Using Social Networks to Aid Homeless Shelters: Dynamic Influence Maximization under Uncertainty - An Extended Version

Amulya Yadav, Hau Chan[1], Albert Jiang[1], Haifeng Xu, Eric Rice, Milind Tambe
University of Southern California, Los Angeles, CA 90089
[1]Trinity University, San Antonio, TX 78212
{amulyaya, haifengx, ericr, tambe}@usc.edu [1]{hchan, xjiang}@trinity.edu

ABSTRACT

This paper presents HEALER, a software agent that recommends sequential intervention plans for use by homeless shelters, who organize these interventions to raise awareness about HIV among homeless youth. HEALER's sequential plans (built using knowledge of social networks of homeless youth) choose intervention participants strategically to maximize influence spread, while reasoning about uncertainties in the network. While previous work presents influence maximizing techniques to choose intervention participants, they do not address three real-world issues: (i) they *completely fail* to scale up to real-world sizes; (ii) they do not handle deviations in execution of intervention plans; (iii) constructing real-world social networks is an expensive process. HEALER handles these issues via four major contributions: (i) HEALER casts this influence maximization problem as a POMDP and solves it using a novel planner which scales up to previously unsolvable real-world sizes; (ii) HEALER allows shelter officials to modify its recommendations, and updates its future plans in a deviation-tolerant manner; (iii) HEALER constructs social networks of homeless youth at low cost, using a Facebook application. Finally, (iv) we show hardness results for the problem that HEALER solves. HEALER will be deployed in the real world in early Spring 2016 and is currently undergoing testing at a homeless shelter.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Algorithms, HIV Prevention

Keywords

POMDP; Influence Maximization; Social Networks; Multi-Step planning

1. INTRODUCTION

HIV-AIDS kills 2 million people worldwide every year [26]. In USA alone, AIDS kills around 10,000 people per annum [2]. HIV has an extremely high incidence among homeless youth, as they are more likely to engage in high HIV-risk behaviors (e.g., unprotected

sexual activity, injection drug use) than other sub-populations. In fact, previous studies show that homeless youth are at 10X greater risk of HIV infection than stably housed populations [5]. Thus, any attempt at eradicating HIV crucially depends on our success at minimizing rates of HIV infection among homeless youth.

As a result, many homeless shelters organize intervention camps for homeless youth in order to raise awareness about HIV prevention and treatment practices. These intervention camps consist of day-long educational sessions in which the participants are provided with information about HIV prevention measures [20].

However, due to financial/manpower constraints, the shelters can only organize a limited number of intervention camps. Moreover, in each camp, the shelters can only manage small groups of youth ($\sim$3-4) at a time (as emotional and behavioral problems of youth makes management of bigger groups difficult). Thus, the shelters prefer a series of small sized camps organized sequentially [19]. As a result, the shelter cannot intervene on the entire target (homeless youth) population. Instead, it tries to maximize the spread of awareness among the target population (via word-of-mouth influence) using the limited resources at its disposal. To achieve this goal, the shelter uses the friendship based social network of the target population to strategically choose the participants of their limited intervention camps. Unfortunately, the shelters' job is further complicated by a lack of complete knowledge about the social network's structure [17]. Some friendships in the network are known with certainty whereas there is uncertainty about other friendships.

Thus, the shelters face an important challenge: they need a sequential plan to choose the participants of their sequentially organized interventions. This plan must address four key points: (i) it must deal with network structure uncertainty; (ii) it needs to take into account new information uncovered during the interventions, which reduces the uncertainty in our understanding of the network; (iii) the plan needs to be deviation tolerant, as sometimes homeless youth may refuse to be an intervention participant, thereby forcing the shelter to modify its plan; (iv) the intervention approach should address the challenge of gathering information about social networks of homeless youth, which usually costs thousands of dollars and many months of time [19].
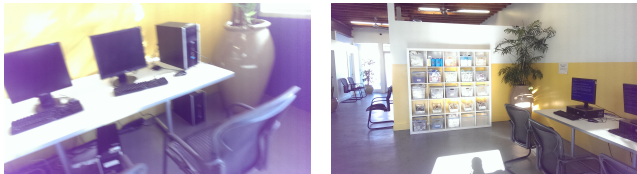
In this paper, we model the shelters' problem by introducing the *Dynamic Influence Maximization under Uncertainty* (or DIME) problem. The sequential selection of intervention participants under network uncertainty in DIME sets it apart from any other previous work on influence maximization, which mostly focuses on single shot choices [1, 25, 10, 14]. Additionally, in previous work, PSINET [29], a POMDP based tool, was proposed for solving this problem, but it has three limitations. First, PSINET completely fails to scale up to the problem's requirements; running slowly

and out of memory. It runs very slowly for moderate-sized networks, and runs out of memory as the network is scaled up. Worse still, even on these moderate sized networks, it runs out of memory when the number of participants in an intervention are increased (as shown later). Second, PSINET did not explicitly allow officials to modify its recommended plans if some participants refuse to attend the intervention. Third, PSINET requires entire social networks of homeless youth as input, while homeless shelters lack the money/time/manpower required to generate these input networks.

In this paper, we build a new software agent, HEALER (**H**ierarchical **E**nsembling based **A**gent which p**L**ans for **E**ffective **R**eduction in HIV Spread), to provide an end-to-end solution to the DIME problem. HEALER addresses PSINET's shortcomings via four contributions. First, HEALER casts the DIME problem as a Partially Observable Markov Decision Process (POMDP) and solves it using HEAL (**H**ierarchical **E**nsembling **A**lgorithm for p**L**anning), a novel POMDP planner which quickly generates high-quality recommendations (of intervention participants) for homeless shelter officials. HEAL uses a hierarchical ensembling heuristic to ensure low memory utilization, thereby enabling scale up. HEAL hierarchically subdivides our *original POMDP* at two layers: (i) In the top layer, graph partitioning techniques are used to divide the *original POMDP* into *intermediate POMDPs*; (ii) In the second level, each of these *intermediate POMDPs* is further simplified by sampling uncertainties in network structure repeatedly to get *sampled POMDPs*; (iii) Finally, we use aggregation techniques to combine the solutions to these simpler POMDPs, in order to generate the overall solution for the *original POMDP*. Our simulations show that even on small settings, HEAL achieves a 100X speed up and 70% improvement in solution quality over PSINET; and on larger problems *where PSINET is unable to run at all*, HEAL continues to provide high quality solutions quickly. Second, HEALER tolerates deviations in execution of intervention plans, as it periodically receives feedback from shelter officials about executed plans, reasons about any deviations from its recommended plans, and updates its plan accordingly to maximize solution quality. Third, HEALER quickly gathers information about the homeless youth social network (at low cost) by interacting with youth via a Facebook application. Fourth, we analyze several novel theoretical aspects of the DIME problem, which illustrates its hardness.

(a) Computers at Homeless Shelter where HEALER is deployed

(b) Emergency Resource Shelf at the Homeless Shelter

Figure 1: Facilities at our Collaborating Homeless Shelter

We deploy HEALER in a real-world pilot study, in collaboration with a homeless shelter (name withheld for anonymity), which provides food and lodging to homeless youth aged 12-25. They provide these facilities for ∼55-60 homeless youth every day. They also operate an on-site medical clinic where free HIV and Hepatitis-C testing is provided. HEALER has been reviewed by officials at our collaborating homeless shelter and their feedback has been positive. We are currently preparing to register 100 youth in our deployment of HEALER at this shelter. To the best of our knowledge, this pilot study represents the first real-world evaluation of such sequential influence maximization algorithms. We expect deployment to commence in early Spring 2016.

2. RELATED WORK

First, we discuss work related to influence maximization. There are many algorithms for finding 'seed sets' of nodes to maximize influence spread in networks [10, 14, 1, 25]. However, all these algorithms assume *no uncertainty in the network structure* and select a single seed set. In contrast, we select several seed sets sequentially in our work to select intervention participants. Also, our problem takes into account uncertainty about the network structure and influence status of network nodes (i.e., whether a node is influenced or not). Finally, unlike [10, 14, 1, 25], we use a different diffusion model as we explain later. Golovin et. al. [8] introduced adaptive submodularity and discussed adaptive sequential selection (similar to our problem), and they proved that a Greedy algorithm has a $(1 - 1/e)$ approximation guarantee. However, unlike our work, they assume no uncertainty in network structure. Also, while our problem can be cast into the adaptive stochastic optimization framework of [8], our influence function is not adaptive submodular (see Section 5), because of which their Greedy algorithm loses its approximation guarantees.

Next, we discuss literature from *social work*. The general approach to these interventions is to use Peer Change Agents (PCA) (i.e., peers who bring about change in attitudes) to engage homeless youth in interventions, but most studies don't use network characteristics to choose these PCAs [22]. A notable exception is Valente et. al. [27], who proposed selecting intervention participants with highest *degree centrality* (the most ties to other homeless youth). However, previous studies [3, 29] show that *degree centrality* performs poorly, as it does not account for potential overlaps in influence of two high degree centrality nodes.

The final field of related work is planning for reward/cost optimization. We only focus on the literature on Monte-Carlo (MC) sampling based online POMDP solvers since this approach allows significant scale-up [21]. The POMCP solver [23] uses Monte-Carlo UCT tree search in online POMDP planning. Also, Somani et. al. [24] present the DESPOT algorithm, that improves the worst case performance of POMCP. Our initial experiments with POMCP and DESPOT showed that they run out of memory on even our small sized networks. A recent paper [29] introduced PSINET-W, a MC sampling based online POMDP planner. We have discussed PSINET's shortcomings in Section 1 and how we remedy them. In particular, as we show later, HEALER scales up whereas PSINET fails to do so. *HEALER's algorithmic approach also offers significant novelties in comparison with PSINET (see Section 6.1)*. Further, a recent paper [15] looks at an extension of the same problem by considering the case that not all nodes in the network are known ahead of time (as opposed to our work where we only assume that some edges are not known ahead of time). However, unlike our work, they do not consider sequential selection of node subsets.

3. HEALER'S DESIGN

We now explain the high-level design of HEALER. It consists of two major components: (i) a Facebook application for gathering information about social networks; and (ii) a DIME Solver, which solves the DIME problem (introduced in Section 5). We first explain HEALER's components and then explain HEALER's design.

Facebook Application: HEALER gathers information about social ties in the homeless youth social network by interacting with youth via a Facebook application. We choose Facebook for gathering information as Young et. al. [31] show that ∼80% of home-
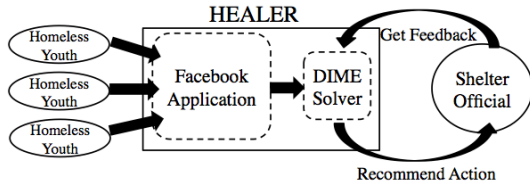
Figure 2: HEALER's Design

less youth are active on Facebook. Once a fixed number of homeless youth register in the Facebook application, HEALER parses the Facebook contact lists of all the registered homeless youth and generates the social network between these youth. HEALER adds a link between two people, if and only if both people are (i) friends on Facebook; and (ii) are registered in its Facebook application. Unfortunately, there is *uncertainty* in the generated network as friendship links between people who are only friends in real-life (and not on Facebook) are not captured by HEALER.

Previously, homeless shelters gathered this social network information via tedious face-to-face interviews with homeless youth, a process which cost thousands of dollars and many months of time. HEALER's Facebook application allows homeless shelters to quickly generate a (partial) homeless youth social network at low cost. This Facebook application has been tested by our collaborating homeless shelter with positive feedback.

DIME Solver: The DIME Solver then takes the approximate social network (generated by HEALER's Facebook application) as input and solves the DIME problem (formally defined in Section 5) using our new algorithm (explained in Section 6.1). HEALER provides the solution of this DIME problem as a series of recommendations (of intervention participants) to homeless shelter officials.

HEALER Design: HEALER's design (shown in Figure 2), begins with the Facebook application constructing an *uncertain* network (as explained above). HEALER has a *sense-reason-act* cycle; where it repeats the following process for T interventions.

It *reasons* about different long-term plans to solve the DIME problem, it *acts* by providing DIME's solution as a recommendation (of intervention participants) to homeless shelter officials. The officials may choose to not use HEALER's recommendation in selecting their intervention's participants. Upon the intervention's completion, HEALER *senses* feedback about the conducted intervention from the officials. This feedback includes new observations about the network, e.g., uncertainties in some links may be resolved as intervention participants are interviewed by the shelter officials (explained more in Section 5). HEALER uses this feedback to update and improve its future recommendations.

4. NETWORK GENERATION

First, we explain our model for influence spread in *uncertain social networks*. Then, we describe how HEALER generates a social network using its' Facebook application.

4.1 Background

We represent social networks as directed graphs (consisting of *nodes* and *directed edges*) where each *node* represents a person in the social network and a *directed edge* between two nodes A and B (say) represents that node A *considers* node B as his/her friend. *We assume directed-ness of edges as sometimes homeless shelters assess that the influence in a friendship is very much uni-directional; and to account for uni-directional follower links on Facebook.* Otherwise friendships are encoded as two uni-directional links.

Uncertain Network: The uncertain network is a directed graph $G = (V, E)$ with $|V| = N$ nodes and $|E| = M$ edges. The edge set E consists of two disjoint subsets of edges: E_c (the set of certain edges, i.e., friendships which we are certain about) and E_u (the set of uncertain edges, i.e., friendships which we are uncertain about). Note that uncertainties about friendships exist because HEALER's Facebook application misses out on some links between people who are friends in real life, but not on Facebook.

To model the uncertainty about missing edges, every uncertain edge $e \in E_u$ has an existence probability $u(e)$ associated with it, which represents the likelihood of "existence" of that uncertain edge. For example, if there is an uncertain edge (A, B) (i.e., we are unsure whether node B is node A's friend), then $u(A, B) = 0.75$ implies that B is A's friend with a 0.75 chance. In addition, each edge $e \in E$ (both certain and uncertain) has a propagation probability $p(e)$ associated with it. A propagation probability of 0.5 on directed edge (A, B) denotes that if node A is influenced (i.e., has information about HIV prevention), it influences node B (i.e., gives information to node B) with a 0.5 probability in each subsequent time step (our full influence model is defined below). This graph G with all relevant $p(e)$ and $u(e)$ values represents an uncertain network and serves as an input to the DIME problem. Figure 3 shows an uncertain network on 6 nodes (A to F) and 7 edges. The dashed and solid edges represent uncertain (edge numbers 1, 4, 5 and 7) and certain (edge numbers 2, 3 and 6) edges, respectively. Next, we explain the influence diffusion model that we use in HEALER.

Influence Model We use a variant of the independent cascade model [30]. In the standard independent cascade model, all nodes that get influenced at round t get a **single** chance to influence their un-influenced neighbors at time $t + 1$. If they fail to spread influence in this **single** chance, they don't spread influence to their neighbors in future rounds. Our model is different in that we assume that nodes get **multiple** chances to influence their un-influenced neighbors. If they succeed in influencing a neighbor at a given time step t', they stop influencing that neighbor for all future time steps. Otherwise, if they fail in step t', they try to influence again in the next round. This variant of independent cascade has been shown to empirically provide a better approximation to real influence spread than the standard independent cascade model [4, 30]. Further, we assume that nodes that get influenced at a certain time step remain influenced for all future time steps. We now explain how HEALER generates an *uncertain social network*.
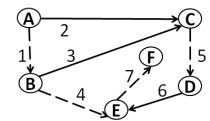

Figure 3: Uncertain Network

4.2 HEALER's Facebook application

HEALER generates an *uncertain network* by (i) using its Facebook application to generate a network with no uncertain edges; (ii) using well known link prediction techniques such as KronEM [11] to infer existence probabilities $u(e)$ for all possible *missing* edges that are not present in the network; (iii) deciding on a threshold probability τ (in consultation with homeless shelter officials), so that we *only* add a *missing* edge as an uncertain edge if its inferred existence probability $u(e) > \tau$; and (iv) asking homeless shelter officials to provide $p(e)$ estimates for network edges.

Choosing τ: Rice et. al [18] show that real-world homeless youth networks are *relatively sparse*. Thus, shelter officials choose the threshold probability value τ such that the number of uncertain edges that get added because of τ does not make our input uncertain network *overly dense*. Next, we introduce the DIME problem.

5. DIME PROBLEM

We now provide some background information that helps us define a precise problem statement for DIME. After that, we will show some hardness results about this problem statement.

Given the *uncertain network* as input, HEALER runs for T *rounds* (corresponding to the number of interventions organized by the homeless shelter). In each round, HEALER chooses K *nodes* (youth) as intervention participants. These participants are assumed to be influenced post-intervention with certainty. Upon influencing the chosen nodes, HEALER '*observes*' the true state of the *uncertain edges* (friendships) out-going from the selected nodes. This translates to asking intervention participants about their 1-hop social circles, which is within the homeless shelter's capabilities [18].

After each round, influence spreads in the network according to our influence model for L *time steps*, before we begin the next round. This L represents the time duration in between two successive intervention camps. *In between rounds, HEALER does not observe the nodes that get influenced during L time steps.* HEALER only knows that explicitly chosen nodes (our intervention participants in all past rounds) are influenced. Informally then, given an uncertain network $G_0 = (V, E)$ and integers T, K, and L (as defined above), HEALER finds an online policy for choosing *exactly* K nodes for T successive rounds (interventions) which maximizes influence spread in the network at the end of T rounds.

We now provide notation for defining HEALER's policy formally. Let $\mathcal{A} = \{A \subset V \text{ s.t. } |A| = K\}$ denote the set of K sized subsets of V, which represents the set of possible choices that HEALER can make at every time step $t \in [1, T]$. Let $A_i \in \mathcal{A} \, \forall i \in [1, T]$ denote HEALER's choice in the ith time step. Upon making choice A_i, HEALER '*observes*' uncertain edges adjacent to nodes in A_i, which updates its understanding of the network. Let $G_i \, \forall \, i \in [1, T]$ denote the uncertain network resulting from G_{i-1} with *observed* (additional edge) information from A_i. Formally, we define a history $H_i \, \forall \, i \in [1, T]$ of length i as a tuple of past choices and observations $H_i = \langle G_0, A_1, G_1, A_2, .., A_{i-1}, G_i \rangle$. Denote by $\mathcal{H}_i = \{H_k \text{ s.t. } k \leqslant i\}$ the set of all possible histories of length less than or equal to i. Finally, we define an i-step policy $\Pi_i : \mathcal{H}_i \to \mathcal{A}$ as a function that takes in histories of length less than or equal to i and outputs a K node choice for the current time step. We now provide an explicit problem statement for DIME.

PROBLEM 1. *__DIME Problem__ Given as input an uncertain network $G_0 = (V, E)$ and integers T, K, and L (as defined above). Denote by $\mathcal{R}(H_T, A_T)$ the expected total number of influenced nodes at the end of round T, given the T-length history of previous observations and actions H_T, along with A_T, the action chosen at time T. Let $\mathbb{E}_{H_T, A_T \sim \Pi_T}[\mathcal{R}(H_T, A_T)]$ denote the expectation over the random variables $H_T = \langle G_0, A_1, .., A_{T-1}, G_T \rangle$ and A_T, where A_i are chosen according to $\Pi_T(H_i) \, \forall \, i \in [1, T]$, and G_i are drawn according to the distribution over uncertain edges of G_{i-1} that are revealed by A_i. The objective of DIME is to find an optimal T-step policy $\Pi_T^* = \text{argmax}_{\Pi_T} \mathbb{E}_{H_T, A_T \sim \Pi_T}[\mathcal{R}(H_T, A_T)]$.*

Next, we show hardness results about the DIME problem. First, we analyze the value of having complete information in DIME. Then, we characterize the computational hardness of DIME.

The Value of Information. We characterize the impact of insufficient information (about the uncertain edges) on the achieved solution value. We show that no algorithm for DIME is able to provide a good approximation to the *full-information solution value* (i.e., the best solution achieved w.r.t. the underlying ground-truth network), even with infinite computational power.
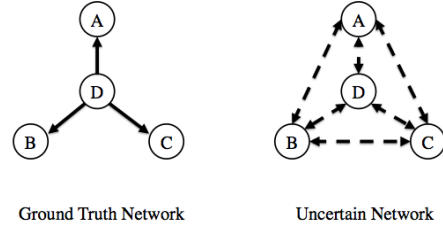

Figure 4: Counter-example for Theorem 5.1

THEOREM 5.1. *Given an uncertain network with n nodes, for any $\epsilon > 0$, there is no algorithm for the DIME problem which can guarantee a $n^{-1+\epsilon}$ approximation to OPT_{full}, the full-information solution value.*

PROOF. We prove this statement by providing a counter-example in the form of a specific (ground truth) network for which there can exist no algorithm which can guarantee a $n^{-1+\epsilon}$ approximation to OPT_{full}. Consider an input to the DIME problem, an *uncertain network* with n nodes with $2 * \binom{n}{2}$ uncertain edges between the n nodes, i.e., it's a completely connected uncertain network consisting of *only* uncertain edges (an example with $n = 3$ is shown in Figure 4). Let $p(e) = 1$ and $u(e) = 0.5$ on all edges in the *uncertain network*, i.e., all edges have the same propagation and existence probability. Let $K = 1$, $L = 1$ and $T = 1$, i.e., we just select a single node in one shot (in a single round).

Further, consider a star graph (as the ground truth network) with n nodes such that propagation probability $p(e) = 1$ on all edges of the star graph (shown in Figure 1). Now, any algorithm for the DIME problem would select a single node in the *uncertain network* uniformly at random with equal probability of $1/n$ (as information about all nodes is symmetrical). In expectation, the algorithm will achieve an expected reward $\{1/n \times (n)\} + \{1/n \times (1) + ... + 1/n \times (1)\} = 1/n \times (n) + (n-1)/n \times 1 = 2 - 1/n$. However, given the ground truth network, we get $OPT_{full} = n$, because we always select the star node. As n goes to infinity, we can at best achieve a n^{-1} approximation to OPT_{full}. Thus, no algorithm can achieve a $n^{-1+\epsilon}$ approximation to OPT_{full} for any $\epsilon > 0$. $\square$

Computational Hardness. We now analyze the hardness of computation in the DIME problem in the next two theorems.

THEOREM 5.2. *The DIME problem is NP-Hard.*

PROOF. Consider the case where $E_u = \Phi$, $L = 1$, $T = 1$ and $p(e) = 1 \, \forall \, e \in E$. This degenerates to the standard influence maximization problem which is shown to be NP-Hard [10]. Thus, the DIME problem is also NP-Hard. $\square$

Some NP-Hard problems exhibit nice properties that enable approximation guarantees for them. Golovin et. al. [8] introduced adaptive submodularity, an analog of submodularity for adaptive settings. Presence of adaptive submodularity ensures that a simply greedy algorithm provides a $(1 - 1/e)$ approximation guarantee w.r.t. the optimal solution defined on the *uncertain network*. However, as we show next, while DIME can be cast into the adaptive stochastic optimization framework of [8], our influence function is not adaptive submodular, because of which their Greedy algorithm does not have a $(1 - 1/e)$ approximation guarantee.

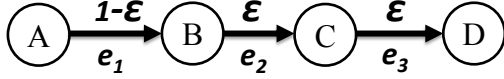THEOREM 5.3. *The influence function of DIME is not adaptive submodular.*

Figure 5: Failure of Adaptive Submodularity

PROOF. The definition of adaptive submodularity requires that the expected marginal increase of influence by picking an additional node v is more when we have less observation. Here the expectation is taken over the random states that are consistent with current observation. We show that this is not the case in DIME problem. Consider a path with 4 nodes a, b, c, d and three *directed* edges $e_1 = (a, b)$ and $e_2 = (b, c)$ and $e_3 = (c, d)$ (see Figure 5). Let $p(e_1) = p(e_2) = p(e_3) = 1$, i.e., propagation probability is 1; $L = 2$, i.e., influence stops after two round; and $u(e_1) = 1 - \epsilon$ $u(e_2) = u(e_3) = \epsilon$ for some small enough ϵ to be set. That is the only uncertainty comes from incomplete knowledge of the existence of edges.

Let $\Psi_1 = \{e_1 \text{ exists}\}$ and $\Psi_2 = \{e_1, e_3 \text{ exists}\}$. Then $\mathbb{E}_\Phi [f(a, b, c)|\Phi \sim \Psi_2] = 4$ since all nodes will be influenced. $\mathbb{E}_\Phi [f(a, c)|\Phi \sim \Psi_2] = 4 - \epsilon$ since the only uncertain node is b which will be influenced with probability $1 - \epsilon$. Therefore,

$$\mathbb{E}_\Phi [f(a, b, c)|\Phi \sim \Psi_2] - \mathbb{E}_\Phi [f(a, c)|\Phi \sim \Psi_2] = \epsilon. \quad (1)$$

Now $\mathbb{E}_\Phi [f(a, b)|\Phi \sim \Psi_1] = 2 + \epsilon + \epsilon^2$ since a, b will be surely influenced, c and d will be influenced with probability ϵ and ϵ^2 respectively. On the other hand, $\mathbb{E}_\Phi [f(a)|\Phi \sim \Psi_1] = 2 + \epsilon$ since b will be surely influenced (since e_1 exists) and c will be influenced with probability ϵ. Since $L = 2$, d cannot be influenced. As a result,

$$\mathbb{E}_\Phi [f(a, b)|\Phi \sim \Psi_2] - \mathbb{E}_\Phi [f(a)|\Phi \sim \Psi_2] = \epsilon^2. \quad (2)$$

Combining Equation (1) and (2), we know that DIME is not adaptive submodular. $\square$

6. HEAL: DIME PROBLEM SOLVER

The above theorems show that DIME is a hard problem as it is difficult to even obtain any reasonable approximations. We model DIME as a POMDP [16] because of two reasons. First, POMDPs are a good fit for DIME as (i) we conduct several interventions sequentially, similar to sequential POMDP actions; and (ii) we have *partial observability* (similar to POMDPs) due to uncertainties in network structure and influence status of nodes. Second, POMDP solvers have recently shown great promise in generating near-optimal policies efficiently [23]. We now explain how we map DIME onto a POMDP.

States. A POMDP state in our problem is a pair of binary tuples $s = \langle W, F \rangle$ where W and F are of lengths $|V|$ and $|E_U|$, respectively. Intuitively, W denotes the influence status of network nodes, where $W_i = 1$ denotes that node i is influenced and $W_i = 0$ otherwise. Moreover, F denotes the existence of uncertain edges, where $F_i = 1$ denotes that the ith uncertain edge exists in reality, and $F_i = 0$ otherwise.

Actions. Every choice of a subset of K nodes is a POMDP action. More formally, $A = \{a \subset V s.t. |a| = K\}$. For example, in Figure 3, one possible action is $\{A, B\}$ (when $K = 2$).

Observations. Upon taking a POMDP action, we "*observe*" the ground reality of the uncertain edges outgoing from the nodes chosen in that action. Consider $\Theta(a) = \{e \mid e = (x,y) \text{ s.t. } x \in a \wedge e \in E_u\} \forall a \in A$, which represents the (ordered) set of uncertain edges that are observed when we take action a. Then, our POMDP observation upon taking action a is defined as $o(a) = \{F_e | e \in \Theta(a)\}$, i.e., the F-values of the observed uncertain edges. For example, by taking action $\{B, C\}$ in Figure 3, the values of F_4 and F_5 (i.e., the F-values of uncertain edges in the 1-hop social circle of nodes B and C) would be observed.

Rewards. The reward $R(s, a, s')$ of taking action a in state s and reaching state s' is the number of newly influenced nodes in s'. More formally, $R(s, a, s') = (\|s'\| - \|s\|)$, where $\|s'\|$ is the number of influenced nodes in s'.

Initial Belief State. The initial belief state is a distribution β_0 over all states $s \in S$. The support of β_0 consists of all states $s = \langle W, F \rangle$ s.t. $W_i = 0 \forall i \in [1, |V|]$, i.e., all states in which all network nodes are un-influenced (as we assume that all nodes are un-influenced to begin with). Inside its support, each F_i is distributed independently according to $P(F_i = 1) = u(e)$.

Transition And Observation Probabilities. Computation of exact transition probabilities $T(s'|s, a)$ requires considering all possible paths in a graph through which influence could spread, which is $\mathcal{O}(N!)$ (N is number of nodes in the network) in the worst case. Moreover, for large social networks, the size of the transition and observation probability matrix is prohibitively large (due to exponential sizes of state and action space).

Therefore, instead of storing huge transition/observation matrices in memory, we follow the paradigm of large-scale online POMDP solvers [23, 7, 6] by using a generative model $\Lambda(s, a) \sim (s', o, r)$ of the transition and observation probabilities. This generative model allows us to generate on-the-fly samples from the exact distributions $T(s'|s, a)$ and $\Omega(o|a, s')$ at very low computational costs. Given an initial state s and an action a to be taken, our generative model Λ simulates the random process of influence spread to generate a random new state s', an observation o and the obtained reward r. Simulation of the random process of influence spread is done by "*playing*" out propagation probabilities (i.e., flipping weighted coins with probability $p(e)$) according to our influence model to generate sample s'. The observation sample o is then determined from s' and a. Finally, the reward sample $r = (\|s'\| - \|s\|)$ (as defined above). This simple design of the generative model allows significant scale and speed up (as seen in previous work [23] and also in our experimental results section).

We solve this POMDP using a novel algorithm (described in Section 6.1) to find the optimal policy Π_T^* for the DIME problem.

6.1 HEALER's DIME Solver

Initial experiments with the POMCP solver [23] showed that it ran out of memory on 30 node graphs. Similarly, PSINET-W [29] was simply unable to scale up to real world demands (as shown in our experiments). Hence, we propose HEAL, a new heuristic based online POMDP planner (for solving the DIME problem) which scales up to our collaborating shelter's real world demands.

6.1.1 HEAL

HEAL solves the *original POMDP* using a novel *hierarchical ensembling heuristic*: it creates ensembles of imperfect (and smaller) POMDPs at *two* different layers, in a hierarchical manner (see Figure 6). HEAL's *top layer* creates an ensemble of smaller sized *intermediate POMDPs* by subdividing the original *uncertain network* into several smaller sized *partitioned networks* by using graph partitioning techniques [13]. Each of these partitioned
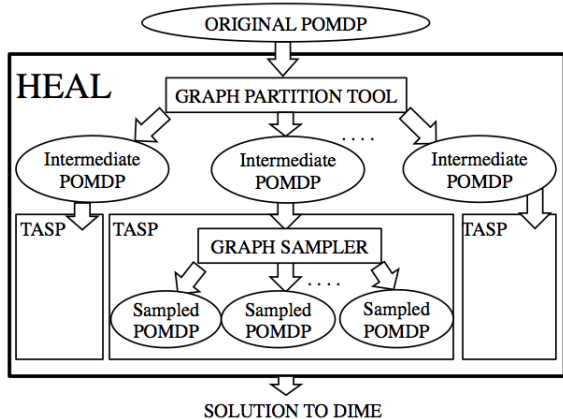
Figure 6: Hierarchical decomposition in HEAL

networks is then mapped onto a POMDP, and these *intermediate POMDPs* form our *top layer* ensemble of POMDP solvers.

In the bottom layer, each *intermediate POMDP* is solved using TASP (**T**ree **A**ggregation for **S**equential **P**lanning), our novel POMDP planner, which subdivides the POMDP into another ensemble of smaller sized *sampled POMDPs*. Each member of this *bottom layer* ensemble is created by randomly sampling uncertain edges of the partitioned network to get a sampled network having no uncertain edges, and this sampled network is then mapped onto a *sampled POMDP*. Finally, the solutions of POMDPs in both the *bottom* and *top layer* ensembles are aggregated using novel techniques to get the solution for HEAL's original POMDP.

HEAL uses several novel heuristics. First, it uses a novel two-layered *hierarchical ensembling heuristic*. Second, it uses graph partitioning techniques to partition the uncertain network, which generates partitions that minimize the edges going across partitions (while ensuring that partitions have similar sizes). Since these partitions are "almost" disconnected, we solve each partition separately. Third, it solves the *intermediate POMDP* for each partition by creating smaller-sized *sampled POMDPs* (via sampling uncertain edges), each of which is solved using a novel tree search algorithm, which avoids the exponential branching factor seen in PSINET [29]. Fourth, it uses novel aggregation techniques to combine solutions to these smaller POMDPs rather than simple plurality voting techniques seen in previous ensemble techniques [29].

These heuristics enable scale up to real-world sizes (at the expense of sacrificing performance guarantees), as instead of solving one huge problem, we now solve several smaller problems. However, these heuristics perform very well in practice. Our simulations show that even on smaller settings, HEAL achieves a 100X speed up over PSINET, while providing a 70% improvement in solution quality; and on larger problems, *where PSINET is unable to run at all*, HEAL continues to provide high solution quality. Now, we elaborate on these heuristics by first explaining the TASP solver.

6.1.2 Bottom layer: TASP

We now explain TASP, our new POMDP solver that solves each *intermediate POMDP* in HEAL's bottom layer. Given an *intermediate POMDP* and the uncertain network it is defined on, as input, TASP goes through four steps (see Algorithm 1).

First, Step 1 makes our *intermediate POMDP* more tractable by creating an ensemble of smaller sized *sampled POMDPs*. Each member of this ensemble is created by sampling uncertain edges of the input network to get an *instantiated* network. Each uncertain edge in the input network is randomly kept with probability $u(e)$,

---

Algorithm 1: TASP Solver

Input: Uncertain network G, Parameters K, T, L
Output: Best K node action κ

1   Create ensemble of Δ different POMDPs;
2   **for** $\delta \in \Delta$ **do**
3     $\alpha^\delta = Evaluate(\delta)$;
4   $r = Expectation(\alpha)$;
5   $\kappa = \mathrm{argmax}_j \, r_j$;
6   return κ;

Algorithm 2: Evaluate Step

Input: Instantiated network δ, Number of simulations **NSim**
Output: Ranked Ordering of actions α^δ

1   $tree = Initialize\_K\_Level\_Tree()$;
2   $counter = 0$;
3   **while** $counter + + < $ **NSim do**
4     $K\_Node\_Act = FindStep(tree)$;
5     $LT\_Reward = SimulateStep(K\_Node\_Act)$;
6     $UpdateStep(tree, LT\_Reward, K\_Node\_Act)$;
7   $\alpha^\delta = Get\_All\_Leaf\_Values(tree)$;
8   return α^δ;

---

or removed with probability $1 - u(e)$, to get an *instantiated* network with no uncertain edges. We repeat this sampling process to get Δ (a variable parameter) different *instantiated* networks. These Δ different *instantiated* networks are then mapped onto to Δ different POMDPs, which form our ensemble of *sampled POMDPs*. Each *sampled POMDP* shares the same action space (defined on the input partitioned network) as the different POMDPs only differ in the sampling of uncertain edges. Note that each member of our ensemble is a POMDP as even though sampling uncertain edges removes uncertainty in the F portion of POMDP states, there is still partial observability in the W portion of POMDP state.

In Step 3 (called the Evaluate Step), for each instantiated network $\delta \in [1, \Delta]$, we generate an α^δ list of rewards. The ith element of α^δ gives the long term reward achieved by taking the ith action in *instantiated* network δ. In Step 4, we find the expected reward r_i of taking the ith action, by taking a reward expectation across the α^δ lists (for each $\delta \in [1, \Delta]$) generated in the previous step. For e.g., if $\alpha_1^{\delta_1} = 10$ and $\alpha_1^{\delta_2} = 20$, i.e., the rewards of taking the 1st action in instantiated networks δ_1 and δ_2 (which occurs with probabilities $P(\delta_1)$ and $P(\delta_2)$) are 10 and 20 respectively, then the expected reward $r_1 = P(\delta_1) \times 10 + P(\delta_2) \times 20$. Note that $P(\delta_1)$ and $P(\delta_2)$ are found by multiplying existence probabilities $u(e)$ (or $1 - u(e)$) for uncertain edges that were kept (or removed) in δ_1 and δ_2. Finally, in Step 5, the action $\kappa = \mathrm{argmax}_j \, r_j$ is returned by TASP. Next, we discuss the Evaluate Step (Step 3).

Evaluate Step Algorithm 2 generates the α^δ list for a single instantiated network $\delta \in [1, \Delta]$. This algorithm works similarly for all instantiated networks. For each instantiated network, the Evaluate Step uses **NSim** (we use 2^{10}) number of MC simulations to evaluate the long term reward achieved by taking actions in that network. Due to the combinatorial action space, the Evaluate Step uses a UCT [12] driven approach to strategically choose the actions whose long term rewards should be calculated. UCT has been used to solve POMDPs in [23, 29], but these algorithms suffer from a $\binom{N}{K}$ branching factor (where K is number of nodes picked per round, N is number of network nodes). We exploit the structure of our domain by creating a K-level UCT tree which has a branching factor of just N (explained below). This K-level tree allows storing reward values for smaller sized node subsets as well (instead of just

---

Algorithm 3: FindStep

Input: K level deep tree - $tree$
Output: Action set of size K nodes - Act
1   $Act = \Phi$;
2   $tree\_node = tree.Root$;
3   **while** $is\_Leaf(tree\_node) == false$ **do**
4      $MAB_{node} = Get\_UCB\_at\_Node(node)$;
5      $next\_node = Ask\_UCB(MAB_{node})$;
6      $Act = Act \cup next\_node$;
7      $tree\_node = tree\_node.branch(next\_node)$;
8   **return** Act;

K sized subsets), which helps in guiding the UCT search better.

Algorithm 2 takes an *instantiated* network and creates the aforementioned K-level tree for that network. The first level of the tree has N branches (one for each network node). For each branch i in the first level, there are $N - 1$ branches in the second tree level (one for each network node, except for node i, which was covered in the first level). Similarly, for every branch j in the mth level ($m \in [2, K - 1]$), there are $N - m$ branches in the $(m + 1)^{th}$ level. Theoretically, this tree grows exponentially with K, however, the values of K are usually small in practice (e.g., 4).

In this K level tree, each leaf node represents a particular POMDP action of K network nodes. Similarly, every non-leaf tree node v represents a subset S_v of network nodes. Each tree node v maintains a value R_v, which represents the average long term reward achieved by taking our POMDP's actions (of size K) which contain S_v as a subset. For example, in Figure 3, if $K = 5$, and for tree node v, $S_v = \{A, B, C, D\}$, then R_v represents the average long term reward achieved by taking POMDP actions $A_1 = \{A, B, C, D, E\}$ and $A_2 = \{A, B, C, D, F\}$, since both A_1 and A_2 contain $S_v = \{A, B, C, D\}$ as a subset. To begin with, all nodes v in the tree are initialized with $R_v = 0$ (Step 1). By running **NSim** number of MC simulations, we generate good estimates of R_v values for each tree node v.

Each node in this K-level tree runs a UCB1 [12] implementation of a multi-armed bandit. The arms of the multi-armed bandit running at tree node v correspond to the child branches of node v in the K-level tree. Recall that each child branch corresponds to a network node. The overall goal of all the multi-armed bandits running in the tree is to construct a POMDP action of size K (by traversing a path from the root to a leaf), whose reward is then calculated in that MC simulation (explained in Algorithm 3). Every MC simulation consists of three steps: Find Step (Step 4), Simulate Step (Step 5) and Update Step (Step 6).

Find Step: The Find Step takes a K-level tree for an instantiated network and *finds* a K node action, which is used in the Simulate Step. Algorithm 3 details the process of *finding* this K node action, which is found by traversing a path from the root node to a leaf node, one edge/arm at a time. Initially, we begin at the root node with an empty action set of size 0 (Steps 1 and 2). For each node that we visit on our way from the root to a leaf, we use its multi-armed bandit (denoted by MAB_{node} in Step 4) to choose which tree node do we visit next (or, which network node do we add to our action set). We get a K node action upon reaching a leaf.

Simulate Step: The Simulate Step takes a K node action from the Find Step, to *evaluate* the long term reward of taking that action (called Act) in the instantiated network. Assuming that T_0 interventions remain (i.e., we have already conducted $T - T_0$ interventions), the Simulate Step first uses action Act in the generative model Λ to generate a reward r_0. For all remaining $(T_0 - 1)$ interventions, Simulate Step uses a rollout policy to randomly select K node actions, which are then used in the generative model Λ to

generate future rewards $r_i \forall i \in [1, T_0 - 1]$. Finally, the long term reward returned by Simulate Step is $r_0 + r_1 + ... + r_{T_0 - 1}$.

Update Step: The Update Step uses the long term reward returned by Simulate Step to update relevant R_v values in the K-level tree. It updates the R_v values of all nodes v that were traversed in order to find the K node action in the Find Step. First, we get the tree's leaf node corresponding to the K node action that was returned by the Find Step. Then, we go and update R_v values for all ancestors (including the root) of that leaf node.

After running the Find, Simulate and Evaluate for **NSim** simulations, we return the R_v values of all leaf nodes as the α^δ list. Recall that we then find the expected reward r_i of taking the ith action, by taking an expectation of rewards across the α^δ lists. Finally, TASP returns the action $\kappa = \text{argmax}_j r_j$.

6.1.3 Top layer: Using Graph Partitioning

We now explain HEAL's top layer, in which we use METIS [13], a state-of-the-art graph partitioning technique, to subdivide our original uncertain network into different partitioned networks. These partitioned networks form the ensemble of *intermediate POMDPs* (in Figure 6) in HEAL. Then, TASP is invoked on each intermediate POMDP independently, and their solutions are aggregated to get the final DIME solution. We try two different partitioning/aggregation techniques, which leads to two variants of HEAL:

K Partition Variant (HEAL): Given the *uncertain* network G and the parameters K, L and T as input, we first partition the uncertain network into K partitions. In each round from 1 to T, we invoke the bottom layer TASP algorithm to select 1 node from each of the K clusters. These singly selected nodes from the K clusters give us an action of K nodes, which is given to shelter officials to execute. Based on the *observation* (about uncertain edges) that officials get while executing the action, we update the partition networks (which are input to the *intermediate POMDPs*) by either replacing the *observed* uncertain edges with certain edges (if the edge was *observed* to exist in reality) or removing the uncertain edge altogether (if the edge was *observed* to *not exist* in reality). The list of K node actions that Algorithm 4 generates serves as an online policy for use by the homeless shelter.

T Partition Variant (HEAL-T): Given the *uncertain* network G and the parameters K, L and T as input, we first partition the uncertain network into T partitions and TASP picks K nodes from the ith partition ($i \in [1, T]$) in the ith round.

7. EXPERIMENTAL RESULTS

In this section, we analyze HEAL and HEAL-T's performance in a variety of settings. All our experiments are run on a 2.33 GHz 12-core Intel machine having 48 GB of RAM. All experiments are averaged over 100 runs. We use a metric of "*Indirect Influence*" throughout this section, which is number of nodes "*indirectly*" influenced by intervention participants. For example, on a 30 node network, by selecting 2 nodes each for 10 interventions (horizon), 20 nodes (a lower bound for any strategy) are influenced with certainty. However, the total number of influenced nodes might be 26 (say) and thus, the *Indirect Influence* is $26 - 20 = 6$. In all experiments, the propagation and existence probability values on all network edges were uniformly set to 0.1 and 0.6, respectively. This was done based on findings in Kelly et. al.[9]. However, we relax these parameter settings later in the section. All experiments are statistically significant under bootstrap-t ($\alpha = 0.05$).

Baselines: We use two algorithms as baselines. We use PSINET-W as a benchmark as it is the most relevant previous algorithm, which was shown to outperform heuristics used in practice; however, we also need a point of comparison when PSINET-
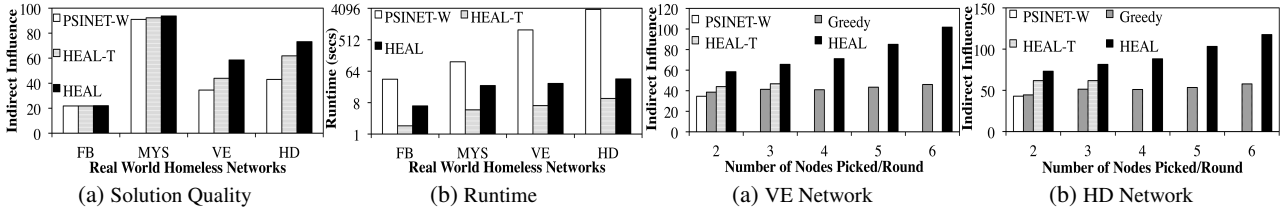
(a) Solution Quality   (b) Runtime

Figure 7: Solution Quality and Runtime on Real World Networks

(a) VE Network   (b) HD Network

Figure 8: Scale up in number of nodes picked per round

W does not scale. No previous algorithm in the influence maximization literature accounts for uncertain edges and uncertain network state in solving the problem of sequential selection of nodes; in-fact we show that even the standard Greedy algorithm [10, 8] has no approximation guarantees as our problem is not adaptive submodular. Thus, we modify Greedy by replacing our uncertain network with a certain network (in which each uncertain edge e is replaced with a certain edge e_0 having propagation probability $p(e_0) = p(e) \times u(e)$), and then run the Greedy algorithm on this *certain network*. We use the Greedy algorithm as a baseline as it is the best known algorithm known for influence maximization and has been analyzed in many previous papers [3, 1, 25, 10, 14, 8].

Datasets: We use *four real world social networks* of homeless youth, provided to us by our collaborators. All four networks are friendship based social networks of homeless youth living in different areas of a big city in USA (name withheld for anonymity). The first and second networks are of homeless youth living in two large areas (denoted by VE and HD to preserve anonymity), respectively. These two networks (each having ~150-170 nodes, 400-450 edges) were created through surveys and interviews of homeless youth (conducted by our collaborators) living in these areas. The third and fourth networks are relatively small-sized online social networks of these youth created from their Facebook (34 nodes, 120 edges) and MySpace (107 nodes, 803 edges) contact lists, respectively. When HEALER is deployed, we anticipate even larger networks, (e.g., 250-300 nodes) than the ones we have in hand and we also show run-time results on artificial networks of these sizes.

Solution Quality/Runtime Comparison. We compare *Indirect Influence* and run-times of HEAL, HEAL-T and PSINET-W on all four real-world networks. We set $T = 5$ and $K = 2$ (since PSINET-W fails to scale up beyond $K = 2$ as shown later). Figure 7a shows the *Indirect Influence* of the different algorithms on the four networks. The X-axis shows the four networks and the Y-axis shows the *Indirect Influence* achieved by the different algorithms. This figure shows that (i) HEAL outperforms all other algorithms on every network; (ii) *it achieves ~70% improvement over PSINET-W in VE and HD networks; (iii) it achieves ~25% improvement over HEAL-T*. The difference between HEAL and other algorithms is not significant in the Facebook (FB) and MySpace (MYS) networks, as HEAL is already influencing almost all nodes in these two relatively small networks. Thus, in experiments to come, we focus more on the VE and HD networks.

Figure 7b shows the run-time of all algorithms on the four networks. The X-axis shows the four networks and the Y-axis (in log scale) shows the run-time (in seconds). This figure shows that (i) *HEAL achieves a 100X speed-up over PSINET-W*; (ii) PSINET-W's run-time increases exponentially with increasing network sizes; (iii) HEAL runs 3X slower than HEAL-T but achieves 25% more *Indirect Influence*. Hence, HEAL is our algorithm of choice.

Next, we check if PSINET-W's run-times become worse on larger networks. Because of lack of larger real-world datasets, we create relatively large artificial Watts-Strogatz networks (model pa-

rameters $p = 0.1, k = 7$). Figure 11a shows the run-time of all algorithms on Watts-Strogatz networks. The X-axis shows the size of networks and the Y-axis (in log scale) shows the run-time (in seconds). This figure shows that *PSINET-W fails to scale beyond 180 nodes, whereas HEAL runs within 5 minutes*. Thus, PSINET-W fails to scale-up to network sizes that are of importance to us.

Scale Up Results. Not only does PSINET-W fail in scaling up to larger network sizes, it even fails to scale-up with increasing number of nodes picked per round (or K), on our real-world networks. Figures 8a and 8b show the *Indirect Influence* achieved by HEAL, HEAL-T, Greedy and PSINET-W on the VE and HD networks respectively ($T = 5$), as we scale up K values. The X-axis shows increasing K values, and the Y-axis shows the *Indirect Influence*. These figures show that (i) PSINET-W and HEAL-T fail to scale up – they cannot handle more than $K = 2$ and $K = 3$ respectively (thereby not fulfilling real world demands); (ii) HEAL outperforms all other algorithms, and the difference between HEAL and Greedy increases linearly with increasing K values. Also, *in the case of $K = 6$, HEAL runs in less than 40.12 seconds on the HD network and 34.4 seconds on the VE network*.

Thus, Figures 7a, 7b, 8a and 8b show that PSINET-W (the best performing algorithm from previous work) fails to scale up with increasing network nodes, and with increasing K values. Even for $K = 2$ and moderate sized networks, it runs very slowly. Moreover, HEAL is the best performing algorithm that runs quickly, provides high-quality solutions, and can scale-up to real-world demands. Since only HEAL and Greedy scale up to $K = 6$, we now analyze their performance in detail.

Scaling up Horizon. Figure 9a shows HEAL and Greedy's *Indirect Influence* in the HD network, with varying horizons (see appendix for VE network results). The X-axis shows increasing horizon values and the Y-axis shows the *Indirect Influence* ($K = 2$). This figure shows that the relative difference between HEAL and Greedy increases significantly with increasing T values.

Next, we scale up K values with increased horizon settings to find the maximum attainable solution quality difference between HEAL and Greedy. Figure 9b shows the *Indirect Influence* achieved by HEAL and Greedy (with $K = 4$ and $T = 10$) on the VE and HD networks. The X-axis shows the two networks and the Y-axis shows the *Indirect Influence*. This figure shows that with these settings, *HEAL achieves ~110% more Indirect Influence than Greedy (i.e.,*
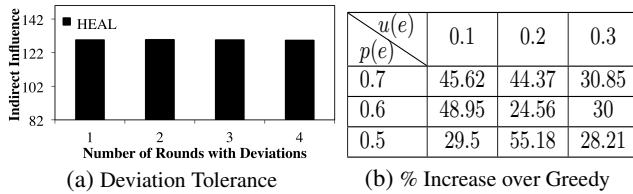

(a) Solution Quality   (b) Maximum Relative Gain

Figure 9: Horizon Scale up & Maximum Gain on HD Network

(a) Deviation Tolerance

$u(e)$ $p(e)$	0.1	0.2	0.3
0.7	45.62	44.37	30.85
0.6	48.95	24.56	30
0.5	29.5	55.18	28.21

(b) % Increase over Greedy

Figure 10: Deviation Tolerance & HEAL vs Greedy

(a) Artificial Networks

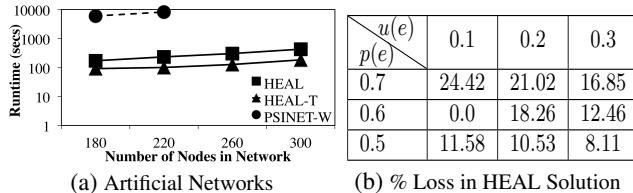| $u(e)$ $p(e)$	0.1	0.2	0.3
0.7	24.42	21.02	16.85
0.6	0.0	18.26	12.46
0.5	11.58	10.53	8.11

(b) % Loss in HEAL Solution

Figure 11: Artificial Networks And Sensitivity Analysis

more than a 2-fold improvement) in the two real-world networks.

HEAL vs Greedy. Figure 10b shows the percentage increase (in *Indirect Influence*) achieved by HEAL over Greedy with varying $u(e)/p(e)$ values. The columns and rows of Figure 10b show varying $u(e)$ and $p(e)$ values respectively. The values inside the table cells show the percentage increase (in *Indirect Influence*) achieved by HEAL over Greedy when both algorithms plan using the same $u(e)/p(e)$ values. For example, with $p(e) = 0.7$ and $u(e) = 0.1$, HEAL achieves 45.62% more *Indirect Influence* than Greedy. This figure shows that *HEAL continues to outperform Greedy across varying $u(e)/p(e)$ values*. Thus, on a variety of settings, HEAL dominates Greedy in terms of both *Indirect Influence* and run-time.

Deviation Tolerance. We show HEAL's tolerance to deviation by replacing a fixed number of actions recommended by HEAL with randomly selected actions. Figure 10a shows the variation in *Indirect Influence* achieved by HEAL ($K = 4, T = 10$) with increasing number of random deviations from the recommended actions. The X-axis shows increasing number of deviations and the Y-axis shows the *Indirect Influence*. For example, when there were 2 random deviations (i.e., two recommended actions were replaced with random actions), HEAL achieves 100.23 *Indirect Influence*. This figure shows that HEAL is highly deviation-tolerant.

Sensitivity Analysis. Finally, we test the robustness of HEAL's solutions in the HD network (see appendix for VE network results), by allowing for error in HEAL's understanding of $u(e)/p(e)$ values. We consider the case that $u(e) = 0.1$ and $p(e) = 0.6$ values that HEAL plans on, are wrong. Thus, HEAL plans its solutions using $u(e) = 0.1$ and $p(e) = 0.6$, but those solutions are evaluated on different (correct) $u(e)/p(e)$ values to get *estimated solutions*. These *estimated solutions* are compared to *true solutions* achieved by HEAL if it planned on the correct $u(e)/p(e)$ values. Figure 11b shows the percentage difference (in *Indirect Influence*) between the *true* and *estimated* solutions, with varying $u(e)$ and $p(e)$ values. For example, when HEAL plans its solutions with wrong $u(e) = 0.1/p(e) = 0.6$ values (instead of correct $u(e) = 0.3/p(e) = 0.5$ values), it suffers a 8.11% loss. This figure shows that HEAL is relatively robust to errors in its understanding of $u(e)/p(e)$ values, as it only suffers an average-case loss of $\sim 15\%$.

8. CONCLUSION

This paper focuses on the important problem of selecting par-

ticipants of sequentially deployed interventions, which are organized by homeless shelters to spread awareness about HIV prevention practices among homeless youth. This is an extremely important problem as homeless youth are at high-risk to HIV ($\sim$10% of homeless youth are HIV positive). While previous work tries to solve this problem, it *simply fails to scale up to real world sizes and demands*. It runs out of memory on large networks, with increased number of intervention participants, and runs very slowly on moderate sized networks. In this paper, we develop HEALER, a new software agent for solving this problem which scales up to real world demands. HEALER casts the problem as a POMDP and uses a completely novel suite of algorithms (HEAL, TASP and Evaluate) to achieve a 100X speedup over state-of-the-art algorithms while outperforming them by 70% in terms of solution quality. More than that, it runs when previous algorithms can't scale up. Also, HEALER saves homeless shelters' thousands of dollars and many months of time by generating uncertain networks at low cost using its Facebook application. Finally, we show some novel theoretical hardness results about the problem that HEALER solves. HEALER is fully ready to be deployed in the real world, in collaboration with a homeless shelter. The shelter officials have tested HEALER's components and their feedback has been positive. HEALER's deployment is expected to commence in early Spring 2016. This is an extended version of our AAMAS 2016 paper by the same name. For the conference version, please refer to [28].

REFERENCES

[1] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier. Maximizing Social Influence in Nearly Optimal Time. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 946–957. SIAM, 2014.

[2] CDC. HIV Surveillance Report. www.cdc.gov/hiv/pdf/g-l/hiv_surveillance_report_vol_25.pdf, Mar. 2013.

[3] E. Cohen, D. Delling, T. Pajor, and R. F. Werneck. Sketch-based Influence Maximization and Computation: Scaling up with guarantees. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 629–638. ACM, 2014.

[4] J.-P. Cointet and C. Roth. How Realistic Should Knowledge Diffusion Models Be? *Journal of Artificial Societies and Social Simulation*, 10(3):5, 2007.

[5] N. H. Council. HIV/AIDS among Persons Experiencing Homelessness: Risk Factors, Predictors of Testing, and Promising Testing Strategies. www.nhchc.org/wp-content/uploads/2011/09/InFocus_Dec2012.pdf, Dec. 2012.

[6] J. S. Dibangoye, G. Shani, B. Chaib-Draa, and A.-I. Mouaddib. Topological Order Planner for POMDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2009.

[7] A. Eck and L.-K. Soh. To Ask, Sense, or Share: Ad Hoc Information Gathering. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 367–376. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[8] D. Golovin and A. Krause. Adaptive Submodularity: Theory and Applications in Active Learning and Stochastic Optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

[9] J. A. Kelly, D. A. Murphy, K. J. Sikkema, T. L. McAuliffe,

R. A. Roffman, L. J. Solomon, R. A. Winett, and S. C. Kalichman. Randomised, Controlled, Community-Level HIV-Prevention Intervention for Sexual-Risk Behaviour among Homosexual men in US cities. *The Lancet*, 350(9090):1500, 1997.

[10] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the Spread of Influence through a Social Network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[11] M. Kim and J. Leskovec. The Network Completion Problem: Inferring Missing Nodes and Edges in Networks. In *Proceedings of the SIAM Conference on Data Mining*. SIAM, 2011.

[12] L. Kocsis and C. Szepesvári. Bandit based Monte-Carlo Planning. In *Machine Learning: ECML 2006*, pages 282–293. Springer, 2006.

[13] D. LaSalle and G. Karypis. Multi-threaded Graph Partitioning. In *Parallel & Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on*, pages 225–236. IEEE, 2013.

[14] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective Outbreak Detection in Networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 420–429. ACM, 2007.

[15] L. Marcolino, A. Lakshminarayanan, A. Yadav, and M. Tambe. Simultaneous Influencing and Mapping Social Networks. In *Proceedings of the Fiftienth International Conference on Autonomous Agents and Multiagent Systems (Short Paper) (AAMAS 2016)*, 2016.

[16] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2009.

[17] E. Rice. The Positive Role of Social Networks and Social Networking Technology in the Condom-using Behaviors of Homeless Young People. *Public health reports*, 125(4):588, 2010.

[18] E. Rice, A. Barman-Adhikari, N. G. Milburn, and W. Monro. Position-specific HIV risk in a Large Network of Homeless Youths. *American journal of public health*, 102(1):141–147, 2012.

[19] E. Rice, A. Fulginiti, H. Winetrobe, J. Montoya, A. Plant, and T. Kordic. Sexuality and Homelessness in Los Angeles public schools. *American Journal of Public Health*, 102, 2012.

[20] E. Rice, E. Tulbert, J. Cederbaum, A. B. Adhikari, and N. G. Milburn. Mobilizing Homeless Youth for HIV Prevention: a Social Network Analysis of the Acceptability of a face-to-face and Online Social Networking Intervention. *Health education research*, 27(2):226, 2012.

[21] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa. Online Planning Algorithms for POMDPs. *Journal of Artificial Intelligence Research*, pages 663–704, 2008.

[22] J. A. Schneider, A. N. Zhou, and E. O. Laumann. A new HIV Prevention Network Approach: Sociometric Peer Change Agent Selection. *Social Science & Medicine*, 125:192–202, 2015.

[23] D. Silver and J. Veness. Monte-Carlo Planning in large POMDPs. In *Advances in Neural Information Processing Systems*, pages 2164–2172, 2010.

[24] A. Somani, N. Ye, D. Hsu, and W. S. Lee. DESPOT: Online POMDP Planning with Regularization. In *Advances In Neural Information Processing Systems*, pages 1772–1780, 2013.

[25] Y. Tang, X. Xiao, and Y. Shi. Influence maximization: Near-Optimal Time Complexity meets Practical Efficiency. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 75–86. ACM, 2014.

[26] UNAIDS. Report on the Global AIDS Epidemic. `www.unaids.org/en/resources/campaigns/20121120_globalreport2012/globalreport/`, Mar. 2012.

[27] T. W. Valente and P. Pumpuang. Identifying Opinion Leaders to Promote Behavior Change. *Health Education & Behavior*, 2007.

[28] A. Yadav, H. Chan, A. Jiang, H. Xu, E. Rice, and M. Tambe. Using Social Networks to Aid Homeless Shelters: Dynamic Influence Maximization under Uncertainty - An Extended Version. In *Proceedings of the Fiftienth International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016)*, 2016.

[29] A. Yadav, L. Marcolino, E. Rice, R. Petering, H. Winetrobe, H. Rhoades, M. Tambe, and H. Carmichael. Preventing HIV Spread in Homeless Populations Using PSINET. In *Proceedings of the Twenty-Seventh Conference on Innovative Applications of Artificial Intelligence (IAAI-15)*, 2015.

[30] Q. Yan, S. Guo, and D. Yang. Influence Maximizing and Local Influenced Community Detection based on Multiple Spread Model. In *Advanced Data Mining and Applications*, pages 82–95. Springer, 2011.

[31] S. D. Young and E. Rice. Online Social Networking Technologies, HIV knowledge, and Sexual Risk and Testing Behaviors among Homeless Youth. *AIDS and Behavior*, 15(2):253–260, 2011.