INFORMATION AGGREGATION IN SENSOR NETWORKS

BY

HEMANT JAGADISH KOWSHIK

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Doctoral Committee:

Professor P. R. Kumar, Chair
Professor R. Srikant
Professor Tamer Başar
Assistant Professor Todd P. Coleman

ABSTRACT

In many sensor network applications, one is interested only in computing some relevant *function* of the sensor measurements. In this thesis, we study optimal strategies for in-network computation and communication in such wireless sensor networks.

We begin by considering a directed graph $G = (\mathcal{V}, \mathcal{E})$ on the sensor nodes, with a designated collector node, where the goal is to characterize the rate region in $\mathbf{R}^{|\mathcal{E}|}$, i.e., the set of vector rates for which there exist feasible encoders and decoders which achieve zero-error computation for large enough block length. For directed tree graphs, we determine a necessary and sufficient condition for each edge that yields the optimal alphabet, from which we then calculate the minimum worst case and average case complexity. For general directed acyclic graphs, we provide an outer bound on the rate region by finding the disambiguation requirements for each cut, and describe examples where this outer bound is tight.

Next, we consider undirected tree networks, where each node has an integer measurement, and all nodes want to compute a symmetric Boolean function. For a class of functions called sum-threshold functions, we derive an optimal strategy which minimizes the worst-case number of bits exchanged on each edge. In the case of general graphs, we present a cut-set lower bound, and an achievable scheme based on aggregation along trees. For complete graphs, the complexity of this scheme is no more than twice that of the optimal scheme.

We then turn to a collocated network of nodes, where each node has a Boolean measurement and we wish to compute a symmetric Boolean function of these measurements with zero error. Our objective is to determine the minimum worst-case total number of bits to be communicated to perform the desired computation. We define three classes of functions, namely threshold functions, delta functions and interval functions. We provide exactly optimal strategies for the first two classes, and an order-optimal strategy with optimal preconstant for interval functions. Using these results, we can characterize the complexity of computing percentile type functions. The results also extend

to the case of integer measurements and certain integer-valued functions. We use lower bounds from communication complexity theory, and provide an achievable scheme using information theoretic tools.

In the collocated network scenario, minimizing the average case complexity presents a variety of interesting problems. We show that the average case complexity of computing a Boolean threshold function of i.i.d. measurements, with threshold θ, is $O(\theta)$, in comparison to the worst case complexity of $\Omega(\log n)$, where n is the number of nodes. In the case of independent but not identically distributed measurements, we show that the optimal order of transmissions is determined by a surprisingly simple rule that depends in a trivial way on the values of the previously transmitted bits and the ordering, but not the exact values of the marginal probabilities. The approach presented can be generalized to the case of block computation, and to alternate models of communication. We also determine the optimal strategy when the number of bits to be communicated is fixed, and one wants to minimize the conditional entropy of the parity function.

Finally, we consider the problem of determining the connectivity of a random graph by sequentially sampling edges. We present optimal strategies for determining connectivity in series graphs, parallel graphs, series-parallel graphs and parallel-series graphs. In the case of general graphs, we consider the related problem of finding a certificate for connectivity, and conjecture the optimal strategy.

To my adviser and my family

ACKNOWLEDGMENTS

This thesis would not be possible without the support and encouragement of professors and colleagues whose inspiring presence made my years at the University of Illinois an enriching, multifaceted experience.

First and foremost, I would like to thank my adviser Prof. P. R. Kumar for his unstinting support and encouragement through six years of graduate study. His role as a guide, friend and philosopher have been integral in my professional and personal development over these years. He inculcated in me an unabashedly independent research temperament, encouraging me to explore diverse topics and pursue novel ideas. The freedom afforded me as a graduate student allowed me to work on problems that interested me, and to develop my own individual way of thinking. Throughout these years, Prof. Kumar's smiling personality and sang-froid made him a great mentor and role-model. His infectious enthusiasm for the next interesting problem and his broad span of knowledge made the discussions with him thoroughly uplifting and inspiring. The demand for mathematical rigor on the one hand, and the urge to think about the larger technological context on the other, afforded me a superbly well-rounded experience. Prof. Kumar's constant emphasis on clarity of thought and presentation helped me structure my attempts at solving complex problems. Through rough waters and smooth, his patient hand helped steer the course of my research, and his advice on matters both academic and personal, served as a guiding light.

I would also like to thank Prof. R. Srikant, Prof. Todd Coleman and Prof. Tamer Başar for serving on my PhD committee. I am grateful for the many engaging discussions which helped refine my ideas and provided me with new problems. I greatly appreciate their patience and encouragement through the course of my PhD. Their comments and suggestions have helped greatly improve this thesis. I am also grateful to Janet Peters and Jamie Hutchinson at the ECE Publications Office for carefully proofreading this dissertation, and for patiently guiding me through the submission

process.

Through my years in graduate school, I have been blessed with the opportunity to learn from and interact with some of the finest minds in the world. The open and collegial culture at Illinois helped me build personal relationships with various professors (a veritable Stockholm syndrome, if you will!). I am particularly grateful to Prof. Bruce Hajek for his crystal-clear lectures and open-source books on networks and stochastic processes, and for his advice on various issues. I would also like to thank Prof. Richard Sowers for teaching me how to stop worrying about measurability and start loving probability. His engaging and entertaining lectures, and his approachability as a professor made those years spent attending classes at Altgeld Hall truly special.

The Coordinated Science Laboratory provided me with the opportunity to interact with top-class researchers and very talented fellow students. While being a research assistant sometimes feels like having a merciless job with low pay, it is more than made up for by a fantastic set of colleagues. Prof. Kumar's research group always provided an exciting blend of personalities with diverse professional and personal experiences. I would like to thank students past and present - Arvind, Feng, Kurt, Vivek, Min, Robert, Roberto, Nikos, Kyoung-dae, Anand, I-Hong, Zheng, Yan, Ray, Jonathan, Ahn and Ankur for being a great sounding board for ideas and a first-stop source for guidance and counsel. I will sorely miss the many stimulating discussions and the friendly banter that made 139 CSL a relaxed and enriching work environment, and the unofficial lounge of the first floor. I would like to thank Nikos for making the lab go greek, Jonathan for animated discussions about celebrity gossip, Kunal for keeping things upbeat in the darkest of times, Arvind for the customary hilarious forwarded email of the day, Sophie for helping me polish my French pronunciation and Anand for helping me gargle away the French with the sweet sounds of Kannada. I would also like to thank Aadeel, Jamie, Aaron and Anupama for making 139 CSL a truly wonderful place to work.

The camaraderie between students of different groups and the accomodating nature of professors made CSL a thriving ecosystem of ideas, and the perfect learning environment for a young researcher. I am grateful for the opportunity to attend seminars on diverse topics from networks to neuroscience and stochastic control to social media. I would like to thank Adnan, Srikant, Siva, Ramki and others at the "Coffee and Communications" discussion group, which incidentally never had coffee. The intense interactive meetings served as a the final shot of intellectual stimulation

before launching into the lethargy of the weekend. Of course, the carefree existence I enjoyed as a research assistant was made possible by the enviable efficiency of the staff at CSL. I would like to thank Jana Lenz, Becky Lonberger, Barb Horner, Ronda Rigdon, Francie Bridges and Dan Jordan for abstracting out the logistics and always maintaining a smile in the face of the adversities I brought to them.

The tremendous ride I enjoyed in Urbana-Champaign would not have been possible without the support of all my friends who helped me keep it real - real fun that is. The concerts at Krannert Center, the Illinois basketball games and the winter nights spent in the shelter of The Blind Pig served as a counterpoint to the austere and slow-paced nature of academic life. I would like to thank my tennis buddies Anjan, Rajan, Kunal and Jayanand for all the fun tennis sessions, and for their unwavering support of Federer while Nadal slowly dismantled him (Vamos!). I would like to thank Jayakrishnan for emergency pub tours, Sreeram for discussions on research and philosophy, Gayathri for being a guinea pig in my salsa experiments, Jagan and Anil for great times at volleyball, and Vijay for stomach-hurtingly hilarious impressions of everybody else. I would also like to thank Nikhil and Kunal at Northwestern University for the great days spent in Evanston and Chicago, which were a welcome respite from all the craziness of Urbana-Champaign.

Most importantly, I would like to thank my family for being a source of great inspiration and energy, and helping me become "a real man," in Don Corleone's words. I am grateful to my parents for supporting me through these years and serving up periodic doses of spiritual energy. Their encouragement and love strengthened my self-belief and kept me sane over the years. I would like to thank my brother Sumant, for teaching me the ropes as a young graduate student. His advice and support have been tremendously reassuring over the years and helped create a home away from home. I would like to thank Shobha for the exotic lip-smacking dinner spreads and the long engaging conversations on topics professorial and non-professorial. Her zest for life and *carpe diem* attitude have always been a source of great inspiration. I would like to thank Prashant for being a tremendously supportive elder brother, and Ashwini for her constant encouragement. I would also like to thank my grandparents for displaying heart-warming and almost disproportionate pride at every little accomplishment of mine. Finally, I thank God for standing by my side and guiding me each step of the way.

The material in this thesis is based upon work partially supported by AFOSR under Contract FA9550-09-0121, NSF under Contract No. CNS-1035378, Science & Technology Center Grant CCF-0939370, and Contract CNS-1035340, and USARO under Contract Nos. W911NF-08-1-0238 and W-911-NF-0710287. I would like to thank the above agencies for supporting my research. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the above agencies.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Wireless sensor networks are composed of nodes with sensing, wireless communication and computation capabilities. These networks are designed for applications like fault monitoring, data harvesting and environmental monitoring; tasks which can be broadly classified as information aggregation. Sensor networks should be distinguished from general wireless ad-hoc networks in this regard. While traditional data networks are concerned with end-to-end information transfer, sensor networks may only be interested in gathering certain aggregate *functions of distributed data*. For example, one might want to compute the average temperature for environmental monitoring, or the maximum temperature in fire alarm systems.

In this context, communicating all the relevant data to a central collector node which subsequently computes the function might be inefficient since it requires excessive data transfer or energy. This suggests moving away from a data forwarding paradigm, and focusing on efficient in-network computation and communication strategies for the function of interest. This is particularly important since sensor nodes may be severely limited in terms of power and bandwidth, and can potentially generate enormous volumes of data. Such in-network processing and aggregation is made possible by the fact that sensor networks are often application specific, that is, deployed to achieve a specific goal, and hence nodes may look into the contents of packets and create new packets or discard others. This is in sharp contrast to data networks where nodes only process the headers of the packets, but never the contents of the payload of the packets.

The problem of computing functions of distributed data in sensor networks presents several challenges. A fundamental challenge is to exploit the structure of the particular function, so as to optimally combine transmissions at intermediate nodes. Thus, the problem of function computation could be regarded as being more complex than finding the capacity of a wireless network, since the traditional decode and forward model does not capture the possibility of combining information at

intermediate nodes. Second, we have to contend with the broadcast nature of the wireless medium. On the one hand, nodes have to deal with interference from other transmissions, while on the other hand, nodes can exploit these overheard transmissions to achieve a more efficient description of their own data. Further, the strategy for computation may benefit from interactive information exchange between nodes, which presents an additional degree of freedom vis-a-vis the standard point-to-point communication set-up.

In order to make progress on this general problem, we present a rich array of specific results by considering different architectures, network topologies and computation paradigms. To begin with, there are two possible architectures for sensor networks that one might consider. First, one can designate a single collector node/fusion center which seeks to compute the function, as we consider in Chapters 3 - 4. This goal is more appropriate for data harvesting and centralized fault monitoring, where the collector alone makes decisions. Alternately, one can suppose that every node in the network wants to compute the function, as we consider in Chapters 5 - 7. The latter goal can be viewed as providing situational awareness to each sensor node, which could be very useful in applications like distributed fault monitoring, adaptive sensing and sensor-actuator networks. For example, sensor nodes might want to modify their sampling rate depending on the value of the function.

We consider two specific network scenarios in which we study the problem of function computation. In Chapters 3 and 4, we abstract out the medium access control problem associated with a wireless network, and view the network as a directed graph with edges representing essentially noiseless wired links between nodes. We assume implicitly that there is a transmission schedule which ensures no collisions, and reliable transmission of information on each edge. Thereby, we focus specifically on strategies for combining information at intermediate nodes, and optimal codes for transmissions on each edge. In comparison, in Chapters 5, 6 and 7, we consider a collocated network where each node's transmissions can be heard by every other node. Thus, in order to avoid collisions, nodes must transmit one at a time. In this scenario, we study interactive strategies where the content of previously transmitted messages could determine both the identity of the next transmitting node as well as the content of its transmitted message.

In this thesis, we present results for both the block computation and one-shot computation

paradigms. In Chapters 3 - 5, we consider the zero error block computation framework. We allow for nodes to accumulate a block of measurements and realize greater efficiency using block coding strategies. However, we require the function to be computed with zero error for the block. This corresponds to the zero-error information theoretic paradigm. In contrast, in Chapters 7-8, we consider the problem of single-instance computation.

Throughout this thesis, we consider a packet capture model as in [1] and suppose that collisions do not convey information. Further, we suppose that there is a joint distribution on the node measurements and we consider both worst case and average case performance metrics. Under this framework, we address three fundamental questions.

- Since nodes may possess heterogeneous data which could affect the function to different degrees, we study the problem of identifying which node should transmit next. This could depend on the structure of the function as well as side information gained from overheard transmissions.

- In order to minimize the communication footprint, we study the minimum information that must be conveyed by nodes so as to compute a particular function. Further, we study strategies for combining transmissions at intermediate nodes.

- Given the information that must be conveyed, we study optimal coding strategies that minimize worst-case and average case metrics.

It is particularly interesting to study the benefit of multi-round protocols, possibly involving complex interaction between nodes, versus single round protocols, where each node only transmits once.

The rest of the thesis is organized as follows. In Chapter 3, we consider the problem of general function computation in a directed graph $G = (\mathcal{V}, \mathcal{E})$ with a designated collector node that wants to compute the function. We focus specifically on optimal coding strategies for transmissions on each edge, and strategies for combining information at intermediate nodes. We suppose that there is a joint probability distribution on the node measurements, and consider both the worst case and the average case complexity for zero error block computation. Our goal is to characterize the rate region in $\mathbf{R}^{|\mathcal{E}|}$, i.e., the set of points for which there exist feasible encoders with given rates

which achieve zero-error computation for large enough block length. In the case of tree graphs, we derive a necessary and sufficient condition for the encoder on each edge, which provides a complete characterization of the rate region. The extension of these results to directed acyclic graphs is harder. However, we provide an outer bound on the rate region by finding the disambiguation requirements for each cut, and describe examples where this outer bound is tight.

In Chapter 4, we address the problem of computing symmetric Boolean functions in undirected graphs. The key difference from Chapter 3 is that we consider bidirectional links and study the benefit of interaction between nodes. We show how the approach described in Chapter 3, together with ideas from communication complexity theory, can be synthesized to develop a theory of optimal computation of symmetric Boolean functions in undirected graphs. In the case of tree networks, each edge is a cut-edge, and this allows us to derive a lower bound on the number of bits exchanged on each edge, by considering an equivalent two node problem. Further, we show that a protocol of recursive in-network aggregation along with a smart interactive coding strategy, achieves this lower bound for the class of sum-threshold functions in tree networks. The optimal strategy has a simple structure that is reminiscent of message passing, where messages flow from the leaves towards an interior node, and then flow back from the interior node to the leaves. In the case of general graphs, we present a cut-set lower bound, and an achievable scheme based on aggregation along trees. For complete graphs, the complexity of this scheme is no more than twice that of the optimal scheme.

In Chapter 5, we study the problem of computing symmetric Boolean functions in a collocated network, where each node's transmissions can be heard by every other node. The key challenge is for nodes to systematically exploit previous transmissions to compress their own data. We suppose that each node has a Boolean measurement and we wish to compute a given symmetric Boolean function of these measurements with zero error. We define three classes of functions, namely threshold functions which evaluate to 1 if the number of 1s exceeds a certain threshold, delta functions which evaluate to 1 if the number of 1s is exactly equal to a given value, and interval functions which evaluate to 1 if the number of 1s is between two given lower and upper values. For worst-case computation, we provide exactly optimal strategies for the first two classes, and an order-optimal strategy, as the number of nodes increases, with optimal preconstant for interval functions. In our analysis, we use lower bounds from communication complexity theory, and provide

an achievable scheme using information theoretic tools. Using these results, we can characterize the complexity of computing percentile type functions, which is of great interest. Further, the approach presented can be generalized to compute functions of non-Boolean measurements, as shown in our treatment of general threshold functions and the MAX function. In Chapter 6, we consider the problem of minimizing the average-case complexity of computing a Boolean threshold function in a collocated network. We show that the average-case complexity of computing a Boolean threshold function in a collocated network is $O(\theta)$, where θ is the threshold.

In Chapter 7, we address a class of sequential decision problems that arise in the context of computing symmetric Boolean functions of distributed data. We consider a collocated network, where each node's transmissions can be heard by every other node. Each node has a Boolean measurement and we wish to compute a given Boolean function of these measurements. We suppose that the measurements are independent and Bernoulli distributed. We address the problem of optimal computation of the function so as to minimize the total number of bits that are transmitted. We solve the problem for the class of Boolean threshold functions. The problem reduces to one of determining the optimal order in which the nodes should transmit. We show the surprising result that the optimal order of transmissions depends in an extremely simple way on the values of previously transmitted bits, and the ordering, but not the exact values, of the marginal probabilities of the Boolean variables, according to the *k-th least likely rule*: At any transmission, the node that has the k-th least likely value of its Boolean variable transmits, where k reduces by one each time any node transmits a one. Initially the value of k is ($n + 1$ - Threshold). The approach presented can be generalized to the case where each node has a block of measurements, though the resulting problem is somewhat harder, and we conjecture the optimal strategy. We also show how to generalize to a pulse model of communication. One can also consider the related problem of approximate computation given a fixed number of bits. We show that in this case, the optimal strategy can be significantly different. However, for the special case of the parity function, we show that the greedy strategy is optimal.

Finally, in Chapter 8, we consider the interesting problem of determining $s - t$ connectivity in a random graph G where each edge occurs independently of the others with a given Bernoulli probability. We suppose that each edge is equipped with a sensor that can be queried for existence of

the edge, at an associated sampling cost. Our objective is to determine connectivity by sequentially sampling edges, while incurring minimum expected total cost. This problem arises in the practical context of fault detection in a communication network or a chemical plant. Since we do not know *a priori* if the graph is connected or not, there is a natural tension between showing the existence of an $s-t$ path and showing the absence of an $s-t$ cut. We present some special classes of graphs for which this tension can be resolved, yielding optimal strategies for determining connectivity. In the case of series graphs, we prove that the optimal strategy is to sample the edge with the maximum ratio of probability of absence of the edge to the cost of sampling the edge. Analogously, in the case of parallel graphs, we prove that the optimal strategy is to sample the edge with the maximum ratio of probability of existence of the edge to the cost of sampling the edge. Next, we present optimal strategies for series-parallel graphs and parallel-series graphs. In the former case, the optimal strategy focuses exclusively on $s-t$ cuts, while in the latter case, the optimal strategy focuses exclusivley on $s-t$ paths. The extension to general graphs is harder, and the optimal strategy remains elusive. However, we formulate a conditional subproblem for which we conjecture the optimal strategy.

The results described in this dissertation have appeared in various conference and journal papers [2], [3], [4], [5], [6].

CHAPTER 2

RELATED WORK

The general problem of computing a function of correlated data, over a distributed network of nodes with wireless links, admits a variety of approaches. It is related to fundamental problems in several fields – multi-terminal information theory, distributed source coding, communication complexity and distributed computation. We present a summary of the related literature.

2.1 Zero-Error Block Computation of Functions

Consider a general network of n nodes, with a designated collector node to which the value of the aggregate function of interest needs to be communicated. Each node has a certain transmission range, and can transmit directly to any other node within that range, provided no other transmission is interfering. Two networks of interest are the *collocated* network, where the connectivity graph is complete, and the random *multihop* network where the connectivity graph is a random geometric graph. In [1], the authors have formulated the problem of worst-case block function computation with zero error. Nodes make a block of N measurements, and the collector or fusion node seeks to compute the block of function values. It should be noted that communicating a block of measurements has been shown by Shannon [7] to be critical to achieving reliability with high throughput in point-to-point communication channels. In the multinode function computation problem, the appropriate measure of efficiency is the *computational throughput*, which is defined as the reciprocal of the minimum number of time units required per computation, over all schemes, and over all block lengths. In [1], an order optimal strategy for computing *divisible* functions is derived, given the constraints of the wireless medium,

In sensor networks, we are often interested in *symmetric* functions which only depend on the data of a sensor, not its identity. They include statistical functions like mean, median, maxi-

mum/minimum, and others which are completely determined by the histogram of the set of node measurements. A key property is that the histogram of two disjoint sets can be combined to give the histogram of the union. Among the class of symmetric functions, not all functions are equally hard to compute. For a function like Maximum, if a node knows that the maximum temperature recorded until now is at least 100, then the node need not transmit any further reading unless it has a higher value. Thus the previous transmissions provide side information about the function value, even if the measurements are independent. On the other hand, for a function like Average, if even a single measurement is missing, we could have incorrect function computation, if the goal is zero-error block function computation. The results do change significantly if we allow a vanishing error as block length increases.

In [1], the authors identify two classes of symmetric functions, namely *type-sensitive* functions like Average, Median, Majority and Histogram, and *type-threshold* functions, such as Maximum and Minimum. The maximum rates for computation of type-sensitive and type-threshold functions in a collocated network is shown to be $\Theta(\frac{1}{n})$ and $\Theta(\frac{1}{\log n})$ respectively, where n is the number of nodes. The upper bound on the computational throughput in each case is obtained by generalizing the concept of *fooling sets* in communication complexity [8]. In random planar networks, spatial reuse of the wireless medium leads to an exponential speedup, and the maximum rates for computation of type-sensitive and type-threshold are shown to be $\Theta(\frac{1}{\log n})$ and $\Theta(\frac{1}{\log \log n})$ respectively. Some extensions in the case of finite degree graphs are presented in [9]. If we assume that the measurements are drawn independently and identically from some distribution, one can obtain an even higher computational throughput as shown in Chapter 6.

The assumption of block codes which ensure zero error for the block corresponds to the zero-error information theoretic paradigm [10]. The problem of source coding with side information ensuring zero error for finite block length has been studied in [11] and [12]. The problem reduces to the task of coloring a probabilistic graph defined on the set of source samples. The minimum entropy of such a coloring approaches the graph entropy or Korner entropy, as block length goes to infinity. In Chapter 3, we revisit these ideas in the context of partitioning input blocks into equivalence classes. Further, we extend this simple approach to zero error function computation on graphs. In the zero-error framework, the effect of multi-round interaction on two party communication

complexity has been studied [13]. For the problem of source coding with side information, it is shown that single round schemes are almost optimal [14], [15].

2.2 Multi-Party Computation

The problem of communication complexity was introduced by Yao in [16]. In communication complexity, one seeks to minimize the number of bits that must be exchanged in the worst case between two nodes to achieve zero-error computation of a function of the node variables. A thorough exposition of fundamental results in communication complexity can be found in [8]. The communication complexity of Boolean functions has been studied in [17], [18]. One can go further and consider the *direct-sum problem* [19] where several instances of the problem are considered together to obtain savings. This block computation approach is used to compute the exact complexity of the Boolean AND function in [20]. In this thesis, we generalize this result, which allows us to derive optimal strategies for computing more general classes of symmetric Boolean functions in collocated networks (Chapter 5) and tree networks (Chapter 4).

In Chapters 3 and 4, we consider function computation on graphs obtained after abstracting out medium access control, effectively resulting in wired links between nodes. This resembles the network coding paradigm, except that we are now computing functions. In the simplest case, assuming independent measurements x_i and the function $f(x_1, x_2, \ldots, x_n) = (x_1, x_2, \ldots, x_n)$, we have the reverse of the multicast problem studied in [21]. For the particular case where a set of terminal nodes want to compute the sum of source measurements, ideas from network coding can be used to derive the rate region for some special cases [22],[23]. Computing a function of independent measurements is a *network computation problem* as opposed to a network coding problem. In [24], the min-cut upper bound on the rate of computation is shown to be tight for the computation of divisible functions on tree graphs. In Chapter 3, we generalize this result using a different approach. Further, the simplicity of the approach presented allows extensions in the case of general graphs and collocated networks.

Another approach to the problem of function computation comes from the literature on message passing. Here, one is interested in computing a function efficiently by exploiting the structure of the domains. There are two broad approaches: one using factor graphs [25] and the other using

the generalized distributive law [26]. These approaches have been applied very effectively to the problems of computing marginals and probabilistic inference. This has resulted in low complexity iterative algorithms for decoding of turbo codes. For the specific problem of computing the average, randomized gossip algorithms have been proposed in [27] and shown to be computationally inexpensive and resilient to failures. These algorithms consist of nearest neighbor update rules [28] which eventually result in dissemination of the average value to all nodes in the network. A similar iterative approach is used to design distributed randomized algorithms for computing separable functions in [29]. It must be noted that the above approaches do not address the communication problem and suppose that nodes can communicate real numbers to their neighbors. In [30], the problem of quantized consensus is studied, where nodes seek to compute the average value by communicating over finite capacity channels.

In this thesis, we do not consider communication and computation strategies that are robust to an eavesdropper that is overhearing transmissions and seeks to compute the function. In [31], the problem of Boolean function computation by two nodes using a public channel and a secret private channel was considered. The objective is to minimize the number of bits transmitted over the secret channel to compute the function. An extension to the case where there is noise in the eavesdropper's channel was studied in [32] and the extension to general distributions was studied in [33]. In [34], necessary and sufficient conditions for the secure computability of a function of correlated sources were derived.

Throughout this thesis, we assume implicitly that nodes have identities, and each message transmitted by a node contains a header that identifies the node. Alternately, one can consider the problem of anonymous computation where n indistinguishable processors seek to compute the function. In [35], the authors characterize which functions are computable on an anonymous ring, and show how they can be computed using $O(n^2)$ messages. The results have been generalized to the problem of distributed function computation in a general network of anonymous nodes [36].

2.3 Information Theoretic Formulation

To study the ultimate performance limits of function computation as well as optimal function computation strategies, one needs to turn to an information theoretic formulation. There are

two features that can be incorporated in this most general formulation. First, we can allow for a vanishing error of computation, in contrast with the previous formulation which considers zero-error block computation. Second, we can exploit the correlation in sensor measurements, and achieve higher efficiency. However, there are very few results for this most general framework. We now review some of the basic information theoretic results, which lead to a more general formulation that incorporates computation over wireless networks. An information-theoretic formulation of this problem combines the complexity of source coding of correlated sources with rate distortion, together with the complications introduced by the function structure; see [37]. There is little or no work that addresses this most general framework.

This problem of *decentralized compression of correlated sources* was studied by Slepian and Wolf [38]. Consider two sensors with correlated measurements X, Y drawn according to the joint distribution $p(X, Y)$. One wishes to determine the *rate region*, i.e., the set of all possible pairs of rates (R_1, R_2) at which the sources can be individually compressed so that the receiver can reconstruct the original sources with vanishing probability of error. The region defined by the cut-set bounds was remarkably shown to be achievable using the technique of *random binning* to exploit correlation. The Slepian-Wolf problem can be easily extended to the case of multiple correlated sources communicating to a receiver. However, the extension to tree networks, with the receiver as root, has not been solved.

Another interesting variation of the problem arises if we only desire the reconstruction of sources to some *fidelity*, i.e., the receiver wishes to recover estimates X, Y such that $E[D(X, X')] \leq d$, $E[D(Y, Y')] \leq d$, where $D(\cdot, \cdot)$ is a given *distortion measure*. This problem is open as well. The special case of this problem in which one of the sources is known to the receiver as *side information*, was solved by Wyner and Ziv [39]. It is important to note that function computation can be viewed as a special case of the *rate-distortion problem*, by defining an appropriate distortion metric that is function-dependent. For example,

$$D(X, \hat{X}) := |f(X, Y) - \hat{f}(\hat{X}, Y)|^2.$$

This has been extended in [40] to the case in which the receiver desires to know a certain function $f(X, Y)$ of the single source X and the side information Y; the authors determined the required

capacity of the channel between the source and receiver to be the conditional graph entropy, which is a measure defined on the two random variables and the characteristic graph [11] defined by the function f. Recently, there have been some extensions to the case of two nodes [41] and to the case of tree networks [42]. However, a general single-letter characterization has remained elusive.

Another interesting stand-alone problem was studied in [43], where two correlated binary sources need to encode their sequences in distributed fashion, so that the receiver can compute the XOR function of the two sequences. It is shown that, in some cases, the sum rate required may be substantially less than the joint entropy of the two sources. This clearly displays the advantage of function-aware encoding strategies over standard distributed source coding.

The above formulation considers a single round protocol, where only X communicates with Y. One can further seek the information theoretic limits for interactive computation. The rate region for multi-round interactive function computation has been characterized for two nodes in [44], and for collocated networks in [45]. The characterization closely resembles the Wyner-Ziv result and some interesting connections with communication complexity are made.

2.4 Modeling Channel Noise

In our work, we assume that the channel is a noiseless wired link, resulting possibly after the medium access control problem is solved. However, in practice, all channels are noisy, and one needs to study the rate of computation over noisy channels. In [46], the problem of computing parity in a broadcast network was studied, assuming independent binary symmetric channels between each pair of nodes. In the noiseless case, one would require n bits, and in the noisy case, one can achieve $O(n \log n)$ using repetition coding. However, this approach does not make use of the broadcast nature of wireless medium. For each bit that is transmitted, all the other nodes in the network hear a noisy version of it. The key contribution of [46] is a strategy which systematically uses this side-information to compute the parity function. The hierarchical strategy proposed requires only $\Theta(n \log \log n)$ bits. Remarkably, this upper bound of $n \log \log n$ was shown to be sharp in [47], using noisy binary decision trees. The problem of computing a threshold function in a noisy broadcast network was studied in [48] and the OR function was studed in [49].

In [50], the problem of distributed symmetric function computation in noisy wireless networks

is considered. It is shown that the energy usage for computing a symmetric function in a network with binary symmetric channels is $\Theta\left(n(\log\log n)\left(\sqrt{\frac{\log n}{n}}\right)^{\alpha}\right)$, where α is the path loss exponent. In [51], this approach is extended to derive an order optimal strategy for computing the MAX function in a random planar network.

While the information theoretic formulation described above only considers distributed source coding, one would like to generalize the formulation to incorporate a more general channel model. For example, consider two sources $S1$ and $S2$, which have access to channel inputs X and Y of a multiple access channel, with output $Z = X + Y + N$ being available to a receiver, where N is Gaussian noise. The receiver wishes to compute the sum $X + Y$. The question of interest is to find the optimal *power-distortion curve*; i.e., for a given pair of transmit powers $P1$ and $P2$, what is the minimum distortion D at which the sum $X+Y$ can be communicated to the receiver? Thus, the channel operation itself can be viewed as an implicit computation, as argued in [52].

Finally, the above described solution to the medium access problem can be described broadly as *interference avoidance*, since we assume that collisions do not convey information. We could generalize the formulation further by assuming an interference network. However, the solution to such a problem is very far from the current frontiers of knowledge in information theory.

2.5 Sequential Decision Making

Due to the broadcast nature of the wireless medium, two nodes which are close to each other cannot transmit simultaneously. Thus, nodes need to schedule their transmissions to avoid interfering with one another. The challenge now is to order nodes' transmissions so as to exploit the structure of the function, the side-information gained from previously transmitted bits, and the knowledge of the underlying distribution. Sequential decision problems have been studied in various forms. The most well known problem of designing sequential experiments is the bandit problem [53], [54], [55]. We have N parallel projects evolving according to Markov processes, and generating instantaneous rewards. At each time-step, the player chooses to activate one project so as to maximize the expected discounted reward. Under the optimal strategy, each arm is assigned a dynamic allocation index and the maximum index arm is chosen.

Search problems present another interesting class of sequential decision problems. Due to various

constraints, one might be forced to conduct a sequential search. Thus, the challenge is to design a search strategy that terminates in minimum time or minimum expected time, as per the context. A thorough review of various formulations of search problems is presented in [56]. A class of one-dimensional search problems is studied in [57].

In [58], an interesting problem in sequential decision making is studied, where n nodes have i.i.d. measurements, and a central agent wishes to know the identities of the nodes with the k largest values. One is allowed questions of the type "Is $X \geq t$", in response to which the central agent receives the list of all nodes which satisfy the condition. Under this framework, the optimal recursive strategy of querying the nodes is found. In Chapters 7 and 8, we will consider some problems in optimal sequential sampling. A key difference in our formulation is that we are only allowed to query particular nodes, and not all nodes at once. The problem of minimizing the depth of decision trees for Boolean threshold queries is considered in [59].

CHAPTER 3

FUNCTION COMPUTATION IN DIRECTED GRAPHS

In this chapter, we formulate the problem of zero error block computation on graphs. We abstract out the medium access control problem, and view the network as a directed graph with edges representing noiseless wired links allowing reliable transmission of information between nodes. We suppose that there is a joint probability distribution on the node measurements, and consider both the worst case and the average case complexity. Further, instead of restricting a strategy to compute just one instance of the problem, we allow for a block of N independent instances for which the function is to be computed. Thus nodes can accumulate a block of N measurements, and realize greater efficiency by using block codes, while still ensuring zero error for the block.

Given a graph, the problem we address is to determine the set of rates on the edges which will allow zero error function computation for a large enough block length. In essence, we are exploring the interaction between the function structure and the structure of the graph; how information needs to be routed and combined at intermediate nodes to achieve certain rate vectors. We start with a simple two node example and build up to directed trees and directed acyclic graphs.

In Section 3.1, we begin with the two node problem. We compute the number of bits that node v_X needs to communicate to node v_Y so that the latter can compute a function $f(X, Y)$ with zero error. For correct function computation, an encoder must disambiguate certain pairs of source symbols of node v_X, on which the function disagrees. We show by explicit construction of a code that this necessary condition is in fact sufficient. This yields the optimal alphabet and we calculate the minimum worst case and average case complexity, with the latter obtained by Huffman coding over the optimal alphabet. In Section 3.2, we extend this result to directed trees with the collector as root, exploiting the fact that each edge is a cut-edge. This yields the optimal alphabet for each edge, and we separately optimize the encoders for the worst case and the average case. Thus the rate region consists of all rate points dominating a single point that is coordinate-wise optimal.

In Section 3.3, we consider directed acyclic graphs. A key difference from the tree case is the presence of multiple paths to route the data, which present different opportunities to combine information at intermediate nodes. We arrive at an outer bound to the rate region by finding the disambiguation requirements for each cut of the directed graph. This outer bound is not always tight as we show in Example 12. However, for the worst case computation of finite field parity, and the maximum or minimum functions, the outer bound is shown to be indeed tight. Further, the only extreme points of the rate region are rate points corresponding to activating only a tree subset of edges.

3.1 Two Node Setting

3.1.1 Worst case complexity

We begin by considering the simple two node problem. Suppose nodes v_X and v_Y have measurements $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, where the alphabets $\mathcal{X}$ and $\mathcal{Y}$ are finite sets. Node v_X needs to optimally communicate its information to node v_Y so that a function $f(x, y)$, which takes values in $\mathcal{D}$, can be computed at v_Y with zero error. We do not consider the case where v_X and v_Y interactively compute the function. Thus node v_X has an encoder $\mathcal{C} : \mathcal{X} \to \{0,1\}^*$, which maps its measurement x to the codeword $\mathcal{C}(x)$, and node v_Y has a decoder $g : \{0,1\}^* \times \mathcal{Y} \to \mathcal{D}$ which maps the received codeword $\mathcal{C}(x)$ and its own measurement y to a function estimate, $g(\mathcal{C}(x), y)$. The set of all possible codewords is called the codebook, denoted by $\mathcal{C}(\mathcal{X})$

Definition 1 (Feasible Encoder)**.** An encoder $\mathcal{C}$ is *feasible* if there exists a decoder $g : \{0,1\}^* \times \mathcal{Y} \to \mathcal{D}$ such that $g(\mathcal{C}(x), y) = f(x, y)$ for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$. Thus, a feasible encoder is one that achieves error-free function computation.

Theorem 1 (Characterization of Feasible Encoders)**.** *An encoder $\mathcal{C}$ is feasible if and only if given $x^1, x^2 \in \mathcal{X}$, $\mathcal{C}(x^1) = \mathcal{C}(x^2)$ implies $f(x^1, y) = f(x^2, y)$ for all $y \in \mathcal{Y}$.*

Proof: By definition, if $\mathcal{C}$ is a feasible encoder, then there exists a corresponding decoder g such that $g(\mathcal{C}(x^1), y) = f(x^1, y)$ and $g(\mathcal{C}(x^2), y) = f(x^2, y)$, for all $y \in \mathcal{Y}$. Further, if $\mathcal{C}(x^1) = \mathcal{C}(x^2)$, we have $f(x^1, y) = f(x^2, y)$ for all $y \in \mathcal{Y}$.

To prove the converse, we need to construct a decoding function $g : \{0,1\}^* \times \mathcal{Y} \to \mathcal{D}$. For each codeword C^* in the codebook, define $\mathcal{C}^{-1}(C^*) := \{x \in \mathcal{X} : \mathcal{C}(x) = C^*\}$. For fixed $y \in \mathcal{Y}$ and fixed codeword $C^* \in \mathcal{C}(\mathcal{X})$, the decoder mapping is given by $g(C^*, y) := f(x^{nom}(C^*), y)$ for any arbitrary $x^{nom}(C^*) \in \mathcal{C}^{-1}(C^*)$. We show that this decoder works for any fixed x and y . Indeed, $g(\mathcal{C}(x), y) = f(x^{nom}, y)$ where $x^{nom} \in \mathcal{C}^{-1}(\mathcal{C}(x))$. Thus, $\mathcal{C}(x^{nom}) = \mathcal{C}(x)$ and by assumption $f(x^{nom}, y) = f(x, y)$. Hence, $g(\mathcal{C}(x), y) = f(x^{nom}, y) = f(x, y)$ for all $y \in \mathcal{Y}$. $\square$

Any feasible encoder $\mathcal{C}$ can be viewed as partitioning the set $\mathcal{X}$ into $\Pi(\mathcal{C}) := \{S_1, S_2, \ldots, S_k\}$ such that for $x^1 \in C_i, x^2 \in C_j$, we have $\mathcal{C}(x^1) = \mathcal{C}(x^2)$ if and only if $i = j$. Define an equivalence relation "$\leftrightarrow$" between $x^1, x^2 \in \mathcal{X}$ by:

$$x^1 \leftrightarrow x^2 \text{ if and only if } f(x^1, y) = f(x^2, y) \text{ for all } y \in \mathcal{Y}.$$

Consider the encoder $\mathcal{C}^{OPT}$ which assigns a distinct codeword to each resulting equivalence class. Clearly, $\mathcal{C}^{OPT}$ is a feasible encoder, since $\mathcal{C}^{OPT}(x^1) = \mathcal{C}^{OPT}(x^2)$ implies $x^1 \leftrightarrow x^2$, and hence $f(x^1, y) = f(x^2, y)$ for all $y \in \mathcal{Y}$. $\mathcal{C}^{OPT}$ is optimal in the sense that any other feasible encoder $\mathcal{C}$ must have at least as many codewords as $\mathcal{C}^{OPT}$:

Theorem 2 (Optimality of $\mathcal{C}^{OPT}$)**.** *Let* $\Pi(\mathcal{C}^{OPT}) := \{S_1^{OPT}, S_2^{OPT}, \ldots, S_k^{OPT}\}$ *be the partition of* $\mathcal{X}$ *generated by* $\mathcal{C}^{OPT}$*, and let* $\Pi(\mathcal{C}) := \{S_1, S_2, \ldots, S_l\}$ *be the partition of* $\mathcal{X}$ *generated by any other feasible encoder* $\mathcal{C}$*. Then,*

(i) $\Pi(\mathcal{C})$ *must be a finer partition than* $\Pi(\mathcal{C}^{OPT})$*.*

(ii) The minimum number of bits that node v_X *needs to communicate is* $\lceil \log |\Pi(\mathcal{C}^{OPT})| \rceil$*.*

Proof: First we claim that any subset $S_i \in \Pi(\mathcal{C})$ can have nonempty intersection with exactly one subset $S_j^{OPT} \in \Pi(\mathcal{C}^{OPT})$. Suppose not. Then there exist $x^1, x^2 \in S_i$ such that $x^1 \in S_{j_1}^{OPT}$ and $x^2 \in S_{j_2}^{OPT}$. Since $\mathcal{C}(x^1) = \mathcal{C}(x^2)$, by Theorem 1, we must have $f(x^1, y) = f(x^2, y)$ for all $y \in \mathcal{Y}$. However, by construction of $\mathcal{C}^{OPT}$, x^1 and x^2 must belong to distinct equivalence classes i.e., $x^1 \nleftrightarrow x^2$. Hence, there exists y^* such that $f(x^1, y^*) \neq f(x^2, y^*)$, which is a contradiction. This shows that the partition generated by any encoder $\mathcal{C}$ must be a further subdivision of the partition generated by $\mathcal{C}^{OPT}$, i.e., finer than $\Pi(\mathcal{C}^{OPT})$. So node v_X needs to communicate at least $\lceil \log |\Pi(\mathcal{C}^{OPT})| \rceil$ bits. $\square$

We can extend this to the case where v_X collects a block of N measurements $\underline{x} = (x_1, x_2, \ldots, x_N) \in \mathcal{X}^N$, and v_Y collects a block of N measurements $\underline{y} = (y_1, y_2, \ldots, y_N) \in \mathcal{Y}^N$. We want to find a block encoder $\mathcal{C}^N : \mathcal{X}^N \to \{0,1\}^*$ so that the vector function $f^{(N)}(\underline{x}, \underline{y}) = (f(x_1, y_1), \ldots, f(x_N, y_N))$ can be computed without error, for all $\underline{x} \in \mathcal{X}^N, \underline{y} \in \mathcal{Y}^N$. All the above results carry over to the error-free block computation case. As before, we define an equivalence $\leftrightarrow$ between $\underline{x}^1, \underline{x}^2 \in \mathcal{X}^N$ if $f^{(N)}(\underline{x}^1, \underline{y}) = f^{(N)}(\underline{x}^2, \underline{y})$ for all $\underline{y} \in \mathcal{Y}^N$. The optimal encoder $\mathcal{C}^{N,OPT}$ is once again obtained by assigning distinct codewords to each equivalence class. Since we are stringing together N independent instances, we have $|\Pi(\mathcal{C}^{N,OPT})| = |\Pi(\mathcal{C}^{OPT})|^N$. Hence the minimum number of bits per computation that node v_X needs to communicate is $\frac{\lceil N \log |\Pi(\mathcal{C}^{OPT})| \rceil}{N}$ which converges to $\log |\Pi(\mathcal{C}^{OPT})|$ as $N \to \infty$.

3.1.2 Average case complexity

Suppose now that the measurements X, Y are drawn from the joint probability distribution $p(X,Y)$, with the goal being to minimize the average number of bits that need to be communicated, i.e., the *average case complexity*.

Definition 2 (Feasible Encoder)**.** An encoder $\mathcal{C} : \mathcal{X} \to \{0,1\}^*$ is *feasible* if there exists a decoder $g : \{0,1\}^* \times \mathcal{Y} \to \mathcal{D}$ such that $g(\mathcal{C}(x), y) = f(x,y)$ for all $\{(x,y) \in \mathcal{X} \times \mathcal{Y} : p(x,y) > 0\}$.

Theorem 3. *An encoder $\mathcal{C}$ is feasible if and only if, given $x^1, x^2 \in \mathcal{X}$, $\mathcal{C}(x^1) = \mathcal{C}(x_2)$ implies $f(x_1, y) = f(x_2, y)$ for $\{y \in \mathcal{Y} : p(x_1, y)p(x_2, y) > 0\}$.*

Proof: By definition, if $\mathcal{C}$ is a feasible encoder, then there exists a corresponding decoder g such that $g(\mathcal{C}(x^1), y) = f(x^1, y)$ and $g(\mathcal{C}(x^2), y) = f(x^2, y)$, for all $\{y \in \mathcal{Y} : p(x^1, y)p(x^2, y) > 0\}$. Further, if $\mathcal{C}(x^1) = \mathcal{C}(x^2)$, we have $f(x^1, y) = f(x^2, y)$ for $\{y \in \mathcal{Y} : p(x^1, y)p(x^2, y) > 0\}$.

To prove the converse, for fixed $y \in \mathcal{Y}$ and fixed codeword $\mathcal{C}^* \in \mathcal{C}(\mathcal{X})$, define the decoder by $g(C^*, y) := f(x^{nom}(C^*, y), y)$ for any arbitrary $x^{nom}(C^*, y) \in \mathcal{C}^{-1}(C^*)$ with $p(x^{nom}(C^*, y), y) > 0$. We show that this decoder works for any fixed x and y with $p(x,y) > 0$. Indeed, $g(\mathcal{C}(x), y) = f(x^{nom}, y)$ where $x^{nom} \in \mathcal{C}^{-1}(\mathcal{C}(x))$ with $p(x^{nom}, y) > 0$. Thus, $\mathcal{C}(x^{nom}) = \mathcal{C}(x)$ and by assumption $f(x^{nom}, y) = f(x, y)$ since $p(x^{nom}, y)p(x, y) > 0$. Hence, $g(\mathcal{C}(x), y) = f(x^{nom}, y) = f(x, y)$. $\square$

We now define "$x^1 \leftrightarrow x^2$" when $f(x^1, y) = f(x^2, y)$ for $\{y \in \mathcal{Y} : p(x^1, y)p(x^2, y) > 0\}$. Now the

$\leftrightarrow$ relation is reflexive and symmetric, but not necessarily transitive. This upsets the equivalence class structure and the problem of finding the optimal encoder is much harder as we see in Section 3.1.3. However, if $p(x,y) > 0$ for all $(x,y) \in \mathcal{X} \times \mathcal{Y}$, then $\leftrightarrow$ is an equivalence relation. We can construct an encoder $\mathcal{C}^{OPT}$ which assigns a distinct codeword to each equivalence class. Let $\Pi(\mathcal{C}^{OPT}) := \{S_1^{OPT}, S_2^{OPT}, \ldots, S_k^{OPT}\}$ be the partition of $\mathcal{X}$ generated by $\mathcal{C}^{OPT}$. Analogous to Theorem 2, we can show that the encoder $\mathcal{C}^{OPT}$ has the *optimal alphabet* $\mathcal{A}$, with the probability distribution vector $\underline{q} = \{q_1, q_2, \ldots, q_k\}$ where $q_i := \sum_{x \in S_i^{OPT}} \sum_{y \in \mathcal{Y}} p(x,y)$.

Once the optimal alphabet is fixed, the optimal code $\mathcal{C}^{OPT}$ is the *binary Huffman code* for the probability vector $\underline{q}$. Since the Huffman code has an average code length within one bit of the entropy,

$$H(q_1, q_2, \ldots, q_k) \leq E[l(\mathcal{C}^{OPT})] \leq H(q_1, q_2, \ldots, q_k) + 1.$$

The extension to the case where nodes v_X,v_Y collect a block of N i.i.d. measurements is straightforward. We want to find a block encoder $\mathcal{C}^N : \mathcal{X}^N \to \{0,1\}^*$ so that the vector function $f^{(N)}(\underline{x}, \underline{y})$ can be computed with zero probability of error, using $\mathcal{C}^N(\underline{x})$ and $\underline{y}$. As before, if $p(\underline{x}, \underline{y}) > 0$ for all $\underline{x}, \underline{y} \in \mathcal{X}^N \times \mathcal{Y}^N$, we define $x^1 \leftrightarrow x^2$ if $f^{(N)}(\underline{x}^1, \underline{y}) = f^{(N)}(\underline{x}^2, \underline{y})$ for all $\underline{y} \in \mathcal{Y}^N$. The optimal alphabet is $\mathcal{A}^N$, which has the product distribution $\underline{q}^N$. The optimal encoder is obtained via the Huffman code for the optimal alphabet. Its expected length satisfies

$$\frac{H(\underline{q}^N)}{N} \leq \frac{E[l(\mathcal{C}^{N,OPT})]}{N} \leq \frac{H(\underline{q}^N) + 1}{N}.$$

Hence the minimum number of bits per computation that node v_X needs to communicate converges to $H(\underline{q})$ as $N \to \infty$.

3.1.3 Graphical interpretation

The problem of source coding with side information can also be viewed in graph-theoretic terms. The contrapositive of the statement of Theorem 3 tells us which pairs $x^1, x^2 \in \mathcal{X}$ must be *separated* for any feasible encoder.

Consider a graph $G = (V, E)$ with vertex set $V = \mathcal{X}$, where $(x^1, x^2) \in E$ if there exists $y^* \in \mathcal{Y}$ such that $f(x^1, y^*) = f(x^2, y^*)$ and $p(x^1, y^*)p(x^2, y^*) > 0$. Note that we have an induced distribution

on the vertices, i.e., for $x \in \mathcal{X}$, we have $p_X(x) = \sum_{y \in \mathcal{Y}} p(x,y)$. This graph is referred to as the Witsenhausen characteristic graph [11]. Clearly, an encoder is feasible if and only if it corresponds to a proper *coloring* of G. Further, we want to find colorings which minimize average case complexity, i.e., colorings which minimize the entropy of colors. The minimum entropy of colors is called the *chromatic entropy* of the graph, denoted by $H_G^\chi(X)$. After finding the minimum entropy coloring, we can construct the Huffman code for the colors, which achieves an average code length within one bit of $H_G^\chi(X)$). When we consider a block of N measurements, the characteristic graph is the OR-graph $G^N := G \vee G \vee \ldots \vee G$, with chromatic entropy $H_{G^N}^\chi(X)$. Thus, the optimal coloring requires at most $\frac{H_{G^N}^\chi(X)+1}{N}$ bits per computation. From a result in [12], we know that $\lim_{N\to\infty} \frac{1}{N} H_{G^N}^\chi(X) = H_G(X)$, where $H_G(X)$ is the graph entropy or Korner entropy. Thus the asymptotic rate for zero-error function computation is $H_G(X)$. If, however, we allow a vanishing error-rate, the minimum rate required is the *conditional graph entropy* $H_G(X|Y)$ [40].

For a general graph, finding the minimum entropy coloring is NP-hard. However, under the simplifying condition $p(x,y) > 0$ for all $(x,y) \in \mathcal{X} \times \mathcal{Y}$, the graph becomes a complete k-partite graph, for which greedy coloring achieves a $\chi(G)$-coloring, where $\chi(G)$ is the chromatic number of the graph G. Further, any other proper $\chi(G)$-coloring is merely a relabelling of the colors in the greedy coloring. This is analogous to the result in Theorem 2. Thus the optimal coloring can be found in polynomial time, and the optimal encoder can be obtained by constructing a Huffman code for the colors in the optimal coloring. Further, by considering long blocks of N measurements, we can achieve an asymptotic rate of $H(c(X))$, while still ensuring zero error for the block. Here, $c(X)$ denotes the color class to which X belongs.

3.2 Function Computation on Directed Trees

Let us now consider computation on a *tree graph*. Consider a directed tree $G = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V} := \{v_1, v_2, \ldots, v_n\}$ and root node v_1. Edges represent communication links, so that node v_j can transmit to node v_i if $(v_j, v_i) \in \mathcal{E}$. Each node v_i makes a measurement $x_i \in \mathcal{X}_i$, and the collector node v_1 wants to compute a function $f(x_1, x_2, \ldots, x_n)$ with no error. We seek to minimize the worst case complexity on each edge.

For each node i, let $\pi(v_i)$ be the unique node to which node i has an outgoing edge, and let

$\mathcal{N}^-(v_i) := \{v_j \in \mathcal{V} : (v_j, v_i) \in \mathcal{E}\}$. The *height* of a node v_i is the length of the longest directed path from a leaf node to v_i. Define the *descendant set* $D(v_i)$ to be the subset of nodes in $\mathcal{V}$ from which there exist directed paths to node v_i. The graph induced on $D(v_i)$ is a tree with node v_i as root. Each node transmits exactly once and the computation proceeds in a bottom-up fashion, starting from the leaf nodes and proceeding up the tree. Each leaf node v_i has an encoder $\mathcal{C}_i : \mathcal{X}_i \to \{0,1\}^*$ that maps its measurement x_i to a codeword $\mathcal{C}_i(x_i)$ which is transmitted on the edge $(v_i, \pi(v_i))$. Each non-leaf node v_j for $j \neq 1$ has an encoder $\mathcal{C}_j$ which maps its measurement x_j as well as the codewords received from $\mathcal{N}^-(v_j)$, to a codeword transmitted on the edge $(v_j, \pi(v_j))$. Thus the computation proceeds in a bottom-up fashion. Let C_i denote the codeword transmitted by node v_i, and $C_S := \{C_i : v_i \in S\}$ denote the set of codewords transmitted by nodes in S. Thus the codeword C_i is implicitly a function of $x_{D(v_i)} := \{x_j : v_j \in D(v_i)\}$.

Definition 3. A set of encoders $\{\mathcal{C}_i : 2 \leq i \leq n\}$ is said to be feasible if there is a decoding function g_1 at the collector node v_1 such that $g(x_1, \mathcal{C}_{\mathcal{N}^-(v_1)}) = f(x_1, x_2, \ldots, x_n)$ for all $(x_1, x_2, \ldots, x_n) \in \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_n$.

Lemma 4. *If a set of encoders $\{\mathcal{C}_i : 2 \leq i \leq n\}$ is feasible, then the encoder $\mathcal{C}_i$ at node v_i must separate*[1] *$x^1_{D(v_i)} \in \mathcal{X}_{D(v_i)}$ from $x^2_{D(v_i)} \in \mathcal{X}_{D(v_i)}$, if there exists an assignment $x^*_{\mathcal{V}\setminus D(v_i)}$ such that $f(x^1_{D(v_i)}, x^*_{\mathcal{V}\setminus D(v_i)}) \neq f(x^2_{D(v_i)}, x^*_{\mathcal{V}\setminus D(v_i)})$.*

Proof: The removal of edge $(v_i, \pi(v_i))$ separates the graph into two disconnected subtrees $D(v_i)$ and $\mathcal{V}\setminus D(v_i)$. We combine all the nodes in $D(v_i)$ into a supernode v_α, and all the nodes in $\mathcal{V}\setminus D(v_i)$ into a supernode v_β. The result now follows from Theorem 1. $\square$

To prove the converse, we explicitly define the encoders $\mathcal{C}_2, \mathcal{C}_3, \ldots, \mathcal{C}_n$ and a decoding function g, and prove that it achieves correct function computation. Define the alphabet for encoder $\mathcal{C}_i$ on edge $(v_i, \pi(v_i))$ as,

$$\mathcal{A}_i := \{h_i : \mathcal{X}_{\mathcal{V}\setminus D(v_i)} \to \mathcal{D} \text{ s.t. } \exists x^*_{D(v_i)} \in \mathcal{X}_{D(v_i)} \text{ satisfying } h_i(x_{\mathcal{V}\setminus D(v_i)}) = f(x^*_{D(v_i)}, x_{\mathcal{V}\setminus D(v_i)}) \ \forall x_{\mathcal{V}\setminus D(v_i)} \in \mathcal{X}_{\mathcal{V}\setminus D(v_i)}\}.$$

[1]Node v_i does not have access to $x_{D(v_i)}$ directly but only the codewords received from $\mathcal{N}^-(v_i)$. When we say that the encoder $\mathcal{C}_i$ must separate $x_{D(v_i)}, \tilde{x}_{D(v_i)}$, we are considering $\mathcal{C}_i$ as an implicit function of $x_{D(v_i)}$.

Thus codewords sent by node v_i can be viewed as *normal forms* on variables $X_{\mathcal{V}\setminus D(v_i)}$, or as partial functions on $X_{\mathcal{V}\setminus D(v_i)}$.

Encoder at node v_i: On receiving the codeword corresponding to $h_j : \mathcal{X}_{\mathcal{V}\setminus D(v_j)} \to \mathcal{D}$, on incoming edge (v_j, v_i), node v_i assigns nominal values, $x^{nom}_{D(v_j)}$ to variables $X_{D(v_j)}$ such that

$$f(x^{nom}_{D(v_j)}, x_{\mathcal{V}\setminus D(v_j)}) = h_j(x_{\mathcal{V}\setminus D(v_j)}) \quad \forall x_{\mathcal{V}\setminus D(v_j)} \in \mathcal{X}_{\mathcal{V}\setminus D(v_j)}. \tag{3.1}$$

Given nominal values for all nodes in $D(v_i)\setminus\{v_i\}$, and its own measurement x_i, node v_i substitutes these values to obtain a function $h_i : \mathcal{X}_{\mathcal{V}\setminus D(v_i)} \to \mathcal{D}$ such that $h_i(x_{\mathcal{V}\setminus D(v_i)}) = f(x^{nom}_{D(v_i)\setminus\{v_i\}}, x_i, x_{\mathcal{V}\setminus D(v_i)})$ for all $x_{\mathcal{V}\setminus D(v_i)} \in \mathcal{X}_{\mathcal{V}\setminus D(v_i)}$.

If $v_i \neq v_1$, node v_i then transmits the codeword $\mathcal{C}_i$ corresponding to function $h_i \in \mathcal{A}_i$ on the edge $(v_i, \pi(v_i))$.

Decoding function g: The collector node v_1 assigns nominal values to the variables $X_{D(v_1)\setminus\{v_1\}}$. The decoding function g is given by $g(x_1, C_{\mathcal{N}^-(v_1)}) := h_1 = f(x_1, x^{nom}_{D(v_1)\setminus\{v_1\}})$.

Theorem 5. *Let $x_1^{fix}, x_2^{fix}, \ldots, x_n^{fix}$ be any fixed assignment of node values. Let the encoders at node $v_2, v_3, \ldots, v_n$ be as above. Then function h_i computed by node v_i is*

$$h_i(x_{\mathcal{V}\setminus D(v_i)}) = f(x^{fix}_{D(v(i))}, x_{\mathcal{V}\setminus D(v_i)}) \qquad \forall x_{\mathcal{V}\setminus D(v_i)} \in \mathcal{X}_{\mathcal{V}\setminus D(v_i)}.$$

Consequently the decoding function g satisfies $g(x_1^{fix}, \mathcal{C}_{\mathcal{N}^-(v_1)}) = f(x_1^{fix}, x_2^{fix}, \ldots, x_n^{fix})$.

Proof: The proof proceeds by induction. The theorem is trivially true for all leaf nodes v_i, since by assumption $h_i(x_{\mathcal{V}\setminus D(v_i)}) = f(x^{fix}_{v_i}, x_{\mathcal{V}\setminus D(v_i)})$ for all $x_{\mathcal{V}\setminus D(v_i)} \in \mathcal{X}_{\mathcal{V}\setminus D(v_i)}$. Suppose it is true for all nodes with height less than κ. Consider a node v_i with height κ. All the nodes in $\mathcal{N}^-(v_i)$ must have height less than κ. On receiving the codeword corresponding to h_j on edge (v_j, v_i), node v_i assigns nominal values to variables in $X_{D(v_j)}$ so that (3.1) is satisfied. From the induction assumption, we have

$$h_j(x_{\mathcal{V}\setminus D(v_j)}) = f(x^{fix}_{D(v_j)}, x_{\mathcal{V}\setminus D(v_j)}) \quad \forall x_{\mathcal{V}\setminus D(v_j)} \in \mathcal{X}_{\mathcal{V}\setminus D(v_j)}. \tag{3.2}$$

Since (3.2) is true for all $v_j \in \mathcal{N}^-(v_i)$, we can simultaneously substitute the nominal values $x^{nom}_{D(v_i)\setminus\{v_i\}}$ for the variables $X_{D(v_i)\setminus\{v_i\}}$ and the value x_i^{fix} for the variable $X_{\{v_i\}}$, to obtain a function

h_i satisfying

$$\begin{aligned} h_i(x_{\mathcal{V}\setminus D(v_i)}) &= f(x^{nom}_{D(v(i))\setminus\{v_i\}}, x^{fix}_{v_i}, x_{\mathcal{V}\setminus D(v_i)}) \quad \forall x_{\mathcal{V}\setminus D(v_i)} \\ &= f(x^{fix}_{D(v(i))}, x_{\mathcal{V}\setminus D(v_i)}) \qquad\qquad \forall x_{\mathcal{V}\setminus D(v_i)}, \end{aligned} \tag{3.3}$$

where (3.3) follows from (3.1) and (3.2). This establishes the induction step and completes the proof. For the special case of the collector node v_i, we have

$$g(x_1^{fix}, \mathcal{C}_{\mathcal{N}^-(v_1)}) = h_1 = f(x^{fix}_{D(v_1)}) = f(x_1^{fix}, x_2^{fix}, \ldots, x_n^{fix}).$$

Since this is true for every fixed assignment of the node values, we can achieve error-free computation of the function. Hence the set of encoders described above is feasible. □

For node v_i, consider the equivalence relation "$\leftrightarrow_i$" where $x^1_{D(v_i)} \leftrightarrow_i x^2_{D(v_i)}$ if $f(x^1_{D(v_i)}, x_{\mathcal{V}\setminus D(v_i)}) = f(x^2_{D(v_i)}, x_{\mathcal{V}\setminus D(v_i)})$ for all $x_{\mathcal{V}\setminus D(v_i)} \in \mathcal{X}_{\mathcal{V}\setminus D(v_i)}$. It is easy to check that the equivalence classes generated by $\leftrightarrow_i$ are captured exactly by the alphabet $\mathcal{A}_i$. Thus the above encoders use exactly the optimal alphabet. Hence, the minimum worst case complexity for encoder $\mathcal{C}_i$ is $\lceil \log(|\mathcal{A}_i|) \rceil$ on the edge $(v_i, \pi(v_i))$.

The extension to the case where node v_i collects a block of N independent measurements $\underline{X}_i \in \mathcal{X}_i^N$, and the collector node v_1 wants to compute the vector function $f^{(N)}(\underline{X}_1, \underline{X}_2, \ldots, \underline{X}_n)$, is straightforward. We can thus achieve a minimum worst case complexity arbitrarily close to $\log |\mathcal{A}_i|$ bits for encoder $\mathcal{C}_i$. It should be noted that the minimum worst case complexity of encoder $\mathcal{C}_i$ does not depend on the encoders of the other nodes.

If there is a probability distribution $p(X_1, X_2, \ldots, X_n)$ on the measurements, then we can obtain a necessary and sufficient condition by considering all edge cuts.

Lemma 6. *Consider a cut which partitions the nodes into S and $\mathcal{V} \setminus S$ with $v_1 \in \mathcal{V} \setminus S$. Let $\delta^+(S)$ be the set of all edges from nodes in S to nodes in $\mathcal{V} \setminus S$. Then the set of encoders $\{\mathcal{C}_i : 2 \leq i \leq n\}$ is feasible if and only if for every cut, the encoder on at least one of the edges in $\delta^+(S)$ separates $x^1_S, x^2_S \in \mathcal{X}_S$ if there exists an assignment $x^*_{\mathcal{V}\setminus S}$ such that $f(x^1_S, x^*_{\mathcal{V}\setminus S}) \neq f(x^2_S, x^*_{\mathcal{V}\setminus S})$ and $p(x^1_S, x^*_{\mathcal{V}\setminus S}) p(x^2_S, x^*_{\mathcal{V}\setminus S}) > 0$.*

Proof: Necessity is as before. For the converse, suppose the set of encoders is not feasible. Then there exist $(x_1^*, x^A_{\mathcal{V}\setminus v_1})$ and $(x_1^*, x^B_{\mathcal{V}\setminus v_1})$ such that

$$f(x_1^*, x^A_{\mathcal{V}\setminus v_1}) \neq f(x_1^*, x^B_{\mathcal{V}\setminus v_1}) \text{ and } p(x_1^*, x^A_{\mathcal{V}\setminus v_1})p(x_1^*, x^B_{\mathcal{V}\setminus v_1}) > 0.$$

However, the codewords received from nodes in $\mathcal{N}^-(v_1)$ are the same for both assignments. For the cut which separates v_1 from $\mathcal{V} \setminus v_1$, there is no encoder on $\delta^+(S)$ which separates $x^A_{\mathcal{V}\setminus v_1}$ and $x^B_{\mathcal{V}\setminus v_1}$. □

The above proof of the converse is not constructive. The construction is much harder now since the encoders are coupled, as shown by the following example.

Example 7. *Consider the three node network $G = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{v_1, v_2, v_3\}$ and $\mathcal{E} = \{(v_2, v_1), (v_3, v_1)\}$ (see Figure 3.1(a)). Let $\mathcal{X}_1 = \{x^{1a}\}, \mathcal{X}_2 = \{x^{2a}, x^{2b}\}, \mathcal{X}_3 = \{x^{3a}, x^{3b}\}$. Let us suppose $p(x^{1a}, x^{2a}, x^{3a}) = p(x^{1a}, x^{2b}, x^{3b}) = \frac{1}{2}$. The function is given by $f(X_1, X_2, X_3) = (X_1, X_2, X_3)$. Considering the cut $(\{v_2, v_3\}, \{v_1\})$, either v_2 or v_3 needs to separate its two values. Thus the two encoders are no longer independent.*

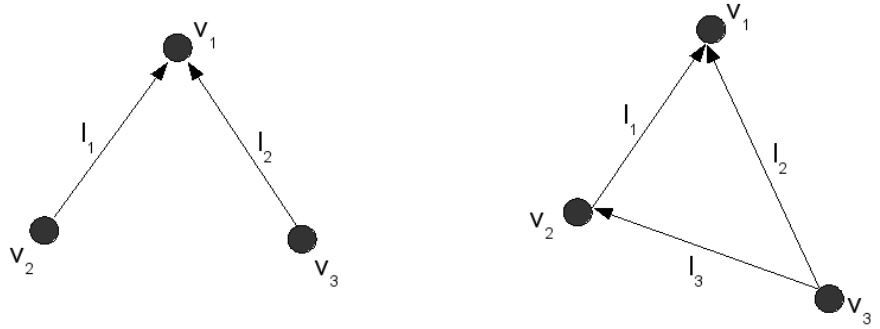

Figure 3.1: Two simple networks of Examples 1 and 2

In general, we can trade off between the encoders on different edges and it may not be possible to simultaneously minimize the average description lengths of all encoders. However, if we assume that $p(x_1, x_2, \ldots, x_n) > 0$ for all $(x_1, x_2, \ldots, x_n)$, we can separately minimize the average description length of each encoder. The optimal encoder constructs a Huffman code on the optimal alphabet $\mathcal{A}_i$. Suppose $\underline{q}_i$ is the probability vector induced on the alphabet $\mathcal{A}_i$. Then, by taking long blocks

of measurements, we can achieve a minimum average case complexity arbitrarily close to $H(\underline{q}_i)$ for encoder $\mathcal{C}_i$.

It is worth noting that we have considered zero error function computation, and shown optimal strategies for function computation on tree graphs, where node measurements may be correlated, but $p(x_1, x_2, \ldots, x_n) > 0$ for all $(x_1, x_2, \ldots, x_n)$. In contrast, in the vanishing error paradigm, the extension of Slepian-Wolf coding to tree graphs with more than three nodes is not known. As an aside, we note that under some additional assumptions, we can do recursive Slepian-Wolf coding to obtain the rate region for computing the function $f(X_1, X_2, \ldots, X_n) = (X_1, X_2, \ldots, X_n)$.

Theorem 8 (Distributed Source Coding on a Tree)**.** *Consider a directed tree $G = (\mathcal{V}, \mathcal{E})$ with collector node v_1. The node measurements $X_1, X_2, \ldots, X_n$ are drawn from the probability distribution $p(X_1, X_2, \ldots, X_n)$ which satisfies the following condition: if $(v_i, v_j), (v_j, v_k)$ are edges in the undirected tree $\tilde{G}$, then X_i is conditionally independent of X_k, given X_j. That is, the edges of the undirected tree $\tilde{G}$ specify Markov chain constraints on the variables $(X_1, X_2, \ldots X_n)$. If R_i is the rate of the encoder on the edge $(v_i, \pi(v_i))$, then the rate region is described by*

$$R_i \geq H(X_{D(v_i)} | X_{\{\pi(v_i)\}}) \text{ for every node } v_i \in \mathcal{V} \setminus \{v_1\}$$

3.3 Function Computation on Directed Acyclic Graphs

The extension from trees to directed acyclic graphs presents significant challenges, since there is no longer a unique path from every node to the collector. Consider a weakly connected directed acyclic graph (DAG) $G = (\mathcal{V}, \mathcal{E})$, where each node v_i collects a block of N measurements $\underline{X}_i \in \mathcal{X}_i^N$. The collector node v_1 is the unique node with only incoming edges, which wants to compute the vector function $f^N(\underline{X}_1, \underline{X}_2, \ldots, \underline{X}_n)$ with zero error.

Let the encoder mapping on edge (v_j, v_i) be denoted by $\mathcal{C}_{ji}^N$, which maps the measurement vector $\underline{X}_j$ and the codewords received thus far, to a codeword transmitted on edge (v_j, v_i). Since there are no cycles in G, function computation proceeds in a bottom-up fashion. Node v_i receives codewords $\mathcal{C}_{ji}^N$ on each incoming edge (v_j, v_i) and then transmits a codeword $\mathcal{C}_{ik}$ on each outgoing edge (v_i, v_k). A set of encoders is said to be feasible if there is a decoding function at the collector node v_1 which maps the received codewords to the correct function value. Let $l_{wc}(\mathcal{C}_{ij}^N)$ and $l_{avg}(\mathcal{C}_{ij}^N)$ denote the

worst case and average case complexity, respectively, of the encoder $\mathcal{C}_{ij}^N$. The rate of encoder $\mathcal{C}_{ij}^N$ is given by

$$R_{wc}(\mathcal{C}_{ij}^N) = \frac{l_{wc}(\mathcal{C}_{ij}^N)}{N} \text{ and } R_{avg}(\mathcal{C}_{ij}^N) = \frac{l_{avg}(\mathcal{C}_{ij}^N)}{N}.$$

Thus we can assign a rate vector in $\mathbf{R}^{|\mathcal{E}|}$ to every feasible set of encoders. Let $\mathcal{R}_{wc}^{(N)}$ in the worst case (or $\mathcal{R}_{avg}^{(N)}$ in the average case) be the set of feasible rate vectors for encoders of block length N. Then the rate region $\mathcal{R}_{wc}$ (or $\mathcal{R}_{avg}$) is given by the closure in $\mathbf{R}^{|\mathcal{E}|}$ of the finite block length rate vectors:

$$\mathcal{R}_{wc} := \overline{\bigcup_{N\geq 1} \mathcal{R}_{wc}^{(N)}} \text{ and } \mathcal{R}_{avg} := \overline{\bigcup_{N\geq 1} \mathcal{R}_{avg}^{(N)}}.$$

Consider the following example.

Example 9. *We have three nodes $\{v_1, v_2, v_3\}$ connected as shown in Figure 3.1(b). Let $\mathcal{X}_1 = \mathcal{X}_2 = \mathcal{X}_3 = \{0, 1, 2, 3\}$, and suppose node v_1 wants to compute $f(X_1, X_2, X_3) = (X_1 + X_2 + X_3) mod 4$. It is easy to check that $(2, 0, 2)$ and $(2, 2, 0)$ are feasible rate vectors for (l_1, l_2, l_3). These are rate vectors associated with the two tree subgraphs. Further, one can also check that $(2, 1, 1)$ is also feasible. This is achieved by node 3 dividing its measurement by 2, and transmitting the quotient on l_3 and the remainder on l_2. Alternately, in the block computation setting, the rate $(2, 1, 1)$ is achievable by time-sharing between $(2, 0, 2)$ and $(2, 2, 0)$.*

3.3.1 Outer bound on the rate region

Consider any cut of the graph G which partitions nodes into subsets S and $\mathcal{V} \setminus S$ with $v_1 \in \mathcal{V} \setminus S$. Let $\delta^+(S)$ be the set of edges from some node in S to some node in $\mathcal{V} \setminus S$.

Lemma 10. *Consider a set of encoders which achieve error free block function computation with rate vector $\{R_{wc}(i, j)\}_{(v_i,v_j)\in\mathcal{E}}$. Given any assignments $\underline{x}_S^1$ and $\underline{x}_S^2$ of the nodes in S, if there exists an assignment $\underline{x}_{\mathcal{V}\setminus S}$ such that $f^{(N)}(\underline{x}_{\mathcal{V}\setminus S}, \underline{x}_S^1) \neq f^{(N)}(\underline{x}_{\mathcal{V}\setminus S}, \underline{x}_S^2)$, then the encoders on at least one of the edges in $\delta^+(S)$ must separate $\underline{x}_S^1$ and $\underline{x}_S^2$.*

(i) *In the worst case block computation scenario, an outer bound on the rate region is given by*

$$\sum_{(v_i,v_j)\in\delta^+(S)} R_{ij} \geq \log |\Pi(\mathcal{C}_S^1)| \text{ for all cuts } (S, \mathcal{V} \setminus S),$$

where $\Pi(\mathcal{C}_S^1)$ is the partition of $\mathcal{X}_S$ into the appropriate equivalence classes.

(ii) *Suppose we have a probability distribution with $p(X_1, X_2, \ldots, X_n) > 0$. Given a cut $(S, \mathcal{V} \setminus S)$, let $R \subset \mathcal{V} \setminus S$ be the subset of nodes which have a directed path to some node in S. In the average case block computation scenario, an outer bound on the rate region is given by*

$$\sum_{(v_i,v_j)\in\delta^+(S)} R_{ij} \geq H([X_S]|X_R) \text{ for all cuts } (S, \mathcal{V} \setminus S),$$

where $[X_S]|X_R$ is the equivalence class to which X_S belongs, given X_R and a particular function.

3.3.2 Achievable region

Lemma 11. *Consider any directed tree subgraph G_T with root node v_1. Let us suppose that only the edges in G_T can be used for communication. Then we can construct encoders on each edge, which minimize worst case or average case complexity. The rate vector corresponding to a tree G_T is the limit of the rate vectors for the optimal finite block length encoders for G_T. Thus, for a given tree G_T:*

(i) *The worst case rate vector corresponding to the tree G_T is an extreme point of the worst case rate region $\mathcal{R}_{wc}$.*

(ii) *If $p(x_1, x_2, \ldots, x_n) > 0$ for all $(x_1, x_2, \ldots, x_n)$, the rate vector corresponding to the tree G_T is an extreme point of the average case rate region $\mathcal{R}_{avg}$.*

The convex hull of the rate points corresponding to trees is achievable. However, we do not know if these are the *only* extreme points of the rate region $\mathcal{R}$.

3.3.3 Some examples

Example 12 (Arithmetic Sum)**.** *Consider three nodes v_1, v_2, v_3 connected as in Figure 3.1(b). Let $\mathcal{X}_2 = \mathcal{X}_3 = \{0, 1\}$, with node v_1 having no measurements. Suppose node v_1 wants to compute $f(X_1, X_2, X_3) = X_2 + X_3$. Let (R_{21}, R_{31}, R_{32}) be the rate vector associated with edges (l_1, l_2, l_3). The outer bound on $\mathcal{R}_{wc}$ is:*

$$R_{21} \geq 1; \quad R_{21} + R_{31} \geq \log 3; \quad R_{32} + R_{31} \geq 1.$$

The subset of the rate region achievable by trees is:

$$R_{21} = \lambda + (1-\lambda)\log 3, R_{31} = \lambda, R_{32} = (1-\lambda) \text{ for } 0 \leq \lambda \leq 1.$$

Suppose that X_1, X_2 are i.i.d. with $p(X_1 = 0) = p(X_1 = 1) = 0.5$. The outer bound on $\mathcal{R}_{avg}$ is:

$$R_{21} \geq 1; \quad R_{21} + R_{31} \geq \frac{3}{2}; \quad R_{32} + R_{31} \geq 1.$$

The subset of the rate region achievable by trees is:

$$R_{21} = \lambda + (1-\lambda)\frac{3}{2}, R_{31} = \lambda, R_{32} = (1-\lambda) \text{ for } 0 \leq \lambda \leq 1.$$

Example 13 (Finite field parity)**.** *Let $\mathcal{X}_i = \{0, 1, \ldots, D-1\}$ for each node v_i. Suppose the collector node v_1 wants to compute the function $(X_1 + X_2 + \ldots + X_n) \bmod D$. In this case, the outer bound on the worst case rate region described in Lemma 10 is tight. Indeed, since the set of all outgoing links from a node is a valid cut, we have $\sum_{(v_i,v_j)\in\mathcal{E}} R_{ij} \geq \log_2 D$*

An obvious achievable strategy is for every leaf node v_i to split its block and transmit it on the outgoing edges from v_i. Next, we move to a node at height 1. This node receives partial blocks from various leaf nodes, and can hence compute an intermediate parity for some instances of the block. It then splits its block along the various outgoing edges. The crucial point is that the worst case description length per instance remains $\log_2 D$. Proceeding recursively up the DAG, we see that we can achieve the outer bound.

Example 14 (Max/Min)**.** *Let $\mathcal{X}_i = \{0, 1, \ldots, D-1\}$ for each node v_i. Suppose the collector node v_1 wants to compute $max(X_1, X_2, \ldots, X_n)$. The outer bound to the worst case rate region described in Lemma 10 is tight. The achievable strategy is similar to the parity case, where nodes compute intermediate maximum values and split their blocks on the outgoing edges. Once again, we utilize the fact that the range of the Max function remains constant irrespective of the number of nodes.*

3.4 Concluding Remarks

We have addressed the problem of zero error function computation on graphs, and analyzed both worst case and average case metrics. We have provided necessary and sufficient conditions for the computation of a general function of correlated measurements on a directed acyclic graph. For tree graphs, this leads to finding the optimal encoders on each edge. For general DAGs, we have provided an outer bound on the rate region, and an achievable region based on aggregating along subtrees.

CHAPTER 4

COMPUTING SYMMETRIC BOOLEAN FUNCTIONS IN UNDIRECTED GRAPHS

In this chapter, we consider a network where each node has a Boolean variable and *all* nodes want to compute a given symmetric Boolean function. We abstract out the medium access control problem, and view the network as an undirected graph with edges representing essentially noiseless wired links between nodes. We adopt a deterministic formulation of the problem of function computation, requiring zero error. We consider the problem of worst-case function computation, without imposing a probability distribution on the node measurements. As in the previous chapter, we allow nodes to accumulate a block of N measurements, and realize greater efficiency by using block codes.

The key difference from Chapter 3 is that we consider undirected graphs with bidirectional links which allow interactive information exchange. Thus, the set of admissible strategies includes all interactive strategies, where a node may exchange several messages with other nodes, with node i's transmission being allowed to depend on all previous transmissions heard by node i, and node i's block of measurements. Our objective is to determine the set of rate vectors that achieve zero error block computation. In essence, we are exploring how the function structure and the structure of the graph affect the communication strategy. We can further study the benefit of interactive strategies versus single-round strategies. In particular, we would like to study the nature of interaction that is appropriate on a graph.

In this chapter, we are particularly interested in computing functions which depend only on the sum of nodes' measurements. We begin by reviewing a toy problem from [20] where the exact communication complexity of the AND function of two variables is shown to be $\log_2 3$ bits, for block computation. The lower bound was established using fooling sets, and a novel achievable scheme was presented which minimizes the worst case total number of bits exchanged.

In Section 4.1, we generalize the approach described in [20] to the two node problem where each

node i has an integer variable X_i and both nodes want to compute a function $f(X_1, X_2)$ which only depends on $X_1 + X_2$. Once again, we establish a lower bound by constructing an appropriate fooling set. For achievability, we devise a single-round strategy as follows. Node 1 constructs an effective alphabet based on separation requirements of the function, and then uses a prefix-free codebook. Node 2 simply replies with the value of the function block. The codebook for node 1 is chosen so as to minimize the worst-case total number of bits exchanged. The existence of this codebook is guaranteed by the Kraft inequality. We also construct a similar achievable strategy starting with node 2. We define a class of functions called sum-threshold functions, which evaluate to 1 if $X_1 + X_2$ exceeds a threshold. For this class, we show that this single-round strategy is indeed optimal. However, for a class of functions called sum-interval functions, which evaluate to 1 if $a \leq X_1 + X_2 \leq b$, the upper and lower bounds do not match. However, the achievable strategy involving separation, followed by coding, can be used for any general function.

In Section 4.2, we consider Boolean symmetric function computation on trees. Since every edge is a cut-edge, we can obtain a lower bound for the number of bits that must be exchanged on an edge, by reducing it to a two node problem. This lower bound is, in fact, a cut-set bound. For the class of sum-threshold functions, we construct an achievable strategy which achieves the cut-set bound. In fact, we use exactly the same achievable strategy as in the two node example, but we order the transmissions so that information flows up the tree to the root node and then back down the tree to the leaves. This is reminiscent of message passing algorithms. This result can be generalized to the case where nodes have non-binary measurements, and want to compute a sum-threshold function.

In Section 4.3, for general graphs, we can still derive a cut-set lower bound by considering all partitions of the vertices. We also propose an achievable scheme that consists of activating a subtree of edges and using the optimal strategy for transmissions on the tree. While the upper and lower bounds do not seem to match even for very simple functions, for complete graphs, we are able to show that the minimum symmetric rate point which satisfies the cut-set constraints is no less than one-half of the symmetric rate point achieved by employing star graphs. Thus, for complete graphs, aggregation along trees provides a 2-OPT solution.

4.1 The Two Node Problem

Our objective in this chapter is to derive provably optimal strategies for the computation of symmetric Boolean functions in tree networks, and, further, in general network topologies. The value of a symmetric Boolean function of n variables only depends on the number of 1s, or viewed differently, the sum of the n variables. A key component in the solution of the main problem is the understanding of the two node problem.

Consider two nodes 1 and 2 with variables $X_1 \in \{0, 1, \ldots, m_1\}$ and $X_2 \in \{0, 1, \ldots, m_2\}$. Both nodes wish to compute a function $f(X_1, X_2)$ which only depends on the value of $X_1 + X_2$. To put this in context, one can suppose there are m_1 Boolean variables collocated at node 1 and m_2 Boolean variables at node 2, and both nodes wish to compute a symmetric Boolean function of the $n := m_1 + m_2$ variables. We pose the problem in a block computation setting, where each node i has a block of N independent measurements, denoted by X_i^N. We consider the class of all interactive strategies, where nodes 1 and 2 transmit messages alternately with the value of each subsequent message being allowed to depend on all previous transmissions, and the block of measurements available at the transmitting node. We define a round to include one transmission by each node. A strategy is said to achieve correct block computation if for *every* choice of input (X_1^N, X_2^N), each node i can correctly decode the value of the function block $f^N(X_1, X_2)$ using the sequence of transmissions $b_1, b_2, \ldots$ and its own measurement block X_i^N. This is the direct-sum problem in communication complexity.

Let $\mathcal{S}_N$ be the set of strategies for block length N, which achieve zero-error block computation, and let $C(f, S_N, N)$ be the worst-case total number of bits exchanged under strategy $S_N \in \mathcal{S}_N$. The worst-case per-instance complexity of computing a function $f(X_1, X_2)$ is defined as

$$C(f) := \lim_{N \to \infty} \min_{S_N \in \mathcal{S}_N} \frac{C(f, S_N, N)}{N}.$$

4.1.1 Complexity of sum-threshold functions

In this chapter, we are interested in functions $f(X_1, X_2)$ which only depend on $X_1 + X_2$. Let us suppose without loss of generality that $m_1 \leq m_2$. We define an interesting class of $\{0, 1\}$-valued

functions called sum-threshold functions.

Definition 4 (sum-threshold functions)**.** A sum-threshold function $\Pi_\theta(X_1, X_2)$ with threshold θ is defined as follows:

$$\Pi_\theta(X_1, X_2) = \begin{cases} 1 & \text{if } X_1 + X_2, \geq \theta \\ 0 & \text{otherwise.} \end{cases}$$

For the special case where $m_1 = 1, m_2 = 1$ and $\theta = 2$, we recover the Boolean AND function, which was studied in [20]. It is critical to understand this problem before we can address the general problem of computing symmetric Boolean functions. Consider two nodes with measurement blocks $X_1^N \in \{0,1\}^N$ and $X_2^N \in \{0,1\}^N$, which want to compute the element-wise AND of the two blocks, denoted by $\wedge^N(X_1, X_2)$.

Theorem 15. *Given any strategy S_N for block computation of $X_1 \wedge X_2$,*

$$C(X_1 \wedge X_2, S_N, N) \geq N \log_2 3.$$

Further, there exists a strategy S_N^* which satisfies

$$C(X_1 \wedge X_2, S_N^*, N) \leq \lceil N \log_2 3 \rceil.$$

Thus, the complexity of computing $X_1 \wedge X_2$ is given by $C(X_1 \wedge X_2) = log_2 3$.

Proof of achievability: Suppose node 1 transmits first using a prefix-free codebook. Let the length of the codeword transmitted be $l(X_1^N)$. At the end of this transmission, both nodes know the value of the function at the instances where $X_1 = 0$. Thus node 2 only needs to indicate its bits for the instances of the block where $X_1 = 1$. Thus the total number of bits exchanged under this scheme is $l(X_1^N) + w(X_1^N)$, where $w(X_1^N)$ is the number of 1s in X_1^N. For a given scheme, let us define

$$L := \max_{X_1^N} (l(X_1^N) + w(X_1^N))$$

to be the worst case total number of bits exchanged. We are interested in finding the codebook which will result in the minimum worst-case number of bits.

Any prefix-free code must satisfy the Kraft inequality given by $\sum_{X_1^N} 2^{-l(X_1^N)} \leq 1$. Consider a codebook with $l(X_1^N) = \lceil N \log_2 3 \rceil - w(x_1^N)$. This satisfies the Kraft inequality since $\sum_{X_1^N} w(X_1^N) = 3^N$. Hence there exists a valid prefix free code for which the worst case number of bits exchanged is $\lceil N \log_2 3 \rceil$, which establishes that $C(X_1 \wedge X_2) \leq \log_2 3$.

The lower bound is shown by constructing a *fooling set* [8] of the appropriate size. We digress briefly to introduce the concept of fooling sets in the context of two-party communication complexity [8]. Consider two nodes X and Y, each of which take values in finite sets $\mathcal{X}$ and $\mathcal{Y}$, and both nodes want to compute some function $f(X, Y)$ with zero error.

Definition 5 (Fooling Set)**.** A set $E \subseteq \mathcal{X} \times \mathcal{Y}$ is said to be a fooling set, if for any two distinct elements $(x_1, y_1), (x_2, y_2)$ in E, we have either

- $f(x_1, y_1) \neq f(x_2, y_2)$, or
- $f(x_1, y_1) = f(x_2, y_2)$, but either $f(x_1, y_2) \neq f(x_1, y_1)$ or $f(x_2, y_1) \neq f(x_1, y_1)$.

Given a fooling set E for a function $f(X_1, X_2)$, we have $C(f(X_1, X_2)) \geq \log_2 |E|$. We have described two dimensional fooling sets above. The extension to multi-dimensional fooling sets is straightforward and gives a lower bound on the communication complexity of the function $f(X_1, X_2, \ldots, X_n)$. We will encounter multi-dimensional fooling sets in Chapter 5.

Lower bound for Theorem 15: We define the measurement matrix M to be the matrix obtained by stacking the row X_1^N over the row X_2^N. Thus we need to find a subset of the set of all measurement matrices which forms a fooling set. Let E be the set of all measurement matrices which are made up of only the column vectors $\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix} \}$. We claim that E is the appropriate fooling set. Consider two distinct measurement matrices $M_1, M_2 \in E$. Let $f^N(M_1)$ and $f^N(M_2)$ be the block function values obtained from these two matrices. If $f^N(M_1) \neq f^N(M_2)$, we are done. Let us suppose $f^N(M_1) = f^N(M_2)$ and since $M_1 \neq M_2$, there must exist one column where M_1 has $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ but M_2 has $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$. Now if we replace the first row of M_1 with the first row of M_2, the resulting measurement matrix, say M^*, is such that $f(M^*) \neq f(M_1)$. Thus, the set E is a valid fooling set. It is easy to verify that the E has cardinality 3^N. Thus, for *any* strategy

$S_N \in \mathcal{S}_N$, we must have $C(X_1 \wedge X_2, S_N, N) \geq N \log_2 3$, implying that $C(X_1 \wedge X_2) \geq \log_2 3$. This concludes the proof of Theorem 15. $\square$

We now return to the general two node problem with $X_1 \in \{0, 1, \ldots, m_1\}$ and $X_2 \in \{0, 1, \ldots, m_2\}$ and the sum-threshold function $\Pi_\theta(X_1, X_2)$. We will extend the approach presented above to this general scenario.

Theorem 16. *Given any strategy S_N for block computation of the function $\Pi_\theta(X_1, X_2)$,*

$$C(\Pi_\theta(X_1, X_2), S_N, N) \geq N \log_2\{\min(2\theta + 1, 2m_1 + 2, 2(n - \theta + 1) + 1)\}.$$

Further, there exist single-round strategies S_N^* and S_N^{**}, starting with nodes 1 and 2 respectively, which satisfy

$$C(\Pi_\theta(X_1, X_2), S_N^*, N) \leq \lceil N \log_2\{\min(2\theta + 1, 2m_1 + 2, 2(n - \theta + 1) + 1)\}\rceil.$$

$$C(\Pi_\theta(X_1, X_2), S_N^{**}, N) \leq \lceil N \log_2\{\min(2\theta + 1, 2m_1 + 2, 2(n - \theta + 1) + 1)\}\rceil.$$

Thus, the complexity of computing $\Pi_\theta(X_1, X_2)$ is given by $C(\Pi_\theta(X_1, X_2)) = \log_2\{\min(2\theta+1, 2m_1+2, 2(n-\theta+1)+1)\}$.

Proof of achievability: We consider three cases:

(a) Suppose $\theta \leq m_1 \leq m_2$. We specify a strategy S_N^* in which node 1 transmits first. We begin by observing that inputs $X_1 = \theta, X_1 = (\theta + 1) \ldots, X_1 = m_1$ need not be *separated*. By this we mean that, for each of these values of X_1, $\Pi_\theta(X_1, X_2) = 1$ for all values of X_2. Loosely speaking, we can replace any instance of $X_1 \geq \theta$ with $X_1 = \theta$ without affecting the value of the function. Thus node 1 has an effective alphabet of $\{0, 1, \ldots, \theta\}$. Suppose node 1 transmits using a prefix-free codebook on this reduced alphabet. Let the length of the codeword transmitted be $l(X_1^N)$. At the end of this transmission, node 2 only needs to indicate one bit for the instances of the block where $X_1 = 0, 1, \ldots, (\theta - 1)$. Thus the worst-case total number of bits is

$$L := \max_{X_1^N}(l(X_1^N) + w^0(X_1^N) + w^1(X_1^N) + \ldots + w^{\theta-1}(X_1^N)),$$

where $w^j(X_1^N)$ is the number of instances in the block where $X_1 = j$. We are interested in

finding the codebook which will result in the minimum worst-case number of bits. From the Kraft inequality for prefix-free codes we have

$$\sum_{X_1^N \in \{0,1,\ldots,\theta\}^N} 2^{-l(X_1^N)} \leq 1 \Rightarrow \sum_{X_1^N \in \{0,1,\ldots,\theta\}^N} 2^{-L+w^0(X_1^N)+w^1(X_1^N)+\ldots+w^{\theta-1}(X_1^N))} \leq 1.$$

Consider a codebook with $l(X_1^N) = \lceil N \log_2(2\theta+1) \rceil - w(x_1^N)$. This satisfies the Kraft inequality since

$$\sum_{X_1^N \in \{0,1,\ldots,\theta\}^N} 2^{w^0(X_1^N)+w^1(X_1^N)+\ldots+w^{\theta-1}(X_1^N))} . 1^{w^\theta(X_1^N)} = (2\theta+1)^N.$$

Hence there exists a valid prefix-free code for which the worst-case number of bits exchanged is $\lceil N \log_2(2\theta+1) \rceil$. Since $\theta \leq m_1 \leq m_2$, we have

$$C(\Pi_\theta(X_1, X_2), S_N^*, N) \leq \lceil N \log_2\{\min(2\theta+1, 2m_1+2, 2(n-\theta+1)+1)\} \rceil.$$

Clearly the strategy S_N^* has exactly one round. An analogous achievable strategy S_N^{**} can be derived with node 2 transmitting first. Again, node 2 does not need to separate the inputs $X_2 = \theta, (\theta+1), \ldots, m_2$, yielding an effective alphabet of $\{0, 1, \ldots, \theta\}$, which gives

$$C(\Pi_\theta(X_1, X_2), S_N^{**}, N) \leq \lceil N \log_2(2\theta+1) \rceil.$$

(b) Suppose $m_1 \leq m_2 \leq \theta$. Consider a strategy S_N^* in which node 1 transmits first. We begin by observing that inputs $X_1 = 0, X_1 = 1, \ldots, X_1 = \theta - m_2 - 1$ need not be *separated* since for each of these values of X_1, $\Pi_\theta(X_1, X_2) = 0$ for all values of X_2. Thus node 1 has an effective alphabet of $\{\theta - m_2 - 1, \theta - m_2, \ldots, m_1\}$. Suppose node 1 transmits using a prefix-free codebook on this reduced alphabet. Let the length of the codeword transmitted be $l(X_1^N)$. At the end of this transmission, node 2 only needs to indicate one bit for the instances of the block where $X_1 = \theta - m_2, \ldots, m_1$. Thus the worst-case total number of bits is

$$L := \max_{X_1^N}(l(X_1^N) + w^{\theta-m_2}(X_1^N) + \ldots + w^{m_1}(X_1^N)).$$

From the Kraft inequality for prefix-free codes we have

$$\sum_{X_1^N \in \{\theta-m_1-1, \theta-m_1, \ldots, m_1\}^N} 2^{-L+w^{\theta-m_2}(X_1^N)+\ldots+w^{m_1}(X_1^N)} \leq 1.$$

Consider a codebook with $l(X_1^N) = \lceil N \log_2(2(m_1+m_2-\theta+1)+1)\rceil - w^{\theta-m_2}(X_1^N) - \ldots - w^{m_1}(X_1^N)$. This satisfies the Kraft inequality since

$$\sum_{X_1^N \in \{\theta-m_2-1, \theta-m_2, \ldots, m_1\}^N} 2^{w^{\theta-m_2}(X_1^N)+\ldots+w^{m_1}(X_1^N)} . 1^{w^{\theta-m_2-1}(X_1^N)} = (2(m_1+m_2-\theta+1)+1)^N.$$

Hence there exists a valid prefix free code for which $L = \lceil N \log_2(2(n-\theta+1)+1)\rceil$. Since $m_1 \leq m_2 \leq \theta$, we have that

$$C(\Pi_\theta(X_1, X_2), S_N^*, N) \leq \lceil N \log_2\{\min(2\theta+1, 2m_1+2, 2(n-\theta+1)+1)\}\rceil.$$

Clearly, S_N^* is a single-round strategy, and the strategy S_N^{**} which starts with node 2 can be analogously derived.

(c) Suppose $m_1 \leq \theta \leq m_2$. For the case where node 1 transmits first, we construct a trivial strategy S_N^* where node 1 uses a codeword of length $\lceil N \log_2(m_1+1)\rceil$ bits and node 2 replies with a string of N bits indicating the function block. Thus we have $C(\Pi_\theta(X_1, X_2), S_N^*, N) \leq \lceil N \log_2(2m_1+2)\rceil$.

Now consider a strategy S_N^{**} where node 2 transmits first. Observe that the inputs $X_2 = 0, X_2 = 1, \ldots, X_2 = \theta - m_1 - 1$ need not be separated since for each of these values of X_2, $\Pi_\theta(X_1, X_2) = 0$ for all values of X_2. Further, the inputs $X_2 = \theta, X_2 = \theta+1, \ldots, X_2 = m_2$ need not be *separated*. Thus node 1 has an effective alphabet of $\{\theta - m_1 - 1, \theta - m_1, \ldots, \theta\}$. Suppose node 2 transmits using a prefix-free codebook on this reduced alphabet. Then, node 1 only needs to indicate one bit for the instances of the block where $X_1 = \theta - m_1, \ldots, \theta - 1$. Thus the worst-case total number of bits is

$$L := \max_{X_2^N}(l(X_2^N) + w^{\theta-m_1}(X_2^N) + \ldots + w^{\theta-1}(X_2^N)).$$

From the Kraft inequality for prefix-free codes we have

$$\sum_{X_1^N \in \{\theta-m_1-1,\theta-m_1,\ldots,\theta\}^N} 2^{-L+w^{\theta-m_1}(X_1^N)+\ldots+w^{\theta}(X_1^N)} \leq 1.$$

Consider a codebook with $l(X_1^N) = \lceil N \log_2(2m_1+2) \rceil - w^{\theta-m_1}(X_1^N) - \ldots - w^{\theta-1}(X_1^N)$. This satisfies the Kraft inequality since

$$\sum_{X_1^N \in \{\theta-m_1-1,\theta-m_1,\ldots,\theta\}^N} 2^{w^{\theta-m_1}(X_1^N)+\ldots+w^{m_2}(X_1^N)} = (2m_1+2)^N.$$

Hence there exists a valid prefix-free code for which $L = \lceil N \log_2(2(n-\theta+1)+1) \rceil$. Since $m_1 \leq \theta \leq m_2$, we have that

$$C(\Pi_\theta(X_1, X_2), S_N^{**}, N) \leq \lceil N \log_2\{\min(2\theta+1, 2m_1+2, 2(n-\theta+1)+1)\} \rceil.$$

Proof of Lower Bound: The lower bound is shown by constructing a fooling set of the appropriate size. Define the measurement matrix M to be the matrix obtained by stacking the row X_1^N over the row X_2^N. We need to find a subset of the set of all measurement matrices, which form a fooling set. Let E denote the set of all measurement matrices which are made up only of the column vectors from the set

$$Z = \{ \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} : 0 \leq z_1 \leq m_1, 0 \leq z_2 \leq m_2, (\theta-1) \leq z_1+z_2 \leq \theta \}.$$

We claim that E is the appropriate fooling set. Consider two distinct measurement matrices $M_1, M_2 \in E$. Let $f^N(M_1)$ and $f^N(M_2)$ be the block function values obtained from these two matrices. If $f^N(M_1) \neq f^N(M_2)$, we are done. Let us suppose $f^N(M_1) = f^N(M_2)$, and note that since $M_1 \neq M_2$, there must exist one column where M_1 and M_2 differ. Suppose M_1 has $\begin{bmatrix} z_{1a} \\ z_{2a} \end{bmatrix}$ while M_2 has $\begin{bmatrix} z_{1b} \\ z_{2b} \end{bmatrix}$, where $z_{1a}+z_{2a} = z_{1b}+z_{2b}$. We can assume without loss of generality that $z_{1a} < z_{1b}$ and $z_{2a} > z_{2b}$.

- If $z_{1a} + z_{2a} = z_{1b} + z_{2b} = \theta - 1$, then the *diagonal* element $f(z_{1b}, z_{2a}) = 1$ since $z_{1b} + z_{2a} \geq \theta$. Thus, if we replace the first row of M_1 with the first row of M_2, the resulting measurement matrix, say M^*, is such that $f(M^*) \neq f(M_1)$.

- If $z_{1a} + z_{2a} = z_{1b} + z_{2b} = \theta$, then the *diagonal* element $f(z_{1a}, z_{2b}) = 0$ since $z_{1b} + z_{2a} < \theta$. Thus, if we replace the second row of M_1 with the second row of M_2, the resulting measurement matrix, say M^*, is such that $f(M^*) \neq f(M_1)$.

Thus, the set E is a valid fooling set. It is easy to verify that the E has cardinality $|Z|^N$. Thus, for *any* strategy $S_N \in \mathcal{S}_N$, we must have $C(\Pi_\theta(X_1, X_2), S_N, N) \geq N \log_2 |Z|$ implying that $C(f) \geq \log_2 |Z|$. We are left with the task of computing the cardinality of the set Z. This can be modeled as finding the sum of the coefficients of Y^θ and $Y^{\theta-1}$ in a carefully constructed polynomial:

$$\begin{aligned} |Z| &= \left[Y^\theta\right] + \left[Y^{\theta-1}\right](1 + Y + \ldots + Y^{m_1})(1 + Y + \ldots + Y^{m_2}) \\ &= \left[Y^\theta\right] + \left[Y^{\theta-1}\right] \frac{(1 - Y^{m_1+1})(1 - Y^{m_2+1})}{(1 - Y)^2}. \end{aligned}$$

Using the binomial expansion for $\frac{1}{(1-Y)^k}$ (see [60]),

$$|Z| = \left[Y^\theta\right] + \left[Y^{\theta-1}\right](1 - Y^{m_1+1})(1 - Y^{m_2+1}) \sum_{k=0}^{\infty} \left(\begin{array}{c} k+1 \\ 1 \end{array} \right) Y^k.$$

(a) Suppose $\theta \leq m_1 \leq m_2$. Then $|Z| = \theta + \theta + 1$.

(b) Suppose $m_1 \leq \theta \leq m_2$. Then $|Z| = (2\theta + 1) - \theta + m_1 - \theta + m_1 + 1 = 2m_1 + 2$.

(c) Suppose $m_1 \leq m_2 \leq \theta$. Then $|Z| = 2m_1 + 2 - 2\theta + 2m_2 + 1 = 2(n - \theta + 1) + 1$.

This completes the proof of Theorem 16. $\square$

4.1.2 Complexity of sum-interval functions

Definition 6 (sum-interval functions)**.** A sum-interval function $\Pi_{[a,b]}(X_1, X_2)$ on the interval $[a, b]$ is defined as follows:

$$\Pi_{[a,b]}(X_1, X_2) := \begin{cases} 1 & \text{if } a \leq X_1 + X_2 \leq b, \\ 0 & \text{otherwise.} \end{cases}$$

Theorem 17. *Given any strategy S_N for block computation of $\Pi_{[a,b]}(X_1, X_2)$ where $b \leq n/2$,*

$$C(\Pi_{[a,b]}(X_1, X_2), S_N, N) \geq N \log_2\{\min(2b - a + 3, m_1 + 1)\}.$$

Further, there exists a single-round strategy S_N^* which satisfies

$$C(\Pi_{[a,b]}(X_1, X_2), S_N^*, N) \leq \lceil N \log_2\{\min(2(b + 1) + 1, 2m_1 + 2)\}\rceil.$$

Thus, we have obtained the complexity of computing $\Pi_\theta(X_1, X_2)$ to within one bit.

Proof of Achievability:

(a) Suppose $b \leq m_1 \leq m_2$. Node 1 has an effective alphabet of $\{0, 1, \ldots, b+1\}$. Then the worst-case total number of bits exchanged is given by

$$L := \max_{X_1^N}(l(X_1^N) + w^0(X_1^N) + \ldots + w^b(X_1^N)).$$

From the Kraft inequality, we can obtain a prefix free codebook with $L = \lceil N \log_2(2b + 1) + 1)\rceil$. Thus we have

$$C(\Pi_{[a,b]}(X_1, X_2), S_N^*, N) \leq \lceil N \log_2(2(b + 1) + 1)\rceil.$$

(b) Suppose $m_1 \leq a \leq b \leq m_2$ or $a \leq m_1 \leq b \leq m_2$. In either of these scenarios, node 1 has an effective alphabet of $\{0, 1, \ldots, m_1\}$. Then the worst-case total number of bits exchanged is given by

$$L := \max_{X_1^N}(l(X_1^N) + w^{a-m_2}(X_1^N) + \ldots + w^{m_1}(X_1^N))$$

From the Kraft inequality, we can obtain a prefix free codebook with $L = \lceil N \log_2(2m_1 + 2)\rceil$. Thus we have $C(\Pi_{[a,b]}(X_1, X_2), S_N^*, N) \leq \lceil N \log_2(2m_1 + 2)\rceil$.

Proof of Lower Bound: We attempt to find a fooling subset E of the set of measurement matrices. Our first guess would be the set of measurement matrices which are composed of only column vectors which sum up to b or $b+1$. However we see that this is not necessarily a fooling set, because if $[z_{1a}, z_{2a}]^T$ and $[z_{1b}, z_{2b}]^T$ are two columns which sum to $b+1$, and if $z_{1a} \leq z_{1b}-(b-a+2)$, then neither of the diagonal elements evaluate to function value 1. Thus, we can pick a maximum of $(b-a+2)$ consecutive elements along the line $z_1 + z_2 = b+1$, and, as before, all the elements on the line $z_1 + z_2 = b$. It is easy to check that this modified set of columns indeed yields a fooling set of measurement matrices. Now we need to compute the number of such columns.

(a) Suppose $b \leq m_1 \leq m_2$. The number of columns which sum up to b is equal to $b+1$. Thus the size of the fooling set is given by $(2b-a+3)^N$.

(b) Suppose $a \leq m_1 \leq b \leq m_2$ or $m_1 \leq a \leq b \leq m_2$. The number of columns which sum up to b is equal to m_1+1 and the number of columns which sum up to $b+1$ is equal to m_1+1. Thus, the size of the fooling set is given by $\{(m_1+1) + \min(m_1+1, b-a+2)\}^N$.

4.1.3 A general strategy for achievability

The strategy for achievability used in Theorems 16 and 17 suggests an achievable scheme for any general function $f(X_1, X_2)$ of variables $X_1 \in \mathcal{X}_1$ and $X_2 \in \mathcal{X}_2$ which depends only on the value of $X_1 + X_2$. This is done in two stages.

- **Separation:** Two inputs x_{1a} and x_{1b} need not be *separated* if $f(x_{1a}, x_2) = f(x_{1b}, x_2)$ for all values x_2. By checking this condition for each pair (x_{1a}, x_{1b}), we can arrive at a partition of $\{0, 1 \ldots, m_1\}$ into equivalence classes, which can be considered a reduced alphabet, say $A := \{a_1, \ldots, a_l\}$.

- **Coding:** Let A_0 denote the subset of the alphabet A for which the function evaluates only to 0, irrespective of the value of X_2, and let A_1 denote the subset of A which always evaluates to 1. Clearly, from the equivalence class structure, we have $|A_0| \leq 1$ and $|A_1| \leq 1$. Using the Kraft inequality as in Theorems 16 and 17, we obtain a scheme S_N^* with complexity $\log_2(2l - |A_0| - |A_1|)$.

4.2 Computing Symmetric Boolean Functions on Tree Networks

Consider a tree graph $T = (V, E)$, with node set $V = \{0, 1, \ldots, n\}$ and edge set E. Each node i has a Boolean variable $X_i \in \{0, 1\}$, and every node wants to compute a given symmetric Boolean function $f(X_1, X_2, \ldots, X_n)$. Again, we allow for block computation and consider all strategies where nodes can transmit in any sequence with possible repititions, subject to:

- On any edge $e = (i, j)$, either node i transmits or node j transmits, or neither, and this is determined from the previous transmissions.
- Node i's transmission can depend on the previous transmissions and the measurement block X_i^N.

The following theorem provides a computation and communication strategy for sum-threshold functions that is optimal for each link.

Theorem 18. *Consider a tree network where we want to compute the function $\Pi_\theta(X_1, \ldots, X_n)$. Let us focus on a single edge $e \equiv (i, j)$ whose removal disconnects the graph into components A_e and $V \setminus A_e$, with $|A_e| \leq |V \setminus A_e|$. For any strategy $S_N \in \mathcal{S}_N$, the number of bits exchanged along edge $e \equiv (i, j)$, denoted by $C_e(\Pi_\theta(X_1, \ldots, X_n), S_N, N)$, is lower bounded by*

$$C_e(\Pi_\theta(X_1, \ldots, X_n), S_N, N) \geq N \log_2\{\min(2\theta + 1, 2|A_e| + 2, 2(n - \theta + 1) + 1)\}.$$

Further, there exists a strategy S_N^* such that for any edge e,

$$C_e(\Pi_\theta(X_1, \ldots, X_n), S_N^*, N) \leq \lceil N \log_2\{\min(2\theta + 1, 2|A_e| + 2, 2(n - \theta + 1) + 1)\}\rceil.$$

Thus the complexity of computing $\Pi_\theta(X_1, \ldots, X_n)$ is given by

$$C_e(\Pi_\theta(X_1, \ldots, X_n)) = \log_2\{\min(2\theta + 1, 2|A_e| + 2, 2(n - \theta + 1) + 1)\}.$$

Proof: Given a tree network T, every edge e is a cut edge. Consider an edge e whose removal creates components A_e and $V \setminus A_e$, with $|A_e| \leq |V \setminus A_e|$. Now let us aggregate the nodes in A_e and also those in $V \setminus A_e$, and view this as a problem with two nodes connected by edge e. Clearly

the complexity of computing the function $\Pi_\theta(X_{A_e}, X_{V\setminus A_e})$ is a lower bound on the worst-case total number of bits that must be exchanged on edge e under any strategy S_N. Hence we obtain

$$C_e(\Pi_\theta(X_1,\ldots,X_n), S_N, N) \geq N\log_2\{\min(2\theta+1, 2|A_e|+2, 2(n-\theta+1)+1)\}.$$

The achievable strategy S_N^* is derived from the achievable strategy for the two node case in Theorem 16. That is to say, the transmissions back and forth along any edge will be exactly the same as in the two node case. However, we need to orchestrate these transmissions so that conditions of causality are maintained. Pick any node, say r, to be the root. This induces a partial order on the tree network. We start with each leaf in the network transmitting its codeword to the parent. Once a parent node obtains a codeword from each of its children, it has knowledge of the number of 1s in its subtree, and can transmit a codeword to its parent. Thus codewords are transmitted from child nodes to parent nodes until the root is reached. The root can then compute the value of the function and now sends the appropriate replies to its children. The children then compute the function and send appropriate replies, and so on. This sequential strategy depends critically on the fact that, in the two node problem, we derived optimal strategies starting from either node. Under the strategy described above, for any edge e, the worst-case total number of bits exchanged is given by

$$C_e(\Pi_\theta(X_1,\ldots,X_n), S_N^*, N) \leq \lceil N\log_2\{\min(2\theta+1, 2|A_e|+2, 2(n-\theta+1)+1)\}\rceil.\square$$

Using Theorem 17, one can similarly derive an approximately optimal strategy for sum-interval functions, which we state here without proof.

Theorem 19. *Consider a tree network where we want to compute the function $\Pi_{[a,b]}(X_1,\ldots,X_n)$, with $b \leq \frac{n}{2}$. Let us focus on a single edge $e \equiv (i,j)$ whose removal disconnects the graph into components A_e and $V\setminus A_e$, with $|A_e| \leq |V\setminus A_e|$. For any strategy $S_N \in \mathcal{S}_N$, the number of bits exchanged along edge $e \equiv (i,j)$, denoted by $C_e(f, S_N, N)$, is lower bounded by*

$$C_e(\Pi_{[a,b]}(X_1,\ldots,X_n), S_N, N) \geq N\log_2\{\min(2b-a+3, |A_e|+1)\}.$$

Further there exists a strategy S_N^* such that for any edge e,

$$C_e(\Pi_{[a,b]}(X_1,\ldots,X_n), S_N^*, N) \leq \lceil N \log_2\{\min(2(b+1)+1, 2|A_e|+2)\}\rceil.$$

4.2.1 Extension to non-binary alphabets

The extension to the case where each node draws measurements from a non-binary alphabet is immediate, since we have already solved the two node problem for a general integer alphabet. Consider a tree network with n nodes where node i has a measurement $X_i \in \{0,1,\ldots,l_i-1\}$. Suppose all nodes want to compute a given function which only depends on the value of $X_1+X_2+\ldots X_n$. We can define sum-threshold functions in analogous fashion and derive an optimal strategy for computation, along the lines of Theorem 18. We state the following theorem without proof.

Theorem 20. *Consider a tree network where we want to compute a sum-threshold function, $\Pi_\theta(X_1,\ldots,X_n)$, of non-binary measurements. Let us focus on a single edge e whose removal disconnects the graph into components A_e and $V\setminus A_e$. Let us define $l_{A_e} := \sum_{i\in A_e} l_i$. Then the complexity of computing $\Pi_\theta(X_1,\ldots,X_n)$ is given by*

$$C_e(\Pi_\theta(X_1,\ldots,X_n)) = \log_2\{\min(2\theta+1, 2\min(l_{A_e}, l_{V\setminus A_e})+2, 2(l_V-\theta+1)+1)\}.$$

We can similarly extend Theorem 19 to the case of non-binary alphabets.

4.3 Computing Sum-Threshold Functions in General Graphs

We now consider the computation of sum-threshold functions in general graphs where node measurements can take more than two values, i.e., the alphabet is not restricted to be binary. A cut is defined to be a set of edges $F \subseteq E$ which disconnect the network into two components A_F and $V\setminus A_F$. The lower bound for the two node problem yields a general cut-set bound for any given cut.

Lemma 21 (Cut-set bound)**.** *Consider a general network $G=(V,E)$, where node i has measurement $X_i \in \{0,1,\ldots,l_i-1\}$ and all nodes want to compute the function $\Pi_\theta(X_1,\ldots,X_n)$. Given a*

cut F which separates A_F from $V \setminus A_F$, the cut-set lower bound specifies that: For any strategy S_N, the number of bits exchanged on the edges in F is lower bounded by

$$C_F(\Pi_\theta(X_1, \ldots, X_n), S_N, N) \geq N \log_2(\min\{2\theta + 1, 2\min(l_{A_F}, l_{V \setminus A_F}) + 2, 2(l_V - \theta + 1) + 1)\},$$

where $l_{A_F} = \sum_{i \in A_F} l_i$.

A natural achievable strategy is to pick a spanning subtree of edges and use the optimal strategy on this spanning subtree. Suppose we plot the vector of rates on each edge as a point in $\mathbf{R}^{|E|}$. Then the convex hull of the rate vectors of the subtree aggregation schemes is an achievable region. We would like to study how far this is from the cut-set region.

To simplify matters, let us consider a complete graph G and suppose each node i has a measurement $X_i \in \{0, 1, \ldots, l_i - 1\}$. Define the minimum symmetric achievable rate R_{ach} to be the smallest value of R such that $(R, R, \ldots, R)$ is achievable by aggregating along trees. Similarly, define the minimum symmetric cutset point R_{cut} to be the minimum R such that $(R, R, \ldots, R)$ satisfies the cut-set constraints.

Theorem 22. *For the computation of sum-threshold functions on complete graphs,*

$$R_{ach} \leq 2\left(1 - \frac{1}{n}\right) R_{cut}.$$

In fact, this approximation ratio is tight.

Proof: For simplicity of treatment, let us suppose that $\theta \leq \frac{\sum_{i=1}^n l_i}{2}$. Consider all cuts of the type $(\{i\}, V \setminus \{i\})$. This yields

$$R_{cut} \geq \max_{i \in V} \left(\frac{\min(\log_2(2\theta + 1), \log_2(2l_i + 2))}{n - 1} \right).$$

Now consider the achievable scheme which employs each of the n star graphs for equal sized subblocks of measurements. In a given star graph with i as root, each of the activated edges (i, j) needs $\min(\log_2(2\theta + 1), \log_2(2l_i + 2))$ bits. Further, each edge in the graph G belongs to 2 of the n

star graphs. Hence, the rate on edge (i,j) is given by

$$\frac{1}{n}\left(\min(\log_2(2\theta+1),\log_2(2l_i+2))+\min(\log_2(2\theta+1),\log_2(2l_j+2))\right).$$

Hence we have

$$R_{ach} \;\leq\; \frac{2}{n}(\min(\log_2(2\theta+1),\max_{i\in V}\{\log_2(2l_i+2)\})) \;\leq\; 2\left(1-\frac{1}{n}\right)R_{cut}.$$

Tight Example: Suppose $l_1 = l_2 = \ldots = l_n = l$ and $\theta > l$, then

$$R_{cut} = \frac{1}{n-1}\min(\log_2(2\theta+1),\log_2(2l+2)).$$

Further, from the symmetry of the problem, it is clear that the optimal scheme is to employ the n star graphs for equal sub-blocks of measurements. This gives a symmetric achievable point of

$$R_{ach} = \frac{2}{n}\min(\log_2(2\theta+1),\log_2(2l+2)) = 2\left(1-\frac{1}{n}\right)R_{cut}.$$

4.3.1 LP formulation for tree aggregation

The above approach of restricting attention to aggregation along star graphs leads to a convenient LP formulation. Consider a complete graph G. Let us define the rate region achievable by star graphs in the following way:

$$\tilde{\mathcal{R}}_{ach} = \{A\underline{\lambda} : ||\lambda||_1 = 1\},$$

where A is a $n \times \frac{n(n-1)}{2}$ matrix where the a_{ie}th entry is the minimum number of bits that must be sent along edge e under tree aggregation scheme T_i. Let the vector $\underline{\lambda}$ represent the time fractions assigned to different trees. We want to compare the rate vectors achieved by this scheme with the rate vectors that satisfy the cut constraints. Let $\underline{r} \in \mathcal{R}_{cut}$ be a given rate vector which satisfies the cut constraints of Lemma 1. Now, we seek to find an achievable rate vector that is within a θ factor of $\underline{r}$, and further, we want to find the minimum value of such a θ. This can be formulated as a linear program.

$$\text{Min. } \theta$$

$$\text{s.t. } A\underline{\lambda} \leq \theta \underline{r}$$

$$||\lambda||_1 \geq 1$$

$$\lambda \geq 0, \theta \geq 0$$

Thus we can obtain the optimal assignment λ^* and the optimal factor θ^*. Note that this assignment depends on the given rate vector $\underline{r} \in \mathcal{R}_{cut}$. We can also write similar such LPs for other classes of trees.

4.4 Concluding Remarks

In this chapter, we have addressed the computation of symmetric Boolean functions in tree networks, where all nodes want to compute the function. Toward this objective, we derived lower bounds on the number of bits that must be exchanged on each edge of the tree, using communication complexity theory. Further, for each edge, we have devised an achievable scheme for block computation that involves separation followed by prefix-free coding. We then sequence the transmissions so that information flows up the tree to a root node and then back down to the leaves. For the case of sum-threshold functions, we have provided an optmal in-network information processing strategy.

The approach presented also provides lower and upper bounds for the complexity of other functions like sum-interval functions. Our framework can be generalized to handle functions of integer measurements which only depend on the sum of the measurements. The extension to general graphs is very interesting and appears significantly harder. However, a cut-set lower bound can be immediately derived, and in some special cases we can show that subtree aggregation schemes provide a 2-OPT solution.

CHAPTER 5

COMPUTING SYMMETRIC BOOLEAN FUNCTIONS IN COLLOCATED NETWORKS - WORST CASE

In this chapter, we consider a collocated network where each node's transmissions can be heard by every other node. This could correspond to a collocated subnet in a sensor network. We restrict attention to the problem where *all* nodes want to compute a given function of the sensor measurements. This formulation might be more appropriate in the context of sensor-actuator networks, distributed fault monitoring and adaptive sensing. Each node has a Boolean variable and we focus on the specific problem of symmetric Boolean function computation. As in Chapter 4, we adopt a deterministic formulation of the problem of function computation, requiring zero error. We consider the problem of worst-case block function computation where we allow nodes to accumulate a block of N measurements, and realize greater efficiency by using block codes.

We assume a packet capture model as in [1], where collisions do not convey information. Thus, the problem of medium access is resolved by allowing at most one node to transmit successfully at any time. The set of admissible strategies includes all interactive strategies, where a node may exchange several messages with other nodes, with node i's transmission being allowed to depend on all previous transmissions, and node i's block of measurements. However, since exactly one node can access the wireless medium at a time, the identity of the node which transmits the next bit must be a function of all previous transmissions. This ensures a collision free strategy [1]. It is of particular interest to study the benefit of interactive strategies versus single-round strategies, where each node transmits only one message.

In this chapter, we are particularly interested in computing symmetric Boolean functions. We begin by recalling Theorem 15, which states that, for the AND function of two variables, the exact communication complexity is $\log_2 3$ bits. In Section 5.2, we generalize this result to obtain the *broadcast communication complexity* of the AND function of n variables. In Section 5.3, we consider *threshold functions*, which evaluate to 1 if and only if the total number of 1s are above a

certain threshold. For this class of functions, we devise an achievable strategy which involves each node transmitting in turn using a prefix-free codebook. Further, by intelligent construction of a fooling set, we obtain a matching lower bound for the complexity of computing threshold functions. It is interesting to note that the optimal strategy requires no back-and-forth interaction between nodes.

In Section 5.4, we derive the exact complexity of computing *delta functions* which evaluate to 1 if and only if there are a certain number of 1s. As a natural generalization, in Section 5.5, we study the complexity of computing *interval functions*, which evaluate to 1 if and only if the total number of 1s belong to a given interval $[a, b]$. For a fixed interval $[a, b]$, we propose a strategy for achievability that is order-optimal with optimal preconstant. Additionally, for the interesting class of percentile functions, the proposed single-round strategy is order optimal. Finally, in Section 5.6, we present some extensions to the case of non-Boolean measurements and to the case of non-Boolean functions. Using the intuition gained from the Boolean case, we show how the achievability scheme and fooling set lower bounds can be adapted. In particular we study general threshold functions and the MAX function. While the proposed achievability strategy is exactly optimal for general threshold functions, it is only order-optimal for the MAX function.

5.1 General Problem Setting

Consider a collocated network with nodes 1 through n, where each node i has a Boolean measurement $X_i \in \{0, 1\}$. *Every* node wants to compute the same function $f(X_1, X_2, \ldots, X_n)$ of the measurements. We seek to find communication schemes which achieve correct function computation at each node, with minimum worst-case total number of bits exchanged. We allow for the efficiencies of block computation, where each node i has a block of N independent measurements, denoted by X_i^N. We consider the broadcast scenario, and restrict ourselves to collision-free strategies. This means that for the k^{th} bit b_k, the identity of the transmitting node T_k depends only on previously broadcast bits $b_1, b_2, \ldots, b_{k-1}$, while the value of the bit it sends can depend arbitrarily on all previous broadcast bits as well as its block of measurements $X_{T_k}^N$.

It is important to note that all *interactive* strategies are subsumed within the class of collision-free strategies. A collision-free strategy is said to achieve correct block computation if each node i

can correctly determine the value of the function block $f^N(X_1, X_2, \ldots, X_n)$ using the sequence of bits $b_1, b_2, \ldots$ and its own measurement block X_i^N. Let $\mathcal{S}_N$ be the class of collision-free strategies for block length N which achieve zero-error block computation, and let $C(f, S_N, N)$ be the worst-case total number of bits exchanged under strategy $S_N \in \mathcal{S}_N$. The worst-case per-instance complexity of computing a function $f(X_1, X_2, \ldots, X_n)$ is defined by

$$C(f) = \lim_{N\to\infty} \min_{S_N \in \mathcal{S}_N} \frac{C(f, S_N, N)}{N}.$$

We call this the *broadcast computation complexity* of the function f.

5.2 Complexity of Computing the AND Function

Before we can address the general problem of computing symmetric Boolean functions, we consider the specific problem of computing the AND function, which is 1 if all its arguments are 1, and 0 otherwise. The AND function of two variables was studied in [20] where it was shown that the exact communication complexity is $\log_2 3$ bits, for block computation. We presented an elaborate proof of this result in Theorem 15 in Chapter 4. We now proceed to consider a collocated network with n nodes, where each node wants to compute the AND function, denoted by $\wedge(X_1, X_2, \ldots, X_n)$. We have the following result.

Theorem 23. *For any strategy S_N,*

$$C(\wedge(X_1, X_2, \ldots, X_n), S_N, N) \geq N \log_2(n+1).$$

Further, there exists a strategy S_N^* such that

$$C(\wedge(X_1, X_2, \ldots, X_n), S_N^*, N) \leq \lceil N \log_2(n+1) \rceil + (n-2).$$

Thus, the complexity of the multiple node AND function is given by $C(\wedge(X_1, X_2, \ldots X_n)) = \log_2(n+1)$.

Proof of Achievability: The upper bound is established using induction on the number of

nodes n. From Theorem 15, the result is true for $n = 2$ which is the basis step. Suppose the result is true for a collocated network of $(n-1)$ nodes. Consider an achievable scheme in which node n transmits first, using a prefix free codebook. Let the length of the codeword transmitted be $l(X_n^N)$. After this transmission, the function is determined for the instances where $X_n = 0$. For the instances where $X_n = 1$, the remaining $(n-1)$ nodes need to compute $\wedge(X_1, X_2, \ldots, X_{n-1})$. From the induction hypothesis, we know that this can be done using $\lceil w(X_n^N) \log_2 n \rceil + (n-3)$ bits. Thus the worst-case total number of bits exchanged is $L := \max_{X_n^N}(l(X_n^N) + \lceil w(X_n^N) \log_2 n + (n-3) \rceil)$. As before, we want to minimize this quantity subject to the Kraft inequality. Consider a prefix-free codebook for node n which satisfies

$$l(X_n^N) = \lceil N \log_2(n+1) \rceil + (n-2) - \lceil w(X_n^N) \log_2 n \rceil - (n-3).$$

This satisfies Kraft inequality since

$$\sum_{X_n^N} 2^{\lceil w(X_n^N) \log_2 n \rceil} \leq \sum_{X_n^N} 2^{w(X_n^N) \log_2 n + 1} \leq 2(n+1)^N \leq 2^{\lceil N \log_2(n+1) \rceil + 1}.$$

Proof of lower bound: We need to devise a subset of the set of all $n \times N$ measurement matrices which is a valid fooling set. Consider the subset E of measurement matrices which are only comprised of columns which sum to $(n-1)$ or n. Since there are N columns, there are $(n+1)^N$ such matrices. Let M_1, M_2 be two distinct matrices in this subset. If $f^N(M_1) \neq f^N(M_2)$, then we are done. Suppose not. Then there must exist one instance where the function evaluates to zero and for which M_1 and M_2 have different columns. Let us suppose M_1 has $1_n - e_i$ and M_2 has $1_n - e_j$. Now if we replace the i^th row of M_1 with the i^th row of M_2, the resulting measurement matrix, say M^*, is such that $f(M^*) \neq f(M_1)$. Thus, the set E is a valid fooling set. From the fooling set lower bound, for *any* strategy $S_N \in \mathcal{S}_N$, we must have $C(\wedge(X_1, X_2), S_N, N) \geq N \log_2 3$, implying that $C(f) \geq \log_2 3$. This concludes the proof of Theorem 23. □

By symmetry, we can derive the complexity of the OR function, which is defined to be 0 if all its arguments are 0, and 1 otherwise. Consider a collocated network with n nodes, each of which wants to compute the OR function, denoted by $\vee(X_1, X_2, \ldots, X_n)$.

Corollary 24. *The complexity of the OR function is given by* $C(\vee(X_1, X_2, \ldots, X_n)) = \log_2(n+1)$,

since we can view it as $\overline{\wedge(\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n)}$*, by deMorgan's laws.*

Note: Throughout the rest of this chapter, for ease of exposition, we will ignore the fact that terms like $N \log_2(n+1)$ may not be integer. Since our achievability strategy involves each node transmitting exactly once, this will result in a maximum of one extra bit per node, and since we are amortizing this over a long block length N, it will not affect any of the results.

5.3 Complexity of Computing Boolean Threshold Functions

Definition 7 (Boolean threshold functions)**.** A Boolean threshold function $\Pi_\theta(X_1, X_2, \ldots, X_n)$ is defined as

$$\Pi_\theta(X_1, X_2, \ldots, X_n) = \begin{cases} 1 & \text{if } \sum_i X_i \geq \theta \\ 0 & \text{otherwise.} \end{cases}$$

Theorem 25. *The complexity of computing a Boolean threshold function is* $C(\Pi_\theta(X_1, X_2, \ldots X_n)) = \log_2 \begin{pmatrix} n+1 \\ \theta \end{pmatrix}$.

Proof of Achievability: The upper bound is established by induction on n. For $n = 2$ and for *all* $1 \leq \theta \leq n$, the result follows from Theorem 23 and Corollary 24. This establishes the basis step of the induction. Suppose the upper bound is true for a collocated network of $(n-1)$ nodes, for all $1 \leq \theta \leq (n-1)$. Given a function $\Pi_\theta(X_1, X_2, \ldots, X_n)$ of n variables, consider an achievable strategy in which node n transmits first, using a prefix free codeword of length $l(X_n^N)$. After this transmission, nodes 1 through $n-1$ can decode the block X_n^N. For the instances where $X_n = 0$, these $(n-1)$ nodes now need to compute $\Pi_\theta(X_1, X_2, \ldots, X_{n-1})$. For the instances where $X_n = 1$, the remaining $(n-1)$ nodes need to compute $\Pi_{\theta-1}(X_1, X_2, \ldots, X_{n-1})$. From the induction hypothesis, we have optimal strategies for computing these functions. Let $w^i(X_n^N)$ denote the number of instances of i in the block X_n^N. Under the above strategy, the worst-case total number of bits exchanged is

$$L = \max_{X_n^N} \left(l(X_n^N) + w^0(X_n^N) \log_2 \begin{pmatrix} n \\ \theta \end{pmatrix} + w^1(X_n^N) \log_2 \begin{pmatrix} n \\ \theta - 1 \end{pmatrix} \right).$$

We want to minimize this quantity subject to the Kraft inequality. Consider a prefix-free codebook which satisfies

$$l(X_n^N) = N \log_2 \begin{pmatrix} n+1 \\ \theta \end{pmatrix} - w^0(X_n^N) \log_2 \begin{pmatrix} n \\ \theta \end{pmatrix} - w^1(X_n^N) \log_2 \begin{pmatrix} n \\ \theta - 1 \end{pmatrix}.$$

This assignment of codelengths satisfies the Kraft inequality since

$$\begin{aligned} \sum_{X_n^N} 2^{-l(X_n^N)} &= \begin{pmatrix} n+1 \\ \theta \end{pmatrix}^{-N} \sum_{X_n^N} \begin{pmatrix} n \\ \theta \end{pmatrix}^{w^0(X_n^N)} \begin{pmatrix} n \\ \theta - 1 \end{pmatrix}^{w^1(X_n^N)} \\ &= \begin{pmatrix} n+1 \\ \theta \end{pmatrix}^{-N} \left[\begin{pmatrix} n \\ \theta \end{pmatrix} + \begin{pmatrix} n \\ \theta - 1 \end{pmatrix} \right]^N = 1. \end{aligned}$$

Hence there exists a prefix-free code which satisfies the specified codelengths. The worst case total number of bits exchanged under this strategy is given by $L = N \log_2 \begin{pmatrix} n+1 \\ \theta \end{pmatrix}$. This establishes the induction step and completes the proof of the upper bound.

Proof of lower bound: We need to find a subset of the set of all $n \times N$ measurement matrices which is a valid fooling set. Consider the subset E of measurement matrices which consist of only columns which sum to $(\theta - 1)$ or θ. Since there are N columns, there are $\left[\begin{pmatrix} n \\ \theta \end{pmatrix} + \begin{pmatrix} n \\ \theta - 1 \end{pmatrix} \right]^N$ such matrices. We claim that the set E is a valid fooling set. Let M_1, M_2 be two distinct matrices in this subset. If $f^N(M_1) \neq f^N(M_2)$, then we are done. Suppose not. Then there must exist at least one column at which M_1 and M_2 disagree, say $M_1^{(j)} \neq M_2^{(j)}$. However, both $M_1^{(j)}$ and $M_2^{(j)}$ have the same number of ones. Thus there must exist some row, say i^*, where $M_1^{(j)}$ has a zero, but $M_2^{(j)}$ has a one.

- Suppose $f(M_1^{(j)}) = f(M_2^{(j)}) = 0$. Then, consider the matrix M_1^* obtained by replacing the i^*th row of M_1 with the i^*th row of M_2. The j^{th} column of M_1^* has θ ones, and hence $f(M_1^{*(j)}) = 1$. Hence we have $f(M_1^*) \neq f(M_1)$.

- Suppose $f(M_1^{(j)}) = f(M_2^{(j)}) = 1$. Then, consider the matrix M_2^* obtained by replacing the

i^*th row of M_2 with the i^*th row of M_1. The j^{th} column of M_2^* has $\theta - 1$ ones, and hence $f(M_2^{*(j)}) = 1$. Hence we have $f(M_2^*) \neq f(M_2)$.

Thus, E is a valid fooling set. From the fooling set lower bound, for *any* strategy $S_N \in \mathcal{S}_N$, we must have $C(\Pi_\theta(X_1, X_2, \ldots, X_n), S_N, N) \geq N \log_2 \begin{pmatrix} n+1 \\ \theta \end{pmatrix}$ implying that $C(\Pi_\theta(X_1, X_2, \ldots, X_n)) \geq \log_2 \begin{pmatrix} n+1 \\ \theta \end{pmatrix}$. This concludes the proof of Theorem 25. $\square$

5.4 Complexity of Boolean Delta Functions

Definition 8 (Boolean delta function)**.** A Boolean delta function $\Pi_{\{\theta\}}(X_1, X_2, \ldots, X_n)$ is defined as:

$$\Pi_{\{\theta\}}(X_1, X_2, \ldots, X_n) = \begin{cases} 1 & \text{if } \sum_i X_i = \theta \\ 0 & \text{otherwise.} \end{cases}$$

Theorem 26. *The complexity of computing* $\Pi_{\{\theta\}}(X_1, X_2, \ldots, X_n)$ *is given by*

$$C(\Pi_{\{\theta\}}(X_1, X_2, \ldots, X_n)) = \log_2 \left[\begin{pmatrix} n+1 \\ \theta \end{pmatrix} + \begin{pmatrix} n \\ \theta+1 \end{pmatrix} \right].$$

Sketch of Proof: The proof of achievability follows from an inductive argument as before. The fooling set E consists of measurement matrices composed of only columns which sum up to $\theta - 1$, θ or $\theta + 1$. Thus the size of the fooling set is

$$\left[\begin{pmatrix} n \\ \theta-1 \end{pmatrix} + \begin{pmatrix} n \\ \theta \end{pmatrix} + \begin{pmatrix} n \\ \theta+1 \end{pmatrix} \right]^N . \square$$

5.5 Complexity of Computing Boolean Interval Functions

A Boolean *interval function* $\Pi_{[a,b]}(X_1, \ldots, X_n)$ is defined as:

$$\Pi_{[a,b]}(X_1, X_2, \ldots, X_n) = \begin{cases} 1 & \text{if } a \leq \sum_i X_i \leq b \\ 0 & \text{otherwise.} \end{cases}$$

A naive strategy to compute the function $\Pi_{[a,b]}(X_1, \ldots, X_n)$ is to compute the threshold functions $\Pi_a(X_1, \ldots, X_n)$ and $\Pi_{b+1}(X_1, X_2, \ldots, X_n)$. However, this strategy gives us more information than we seek; i.e., if $\sum_i X_i \in [a, b]^C$, then we also know if $\sum_i X_i < a$, which is superfluous information and perhaps costly to obtain. Alternately, we can derive a strategy which explicitly deals with intervals, as against thresholds. This strategy has significantly lower complexity.

Theorem 27. *The complexity of computing a Boolean interval function* $\Pi_{[a,b]}(X_1, X_2, \ldots, X_n)$ *with* $a + b \leq n$ *is bounded as follows:*

$$\log_2 \left[\begin{pmatrix} n+1 \\ b+1 \end{pmatrix} + \begin{pmatrix} n \\ a-1 \end{pmatrix} \right] \leq C(\Pi_{[a,b]}(X_1, X_2, \ldots X_n)) \leq \log_2 \left[\begin{pmatrix} n+1 \\ b+1 \end{pmatrix} + (b-a+1) \begin{pmatrix} n \\ a-1 \end{pmatrix} \right]. \quad (5.1)$$

The complexity of computing a Boolean interval function $\Pi_{[a,b]}(X_1, \ldots, X_n)$ *with* $a + b \geq n$ *is bounded as follows:*

$$\log_2 \left[\begin{pmatrix} n+1 \\ a \end{pmatrix} + \begin{pmatrix} n \\ b+1 \end{pmatrix} \right] \leq C(\Pi_{[a,b]}(X_1, X_2, \ldots X_n)) \leq \log_2 \left[\begin{pmatrix} n+1 \\ a \end{pmatrix} + (b-a+1) \begin{pmatrix} n \\ b+1 \end{pmatrix} \right]. \quad (5.2)$$

Proof of lower bound: Suppose $a + b \leq n$. Consider the subset E of measurement matrices which consist of only columns which sum to $(a-1)$, b or $(b+1)$. We claim that the set E is a valid fooling set. Let M_1, M_2 be two distinct matrices in this subset. If $f^N(M_1) \neq f^N(M_2)$, we

are done. Suppose not. Then there must exist at least one column at which M_1 and M_2 disagree, say $M_1^{(j)} \neq M_2^{(j)}$.

- Suppose $f(M_1^{(j)}) = f(M_2^{(j)}) = 1$. Then, both $M_1^{(j)}$ and $M_2^{(j)}$ have exactly b 1s. Thus there exists some row, say i^*, where $M_1^{(j)}$ has a 0, but $M_2^{(j)}$ has a 1. Consider the matrix M_1^* obtained by replacing the i^*th row of M_1 with the i^*th row of M_2. The j^{th} column of M_1^* has $(b+1)$ 1s, and hence $f(M_1^{*(j)}) = 0$, which means $f(M_1^*) \neq f(M_1)$.
- Suppose $f(M_1^{(j)}) = f(M_2^{(j)}) = 0$. If both $M_1^{(j)}$ and $M_2^{(j)}$ have the same number of 1s, then the same argument as in (i) applies. However, if $M_1^{(j)}$ has $(a-1)$ 1s and $M_2^{(j)}$ has $(b+1)$ 1s, then there exists some row i^* where $M_1^{(j)}$ has a 0, but $M_2^{(j)}$ has a 1. Then, the matrix M_2^* obtained by replacing the i^*th row of M_2 with the i^*th row of M_1 is such that $f(M_2^*) \neq f(M_2)$.

Thus, the set E is a valid fooling set and $|E| = \left[\binom{n}{b+1} + \binom{n}{a-1} + \binom{n}{b} \right]^N$. This gives us the required lower bound in (5.1).

For the case where $a + b \geq n$, we consider the fooling set E' of matrices which are comprised of only columns which sum to $a-1$, a or $b+1$. This gives us the lower bound in (5.2).

Proof of achievability: Consider the general strategy for achievability where node n transmits a prefix-free codeword of length $l(X_1^N)$, leaving the remaining $(n-1)$ nodes the task of computing a residual function. This approach yields a recursion for computing the complexity of interval functions.

$$C(\Pi_{[a,b]}(X_1, \ldots, X_n)) \leq \log_2[2^{C(\Pi_{[a-1,b-1]}(X_1,\ldots,X_{n-1}))} + 2^{C(\Pi_{[a,b]}(X_1,\ldots,X_{n-1})}].$$

The boundary conditions for this recursion are obtained from the result for Boolean threshold functions in Theorem 25. We could simply solve this recursion computationally, but we want to study the behavior of the complexity as we vary a, b and n. Define $f_{a,b,n} := 2^{C(\Pi_{[a,b]}(X_1,\ldots,X_n))}$. We have the following recursion for $f(a,b,n)$:

$$f(a,b,n) \leq f(a-1,b-1,n-1) + f(a,b,n-1). \tag{5.3}$$

We proceed by induction on n. From Theorems 25 and 26, the upper bounds in (5.1) and (5.2)

are true for $n = 2$ and all intervals $[a, b]$. Suppose the upper bound is true for all intervals $[a, b]$ for $(n-1)$ nodes. Consider the following cases.

(i) Suppose $a + b \leq n - 1$. Substituting the induction hypothesis in (5.3), we get

$$\begin{aligned} f(a,b,n) &\leq \binom{n}{b} + (b-a+1)\binom{n-1}{a-2} \\ &\quad + \binom{n}{b+1} + (b-a+1)\binom{n-1}{a-1} \\ &= \binom{n+1}{b+1} + (b-a+1)\binom{n}{a-1}. \end{aligned}$$

(ii) Suppose $a + b \geq n + 1$. Proof is similar to case (i).

(iii) Suppose $a + b = n$. Substituting the induction hypothesis in (5.3), we get

$$\begin{aligned} f(a,b,n) &\leq \binom{n}{b} + (b-a+1)\binom{n-1}{a-2} \\ &\quad + \binom{n}{a} + (b-a+1)\binom{n-1}{b+1} \\ &\leq \binom{n+1}{b+1} + (b-a+1)\binom{n}{a-1} \\ &= \binom{n+1}{a} + (b-a+1)\binom{n}{b+1}, \end{aligned}$$

where some steps have been omitted in the proof of the last inequality. This establishes the induction step and completes the proof. □

5.5.1 Discussion of Theorem 27

(a) The gap between the lower and upper bounds in (5.1) and (5.2) is *additive*, and is upper bounded by $\log_2(b-a+2)$ which is $\log_2(n+2)$ in the worst case.

(b) For fixed a and b, as the number of nodes increases, we have $a+b \leq n$ for large enough n. Consider the residual term, $(b-a+1)\begin{pmatrix} n \\ a-1 \end{pmatrix}$ on the RHS in (5.1). We have

$$(b-a+1)\begin{pmatrix} n \\ a-1 \end{pmatrix} = o\left(\begin{pmatrix} n+1 \\ b+1 \end{pmatrix}\right).$$

Hence, $C(\Pi_{[a,b]}(X_1,\ldots,X_n)) = \log_2\left(\begin{pmatrix} n+1 \\ b+1 \end{pmatrix}(1+o(1))\right)$. Thus, for any fixed interval $[a,b]$, we have derived an order optimal strategy with optimal preconstant. The orderwise complexity of this strategy is the same as that of the threshold function $\Pi_{b+1}(X_1,\ldots,X_n)$. Similarly, we can derive order optimal strategies for computing $C(\Pi_l[n-a,n-b](X_1,\ldots,X_n))$ and $C(\Pi_{[a,n-b]}(X_1,\ldots,X_n))$, for fixed a and b.

(c) Consider a *percentile* type function where $[a,b] = [\alpha n, \beta n]$, with $(\alpha+\beta) \leq 1$. Using Stirling's approximation, we can still show that

$$(\beta-\alpha)n\begin{pmatrix} n \\ \alpha n-1 \end{pmatrix} = o\left(\begin{pmatrix} n+1 \\ \beta n+1 \end{pmatrix}\right).$$

Thus we have derived an order optimal strategy with optimal preconstant for percentile functions.

(d) Consider the function $f := \Pi_{\cup_i[a_i,b_i]}(X_1,\ldots,X_n)$ where the intervals $[a_i,b_i]$ are disjoint, and may be fixed or percentile type. We can piece together the result for single intervals and show that

$$C(f(X_1,\ldots,X_n)) = \log_2\left(\sum_{i=1}^{m} g(a_i,b_i,n)(1+o(1))\right),$$

$$\text{where } g(a_i, b_i, n) = \begin{cases} \begin{pmatrix} n+1 \\ b_i+1 \end{pmatrix} & \text{if } a_i + b_i \leq n \\ \begin{pmatrix} n+1 \\ a_i \end{pmatrix} & \text{if } a_i + b_i \geq n. \end{cases}$$

5.6 Extension to General Alphabets

In Sections 5.3 - 5.5, we studied optimal strategies for computing threshold functions, delta functions and interval functions of Boolean measurements. In this section, we will show that these results can be generalized to the case where nodes have general integer alphabets, i.e., $X_i \in \{0, 1, \ldots, m_i\}$. The proofs are more tedious in this case, and to maintain clarity of presentation, we will focus on threshold functions and the MAX function.

5.6.1 Complexity of general threshold functions

Consider a collocated network of n nodes, where node i has measurement $X_i \in \{0, 1, \ldots, m_i\}$.

Definition 9. A general threshold function $\Pi_\theta(X_1, X_2, \ldots, X_n)$ is defined as below.

$$\Pi_\theta(X_1, X_2, \ldots, X_n) := \begin{cases} 1 & \text{if } \sum_{i=1}^n X_i \geq \theta \\ 0 & \text{otherwise} \end{cases}.$$

We employ the same notation as for Boolean threshold functions, which constitute a special case of general threshold functions.

Theorem 28. *The complexity of computing $\Pi_\theta(X_1, \ldots, X_n)$ is given by*

$$\begin{aligned} C(\Pi_\theta(X_1, \ldots, X_n)) &= \log_2 \left(\left[Y^\theta \right] + \left[Y^{\theta-1} \right] \left(\prod_{i=1}^n \frac{1 - Y^{m_i+1}}{1 - Y} \right) \right) \\ &= \log_2 \left(\left[Y^\theta \right] + \left[Y^{\theta-1} \right] \left(\prod_{i=1}^n (1 - Y^{m_i+1}) \right) \left(\sum_{k=1}^\infty \begin{pmatrix} n+k-1 \\ n-1 \end{pmatrix} Y^k \right) \right), \end{aligned}$$

where the notation $[Y^\theta]$ refers to the coefficient of Y^θ in the expression on the RHS.

Proof: The proof proceeds by induction on the number of nodes n. From Theorem 16 in Chapter 4, we know that the result is true for $n = 2$ and all choices of m_1, m_2 and θ. This serves as a basis step for the induction. Let us suppose the result is true for a collocated network of $n-1$ nodes and all choices of $m_1, m_2, \ldots, m_{n-1}$ and θ. We now proceed to prove the result for a network of n nodes.

We specify a strategy S_N^* in which node n transmits first. As described in Chapter 4, the optimal strategy consists of two stages, namely separation and coding. We begin by identifying the symbols in $\{0, 1, \ldots, m_n\}$ that need to be *separated* by node n. Let $\tilde{X}_n$ be the mapping of X_n to the reduced alphabet given by $\{a_n, \ldots, b_n\}$. Subsequently, we construct a prefix-free codeword on the reduced alphabet. Let the length of the codeword transmitted be $l(X_1^N)$. At the end of this transmission, the remaining $n-1$ nodes need to compute a residual threshold function for each instance of the block. For example, if $X_n = k$, we are left with the task of computing $\Pi_{\theta-k}(X_1, \ldots, X_{n-1})$. By the induction hypothesis, there is an achievable strategy to compute this residual threshold function, with complexity $C(\Pi_{\theta-k}(X_1, \ldots, X_{n-1}))$. Thus the worst case total number of bits exchanged under this strategy is given by

$$L := \max_{\tilde{X}_n^N} (l(\tilde{X}_n^N) + w^{a_n}(\tilde{X}_n^N) C(\Pi_{\theta-a_n}(X_1, \ldots, X_{n-1})) + w^{a_n+1}(\tilde{X}_n^N) C(\Pi_{\theta-a_n-1}(X_1, \ldots, X_{n-1})) + \ldots + w^{b_n}(\tilde{X}_n^N) C(\Pi_{\theta-b_n}(X_1, \ldots, X_{n-1})),$$

where $w^j(\tilde{X}_n^N)$ is the number of instances in the block where $\tilde{X}_n = j$. Our objective is to find the smallest L that satisfies the Kraft inequality for prefix free codes, which states that $\sum_{\tilde{X}_n^N} 2^{-l(\tilde{X}_n^N)} \leq 1$. From the definition of L, we can lower bound the LHS of the Kraft inequality.

$$\sum_{X_n^N} 2^{-l(\tilde{X}_n^N)} \geq 2^L \sum_{\tilde{X}_n^N} \left(2^{-w^{a_n}(\tilde{X}_n^N) C(\Pi_{\theta-a_n}(X_1,\ldots,X_{n-1}))} \ldots 2^{-w^{b_n}(\tilde{X}_n^N) C(\Pi_{\theta-b_n}(X_1,\ldots,X_{n-1}))} \right).$$

From the induction hypothesis, we have that

$$C(\Pi_{\theta-k}(X_1, \ldots, X_{n-1})) = \log_2 \left(\left[Y^{\theta-k} \right] + \left[Y^{\theta-k-1} \right] \left(\prod_{i=1}^{n-1} \frac{(1 - Y^{m_i+1})}{1-Y} \right) \right).$$

Thus, the smallest feasible value of L is given by

$$
\begin{aligned}
2^L &= \sum_{\tilde{X}_n^N} \left(\left[Y^{\theta - a_n} \right] + \left[Y^{\theta - a_n - 1} \right] \prod_{i=1}^{n-1} \left(\frac{1 - Y^{m_i+1}}{1 - Y} \right) \right)^{w^{a_n}(\tilde{X}_n^N)} \cdot \ldots \\
&\qquad \cdot \ldots \cdot \left(\left[Y^{\theta - b_n} \right] + \left[Y^{\theta - b_n - 1} \right] \prod_{i=1}^{n-1} \left(\frac{1 - Y^{m_i+1}}{1 - Y} \right) \right)^{w^{b_n}(\tilde{X}_n^N)} \\
&= \left(\sum_{k=a_n}^{b_n} \left(\left[Y^{\theta - k} \right] + \left[Y^{\theta - k - 1} \right] \prod_{i=1}^{n-1} \left(\frac{1 - Y^{m_i+1}}{1 - Y} \right) \right) \right)^N \\
&= \left(\sum_{k=0}^{m_n} \left(\left[Y^{\theta - k} \right] + \left[Y^{\theta - k - 1} \right] \prod_{i=1}^{n-1} \left(\frac{1 - Y^{m_i+1}}{1 - Y} \right) \right) \right)^N \qquad (5.4) \\
&= \left(\left[Y^{\theta} \right] + \left[Y^{\theta - 1} \right] (1 + Y + \ldots + Y^{m_n}) \prod_{i=1}^{n-1} \left(\frac{1 - Y^{m_i+1}}{1 - Y} \right) \right)^N \\
&= \left(\left[Y^{\theta} \right] + \left[Y^{\theta - 1} \right] \prod_{i=1}^{n} \left(\frac{1 - Y^{m_i+1}}{1 - Y} \right) \right)^N . \\
L &= N \log_2 \left(\left[Y^{\theta} \right] + \left[Y^{\theta - 1} \right] \prod_{i=1}^{n} \left(\frac{1 - Y^{m_i+1}}{1 - Y} \right) \right), \qquad (5.5)
\end{aligned}
$$

where (5.4) follows from the fact that for $k < a_n$ and $k > b_n$, the coefficients of $Y^{\theta-k}$ and $Y^{\theta-k-1}$ are both zero. Thus, we have derived an upper bound on the complexity of computing general threshold functions in collocated networks.

Proof of lower bound: We need to find a subset of the set of all $n \times N$ measurement matrices which is a valid fooling set. Consider the subset E of measurement matrices which are made up only of the column vectors which sum to $(\theta - 1)$ or θ. Consider two distinct measurement matrices $M_1, M_2 \in E$. Let $f^N(M_1)$ and $f^N(M_2)$ be the block function values obtained from these two matrices. If $f^N(M_1) \neq f^N(M_2)$, we are done. Let us suppose $f^N(M_1) = f^N(M_2)$, and note that since $M_1 \neq M_2$, there must exist one column, say column j, where M_1 and M_2 differ. However, since $f^N(M_1) = f^N(M_2)$, each column of M_1 must sum to the same value as the corresponding column in M_2. Thus there must exist rows i_1 and i_2 such that $M_1(i_1, j) < M_2(i_1, j)$ and $M_1(i_2, j) < M_2(i_2, j)$.

- If column j in M_1 and M_2 sum to $\theta - 1$, then consider the new measurement matrix M^* got by replacing the i_1^{th} row of M_1 with the i_1^{th} row of M_2. The j^{th} column of M^* sums to a value that is greater than $\theta - 1$. Thus, we have $f(M^*) \neq f(M_1)$.

- If column j in M_1 and M_2 sum to θ, then consider the new measurement matrix M^* got by replacing the i_2^{th} row of M_1 with the i_2^{th} row of M_2. The j^{th} column of M^* sums to a value that is less than θ. Thus, we have $f(M^*) \neq f(M_1)$.

Thus, the set E is a valid fooling set. We now need to evaluate the size of E. The number of columns which sum to $\theta - 1$ and θ respectively, can be evaluated by looking at the coefficients at a carefully constructed generating polynomial given by

$$(1 + Y + \ldots + Y^{m_1})(1 + Y + \ldots + Y^{m_2}) \ldots (1 + Y + \ldots + Y^{m_n}).$$

This polynomial models all possible measurement vectors $(X_1, X_2, \ldots, X_n)$. Thus, we can now calculate the size of E by looking at the coefficients of Y^θ and $Y^{\theta-1}$ in this polynomial.

$$\begin{aligned} |E| &= \left[Y^\theta\right] + \left[Y^{\theta-1}\right] \left(\prod_{i=1}^{n} (1 + Y + \ldots + Y^{m_i})\right) && (5.6) \\ &= \left[Y^\theta\right] + \left[Y^{\theta-1}\right] \left(\prod_{i=1}^{n} \frac{1 - Y^{m_i+1}}{1 - Y}\right) && (5.7) \\ &= \left[Y^\theta\right] + \left[Y^{\theta-1}\right] \left(\prod_{i=1}^{n} (1 - Y^{m_i+1})\right) \left(\sum_{k=1}^{\infty} \binom{n+k-1}{n-1} Y^k\right), && (5.8) \end{aligned}$$

where the last equation follows from the binomial expansion for negative exponents. Thus, we have established the required lower bound.□

5.6.2 The MAX function

In this section, we use the tools that we have developed to study a particular example, namely the MAX function. However, we no longer obtain exact results, which is to say that the single-round achievable scheme does not match the fooling set lower bound. This suggests that single round strategies are no longer optimal and it might be necessary to consider multi-round block computation strategies. Indeed, previous work in the area of communication complexity has shown a multi-round protocol that does better that our single-round scheme for the two node case. However, our proposed strategy is still exponentially better than the naive strategy of communicating all measurements. Further, it provides reasonably tight bounds and achieves the optimal scaling

as the number of nodes increases.

Consider nodes 1 through n organized in a collocated network as before. For simplicity, let us suppose that for each node i, $X_i \in \{0, 1, \ldots, m\}$. The MAX function of n measurements is defined in the natural way and is denoted by $MAX_m(X_1, X_2, \ldots, X_n)$. We want to determine the worst case complexity of computing the MAX function.

Theorem 29. *The complexity of the MAX function of n variables from the alphabet $\{0, 1, \ldots, m\}$ is bounded as follows.*

$$\log_2(mn + 1) \leq C(MAX_m(X_1, \ldots, X_n)) \leq \log_2 \begin{pmatrix} n+m \\ m \end{pmatrix}.$$

Proof: We prove the result by induction on the number of nodes n. For the basis step, we consider the two node problem. Consider the general achievable scheme where node 1 sends a prefix free codeword of length $l(X_1^N)$, and node 2 indicates its exact value for each of the instances of the block where $X_1 < X_2$. For example, if $X_1 = k$, node 2 needs to indicate its value in the set $\{k, k+1, \ldots, m\}$. Thus, the worst case total number of bits exchanged under this scheme is given by

$$L = \max_{X_1^N} \left(l(X_1^N) + w^0(X_1^N) \log_2(m+1) + w^1(X_1^N) \log_2 m + \ldots + w^m(X_1^N) \log_2 1 \right).$$

Proceeding as before, we can show that, in order to ensure a valid prefix free code with codelengths $l(X_1^N)$ that satisfy Kraft inequality, the minimum L is given by

$$L = \log_2(m + 1 + m + \ldots + 1) = N \log_2 \begin{pmatrix} m+2 \\ 2 \end{pmatrix}.$$

For the lower bound, we can verify that the set of measurement matrices with columns exclusively from the set $E := \{(0,0), (0,1), (1,0), \ldots, (0,m), (m,0)\}$ is a valid fooling set. Thus we have

$$\log_2(2m + 1) \leq C(MAX_m(X_1, X_2)) \leq \log_2 \begin{pmatrix} m+2 \\ 2 \end{pmatrix},$$

which establishes the basis step for the induction.

Now, let us suppose that the result is true for a network of $(n-1)$ nodes. Consider the following achievable scheme for the n node network. Node n transmits a prefix-free codeword of length $l(X_n^N)$. At the end of this transmission, the remaining $(n-1)$ nodes need to compute the residual MAX function for each instance of the block. For example, if $X_n = k$, we are left with the task of computing the MAX function of $(n-1)$ nodes on the reduced alphabet $\{k, k+1, \ldots, n\}$. Since $\{k, k+1, \ldots, n\}$ is isomorphic to $\{0, 1, \ldots, n-k\}$, this is equivalent to computing $MAX_{n-k}(X_1, \ldots, X_{n-1})$. By the induction hypothesis, there is an achievable strategy to compute this residual MAX function, which we can unroll recursively. Thus the worst case total number of bits exchanged under this strategy is given by

$$L = \max_{X_n^N} \left(l(X_n^N) + w^0(X_n^N) C(MAX_m(X_1, \ldots, X_{n-1})) + \ldots + w^m(X_n^N) C(MAX_0(X_1, \ldots, X_{n-1})) \right).$$

In order to satisfy the Kraft inequality, the smallest L that is feasible is given by

$$\begin{aligned} L &= N \log_2 \sum_{i=0}^{m} 2^{C(MAX_{m-i}(X_1,\ldots,X_{n-1}))} \\ &\leq N \log_2 \sum_{i=0}^{m} \left(\begin{array}{c} m+n-i-1 \\ m-i \end{array} \right) \\ &= N \log_2 \left(\begin{array}{c} m+n \\ m \end{array} \right) \end{aligned}$$

which establishes the upper bound in the induction step.

In order to prove the lower bound, we need to construct a fooling set of the appropriate size. Consider the set of $n \times N$ measurement matrices which consist of columns from the set E defined by

$$E = \left\{ \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \ldots, \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}, \ldots, \begin{bmatrix} m \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ m \\ \vdots \\ 0 \end{bmatrix}, \ldots, \begin{bmatrix} 0 \\ 0 \\ \vdots \\ m \end{bmatrix} \right\}.$$

It is easy to check that this is a valid fooling set of size $(mn+1)^N$ which gives us the required

lower bound for the induction step. □

We make some observations regarding the result in Theorem 29

- For fixed m, we have that $C(MAX_m(X_1, X_2, \ldots, X_n)) = \Theta(\log_2 n)$. This agrees with the result in [1] that the maximum rate of computing a type-threshold function is $\Theta(\frac{1}{\log_2 n})$. Thus, the proposed achievable strategy is order-optimal. Further, we obtain better bounds on the complexity.

$$\log_2(mn+1) \leq C(MAX_m(X_1, X_2, \ldots, X_n)) \leq \log_2 \left(\begin{array}{c} n+m \\ m \end{array} \right) \leq \min(n \log_2(m+1), m \log_2(n+1)).$$

- The naive strategy for computing the MAX function consists of each node communicating its measurement which has a complexity of $n \log_2(m+1)$. For fixed m, the complexity of the proposed scheme is upper bounded by $m \log_2(n+1)$, which is exponentially better than the naive strategy ($O(\log_2 n)$ vs. $O(n)$).

5.7 Concluding Remarks

We have addressed the problem of computing symmetric Boolean functions in a collocated network. We have derived optimal strategies for computing threshold functions and order optimal strategies with optimal preconstant for interval functions. Thus, we have sharply characterized the complexity of various classes of symmetric Boolean functions. Further, since the thresholds and intervals are allowed to depend on n, we have provided a unified treatment of type-sensitive and type-threshold functions.

The results can be extended in two directions. First, we have considered non-Boolean alphabets and general threshold functions which depend only on $\sum_i X_i$. Second, we have considered a particular example of a non-Boolean valued function, namely the MAX function. The fooling set lower bound and the strategy for achievability can be generalized to both these cases.

CHAPTER 6

COMPUTING SYMMETRIC BOOLEAN FUNCTIONS IN COLLOCATED NETWORKS - AVERAGE CASE

In this chapter, we consider the collocated network scenario where all nodes can hear all transmissions. Its symmetry makes it a desirable starting point for studying random planar networks. However, in contrast with Chapter 5 where we consider the worst case scenario, here we impose a joint probability distribution on the node measurements. Since we have random data, the evolution of the computation depends on the particular instances of measurements, and the time of termination of the computation is random. Thus, we seek to minimize the total expected number of bits exchanged to achieve zero error block computation. The linearity of the expectation operator means that we cannot trade off between transmissions as in Chapter 5. However, there are some interesting problems that arise.

We first consider a collocated network with a prespecified order of transmission in Section 6.1, and derive a necessary and sufficient condition for each encoder based on transmissions received. This is similar in spirit to the results in Chapter 3. In Section 6.1.1, we show that the average case complexity of the Boolean threshold functions is $O(\theta)$, where θ is the threshold. Thus, if $\theta = O(1)$, we have that the average case complexity is $\Theta(1)$ in comparison to the worst case complexity of $\Theta(\log n)$ implied by Theorem 25 in Chapter 5.

6.1 General Function Computation in Collocated Networks

Consider a set of nodes $\mathcal{V} := \{v_1, v_2, \ldots, v_n\}$ in a collocated network, with measurement vectors $\underline{X}_1, \underline{X}_2, \ldots, \underline{X}_n$, drawn i.i.d. from a distribution $p(X_1, X_2, \ldots, X_n) > 0$. There is a collector node v_1 which wants to compute the vector function $f^{(N)}(\underline{X}_1, \underline{X}_2, \ldots, \underline{X}_n)$. We consider a restricted class of strategies, with a prespecified order of transmissions, assumed to be $v_n, v_{n-1}, \ldots, v_1$ without loss of generality. Further, the block code $\mathcal{C}_i^N$ for node v_i must be prefix-free, so that all nodes know

when node v_i's transmission is complete. Let R_i^N be the rate of encoder $\mathcal{C}_i^N$. The following result characterizes when error-free function computation is feasible.

Lemma 30. *View codeword C_i^N transmitted by node i as a random variable that depends on $\{C_j^N : (i+1) \leq j \leq n\}$ and the measurement vector $\underline{X}_i$. For a set of encoders which achieves correct computation, the following is a necessary and sufficient condition for encoder $\mathcal{C}_i^N$ given $C_n, C_{n-1}, \ldots, C_{i+1}$. Encoder $\mathcal{C}_i$ separates $\underline{x}_i^1, \underline{x}_i^2 \in \mathcal{X}_i^N$ whenever there exists $\underline{x}^*_{\{1,2,\ldots,i-1\}}$ such that*

$$f^{(N)}(\underline{x}^*_{\{1,\ldots,i-1\}}, \underline{x}_i^1, \underline{x}^{nom}_{\{i+1,\ldots,n\}}) \neq f^{(N)}(\underline{x}^*_{\{1,\ldots,i-1\}}, \underline{x}_i^2, \underline{x}^{nom}_{\{i+1,\ldots,n\}}).$$

Though we have a necessary and sufficient condition, we can trade off between encoders to obtain different rate points. Thus finding the coding scheme for minimum sum rate is not straightforward. The worst case scenario has been studied in [1] establishing a $\Theta(\log n)$ complexity for type-threshold functions in collocated networks and a $\Theta(n)$ complexity for type-sensitive functions.

6.1.1 Average case complexity of computing Boolean threshold functions

In this section, we consider the specific problem of computing Boolean threshold functions in collocated networks. For simplicity, we suppose that nodes' measurements are independent and identically distributed, and propose a natural block computation strategy and quantify its average case complexity.

Theorem 31. *Suppose that the nodes' measurements $X_1, X_2, \ldots, X_n$ are independent and identically distributed with $p(X_i = 1) = p$. Then, the average case complexity of zero error block computation of the threshold function $\Pi_\theta(X_1, X_2, \ldots, X_n)$ is $O(\theta)$ bits.*

Proof: We need to describe a coding strategy which achieves zero error block computation, as block length N goes to infinity. Let us suppose that nodes communicate in reverse order starting with node n. Node n encodes its block of N measurements using a Huffman code which requires $\lceil NH(p) \rceil$ bits. Having heard all previous transmissions, each successive node discards the instances of the block that are already *determined*, i.e., those instances of the block that have already recorded θ ones. It then constructs the Huffman code for the remaining instances of the

block. Let the number of determined instances after node $i+1$ transmits be denoted by random variable Z_i. Then, the average complexity of computing a function block of length N is given by

$$\sum_{i=1}^{n}(N-\mathbf{E}(Z_i))H(p) = \theta NH(p) + NH(p)\sum_{i=\theta}^{n-1}\sum_{j=0}^{\theta-1}\binom{i}{j}p^j(1-p)^{i-j}. \tag{6.1}$$

We need to intelligently upper bound the RHS in the (6.1). We start by establishing the following lemma.

Lemma 32. *Define $g_\theta := \frac{x^\theta}{1-x}$ for θ a positive integer. Then*

$$f_\theta^{(\theta-1)} := \frac{d^{(\theta-1)}g_\theta}{dx^{(\theta-1)}} = (\theta-1)!\left(\frac{1}{(1-x)^\theta}-1\right).$$

Proof of lemma: The proof is by induction on θ. For $\theta = 1$, we have $g_1 = \frac{x}{1-x} = g_1^{(0)}$ trivially. For $\theta > 1$, observe that $g_{\theta-1} - g_\theta = x^{\theta-1}$ and hence $g_\theta^{(\theta-1)} = g_{\theta-1}^{(\theta-1)} - (\theta-1)!$. By the induction assumption, we have

$$g_\theta^{(\theta-1)} = \frac{d}{dx}\left(\frac{(\theta-2)!}{(1-x)^{\theta-1}}\right) - (\theta-1)! = \left(\frac{(\theta-1)!}{(1-x)^\theta}-1\right),$$

which completes the induction. □

We now proceed to show that the second term on the RHS in (6.1) is smaller than $\theta NH(p)\left(\frac{1-p}{p}\right)$ for each n. The proof is by induction on θ. For $\theta = 1$, the second term is given by

$$\sum_{i=1}^{n-1} NH(p)p^\theta(1-p)^i = NH(p)\frac{(1-p)-(1-p)^n}{p} < \frac{NH(p)(1-p)}{p}.$$

Define $R_\theta^n := \sum_{i=\theta}^{n-1}\sum_{j=0}^{\theta-1}\binom{i}{j}p^j(1-p)^{i-j}$. Then, we have the following recursion:

$$R_\theta^n = R_{\theta-1}^n + \sum_{i=\theta}^{n-1}\binom{i}{\theta-1}p^{\theta-1}(1-p)^{i-\theta+1} - \sum_{j=0}^{\theta-2}\binom{\theta-1}{j}p^j(1-p)^{\theta-1-j}.$$

From the induction hypothesis, we have that

$$\begin{aligned} R_\theta^n &\leq \frac{(\theta-1)(1-p)}{p} + \left(\sum_{i=\theta}^{n-1} \binom{i}{\theta-1} p^{\theta-1}(1-p)^{i-\theta+1} \right) - 1 + p^{\theta-1} \\ &\leq \frac{(\theta-1)(1-p)}{p} + \left(\frac{p^{\theta-1}}{(\theta-1)!} \sum_{i=\theta}^{\infty} i(i-1)\ldots(i-\theta+2)(1-p)^{i-\theta+1} \right) - 1 + p^{\theta-1} \\ &= \frac{(\theta-1)(1-p)}{p} + \frac{p^{\theta-1}}{(\theta-1)!} \frac{d^{(\theta-1)}}{dx^{(\theta-1)}} \left(\frac{x^\theta}{1-x} \right) - 1 + p^{\theta-1}. \end{aligned} \tag{6.2}$$

Now, applying Lemma 32 in (6.2), we can show $R_\theta^n \leq \frac{\theta(1-p)}{p}$, which establishes the induction step. Substituting the upper bound for the second term in the RHS of (6.1), we obtain that the total number of bits transmitted is less than $\frac{\theta N H(p)}{p}$ for all n. This yields a sum rate of $\frac{\theta H(p)}{p}$ which completes the proof. □

We make some observations regarding the above result.

(i) For a type-threshold function [1] with threshold vector $[\theta_1, \theta_2]$, we can run two parallel schemes with thresholds $[\theta_1, 0]$ and $[0, \theta_2]$, thus attaining a sum rate $\frac{(\theta_1+\theta_2)H(p)}{p}$. Since we typically consider θ_1, θ_2 to be constants independent of n, we obtain that the average case complexity of computing Boolean threshold functions is $O(1)$.

(ii) As a special case, the average case complexity of computing a symmetric Boolean Disjunctive Normal Form with bounded minterms is $\Theta(1)$.

6.2 Concluding Remarks

In this chapter, we derived necessary and sufficient conditions for function computation in collocated networks with a prespecified order of transmission. We also showed that the average case complexity of computing a Boolean threshold function with threshold θ, is $O(\theta)$ bits. The special case where $\theta = \Theta(1)$, corresponding to Boolean type-threshold functions, was studied in [1]. Thus, we have shown that the average case complexity of computing a type-threshold function is $\Theta(1)$ is in comparison with the worst case complexity of $\Theta(\log n)$.

CHAPTER 7

OPTIMAL ORDERING OF TRANSMISSIONS FOR COMPUTING SYMMETRIC BOOLEAN FUNCTIONS

In this chapter, we consider the collocated network scenario where each node's transmissions can be heard by every other node. The medium access problem is trivially resolved by allowing at most one node to transmit successfully at any time. Each node has a Boolean variable and we focus on the problem of symmetric Boolean function computation with zero error, as in Chapters 5 and 6. We suppose that node measurements are independent and distributed according to given marginal Bernoulli distributions. In this chapter, we primarily focus on optimal strategies for Boolean threshold functions, which are equal to 1 if and only if the number of nodes with measurement 1 is greater than a certain threshold. The set of admissible strategies includes all interactive strategies, where a node may exchange several messages with other nodes.

Let us consider the case where each node has a single bit; the communication problem is rendered trivial, since it is optimal for the transmitting node to simply indicate its bit value. Thus, it only remains to determine the optimal ordering of nodes' transmissions so as to minimize the expected total number of bits exchanged. For the class of Boolean threshold functions, we present a simple policy for ordering the transmissions and prove its optimality. The optimal policy is dynamic, depending in a particularly simple way on the previously transmitted bits, and on the relative ordering of the marginal probabilities, but surprisingly not on their values.

The problem of optimally ordering transmissions of nodes is a sequential decision problem and can in principle be solved by dynamic programming. However, this would require solving the dynamic program for all thresholds and all probability distributions, which is computationally hard. We avoid this, and establish a more insightful solution, in the form of a simple rule defining the optimal policy. The proposed solution solves a more general dynamic program, thus permitting a unified treatment of the problems of single instance computation, block computation and computation under alternate communication models.

In Section 7.2, we turn to the case where each node has a block of bits, and we seek to compute the Boolean threshold function for each instance of the block. Nodes are allowed to employ block coding to achieve greater efficiency, thus adding another layer of complexity. This problem appears formidable due to the plethora of possibilities, and due to a far more complex class of interactive strategies for computation. However, for a certain natural restricted class of strategies, we can establish that an analogous policy is optimal, thus establishing an upper bound on the optimal cost. In order to establish a lower bound across all strategies, we propose the approach of calculating the minimum entropy over all valid protocol partitions which respect fooling set constraints. While this lower bound matches the upper bound for small examples, a proof has remained elusive.

In Section 7.3, we proceed to consider an alternate model of communication, where nodes use pulses of unit energy to convey information. Binary information can be encoded by the presence or absence of a pulse. This model represents a paradigm where silence can be used to convey information. We show the generalizability of our proof technique by deriving a similar result for computing Boolean threshold functions under this model of communication.

When we consider exact computation of functions of random data, we note that the time of termination is a random variable. While the optimal strategy minimizes the expected time of termination, some instances of computation might terminate earlier and some much later. This behavior might be inconvenient for certain applications with deterministic constraints on the time of termination. Thus, it becomes necessary to study the problem of approximate function computation given a fixed number of timeslots. In Section 7.4, we show that the optimal strategy for the approximate computation of threshold functions lacks the same elegant structure as in the case of exact computation. However, for the special case of the parity function, we show that the logical strategy of first querying the node with maximum uncertainty (i.e., entropy) is optimal.

7.1 Single Instance Computation of Boolean Threshold Functions

Consider a collocated network with nodes 1 through n, where each node i has a Boolean measurement $X_i \in \{0, 1\}$. X_i is drawn from a Bernoulli distribution with $P(X_i = 1) =: p_i$, and $\{X_i\}_{i=1}^n$ are independent of each other. Without loss of generality, we assume that $p_1 \leq p_2 \leq \ldots \leq p_n$.

We address the following problem. *Every* node wants to compute the same function $f(X_1, \ldots, X_n)$

of the measurements. Given a strategy for computing $f(X_1, \ldots, X_n)$, the time of termination is a random variable. Our objective is to find communication strategies which achieve correct function computation at each node, with minimum expected total number of bits exchanged. Throughout this chapter, we consider the broadcast scenario where each node's transmission can be heard by every other node. We also suppose that collisions do not convey information, thus restricting ourselves to *collision-free strategies* as in [1]. This means that for the k^{th} bit b_k, the identity of the transmitting node T_k depends only on previously broadcast bits $b_1, b_2, \ldots, b_{k-1}$, while the value of b_k can depend arbitrarily on all previous broadcast bits as well as its own measurement X_{T_k}.

First, we note that since each node has exactly one bit of information, it is optimal to set $b_k = X_{T_k}$. Indeed, for any other choice $b'_k = g(b_1, \ldots, b_{k-1}, X_{T_k})$, the remaining nodes can reconstruct b'_k since they already know $b_i, \ldots, b_{k-1}$. Thus the only freedom available is in choosing the transmitting node T_k as a function of $b_1, b_2, \ldots, b_{k-1}$, for otherwise the transmission itself could be avoided. We call this the *ordering problem*. Thus, by definition, the order can dynamically depend on the previous broadcast bits. In this section, we address the ordering problem for a class of Boolean functions, namely threshold functions.

We begin by establishing some notation. The set of measurements of nodes 1 through n is denoted by $(X_1, X_2, \ldots, X_n)$ which is abbreviated as $\mathbf{X}^n$. In the sequel, we will use $\mathbf{X}^n_{-i}$ to denote the set of measurements $(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)$. As a natural extension, we use $\mathbf{X}^n_{-(i,j)}$ to denote the set of measurements $(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_{j-1}, X_{j+1}, \ldots, X_n)$, where $i < j$.

We recall that a Boolean threshold function $\Pi_\theta(X_1, X_2, \ldots, X_n)$ is defined by

$$\Pi_\theta(X_1, X_2, \ldots, X_n) = \begin{cases} 1 & \text{if } \sum_i X_i \geq \theta, \\ 0 & \text{otherwise.} \end{cases}$$

The class of threshold functions has the property that, if one of the nodes' measurements is known, the residual function is still a threshold function. Given a function $\Pi_{n-k}(\mathbf{X}^n)$, if node i transmits its bit, we are left with the residual task of computing $\Pi_{n-k-1}(\mathbf{X}^n_{-i})$ if $X_i = 1$, and $\Pi_{n-k}(\mathbf{X}^n_{-i})$ if $X_i = 0$. Thus, the ordering problem can be solved using dynamic programming. Let $C(\Pi_{n-k}(\mathbf{X}^n))$ denote the minimum expected number of bits required to compute $\Pi_{n-k}(\mathbf{X}^n)$. The dynamic programming

equation is

$$C(\Pi_{n-k}(\mathbf{X}^n)) = \min_i \{1 + p_i C(\Pi_{n-k-1}(\mathbf{X}^n_{-i})) + (1-p_i) C(\Pi_{n-k}(\mathbf{X}^n_{-i}))\} \tag{7.1}$$

with boundary condition $C(\Pi_a(\mathbf{X}^m)) = 0$ if $a = 0$ or $a > m$.

To begin with, we argue that solving (7.1) for each n and k does indeed yield the optimal strategy for computing Boolean threshold functions. In particular, to derive the optimal strategy for computing $\Pi_{n-k}(\mathbf{X}^n)$, we first determine which node must transmit first, by solving (7.1) for n, k. Then, depending on whether $X_{T(1)} = 0$ or $X_{T(1)} = 1$, we are left with the residual task of computing $\Pi_{n-k}(\mathbf{X}^n_{-T(1)})$ or $\Pi_{n-k-1}(\mathbf{X}^n_{-T(1)})$. We can determine which node should transmit next in either case, from the solution of (7.1) for $n-1, k-1$ or $n-1, k$ respectively. Proceeding recursively, one can unroll the optimal strategy for computing $\Pi_{n-k}(X_1, X_2, \ldots X_n)$.

In (7.1), we recognize that the single-stage cost is uniformly 1. More generally, given a function $f(\cdot) : [0,1] \to \mathbf{R}^+$, one can write down a more general dynamic programming equation.

$$C(\Pi_{n-k}(\mathbf{X}^n)) = \min_i \{f(p_i) + p_i C(\Pi_{n-k-1}(\mathbf{X}^n_{-i})) + (1-p_i) C(\Pi_{n-k}(\mathbf{X}^n_{-i}))\}. \tag{7.2}$$

Here, one can view $f(p_i)$ as the cost of communicating the information of node i which has $P(X_i = 1) = p_i$. Indeed, for the case of single instance computation, we have $f(p) \equiv 1$. In the sequel, we will see how this general dynamic programming formulation will allow us to study other problems of interest.

For general $f(\cdot)$, solving the dynamic programing equation (7.2) may be intractable. Further, it is unclear at the outset if the optimal strategy will depend only on the ordering of the p_is, or their particular values. This makes the explicit solution of (7.2), or even (7.1), for all n, k and $(p_1, p_2, \ldots p_n)$ notoriously hard. However, under some conditions on $f(\cdot)$, we can derive a very simple characterization of the optimal strategy for each n and $0 \le k \le n-1$. Further, we observe that optimal strategy is independent of the particular values of the p_is, but only depends on their relative ordering.

Lemma 33. *Let $f(\cdot) : [0,1] \to \mathbf{R}^+$ be a function such that*

- $f(p) = f(1-p)$.

- $\frac{f(p)}{p}$ *is a monotone non-increasing function of* p.

Then the minimum in (7.2) is attained by $k+1$. *That is,*

$$k+1 \in \underset{i}{\operatorname{argmin}} \left\{ f(p_i) + p_i C(\Pi_{n-k-1}(\mathbf{X}^n_{-i})) + (1-p_i) C(\Pi_{n-k}(\mathbf{X}^n_{-i}) \right\}. \tag{7.3}$$

This result is true for all n *and all* $0 \leq k \leq n-1$ *and all probability distributions with* $p_1 \leq p_2 \leq \ldots \leq p_n$.

Proof: We define the following expressions.

$$T_{m,k,i}(\mathbf{X}^m) = p_{k+1} C(\Pi_{m-k-1}(\mathbf{X}^m_{-(k+1)}) + (1-p_{k+1}) C(\Pi_{m-k}(\mathbf{X}^m_{-(k+1)}) \\ - p_i C(\Pi_{m-k-1}(\mathbf{X}^m_{-i}) - (1-p_i) C(\Pi_{m-k}(\mathbf{X}^m_{-i})$$

$$S^{(1)}_{m,k,i}(\mathbf{X}^m) := (p_{k+1}-p_i) C(\Pi_{m-k-1}(\mathbf{X}^m_{-(k+1,i)})) + (1-p_{k+1}) C(\Pi_{m-k}(\mathbf{X}^m_{-(k+1)})) \\ - (1-p_i) C(\Pi_{m-k}(\mathbf{X}^m_{-i})).$$

$$S^{(2)}_{m,k,i}(\mathbf{X}^m) := (p_i - p_{k+1}) C(\Pi_{m-k-1}(\mathbf{X}^m_{-(i,k+1)})) + p_{k+1} C(\Pi_{m-k-1}(\mathbf{X}^m_{-(k+1)})) - p_i C(\Pi_{m-k-1}(\mathbf{X}^m_{-i})).$$

We establish the above theorem by induction on the number of nodes n. However, we need to load the induction hypothesis. Consider the following induction hypothesis.

(a) $T_{m,k,i}(\mathbf{X}^m) \leq f(p_i) - f(p_{k+1})$ for all $0 \leq k \leq (m-1), 1 \leq i \leq m$

(b) $S^{(1)}_{m,k,i}(\mathbf{X}^m) \leq (1-p_{k+1}) f(p_i) - (1-p_i) f(p_{k+1})$ for all $0 \leq k+1 \leq (m-1), k+2 \leq i \leq m$

(c) $S^{(2)}_{m,k,i}(\mathbf{X}^m) \leq p_{k+1} f(p_i) - p_i f(p_{k+1})$ for all $0 \leq k \leq (m-1), 1 \leq i < k+1$

Observe that part (a) immediately establishes (7.3).

The basis step for $m = 1$ is trivially true. Let us suppose the induction hypothesis is true for all $m \leq n$. We now proceed to prove the hypothesis for $m = n+1$.

Lemma 34. *For fixed* k *and* $i \geq k+2$, *we have* $S^{(1)}_{n+1,k,i}(\mathbf{X}^{n+1}) \leq (1-p_{k+1}) f(p_i) - (1-p_i) f(p_{k+1})$.

Proof: See Section 7.1.1.

Lemma 35. *For fixed k and $i \leq k$, we have $S^{(2)}_{n+1,k,i}(\mathbf{X}^{n+1}) \leq p_{k+1}f(p_i) - p_i f(p_{k+1})$.*

Proof: See Section 7.1.1.

Lemmas 34 and 35 establish the induction step for parts (b) and (c) of the induction hypothesis. We now proceed to show the induction step for part (a).

Lemma 36. *For fixed k and $i \geq k+2$, we have $T_{n+1,k,i}(\mathbf{X}^{n+1}) \leq S^{(1)}_{n+1,k,i}(\mathbf{X}^{n+1}) + p_{k+1}f(p_i) - p_i f(p_{k+1})$.*

Proof: See Section 7.1.1.

Lemma 37. *For fixed k and $i \leq k$, we have $T_{n+1,k,i}(\mathbf{X}^{n+1}) \leq S^{(2)}_{n+1,k,i}(\mathbf{X}^{n+1}) + (1-p_{k+1})f(p_i) - (1-p_i)f(p_{k+1})$.*

Proof: See Section 7.1.1.

Applying Lemmas 36 and 37 together with Lemmas 34 and 35, we see that $T_{n+1,k,i}(\mathbf{X}^{n+1}) \leq 0$ for all $0 \leq k \leq n$ and $i \neq k+1$. For the case $i = k+1$, we have $T(n+1,k,k+1) = 0$ trivially. This completes the induction step for part (a), and the proof of the Theorem. $\square$

Using Lemma 33, we can now simply derive the optimal sequential communication strategy for computing a single instance of the Boolean threshold function $\Pi_{n-k}(\mathbf{X}^n)$.

Theorem 38. *In order to compute a single instance of the Boolean threshold function $\Pi_{n-k}(\mathbf{X}^n)$, it is optimal for node $(k+1)$ to transmit its bit first.*

Proof: In the case of single instance computation, we have $f(p) \equiv 1$. Hence, trivially, we have that $f(p) = f(1-p)$, and that $\frac{f(p)}{p}$ is a monotone non-increasing function of p. From Lemma 33, we have

$$k+1 \in \operatorname*{argmin}_i \left\{ f(p_i) + p_i C(\Pi_{n-k-1}(\mathbf{X}^n_{-i})) + (1-p_i)C(\Pi_{n-k}(\mathbf{X}^n_{-i}) \right\}.$$

Thus, in order to compute the Boolean threshold function $\Pi_{n-k}(\mathbf{X}^n)$, it is optimal for node $k+1$ to transmit first. $\square$

Note: At the outset, there are two heuristics that one may apply to the ordering problem. First, if we believe that $\Pi_{n-k}(\mathbf{X}^n)$ evaluates to 0, the conditional optimal strategy is for nodes to transmit in order starting with node 1. Alternately, if we believe that $\Pi_{n-k}(\mathbf{X}^n)$ evaluates to

1, the conditional optimal strategy is for nodes to transmit in reverse order starting with node n. Thus, the result in Theorem 38 can be viewed as an appropriate *hedging* solution which safeguards against the event that $\Pi_{n-k}(\mathbf{X}^n)$ could evaluate to 0 or 1. It is indeed surprising that a particularly simple hedging strategy is optimal for all n, all k and all probability distributions.

7.1.1 Proofs of Lemma 33

Proof of Lemma 34

First, let us suppose $k = 0$. In this case

$$S^{(1)}_{n+1,0,i}(\mathbf{X}^{n+1}) = (p_1 - p_i)C(\Pi_n(\mathbf{X}^{n+1}_{-(1,i)})) + (1-p_1)C(\Pi_{n+1}(\mathbf{X}^{n+1}_{-1})) - (1-p_i)C(\Pi_{n+1}(\mathbf{X}^{n+1}_{-i})) = 0$$

However, by assumption, we have $0 \leq (1-p_1)f(p_i) - (1-p_i)f(p_1)$.

Next, consider the case where $k \neq 0$.

$$\begin{aligned}
& (p_{k+1} - p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,i)})) + (1-p_{k+1})C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k+1)})) - (1-p_i)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-i})) \\
= \;& (p_{k+1} - p_i)\left[f(p_k) + p_k C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k,k+1,i)})) + (1-p_k)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1,i)}))\right] \\
& + (1-p_{k+1})\left[f(p_k) + p_k C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})) + (1-p_k)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k,k+1)}))\right] \\
& - (1-p_i)\left[f(p_k) + p_k C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,i)})) + (1-p_k)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k,i)}))\right] \qquad (7.4) \\
= \;& p_k\Big[(p_{k+1} - p_i)C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k,k+1,i)})) \\
& \qquad + (1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})) - (1-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,i)}))\Big] \\
& + (1-p_k)\Big[(p_{k+1} - p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1,i)})) \qquad (7.5) \\
& \qquad + (1-p_{k+1})C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k,k+1)})) - (1-p_i)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k,i)}))\Big] \\
\leq \;& p_k\Big[(p_{k+1} - p_i)C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k,k+1,i)})) \\
& \quad + (1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})) - (1-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,i)}))\Big] + (1-p_k)S^{(1)}_{n,k-1,i-1}(\mathbf{X}^{n+1}_{-k}) \\
\leq \;& p_k\Big[(p_{k+1} - p_i)C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k,k+1,i)})) \\
& \qquad + (1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})) - (1-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,i)}))\Big] \\
& + (1-p_k)\left[(1-p_{k+1})f(p_i) - (1-p_i)f(p_{k+1})\right] \qquad (7.6)
\end{aligned}$$

$$
\begin{aligned}
&= p_k \Big[(p_{k+1}-p_i)C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k,k+1,i)})) + (1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})) \\
&\qquad -(1-p_i)[f(p_{k+1}+p_{k+1}C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k,k+1,i)})) + (1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1,i)}))]\Big] \\
&\quad +(1-p_k)\left[(1-p_{k+1})f(p_i) - (1-p_i)f(p_{k+1})\right] \qquad (7.7) \\
&= p_k(1-p_{k+1})\Big[C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})) - p_i C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k,k+1,i)})) - (1-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1,i)}))\Big] \\
&\quad -p_k(1-p_i)f(p_{k+1}) + (1-p_k)\left[(1-p_{k+1})f(p_i) - (1-p_i)f(p_{k+1})\right] \\
&\leq p_k(1-p_{k+1})f(p_i) - (1-p_i)f(p_{k+1}) + (1-p_k)(1-p_{k+1})f(p_i) \qquad (7.8) \\
&= (1-p_{k+1})f(p_i) - (1-p_i)f(p_{k+1})
\end{aligned}
$$

Equation (7.4) follows from the optimal ordering for computing $\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,i)})$, $\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k+1)})$ and $\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-i})$, which is true by the induction hypothesis for $m = n$. The inequality (7.6) follows from the induction hypothesis that $S^{(1)}_{n,k-1,i}(\mathbf{X}^{n+1}_{-k}) \leq (1-p_{k+1})f(p_i) - (1-p_i)f(p_{k+1})$. Equality in (7.7) and (7.8) follows from the optimal ordering for computing $\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,i)})$ and $\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})$ respectively. □

Proof of Lemma 35

First, let us suppose $k = n$. In this case

$$
S^{(2)}_{n+1,n,i}(\mathbf{X}^{n+1}) = (p_i - p_{n+1})C(\Pi_0(\mathbf{X}^{n+1}_{-(i,n+1)})) + p_{n+1}C(\Pi_0(\mathbf{X}^{n+1}_{-(n+1)})) - p_i C(\Pi_0(\mathbf{X}^{n+1}_{-i})) = 0.
$$

However, by assumption, we have $0 \leq p_{n+1}f(p_i) - p_i f(p_{n+1})$.

Next, consider the case where $k < n$.

$$
\begin{aligned}
&(p_i - p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1)})) + p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1)})) - p_i C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-i})) \\
&= (p_i - p_{k+1})\Big[f(p_{k+2}) + p_{k+2}C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)})) + (1-p_{k+2})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)}))\Big] \\
&\quad +p_{k+1}\Big[f(p_{k+2}) + p_{k+2}C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) + (1-p_{k+2})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)}))\Big] \\
&\quad -p_i\Big[f(p_{k+2}) + p_{k+2}C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(i,k+2)})) + (1-p_{k+2})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+2)}))\Big] \qquad (7.9)
\end{aligned}
$$

$$
\begin{aligned}
&= p_{k+2}\Big[(p_i - p_{k+1})C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)})) \\
&\qquad\qquad + p_{k+1}C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) - p_i C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(i,k+2)}))\Big] \\
&\quad + (1-p_{k+2})\Big[(p_i - p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)})) + p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) \\
&\qquad\qquad + p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) - p_i C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+2)}))\Big] \\
&\leq (1-p_{k+2})\Big[(p_i - p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)})) \\
&\qquad + p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) - p_i C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+2)}))\Big] + p_{k+2}\Big[S^{(2)}_{n,k,i}(\mathbf{X}^{n+1}_{-(k+2)})\Big] \\
&\leq (1-p_{k+2})\Big[(p_i - p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)})) \\
&\qquad\qquad + p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) - p_i C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+2)}))\Big] \\
&\quad + p_{k+2}\left[p_{k+1}f(p_i) - p_i f(p_{k+1})\right] \qquad (7.10) \\
&= (1-p_{k+2})\Big[(p_i - p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)})) + p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) \\
&\qquad - p_i[f(p_{k+1}) + p_{k+1}C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)})) + (1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)}))]\Big] \\
&\quad + p_{k+2}\left[p_{k+1}f(p_i) - p_i f(p_{k+1})\right] \qquad (7.11) \\
&= (1-p_{k+2})p_{k+1}\Big[C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) - p_i C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)})) \\
&\qquad\qquad - (1-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1,k+2)}))\Big] \\
&\quad - (1-p_{k+2})p_i f(p_{k+1}) + p_{k+2}\left[p_{k+1}f(p_i) - p_i f(p_{k+1})\right] \\
&\leq (1-p_{k+2})p_{k+1}f(p_i) - p_i f(p_{k+1}) + p_{k+2}p_{k+1}f(p_i) \qquad (7.12) \\
&= p_{k+1}f(p_i) - p_i f(p_{k+1})
\end{aligned}
$$

Equation (7.9) follows from the optimal ordering for computing $\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1)})$, $\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1)})$ and $\Pi_{n-k}(\mathbf{X}^{n+1}_{-i})$, which follows from the induction hypothesis for $m = n$. The inequality (7.10) follows from the induction hypothesis that $S^{(2)}_{n,k,i}(\mathbf{X}^{n+1}_{-(k+2)}) \leq p_{k+1}f(p_i) - p_i f(p_{k+1})$. Equations (7.11) and (7.12) follow from the optimal ordering for computing $\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+2)})$ and $\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)})$ respectively. □

Proof of Lemma 36

First, we observe that

$$T_{n+1,k,i}(\mathbf{X}^{n+1}) - S^{(1)}_{n+1,k,i}(\mathbf{X}^{n+1}) = p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1)})) - p_iC(\Pi_{n-k}(\mathbf{X}^{n+1}_{-i})) \\ - (p_{k+1}-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,i)})).$$

Thus it is enough to show that

$$p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1)})) - p_iC(\Pi_{n-k}(\mathbf{X}^{n+1}_{-i})) \\ \leq (p_{k+1}-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,i)})) + p_{k+1}f(p_i) - p_if(p_{k+1}) \qquad \text{for } i \geq k+2.$$

First, observe that for $k = n$, the statement is vacuously true since $i \geq n+2$ is impossible. Hence, let us suppose that $k < n$. We have

$$\begin{aligned}
& p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1)})) - p_iC(\Pi_{n-k}(\mathbf{X}^{n+1}_{-i})) \\
= \; & p_{k+1}\left[f(p_{k+2}) + p_{k+2}C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) + (1-p_{k+2})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)}))\right] \\
& -p_i\left[f(p_{k+1} + p_{k+1}C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k+1,i)})) + (1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,i)}))\right] \qquad (7.13) \\
= \; & p_{k+1}\left[f(p_{k+2}) + p_{k+2}C(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) - p_iC(\Pi_{n-k-1}(\mathbf{X}^{n+1}_{-(k+1,i)}))\right] \\
& +p_{k+1}(1-p_{k+2})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) - p_i(1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,i)})) - p_if(p_{k+1}) \\
\leq \; & p_{k+1}\left[f(p_i) + (1-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,i)})) - (1-p_{k+2})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)}))\right] \\
& +p_{k+1}(1-p_{k+2})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,k+2)})) - p_i(1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,i)})) - p_if(p_{k+1}) \qquad (7.14) \\
= \; & (p_{k+1}-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1,i)})) + p_{k+1}f(p_i) - p_if(p_{k+1})
\end{aligned}$$

Equation 7.13 follows from the optimal order for computing $\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k+1)})$ and $\Pi_{n-k}(\mathbf{X}^{n+1}_{-i})$. The inequality in 7.14 follows from the induction hypothesis $T_{n,k,i}(\mathbf{X}^{n+1}_{-(k+1)}) \leq f(p_i) - f(p_{k+2})$. $\square$

Proof of Lemma 37

First, we observe that

$$T_{n+1,k,i}(\mathbf{X}^{n+1}) - S^{(2)}_{n+1,k,i}(\mathbf{X}^{n+1}) = (1-p_{k+1})C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k+1)})) - (1-p_i)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-i})) - (p_i - p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1)})).$$

Thus it is enough to show that

$$(1-p_{k+1})C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k+1)})) - (1-p_i)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-i})) \leq (p_i - p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1)})) + (1-p_{k+1})f(p_i) - (1-p_i)f(p_{k+1}) \qquad \text{for } i \leq k.$$

First, observe that for $k = 0$, the statement is vacuously true since $i \leq 0$ is impossible. Hence, let us suppose that $k > 0$. We have

$$\begin{aligned}
& (1-p_{k+1})C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k+1)})) - (1-p_i)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-i})) \\
= \; & (1-p_{k+1})\left[f(p_k) + p_k C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})) + (1-p_k)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k,k+1)}))\right] \\
& -(1-p_i)\left[f(p_{k+1}) + p_{k+1}C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1)})) + (1-p_{k+1})C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(i,k+1)}))\right] \qquad (7.15) \\
= \; & (1-p_{k+1})\left[f(p_k) + (1-p_k)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k,k+1)})) - (1-p_i)C(\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(i,k+1)}))\right] \\
& +p_k(1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})) - p_{k+1}(1-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1)})) - (1-p_i)f(p_{k+1}) \\
\leq \; & (1-p_{k+1})\left[f(p_i) + p_i C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1)})) - p_k C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)}))\right] \\
& +p_k(1-p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(k,k+1)})) - p_{k+1}(1-p_i)C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1)})) \qquad (7.16) \\
= \; & (p_i - p_{k+1})C(\Pi_{n-k}(\mathbf{X}^{n+1}_{-(i,k+1)})) + (1-p_{k+1})f(p_i) - (1-p_i)f(p_{k+1})
\end{aligned}$$

Equation (7.15) follows from the optimal order for computing $\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-(k+1)})$ and $\Pi_{n-k+1}(\mathbf{X}^{n+1}_{-i})$. The inequality in (7.16) follows from the induction hypothesis $T_{n,k-1,i}(\mathbf{X}^{n+1}_{-(k+1)}) \leq f(p_i) - f(p_k)$ $\square$.

7.2 Block Computation of Boolean Threshold Functions

We now shift attention to the case where we allow nodes to accumulate a block of N measurements, and thus achieve improved efficiency by using block codes. The most general class of interactive strategies are those where the identity of the node transmitting the k^{th} bit, say T_k, can depend arbitrarily on all previously broadcast bits, and the k^{th} bit itself can depend arbitrarily on all previously broadcast bits as well as T_k's block of measurements. We require that all nodes compute the function with zero error for the block, and wish to minimize the expected number of bits exchanged per instance of computation, denoted $\mathcal{C}(\Pi_{n-k}(\mathbf{X}^n))$. While the problem of finding the optimal strategy in this general class of strategies appears intractable, we derive the optimal solution under a restricted class of strategies. The restriction we impose is natural, and we conjecture that the optimal strategy in this restricted class is also optimal among all interactive strategies.

Theorem 39. *Consider the following restricted class of strategies. When we compute $\Pi_{n-k}(\mathbf{X}^n)$ for a block of N measurements, we mandate that the first node to transmit, say node $T(1)$, must declare its entire block using a Huffman code. Note that this does not exclude interactive strategies, since, subsequent to node $T(1)$'s transmission, we have two subproblems over sub-blocks of measurements corresponding to instances where $X_{T(1)} = 0$ and $X_{T(1)} = 1$. For each of these subproblems, we could potentially have different nodes transmitting first. However each of these nodes are again constrained to communicate their entire subblock of measurements. Under this restricted class of strategies, in order to compute $\Pi_{n-k}(\mathbf{X}^n)$ for a block of measurements, it is optimal for node $k+1$ to transmit its entire block first, using the Huffman code. This result is true for asymptotically long block lengths, for all n, all $0 \leq k \leq n-1$, and all probability distributions with $p_1 \leq p_2 \leq \ldots \leq p_n$.*

Proof: Let us suppose node i transmits first. Under this restricted class of strategies, node i must communicate its entire block which requires an average description length of $H(p_i)$ bits per instance. This can be achieved asymptotically by using the Huffman code to compress node i's block of measurements. Subsequent to node i's transmission, we are left with the residual tasks of computing $\Pi_{n-k-1}(\mathbf{X}^n_{-i})$ for the subblock where $X_i = 1$, and $\Pi_{n-k}(\mathbf{X}^n_{-i})$ for the subblock where $X_i = 0$. These are two block computation problems again. Let $\mathcal{C}_U(\Pi_{n-k}(\mathbf{X}^n)$ denote the minimum number of bits per instance, that must be exchanged under this restricted class of strategies. We

can write a dynamic programming equation as before.

$$\mathcal{C}_U(\Pi_{n-k}(\mathbf{X}^n)) = \min_i \{H(p_i) + p_i \mathcal{C}_U(\Pi_{n-k-1}(\mathbf{X}^n_{-i})) + (1-p_i)\mathcal{C}_U(\Pi_{n-k}(\mathbf{X}^n_{-i}))\} \tag{7.17}$$

where $H(p)$ is the standard binary entropy function defined by $H(p) = -p\log_2(p) - (1-p)\log_2(1-p)$. The boundary condition for (7.2) is given by $\mathcal{C}_U(\Pi_a(\mathbf{X}^m)) = 0$ if $a = 0$ or $a > m$.

Observe that (7.17) is a special case of (7.2) where $f(p) = H(p)$. Thus, for this restricted class of strategies, the problem of optimal computation once again reduces to an ordering problem. If we can show that $H(p)$ satisfies the conditions in Lemma 33, the result follows immediately. Clearly $H(p) = H(1-p)$ and one can verify that

$$\frac{d\left(\frac{H(p)}{p}\right)}{dp} = \frac{\log_2(1-p)}{p^2} \leq 0.$$

Thus, we have that $\frac{H(p)}{p}$ is a non-decreasing function of p. Hence, from Lemma 33, the optimal strategy for computing $\Pi_{n-k}(\mathbf{X}^n)$ for a block of measurements is for node $k+1$ to transmit its entire block first, using the Huffman code. $\square$

Note 1: The proposed optimal strategy is inherently interactive, since nodes do transmit more than once. This is due to the recursive division of the original block of measurements depending on nodes' transmissions. In practice, all nodes need to agree *a priori* on a traversal order in the computation tree, e.g., depth-first traversal or breadth-first traversal.

Note 2: The proposed optimal strategy is asymptotically optimal in the limit of long blocks. This is necessary to achieve an average description length of $H(p_i)$ bits per instance. In practice, one could simply choose a large enough block length N so that the average description length is close enough to the entropy. In this context, it is important to note that, as the computation proceeds, the original block gets recursively subdivided into smaller and smaller subblocks of measurements. Each of these subblocks need to be large enough to achieve an average description length that is close enough to the entropy of the transmitting node. Thus, in the worst case, we could have up to 2^n subblocks in the computation tree, and one must take care to ensure that each of these subblocks are large enough by choosing N to be suitably large.

7.2.1 A strategy-independent lower bound

Next, we would like to determine if the restricted class of strategies considered above is rich enough to include the absolute optimal strategy for interactive block computation without any restrictions on a node encoding all its information using a Huffman code. Intuitively, since all the instances of the block are independent and identically distributed, it appears suboptimal for nodes to communicate only partial information regarding their blocks at any stage. Thus, it is plausible that, under the optimal strategy, one node communicates its entire block, and the computation proceeds recursively from there. However, establishing this fact rigorously is a formidable challenge. In this subsection, we describe a possible approach toward establishing this result, by adapting the concept of *fooling sets*. Fooling sets are a classical tool for establishing lower bounds in communication complexity [8], and have been used to establish tight lower bounds on the minimum number of bits exchanged in the worst-case in collocated networks (Chapter 5), and tree networks (Chapter 4). We describe an extension of fooling sets to the scenario where we have probability distributions on nodes' measurements, and use this to establish a lower bound. This is done as follows.

We recall from Theorem 25 in Chapter 5 that, for the threshold function $\Pi_{n-k}(\mathbf{X}^n)$, a valid fooling set of maximum size is given by

$$E_{n,n-k} := \{\mathbf{X}^n : \sum_{i=1}^{n} X_i = n-k \text{ or } \sum_{i=1}^{n} X_i = n-k-1\}.$$

Any correct protocol for distributed computation of $\Pi_{n-k}(\mathbf{X}^n)$ partitions the function matrix into monochromatic rectangles [8]. Further, each rectangle in the partition can contain at most one element of $E_{n,n-k}$. Let $D(\Pi_{n-k}(\mathbf{X}^n))$ be the set of all protocol partitions of the function matrix of $\Pi_{n-k}(\mathbf{X}^n)$ which respect the fooling set constraints. Suppose we use a protocol with associated partition d; the number of bits that must be exchanged under this protocol is lower bounded by the entropy of this partition, denoted by $H(p(d))$, where $p(d)$ is the implied probability distribution on the elements of the partition. Thus, the number of bits that must be exchanged under *any* protocol is bounded by

$$\mathcal{C}(\Pi_{n-k}(\mathbf{X}^n)) \geq \min_{d \in D(\Pi_{n-k}(\mathbf{X}^n))} H(p(d)) =: \mathcal{C}_L(\Pi_{n-k}(\mathbf{X}^n)). \tag{7.18}$$

We conjecture that this lower bound is achievable and, in particular, the optimal strategy described

in Theorem 39 achieves it.

Conjecture 40. *The lower bound described in (7.18) satisfies the dynamic programming equation in (7.17).*

$$\mathcal{C}_L(\Pi_{n-k}(\mathbf{X}^n)) = \min_i \{H(p_i) + p_i \mathcal{C}_L(\Pi_{n-k-1}(\mathbf{X}^n_{-i})) + (1-p_i)\mathcal{C}_L(\Pi_{n-k}(\mathbf{X}^n_{-i}))\}.$$

Since $\mathcal{C}_L(\Pi_{n-k}(\mathbf{X}^n)) \leq \mathcal{C}(\Pi_{n-k}(\mathbf{X}^n)) \leq \mathcal{C}_U(\Pi_{n-k}(\mathbf{X}^n))$*, we conjecture that the optimal strategy described in Theorem 39 is in fact optimal among all interactive strategies.*

We note that the above conjecture has been verified by hand for all threshold functions of three variables. A formal proof of the conjecture, however, remains a challenge for the future.

7.3 Computation under an Alternate Communication Model

In this section, we illustrate how we can adapt the solution to the general dynamic programming equation described in Lemma 33 to a different communication model. We return to the problem of computing a single instance of a Boolean threshold function $\Pi_{n-k}(\mathbf{X}^n)$ in the broadcast scenario. Let us suppose that time is slotted, and that nodes transmit information in the form of pulses, which have unit energy cost. Under this alternate communication model, our modified objective is to minimize the expected total energy expended in transmissions.

In contrast to Section 7.1 where the cost of transmitting a bit is uniformly 1, under the pulse model of communication, silence can be used to convey information with zero cost. This is similar to the model for silence-based communication studied in [61]. Thus, the communication problem is no longer trivial. However, since each node makes a Boolean measurement, the value of its bit can be mapped to the presence or absence of a pulse in two ways. Either node i transmits a pulse to indicate $X_i = 1$ and remains silent to indicate $X_i = 0$, or vice versa. Clearly, the optimal communication strategy is as follows:

- If $p_i \leq \frac{1}{2}$, then node i transmits a pulse to indicate $X_i = 1$.
- If $p_i \geq \frac{1}{2}$, then node i transmits a pulse to indicate $X_i = 0$.

We are still left with the problem of determining the optimal ordering of transmissions.

Let $\mathcal{E}(\Pi_{n-k}(\mathbf{X}^n))$ be the minimum expected total energy required in order to compute the threshold function $\Pi_{n-k}(\mathbf{X}^n)$ under this communication model. The problem of minimizing the expected total energy can be formulated as a dynamic programming equation as follows:

$$\mathcal{E}(\Pi_{n-k}(\mathbf{X}^n)) = \min_i \{\min(p_i, 1-p_i) + p_i \mathcal{E}(\Pi_{n-k-1}(\mathbf{X}^n_{-i})) + (1-p_i)\mathcal{E}(\Pi_{n-k}(\mathbf{X}^n_{-i}))\}. \tag{7.19}$$

From Lemma 33, we have the following result.

Theorem 41. *In order to compute a single instance of the Boolean threshold function $\Pi_{n-k}(\mathbf{X}^n)$ under the pulse communication model, it is optimal for node $k+1$ to transmit first.*

Proof: Observe that (7.19) is a special case of (7.2) where $f(p) = \min(p, 1-p)$. Hence, in order to establish the result, it is sufficient to show that $\min(p, 1-p)$ satisfies the conditions in Lemma 33. Indeed, $\min(p, 1-p)$ is symmetric about $p = \frac{1}{2}$ and we have

$$g(p) = \frac{\min(p, 1-p)}{p} = \begin{cases} 1 & \text{if } p \leq \frac{1}{2}, \\ \frac{1-p}{p} & \text{if } p > \frac{1}{2}. \end{cases}$$

Thus, $\frac{\min(p,1-p)}{p}$ is a monotone non-increasing function of p. The theorem then follows directly from Lemma 33. $\square$

7.4 Approximate Function Computation

In Sections 7.1 - 7.3, we considered the problem of computing Boolean threshold functions with zero error. While we focused on constructing optimal strategies to minimize the expected total number of bits exchanged during computation, we note that the worst-case total number of bits exchanged might still be n. In some applications however, we might have a constraint on the number of bits exchanged, or equivalently, the number of timeslots available for computation. In this case, one cannot always hope to compute the function exactly. Instead, we consider *approximate* function computation where we seek to minimize certain error metrics.

To begin with, let us consider the class of Boolean threshold functions. As before, we permit

all interactive strategies where the choice of next transmitting node can depend arbitrarily on all previously broadcast bits. Let us suppose that we are allowed to exchange at most $(n-\theta)$ bits in order to compute the threshold function $\Pi_{n-k}(\mathbf{X}^n)$. We propose two metrics of error, namely probability of error and conditional entropy of the function.

- **Probability of error:** Let $P_e^{(\theta)}(\Pi_{n-k}(\mathbf{X}^n))$ denote the minimum probability of error after at most $(n-\theta)$ bits are exchanged. Note that these bits are exchanged in sequential fashion, since we are computing in a broadcast network. Hence, the identity of each successive transmitting node can depend on the previously transmitted bits. The sequential nature of this problem permits a dynamic programming formulation analogous to (7.2).

$$P_e^{(\theta)}(\Pi_{n-k}(\mathbf{X}^n)) = \min_i \{p_i P_e^{(\theta)}(\Pi_{n-k-1}(\mathbf{X}_{-i}^n) + (1-p_i) P_e^{(\theta)}(\Pi_{n-k}(\mathbf{X}_{-i}^n)\}, \tag{7.20}$$

  with the boundary condition $P_e^{(\theta)}(\Pi_{\theta-k}(\mathbf{X}^\theta)) = \min(P(\Pi_{\theta-k}(\mathbf{X}^\theta)=1), P(\Pi_{\theta-k}(\mathbf{X}^\theta)=0))$.

- **Conditional entropy of function:** Let $H^{(\theta)}(\Pi_{n-k}(\mathbf{X}^n))$ denote the minimum conditional entropy of the function after at most $(n-\theta)$ bits are exchanged. As before, the identity of each successive transmitting node can depend on the previously transmitted bits. Once again, the sequential nature of this problem permits a dynamic programming formulation analogous to (7.2).

$$H^{(\theta)}(\Pi_{n-k}(\mathbf{X}^n)) = \min_i \{p_i H^{(\theta)}(\Pi_{n-k-1}(\mathbf{X}_{-i}^n) + (1-p_i) H^{(\theta)}(\Pi_{n-k}(\mathbf{X}_{-i}^n)\}, \tag{7.21}$$

  with the boundary condition $H^{(\theta)}(\Pi_{\theta-k}(\mathbf{X}^\theta)) = H(\Pi_{\theta-k}(\mathbf{X}^\theta))$.

7.4.1 Counter-example

At first glance, one would expect that the optimal strategy for approximate function computation would match the strategy for exact function computation, thus verifying that the strategy proposed in Theorem 38 is *increasingly correct*. Unfortunately, this is not true as shown by the following counter-example.

Let us suppose that we want to compute $\Pi_2(X_1, X_2, X_3)$ and we are allowed to exchange exactly

one bit. We have exactly three choices of strategy. Either node 1 transmits first, or node 2, or node 3. Consider the case where $p_1 = 0.7, p_2 = 0.82, p_3 = 0.84$; then one can calculate the conditional entropy when node 1 transmits first (respectively node 2 and node 3).

$$\begin{aligned}
H^{(2)}(\Pi_2(X_1, X_2, X_3)|X_1) &= p_1 H((1-p_2)(1-p_3)) + (1-p_1)H(p_2 p_3) = 0.4002. \\
H^{(2)}(\Pi_2(X_1, X_2, X_3)|X_2) &= p_2 H((1-p_1)(1-p_3)) + (1-p_2)H(p_1 p_3) = 0.4991. \\
H^{(2)}(\Pi_2(X_1, X_2, X_3)|X_3) &= p_3 H((1-p_1)(1-p_2)) + (1-p_3)H(p_1 p_2) = 0.4121.
\end{aligned}$$

Contrary to our expectation, it is not always optimal for node 2 to transmit first. This is also true for the probability of error metric. Again, consider the approximate computation of $\Pi_2(X_1, X_2, X_3)$ where we are only allowed to exchange exactly one bit. For the case where $p_1 = 0.6, p_2 = 0.72, p_3 = 0.84$; the probability of error expressions for the three strategies are given by

$$\begin{aligned}
P_e^{(2)}(\Pi_2(X_1, X_2, X_3)|X_1) &= p_1 \min((1-p_2)(1-p_3), 1-(1-p_2)(1-p_3)) \\
&\quad + (1-p_1)\min(p_2 p_3, 1-p_2 p_3) = 0.1850, \\
P_e^{(2)}(\Pi_2(X_1, X_2, X_3)|X_2) &= p_2 \min((1-p_1)(1-p_3), 1-(1-p_1)(1-p_3)) \\
&\quad + (1-p_2)\min(p_1 p_3, 1-p_1 p_3) = 0.1850, \\
P_e^{(2)}(\Pi_2(X_1, X_2, X_3)|X_3) &= p_3 \min((1-p_1)(1-p_2), 1-(1-p_1)(1-p_2)) \\
&\quad + (1-p_3)\min(p_1 p_2, 1-p_1 p_2) = 0.1632.
\end{aligned}$$

Thus, it appears that the structure of the optimal solution when we seek approximate computation given a fixed number of bits, is somewhat different from the optimal strategy for zero error computation.

7.4.2 Special case of the parity function

While the structure of the optimal strategy for the approximate computation of threshold functions remains elusive, the case of the parity function is solvable. In this section, we show that an intuitive greedy strategy is optimal for the approximate computation of the parity function. To begin with,

the parity function of n Boolean variables labeled X_1 through X_n is defined as follows:

$$\Phi(\mathbf{X}^n) := \begin{cases} 0 & \text{if } \sum_i X_i \text{ is even} \\ 1 & \text{if } \sum_i X_i \text{ is odd.} \end{cases}$$

We consider the computation of $\Phi(\mathbf{X}^n)$ in a broadcast scenario where $X_i \sim Bern(p_i)$. If we seek exact computation, the problem becomes trivial since each node must transmit its bit. Hence, we will consider approximate computation of parity under the conditional entropy metric. Let us suppose that nodes are allowed to exchange up to $(n-\theta)$ bits. Let $H^{(\theta)}(\Phi(\mathbf{X}^n))$ denote the minimum conditional entropy of the function after $(n-\theta)$ bits are exchanged. The dynamic programming equation analogous to (7.21) is

$$\begin{aligned} H^{(\theta)}(\Phi(\mathbf{X}^n)) &= \min_i \left\{ p_i H^{(\theta)}(\Phi(\mathbf{X}^n_{-i})) + (1-p_i) H^{(\theta)}(\Phi(\mathbf{X}^n_{-i})) \right\} \\ &= \min_i \left\{ H^{(\theta)}(\Phi(\mathbf{X}^n_{-i})) \right\} \end{aligned} \tag{7.22}$$

with the boundary condition $H^{(\theta)}(\Phi(\mathbf{X}^\theta)) = h(P(\Phi(\mathbf{X}^\theta) = 1))$. One can derive the solution to (7.22) and hence deduce the optimal strategy for approximate computation of parity.

Theorem 42. *In order to minimize the conditional entropy of $\Phi(\mathbf{X}^n)$ after $(n-\theta)$ bits are exchanged, it is optimal for the node with highest binary entropy to transmit first. Subsequently, the node with the next highest entropy transmits, and so on until $(n-\theta)$ bits are transmitted.*

Proof: First, we note that (7.21) implies that the optimal strategy for approximate computation of $\Phi(\mathbf{X}^n)$ is not data-dependent. Indeed, if node i transmits first, irrespective of whether $X_i = 0$ or $X_i = 1$, we have the residual task of computing $\Phi(\mathbf{X}^n_{-i})$ given at most $(n-\theta-1)$ bits. Thus, the optimal strategy can be specified *a priori* and does not depend on the particular values of the bits exchanged. Further, if our objective is to minimize the conditional entropy after $(n-\theta)$ bits, we are only interested in determining the optimal subset of nodes that must transmit, and the order of transmission within this set is irrelevant. Thus, we have

$$H^{(\theta)}(\Phi(\mathbf{X}^n)) = \min_{\substack{S \subseteq \{1,\ldots,n\} \\ |S| = n-\theta}} H(\Phi(\mathbf{X}^n)|\mathbf{X}_S).$$

Let $A = \{a_1, a_2, \ldots, a_{n-\theta}\}$ be an optimal set of nodes. We claim that A consists of nodes with the $(n-\theta)$ highest entropies among the n nodes. Suppose not. Then there exists a node $a^* \notin A$ and $a_i \in A$ such that $H(p_{a^*}) > H(p_{a_i})$. Consider the set $A^* := (A \setminus \{a_i\}) \bigcup \{a^*\}$.

$$\begin{aligned}
H(\Phi(\mathbf{X}^n)|\mathbf{X}_{A^*}) &= H(\Phi(\mathbf{X}^n)|\mathbf{X}_{A\setminus\{a_i\}}, X_{a^*}) \\
&= H(\Phi(\mathbf{X}^n_{-((A\setminus\{a_i\}),a^*)})) \\
&= H(p_{a_i} P(\Phi(\mathbf{X}^n_{-(A,a^*)})) = 1) + (1-p_{a_i}) P(\Phi(\mathbf{X}^n_{-(A,a^*)})) = 0)) \\
&\leq H(p_{a^*} P(\Phi(\mathbf{X}^n_{-(A,a^*)})) = 1) + (1-p_{a^*}) P(\Phi(\mathbf{X}^n_{-(A,a^*)})) = 0)) \\
&= H(\Phi(\mathbf{X}^n_{-((A\setminus a^*),a_i)})),
\end{aligned}$$

which contradicts the assumption that A is an optimal subset. Thus, under the toptimal strategy, the set of transmitting nodes must be those with the highest entropies. A candidate strategy which achieves this is one where nodes transmit in decreasing order of their binary entropies. □

7.5 Concluding Remarks

We have considered some sequential decision problems that arise in the context of optimal computation of Boolean functions in collocated networks. The broadcast nature of the medium forces nodes to communicate sequentially, and the challenge is to order nodes' transmissions so as to exploit both the structure of the function and the knowledge of the underlying distribution.

For single instance computation of Boolean threshold functions, we showed that the optimal strategy has an elegant structure, which depends only on the ordering of the marginal probabilities, and not on their exact values. The extension to the case of block computation is harder. However, we have derived the optimal strategy in a logical restricted class of strategies, which we conjecture to be optimal in general. The proof technique presented allows a unified treatment of these two problems, and also allows extension to alternate pulse models of communication where nodes transmit pulses of energy.

Finally, we have considered the problem of approximate function computation, where we are given a fixed number of bits and seek to minimize the error in the estimate of the function. We

have shown that this problem is more complicated and the optimal strategy lacks the structure that we observed in the case of exact computation. However, for the special case of the parity function, a simple greedy strategy is optimal. There are several directions for future work including extending these results to the case of correlated measurements, and generalizing the sequential decision making approach to handle general functions.

CHAPTER 8

DETERMINING THE CONNECTIVITY OF RANDOM GRAPHS

In this chapter, we consider the problem of determining the $s-t$ connectivity of a given random graph. We suppose that one can sample the edges of the graph. Each sample has an associated cost and we consider the optimization problem of minimizing the expected total cost incurred, in order to determine if the graph is $s-t$ connected. Such a problem arises in a network where each link has a sensor and we wish to determine connectivity in minimum expected time. The problem of determing $s-t$ connectivity can be formulated as the computation of a Boolean disjunctive normal form, corresponding to the presence of *at least* one $s-t$ path. Alternately, it can be formulated as the computation of a Boolean conjunctive normal form, corresponding to the absence of *every* $s-t$ cut. Thus, we are faced with the problem of computing an *asymmetric* Boolean function of Boolean measurements. This is in contrast with Chapter 7 where we considered a class of symmetric Boolean functions. However, the inherent sequential nature of the problem leads to a similar framework.

The general problem can be formulated as follows. Consider a graph $\mathcal{G} = (V, E)$, with designated source node s, and terminal node t. Edges occur independently of each other, and each edge $e \in E$ is present with probability p_e. Thus, a particular instance of the random graph will have vertex set V and edge set equal to a subset of E. Our objective is to construct a sequential sampling strategy to determine if the given instance of a random graph is $s-t$ connected or not. We consider the most general class of strategies where the choice of which edge to sample next is allowed to depend on all previous samples. We suppose that the cost of sampling an edge e is given by t_e. Any correct sampling strategy terminates with one of two possible outcomes, namely

- Connectivity, when we have verified the existence of all edges on an $s-t$ path, and
- Disconnectivity, when we have verified the absence of all edges on an $s-t$ cut.

Since we do not know *a priori* if the graph is connected or not, the strategy possibly alternates between trying to show the existence of an $s-t$ path and the absence of an $s-t$ cut. Since the underlying graph is random, the cost of a strategy is a random variable and we seek to minimize the expected cost. Let $C(\mathcal{G})$ denote the minimum expected cost to check connectivity of $\mathcal{G}$. We can write a dynamic programming equation for $C(\mathcal{G})$.

$$C(\mathcal{G}) = \min_{e \in E}(t_e + p_e C(\mathcal{G} \cdot e) + (1 - p_e)C(\mathcal{G} \setminus e)) \tag{8.1}$$

where $\mathcal{G} \cdot e$ is the graph obtained by contracting the edge e in $\mathcal{G}$, and $\mathcal{G} \setminus e$ is the graph obtained by removing the edge e from $\mathcal{G}$. Note that both these graphs are random graphs. The boundary condition for the above recursion is given by $C(e) = t_e$ which is obtained by considering graphs with a single edge. The recursion in (8.1) is much more complicated than the recursion for Boolean threshold functions. This is due to the fact that the set of possible graphs with a given skeleton structure is far more complex than the set of threshold functions. We have not been able to solve (8.1) for general graphs. However, for some specific classes of graphs, we obtain an elegant structure for the optimal policy of sampling edges. The intuition gained from these specific examples can be instructive in constructing heuristics for verifying connectivity of general graphs.

We start by considering the simplest class of graphs. In Section 8.1, we consider series graphs and parallel graphs where a set of links are arranged in series and parallel, respectively, between the source node s and terminal node t. We derive optimal sequential sampling strategies for determining the connectivity of these graphs. We then add another layer of complexity, and consider series-parallel graphs and parallel-series graphs in Section 8.2. In the former case, the optimal strategy involves indexing all the $s-t$ cuts in the graph, while in the latter case, the optimal strategy involves indexing all the $s-t$ paths. The extension to general graphs is harder, and could potentially involve hybrid strategies where we consider both paths and cuts.

Further, we show that the general problem is composed of two subproblems which are possibly simpler to analyze. The first problem is one of minimizing the expected cost of finding a valid $s-t$ path conditioned on the fact that the graph is connected. The second problem is one of minimizing the expected cost of finding a valid $s-t$ cut conditioned on the fact that the graph is disconnected. We conjecture the structure of the optimal solution for these two problems. The

optimal strategy for determining connectivity in general graphs could be a hedging strategy, which alternates between the optimal approaches for the two conditional subproblems.

8.1 Checking Connectivity in Series Graphs and Parallel Graphs

Consider a graph $\mathcal{S}_n$ which consists of n links in series, with source node at the leftmost end and terminal node at the rightmost end, as shown in Figure 8.1. Let us suppose that the edges are labeled e_1 through e_n, where edge e_i is present with probability p_i and has sampling cost t_i. The following theorem describes the optimal strategy for sequentially sampling a series graph.

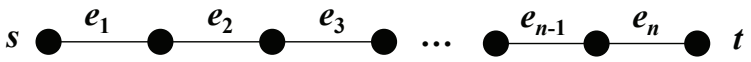

Figure 8.1: Series graph $\mathcal{S}_n$

Theorem 43. *In order to determine the $s-t$ connectivity of the series graph $\mathcal{S}_n$, it is optimal to first sample the edge e_i with the largest value of $\frac{1-p_i}{t_i}$.*

Proof: We proceed by induction on the number of edges n. For $n = 1$, the result is trivially true. Let us suppose that the result is true for the series graph $\mathcal{S}_{n-1}$. Consider a series graph $\mathcal{S}_n$ with n edges. When we sample edge e_i, there are two possible outcomes.

- Either edge e_i is absent and hence $\mathcal{S}_n$ is disconnected, or
- edge e_i is present and we need to determine the connectivity of $\mathcal{S}_n \cdot e_i$, for which we know the optimal strategy from the induction hypothesis.

Without loss of generality, we suppose that edges are indexed in decreasing order of $\frac{1-p_i}{t_i}$ to begin with. Thus, we are left to prove that it is optimal to sample the edge e_1 first. Let us denote the expected cost of a strategy where e_i is sampled first by $C(\mathcal{S}_n|e_i \text{ is sampled first})$.

$$\begin{aligned} C(\mathcal{S}_n|e_1 \text{ is sampled first}) &= t_1 + p_1 C(\mathcal{S}_n \cdot e_1) \\ &\leq t_1 + p_1(t_i + p_i C(\mathcal{S}_n \cdot \{e_1, e_i\})) \\ &= t_1 + p_1 t_i + p_1 p_i C(\mathcal{S}_n \cdot \{e_1, e_i\}) \end{aligned}$$

$$
\begin{aligned}
C(\mathcal{S}_n|e_1 \text{ is sampled first}) &\leq t_i + p_i t_1 + p_1 p_i C(\mathcal{S}_n \cdot \{e_1, e_i\}) \qquad (8.2)\\
&= t_i + p_i(t_1 + p_1 C(\mathcal{S}_n \cdot \{e_1, e_i\}))\\
&= C(\mathcal{S}_n|e_i \text{ is sampled first}),
\end{aligned}
$$

where (8.2) follows from the assumption that $\frac{1-p_1}{t_1} \geq \frac{1-p_i}{t_i}$. Thus we have shown that the expected cost is minimized when e_1 is sampled first, which establishes the induction step. □

In series graphs, the optimal strategy can be interpreted as follows. Each $s-t$ cut in the graph is indexed by the ratio of the probability of absence of the cut to the expected cost to check the cut. We then pick the cut with the highest index.

Analogously, one can define the parallel graph $\mathcal{P}_m$ as a graph with m parallel edges between nodes s and t. Edges are labeled e_1 through e_m as shown in Figure 8.2. In exactly analogous fashion, one can describe the optimal strategy for sequentially sampling a parallel graph.

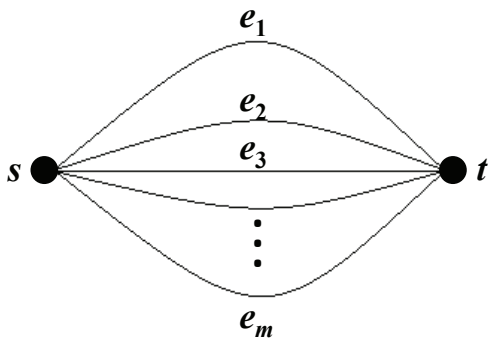

Figure 8.2: Parallel graph $\mathcal{P}_m$

Theorem 44. *In order to determine the $s-t$ connectivity of the parallel graph $\mathcal{P}_m$, it is optimal to first sample the edge e_i with the largest value of $\frac{p_i}{t_i}$.*

Proof: We proceed by induction on the number of edges m. For $m = 1$, the result is trivially true. Let us suppose that the result is true for the parallel graph $\mathcal{P}_{m-1}$. Consider a parallel graph $\mathcal{P}_m$ with m edges. When we sample edge e_i, there are two possibilities. Either e_i is present and we immediately know that $\mathcal{P}_m$ is connected, or e_i is absent and we are left with the problem of determining the connectivity of $\mathcal{P}_m \setminus e_i$. In the latter case, the optimal strategy is known from the induction hypothesis.

Without loss of generality, we suppose that edges are indexed in decreasing order of $\frac{p_i}{t_i}$ to begin with. Thus, we are left to prove that it is optimal to sample the edge e_1 first. Let us denote the expected cost of a strategy where e_i is sampled first by $C(\mathcal{P}_m|e_i \text{ is sampled first})$.

$$\begin{aligned}
C(\mathcal{P}_m|e_1 \text{ is sampled first}) &= t_1 + (1-p_1)C(\mathcal{P}_m \setminus e_1) \\
&\leq t_1 + (1-p_1)(t_i + (1-p_i)C(\mathcal{P}_m \setminus \{e_1, e_i\})) \\
&= t_1 + (1-p_1)t_i + (1-p_1)(1-p_i)C(\mathcal{P}_m \setminus \{e_1, e_i\}) \\
&\leq t_i + (1-p_i)t_1 + (1-p_1)(1-p_i)C(\mathcal{P}_m \setminus \{e_1, e_i\}) \qquad (8.3) \\
&= t_i + (1-p_i)(t_1 + (1-p_1)C(\mathcal{P}_m \setminus \{e_1, e_i\})) \\
&= C(\mathcal{P}_m|e_i \text{ is sampled first}),
\end{aligned}$$

where (8.3) follows from the assumption that $\frac{p_1}{t_1} \geq \frac{p_i}{t_i}$. Thus we have shown that the expected cost is minimized when e_1 is sampled first, which establishes the induction step. □

In parallel graphs, the optimal strategy can be interpreted as follows. Each $s-t$ path in the graph is indexed by the ratio of the probability of existence of the path to the expected cost to check the path. We then pick the path with the highest index.

8.2 Checking Connectivity of Series-Parallel Graphs and Parallel-Series Graphs

We now proceed to study optimal strategies for determining connectivity in slightly more complicated random graphs obtained from composing series and parallel graphs in prescribed ways.

8.2.1 Series-parallel graphs

We begin by considering *series-parallel* graphs, which consist of several parallel graphs arranged in series as shown in Figure 8.3. The series-parallel graph $\mathcal{SP}_n$ consists of n parallel graphs labeled $\mathcal{P}^1_{m_1}, \ldots, \mathcal{P}^n_{m_n}$, arranged in series. Further let the edges in the graph be labeled $\{e_{ij} : 1 \leq j \leq n, 1 \leq i \leq m_j\}$, where e_{ij} is the i^{th} edge in the j^{th} parallel graph. Edge e_{ij} is present with probability

p_{ij}, and has sampling cost t_{ij}. Further, suppose that within each cut $\mathcal{P}^j_{m_j}$, edges are indexed in decreasing order of $\frac{1-p_{ij}}{t_{ij}}$, i.e. $\frac{1-p_{1j}}{t_{1j}} \geq \frac{1-p_{2j}}{t_{2j}} \geq \ldots \geq \frac{1-p_{m_j j}}{t_{m_j j}}$.

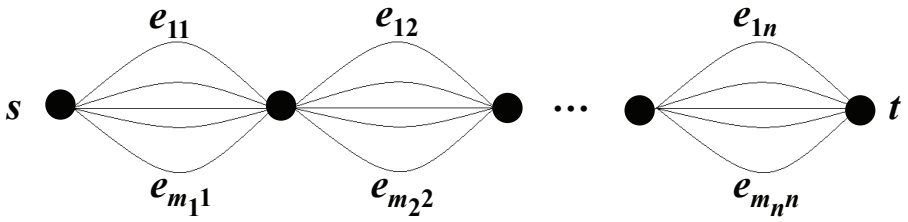

Figure 8.3: Series-parallel graph $\mathcal{SP}_n$

Theorem 45. *Consider the problem of determining the $s-t$ connectivity of a series parallel graph $\mathcal{SP}_n$. Index each $s-t$ cut in $\mathcal{SP}_n$, equivalently each component parallel graph, in the following way:*

$$J_\alpha(\mathcal{P}^j_{m_j}) = \frac{(1-p_{1j})(1-p_{2j})\cdots(1-p_{m_j j})}{t_{1j} + (1-p_{1j})t_{2j} + \ldots + (1-p_{1j})\cdots(1-p_{(m_j-1)j})t_{m_j j}}.$$

An optimal strategy is to sample the edge e with the maximum value of $\frac{p_e}{t_e}$ on the maximum index cut.

Proof: We first present a lemma regarding the behavior of the indexing function J_α, which will be critical to the proof.

Lemma 46. *Let $\mathcal{A} = \{a_1, a_2, \ldots a_m\}$ be a set of edges, where edge a_i is present with probability p_i and incurs sampling cost t_i. Then, the J_α index of any subset of $\mathcal{A}$ is greater than or equal to the J_α index of $\mathcal{A}$. Equivalently, we have $J_\alpha(\mathcal{A} \setminus a_i) \geq J_\alpha(\mathcal{A})$.*

Proof: Without loss of generality, we can suppose that $\frac{p_i}{t_i} \geq \frac{p_j}{t_j}$ if $i \leq j$. Then, we have

$$\begin{aligned}
J_\alpha(\mathcal{A} \setminus a_i) &= \frac{(1-p_1)\cdots(1-p_{i-1})(1-p_{i+1})\ldots(1-p_m)}{\Big(t_1 + (1-p_1)t_2 + \ldots + (1-p_1)\cdots(1-p_{i-1})t_{i+1} \\
&\qquad + \ldots + (1-p_1)\cdots(1-p_{i-1})(1-p_{i+1})\cdots(1-p_{m-1})t_m\Big)} \\
&= \frac{J_\alpha(\mathcal{A})}{(1-p_i)} \frac{t_1 + (1-p_1)t_2 + \ldots + (1-p_1)\cdots(1-p_{m-1})t_m}{\Big(t_1 + (1-p_1)t_2 + \ldots + (1-p_1)\cdots(1-p_{i-1})t_{i+1} \\
&\qquad + \ldots + (1-p_1)\cdots(1-p_{i-1})(1-p_{i+1})\cdots(1-p_{m-1})t_m\Big)} \\
&\geq J_\alpha(\mathcal{A}).
\end{aligned}$$

From Lemma 46, we can decipher the proposed structure of the optimal strategy. Since the index of a cut only increases when we sample an edge, the optimal strategy begins with the maximum index cut and sequentially samples edges in decreasing order of $\frac{p_e}{t_e}$. Upon finding an edge, we proceed to the next highest index cut and repeat the same procedure.

The proof of Theorem 45 proceeds by induction on the number of edges in E. For a graph with one edge, the result is trivial. Let us suppose that the result is true for all graphs with $|E| < k$. Consider a graph with $|E| = k$. Without loss of generality, we can suppose that

$$J_\alpha(\mathcal{P}^1_{m_1}) \geq J_\alpha(\mathcal{P}^2_{m_2}) \geq \dots J_\alpha(\mathcal{P}^n_{m_n}).$$

Hence, $\mathcal{P}^1_{m_1}$ is the maximum index cut and e_{11} is the prescribed edge to sample. Let e_{ij} be some other edge. When we sample edge e_{ij}, it could be present or absent, resulting in two residual graphs. Both these graphs have a smaller edge set than the original graph. From the induction hypothesis, we know the optimal strategy for each of these smaller graphs. There are two cases we need to consider.

(i) First, suppose $j = 1$. In this case, both e_{11} and e_{ij} are in the same cut. Let $C(\mathcal{SP}_n|e_{ij}$ is sampled first) denote the minimum expected cost of the strategy where we sample edge e_{ij} first and proceed optimally from there on. Suppose we sample the wrong edge e_{i1} first. Then, the index of the cut increases, and the next optimal edge to sample is e_{11} from the induction hypothesis. We have the following series of inequalities:

$$\begin{aligned}
& C(\mathcal{SP}_n|e_{i1} \text{ is sampled first}) \\
= \;& t_{i1} + p_{i1}C(\mathcal{SP}_n \cdot e_{i1}) + (1-p_{i1})C(\mathcal{SP}_n \setminus e_{i1}) \\
= \;& t_{i1} + p_{i1}C(\mathcal{SP}_n \cdot e_{i1}) + (1-p_{i1})(t_{11} + p_{11}C(\mathcal{SP}_n \cdot e_{11} \setminus e_{i1}) + (1-p_{11})C(\mathcal{SP}_n \setminus \{e_{i1}, e_{11}\}) \\
= \;& t_{i1} + (1-p_{i1})t_{11} + (p_{11} + p_{i1} - p_{i1}p_{11})C(\mathcal{SP}_n \cdot e_{11}) + (1-p_{i1})(1-p_{11})C(\mathcal{SP}_n \setminus \{e_{i1}, e_{11}\}) \\
\geq \;& t_{11} + (1-p_{11})t_{i1} + (p_{11} + p_{i1} - p_{i1}p_{11})C(\mathcal{SP}_n \cdot e_{11}) + (1-p_{i1})(1-p_{11})C(\mathcal{SP}_n \setminus \{e_{i1}, e_{11}\}) \quad (8.4) \\
= \;& t_{11} + p_{11}C(\mathcal{SP}_n \cdot e_{11}) + (1-p_{11})(t_{i1} + p_{i1}C(\mathcal{SP}_n \cdot e_{i1} \setminus e_{11}) + (1-p_{i1})C(\mathcal{SP}_n \setminus \{e_{i1}, e_{11}\})) \quad (8.5) \\
\geq \;& t_{11} + p_{11}C(\mathcal{SP}_n \cdot e_{11}) + (1-p_{11})C(\mathcal{SP}_n \setminus e_{11}) \\
= \;& C(\mathcal{SP}_n|e_{11} \text{ is sampled first}),
\end{aligned}$$

where (8.4) follows from the fact that $\frac{p_{11}}{t_{11}} \geq \frac{p_{i1}}{t_{i1}}$, and (8.5) follows from the fact that the graphs $(\mathcal{SP}_n \cdot e_{11}), (\mathcal{SP}_n \cdot e_{i1}), (\mathcal{SP}_n \cdot e_{11} \setminus e_{i1}), (\mathcal{SP}_n \cdot e_{i1} \setminus e_{11})$ are all the same. Thus we have shown that the strategy of sampling edge e_{11} first minimizes the expected cost of determining connectivity of $\mathcal{SP}_n$. This completes the induction step for the case when $j = 1$.

(ii) Second, suppose $j \neq 1$. In this case, e_{11} and e_{ij} are in different cuts, with $e_{11} \in \mathcal{P}^1_{m_1}$ and $e_{ij} \in \mathcal{P}^j_{m_j}$. Suppose we sample edge e_{ij} first. We have two subcases.

- $J_\alpha(\mathcal{P}^j_{m_j} \setminus e_{ij}) \leq J_\alpha(\mathcal{P}^1_{m_1})$. Then, by the induction hypothesis, the optimal strategy is to return to the cut $\mathcal{P}^1_{m_1}$ and sequentially sample edges until we find one. Upon finding an edge, we move to the next highest indexed cut, and repeat the procedure. At some point we will return to cut $\mathcal{P}^j_{m_j}$ and run through the rest of the edges. The expected cost of this strategy is given by

$$C(\mathcal{SP}_n | e_{ij} \text{ is sampled first}) = t_{ij} + p_{ij} C(\mathcal{SP}_n \cdot e_{ij}) + (1 - p_{ij}) C(\mathcal{SP}_n \setminus e_{ij}).$$

  To simplify the notation, we define two quantities P_j and T_j which represent the probability that $\mathcal{P}^j_{m_j}$ is connected and the minimum expected cost to check $\mathcal{P}^j_{m_j}$ respectively:

$$\begin{aligned} 1 - P_j &:= (1 - p_{1j})(1 - p_{2j}) \dots (1 - p_{m_j j}) \\ T_j &:= t_{1j} + (1 - p_{1j}) t_{2j} + \dots + (1 - p_{1j}) \dots (1 - p_{(m_j - 1)j}) t_{m_j j} = C(\mathcal{P}^j_{m_j}). \end{aligned}$$

  By definition, we see that $J_\alpha(\mathcal{P}^j_{m_j}) = \frac{1 - P_j}{T_j}$. We have the following series of inequalities:

$$\begin{aligned} & t_{ij} + p_{ij} C(\mathcal{SP}_n \cdot e_{ij}) + (1 - p_{ij}) C(\mathcal{SP}_n \setminus e_{ij}) \\ = \; & t_{ij} + p_{ij} \left(T_1 + \dots + P_1 \cdots P_{j-1} T_{j+1} + P_1 \cdots P_{j-1} P_{j+1} T_{j+2} \right. \\ & \qquad \left. + \dots + P_1 \cdots P_{j-1} P_{j+1} \cdots P_{n-1} T_n \right) \\ & + (1 - p_{ij}) \left(T_1 + P_1 T_2 + \dots + P_1 \cdots P_a C(\mathcal{P}^j_{m_j} \setminus e_{ij}) + P_1 \cdots P_a \left(\frac{P_j - p_{ij}}{1 - p_{ij}} \right) T_{a+1} \right. \\ & \qquad \left. + \dots + P_1 \cdots P_a \left(\frac{P_j - p_{ij}}{1 - p_{ij}} \right) P_{a+1} \cdots P_{j-1} P_{j+1} \cdots P_{n-1} T_n \right) \end{aligned}$$

$$
\begin{aligned}
&= (T_1 + P_1T_2 + \ldots + P_1 \cdots P_{a-1}T_a) + P_1P_2 \cdots P_a(1-p_{ij})C(\mathcal{P}^j_{m_j} \setminus e_{ij}) + t_{ij} \\
&\quad +(p_{ij} + P_j - p_{ij})\left(P_1 \cdots P_aT_{a+1} + \ldots + P_1 \cdots P_{j-1}T_{j+1} + P_1 \cdots P_{j-1}P_{j+1}T_{j+2}\right. \\
&\qquad\qquad \left. + \ldots + P_1 \cdots P_{j-1}P_{j+1} \cdots P_{n-1}T_n\right) \\
&\geq (T_1 + P_1T_2 + \ldots + P_1 \cdots P_{a-1}T_a) + P_1P_2 \cdots P_a(T_j - t_{ij}) + t_{ij} \\
&\quad +P_1 \cdots P_aP_jT_{a+1} + P_1 \cdots P_aP_jP_{a+1}T_{a+2} + \ldots + P_1 \cdots P_jT_{j+1} + \ldots + P_1 \cdots P_{n-1}T_n \quad (8.6) \\
&\geq T_1 + P_1T_2 + \ldots + P_1 \cdots P_{a-1}T_a + P_1P_2 \cdots P_aT_j + P_1 \cdots P_aP_jT_{a+1} + P_1 \cdots P_aP_jP_{a+1}T_{a+2} \\
&\quad + \ldots + P_1 \cdots P_{j-1}T_{j+1} + P_1 \cdots P_{j-1}P_{j+1}T_{j+2} + \ldots + P_1 \cdots P_{j-1}P_{j+1} \cdots P_{n-1}T_n. \\
&\geq T_1 + P_1T_2 + \ldots + P_1 \cdots P_{n-1}T_n \quad (8.7) \\
&= C(\mathcal{SP}_n | e_{11} \text{ is sampled first}),
\end{aligned}
$$

where (8.6) follows from the fact that $C(\mathcal{P}^j_{m_j}) \leq t_{ij} + (1-p_{ij})C(\mathcal{P}^j_{m_j} \setminus e_{ij})$, and (8.7) follows from Theorem 43 and the assumption that $\frac{1-P_1}{T_1} \geq \frac{1-P_2}{T_2} \geq \ldots \geq \frac{1-P_n}{T_n}$. This completes the induction step.

- $J_\alpha(\mathcal{P}^j_{m_j} \setminus e_{ij}) > J_\alpha(\mathcal{P}^1_{m_1})$. Then, by the induction hypothesis, the optimal strategy is to continue to sample the cut $\mathcal{P}^j_{m_j}$ until we find an edge. Upon finding an edge, we proceed to check the remaining cuts in the indexed order, viz., $(1, 2, \ldots, j-1, j+1, \ldots n)$. The expected cost of this strategy is given by

$$
\begin{aligned}
&C(\mathcal{SP}_n | e_{ij} \text{ is sampled first}) \\
&= T_j + P_jT_1 + P_jP_1T_2 + \ldots + P_jP_1 \cdots P_{j-1}T_{j+1} + \ldots + P_1 \cdots P_{n-1}T_n \\
&\geq T_1 + P_1T_2 + P_1P_2T_3 + \ldots + P_1 \cdots P_{n-1}T_n \quad (8.8) \\
&= C(\mathcal{SP}_n | e_{11} \text{ is sampled first}),
\end{aligned}
$$

where (8.8) follows from the optimal strategy for series graphs described in Theorem 43 and the assumption that $\frac{1-P_1}{T_1} \geq \frac{1-P_2}{T_2} \geq \ldots \geq \frac{1-P_n}{T_n}$. This completes the induction step. □

The optimal strategy for series-parallel graphs has a logical interpretation. We index each $s-t$ cut in the graph by the ratio of the probability that the cut is absent to the expected cost incurred to check the cut. We pick the maximum index cut and treat this like a parallel graph where we

pick the edge with the maximum value of $\frac{p_e}{t_e}$. Thus, the strategy exclusively focuses on cuts, and completely ignores paths.

8.2.2 Parallel - series graphs

The case of parallel-series graphs is exactly analgous. The parallel-series graph $\mathcal{PS}_m$ consists of m series graphs labeled $\mathcal{S}^1_{n_1}, \ldots, \mathcal{S}^n_{n_m}$, arranged in parallel as shown in Figure 8.4. Further let the edges in the graph be labeled $\{e_{ij} : 1 \leq i \leq m, 1 \leq j \leq n_i\}$, where e_{ij} is the j^{th} edge in the i^{th} series graph. Edge e_{ij} is present with probability p_{ij}, and has sampling cost t_{ij}. Further, suppose that within path $\mathcal{S}^i_{n_i}$, edges are indexed in decreasing order of $\frac{p_{ij}}{t_{ij}}$, i.e. $\frac{p_{i1}}{t_{i1}} \geq \frac{p_{i2}}{t_{i2}} \geq \ldots \geq \frac{p_{in_i}}{t_{in_i}}$.

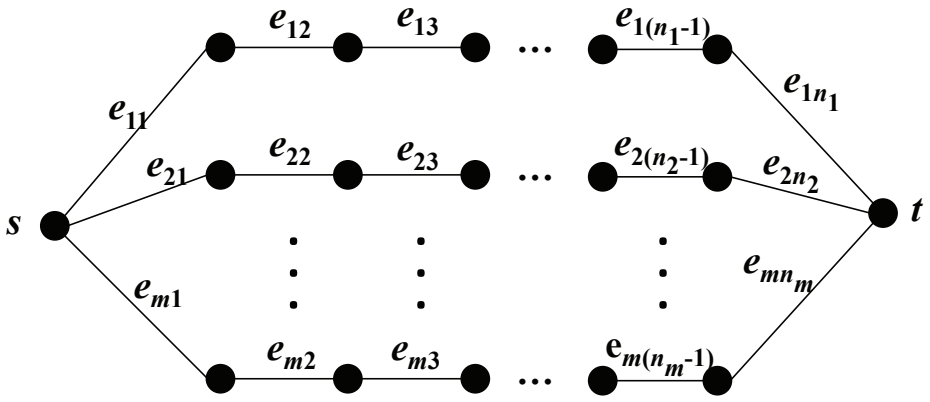

Figure 8.4: Parallel-series graph $\mathcal{PS}_m$

Theorem 47. *Consider the problem of determining the $s-t$ connectivity of a parallel-series graph $\mathcal{PS}_m$. Index each $s-t$ path in $\mathcal{PS}_m$, equivalently each component series graph, in the following way:*

$$J_\beta(\mathcal{S}^i_{n_i}) = \frac{p_{i1}p_{i2}\cdots p_{in_i}}{t_{i1} + p_{i1}t_{i2} + \ldots + p_{i1}\cdots p_{i(n_i-1)}t_{in_i}}.$$

The optimal strategy is to sample the edge e with the maximum value of $\frac{1-p_e}{t_e}$ on the maximum index path.

Proof: We first present a lemma regarding the behavior of the indexing function J_β, that will be critical to the proof.

Lemma 48. *Let $\mathcal{A} = \{a_1, a_2, \ldots a_n\}$ be a set of edges, where edge a_i is present with probability p_i and incurs sampling cost t_i. Then, the J_β index of any subset of $\mathcal{A}$ is greater than or equal to the*

J_β index of $\mathcal{A}$. Equivalently, we have $J_\beta(\mathcal{A} \setminus a_i) \geq J_\beta(\mathcal{A})$.

Proof: Without loss of generality, we can suppose that $\frac{1-p_i}{t_i} \geq \frac{1-p_j}{t_j}$ if $i \leq j$. Then, we have

$$\begin{aligned} J_\beta(\mathcal{A} \setminus a_i) &= \frac{p_1 \cdots p_{i-1} p_{i+1} \cdots p_n}{t_1 + p_1 t_2 + \ldots + p_1 \cdots p_{i-1} t_{i+1} + \ldots + p_1 \cdots p_{i-1} p_{i+1} \cdots p_{n-1} t_n} \\ &= \frac{J_\beta(\mathcal{A})}{p_i} \frac{t_1 + p_1 t_2 + \ldots + p_1 \cdots p_{n-1} t_n}{t_1 + p_1 t_2 + \ldots + p_1 \cdots p_{i-1} t_{i+1} + \ldots + p_1 \cdots p_{i-1} p_{i+1} \cdots p_{n-1} t_n}. \\ &\geq J_\beta(\mathcal{A}). \end{aligned}$$

From Lemma 48, we can decipher the proposed structure of the optimal strategy. Since the index of a path only increases when we sample an edge, the optimal strategy begins with the maximum index path and sequentially samples edges in decreasing order of $\frac{1-p_e}{t_e}$. Upon finding a missing edge, we proceed to the next highest index path and repeat the same procedure.

The proof of Theorem 47 proceeds by induction on the number of edges in E. For a graph with one edge, the result is trivial. Let us suppose that the result is true for all graphs with $|E| < k$. Consider a graph with $|E| = k$. Without loss of generality, we can suppose that

$$J_\beta(\mathcal{S}^1_{n_1}) \geq J_\beta(\mathcal{S}^2_{n_2}) \geq \ldots J_\beta(s^m_{n_m}).$$

Hence, $\mathcal{S}^1_{n_1}$ is the maximum index path and e_{11} is the prescribed edge to sample. Let e_{ij} be some other edge. When we sample edge e_{ij}, it could be present or absent, resulting in two residual graphs. Both these graphs have a smaller edge set than the original graph. From the induction hypothesis, we know the optimal strategy for each of these smaller graphs. There are two cases we need to consider.

(i) First, suppose $i = 1$. In this case, both e_{11} and e_{ij} are on the same path. Let $C(\mathcal{SP}_n | e_{ij}$ is sampled first) denote the minimum expected cost of the strategy where we sample edge e_{ij} first and proceed optimally from there on. Suppose we sample the wrong edge e_{1j} first. Then, the index of path increases, and the next optimal edge to sample is e_{11} from the induction hypothesis. We

have the following series of inequalities:

$$
\begin{aligned}
& C(\mathcal{PS}_m | e_{1j} \text{ is sampled first}) \\
= \;& t_{1j} + (1-p_{1j})C(\mathcal{PS}_m \setminus e_{1j}) + p_{1j}C(\mathcal{PS}_m \cdot e_{1j}) \\
= \;& t_{1j} + (1-p_{1j})C(\mathcal{PS}_m \setminus e_{1j}) + p_{1j}(t_{11} + (1-p_{11})C(\mathcal{PS}_m \cdot e_{1j} \setminus e_{11}) + p_{11}C(\mathcal{PS}_m \cdot \{e_{11}, e_{1j}\})) \\
= \;& t_{1j} + p_{1j}t_{11} + (1-p_{1j}p_{11})C(\mathcal{PS}_m \cdot e_{11}) + p_{1j}p_{11}C(\mathcal{PS}_m \cdot \{e_{11}, e_{1j}\}) \\
\geq \;& t_{11} + p_{11}t_{1j} + (1-p_{1j}p_{11})C(\mathcal{PS}_m \cdot e_{1j} \setminus e11) + p_{1j}p_{11}C(\mathcal{PS}_m \cdot \{e_{11}, e_{1j}\}) \qquad (8.9) \\
= \;& t_{11} + (1-p_{11})C(\mathcal{PS}_m \setminus e_{11}) + p_{11}(t_{1j} + (1-p_{1j})C(\mathcal{PS}_m \cdot e_{11} \setminus e_{1j}) + p_{1j}C(\mathcal{PS}_m \cdot \{e_{11}, e_{1j}\}) \\
& \qquad (8.10) \\
\geq \;& t_{11} + (1-p_{11})C(\mathcal{PS}_m \setminus e_{11}) + p_{11}C(\mathcal{PS}_m \cdot e_{11}) \\
= \;& C(\mathcal{PS}_m | e_{11} \text{ is sampled first}),
\end{aligned}
$$

where (8.9) follows from the fact that $\frac{1-p_{11}}{t_{11}} \geq \frac{1-p_{1j}}{t_{1j}}$, and (8.10) follows from the fact that the graphs $(\mathcal{PS}_m \setminus e_{11}), (\mathcal{PS}_m \setminus e_{i1}), (\mathcal{PS}_m \cdot e_{1j} \setminus e_{11}), (\mathcal{SP}_n \cdot e_{11} \setminus e_{1j})$ are all the same. Thus we have shown that the strategy of sampling edge e_{11} first minimizes the expected cost of determining the $s-t$ connectivity of $\mathcal{PS}_m$. This completes the induction step for the case when $i = 1$.

(ii) Second, suppose $i \neq 1$. In this case, e_{11} and e_{ij} are on different paths, with $e_{11} \in \mathcal{S}^1_{n_1}$ and $e_{ij} \in \mathcal{S}^i_{n_i}$. Suppose we sample edge e_{ij} first. We have two subcases.

- $J_\beta(\mathcal{S}^i_{n_i} \setminus e_{ij}) \leq J_\beta(\mathcal{S}^1_{n_1})$. Then, by the induction hypothesis, the optimal strategy is to return to the path $\mathcal{S}^1_{n_1}$ and sequentially sample edges until we find a missing edge. Upon finding a missing edge, we move to the next highest indexed path, and repeat the procedure. At some point, we will return to path $\mathcal{S}^i_{n_i}$ and run through the rest of the edges. The expected cost of this strategy is given by

$$C(\mathcal{PS}_m | e_{ij} \text{ is sampled first}) = t_{ij} + p_{ij}C(\mathcal{PS}_m \cdot e_{ij}) + (1-p_{ij})C(\mathcal{PS}_m \setminus e_{ij}).$$

  To simplify the notation, we define two quantities P_i and T_i which represent the probability

that $\mathcal{S}^i_{n_i}$ is connected and the minimum expected cost to check $\mathcal{S}^i_{n_i}$ respectively:

$$\begin{aligned} P_i &:= p_{i1}p_{i2}\dots p_{in_i} \\ T_i &:= t_{i1} + p_{i1}t_{i2} + \dots + p_{i1}\dots p_{i(n_i-1)})t_{in_i} = C(\mathcal{S}^i_{n_i}). \end{aligned}$$

By definition, we see that $J_\beta(\mathcal{S}^i_{n_i}) = \frac{P_i}{T_i}$. Let us also define $Q_i := 1 - P_i$. We have the following series of inequalities:

$$\begin{aligned} & t_{ij} + p_{ij}C(\mathcal{PS}_m \cdot e_{ij}) + (1-p_{ij})C(\mathcal{PS}_m \setminus e_{ij}) \\ = \;& t_{ij} + (1-p_{ij})\left(T_1 + \dots + Q_1\cdots Q_{i-1}T_{i+1} + Q_1\cdots Q_{i-1}Q_{i+1}T_{i+2}\right. \\ & \qquad \left. + \dots + Q_1\cdots Q_{i-1}Q_{i+1}\cdots Q_{m-1}T_m\right) \\ & + p_{ij}\left(T_1 + Q_1T_2 + \dots + Q_1\cdots Q_aC(S^i_{n_i}\cdot e_{ij}) + Q_1\cdots Q_a\left(\frac{p_{ij}-P_i}{p_{ij}}\right)T_{a+1}\right. \\ & \qquad \left. + \dots + Q_1\cdots Q_a\left(\frac{p_{ij}-P_i}{p_{ij}}\right)Q_{a+1}\cdots Q_{i-1}Q_{i+1}\cdots Q_{m-1}T_m\right) \\ = \;& (T_1 + Q_1T_2 + \dots + Q_1\cdots Q_{a-1}T_a) + Q_1\cdots Q_ap_{ij}C(\mathcal{S}^i_{n_i}\cdot e_{ij}) + t_{ij} \\ & + (p_{ij} - P_i + 1 - p_{ij})\left(Q_1\cdots Q_aT_{a+1} + \dots + Q_1\cdots Q_{i-1}T_{i+1} + Q_1\cdots Q_{i-1}Q_{i+1}T_{i+2}\right. \\ & \qquad \left. + \dots + Q_1\cdots Q_{i-1}Q_{i+1}\cdots Q_{m-1}T_m\right) \\ \geq \;& (T_1 + Q_1T_2 + \dots + Q_1\cdots Q_{a-1}T_a) + Q_1Q_2\cdots Q_a(T_j - t_{ij}) + t_{ij} + Q_1\cdots Q_aQ_jT_{a+1} \\ & + Q_1\cdots Q_aQ_jQ_{a+1}T_{a+2} + \dots + Q_1\cdots Q_iT_{i+1} + \dots + Q_1\cdots Q_{m-1}T_n. \qquad (8.11) \\ \geq \;& T_1 + Q_1T_2 + \dots + Q_1\cdots Q_{a-1}T_a + Q_1\cdots Q_aQ_jT_{a+1} + Q_1\cdots Q_aQ_jQ_{a+1}T_{a+2} \\ & + \dots + Q_1\cdots Q_{j-1}T_{j+1} + Q_1\cdots Q_{j-1}Q_{j+1}T_{j+2} + \dots + Q_1\cdots Q_{j-1}Q_{j+1}\cdots Q_{m-1}T_m. \\ \geq \;& T_1 + Q_1T_2 + \dots + Q_1\cdots Q_{m-1}T_m \qquad (8.12) \\ = \;& C(\mathcal{PS}_m | e_{11} \text{ is sampled first}), \end{aligned}$$

where (8.11) follows from the fact that $C(\mathcal{S}^i_{n_i}) \leq t_{ij} + p_{ij}C(\mathcal{S}^i_{n_i} \setminus e_{ij})$, and (8.12) follows from Theorem 44 and the assumption that $\frac{P_1}{T_1} \geq \frac{P_2}{T_2} \geq \dots \geq \frac{P_n}{T_n}$. This completes the induction step.

- $J_\beta(\mathcal{S}^i_{n_i} \setminus e_{ij}) > J_\beta(\mathcal{S}^1_{n_1})$. Then, by the induction hypothesis, the optimal strategy is to continue to sample the path $\mathcal{S}^i_{n_i}$ until we find a missing edge. Upon finding a missing edge, we proceed

to check the remaining paths in the indexed order, viz., $(1, 2, \ldots, i-1, i+1, \ldots n)$. The expected cost of this strategy is given by

$$\begin{aligned} & C(\mathcal{PS}_m | e_{ij} \text{ is sampled first}) \\ = \; & T_i + Q_i T_1 + Q_i Q_1 T_2 + \ldots + Q_i Q_1 \cdots Q_{i-1} T_{i+1} + \ldots + Q_1 \cdots Q_{n-1} T_n \\ \geq \; & T_1 + Q_1 T_2 + Q_1 Q_2 T_3 + \ldots + Q_1 \cdots Q_{n-1} T_n \qquad (8.13) \\ = \; & C(\mathcal{PS}_m | e_{11} \text{ is sampled first}), \end{aligned}$$

where (8.13) follows from the optimal strategy for series graphs described in Theorem 43 and the assumption that $\frac{1-P_1}{T_1} \geq \frac{1-P_2}{T_2} \geq \ldots \geq \frac{1-P_n}{T_n}$. This completes the induction step. □

The optimal strategy for parallel-series graphs has the following interpretation. We index each $s-t$ path in the graph by the ratio of the probability that the path is present to the minimum expected cost incurred to check the path. We pick the maximum index path and treat this like a series graph where we pick the edge with the maximum value of $\frac{1-p_e}{t_e}$. Thus the strategy exclusively focuses on paths and completely ignores cuts.

8.3 Extension to General Graphs

Paths and cuts are natural primitives to consider when we speak of connectivity. The question of whether a graph is connected can be resolved if we can display an $s-t$ path or an $s-t$ cut. Indeed, any strategy which successfully determines connectivity by sequentially sampling edges, will also deduce an $s-t$ path or $s-t$ cut as a certificate. However, since we do not know *a priori* whether the given instance of the random graph is connected or not, any strategy for determining connectivity oscillates between trying to show that all edges of some $s-t$ cut are absent, and trying to show that all edges of an $s-t$ path are present. Further, the optimal strategy is likely to be dynamic, changing based on the presence or absence of previously sampled edges. This inherent tension in the problem of determining connectivity makes the problem intractable in general.

However, in the case of series-parallel and parallel-series graphs, it is possible to find the optimal strategy as shown in Section 8.2. In these cases, one can resolve the tension between displaying a

cut and displaying a path. For series-parallel graphs, the optimal strategy focuses entirely on cuts, while for parallel-series graphs, the optimal strategy focuses entirely on paths. Unfortunately, for even the smallest graph which is not series-parallel or parallel-series (e.g., the graph in Figure 8.5), the optimal strategy remains elusive.

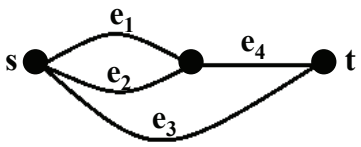

Figure 8.5: Smallest graph for which optimal strategy is unknown

In order to make progress on the problem of verifying connectivity in general graphs, one can consider the following approach. Let us suppose we know if the graph is connected to start with. Our task is to find an $s-t$ cut or an $s-t$ path which serves as a certificate. This is equivalent to minimizing the conditional expected cost of *verifying* the $s-t$ connectivity given that the graph is $s-t$ connected or $s-t$ disconnected. We formulate the following two conditional problems that are intimately connected with the original problem of determining $s-t$ connectivity in a general graph.

- Given that the particular instance of the random graph is $s-t$ disconnected, we want to find an $s-t$ cut using a sequential sampling strategy that incurs minimum expected cost. Mathematically speaking, for a given graph $\mathcal{G}$, we seek to minimize $\mathbf{E}(C(\mathcal{G})|\mathcal{G}$ is $s-t$ disconnected).

- Given that the particular instance of the random graph is $s-t$ connected, we would like to find an $s-t$ path using a sequential sampling strategy that incurs minimum expected cost. Mathematically speaking, for a given graph $\mathcal{G}$, we seek to minimize $\mathbf{E}(C(\mathcal{G})|\mathcal{G}$ is connected).

Aside from the fact that the solution of the above problems sheds some light on the nature of the solution for the unconditional problem, the conditional subproblem is also interesting in its own right. For instance, given a communication network or a network of valves in a chemical plant, one would like to construct a strategy which discovers if the network is disconnected in minimum time. On the other hand, we can afford to ignore the time consumed in the case that the network is connected.

The devolution of the general problem into two conditional problems resolves the tension between checking paths and cuts. We conjecture that these problems admit an optimal solution that focuses exclusively on cuts, or paths. However, a formal rigorous proof of this fact remains a challenge for the future.

Conjecture 49. *Consider a graph $\mathcal{G}$ with source node s and terminal node t.*

- *In order to minimize $\mathbf{E}(C(\mathcal{G})|\mathcal{G}$ is $s-t$ disconnected$)$, the following strategy is optimal. Index each $s-t$ cut in the graph by the ratio of the probability that the edges in the cut are absent to the expected cost incurred to check the cut. We first sample the edge with the maximum value of $\frac{1-p_e}{t_e}$ on the maximum index cut.*
- *In order to minimize $\mathbf{E}(C(\mathcal{G})|\mathcal{G}$ is $s-t$ connected$)$, the following strategy is optimal. Index each $s-t$ path in the graph by the ratio of the probability that the path is present to the expected cost incurred to check the path. We first sample the edge with the maximum value of $\frac{p_e}{t_e}$ on the maximum index path.*

8.4 Concluding Remarks

In this chapter, we have considered the problem of determining $s-t$ connectivity in a random graph $\mathcal{G}$, by sampling edges one-by-one. A key aspect of the problem is that since we do not know if the graph is connected or not, so we are torn between proving the absence of an $s-t$ cut and proving the existence of an $s-t$ path. In some special cases, this tension can be resolved, yielding optimal strategies for series graphs, parallel graphs, series-parallel graphs and parallel series graphs. While the optimal solution has an elegant intuitive interpretation, the extension to general graphs remains a challenge. We have formulated a related conditional problem for which we have conjectured the nature of the optimal strategy.

CHAPTER 9

CONCLUDING REMARKS

In this thesis, we have addressed a variety of problems that arise in the context of information aggregation in sensor networks. While the general problem of devising optimal strategies for function computation in wireless networks appears formidable, we have addressed different aspects of the problem and presented optimal strategies for computing specific classes of functions and specific network topologies. We have addressed three key questions that arise, namely which node should transmit, what information it should send, and how that information must be encoded.

Beginning with the problem of zero error function computation in directed graphs, we have analyzed both worst case and average case metrics. For directed tree graphs, we have constructed optimal encoding schemes on each edge. This matches the cut-set lower bounds. For general DAGs, we have provided an outer bound on the rate region, and an achievable region based on aggregating along subtrees. It remains a challenge to quantify the sub-optimality of tree aggregation schemes in general, and to develop better lower bounds for computation on graphs.

Next, we have addressed the problem of computing sum-threshold functions in undirected trees, where all nodes seek to compute the function. In this case, the optimal strategy for block computation involves a layering of transmissions that is reminiscent of message passing. For general undirected graphs, we have presented cut-set lower bounds and a subtree-based aggregation scheme for achievability. While we have shown that tree aggregation is a 2-OPT solution for complete graphs, it remains a challenge to bound their suboptimality in general graphs.

Next, we have addressed the problem of computing symmetric Boolean functions in collocated networks. We have derived optimal strategies for computing threshold functions and order optimal strategies with optimal preconstant for interval functions. The approach presented can be extended to non-Boolean alphabets and non-Boolean valued functions. It remains a challenge to derive optimal strategies for other classes of functions, and to study examples where multi-round

interaction is beneficial.

Finally, we have addressed some sequential decision problems that arise in the context of computing Boolean functions of random data. We have derived the optimal transmission policy for computing a single instance of a Boolean threshold function in a collocated network. The proof of an analogous result for block computation remains a challenge. We have also considered the problem of determining connectivity in random graphs by sampling edges, and have presented optimal sampling strategies for some special classes of graphs. For general graphs, there are several open problems that remain.

REFERENCES

[1] A. Giridhar and P. R. Kumar, "Computing and communicating functions over sensor networks," *IEEE Journal on Selected Areas in Communication*, vol. 23, no. 4, pp. 755–764, April 2005.

[2] H. Kowshik and P. R. Kumar, "Zero-error function computation in sensor networks," in *Proceedings of the 48th IEEE Conference on Decision and Control (CDC)*, December 2009, pp. 3787–3792.

[3] H. Kowshik and P. R. Kumar, "Optimal computation of symmetric Boolean functions in tree networks," in *Proceedings of the International Symposium on Information Theory (ISIT)*, June 2010, pp. 1873–1877.

[4] H. Kowshik and P. R. Kumar, "Optimal strategies for computing symmetric Boolean functions in collocated networks," in *Proceedings of the Information Theory Workshop (ITW)*, January 2010, pp. 1–5.

[5] H. Kowshik and P. R. Kumar, "Optimal ordering of transmissions for computing boolean threshold functions," in *Proceedings of the International Symposium on Information Theory, Austin, TX*, June 2010, pp. 1863–1867.

[6] N. M. Freris, H. Kowshik, and P. R. Kumar, "Fundamentals of large sensor networks: Connectivity, capacity, clocks, and computation," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1828–1846, November 2010.

[7] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, July 1948.

[8] E. Kushilevitz and N. Nisan, *Communication Complexity*. New York, NY, USA: Cambridge University Press, 1997.

[9] S. Subramanian, P. Gupta, and S. Shakkottai, "Scaling bounds for function computation over large networks," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, June 2007, pp. 136–140.

[10] J. Korner and A. Orlitsky, "Zero error information theory," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2207–2229, October 1998.

[11] H. Witsenhausen, "The zero-error side information problem and chromatic numbers," *IEEE Transactions on Information Theory*, vol. 22, pp. 592–593, September 1976.

[12] N. Alon and A. Orlitsky, "Source coding and graph entropies," *IEEE Transactions on Information Theory*, vol. 42, pp. 1329–1339, September 1996.

[13] R. Ahlswede, N. Cai, and Z. Zhang, "On interactive communication," *IEEE Transactions on Information Theory*, vol. 43, no. 1, pp. 22–37, January 1997.

[14] A. Orlitsky, "Worst-case interactive communication I: Two messages are almost optimal," *IEEE Transactions on Information Theory*, vol. 36, no. 5, pp. 1111–1127, 1990.

[15] A. Orlitsky, "Worst-case interactive communication II: Two messages are not optimal," *IEEE Transactions on Information Theory*, vol. 37, no. 4, pp. 995–1005, 1991.

[16] A. C. Yao, "Some complexity questions related to distributed computing," in *Proceedings of the 11th Symposium on the Theory of Computing (STOC)*, 1979, pp. 209–213.

[17] I. Wegener, *The Complexity of Boolean Functions*. New York, NY, USA: J. Wiley & Sons, Inc., 1987.

[18] A. Orlitsky and A. E. Gamal, "Average and randomized communication complexity," *IEEE Transactions on Information Theory*, vol. 36, pp. 3–16, 1990.

[19] M. Karchmer, R. Raz, and A. Wigderson, "Super-logarithmic depth lower bounds via direct sum in communication coplexity," in *Structure in Complexity Theory Conference*, 1991, pp. 299–304.

[20] R. Ahlswede and N. Cai, "On communication complexity of vector-valued functions," *IEEE Transactions on Information Theory*, vol. 40, pp. 2062–2067, 1994.

[21] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.

[22] A. Ramamoorthy, "Communicating the sum of sources over a network," in *Proceedings of the International Symposium on Information Theory*, July 2008, pp. 1646–1650.

[23] A. Karandikar, B. K. Rai, and B. K. Dey, "Some results on communicating the sum of sources over a network," in *Proceedings of the Workshop on Network Coding, Theory, and Applications (NetCod)*, June 2009, pp. 92–97.

[24] R. Appuswamy, M. Franceschetti, N. Karamchandani, and K. Zeger, "Network coding for computing," in *Proceedings of the 46th Annual Allerton Conference on Communication, Control, and Computing*, September 2008, pp. 1–6.

[25] F. R. Kschischang, B. J. Frey, and H. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, February 2001.

[26] S. Aji and R. Mceliece, "The generalized distributive law," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 325–343, 2000.

[27] R. Karp, C. Schindelhauer, S. Shenker, and B. Vcking, "Randomized rumor spreading," in *IEEE Symposium on Foundations of Computer Science*, 2000, pp. 565–574.

[28] A. Jadbabaie, J. Lin, and S. A. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, vol. 48, pp. 988–1001, June 2003.

[29] D. Mosk-Aoyama and D. Shah, "Computing separable functions via gossip," in *Proceedings of the 25th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, 2006, pp. 113–122.

[30] A. Kashyap, T. Basar, and R. Srikant, "Quantized consensus," *Automatica*, vol. 43, no. 7, pp. 1192–1203, July 2007.

[31] A. Orlitsky and A. E. Gamal, "Communication with secrecy constraints," in *Proceedings of the 16th Annual ACM Symposium on the Theory of Computing*, April 1984, pp. 217–224.

[32] E. Modiano and A. Ephremides, "Communication complexity of secure distributed computation in the presence of noise," *IEEE Transactions on Information Theory*, vol. 38, no. 4, pp. 1193–1202, July 1992.

[33] E. Modiano and A. Ephremides, "Communication protocols for secure distributed computation of binary functions," *Information and Computation*, vol. 58, no. 2, pp. 71–97, April 2000.

[34] H. Tyagi, P. Narayan, and P. Gupta, "Secure computing," in *Proceedings of the International Symposium on Information Theory*, June 2010, pp. 2612–2616.

[35] H. Attiya, M. Snir, and M. K. Warmuth, "Computing on an anonymous ring," *Journal of the Association for Computing Machinery*, vol. 35, no. 4, pp. 845–875, October 1988.

[36] J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, "Distributed anonymous function computation in information fusion and multiagent systems," in *Proceedings of the 47th Annual Allerton Conference on Communication, Control, and Computing*, September 2009, pp. 1582–1589.

[37] A. Giridhar and P. R. Kumar, "Toward a theory of in-network computation in wireless sensor networks," *IEEE Communications Magazine*, vol. 44, pp. 98–107, 2006.

[38] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, no. 4, pp. 471–480, 1973.

[39] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Transactions on Information Theory*, vol. 22, no. 1, pp. 1–10, January 1976.

[40] A. Orlitsky and J. R. Roche, "Coding for computing," *IEEE Transactions on Information Theory*, vol. 47, pp. 903–917, 2001.

[41] V. Doshi, D. Shah, M. Medard, and S. Jaggi, "Distributed functional compression through graph coloring," in *Data Compression Conference*, 2007, pp. 93–102.

[42] S. Feizi and M. Medard, "When do only sources need to compute? on functional compression in tree networks," in *Proceedings of the 47th Annual Allerton Conference on Communication, Control, and Computing*, 2009, pp. 447–454.

[43] J. Korner and K. Marton, "How to encode the modulo-two sum of binary sources," *IEEE Transactions on Information Theory*, vol. 25, no. 2, pp. 219–221, March 1979.

[44] N. Ma and P. Ishwar, "Two-terminal distributed source coding with alternating messages for function computation," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2008, pp. 51–55.

[45] N. Ma, P. Ishwar, and P. Gupta, "Information-theoretic bounds for multiround function computation in collocated networks," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2009, pp. 2306–2310.

[46] R. D. Gallager, "Finding parity in a simple broadcast network," *IEEE Transactions on Information Theory*, vol. 34, no. 2, pp. 176–180, March 2008.

[47] C. Dutta, Y. Kanoria, D. Manjunath, and J. Radhakrishnan, "A tight lower bound for parity in noisy communication networks," in *Proceedings of the 20th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, January 2008, pp. 1056–1065.

[48] E. Kushilevitz and Y. Mansour, "Computation in noisy radio networks," in *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1998, pp. 236–243.

[49] U. Feige and J. Kilian, "Finding OR in a noisy broadcast network," *Information Processing Letters*, vol. 73, no. 1-2, pp. 69–75, January 2000.

[50] L. Ying, R. Srikant, and G. E. Dullerud, "Distributed symmetric function computation in noisy wireless sensor networks," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4826–4833, December 2007.

[51] Y. Kanoria and D. Manjunath, "On distributed computation in noisy random planar networks," in *Proceedings of IEEE International Symposium on Information Theory*, June 2007, pp. 626–630.

[52] B. Nazer and M. Gastpar, "Computation over multiple-access channels," *IEEE Transactions on Information Theory*, vol. 53, no. 10, pp. 3498–3516, October 2007.

[53] J. C. Gittins, "Bandit processes and dynamic allocation indices," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 41, no. 2, pp. 148–177, 1979.

[54] J. Gittins and D. Jones, "A dynamic allocation index for the sequential design of experiments," in *Progress in Statistics*, J. Gani, K. Sarkadi, and I. Vincze, Eds. Amsterdam, NL: North-Holland, 1974, pp. 241–266.

[55] P. Whittle, "Multi-armed bandits and the Gittins index," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 42, no. 2, pp. 143–149, 1980.

[56] R. Ahlswede and I. Wegener, *Search Problems*. New York, NY, USA: John Wiley & Sons, Inc., 1987.

[57] E. Wong, "A linear search problem," *SIAM Review*, vol. 6, no. 2, pp. 168–174, April 1964.

[58] K. J. Arrow, L. Pesotchinsky, and M. Sobel, "On partitioning of a sample with binary-type questions in lieu of collecting observations," *Journal of the American Statistical Association*, vol. 76, no. 374, pp. 402–409, June 1981.

[59] Y. Ben-Asher and I. Newman, "Decision trees with boolean threshold queries," *Journal of Computer and System Sciences*, vol. 51, no. 3, pp. 495–502, 1995.

[60] D. West, *Combinatorial Mathematics*. Course notes for ECE 580, Department of Mathematics, University of Illinois at Urbana-Champaign, 2008.

[61] A. K. Dhulipala, C. Fragouli, and A. Orlitsky, "Silence-based communication," *IEEE Transactions on Information Theory*, vol. 56, no. 1, pp. 350–366, January 2010.

AUTHOR'S BIOGRAPHY

Hemant Kowshik received his B.Tech in Electrical Engineering at the Indian Institute of Technology (IIT), Madras, in 2005. He received the M.S. degrees in Electrical and Computer Engineering and Mathematics from the University of Illinois at Urbana-Champaign in 2007 and 2010, respectively. He is currently a PhD candidate in the Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign.

His research has been on provable safety guarantees in automated vehicular systems and on distributed function computation in sensor networks. His current research interests include wireless networks, information theory, distributed systems and control. He is a student member of the IEEE.