

Distributed Submodular Maximization with Parallel Execution

Haoyuan Sun, David Grimsman, and Jason R. Marden ^{*†‡§}

Abstract

The submodular maximization problem is widely applicable in many engineering problems where objectives exhibit diminishing returns. While this problem is known to be NP-hard for certain subclasses of objective functions, there is a greedy algorithm which guarantees approximation at least $1/2$ of the optimal solution. This greedy algorithm can be implemented with a set of agents, each making a decision sequentially based on the choices of all prior agents. In this paper, we consider a generalization of the greedy algorithm in which agents can make decisions in parallel, rather than strictly in sequence. In particular, we are interested in partitioning the agents, where a set of agents in the partition all make a decision simultaneously based on the choices of prior agents, so that the algorithm terminates in limited iterations. We provide bounds on the performance of this parallelized version of the greedy algorithm and show that dividing the agents evenly among the sets in the partition yields an optimal structure. It is shown that such optimal structures holds even under very relaxed information constraints. We additionally show that this optimal structure is still near-optimal, even when additional information (i.e., total curvature) is known about the objective function.

^{*}H. Sun (hsun2@caltech.edu) is with the California Institute of Technology (Caltech), Pasadena, CA. He was supported by Caltech's SURF program as the Øistein and Rita A. Skjellum SURF Fellow. He was also co-mentored by A. Wierman of Caltech.

[†]D. Grimsman (davidgrimsman@ucsb.edu), J. R. Marden (jmarden@ece.ucsb.edu) are with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA

[‡]This research was supported by NSF Grant #ECCS-1638214 and U.S. Office of Naval Research (ONR) grant #N00014-17-1-2060.

[§] 2020. This work has been accepted to American Control Conference for publication under a Creative Commons Licence CC-BY-NC-ND

1 Introduction

Submodular maximization is an important topic with relevance to many fields and applications, including sensor placement [1], outbreak detection in networks [2], maximizing and inferring influence in a social network [3, 4], document summarization [5], clustering [6], assigning satellites to targets [7], path planning for multiple robots [8], and leader selection and resource allocation in multiagent systems [9, 10]. An important similarity among these applications is the presence of an objective function which exhibits a “diminishing returns” property. For instance, a group of leaders can impose some influence on a social network, but the marginal gain in influence achieved by adding a new leader to the group decreases as the size of the group increases. Objective functions (such as influence) satisfying this property are *submodular*.

The submodular maximization problem is to choose a set of elements (such as social leaders) which maximize the submodular objective function, according to some constraints. This problem is known to be NP-hard for certain subclasses of submodular functions [11]. Therefore, much research has focused on how to approximate the optimal solution [12, 13, 14, 15]. The overall message of this research is that simple algorithms can perform well by providing solutions which are guaranteed to be within some factor of optimal.

One such algorithm is the greedy algorithm, first proposed in [16]. It was shown in this seminal work that for certain classes of constraints the solution provided by the greedy algorithm is guaranteed to be within $1 - 1/e$ of the optimal, and within $1/2$ of the optimal for the more general cases [17]. Since then, more sophisticated algorithms have been developed to show that there are many instances of the submodular maximization problem which can be solved efficiently within the $1 - 1/e$ guarantee [12, 18]. It has also been shown that progress beyond this level of optimality is not possible using a polynomial-time algorithm [19].

More recent research has focused on distributed algorithms, since in many cases having a centralized agent with access to all the relevant data is untenable [20, 6, 21]. In this case, the greedy algorithm can be generalized using a set of n agents, each with its own decision set. The combined set of decisions by the agents is evaluated by the submodular objective function, which they seek to maximize. Each agent chooses sequentially, maximizing its marginal contribution with respect to the decisions of the prior agents. In this setting, the greedy algorithm has been shown to provide a solution within $1/2$ of the optimal. While simplistic from an algorithmic perspective, the informational requirement can be quite demanding as agents are required to observe all previous choices. Accordingly, researchers have sought to characterize the impact of reducing this informational dependencies on the resulting performance guarantees [5, 22, 23].

In this paper, we consider the case where no such centralized authority is present, i.e., the agents’ decision sets are determined a priori and cannot be modified. Terminating the greedy algorithm in $q < n$ iterations thus requires a partitioning of the agents into sets $1, \dots, q$, where now each agent in set j simultaneously chooses an action which maximizes its marginal contribution relative to the actions chosen by all the agents in sets $1, \dots, j - 1$. In this setting, one can ask the following questions:

- What is the best way to partition the agents? Should the agents be spread evenly across the sets in the partition, or should set 1 or set q be larger than the others?
- Can we further reduce the amount of informational dependencies without sacrificing any performance guarantees?
- If some additional structure is known about the submodular objective function, how does that affect the partitioning strategy?

In response to the questions listed above, the contributions of this paper are the following:

1. Theorem 1 shows that given time constraint q , partitioning the agents into equally-sized sets is the best strategy to parallelize the greedy algorithm.
2. Theorem 2 proves that we can relax the information sharing constraints under a graph-theoretic interpretation and still achieve the same performance guarantees.
3. Theorem 3 shows that if we know some additional properties of the submodular function, i.e., their total curvature, then we have improved performance guarantees and the strategy in Theorem 2 is nearly optimal.

This paper is organized as follows:

1. Section 2 introduces general definitions and the greedy algorithm;
2. Section 3 introduces Theorem 1;
3. Section 4.1 gives a more general graph-theoretic interpretation of the model and introduces Theorem 2.
4. Section 5 defines the total curvature property and introduces Theorem 3.

2 Model

2.1 Distributed Submodular Optimization

Consider a base set S and a set function $f : 2^S \rightarrow \mathbb{R}_{\geq 0}$. Given $A, B \subseteq S$, we define the *marginal contribution* of A with respect to B as

$$f(A | B) = f(A \cup B) - f(B) \tag{1}$$

In this paper, we are interested in distributed algorithms for maximization of submodular functions, where there is a collection of n decision-making agents, named as $N = \{1, \dots, n\}$, and a set of decisions S . Each decision-making agent is associated with a set of permissible decisions $X_i \subseteq S$ so that X_1, \dots, X_n form a partition of S . For rest of the paper, we refer to decision-making agents as simply agents. Then the collection of decisions by all agents $x = \{x_1, \dots, x_n\}$ is called an

action profile, and we denote the set of all action profiles as $X = X_1 \times \cdots \times X_n$. An action profile $x \in X$ is evaluated by an *objective function* $f : 2^S \rightarrow \mathbb{R}_{\geq 0}$ as $f(x) = f(\cup_i \{x_i\})$. Furthermore, we restrict our attention to f 's which satisfy the following properties:

- **Normalized** $f(\emptyset) = 0$.
- **Monotonic** $f(\{e\} | A) \geq 0$ for all $e \in S$ and $A \subseteq S$.
- **Submodular** $f(\{e\} | A) \geq f(\{e\} | B)$ for all $A \subseteq B \subseteq S$ and $e \in S \setminus B$.

For simplicity, define \mathcal{F} as the set of functions satisfying these three properties. For any $f \in \mathcal{F}$, the goal of the submodular maximization problem is to find:

$$x^{\text{opt}} \in \arg \max_{x \in X} f(x) \tag{2}$$

For convenience, we overload f with the understanding that $f(e) = f(\{e\})$ for $e \in S$, and $f(A, B) = f(A \cup B)$ for $A, B \subseteq S$. We also introduce the following notation. For any $j, k \in \{1, \dots, n\}$, $j < k$, let $[k] = \{1, \dots, k\}$ and $j : k = \{j, i + 1, \dots, k\}$. Furthermore, for any action profile $x \in X$ and set $M \subseteq N$, let $f(x_M) = f(\cup_{i \in M} \{x_i\})$. For instance, $f(x_{1:10}) = f(\{x_1, \dots, x_{10}\})$.

An example of the submodular maximization problem is the weighted set cover problem [18]. Given a target set Y and a mapping $T : S \rightarrow 2^Y$ from the decisions to subsets of Y ,¹ the value of an action profile $x \in X$ is determined by a weight function $w : Y \rightarrow \mathbb{R}_{\geq 0}$:

$$f(x) = \sum_{y \in \cup_i T(x_i)} w(y) \tag{3}$$

Intuitively, this problem aims to “cover” as much of the target set as possible. An instance of this problem is illustrated by Figure 1.

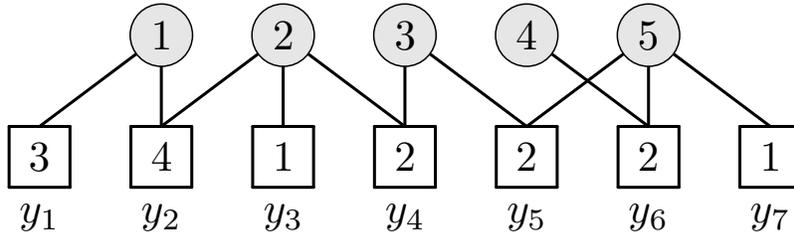
2.2 Greedy Algorithm and Parallelization

A distributed greedy algorithm can be used to approximate the problem as stated in (2). In the greedy algorithm, the agents make decision in sequential order according to their names $i \in N$ and each agent attempts to maximize its marginal contribution with respect to the choices of prior agents:

$$x_i^{\text{sol}} \in \arg \max_{x_i \in X_i} f(x_i | x_{1:i-1}) \tag{4}$$

Note that the greedy solution x^{sol} may not be unique. In the context of this paper, we assume that x^{sol} is the worst solution among all possible greedy solutions. Define $\gamma(f, X)$ to be $f(x^{\text{sol}})/f(x^{\text{opt}})$ from the greedy algorithm subject to objective function f and set of action profiles X . Then, to

¹ Note that the decisions are related to assignments to the subsets of Y in the sense that each decision is an ordered pair (agent ID, subset of Y). This way we allow agents to choose the same element from Y while maintaining that the decisions are distinct.



(a) Illustration of weighted subset cover problem. The 5 agents are represented by circles. The target set $Y = \{y_1, \dots, y_7\}$ is represented by boxes. For each box, its label indicates its weight, and the black lines indicate the permissible decisions, for example, $T(X_1) = \{\{y_1\}, \{y_2\}\}$.

Algorithm	x_1	x_2	x_3	x_4	x_5	$f(x_{1:5})$
Optimal	y_1	y_2	y_4	y_6	y_5	13
Greedy	y_2	y_4	y_5	y_6	y_7	11
Parallel (P_1)	y_2	y_2	y_5	y_6	y_6	8
Parallel (P_2)	y_2	y_2	y_5	y_6	y_7	9

(b) For the above weighted set cover problem, this table shows the decisions x_1, \dots, x_5 by several algorithms described in this paper. P_1 and P_2 are the optimal iteration assignments as shown in Fig. 2a and 2c, respectively. For simplicity, we used element instead of set to denote each decision, e.g. we wrote y_1 but we mean $\{y_1\}$. To illustrate how the greedy algorithm works, let's consider P_2 . We have $x_1 = x_2 = \{y_2\}$ because agents 1 and 2 are deciding at the same time. Then agents 3 and 4 choose at the same time, and one possible result is $x_3 = \{y_5\}, x_4 = \{y_6\}$. Lastly, agent 5 makes a decision, it sees that agents 3 and 4 already took y_5 and y_6 , so it chooses $\{y_7\}$ as the decision maximizing its marginal contribution.

Figure 1: An instance of weighted subset cover problem and the behavior of various algorithms for this problem.

measure the quality of the greedy algorithm, we consider its *competitive ratio*:

$$\gamma = \inf_{f \in \mathcal{F}, X} \gamma(f, X) \tag{5}$$

The well known result from [17] states that $\gamma = 1/2$, which means that (4) guarantees that the performance of the greedy solution is always within 1/2 of optimal. Furthermore, only one agent can make a decision at any given time and thus a solution will be found in precisely n iterations.

The focus of this work is on characterizing the achievable competitive ratio for situations where a system-design does not have the luxury of having n iterations to construct a decision. To this end, we introduce the parallelized greedy algorithm to allow multiple agents to make decisions at the same time. More formally, we consider a situation where the system is given limited number of iterations $q \leq n$ and needs to come up with an iteration assignment $P : [n] \rightarrow [q]$ so that each agent makes decision only based off the choices made by agents from earlier iterations:

$$x_i^{\text{sol}} \in \arg \max_{x_i \in X_i} f(x_i \mid x_{\mathcal{N}_i}) \tag{6}$$

where $\mathcal{N}_i = \{j \mid P(j) < P(i)\} \subseteq \{1, \dots, i - 1\}$. Note, that to maintain consistency with (4), we require that P preserves the implicit ordering of the greedy algorithm: $P(i) \leq P(j)$ whenever $i < j$.

We are interested in the performance of (6) given a fixed number of agents and number of iterations. We define the competitive ratio of a time step assignment P as:

$$\gamma(P) = \inf_{f \in \mathcal{F}, X} \gamma(f, X, P) \tag{7}$$

where $\gamma(f, X, P)$ is $f(x^{\text{sol}})/f(x^{\text{opt}})$ from the parallelized greedy algorithm subject to objective function f , set of action profiles X and iteration assignment P . Denote the set of all possible iteration assignments with n agents and at most q iterations as:

$$\mathcal{P}_{n,q} = \{P : [n] \rightarrow [q]\} \tag{8}$$

We seek to find the best possible competitive ratio, and the iteration assignment which achieves the optimal competitive ratio (if such an assignment exists):

$$\rho(n, q) = \sup_{P \in \mathcal{P}_{n,q}} \gamma(P) \tag{9}$$

$$P_{n,q}^* \in \arg \max_{P \in \mathcal{P}_{n,q}} \gamma(P) \tag{10}$$

3 Optimal Parallel Structures

In this section, we present the best possible competitive ratio for the parallelized greedy algorithm and optimal iteration assignment which achieves such a bound.

Note that according to (6), a pair of agents do not utilize each other’s decisions (in either direction) when they are in the same iteration. This intuitively represent some “blindspots” in the parallelized greedy algorithm compared to the original greedy algorithm. One way to reduce those “blindspots” is to divide the agents evenly among the available iterations, in other words, we want the number of agents deciding in parallel to be as close to the average number as possible. Theorem 1 illustrates that this simple idea is nearly sufficient to yield the best possible iteration assignment.

Theorem 1. *Given a parallelized submodular maximization problem with n agents and q iterations, the competitive ratio is:*

$$\rho(n, q) = \begin{cases} \frac{1}{r} & \text{if } n \equiv 1 \pmod{q} \\ \frac{1}{r+1} & \text{otherwise} \end{cases} \quad (11)$$

where $r = \lceil n/q \rceil$. In particular, when $n \equiv 1 \pmod{q}$,

$$P_{n,q}^* = \begin{cases} \left\lceil \frac{i}{r-1} \right\rceil & \text{if } i < n \\ q & \text{if } i = n \end{cases} \quad (12)$$

Otherwise,

$$P_{n,q}^* = \left\lceil \frac{i}{r} \right\rceil \quad (13)$$

Figure 2 illustrates some example of optimal and non-optimal assignments. In particular, Figure 2b gives an edge case where our simple intuition is not enough to make the optimal assignment. And Figure 2d shows that other approaches, such as assigning more agents to earlier iterations to reduce the “blindness” in later iterations, are not optimal.

We observe that the competitive ratio is roughly inversely proportional to the value r . Theorem 1 as stated does not offer a good explanation for why this relation holds. To better understand this, we will generalize our model in Section 4.1 and present a strengthening of Theorem 1 in Section 4.3.

4 Parallelization as Information Exchange

We observe that our parallelization approach implicitly defines a certain type of informational dependencies. In both (4) and (6), each agent uses the choices of prior agents to make its own decision. Alternatively, we can view this process as an agent communicating its decision to other agents who depend on this piece of information. In our parallelization model, each agent is required to broadcast its decision to all of the later agents. This requirements seems rather restrictive and thus in this section, we will reduce the amount of informational dependencies by relaxing

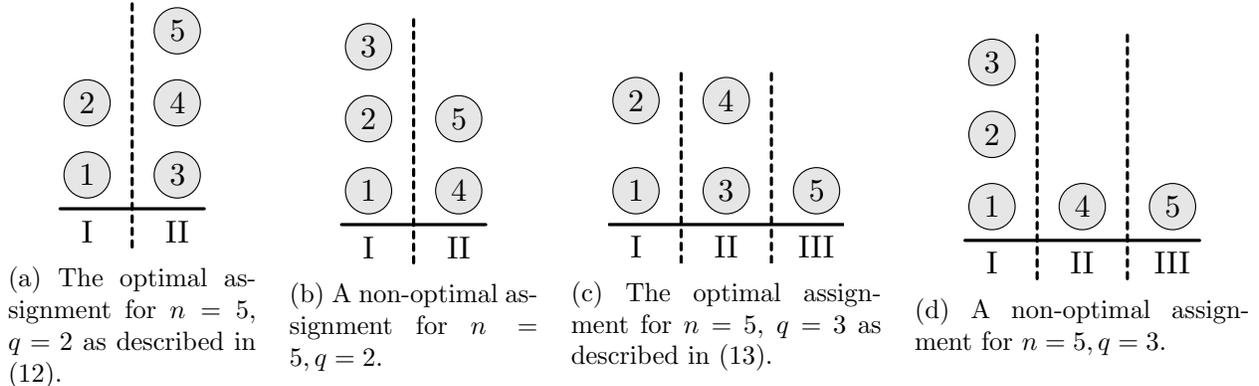


Figure 2: Examples of optimal iteration assignments as given by Theorem 1. Each agent, represented by a node, is named as $1, \dots, n$. Each column, labeled by a Roman numeral, contains all agents executed at a given iteration. Also shown are some examples of iteration assignments that are not optimal. According to Theorem 1, the competitive ratio in Fig. 2a is $1/3$ and the competitive ratio in Fig. 2c is also $1/3$. On the other hand, in Section 4.3, Lemma 3 shows that the competitive ratio in Fig. 2b and Fig. 2d are both $1/4$.

the constraints. Specifically, we will treat the communication between one pair of agents as an individual entity so that an agent only has to send its choice to selected later agents and receive choices from selected previous agents.

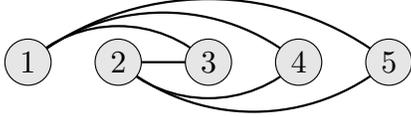
4.1 Parallelization and Information

We can model the information exchange as an undirected graph $G = (V, E)$, where $V = N$ represents the agents and each edge $(i, j) \in E$ represents information being exchanged between two agents. Since the greedy algorithm has an implicit ordering induced by the agents' names, for every pair $i < j$ so that $(i, j) \in E$, agent j requires knowing the choice of agent i , but agents i does not access agent j 's decision. Hence, we can use G , which we will call the *information graph*, to determine the set of prior agents which agent $i \in N$ accesses as $\mathcal{N}_i = \{j < i \mid (j, i) \in E\} \subseteq [i - 1]$. From here, it is natural to consider the generalized greedy algorithm proposed in [24]:

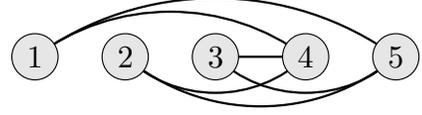
$$x_i^{\text{sol}} \in \arg \max_{x_i \in X_i} f(x_i \mid x_{\mathcal{N}_i}) \quad (14)$$

First we noticed that the parallelized greedy algorithm as defined in (6) is a special case of (14). In the parallelized greedy algorithm, each agent requires information from all agents executed in prior iterations. Therefore, an iteration assignment $P \in \mathcal{P}_{n,q}$ induces a corresponding information graph $G = (V, E)$ so that $V = N$, $E = \{(i, j) \mid P(i) < P(j)\}$. Figure 3 illustrates how the iteration assignments from Figure 2 induce corresponding information graphs. For rest of this section, we mention an iteration assignment with the understanding that we are interested in its induced information graph.

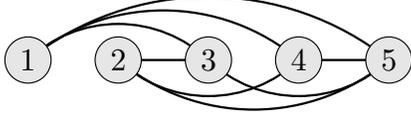
Conversely, an information graph induces an ordering to parallelize the generalized greedy



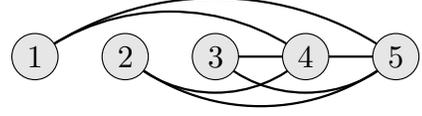
(a) The optimal assignment for $n = 5$, $q = 2$ as described in (12).



(b) A non-optimal assignment for $n = 5$, $q = 2$.



(c) The optimal assignment for $n = 5$, $q = 3$ as described in (13).



(d) A non-optimal assignment for $n = 5$, $q = 3$.

Figure 3: The induced information graph of iteration assignments from Figure 2 as discussed in Section 4. In the information graphs, an edge (i, j) where $i < j$ represents that the iteration of agent i is before the iteration of agent j and hence agent j requires knowing the choice made by agent i before making its own decision. Each edge can be thought of as the flow of information exchanges between the agents in which a prior agent share its choice so later agents can decide.

algorithm. If all agents in \mathcal{N}_i have made their decisions but agent $j < i$ is still undecided, then we can let agents i and j to make their decisions in the same iteration without affecting the algorithm. Intuitively, we can think of information being relayed through some paths in G and the earliest iteration for which agent i can decide depends on the longest path in G leading to node i . Formally, a simple induction argument shows that the function $\bar{P} : G \times [n] \rightarrow [n]$ determines the earliest iteration in which an agent could decide:

$$\bar{P}(G; i) = \begin{cases} 1 & \text{if } \mathcal{N}_i = \emptyset \\ 1 + \max_{j \in \mathcal{N}_i} \bar{P}(G; j) & \text{otherwise} \end{cases} \quad (15)$$

This function \bar{P} can be used to construct a parallelization of the greedy algorithm in which agent i makes its decision in iteration $\bar{P}(G; i)$ and for every $j \in \mathcal{N}_i$, $\bar{P}(G; j) < \bar{P}(G; i)$. From here we can define $\mathcal{G}_{n,q}$ as the set of graphs which induces a parallelization with at most q iterations:

$$\mathcal{G}_{n,q} = \{G = (V, E) : |V| = n, \max_i \bar{P}(G; i) \leq q\} \quad (16)$$

From the the definitions, it is clear that $\mathcal{P}_{n,q} \subseteq \mathcal{G}_{n,q}$. Hence we can adopt competitive ratios for information graphs so that (7), (9) and (10) become:

$$\gamma(G) = \inf_{f, X} \gamma(f, X, G) \quad (17)$$

$$\eta(n, q) = \sup_{G \in \mathcal{G}_{n,q}} \gamma(G) \quad (18)$$

$$G_{n,q}^* \in \arg \max_{G \in \mathcal{G}_{n,q}} \gamma(G) \quad (19)$$

where $\gamma(f, X, G)$ is $f(x^{\text{sol}})/f(x^{\text{opt}})$ from the generalized greedy algorithm subject to objective function f , set of action profiles X and information graph G .

It is natural to ask whether the above generalization of parallelized greedy algorithm yields higher competitive ratio under the same number of agents and iterations. Also, in some applications, the information exchange may incur some costs, and therefore having a information graph with many edges is undesirable. So we wish to answer a second question, whether we can achieve the same competitive ratio in (11) with information graphs that have fewer edges than that of (12) and (13). Theorems 2, as presented below, show that generalizing to information graphs does not yield higher competitive ratio but allows us to achieve the same optimal competitive ratio with fewer edges.

4.2 Preliminary: Graph Theory

The main result on information graphs leverages several concepts from graph theory. We assume that we have an undirected graph $G = (V, E)$:

Definition 1. Nodes $K \subseteq V$ form a *clique* if for all distinct $i, j \in K$, $(i, j) \in E$. The *clique number* $\omega(G)$ of a graph G is the size of the maximum clique in G .

Definition 2. A *clique cover* of a graph G is a partition of V so that each set in the partition forms a clique. And the *clique cover number* $\theta(G)$ is the least number of cliques necessary to form a clique cover.

Definition 3. Nodes $I \subseteq V$ form an *independent set* if for all distinct $i, j \in I$, $(i, j) \notin E$. The *maximum independent set* is an independent set with the largest possible number of nodes. And the *independence number* $\alpha(G)$ is the size of the maximum independent set.

Lastly, a *Turán graph* $T(n, r)$ of n vertices and clique number r is constructed as follows:

- We partition the n vertices into r subsets as evenly as possible. More specifically, we have $(n \bmod r)$ subsets with $\lceil n/r \rceil$ vertices and $(r - n \bmod r)$ subsets with $\lfloor n/r \rfloor$ vertices.
- We draw an edge between two vertices if and only if they belong to different subsets.

And it has the following interesting property [25]:

Lemma 1 (Turán). *The Turán graph $T(n, r)$ is the n -vertex graph with the largest number of edges that does not contain an $(r + 1)$ -clique. In other words:*

$$T(n, r) = \arg \max_{G=(V,E): |V|=n, \omega(G) \leq r} |E| \quad (20)$$



(a) The optimal graph for $n = 5$, $q = 2$ as described in (22). (b) The optimal graph for $n = 5$, $q = 3$ as described in (23).

Figure 4: Examples of optimal information graph as given by Theorem 2. The agents, represented by nodes, are implicitly named as $1, \dots, n$ from left to right and the Roman numeral labels indicate the iteration in which the agents will be executed. Note that Fig. 4a is a subgraph of Fig. 3a and Fig. 4b is a subgraph of Fig. 3c. All four of these graphs achieve their respective optimal competitive ratio, which all happen to be $1/3$. However, both Fig. 4a and Fig. 4b have 4 edges, whereas Fig. 3a has 6 edges and Fig. 3c has 8 edges. Also, by Theorem 3, when the objective function has total curvature λ , Fig. 4a and Fig. 4b are still nearly-optimal.

4.3 Main Result on Information Graphs

We will present a stronger version of Theorem 1 that generalizes to the setting of information graphs.

Theorem 2. *Given a parallelized submodular maximization problem with n agents and q iterations, the competitive ratio is (recall that $r = \lceil n/q \rceil$):*

$$\eta(n, q) = \begin{cases} \frac{1}{r} & \text{if } n \equiv 1 \pmod{q} \\ \frac{1}{r+1} & \text{otherwise} \end{cases} \quad (21)$$

In particular, when $n \equiv 1 \pmod{q}$, for the following edge set E , $G = (V, E)$ achieves the equality $\gamma(G) = 1/r$:

$$E = \left\{ \begin{array}{l} (i, j) \text{ for all } i < j < n, i \equiv j \pmod{r-1} \\ (i, n) \text{ for all } i \leq (q-1)(r-1) \end{array} \right\} \quad (22)$$

Otherwise, for the following edge set E , $G = (V, E)$ achieves the equality $\gamma(G) = 1/(r+1)$:

$$E = \{(i, j) \mid i \neq j, i \equiv j \pmod{r}\} \quad (23)$$

Figure 4 illustrates the aforementioned optimal information graphs. Note that (22) and (23) are subgraphs of the information graph induced by (12) and (13), respectively. Furthermore, (22) has approximately $1/(r-1)$ times as many edges as (12) and (23) has approximately $1/r$ times as many edges as (13). This is true because every agent (except agent n when $n \equiv 1 \pmod{q}$) requires information from just one agent from each prior time step. Additionally, (23) is “delightfully parallel” so that we can partition the agents into threads according to the clique in which they reside, and no information need to be exchanged among threads.

The following lemma shows that the value r is related to the independence number of the information graph:

Lemma 2. *For any $G \in \mathcal{G}_{n,q}$, $\alpha(G) \geq r = \lceil n/q \rceil$.*

Proof. Let $D_k = \{i \mid \bar{P}(G; i) = k\}$ be the set of agents executed in the k th iteration. From our construction, $\bigcup_{1 \leq k \leq q} D_k = [n]$. By the pigeonhole principle, $\max_{1 \leq k \leq q} |D_k| \geq r$. Since D_k must be an independent set for any $1 \leq k \leq q$, we conclude that $\alpha(G) \geq r$. \square

Since the agents in an independent set do not rely on each other to make a decision, they can be assigned to the same iteration. Therefore r is a lower bound on how many agents can decide in parallel.

The proof of Theorem 2 relies on Theorem 1 and Corollary 1 from [26] that relates the competitive ratio of an information graph to its independence number and clique cover number:

Lemma 3. *Given any information graph $G = (V, E)$,*

$$\frac{1}{\alpha(G)} \geq \gamma(G) \geq \frac{1}{\theta(G) + 1} \quad (24)$$

Lemma 4. *If there exists $w \in V$ and a maximum independent set I s.t. $i \in \mathcal{N}_w$ for some $i \in I$, then*

$$\gamma(G) \leq \frac{1}{\alpha(G) + 1} \quad (25)$$

By combining Lemmas 2 and 3, we can see that, intuitively, the competitive ratio η should be approximately $1/r$. Lastly, we note that the graph described by (23) is in fact a complement Turán graph. This reinforces our earlier intuitions and relates to Lemma 3 that the optimal parallel structure arises from having as few agents decide in parallel as possible. For the full proof of Theorem 2, please refer to Appendix A.

5 Submodular Functions with Total Curvature

Recent research, such as [21], has been focused on whether imposing additional structure on the objective function and the set of action profiles would help the underlying greedy algorithm to yield better performing solutions. One intuition is to consider the scenario when actions do not “overlap” in the sense that each decision is guaranteed to make some marginal contribution. Here, we will analyze how the property introduced in [27] enables a parallelized greedy algorithm to yield higher competitive ratio.

Definition 4. Objective function $f : 2^S \rightarrow \mathbb{R}_{\geq 0}$ has *total curvature* $\lambda \in (0, 1)$ if $f(e \mid A) \geq (1 - \lambda)f(e)$ for $e \in S$ and $A \subseteq S \setminus \{e\}$. We denote the set of normalized, monotonic and submodular functions with total curvature λ as \mathcal{F}_λ .

We can extend competitive ratio defined in (17) and (18) to when they are subject to total curvature of λ :

$$\gamma_\lambda(G) = \inf_{f \in \mathcal{F}_\lambda, X} \gamma(f, X, G) \quad (26)$$

$$\eta_\lambda(n, q) = \sup_{G \in \mathcal{G}_{n, q}} \gamma_\lambda(G) \quad (27)$$

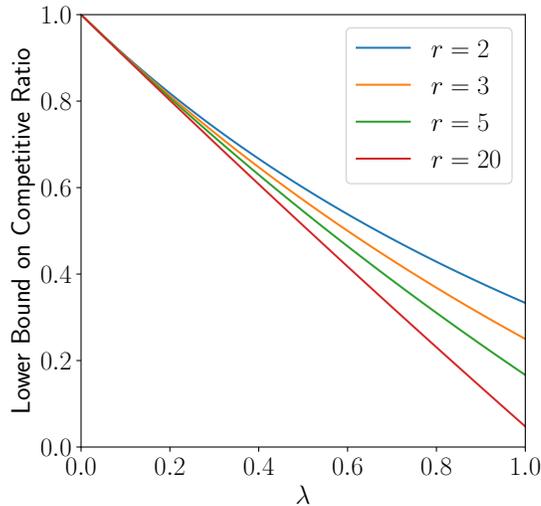


Figure 5: The lower bound (29) on the optimal competitive ratio when the objective function has total curvature λ , as given in Theorem 3. For $r = 2, 3, 5, 20$, we plot the lower bound against λ . Note that at $\lambda = 0$, the competitive ratio becomes 1, as expected from our intuition, and at $\lambda = 1$, the lower bound is $1/(r + 1)$, which is consistent with (21).

We can generalize the result of Lemma 3 to the total curvature setting:

Lemma 5. *Given any information graph G ,*

$$\frac{\alpha(G) - (\alpha(G) - 1)\lambda}{\alpha(G)} \geq \gamma_\lambda(G) \geq \frac{\theta(G) - (\theta(G) - 1)\lambda}{\theta(G) + \lambda} \quad (28)$$

We then generalize Theorem 2 to the total curvature setting and demonstrates that the iteration assignment (23) is nearly optimal.

Theorem 3. *Given a parallelized submodular maximization problem with n agents and q iterations, where the objective function has total curvature λ , then:*

$$\frac{r - (r - 1)\lambda}{r} \geq \eta_\lambda(n, q) \geq \frac{r - (r - 1)\lambda}{r + \lambda} \quad (29)$$

where $r = \lceil n/q \rceil$. Furthermore, the graph G as defined in (23) achieves the lower bound $\gamma_\lambda(G) \geq \frac{r - (r - 1)\lambda}{r + \lambda}$.

If there is no parallelization (when $r = 1$), [27] showed that the lower bound in (29) is tight.

Figure 5 illustrates how the lower bound changes with respect to λ at different r . We can make some quick observations on how the lower bound changes with respect to λ . In the limit of $\lambda \rightarrow 0$, the competitive ratio tends to 1, as expected from our intuition, and in the limit of $\lambda \rightarrow 1$, the lower bound tends to $1/(r + 1)$, which is consistent with (21). This confirms the intuition that for smaller λ , decisions have less “overlap” and as a result the greedy algorithm can perform closer to

the optimal. Also, as $r \rightarrow \infty$ both the lower and upper bounds converge to $1 - \lambda$. So the lower and upper bounds are close to each other if r is large (when there are a lot of parallelization).

For full proofs of the results in this section, please refer to Appendices B and C.

6 Conclusion

In this paper, we derived bounds on the competitive ratio of the parallelized greedy algorithm for both submodular objective functions and those with an additional property of total curvature. We also provided the optimal design which achieves such bound and showed that a graph theoretic approach yields more effective parallelization that still achieves the same competitive ratio.

There are several directions of future research. One possibility is to consider whether employing other structural properties on the objective functions can also improve the competitive ratio of greedy algorithm. In particular, we are interested in properties that consider fixed number decisions at once because the total curvature property considers arbitrarily many decisions at once. Another possible direction is to consider applying something other than the marginal contribution in making the greedy decisions.

References

- [1] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg, “Near-optimal sensor placements: Maximizing information while minimizing communication cost,” in *Proceedings of the International Conference on Information Processing in Sensor Networks*. ACM, 2006.
- [2] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, C. Faloutsos, J. VanBriesen, and N. Glance, “Cost-effective outbreak detection in networks,” in *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2007.
- [3] D. Kempe, J. Kleinberg, and É. Tardos, “Maximizing the spread of influence through a social network,” in *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003.
- [4] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, “Inferring networks of diffusion and influence,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 4, 2012.
- [5] H. Lin and J. Bilmes, “A class of submodular functions for document summarization,” in *Proceedings of the Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011.
- [6] B. Mirzasoleiman, A. Karbasi, R. Sarkar, and A. Krause, “Distributed submodular maximization: Identifying representative elements in massive data,” in *Advances in Neural Information Processing Systems*, 2013.

- [7] G. Qu, D. Brown, and N. Li, “Distributed greedy algorithm for multi-agent task assignment problem with submodular utility functions,” *Automatica*, vol. 105, 2019.
- [8] A. Singh, A. Krause, C. Guestrin, W. J. Kaiser, and M. A. Batalin, “Efficient planning of informative paths for multiple robots.” in *International Joint Conferences on Artificial Intelligence*, vol. 7, 2007.
- [9] A. Clark and R. Poovendran, “A submodular optimization framework for leader selection in linear multi-agent systems,” in *Conference on Decision and Control and European Control Conference*. IEEE, 2011.
- [10] J. R. Marden, “The role of information in distributed resource allocation,” *IEEE TCNS*, vol. 4, no. 3, Sept 2017.
- [11] L. Lovász, “Submodular functions and convexity,” in *Mathematical Programming The State of the Art*. Springer, 1983.
- [12] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, “Maximizing a submodular set function subject to a matroid constraint,” in *International Conference on Integer Programming and Combinatorial Optimization*. Springer, 2007.
- [13] M. Minoux, “Accelerated greedy algorithms for maximizing submodular set functions,” in *Optimization Techniques*. Springer, 1978.
- [14] N. Buchbinder, M. Feldman, J. Seffi, and R. Schwartz, “A tight linear time $(1/2)$ -approximation for unconstrained submodular maximization,” *SIAM Journal on Computing*, vol. 44, no. 5, 2015.
- [15] J. Vondrák, “Optimal approximation for the submodular welfare problem in the value oracle model,” in *ACM Symposium on Theory of Computing*. ACM, 2008.
- [16] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, “An analysis of approximations for maximizing submodular set functions I,” *Mathematical Programming*, vol. 14, no. 1, 1978.
- [17] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, “An analysis of approximations for maximizing submodular set functionsii,” in *Polyhedral combinatorics*. Springer, 1978.
- [18] M. Gairing, “Covering games: Approximation through non-cooperation,” in *International Workshop on Internet and Network Economics*. Springer, 2009.
- [19] U. Feige, “A threshold of $\ln n$ for approximating set cover,” *Journal of the ACM (JACM)*, vol. 45, no. 4, 1998.
- [20] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, *Submodularity in dynamics and control of networked systems*. Springer, 2016.

- [21] M. Corah and N. Michael, “Distributed submodular maximization on partition matroids for planning on large sensor networks,” in *Conference on Decision and Control*. IEEE, 2018.
- [22] X. Pan, S. Jegelka, J. E. Gonzalez, J. K. Bradley, and M. I. Jordan, “Parallel double greedy submodular maximization,” in *Advances in Neural Information Processing Systems*, 2014.
- [23] A. Ene, H. L. Nguyen, and A. Vladu, “A parallel double greedy algorithm for submodular maximization,” *arXiv preprint arXiv:1812.01591*, 2018.
- [24] B. Gharesifard and S. L. Smith, “Distributed submodular maximization with limited information,” *Trans. on Control of Network Systems*, vol. 5, no. 4, 2017.
- [25] P. Turán, “On an external problem in graph theory,” *Mat. Fiz. Lapok*, vol. 48, pp. 436–452, 1941.
- [26] D. Grimsman, M. S. Ali, J. P. Hespanha, and J. R. Marden, “The impact of information in greedy submodular maximization,” *Trans. on Control of Network Systems*, 2018.
- [27] M. Conforti and G. Cornuéjols, “Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem,” *Discrete applied mathematics*, vol. 7, no. 3, 1984.

A Proof of Theorem 2

We shall prove Theorem 2 by applying Lemmas 2, 3 and 4.

We first consider when $n \equiv 1 \pmod{q}$. Combining Lemmas 2 and 3 yields that $\gamma(G) \leq 1/r$. Now let G be the graph described in (22) and we can show that $\gamma(G) \geq 1/r$. First note that for any $i < n$, $\bar{P}(G; i) = \lceil i/(r-1) \rceil$ and if $i = n$, $\bar{P}(G; i) = q$. Hence this graph is in $\mathcal{G}_{n,q}$. Let $T = \{i \mid i \leq (r-1)(q-1)\}$, and note that for all $i \leq n$, $\mathcal{N}_i \subseteq T$. Then we have:

$$f(x^{\text{opt}}) \leq f(x^{\text{opt}}, x_T^{\text{sol}}) \tag{30a}$$

$$= f(x_T^{\text{sol}}) + \sum_{i=1}^n f(x_i^{\text{opt}} \mid x_{1:i-1}^{\text{opt}}, x_T^{\text{sol}}) \tag{30b}$$

$$\leq f(x_T^{\text{sol}}) + \sum_{i=1}^n f(x_i^{\text{opt}} \mid x_{\mathcal{N}_i}^{\text{sol}}) \tag{30c}$$

$$= f(x_T^{\text{sol}}) + \sum_{j=1}^{r-1} \sum_{i=0}^{q-1} f(x_{i(r-1)+j}^{\text{opt}} \mid x_{\mathcal{N}_i}^{\text{sol}}) + f(x_n^{\text{opt}} \mid x_T^{\text{sol}}) \tag{30d}$$

$$\leq f(x_T^{\text{sol}}) + \sum_{j=1}^{r-1} \sum_{i=0}^{q-1} f(x_{i(r-1)+j}^{\text{sol}} \mid x_{\mathcal{N}_i}^{\text{sol}}) + f(x_n^{\text{sol}} \mid x_T^{\text{sol}}) \tag{30e}$$

$$= \sum_{j=1}^{r-1} f(x_{\{j, j+(r-1), \dots, j+(r-1)(q-1)\}}^{\text{sol}}) + f(x_n^{\text{sol}}, x_T^{\text{sol}}) \tag{30f}$$

$$\leq rf(x^{\text{sol}}) \tag{30g}$$

where (30a) and (30g) follow from monotonicity, (30b) and (30f) follow from telescoping sums, (30c) follows from submodularity, (30d) rearranges the sum according to the graph structure as defined in (22), and (30e) follows from the definition of the greedy algorithm as stated in (14). From here we can conclude that $\eta(n, q) = 1/r$ when $n \equiv 1 \pmod{q}$.

Now we consider the case where $n \not\equiv 1 \pmod{q}$. If $\alpha(G) > r$, then by Lemma 3, $\gamma(G) \leq 1/(r+1)$. If $\alpha(G) = r$, we suppose for the sake of contradiction that condition of Lemma 4 fails to hold. Let $q' = \arg \max_k |D_k| > 0$. Then we need $|D_k| < r$ for any $k < q'$, otherwise we can pick w be any agent in iteration $k+1$. So we can have at most $(r-1)(q'-1) + r \leq (r-1)q + 1$ agents. But the condition $n \not\equiv 1 \pmod{q}$ implies that $n > q(r-1) + 1$, contradiction. So by Lemma 4, $\gamma(G) \leq 1/(r+1)$.

Now let G be the graph described by (23). For any $i \in n$, $\overline{P}(G; i) = \lceil i/r \rceil \leq q$, hence this graph is in $\mathcal{G}_{n,q}$. Also note that this graph consists of r disjoint cliques, therefore $r = \alpha(G) = \theta(G)$. By Lemma 3, we have $\gamma(G) \geq 1/(r+1)$, hence the equality is achieved. \blacksquare

B Proof of Lemma 5

We first show the lower bound on $\gamma_\lambda(G)$. In this proof, we consider some minimum clique covering $\{K_1, K_2, \dots, K_\theta\}$ of G , where we can choose the K 's so they are disjoint. Let $K(i)$ denotes the clique containing the i node, and $\sigma_j = f(x_{K_j}^{\text{sol}})$. We will upper bound $f(x^{\text{opt}})$ in terms of the σ 's. Then we express the quantity $\sigma_j/f(x^{\text{sol}})$ as a concave function in σ_j and using convexity to derive the final lower bound.

First, we need to resolve some technicalities so we have an appropriate objective function f and action profile X .² Given $(f \in \mathcal{F}_\lambda, X)$, we can assume that $X_i = \{x_i^{\text{sol}}, x_i^{\text{opt}}\}$ without affecting the competitive ratio. We want $x_i^{\text{opt}} \neq x_i^{\text{sol}}$ for all $i \in N$, which ensures that all decisions are distinct. Suppose not and for some $i \in N$, $x_i^{\text{opt}} = x_i^{\text{sol}} = u$ for some decision u . Now we create a new decision v and transform (f, X) to $(\overline{f}, \overline{X})$ so that $\overline{x_j^{\text{sol}}} = x_j^{\text{sol}}$, $\overline{x_j^{\text{opt}}} = x_j^{\text{opt}}$ for any $j \neq i$, $\overline{x_i^{\text{sol}}} = u$ and $\overline{x_i^{\text{opt}}} = v$. And we define \overline{f} as:

$$\begin{cases} \overline{f}(u, v, A) = f(u, A) + (1-\lambda)f(u) \\ \overline{f}(u, A) = \overline{f}(v, A) = f(u, A) \\ \overline{f}(A) = f(A) \end{cases} \tag{31}$$

for any $A \subseteq S \setminus X_i$. Note that the above transformation does not affect the competitive ratio because our construction guarantees $\overline{f}(\overline{x^{\text{sol}}}) = f(x^{\text{sol}})$ and $\overline{f}(\overline{x^{\text{opt}}}) = f(x^{\text{opt}})$. Also, from some

²The readers can safely ignore this paragraph without affecting their understanding of the rest of this proof.

simple algebra:

$$\bar{f}(u | v, A) = \bar{f}(v | u, A) = (1 - \lambda)f(u) \quad (32a)$$

$$\bar{f}(y | u, v, A) = f(y | u, A) \quad (32b)$$

where $A \subseteq S \setminus X_i$, $y \in S \setminus X_i$. From (32), it is easy to verify that (\bar{f}, \bar{X}) satisfies all properties of submodular functions and has total curvature λ ; but for brevity, we will not explicitly show them here. Hence, for rest of the proof, we can safely assume that $x_i^{\text{opt}} \neq x_i^{\text{sol}}$ for all $i \in N$.

Now we bound $f(x^{\text{opt}})$ through total curvature:

$$f(x^{\text{opt}}) + (1 - \lambda) \sum_{j=1}^{\theta} \sigma_j \leq f(x^{\text{opt}}) + \sum_{j=1}^{\theta} \sum_{i \in K_j} (1 - \lambda) f(x_i^{\text{sol}}) \quad (33a)$$

$$= f(x^{\text{opt}}) + \sum_{i=1}^n (1 - \lambda) f(x_i^{\text{sol}}) \quad (33b)$$

$$\leq f(x^{\text{opt}}) + \sum_{i=1}^n f(x_i^{\text{sol}} | x_{1:i-1}^{\text{sol}}, x^{\text{opt}}) \quad (33c)$$

$$= f(x^{\text{opt}}, x^{\text{sol}}) \quad (33d)$$

where (33a) follows from submodularity, (33b) follows from the fact that K 's are disjoint, (33c) follows from total curvature, and (33d) is a telescoping sum.

Then we bound $f(x^{\text{opt}}, x^{\text{sol}})$ using properties of submodular functions and the greedy algorithm.

$$f(x^{\text{opt}}, x^{\text{sol}}) = f(x^{\text{sol}}) + \sum_{i=1}^n f(x_i^{\text{opt}} | x_{1:i-1}^{\text{opt}}, x^{\text{sol}}) \quad (34a)$$

$$\leq f(x^{\text{sol}}) + \sum_{i=1}^n f(x_i^{\text{opt}} | x_{\mathcal{N}_i}^{\text{sol}}) \quad (34b)$$

$$\leq f(x^{\text{sol}}) + \sum_{i=1}^n f(x_i^{\text{sol}} | x_{\mathcal{N}_i}^{\text{sol}}) \quad (34c)$$

$$\leq f(x^{\text{sol}}) + \sum_{i=1}^n f(x_i^{\text{sol}} | x_{\mathcal{N}_i}^{\text{sol}} \cap K(i)) \quad (34d)$$

$$= f(x^{\text{sol}}) + \sum_{j=1}^{\theta} \sum_{i \in K_j} f(x_i^{\text{sol}} | x_{\mathcal{N}_i}^{\text{sol}} \cap K_j) \quad (34e)$$

$$= f(x^{\text{sol}}) + \sum_{j=1}^{\theta} \sigma_j \quad (34f)$$

where (34a) and (34f) are telescoping sums, (34b) and (34d) follow from submodularity, (34e) follows from the fact that K 's are disjoint, and (34c) follows from the greedy algorithm as defined

in (14).

Combining (33) and (34), we have that

$$f(x^{\text{opt}}) + (1 - \lambda) \sum_{j=1}^{\theta} \sigma_j \leq f(x^{\text{sol}}) + \sum_{j=1}^{\theta} \sigma_j \quad (35)$$

Rearrange the terms yields:

$$\frac{f(x^{\text{opt}})}{f(x^{\text{sol}})} \leq 1 + \lambda \sum_{j=1}^{\theta} \frac{\sigma_j}{f(x^{\text{sol}})} \quad (36)$$

And we can upper bound the σ 's using total curvature in a manner similar to that of (33).

$$\sigma_j + \sum_{\ell \neq j} (1 - \lambda) \sigma_\ell \leq \sigma_j + \sum_{i \notin K_j} (1 - \lambda) f(x_i^{\text{sol}}) \quad (37a)$$

$$\leq \sigma_j + \sum_{i \notin K_j} f(x_i^{\text{sol}} \mid x_{1:i-1}^{\text{sol}} \cup x_{K_j}^{\text{sol}}) \quad (37b)$$

$$= f(x^{\text{sol}}) \quad (37c)$$

where (37a) follows from submodularity, (37b) follows from total curvature, and (37c) follows from telescoping sum.

Without loss of generality, we impose the normalization that $\sum_{\ell=1}^{\theta} \sigma_\ell = \theta$. Then for any $1 \leq j \leq \theta$,

$$\frac{\sigma_j}{f(x^{\text{sol}})} \leq \frac{\sigma_j}{(1 - \lambda) \sum_{\ell=1}^{\theta} \sigma_\ell + \lambda \sigma_j} = \frac{\sigma_j}{(1 - \lambda) \theta + \lambda \sigma_j} \quad (38)$$

We can denote the right hand side of (38) by the function $g(x) = x / ((1 - \lambda)x + \lambda x)$. Note that g is a concave function. We then apply the Jensen's inequality:

Lemma 6 (Jensen). *For a concave function g and values y_1, \dots, y_m in its domain, then*

$$\frac{1}{m} \sum_{i=1}^m g(y_i) \leq g\left(\frac{\sum_{i=1}^m y_i}{m}\right) \quad (39)$$

And the equality holds if and only if $y_1 = \dots = y_m$ or when g is linear.

We substitute our normalization into (38) and combine it with (36):

$$\frac{1}{\gamma_\lambda(G)} = \frac{f(x^{\text{opt}})}{f(x^{\text{sol}})} \leq 1 + \frac{\lambda \theta}{(1 - \lambda) \theta + \lambda} = \frac{\theta + \lambda}{\theta - (\theta - 1) \lambda} \quad (40)$$

Hence we derived the lower bound.

Now we show the upper bound on $\gamma_\lambda(G)$. Let I be a maximum independent set of G and $\alpha = |I|$. Define two sets of distinct decisions $U = \{u_1, u_2, \dots, u_\alpha\}$ and $V = \{v_1, v_2, \dots, v_\alpha\}$.

Consider $S = U \cup V$ and action profile determined by:

$$X_i = \begin{cases} \{u_i, v_i\} & \text{if } i \in I \\ \emptyset & \text{otherwise} \end{cases} \quad (41)$$

Define the objective function f so that for any $x \in X$ (with x interpreted as subset of S),

$$f(x) = \min(1, |x \cap U|)\lambda + |x \cap U|(1 - \lambda) + |x \cap V| \quad (42)$$

Note that through this construction, f is a submodular function and has total curvature λ . Also, f has the following properties for any $i \in I$:

1. $f(u_i) = f(v_i) = 1$.
2. $f(u_i | x_{\mathcal{N}_i}) = f(u_i)$ for any $x \in X$ because by definition, we have $\mathcal{N}_i \cap I = \emptyset$.
3. $f(v_i | B) = f(v_i)$ for any $B \subseteq S \setminus \{v_i\}$

From these properties, the agents in I are equally incentivized to pick either option. In one of the greedy solutions, the action would be the u 's. And in the optimal solution, the action would be the v 's. This results in $f(x^{\text{sol}}) \leq \lambda + \alpha(1 - \lambda)$ and $f(x^{\text{opt}}) = \alpha$, hence $\gamma_\lambda \leq \frac{\alpha - (\alpha - 1)\lambda}{\alpha}$. ■

C Proof of Theorem 3

First we note that the function $g(x) = (x - (x - 1)\lambda)/x$ is decreasing, and the upper bound in (28) is equal to $g(\alpha(G))$. The upper bound in (29) then follows from combining Lemma 2 and the upper bound in Lemma 5.

Consider the graph G described by (23); as we already argued previously in Appendix A, $r = \alpha(G) = \theta(G)$ and $G \in \mathcal{G}_{n,q}$. Hence, from the lower bound in Lemma 5, we conclude that this graph achieves the lower bound in (29), so we are done. ■