Maximizing the Spread of Influence through a Social Network

Problem/Motivation:

Suppose we want to market a product or promote an idea or behavior in a society. In order to do so, we can "target" individuals, say by offering free samples of the product or explaining an idea, such as the consequences of drug abuse to teenagers (or college students). These people can then influence other people by word of mouth, and those people in turn can influence more people, and so on, creating a cascade of recommendations. The problem becomes how to find a given number of individuals who together will have the greatest influence. This is the "Influence Maximization Problem", which we shall formalize and discuss later.

Diffusion in a Social Network:

We give a simple deterministic example of diffusion in a social network. The social network here consists of two teachers A, B, and three students 1, 2, 3. The government would like to reduce the rate of drug abuse amongst teens so they want to pay one teacher to preach the negative side effects of drugs. Teacher A is not very popular so his class has only two students A and B, while teacher B is very popular so all students 1, 2, and 3 go to her class. Thus the obvious choice for the government (can they do it?) is to choose teacher B.

Diffusion Model:

In order to discuss the size of the cascade, we have to introduce a model by which the diffusion process takes place. We assume that the diffusion process is carried out in discrete steps, i.e. the diffusion process can be indexed by integer values, which we can think of as a time variable t. We also use a probabilistic model to more closely emulate real world conditions. Because of this, we must instead consider the expected size of the cascade.

Independent Cascade Model:

In the "Maximizing the Spread of Influence through a Social Network" paper, the authors considered the "Independent Cascade Model". Formally, each edge (u, v) is assigned a real value p(u,v) between 0 and 1, called the "success probability". A node that has been influenced is called "active", and it is "contagious" at time t if is active at time t but inactive at time t-1, i.e. when it just became active. When a node u is contagious, it attempts to influence all its neighbors v, each with probability p(u,v). If successful, v becomes active at time t+1. At time t+1, u is no longer contagious, and is not able to influence any more nodes. The model also assumes that an active node remains active forever.

Example 1:

We give a simple example to illustrate the diffusion process in the Independent Cascade Model. We have three nodes 1, 2, and 3. $p(1,2) = 1.0$, i.e. node 1 will always succeed in influencing node 2 when it becomes contagious. Also, $p(1,3) = 0.9$, and $p(2,3) = 0.4$. We start by targeting just node 1, so node 1 is contagious at time 0. In the diagrams, contagious nodes are encircled with a star shape.

From time 0 to time 1, node 1 influences node 2 with probability 1, i.e. node 2 will definitely be activated. Node 1 also attempts to influence node 3, with probability 0.9. So at time t = 1, there are two possibilities, either node 3 is active or not. In the case where node 1 failed to activate node 3, node 2 can try to activate node 3, with success probability of 0.4. In the other case, no more changes can happen. So there are 3 possibilities at time t = 2. Finally we let the process continue until there are no more contagious nodes, and easily calculate the expected value of the size of the cascade.

Note that the probability by which node 3 is influenced by a contagious node 2 did not change after node 1's attempt to influence it. This is the "independent" aspect of the model.

Example 2:

Here's another example of the Independent Cascade Model. We have four nodes 1 through 4, with p(1,2) = 0.5, p(2,3) = 0.2, p(2,4) = 0.7. We start by targeting node 1. If node 2 fails to be influenced, the diffusion process halts. Otherwise, node 2 becomes contagious and attempts to influence nodes 3 and 4. The result of these attempts happen between time t=1 and t=2, and manifest at t=2. Note that the outcome is well-defined because of the independence of the success probabilities, i.e. order does not matter. So we can assume without loss of generality that node 2 attempts to influence node 3 first, say at some imaginary time t=1.5, and then node 4.

Submodular functions:

We do a quick recap of submodular functions. A function f from the power set of some ground set M to R, the real numbers, is submodular if it satisfies the "diminishing returns" condition, captured by the equation (on the slide). Last week John gave an excellent introduction to unconstrained submodular functions, so half my work here is done. Note however that we are dealing with a slightly different problem here. Firstly, it is clear that we are dealing with just monotone functions, since the influence of a target set will obviously be larger if we take a superset of it. While this would make the maximization problem trivial in the unconstrained case, here we want to find the optimal target set of size k which is given. Monotone submodular functions can be approximated up to a factor of (1-1/e), about 0.632, which we will prove later on.

Also, recall that the "Maximizing Spread" paper proved that the expected size of the cascade is submodular in terms of the target set. We shall show that this is true also of the more general model called the Decreasing Cascade Model, or DCM, for brevity, but the proof requires more subtle considerations.

Decreasing Cascade Model:

This model is similar to the Independent Cascade Model, only differing in how nodes are affected by their neighbors. DCM generalizes the Independent Cascade Model by taking into account the effect of previous attempts to activate a node on its chances of being activated in the future, hence it is no longer "independent". Each node v is assigned a function p_v in two variables, the first a neighbor node of v and the second a set of neighbors of v. Then p_v(u, S) is the probability that a contagious u manages to influence v given that the set of neighbors S has

already attempt to influence v but failed. The “decreasing” aspect is the assumption that the more neighbors try to influence v, the less likely v will become active. This reflects the quality that the authors refer to as “market-saturation”. Formally, p_v(u, S) >= p_v(u, T) whenever S is a subset of T.

Example DCM:

We give a simple example of DCM. This graph has three nodes, and we are given p_3(2, {}) = 0.6, p_3(2, {1}) = 0.4 ({} is the empty set). In the right diagram, we see that node 2 is contagious and attempts to influence node 3, and node 1 is still inactive. So in this case, the probability that node 3 is influenced is 0.6. In the left diagram however, we see that node 2 is contagious, and wants to influence node 3, but node 1 has already attempted (and failed) to influence node 3. From the given p_3, we see that in this case node 3 will become active with probability 0.4. Observe that {} is a subset of {1}, and in the left diagram the success probability is lower, so we have a valid DCM.

Order-independence:

Note that by generalizing the cascade model this way, we potentially make the process not well-defined. Consider what happens when two nodes u and w become contagious at time t=0, and they are both neighbors of v. There is no implicit order by which u and w attempt to influence v, so the outcome of their attempts at t=2 may not be well-defined. Thus, we must assume order-independence of the success probabilities, which is expressed in the complicated looking expression in the slide. Each side of the equation gives the probability that v is not active after attempts by the same set of neighbors, but in different orders.

Example of DCM: Order Independence

We give an example to exhibit order independence of DCM. Our graph consists of three nodes, with nodes 1, 2 neighbors of node 3. The success probabilities are: p(1, {}) = 0.8, p(2, {}) = 0.6, p(1, {2}) = 0.7, p(2, {1}) = 0.4. We target nodes 1 and 2. If we let node 1 attempt first then node 2, we get that v is inactive at time t=1 with probability (1 – p(1, {}))*(1 – p(2, {1})) = 0.2 * 0.6 = 0.12. If node 2 before node 1 instead, probability that v remains inactive is (1 – p(2, {}))*(1 – p(1, {2})) = 0.4 * 0.3 = 0.12. These values line up, showing order independence, at least for node 3.

Main theorem:

The main result proved in the paper is that the expected size of cascade in the DCM is submodular. This result is good news since we are then guaranteed a (1-1/e)-approximation with a simple greedy algorithm: keep adding the element which gives the best marginal gain. Note that this fact alone is not enough to implement a good approximation algorithm for the Influence Maximization Problem. We must also take into consideration the time it takes to find the element with the greatest marginal gain. One way this is done is by randomly selecting potential candidates, and play out the diffusion process for several runs, and find estimate the expected size of cascade. The paper does not focus on this, and just mentions this in passing and refers to other works on this.

Example of DCM: decreasing -> submodularity

We give the intuition behind why the "decreasing" aspect of the DCM leads naturally to submodularity of expected size of cascade, denoted by σ. We consider the same graph as in the previous example. We want to compare the marginal gain in σ when node 2 is added to the target set. Consider first what happens when node 2 is added to the empty set. Simple computation shows that the marginal gain in expectation is 1.6. Now consider what happens when node 2 is added to the set {1}. Observe that when finding the marginal gain, we can focus only nodes 2 and 3, since no matter what node 1 will be active in the end.

By order independence, we may assume that node 1 attempts to activate node 3 first. We break down the analysis to two cases: whether or not node 1 succeeds. If node 1 succeeds, then the marginal gain is clearly exactly 1, since there is no gain from node 2 influencing node 3. If node 1 fails, however, then we see that the marginal gain is 1.4. The lower marginal gain is directly related to the "decreasing" assumption on the success probabilities. Combining these two results, we see that overall the marginal gain by adding node 2 to empty set is greater than adding node 2 to {1}, which is to be expected.

Old proof idea doesn't work:

In the "Maximizing Spread" paper, σ is proven to be submodular by breaking it down to simpler functions which are submodular. Each such function corresponds to a simple graph, with edges subset of those in the original graph, and the function gives the number of nodes reachable from the target set via a path in the graph, i.e. it is the size of the cascade if the diffusion process was deterministic on the associated graph. These functions are easily seen to be submodular, and σ can be written as a linear combination (with nonnegative coefficients) of these functions. The idea is that the "coin flip" to determine the success of an influencing attempt can be carried out before the process begins, and each possible outcome of coin flips we just associate the subgraph with edges where the attempt should be successful, no edge otherwise. We refer to Example 2, when t=3.

This cannot be expected to work in the DCM. An essential part of decomposing σ into simpler functions is the fact that the edges can be drawn independently. In DCM, success probabilities not only depends on the node attempting to activate v, but also the set of neighbors that have attempted to influence it. In particular, we are dealing with sets of neighbors, and so any sort of independence (besides order) is difficult to achieve. To prove that it is indeed impossible to decompose σ in this way for DCM, we suppose that σ(A) = \sum q_G * σ_G(A), G a subgraph of initial graph. The counterexample presented in the paper is simple: five nodes, u_1 to u_4 can influence v. The success probability is always p_v(u_i, S) = ½ if |S| < 2, 0 otherwise. Then plugging in possible target sets, we try to figure out what the coefficients q_G are, and eventually a contradiction is reached.

(1-1/e)-approximability of (monotone) Submodular Functions:

The intuition behind the proof is that by choosing to add the element with the largest marginal gain, our solution becomes closer to the optimal solution by 1/k, where k is the size of

optimal set we want. That is, suppose k = 5 and we currently have σ(A) = 20, and σ(OPT) = 30, where OPT is the optimal solution. Then if u is the node that maximizes marginal gain, then σ(A + {u}) – σ(A) is at least (30 – 20)/5 = 2.

We run through the outline of the proof for the case of set cover: given a collection of subsets S_i of a universe U, the problem is to find the largest possible union of k subsets. We index the subsets with 1,2,…,n, and let f:{1,2,…,n} -> {S_1, … S_n}, f(A) denote the size of the union of subsets indexed by elements of A. The greedy algorithm is given simply as follows: start with A being the empty set. At iteration, add the element which gives the largest marginal gain when added to A. Update A to include this new element. Stop when |A| = k.

Consider the first iteration. Then the first iteration is just finding the largest subset in the collection. Now the optimal solution is the union of k subsets, and so its size must not be more that k times the largest subset. Thus, in the first iteration, f({u}) – f({}) = f({u}) >= 1/k * f(OPT). Hence, our current set is 1/k closer to f(OPT) than f({}) = 0. The other iterations are similar, but make explicit use of the submodularity of f.

Granted we have proven that we always get 1/k times closer, we see that at the end of k iterations, the distance to f(OPT) is at most (1 – 1/k)^k ≈ 1/e as k large. Thus f(A) will be at least 1 – 1/e times of f(OPT).

Conclusion:

The primary contribution of the paper is proving that DCM is submodular. We note that the proof of this requires new techniques which we do not present here.

Technical aspects:

Here we discuss more technical aspects of the paper and dive into the details at more depth.

In the presentation, we talked about the simple greedy algorithm that, by submodularity and monotonicity of expected size, will give a (1-1/e)-approximation. It is true in theory, but in practice, even in the simple independent cascade model, it is not clear how to evaluate $\sigma(A)$ exactly, or whether this can be done in polynomial time, so let alone how to find the optimal node to append to the current set. However, the cascade process has the property that it can be efficiently simulated, simply by running the probabilistic rule for influence propagation until quiescence (when there are no more contagious nodes, which must clearly happen in no more than n + 1 rounds). By repeatedly simulating the cascade process and sampling the size of the cascade, we can compute arbitrarily close approximations to $\sigma(A)$.

This is reflected in the paper's more detailed statement of the main theorem: If the node added in each iteration of the greedy algorithm is a 1-ε approximate best node, then the greedy algorithm is a $(1 - 1/e - \varepsilon')$-approximation, where ε' depends on ε polynomially. By virtue of a theorem due to Nemhauser, Wolsey and Fischer, this follows almost directly from Theorem 3, which states that for the DCM, $\sigma(A)$ is a monotone and submodular function of A. The proof of $(1 - 1/e - \varepsilon')$-approximation can be attained for the 1-ε approximate greedy algorithm is easily proven by tweaking the proof.

Before discussing the proof of the main theorem, let us introduce the General Threshold Model (GTM), which was in fact discussed in "Maximizing Spread". The GTM is a diffusion model that is shown in Lemma 1 to be equivalent to the cascade model in a precise sense, but focuses on the "cumulative effect" of a node set S's influence on node v, instead of the individual attempts of nodes u in S. In the GTM, to each node v is associated a monotone 'activation function' f_v:2^V -> [0,1], and a threshold θ_v, chosen independently and uniformly at random from the interval (0,1]. A node v becomes active at time t+1 if f_v(S) >= θ_v, where S is the set of nodes active at time t. Thus, when the amount of influence on node v has exceeded a critical point, v becomes active. Note that the set S does not have to be a subset of neighbors of v, and we can indeed assume the same for the DCM by keeping the success probabilities the same when set of active nodes S in p_v(u, S) is extended to include non-neighbors of v. As with DCM, the diffusion process starts with the activation of a select set A at time 0 (hmm seem to have an off by one discrepancy here…).

DCM and GTM are equivalent in the following precise sense: for any instance of DCM, we can find an instance of GTM (i.e. activation functions) such that, starting from the same target set, the distribution of the set of active nodes at quiescence is the same as that given by the DCM, and vice versa. This is captured in Lemma 1, where (natural) explicit formulas to convert between activation function and success probabilities are proved to give this equivalence. The proof is just induction on a slightly stronger but natural hypothesis.

Observe that since at each round, each node's likelihood of becoming active is dependent on the entire set of active nodes and not individual nodes activating it one-by-one, it is clear that

we may make all random choices at time 0 before the process starts. This is similar to flipping coins to decide whether or not to include an edge at the start in the Independent Cascade Model. Note that this cannot in general be done for the DCM, since it once the outcomes have been decided beforehand, order-independence may be violated. Reusing the "Example of DCM: Order Independence" graph from above, suppose before we began we decided that node 1 will never activate node 3, node 2 will fail to activate node 3 if node 1 has already attempted and failed, and node 2 will succeed if node 1 has not attempted. Then if we allow node 1 to attempt first and then node 2, we find that node 3 will not become active; however if node 2 goes first, then node 3 will become active. This contradicts order-independence.

In particular, by switching views from DCM to GTM, order-independence becomes more natural and easier to deal with. Using the GTM view, we can then prove a strong generalization of order-independence. Namely, each node is associated a finite 'waiting time' τ_v, meaning that when v's criterion for activation has been met just before time t (i.e. should become active at time t), v only becomes active at time $t + \tau_v$. Lemma 2 states that the distribution of the set of active nodes at quiescence is independent of choice of τ_v. This is a very natural thing to want from our model, at least in order to prove submodularity: it makes analysis of the marginal gain simpler by thinking of the additional node not as a new element in the initial target set, but instead as a new node we target at time n+1, when the process is guaranteed to have reached quiescence. Thus we can think of the difference in cascade size coming just from one new node on a distribution of graphs. This is the "breaking down" step in the proof of the main theorem, similar to splitting $\sigma(A)$ into simpler functions in "Maximizing Spread". The proof of Lemma 2 boils down to observing that if node v is meant to be activated in the case without waiting time, that means at some point v should have enough 'incentive' to become active, regardless of waiting time.

Now we are ready to run through the proof of the main theorem, specifically submodularity of $\sigma(A)$. The basic idea of the proof is mentioned above: we want to characterize $\sigma(A + \{w\}) - \sigma(A)$ in terms of a 'residual process' in which only node w is targeted, and has appropriately modified success probabilities (the '+' stands in for union U).

Given a node set B, we define the 'residual process' on the set $V \setminus B$: the success probabilities are $p_v^{(B)}(u, S) := p_v(u, S + B)$, and the only node initially targeted is w. Let $\varphi(A)$ denote the random variable giving the distribution of the set of active nodes at quiescence in the original model, and similarly φ_B for the residual process on $V\setminus B$. It is not hard to see that, by Lemma 2, given that, the distribution of $\varphi_B(w)$ is the same as the distribution of $\varphi(A + \{w\})\setminus\varphi(A)$. Indeed, $\varphi_B(w)$ is naturally chosen so that these will coincide.

Next we want to compare the expected sizes $\varphi_B(w)$ of $\varphi_{B'}(w)$, when B subset of B'. Let $\sigma_B(w) = E[|\varphi_B(w)|]$ be the expected size of $\varphi_B(w)$, and likewise $\sigma_{B'}(w) = E[|\varphi_{B'}(w)|]$. Observe that the residual process on $V\setminus B$ has more nodes than that of $V\setminus B'$. Also, by the decreasing condition of DCM, $p_v^{(B)}(u, S) = p_v(u, S + B) >= p_v(u, S + B') = p_v^{(B')}(u, S)$. Hence, by the combination of larger ground set of nodes and larger success probabilities, it should be clear that $\sigma_B(w) >= \sigma_{B'}(w)$ (Lemma 3 proves this formally).

Finally, the proof is finished by a string of equalities and one inequality, comparing σ(A + {w}) - σ(A) with σ(A' + {w}) - σ(A'), where A subset of A'. Essentially, we add inequalities σ_B(w) >= σ_B'(w) over all pairs B subset of B', with weights Prob[φ(A) = B, φ(A') = B']. Note that on the left hand side, fixing B and letting B' range over all possibilities, the sum collapses to sum of σ_B(w)*Prob[φ(A) = B] over all B. This is just σ(A + {w}) - σ(A) as we have seen above. Similarly on right hand side, we first fix B', getting σ_B'(w)*Prob[φ(A') = B'] over all B', and this is just σ(A' + {w}) - σ(A'), and we are done.

Now let us give the full proof that the greedy algorithm gives a (1-1/e)-approximation. Let f be a monotone, submodular, normalized function on power set of universe U (normalized means f({}) = 0), and we want to find subset OPT of size k which maximizes f. The algorithm begins with S_0 = {}, and at each iteration i from i=1 to i=k, let u_i be the element of U that gives greatest marginal gain when added to S_(i-1), and let S_i = S_(i-1) + {u_i}. Suppose OPT = {t_1, t_2, … t_k}, T_j = {t_1, … t_j}.

By choice of u_i and submodularity, f(S_(i+1)) – f(S_i) >= f(S_i + {t_j}) – f(S_i) >= f(S_i + T_(j+1)) – f(S_i + T_j), so adding these inequalities together, they telescope, giving f(S_(i+1)) - f(S_i) >= (f(S_i + OPT) – f(S_i))/k >= (f(OPT) – f(S_i))/k. So the marginal gain in iteration i increases f to make f(S_i) at least 1/k times closer to to f(OPT). Finally, after k iterations, f(S_k) is at least (1-1/k)^k times closer to f(OPT) than when we began (which was 0), so f(S_k) >= (1 – (1-1/k)^k)*f(OPT) ≈ (1-1/e)*f(OPT).