CSE 8803 EPI: Data Science for Epidemiology, Fall 2024

Lecturer: B. Aditya Prakash
Scribe: Vanshika Shah

September 19, 2024
Lecture 10 : Outbreak Detection - 1

1 Introduction

Till now we have been discussing on the various models on the disease to find out if there will be an outbreak or not. But consider there is already an outbreak, one important aspect would be to detect it as soon as possible in order to conduct necessary interventions. A major problem in epidemiology is figuring out how to detect an outbreak of a contagion as early as possible. One answer to this is to track a subset of a population and ascertain if there is an outbreak. However, due to lack of resources, we cannot track a significant enough subset of the population for this method to work. So, we need to find the best candidates within the population, sensors, to track.

There are multiple methodologies for picking sensors. One such methodology is to track the friends of a random sample of the population. This can be more effective than just sampling randomly from a population. Another methodology for picking sensors is the idea of dominator trees, where nodes that are present along the shortest paths between other nodes are often good choices for sensors. Finally, we consider the problem of detecting outbreaks in a cascade, in which there is a submodular function that can provide a fast and effective approximation for the optimal set of nodes to select in graph G.

2 Idea of Social Network Sensors

2.1 Social Network Study

Social network sensors attempts to answer the question if we can effectively track a handful of individuals (so called "canaries in a coal mine") to forecast contagious outbreaks.

2.1.1 Harvard study on Early Detection of Contagious Outbreaks

This brings light to the study of the outbreak of influenza among Harvard students in 2009 [1]. In this study, a social network of 774 undergraduate students was constructed from 6650 undergraduates with two major data groups. One group was a random sample of 319 students and the other group was random sample of 425 of their friends. Both groups were tracked for the spread of influenza and the observations between the two were compared.

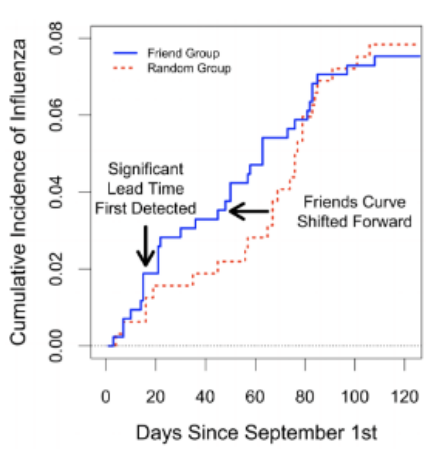


Fig. 1. Cumulative incidence of influenza after September 1st for friend group and random group. Depicts friend group having a significant lead time compared to the random group.

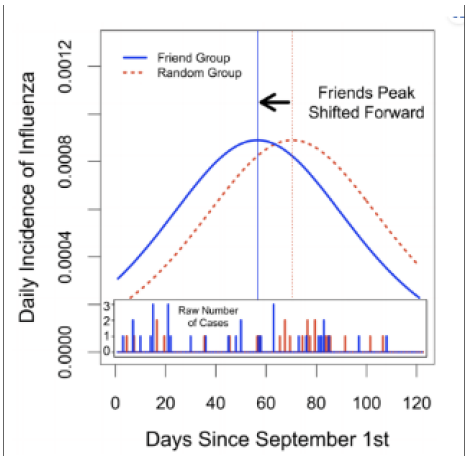


Fig. 2. Daily incidence of influenza after September 1st for friend group and random group. Depicts peak of influenza cases for friend group occurring earlier than random group.

From the figures above, there was an observed trend shift between the friends group and the random group. This trend shift meant that the friend group tended to get influenza before the individuals in the random group. As a result, a significant lead time was observed between the groups as shown in Fig. 1. Due to this significant lead time, the friends peak of daily incidence was shifted forward as shown in Fig. 2.

This observation can be scaled to identify an outbreak of contagion and applied generally, where we can track a subset of people which can give us a lead time advantage. In order to identify the right folks for the job, one should recall the friendship paradox's key statement regarding the average variance of friends of friends one has will always be greater than the average variance of friends one has. In this case study context, random samples will be less essential than the friends of random samples in outbreak detection.

Please refer to the Mathematical depiction of the friendship paradox (friends are likely better connected) below:

$$E[X] = \frac{\sum x_i}{N}$$

$$\mathrm{Var}[X] = E[(X - E[X])^2] = E[X^2] - E[X]^2$$

$$\frac{E[X^2]}{E[X]} = E[X] + \frac{\mathrm{Var}[X]}{E[X]}$$

Friendship paradox remains true even if we consider medians instead of averages. The median number of friends you have will be lesser than the median number of friends your friends have.

2.1.2 Majority Illusion

The idea of the "Friendship Paradox" can also create an issue of majority illusion as exhibited by Lerman et al 2015 [4]. This is widely observed in political science scenarios where

the senate might have an apparent majority in raw numbers. In class, we saw an example illustrating the majority illusion.

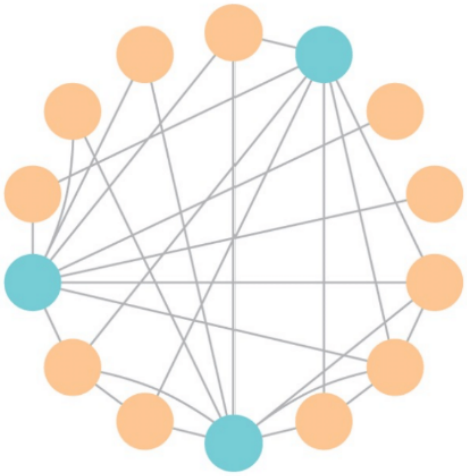


Fig. 3. Network depicting people in favor of baseball caps (blue) and people against baseball caps (orange).

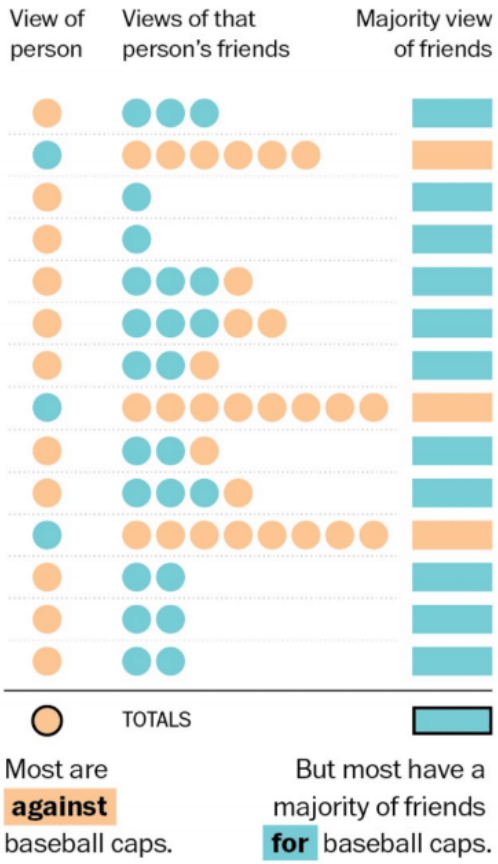


Fig. 4. Local view of the network for each person. Depicts distorted view of network when comparing to the global view.

In Fig. 3 above, we can see the global view of the network in which a majority of people are against baseball caps (oranges) while the people who are for baseball caps are in the minority (blues). Fig. 4 shows the local view of each person's friends in the network. When analyzing this, we see that people may have a different perception of who is in the majority based on their local view. This perception may be incorrect, as seen in Fig. 4, which is only able to be fixed by disseminating the global view of the network to everybody. A real-world example of this is the support for same-sex marriage. Once you get to know someone who advocates that opinion, your own viewpoint changes. In public health scenarios, the majority are susceptible population groups and the minority are the infected populations which can infect others.

2.1.3 Same sex marriage - Pew research center

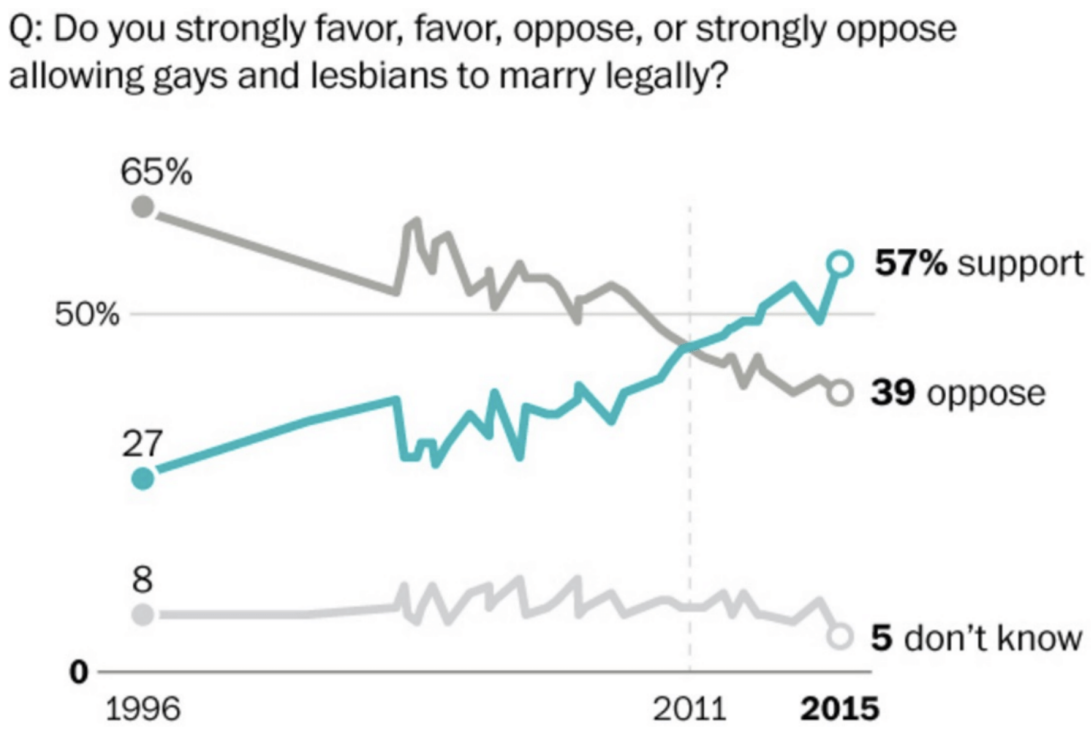


Fig. 5. Typical cumulative distribution functions of messages posted by a random control group and its corresponding sensor group.

The above figure shows the change in trends of public opinions on same sex marriage in the Unites States from 1996 to 2015. In 1996, only 27% supported legalizing same sex marriage. This support has increased with time with fluctuations in the early 2000s. The support eventually increased to 57% by 2015. As the support has increased over time, the opposition has decreased gradually over time, crossing below the support line around 2011. Here it can be seen that the support split as soon as people began to know other friends or other family members in their social circles going through the same situation. We can see that the influence of known entities significantly affected the trends.

2.1.4 Social network sensor in Twitter

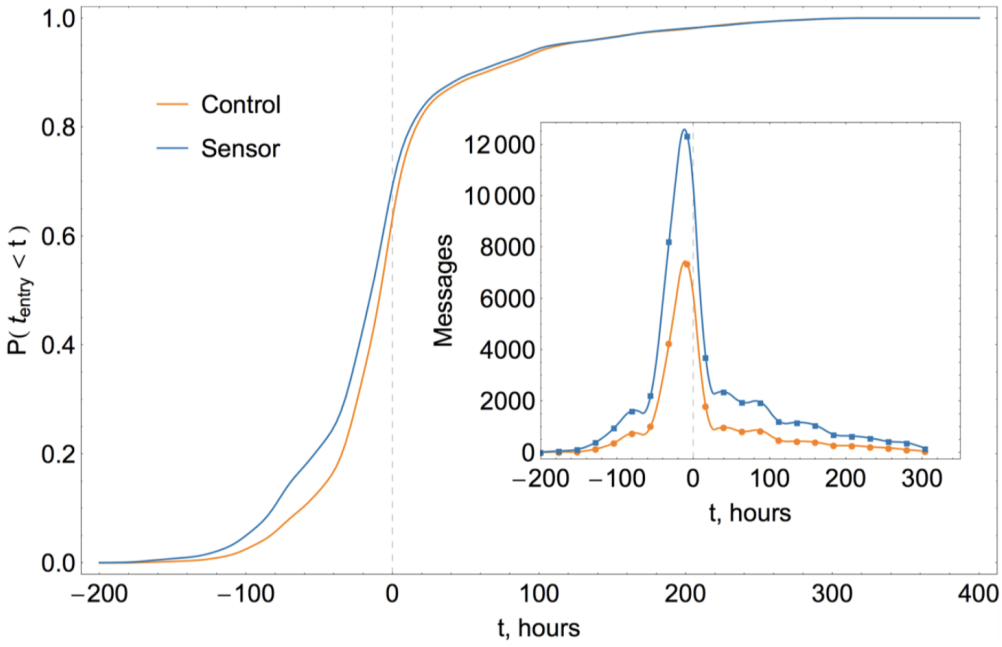


Fig. 6. Pew Research Center - trends on public opinion on legality of same-sex marriage

The study "Performance of Social Network Sensors during Hurricane Sandy" [3] investigates the performance of social network sensors, particularly Twitter users, during Hurricane Sandy. It examines how well-connected users (sensor groups) gain awareness of disasters earlier than randomly selected users (control groups). The study selected sensor groups based on social network characteristics, specifically focusing on Twitter users who are highly connected, influential, or active. These sensors include users with many followers, those who frequently tweet, and those involved in multiple interactions. The idea is that these users are more likely to be early detectors or spreaders of information during events like disasters, giving them an edge in awareness compared to randomly selected control groups. The above figure compares tweet activity between control and sensor groups. Sensor groups tweet earlier and more frequently, showing they become aware of events faster. A smaller graph inside shows daily tweet numbers, with both groups peaking on the hurricane's landfall day. The sensor group consistently tweets more, highlighting their quicker response.

2.1.5 Study on Forecasting the flu - obtaining a better lead time

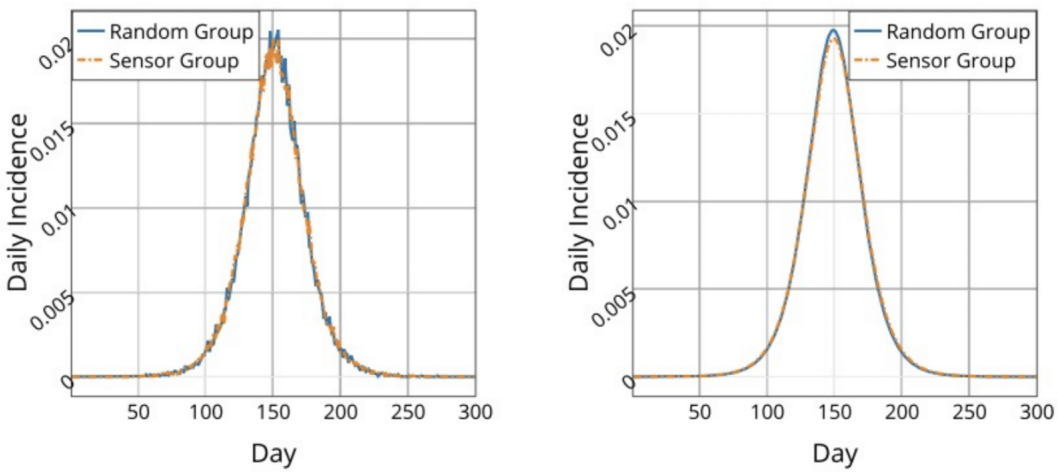


Fig. 7. Illustration of the Friend-of-Friend approach on the Miami dataset. True daily incidence curve (left), fitted daily incidence curve with logistic function (right)

The study "Forecasting the Flu: Designing Social Network Sensors for Epidemics" [6] focuses on selecting individuals within a social network (sensors) to detect flu outbreaks early. The approach leverages network properties to optimize the selection of these sensors, providing lead times that can guide public health responses. The study evaluates methods on large-scale, city-based networks to validate their effectiveness. The figure above illustrates the results of the Friend-of-Friend approach applied to the Miami dataset. It shows two panels: the true daily incidence curve of flu infections and a fitted curve using a logistic function. The Miami dataset reveals no significant lead time advantage using this method, suggesting that the approach's effectiveness varies with network structure. Hence, there are other better approaches for sensor selection as discussed below. By using graph-theoretic techniques like dominator trees and transmission trees, the study aims to improve lead times in predicting peaks of epidemic curves. Experiments using city-scale datasets show that these methods can significantly outperform traditional approaches discussed until now.

2.2 Formal Definition for Selecting Sensors

We can formally define the problem for sensor selection with two main methods: PLTM, where we maximize the lead time for the predicted peak, or MAIT, where we are trying to minimize the time to detection for infected nodes.

(ϵ, k)-Peak Lead Time Maximization (PLTM)

Given: Parameters ϵ and k, network G, and the epidemic model
Find: A set of notes S from G such that

$$S = \underset{S}{\operatorname{argmax}}\, E\left[t_{pk} - t_{pk}(S)\right]$$
$$\text{s.t. } f(S) \geq \epsilon,\ |S| = k$$

(ϵ, k)-Minimum Average Infection Time (MAIT)

Given: Parameters ϵ and k, network G, and the epidemic model
Find: A set of notes S such that

$$S = \underset{S}{\operatorname{argmin}} \sum_{v \in S} \frac{t_{\inf}(v)}{|S|}$$
$$\text{s.t. } f(S) \geq \epsilon,\ |S| = k$$

2.3 Dominator Trees

Another method for selecting effective sensor nodes is to use dominator trees. The idea is that nodes that are present on many of the shortest paths between other nodes are more likely to be infected when an epidemic spreads throughout the graph. Following this idea, we can generate dominator trees for dendrograms on a graph, and the top k nodes in that tree will become our sensor set. While it has limitations, this algorithm has the merit of being especially fast, running in linear time over a graph.

1. **generate dominator trees corresponding to each dendrogram;**
2. **compute the average depth of each node v in the dominator tree (as in the transmission tree heuristic);**
3. **discard nodes whose average depth is smaller than ϵ_0;**
4. **we order nodes based on their average depth to the dominator tree, and pick S to be the set of the first k nodes.**

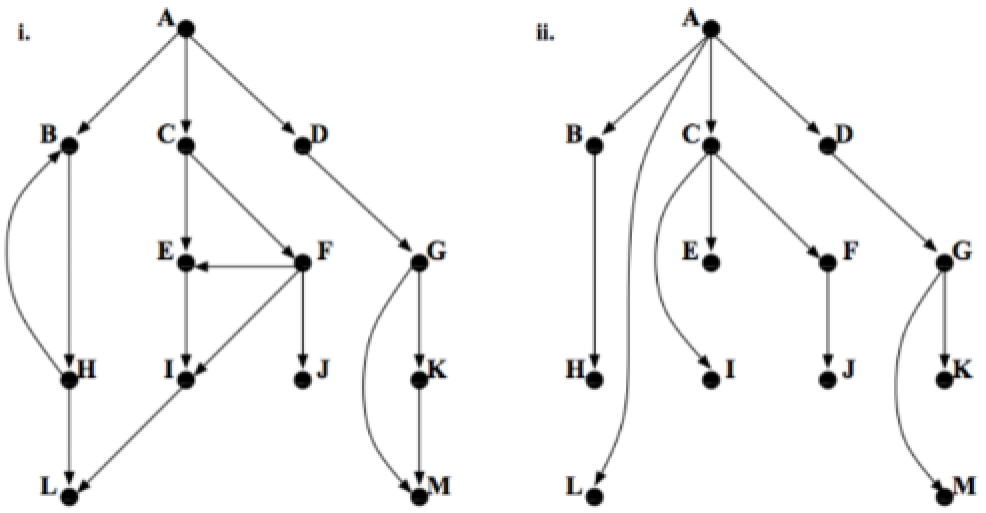


Fig. 8. A graph (i.) and the graph's dominator tree (ii.).

2.4 Surrogates - Redescriptions

We know there are methods for finding the best theoretical nodes to be sensors in a graph, but how can apply this knowledge to the real world? In other words, how can we use the information from these methods to determine who in the real world is an effective sensor?

One way is a decision tree that determines if a person is a good sensor candidate. We can use this to correlate which demographic features correlate to sensors found.

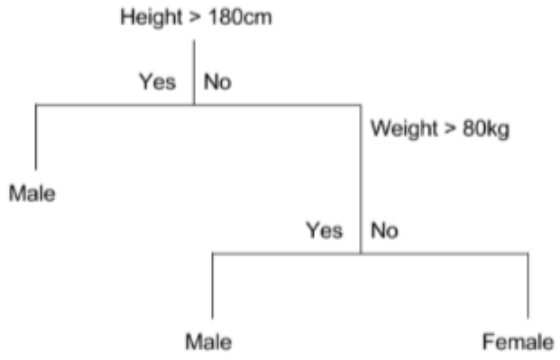


Fig. 9. An example decision tree with demographic features.

2.5 Performance of Dominators and Surrogates

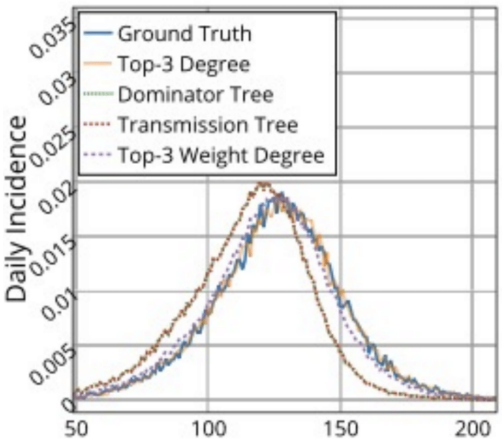


Fig. 10. Daily incidence of sensor sets selected by the heuristic approaches compared to the true daily incidence in the simulated epidemic on Miami dataset.

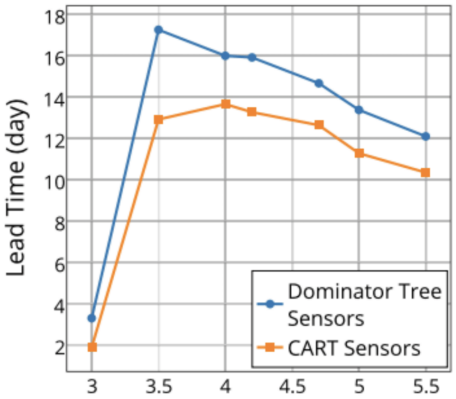


Fig. 11. Mean lead times estimated with surrogate sensor set and dominator tree based social sensors for various flu transmission rates.

The above figures are from the study "Forecasting the Flu: Designing Social Network Sensors for Epidemics" [6]. Fig. 10 compares daily incidence curves of flu infections between sensor sets in the Miami datasets. In this dataset, the proposed Transmission Tree (TT) and Dominator Tree (DT) heuristics show a much larger lead time (around 10 days), while other methods demonstrate minimal lead time, highlighting the effectiveness of the proposed strategies in complex networks. Fig. 11 shows the comparison between surrogate sensor sets selected using demographic data and dominator tree-based sensor sets across various flu transmission rates. Although the dominator tree sensors perform better, the surrogate sets still provide significant lead times, making them valuable in real-world scenarios where full network data is unavailable.

3 Detection in Cascades

3.1 Water network

Given a water distribution network and data on how contaminants spread over the network, we want to select nodes on where to place the sensors on the water quality to detect the quality. Here, the challenge is to make sure that every contaminant is detected. Hence cascade detection helps place sensors so that every outbreak is detected.

3.2 Cascades in blogs

In this problem, instead of placing sensors to detect outbreaks before they happen, we are given the cascade of an outbreak beforehand and want to place sensors to detect all possible infected nodes. Lescovec *et al.* investigate an analogous problem domain of information propagation between online bloggers' blog posts, for which each story or topic posted about and spread corresponds to an "information cascade" [5]. Selecting the fewest blogs that soonest participate in the most cascades (i.e., the blogs that are the most up-to-date for the greatest number of stories) is analogous to the individuals that, for multiple epidemics, are consistently closest to and soonest infected by the epidemics' patient zeroes. For a solution A to these problems – any set of blog posts, or epidemic patients – Lescovec *et al.* identify multiple criteria to be optimized, each scored and packaged into the vector $R(A)$: (1) *detection likelihood*, the fraction of cascade events any of A's elements participate in; (2) *detection time*, the time elapsed until an element of A becomes involved in a cascade; and (3) *population affected*, those *not* part of a cascade at the moment it is detected by an element of A (in an epidemiological context, those "saved" by detecting an outbreak early).

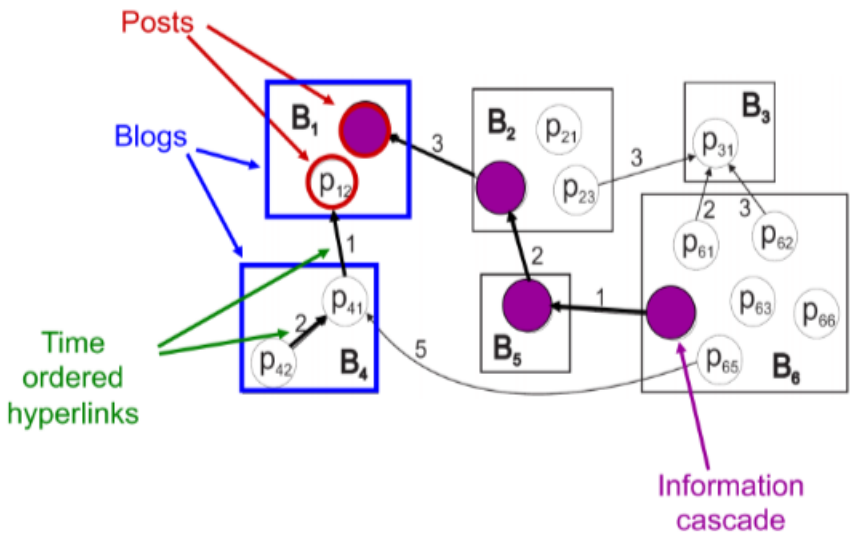


Fig. 12. The time evolution of an information cascade through the posts (p_{ij}, red circles) of blogs (B_i, blue boxes).

Given a series of cascades over a network, place sensors to detect the outbreaks of those cascades. The problem is formulated as follows:

Given: A graph $G = (V, E)$, a budget B for sensors, and cascades
Find: A subset A of nodes that maximize the expected reward R, where:

$$R = \sum_i P(i) R_i(A) = \pi(\varnothing) - \pi(A)$$
$$\text{s.t. } \text{cost}(A) < B$$

To put it simply, we simply trying to pick nodes up to a budget B such that we maximize the expected reward R of those nodes. We calculated the reward of a set of nodes by looking over all i cascades and summing the reward of the sensors in each of those cascades.

An important property of this function is that $R(A)$ is submodular (for discrete state functions), meaning it can be approximated in a reasonable amount of time. This is crucial because trying to solve this problem by brute force is would be an impossibly expensive task on large graphs.

By applying submodularity for unit cost, each time we select a node with the maximum marginal gain. Through this we get a (1-1/e) approximation algorithm which is 63% of the optimal.

3.3 CELF: Speed up Greedy Algorithms

In the above cases, greedy algorithms are implemented.The CELF (Cost-Effective Lazy Forward) algorithm is a variant of the greedy algorithm designed to speed up the process of selecting the most influential nodes in a network, particularly for influence maximization tasks. It optimizes the standard greedy approach by maintaining a priority queue of nodes based on their marginal gain, recalculating only when necessary. This "lazy evaluation" significantly reduces the number of evaluations required, making the selection process faster and more efficient without sacrificing accuracy. Below summarize the key features of CELF algorithm.

- **CELF (Cost Effective Lazy Forward)**
  - Idea: marginal gain decreases as the solution size increases:

    $$\delta_a(S_{t-1}) \geq \delta_a(S_t)$$

  - Each time sort the marginal gain
    - If $\delta_b(S_t) \geq \delta_a(S_{t-1})$, we can make sure $\delta_b(S_t)$ is the maximum marginal gain at time t
  - Lazy evaluations!
    - Evaluating only top values of marginal gain

Please refer to the figure below to see the comparison between CELF algorithm to other algorithms. We can see in the below figures that CELF performs best achieving the highest reduction in affected population.

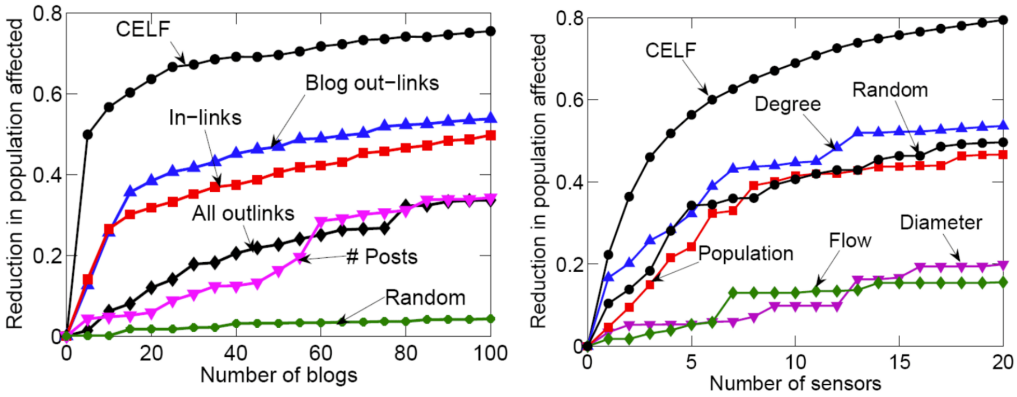


Fig. 13. The diminishing-returns performance of the CELF algorithm in detecting information diffusion in blogs (left) and epidemic outbreaks (right). Because CELF is non-negative, monotonically increasing, and exhibits the diminishing-returns property, we can conclude it to be an acceptable approximation of a submodular function, and thus a valid outbreak detector.

Additionally the below figures depict the running time of the algorithm. It can be seen that the CELF algorithm runs significantly faster than the naive greedy method.

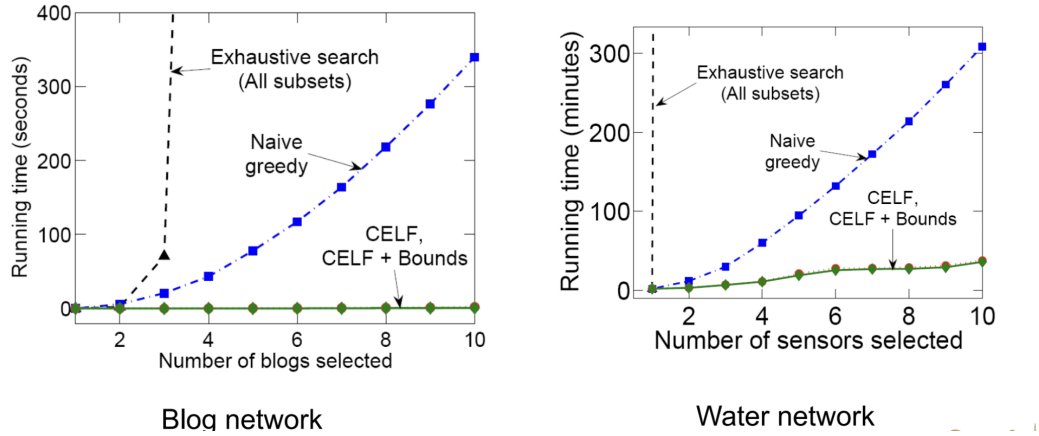


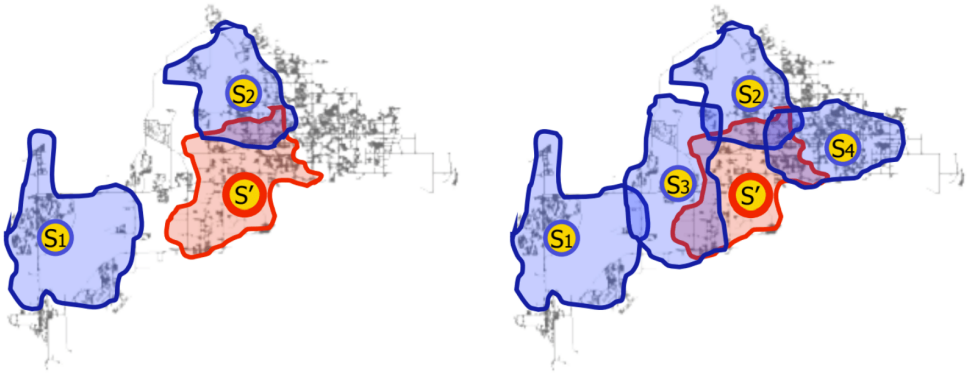
Fig. 14. Running time of CELF algorithm compared to Naive Greedy method.

4 Submodular functions

Submodular functions are basically a property of a function. For a greedy algorithm to work, the outbreak detection objective functions f (such as detection likelihood and detection time for a subset S of nodes) must be submodular, or change more slowly as the size of S increases (*diminishing returns*: every new node added to S should contribute less and less to the value of the objective $f(S)$). $f(S)$ is submodular when:

- Non-negative (the size of the set)
- Monotone (adding elements should increase the function) $f(S + v) \geq f(S)$

- Has diminishing returns property, where $f(S+v) - f(S) \geq f(T+v) - f(T)$ for all $S \subseteq T$ (the gain of adding a node v to a smaller set S is greater than adding v to a larger set T)



(a) *Adding s' to set $\{s_1, s_2\}$* (b) *Adding s' to superset $\{s_1, \ldots, s_4\}$*

Fig. 15. Illustration of the diminishing returns effect in context of placing sensors in a water distribution network to detect contaminations. The blue regions indicate nodes where contamination is detected quickly using the existing sensors S. The red region indicates the additional coverage by adding a new sensor s'. If more sensors are already placed (b), there is more overlap, hence less gain in utility.

The above figure is from "Submodular Function Maximization" by Andreas Krause and Daniel Golovin [2]. The figures show the water distribution network for a city where we have sensors. Each sensor will have some voronoid space. Each sensor covers an area. In Fig. 15(a), Suppose there are S1 and S2 and we add s', the additional area covered by s' is large. In Fig. 15(b), there are four sensors S1, S2, S3 and S4 already present. In this case adding s' will be give a benefit but it will not be as large. Hence the marginal utility will be lesser.

Hence, submodular functions are sort of like convexity and sort of like concavity. Convex functions are sued for continuous optimization whereas submodular functions are used for discrete optimization.

Property of submodular functions: If $f_1(x), f_2(x), \ldots, f_k(x)$ are submodular functions, then for any constants $c_1, c_2, \ldots, c_k \geq 0$, $F(x) = \sum_i c_i \cdot f_i(x)$ is also submodular. Linear combination of submodularity.

Example: Consider the following function for sets X_1,X_2,...., X_m

$$f(S) = \left| \bigcup_{i \in S} X_i \right|$$

The above functions is submodular since:

- The function always produces non-negative values since the size of a union of sets cannot be negative.

- The function value never decreases as the set S grows; adding more elements always increases or maintains the size of the union.
- Adding an element to a smaller set provides a large marginal gain than adding it to a larger set.

There are many other cases of submodular functions like Influence maximization (Viral marketing problem to pick the most influential nodes since every influencer you spend money on, you get diminishing marginal benefit), Facility Locations, Entropy, Immunization etc.

4.1 Efficient Optimization

We can do efficient optimization for submodular functions. In this case, we usually have a budget constraint where the size of the set should not be more than k. We need to pick k sensors. For this we need to pick the best set to maximize the rewards. Note the following:

$$S^* = \arg\max_S f(S), \text{ s.t. } |S| = k$$

The efficient optimization of submodular functions is an NP-Hard problem, but designing the outbreak detection reward function R to be submodular permits the use of greedy algorithms to get an *approximation*, such as hill-climbing techniques.

References

[1] N. A. Christakis and J. H. Fowler. Social network sensors for early detection of contagious outbreaks. *PLoS ONE*, 5(9), 2010.

[2] A. Krause and D. Golovin. Submodular function maximization. *Survey on Submodular Function Maximization*, 2012.

[3] Y. Kryvasheyeu, H. Chen, E. Moro, P. Van Hentenryck, and M. Cebrian. Performance of social network sensors during hurricane sandy. *PLOS ONE*, 2015.

[4] K. Lerman, X. Yan, and X.-Z. Wu. The "majority illusion" in social networks. *PLOS ONE*, 11(2), 2016.

[5] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2007.

[6] H. Shao, K. T. Hossain, H. Wu, M. Khan, A. Vullikanti, B. A. Prakash, M. Marathe, and N. Ramakrishnan. Forecasting the flu: Designing social network sensors for epidemics. *Proceedings of ACM International Workshop on Epidemiology meets Data Mining and Knowledge Discovery (KDD epiDAMIK'18)*, 2018.