3-2016

Optimization Algorithms for Big Data with Application in Wireless Networks

Mingyi Hong  
Iowa State University  

Wei-Cheng Liao  
University of Minnesota  

Ruoyu Sun  
University of Minnesota  

Zhi-Quan Luo  
University of Minnesota

Follow this and additional works at: https://lib.dr.iastate.edu/imse_pubs

Part of the Signal Processing Commons, Systems and Communications Commons, and the Systems Engineering Commons  

The complete bibliographic information for this item can be found at https://lib.dr.iastate.edu/imse_pubs/171. For information on how to cite this item, please visit http://lib.dr.iastate.edu/howtocite.html.
Optimization Algorithms for Big Data with Application in Wireless Networks

Abstract
This chapter proposes the use of modern first-order large-scale optimization techniques to manage a cloud-based densely deployed next-generation wireless network. In the first part of the chapter we survey a few popular first-order methods for large-scale optimization, including the block coordinate descent (BCD) method, the block successive upper-bound minimization (BSUM) method and the alternating direction method of multipliers (ADMM). In the second part of the chapter, we show that many difficult problems in managing large wireless networks can be solved efficiently and in a parallel manner, by modern first-order optimization methods. Extensive numerical results are provided to demonstrate the benefit of the proposed approach.

Disciplines
Signal Processing | Systems and Communications | Systems Engineering

Comments
This is a chapter published as Mingyi Hong, Wei-Cheng Liao, Ruoyu Sun and Zhi-Quan Luo "Optimization Algorithms for Big Data with Application in Wireless Networks," in Big Data over Networks, ed. Shuguang Cui, Alfred O. Hero III, Zhi-quan Luo, and Jose M. F. Moura (Cambridge: Cambridge University Press, 2016), pp. 66-100. Posted with permission.
This chapter proposes the use of modern first-order large-scale optimization techniques to manage a cloud-based densely deployed next-generation wireless network. In the first part of the chapter we survey a few popular first-order methods for large-scale optimization, including the block coordinate descent (BCD) method, the block successive upper-bound minimization (BSUM) method and the alternating direction method of multipliers (ADMM). In the second part of the chapter, we show that many difficult problems in managing large wireless networks can be solved efficiently and in a parallel manner, by modern first-order optimization methods. Extensive numerical results are provided to demonstrate the benefit of the proposed approach.

3.1 Introduction

3.1.1 Motivation

The ever-increasing demand for rapid access to large amounts of data anywhere anytime has been the driving force in the current development of next-generation wireless network infrastructure. It is projected that within 10 years, the wireless cellular network will offer up to $1000\times$ throughput performance over the current 4G technology [1]. By that time the network should also be able to deliver a fiber-like user experience, boasting 10 Gb/s individual transmission rate for data-intensive cloud-based applications.

Achieving this lofty goal requires revolutionary infrastructure and highly sophisticated resource management solutions. A promising network architecture to meet this requirement is the so-called cloud-based radio access network (RAN), where a large number of networked base stations (BSs) are deployed for wireless access, while powerful cloud centers are used at the back end to perform centralized network management [1–4]. Intuitively, a large number of networked access nodes, when intelligently provisioned, will offer significantly improved spectrum efficiency, real-time load balancing and hotspot coverage. In practice, the optimal network provisioning is extremely challenging, and its success depends on smart joint backhaul provisioning, physical layer transmit/receive schemes, BS/user cooperation and so on.
This chapter proposes the use of modern first-order large-scale optimization techniques to manage a cloud-based densely deployed next-generation wireless network. We show that many difficult problems in this domain can be solved efficiently and in a parallel manner, by advanced optimization algorithms such as the block successive upper-bound minimization (BSUM) method and the alternating direction methods of multipliers (ADMM) method.

3.1.2 The organization of the chapter

To begin with, we introduce a few well-known first-order optimization algorithms. Our focus is on algorithms suitable for solving problems with certain block-structure, where the optimization variables can be divided into (possibly overlapping) blocks. Next we show that this type of block-structured problem turns out to be crucial in modeling many network provisioning problems arising in next-generation network design. A few detailed examples are provided to demonstrate the applicability of the first-order optimization algorithms in large-scale data delivery and network provisioning. Numerical examples are given at the end to demonstrate the efficiency of the algorithms studied throughout the article.

3.2 First-order algorithms for big data

In this chapter we consider algorithms that can solve the block-structured optimization problems of the following form

$$\min_{x} f(x_1, x_2, \ldots, x_n), \quad \text{s.t. } (x_1, x_2, \ldots, x_n) \in \mathcal{X},$$

(3.1)

where $f(\cdot)$ is a continuous function (possibly nonconvex and nonsmooth), $\mathcal{X}$ is a closed convex set, and each $x_i \in \mathbb{R}^{m_i}$ is a block variable, $i = 1, 2, \ldots, n$. Later we will see that this type of problem appears frequently in many network provisioning problems that arise in next-generation network design.

3.2.1 The block coordinate descent algorithm

In practice, solving (3.1) directly can be very challenging, due to either its nonconvexity, nonsmoothness, or the sheer problem size. However, consider the special case of (3.1) where the constraint set has a Cartesian product structure: $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_i$, and the nonsmooth part of the objective is separable among the variables. A well-known technique for such special case is the so-called block coordinate descent (BCD) method whereby, at every iteration, a single block of variables is optimized while the remaining blocks are held fixed. More specifically, we consider the following special case of problem (3.1)

$$\min_{x} f(x) = h_0(x_1, x_2, \ldots, x_n) + \sum_{i=1}^{n} h_i(x_i),$$

(3.2)

s.t. $x_i \in \mathcal{X}_i, \ i = 1, \ldots, n,$
where $h_0(\cdot)$ is a smooth function (possibly nonconvex), and $h_i(\cdot), i = 1, \ldots, n$ are convex functions (possibly nonsmooth). When following the classic Gauss–Seidel (G-S) update rule, at iteration $t$, the block $i = (t \mod n) + 1$ is updated by

$$
x_i^{(t)} \in \operatorname{arg\,min}_{y_i \in \mathbb{Y}_i} f(x_1^{(t-1)}, \ldots, x_{i-1}^{(t-1)}, y_i, x_{i+1}^{(t-1)}, \ldots, x_n^{(t-1)})
$$

(3.3)

while the remaining blocks are kept unchanged, i.e. $x_k^{(t)} = x_k^{(t-1)}$ for all $k \neq i$. Since each step involves solving a simple subproblem of small size, the BCD method can be quite effective for solving large-scale problems, provided that certain regularity conditions are met. For instance, the existing analysis of the BCD method [5–7] requires the uniqueness of the minimizer for the subproblems (3.3), or the quasi-convexity of $f$.

Below is a summary of the convergence results of the BCD method for solving (3.2).

**Theorem 3.1** Assume that the level set $X^0 = \{f(x) \leq f(x^0)\}$ is compact. Then the sequence $\{x^{(t)} = (x_1^{(t)}, \ldots, x_n^{(t)})\}$ generated by the BCD method is well-defined and bounded. Further, we have the following.

1. If $f(x_1, \ldots, x_n)$ is pseudoconvex in $(x_k, x_i)$ for every $(i, k) \in \{1, \ldots, n\}$, then every cluster point of $\{x^{(t)}\}$ is a stationary point of $f$.
2. If $f(x_1, \ldots, x_n)$ has at most one minimum in $x_k$ for $k = 2, \ldots, n-1$, then every cluster point $z$ of $\{x^{(t)}\}_{t=(n-1)\mod n}$ is a stationary point of $f$.

This result is adapted from [5, Theorem 4.1], where the “regularity” of $f$ therein is implied by the smooth plus separable nonsmooth objective of problem (3.2). Further, the “stationary solutions” here are the solutions that satisfying the first-order optimality condition; see [5] for the precise definition.

When $f(\cdot)$ is a convex function, it is possible to characterize the rate of convergence for BCD-type algorithm. For example, when the objective function is strongly convex, the BCD algorithm converges globally linearly [8], that is

$$
f(x^{(t+1)}) - f(x^*) \leq c (f(x^{(t)}) - f(x^*))
$$

(3.4)

for some constant $0 < c < 1$. When the objective function is smooth but not strongly convex, Luo and Tseng have shown that the BCD method with the G-S rule converges linearly, provided that a certain local error bound is satisfied around the solution set [8–10]. For more general convex problems, several recent studies have established the $\mathcal{O}(1/t)$ iteration complexity for various BCD-type algorithms [11–14]. In these works, it is shown that when the problem satisfies certain regularity conditions, and when the coordinates are selected according to certain probability distribution, then the bound of the following type is true:

$$
\mathbb{E} \left[ f(x^{(t)}) - f(x^*) \right] \leq \frac{d}{t},
$$

(3.5)

where the expectation is taken over the randomization of the choice of the coordinates, and $d > 0$ is some constant. When the coordinates are updated according to the traditional G-S rule, a few recent works [15–17] have proven the $\mathcal{O}(1/t)$ rate for the G-S BCD algorithm when applied to certain special convex problems. Some recent works [18, 19] propose BCD-based algorithms with parallel block update rules. These algorithms are
designed for both convex and nonconvex problems, and the built-in parallelism offers a significant speed up in computation when multiple computing nodes are available.

It is important to note that without the assumptions such as the uniqueness of the minimizers of the subproblems or the separability of the constraint set, the BCD method may get stuck at a non-stationary point of the problem (see [20] and [21] for well-known examples). Unfortunately, sometimes these assumptions can be restrictive in practice. We will show how to generalize the BCD method when these assumptions are not satisfied in the following subsections.

3.2.2 The ADMM algorithm

In many contemporary applications involving big data, the objective function of (3.1) is convex separable, and the block variables are linearly coupled in the constraint:

\[
\begin{align*}
\text{minimize} & \quad f(x) = f_1(x_1) + f_2(x_2) + \cdots + f_n(x_n) \\
\text{s.t.} \quad & Ex = E_1x_1 + E_2x_2 + \cdots + E_nx_n = q, \\
& x_i \in \mathcal{X}_i, \quad i = 1, \ldots, n
\end{align*}
\]

(3.6)

where \( E = (E_1, \ldots, E_n) \) is the partition of matrix \( E \) corresponding to the block variables \( x_1, \ldots, x_n \).

Directly applying the BCD method to problem (3.6) may fail to find any (local) optimal solution. For instance, the following simple quadratic problem has an optimal objective of 0, but the BCD method can get stuck at the non-interesting point \((1, -1)\):

\[
\begin{align*}
\text{minimize} & \quad x_1^2 + x_2^2, \quad \text{s.t.} \quad x_1 + x_2 = 0.
\end{align*}
\]

In the ADMM method, instead of maintaining feasibility all the time, the constraint \( Ex = q \) is dualized using the Lagrange multiplier \( y \) and a quadratic penalty term is added. The resulting augmented Lagrangian function is of the form:

\[
L(x; y) = f(x) + \langle y, q - Ex \rangle + \frac{\rho}{2} \|q - Ex\|^2, 
\]

(3.7)

where \( \rho > 0 \) is a constant and \( \langle \cdot, \cdot \rangle \) denotes the inner product operator. The ADMM method updates the primal block variables \( x_1, \ldots, x_n \) similarly to BCD to minimize \( L(x; y) \), which often leads to simple subproblems with closed-form solutions. These updates are followed by a gradient ascent update of the dual variable \( y \). Equation (3.8) summarizes the ADMM method.

\[
\begin{align*}
x_i^{(t+1)} &= \arg \min_{x_i} L(x_1^{(t+1)}, \ldots, x_{i-1}^{(t+1)}, x_i, x_{i+1}^{(t)}, \ldots, x_n^{(t)}, y^{(t)}), \quad i = 1, 2, \ldots, n, \\
y^{(t+1)} &= y^{(t)} + \alpha (q - Ex^{(t+1)}) = y^{(t)} + \alpha \left( q - \sum_{i=1}^n E_i x_i^{(t+1)} \right)
\end{align*}
\]

(3.8)

where \( \alpha > 0 \) is the step size for the dual update.
Although the ADMM algorithm was introduced as early as 1976 by Gabay, Mercier, Glowinski, and Marrocco [22, 23], it became popular only recently due to its applications in modern large-scale optimization problems arising from machine learning and computer vision [24–28]. In practice, the algorithm is often computationally very efficient and exhibits much faster convergence than other traditional algorithms such as the dual ascent algorithm [29–31] or the method of multipliers [32].

When there are only two block variables \( n = 2 \), the ADMM converges under very mild conditions; see the following basic result from [7, Proposition 4.2].

**Theorem 3.2** Suppose that \( n = 2 \) and \( \alpha = \rho \). Assume that the optimal solution set of problem (3.6) is non-empty, and \( E_1^T E_1 \) and \( E_2^T E_2 \) are invertible. Then the sequence of \( \{x(t), y(t)\} \) generated by the ADMM algorithm is bounded and every limit point of \( \{x(t), y(t)\} \) is an optimal primal-dual solution of problem (3.6).

Several recent works [33, 34] have shown that the ADMM method converges with the rate of \( O(\frac{1}{t}) \). Moreover, references [35–37] have shown that the ADMM converges linearly when the objective function is strongly convex and there are only two blocks of variables. Unfortunately, the understanding of the algorithm for the case of \( n \geq 3 \) is still very limited. In fact, the convergence of the ADMM method for the case of \( n \geq 3 \) has been an open question since the late 1980s, precluding its direct application to many important problems such as the robust PCA [38]. Recent advances in extending the convergence analysis of ADMM to multiple-block case can be found for example in [39–42].

### 3.2.3 The BSUM method

If the per-block subproblem (3.3) is nonconvex, the BCD algorithm cannot be used due to the difficulty in solving each of the subproblems. To broaden the applicability of the BCD method, a block successive upper-bound minimization (BSUM) is proposed in [43], in which a sequence of approximate versions (e.g. upper bounds) of the objective function is minimized. It is shown that in many applications it is possible to construct subproblems with simple solutions.

Specifically, at each iteration \( t \) of the BSUM method, one chooses an index set \( I(t) \subseteq \{1, 2, \ldots, n\} \) and performs the following update

\[
\begin{align*}
    x_i^{(t+1)} &= \arg\min_{x_i \in \mathcal{X}_i} \ g_i(x_i; x^{(t)}) + h_i(x_i), \quad \forall \ i \in I(t) \\
    x_i^{(t+1)} &= x_i^{(t)}, \quad \forall \ i \notin I(t),
\end{align*}
\]  

(3.9)

where \( g_i(x_i; z) \) is an approximation of the smooth function \( h_0(x_i, z_{-i}) \) at a given \( z \) which satisfies the following assumption.

**Assumption A.**

\[
\begin{align*}
    g_i(x_i; x) &= h_0(x), \quad \forall \ x \in \mathcal{X}, \quad \forall \ i \tag{A1} \\
    g_i(x_i; z) &\geq h_0(x_i, z_{-i}), \quad \forall \ x_i \in \mathcal{X}_i, \quad \forall \ z \in \mathcal{X}, \quad \forall \ i \tag{A2}
\end{align*}
\]
\[ \nabla g_i(z_i; z) = \nabla_i h_0(z), \quad \forall \ z_i \in \mathcal{X}_i, \ \forall \ i \]  
(A3)

\[ g_i(x_i; z) \text{ is continuous in } (x_i, z), \quad \forall \ i \]  
(A4)

\[ g_i(x_i; z) \text{ is strictly convex in } x_i, \quad \forall \ i. \]  
(A5)

The assumptions (A1) and (A2) imply that the approximation function is a global upper bound of \( h_0(x) \); while the assumption (A3) guarantees that the first-order behavior of the objective function and the approximation function are the same.

The BSUM method has wide application in various engineering domains. Many well-known existing algorithms for solving both convex and nonconvex problems are in fact special cases of BSUM. Examples include the proximal gradient method [44], the alternating least square (ALS) method for tensor decomposition [45], the weighted minimum mean square error (WMMSE) algorithm in wireless communication [46], the EM algorithm in statistics [47], the convex concave procedure (CCP) [48], the majorization minimization method (MM) [49] and the nonnegative matrix factorization [50] for machine learning. One related method is the inner approximation algorithm (IAA) developed by Marks and Wright in [51]. Its convergence analysis is quite restrictive: it is applicable only to smooth problems with a single block variable. Moreover, convergence to a stationary solution is established only under the unreasonable assumption that the whole iterate sequence converges (see [51, Theorem 1]).

Below we present a general convergence theorem for the BSUM method.

**Theorem 3.3** The following hold true.

(a) Suppose that the function \( g_i(x_i; y) \) is quasi-convex in \( x_i \) and Assumptions (A1)–(A4) hold. Further assume that the subproblem (3.9) has a unique solution for all \( x^{(t-1)} \in \mathcal{X} \). Then every limit point \( z \) of the iterates generated by the BSUM algorithm is a stationary point of (3.2).

(b) Suppose the level set \( \mathcal{X}^{(0)} = \{x \mid f(x) \leq f(x^{(0)})\} \) is compact and Assumptions (A1)–(A4) hold. Further assume that the subproblem (3.9) has a unique solution for any point \( x^{(t-1)} \in \mathcal{X} \) for at least \( n - 1 \) blocks. Then the iterates generated by the BSUM algorithm converge to the set of stationary points.

This result is adapted from [43, Theorem 2], where again the "regularity" of \( f \) is implied by the smooth plus nonsmooth structure of the objective in (3.2). The convergence of BSUM algorithm can also be established under other assumptions. For example, it is possible to drop the uniqueness requirement in the solution of subproblems provided we update the block that provides the maximum amount of improvement; see [43, 52].

To close this section, we remind the readers that the main strength of all the first-order algorithms discussed in this chapter lies in the simplicity of solving their subproblems. Therefore, when applying these algorithms to solve practical problems, it is often desirable to find the right problem structure that leads to easy updates. In the next section we will show how this can be done for a wide class of network provisioning problems.
3.3 Application to network provisioning problem

In this section, we provide a few concrete examples to demonstrate the applicability of the first-order algorithms such as BSUM and ADMM for large-scale network provisioning problems. We start with describing the general network setting.

3.3.1 The setting

We first describe the generic network model to be studied in the subsequent discussion; see Figure 3.1 for an illustration. For simplicity, we consider the downlink direction in which the traffic flows from the network to the users.

Consider the next-generation access network consisting both the wired backhaul network, which delivers the data flow from the core network to the BSs, and the wireless radio access network (RAN) that transfers the data wirelessly to the users. The wireless RAN consists of a set of BSs \( B \), a set of mobile users \( U \) and a set of wireless links:

\[
\mathcal{L}^w = \{(s_k, d_e) \in B \times U\}. \tag{3.10}
\]

Here we have used \( s_k \) and \( d_e \) to denote the BS–user pair that uniquely defines a link \( \ell \).

Also suppose that each node in the system has a single antenna, and use \( h_{d_e s_k} \in \mathbb{C} \), or simply \( h_{d_e} \in \mathbb{C} \), to denote the channel between BS \( s_k \) and user \( d_e \). Using this notation, the wireless link \( \ell \) is said to be interfered by the set of wireless links \( \mathcal{I}(\ell) = \{ k \in \mathcal{L}^w \mid h_{d_e} \neq 0 \} \). See Figure 3.2 for an illustration of this simple network setting.

For a wireless link \( \ell \in \mathcal{L}^w \), BS \( s_k \) uses a linear precoder \( v_k \in \mathbb{C} \) to transmit to user \( d_e \). Use \( v \) to collect the precoders from all the BSs. Then the transmit rate achievable over
Optimization algorithms for big data with application in wireless networks

Figure 3.2 Illustration of the wireless part of the network.

The link \( \ell \) is given by

\[
 r_\ell(v) \triangleq \log \left( 1 + \frac{|h_\ell|^2 |v_\ell|^2}{\sum_{n \in \mathcal{L}(\ell) \setminus \{\ell\}} |h_{\ell n}|^2 |v_n|^2 + \sigma_\ell^2} \right),
\] (3.11)

where \( \sigma_\ell^2 \) is the variance of AWGN noise at mobile user \( d_\ell \).

To describe the entire access network, let \( \mathcal{V} \) denote the set of nodes in the network, including a set of network routers \( \mathcal{N} \), a set of BSs \( \mathcal{B} \), and a set of mobile users \( \mathcal{U} \). Let \( \mathcal{L} \) denote the set of directed links that connect the nodes of \( \mathcal{V} \). The set \( \mathcal{L} \) consists both wireless links and wired links: the former is defined in (3.10), and the latter connects the nodes in the backhaul and is given by

\[
 \mathcal{L}^w \triangleq \{(s_\ell, d_\ell) : (s_\ell, d_\ell) \in \mathcal{L} \land s_\ell \in \mathcal{N} \cup \mathcal{B} \}. \] (3.12)

Suppose a set of \( \mathcal{M} \) flows is to be delivered from the network to the users, and each \( m \in \mathcal{M} \) has a source \( s(m) \in \mathcal{V} \) and a sink \( d(m) \in \mathcal{V} \). Owing to the fact that only the downlink direction is considered, \( d(m) \) must be one of the users. We use \( r(m) \geq 0 \) and \( r_\ell(m) \geq 0 \) to denote the rate of flow \( m \) and the rate of flow \( m \) on link \( \ell \in \mathcal{L} \), respectively. Define

\[
 r \triangleq \{r(m), r_\ell(m) \}_{m \in \mathcal{M}, \ell \in \mathcal{L}}.
\]

We use the following notation for the set of links going into and coming out of a node \( v \) respectively

\[
 \text{In}(v) \triangleq \{ \ell \in \mathcal{L} \mid d_\ell = v \} \quad \text{and} \quad \text{Out}(v) \triangleq \{ \ell \in \mathcal{L} \mid s_\ell = v \}. \] (3.13)

It is important to note that using the above network model, we implicitly allow a mobile user to be served by more than one BSs; see Figure 3.3 for an illustration.
Below we describe a few link-level constraints that regulate the data flows.

(1) **Wired link capacity constraint** Assume each wired link $\ell \in \mathcal{L}^w$ has a fixed capacity $C_\ell$. The total flow rate on link $\ell$ is constrained by

$$\sum_{m \in \mathcal{M}} r_\ell(m) \leq C_\ell, \quad \forall \ell \in \mathcal{L}^w. \quad (3.14)$$

(2) **Wireless link capacity constraint** The total flow rates on a given wireless link $\ell \in \mathcal{L}^{wi}$ should not exceed the capacity (3.11):

$$\sum_{m \in \mathcal{M}} r_\ell(m) \leq r_\ell(v) = \log \left( 1 + \frac{|h_\ell|}{\sum_{n \in \mathcal{L}(\ell) \setminus \ell} |h_{\ell n}|^2 |v_n|^2 + \sigma_\ell^2} \right), \quad \forall \ell \in \mathcal{L}^{wi}. \quad (3.15)$$

(3) **Flow conservation constraint** For any node $v \in \mathcal{V}$, the total incoming flow should be equal to the total outgoing flow:

$$\sum_{\ell \in \text{In}(v)} r_\ell(m) + 1_{s(m)(v)} r(m) = \sum_{\ell \in \text{Out}(v)} r_\ell(m) + 1_{d(m)(v)} r(m), \quad \forall m \in \mathcal{M}, \forall v \in \mathcal{V} \quad (3.16)$$

where the notation $1_A(x)$ denotes the indicator function for a set $A$, i.e. $1_A(x) = 1$ if $x \in A$, and $1_A(x) = 0$ otherwise.
Optimization algorithms for big data with application in wireless networks

(4) **BS power budget constraint** The transmit power used by each BS $b \in B$ should not exceed given budget $\bar{p}_b \geq 0$:

$$\sum_{\ell \in \text{Out}(b) \cap L^w} |v_\ell|^2 \leq \bar{p}_b, \; \forall \; b \in B.$$  \hspace{1cm} (3.17)

We are interested in data delivery problem formulated in the following utility maximization form

$$\max_{v, r} \quad U ([r(m)]_{m \in \mathcal{M}})$$

$$\text{s.t.} \quad (3.14), (3.15), (3.16), (3.17)$$

$$r(m) \geq 0, \; \forall \; m \in \mathcal{M}$$  \hspace{1cm} (3.18)

where $U(\cdot)$ is the system utility function which measures the performance of the entire network. We note that the system utility maximization problem (3.18) is described in a fairly simple manner to facilitate presentation. The solutions described here can be applied to more complicated formulations that involve nodes with multiple transmit/receive antennas as well as nodes capable of operating on multiple frequency channels. We refer the readers to [2, 53] for extended discussions.

Next we describe a decomposition-based optimization approach to solve the utility maximization problem (3.18). To gain some insights into the problem, we first consider the idealized scenario in which the capacity of the backhaul links is infinite. In this case the problem reduces to a resource management problem for the wireless access network only. We will show that for a large-family of utility functions, problem (3.18) can be solved effectively by BSUM algorithm. Using the insights obtained from this special case, we then generalize the approach to the full-fledged network provisioning problem with limited backhaul capacity.

### 3.3.2 Network with an uncapacitated backhaul

In this section, we consider a simplified network that has infinite backhaul capacity, and each user gets precisely a single flow; see Figure 3.4 for an illustration. In this case $\mathcal{M} = \mathcal{U}$, and each wireless link $\ell \in L^w$ carries a single flow, denoted as $r_\ell(v)$. The utility maximization problem (3.18) reduces to

$$\max_{v, r} \quad U ([r_\ell(v)]_{\ell \in L^w})$$

$$\text{s.t.} \quad \sum_{\ell \in \text{Out}(b) \cap L^w} |v_\ell|^2 \leq \bar{p}_b, \; \forall \; b \in B,$$

$$r_\ell(v) \leq \log \left(1 + \frac{|h_\ell|^2 |v_\ell|^2}{\sum_{n \in N(\ell) \setminus \{\ell\}} |h_{\ell n}|^2 |v_n|^2 + \sigma_\ell^2}\right), \; \forall \; \ell \in L^w.$$  \hspace{1cm} (3.19)
The sum rate maximization problem

For illustration purpose, in the following we specialize the utility function to be the well-known sum rate utility. The problem becomes

\[
\max_{\mathbf{v}} \ U(\mathbf{v}) = \sum_{\ell \in \mathcal{C}^w} \log \left( 1 + \frac{\sum_{n_{\ell} \in \mathcal{L}(\ell) \setminus \{\ell\}} |h_{\ell n}|^2 |v_{\ell n}|^2}{\sum_{n_{\ell} \in \mathcal{L}(\ell) \setminus \{\ell\}} |h_{\ell n}|^2 |v_{\ell n}|^2 + \sigma_n^2} \right)
\]

s.t. \[ \sum_{\ell \in \text{Out}(b) \cap \mathcal{C}^w} |v_{\ell}|^2 \leq \bar{p}_b, \ \forall \ b \in \mathcal{B}. \] (3.20)

This problem is precisely the block-structured problem discussed in Section 3.1. More specifically, it falls into the category of problem (3.2), where the precoder for a given BS \( b \), \( \{v_{\ell n} \in \text{Out}(b) \cap \mathcal{C}^w\} \), corresponds to a block variable \( x_i \) in (3.2).

The difficulty in solving problem (3.20) is quite obvious now: the variables \( v_{\ell n} \) are coupled in a nonlinear way in the objective through mutual interference, making the problem highly nonconvex. One may resort to general purpose algorithm such as gradient projection, but its dependence on stepsize as well as the requirement to perform projection make it difficult to implement for large-scale problems. What we propose here is to use the BSUM approach discussed in Section 3.2.3, in which approximate versions of the original problem are successively solved to progressively obtain improved solutions; see Figure 3.5 for the illustration of the algorithm.\(^1\) Clearly the key here is to find an appropriate lower bound of the objective function at any given point \( \tilde{\mathbf{v}} \), so that the resulting subproblem can be solved cheaply.

\(^1\) Note that here the problem is formulated in a maximization form, rather than the minimization form considered in Section 3.2. So the BSUM algorithm successively constructs and solves lower bounds as opposed to the upper bounds stated in Section 3.2.3.
Optimization algorithms for big data with application in wireless networks

Figure 3.5 Illustration of the BSUM algorithm. Superscript \((t)\) denotes the iteration number, \(g(v; v^{(t)})\) denotes the approximate function (lower bound) constructed at iteration \((t)\); \(v^{(t)}\)s are the iterates generated by the algorithm.

To this end, let us first introduce two useful quantities. For a given collection of precoder \(v\) and a given link \(\ell\), let us define \(c_\ell(v)\) and \(e_\ell(v)\) as

\[
\begin{align*}
gen_\ell(v) &= \sigma^2_{\ell} + \sum_{n \in \mathcal{I}(\ell)} |h_{\ell n}|^2 |v_n|^2 \\
e_\ell(v) &= 1 - c_\ell^{-1}(v)|h_{\ell \ell}|^2 |v_\ell|^2.
\end{align*}
\]

Here \(c_\ell(v)\) can be interpreted as the total signal plus interference power received by user \(d_\ell\), while \(e_\ell(v)\) is the minimum mean square error for decoding user \(d_\ell\)'s message (see, e.g., [46] for a more detailed explanation).

Our first lemma finds a lower bound \(f_\ell(v; \hat{v})\) for the rate \(r_\ell(v)\) at a given point \(\hat{v}\).

**Lemma 3.4** For any feasible solution \(\hat{v}\), define

\[
\hat{e}_\ell \triangleq 1 - c_\ell^{-1}(\hat{v})|h_{\ell \ell}|^2 |\hat{v}_\ell|^2.
\]

Then we have

\[
r_\ell(v) \geq r_\ell(\hat{v}) + \frac{1}{\hat{e}_\ell} \left( v_\ell^H h_{\ell \ell}^H c_\ell^{-1}(\hat{v}) h_{\ell \ell} v_\ell - 1 \right) + 1 \triangleq f_\ell(v; \hat{v}).
\]

To see why this result is true, let us first express \(r_\ell(v)\) as follows

\[
r_\ell(v) = \log \left( 1 + v_\ell^H h_{\ell \ell}^H \left( \sum_{n \in \mathcal{I}(\ell) \setminus \{\ell\}} |h_{\ell n}|^2 |v_n|^2 + \sigma^2_{\ell} \right)^{-1} h_{\ell \ell} v_\ell \right).
\]

Applying the following inversion lemma

\[
1 + ab^{-1}c = (1 - a(b + ac)^{-1}c)^{-1}
\]

we have

\[
r_\ell(v) = -\log \left( 1 - v_\ell^H h_{\ell \ell}^H c^{-1}_\ell(v) h_{\ell \ell} v_\ell \right).
\]
Now it is clear that the right-hand side (RHS) is a convex function on $\mathbf{1} - v^H H H \mathbf{c}^{-1}_e(v) h_{ee} v_e$. Therefore we can linearize the RHS at any given $\mathbf{v}$ and use the defining property of a convex function to obtain

$$
r_{\ell}(v) \geq r_{\ell}(\mathbf{v}) - \frac{(1 - v^H H H \mathbf{c}^{-1}_e(v) h_{ee} v_e) - (1 - \mathbf{v}^H H H \mathbf{c}^{-1}_e(\mathbf{v}) h_{ee} \mathbf{v}_e)}{1 - \mathbf{v}^H H H \mathbf{c}^{-1}_e(\mathbf{v}) h_{ee} \mathbf{v}_e}
\]

$$
= r_{\ell}(\mathbf{v}) + \frac{1}{\mathbf{e}_\ell} (v^H H H \mathbf{c}^{-1}_e(v) h_{ee} v_e - 1) + 1
\]

$$
\triangleq f_{\ell}(v; \mathbf{v}). \quad (3.28)
$$

Summing over all the links, we obtain

$$
U(v) = \sum_{\ell} r_{\ell}(v) \geq \sum_{\ell} f_{\ell}(v; \mathbf{v}) \triangleq f(v; \mathbf{v}). \quad (3.29)
$$

Unfortunately, the lower bound $f(v; \mathbf{v})$ obtained is not so useful yet, as it still couples all the variables and is again a nonconvex function w.r.t. the optimization variable $v$. What we will do next is to further construct a concave lower bound for $f(v; \mathbf{v})$, which in turn is a concave lower bound for $U(v)$.

**Lemma 3.5** For any given $\mathbf{v}$ we have

$$
v^H H H \mathbf{c}^{-1}_e(v) h_{ee} v_e \geq \mathbf{u}^H H h_{ee} \mathbf{u} + v^H H H \mathbf{c}_e(v) u_e - \mathbf{u}^H H \mathbf{c}_e(v) u_e \quad (3.30)
$$

where we have defined

$$
\mathbf{u} \triangleq \mathbf{c}^{-1}_e(v) h_{ee} \mathbf{v}_e. \quad (3.31)
$$

The proof of this result is again very simple. First note that the function $h(x, y) = \frac{x^2}{y}$ is jointly convex on $(x, y)$ over the domain $y > 0$. Thus using the property of the convex function, for any given tuple $(\hat{x}, \hat{y})$ with $\hat{y} > 0$ we have

$$
h(x, y) \geq h(\hat{x}, \hat{y}) + \frac{\partial h(\hat{x}, \hat{y})}{\partial x} (x - \hat{x}) + \frac{\partial h(\hat{x}, \hat{y})}{\partial y} (y - \hat{y}). \quad (3.32)
$$

Now applying this inequality to $v^H H H \mathbf{c}^{-1}_e(v) h_{ee} v_e$ with the identification that $y = \mathbf{c}^{-1}_e(v)$ and $x = v^H H H \mathbf{c}_e(v)$, we obtain the desired result

$$
v^H H H \mathbf{c}^{-1}_e(v) h_{ee} v_e \geq \mathbf{u}^H H h_{ee} \mathbf{u} + \mathbf{u}^H H h_{ee} \mathbf{u} - \mathbf{u}^H H \mathbf{c}_e(v) \mathbf{u} - \mathbf{u}^H H \mathbf{c}_e(v) \mathbf{u}
\]

$$
= \mathbf{u}^H H h_{ee} \mathbf{u} + v^H H H \mathbf{c}_e(v) \mathbf{u} - \mathbf{u}^H H \mathbf{c}_e(v) \mathbf{u}. \quad (3.33)
$$

As a result, combining Lemma 3.4 and Lemma 3.5, we have

$$
r_{\ell}(v) \geq f_{\ell}(v; \mathbf{v}) \geq \frac{1}{\mathbf{e}_\ell} (\mathbf{u}^H H h_{ee} \mathbf{u} + v^H H H \mathbf{c}_e \mathbf{u} - \mathbf{u}^H H \mathbf{c}_e \mathbf{u}) + 1 + r_{\ell}(\mathbf{v}) - \frac{1}{\mathbf{e}_\ell} \quad \text{concave quadratic function on } v \quad \text{constants} \quad (3.33)
$$
Using this bound, the objective of the sum rate maximization problem can be bounded below by

$$U(r) = \sum_{\ell} r_{\ell}(v) \geq \sum_{\ell} f_{\ell}(v; \hat{v})$$

$$\geq \sum_{\ell} \left( \frac{1}{\hat{e}_{\ell}} \left( \hat{u}_{\ell}^{H} h_{\ell} v_{\ell} + v_{\ell}^{H} h_{\ell}^{H} \hat{u}_{\ell} - \hat{u}_{\ell}^{H} c_{\ell}(v) \hat{u}_{\ell} \right) + 1 + r_{\ell}(\hat{v}) - \frac{1}{\hat{e}_{\ell}} \right)$$

concave quadratic function on $v$

constants

$$= \sum_{\ell} \left( \frac{1}{\hat{e}_{\ell}} \left( \hat{u}_{\ell}^{H} h_{\ell} v_{\ell} + v_{\ell}^{H} h_{\ell}^{H} \hat{u}_{\ell} - \sum_{n \in \mathcal{E}^{(\ell)}} \frac{1}{\hat{e}_{n}} (|\hat{u}_{n}|^{2} |h_{n\ell}|^{2} |v_{\ell}|^{2}) \right) + \text{constant} \right)$$

concave quadratic on $v_{\ell}$

$$\triangleq \sum_{\ell} g_{\ell}(v_{\ell}; \hat{v}) \triangleq g(v; \hat{v}) \quad (3.34)$$

The above two-layer convex approximation process is illustrated in Figure 3.6. From (3.34) it is clear that the lower bound function $g(v; \hat{v})$ is not only a concave quadratic function on $v$, but is also completely separable among $v_{i}$'s. Utilizing this favorable structure of the lower bound, we can successively minimize the lower bound function $g(v; \hat{v})$ by solving $|B|$ independent subproblems of the following form, one for each BS $b$:

$$\max_{\ell \in \text{Out}(b) \cap \mathcal{C}^{\ell}} g_{\ell}(v_{\ell}; \hat{v})$$

s.t. $\sum_{\ell \in \text{Out}(b) \cap \mathcal{C}^{\ell}} |v_{\ell}|^{2} \leq \bar{p}_{b}. \quad (3.35)$
The solution of problem (3.35) can be written down in closed form

\[
v^*_\ell = \left( \sum_{n \in \mathcal{Z}(t)} |\hat{u}_n|^2 |h_{n\ell}|^2 \frac{1}{\hat{e}_n} + \lambda_b \right)^{-1} h_{\ell\ell}^H \hat{u}_\ell \hat{e}_\ell^{-1}, \quad \forall \, \ell \in \text{Out}(b) \cap \mathcal{L}_w. \tag{3.36}
\]

where \(\lambda_b \geq 0\) is the Lagrangian multiplier used to guarantee the total power constraint for BS \(b\).

The overall algorithm is summarized in the following box, where we use the superscript \((t)\) to denote the iteration index. It can be easily checked that the lower bound \(g(v; \hat{v})\) satisfies Assumptions (A1)–(A5). By appealing to Theorem 3.3, this algorithm is guaranteed to converge to a stationary solution of problem (3.3.2).

1. Update \(\{\hat{u}_\ell\}\): \(\hat{u}_\ell \leftarrow c_\ell^{-1}(v^{(t)}(\ell)) h_{\ell\ell} v^{(t)} \).
2. Update \(\{\hat{e}_\ell\}\): \(\hat{e}_\ell \leftarrow 1 - c_\ell^{-1}(v^{(t)}(\ell)) h_{\ell\ell}^2 (v^{(t)}(\ell))^2\).
3. Update \(\{v_\ell\}\):

\[
v^{(t+1)}_\ell = \left( \sum_{n \in \mathcal{Z}(t)} |\hat{u}_n|^2 |h_{n\ell}|^2 \frac{1}{\hat{e}_n} + \lambda_b \right)^{-1} h_{\ell\ell}^H \hat{u}_\ell \hat{e}_\ell^{-1}, \quad \forall \, \ell \in \text{Out}(b) \cap \mathcal{L}_w, \forall \, b \in \mathcal{B}.
\]

4. Let \(t = t + 1\), go to step (1).

It is important to note that by exploring the hidden convexity of the rate function \(r_\ell(v)\), problem (3.20) can be solved with simple closed-form updates, and the computation can be further carried out in parallel by all the BSs. The algorithm described above is the so-called weighted minimum mean square error (WMMSE) algorithm, which is originally developed using certain equivalence argument between problem (3.20) and certain weighted MSE minimization problem [46, 54]. The preceding derivation based on BSUM is first given in [53], which provides an interesting alternative interpretation of the algorithm. We remark that the above analysis and the WMMSE algorithm can be easily generalized to networks with multi-antenna wireless nodes, or to problems having different (possibly nonsmooth) utility functions [55].

**The min rate maximization problem**

In this section we briefly discuss how the bounds derived in the previous section can be utilized to solve another popular problem – the min rate maximization problem – which results in a fair rate allocation. In particular, we are interested in maximizing the minimum rate achieved by all the users \(u \in \mathcal{U}\):

\[
\max_v \min_u \sum_{\ell: u = d_i} \log \left( 1 + \frac{|h_{\ell\ell}|^2 |v_\ell|^2}{\sum_{n \in \mathcal{Z}(t) \setminus \{\ell\}} |h_{n\ell}|^2 |v_n|^2 + \sigma^2_\ell} \right), \tag{3.37}
\]

s.t. \(\sum_{\ell \in \text{Out}(b) \cap \mathcal{L}_w} |v_\ell|^2 \leq \bar{p}_b, \forall \, b \in \mathcal{B}\).
First we introduce a new variable $r \geq 0$ and transform the above problem to the following equivalent form

$$\max_{v, r} r$$

subject to

$$\sum_{e u = d} \log \left( 1 + \frac{\sum_{n \in \mathcal{L}\setminus\{e\}} |h_{e n}|^2 |v_n|^2}{\sum_{n \in \mathcal{L}\setminus\{e\}} |h_{e n}|^2 |v_n|^2 + \sigma_e^2} \right) \geq r, \ \forall u \in \mathcal{U} \quad (3.38)$$

$$\sum_{e \in \text{Out}(b) \cap \mathcal{L}_{=1}} |v_e|^2 \leq \bar{p}_b, \ \forall b \in \mathcal{B}.$$ 

The transformed problem now has a simple linear objective, but a set of difficult rate constraints. Once again we can utilize the bounds developed in the previous section, but to successively approximate the feasible set instead. That is, for a given $\tilde{v}$, we first compute $\tilde{u}_e$ and $\tilde{e}_e$ just as before, and then solve the following convex problem

$$\max_{v, r} r$$

subject to

$$\sum_{e u = d} g_e(v; \tilde{v}) \geq r, \ \forall u \in \mathcal{U} \quad (3.39)$$

$$\sum_{e \in \text{Out}(b) \cap \mathcal{L}_{=1}} |v_e|^2 \leq \bar{p}_b, \ \forall b \in \mathcal{B}.$$ 

Unlike the original problem (3.38), this problem is now a convex problem as it has a convex, albeit smaller, feasible set. The overall algorithm is presented in the following box.

1. Update $\{\tilde{u}_e\}$: $\tilde{u}_e \leftarrow c_e^{-1}(v^{(t)})h_{e e}v_e^{(t)}$.
2. Update $\{\tilde{e}_e\}$: $\tilde{e}_e \leftarrow 1 - c_e^{-1}(v^{(t)})h_{e e}|v_e^{(t)}|^2$.
3. Update $\{v_e\}$: obtain $v_e^{(t+1)}$ by solving problem (3.39).
4. Let $t = t + 1$, go to step (1).

We note that the above algorithm is not a special case of BSUM, because it is the feasible set that has been approximated here. Therefore the previous analysis of BSUM in Section 3.2.3 does not apply. Fortunately by carefully studying the optimality conditions of the resulting subproblems, one can still show that the iterates $\{v^{(t)}\}$ converge to the set of stationary solutions of problem (3.37); see [56] for detailed analysis.

At this point it should be noted that the subproblem for solving $v$ is convex but does not have closed-form solution. Therefore general purpose solvers need to be used repeatedly for this subproblem, which can be computationally expensive when the problem size becomes large (i.e. large number of BSSs, flows, users, etc.). Later when we discuss the general network provisioning problem, we will revisit this issue and design an efficient algorithm for solving the related subproblem.
3.3.3 Network with a capacitated backhaul

Now we are ready to solve the problem posed in Section 3.3.1 in the setting of large-scale cloud-based RAN. Without loss of generality, we let $v_\ell \in \mathbb{R}$ for all $\ell$. We focus on the following per-flow min rate maximization problem

\[
\max_{v, r} r \quad \text{s.t.} \quad r \geq 0, r(m) \geq r, \ m \in \mathcal{M} \\
\sum_{m \in \mathcal{M}} r_\ell(m) \leq C_\ell, \ \forall \ell \in \mathcal{L}^w \tag{3.40b} \\
\sum_{m \in \mathcal{M}} r_\ell(m) \leq r_\ell(v) = \log \left( 1 + \frac{|h_\ell|^2 v_\ell^2}{\|h_\ell\|^2 v_n^2 + \sigma_\ell^2} \right), \ \forall \ell \in \mathcal{L}^w \tag{3.40c} \\
\sum_{\ell \in \text{In}(v)} r_\ell(m) + 1_{s(m)}(v)r(m) = \sum_{\ell \in \text{Out}(v)} r_\ell(m) + 1_{d(m)}(v)r(m), \ \forall m \in \mathcal{M}, \ \forall v \in \mathcal{V}, \tag{3.40d} \\
\sum_{\ell \in \text{Out}(b) \cap \mathcal{L}^w} v_\ell^2 \leq \tilde{p}_b, \ \forall b \in \mathcal{B}. \tag{3.40e} \\
\]

Here with a little abuse of notation, we have defined

\[r \triangleq [r, \{r(m), r_\ell(m) | \ell \in \mathcal{L} \}_{m \in \mathcal{M}}]^T.\]

The constraints (3.40c)–(3.40f) are, respectively, the wired link capacity constraints, the wireless link capacity constraints, the flow conservation constraints and the BS power budget constraint introduced in Section 3.3.1.

**The N-MaxMin algorithm**

In practice, problem (3.40) needs to be solved frequently to determine the dynamic resource and flow allocation. However, this is very challenging because:

- the problem is nonconvex due to the wireless rate constraints (3.40d);
- the design variables $v$ and $r$ are tightly coupled through the rate expressions; and
- the size of the problem can be huge.

To obtain an effective algorithm, our first step is again to approximate the rate $r_\ell(v)$ using its lower bound. To this end, let us simplify the expression for $g_\ell(v; \hat{v})$ in (3.34) by the following:

\[g_\ell(v; \hat{v}) = \hat{c}_{1,\ell} v_\ell + \hat{c}_{2,\ell} v_n - \sum_{n \in \mathcal{N}(\ell)} \hat{c}_{3,n} v_n^2, \tag{3.41} \]
where the constants \((\hat{c}_{1,\ell}, \hat{c}_{2,\ell}, \hat{c}_{3,\ell n})\) are given by

\[
\begin{align*}
\hat{c}_{1,\ell} &= 1 + r_\ell(\hat{v}) - \frac{1}{\hat{e}_\ell}(1 + \sigma_\ell^2 \hat{u}_\ell^2) \\
\hat{c}_{2,\ell} &= \frac{2}{\hat{e}_\ell} \hat{u}_\ell |h_{\ell\ell}| \\
\hat{c}_{3,\ell n} &= \frac{1}{\hat{e}_\ell} \hat{u}_\ell^2 |h_{\ell n}|^2.
\end{align*}
\] (3.42)

(3.43)

(3.44)

Then at any given point \(\hat{v}\), we can approximate problem (3.40) by

\[
\begin{align*}
\max_{v,r} & \quad r \\
\text{s.t.} & \quad (3.40b), (3.40c), (3.40e), (3.40f), \\
& \quad \sum_{m \in M} r_\ell(m) \leq g_\ell(v; \hat{v}) = \hat{c}_{1,\ell} + \hat{c}_{2,\ell} v_\ell - \sum_{n \in \mathcal{L}(\ell)} \hat{c}_{3,\ell n} v_n^2, \forall \ell \in \mathcal{L}^\text{in}.
\end{align*}
\] (3.45a)

(3.45b)

(3.45c)

Similarly as solving the max-min problem in Section 3.3.2, the above problem is again convex and can be solved by using general-purpose solvers. The resulting algorithm, termed the network max-min WMMSE (N-MaxMin) algorithm, is given in the following box. Again one can show that this algorithm converges to the set of stationary solutions of the network provisioning problem (3.40); see [2] for detailed analysis.

1. Update \([\hat{u}_\ell]\): \(\hat{u}_\ell \leftarrow c_\ell^{-1}(v^{\ell(0)})|h_{\ell\ell}|v_\ell^{(0)}\).
2. Update \([\hat{e}_\ell]\): \(\hat{e}_\ell \leftarrow 1 - c_\ell^{-1}(v^{\ell(0)})|h_{\ell\ell}|^2 (v_\ell^{(0)})^2\).
3. Update \((v, r)\): obtain \((v^{\ell+1}, r^{\ell+1})\) by solving problem (3.45).
4. Let \(t = t + 1\), go to step (1).

Once again, the computation of \(\hat{u}_\ell\)'s and \(\hat{e}_\ell\)'s is in closed form. The main computational complexity is in step (3) where \((v, r)\) are updated. When the number of variables and constraints are large, the efficiency of the entire algorithm critically depends on the implementation of this step. How this can be done is the topic that we address in the following section.

**An ADMM approach for updating \((v, r)\)**

We propose to use the ADMM algorithm for solving problem (3.45). ADMM is chosen because it allows us to implement a highly parallelizable algorithm that fits ideally to the cloud-based architecture of the next-generation wireless networks.

In order to apply the ADMM, the first step is to formulate problem (3.45) into the form of (3.6). Our main approach is to properly split the variables in the coupling constraints (3.40e) and (3.45c), so that these constraints decompose nicely over the variables.

Let us first look at the flow conservation constraint (3.40e), restated below for convenience:

\[
\sum_{\ell \in \text{in}(v)} r_\ell(m) + 1_{s(m)}(v)r(m) = \sum_{\ell \in \text{out}(v)} r_\ell(m) + 1_{d(m)}(v)r(m), \quad \forall v, m.
\]
Figure 3.7 Illustration of flow rate splitting.

Figure 3.8 Illustration of link rate splitting.

At first sight it appears that the link rate variables \( \{ r(m) \} \) are all tightly coupled together, making (3.40e) a very difficult constraint to satisfy. However, a careful study reveals that each link rate \( r(m) \) appears exactly twice, in the constraints for node \( s(m) \) and \( d(m) \). It follows that if we introduce two copies of \( r(m) \) (denoted as \( r^{s(e)}(m) \) and \( r^{d(e)}(m) \)) and use \( r^{s(e)}(m) \) (resp. use \( r^{d(e)}(m) \)) in the constraint for node \( s(m) \) (resp. for node \( d(m) \)), then each of these new auxiliary variables only appears in a single flow conservation constraint. The same is true for the flow rate variable \( r(m) \): they appear only twice, in those constraints defined by the source node \( s(m) \) and the destination node \( d(m) \). Similarly we introduce two variables \( r^{s(m)} \) and \( r^{d(m)} \) for each flow \( m \), and use them in the constraints for the source and the destination node of flow \( m \), respectively. See Figure 3.7 and Figure 3.8 for illustrations of the above splitting process.

Mathematically, we have introduced the following auxiliary variables:

\[
\hat{r}_e^{s(e)}(m) = r_e^{s(e)}(m), \quad \hat{r}_e^{d(e)}(m) = r_e^{d(e)}(m), \quad \forall \ e \in \mathcal{E}, \ m \in \mathcal{M};
\]

\[
\hat{r}^{s(m)} = r^{s(m)}, \quad \hat{r}^{d(m)} = r^{d(m)}, \quad \forall \ m \in \mathcal{M}.
\]

We have also modified the flow rate conservation constraints to

\[
\sum_{e \in \text{In}(v)} \hat{r}_e^{s(e)}(m) + 1_{s(m)}(v) \hat{r}^{s(m)} = \sum_{e \in \text{Out}(v)} \hat{r}_e^{d(e)}(m) + 1_{d(m)}(v) \hat{r}^{d(m)}, \quad \forall \ m, v.
\]

To facilitate analysis, we also split \( r \) by introducing \( \hat{r} \) that satisfies \( r = \hat{r} \). Let us collect the new variables and define

\[
\hat{r} \triangleq \left[ \hat{r}, \left\{ \hat{r}(m)^{s(m)}, \hat{r}_e^{s(e)(m)} \mid e \in \mathcal{E} \right\}_{m=1}^M, \left\{ \hat{r}(m)^{d(m)}, \hat{r}_e^{d(e)(m)} \mid e \in \mathcal{E} \right\}_{m=1}^M \right]^T.
\]
Next let us look at the rate constraint (3.45c), restated below for convenience
\[ \sum_{m \in \mathcal{M}} r_{\ell}(m) \leq \hat{c}_{1,\ell} + \hat{c}_{2,\ell} v_{\ell} - \sum_{n \in \mathcal{I}(\ell)} \hat{c}_{3,\ell n} v_{n}^2, \quad \forall \ell \in \mathcal{L}^{wl}. \] (3.48)

Again these constraints are coupled because each variable \( v_n \) appears in multiple constraints, i.e. constraints for the links that are interfered by \( n \). To decouple the constraints, we introduce several copies of the transmit precoders for each \( v_n \), denoted by \( \hat{v}_{\ell n} \), one for each link \( \ell \) interfered by \( n \) (i.e. for nodes \( \ell \) satisfying \( n \in \mathcal{I}(\ell) \)). By doing so, each variable \( \hat{v}_{\ell n} \) appears only in a single constraint. See Figure 3.9 for an illustration of the splitting process.

Formally, we have introduced a set of new variables
\[ \hat{v}_{\ell n} = v_n, \quad \forall \ell \text{ such that } n \in \mathcal{I}(\ell), \forall n \in \mathcal{L}^{wl}. \] (3.49)

We have also modified the rate constraints to
\[ \sum_{m=1}^{M} r_{\ell}(m) \leq \hat{c}_{1,\ell} + \hat{c}_{2,\ell} \hat{v}_{\ell \ell} - \sum_{n \in \mathcal{I}(\ell)} \hat{c}_{3,\ell n} \hat{v}_{\ell n}^2, \quad \forall \ell \in \mathcal{L}^{wl}. \] (3.50)

For notational convenience, define
\[ \hat{\mathcal{I}}(\ell) \triangleq \{ n \mid \ell \in \mathcal{I}(n) \} \]
with
\[ \tilde{\mathcal{I}}(\ell) \triangleq \{ n \mid \ell \in \tilde{\mathcal{I}}(n) \}. \]

being the set of wireless links interfered by \( \ell \).

By utilizing the new variables introduced so far, problem (3.45) is equivalently expressed as
\[
\begin{align*}
\max \quad & (r + \hat{r})/2 \\
\text{s.t.} \quad & (3.40b), \ (3.40c), \ (3.40f), \ (3.47), \ (3.50), \\
& (3.46), \ (3.49), \ \text{and } r = \hat{r}. \\
\end{align*}
\] (3.51)

We will see shortly that the above equivalent formulation decomposes the constraints (except the linear equality constraints \( r = \hat{r} \), (3.46) and (3.49)) between the variable sets
Table 3.1 The ADMM based algorithm for (3.51)

<table>
<thead>
<tr>
<th>Algorithm 1</th>
</tr>
</thead>
</table>

1. Initialize all primal variables $r^{(0)}$, $\tilde{r}^{(0)}$, $v^{(0)}$, $\tilde{v}^{(0)}$ (not necessarily a feasible solution for (3.51)), and all dual variables $\delta^{(0)}$, $\theta^{(0)}$; set $t = 0$.

2. Repeat.

3. Solve the following problem and obtain $r^{(t+1)}$, $\tilde{r}^{(t+1)}$:

$$\max_{r, \tilde{r}, v, \tilde{v}} L_{\rho_1, \rho_2}(r, \tilde{r}, v, \tilde{v}, \delta^{(t)}, \theta^{(t)})$$

subject to (3.40c), (3.40b), and (3.50).

4. Solve the following problem and obtain $\tilde{r}^{(t+1)}$, $v^{(t+1)}$:

$$\max_{\tilde{r}, v} L_{\rho_1, \rho_2}(\tilde{r}^{(t+1)}, v^{(t+1)}, \tilde{r}, v, \delta^{(t)}, \theta^{(t)})$$

subject to (3.17) and (3.47).

5. Update the Lagrange dual multipliers $\delta^{(t+1)}$ and $\theta^{(t+1)}$ by

$$\delta^{(t+1)} = \delta^{(t)} - \rho_1 (\tilde{r}^{(t+1)} - Cr^{(t+1)}),$$

$$\theta^{(t+1)} = \theta^{(t)} - \rho_2 (Dv^{(t+1)} - v^{(t+1)}).$$

6. $t = t + 1$.

7. Until Desired stopping criterion is met.

(r, $\tilde{v}$) and ($\tilde{r}$, v). In particular, we can write the linear equalities $r = \tilde{r}$, (3.46) and (3.49) as $Cr = \tilde{r}$, $Dv = v$ with

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix}^T; \quad D = \text{blkdiag}([1_{|E|}]_{E=E^c}),$$

where $1_{|E|}$ is an all one column vector of size equal to $|E|$.

Now ADMM can be used to solve (3.51), where the linear equality constraints $Cr = \tilde{r}$, $Dv = v$ are to be relaxed and penalized in the augmented Lagrangian. To write down the ADMM iteration, let us use $\delta$, $\{\delta^{ni}(m), \delta^{di}(m)\}$, $\{\delta^{ni(m)}, \delta^{di(m)}\}$, and $\{\theta_{nE}\}$ to denote the Lagrangian multipliers for equality constraints $r = \tilde{r}$, (3.46a), (3.46b), and (3.49), respectively. Collect the multipliers in the vectors $\delta$ and $\theta$. Let $\rho_1 > 0$ and $\rho_2 > 0$ denote the dual stepsizes. Then the partial augmented Lagrangian for problem (3.51) is given by

$$L_{\rho_1, \rho_2}(r, \tilde{r}, v, \delta, \theta) = \frac{(r + \tilde{r})}{2} + \left[ \delta^T (\tilde{r} - Cr) - \frac{\rho_1}{2} \|	ilde{r} - Cr\|^2 \right] + \left[ \theta^T (Dv - v) - \frac{\rho_2}{2} \|Dv - v\|^2 \right].$$

relaxing (3.46) relaxing (3.49)

The resulting algorithm, named Algorithm 1, is described in Table 3.1. The convergence of this algorithm to the optimal solutions of problem (3.51) (hence the original
subproblem \((3.45)\) for \((v, r)\) is readily implied by the standard analysis of ADMM (cf. Theorem 3.2). In the appendix, we provide guidelines on solving the two primal subproblems \((3.52)\) and \((3.53)\). Our focus is given to demonstrating the distinctive feature of these subproblems, that they naturally decompose into a series of independent small subproblems, which can be solved easily and in parallel.

**Implementation of Algorithm 1 in cloud-based RAN**

We reiterate here that each step of the N-MaxMin algorithm is in closed form. Further, the computation can be completely distributed to each node or link of the network. For the detailed discussion on the distributed implementation we refer the readers to [2, Sec. III]. However, in a cloud-based RAN, it is more desirable that a few cloud centers handle the computation centrally, each of them taking care of a subset of nodes located in a specific geographical zone. The key question here is whether the proposed algorithm can also be used in this scenario. Below we show that a properly modified version of Algorithm 1 does the trick. For simplicity, we will only focus on the backhaul network, but the extension for incorporating the wireless links follows the same idea.

Let us revisit the variable splitting in \((3.46)\), which is introduced to decompose the flow conservation constraints \((3.16)\) into each node. We assume that the set of nodes \(\mathcal{V}\) is partitioned into \(\mathcal{Z}\) non-overlapping zones, and \(v \in \mathcal{Z}_i\) if node \(v\) is within the \(i\)th zone. We modify the variable splitting procedure \((3.46a)\) as follows:

\[
\begin{align*}
\hat{r}_i^v(m) &= r_i(m), \quad \forall s_i \in \mathcal{Z}_i, d_i \in \mathcal{Z}_j, \quad i \neq j, \quad m = 1 \sim M, \\
\hat{r}_i^d(m) &= r_i(m), \quad \forall s_i \in \mathcal{Z}_i, d_i \in \mathcal{Z}_j, \quad i = j, \quad m = 1 \sim M.
\end{align*}
\]

That is, we only split the link rates on the bordering links.

Given this new variable splitting method, the flow conservation constraints for each node within zone \(i\) becomes

\[
\begin{align*}
\sum_{l \in \text{In}(v)} \left(1_{U_{s_l, d_l}}(s_i)\hat{r}_i^v(m) + 1_{Z_j}(s_i)\hat{r}_i(m)\right) + 1_{s(m)}(v)\hat{r}(m)^v
\end{align*}
\]

\[
= \sum_{l \in \text{Out}(v)} \left(1_{U_{s_l, d_l}}(d_i)\hat{r}_i^d(m) + 1_{Z_j}(d_i)\hat{r}_i(m)\right) + 1_{d(m)}(v)\hat{r}(m)^v,
\]

\[
\forall v \in \mathcal{Z}_i, \forall m = 1 \sim M.
\]

We can observe that the variables are now decoupled over each zone of nodes instead of each node.

With this new variable splitting we can again apply the ADMM. The resulting algorithm has closed-form updates except for the step related to the flow conservation constraints (i.e. the corresponding subproblem \((3.63)\)). This step now is decomposable into each zone. To describe the subproblem in detail, let us first introduce the following sets of links

\[
\begin{align*}
\text{BD}_i & \triangleq \{l \in \mathcal{L} \mid \forall s_l \in \mathcal{Z}_i, \quad d_l \in \mathcal{Z}_k, \quad \text{or} \quad \forall d_l \in \mathcal{Z}_i, \quad s_l \in \mathcal{Z}_k, \quad k \neq i\} \quad \text{(bordering links)} \\
\text{IT}_i & \triangleq \{l \in \mathcal{L} \mid \forall s_l, d_l \in \mathcal{Z}_i\} \quad \text{(interior links)}.
\end{align*}
\]
Then the per-zone subproblem can be explicitly expressed as the following quadratic problem with a set of linear constraints:

\[
\begin{align*}
\min & \sum_{l \in \mathcal{B}_v} \left( \hat{r}_l^v(m) - r_l(m) - \frac{\delta_l^v(m)}{\rho_1} \right)^2 \\
& + \sum_{l \in \mathcal{I}_v} \left( \hat{r}_l(m) - r_l(m) - \frac{\delta_l^v(m)}{\rho_1} \right)^2 + 1_{\{S(m),D(m)\}}(v) \left( \hat{r}_m^v - r_m - \frac{\delta_m^v}{\rho_1} \right)^2 \\
\text{s.t.} & \quad (3.57).
\end{align*}
\] (3.58)

Although problem (3.58) does not have an easy closed-form solution, it can be solved efficiently in a centralized way via well-known network optimization algorithms such as the relax code [57]. Moreover, when each zone has a single node, the above modified algorithm reduces to the original Algorithm 1.

We emphasize that this modified approach is particularly suitable for the cloud based RAN architecture, where the computation is distributed to each cloud center. One additional benefit offered by this zone-based algorithm is that the splitting procedure is performed less frequently, leading to far fewer number of slack variables. Therefore compared with original Algorithm 1, the modified approach also enjoys faster convergence speed (measured by the number of iterations). This will be demonstrated in the subsequent numerical experiments.

3.4 Numerical results

In this section, we report some numerical results on the performance of the proposed algorithms. Most of the numerical experiments are conducted on a network with 57 BSs and 11 network routers; see Figure 3.10 for an illustration of the network. The detailed specification of the network is given below.

(1) The backhaul network Each link \( \ell \in \mathcal{L}_w \) is bidirectional, and the capacities for both directions are the same. Detailed link capacities are given below:

- links between routers and those between gateway BSs and the routers: 1 (Gnats/s);
- 1-hop to the gateways: 100 (Mnats/s);
- 2-hop to the gateways: [10,50] (Mnats/s);
- 3-hop to the gateways: [2,5] (Mnats/s);
- more than 4-hop to the gateways: 0 (nats/s).

(2) The wireless access network We use a slightly more general model in which the BSs can also operate on \( K = 3 \) subchannels, each with 1 MHz bandwidth. The power budget for each BS is chosen equally by \( p = p_s, \forall s \in \mathcal{B} \), and \( \sigma_l = 1, \forall l \in \mathcal{L} \). The wireless links follow the Rayleigh distribution with \( \mathcal{CN}(0, (200/\text{dist})^3) \), where “dist” represents the distance between BS and the corresponding user. The source (destination) node of each commodity is randomly selected from network routers (mobile users), and all simulation results are averaged over 100 randomly
selected end-to-end commodity pairs. Below we refer to one round of the N-MaxMin iteration as an outer iteration, and one round of Algorithm 1 in Table 3.1 for solving \((r, v)\) as an inner iteration.

3.4.1 Scenario 1: Performance comparison with heuristic algorithms

In the first experiment, we assume that each mobile user can be served by the BSs within 300 m radius. Further, each user is interfered by all BSs in the network. For this problem, the parameters of N-MaxMin algorithm are set to be \(\rho_1 = 0.1\) and \(\rho_2 = 0.1, 0.05,\) and \(0.01\) for, respectively, \(p = 0\) dB, 10 dB, and 20 dB. The termination criteria are

\[
\frac{(r^{(l+1)} + \hat{r}^{(l+1)}) - (r^{(l)} + \hat{r}^{(l)})}{r^{(l)} + \hat{r}^{(l)}} < 10^{-3}
\]

\[
\max\{\|Cr^{(l)} - \hat{r}^{(l)}\|_{\infty}, \|(Dv^{(l)})^2 - (\hat{v}^{(l)})^2\|_{\infty}\} < 5 \times 10^{-4},
\]

where \((\cdot)^2\) represents elementwise square operation.

For comparison purposes, the following two heuristic algorithms are considered.

- **Heuristic 1 (greedy approach)**
  We assume that each mobile user is served by a single BS on a specific frequency tone. For each user, we pick the BS and channel pair that has the strongest channel as its serving BS and channel. After BS–user association is determined, each BS uniformly allocates its power budget to the available frequency tones as well as to the served users on each tone. With the obtained power allocation and BS–user association, the
capacity of all wireless links are available and fixed. Therefore the min rate of all commodities can be maximized by solving a wireline routing problem.

- **Heuristic 2 (orthogonal wireless transmission)**
  For the second heuristic algorithm, each BS uniformly allocates its power budget to each subchannel. To obtain a tractable problem formulation, we further assume that each active wireless link is interference free. Hence, each wireless link rate constraint now becomes convex. To impose this interference free constraint, additional variables $\beta_l \in \{0, 1\}, \forall l \in \mathcal{L}^{wl}$ are introduced, where $\beta_l = 1$ if wireless link $l$ is active, otherwise $\beta_l = 0$. In this way, there is no interference on wireless link $l$ if $\sum_{n \in l(i)} \beta_n = 1$. To summarize, we solve the following optimization problem:

$$\max_{\beta} r$$

s.t. $\sum_{m=1}^{M} r_l(m) \leq \beta_l \log \left( 1 + \frac{|h_l|^2 P_n / K}{\sigma_d^2} \right)$,

$$\sum_{n \in l(i)} \beta_n = 1, \beta_l \in \{0, 1\}, \forall l, n \in \mathcal{L}^{wl},$$

(3.40b), (3.40c), and (3.40e).

Since the integer constraints on $\{\beta_l | \forall l \in \mathcal{L}^{wl}\}$ are also intractable, we relax it to $\beta_l = [0, 1]$. In this way the problem becomes a large-scale LP, whose solution represents an upper bound value of this heuristic.

In Figure 3.11, we show the min rate performance of different algorithms for different numbers of commodities and power budget. We observe that the minimum rates achieved by the N-MaxMin algorithm are more than twice of those achieved by the heuristic algorithms.
Optimization algorithms for big data with application in wireless networks

3.4.2 Scenario 2: The efficiency of N-MaxMin WMMSE algorithm

In the second set of numerical experiments, we evaluate the proposed N-MaxMin algorithm using different number of commodity pairs and different power budgets at the BSs. Here we use the same settings as in the previous experiment, except that all mobile users are interfered by the BSs within a distance of 800 m, and that we set $p_2 = 0.005$ (resp. $p_2 = 0.001$) when $p = 10$ dB (resp. $p = 20$ dB). The min rate performance for the N-MaxMin algorithm and the required number of inner iterations are plotted in Figure 3.12. Owing to the fact that the obtained $(r, v)$ is far from the stationary solution in the first few outer iterations, there is no need to complete the inner Algorithm 1 at the very beginning. Hence, we limit the number of inner iterations to be no more than 500 for the first five outer iterations. After the early termination of the inner algorithm, we use the obtained $v$ to update $\{\hat{u}_i\}_i$ and $\{\hat{e}_i\}_i$.
Figure 3.13 The considered network consists of 114 BSs and 11 routers with the locations and the connectivity between these nodes. Each computation core is responsible for one group of nodes shown in the figure.

In Figure 3.12(a), (b), we see that when $p = 10$ dB, the min rate converges at about the tenth outer iteration when the number of commodities is up to 30, while less than 500 inner iterations are needed per outer iteration. Moreover, after the tenth outer iteration, the number of inner ADMM iterations reaches below 100. In Figure 3.12(c), (d), the case with $p = 20$ dB is considered. Clearly the required number of outer iterations is slightly larger than that in the case of $p = 10$ dB, since the objective value and the feasible set are both larger. However, in all cases the algorithm still converges fairly quickly.

3.4.3 Scenario 3: Multi-commodity routing problem with parallel implementation

In this set of numerical experiments, we demonstrate how parallel implementation can speed up the inner Algorithm 1 considerably. To illustrate the benefit of parallelization, we consider a larger network (see Figure 3.13) which is derived by merging two identical BS networks shown in Figure 3.10. The new network consists of 126 nodes (12 network routers and 114 BSs).

For simplicity, we removed all the wireless links, so constraints (3.40d) and (3.40f) of problem (3.40) are absent. This reduces problem (3.40) to a network flow problem (a very large linear program).

We implement Algorithm 1 using the Open MPI package, and compare its efficiency with the commercial LP solver, Gurobi [58]. For the Open MPI implementation, we use nine computation cores for each set of network nodes as illustrated in Figure 3.13.
Table 3.2 Comparison of computation time used by different implementations of the ADMM approach for the routing only problem. The size of the problems solved are specified using a range of metrics (total number of commodities, variables and constraints).

<table>
<thead>
<tr>
<th># of Commodities</th>
<th>50</th>
<th>100</th>
<th>300</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Variables</td>
<td>1.4x10^4</td>
<td>2.9x10^4</td>
<td>8.7x10^4</td>
</tr>
<tr>
<td># of Constraints</td>
<td>2.1x10^4</td>
<td>4.2x10^4</td>
<td>1.3x10^5</td>
</tr>
<tr>
<td>Time (s) (Sequential)</td>
<td>1.04</td>
<td>2.03</td>
<td>8.53</td>
</tr>
<tr>
<td>Time (s) (Parallel)</td>
<td>0.20</td>
<td>0.37</td>
<td>1.10</td>
</tr>
<tr>
<td>Time (s) (Gurobi)</td>
<td>0.20</td>
<td>0.64</td>
<td>2.51</td>
</tr>
</tbody>
</table>

Figure 3.14 The considered network consists of 57 BSs and 11 routers with the locations and the connectivity between these nodes. Each zone of nodes is labeled by one group of nodes shown in the figure.

We choose $\rho_1 = 0.01$ and let the BSs serve as the destination nodes for commodities. Table 3.2 compares the computation time required for different implementation of Algorithm 1 and that of Gurobi. We observe that parallel implementation of the ADMM approach leads to more than five-fold improvement in computation time. We also note that when the problem size increases, the performance of Gurobi becomes worse than that achieved by the parallel implementation of Algorithm 1. Thus, the proposed algorithm (implemented in parallel) appears to scale nicely to large problem sizes.

---

2 All the computation is performed on a SunFire X4600 server with AMD Opteron 8356 2.3 GHz CPUs.
Scenario 4: Performance evaluation for Algorithm 1 with zones of nodes

In the last set of numerical experiments, the advantage of applying the modified Algorithm 1 with predefined zones of nodes will be demonstrated (cf. Section 3.3.3). In Figure 3.14, we provide the considered mesh network with 57 BSs and 11 network routers where the light gray lines label each zone of the nodes. The parameter $\rho_1$ of ADMM is set to be 0.001. In Figure 3.15, the CDF of the required number of iterations is illustrated for Algorithm 1 that decomposes to nodes or zones. One can easily observe that using the modified Algorithm 1 with predetermined zones of nodes, the number of ADMM iterations can be greatly decreased. This is because fewer slack variables are introduced for the zone-based implementation.

3.5 Appendix

In the appendix, we provide guidelines for solving subproblems (3.52) and (3.53).

Solving subproblem (3.52)

A closer look at this problem reveals that it naturally decomposes over the following three sets of variables

$$\{r, \{r(m)\}_{m \in M}\}, \{\{r_\ell(m)\}_{m \in M} \mid \forall \ell \in L^w\}, \tilde{\chi} \cup \{\{r_\ell(m)\}_{m \in M} \mid \forall \ell \in L^w\}.$$ 

Note that the first two subblocks only have to do with the wired links, while the last subblock corresponds to the variables over the wireless link. In the following we provide
explicit forms of these subproblems. We refer the interested readers to [2, Appendix B] for detailed expressions for their solutions.

(i) **Subproblem for** \{\{r, \{r(m)\}_{m \in \mathcal{M}}\}

This subproblem updates the current minimum flow rate among all commodities. It takes the following form:

\[
\max \frac{r}{2} - \frac{\rho_1}{2} \left( \hat{r} - r - \frac{\delta}{\rho_1} \right)^2 - \frac{\rho_1}{2} \sum_{m \in \mathcal{M}} \sum_{v \in \{s(m), d(m)\}} \left( \hat{r}(m) - r(m) - \frac{\delta^v_m}{\rho_1} \right)^2
\]

\[
\text{s.t. } r \geq 0, \ r(m) \geq r, \ m \in \mathcal{M}.
\]

(3.59)

This problem is a quadratic problem with simple nonnegativity constraints. By checking the first-order optimality condition, its solution can be written down in (semi)closed form.

(ii) **Subproblem for** \{\{r_{\ell(m)}\}_{m \in \mathcal{M}} \mid \forall \ell \in \mathcal{L}^w\}

The problem takes the following form:

\[
\min \sum_{\ell \in \mathcal{L}^w} \sum_{m \in \mathcal{M}} \sum_{v \in \{s, d\}} \left( \hat{r}_{\ell}^v(m) - r_{\ell}(m) - \frac{\delta^v_{\ell}(m)}{\rho_1} \right)^2
\]

\[
\text{s.t. } \sum_{m \in \mathcal{M}} r_{\ell}(m) \leq C_\ell, \ r_{\ell}(m) \geq 0, \ m \in \mathcal{M}, \ \ell \in \mathcal{L}^w.
\]

(3.60)

Obviously it can be further decomposed into \(|\mathcal{L}^w|\) subproblems, expressed below, one for each link \(\ell \in \mathcal{L}^w\)

\[
\min \sum_{m \in \mathcal{M}} \sum_{v \in \{s, d\}} \left( \hat{r}_{\ell}^v(m) - r_{\ell}(m) - \frac{\delta^v_{\ell}(m)}{\rho_1} \right)^2
\]

\[
\text{s.t. } \sum_{m \in \mathcal{M}} r_{\ell}(m) \leq C_\ell, \ r_{\ell}(m) \geq 0, \ m \in \mathcal{M}.
\]

This problem is a quadratic problem with a single linear inequality constraint and a number of simple nonnegativity constraints. Its solution can be again written down in closed form.

(iii) **Subproblem for** \(\hat{v} \cup \{r_{\ell(m)}\}_{m \in \mathcal{M}} \mid \forall \ell \in \mathcal{L}^{wl}\)

This problem is given by

\[
\min \frac{\rho_1}{2} \sum_{\ell \in \mathcal{L}^{wl}} \sum_{m \in \mathcal{M}} \sum_{v \in \{s, d\}} \left( \hat{r}_{\ell}^v(m) - r_{\ell}(m) - \frac{\delta^v_{\ell}(m)}{\rho_1} \right)^2 + \frac{\rho_2}{2} \sum_{n \in \mathcal{I}(\ell)} \left( v_n - \hat{v}_{ln} - \frac{\theta_{\ell n}}{\rho_2} \right)^2
\]

\[
\text{s.t. } r_{\ell}(m) \geq 0, \ m \in \mathcal{M}, \text{ and } (3.50).
\]

First note that the objective as well as the constraints in (3.50) are separable among the wireless links. Further due to variable splitting, each variable \(\hat{v}_{ln}\) and \(r_{\ell}(m)\) only appears in a single constraint in (3.50). Consequently the above problem can
be decomposed into \(|L^w|\) subproblems, one for each wireless link \(\ell \in L^w\):

\[
\min \frac{\rho_1}{2} \sum_{m \in \mathcal{M}, v \in \{s_i, d_i\}} \left( \hat{r}^p_\ell(m) - r_\ell(m) - \frac{\delta^p_\ell(m)}{\rho_1} \right)^2 + \frac{\rho_2}{2} \sum_{n \in \Xi(\ell)} \left( v_n - \hat{v}_\ell n - \frac{\theta_\ell n}{\rho_2} \right)^2
\]

s.t. \(r_\ell(m) \geq 0, m \in \mathcal{M},\)

\[
\sum_{m \in \mathcal{M}} r_\ell(m) \leq \hat{c}_1, \hat{c}_2 r_\ell - \sum_{n \in \Xi(\ell)} \hat{c}_3, \hat{v}_\ell^2 n,
\]

(3.61)

Each of these problems is a quadratic problem with a single quadratic constraint and a number of nonnegativity constraints, therefore has closed-form solutions.

**Solving subproblem (3.53)**

This problem can be decomposed into two parts: one optimizes \(\hat{r}\) subject to the flow rate conservation constraint, and the other optimizes \(\hat{v}\). The respective forms of the subproblem will be shown shortly. Again we refer the interested readers to [2, Appendix B] for exact solutions for these problems.

**(i) Subproblem for \(\hat{r}\)** This subproblem decomposes into two independent problems.

The first one optimizes \(\hat{r}\)

\[
\arg \max \frac{\hat{r}}{2} - \frac{\rho_1}{2} \left( \hat{r} - r - \frac{\delta}{\rho_1} \right)^2 = \hat{r} + \frac{1 + 2\delta}{2\rho_1}.
\]

(3.62)

The second one optimizes \(\{\hat{r}(m)^{v(m)}, \hat{r}(m)^{d(m)}, \hat{r}_i^p(m), \hat{r}_i^d(m)\}\), subject to the conservation constraints of flow rate. Observe that the set of flow conservation constraints (3.47) decomposes among each node and each commodity. Further the introduction of the auxiliary variables made sure that each variable in \(\hat{r}\) only appears in a single constraint in (3.47). As such, the second problem further decomposes into a number of simpler problems, one for each commodity-node tuple \((m, v)\)

\[
\min \sum_{\ell \in \ln(v) \cup \text{Out}(v)} \left( \hat{r}_\ell^p(m) - r_\ell(m) - \frac{\delta^p_\ell(m)}{\rho_1} \right)^2 + 1_{s(m), d(m)}(v) \left( \hat{r}(m)^v - r(m) - \frac{\delta^v m}{\rho_1} \right)^2
\]

s.t. \(\hat{r}_\ell^p(m) + 1_{s(m)}(v)\hat{r}(m)^p = \sum_{\ell \in \text{Out}(v)} \hat{r}_\ell^p(m) + 1_{d(m)}(v)\hat{r}(m)^p.\)

(3.63)

Since problem (3.63) is a quadratic problem with a single equality constraint, it admits a closed-form solution.

**(ii) Subproblem for \(\hat{v}\)** The variable \(\hat{v}\) is constrained by the per-BS power constraint, therefore its related subproblem naturally decomposes over the BSs. For BS \(s \in B,\)
the problem is given by

$$\min_{\ell \in \text{Out}(s) \cap L^u} \sum_{n \in \tilde{L}(\ell)} \left( vE - \tilde{v}_{nt} - \frac{\theta_{nt}}{\rho_2} \right)^2$$

subject to

$$\sum_{\ell \in \text{Out}(s) \cap L^u} vE^2 \leq \tilde{p}_s,$$

which is a simple quadratic problem with a single quadratic constraint. Its solution can be again obtained analytically.

References


Optimization algorithms for big data with application in wireless networks


