Computing Separable Functions via Gossip

Damon Mosk-Aoyama
Stanford University
damonma@cs.stanford.edu

Devavrat Shah
MIT
devavrat@mit.edu

Abstract

Motivated by applications to sensor, peer-to-peer, and ad-hoc networks, we study the problem of computing functions of values at the nodes in a network in a *totally distributed* manner. In particular, we consider separable functions, which can be written as linear combinations or products of functions of individual variables.

The main contribution of this paper is the design of a distributed algorithm for computing separable functions based on properties of exponential random variables. We bound the running time of our algorithm in terms of the running time of an information spreading algorithm used as a subroutine by the algorithm. Since we are interested in totally distributed algorithms, we consider a randomized *gossip* mechanism for information spreading as the subroutine. Combining these algorithms yields a complete and simple distributed algorithm for computing separable functions.

The second contribution of this paper is a characterization of the information spreading time of the gossip algorithm, and therefore the computation time for separable functions, in terms of the *conductance* of an appropriate stochastic matrix. Specifically, we find that for a class of graphs with small spectral gap, this time is of a smaller order than the time required to compute averages for a known iterative gossip scheme [4].

1 Introduction

The development of peer-to-peer, sensor, and ad hoc wireless networks has stimulated interest in distributed algorithms for data aggregation, in which nodes in a network compute a function of local values at the individual nodes. These networks typically do not have centralized agents for organizing computation and communication among the nodes. Furthermore, the nodes in such a network may not know the complete topology of the network, and the topology may change over time as nodes are added and other nodes fail. In light of the preceding considerations, distributed computation is of vital importance in these modern networks.

We consider the problem of computing separable functions in a distributed fashion in this paper. Given a network in which each node has a number, we seek a distributed protocol for computing the value of a separable function of the numbers at the nodes. Each node has its own estimate of the value of the function, which evolves as the protocol proceeds. Our goal is to minimize the amount of time required for all of these estimates to be close to the actual function value.

In this work, we are interested in *totally distributed* computations, in which nodes have a local view of the state of the network. To accurately estimate the value of a separable function that depends on the numbers at all of the nodes, each node must obtain information about the other nodes in the network. This is accomplished through communication between neighbors in the

network. Over the course of the protocol, the global state of the network effectively diffuses to each individual node via local communication among neighbor nodes.

More concretely, we assume that each node in the network knows only its neighbors in the network topology, and can contact any neighbor to initiate a communication. On the other hand, we assume that the nodes do not have unique identities (i.e., a node has no unique identifier that can be attached to its messages to identify the source of the messages). This constraint makes it difficult, without global coordination, to simply transmit every node's value throughout the network so that each node can identify the values at all the nodes. As such, we develop an algorithm for computing separable functions that relies on an *order- and duplicate-insensitive* statistic [14] of a set of numbers, the minimum. The algorithm is based on properties of exponential random variables, and reduces the problem of computing the value of a separable function to the problem of determining the minimum of a collection of numbers, one for each node.

This reduction leads us to study the problem of *information spreading* or *information dissemination* in a network. In this problem, each node starts with a message, and the nodes must spread the messages throughout the network using local communication so that every node eventually has every message. Because the minimum of a collection of numbers is not affected by the order in which the numbers appear, nor by the presence of duplicates of an individual number, the minimum computation required by our algorithm for computing separable functions can be performed by any information spreading algorithm. Our analysis of the algorithm for computing separable functions establishes an upper bound on its running time in terms of the running time of the information spreading algorithm it uses as a subroutine.

In view of our goal of distributed computation, we analyze a randomized *gossip* algorithm for information spreading. This algorithm consists of repeated communication between neighbors in the network, with the communication partner of a node at any time determined by a simple probabilistic choice. We provide an upper bound on the running time of the gossip algorithm for information spreading in terms of the *conductance* of a stochastic matrix that governs how nodes choose communication partners. By using the gossip algorithm to compute minima in the algorithm for computing separable functions, we obtain an algorithm for computing separable functions whose performance on certain graphs compares favorably with that of known iterative distributed algorithms [4] for computing averages in a network.

1.1 Organization

The rest of the paper is organized as follows. Section 2 presents the distributed computation problems we study and an overview of our results. In Section 3, we develop and analyze an algorithm for computing separable functions in a distributed manner. Section 4 contains an analysis of a simple randomized gossip algorithm for information spreading, which can be used as a subroutine in the algorithm for computing separable functions. In Section 5, we discuss applications of our results to particular types of graphs. Section 6 presents related work and compares our results to previous results for computing averages. Finally, we present conclusions and future directions in Section 7.

2 Preliminaries

We consider an arbitrary connected network, represented by an undirected graph $G = (V, E)$, with $|V| = n$ nodes. For notational purposes, we assume that the nodes in V are numbered arbitrarily

so that $V = \{1, \ldots, n\}$. A node, however, does not have a unique identity that can be used in a computation. Two nodes i and j can communicate with each other if (and only if) $(i, j) \in E$.

To capture some of the resource constraints in the networks in which we are interested, we impose a *transmitter gossip* constraint on node communication. Each node is allowed to contact at most one other node at a given time for communication. However, a node can be contacted by multiple nodes simultaneously.

Let 2^V denote the power set of the vertex set V (the set of all subsets of V). For an n-dimensional vector $\vec{x} \in \mathbf{R}^n$, let $x_1, \ldots, x_n$ be the components of $\vec{x}$.

Definition 1 (Separable). We say that a function $f : \mathbf{R}^n \times 2^V \to \mathbf{R}$ is *separable* if there exist functions $f_1, \ldots, f_n$ such that, for all $S \subseteq V$,

$$f(\vec{x}, S) = \sum_{i \in S} f_i(x_i). \tag{1}$$

Goal. Let $\mathcal{F}$ be the class of separable functions f for which $f_i(x) \geq 1$ for all $x \in \mathbf{R}$ and $i = 1, \ldots, n$. Given a function $f \in \mathcal{F}$, and a vector $\vec{x}$ containing initial values x_i for all the nodes, the nodes in the network are to compute the value $f(\vec{x}, V)$ by a distributed computation, using repeated communication between nodes.

Note. Consider a function g for which there exist functions $g_1, \ldots, g_n$ satisfying, for all $S \subseteq V$, the condition $g(\vec{x}, S) = \prod_{i \in S} g_i(x_i)$ in lieu of (1). Our algorithm for computing separable functions can be used to compute the function $f = \log_b g$. The condition $f_i(x) \geq 1$ corresponds to $g_i(x) \geq b$ in this case. This lower bound of 1 on $f_i(x)$ is arbitrary, although our algorithm does require the terms $f_i(x_i)$ in the sum to be positive.

Before proceeding further, we list some practical situations where the distributed computation of separable functions arises naturally. By definition, the sum of a set of numbers is a separable function.

(1) *Summation.* Let the value at each node be $x_i = 1$. Then, the sum of the values is the number of nodes in the network.

(2) *Averaging.* According to Definition 1, the average of a set of numbers is not a separable function. However, the nodes can estimate the separable function $\sum_{i=1}^{n} x_i$ and n separately, and use the ratio between these two estimates as an estimate of the mean of the numbers.

Suppose the values at the nodes are measurements of a quantity of interest. Then, the average provides an unbiased maximum likelihood estimate of the measured quantity. For example, if the nodes are temperature sensors, then the average of the sensed values at the nodes gives a good estimate of the ambient temperature.

For more sophisticated applications of a distributed averaging (summation) algorithm, we refer the reader to [12] and [13]. Averaging is used for the distributed computation of the top k eigenvectors of a graph in [12], while in [13] averaging is used in a throughput-optimal distributed scheduling algorithm in a wireless network.

Time model. In a distributed computation, a time model determines when nodes communicate with each other. We consider both a synchronous and an asynchronous time model in this paper. The two models are described as follows.

(1) *Synchronous time model:* Time is slotted commonly across all nodes in the network. In any time slot, each node may contact one of its neighbors according to a random choice that is independent of the choices made by the other nodes. The simultaneous communication between the nodes satisfies the gossip constraint.

(2) *Asynchronous time model:* Each node has a clock which ticks at the times of a rate 1 Poisson process. Equivalently, a common clock ticks according to a rate n Poisson process at times $Z_k, k \geq 1$, where $\{Z_{k+1} - Z_k\}$ are i.i.d. exponential random variables of rate n. On clock tick k, one of the n nodes, say I_k, is chosen uniformly at random. We consider this global clock tick to be a tick of the clock at node I_k. When a node's clock ticks, it contacts one of its neighbors at random. In this model, time is discretized according to clock ticks. On average, there are n clock ticks per one unit of absolute time.

In this paper, we measure the running times of algorithms in absolute time, which is the number of time slots in the synchronous model. A precise relationship between clock ticks and absolute time in the asynchronous model is provided by Corollary 2, which is stated below. Corollary 2 is a consequence of the following lemma, which follows from Cramér's Theorem (see [6], pp. 30, 35) and properties of exponential random variables.

Lemma 1. *For any $k \geq 1$, let $Y_1, \ldots, Y_k$ be i.i.d. exponential random variables with rate λ. Let $R_k = \frac{1}{k}\sum_{i=1}^{k} Y_i$. Then, for any $\epsilon \in (0, 1/2)$,*

$$\Pr\left(\left|R_k - \frac{1}{\lambda}\right| \geq \frac{\epsilon}{\lambda}\right) \quad \leq \quad 2\exp\left(-\frac{\epsilon^2 k}{3}\right). \tag{2}$$

Corollary 2. *For $k \geq 1$, $E[Z_k] = k/n$. Further, for any $\epsilon \in (0, 1/2)$,*

$$\Pr\left(\left|Z_k - \frac{k}{n}\right| \geq \frac{\epsilon k}{n}\right) \quad \leq \quad 2\exp\left(-\frac{\epsilon^2 k}{3}\right). \tag{3}$$

To measure the performance of an algorithm $\mathcal{C}$ that computes (an estimate of) $f(\vec{x}, V) = \sum_{i=1}^{n} f_i(x_i)$ at each node, we define the following quantity. Let $\hat{y}_i(t)$ be the estimate of $f(\vec{x}, V)$ at node i at time t.

Definition 2. For any $\epsilon > 0$ and $\delta \in (0, 1)$, the (ϵ, δ)-computing time of $\mathcal{C}$, denoted $T_{\mathcal{C}}^{\text{cmp}}(\epsilon, \delta)$, is

$$T_{\mathcal{C}}^{\text{cmp}}(\epsilon, \delta) = \sup_{f \in \mathcal{F}} \sup_{\vec{x} \in \mathbf{R}^n} \inf\left\{t : \Pr\left(\cup_{i=1}^{n} \{\hat{y}_i(t) \notin [(1-\epsilon)f(\vec{x}, V), (1+\epsilon)f(\vec{x}, V)]\}\right) \leq \delta\right\}.$$

As noted before, our algorithm for computing separable functions is based on a reduction to the problem of information spreading, which is described as follows. Suppose that, for $i = 1, \ldots, n$, node i has the one message m_i. The task of information spreading is to disseminate all n messages to all n nodes via a sequence of local communications between neighbors in the graph. In any single communication between two nodes, each node can transmit to its communication partner any of the messages that it currently holds. We assume that the data transmitted in a communication must be a set of messages, and therefore cannot be arbitrary information.

Consider an information spreading algorithm $\mathcal{D}$, which specifies how nodes communicate. For each node $i \in V$, let $S_i(t)$ denote the set of nodes that have the message m_i at time t.

Definition 3. For any $\delta \in (0,1)$, the δ-information-spreading time of the algorithm $\mathcal{D}$, denoted $T^{\text{spr}}_{\mathcal{D}}(\delta)$, is

$$T^{\text{spr}}_{\mathcal{D}}(\delta) = \inf\left\{t : \Pr\left(\cup_{i=1}^{n}\{S_i(t) \neq V\}\right) \leq \delta\right\}.$$

In our analysis of the gossip algorithm for information spreading, we assume that when two nodes communicate, one can send all of its messages to the other in a single communication. This rather unrealistic assumption of *infinite* link capacity is merely for convenience, as it provides a simpler analytical characterization of $T^{\text{cmp}}_{\mathcal{C}}(\epsilon,\delta)$ in terms of $T^{\text{spr}}_{\mathcal{D}}(\delta)$. Our algorithm for computing separable functions requires only links of unit capacity.

2.1 Our contribution

The main contribution of this paper is the design of a distributed algorithm to compute separable functions of node values in an arbitrary connected network. Our algorithm is randomized and is based on the following property of the exponential distribution.

Property 3. *Consider n independent random variables $W_1,\ldots,W_n$, where, for $i = 1,\ldots,n$, the distribution of W_i is exponential with rate λ_i. Let $\bar{W}$ be the minimum of $W_1,\ldots,W_n$. Then, $\bar{W}$ is distributed as an exponential random variable of rate $\lambda = \sum_{i=1}^{n}\lambda_i$.*

Our algorithm uses an information spreading algorithm as a subroutine, and as a result its running time is a function of the running time of the information spreading algorithm it uses. The faster the information spreading algorithm is, the better our algorithm performs. Specifically, the following result provides an upper bound on the (ϵ, δ)-computing time of the algorithm.

Theorem 4. *Given an information spreading algorithm $\mathcal{D}$ with δ-spreading time $T^{spr}_{\mathcal{D}}(\delta)$ for $\delta \in (0,1)$, there exists an algorithm $\mathcal{A}$ for computing separable functions $f \in \mathcal{F}$ such that, for any $\epsilon \in (0,1)$ and $\delta \in (0,1)$,*

$$T^{cmp}_{\mathcal{A}}(\epsilon,\delta) = O\left(\epsilon^{-2}(1+\log\delta^{-1})T^{spr}_{\mathcal{D}}(\delta/2)\right).$$

Motivated by our interest in decentralized algorithms, we analyze a simple randomized gossip algorithm for information spreading. When node i initiates a communication, it contacts each node $j \neq i$ with probability P_{ij}. With probability P_{ii}, it does not contact another node. The $n \times n$ matrix $P = [P_{ij}]$ characterizes the algorithm; each matrix P gives rise to an information spreading algorithm $\mathcal{P}$. We assume that P is stochastic, and that $P_{ij} = 0$ if $(i,j) \notin E$, as nodes that are not neighbors in the graph cannot communicate with each other. Section 4 describes the data transmitted between two nodes when they communicate.

We obtain an upper bound on the δ-information-spreading time of this gossip algorithm in terms of the *conductance* of the matrix P, which is defined as follows.

Definition 4 (Conductance). For a stochastic matrix P, the conductance of P, denoted $\Phi(P)$, is

$$\Phi(P) = \min_{S\subset V,\, 0<|S|\leq n/2} \frac{\sum_{i\in S, j\notin S} P_{ij}}{|S|}.$$

In general, the above definition of conductance is not the same as the classical definition [18]. However, we restrict our attention in this paper to symmetric matrices P. When P is symmetric, the two definitions are equivalent. Note that the definition of conductance implies that $\Phi(P) \leq 1$.

Theorem 5. *Consider any stochastic and symmetric matrix P such that if $(i,j) \notin E$, then $P_{ij} = 0$. There exists an information dissemination algorithm $\mathcal{P}$ such that, for any $\delta \in (0,1)$,*

$$T_{\mathcal{P}}^{spr}(\delta) = O\left(\frac{\log n + \log \delta^{-1}}{\Phi(P)}\right).$$

Note. The results of Theorems 4 and 5 hold for both the synchronous and asynchronous models. Recall that the quantities $T_{\mathcal{C}}^{\text{cmp}}(\epsilon, \delta)$ and $T_{\mathcal{D}}^{\text{spr}}(\delta)$ are defined with respect to absolute time in both models.

3 Function Computation

In this section, we describe our algorithm for computing the value $y = f(\vec{x}, V) = \sum_{i=1}^{n} f_i(x_i)$ of the separable function f, where $f_i(x_i) \geq 1$. For simplicity of notation, let $y_i = f_i(x_i)$. Given x_i, each node can compute y_i on its own. Next, the nodes use the algorithm shown in Figure 1 to compute estimates $\hat{y}_i$ of $y = \sum_{i=1}^{n} y_i$. The quantity r is a parameter to be chosen later.

Algorithm COMP

0. Initially, for $i = 1, \ldots, n$, node i has the value $y_i \geq 1$.

1. Each node i generates r independent random numbers $W_1^i, \ldots, W_r^i$, where the distribution of each W_ℓ^i is exponential with rate y_i (i.e., with mean $1/y_i$).

2. Each node i computes, for $\ell = 1, \ldots, r$, an estimate $\hat{W}_\ell^i$ of the minimum $\bar{W}_\ell = \min_{i=1}^{n} W_\ell^i$. This computation can be done using an information spreading algorithm as described below.

3. Each node i computes $\hat{y}_i = \frac{r}{\sum_{\ell=1}^{r} \hat{W}_\ell^i}$ as its estimate of $\sum_{i=1}^{n} y_i$.

Figure 1: An algorithm for computing separable functions.

We describe how the minimum is computed as required by step **2** of the algorithm in Section 3.1. The running time of the algorithm COMP depends on the running time of the algorithm used to compute the minimum.

Now, we show that COMP effectively estimates the function value y when the estimates $\hat{W}_\ell^i$ are all correct by providing a lower bound on the conditional probability that the estimates produced by COMP are all within a $1 \pm \epsilon$ factor of y. The following lemma is proven in Appendix A.

Lemma 6. *Let $y_1, \ldots, y_n$ be real numbers (with $y_i \geq 1$ for $i = 1, \ldots, n$), $y = \sum_{i=1}^{n} y_i$, and $\bar{W} = (\bar{W}_1, \ldots, \bar{W}_r)$, where the $\bar{W}_\ell$ are as defined in the algorithm COMP. For any node i, let $\hat{W}^i = (\hat{W}_1^i, \ldots, \hat{W}_r^i)$, and let $\hat{y}_i$ be the estimate of y obtained by node i in COMP. For any $\epsilon \in (0, 1/2)$,*

$$\Pr\left(\cup_{i=1}^{n} \{|\hat{y}_i - y| > 2\epsilon y\} \mid \forall i \in V,\, \hat{W}^i = \bar{W}\right) \leq 2\exp\left(-\frac{\epsilon^2 r}{3}\right).$$

3.1 Computing minimum via information spreading

We now elaborate on step **2** of the algorithm COMP. Each node i in the graph starts this step with a vector $W^i = (W^i_1, \ldots, W^i_r)$, and the nodes seek the vector $\bar{W} = (\bar{W}_1, \ldots, \bar{W}_r)$, where $\bar{W}_\ell = \min_{i=1}^n W^i_\ell$. In the information spreading problem, each node i has a message m_i, and the nodes are to transmit messages across the links until every node has every message.

If all link capacities are infinite (i.e., in one time unit, a node can send an arbitrary amount of information to another node), then an information spreading algorithm $\mathcal{D}$ can be used directly to compute the minimum vector $\bar{W}$. To see this, let the message m_i at the node i be the vector W^i, and then apply the information spreading algorithm to disseminate the vectors. Once every node has every message (vector), each node can compute $\bar{W}$ as the component-wise minimum of all the vectors. This implies that the running time of the resulting algorithm for computing $\bar{W}$ is the same as that of the information spreading algorithm.

The assumption of infinite link capacities allows a node to transmit an arbitrary number of vectors W^i to a neighbor in one time unit. A simple modification to the information spreading algorithm, however, yields an algorithm for computing the minimum vector $\bar{W}$ using links of capacity r. To this end, each node i maintains a single r-dimensional vector $w^i(t)$ that evolves in time, starting with $w^i(0) = W^i$.

Suppose that, in the information dissemination algorithm, node j transmits the messages (vectors) $W^{i_1}, \ldots, W^{i_c}$ to node i at time t. Then, in the minimum computation algorithm, j sends to i the r quantities $w_1, \ldots, w_r$, where $w_\ell = \min_{u=1}^c W^{i_u}_\ell$. The node i sets $w^i_\ell(t^+) = \min(w^i_\ell(t^-), w_\ell)$ for $\ell = 1, \ldots, r$, where t^- and t^+ denote the times immediately before and after, respectively, the communication. At any time t, we will have $w^i(t) = \bar{W}$ for all nodes $i \in V$ if, in the information spreading algorithm, every node i has all the vectors $W^1, \ldots, W^n$ at the same time t. In this way, we obtain an algorithm for computing the minimum vector $\bar{W}$ that uses links of capacity r and runs in the same amount of time as the information spreading algorithm.

An alternative to using links of capacity r in the computation of $\bar{W}$ is to make the time slot r times larger, and impose a unit capacity on all the links. Now, a node transmits the numbers $w_1, \ldots, w_r$ to its communication partner over a period of r time slots, and as a result the running time of the algorithm for computing $\bar{W}$ becomes greater than the running time of the information spreading algorithm by a factor of r. This discussion leads to the following lemma.

Lemma 7. *Suppose that the COMP algorithm is implemented using an information spreading algorithm $\mathcal{D}$ as described above. For any $\delta \in (0,1)$, let $\hat{W}^i(t_m)$ denote the estimate of $\bar{W}$ at node i at time $t_m = rT^{spr}_{\mathcal{D}}(\delta)$. With probability at least $1-\delta$, $\hat{W}^i(t_m) = \bar{W}$ for all nodes $i \in V$.*

Theorem 4, which follows from Lemmas 6 and 7 for the parameter choice $r = \Theta(\epsilon^{-2}(1+\log \delta^{-1}))$, is proven in Appendix B. Note that the amount of data communicated between nodes during the algorithm COMP depends on the values of the exponential random variables generated by the nodes. Since the nodes compute minima of these variables, we are interested in a probabilistic lower bound on the values of these variables (e.g., suppose that the nodes transmit the values $1/W^i_\ell$ when computing the minimum $\bar{W}_\ell = 1/\max_{i=1}^n\{1/W^i_\ell\}$). To this end, we use the fact that each $\bar{W}_\ell$ is an exponential random variable with rate y to obtain that, for any constant $c > 1$, the probability that any of the minimum values $\bar{W}_\ell$ is less than $1/B$ (i.e., any of the inverse values $1/W^i_\ell$ is greater than B) is at most δ/c, where B is proportional to cry/δ.

4 Information spreading

In this section, we analyze a randomized gossip algorithm for information spreading. The method by which nodes choose partners to contact to initiate a communication and the data transmitted during the communication are the same for both time models. The models differ in when nodes contact each other: in the asynchronous model, only one node can start a communication at any time, while in the synchronous model all the nodes can communicate in each time slot.

Information spreading algorithm. When a node initiates a communication at (absolute) time t, it chooses another node to contact at random as described in Section 2.1. We now specify the gossip protocol, which determines the data transmitted during the communication.

Let $M_v(t)$ denote the set of messages node v has at time t. Initially, $M_v(0) = \{m_v\}$ for all $v \in V$. Suppose that node i contacts node j at time t, and let t^- and t^+ denote the times immediately before and after, respectively, the communication occurs. We assume that the data transmitted between the two nodes conform to the *pull mechanism*: node j sends information to node i, but i does not send information to j. Specifically, j sends all of the messages it has to i, so that

$$M_i(t^+) = M_i(t^-) \cup M_j(t^-).$$

This algorithm is simple, distributed, and satisfies the transmitter gossip constraint. We now present analysis of its information spreading time in the two time models to prove Theorem 5. To this end, for any $i \in V$, let $S_i(t) \subseteq V$ denote the set of nodes that have the message m_i after any communication events that occur at absolute time t (communication events occur on a global clock tick in the asynchronous time model, and in each time slot in the synchronous time model). At the start of the algorithm, $S_i(0) = \{i\}$.

4.1 Asynchronous model

As described earlier, the global clock ticks according to a Poisson process of rate n, and on a tick one of the n nodes is chosen uniformly at random. This node initiates a communication, so the times at which the communication events occur correspond to the ticks of the clock. On any clock tick, at most one node can receive messages by communicating with another node.

Let $k \geq 0$ denote the index of a clock tick. Initially, $k = 0$, and the corresponding absolute time is 0. For simplicity of notation, we identify the time at which a clock tick occurs with its index, so that $S_v(k)$ denotes the set of nodes that have the message m_v at the end of clock tick k. The following lemma, which is proven in Appendix C, provides a bound on the number of clock ticks required for every node to receive every message.

Lemma 8. *For any $\delta \in (0, 1)$, define*

$$K(\delta) = \inf\{k \geq 0 : \Pr(\cup_{i=1}^{n}\{S_i(k) \neq V\}) \leq \delta\}.$$

Then,

$$K(\delta) = O\left(n\frac{\log n + \log \delta^{-1}}{\Phi(P)}\right).$$

To extend the bound in Lemma 8 to absolute time, observe that Corollary 2 implies that the probability that $\kappa = K(\delta/3) + 27\ln(3/\delta) = O(n(\log n + \log \delta^{-1})/\Phi(P))$ clock ticks do not occur in absolute time $(4/3)\kappa/n = O((\log n + \log \delta^{-1})/\Phi(P))$ is at most $2\delta/3$. Applying the union bound now establishes the upper bound in Theorem 5 for the asynchronous time model.

4.2 Synchronous model

In the synchronous time model, in each time slot every node contacts a neighbor to receive messages. Thus, n communication events may occur simultaneously. Recall that absolute time is measured in rounds or time slots in the synchronous model.

The analysis of the gossip algorithm for information spreading in the synchronous model is very similar to the analysis for the asynchronous model. In Appendix D, we sketch a proof of the time bound in Theorem 5, $T_{\mathcal{P}}^{\text{spr}}(\delta) = O((\log n + \log \delta^{-1})/\Phi(P))$, for the synchronous time model.

5 Applications

We study here the application of our preceding results to several types of graphs. In particular, we consider complete graphs, constant-degree expander graphs, and grid graphs. For each of these graphs, we study the δ-information-spreading time $T_{\mathcal{P}}^{\text{spr}}(\delta)$, where $\mathcal{P}$ is the gossip algorithm defined by a symmetric matrix P that assigns equal probability to each of the neighbors of any node. Specifically, the probability P_{ij} that a node i contacts a node $j \neq i$ when i becomes active is $1/\Delta$, where Δ is the maximum degree of the graph, and $P_{ii} = 1 - d_i/\Delta$, where d_i is the degree of i. Recall from Theorem 4 that each information dissemination algorithm $\mathcal{P}$ can be used to compute separable functions, with the running time of the resulting algorithm being a function of $T_{\mathcal{P}}^{\text{spr}}(\delta)$.

5.1 Complete graph

On a complete graph, the transition matrix P has $P_{ii} = 0$ for $i = 1, \ldots, n$, and $P_{ij} = 1/(n-1)$ for $j \neq i$. This regular structure allows us to directly evaluate the conductance of P, which is $\Phi(P) \approx 1/2$. This implies that the (ϵ, δ)-computing time of the algorithm for computing separable functions based on P is $O(\epsilon^{-2}(1 + \log \delta^{-1})(\log n + \log \delta^{-1}))$. Thus, for a constant $\epsilon \in (0, 1)$ and $\delta = 1/n$, the computation time scales as $O(\log^2 n)$.

5.2 Expander graph

Expander graphs have been used for numerous applications, and explicit constructions are known for constant-degree expanders [17]. We consider here an undirected graph in which the maximum degree of any vertex, Δ, is a constant. Suppose that the edge expansion of the graph is

$$\min_{S \subset V,\, 0 < |S| \leq n/2} \frac{|C(S, S^c)|}{|S|} = \alpha,$$

where $C(S, S^c)$ is the set of edges in the cut (S, S^c), and $\alpha > 0$ is a constant. The transition matrix P satisfies $P_{ij} = 1/\Delta$ for all $i \neq j$, from which we obtain $\Phi(P) \geq \alpha/\Delta$. When α and Δ are constants, this leads to a similar conclusion as in the case of the complete graph: for any constant $\epsilon \in (0, 1)$ and $\delta = 1/n$, the computation time is $O(\log^2 n)$.

5.3 Grid

We now consider a d-dimensional grid graph on n nodes, where $c = n^{1/d}$ is an integer. Each node in the grid can be represented as a d-dimensional vector $a = (a_i)$, where $a_i \in \{1, \ldots, c\}$ for $1 \leq i \leq d$. There is one node for each distinct vector of this type, and so the total number of nodes in the

graph is $c^d = (n^{1/d})^d = n$. For any two nodes a and b, there is an edge (a, b) in the graph if and only if, for some $i \in \{1, \ldots, d\}$, $|a_i - b_i| = 1$, and $a_j = b_j$ for all $j \neq i$.

In [1], it is shown that the isoperimetric number of this grid graph is

$$\min_{S \subset V,\, 0<|S| \leq n/2} \frac{|C(S, S^c)|}{|S|} = \Theta\left(\frac{1}{c}\right) = \Theta\left(\frac{1}{n^{1/d}}\right).$$

By the definition of the edge set, the maximum degree of a node in the graph is $2d$. This means that $P_{ij} = 1/(2d)$ for $i \neq j$, and it follows that $\Phi(P) = \Omega\left(\frac{1}{dn^{1/d}}\right)$. Hence, for any $\epsilon \in (0, 1)$ and $\delta \in (0, 1)$, the (ϵ, δ)-computing time of the algorithm for computing separable functions is $O(\epsilon^{-2}(1 + \log \delta^{-1})(\log n + \log \delta^{-1})dn^{1/d})$.

6 Related Work

In this section, we present a brief summary of related work and compare our results with some previous results on distributed computation. Algorithms for computing the number of distinct elements in a multiset or data stream [7, 2] can be adapted to compute separable functions using information spreading [5]. We are not aware, however, of a previous analysis of the amount of time required for these algorithms to achieve a certain accuracy in the estimates of the function value when the computation is totally distributed (i.e., when nodes do not have unique identities). These adapted algorithms require the nodes in the network to make use of a common hash function. In addition, the discreteness of the counting problem makes the resulting algorithms for computing separable functions suitable only for functions in which the terms in the sum are integers. Our algorithm is simpler than these algorithms, and can compute functions with non-integer terms.

There has been a lot of work on the distributed computation of averages, a special case of the problem of reaching agreement or consensus among processors via a distributed computation. Distributed algorithms for reaching consensus under appropriate conditions are known [19, 20, 3]. Recently, Kempe, Dobra, and Gehrke [11] showed the existence of an averaging algorithm with the optimal averaging time of $\Theta(\log n + \log \epsilon^{-1} + \log \delta^{-1})$ for a complete graph. In [4], Boyd et al. analyzed a large class of iterative averaging algorithms on arbitrary graphs, and found the averaging time to be related to the mixing time of a random walk related to the algorithm.

On the topic of information spreading, the results of [9] established that when the graph is complete and communication partners are chosen uniformly at random, the information spreading time of the gossip algorithm is $\Theta(\log n)$ for $\delta = 1/n$. In this work, we have provided an analysis of the information spreading time of the gossip algorithm for the more general setting of arbitrary graphs and non-uniform random choices. For other related results, we refer the reader to [15, 16, 10, 11]. We take note of the somewhat related recent work of Ganesh, Massoulie, and Towsley [8] on the spread of epidemics in a network.

We now briefly contrast the performance of our algorithm for computing separable functions with that of the iterative averaging algorithm in [4]. When our algorithm is used to compute the average of a set of numbers (by estimating the sum of the numbers and the number of nodes in the graph) on a d-dimensional grid graph, it follows from the analysis in Section 5.3 that the amount of time required to ensure the estimate is within a $1 \pm \epsilon$ factor of the average with probability at least $1 - \delta$ is $O(\epsilon^{-2}(1 + \log \delta^{-1})(\log n + \log \epsilon^{-1})dn^{1/d})$ for any $\epsilon \in (0, 1)$ and $\delta \in (0, 1)$. So, for a constant $\epsilon \in (0, 1)$ and $\delta = 1/n$, the computation time scales as $O(dn^{1/d} \log^2 n)$ with the size of the graph, n. The algorithm in [4] requires $\Omega(n^{2/d} \log n)$ time for this computation. Hence, the running time

of our algorithm is (for fixed d, and up to logarithmic factors) the *square root* of the runnning time of the iterative algorithm! This relationship holds on other graphs for which the spectral gap is proportional to the square of the conductance.

7 Conclusions and Future Work

In this paper, we presented a novel algorithm for computing separable functions in a totally distributed manner. The algorithm is based on properties of exponential random variables, and the fact that the minimum of a collection of numbers is an order- and duplicate-insensitive statistic.

Operationally, our algorithm utilizes an information spreading mechanism as a subroutine. This led us to the analysis of a randomized gossip mechanism for information spreading. We obtained an upper bound on the information spreading time of this algorithm in terms of the conductance of a matrix that characterizes the algorithm.

In addition to computing separable functions, our algorithm improves the computation time for the canonical task of averaging. For example, on graphs such as paths, rings, and grids, the performance of our algorithm is of a smaller order than that of a known iterative algorithm.

We believe that our algorithm will lead to the following totally distributed computations: (1) an approximation algorithm for linear programming; (2) an approximation algorithm for concave maximization with linear constraints; and (3) a "packet marking" mechanism in the Internet. These areas, in which summation is a key subroutine, will be topics of our future research.

Acknowledgments

We thank Ashish Goel for useful discussions and suggestions.

References

[1] M. C. Azizoğlu and Ö. Eğecioğlu. The isoperimetric number of d-dimensional k-ary arrays. *International Journal of Foundations of Computer Science*, 10(3):289–300, 1999.

[2] Z. Bar-Yossef, T. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan. Counting distinct elements in a data stream. In *Proceedings of RANDOM 2002*, pages 1–10, 2002.

[3] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Prentice Hall, 1989.

[4] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Gossip algorithms: Design, analysis and applications. In *Proceedings of IEEE INFOCOM 2005*, pages 1653–1664, 2005.

[5] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *Proceedings of the 20th International Conference on Data Engineering*, pages 449–460, 2004.

[6] A. Dembo and O. Zeitouni. *Large Deviations Techniques and Applications.* Springer, second edition, 1998.

[7] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31(2):182–209, 1985.

[8] A. Ganesh, L. Massoulie, and D. Towsley. The effect of network topology on the spread of epidemics. In *Proceedings of IEEE INFOCOM 2005*, pages 1455–1466, 2005.

[9] R. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking. Randomized rumor spreading. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, pages 565–574, 2000.

[10] D. Kempe and J. Kleinberg. Protocols and impossibility results for gossip-based communication mechanisms. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 471–480, 2002.

[11] D. Kempe, J. Kleinberg, and A. Demers. Spatial gossip and resource location protocols. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 163–172, 2001.

[12] D. Kempe and F. McSherry. A decentralized algorithm for spectral analaysis. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 561–568, 2004.

[13] E. Modiano, D. Shah, and G. Zussman. Maximizing throughput in wireless networks via gossip. *Submitted*, 2005.

[14] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pages 250–262, 2004.

[15] B. Pittel. On spreading a rumor. *SIAM Journal of Applied Mathematics*, 47(1):213–223, 1987.

[16] R. Ravi. Rapid rumor ramification: Approximating the minimum broadcast time. In *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science*, pages 202–213, 1994.

[17] O. Reingold, S. Vadhan, and A. Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, pages 3–13, 2000.

[18] A. Sinclair. *Algorithms for Random Generation and Counting: A Markov Chain Approach*. Birkhäuser, Boston, 1993.

[19] J. N. Tsitsiklis. *Problems in Decentralized Decision Making and Computation*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1984.

[20] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986.

Appendix

A Proof of Lemma 6

Observe that the estimate $\hat{y}_i$ of y at node i is a function of r and $\hat{W}^i$. Under the hypothesis that $\hat{W}^i = \bar{W}$ for all nodes $i \in V$, all nodes produce the same estimate $\hat{y} = \hat{y}_i$ of y. This estimate is $\hat{y} = r\left(\sum_{\ell=1}^{r} \bar{W}_\ell\right)^{-1}$, and so $\hat{y}^{-1} = \left(\sum_{\ell=1}^{r} \bar{W}_\ell\right) r^{-1}$.

Property 3 implies that each of the random variables $\bar{W}_1, \ldots, \bar{W}_r$ has an exponential distribution with rate y. From Lemma 1, it follows that for any $\epsilon \in (0, 1/2)$,

$$\Pr\left(\left|\hat{y}^{-1} - \frac{1}{y}\right| > \frac{\epsilon}{y} \;\middle|\; \forall i \in V,\, \hat{W}^i = \bar{W}\right) \;\leq\; 2\exp\left(-\frac{\epsilon^2 r}{3}\right). \tag{4}$$

This inequality bounds the conditional probability of the event $\{\hat{y}^{-1} \notin [(1-\epsilon)y^{-1}, (1+\epsilon)y^{-1}]\}$, which is equivalent to the event $\{\hat{y} \notin [(1+\epsilon)^{-1}y, (1-\epsilon)^{-1}y]\}$. Now, for $\epsilon \in (0, 1/2)$,

$$(1-\epsilon)^{-1} \;\in\; [1+\epsilon, 1+2\epsilon]\,, \qquad (1+\epsilon)^{-1} \;\in\; [1-\epsilon, 1-2\epsilon/3]\,. \tag{5}$$

Applying the inequalities in (4) and (5), we conclude that for $\epsilon \in (0, 1/2)$,

$$\Pr\left(|\hat{y} - y| > 2\epsilon y \mid \forall i \in V,\, \hat{W}^i = \bar{W}\right) \;\leq\; 2\exp\left(-\frac{\epsilon^2 r}{3}\right). \tag{6}$$

Noting that the event $\cup_{i=1}^{n}\{|\hat{y}_i - y| > 2\epsilon y\}$ is equivalent to the event $\{|\hat{y} - y| > 2\epsilon y\}$ when $\hat{W}^i = \bar{W}$ for all nodes i completes the proof of Lemma 6.

B Proof of Theorem 4

Consider using an information spreading algorithm $\mathcal{D}$ with δ-spreading time $T_{\mathcal{D}}^{\mathrm{spr}}(\delta)$ for $\delta \in (0,1)$ as the subroutine in the COMP algorithm. By Lemma 7, for any $\delta \in (0,1)$, the probability that $\hat{W}^i \neq \bar{W}$ for any node i after $O(rT_{\mathcal{D}}^{\mathrm{spr}}(\delta/2))$ time is at most $\delta/2$. In the case that $\hat{W}^i = \bar{W}$ for all nodes i, for any $\epsilon \in (0,1)$, by choosing $r \geq 12\epsilon^{-2}\log(4\delta^{-1})$ we obtain from Lemma 6 that

$$\Pr\left(\cup_{i=1}^{n}\left\{\hat{y}_i \notin [(1-\epsilon)y, (1+\epsilon)y]\right\} \mid \forall i \in V,\, \hat{W}^i = \bar{W}\right) \;\leq\; \delta/2. \tag{7}$$

Recall that $T_{COMP}^{\mathrm{cmp}}(\epsilon, \delta)$ is the time by which, under the algorithm COMP, all the nodes have an estimate of the function value y within a factor of $1 \pm \epsilon$ with probability at least $1 - \delta$. By a straightforward union bound of events and (7), we conclude that, for any $\epsilon \in (0,1)$ and $\delta \in (0,1)$,

$$T_{COMP}^{\mathrm{cmp}}(\epsilon, \delta) = O\left(\epsilon^{-2}(1 + \log\delta^{-1})T_{\mathcal{D}}^{\mathrm{spr}}(\delta/2)\right).$$

This completes the proof of Theorem 4.

C Proof of Lemma 8

Fix any node $v \in V$. We study the evolution of the size of the set $S_v(k)$. For simplicity of notation, we drop the subscript v, and write $S(k)$ to denote $S_v(k)$.

Under the gossip algorithm, after clock tick $k+1$, we have either $|S(k+1)| = |S(k)|$ or $|S(k+1)| = |S(k)|+1$. Further, the size increases if a node $j \notin S(k)$ contacts a node $i \in S(k)$. For each such pair of nodes i, j, the probability that this occurs on clock tick $k+1$ is P_{ji}/n. Hence,

$$E[|S(k+1)| - |S_k| \mid S(k)] = \sum_{i \in S(k), j \notin S(k)} \frac{P_{ji}}{n}. \tag{8}$$

By the symmetry of P,

$$E[|S(k+1)| \mid S(k)] = |S(k)| \left(1 + \frac{\sum_{i \in S(k), j \notin S(k)} P_{ij}}{n|S(k)|}\right). \tag{9}$$

Now, we divide the execution of the algorithm into two phases based on the size of the set $S(k)$. In the first phase, $|S(k)| \leq n/2$, and in the second phase $|S(k)| > n/2$. When $|S(k)| \leq n/2$, it follows from (9) and the definition of the conductance $\Phi(P)$ of P that

$$E[|S(k+1)| \mid S(k)] \geq |S(k)| \left(1 + \hat{\Phi}\right), \tag{10}$$

where $\hat{\Phi} = \frac{\Phi(P)}{n}$.

Let $Z(k) = |S(k)| - (1+\hat{\Phi})^k$. Define the stopping time $L = \inf\{k : |S(k)| > n/2\}$, and $L \wedge k = \min(L, k)$. The lower bound in (10) on the conditional expectation of $|S(k+1)|$ implies that $Z(L \wedge k)$ is a submartingale. To see this, first observe that if $|S(k)| > n/2$, then $L \wedge (k+1) = L \wedge k = T$, and thus $E[Z(L \wedge (k+1)) \mid S(L \wedge k)] = Z(L \wedge k)$. In the case that $|S(k)| \leq n/2$, we apply the inequality in (10) and the fact that $L \wedge (k+1) = (L \wedge k) + 1$ to verify the submartingale condition.

$$\begin{aligned} E[Z(L \wedge (k+1)) \mid S(L \wedge k)] &= E[|S(L \wedge (k+1))| \mid S(L \wedge k)] \\ &\quad - E\left[\left(1+\hat{\Phi}\right)^{L \wedge (k+1)} \;\middle|\; S(L \wedge k)\right] \\ &\geq \left(1+\hat{\Phi}\right)|S(L \wedge k)| - \left(1+\hat{\Phi}\right)^{(L \wedge k)+1} \\ &= \left(1+\hat{\Phi}\right) Z(L \wedge k) \end{aligned}$$

Since $Z(L \wedge k)$ is a submartingale, we have the inequality $E[Z(L \wedge 0)] \leq E[Z(L \wedge k)]$ for any $k > 0$, which implies that $E\left[\left(1+\hat{\Phi}\right)^{L \wedge k}\right] \leq E[|S(L \wedge k)|]$ because $Z(L \wedge 0) = Z(0) = 0$. Using the fact that the set $S(k)$ can contain at most the n nodes in the graph, we conclude that

$$E\left[\left(1+\hat{\Phi}\right)^{L \wedge k}\right] \leq n. \tag{11}$$

From the Taylor series expansion of $\ln(1+z)$ at 1, for $z \geq 0$ we have the inequality $\ln(1+z) \geq z - z^2/2 = z(1 - z/2)$. By the definition of $\hat{\Phi}$, and the fact that the sum of each row of the matrix P is at most 1, we have $\hat{\Phi} \leq 1$. It follows that $\ln(1+\hat{\Phi}) \geq \hat{\Phi}(1 - \hat{\Phi}/2) \geq \hat{\Phi}/2$, and so $\exp(\hat{\Phi}z/2) \leq (1+\hat{\Phi})^z$ for all $z \geq 0$. Substituting this inequality into (11), we obtain

$$E\left[\exp\left(\frac{\hat{\Phi}(L \wedge k)}{2}\right)\right] \leq n.$$

Because $\exp(\hat{\Phi}(L \wedge k)/2) \uparrow \exp(\hat{\Phi}L/2)$ as $k \to \infty$, the monotone convergence theorem implies that

$$E\left[\exp\left(\frac{\hat{\Phi}L}{2}\right)\right] \leq n.$$

Applying Markov's inequality, we obtain that, for $k_1 = 2(\ln 2 + 2\ln n + \ln(1/\delta))/\hat{\Phi}$,

$$\Pr(L > k_1) = \Pr\left(\exp\left(\frac{\hat{\Phi}L}{2}\right) > \frac{2n^2}{\delta}\right) < \frac{\delta}{2n}. \tag{12}$$

For the second phase of the algorithm, when $|S(k)| > n/2$, we study the evolution of the size of the set of nodes that do not have the message, $|S(k)^c|$. This quantity will decrease as the message spreads from nodes in $S(k)$ to nodes in $S(k)^c$. For simplicity, let us consider restarting the process from clock tick 1 after L (i.e., when more than half the nodes in the graph have the message), so that we have $|S(0)^c| \leq n/2$. The analysis is similar to that for the first phase.

Since $|S(k)| + |S(k)^c| = n$, the equation in (8) gives the conditional expectation $E[|S(k)^c| - |S(k+1)^c| \mid S(k)^c]$ of the decrease in $|S_k^c|$. We use this and the fact that $|S(k)^c| \leq n/2$ to obtain

$$E[|S(k+1)^c| \mid S(k)^c] \quad \leq \quad \left(1 - \hat{\Phi}\right)|S(k)^c|.$$

We note that this inequality holds even when $|S(k)^c| = 0$, and as a result it is valid for all clock ticks k in the second phase. Repeated application of the inequality yields

$$\begin{aligned} E[|S(k)^c|] = E[E[|S(k)^c| \mid S(k-1)^c]] \quad &\leq \quad \left(1 - \hat{\Phi}\right) E[|S(k-1)^c|] \leq \left(1 - \hat{\Phi}\right)^k E[|S(0)^c|] \\ &\leq \quad \left(1 - \hat{\Phi}\right)^k \left(\frac{n}{2}\right). \end{aligned}$$

The Taylor series expansion of e^{-z} implies that $e^{-z} \geq 1 - z$ for $z \geq 0$, and so

$$E[|S(k)^c|] \leq \exp\left(-\hat{\Phi}k\right)\left(\frac{n}{2}\right).$$

For $k_2 = \ln(n^2/\delta)/\hat{\Phi} = (2\ln n + \ln(1/\delta))/\hat{\Phi}$, we have $E[|S(k_2)^c|] \leq \delta/(2n)$. Markov's inequality now implies the following upper bound on the probability that not all of the nodes have the message at the end of clock tick k_2 in the second phase.

$$\Pr(|S(k_2)^c| > 0) = \Pr(|S(k_2)^c| \geq 1) \leq E[|S(k_2)^c|] \leq \frac{\delta}{2n}. \tag{13}$$

Combining the analysis of the two phases, i.e., the inequalities in (12) and (13), we obtain that, for $k' = k_1 + k_2 = O((\log n + \log \delta^{-1})/\hat{\Phi})$, $\Pr(S_v(k') \neq V) < \delta/n$. Applying the union bound over all the nodes in the graph, and recalling that $\hat{\Phi} = \Phi(P)/n$, we conclude that

$$K(\delta) \leq k' = O\left(n\frac{\log n + \log \delta^{-1}}{\Phi(P)}\right).$$

This completes the proof of Lemma 8.

D Proof Sketch for Theorem 5 in Synchronous Model

Since the proof of the upper bound on the information spreading time in the synchronous model follows the same structure as the proof of Lemma 8, we only point out the significant differences in this section. We fix a node $v \in V$, and study the evolution of the size of the set $S(t) = S_v(t)$.

Consider a time slot $t+1$. For any $j \notin S(t)$, let X_j be an indicator random variable that is 1 if node j receives the message m_v in round $t+1$ from some node $i \in S(t)$, and is 0 otherwise. Then,

$$\begin{aligned} E[|S(t+1)| \mid S(t)] &= |S(t)| + E\left[\sum_{j \notin S(t)} X_j \;\middle|\; S(t)\right] = |S(t)| + \sum_{i \in S(t), j \notin S(t)} P_{ji} \\ &= |S(t)| \left(1 + \frac{\sum_{i \in S(t), j \notin S(t)} P_{ij}}{|S(t)|}\right). \end{aligned} \tag{14}$$

Here, we have used the fact that P is symmetric.

It follows from (14) and the definition of conductance that for $|S(t)| \leq n/2$,

$$E[|S(t+1)| \mid S(t)] \geq |S(t)|(1 + \Phi(P)). \tag{15}$$

The inequality in (15) is exactly the same as (10), with a factor n missing. The remainder of the proof is analogous to that for Lemma 8, and hence we skip the details.