Microsoft Research

Microsoft

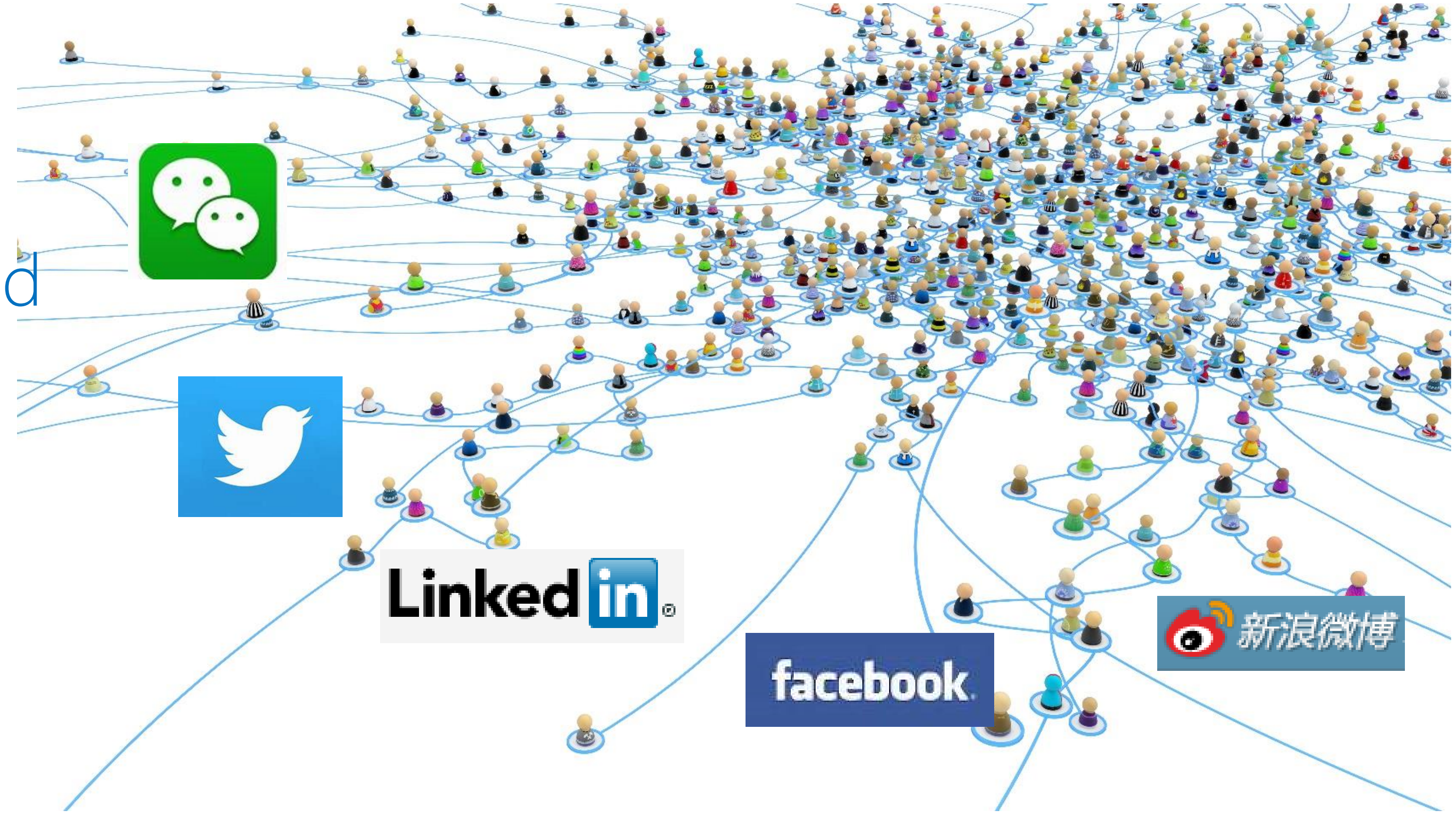
Influence Maximization: Integrating and Expanding Classical Algorithms into the Social Network Context

Wei Chen 陈卫
Microsoft Research Asia

CNCC'2019, Classical Algorithms Forum, Oct. 18, 2019

Social Network Mining

- Social network mining
  - Community detection
  - Influence propagation and maximization
  - Link prediction
  - frequent pattern mining
  - etc.



Classical Algorithms

- Meta algorithms (algorithmic techniques):
  - greedy
  - dynamic programming (1955),
  - linear programming (~1939)
  - divide and conquer (~1945)
- Graph algorithms:
  - BFS/DFS, Dijkstra shortest path algorithm (1959)
- Online learning:
  - Thompson sampling (1933)
  - UCB1 (2002)

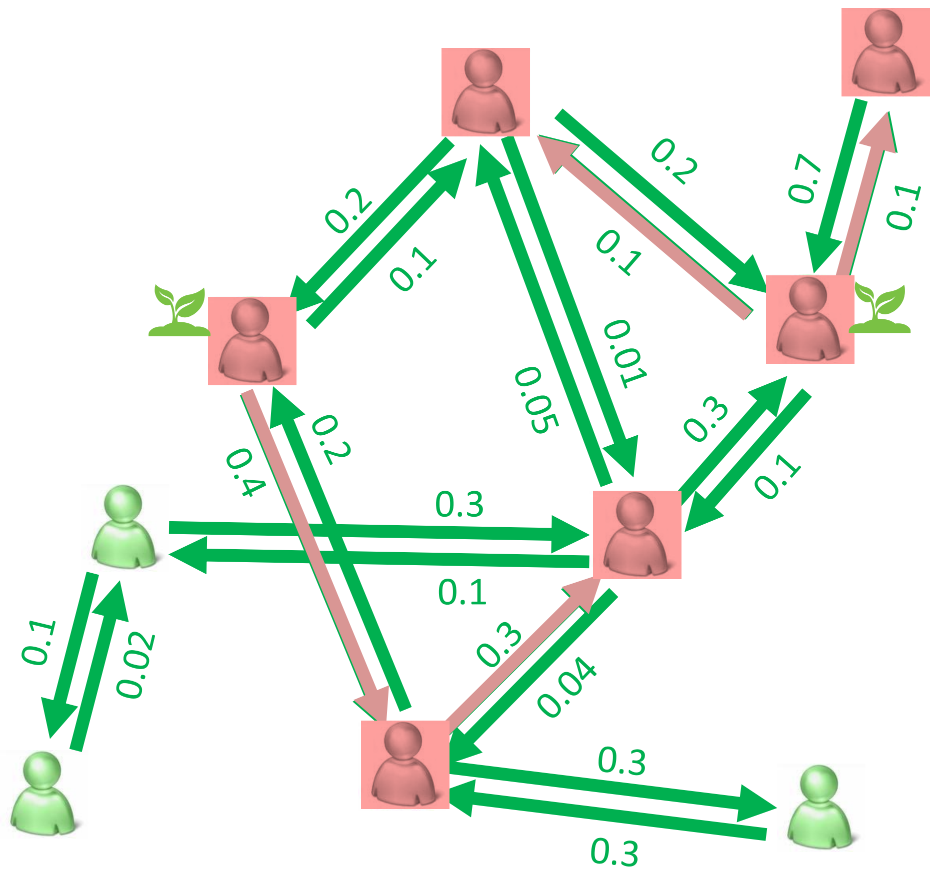
Research on Influence Maximization

Influence Propagation Modeling and Influence maximization task

- Stochastic diffusion models: how information/influence propagates in social networks
  - Its properties, e.g. submodularity
- Influence maximization: given a budget $\boldsymbol{k}$, select at most $\boldsymbol{k}$ nodes in a social network as seeds to maximize the influence spread of the seeds
  - Applications in viral marketing, diffusion monitoring, rumor control, etc.

Independent cascade model

- Each edge (u, v) has a *influence probability* $p(u, v)$
- Initially seed nodes in S_0 are activated
- At each step t, each node u activated at step $t-1$ activates its neighbor v independently with probability $p(u, v)$
- Influence spread $\sigma(S)$: expected number of activated nodes
- Other models: linear threshold (LT), general threshold, etc.



Influence maximization

- Given a social network, a diffusion model with given parameters, and a number $\boldsymbol{k}$, find a seed set $\boldsymbol{S}$ of at most $\boldsymbol{k}$ nodes such that the influence spread of $\boldsymbol{S}$ is maximized.
- Based on *submodular function* maximization
- [Kempe, Kleinberg, and Tardos, KDD'2003]

Kempe D, Kleinberg J M, and Tardos É. Maximizing the spread of influence through a social network. KDD'2003

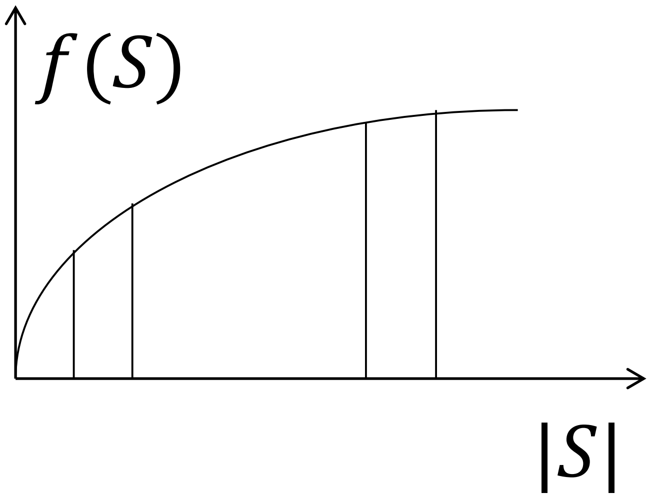
Active Research on Influence Maximization

- Scalable influence maximization
  - make the algorithm run efficiently on large networks
- Variants of influence maximization
  - seed minimization, profit maximization, time-constraint IM
- Adaptive influence maximization
  - adaptive to feedback from already selected seeds
- Online influence maximization
  - learn propagation model parameters while doing maximization
- Multi-item influence maximization
  - competitive IM, complementary IM, welfare maximization

Basic Solution:
Based on the Greedy Algorithm

Submodular set functions

- Sumodularity of set functions $f: 2^V \to R$
  - for all $S \subseteq T \subseteq V$, all $v \in V \setminus T$,
    $$f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T)$$
  - diminishing marginal return
  - an equivalent form: for all $S, T \subseteq V$
    $$f(S \cup T) + f(S \cap T) \leq f(S) + f(T)$$
- Monotonicity of set functions f: for all $S \subseteq T \subseteq V$,
  $$f(S) \leq f(T)$$



Greedy algorithm for submodular function maximization

1: initialize $S = \emptyset$;

2: for $i = 1$ to k do

3: select $u = \mathrm{argmax}_{w \in V \setminus S} [f(S \cup \{w\}) - f(S))]$

4: $S = S \cup \{u\}$

5: end for

6: output S

Property of the greedy algorithm

- Theorem: If the set function f is monotone and submodular with $f(\emptyset) \geq 0$, then the greedy algorithm achieves $(1 - 1/e)$ approximation ratio, that is, the solution S found by the greedy algorithm satisfies:

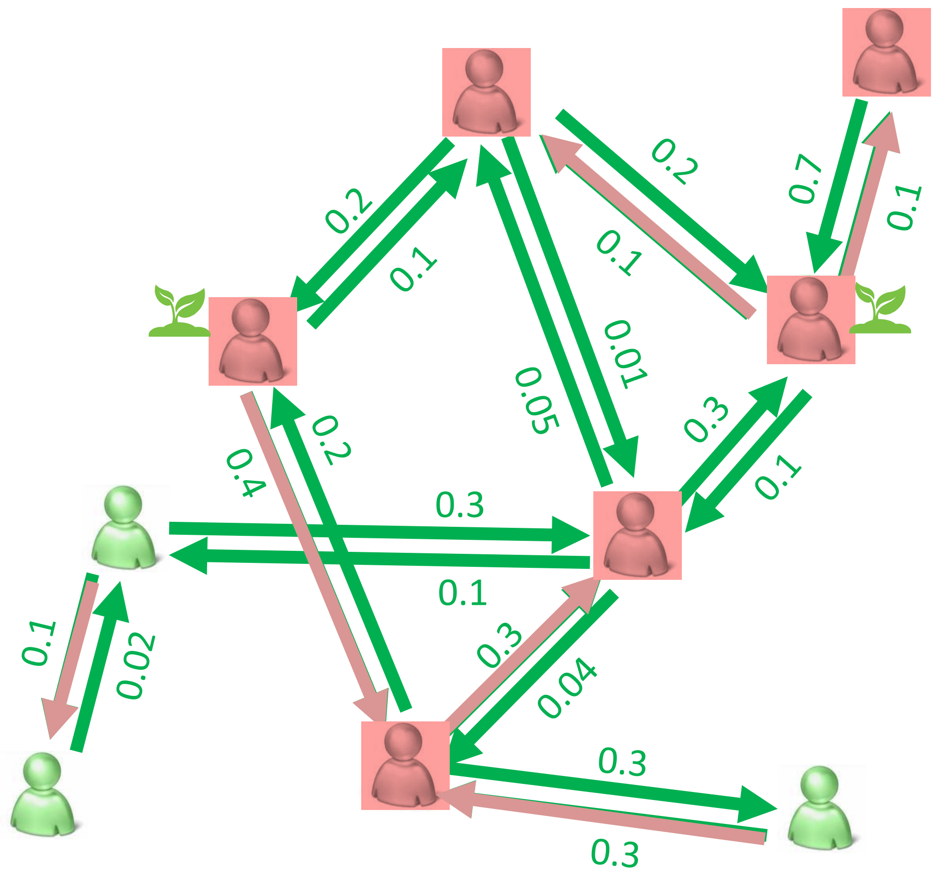
$$f(S) \geq \left(1 - \frac{1}{e}\right) \max_{S' \subseteq V, |S'| = k} f(S')$$

- [Nemhauser, Wolsey and Fisher, 1978]
- Widely used in data mining and machine learning (as approximation algorithms or heuristics)
  - Document summarization, image segmentation, decision tree learning, influence maximization

Nemhauser G L, Wolsey L A, and Fisher M L. An analysis of approximations for maximizing submodular set functions. Mathematical Programming 1978

Submodularity of influence spread function $\sigma(S)$

- Independent cascade model is equivalent to
  - sample live edges by edge probabilities
  - activate nodes reachable from S in the live-edge graph
- $\sigma(S) = \sum_L \Pr\{L\} \cdot |\Gamma(L,S)|$
  - $\Gamma(L,S)$： number of nodes reachable from S in live-edge graph L
  - $|\Gamma(L,S)|$ is a coverage function, easy to show it is submodular



Challenges to the Basic Greedy Solution

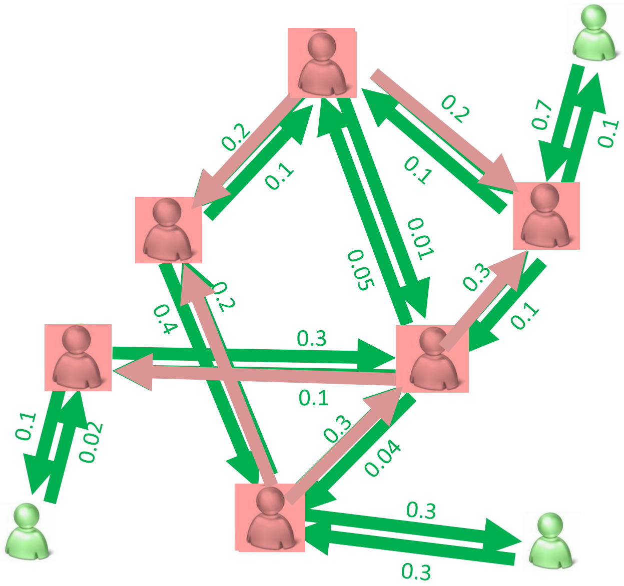
- Scalability challenge:
  - In IC (and LT) models, computing influence spread $\sigma(S)$ for any given S is #P-hard [Chen et al. KDD'2010, ICDM'2010].
  - Implication of #P-hardness of computing $\sigma(S)$
    - Greedy algorithm needs adaptation --- using Monte Carlo simulations
    - But MC-Greedy is very slow: 70+ hours on a 15k node graph to find 50 seeds
- Learning challenge:
  - How to learn the diffusion model?
  - How to use online feedback for optimization --- online influence maximization
- Complex model challenge:
  - Other variants of influence diffusion models, may not be submodular

Scalable Algorithms: Integrating Graph Algorithms

Ways to improve scalability

- Fast deterministic heuristics
  - Utilize model characteristic
  - MIA/IRIE heuristic for IC model [Chen et al. KDD'10, Jung et al. ICDM'12]
  - LDAG/SimPath heuristics for LT model [Chen et al. ICDM'10, Goyal et al. ICDM'11]
  - based on classical graph algorithms, e.g. Dijkstra shortest path algorithm
- Monte Carlo simulation based
  - Lazy evaluation [Leskovec et al. KDD'2007], Reduce the number of influence spread evaluations
- New approach based on Reverse Influence Sampling (RIS)
    - First proposed by Borgs et al. SODA'2014
    - Improved by Tang et al. SIGMOD'2014, 2015 (TIM/TIM+, IMM)

Reverse Influence Sampling (an Illustration)



- Generate RR sets
  - BFS
- Greedily find top k nodes covering most number of RR sets

Reverse Influence Sampling

- Reverse Reachable sets: (use IC model as an example)
  - Select a node $\boldsymbol{v}$ uniformly at random, call it a root
  - From $\boldsymbol{v}$, simulate diffusion, but in reverse order --- every edge direction is reversed, with same probability
  - The set of all nodes reached is the reverse reachable set $\boldsymbol{R}$ (rooted at $\boldsymbol{v}$).
  - [Borgs, Brautbar, Chayes, Lucier'2014]
- Intuition:
  - If a node $\boldsymbol{u}$ often appears in RR sets, it means that if using $\boldsymbol{u}$ as the seed, its influence is large
- Technical guarantee: For any seed set $\boldsymbol{S}$,

$$\sigma(S) = n \cdot \Pr\{S \cap \boldsymbol{R}\}$$

Borgs C, Brautbar M, Chayes J, and Lucier B. Maximizing social influence in nearly optimal time. SODA'2014

IMM: Influence Maximization via Martingales --- Theoretical Result

- Thoerem: For any $\varepsilon > 0$ and $\ell > 0$, IMM achieves $1 - \frac{1}{e} - \varepsilon$ approximation of influence maximization with at least probability $1 - \frac{1}{n^{\ell}}$. The expected running time of IMM is $O\left(\frac{(k+\ell)(m+n)\log n}{\varepsilon^2}\right)$.
- Martingale based probabilistic analysis
  – RR sets are not independent --- early RR sets determine whether later RR sets are generated --- form a martingale

Tang Y, Shi Y, and Xiao X. Influence maximization in near-linear time: A martingale approach. SIGMOD'2015

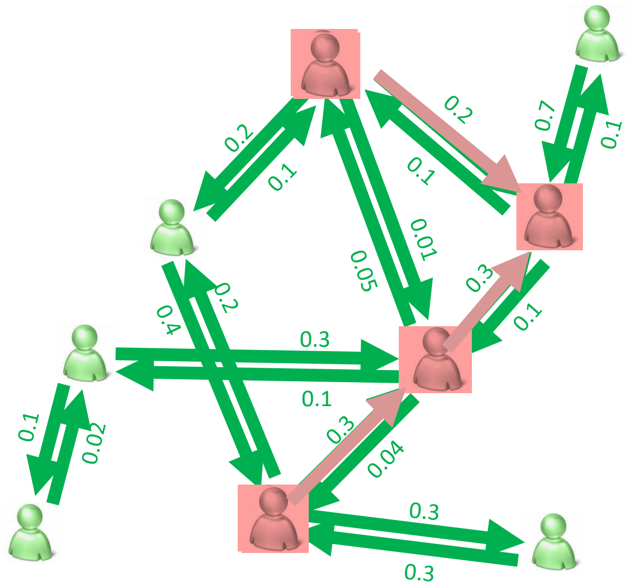
Extension to Spontaneous Adoption

- Node may not be activated by propagation from seeds
  - may be self-activated (e.g. exposure to mass-media marketing)
- We want to identify a set of nodes that can activate most number of nodes before other self-activated reach them
  - preemptive influence maximization [Sun et al. WSDM'2020]
- Expand the model:
  - node has self-activation probabilities, and self-activation delay distribution
  - edge propagation has a delay distribution

Sun L, Chen A, Yu P S, and Chen W. Influence maximization with spontaneous user adoption. WSDM'2020

Extending Reserve Sampling

- When reverse sampling from a node $\boldsymbol{v}$
  - need to sample edge delays to $\boldsymbol{v}$ and self-activation delay of $\boldsymbol{v}$
- Need to guarantee that only sample nodes $\boldsymbol{u}$ whose delay to $\boldsymbol{v}$ is less than or equal to the minimum delay of any self-activated node to $\boldsymbol{v}$
  - How? --- Always do reserve sampling from a node $\boldsymbol{u}$ with minimum delay to $\boldsymbol{v}$
  - Sound familiar? --- It is just like the Dijkstra shortest path algorithm!

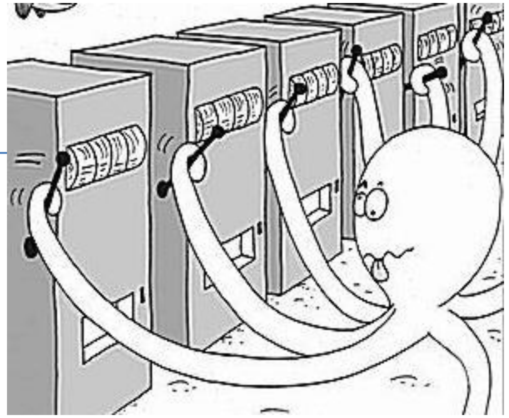


Online Influence Maximization: Expanding Classical Online Learning Algorithms

Online Influence Maximization

- Edge influence probabilities are unknown, need to be learned
- Multiple rounds of online influence maximization. In each round,
  - select $\boldsymbol{k}$ seeds to influence the network
  - observe the diffusion paths and results
  - collect the reward --- the number of nodes activated
  - use the observed feedback to update learning statistics, which is used for seed selection in later rounds
- Falls into the online learning (multi-armed bandit) framework

Multi-armed bandit problem



- There are m arms (machines)
- Arm i has an unknown reward distribution on $[0,1]$ with unknown mean μ_i
  - best arm $\mu^* = \max \mu_i$
- In each round, the player selects one arm to play and observes the reward
- Performance metric: Regret:
  - Regret after playing T rounds $= T\mu^* - \mathbb{E}[\sum_{t=1}^{T} R_t(i_t^A)]$
- Objective: minimize regret in T rounds
- Balancing exploration-exploitation tradeoff
  - exploration (探索): try new arms
  - exploitation (守成): keep playing the best arm so far
- Wide applications: Any scenario requiring selecting best choice from online feedback
  - online recommendations, advertising, wireless channel selection, social networks, A/B testing

Classical MAB Algorithm: UCB1

1: for each arm i: $\hat{\mu}_i = 1$ (empirical mean), $T_i = 0$ (number of observation)

2: for $t = 1, 2, 3, \ldots$ do

3: for each arm i: $\rho_i = \sqrt{\frac{3 \ln t}{2T_i}}$ (confidence radius)

4: for each arm i: $\bar{\mu}_i = \min\{\hat{\mu}_i + \rho_i, 1\}$ (upper confidence bound, UCB)

5: $j = \mathrm{argmax}_i \, \bar{\mu}_i$

6: play arm j, observe its reward $X_{j,t}$

7: update $\hat{\mu}_j = (\hat{\mu}_j \cdot T_j + X_{j,t})/(T_j + 1)$; $T_j = T_j + 1$

6: end-for

For exploration (ρ_i)

For exploitation ($\hat{\mu}_i$)

Guarantee of the UCB1 Algorithm

- Finite-horizon regret:
  - distribution dependent: $O\left(\sum_{\Delta_i>0} \frac{1}{\Delta_i} \ln T\right)$, $\Delta_i = \mu^* - \mu_i$
  - distribution independent: $O(\sqrt{mT \ln T})$

- [Auer, Cesa-Bianchi, and Fischer, 2002]

Auer P, Cesa-Bianchi N, and Fischer P. Finite-time analysis of the multiarmed bandit problem. Machine Learning Journal, 2002(47.2-3):235~256

Challenges applying UCB1 to Online IM

- exponential number of seed sets
  - cannot treat each seed set as an arm
- non-linear reward functions
- offline problem is already NP-hard
- probabilistically triggering new arms in a play

Extending the MAB Framework

- Extend MAB to combinatorial MAB framework with probabilistically triggered arms (CMAB-T)
  - Model: In each round one action/super-arm is played, which triggers a set of base arms (triggering may be probabilistic)
  - precisely characterize the bounded smoothness condition required to solve CMAB-T
  - propose the CUCB algorithm based on an offline approximation oracle
  - distribution-dependent and distribution-independent regret analysis
  - applicable to a large class of combinatorial online learning problems
- [Chen et al JMLR'2016, Wang and Chen, NIPS'2017]

Chen W, Wang Y, Yuan Y, and Wang Q. Combinatorial multi-armed bandit and its extension to probabilistically triggered arms. Journal of Machine Learning Research, 2016(17.50):1~33.

Wang Q and Chen W. Improving regret bounds for combinatorial semi-bandits with probabilistically triggered arms and its applications. NIPS'2017

CUCB Algorithm

1: for each arm i: $\hat{\mu}_i = 1$ (empirical mean), $T_i = 0$ (number of observation)

2: for $t = 1, 2, 3, \dots$ do

3: for each arm i: $\rho_i = \sqrt{\frac{3 \ln t}{2T_i}}$ (confidence radius)

4: for each arm i: $\bar{\mu}_i = \min\{\hat{\mu}_i + \rho_i, 1\}$ (upper confidence bound, UCB)

5: $S = \mathrm{OfflineOracle}(\bar{\mu}_1, \dots, \bar{\mu}_m)$

6: play action/super-arm S, observe triggered arm outcomes $\{X_{j,t}\}$

7: for each observed j: update $\hat{\mu}_j = (\hat{\mu}_j \cdot T_j + X_{j,t})/(T_j + 1)$; $T_j = T_j + 1$

6: end-for

Regret Bounds

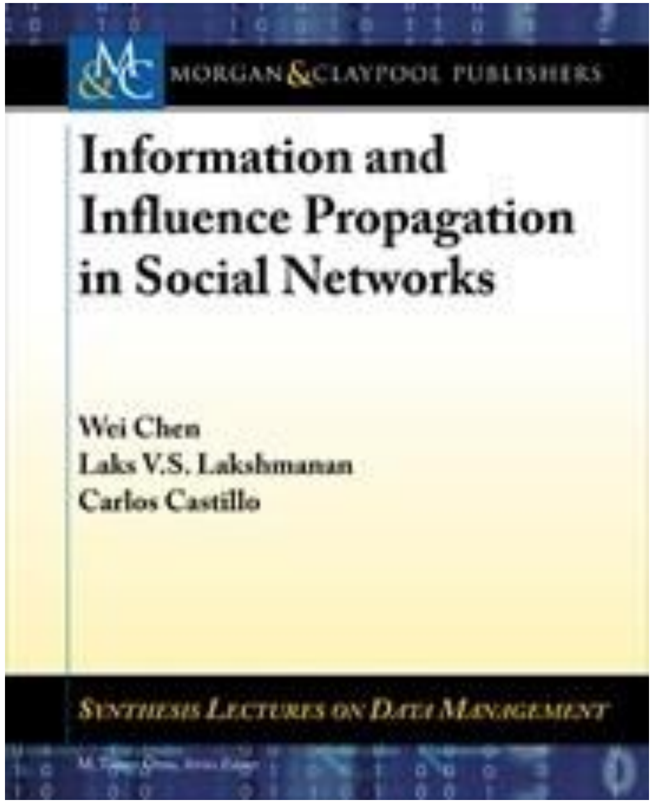
- $O\left(\sum_i \frac{1}{\Delta_{\min}^i} B_1^2 K \ln T\right)$ distribution-dependent regret
  - i: base arm index
  - B_1: one-norm bounded-smoothness constant
  - K: maximum number of arms any action can trigger
  - T: time horizon, total number of rounds
  - $\Delta_{\min}^i$: minimum gap between α fraction of the optimal reward and the reward of any action that could trigger arm i (α is the offline approximation ratio)
- $O\left(B_1\sqrt{mKT \ln T}\right)$ distribution-independent regret
- For influence maximization, B_1 is the largest number of nodes any node can reach

Conclusion and Future Work

- Influence maximization is a rich application context to study
  - connect with many classical algorithms
  - require new extensions and adaptations
  - many optimization, learning and game theoretic studies can be instantiated on the influence maximization task
- Many possible new directions, may require new algorithms and techniques
  - Non-submodular influence maximization
  - Influence maximization in dynamic networks

Reference Resources

- Search “Wei Chen Microsoft”
  - Monograph: “Information and Influence Propagation in Social Networks”, Morgan & Claypool, 2013
  - 社交网络影响力传播研究，大数据期刊，2015
  - my papers and talk slides
  - My upcoming book: 大数据网络传播模型和算法



Thanks!