Influence maximisation

Social and Technological Networks

Rik Sarkar

University of Edinburgh, 2018.

Course

- Make sure you are comfortable with graph theory, exercise and notes sets 0 and 1.
- And programming graphs, ipython etc
- These are simple basics. We will build upon these throughout
- Important for theory, *and* coursework

Course

- Piazza forum up at: http://piazza.com/ed.ac.uk/fall2018/infr11124/
- Please join. We will post announcements etc there.
- Its main purpose is as a forum for you to discuss course material
  - Ask questions and answer them. Post relevant things
  - We will answers some questions, not all (and we may be wrong!)
  - Discuss and find answers yourself
  - If you are not sure if your answer is correct, try to articulate the doubt exactly, and the search for answers!

Unknown optimals

- The final size of the cascade depends on the initial choice of k nodes
- Suppose the best possible choice gives us a cascade of size OPT (short for optimal)
- We do not know what OPT is
- We do not know which starting set gives OPT
- Computing the OPT or the OPT set is NP-hard
  - Computationally intractable. Takes exponential time. Almost as bad as trying all possible starting sets.

Approximations

- Exact OPT solution is impractical
- In many cases a “good” solution will suffice. We do not need the perfect solution
  - As long as we can say what we mean by good

- Suppose we find an algorithm produces a cascade of size c*OPT
  - It is a c-approximation
  - E.g. ½ approximation reaches ½ of OPT nodes

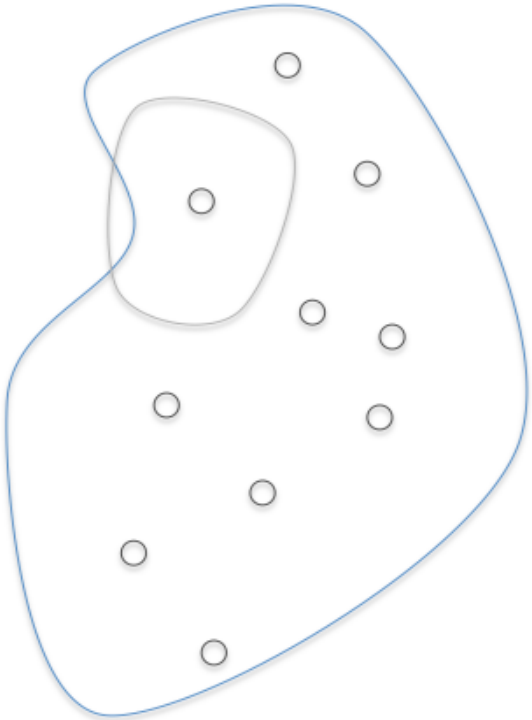
- For the maximizing sphere of influence problem, there is a simple algorithm that gives an approximation of

$$\left(1 - \frac{1}{e}\right)$$

- To prove this, we will use a property called *submodularity*
  - A fundamental concept in machine learning

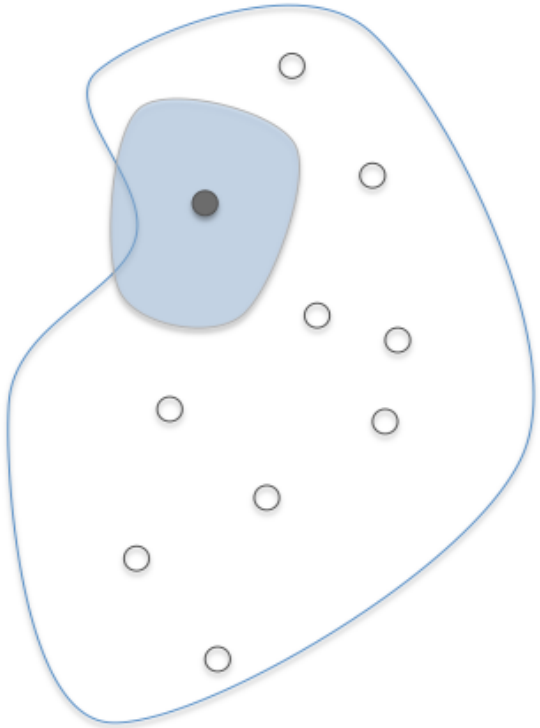
Example: Camera coverage

- Suppose you are placing sensors/cameras to monitor a region (eg. cameras, or chemical sensors etc)
- There are n possible camera locations
- Each camera can “see” a region
- A region that is in the view of one or more sensors is *covered*
- With a budget of k cameras, we want to cover the largest possible area
  - Function f: Area covered



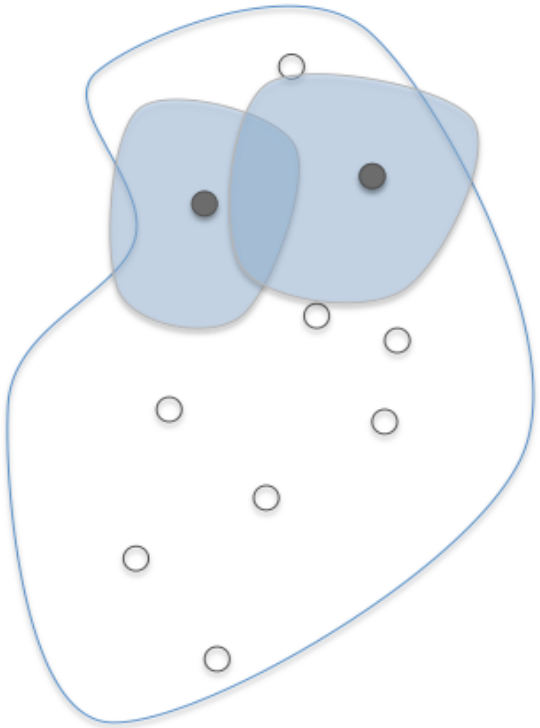
Marginal gains

- Observe:
- Marginal coverage depends on other sensors in the selection



Marginal gains

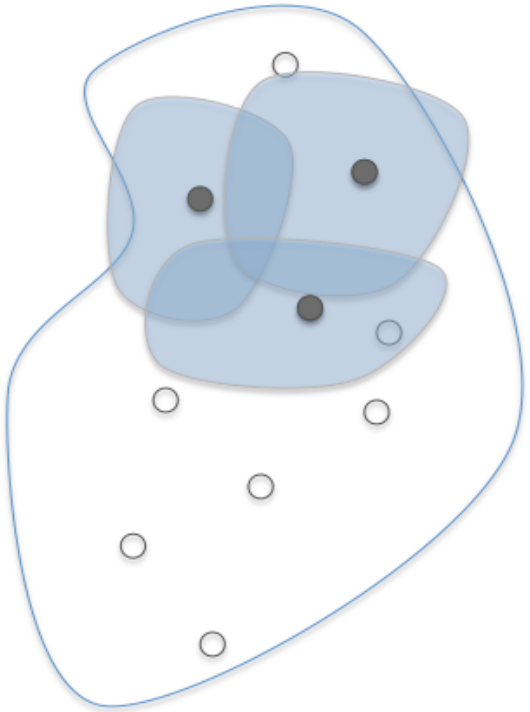
- Observe:
- Marginal coverage depends on other sensors in the selection



Marginal gains

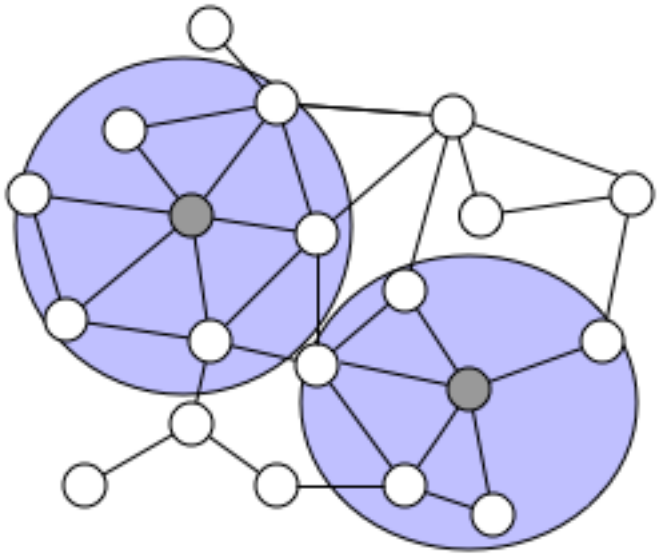
- Observe:
- Marginal coverage depends on other sensors in the selection
- More selected sensors means less marginal gain from each individual

- Exactly the same as social sphere of influence problem



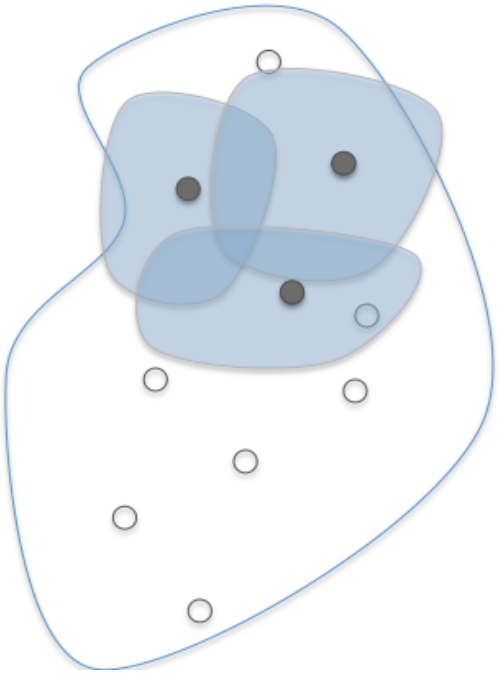
Influences in networks

- We are considering simple neighborhoods
  - E.g. In the figure below, selection of node x covers the set of neighbors N(x)
- A node can influence other nodes in its neighborhood: E.g. expose a friend to a new product.
- We are trying to maximize the number of people exposed to our product



Submodular functions

- Suppose function f(x) represents the total benefit of selecting x
  - Like area covered
  - And f(S) the benefit of selecting set S
- Function f is submodular if:

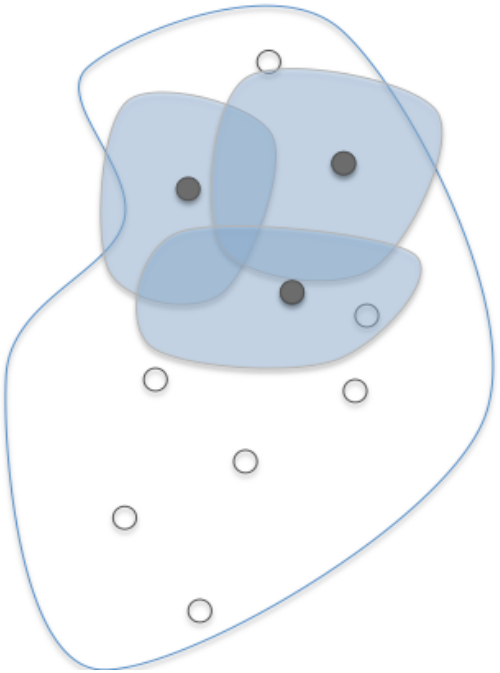


$$S \subseteq T \implies$$

$$f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$$

Submodular functions

- Means *diminishing returns*
- A selection of x gives smaller benefits if many other elements have been selected

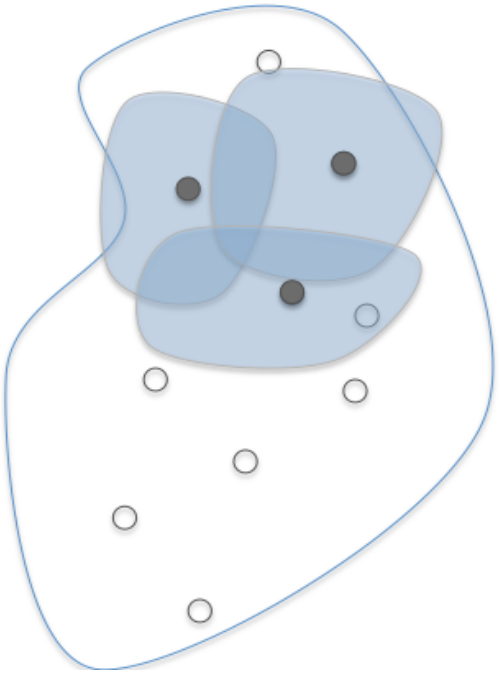


$$S \subseteq T \implies$$

$$f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$$

Submodular functions

- Our Problem: select locations set of size k that maximizes coverage
- NP-Hard



$$S \subseteq T \implies$$

$$f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$$

Greedy Approximation algorithm

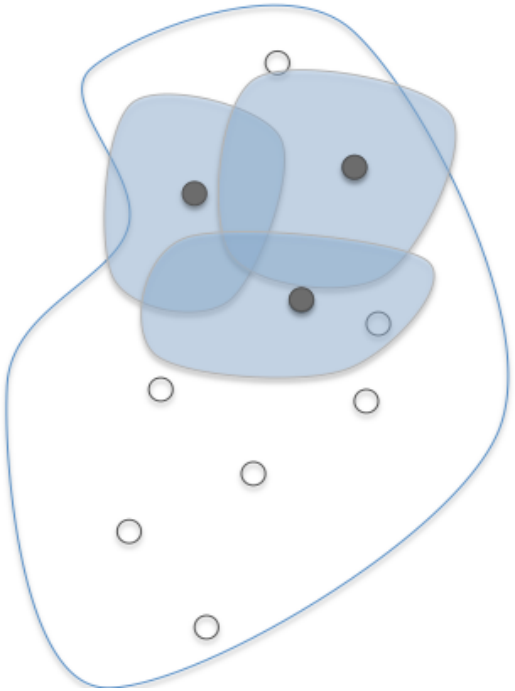
- Start with empty set S = $\varnothing$
- Repeat k times:
- Find v that gives maximum marginal gain:

$$f(S \cup \{v\}) - f(S)$$

- Insert v into S

- Observation 1: Coverage function is submodular
- Observation 2: Coverage function is monotone:
- Adding more sensors always increases coverage

$$S \subseteq T \Rightarrow f(S) \leq f(T)$$



Theorem

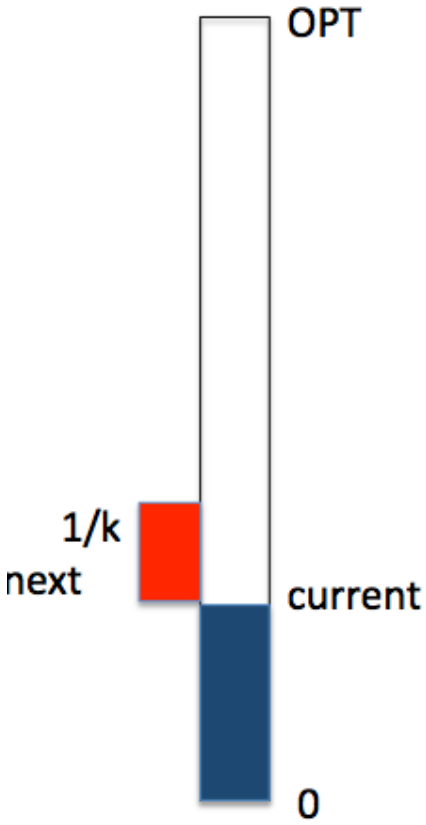
- For monotone submodular functions, the greedy algorithm produces a $\left(1 - \frac{1}{e}\right)$ approximation

- That is, the value f(S) of the final set is at least
  - [Nemhauser et al. 1978] $\left(1 - \frac{1}{e}\right) \cdot OPT$

- (Note that this algorithm applies to submodular maximzation problems, not to minimization)

Proof

- Idea:
- OPT is the max possible
- At every step there is at least one element that covers at least 1/k of remaining:
  - So ≥(OPT - current) * 1/k
- Greedy selects one such element

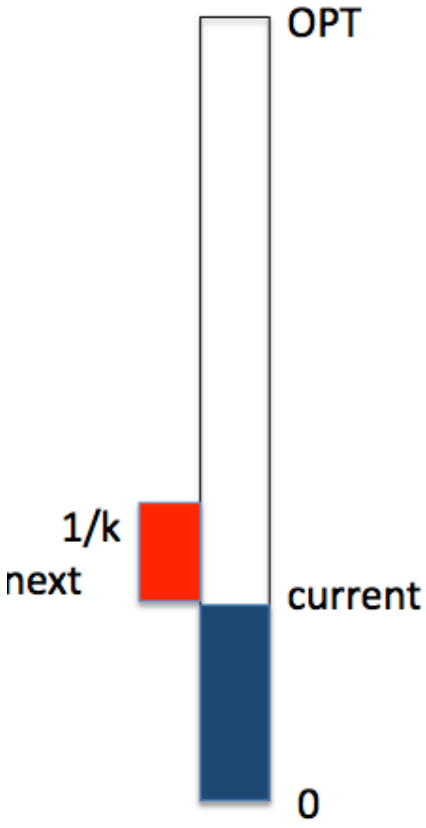


Proof

- Idea:
- At each step coverage remaining becomes

$$\left(1 - \frac{1}{k}\right)$$

- Of what was remaining after previous step



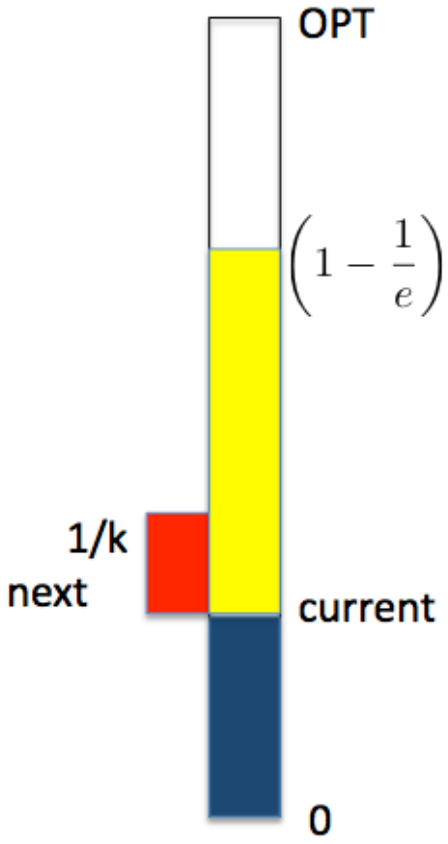
Proof

- After k steps, we have remaining coverage of OPT

$$\left(1-\frac{1}{k}\right)^{k} \simeq \frac{1}{e}$$

- Fraction of OPT covered:

$$\left(1-\frac{1}{e}\right)$$



Proof of the main claim

- Suppose the unknown set of elements that gives OPT is given by set C
- And suppose S_i is the set selected by greedy upto step i
- At every step there is at least one element that covers 1/k of remaining:
  - So $(f(C) - f(S_i)) * 1/k$
- Why is this true?

Proof of the main claim

- At every step there is at least one element that covers 1/k of remaining:
  - So (OPT - current) * 1/k

- At step 0: Suppose to the contrary, there is no such element.
  - Then k cannot give OPT: contradiction.

Proof of the main claim

- At S_i, OPT has available at least on element in C that covers at least at least 1/k of the rest of OPT
  - Since we can simply add k elements from C to get OPT
- Now consider Greedy
  - If greedy chose s_i at step i, that is because it gives at least as much marginal gain as any element in C
    - So, s_i covers at least $(f(C) – f(S_i))/k$

Homework

- Write out the proof nicely!

- Theorem:
  - Positive linear combinations of monotone submodular functions is monotone submodular

- We have shown that monotone submodular maximization can be approximated using greedy selection

- To show that maximizing spread of cascading influence can be approximated:
  - We will show that the function is monotone and submodular

Cascades

- We are going to use a simplified model

Cascade in independent activation model

- If node u activates to use A, then u causes neighbor v to activate and use A with probability $p_{u,v}$
  - E.g. This probability may depend on strength of friendship
- Now suppose u has been activated
  - Neighbor v will be activated with prob. $p_{u,v}$
  - Neighbor w will be activated with prob. $p_{u,w}$ etc..
  - On a particular trial, on activation of u, a certain set of other nodes will be activated.
    - (depending on random choices in that trial.)

Cascade in independent activation model

- Let us take one such trial activations (call it X1).
- Tells us which edges of u are “effective” when u is “on”
- Similarly for other nodes v, w, y ….
- Gives us exactly which nodes will be activated as a consequence of u being activated
  - A tree rooted at u
- Say, c(u) is the set of nodes *covered* by u in X1
  - The nodes in the activation tree

- Say, c(u) is the set of nodes covered by u.
- c(S) is the set of nodes covered by a set S
- f(S) = |c(S)| is submodular

- Remember that we had made the probabilistic choices for each edge uv:
- That is, we made a set of choices representing the entire network
- We used X1 to represent this configuration

- We showed that on a particular trial X1, the function is submodular, since this is just a coverage problem

- But what about other X?
  - Can we say that over all X we have submodularity?

- We sum over all possible Xi, weighted by their probability.

- Non-negative linear combinations of submodular functions are submodular,
  - Therefore the sum of all x is submodular
  - (homework!)

- The approximation algorithm for submodular maximization is an approximation for the cascade in independent activation model with same factor

Linear threshold model

- Linear contagion threshold model:
- The model we have used: node activates to use A if benefit of using $p > q$

- Also submodular and monotone

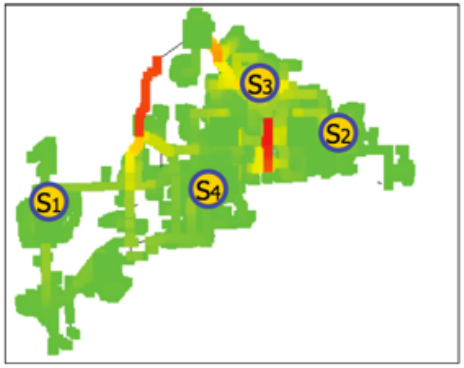
- Proof ommitted.
  - If you are interested, see additional reading: Kempe, Kleinberg, Tardos; KDD03

Applications of submodular optimization

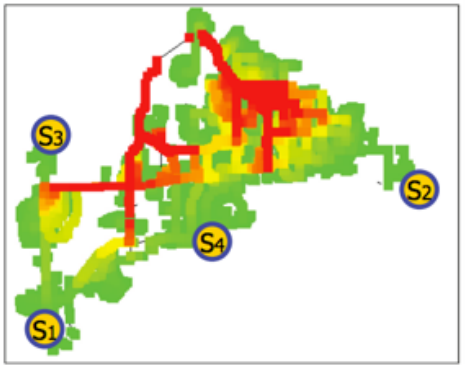
- Sensing the contagion
- Place sensors to detect the spread
- Find “representative elements”: Which blogs cover all topics?
- Machine learning selection of sets
- Exemplar based clustering (eg: what are good seed for centers?)
- Image segmentation

Sensing the contagion

- Consider a different problem:
- A water distribution system may get contaminated
- We want to place sensors such that contamination is detected



(c) effective placement



(d) poor placement

Social sensing

- Which blogs should I read? Which twitter accounts should I follow?
  - Catch big breaking stories early
- Detect cascades
  - Detect large cascades
  - Detect them early…
  - With few sensors
- Can be seen as submodular optimization problem:
  - Maximize the "quality" of sensing

- Ref: Krause, Guestrin; Submodularity and its application in optimized information gathering, TIST 2011

Representative elements

- Take a set of Big data
- Most of these may be redundant and not so useful
- What are some useful "representative elements"?
  - Good enough sample to understand the dataset
  - Cluster representatives
  - Representative images
  - Few blogs that cover main areas…

Problem with submodular maximization

- Can be expensive!
- Each iteration costs O(n): have to check each element to find the best
  - May be more: “checks” are complex and depend on current selection
- Problem in large datasets
- Distributed cluster computation can help
  - Split data into multiple computers
  - Compute and merge back results: Works for many types of problems

- Ref: Mirzasoleiman, Karbasi, Sarkar, Krause; Distributed submodular maximization: Finding representative elements in massive data. NIPS 2013.

Summary

- Approximation algorithms
- Critical in practical scenario, since “perfect” answer may be elusive
  - We can find approximations without even knowing the OPT!
- Critical in Machine learning
  - Learning is always approximate
  - We never know the perfect answer for future
  - Learning theory relies on probability and approximations
- Submodular optimisations are a powerful set of tools

- Notes for this class will be put up

Next class

- Prepare: Web graphs and search
- **Important:** Read beforehand: Kleinberg & Easley 2010– Chapter 13 & 14.
- Look up on: Representation of networks, Matrices, adjacency matrices, matrix multiplication.