consider the base case where x is a neighbor of u and lies on $path(u,v)$. At time $\tau_u + 1$ the vaccination from u will reach x and using the above mentioned inequality it can be seen that x will be vaccinated indirectly before infection from s can reach it. The same argument works for the inductive step where we assume that for some k on $path(u,v)$, all nodes in $path(u,k)$ are vaccinated and consider the next node on the same path. This in turn implies that v will be vaccinated.

Now, consider the claim that if node v is vaccinated then $\Gamma(v) \cap \Psi \neq \emptyset$.

If v is vaccinated directly, then it must be vaccinated at time $0 < \tau_v \leq d(s,v)$ (Lemma 1). This element (v, τ_v) will belong to the set $\Gamma(v)$ according to its definition. Hence the claim is true for direct vaccination. If v is vaccinated indirectly then let the first vaccination to reach v be from u. It implies that u had to vaccinated at time $0 < \tau_u \leq d(s,u)$ (Lemma 1). This means $(u, \tau_u) \in \Psi$. Also, since the first node from which vaccination reached v was u and it reached v before the infection from s, then $\tau_u \leq d(s,v) - d(u,v)$. Such an element (u, τ_u) belongs to the set $\Gamma(v)$ by definition.

From the above claims we can conclude that v is vaccinated **iff** $\Psi \cap \Gamma(v) \neq \emptyset$.

Proof of Theorem 2

For the sake of simplicity, we consider the MaxSave Problem where $T = V$ but the same analysis holds for any $T \subseteq V$. We first show that finding A vaccination strategy Ψ that maximizes the number of nodes saved is the same as maximizing a submodular function subject to a partition matroid constraint. Maximizing a set function subject to a matroid constraint means finding a set $S \in \mathcal{F}$ on which this function is maximized.

We define $\mathcal{E}$ as the set of all possible direct vaccination tuples (v, τ) where $v \in V$ and $\tau = 1 \ldots n$. $\mathcal{E}$ can be represented as a union of disjoint subsets as follows:

$$\mathcal{E} = \bigcup_{\tau=1}^{n} \mathcal{E}_\tau \quad \text{where} \quad \mathcal{E}_\tau = \{(v,\tau) | v \in V\}$$

A valid vaccination strategy should directly vaccinate at most B nodes at each time-step. So, if Ψ is a valid vaccination strategy then it should decompose as

$$\Psi = \bigcup_{\tau=1}^{n} \Psi_\tau \quad \text{where} \quad \Psi_\tau \subseteq \mathcal{E}_\tau, |\Psi_i| \leq B$$

Let $\mathcal{S}$ be the set of all possible valid vaccination strategies for the instance $\textsc{MaxSave}(G(V,E), s, B)$. We can immediately observe that the system Thus, if $(\mathcal{E}, \mathcal{S})$ is the partition matroid with with $l_i, \ldots, l_n = B$ (as described in Section B), then any valid vaccination strategy is an independent set in this matroid. The converse need not be true, however note that if Ψ is an independent set in $(\mathcal{E}, \mathcal{S})$, then $\Psi' = \{(v,\tau) \in \Psi : \tau \leq d(s,v)\}$ *is* a valid vaccination strategy; we define below a monotone submodular function $f : 2^{\mathcal{E}} \to \mathbb{R}_+$ such that $f(\Psi) = f(\Psi')$, and $f(\Psi')$ gives the number of nodes saved by the valid vaccination strategy Ψ'. It follows that finding an independent set Ψ in $(\mathcal{E}, \mathcal{S})$ that (approximately) maximizes $f(.)$ yields, via the above conversion to a valid vaccination strategy, an (approximately) optimal (and valid) vaccination strategy.

We now define the function $f : 2^{\mathcal{E}} \to \mathbb{R}_+$. Recall the definition of the set $\Gamma(v)$:

$$\Gamma(v) = \{(u,\tau) | u \in V \text{ and } 0 < \tau \leq \min(d(s,u), d(s,v) - d(u,v))\}$$

In Theorem 1 we proved that a node v is saved iff at least one element of the set $\Gamma(v)$ is part of the vaccination strategy. Hence we can also define an inverse function of Γ for each element $(u, \tau_u) \in E$ as follows,

$$\Gamma^{-1}(u,\tau) = \{v | v \in V, (u,\tau) \in \Gamma(v)\}$$

It follows from the definition of $\Gamma^{-1}(u,\tau)$ that it is the set of all the nodes of the graph G that will be saved if u is vaccinated at time τ. To evaluate the value of $f(\Psi)$, we need to find the set of all the nodes that will be saved by the vaccination strategy in Ψ. This set can be represented as a union of the sets of nodes saved by each element in Ψ.

Using this definition, the value of $f(\Psi)$ can be calculated as follows,

$$f(\Psi) = |\bigcup_{(u,\tau) \in \Psi} \Gamma^{-1}(u,\tau)| \tag{7}$$

Note that if Ψ' is obtained from Ψ as mentioned previously, (i.e., $\Psi' = \{(v,\tau) \in \Psi : \tau \leq d(s,v)\}$), then $f(\Psi) = f(\Psi')$ since $\Gamma^{-1}(u,\tau) = \emptyset$ if $\tau > d(s,u)$. Since $f(\Psi)$ is the size of union of sets, it is inherently non-decreasing. So now we only have to prove that $f(\Psi)$ is submodular.

Consider two valid vaccination strategies A and B. Then,

$$f(A) = |\bigcup_{(u,\tau)\in A} \Gamma^{-1}(u,\tau)| \quad = |\Big\{ \bigcup_{(u,\tau)\in A\setminus B} \Gamma^{-1}(u,\tau)\Big\} \cup \Big\{ \bigcup_{(u,\tau)\in A\cap B} \Gamma^{-1}(u,\tau)\Big\}|$$
$$f(B) = |\bigcup_{(u,\tau)\in B} \Gamma^{-1}(u,\tau)| \quad = |\Big\{ \bigcup_{(u,\tau)\in B\setminus A} \Gamma^{-1}(u,\tau)\Big\} \cup \Big\{ \bigcup_{(u,\tau)\in A\cap B} \Gamma^{-1}(u,\tau)\Big\}|$$

Also,

$$f(A\cup B) = |\Big\{ \bigcup_{(u,\tau)\in A\setminus B} \Gamma^{-1}(u,\tau)\Big\} \cup \Big\{ \bigcup_{(u,\tau)\in B\setminus A} \Gamma^{-1}(u,\tau)\Big\} \cup \Big\{ \bigcup_{(u,\tau)\in A\cap B} \Gamma^{-1}(u,\tau)\Big\}|$$
$$f(A\cap B) = |\bigcup_{(u,\tau)\in A\cap B} \Gamma^{-1}(u,\tau)|$$

From the above equations we easily conclude that $f(A) + f(B) = f(A\cup B) + f(A\cap B)$ for all valid vaccination strategies A, B. This complies with the definition of submodular functions described previously. Hence, the function $f(\cdot)$ is submodular.

In order to prove the theorem we make use of result by Calinescu et al. [7].

Theorem 8. *[7] There is a randomized algorithm which gives with high probability a* $(1 - 1/e)$*-approximation to the problem* $\max\{f(S) : S \in \mathcal{I}\}$*, where* $f : 2^X \to \mathbb{R}_+$ *is a monotone submodular function given by a value oracle, and* $\mathcal{M} = (X, \mathcal{I})$ *is a matroid given by a membership oracle.*

To use this result for approximating MaxSave, we must show how to evaluate $f(\Psi)$ in poly-time, which can be done by simulating the infection process while vaccinating nodes according to Ψ. Therefore, having proved that the set of valid solutions to the MaxSave problem is a partition matroid and the function $f(\cdot)$ is monotone non-decreasing and submodular, we can make use of theorem 8 to give a randomized algorithm which gives with high probability a $(1 - 1/e)$ approximation for the MaxSave problem. Using the result from [17], we also know that a simple greedy algorithm gives a 1/2-approximation. Specifically, this algorithm simply picks nodes to vaccinate one at a time for each time-step, every time picking the node that will save the most.

Proof of Theorem 3

Consider an instance of Set Cover: a collection C of subsets of finite sets of U. Let $|C| = k$ and $|U| = n$. An instance of the MinBudget problem can be constructed as follows:

- Construct node s which will be the root node.
- For each subset $c \in C$, construct a node v_c with a directed edge (s, v_c).
- For each element $e \in U$, construct k nodes $v_{e_1} \ldots v_{e_k}$ such that there is a directed edge (v_c, v_{e_i}) for $i = 1 \ldots k$ if $e \in c$.

This graph forms the input for the MinBudget problem with $T = \{v_{e_1} \dots v_{e_k}\ \forall e \in U\}$. Each element of U is represented by k nodes, each of them connected by an edge coming from the nodes that represent the sets to which the elements belong.

The idea behind this reduction is that an element e is covered in the instance of Set Cover if all the nodes $v_{e_1} \dots v_{e_k}$ are saved. Since T contains all v_{e_i} nodes, the objective of the MinBudget problem is analogous to covering all elements of U. We make the following claims:

Claim. If $C' \subseteq C$ is a valid set cover then by vaccinating $\{v_c | c \in C'\}$ at the first time-step, we can save all nodes in T.

Proof. Since C' is a valid set cover, it means that $\bigcup_{c \in C'} c$ contains all elements of U. Therefore, by construction, there exist edges (v_c, v_{e_i}) such that $c \in C'$ for all $v_{e_i} \in T$. So if the nodes $\{v_c | c \in C'\}$ are vaccinated at the first time-step, then the vaccination will spread to all the v_{e_i} nodes in the second time-step before the infection can reach the nodes. Thus all the v_{e_i} nodes would be saved.

Claim. If all nodes in T can be saved by vaccinating at most B nodes per time-step, then there exists a set cover of size B.

Proof. We assume that $B < k$, since all the elements in U can be trivially covered if k sets are selected. There are only two ways in which a node v_{e_i} can be saved. One is by directly vaccinating a node v_c that has an edge to v_{e_i} during the first time-step, which vaccinates all nodes $v_{e_1} \dots v_{e_k}$ during the second time-step. The other is by directly vaccinating the node v_{e_i}. We thus note that no optimal vaccination strategy will ever directly vaccinate any v_{e_i} nodes in the first time-step, since we can save more nodes along with the same v_{e_i} nodes by vaccinating any v_c node that has an edge to the v_{e_i} node instead. Let Ψ be the vaccination strategy that saves all of T, and let Ψ_1 be the set of nodes v_c that are vaccinated by Ψ during the first time-step. We now prove that $C' = \{c | v_c \in \Psi_1\}$ is a set cover. If it were not, then at the start of the second time-step there is at least one set of nodes $\{v_{e_1} \dots v_{e_k}\}$ which is not vaccinated, with all the nodes v_c adjacent to it already being infected. To save this set of nodes, we must vaccinate them all directly, which requires a budget of k. But this is not possible as the budget for direct vaccinations is only $B < k$. Therefore, C' is a set cover.

It is implied by the above two claims that any α-approximation algorithm for the MinBudget problem would give a α-approximation algorithm for Set Cover. Since Set Cover is $\log n$ inapproximable [33], then the MinBudget problem in $\log n$ inapproximable.

Proof of Theorem 4

Let $G_i = (V_i, E_i)$ be the graph that is generated after the i'th iteration of the greedy algorithm. Also, let T_i be the set of nodes that are common to T and V_i, i.e., $T_i = T \bigcap V_i$.

We can now make the following observations. First notice that all nodes of T_i can be saved by vaccinating at most B nodes per time-step in the graph G_i, since all nodes of T can be saved in this way in the graph G. Therefore running the greedy algorithm with budget B on G_i must save at least half of the nodes of T_i.

Since at each step the the size of T_i reduces by half, after $\log_2 n$ iterations of the greedy algorithm we have saved all the nodes, since $|T| < n$. Therefore, the algorithm RepGreedy which vaccinates $B \log n$ nodes at each time-step saves all nodes of T.

D Proofs from Section 4

Proof of Theorem 5

Consider an instance of the SAVE-t problem $\{G(V,E), s, t\}$. Let $n = |V|$. We construct an instance of the MAXSAVE problem $\{\bar{G}(\bar{V},\bar{E}), \bar{s}, 1, \bar{V}\}$ as follows:

i. Let $\beta \in \mathbb{Z}_+$ such that $\beta > \max\{\frac{\ln 2n^2}{(1-\alpha)\ln n}, 3\}$.
ii. Let $N(t)$ be the set of nodes that are neighbors of t. We construct $\bar{G}(\bar{V},\bar{E})$ as follows:
   a. $\bar{V} = V \cup T$ such that $|T| = n^\beta - n$,
   b. $\bar{E} = E \cup \{(u,v) : u \in N(t) \,\text{and}\, v \in T\}$.
iii. $\bar{s} = s$.

So we essentially construct an instance of MAXSAVE by creating multiple copies of the sink node of the SAVE-t problem. Let OPT be the best possible solution for the SAVE-t instance and $\overline{OPT}$ be the optimal solution to the corresponding MAXSAVE instance. We state the following lemmas.

Lemma 2. *If OPT can save t in the instance of* SAVE*-t, then $\overline{OPT}$ can save at least $n^\beta - n$ nodes in the corresponding instance of* MAXSAVE*.*

Proof. If OPT can save t then $\overline{OPT}$ will be able to save all nodes that belong to T in $\bar{G}$. There are $n^\beta - n$ such nodes.

Lemma 3. *If OPT cannot save t in the instance of* SAVE*-t, then $\overline{OPT}$ will be able to save at most $2n$ nodes in the corresponding instance of* MAXSAVE*.*

Proof. If OPT cannot save t in G then it is clear that $\overline{OPT}$ will not be able to cut off the nodes of T in $\bar{G}$ from the infection since both can vaccinate only one node at each time-step. The infection will reach the nodes of T in at most n time-steps. The only way $\overline{OPT}$ can save nodes of T is by vaccinating them. In n time-steps, the optimal algorithm can save at most n nodes in T. Even if the optimal algorithm manages to save all other $\bar{V}\backslash T$ nodes ($|\bar{V}\backslash T| = n$), it will be able to save at most $n + n = 2n$ nodes.

Now consider the condition for choosing the value β:

$$\begin{aligned}
&\beta > \frac{\ln(2n^2)}{(1-\alpha)\ln n} \\
&\Rightarrow \beta \cdot (1-\alpha)\ln n > \ln(2n^2) \\
&\Rightarrow \beta \ln n - \alpha\beta \ln n > \ln 2 + 2\ln n \\
&\Rightarrow (\beta - 2)\ln n > \ln 2 + \alpha\beta \ln n \\
&\Rightarrow \ln(n^{\beta-2}) > \ln(2n^{\alpha\beta}) \\
&\Rightarrow n^{\beta-2} > 2n^{\alpha\beta} \qquad (8)
\end{aligned}$$

Also we can safely assume that $n \geq 2$. This leads to the following:

$$\begin{aligned}
&\Rightarrow n - 1 \geq 1 \\
&\Rightarrow n^{\beta-2}(n-1) > 1 \quad \ldots (\because n \geq 2,\ \beta > 3) \\
&\Rightarrow n^{\beta-1} - 1 > n^{\beta-2} \qquad (9)
\end{aligned}$$

Using inequalities (8) and (9), it follows that:

$$\begin{aligned}
&n^{\beta-1} - 1 > 2n^{\alpha\beta} \\
&\Rightarrow n^{\beta} - n > n^{\alpha\beta}2n \qquad (10)
\end{aligned}$$

Suppose to the contrary that there exists an algorithm that gives a n^{α} approximation for the MaxSave problem. This means that the algorithm should be able to give a $n^{\alpha\beta}$ approximation for the above constructed instance of MaxSave since $|\bar{V}| = n^{\beta}$. Let the value obtained by running this algorithm on the above instance of MaxSave be ϕ. We consider two cases for the value of ϕ.

1. $\phi \leq 2n$; Since ϕ is an $n^{\alpha\beta}$ approximation, we get the following inequalities,

$$\begin{aligned}
&\overline{OPT} \leq n^{\alpha\beta}\phi \\
&\overline{OPT} \leq n^{\alpha\beta}2n \\
&\overline{OPT} < n^{\beta} - n \qquad \ldots (\text{from inequality (10)})
\end{aligned}$$

   According to Lemma 2 if OPT could save t then $\overline{OPT}$ can never be lower than $n^{\beta} - n$. Hence we can infer that OPT in the original Save-t instance does not save t.
2. $2n < \phi$; Since ϕ is an approximation, the optimal solution is at least as big as ϕ. Hence, $\overline{OPT} > 2n$. According to Lemma 3, this precludes the possibility that OPT in the original Save-t instance does not save t. Hence, we can infer that t can be saved in the original Save-t instance.

Therefore we see that if there exists an n^{α} approximation for MaxSave, then we can solve the Save-t decision problem which is NP-complete in polynomial time. Hence the MaxSave problem is n^{α} inapproximable unless P=NP.

Proof of Theorem 6

For this section, we will assume for ease of exposition, n is a perfect square. We solve the LP (Primal) and get the fractions x_v^τ for all vertices v and $\tau = 1, \ldots, \ell$. Let

$$W_\tau := \{v : \sum_{i=1}^{\tau} x_v^i \geq \frac{1}{\sqrt{n}}\}$$

be the set of vertices such that the total 'fraction' of v vaccinated by time τ is at least $1/\sqrt{n}$. Note that since x_v^τ's are non-negative, $W_1 \subseteq W_2 \subseteq \cdots \subseteq W_\ell$. Also, the first constraint of (Primal) implies

$$|W_i| \leq i\sqrt{n}B, \quad \forall \tau = 1, \ldots, \ell \tag{11}$$

To see this add the constraints for $\tau = 1, ..., i$. Order the vertices in the following way: first order the vertices of W_1, followed by vertices in $W_2 \setminus W_1$, and so on till vertices in $W_{\sqrt{n}} \setminus W_{(\sqrt{n}-1)}$. The first try of the algorithm will then pick $\sqrt{n}B$ vertices in this order for the first $\sqrt{n}$ days. Note that from (11), by time i, all vertices in W_i are vaccinated.

Now consider any path $P = (s, v_1, \cdots, v_k, t)$ from s to t. If $k \leq \sqrt{n}$, then by the second constraint of the LP, one of the vertices v_i must lie in W_i, and thus the infection cannot reach from s to t via this path. Consider now all the paths via which infection does reach t after the first try of the algorithm; from the above discussion all these paths have length at least $\sqrt{n}$. But this means there is a set of vertices X of size at most $n/\sqrt{n} = \sqrt{n}$ whose removal disconnects all these paths. Our final algorithm vaccinates X on the first day along with the first-try algorithm. The proof of the theorem follows from the above discussion.

1. vaccinate all vertices in X at time step 1.
2. Order all the vertices of $W_{\sqrt{n}}$ in the following way: first we have the vertices in W_1, then vertices in $W_2 \setminus W_1$, and so on.
3. At time step $\tau = 1, \ldots, \ell$, vaccinate $\sqrt{n}B$ vertices in this order.

Proof of Theorem 7

We first describe the integrality gap example for (Primal). The example is an ℓ layered graph containing $O(\ell^3)$ vertices. We show a gap of $H_\ell = \Omega(\log n)$, where $H_r = 1 + 1/2 + \cdots + 1/r$.

Example 2. Consider a ℓ layered graph with layers $(L_1, \ldots, L_\ell)$ with an arc from s to all vertices of L_1 and an arc from any vertex in L_ℓ to t. Furthermore, there will be an arc from any vertex in L_i to any vertex in L_{i+1} for $i = 1, \ldots, \ell - 1$. The size of L_i is $i\ell$. Thus, the total number of vertices is $O(\ell^3)$. Note that the integral optimum will be ℓ; one can show if one vaccinates less than $i\ell$ nodes by time i, then the infection spreads to layer $i + 1$. To see a fractional solution, consider the solution where for each i, we set $x_v^i = \frac{1}{iH_\ell}$ for all $v \in L_i$, and $x_v^i = 0$ for all other v. Any path from s to t then has the LHS of constraint 2 precisely equal to $\frac{1}{H_\ell}(1 + 1/2 + \cdots + 1/\ell) = 1$. The value of the solution is $i\ell\frac{1}{iH_\ell} = \frac{\ell}{H_\ell}$.

Now we state the algorithm for layered graphs which is inspired by the above example. The algorithm is just one standard min-s-t vertex cut computation as follows. The LP, in fact the dual, will only be used for the sake of analysis.

1. Set the capacity of each vertex $v \in L_i$ at $\frac{1}{iH_\ell}$.
2. Find the minimum cut in this capacitated network, let it be $(N_1 \cup \cdots \cup N_\ell)$ with $N_i \subseteq L_i$.
3. vaccinate in the following way: on day 1, vaccinate $|N_1|$ vertices of N_1, $|N_2|/2$ vertices of N_2, $|N_3|/3$ vertices of N_3 and so on. On day 2 vaccinate $|N_2|/2$ vertices of N_2, $|N_3|/3$ vertices of N_3, and so on. In general, on day i, vaccinate $|N_i|/i$ vertices of N_i, $|N_{i+1}|/(i+1)$ vertices of N_{i+1} and so on.

Note that the value of the solution is

$$|N_1| + \frac{|N_2|}{2} + \cdots + \frac{|N_\ell|}{\ell}$$

First we show that the above algorithm is valid. To see this note that by day i, the whole set N_i is vaccinated. Therefore, since any path $(s, v_1, v_2, \ldots, v_k, t)$ from s to t (where $k \leq \ell$), must have $v_i \in N_i$ for some i (N_i's form a cut), we are done. We need to show that our solution is not too large.

To do this, we use (Dual), and the properties of the layered network. First lets look at the LHS of the second constraint in the dual. We claim that for layered networks, if vertex v lies in L_i,

$$\sum_{P : v \in P^{(\tau)}} f_P = \begin{cases} \sum_{P : v \in P} f_P & \text{if} \quad t \leq i \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

This is because *every path that contains $v \neq t$, contains it at exactly the ith position.* Now we exhibit a dual solution of value $\frac{1}{H_\ell}\left(|N_1| + \frac{|N_2|}{2} + \cdots + \frac{|N_\ell|}{\ell}\right)$, and we will be done. To do so, note that corresponding to the minimum vertex cut $(N_1 \cup \ldots \cup N_\ell)$, we have a feasible flow f of the same value. Furthermore, because of the capacity constraints, we get for $v \in L_i$,

$$\sum_{P : v \in P} f_P \leq \frac{1}{iH_\ell} \tag{13}$$

Construct the dual solution with this f and let $z_i = \frac{1}{iH_\ell}$ for $i = 1, \ldots, \ell$. Note that the first constraint of the dual is satisfied. The second constraint follows from (12) and (13), and the fact that z_i's are decreasing. Note that the value of the dual is equal to the capacity of the minimum cut which is

$$\frac{1}{H_\ell}|N_1| + \frac{1}{2H_\ell}|N_2| + \cdots + \frac{1}{\ell H_\ell}|N_\ell|$$

Remark 1. The above proof is incomplete, for the fraction $|N_i|/i$ need not be integral and one needs to be careful about it (for what does it mean to pick a fractional number of vertices). However, one can obtain the same guarantee using the fact which follows from network flows (see [1], for example): given a matrix M with possibly fractional entries, one can obtain another matrix M' which is integral and whose row and column sums are floors or ceilings of the row and column sums of M'.

E Other Results and Extensions for Spreading Vaccination Model

1. Another objective that we consider is that of minimizing the *total* number of nodes directly vaccinated in order to save a subset of nodes. This objective could be important when there is not only a limitation on the number of direct vaccinations at each time-step, but also on the total number of vaccinations that can performed. Using the IP mentioned in section 3.3 gives us a $(\log n, \log n)$ bicriteria approximation that would directly vaccinate at most $O(\log n)$ times the optimal number of nodes over all time-steps while vaccinating at most $O(\log n{\cdot}B)$ nodes at each time-step. This problem can be represented as an Integer Program (IP). For representing the tuple (v, τ) (immunizing vertex v at time τ), we define a variable $x_v^\tau \,\forall v \in V, \tau = 1 \ldots |V|$. The value of $x_v^\tau = 1$ if node v is directly immunized at time τ and 0 otherwise. Also T is the set of nodes that has to be saved. The IP is as follows,

$$\text{Minimize} \quad \sum_{\tau=1}^{n} \sum_{v \in V} x_v^\tau$$

$$\text{Subject to:} \quad x_v^\tau \in \{0,1\} \qquad \forall v \in V, \tau = 1 \ldots |V| \tag{14}$$

$$\sum_{v \in V} x_v^\tau \leq B \qquad \tau = 1 \ldots |V| \tag{15}$$

$$\sum_{(u,\tau) \in \Gamma(v)} x_u^t \geq 1 \qquad \forall v \in T \tag{16}$$

Condition (15) refers to the fact that only B nodes can be immunized at each time-step and condition (16) refers to the fact that the immunization strategy has to select at least one tuple from $\Gamma(v)$ for all $v \in T$. By relaxing the condition on x_v^τ such that $0 \leq x_v^\tau \leq 1$, we get a linear program (LP) whose solution can be found in polynomial time.

Once we obtain the solution to the LP, say ${x_v^t}^*$, we apply the following randomized rounding technique to the get the immunization strategy Ψ:

- Add element (v, τ) to Ψ with probability $x_v^{\tau *}$.
- Repeat the above step $O(\log n)$ times. (Here $n = |V|$)

It becomes necessary to repeat the selection process so that all the nodes in set T are saved with high probability.

Analysis: The analysis of this algorithm is similar to the analysis of the randomized rounding algorithm for Set Cover. The main difference is that while in Set Cover the goal is to guarantee that the total number of sets chosen is small w.h.p., in our case we must guarantee that with high probability, there is no time-step where the number of nodes immunized is large. For this we require a slightly stronger analysis.

2. We also consider a requirement that only nodes that have infected neighbors can be vaccinated directly. This is important because vaccinating a node that is not directly a neighbor of an infected node means that the strategy is guaranteeing that a certain node will be infected when it could have been saved. When nodes represent people and infection a fatal disease, saving people who are next in the line of fire, even though it is not optimal, can become a necessity.

In such cases the techniques discussed in previous sections can be applied and we get similar results.

By Lemma 1 we can infer that at time τ all vulnerable nodes that have infected neighbors lie at distance of τ from s. Since we know that a node v will be directly vaccinated at time $\tau = d(s, v)$ or not at all. For the MaxSave and MinBudget problems, this additional constraint on choosing a valid vaccination strategy preserves the partition matroid property of the set of strategies. Hence the theorems used to prove the approximation guarantees still hold.

3. Consider the weighted version of the MaxSave problem where all nodes have weights $w_v \in \mathbb{Z}^+$ associated with them that signify their importance. Now the objective would be to save a set of nodes with maximum total weight. The results for MaxSave apply to this version as well.

F Bi-Criteria Approximation for MinBurn on Trees

We are also interested in a complementary problem to MaxSave, defined as follows.

MinBurn(G, B, s, T)
Instance: A rooted graph $(G(V, E), s)$, integer $B \geq 1$ and $T \subseteq V$
Objective: Find a valid vaccination strategy Ψ such that if s is the only infected node at time 0, then at the end of the above process the number of infected nodes that belong to T is minimized.

Since we proved that MaxSave in NP-complete, it follows that the MinBurn problem is also NP-complete. For the non-spreading model in general graphs, the argument in Section 4 also shows that MinBurn is inapproximable. For the spreading model, or the special case of trees, however, the arguments for MaxSave no longer work, so it is necessary to use a different approach.

We approach this problem using the technique suggested by Hayrapetyan et al [23], which involves a graph with an identified source node and seeks to find a cut minimizing the number of nodes on the source side of the cut. We obtain the following result,

Theorem 9. *A bi-criteria* $(2,2)$ *approximation in expectation can be obtained for the* MinBurn *problem on trees in poly-time.*

The algorithm gives a solution where the number of nodes infected is at most twice as much as in the optimal solution, and the number of nodes vaccinated at each time-step is $2B$ in expectation, for a given budget B. However, this result *does not* come with a high probability guarantee. In other words, while the expected number of nodes vaccinated at each time-step is small, the probability that there exists *some* time-step where the number vaccinated will be large can be quite high.

Problem Formulation We try to solve the problem by expressing it as an Integer program(IP), then relaxing it to a Linear Program(LP) and then using the randomized rounding technique to obtain a solution.

First, vaccinating a node in a tree is identical to cutting an edge leading to its parent, so from now on we will talk about cutting edges instead of vaccinating nodes. We have already established that at each time-step the optimal strategy will vaccinate nodes at a fixed distance away from the root. This helps us in formulating the IP because now our algorithm only needs to cut B edges that are incident to the root on the first step. Similarly in the second time-step, the algorithm only needs to cut edges at a distance 1 from the root. Therefore, the algorithm only needs to consider edges at a distance of 0 at time-step 0, edges at distance 1 at time-step 1 and so forth.

For representing the vertices, let there be a variable x_v for each vertex and for the the edges let there be a variable y_e. The variable x_v assumes the value 1 if the vertex is burnt or 0 if it is saved. The variable y_e assumes the value 1 if cut or 0 if not. Now, we can only cut a maximum of B edges in every layer. Also once an ancestor of a vertex is saved, it no longer needs to saved. Hence an IP to minimize the number of burnt vertices can be written as follows:

$$\begin{array}{rll} \text{Minimize} \sum_{v\in V} x_v & & \\ \text{subject to} \quad & x_v = 0,1; \quad x_s = 1 & \\ & y_e = 0,1 & \\ & y_e \geq x_u - x_v & \text{for all } e=(u,v)\in E \\ \sum_{e\in i} & y_e \leq B & \text{for each layer } i \end{array}$$

By relaxing the conditions on x_v and y_e such that $0 \leq x_v \leq 1$ and $0 \leq y_e \leq 1$ we get a LP whose solution can found in linear time.

Algorithm Once we obtain the solution to the LP, say $\{x^*, y^*\}$, the following randomized rounding technique can be applied.

- Let c be chosen uniformly at random from the interval [0.5,1].
- Let $S = \{v|x^*_v \geq c\}$, and cut all edges leaving S.

Analysis This algorithm lets at most twice the number of vertices be burnt as compared to the optimal solution (OPT). To prove this, observe that for each $v \in S$, $x_v^* \geq 1/2$. Also $\sum_{v \in V} x_v^* \geq \sum_{v \in S} x_v^* \geq |S|/2$. Now, since we know that $OPT \geq \sum_{v \in V} x_v^*$, therefore,

$$OPT \geq |S|/2 \tag{17}$$

Now, we bound the number of edges that are cut. An edge $e = (u, v)$ is cut only if c lies between x_v^* and x_u^*. The probability of that happening is thus at most $|x_v^* - x_u^*|/0.5$. So from the constraints of the LP we have,

$$\begin{aligned} \Pr[e \text{ is cut}] &\leq |x_v^* - x_u^*| \cdot 2 \\ &\leq 2 \cdot y_e \qquad \forall e \in E \end{aligned} \tag{18}$$

In order to analyze the total number of edges that are cut by this algorithm, let us define the following random variables. Let X_i for $i = 1, 2, 3 \ldots$ denote the number of edges cut at layer i.

We can see that the expected number of edges that are cut at layer i is at most the sum of the probabilities of the edges being cut over all the edges at the i^{th} layer. From (18) and the constraints for the LP, we have the following,

$$\begin{aligned} E[X_i] &\leq \textstyle\sum_{e \in i} \Pr[e \text{ is cut}] \\ &\leq \textstyle\sum_{e \in i} 2 \cdot y_e \\ &\leq 2B \qquad \forall i \text{ layers} \end{aligned} \tag{19}$$

Hence, we can see that the number of edges cut at each layer is bounded by $2B$ in expectation.

We thus proved that the number of edges cut at each layer is at most $2B$ in expectation. However, the complex correlation between the probabilities of edges being cut prevents us from using Chernoff bounds or Martingale methods to bound the occurrences of the bad events, and thus the probability that at some time-step, the number of edges cut is much larger than B can be high.