How to Schedule a Cascade in an Arbitrary Graph

Flavio Chierichetti, Cornell University
Jon Kleinberg, Cornell University
Alessandro Panconesi, Sapienza University

When individuals in a social network make decisions that depend on what others have done earlier, there is the potential for a *cascade* to form — a run of behaviors that are highly correlated. In an arbitrary network, the outcome of such a cascade can depend sensitively on the order in which nodes make their decisions, but to do date there has been very little investigation of how this dependence works, or how to choose an order to optimize various parameters of the cascade.

Here we formulate the problem of ordering the nodes in a cascade to maximize the expected number of "favorable" decisions — those that support a given option. We provide an algorithm that ensures an expected linear number of favorable decisions in any graph, and we show that the performance bounds for our algorithm are essentially the best achievable assuming P $\neq$ NP.

1. INTRODUCTION

When people in a social network are influenced by each other's decisions, one can see long runs of behavior in which people's choices become highly correlated, resulting in a *cascade* of decisions. A striking aspect of such phenomena is the fact that the population's preferences might be quite heterogeneous at an individual level, but this heterogeneity is washed away by a global pattern of behavior in which most people arrive at the same observable decision. For example, it often happens that two competing products A and B each perform well in preliminary market research, but one quickly captures most of the market once they are both introduced.

Cascades of this type are sensitive to the order in which people make decisions; when a group of people are choosing sequentially from among a set of options, the consequences of certain early decisions can be magnified through the effects they produce on the rest of the population [Arthur 1989; Banerjee 1992; Bikhchandani et al. 1992; Salganik et al. 2006]. Thus, if we imagine the problem faced by a planner who is trying to promote a certain option within a population — for example, to manage the marketing of a new product — there is a complex scheduling problem inherent in the question of how to bring this product to people's attention over time. However, despite considerable research on the properties of cascades, as well as approaches to "seeding" cascades with initial adopters [Domingos and Richardson 2001; Kempe et al.

Flavio Chierichetti was supported in part by the NSF grant CCF-0910940. Jon Kleinberg was supported in part by a John D. and Catherine T. MacArthur Foundation Fellowship, a Google Research Grant, a Yahoo! Research Alliance Grant, and NSF grants IIS-0705774, IIS-0910664, and CCF-0910940. Alessandro Panconesi was supported in part by a Google Research Grant.

Author's addresses: F. Chierichetti, Department of Computer Science, Cornell University, Ithaca, NY 14853; J. Kleinberg, Department of Computer Science, Cornell University, Ithaca, NY 14853; A. Panconesi, Dipartimento di Informatica, Sapienza University, Rome, Italy.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

2003; Richardson and Domingos 2002] and offering time-varying incentives [Hartline et al. 2008], very little is known about this scheduling aspect of the problem.

In this paper, we provide an algorithm for this type of scheduling problem in an arbitrary graph representing an underlying social network. We work within a model adapted from a widely-used framework in the economic theory literature; in what follows, we first describe the model, and then our results.

A model: Sequential decisions with positive externalities. In an influential paper in the economic theory literature, Brian Arthur proposed a simple model for how cascades form [Arthur 1989]. He based his model on a scenario with two competing products $\mathcal{Y}$ and $\mathcal{N}$ (mnemonic shorthand for "yes" and "no") that each exhibit *positive externalities* — they become more valuable as more people use them.[1]

In the model, there are two types of consumers — those with a preference for $\mathcal{Y}$, and those with a preference for $\mathcal{N}$. (We will refer to these two types of consumers as *Y-types* and *N-types*, to avoid confusion between decisions and preferences.) We assume that when there are no other users of a product, each type gets a payoff of π_1 from using her preferred product, and a payoff of π_0 from using her non-preferred product, with $\pi_1 > \pi_0$. The positive externalities are manifested by a term that adds $\delta > 0$ to the payoff of a product for each user it has; thus, if there are currently $m_\mathcal{Y}$ users of $\mathcal{Y}$ and $m_\mathcal{N}$ users of $\mathcal{N}$, then the payoff of a Y-type for product $\mathcal{Y}$ is $\pi_1 + \delta m_\mathcal{Y}$, and the payoff of a Y-type for product $\mathcal{N}$ is $\pi_0 + \delta m_\mathcal{N}$. Analogous payoffs hold for an N-type: $\pi_1 + \delta m_\mathcal{N}$ for product $\mathcal{N}$ and $\pi_0 + \delta m_\mathcal{Y}$ for product $\mathcal{Y}$.

Given this, if a consumer must decide which product to purchase based on the current number of users of each product, then the decision rule is very simple: a Y-type compares $\pi_1 + \delta m_\mathcal{Y}$ to $\pi_0 + \delta m_\mathcal{N}$, while an N-type compares $\pi_1 + \delta m_\mathcal{N}$ to $\pi_0 + \delta m_\mathcal{Y}$.[2] In both cases, if we define $c = \lceil \delta^{-1} |\pi_1 - \pi_0| \rceil$, the rule is equivalent to the following:

> *($*$) When $|m_\mathcal{Y} - m_\mathcal{N}| \geq c$ the consumer should choose the product with more users; when $|m_\mathcal{Y} - m_\mathcal{N}| < c$, the consumer should choose based on her own preference.*

We will call c the *decision parameter* in this rule.

Now, suppose that consumers make purchasing decisions one at a time, starting from $m_\mathcal{Y} = m_\mathcal{N} = 0$, and each new consumer who arrives is a Y-type with probability $p > 0$ and an N-type with probability $1-p > 0$. The key point of the model is that if consumers can observe the current values of $m_\mathcal{Y}$ and $m_\mathcal{N}$ as they make their decisions, then as soon as one product has c users more than the other, it becomes "locked in" — all future decisions will be in favor of the majority product. And since the decisions before this moment will favor each consumer's preferred product, the quantity $m_\mathcal{Y} - m_\mathcal{N}$ over time can be analyzed quite simply: it is performing a random walk in which it changes by $+1$ each step with probability p and by -1 with probability $1 - p$. Once this walk reaches the value c or $-c$, one of the products has become locked in. If (by symmetry) we assume that $p < 1/2$, it is not hard to show that the probability product Y is the one that becomes locked in is $\Theta(p^c n)$.

Arthur's model thus captures, in a very simple fashion, the way in which cascades can lead to highly correlated behavior, and can be sensitive to the outcomes of a few early decisions. Subsequent models by Banerjee [Banerjee 1992] and Bikhchandani, Hirshleifer, and Welch [Bikhchandani et al. 1992], also very influential in economic

[1] Examples of such products include cell phones, social media sites, and computer operating systems, since in all these cases they tend to acquire better service and more third-party applications as their audiences grow.

[2] We can assume that in the case of an exact tie in payoffs, a consumer uses a canonical tie-breaking rule such as following the majority.

theory, have shown how similar phenomena can arise when people are influenced by earlier decisions not because of positive externalities, but because these earlier decisions convey information that they can use in their present decision.

Given the generality of the phenomenon, in what follows we abstract away from the language of products and purchases, and instead cast the discussion more broadly simply in terms of *decisions* and *types* (or *signals*): each person chooses $\mathcal{Y}$ (which we also refer to as "yes") or $\mathcal{N}$ (also referred to as "no"), and is of type Y or N (or, equivalently, gets a *positive* or *negative* signal).

The problem: Scheduling a cascade in a graph. This model can be directly generalized to operate on an arbitrary graph $G = (V, E)$, representing a social network on the individuals. As before, each individual i is a Y-type (that is, individual i gets a *positive* signal) with probability $p > 0$ and an N-type (the individual gets a *negative* signal) with probability $1 - p > 0$, and i has the same payoffs as before, except that $m_{\mathcal{Y}}$ and $m_{\mathcal{N}}$ now denote the number of neighbors of i in G who have already chosen $\mathcal{Y}$ and $\mathcal{N}$ respectively. This reflects the fact that i only derives benefit when a neighbor in G makes the same decision that i does. With this new definition of $m_{\mathcal{Y}}$ and $m_{\mathcal{N}}$, the decision rule governing i's decision is the same as before. Note that Arthur's original model corresponds precisely to this generalized model in the case when G is a complete graph.

We now return to the scheduling question raised at the outset. Suppose that people choose between $\mathcal{Y}$ and $\mathcal{N}$ using rule $(*)$ with some decision parameter c. Suppose further that we are interested in helping to maximize the number of people who choose $\mathcal{Y}$. While the relative attractiveness of $\mathcal{Y}$ and $\mathcal{N}$ to any one individual is fixed by rule $(*)$, we have the ability to affect the *order* in which the cascade progresses — that is, the order in which individuals will make their decisions. Can we do this in a way that guarantees a reasonably large expected number $\mathcal{Y}$ decisions?

There are several points to note about this type of question. First, this scheduling problem will turn out to be trivial in complete graphs, but as we will see, in general graphs it becomes much more complex. Second, the question is interesting both when $p \geq 1/2$ and when $p < 1/2$ — that is, both when Y is the majority preference and when Y is the minority preference — and we will consider both of these; but our main focus will be on the case when $p < 1/2$, since this is the more challenging situation when we are trying to promote an option that is inherently favored by a smaller fraction of people.

Finally, it is also important to note that Arthur's model of cascades, which we are following here, is distinct in some fundamental and important ways from the notion of threshold contagion that has been studied in a number of recent papers (e.g. [Blume et al. 2011; Centola and Macy 2007; Dodds and Watts 2004; Kempe et al. 2003; Mossel and Roch 2007; Watts 2002]). In threshold contagion, an individual decides whether to change to a state (say, $\mathcal{Y}$) based only on the number of neighbors who have previously chosen $\mathcal{Y}$. This makes the outcome order-independent, and hence there is no scheduling issue. In Arthur's cascade model, on the other hand, nodes base their decision on the number of neighbors who have previously chosen $\mathcal{Y}$ *and* the number of neighbors who have previously chosen $\mathcal{N}$. As we will see, in general graphs this makes the outcome sensitively dependent on the order in which the decisions are made, and hence leads naturally to questions of scheduling.

Thus, concretely, the problem we consider is as follows. We are given a graph G and the (constant) value of c used as the decision parameter in rule $(*)$ above; at the outset, we do not know the types of the nodes. One at a time, we select a node from G, reveal its type to be a Y-type or an N-type (with probability p and $1 - p$ respectively), and then have it make its decision according to rule $(*)$. The selection of the next node can depend on the outcome of the decisions made by earlier nodes. Our overall goal is to have a large expected number of nodes choose $\mathcal{Y}$; note here that (even for deterministic selection rules) the expectation is computed over the random exposure of a type (Y or N) at each node as the selection proceeds.

Here is a basic question about the attainability of this goal. Let $\mathcal{G} = \{G_1, G_2, G_3, \ldots\}$ be a family of graphs, with $n_i = |V(G_i)|$ denoting the number of nodes in G_i. We say that the set $\mathcal{G}$ *achieves constant adoption* with decision parameter c if it is possible to achieve a linear expected number of nodes choosing $\mathcal{Y}$ for all graphs in the family — that is, if there exists a function $f_c(p)$ such that for all graphs G_i in the family and all $p > 0$, there is a way of selecting the nodes of G_i for which the expected number of decisions equal to $\mathcal{Y}$ is at least $f_c(p) \cdot n_i$.

A natural question is then the following: which families of graphs achieve constant adoption? We note first that the discussion of Arthur's model above shows that the family of complete graphs achieves constant adoption with $f_c(p) = p^c$. In a different direction, it is not hard to see that the family of empty graphs (those with no edges) achieves constant adoption with $f_c(p) = p$, since in such a graph each node is simply following its own preference. Of course, the scheduling problem in both of these types of graphs is trivial, since all orderings yield the same expected value. But one intriguing point about even this very simple pair of examples — cliques and empty graphs — is that it shows how different families of graphs can achieve constant adoption for very different reasons. In the former case, the tight dependence among all nodes in the complete graph results in a constant probability that all but a constant number of nodes will choose $\mathcal{Y}$; in the latter case, the complete independence among all nodes in the empty graph ensures that a p fraction of them will choose $\mathcal{Y}$ in expectation. For more complex graphs, it is not a priori clear whether the cascade dynamics can exhibit properties qualitatively different from either of these.

Our Results. Our first main result is that the family of all graphs achieves constant adoption. Specifically, we show that for any n-node graph G, there is a way of ordering nodes so that the expected number of nodes choosing $\mathcal{Y}$ is at least $\Theta(p^c n)$. Thus, the family of cliques is in fact the worst case for ensuring many decisions equal to $\mathcal{Y}$.

We prove this result through an explicit method for ordering the nodes, which roughly speaking combines ideas from the analysis of the complete graph and the empty graph. In particular, we first identify a maximal set S of nodes that is sufficiently sparse that decisions within S can be made independently; we then try to extend S by adding nodes whose decisions have been forced by the decisions made in S.

We also show that there are some graphs in which it is possible to do much better than is possible in either the complete graph or the empty graph: there exist n-node graphs for which one can order the nodes to achieve an expected number of $\mathcal{Y}$ decisions that is $n - O(1/p)$, and this is the best possible for any graph. Moreover, it is computationally hard to distinguish among these extremes: for any integer $c \geq 1$, we show that it is NP-hard to distinguish between graphs for which the maximum expected number of nodes choosing $\mathcal{Y}$ is at least $n - o(n)$ and those for which it is at most $p^{c-\epsilon} n$.

Thus far our model for ordering has been *adaptive* — when we go to choose the next node in the ordering, we are able to see the types and decisions of all earlier nodes. It is also interesting to consider a *non-adaptive* version of the problem; here the requirement is that we choose an ordering of the full node set in advance, before seeing the types or decisions of any nodes.

For this non-adaptive version of the problem, we show that when $p \leq \frac{1}{2}$, it is not possible to get $n - o(n)$ nodes in expectation to choose $\mathcal{Y}$. We also show that if $p \geq \frac{1}{2}$, then any ordering guarantees that the expected number of nodes choosing $\mathcal{Y}$ is at least

$\frac{n}{2}$ — so any ordering is a 2-approximation if $p \geq \frac{1}{2}$. Finally, we prove that for small p, it is NP-hard to distinguish between the case where the maximum expected number of nodes choosing $\mathcal{Y}$ is at least $p^{1+\epsilon}n$ and the case where it is at most $p^{c-\epsilon}n$.

2. AN ALGORITHM

In this section we develop and analyze an algorithm that allows us to obtain an expected number of $\mathcal{Y}$'s that is at least $p^c \cdot n$ in any graph. The algorithm makes use of the notions of *t-degenerate induced subgraphs* and ***Erdös-Hajnal sequences***, so we begin by defining these here.

Given a graph G, the set of nodes $S \subseteq V(G)$ induces a *t-degenerate graph* $G[S]$ if there exists an ordering $v_1, \ldots, v_{|S|}$ of the nodes in S such that, for each $i = 1, \ldots, |S|$, the degree of v_i in $G\left[\{v_1, \ldots, v_{i-1}\}\right]$ is at most t. Such an ordering of the nodes in S is called a ***Erdös-Hajnal sequence*** for the t-degenerate graph $G[S]$, and can be easily computed in polynomial time.

Finally, an induced subgraph $G[S]$ is *maximal t-degenerate* if it is t-degenerate, and for every node $v \in V(G) - S$ it holds that $G[S \cup \{v\}]$ is not t-degenerate.

We are now ready to introduce Algorithm 1. If a node v has already made its choice at some point in the process, we say that it has been "activated"; otherwise that it is "unactivated". Algorithm 1 starts by selecting a maximal induced subgraph S with the informal properties that (i) any node in the subgraph will make its choice somewhat independently from other nodes in the subgraph, (ii) every node outside the subgraph will get a chance of being forced to choose $\mathcal{Y}$ with probability at least p^c. Then, it schedules the maximal induced subgraph S interleaved with the nodes in $V(G) - S$, whenever they get a chance of being forced to choose $\mathcal{Y}$.

ALGORITHM 1: A scheduling algorithm.

```
Let $W \subseteq V(G)$ be any set inducing a maximal $(c-1)$-degenerate graph in $G$
Let $v_1, v_2, \ldots, v_{|W|}$ be a Erdös-Hajnal sequence of the nodes in $W$
for $i = 1, \ldots, |W|$ do
    Schedule $v_i$
    while there exists an unactivated node $v \in V(G) - W$ with exactly $c$ activated neighbors
    in $W$, all of which have chosen $\mathcal{Y}$ do
        Schedule $v$
Schedule the remaining nodes arbitrarily
```

The following theorem characterizes the performance of Algorithm 1.

THEOREM 2.1. *Let G be any graph on n nodes, let p be the probability that a node receives a positive signal, and let $c \geq 1$ be the threshold. Then, the random variable X representing the number of nodes having chosen $\mathcal{Y}$ with the scheduling produced by Algorithm 1, is such that $E[X] \geq p^c \cdot n$.*

We will prove Theorem 2.1, after having established a series of lemmas.

The first lemma claims that every node in $V(G) - W$ will have exactly c activated neighbors in W at some point in the execution.

LEMMA 2.2. *For every node $v \in V(G) - W$, there will be (at least) one iteration $k(v)$ in which v will have exactly c neighbors in W that have been scheduled.*

PROOF. By fixing a maximal $(c-1)$-degenerate set W, we have that every node $v \in V(G) - W$ will be connected to at least c nodes in W (for otherwise, we could add

v to W while still keeping $W \cup \{v\}$ has a $(c-1)$-degenerate graph, and therefore W would not be maximal.)

Given W, $v \in V(G) - W$, and the ordering $v_1, \dots, v_{|W|}$ of the nodes in W, let $k = k(v)$ be the smallest integer such that v has at least c neighbors in the prefix $v_1, v_2, \dots, v_k$. After having scheduled v_k, node v will have exactly c activated neighbors in W. □

The next lemma shows how nodes in $V(G) - W$ will not choose $\mathcal{N}$ until the end of the algorithm.

LEMMA 2.3. *If $v \in V(G) - W$, either node v will choose $\mathcal{Y}$ at iteration $k(v)$, or it will remain unactivated until line 7.*

PROOF. By induction, assume that $\ell \geq 0$ nodes in $V(G) - W$ have been scheduled. By induction, each of them chose $\mathcal{Y}$. By the test at line 5, if a new node v is scheduled at line 6, its only c activated neighbors $w_1, w_2, \dots, w_c$ in W have chosen $\mathcal{Y}$ — and by the induction hypothesis all its actived neighbors (if any) in $V(G) - W$ have also chosen $\mathcal{Y}$. Therefore v will choose $\mathcal{Y}$. If, on the other hand, one of $w_1, w_2, \dots, w_c$ chose $\mathcal{N}$, then v will not be scheduled until line 7 is reached. □

We then prove that each node in W will choose $\mathcal{Y}$ if its random signal is positive.

LEMMA 2.4. *When a node $v \in W$ is activated, it will choose $\mathcal{Y}$ if its random signal is positive; therefore it will choose $\mathcal{Y}$ with probability at least p.*

PROOF. Indeed, before reaching line 7, every activated node in $V(G) - W$ will choose $\mathcal{Y}$. Thanks to our choice of the ordering of W's nodes, when we activate a node $v \in W$, there will be at most $c-1$ of its neighbors in W that have already been activated. Therefore, either v will be forced to choose $\mathcal{Y}$, or its choice will be equal to its random signal. □

We now prove a lower bound on the probability that a node in $V(G) - W$ will choose $\mathcal{Y}$.

LEMMA 2.5. *Every node $v \in V(G) - W$ will choose $\mathcal{Y}$ with probability at least p^c.*

PROOF. At iteration $k(v)$, when v has exactly c active neighbors $w_1, w_2, \dots, w_c$ in W, we execute v iff each of the w_i's chose $\mathcal{Y}$. Since, for $i = 1, 2, \dots, c$, w_i will choose $\mathcal{Y}$ if its signal is positive, and since w_i's signal is independent of other signals (and positive with probability p), we have that $w_1, w_2, \dots, w_c$ will all choose $\mathcal{Y}$, and therefore v will choose $\mathcal{Y}$, with probability at least p^c. □

We can finally prove Theorem 2.1.

PROOF PROOF OF THM. 2.1. Since every node v of G is either part of W or $V(G) - W$, we have that the expected value of the random variable indicating a yes-choice of v is at least p^c. Linearity of expectation then gives the claim. □

We observe how the previous proof actually entails the following stronger corollary:

COROLLARY 2.6. *Let W^* be a maximum $(c-1)$-degenerate induced subgraph of G. Then, there exists a scheduling that guarantees an expected number of $\mathcal{Y}$'s of value at least*

$$p\,|W^*| + p^c\,|V(G) - W^*|\,.$$

It is now useful to recall that the size of the $(c-1)$-degenerate subgraph is lower bounded by the size of the maximum independent set, and the size of the maximum independent set is lower bounded by the size of the maximum $(c-1)$-degenerate subgraph divided by c. Therefore, since c is a constant, Corollary 2.6 claims that it is always possible to get a scheduling of value $\Theta\left(p \cdot \alpha(G) + p^c \cdot (n - \alpha(G))\right)$, where $\alpha(G)$ is the size of the maximum independent set of G.

We now give an easy example showing that Corollary 2.6 is tight:

OBSERVATION 2.1. *For any $n \geq c$, and any $c \leq w \leq n$, there exists a graph of n nodes whose maximum $(c-1)$-degenerate induced subgraph has size w, whose maximum expected number of $\mathcal{Y}$'s has value*

$$p \cdot w + \Theta\left(p^c \cdot (n - w)\right).$$

PROOF. Take an independent set on $w - c$ nodes and a disjoint clique on $n - w + c$ nodes. Clearly the maximum expected number of $\mathcal{Y}$'s is the sum of the maximum expected number of $\mathcal{Y}$'s for a scheduling of the clique plus the maximum expected number of $\mathcal{Y}$'s for a scheduling of the independent set.

Any scheduling for the independent set, and for the clique, has the same expectation. For the independent set it is p times its size. For the clique it is $\Theta(p^c)$ times its size — this is a consequence of the unfair gambler's ruin problem. Suppose that two gamblers, starting with c coins each, play a sequence of games in such a way that player 1 will win any fixed game independently with probability $p < \frac{1}{2}$, and player 2 will win that same game with probability $1 - p$. After a game is played, the loser gives one coin to the winner. The process ends when one player goes bankrupt. It is known (see, for instance, exercise 7.20 of [Mitzenmacher and Upfal 2005]) that the probability that player 1 will end up with $2c$ coins (and therefore, the probability that player 2 will go bankrupt) is

$$\frac{\left(\frac{1-p}{p}\right)^c - 1}{\left(\frac{1-p}{p}\right)^{2c} - 1} = \Theta\left(p^c\right).$$

Since the expected number of steps needed to end the process is a constant, if c is a constant, the claim that the expected number of $\mathcal{Y}$'s in any scheduling of a clique of size $n - w$ is $\Theta\left(p^c \cdot (n - w)\right)$ is proved — therefore the main claim is also proved. □

3. MAXIMUM NUMBER OF $\mathcal{Y}$'S

We just saw that, on every graph on n nodes, one can find a scheduling guaranteeing at least $p^c \cdot n$ $\mathcal{Y}$'s on expectation, and that this lower bound is tight. The next question we answer is: what is the largest possible number of $\mathcal{Y}$'s ?

LEMMA 3.1. *There exists a graph G on n nodes that, for any $p \geq \Omega\left(\frac{\log n}{\sqrt[c]{n}}\right)$ guarantees an expected number of $\mathcal{Y}$'s , for the adaptive scheduling with threshold c, of at least $n - O(1/p)$.*

PROOF. Let n be $n = c\binom{t}{c} + t + c$, for a positive integer t. Create a clique A on $c\binom{t}{c} + c$ nodes: c nodes $x_1, x_2, \ldots, x_c$ and one node $v^j_{\{i_1,\ldots,i_c\}}$ for each $j = 1, \ldots, c$, and for each $\{i_1, \ldots, i_c\} \in \binom{[t]}{c}$.

Also, create a disjoint independent set B on t nodes $w_1, \ldots, w_t$. Connect each node w_i to each node v^j_A such that $i \in A$. We choose $p \geq \Omega\left(\frac{\ln n}{t}\right)$. Observe that $t = \Theta\left(\sqrt[c]{n}\right)$.

Now we describe the scheduling. Until we get c distinct $\mathcal{Y}$ choices, activate in order the nodes $w_1, w_2, \ldots$. Suppose we get c choices of type $\mathcal{Y}$ at nodes $w_{i_1}, w_{i_2}, \ldots, w_{i_c}$. Observe that those c nodes form a side of a complete bipartite graph whose other side is composed by the nodes $X = \left\{w^j_{i_1,i_2,\ldots,i_c}\right\}_{j=1}^{c}$. We execute the nodes in X in any order. Since each of them has at least c neighbors that have chosen $\mathcal{Y}$ (the $w_{i_1}, \ldots, w_{i_c}$ nodes) and zero $\mathcal{N}$ neighbors, they will all choose $\mathcal{Y}$. We then schedule the nodes $x_1, x_2, \ldots, x_c$, that are not connected to any node w_i. They will all choose $\mathcal{Y}$. We then schedule the remainder of the clique — since every remaining node in the clique has at least $2c$ neighbors that have chosen $\mathcal{Y}$, and at most c neighbors that have chosen $\mathcal{N}$, they will all choose $\mathcal{Y}$. Finally, we schedule every remaining w_i node — each of them is only connected to c different $\mathcal{Y}$ nodes in the clique. Therefore they will all choose $\mathcal{Y}$.

The above scheduling has the property that after we obtain at least c different $\mathcal{Y}$'s every remaining node will choose $\mathcal{Y}$. Furthermore each w_i node activated before we get the $(c+1)$st choice of type $\mathcal{Y}$, is activated independently from the others. Therefore the expected number of $\mathcal{N}$'s is at most

$$c \cdot \frac{1-p}{p} + n \cdot \sum_{i=0}^{c-1} \left(\binom{t}{i} \cdot p^i \cdot (1-p)^{t-i} \right) \leq \frac{c}{p} + n \cdot (1-p)^{t-c} \cdot \sum_{i=0}^{c-1} (tp)^i = O\left(\frac{1}{p}\right)$$

Therefore the expected number of $\mathcal{Y}$'s is at least $n - O\left(\frac{1}{p}\right)$. □

Again, we prove tightness of the above bound.

OBSERVATION 3.1. *For every graph G on n nodes, and for every $p \geq \Omega\left(\frac{\log n}{n}\right)$, the smallest expected number of $\mathcal{N}$'s is at least $\Omega\left(\frac{1}{p}\right)$.*

PROOF. The expected number of $\mathcal{N}$'s , with any scheduling, and on any graph is at least the expected number of $\mathcal{N}$'s until a positive signal is received, that is:

$$\sum_{i=0}^{n} \left(i \cdot (1-p)^i \cdot p\right) = \frac{1-p}{p} - (1-p)^{n+1} \cdot \left(n + \frac{1}{p}\right).$$

□

4. APPROXIMATING THE MAXIMUM

Since the maximum number of $\mathcal{Y}$'s is at most n, Algorithm 1 gives a p^c approximation to the problem of finding the scheduling that maximizes the (expected) number of $\mathcal{Y}$'s . In this section we show that a better approximation is not achievable if P $\neq$ NP.

We start with a lemma that will turn out to be a crucial ingredient in the hardness of approximation proof.

LEMMA 4.1. *In the adaptive case, if G is a graph of order n, with maximum independent set of size α, and if $\alpha \cdot p \geq \Omega\left(\frac{1}{\sqrt[c]{n}}\right)$, then the maximum expected number of $\mathcal{Y}$'s is at most $O(n \cdot \alpha^c \cdot p^c)$.*

PROOF. If $p > \frac{1}{2(3c-2)\alpha}$, there is nothing to prove. We assume the contrary.

We will base our analysis on the following consideration: if at most $c-1$ nodes chose $\mathcal{Y}$ during the process, then if we activate a node v connected to $3c-2$ or more already activated nodes, it will necessarily choose $\mathcal{N}$ since at least $2c-1$ of the $3c-2$ activated neighbors will have chosen $\mathcal{N}$ themselves, and at most $c-1$ of them will have chosen $\mathcal{Y}$. Therefore in the neighborhood of v there will be at least c more $\mathcal{N}$'s than $\mathcal{Y}$'s .

Take any scheduling, and let S be the set of nodes that, at the moment they are activated, are connected to at most $3c-3$ already-activated nodes.

If we induce G on S, we obtain a graph $G[S]$ that is $(3c-2)$-colorable: indeed, a $(3c-2)$-coloring can be obtained by coloring the nodes of S greedily in the order dictated by the scheduling. Since any independent set in $G[S]$ is also an independent set in G, we have that the independence number of $G[S]$ is at most α — since $G[S]$ is $(3c-2)$-colorable, this implies that the cardinality of S is at most $|S| \leq (3c-2)\cdot\alpha$.

Suppose that at most $c-1$ nodes in S get a positive signal. Then, the number of nodes in S that choose $\mathcal{Y}$ is at most $c-1$. Furthermore, every node outside S will choose deterministically $\mathcal{N}$. Therefore, the total number of $\mathcal{Y}$'s will be at most $c-1$.

We compute the probability that at least c nodes in S will have a positive signal:

$$\sum_{i=c}^{|S|}\left(\binom{|S|}{i}\cdot p^i\cdot(1-p)^{|S|-i}\right) \leq \sum_{i=c}^{|S|}\left(|S|\cdot p\right)^i \leq \sum_{i=c}^{\infty}\left((3c-2)\cdot\alpha\cdot p\right)^i =$$
$$\frac{\left((3c-2)\alpha p\right)^c}{1-(3c-2)\alpha p} \leq 2\cdot\left((3c-2)\alpha p\right)^c = 2\cdot(3c-2)^c\cdot\alpha^c\cdot p^c.$$

In general, the number of $\mathcal{Y}$'s we can get is at most n. It follows that the expected number of $\mathcal{Y}$'s is at most $n\cdot O\left(c^c\alpha^c p^c\right)+c-1 = O\left(n\cdot\alpha^c\cdot p^c\right)$, where the last step follows from $c=O(1)$ and $\alpha p \geq \Omega\left(n^{-\frac{1}{c}}\right)$. □

We are now ready to prove our hardness result. As already mentioned it makes use of Lemma 4.1, and of ideas developed in the proof of Lemma 3.1.

THEOREM 4.2. *In the adaptive case, it is NP-hard to distinguish between graphs G for which the maximum number of $\mathcal{Y}$'s is at least $n-o(n)$ and graphs for which the maximum number of $\mathcal{Y}$'s is at most $n\cdot p^{c-\epsilon}$, for p equal to an inverse polynomial in n.*

PROOF. We reduce from an independent set instance H on $k=|V(H)|$ nodes. Zuckerman [Zuckerman 2007] proved that it is NP-hard to distinguish whether H has an independent set of size $\Omega\left(k^{1-\epsilon}\right)$, or if the maximum independent set of H has size $O\left(k^{\epsilon}\right)$, for an arbitrary small constant $\epsilon>0$.

We choose $p=k^{-1+2\epsilon}$; our graph G will contain a copy of H and a disjoint clique C of order $c\binom{k}{c}+k^d$, where d is an arbitrary constant larger than c. Therefore, the number of nodes will be $n=k+c\binom{k}{c}+k^d$. (Observe that it holds $p=n^{-\frac{1}{d}+\Theta\left(\frac{\epsilon}{d}\right)}$.)

For each set S of k nodes in H, we choose a set S' of k unique nodes in C, and we add each edge from S to S'.

Observe that if the maximum independent set of H has size $O\left(k^{\epsilon}\right)$, then the maximum independent set of G has also size $O\left(k^{\epsilon}\right)$, and by Lemma 4.1 the maximum expected number of $\mathcal{Y}$'s in G will be at most

$$O\left(n\cdot k^{c\epsilon}\cdot p^c\right) = O\left(k^d\cdot k^{c\epsilon}\cdot k^{-c+2c\epsilon}\right) = O\left(k^{d-c+6c\epsilon}\right) = n\cdot p^{c-O(\epsilon)}.$$

Lemma 4.1 can be applied since $k\cdot p = k\cdot k^{-1+2\epsilon} = k^{2\epsilon}\geq 1$.

On the other hand, if the maximum independent set of H has size $\Omega\left(k^{1-\epsilon}\right)$, by scheduling all the nodes in that independent set we will have that with probability $1-o(1)$ at least c nodes in the independent set will choose $\mathcal{Y}$. Let T be the set of c nodes in the independent set that chose $\mathcal{Y}$, and let T' be the set of c nodes in the clique that they were mapped to by the reduction. We schedule all the nodes in T' — they will all choose $\mathcal{Y}$ deterministically; then we can schedule the subset (of size k^d) of the nodes in C that are not connected to any node in H. Every such node will choose $\mathcal{Y}$ deterministically.

Since $k^d = n-o(n)$ (for any $d>c$), the claim is proved. □

5. THE NON-ADAPTIVE PROBLEM

Here, we consider the "non-adaptive" version of the scheduling problem: first we fix a scheduling, and then we activate the nodes in the ordering it dictates.

As the next theorem shows, in the non-adaptive case there is an interesting threshold phenomenon at $p = \frac{1}{2}$:

THEOREM 5.1. *If $p \geq \frac{1}{2}$ (resp., $p \leq \frac{1}{2}$), then for each non-adaptive scheduling, the expected number of $\mathcal{Y}$'s is greater than or equal (resp., less than or equal) $\frac{1}{2} \cdot n$.*

PROOF.
Let us label the nodes in terms of an arbitrary scheduling, $v_1, v_2, \ldots, v_n$, so that node v_i will be activated before node v_j iff $i < j$. Our random process assigns a signal $X_i \in \{Y, N\}$ to each node v_i independently — in such a way that $\Pr[X_i = Y] = p$. Furthermore, after having fixed a schedule, and having sampled the variables $X_1, \ldots, X_n$, the process becomes deterministic.

Take the ith node in the scheduling v_i, for any $1 \leq i \leq n$. Let $\mathbf{X}_i$ be the random vector $\mathbf{X}_i = (X_1, X_2, \ldots, X_i)$.

For now, let us assume $p = \frac{1}{2}$. Then, clearly, every realization of $\mathbf{X}_i$ is equiprobable.

Given a realization $x = (x_1, x_2, \ldots, x_i)$ of $\mathbf{X}_i$ (that is, given a boolean vector on i coordinates), we denote its coordinate-wise complement by $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_i)$. Observe that the ith node chooses $\mathcal{Y}$ with realization x iff the ith node chooses $\mathcal{N}$ with realization $\bar{x}$. This is trivially true if $i = 1$ (since in that case, $v_i = v_1$ will make the choice based only on its signal). Now, if it is true up until i, then it is also true at $i+1$: indeed, each neighbor of v_{i+1} has an index less than or equal i, and therefore will change its choice. If, given x, the neighbors of v_{i+1} forced it to make a choice, then – with $\bar{x}$ – they will force it to make the opposite choice: that is, if, with x, there were at least c more $\mathcal{Y}$'s than $\mathcal{N}$'s (resp., $\mathcal{N}$'s than $\mathcal{Y}$'s), then v_{i+1} was forced to choose $\mathcal{Y}$ (resp., $\mathcal{N}$) — but, the opposite will happen on $\bar{x}$ and the ith node will be forced to choose $\mathcal{N}$ (resp., $\mathcal{Y}$). If on the other hand, given x, v_{i+1} made its choice according to its signal, it will make the choice dictated by the opposite signal with $\bar{x}$.

Since all realizations of $\mathbf{X}_i$ are equally likely, and since we gave a perfect matching between that guarantees that only contains edges connecting a realization where v_i chose $\mathcal{Y}$ with a realization where v_i chose $\mathcal{N}$, it holds that the probability that node i chooses $\mathcal{Y}$ is exactly $\frac{1}{2}$.

Therefore the claim for $p = \frac{1}{2}$, follows from linearity of expectation.

Let us now consider $p > \frac{1}{2}$ (resp., $p < \frac{1}{2}$). We define a coupling between the user signal process $\mathcal{P}$ and a new process $\mathcal{P}'$: namely, each node first get a positive signal with probability $\frac{1}{2}$ and a negative signal with probability $\frac{1}{2}$ — if the first signal is negative (resp., positive), then the node gets a second signal that is positive (resp., negative) with probability $2p - 1$ (resp., $1 - 2p$). The actual user signal in $\mathcal{P}'$ will be the last signal that the user receives. A simple calculation shows that the probability of a positive signal in $\mathcal{P}'$ is exactly p.

Since the function that maps a node's neighbors' choices and the node signal to the node choice is monotonically non-decreasing in each coordinate (i.e., in the neighbors' choices and the node signal), if $p > \frac{1}{2}$ (resp., $p < \frac{1}{2}$), then the first signal that a node receives is sufficient to guarantee that the probability that the node chooses $\mathcal{Y}$ is at least (resp., at most) $\frac{1}{2}$. The second signal can only increase (resp., decrease) that probability. □

Theorem 5.1 directly translates to a 2-approximation algorithm for finding the non-adaptive schedule that maximizes the number of $\mathcal{Y}$'s , when $p \geq \frac{1}{2}$: any scheduling will guarantee at least $\frac{n}{2}$ $\mathcal{Y}$'s on expectation, and the maximum number of $\mathcal{Y}$'s is n. We will show later that, for small p, a constant approximation is not achievable.

Clearly, the bound of $\frac{n}{2}$ is not always tight: if $p < \frac{1}{2}$, and the graph is a clique, the expected number of $\mathcal{Y}$'s is at most $O(p^c) \cdot n$; on the other hand, if $p > \frac{1}{2}$, then the expected number of $\mathcal{Y}$'s is at least $(1 - O(1-p)^c) \cdot n$.

Also, if the graph is an independent set, then the expected number of $\mathcal{Y}$'s is exactly $p \cdot n$. More generally, we have the following easy lemma:

LEMMA 5.2. *If the graph G contains an induced $(c-1)$-subgraph of size k, then the maximum expected number of $\mathcal{Y}$'s , with a non-adaptive scheduling, is at least $k \cdot p$.*

PROOF. The scheduling is similar to the one of Algorithm 1. Specifically, given a set W of nodes that induces a $(c-1)$-degenerate subgraph of G, we schedule the nodes in W according to some Erdös-Hajnal sequence (that is, we schedule them in such a way that whenever a node gets activated, that node has at most $c-1$ activated neighbors).

The choice of $v \in W$, then, depends only on its private signal; that is, v will choose $\mathcal{Y}$ independently with probability p. The claim follows from linearity of expectation. □

We observe that the upper bound of Lemma 4.1, that held in the adaptive setting, holds also in the non-adaptive case.

We now show that the non-adaptive scheduling problem cannot be approximated to better than $\Omega\left(p^{c-1-\epsilon}\right)$ for sufficiently small p.

THEOREM 5.3. *In the non-adaptive case, it is NP-hard to distinguish between graphs G for which the maximum number of $\mathcal{Y}$'s is at least $n \cdot p^{1+\epsilon}$ and graphs for which the maximum number of $\mathcal{Y}$'s is at most $n \cdot p^{c-\epsilon}$, for p equal to an inverse polynomial in n.*

PROOF. As in our other reduction, we start from an independent set instance G on $n = |V(G)|$ nodes. It is NP-hard to distinguish whether G has an independent set of size $\Omega\left(n^{1-\epsilon}\right)$, or if the maximum independent set of G has size $O\left(n^{\epsilon}\right)$, for an arbitrary small constant $\epsilon > 0$ (see [Zuckerman 2007]).
We choose $p = n^{-\frac{1}{c}+\epsilon}$.

Now, if the independent set of G had size at most $O\left(n^{\epsilon}\right)$, then Lemma 4.1 guarantees that in the adaptive-case, and therefore in the non-adaptive case, the maximum number of $\mathcal{Y}$'s is at most

$$O\left(n \cdot p^c \cdot n^{c\epsilon}\right) = n \cdot p^{c-O(\epsilon)}.$$

Suppose instead that the independent set of G had size at least $\Omega\left(n^{1-\epsilon}\right)$, then Lemma 5.2 guarantees that in the non-adaptive case the maximum number of $\mathcal{Y}$'s is at least

$$O\left(n^{1-\epsilon} \cdot p\right) = n \cdot p^{1+O(\epsilon)}.$$

□

REFERENCES

ACEMOGLU, D., DAHLEH, M. A., LOBEL, I., AND OZDAGLAR, A. to appear. Bayesian learning in social networks. *Review of Economic Studies*.

ARTHUR, W. B. 1989. Competing technologies, increasing returns, and lock-in by historical events. *The Economic Journal 99,* 394, 116–131.

BANERJEE, A. 1992. A simple model of herd behavior. *Quarterly Journal of Economics 107*, 797–817.

BIKHCHANDANI, S., HIRSHLEIFER, D., AND WELCH, I. 1992. A theory of fads, fashion, custom and cultural change as information cascades. *Journal of Political Economy 100*, 992–1026.

BLUME, L., EASLEY, D., KLEINBERG, J., KLEINBERG, R., AND TARDOS, É. 2011. Which networks are least susceptible to cascading failures? In *Proc. 52nd IEEE Symposium on Foundations of Computer Science*.

CENTOLA, D. AND MACY, M. 2007. Complex contagions and the weakness of long ties. *American Journal of Sociology 113*, 702–734.

DODDS, P. AND WATTS, D. 2004. Universal behavior in a generalized model of contagion. *Physical Review Letters 92*, 218701.

DOMINGOS, P. AND RICHARDSON, M. 2001. Mining the network value of customers. In *Proc. 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 57–66.

HARTLINE, J. D., MIRROKNI, V. S., AND SUNDARARAJAN, M. 2008. Optimal marketing strategies over social networks. In *WWW*. 189–198.

KEMPE, D., KLEINBERG, J., AND TARDOS, É. 2003. Maximizing the spread of influence in a social network. In *Proc. 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 137–146.

MITZENMACHER, M. AND UPFAL, E. 2005. *Probability and computing: randomized algorithms and probabilistic analysis*. Cambridge University Press.

MOSSEL, E. AND ROCH, S. 2007. On the submodularity of influence in social networks. In *Proc. 39th ACM Symposium on Theory of Computing*.

RICHARDSON, M. AND DOMINGOS, P. 2002. Mining knowledge-sharing sites for viral marketing. In *Proc. 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 61–70.

SALGANIK, M., DODDS, P., AND WATTS, D. 2006. Experimental study of inequality and unpredictability in an artificial cultural market. *Science 311*, 854–856.

WATTS, D. J. 2002. A simple model of global cascades on random networks. *Proc. Natl. Acad. Sci. USA 99,* 9, 5766–5771.

ZUCKERMAN, D. 2007. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing 3,* 1, 103–128.