Chapter 7
Networked Distributed Source Coding

Shizheng Li and Aditya Ramamoorthy

Abstract The data sensed by different sensors in a sensor network is typically correlated. A natural question is whether the data correlation can be exploited in innovative ways along with network information transfer techniques to design efficient and distributed schemes for the operation of such networks. This necessarily involves a coupling between the issues of compression and networked data transmission that have usually been considered separately. In this work we review the basics of classical distributed source coding and discuss some practical code design techniques for it. We argue that the network introduces several new dimensions to the problem of distributed source coding. The compression rates and the network information flow constrain each other in intricate ways. In particular, we show that network coding is often required for optimally combining distributed source coding and network information transfer and discuss the associated issues in detail. We also examine the problem of resource allocation in the context of distributed source coding over networks.

7.1 Introduction

There are various instances of problems where correlated sources need to be transmitted over networks, e.g., a large-scale sensor network deployed for temperature or humidity monitoring over a large field or for habitat monitoring in a jungle. This is an example of a network information transfer problem with correlated sources. A natural question is whether the data correlation can be exploited in innovative ways along with network information transfer techniques to design efficient and distributed schemes for the operation of such networks. This necessarily involves a coupling between the issues of compression and networked data transmission that have usually been considered separately (see Fig. 7.1 for an illustration).

S. Li (✉)
Iowa State University, Ames, IA, USA
e-mail: szli@iastate.edu

S. Nikoletseas, J.D.P. Rolim (eds.), *Theoretical Aspects of Distributed Computing in Sensor Networks,* Monographs in Theoretical Computer Science. An EATCS Series,
DOI 10.1007/978-3-642-14849-1_7,

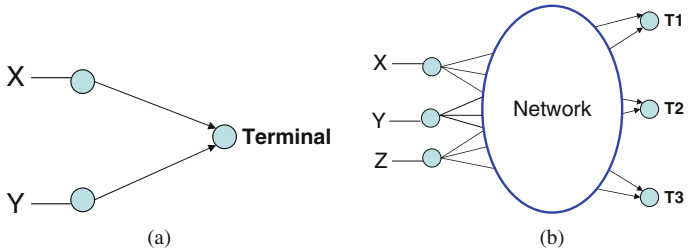

Fig. 7.1 (**a**) Classical Slepian–Wolf problem with sources X and Y with direct links to a terminal. (**b**) Practical scenario with multiple sources and terminals communicating over a network with link capacity and cost constraints. The joint problem of distributed source coding and network information transfer introduces various issues that are overviewed in this work

The correlation in a sensor network can be exploited in multiple ways. One can consider protocols where sensor nodes exchange information among themselves, compress the information, and then transmit the compressed bits to the terminal. At the other extreme, the sensors may operate independently. Intuitively, one would expect that the first scenario would be significantly better from a compression perspective. A surprising and groundbreaking result of Slepian and Wolf [1] shows that in fact under certain situations, the case in which the sensors act independently can be as efficient as the case in which the sensors do communicate with each other. The work of [1] introduced the idea of distributed source coding and demonstrated the existence of encoders and decoders that could leverage the correlation without needing explicit cooperation between the sources.

In this chapter we review various ideas in distributed source coding that are interesting within the context of sensor networks. We begin by an overview of the basic concepts and an outline of certain practical code constructions that have been the focus of much work recently. Next, we examine distributed source coding in a network context. The network introduces several dimensions to the problem of distributed source coding that do not exist in the classical case. It may be tempting to argue that one could simply find paths in the network that act as the direct links in the classical problem, assuming that the paths have enough capacity. However, such an approach is not optimal. The compression rates and the network information flow constrain each other in intricate ways. In particular, it turns out that network coding [2] is essential in certain cases for optimality. Interestingly enough, the flavor of results in this area depends upon the number of sources and terminals in the network. We survey these in a fair amount of detail in this chapter and examine the relationship between network coding and distributed source coding.

The issue of resource allocation is very important in the field of networking. For example, optimal routing of packets that maximizes some utility function of the network is a well-investigated issue in the field of networking [3]. Several techniques for solving these problems in a distributed manner have been studied in the literature [4]. In this chapter we discuss resource allocation problems in the context of transmitting correlated sources over a network. The main difference here is that

one needs to jointly allocate the rates and the flows in the network. In particular, the network capacity region and the feasible rate regions interact in non-trivial ways.

This chapter is organized as follows. We discuss the basics of distributed source coding in Sect. 7.2 and introduce the problem of networked distributed coding in Sect. 7.3. Sect. 7.4 presents the discussion for the case of networks with a single terminal and Sect. 7.5 considers the case of networks with multiple terminals.

7.2 Basics of Distributed Source Coding

A sensor network consists of various sensors that monitor some physical phenomenon, e.g., an agricultural sensor network may be deployed in a field for temperature or humidity monitoring. In this chapter we will use the terms sensor and source interchangeably. Furthermore, a sensor output at a given time is assumed to be a random variable. Hence, over time, the observations of a sensor can be treated as a vector of random variables. We assume that the source outputs a sequence of independent and identically distributed (i.i.d.) random variables. While this assumption may not hold in a strict sense, we will see that it serves to simplify our exposition. Many of the results discussed in this chapter also hold for the case of sources with memory. However, we will not discuss them here.

Formally, we denote n successive realizations of a source X by $X_1, X_2, \ldots, X_n$, such that their joint distribution $p(X_1, \ldots, X_n) = \Pi_{i=1}^{n} p(X_i)$. If there is another correlated source Y, the joint distribution $p(X_1, Y_1, X_2, Y_2, \ldots, X_n, Y_n) = \Pi_{i=1}^{n} p(X_i, Y_i)$, i.e., at a given time instant, the sources are correlated but across time they are independent.

In a sensor network, the main problem is to convey either the sensor readings or their functions (e.g., mean, variance) to a terminal. The transmission protocol needs to be efficient in terms of the number of bits transmitted. If the correlation between the sources is ignored and if the terminal needs to recover the source without any distortion, the compression rate should be at least the entropy [5, 6] of the source. For example, if there are two sources X and Y, this implies that the terminal needs to receive $H(X) + H(Y)$ bits per unit time for recovering both X and Y.

Clearly, if there is correlation across sensors, the overall bit rate required for transmission to the terminal can be reduced. This is certainly feasible if the sources communicate with each other. The famous result of Slepian and Wolf [1] shows that distributed source coding, where the sources do not communicate with each other, can be as efficient as the case in which the sources communicate with each other.

7.2.1 Slepian–Wolf Theorem

Consider two sources X and Y. Let R_X and R_Y denote the rates at which the sources operate. This means that the sources X and Y transmit R_X and R_Y bits per unit time to the terminal.

Theorem 1 (Slepian–Wolf Theorem [1]) *Consider memoryless correlated sources X and Y from finite-sized alphabets $\mathcal{X}, \mathcal{Y}$ respectively, with joint distribution $p(X, Y)$. Suppose that*

$$\begin{aligned} R_X &\geq H(X|Y) \\ R_Y &\geq H(Y|X) \\ R_X + R_Y &\geq H(X, Y) \end{aligned}$$

There exist encoding functions $f_1 : \mathcal{X}^n \to \{1, 2, \ldots, 2^{nR_X}\}$ at source X and $f_2 : \mathcal{Y}^n \to \{1, 2, \ldots, 2^{nR_Y}\}$ at the source Y and a decoding function $g : \{1, 2, \ldots, 2^{nR_X}\} \times \{1, 2, \ldots, 2^{nR_Y}\} \to \mathcal{X} \times \mathcal{Y}$ at the terminal, such that the terminal is able to recover the source sequences with vanishing error probability as n goes to infinity. Conversely, if R_X, R_Y do not satisfy those conditions, it is impossible to recover the sources with vanishing error probability.

The implication of the Slepian–Wolf theorem is rather surprising and profound. Intuitively, it is clear that there is no hope of compressing the sources to a rate of less than $H(X, Y)$ even if they communicate with each other. The Slepian–Wolf theorem shows that in fact one can do this even when the sources do not communicate with each other.

The achievability proof goes as follows. A length n X-sequence is compressed to a binary vector of length nR_X by encoding function f_1 that is chosen at random. This process is referred to as random binning [6] in the literature, as each sequence is assigned a bin whose index is determined by f_1. Similarly, f_2 returns the bin index of a Y-sequence. At the terminal, suppose bin indices (i, j) are received. The decoding function finds all the length n sequences $\mathbf{x}, \mathbf{y}$ such that $f_1(\mathbf{x}) = i$, $f_2(\mathbf{y}) = j$ and find the pair of sequences that are most likely to have been transmitted. When n is large, with high probability, such sequence pair is the actual transmitted sequence pair. In other words, the error probability is vanishing as n goes to infinity.

The rates satisfying conditions are called achievable rates and they form a region in the 2-D plane shown in Fig. 7.2.

The two corner points on the boundary are interesting. They correspond to a rate allocation $(R_X, R_Y) = (H(X), H(Y|X))$ or $(R_X, R_Y) = (H(X|Y), H(Y))$.

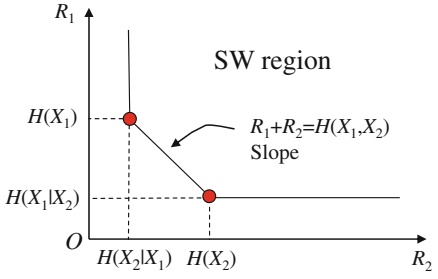

Fig. 7.2 Slepian–Wolf Region in the case of two sources X and Y

In order to achieve one of these points, say the first one, since $R_X = H(X)$, any lossless compression scheme can be used to compress $\mathbf{x}$. Then, $\mathbf{x}$ is used as *side information* to help decode $\mathbf{y}$ at the decoder. The rate of Y is $H(Y|X)$, i.e., the amount of uncertainty given X.

Code design in the case when side information is available at the decoder is called the *asymmetric* Slepian–Wolf coding problem [7]. Code design for achieving any general (non-corner) point is called the *symmetric* Slepian–Wolf coding problem. There are many practical code designs for both asymmetric coding and symmetric coding when we have only two sources. In general, asymmetric Slepian–Wolf coding is easier than the symmetric case, because of a certain equivalence with channel coding, that we will discuss shortly. We refer the reader to [7] and the references therein for detailed descriptions.

The theorem above is stated for two sources. In general, when there are N sources, we have a generalized Slepian–Wolf theorem [8]. Suppose the sources $X_1, X_2, \ldots, X_N$ are generating i.i.d. symbols according to the joint probability distribution $p(X_1, X_2, \ldots, X_N)$. Let R_i denote the rate for source X_i and S denote a nonempty subset of node indices: $S \subseteq \{1, 2, \ldots, N\}$. Let X_S denote the set of random variables $\{X_i : i \in S\}$. If the rate vector $(R_1, R_2, \ldots, R_N)$ satisfies

$$\sum_{i \in S} R_i \geq H(X_S|X_{S^c}) \text{ for all } S \neq \emptyset$$

the decoder is able to recover all sources error free (asymptotically). Conversely, if the rates do not satisfy the condition, lossless recovery is impossible. When there are multiple sources, practical code design is a challenging problem. Some coding schemes exist, e.g., [9–11], but they either suffer suboptimal rate or have strong assumptions on the correlation model.

7.2.2 Equivalence Between Slepian–Wolf Coding and Channel Coding

The proof of the Slepian–Wolf theorem is information theoretic in nature and the corresponding achievability scheme requires exponential (in n) complexity decoding in general. For the case of two sources, and asymmetric Slepian–Wolf coding, Wyner [12] discovered the relation between channel coding and Slepian–Wolf coding. Most existing work on Slepian–Wolf coding for two sources relies on Wyner's idea and exploits powerful channel codes such as Turbo codes and LDPC codes [13–19]. Here, we introduce the basic ideas for asymmetric Slepian–Wolf coding.

First we review the concepts of channel coding [20], especially on linear block codes. A (n, k) linear block code over a finite field $GF(q)$ maps each message of length k (i.e., a k-length vector $\in GF(q)$) to a codeword $\mathbf{c}$ of length n (i.e., an n-length vector $\in GF(q)$). The codeword is transmitted through a channel, which introduces an error $\mathbf{e}$. The receive vector is $\mathbf{r} = \mathbf{c} + \mathbf{e}$ (addition over $GF(q)$), where

e denotes the error vector. The decoder takes **r** as input and attempts to find the correct **c**. In classical coding theory, the errors are modeled according to their Hamming weight, i.e., the number of nonzero elements in **e**. An important design parameter of a code is the minimum Hamming distance d (the number of positions where two codewords take different values). A code with minimum distance d is able to correct up to $\lfloor (d-1)/2 \rfloor$ errors, i.e., as long as the Hamming weight of **e**, $wt(\mathbf{e}) \leq \lfloor (d-1)/2 \rfloor$, the decoder can find the error pattern **e** and the transmitted codeword **c**.

The parity check matrix of a linear block code is a $(n-k) \times n$ matrix H such that $\mathbf{c}H^T = 0$ (matrix multiplication over $GF(q)$) for every codeword **c**. A practical decoding algorithm for a linear block is called syndrome decoding. The decoder computes the syndrome of length $(n-k)$ $\mathbf{s} = \mathbf{r}H^T$. Since $\mathbf{r}H^T = \mathbf{c}H^T + \mathbf{e}H^T$, $\mathbf{s} = \mathbf{e}H^T$, implying that the syndrome only depends on the error pattern. It then attempts to find the **e** with the least weight. This can be done efficiently for specific classes of codes. For example, Berlekamp–Massey algorithm for BCH codes and Reed–Solomon codes [20] can be used to find the error pattern **e** from **s** as long as $wt(\mathbf{e}) \leq (d-1)/2$. Likewise, binary LDPC codes admit efficient decoding.

We now demonstrate that syndrome decoding can be applied to the asymmetric Slepian–Wolf coding problem. Assume that the source sequences **x**, **y** have length n and the correlation model is that the Hamming distance between them is no more than t, i.e., they differ at most t positions. Suppose **y** is available at the decoder. At source X, we transmit $\mathbf{x}H^T$ to the terminal. The terminal computes $\mathbf{y}H^T + \mathbf{x}H^T = (\mathbf{x}+\mathbf{y})H^T = \mathbf{e}H^T$, where $\mathbf{e} = \mathbf{x}+\mathbf{y}$ is the difference between **x** and **y**.[1] We know that **x** and **y** differ in at most t positions, so $wt(\mathbf{e}) \leq t$. The decoder is able to find **e** as long as the minimum distance of the channel code is at least $2t+1$ based on the discussions above. Once **e** is obtained, $\mathbf{x} = \mathbf{y} + \mathbf{e}$ can be easily computed. Thus, a length n vector **x** is compressed to a length $(n-k)$ vector $\mathbf{x}H^T$. Since the minimum distance of a code should satisfy Singleton bound $d \leq n-k+1$ [20], the length $n-k$ should be at least $2t$.

In order to establish a concrete relationship with Slepian–Wolf theorem, next we consider a probabilistic correlation model. Consider binary sources X and Y that are uniformly distributed. The correlation between them is that the probability that they are different is $p < 0.5$. In other words, each bit in the vector $\mathbf{e} = \mathbf{x}+\mathbf{y}$ is 1 with probability p and 0 with probability $1-p$. Then, $H(X|Y) = H_b(p)$,[2] and $H(X,Y) = 1 + H_b(p)$.

Now, consider the channel coding problem for the binary symmetric channel (BSC) with crossover probability p. The codeword **c** is transmitted and $\mathbf{r} = \mathbf{c}+\mathbf{e}$ is received and **e** is i.i.d. taking value 1 with probability p. The capacity of this channel is $1 - H_b(p)$ [6]. The receiver computes the syndrome $\mathbf{s} = \mathbf{r}H^T = \mathbf{e}H^T$. It can be shown that there exists an H and the decoding function $f_{\text{dec}}(\cdot)$ such that the code

[1] In this chapter, assume that the size of the finite field is a power of 2 so addition and subtraction are the same.

[2] $H_b(p)$ is the binary entropy function defined as $H_b(p) = -p\log_2 p - (1-p)\log_2(1-p)$.

rate $k/n \to 1 - H_b(p)$ as $n \to \infty$ and the decoding error can be made arbitrarily small [21]. Such a code is called a capacity-achieving code.

In an asymmetric Slepian–Wolf coding setting, suppose that the decoder knows $\mathbf{y}$. Let $R_Y = H(Y) = 1$ and apply any lossless entropy coding scheme [6], $\mathbf{y}$ can be recovered at the terminal. Take the parity check matrix of a capacity-achieving code H and the source X transmits $\mathbf{x}H^T$. The terminal finds the estimate of $\mathbf{x}$,

$$\hat{\mathbf{x}} = \mathbf{y} + f_{\text{dec}}(\mathbf{x}H^T + \mathbf{y}H^T)$$

The probability that $\hat{\mathbf{x}} \neq \mathbf{x}$ is arbitrary small. Note that the length of vector transmitted by source X is $n - k$, so the rate

$$R_X = (n-k)/n = 1 - k/n = H_b(p) = H(X|Y)$$

Thus, using a capacity-achieving channel code, we can achieve the corner point $(H(X|Y), H(Y))$ of the Slepian–Wolf region.

In practice, LDPC codes [22] come very close to the BSC capacity. The belief propagation algorithm (BPA) [22] acts as the decoding function $f_{\text{dec}}(\cdot)$. Note that in the channel coding setting, the belief propagation algorithm is designed to output a codeword $\mathbf{c}$ with 0 syndrome, whereas in the distributed source coding setting, the BPA needs to be modified so that it outputs a vector satisfying a given syndrome. More generally, even if the correlation model cannot be viewed as a binary symmetric channel, we can provide proper initialization to the BP algorithm according to the correlation model. Turbo codes can also be used to achieve compression via puncturing at the encoder; the extrinsic information exchange at the decoder exploits the correlation between the sources [23–25].

The equivalence in the asymmetric case does not carry over in a straightforward manner to the symmetric case. However, an approach called source splitting [26, 27] allows us to transform the symmetric Slepian–Wolf coding problem for N sources to an asymmetric (corner point) problem where there are $2N - 1$ sources.

7.2.3 Distributed Source Coding with a Fidelity Criterion

In the previous sections we considered the problem of lossless reconstruction. In many practical applications, we may allow a certain amount of distortion in the recovery process. In lossy multiterminal source coding, each source encodes its own data at a certain rate and transmits it to the terminal. The terminal tries to recover all the sources under a fidelity criterion. The fidelity is measured with respect to a distortion metric.

More specifically, the encoders observe source sequences $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ emitted by the sources $X_1, X_2, \ldots, X_N$ and encode them at rate $R_1, R_2, \ldots, R_N$ separately (with no communication between the encoders). Given distortion metrics, $\mathbf{D} = (D_1, D_2, \ldots, D_N)$ for each source, we hope to find the region $\mathcal{R}(\mathbf{D})$ of all rates $\mathbf{R} = (R_1, R_2, \ldots, R_N)$ that allow the decoder to reconstruct $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_N$ such

that the expected distortion between $\mathbf{x}_i$ and $\hat{\mathbf{x}}_i$ is less than D_i for all $i = 1, 2, \ldots, N$. However, the general region even in the case of very specific distortion metrics remains unknown.

The inner bound for a given problem refers to a set of rates that can be shown to be achievable. The outer bound refers to a set of rates that are not achievable under any strategy. Some inner/outer bounds for the general problem can be found in [28–30]. In most cases the inner and outer bounds do not meet, i.e., the exact region is unknown. A tighter outer bound was obtained recently [31] and some insights on the optimal encoders and decoders are given in [32]. The quadratic Gaussian case was considered in [33, 34], where the rate-distortion regions for several special cases are determined. Practical code design for multiterminal rate-distortion problems is discussed in [35, 36].

Next we discuss two special cases of multiterminal source coding problems, for which the rate-distortion regions are relatively well studied.

7.2.3.1 Wyner–Ziv Coding

Consider two correlated sources X and Y that follow joint distribution $p(X, Y)$. The source sequence $\mathbf{x}$ needs to be encoded without knowing $\mathbf{y}$ and transmitted to the decoder, at which side information $\mathbf{y}$ is available. Let the distortion between two n length sequences $\mathbf{x}$ and $\hat{\mathbf{x}}$ be measured as $\frac{1}{n}\sum_{i=1}^{n} d(x_i, \hat{x}_i)$, where d is a non-negative function. The rate-distortion function $R_{WZ}(D)$ gives the minimum required rate such that the expected distortion between the actual source sequence $\mathbf{x}$ and the decoder output $\hat{\mathbf{x}}$ is upper bounded by D, i.e., $E\left(\frac{1}{n}\sum_{i=1}^{n} d(x_i, \hat{x}_i)\right) \leq D$. Clearly, if $D = 0$, it is the special instance of Slepian–Wolf problem at corner point $(H(X|Y), H(Y))$. In general, the rate-distortion function was shown by Wyner and Ziv [37] to be

$$R_{\mathrm{WZ}}(D) = \min_{P_{U|X}(\cdot), f(\cdot): E(d(X, f(U,Y))) \leq D} I(X; U) - I(Y; U)$$

where U is an auxiliary random variable and is such that $U \rightarrow X \rightarrow Y$, i.e., U, X, Y form a Markov chain and the expectation is taken over the joint distribution of X, Y, U. The function f is the decoding function.

In the Slepian–Wolf setting (i.e., $D = 0$), we know that minimum required rate is $H(X|Y)$, whether or not Y is available at the X encoder. When $D > 0$, let us denote the rate required when Y is available at the source encoder as $R_{X|Y}(D)$. It can be shown that in some cases $R_{X|Y}(D) < R_{WZ}(D)$. In other words, we may lose efficiency when encoding correlated sources separately rather than jointly when $D > 0$. In the special case when the sources are correlated by $X = Y + Z$ where Y and Z are both Gaussian and Z is independent of Y, $R_{X|Y}(D) = R_{WZ}(D)$ [37]. In many other correlation models, the equality does not hold.

Practical coding schemes for the Wyner–Ziv problem based on nested codes [38, 39] are known. Nested lattice codes can be used in the quadratic Gaussian case and

can be shown to achieve the Wyner–Ziv bound. Other practical Wyner–Ziv code designs include trellis-based codes [13], nested coding followed by Slepian–Wolf coding [40], quantization followed by Slepian–Wolf coding [41, 42]. The discussion of these techniques is beyond the scope of this survey.

7.2.3.2 The CEO Problem

In the CEO problem [43], there is one source X and N encoders that do not observe the source directly. Instead, each encoder observes a corrupted version of X, denoted as Y_i, $i = 1, 2, \ldots, N$. The Y_i's are assumed to be conditionally independent given X. The encoder encodes $\mathbf{y}_i$ at rate R_i and such that the total encoding rate is $\sum_{i=1}^{N} R_i \leq R$. The decoder finds the $\hat{\mathbf{x}}$ (the estimate of $\mathbf{x}$,), based on the encoded codewords. The aim is to find the rate-distortion function $R(D)$, i.e., the minimum total encoding rate needed such that the expected distortion between $\mathbf{x}$ and $\hat{\mathbf{x}}$ is at most D. This is analogous to a situation when a Chief Executive (or Estimation) Officer obtains information from N agents and wants to estimate the source sequence $\mathbf{x}$ that he or she is interested in. In a sensor network application, we can think of the data fusion center acting as the CEO and the sensors act as the agents. The problem formulation takes into account the noise in the sensing procedure. The original paper [43] determined the asymptotic behavior of the error frequency when $R \to \infty$ for discrete memoryless source. The quadratic Gaussian case of the CEO problem, where X is Gaussian and the observation noises $Y_i - X$ are independently Gaussian distributed, is studied in [44–46] and the rate-distortion function is determined in [45, 46].

7.3 Networked Distributed Source Coding: An Introduction

In the previous sections we have discussed the classical Slepian–Wolf result and its lossy variants. Note that so far we have assumed that there is a direct noiseless link between the sources and the terminal. This is a useful simple case to analyze and captures the core of the problem as far as the basic concept of distributed source coding is concerned. However, in a practical sensor network we expect that the sensors will be communicating with the terminal over a network, possibly with the help of various relay nodes. Therefore, it is natural to investigate whether the process of information transmission over the network influences the compression and vice versa. Our network model represents a wireline network or a wireless network with medium access control (MAC) protocols that make the channels look independent (we discuss the network model in more detail later). In this part of the chapter, we overview relatively recent work that has contributed toward our understanding of this field.

The problem of networked distributed source coding differs from the classical problem in the following ways.

- *Suboptimality of separation between distributed source coding and network information transfer.*
  Note that the problem of distributed source coding over networks would be a straightforward extension of the classical Slepian–Wolf problem if one could essentially "simulate" the presence of direct links between the sources and the terminal. Indeed, one could encode the sources using a classical Slepian–Wolf code and simply "flow" the encoded bits over the appropriate paths. This would amount to separating the tasks of distributed source code design and the problem of network information transfer. It turns out that such a strategy is suboptimal in general.
- *Issues of optimal resource allocation over the network.*
  The network introduces several issues with respect to the allocation of rates and flows such that they are in some sense "optimal" for the operation of a network. For example, in sensor networks, the problem of deciding the appropriate paths over which the data needs to flow for minimum energy or maximum lifetime [47] is of interest. In the context of correlated sources, these issues become more complicated since one needs to jointly optimize the rates and the flows.

Our model of a network is a directed graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. There is a set of source nodes $S \subset V$ that observes the sources and a set of terminals $T \subset V$ that needs to reconstruct the sources. An edge (v_1, v_2) is a communication channel which allows information transmission from v_1 to v_2. The channel can be noisy or deterministic (but typically capacity constrained). The different channels in the network are in general dependent, e.g., in a wireless network, broadcast, and interference induces dependence between different channels. However, characterizing the capacity region in such scenarios, even with independent messages, has proved to be a difficult task [6]. In fact, in many practical situations, protocols such as time division multiple access-TDMA, frequency division multiple access-FDMA are used to provide the semblance of independent channels. In a wireline network, the channels are typically independent. In the discussion in the sequel, we will mostly work under the assumption that the channels are independent. It turns out that the results in this area depend critically on the number of terminals in network. Accordingly, we divide the discussion into two different sections. In Sect. 7.4 we review the results for the single terminal case and in Sect. 7.5 we review the corresponding results for multiple terminals.

7.4 Networked Distributed Source Coding: Single Terminal

In networks with a single terminal, under the assumption that the channels are independent, Han [48] gave necessary and sufficient conditions for a network to be able to transmit the correlated sources to the sink. A simple achievable transmission scheme was proposed and its optimality was proved. Barros et al.[49] obtained

the same result under a more general encoding model, where the form of joint source/channel coding can be arbitrary and the coding can be across multiple blocks. The achievability proof is almost the same as [48] and the converse is proved in a different manner and is stronger because of the more general coding model.

Suppose that there are $N+1$ nodes $v_0, v_1, \ldots, v_N$ in the network observing sources $X_0, X_1, \ldots, X_N$. The graph $G(V, E)$ is complete and each edge (v_i, v_j) is a discrete memoryless channel with capacity C_{ij}. Note that the source entropy could be zero and the capacity could also be zero, so realistic networks can easily fit into this general framework. Node v_0 is the sink that wants to reconstruct the sources $X_1, \ldots, X_N$.

The proposed transmission scheme is very simple and intuitive. Apply good channel codes to each channel so that we can model every edge (v_i, v_j) as a noiseless link with capacity C_{ij}. Each node performs Slepian–Wolf coding at rate R_i. Next, the Slepian–Wolf coded bits need to be routed to the sink v_0. Knowing the rates at each source node, we can find a feasible flow that supports rate R_i at source node v_i and terminates at sink node v_0 as follows.

Add a virtual supersource s^* and introduce an edge (s^*, v_i) with capacity $C_{s^*i} = R_i$ for $i = 1, \ldots, N$. Then compute the max-flow between s^* and v_0 [50]. This returns a flow assignment on each edge. The Slepian–Wolf coded bits are routed according to the flow assignment to v_0.

The node v_0 receives all Slepian–Wolf coded bits and jointly decodes all the sources $X_1, X_2, \ldots, X_N$. In order to reconstruct the sources, the rate vector $(R_1, \ldots, R_N)$ needs to be in the Slepian–Wolf region, i.e., for any nonempty subset of $\{0, \ldots, N\}$, S, such that $0 \in S^c$ (since X_0 is available at v_0 as side information and is not encoded)

$$\sum_{i \in S} R_i \geq H(X_S | X_{S^c}) \tag{7.1}$$

In order to successfully find the flow of value $\sum_{i=1}^{N} R_i$ from s^* to v_0, we need the capacity of any cut separating s^* and v_0 to be greater than $\sum_{i=1}^{N} R_i$. Note that a cut separates the source nodes into S and S^c, where $S \subseteq \{0, \ldots, N\}, 0 \in S^c$ but s^* does not connect to v_0, its capacity is $\sum_{j \in S^c \setminus \{0\}} C_{s^* j} + \sum_{i \in S, j \in S^c} C_{ij} = \sum_{j \in S^c \setminus \{0\}} R_j + \sum_{i \in S, j \in S^c} C_{ij}$. Thus, as long as

$$\sum_{i \in S} R_i \leq \sum_{i \in S, j \in S^c} C_{ij} \tag{7.2}$$

for all nonempty subset S of $\{0, \ldots, N\}$ such that $0 \in S^c$, the flow exists. This is illustrated in Fig. 7.3. Moreover, if

$$H(X_S | X_{S^c}) \leq \sum_{i \in S, j \in S^c} C_{ij} \tag{7.3}$$

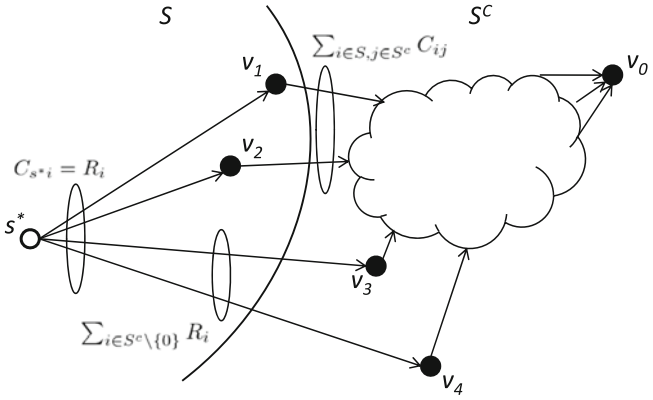

Fig. 7.3 Illustration of the sufficient condition for routing Slepian–Wolf coded bits to the terminal. s^* is the supersource. The cut of interest contains v_1, v_2 in S and v_3, v_4 in S^c. The cut capacity is $\sum_{j\in S^c\setminus\{0\}} R_j + \sum_{i\in S, j\in S^c} C_{ij}$, and it should be no less than $\sum_{i=1}^{N} R_i$. Thus, $\sum_{i\in S, j\in S^c} C_{ij} \geq \sum_{i\in S} R_i$

there exists a rate allocation satisfying (7.1) and (7.2) [48]. Therefore, (7.3) is a sufficient condition for the single sink data collection with Slepian–Wolf coding.

Conversely, it is proved that the above condition is the necessary condition for successful transmission under any joint coding scheme, i.e., if the capacity does not satisfy this condition, the sink cannot recover the sources losslessly, under any kind of coding scheme. Note that the proposed achievability scheme separates source coding, channel coding, and routing. The converse part implies that it is optimal to separately perform channel coding, distributed source coding, and treat the Slepian–Wolf coded bits as commodities and route to the terminal. The main theorem in [49] can also be viewed as a general source–channel separation theorem for networks with one terminal, with independent channels. It implies that the source coding, routing, and channel coding can be put into different layers of the protocol stack separately.

We emphasize, however, that such a separation does not hold in general, i.e., when there are more terminals. As we shall see in Sect. 7.5, even when the channels are independent, if we have more terminals, the compression rates and the network flows are closely coupled.

7.4.1 Optimal Rate and Flow Allocation

From the discussion in previous sections, it is clear that distributed source coding can compress the data effectively. In this section, we discuss resource allocation problems for networked distributed source coding.

A natural resource allocation problem is to determine the rate at which each source should be encoded, and the corresponding flows such that some network

metric is optimized. For simplicity, we first consider the case when there are direct channels between the sources and the sink.

7.4.1.1 Direct Source–Sink Channels

Suppose the sources communicate to the sink directly. We consider two metrics as follows.

1. Sum rate minimization: In this case we consider noiseless source–sink channels and seek to find a feasible rate vector that minimizes $\sum_{i=1}^{N} R_i$.
2. Sum power minimization: Here we assume orthogonal additive white Gaussian noise (AWGN) channels between the sources and the sink and seek to minimize the total power min $\sum_{i=1}^{N} P_i$ (where P_i is the power of the ith source), expended in ensuring that the sources can be reconstructed at the terminal.

For the noisy channel case, the source nodes first use Slepian–Wolf codes to encode the sources. As long as each rate is less than the channel capacity the sources can be recovered losslessly at the terminal (assuming capacity-achieving codes are used). The capacity of the channel between node i and the sink with transmission power P_i and channel gain γ_i is $C_i(P_i) \equiv \log(1 + \gamma_i P_i)$, where the noise power is normalized to one and channel gains are constants that are known to the terminal. Thus, the rate R_i should satisfy $R_i \leq C_i(P_i)$. It is easy to see at the optimum, the sensor node should transmit at the capacity, i.e., $R_i^* = C_i\left(P_i^*\right)$. Thus, the power assignment is given by the inverse function of C_i which we denote by $Q_i(R_i)$, i.e., $P_i^* = Q_i\left(R_i^*\right) = \left(2^{R_i^*} - 1\right)/\gamma_i$. Once we know the optimal rate assignment R_i^* we know the power assignment P_i^* and vice versa. Therefore, the objective function of the sum power minimization problem can also be written as

$$\min \sum_{i=1}^{N} \left(2_i^R - 1\right)/\gamma_i$$

For both problems, if N-dimensional Slepian–Wolf codes are used, the rates should be in the N-dimensional Slepian–Wolf region, which is denoted by SW_N. Then, the sum rate minimization problem can be written as

$$\min_{R_1,\ldots,R_N} \sum_{i=1}^{N} R_i$$
$$\text{subject to } (R_1, \ldots, R_N) \in SW_N$$

The solution to this problem is trivial, i.e., any point at the boundary of the N-dimensional Slepian–Wolf region is the optimal solution. In the sum power minimization problem, besides Slepian–Wolf region constraint, we also add peak power constraints for the transmission power of each sensor node, taking into account the fact that every sensor has limited transmission power in a wireless sensor network. Then, the problem is a convex optimization problem:

$$\min_{R_1,\ldots,R_N} \sum_{i=1}^{N} P_i = \sum_{i=1}^{N} (2^{R_i} - 1)/\gamma_i$$
$$\text{subject to } (2^{R_i} - 1)/\gamma_i \leq P_{\max}, \forall i$$
$$(R_1, \ldots, R_N) \in SW_N$$

This problem can be efficiently solved by, for example, interior-point methods [51].

In practice we do need to impose additional restrictions on the set of feasible rate vectors. This is primarily because the problem of practical code design for the N-dimensional Slepian–Wolf region remains open. It is fair to say that at present, we only know how to design Slepian–Wolf codes for two sources. Thus, it makes sense to impose "pairwise" constraints on the rate vectors, so that two sources can be decoded together. Given the state-of-the-art code designs for two sources case, we could perform encoding and decoding in a pairwise fashion. Before the transmission starts, we determine the source pairs that are jointly decoded together each time and determine the rates of the sources and the corresponding codes. During the transmission, the sources encode the message separately (without communication with other sources) using the preassigned code and the sink performs joint decoding for two nodes each time according to the preassigned combinations. We call this *pairwise distributed source coding*, which is simple and practical. The resource allocation problem is to determine the optimal pairing combinations and the rates for the sensors such that the sum rate or the sum power is minimized. This problem was first considered and solved using the notion of matching in undirected graph in [52]. Later, an improved solution using the notion of minimum weight arborescences and matching forests was proposed in [53] that we shall discuss below.

First, we consider the sum rate minimization problem. Note that any point on the slope of the Slepian–Wolf boundary achieves the minimum sum rate of two sources. Thus, for a pair of sources that will be decoded together, simply choosing the corner point as a rate allocation achieves minimum sum rate. Also note that a decoded source can be used as side information to help decode other sources at the terminal so that the rate of other sources being helped can be as low as the conditional entropy given the decoded source. Since we consider pairwise distributed source coding here and each time only two sources are involved in the decoding, we do not use more than one decoded sources as helper. We say a rate assignment has the pairwise property if it allows the terminal decode the sources in a pairwise fashion. Specifically, the rate assignment is said to satisfy the pairwise property if for each source $X_i, i = 1, 2, \ldots, N$, there exists an ordered sequence of sources $(X_{i_1}, X_{i_2}, \ldots, X_{i_k})$ such that

$$R_{i_1} \geq H(X_{i_1}) \tag{7.4}$$
$$R_{i_j} \geq H(X_{i_j}|X_{i_{j-1}}), \quad \text{for } 2 \leq j \leq k, \text{ and} \tag{7.5}$$
$$R_i \geq H(X_i|X_{i_k}) \tag{7.6}$$

Such a rate assignment allows the possibility that each source can be reconstructed at the decoder by solving a sequence of decoding operations at the SW corner points,

e.g., for decoding source X_i one can use X_{i_1} (since $R_{i_1} \geq H(X_{i_1})$), then decode X_{i_2} using the knowledge of X_{i_1}. Continuing in this manner finally X_i can be decoded. We hope to find rate assignment with pairwise property and with minimum sum rate. Clearly, the optimal rate assignment satisfies conditions (7.4), (7.5), and (7.6) with equality. It is easy to see the sequential decoding procedure of a rate assignment with pairwise property that can be expressed on a tree. The nodes at the higher layer are decoded first and used as side information to help decode the nodes at the lower layer. If we assign edge weights to be entropies and conditional entropies, the weight of the tree is the sum rate. Therefore, this inspires us to find a tree with minimum weight on a proper defined graph.

Now we formally describe our approach. Construct a directed graph $G = (V, E)$ as follows. The node set V consists of N regular nodes: $1, 2, \ldots, N$ and N starred nodes $1^*, 2^*, \ldots, N^*$. The edge set E consists of edges $(i^* \to i)$ with weight $H(X_i)$ for all $i = 1, 2, \ldots, N$, and edges $(i \to j)$ with weight $H(X_j|X_i)$ for all $i, j = 1, 2, \ldots, N$. An example of G is shown in the left figure of Fig. 7.4. Define a subgraph G_{i^*} of G as a graph obtained from G by deleting all starred nodes except i^* and all edges of the form $(j^* \to j)$ for $j \neq i$. For each i, find a minimum weight directed spanning tree[3] on G_{i^*}. This tree gives a rate allocation: $R_i = H(X_i)$, $R_j = H(X_j|X_{\text{inc}(j)})$, where $inc(j)$ is the node such that edge $(inc(j) \to j)$ belongs to the tree. Each subgraph G_{i^*} gives a rate allocation by a minimum weight

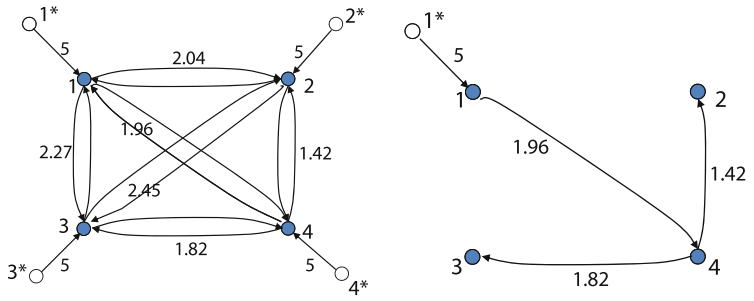

Fig. 7.4 An example of the rate allocation algorithm. The *left* figure shows the graph G. The edge weights on the edges from node i^* to node i are individual entropies and the edge weights on the edges between regular nodes are conditional entropies. In this example, the individual entropies are the same. Thus, $H(X_i|X_j) = H(X_j|X_i)$ and we only label one number between regular nodes i and j. The *right* figure shows the minimum weight directed spanning tree found on G_{1^*} (**a**) The graph G; (**b**) The minimum weight directed spanning tree found on G_{1^*}

[3] A directed spanning tree (also called arborescence) of a directed graph $G = (V, A)$ rooted at vertex $r \in V$ is a subgraph T of G such that it is a spanning tree if the orientation of the edges is ignored and there is a path from r to all $v \in V$ when the direction of edges is taken into account. The minimum weight directed spanning tree can be found by a greedy algorithm in polynomial time [54].

directed spanning tree and the one with minimum weight gives the final optimal rate allocation of the network. Note that if each source has the same entropy, the weights of minimum weight directed spanning trees of G_{i*} are the same for each i, so we only need to pick up an arbitrary subgraph G_{i*} and find the assignment on it. Clearly, the resulting rate assignment has the pairwise property and is optimal. In the example in Fig. 7.4, each source has the same entropy and the minimum weight directed spanning tree rooted at node 1^* is shown in the right figure. The optimal rate allocation is $R_1 = H(X_1)$, $R_4 = H(X_4|X_1)$, $R_2 = H(X_2|X_4)$, and $R_3 = H(X_3|X_4)$. The corresponding decoding procedure is that first decode source X_1, and use X_1 as side information to help decode X_4. Then, X_4 is used as side information to help decode X_2 and X_3.

Next, we show some simulation results. Consider a wireless sensor network example in a square area where the coordinates of the sensors are randomly chosen and uniformly distributed in [0, 1]. The sources are assumed to be jointly Gaussian distributed such that each source has zero mean and unit variance (this model was also used in [55]). The parameter c indicates the spatial correlation in the data. A lower value of c indicates higher correlation. The individual entropy of each source is $H_1 = \frac{1}{2}\log(2\pi e\sigma^2) = 2.05$. In Fig. 7.5, we plot the normalized sum rate found by minimum weight spanning tree (MST) $R_{s0} \equiv \sum_{i=1}^{N} R_i/H_1$ vs. the number of sensors n. If no distributed source coding is used, i.e., the nodes transmit data individually to the sink, $R_i = H_1$ and $R_{s0} = N$. Clearly, by pairwise distributed source coding, the sum rate is reduced. We also plotted the optimal normalized sum rate when N-dimensional Slepian–Wolf code is used $H(X_1, \ldots, H_N)/H_1$ in the figure. It is interesting to note that even though we are doing pairwise distributed source coding, our sum rate is quite close to the theoretical limit which is achieved by N-dimensional distributed source coding.

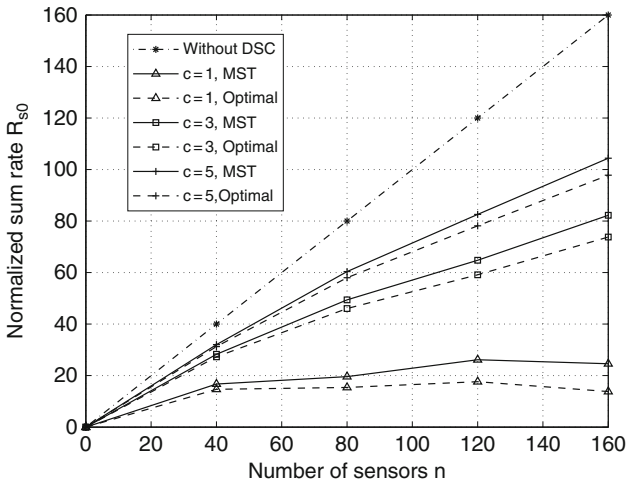

Fig. 7.5 Normalized sum rate vs. number of sensors

Now we consider the sum power minimization problem. Note that for a pair of sources that will be decoded together, the optimal rate allocation that minimizes the sum power of the pair is no longer a corner point but rather a particular point on the slope (which can be found by solving a simple optimization problem). For a node pair i and j, denote the optimal power allocation as $P_{ij}^*(i)$, $P_{ij}^*(j)$. We cannot simply choose the corner points and perform asymmetric Slepian–Wolf coding. We want some source pairs working at corner points while some others working at the optimal point on the slope of the 2-D SW region. Taking this into account, we generalize the concept of pairwise property. Recall that previously, under a rate assignment with pairwise property, the first source in a sequence is encoded at the rate of its entropy. Now we allow the first source in a decoding sequence to be paired with another node and encoded at the rate on the slope of the 2-D Slepian–Wolf region. The appropriate structure for finding the optimal resource allocation turns out to be one that combines the directed spanning tree and the matching. Such a structure is the *matching forest* first introduced in the work of Giles [56]. In fact, we are interested in a specific form of matching forest called *strict matching forest* (SMF). For the formal definitions, we refer the reader to [53]. Roughly speaking, a strict matching forest is a subgraph of a mixed graph[4] that connects every node only once. The SMF plays a role similar to the spanning tree in the sum rate minimization problem. The sequential decoding procedure of a rate assignment with generalized pairwise property can be expressed on a SMF. The node pairs connecting with undirected edges work at the slope of the Slepian–Wolf region and a symmetric coding scheme is used for them. The nodes that are connected with directed edge work at the corner point of the Slepian–Wolf region and the tail (origin) of a directed edge is used as side information to help decode the head (destination) of the edge. If we assign edge weights to be transmission powers, the weight of the SMF is the total transmission power.

Now we formally describe our approach. Construct a mixed graph $G = (V, E, A)$ as follows. The node set V consists of N regular nodes: $1, 2, \ldots, N$ and N starred nodes $1^*, 2^*, \ldots, N^*$. Recall that $Q_i(R)$ is the power consumed in transmission at rate R. For each $i = 1, 2, \ldots, N$, if $Q_i(H(X_i)) \leq P_{\max}$, add edge $(i^* \to i)$ with weight $Q_i(H(X_i))$. For each $i, j = 1, 2, \ldots, N$, if $Q_i(H(X_i|X_j)) \leq P_{\max}$, add edge $(j \to i)$ with weight $Q_i(H(X_i|X_j))$. For each pair i and j, if the optimal power allocation $P_{ij}^*(i)$, $P_{ij}^*(j)$ that minimizes the sum power of the pair of nodes exists, add undirected edge (i, j) with weight $P_{ij}^*(i) + P_{ij}^*(j)$. Then, find the minimum weight SMF on G, which gives the rate/power assignment with the generalized pairwise property and minimum sum power. It is shown in [53] that the problem of finding minimum weight SMF can be transformed and solved in polynomial time [57]. From the simulations we observe that in most cases, the optimal allocation is such that only one pair works on the slope and all other sources work at the corner points.

[4] "Mixed" graph refers to a graph with directed edges and undirected edges.

7.4.1.2 General Multihop Communication Between Sources and Sink

The resource allocation problem in a network with general topology and relay nodes was first considered by Han [48] and a similar formulation is given in [49]. We reformulate the problem as follows.

The network is given by a directed graph $G = (V, E, C)$, where $C = \{C_{ij} : (i, j) \in E\}$ is the capacity of each edge. Edge (i, j) is also associated with a weight w_{ij}. The cost of a flow of value z_{ij} going through the edge can be written as $F(z_{ij})w_{ij}$, where $F(\cdot)$ is a non-negative, increasing function. Then, the optimization problem can be written as

$$\begin{aligned} &\min \sum_{(i,j)\in E} F(z_{ij})w_{ij} \\ \text{s.t.}\quad &0 \le z_{ij} \le C_{ij}, \forall (i, j) \in E \quad \text{(capacity constraint)} \\ &\sum_{j|(i,j)\in E} z_{ij} - \sum_{j|(j,i)\in E} z_{ji} = \sigma_i, \forall i \in V \quad \text{(flow balance constraint)} \\ &(R_1, R_2, \ldots, R_N) \in SW_N \quad \text{(Slepian–Wolf constraint)} \end{aligned}$$

where $\sigma_i = R_l$ if i is the lth source node, $\sigma_i = -\sum_{i=1}^{N} R_i$ if i is the sink and otherwise, $\sigma_i = 0$.

For simplicity, we can consider linear cost $F(z_{ij}) = z_{ij}$. Then, the above optimization is a linear program. If $F(\cdot)$ is a convex function, it is a convex optimization problem.

If there is no capacity constraint, the solution of the problem has a simple form and interpretation [58]. The basic idea is that in the absence of capacity constraints, there is no need to split the flow across different edges. Once a route (path) from a given source to the sink with minimum cost is found, the source simply routes all the data through that path. For example, suppose that the minimum cost path for source X_l is $\mathcal{P}^l$. Then for all edges (i, j) belonging to $\mathcal{P}^l$, we set $z_{ij} = R_l$. In this case, the cost of transmitting the data from X_l to the sink is $\sum_{e\in\mathcal{P}^l} F(R_l)w_e$. Thus, the overall cost function becomes

$$\min_{\{R_l,d_l\},l=1,2,\ldots,N} \sum_{l=1}^{N} F(R_l)d_l$$

where d_l is the total weight of path $\mathcal{P}^l$, i.e., $d_l = \sum_{e\in\mathcal{P}^l} w_e$. Solving this problem involves finding the optimal paths $\mathcal{P}^l, l = 1, 2, \ldots, N$ and finding the optimal rate allocation $R_l, l = 1, 2, \ldots, N$. It is shown in [58] that these two steps are separable, i.e., one can first find the optimal paths $\mathcal{P}^{l*}$ and then find the optimal rate allocation based on the optimal paths $\mathcal{P}^{l*}$. This separation holds even if the function $F(\cdot)$ is nonlinear. It is easy to see the optimal path $\mathcal{P}^{l*}$ is the path with minimum total weight. Then, the optimal routing structure is the shortest path tree rooted at the sink, which can be found effectively and in a distributed manner. Now, suppose that the cost function F is such that $F(R_l) = R_l$. In this case, the problem becomes

$$\min \sum_{l=1}^{N} R_l d_{\mathrm{SPT}}(l,t)$$
$$s.t.(R_1, R_2, \ldots, R_N) \in SW_N$$

where $d_{\mathrm{SPT}}(l,t)$ (known as a constant) is the weight of the path from source l to terminal t on the shortest path tree. This is a linear programming problem with number of constraints exponentially with N. However, because of the contra-polymatroidal structure of the Slepian–Wolf region [59], the solution can be found in a easy greedy manner as follows [58].

1. Find a permutation π such that $d_{\mathrm{SPT}}(\pi(1),t) \geq d_{\mathrm{SPT}}(\pi(2),t) \geq \cdots \geq d_{\mathrm{SPT}}(\pi(N),t)$.
2. The optimal rate allocations is given by

$$\begin{aligned} R_{\pi(1)} &= H(X_{\{\pi(1)\}}|X_{\{\pi(2),\pi(3),\ldots,\pi(N)\}}) \\ R_{\pi(2)} &= H(X_{\{\pi(2)\}}|X_{\{\pi(3),\pi(4),\ldots,\pi(N)\}}) \\ &\vdots \\ R_{\pi(N)} &= H(X_{\{\pi(N)\}}) \end{aligned} \tag{7.7}$$

If the function $F(\cdot)$ is not linear but convex, the problem can still be solved by convex optimization [51] but the simple greedy algorithm may not work here.

From the previous discussion, we know that Slepian–Wolf coding along with routing is the optimal solution for the single sink data collection problem. In fact, it is shown in [60] that in terms of the cost under convex and increasing cost functions, Slepian–Wolf coding plus routing is still the optimal solution even if the wireless network broadcast advantage is considered. Interestingly, because the N-dimensional ($N > 2$) Slepian–Wolf code design problem remains open, [58, 60] also consider several schemes that do not use distributed source coding but allow some cooperation among the sources. Clearly, the communication between the sources will increase the cost. The cost of the Hierarchical Difference Broadcasting in [60] has been shown to have the same order compared to Slepian–Wolf coding. However, the explicit communication scheme in [58] will have significant loss compared to Slepian–Wolf under some conditions.

7.5 Networked Distributed Source Coding: Multiple Terminals

We now consider the variant of the problem when there are multiple terminals that want to reconstruct all the sources. This is called multicast. As before, one could attempt to treat this scenario as a generalization of the single terminal case. For example, one could divide the capacity of each edge into various parts, with each part responsible for conveying the bits to a specific terminal. However, on closer inspection it is possible to realize that such a strategy will in general be suboptimal.

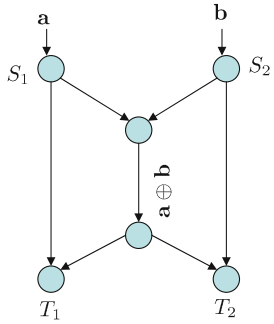

Fig. 7.6 A network with unit capacity edges and sources S_1 and S_2 and terminals T_1 and T_2. Each terminal wants to simultaneously recover the pair of bits (a, b). Under routing this is impossible. However, by computing and sending $a \oplus b$ along the bottleneck edge, we can achieve simultaneous recovery

To see this consider Fig. 7.6 that depicts the celebrated butterfly network of network coding [2]. In this example, each edge has unit capacity. Each terminal seeks to obtain the bits from both the sources. It is easy to see that if we only allow routing in the network, it is impossible to support this since the edge in the middle is a bottleneck. However, if we allow coding at the intermediate nodes and transmit the XOR of the two bits, then both terminals can obtain the two bits by simple XOR decoding. This example shows the potential gain of coding when there are multiple terminals. Of course, in this case the sources are independent. However, since independence is a degenerate case of correlation, one expects that similar conclusions will hold in the correlated case. As we shall see this is indeed the case. Furthermore, several interesting conclusions about the relationship of the coding rates and flow structures can be found.

7.5.1 A network Coding Primer

Traditionally, the intermediate nodes (routers) in the network only copy and forward packets. In a single source single sink unicast connection, routing achieves maximum flow, which equals to the minimum cut between the source and the terminal [61]. However, in a multicast scenario, pure routing may not achieve maximum flow as shown above. But it has been shown in [2] that network coding achieves max-flow min-cut upper bound in multicast. Next, we shall mathematically describe this result.

As usual, we model a network as a graph $G = (V, E, C)$, where $C = \{c_e : e \in E\}$ is the capacity of the edges, where c_e is the capacity on edge e. The seminal work on network coding [2] finds a tight capacity characterization for the single source, multiple terminals multicast problem.

Theorem 2 *Consider a network $G = (V, E, C)$ with source s and L terminals: $t_1, \ldots, t_L$. Suppose that the source node observes a source X, such that its entropy $H(X) = R$. Each terminal can recover X if and only if*

$$min - cut(s, t_l) \geq R, \forall l \in \{1, \ldots, L\} \tag{7.8}$$

The work of [62, 63] shows that the multicast can be supported even with linear codes. Basically, each intermediate node transmits linear combinations of the packets, where a packet is treated as a vector over a finite field. It is possible to show that in this case at each terminal, the received packets are the source messages multiplied by a transfer matrix. By inverting the transfer matrix, the terminal is able to recover the source packets. Moreover, as long as the coefficients of the linear combinations are chosen randomly from a large field and the min-cut between the source and each destination is greater than the source rate, the probability that the transfer matrix is invertible is very high [64]. This fact provides a simple distributed scheme for network coding-based multicast. A practical multicast protocol based on these ideas was developed in [65].

7.5.2 Multicasting Correlated Sources over a Network

In the discussion in the previous section, we only considered multicast with single source. The multiple independent sources case can be reduced to single source case [63], by introducing a virtual supersource that is connected to each source node.

In this section we consider the problem of multicasting correlated sources over a network. We begin by stating the main result. Consider a network $G = (V, E, C)$, with terminals $t_i, i = 1, \ldots, L$ and a set of source node $\mathcal{S} \subset V$. Without loss of generality, we assume a numbering so that these are the first $|\mathcal{S}|$ sources in V. Furthermore, source node i observes a source X_i. The communication requirement for multicasting correlated sources is that each terminal $t_i, i = 1, \ldots, L$ needs to recover all sources $(X_1, \ldots, X_{|\mathcal{S}|})$ losslessly. The admissible rate region is given by [66, 67].

Theorem 3 *The correlated sources* $(X_1, \ldots, X_{|\mathcal{S}|})$ *can be multicast to the terminals* $t_1, \ldots, t_L$ *if and only if*

$$H(X_S|X_{S^c}) \leq \min -\text{cut}(S, t_i) \ \ \forall S \subseteq \mathcal{S} \tag{7.9}$$

An achievability scheme based on random linear network coding for this result was proposed in [64]. Alternative proofs are provided in [66, 67]. We briefly overview the achievability scheme in [64] now.

Consider two correlated sources generating binary vectors $\mathbf{x}_1, \mathbf{x}_2$ of length r_1 and r_2 according to joint probability distribution $Q(\mathbf{x}_1, \mathbf{x}_2)$ each time. After n time slots, the source messages are $\mathbf{x}_1^n$ and $\mathbf{x}_2^n$ of length nr_1 and nr_2, respectively. We assume that c_e is rational for all e. Furthermore assume that n is large enough so that $n \times c_e$ is an integer for all e. This implies that when considered over a block of n time slots we communicate nc_e bits over edge e.

Simply perform random linear coding at each node over a blocklength of n including the source nodes and intermediate nodes, i.e., the bits on an outgoing edge

of node v are a linear combination of bits on incoming edges to node v, where the coefficients are chosen uniformly randomly from $GF(2)$. Each terminal t receives a vector $\mathbf{z}_t^n$ of length $n|\Gamma_i(t)|$, where $|\Gamma_i(t)|$ is the number of incoming edges to terminal t (before the edges are copied). Using the algebraic network coding framework [63], we can conclude that

$$\mathbf{z}_t^n = \left[\mathbf{x}_1^n \ \mathbf{x}_2^n\right] M_t \tag{7.10}$$

where M_t is a $(nr_1 + nr_2) \times n|\Gamma_i(t)|$ transfer matrix from the sources to terminal t. When sources are independent, M_t needs to have full rank so that by inversion we can recover the sources. In the case of correlated sources , M_t need not have full rank because we can take advantage of the correlation between the sources to find $\mathbf{x}_1^n$ and $\mathbf{x}_2^n$.

The decoding is done as follows. Find all possible $\left[\mathbf{x}_1^n \ \mathbf{x}_2^n\right]$ satisfying (7.10). Note that $\mathbf{x}_1^n, \mathbf{x}_2^n$ can be viewed as a length n vector of elements from $GF(2^{r_1})$ and $GF(2^{r_2})$, respectively.[5] Let $\mathbf{x}_{1i}, \mathbf{x}_{2i}$ denote the ith element and $i = 1, 2, \ldots, n$. The number of appearances of $(\mathbf{a}, \mathbf{b}), \mathbf{a} \in GF(2^{r_1}), \mathbf{b} \in GF(2^{r_2})$ is defined to be $N(\mathbf{a}, \mathbf{b}) = |\{i : \mathbf{x}_{1i} = \mathbf{a}, \mathbf{x}_{2i} = \mathbf{b}\}|$. The empirical joint distribution (histogram) $P_{\mathbf{x}_1^n, \mathbf{x}_2^n}$ is $P_{\mathbf{x}_1^n, \mathbf{x}_2^n}(\mathbf{a}, \mathbf{b}) = N(\mathbf{a}, \mathbf{b})/n$ for $\mathbf{a} \in GF(2^{r_1})$ and $\mathbf{b} \in GF(2^{r_2})$. The empirical joint distribution is an approximation of the true joint distribution based on the observation of two sequences $\mathbf{x}_1^n$ and $\mathbf{x}_2^n$. Note that the empirical joint distribution defined for each sequence $\left[\mathbf{x}_1^n, \mathbf{x}_2^n\right]$ has a similar form to a probability mass function. Then, the functions applied on probability mass function, such as entropy function, relative entropy function, can be applied to $P_{\mathbf{x}_1^n, \mathbf{x}_2^n}$.

In the decision procedure, given all sequences $\left[\mathbf{x}_1^n, \mathbf{x}_2^n\right]$ that satisfying $\mathbf{z}_t^n = \left[\mathbf{x}_1^n \mathbf{x}_2^n\right] M_t$, find

$$\left\{\hat{\mathbf{x}}_1^n, \hat{\mathbf{x}}_2^n\right\} = \arg \min_{\left[\mathbf{x}_1^n \mathbf{x}_2^n\right] M_t = \mathbf{z}_t^n} \alpha\left(P_{\mathbf{x}_1^n, \mathbf{x}_2^n}\right)$$

where $\alpha(\cdot)$ is a function that needs to be chosen, depending on the metric to be optimized. The two functions discussed below both achieve the capacity region in Theorem 3.

1. *Maximum-Q probability decoder.* $\alpha\left(P_{\mathbf{x}_1^n, \mathbf{x}_2^n}\right) = D\left(P_{\mathbf{x}_1^n, \mathbf{x}_2^n} || Q\right) + H\left(P_{\mathbf{x}_1^n, \mathbf{x}_2^n}\right)$, where $D(\cdot||\cdot)$ is the relative entropy [6],

$$D\left(P_{\mathbf{x}_1^n, \mathbf{x}_2^n} || Q\right) = \sum_{\mathbf{a} \in F_{2^{r_1}}} \sum_{\mathbf{b} \in F_{2^{r_2}}} P_{\mathbf{x}_1^n, \mathbf{x}_2^n}(\mathbf{a}, \mathbf{b}) \log \frac{P_{\mathbf{x}_1^n, \mathbf{x}_2^n}(\mathbf{a}, \mathbf{b})}{Q(\mathbf{a}, \mathbf{b})}$$

and $H(\cdot)$ is the joint entropy function [6]

[5] A length r vector with elements from $GF(2)$ can be viewed as an element from $GF(2^r)$.

$$H\left(P_{\mathbf{x}_1^n,\mathbf{x}_2^n}\right) = -\sum_{\mathbf{a}\in F_2^{r_1}} \sum_{\mathbf{b}\in F_2^{r_2}} P_{\mathbf{x}_1^n,\mathbf{x}_2^n}(\mathbf{a},\mathbf{b}) \log P_{\mathbf{x}_1^n,\mathbf{x}_2^n}(\mathbf{a},\mathbf{b})$$

From [6, theorem 12.1.2], since $\left(\mathbf{x}_1^n, \mathbf{x}_2^n\right)$ are drawn i.i.d. according to $Q(\mathbf{x}_1, \mathbf{x}_2)$, the probability of $\mathbf{x}_1^n, \mathbf{x}_2^n$ is given by

$$Q^n(\mathbf{x}_1, \mathbf{x}_2) = 2^{-n(D(P_{\mathbf{x}_1,\mathbf{x}_2}||Q)+H(P_{\mathbf{x}_1,\mathbf{x}_2}))}$$

Therefore, finding $\mathbf{x}_1^n, \mathbf{x}_2^n$ that minimizing $\alpha\left(P_{\mathbf{x}_1^n,\mathbf{x}_2^n}\right)$ is equivalent to finding $\mathbf{x}_1^n, \mathbf{x}_2^n$ that maximizing the sequence probability.

2. *Minimum entropy decoder.* $\alpha(P_{\mathbf{x}_1,\mathbf{x}_2}) = H(P_{\mathbf{x}_1,\mathbf{x}_2})$.
Note that here the decoder does not need to know the prior source joint distribution Q. Thus, it is an universal decoder. For a long sequence, the empirical distribution $P_{\mathbf{x}_1,\mathbf{x}_2}$ is very close to the true distribution Q, which causes $D(P_{\mathbf{x}_1,\mathbf{x}_2}||Q)$ to approach zero. Therefore, the minimum entropy decoder is an approximation of maximum-Q probability decoder.

It is shown in [64] that as long as

$$min - cut(s_1, t_i) \geq H(X_1|X_2) \tag{7.11}$$

$$min - cut(s_2, t_i) \geq H(X_2|X_1) \text{ and} \tag{7.12}$$

$$min - cut(s_1 \text{ and } s_2, t_i) \geq H(X_1, X_2) \tag{7.13}$$

for every $i = 1, 2, \ldots, L$, each terminal t_i can recover X_1 and X_2 with vanishing error probability when the one of the two decoders shown above is used. Therefore, the admissible rate region achieves bound (7.9). However, the decoding algorithms above are based on exhaustive search and have a complexity that is unacceptably high.

7.5.3 Separating Distributed Source Coding and Network Coding

The achievability scheme described in the previous section performs distributed source coding and network coding jointly and has high decoding complexity. Perhaps the simplest way to multicast correlated sources is to perform distributed source coding and network coding separately, i.e., the source nodes perform distributed source coding (Slepian–Wolf coding) and the coded bits are multicasted to the terminals through network coding. The terminals first decode the network code to obtain the Slepian–Wolf coded bits, then jointly decode the Slepian–Wolf code (usually is a channel code) to recover the sources. The decoding algorithms for network code and Slepian–Wolf code have been well studied and have polynomial time complexity. However, the separation of distributed source coding and network coding is suboptimal in general [68].

At an intuitive level, this result can be understood as follows. Suppose that the network is such that each terminal can operate at the same point in the Slepian–Wolf region. In such a situation, one could use a Slepian–Wolf code and encode each source. Next, one could treat the encoded sources as independent and multicast the encoded bits to each terminal. The terminal then decodes to obtain the original sources. Roughly speaking, in this case we can reduce the correlated sources multicast to an independent sources multicast.

However, if different terminals in the network are forced to operate at different rate points in the Slepian–Wolf region, because of the nature of their connectivity, then a reduction to the independent sources multicast is not possible in general. In this case, clearly one cannot work with a single distributed source code. It can be shown that there exist instances of networks and source distributions such that performing separate distributed source coding and network coding can be strictly suboptimal with respect to the approach in [64]. A surprising conclusion of [68] is that if there are two sources and two terminals in a network, then it can be shown that there is no loss in using a separation-based approach. This result forms the basis of practical approaches to combining distributed source coding and network coding as explained in the next section.

7.5.4 *Practical Joint Distributed Source Coding and Network Coding*

In this section, we describe practical algorithms to perform joint distributed source coding and network coding. Suppose there are two source nodes $s_1, s_2 \in V$ and observe two binary sources X and Y, respectively. The sources generate bits i.i.d. according to the joint distribution $p(X, Y)$ where the joint distribution satisfies the following symmetry property, i.e., $p(X + Y = 1) = p < 0.5$. Then, as discussed before, the sequences $\mathbf{x}, \mathbf{y}$ are related by $\mathbf{y} = \mathbf{x} + \mathbf{e}$, where e_i equals to 1 with probability $p < 0.5$. Note that $H(X, Y) = 1+H_b(p)$ and $I(X; Y) = 1-H_b(p)$. Let H be the parity check matrix for a channel code approaching the capacity of a binary symmetric channel with crossover probability p with code rate $k/n = I(X; Y) = 1 - H_b(p)$, i.e., there is a decoding function $f(\cdot)$ such that $Pr(\mathbf{e} \neq f(\mathbf{e}H^T))$ is arbitrarily close to zero.

The basic idea is to transmit $\mathbf{x}H^T + \mathbf{y}H^T = \mathbf{e}H^T$ to each terminal such that $\mathbf{e}$ can be recovered. Then, we transmit some additional information so that each terminal can recover either $\mathbf{x}$ or $\mathbf{y}$. We shall see the exact form of this additional information later. The simplest but not necessarily optimal way to convey the sum $\mathbf{e}H^T = \mathbf{x}H^T + \mathbf{y}H^T$ to the terminal is to multicast both $\mathbf{x}H^T$ and $\mathbf{y}H^T$ to each terminal and compute the sum at the terminal. Based on this, a practical joint distributed source coding and network coding is proposed in [69]. We first describe this scheme and then discuss the optimal schemes to multicast the sum to the terminals. The scheme in [69] is not optimal in the sense that in general, it requires more network capacity than the result of [64] requires.

The design scheme can be summarized as follows. The network capacity resource C is partitioned into two shares: C_1 and C_2, where $C_1 + C_2 \leq C$. Each share is used to support two multicast sessions. Let $\bar{H}$ be a matrix such that $[\bar{H}^T H^T]$ has full rank. And let $\mathbf{x}_1 = \mathbf{x}\bar{H}, \mathbf{y}_1 = \mathbf{y}\bar{H}$. The two multicast sessions are described as follows.

1. In the first session, multicast $\mathbf{x}H^T$ and $\mathbf{y}H^T$ to each terminal. This implies $\mathbf{e}H^T$ can be computed, and $\mathbf{e}$ can be recovered at each terminal since H is the parity check matrix of a capacity achieving code. Using this, $\mathbf{e}_1 \equiv \mathbf{y}_1 + \mathbf{x}_1 = \mathbf{e}\bar{H}^T$ can be computed.
   The length of $\mathbf{x}H^T$ is $(n - k) = nH(X|Y)$ (likewise for $\mathbf{y}H^T$). We need to multicast $nH(X|Y)$ bits from node s_1 to the terminal and $nH(Y|X)$ bits from node s_2 to the terminal. This requires $G(V, E, C_1)$ to support a multicast with rate $H(X|Y) + H(Y|X)$ from a virtual supersource connected to s_1, s_2 to each terminal.
2. In the second session, the sources transmit linear combinations of $\mathbf{x}_1$ and $\mathbf{y}_1$ to the network and $\mathbf{x}_1 A_t + \mathbf{y}_1 B_t$ is received by terminal t. A_t and B_t are transfer matrices from s_1 to terminal t and s_2 to terminal t, respectively, and they are assumed known to the terminal t. A_t and B_t are such that given $\mathbf{e}_1$ and $\mathbf{x}_1 A_t + \mathbf{y}_1 B_t$, $\mathbf{x}_1, \mathbf{y}_1$ can be recovered. Since we can compute $(\mathbf{x}_1 + \mathbf{y}_1)B_t = \mathbf{e}_1 B_t = \mathbf{e}\bar{H}^T B_t$ and then $\mathbf{x}_1(A_t + B_t) = \mathbf{x}_1 A_t + \mathbf{y}_1 B_t + \mathbf{e}_1 B_t$, as long as $A_t + B_t$ is invertible, $\mathbf{x}_1$ and $\mathbf{y}_1$ can be recovered. The invertibility of $A_t + B_t$ is guaranteed with high probability (for details see [69]). After $\mathbf{x}_1$ is obtained, we compute $\mathbf{y}_1 = \mathbf{e}_1 + \mathbf{x}_1$. Once $\mathbf{x}_1, \mathbf{y}_1$ are known, $\mathbf{x}, \mathbf{y}$ can be recovered by the inversion of $[\bar{H}^T H^T]$ since $[\mathbf{x}\bar{H}^T \mathbf{x}H^T] = \mathbf{x}[\bar{H}^T H^T]$ and $\mathbf{y} = \mathbf{x} + \mathbf{e}$.

The two multicast sessions are illustrated in Fig. 7.7. The admissible rate region for this design scheme is

$$\mathbf{C}^* = \{C_1 + C_2 : C_1 \in \mathbf{C}(\bar{s}, T, H(X|Y) + H(Y|X)) \text{ and } C_2 \in \mathbf{C}(u, T, I(X;Y))\}$$

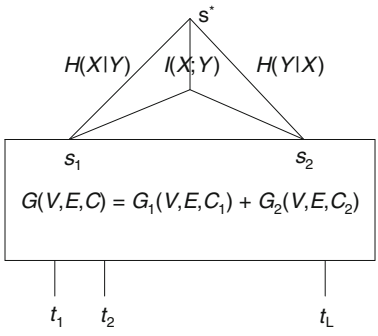

Fig. 7.7 Multicast model for the practical scheme [69]

In general, $\mathbf{C}^*$ requires more capacity than the optimal capacity region [64] because separate multicast sessions are usually suboptimal. But if there are only two terminals (and we are only dealing with two sources), $\mathbf{C}^*$ is optimal, i.e., the practical design scheme is optimal [68, 69].

Computing the sum at the terminals (see [70–72] for related work) may not be optimal in terms of the capacity region. It may in fact be better in terms of resource utilization if the sum is computed at some intermediate nodes and then sent to each terminal. In a network with two sources multiple terminals or two terminals multiple sources, it is shown in [70, 72] that the optimal scheme to convey the symbol sum of the sources to the terminals is to compute the sum in some intermediate nodes and multicast to the terminals. In general, finding the right set of nodes at which the sum should be computed is an open problem. But, the idea of computing the sum at the intermediate nodes leads us to a heuristic approach to the joint distributed source coding and network coding. We can find a set of nodes U and multicast $\mathbf{x}H^T$ and $\mathbf{y}H^T$ to each node in U (*multicast session 1*). Then, compute the sum $\mathbf{e}H^T$ at $u \in U$ and multicast to the terminals so that each terminal can recover $\mathbf{e}$ (*multicast session 2*). Transmit linear combinations $\mathbf{x}A_t + \mathbf{y}B_t$ to the terminals (*multicast session 3*) and if $(A_t + B_t)$ is invertible then both $\mathbf{x}$ and $\mathbf{y}$ can be recovered in a similar manner to the previous scheme. Note that the coded packets in *multicast session 1* can be used in *multicast session 3* since $\mathbf{x}H^T$ and $\mathbf{y}H^T$ are also linear combinations of $\mathbf{x}$ and $\mathbf{y}$. Next we demonstrate an example of this scheme in which we achieve the optimal capacity region.

Consider the network in Fig. 7.8. The capacity on each edge is 0.5. The source nodes s_1, s_2 observe the sources X and Y, and they are correlated such that $H(X) = H(Y) = 1$ and $H(Y|X) = H(X|Y) = 0.5$. The terminals are t_1, t_2, and t_3

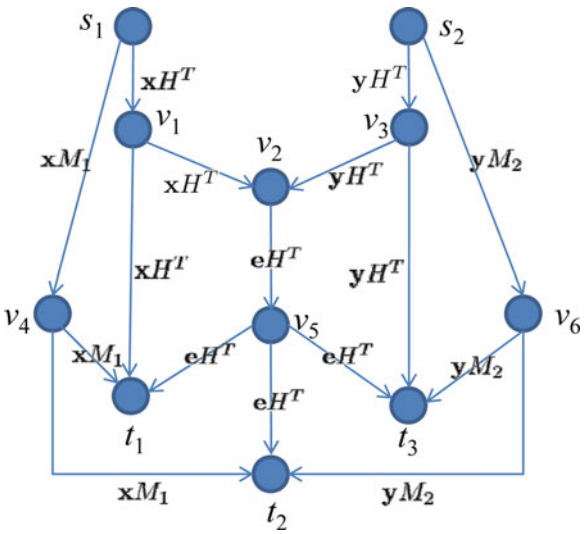

Fig. 7.8 An example where the strategy in [69] is suboptimal. However, our proposed heuristic for selecting the right set of nodes for computing the sum works better

and $\min -\text{cut}(s_i, t_j) = 0.5$ for $i = 1, 2$, $j = 1, 2, 3$, $\min -\text{cut}(\{s_1, s_2\}, t_i) = 1.5$ for $i = 1, 2, 3$. According to Theorem 3, the capacity of this network supports the recovery of the sources at the terminals. Consider the following scheme: s_1, s_2 transmit $\mathbf{x}H^T$ and $\mathbf{y}H^T$ to node v_2 (*multicast session 1*). Node v_2 computes the sum $\mathbf{e}H^T$ and routes it to the terminals (*multicast session 2*). For *multicast session 3*, transmit $\mathbf{x}H^T$, $\mathbf{y}H^T$ on $v_1 - t_1$, $v_3 - t_3$, respectively.[6] In addition, s_1, s_2 transmit random linear combinations on edges $s_1 - v_4$, $s_2 - v_6$, i.e., M_1, M_2 are matrices of dimension $n \times 0.5n$ consisting of entries randomly from $GF(2)$. Then, matrices M_1, M_2, and $[M_1 M_2]$ have full rank with high probability. Terminal t_1 receives $\mathbf{e}H^T$, $\mathbf{x}M_1$ and $\mathbf{x}H^T$. From the first one t_1 can decode $\mathbf{e}$ and from the last two t_1 can recover $\mathbf{x}$, then $\mathbf{y} = \mathbf{x} + \mathbf{e}$ can also be obtained. Terminal t_2 acts in a similar fashion as t_1, while t_3 can decode $\mathbf{e}$ from $\mathbf{e}H^T$ and it also knows $\mathbf{x}M_1$ and $\mathbf{y}M_2$. Therefore, it can compute $\mathbf{x}M_2 = \mathbf{e}M_2 + \mathbf{y}M_2$ then $\mathbf{x}$ can be recovered by the inversion of $[M_1 M_2]$.

As shown in [69], the scheme that multicasts both $\mathbf{x}H^T$ and $\mathbf{y}H^T$ to the terminals cannot achieve the capacity region in the example above. But from some simulations on random graphs, where the optimal set U is found by integer programming, we observe that in many cases, multicasting both $\mathbf{x}H^T$ and $\mathbf{y}H^T$ to the terminals and computing the sum there is as good as computing the sum at some intermediate nodes. Clearly, the best choice of nodes for computing the sum depends on the network topology. The problem of choosing these nodes in an efficient manner is still an open problem.

7.5.5 Resource Allocation for Multicasting Correlated Sources over a Network

Given the admissible region in Sect. 7.5.2, it is natural question to determine the rate at each source and the flow on each edge such that the total cost is minimized. The problem is solved in an efficient manner in [73, 74].

The network is modeled as a directed acyclic graph $G = (V, E, C)$ and each edge is associated with a weight w_{ij}. For simplicity we assume that the cost of the use of an edge (i, j) when the actual data rate on edge (i, j) is z_{ij} is $w_{ij} z_{ij}$. To facilitate the problem formulation we append a virtual super source node s^* to G, so that

$$
\begin{aligned}
V^* &= V \cup \{s^*\} \\
E^* &= \{(s^*, v) |\ v \in S\} \cup E \text{ and} \\
C^*_{ij} &= \begin{cases} C_{ij} & (i, j) \in E \\ H(X_j) & \text{if } i = s^* \text{ and } j \in S \end{cases}
\end{aligned}
$$

We let $G^* = (V^*, E^*, C^*)$. Denote the source node set as $\mathcal{S}$ and the terminal set as T. The admissible region in Theorem 3 requires the min-cut between any

[6] We could also simply perform random linear network coding on these edges.

subset S of nodes and *every* terminal greater than $H(S|S^c)$. From max-flow min-cut theorem, we know the min-cut can be characterized as max-flow. As long as there is a flow of value R from a source s to a terminal t, the min-cut between s and t is R. Thus, to model the conditions on the min-cut, we introduce virtual flows $\mathbf{f}^{(t_k)} = \left\{ f_{ij}^{(t_k)} \right\}$ for each terminal t_k. Note that we only require the existence of the flow for every terminal; the flows corresponding to different terminals can coexist on an edge. So the actual flow rate z_{ij} on edge (i, j) is the maximum (not the sum) of $f_{ij}^{(t_k)}, \forall t_k \in T$, i.e., $z_{ij} \geq f_{ij}^{(t_k)}, \forall t_k \in T$. Based on the discussions above, the problem can be formulated as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{(i,j)\in E} w_{ij} z_{ij} \\
\text{s. t.} \quad & 0 \leq f_{ij}^{(t_k)} \leq z_{ij} \leq C_{ij}^*, \quad (i, j) \in E^*, t_k \in T \\
& \sum_{\{j|(i,j)\in E^*\}} f_{ij}^{(t_k)} - \sum_{\{j|(j,i)\in E^*\}} f_{ji}^{(t_k)} = \sigma_i^{(t_k)}, \text{ for } i \in V^*, t_k \in T, \quad (7.14) \\
& f_{s^*i}^{(t_k)} \geq R_i^{(t_k)}, \quad \text{for } i \in \mathcal{S}, t_k \in T \quad (7.15) \\
& \left(R_1^{(t_k)}, R_2^{(t_k)}, \ldots, R_N^{(t_k)} \right) \in SW_N, \quad \text{for } t_k \in T \quad (7.16)
\end{aligned}$$

where

$$\sigma_i^{(t_k)} = \begin{cases} H(X_1, X_2, \ldots, X_N) & \text{if } i = s^* \\ -H(X_1, X_2, \ldots, X_N) & \text{if } i = t_k \\ 0 & \text{otherwise} \end{cases}$$

The constraint (7.14) is the flow balance constraint for each virtual flow. The constraints (7.15) and (7.16) make sure for each terminal t_k there is a flow of value $H(X_S|X_{S^c})$ from each subset S of sources to t_k. The detailed proof of the correctness of the formulation can be found in [73, 74]. The formulation of *MIN-COST-SW-NETWORK* as presented above is a linear program and can potentially be solved by a regular LP solver. However, the number of constraints due to the requirement that $\mathbf{R} \in \mathcal{SW}_{\mathcal{N}}$ is $|T|(2^N - 1)$ that grows exponentially with the number of sources. For regular LP solvers the time complexity scales with the number of constraints and variables. Thus, using a regular LP solver is certainly not time efficient. Moreover even storing the constraints consumes exponential space and thus using a regular LP solver would also be space inefficient. We now present efficient techniques for solving this problem.

Let $\mathbf{w}, \mathbf{z}, \mathbf{f}^{(t_k)}$ denote the column vectors of $w_{ij}, z_{ij}, f_{ij}^{(t_k)}$ for $(i, j) \in E$ and $\mathbf{R}^{(t_k)}, \mathbf{f}_{s*}^{(t_k)}$ denote the column vectors of $R_i^{(t_k)}, f_{s*i}^{(t_k)}$ for $i = 1, 2, \ldots, |\mathcal{S}|$. Let L be the number of terminals. We form the Lagrangian of the optimization problem with respect to the constraints $\mathbf{R}^{(t_k)} \leq \mathbf{f}_{s*}^{(t_k)}$, for $t_k \in T$. This is given by

$$L(\lambda, \mathbf{z}, \mathbf{f}^{(t_1)}, \ldots, \mathbf{f}^{(t_L)}, \mathbf{R}^{(t_1)}, \ldots, \mathbf{R}^{(t_L)}) = \mathbf{w}^T\mathbf{z} + \sum_{k=1}^{L} \lambda_k^T \left(\mathbf{R}^{(t_k)} - \mathbf{f}_{s^*}^{(t_k)}\right)$$

where $\lambda = \left[\lambda_1^T \ \lambda_2^T \ \ldots \ \lambda_L^T\right]^T$ and $\lambda_k = [\lambda_{k,1}, \lambda_{k,2}, \ldots, \lambda_{k,|\mathcal{S}|}]^T$ are the dual variables such that $\lambda \succeq 0$ (where $\succeq$ denotes component-wise inequality).

For a given λ, let $g(\lambda)$ denote the dual function obtained by

$$g(\lambda) = \text{minimize}_{\mathbf{z}, \mathbf{f}^{(t_1)}, \ldots, \mathbf{f}^{(t_L)}, \mathbf{R}^{(t_1)}, \ldots, \mathbf{R}^{(t_L)}} L\left(\lambda, \mathbf{z}, \mathbf{f}^{(t_1)}, \ldots, \mathbf{f}^{(t_L)}, \mathbf{R}^{(t_1)}, \ldots, \mathbf{R}^{(t_L)}\right)$$

Since strong duality holds in our problem we are guaranteed that the optimal value of *MIN-COST-SW-NETWORK* can be equivalently found by maximizing $g(\lambda)$ subject to $\lambda \succeq 0$ [51]. Thus, if $g(\lambda)$ can be determined in an efficient manner for a given λ then we can hope to solve *MIN-COST-SW-NETWORK* efficiently.

Consider the optimization problem for a given $\lambda \succeq 0$.

$$\begin{aligned} \text{minimize} \quad & \mathbf{w}^T\mathbf{z} + \sum_{k=1}^{L} \lambda_k^T \left(\mathbf{R}^{(t_k)} - \mathbf{f}_{s^*}^{(t_k)}\right) \\ \text{s. t.} \quad & 0 \le f_{ij}^{(T_k)} \le z_{ij} \le C_{ij}, \quad (i,j) \in E^*, t_k \in T \\ & \sum_{\{j|(i,j)\in E^*\}} f_{ij}^{(t_k)} - \sum_{\{j|(j,i)\in E^*\}} f_{ji}^{(t_k)} = \sigma_i^{(t_k)}, \ i \in V^*, t_k \in T \\ & \mathbf{R}^{(t_k)} \in SW_N, \ t_k \in T \end{aligned} \tag{7.17}$$

We realize on inspection that this minimization decomposes into a set of independent subproblems shown below.

$$\begin{aligned} \text{minimize} \quad & \mathbf{w}^T\mathbf{f} - \sum_{k=1}^{L} \lambda_k^T \mathbf{f}_{s^*}^{(t_k)} \\ \text{s. t.} \quad & 0 \le f_{ij}^{(t_k)} \le z_{ij} \le C_{ij}, \quad (i,j) \in E^*, t_k \in T \\ & \sum_{\{j|(i,j)\in E^*\}} f_{ij}^{(t_k)} - \sum_{\{j|(j,i)\in E^*\}} f_{ji}^{(t_k)} = \sigma_i^{(t_k)}, \ i \in V^*, t_k \in T \end{aligned} \tag{7.18}$$

and for each $t_k \in T$,

$$\begin{aligned} \text{minimize} \quad & \lambda_k^T \mathbf{R}^{(t_k)} \\ \text{subject to} \quad & \mathbf{R}^{(t_k)} \in SW_N \end{aligned} \tag{7.19}$$

The optimization problem in (7.18) is a linear program with variables z and $x^{(T_k)}$ for $k = 1, \ldots, N_R$ and a total of $(2|T| + 1)|E^*| + |T||V^*|$ constraints that can be

solved efficiently by using a regular LP solver. It can also be solved by treating it as a minimum cost network flow problem with fixed rates for which many efficient techniques have been developed [50].

However, each of the subproblems in (7.19) still has $2^N - 1$ constraints and therefore the complexity of using an LP solver is still exponential in N. However, recall the contra-polymatroid property of Slepian–Wolf region mentioned in Sect. 7.4.1.2. Using the contra-polymatroid property, the solution to this LP can be found by a greedy allocation of the rates as shown in (7.7), where the permutation π is such that $\lambda_{k,\pi(1)} \geq \lambda_{k,\pi(2)} \geq \cdots \geq \lambda_{k,\pi(N)}$.

The previous algorithm presents us a technique for finding the value of $g(\lambda)$ efficiently. It remains to solve the maximization

$$\max_{\lambda \succeq 0} g(\lambda)$$

For this purpose we use the fact that the dual function is concave (possibly non-differentiable) and can therefore be maximized by using the projected subgradient algorithm [75]. Roughly speaking, the subgradient algorithm is a iterative method to minimize non-differentiable convex (or maximize concave) functions. It is similar to the gradient descent method, though there are notable differences. The subgradient for λ_k can be found as $\mathbf{R}^{(t_k)} - \mathbf{f}_{s^*}^{(t_k)}$ [75].

Let λ^i represent the value of the dual variable λ at the ith iteration and θ_i be the step size at the ith iteration. A step-by-step algorithm to solve *MIN-COST-SW-NETWORK* is presented below.

1. Initialize $\lambda^0 \succeq 0$.
2. For given λ^i solve the problem (7.18) using an LP solver and for each $t_k \in T$, solve the problem (7.19) using the greedy algorithm presented in (7.7).
3. Set $\lambda_k^{i+1} = \left[\lambda_k^i + \theta_i \left(\mathbf{R}^{(t_k)} - \mathbf{f}_{s^*}^{(t_k)}\right)\right]^+$ for all $t_k \in T$, where $[x]^+ = x$ if $x \geq 0$ and zero otherwise. Goto step 2 and repeat until convergence.

While subgradient optimization provides a good approximation on the optimal value of the primal problem, a primal-optimal solution or even a feasible, near-optimal solution is usually not available because the objective function is linear. In our problem, we seek to jointly find the flows and the rate allocations that support the recovery of the sources at the terminals at minimum cost. Thus, finding the appropriate flows and rates specified by the primal-optimal or near primal-optimal $\mathbf{z}, \mathbf{f}^{(t_1)}, \ldots, \mathbf{f}^{(t_L)}, \mathbf{R}^{(t_1)}, \ldots, \mathbf{R}^{(t_L)}$ is important. Toward this end we use the method of Sherali and Choi [76]. We skip the details and refer the interested reader to [73, 74].

7.6 Conclusion

In this survey we have examined the problem of distributed source coding over networks. Distributed source coding has been traditionally studied under a model where there exist direct source destination links. In a general network, the sources

communicate with the destinations over a network whose topology may be quite complicated. It turns out that in this case the problem of distributed source coding and network information transfer needs to be addressed jointly. In particular, treating these problems separately can be shown to be suboptimal in general. Moreover, in certain cases the usage of the network coding [2] becomes essential. We also discussed various resource allocation problems that occur in this space and provided an overview of the solution approaches.

There are several problems that need to be addressed in this area. In the area of sensor networks, it would be interesting to examine if simple protocols can be developed that leverage joint distributed source coding and network coding. In this survey we assumed that the source statistics are known to the intended destination. In practice, the protocols will need to ensure that these statistics are communicated periodically. In a practical sensor network, it is reasonable to assume that some limited communication between the sensors is possible. It would be interesting to see if this reduces the overall complexity of decoding at the destinations.

Acknowledgments This work was supported in part by NSF grant CNS-0721453.

References

1. D. Slepian and J. Wolf. Noiseless coding of correlated information sources. *IEEE Transactions on Information Theory*, 19:471–480, July 1973.
2. R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, 2000.
3. X. Lin and N. Shroff. Utility maximization for communication networks with multipath routing. *IEEE Transactions on Automatic Control*, 51(5):766–781, May 2006.
4. M. Chiang, S. Low, A. Calderbank, and J. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255–312, January 2007.
5. C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948.
6. T. Cover and J. Thomas. *Elements of Information Theory*. Wiley, New York, NY, 1991.
7. Z. Xiong, A. Liveris, and S. Cheng. Distributed source coding for sensor networks. *Signal Processing Magazine, IEEE*, 21(5):80–94, September 2004.
8. T. Cover. A proof of the data compression theorem of slepian and wolf for ergodic sources (corresp.), *IEEE Transactions on Information Theory*, 21(2):226–228, March 1975.
9. V. Stankovic, A. Liveris, Z. Xiong, and C. Georghiades. On code design for the slepian-wolf problem and lossless multiterminal networks. *IEEE Transactions on Information Theory*, 52(4):1495–1507, April 2006.
10. C.-F. Lan, A. Liveris, K. Narayanan, Z. Xiong, and C. Georghiades. Slepian-wolf coding of multiple m-ary sources using ldpc codes. In: *Data Compression Conference, 2004. Proceedings. DCC 2004*, page 549, March 2004.
11. A. Liveris, C. Lan, K. Narayanan, Z. Xiong, and C. Georghiades. Slepian-Wolf coding of three binary sources using LDPC codes. In: *Proceedings of International Symposium on Turbo Codes and Related Topics, Brest, France*, Sep. 2003.
12. A. Wyner. Recent results in Shannon theory. *IEEE Transactions on Information Theory*, 20:2–10, January 1974.

13. S. S. Pradhan and K. Ramchandran. Distributed Source Coding using Syndromes (DISCUS): Design and Construction. *IEEE Transactions on Information Theory*, 49:626–643, March 2003.
14. A. Liveris, Z. Xiong, and C. N. Georghiades. Compression of binary sources with side information at the decoder using LDPC codes. *IEEE Communications Letters*, 6(10):440–442, 2002.
15. D. Varodayan, A. Aaron, and B. Girod. Rate-adaptive codes for distributed source coding. *Signal Processing*, 86(11):3123–3130, 2006.
16. P. Tan and J. L. Tiffany. A general and optimal framework to achieve the entire rate region for slepian-wolf coding. *Signal Processing*, 86(11):3102–3114, 2006.
17. J. Muramatsu and T. Uyematsu, and T. Wadayama. Low-density parity-check matrices for coding of correlated sources. *IEEE Transactions on Information Theory*, 51(10):3645–3654, 2005.
18. M. Sartipi and F. Fekri. Distributed source coding in wireless sensor networks using LDPC coding: the entire Slepian-Wolf rate region. In: *IEEE Wireless Communications and Networking Conference (WCNC)*, New Orleans, LA, USA, March 2005.
19. S. Pradhan and K. Ramchandran. Distributed source coding: symmetric rates and applications to sensor networks. In: *Data Compression Conference, 2000. Proceedings. DCC 2000*, pages 363–372, 2000.
20. S. Lin and D. J. Costello. *Error Control Coding: Fundamentals and Applications*. Prentice Hall, New Jersey, 2004.
21. R. Gallager. *Information Theory and Reliable Communication*, Wiley, New York, 1968.
22. T. Richardson and R. Urbanke. *Modern Coding Theory*, Cambridge University Press, Cambridge, 2008.
23. J. Garcia-Frias, Y. Zhao, and W. Zhong. Turbo-like codes for transmission of correlated sources over noisy channels. *Signal Processing Magazine, IEEE*, 24(5):58–66, September 2007.
24. J. Garcia-Frias and Y. Zhao. Compression of correlated binary sources using turbo codes. *IEEE Communications Letters*, 5(10):417–419, October 2006.
25. A. Aaron and B. Girod. Compression with side information using turbo codes. In: *Proceedings of IEEE Data Compression Conference(DCC)*, pages 252–261, 2002.
26. T. P. Coleman, A. H. Lee, M. Medard, and M. Effros. Low-complexity approaches to Slepian-Wolf near-lossless distributed data compression. *IEEE Transactions on Information Theory*, 52(8):3546–3561, 2006.
27. B. Rimoldi and C. Urbanke. Asynchronous Slepian-Wolf coding via source-splitting. In: *Proceedings of International Symposium on Information Theory, Ulm, Germany*, page 271, Ulm, Germany, June–July 1997.
28. S. Y. Tung, Multiterminal source coding, *Ph.D. Dissertation*, Cornell University, 1978.
29. K. Housewright. Source coding studies for multiterminal systems. *Ph.D. Dissertation*, University of California, Los Angeles, 1977.
30. T. Berger. Multiterminal source coding. In: *CISM Courses and Lectures No. 229, The Information Theory Approach to Communications*, Springer-Verlag, Berlin, 1980.
31. W. Kang and S. Ulukus. An outer bound for the multi-terminal rate-distortion region. In: *IEEE International Symposium on Information Theory*, pages 1419–1423, Seattle, Washington, USA, July 2006.
32. T. Berger. Multiterminal rate-distortion theory revisited. In *Communication, Control, and Computing, 2008 46th Annual Allerton Conference*, pages 1535–1537, September 2008.
33. Y. Oohama. Gaussian multiterminal source coding. *IEEE Transactions on Information Theory*, 43(6):1912–1923, November 1997.
34. Y. Oohama, Rate-distortion theory for gaussian multiterminal source coding systems with several side informations at the decoder. *IEEE Transactions on Information Theory*, 51(7):2577–2593, July 2005.

35. Y. Yang, V. Stankovic, Z. Xiong, and W. Zhao. On multiterminal source code design. In *Data Compression Conference, 2005. Proceedings. DCC 2005*, pages 43–52, Snowbird, Utah, USA, March 2005.
36. Y. Zhang, Z. Xiong, and Y. Yang. Code design for quadratic gaussian multiterminal source coding: The symmetric case. In: *IEEE International Symposium on Information Theory*, 28:1458–1462, July 3, 2009.
37. A. Wyner and J. Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Transactions on Information Theory*, 22(1):1–10, January 1976.
38. R. Zamir, S. Shamai, and U. Erez. Nested linear/lattice codes for structured multiterminal binning. *IEEE Transactions on Information Theory*, 48(6):1250–1276, June 2002.
39. S. Servetto, Lattice quantization with side information: Codes, asymptotics, and applications in sensor networks. *IEEE Transactions on Information Theory*, 53(2):714–731, February 2007.
40. Z. Liu, S. Cheng, A. Liveris, and Z. Xiong. Slepian-wolf coded nested quantization (swc-nq) for wyner-ziv coding: performance analysis and code design. In: *Data Compression Conference, 2004. Proceedings. DCC 2004*, pages 322–331, March 2004.
41. B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero. Distributed video coding. *Proceedings of the IEEE*, 93(1):71–83, January 2005.
42. Y. Yang, S. Cheng, Z. Xiong, and W. Zhao. Wyner-ziv coding based on tcq and ldpc codes. *IEEE Transactions on Communications*, 57(2):376–387, February 2009.
43. T. Berger, Z. Zhang, and H. Viswanathan. The ceo problem [multiterminal source coding]. *IEEE Transactions on Information Theory*, 42(3):887–902, May 1996.
44. H. Viswanathan and T. Berger. The quadratic gaussian ceo problem. *IEEE Transactions on Information Theory*, 43(5):1549–1559, September 1997.
45. Y. Oohama. The rate-distortion function for the quadratic gaussian ceo problem. *IEEE Transactions on Information Theory*, 44(3):1057–1070, May 1998.
46. V. Prabhakaran, D. Tse, and K. Ramachandran. Rate region of the quadratic gaussian ceo problem. In: *IEEE International Symposium on Information Theory*, page 119, June-2 July 2004.
47. J.-H. Chang and L. Tassiulas. Maximum lifetime routing in wireless sensor networks. *IEEE/ACM Transactions on Networking*, 12(4):609–619, August 2004.
48. T. S. Han. Slepian-wolf-cover theorem for networks of channels. *Information and Control*, 47(1):67–83, October 1980.
49. J. Barros and S. Servetto. Network information flow with correlated sources. *IEEE Transactions on Information Theory*, (52):155–170, January 2006.
50. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, New Jersey, 1993.
51. S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, 2004.
52. A. Roumy and D. Gesbert, Optimal matching in wireless sensor networks. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):725–735, December 2007.
53. S. Li and A. Ramamoorthy. Rate and power allocation under the pairwise distributed source coding constraint. *IEEE Transactions on Communications*, 57(12), December 2009.
54. Y. J. Chu and T. H. Liu, On the shortest arborescence of a directed graph. *Science Sinica*, 14:1396–1400, 1965.
55. R. Cristescu and B. Beferull-Lozano. Lossy network correlated data gathering with high-resolution coding. *IEEE Transactions on Information Theory*, 52(6):2817–2824, June 2006.
56. R. Giles. Optimum matching forests I: Special weights. *Mathematical Programming*, 22(1):1–11, December 1982.
57. R. Giles, Optimum matching forests ii: General weights. *Mathematical Programming*, 22(1):12–38, December 1982.

58. R. Cristescu, B. Beferull-Lozano, and M. Vetterli. Networked slepian-wolf: theory, algorithms, and scaling laws. *IEEE Transactions on Information Theory*, 51(12):4057–4073, December 2005.
59. D. Tse and S. Hanly. Multiaccess fading channels. i. polymatroid structure, optimal resource allocation and throughput capacities. *IEEE Transactions on Information Theory*, 44(7):2796–2815, November 1998.
60. J. Liu, M. Adler, D. Towsley, and C. Zhang. On optimal communication cost for gathering correlated data through wireless sensor networks. In: *ACM MobiCom*, Los Angeles, CA, USA, 2006.
61. J. Kleinberg and E. Tardos. *Algorithm Design*. Addison Wesley, New York, NY, 2005.
62. S.-Y. Li, R. W. Yeung, and N. Cai. Linear Network Coding. *IEEE Transactions on Information Theory*, 49(2):371–381, 2003.
63. R. Koetter and M. Médard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking*, 11(5):782–795, 2003.
64. T. Ho, M. Medard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong. A random linear network coding approach to multicast. *IEEE Transactions on Information Theory*, 52(10):4413–4430, 2006.
65. P. A. Chou, Y. Wu, and K. Jain. Practical network coding. In: *41st Allerton Conference on Communication, Control, and Computing*, 2003.
66. T. S. Han. Multicasting of correlated multisource to multisink over a network. *preprint*, 2009. [Online]. Available: http://arxiv.org/abs/0901.0608
67. L. Song and R. W.Yeung. Network information flow – multiple sources. In: *Proceedings of Intl Symp. Information Theory*, Washington DC, USA, 2001.
68. A. Ramamoorthy, K. Jain, P. A. Chou, and M. Effros. Separating Distributed Source Coding from Network Coding, *IEEE Transactions on Information Theory*, 52:2785–2795, June 2006.
69. Y. Wu, V. Stankovic, Z. Xiong, and S.-Y. Kung. On practical design for joint distributed source and network coding. *IEEE Transactions on Information Theory*, 55(4):1709–1720, 2009.
70. A. Ramamoorthy. Communicating the sum of sources over a network. In *Proceedings of International Symposium Information Theory*, pages 1646–1650, Toronto, ON, Canada, 2008.
71. A. Ramamoorthy and M. Langberg. Communicating the sum of sources in a 3-sources/3-terminals network. In *Proceedings of International Symposium Information Theory*, pages 2121–2125, Seoul, Korea, 2009.
72. A. Ramamoorthy and M. Langberg. Communicating the sum of sources over a network. *preprint*, 2010. [Online]. Available: http://arxiv.org/abs/1001.5319
73. A. Ramamoorthy. Minimum cost distributed source coding over a network. In: *IEEE International Symposium on Information Theory*, pages 1761–1765, Nice, France, 2007.
74. A. Ramamoorthy, Minimum cost distributed source coding over a network. In: *IEEE Transactions on Information Theory*
75. D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Nashua, NH, 1999.
76. H. D. Sherali and G. Choi. Recovery of primal solutions when using subgradient optimization methods to solve lagrangian duals of linear programs. *Operations Research Letters*, 19(3):105–113, 1996.