Influence Maximization

Ariah Klages-Mundt

Influence maximization Social networks (among others) involve processes for influence propagation in the network. Examples include diffusion of technological innovation, beliefs, product adoption, and posting of 'viral' content. A natural question is how to engineer such a viral cascade given information about the network. Such a problem is to maximize influence propagation by choosing an optimal set of seed nodes S to directly influence given a budget b.

The *Linear Threshold Model* is a simple model for influence propagation. In this model, a node v is influenced by each of its neighbors w by amount $b_{v,w}$ such that $\sum_{w\sim v} b_{v,w} \leq 1$. Each node v further has a threshold θ_v, which gives the weighted fraction of v's neighbors that, if activated, in turn activate v. *Integral Influence Maximization*, studied in [1], focuses on finding an optimal seed set S on which to spend θ_v on v for each $v \in S$.

A generalization of the integral case leads to *Fractional Influence Maximization*, as studied in [3]. In this problem, we choose a vector $\mathbf{x}$ with $\mathbf{1}^T\mathbf{x} \leq b$ of influence to exert on nodes. The amounts can be a fraction of the thresholds of the nodes. This allows more efficient use of budget b to influence an effective seed set S. In particular, this takes advantage of the fact that we don't have to spend as much to influence a node that already has partial influence exerted from other influenced nodes.

Both the integral problem of selecting the optimal seed set S and the fractional problem of selecting the optimal influence vector $\mathbf{x}$ are NP-hard, as shown in [1], [3]. Further, they are also hard to approximate to within any general nontrivial factor.

However, when we consider a modified problem with uncertain thresholds–e.g., if activation thresholds for influence are uniform random variables–then the problem changes enough in expectation to lower complexity. In particular, the expected cascade size from a given seed set becomes submodular and allows a greedy approximation that is provably within $(1-1/e) \approx 63\%$ of optimal ([1],[3]). This works for the linear threshold model as well as more general threshold models, as shown in [2].

I define the greedy algorithms for integral influence maximization and fractional influence maximization below. The general structure of these algorithms is to start with an empty seed set S and, iteratively, add the node v to S that gives the maximum marginal gain. Since the thresholds are random (distributed by Θ), determining the maximum marginal gain involves estimating the expected size of resulting cascades $\sigma(S\cup\{v\}) = \mathbf{E}_\Theta\Big[\text{cascade size}|S\cup\{v\}\Big]$. A similar definition applies to $\sigma(\mathbf{x})$. This is typically done through Monte Carlo estimation of the expectation, which can make these greedy algorithms prohibitively slow, although still within polynomial time complexity.

I also define the heuristic algorithm `DiscountFrac` used in [3] that tries to estimate the greedy algorithm in faster time. This algorithm uses a similar greedy approach. Starting with an empty seed set S, it iteratively adds the node v to S that would exert the most total influence on the remaining unactivated nodes.

The algorithms below use the following problem setting:

- $f(S)$ outputs the vector of influence exerted by the activation of node set S on each node. In the analysis, we define f to give the linear threshold model.
- $w(S)$ outputs a weight of node set S. In the analysis, we define w to weight each node by 1.

- $\Theta = \text{uniform}[0,1]^n$ is the distribution for node thresholds (n=number of nodes).
- b = budget.

Algorithm 1 `CalcIntCascade`$(S; f, \theta)$

Input: set S, set function f, thresholds θ

Initialize $S_0 \leftarrow \emptyset$, $S_1 \leftarrow S$, $i \leftarrow 1$
while $S_i \neq S_{i-1}$ **do**
  $S_{i+1} = \{\text{node } v | f(S_i)[v] \geq \theta[v]\} \cup S_i$
  $i \leftarrow i + 1$
end while
return S_i

Algorithm 2 $\hat{\sigma}(S)$ estimate of $\sigma(S)$ for integral influence

---

Input: set S, set function f, weight function w, thresholds distr. Θ, sample size $k = 10,000$

Initialize $\sigma \leftarrow 0$
for $i \leq k$ **do**
  Sample $\theta \sim \Theta$
  $T, =$ `CalcIntCascade`$\Big(S; f, \theta\Big)$
  $\sigma \leftarrow \sigma + w(T)$
end for
return σ / k

Algorithm 3 `GreedyIntInfMax` = Greedy algorithm for integral influence maximization

---

Input: set function f, weight function w, thresholds distr. Θ, budget b

Initialize $S_0 \leftarrow \emptyset$, $i \leftarrow 0$
while $|S_i| < b$ **do**
  for node $v \notin S_i$ **do**
    $\mathbf{q}[v] = \hat{\sigma}\Big(S_i \cup \{v\}; f, \Theta, w\Big)$
  end for
  $S_{i+1} \leftarrow S_i \cup \{\arg\max \mathbf{q}\}$, $i \leftarrow i + 1$
end while
if $|S_i| \leq b$ **then**
  return S_i
else
  return S_{i-1}
end if

Algorithm 4 `CalcFracCascade`$(\mathbf{x}; f, \theta)$

Input: vector $\mathbf{x}$, set function f, thresholds θ

Initialize $S_0 \leftarrow \emptyset$, $i \leftarrow 1$
$S_1 \leftarrow \{\text{node } v | \mathbf{x}[v] \geq \theta[v]\}$
while $S_i \neq S_{i-1}$ **do**
  $S_{i+1} = \{\text{node } v | f(S_i)[v] + \mathbf{x}[v] \geq \theta[v]\}$
  $i \leftarrow i + 1$
end while
return S_i

Algorithm 5 $\hat{\sigma}(x)$ estimate of $\sigma(x)$ for fractional influence

Input: vector $\mathbf{x}$, set function f, weight function w, thresholds distr. Θ, sample size $k = 10,000$

Initialize $\sigma \leftarrow 0$
for $i \leq k$ **do**
  Sample $\theta \sim \Theta$
  $T = \texttt{CalcFracCascade}\Big(\mathbf{x}; f, \theta\Big)$
  $\sigma \leftarrow \sigma + w(T)$
end for
return σ / k

Algorithm 6 `GreedyFracInfMax` = Greedy algorithm for fractional influence maximization

Input: set function f, weight function w, thresholds distr. Θ, budget b

Initialize $\mathbf{x_0} \leftarrow \mathbf{0}$, $i \leftarrow 0$
while $\mathbf{1}^T \mathbf{x_i} < b$ **do**
  $S_i = \{\text{node } v | \mathbf{x_i}[v] > 0\}$
  for node $v \notin S_i$ **do**
    $\mathbf{x_v} = \mathbf{x_i} + \Big(\theta_{\max}[v] - \Gamma^+(v, S_i)\Big)\mathbf{1}_v$
    $\mathbf{q}[v] = \hat{\sigma}\Big(\mathbf{x_v}; f, \Theta, w\Big)$
  end for
  $u = \arg\max \mathbf{q}$
  $\mathbf{x_{i+1}} \leftarrow \mathbf{x_i} + \Big(\theta_{\max}[u] - \Gamma^+(u, S_i)\Big)\mathbf{1}_u$, $i \leftarrow i + 1$
end while
if $\mathbf{1}^T \mathbf{x_i} \leq b$ **then**
  return $\mathbf{x_i}$
else
  return $\mathbf{x_{i-1}}$
end if

Algorithm 7 $\Gamma^+(v, A)$ = total sum of weight of edges from set A to node v

Input: set A, set function f, node v

return $f(A)[v]$

Algorithm 8 $\Gamma^-(v, A)$ = total sum of weight of edges from node v to set A

Input: set A, set function f, node v

  return $\mathbf{1}_A^T f(\{v\})$

Algorithm 9 `DiscountFrac` heuristic algorithm

Input: set function f, weight function w, thresholds distr. Θ, budget b

  Initialize $\mathbf{x_0} \leftarrow \mathbf{0}$, $i \leftarrow 0$
  while $\mathbf{1}^T \mathbf{x_i} < b$ **do**
    $S_i = \{\text{node } v | \mathbf{x_i}[v] > 0\}$
    for node $v \notin S_i$ **do**
      $\mathbf{q}[v] = \Gamma^-(v, V \setminus S_i)$
    end for
    $u = \arg\max \mathbf{q}$
    $\mathbf{x_{i+1}} \leftarrow \mathbf{x_i} + \Big(\theta_{\max}[u] - \Gamma^+(u, S_i)\Big)\mathbf{1}_u$, $i \leftarrow i + 1$
  end while
  if $\mathbf{1}^T \mathbf{x_i} \leq b$ **then**
    return $\mathbf{x_i}$
  else
    return $\mathbf{x_{i-1}}$
  end if

References

[1] Kempe, D., Kleinberg, J., Tardos, E. (2003). Maximizing the spread of influence through a social network. In *KDD*, ACM, 137-146.

[2] Mossel, E., Roch, S. (2007). On the submodularity of influence in social networks. In *STOC*, ACM, 128-134.

[3] Demaine, E., Hajiaghayi, M.T., Mahini, H., Malec, D., Raghavan, S., Sawant, A., Zadimoghadam, M. (2014). How to influence people with partial incentives. In *WWW*, ACM, 937-948.