A Random Deflected Subgradient Algorithm for Energy-Efficient Real-time Multicast in Wireless Networks

Guoping Tan, Jianjun Liu and Yueheng Li
College of Computer and Information Engineering, Hohai University
Nanjing, 210098, China
[e-mail: guoping_tan@hotmail.com]
*Corresponding author: Guoping Tan

Received March 21, 2016; revised June 27, 2016; accepted August 19, 2016; published October 31, 2016

Abstract

In this work, we consider the optimization problem of minimizing energy consumption for real-time multicast over wireless multi-hop networks. Previously, a distributed primal-dual subgradient algorithm was used for finding a solution to the optimization problem. However, the traditional subgradient algorithms have drawbacks in terms of i) sensitivity to iteration parameters; ii) need for saving previous iteration results for computing the optimization results at the current iteration. To overcome these drawbacks, using a joint network coding and scheduling optimization framework, we propose a novel distributed primal-dual Random Deflected Subgradient (RDS) algorithm for solving the optimization problem. Furthermore, we derive the corresponding recursive formulas for the proposed RDS algorithm, which are useful for practical applications. In comparison with the traditional subgradient algorithms, the illustrated performance results show that the proposed RDS algorithm can achieve an improved optimal solution. Moreover, the proposed algorithm is stable and robust against the choice of parameter values used in the algorithm.

Keywords: Network coding, scheduling, real-time multicast, wireless multi-hop networks, conflict graph, optimization, random deflected subgradient

This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant 2015B18914, and in part by the Key Laboratory of Wireless Sensor Network and Communication, Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences under Grant 2016001. The optimization framework and a traditional subgradient based solution have been presented in our previous publication "Minimum-Energy Wireless Real-Time Multicast by Joint Network Coding and Scheduling Optimization" in Mathematical Problems in Engineering, 2015, however, we contribute a novel general random-deflected-subgradient (RDS) based solution in this paper, which can outperform exiting algorithms significantly in terms of not only optimum results but also robustness and stability.

http://dx.doi.org/10.3837/tiis.2016.10.013 **ISSN : 1976-7277**

1. Introduction

The development of broadband wireless networks has led to the emergence of mobile commerce applications. Many mobile commerce applications require or can benefit from real-time multicast support in wireless networks [1]. As an example, the general aspects and principles for interfaces supporting Multimedia Broadcast Multicast Services (MBMS) is being specified in 3GPP LTE project [2]. The MBMS in 3GPP provides the possibility of distributed real-time multimedia services for mobile users via IP multicast data over point-to-multipoint radio bearers. Furthermore, emerging Tactile Internet services will be an important feature for the future wireless networks, which have very low latency requirements, e.g. 1ms round-trip time [3]. Energy-efficiency becomes an important requirement for supporting real-time wireless multicast. Hence, in this work, we focus on the aspect of minimum energy consumption for multicast transmission in multi-hop wireless networks by using joint network coding (NC) [4] and scheduling techniques.

In the last few years, NC has enabled new ways to significantly improve network performance such as throughput, robustness, and efficiency. Moreover, it was found that the multicast capacity can be achieved via random network coding (RNC) in multi-hop networks [5]. This result has led to a great progress of RNC-based distributed optimization algorithms for wireless networks. For example, Lun et al. decomposed the minimum cost multicast optimization problem into two sub-problems [6]. In the solution approach of [6], the authors first found an NC subgraph with minimum cost by solving a linear or a convex programming problem using a distributed primal-dual subgradient optimization algorithm. Next, the authors designed an NC scheme for the optimal subgraph, which can be solved by a simple RNC procedure. Similarly, Wu et al. proposed a distributed NC optimization algorithm for mobile ad-hoc networks to achieve a minimum-cost multicast [7].

Another approach for solving this optimization problem is to integrate the medium access control (MAC) techniques into the optimization framework. Compared with the techniques which use simple orthogonal interference-free channel models, the network performance can be improved significantly by using some optimized scheduling strategies in the MAC layer [6][8]. Recently, using an interference graph model, Jaramillo et al. studied the optimization problem about the equitable distribution and scheduling of resources when real-time and non-real-time services coexist in wireless multi-hop networks [9]. In addition, the same authors also studied the optimal rate allocation problem under heterogeneous delay constraints [10]. For non-real-time services, Xiong et al. proposed an efficient scheduling scheme for minimizing the energy consumption in ad-hoc wireless sensor networks with the objective of maximizing the network lifetime [11]. Although these studies consider the resource allocation and scheduling optimization problem in real-time and/or non-real-time scenarios, they do not integrate NC into the optimization framework.

In order to achieve the best performance in wireless multi-hop networks, we should integrate the two optimization problems discussed previously. By considering this aspect, Rajawat et al. improved the throughput performance of the wireless multicast system under strict delay constraints by joint NC and scheduling optimization [12]. Using hyperarcs to model the natural multicast properties of wireless transmissions, researchers proposed to construct a conflict graph model to identify effective network configuration for jointly optimizing scheduling and NC subgraph [13][14][15]. In fact, this conflict graph model can be used to build a protocol interference model of the nodes in wireless networks. To avoid the

interference, we can select the scheduling strategy by sampling stable sets in a conflict graph. This indicates that the feasible set of the network configuration can be modeled as a stable set in a conflict graph. Previous research studies have shown that, comparing with the traditional scheduling strategies using simple interference models, the performance of wireless networks can be improved significantly by jointly optimizing NC subgraph and scheduling using protocol interference models [13], or using more accurate physical interference models [16].

Since real-time streaming multicast services usually require the network to support a fixed data rate, the framework to optimize the overall network throughput performance proposed in [13] is not applicable. Lun [6] and Wu [7] et al. proposed a NC subgraph based optimization framework with minimum cost for supporting multicast scenarios with fixed data rates. However, there is no scheduling strategy integrated into their optimization framework. In addition, the conflict graph based scheduling optimization problem built in [13] is an NP-hard problem. Therefore, the authors proposed a greedy algorithm by sampling maximum weight stable sets to solve the optimization problem [13]. However, a major drawback of this algorithm is that it is very sensitive to the iteration parameters such as the step size. In other words, the results often fall far short of the global optimum value in case that the iteration parameters are not chosen properly. Since those iteration parameters can only be chosen through trial and error methods, it is thus difficult to meet real-time requirements. To address this issue, recently, we proposed a joint optimization algorithm by sampling K number of random maximal stable sets (K-RMSS) in a conflict graph [14]. Furthermore, it has come to our attention that Mohandespour et al. have also observed in an independent study that the near-optimal results can be achieved efficiently by randomly sampling subsets of MSSs [15], which is in principle similar to our proposed K-RMSS scheme in [14]. These independent studies have shown that the proposed K-RMSS scheme has a better performance in terms of convergence and accuracy than the existing algorithms in the literature.

The traditional sub-gradient based primal-dual algorithms, such as the K-RMSS proposed in [14] or the equivalent algorithm with randomly sampling subsets of MSSs proposed in [15], still have two major drawbacks. The first one is that it is quite sensitive to the iteration parameters during the primary recovery procedure. The second drawback is that we have to compute and store all of the previous iterative results during each primary recovery procedure, which is not suitable for the real-time implementation. Due to these drawbacks, to achieve a predefined target accuracy requirement, the iteration parameters have to be chosen carefully in advance through trial and error methods. However, Sherali et al. showed that the accuracy can be controlled up to some extent through random-deflected-subgradient (RDS) based primal-dual recovery algorithms [17]. Inspired by the findings in [17], our main contributions in this work can be summarized as:

1. A RDS based primal-dual algorithm is proposed to solve the optimization problem by joint NC and multicast scheduling. The proposed algorithmic solution is much more robust against the choice of iteration parameters than the traditional sub-gradient based primal-dual algorithms, such as the K-RMSS etc. Moreover, the proposed algorithm can achieve better optimization results;

2. For the proposed RDS algorithm, we derived the corresponding recursive formulas, which can be useful for an online implementation of the proposed algorithm.

The rest of the paper is organized as follows. In Section I, we introduce the network model adopted in this work. We present the construction of the minimum energy real-time multicast optimization framework using a conflict graph in Section II. In Section III, we propose a distributed RDS based optimization algorithm using Lagrangian decomposition. Illustrative

examples highlighting the benefits of the proposed RDS based algorithm are presented in Section IV. Finally, conclusions of this work are provided in Section V.

2. Network Model

We use a directed hypergraph $\mathcal{H}=(\mathcal{N}, \mathcal{A})$ to model a wireless multi-hop network, where $\mathcal{N}$, and $\mathcal{A}$ denote the set of nodes and a collection of hyperarcs, respectively. We define $(i, J)\in \mathcal{A}$ as a hyperarc and $N(i)\subset \mathcal{N}$, as a set of neighbors of node i within the receiving range. In other words, when the node i sends a data packet, it is assumed that all the nodes in $N(i)\subset \mathcal{N}$, can receive this packet. Clearly, for any hyperarc $(i, J)\in \mathcal{A}$, there are $i\in \mathcal{N}$, and $J\subset N(i)$. For each node, there are at most $2^{|N(i)|}-1$ hyperarcs.

Throughout the paper, we assume half-duplex transceivers and consider wireless networks operating with slotted time. In any time slot, a node can either broadcast one constant-length packet or stay idle. The scheduling issue considered here is to schedule all of the hyperarcs defined above. When scheduling multiple hyperarcs to transmit packets simultaneously, we must avoid the interference among them. The conflict situations among specific hyperarcs depend on the network interference model. We consider the following two commonly used protocol interference models: the primary interference model and the secondary interference model. In the primary interference model, it is assumed that each node can only receive data from one node every time. Whereas, in the secondary interference model, besides the previously mentioned constraints, it is also assumed that any receiving node can only receive data correctly when all other neighbors are in a dormant state. The two interference models are defined as follows:

Interference Models: when scheduling any two hyperarcs (i_1, J_1) and (i_2, J_2) simultaneously, the necessary and sufficient conditions for conflict-free transmission between them are:

(1) $i_1\neq i_2$ and $i_1\notin J_2, i_2\notin J_1$;

(2) $J_1\cap J_2=\Phi$ in the *primary interference model*; and $J_1\cap N(i_2)=\Phi$ and $J_2\cap N(i_1)=\Phi$ in *the secondary interference model*, where Φ denotes an empty set.

Note that for both the primary interference model and the secondary interference model, the definitions are symmetric. Therefore, we can use an undirected graph to represent the conflict between any pair of hyperarcs. In this paper, we use the method proposed in [13] to construct the conflict graph, which is defined below:

Conflict graph $\mathcal{G}$: for any hypergraph $\mathcal{H}$ defined above, we can construct an undirected graph $\mathcal{G}$ = ($\mathcal{T}$,$\mathcal{B}$) according to an interference model, where $\mathcal{G}$ represents conflicts among overall hyperarcs. Where, the vertex set $\mathcal{T}$ in $\mathcal{G}$ is the collection of all of the hyperarcs in $\mathcal{H}$. Each edge in $\mathcal{B}$ represents the conflict between two adjacent vertices according to the interference model, which indicates that these two corresponding hyperarcs cannot be scheduled simultaneously due to conflict.

Using the definitions of the interference models and the conflict graph described above, given any hypergraph $\mathcal{H}$, we can easily construct its conflict graph $\mathcal{G}$. Given a conflict graph $\mathcal{G}$, we now define a stable set S as any subset in which there is no edge connected between any two nodes. Any stable set S can be indicated by a column vector of length $|\mathcal{T}|$, which is defined as

$$\mathrm{I}_t^S = \begin{cases} 1, & if \quad t \in S \\ 0, & otherwise \end{cases} \tag{1}$$

A maximal stable set is the one that is not contained in any other stable set. A maximum stable set is a stable set of largest cardinality. The stability number of $\mathcal{G}$ is the cardinality of the maximum stable set. The stable set polytope denoted by $\mathrm{CH_{SS}}$ is the convex hull of the incidence vectors of all stable sets of $\mathcal{G}$.

3. Optimization Framework

In this paper, we adopt the same optimization framework introduced in [14], which minimizes the energy requirement for real-time wireless multicast via joint optimization of NC and scheduling. To present this framework, we provide definitions of some essential variables in **Table 1**.

Table 1. Symbol definitions

Symbols	Definitions
z_{iJ}	denotes the rate of NC data packet injected to hyperarc (i, J) by node i.
Z	denotes a collection representing the coded packet rate injected to overall hyperarcs, i.e. $Z = (z_{iJ})_{(i,J)\in\mathcal{A}}$, which is also known as an NC subgraph.
T	represents the collection of all sink nodes in a multicast session.
$x_{iJj}^{(t)}$	is the transmission rate of the information flow transferred to the terminal $t \in T$ when transmitting from $i \in \mathcal{N}$ to $j \in N(i)$ in case that the coded packet rate injected to the hyperarc $(i, J) \in \mathcal{A}$ is z_{iJ}.
$x_{ij}^{(t)}$	is the transmission rate of the information flow transferred to the sink node $t \in T$ when transmitting from $i \in \mathcal{N}$ to $j \in N(i)$, which is calculated by $x_{ij}^{(t)} = \sum_{J \subset N(i)} x_{iJj}^{(t)}$.
d_{ij}	denotes the distance between node i and node j.
ζ_{iJ}	represents the energy required when the coded packet transferred with the rate z_{iJ} over the hyperarc $(i, J) \in \mathcal{A}$, which is computed by $\zeta_{iJ} = \max_{j\in J} d_{ij}^2$ as in [14].

For simplifying the optimization framework for supporting real-time multicast services requiring higher reliability, we assume that the physical layer can ensure perfect reliable transmissions. This can be achieved for example through appropriate power control, channel coding, modulation and other methods when the coded packet rate on any hyperarc is not more than Z_{iJ}.

We consider the optimization problem of minimizing the energy consumption of real-time multicast. The objective function for this optimization problem can be expressed as a function of the coded packet rate Z_{iJ}, similar to [6], this is defined as

$$f_{iJ}(z_{iJ}) = \zeta_{iJ} z_{iJ} \tag{2}$$

For real-time multicast services requiring different reliabilities, we can modify the objective function in (2) by introducing some factors for describing transmission costs under

different link qualities. However, this general case is beyond the scope of the present contribution and left for future studies. Using the definitions in **Table 1**, as in [14], we can model the optimization problem of minimizing the energy consumption of transmissions with a fixed multicast rate as a joint scheduling and NC subgraph optimization problem, i.e.

$$\text{minimize } \sum_{(i,J)\in\mathcal{A}} f_{iJ}(z_{iJ}) \tag{3}$$

s.t.

Capacity constraints:

$$x_{ij}^{(t)} \le \sum_{\{J \mid j\in J, J\subset N(i)\}} z_{iJ}, \forall i \in \mathcal{N}, j \in N(i), t \in T$$

Flow constraints C_F:

$$\sum_{j\in N(i)} x_{ij}^{(t)} - \sum_{\{j \mid i\in N(j)\}} x_{ji}^{(t)} = \begin{cases} R, & i = s \\ -R, & i = t \\ 0, & else \end{cases} \quad \forall i \in \mathcal{N}, t \in T$$

$$x_{ij}^{(t)} \ge 0, \quad \forall i \in \mathcal{N}, j \in N(i), t \in T$$

Scheduling constraints:

$$Z = (z_{iJ})_{i\in\mathcal{N}} \in \mathrm{CH_{SS}}$$

Note that the objective function in this framework is a convex function and $\mathrm{CH_{SS}}$ is a convex set. Therefore, the optimization problem is a convex optimization problem. Theoretical studies have shown that the optimal NC subgraph can be obtained by solving the above described optimization problem. Then, the multicast transmission can meet all the constraints through RNC [5][6].

Since the optimization in the whole stable set polytope of $\mathrm{CH_{SS}}$, i.e. the scheduling sub-problem in (3), has been proved to be a NP-hard problem [13], there is no effective method to solve it even using centralized algorithms. Accordingly, we recently proposed to construct a distributed subgradient algorithm, i.e. *K*-RMSS, to solve this problem in [14]. As mentioned in the introduction part of this paper, the non-robustness and low efficiency of *K*-RMSS will severely limit its application in practical systems. To overcome its two major drawbacks, we thus propose a novel RDS based primal-dual algorithm to solve (3) in the following section.

4. Primal-dual RDS Algorithm

First, let $\lambda = \left(\lambda_{ij}^{(t)}\right)$ denote the Lagrangian multipliers. Using the Lagrangian relaxation, we can construct a dual function of the objective function of the original problem by moving the capacity constraint into the objective function in (3), i.e.

$$q(\lambda) = \min_{X\in \mathrm{C_F}, Z\in \mathrm{CH_{SS}}} \mathcal{L}(X, Z, \lambda) \tag{4}$$

where $X = (x_{ij}^{(t)})$, $\mathrm{C_F}$ is the flow constraint defined in (3), and $\mathcal{L}(X, Z, \lambda)$ denotes the corresponding Lagrangian function related to the multiplier λ. Finally, the dual objective function can be decomposed into two parts as follows,

$$\mathcal{L}(X,Z,\lambda)=\sum_{t\in T}\sum_{i\in\mathcal{N}}\sum_{j\in N(i)}\lambda_{ij}^{(t)}x_{ij}^{(t)}+\sum_{i\in\mathcal{N}}\sum_{J\subset N(i)}z_{iJ}\left(\xi_{iJ}-\sum_{t\in T}\sum_{j\in J}\lambda_{ij}^{(t)}\right) \tag{5}$$

Note that the two parts in Eq.(5) are coupled with the Lagrangian multiplier $\lambda_{ij}^{(t)}$. Substituting (5) into (4), we can decompose this dual problem into two sub-problems, i.e.

$$\begin{aligned} q(\lambda)&=\min_{X,Z}L(X,Z,\lambda)\\ &=\underbrace{\min_{X\in C_F}\sum_{t\in T}\sum_{i\in\mathcal{N}}\sum_{j\in N(i)}\lambda_{ij}^{(t)}x_{ij}^{(t)}}_{\text{sub-problem 1}}+\underbrace{\min_{Z\in CH_{SS}}\sum_{i\in\mathcal{N}}\sum_{J\subset N(i)}z_{iJ}\left(\xi_{iJ}-\sum_{t\in T}\sum_{j\in J}\lambda_{ij}^{(t)}\right)}_{\text{sub-problem 2}} \end{aligned} \tag{6}$$

Then, the dual problem of the original problem in Eq. (3) can be formulated as

$$\text{maximize } q(\lambda)=\min_{X,Z}\mathcal{L}(X,Z,\lambda) \tag{7}$$

s.t.

$$\lambda\geq 0$$

To solve this dual problem, we use the projected subgradient algorithm with the iterative rule as follows,

$$\lambda[n+1]=\max\left(\left(\lambda[n]+\delta[n]d[n]\right),0\right) \tag{8}$$

where $\delta[n]$ denotes a suitable step size and $d[n]$ denotes the deflected subgradient in the n-th iteration.

***Remark* 1:** for the traditional subgradient algorithms considered in [6][7][14], the authors proposed to utilize the current subgradient for updating the multiplier $\lambda[n]$ during the n-th iteration. Different from the traditional subgradient algorithms, the key idea of the deflected subgradient algorithm is to improve the robustness and accuracy by making use of the subgradient information of the previous iterations in an efficient way. We will explain these differences between the subgradient algorithms in more detail.

To explain our proposed RDS algorithm, let $g[n]=\left(g_{ij}^{(t)}[n]\right)$ denote the original subgradient in the n-th iteration and set $d[1]=g[1]$. As proposed in [17], in case of $n\geq 2$, the deflected subgradient $d[n]$ will be updated using:

$$d[n]=g[n-1]+\varphi\cdot d[n-1] \tag{9}$$

where φ $(0<\varphi<1)$ is a predefined deflected coefficient. It can be chosen randomly at the beginning and then kept constant during all of the remaining iterations [17]. Additionally, the original subgradient $g[n]$ is computed using:

$$g_{ij}^{(t)}[n] = \hat{x}_{ij}^{(t)}[n] - \sum_{\{J \mid j \in J, J \subset N(i)\}} \hat{z}_{iJ}[n] \tag{10}$$

where $\hat{x}_{ij}^{(t)}[n]$, $\hat{z}_{iJ}[n]$ denote the optimization results of the sub-problem 1 and sub-problem 2 in the *n*-th iteration of the algorithm for solving (7), respectively.

As suggested in [17], to control the accuracy of the algorithm, the step size $\delta[n]$ can be updated as a function of the deflection subgradient $d[n]$ using

$$\delta[n] = \frac{\varepsilon}{\|d[n]\|^2} \tag{11}$$

where ε ($0 < \varepsilon < 1$) denotes the accuracy control parameter and $\|d[n]\|$ denotes the Euclidean norm of $d[n]$. Using the iterative rule in (11) proposed for solving (7), Sherali and Choi [17] have proven that the gap between the global optimization results and the $q(\lambda)$ is not more than ε. That is, the solution for this dual problem can be viewed as an ε-optimal solution.

***Remark* 2:** first, by observing (9), we can note that the deflected subgradient $d[n]$ in any given iteration actually includes all of the subgradient information of the previous iterations using a filtering operation. However, the traditional subgradient algorithms only utilize the current subgradient $g[n]$ instead of $d[n]$ when applying the similar iterative rule as in Eq. (8). Secondly, the step size $\delta[n]$ adopted in the traditional subgradient algorithms (e.g $\delta[n] = \alpha/(\beta+n)$, where $\alpha>0$, $\beta\geq 0$ as used in [6][7][13][14]) does not consider the effect of the subgradient during each iteration. However, the step size $\delta[n]$ adopted here as in Eq. (11) considers the effect of the subgradient during each iteration. Therefore, due to these two main differences, it is expected that the proposed deflected subgradient algorithms can achieve a better performance than the traditional subgradient algorithms.

Furthermore, as shown in (10), the calculation of each subgradient $g_{ij}^{(t)}[n]$ is only related to local variables $\hat{x}_{ij}^{(t)}[n]$ and $\hat{z}_{iJ}[n]$. It indicates that all the computations can be carried out in a single-hop range of the node *i*. Therefore, any node-based distributed algorithm can be employed without collecting the information of the entire network. In other words, we can calculate the iteration values for Lagrangian multiplier $\lambda_{ij}^{(t)}$ using (8) and (10) in a distributed manner, where each node only needs to communicate with nodes which are in a one-hop distance rather than all the nodes in the network.

Since the intermediate results $\hat{x}_{ij}^{(t)}[n]$ and $\hat{z}_{iJ}[n]$ obtained by solving the dual sub-problems cannot guarantee the optimal result of the primary problem, we need to use the well-known primary recovery techniques to make the results converge to the optimal value [12]. To simplify the description, the optimal results about the information flow (i.e. $\hat{x}_{ij}^{(t)}[n]$) and NC subgraph (i.e. $\hat{z}_{iJ}[n]$) in the *n*-th iteration are defined as follows, respectively:

$$\begin{aligned} \hat{X}[n] &= (\hat{x}_{ij}^{(t)}[n]) \\ \hat{Z}[n] &= (\hat{z}_{iJ}[n]) \end{aligned} \tag{12}$$

The core idea of the primary recovery technique is to compute a convex combination of all the previous results as the current optimal solution of the primary problem. That is, by evaluating a convex combination of all the previous results, the optimal solution of the primary problem in the *n*-th iteration can be obtained by using the following formulas [18]:

$$X^*[n] = (\left(\hat{x}_{ij}^{(t)}[n]\right)^*) = \sum_{l=1}^{n} \tau_l[n]\hat{X}[l]$$
$$Z^*[n] = (z_{iJ}^*[n]) = \sum_{l=1}^{n} \tau_l[n]\hat{Z}[l] \tag{13}$$
$$\forall \tau_l[n] \geq 0, \sum_{l=1}^{n} \tau_l[n] = 1.$$

When adopting the RDS based algorithm, it is interesting to note that the formulas in (13) can be replaced by the following two simple recursive equations:

◆ *Recursive primary recovery formulas: during the n-th iterative primary recovery procedure, the optimal solution of the information flow and the NC subgraph can be obtained by the following two simple recursive formulas:*

$$X^*[n] = \frac{\varphi\left(1-\varphi^{n-1}\right)}{1-\varphi^n} X^*[n-1] + \frac{1-\varphi}{1-\varphi^n}\hat{X}[n]$$
$$Z^*[n] = \frac{\varphi\left(1-\varphi^{n-1}\right)}{1-\varphi^n} Z^*[n-1] + \frac{1-\varphi}{1-\varphi^n}\hat{Z}[n] \tag{14}$$

A detailed derivation of the *recursive primary recovery formulas* can be found in the Appendix at the end of the paper. After obtaining $Z^*[n]$, the final optimization results of the primary problem at the n-th iteration can be computed using

$$f^*[n] = f(Z^*[n]) = \sum_{(i,J)\in \mathbf{A}} f_{iJ}(z_{iJ}^*[n]) \tag{15}$$

Now let N_{max} denote the maximum allowed number of iterations, ε denote a predefined accuracy requirement (where $0< \varepsilon <1$), and φ denote a randomly chosen deflected coefficient ($0<\varphi<1$). As a result, the proposed primal-dual RDS optimization algorithm for solving (3) can be summarized as in **Table 2**.

Table 2. The proposed primal-dual RDS algorithm

Input: $\mathcal{G}$, **T**, K, ε, φ, N_{max}

Initialization: set $0<\lambda<1$ randomly, n=1

While $n \leq N_{max}$

Step 1: solve the sub-problem 1 in (6), then obtain $\hat{X}[n]$;

Step 2: randomly sample K maximal stable sets in $\mathcal{G}$, denote the sets by $\mathbf{\Omega_k}$;

Step 3: solve the sub-problem 2 in (6) based on $\mathbf{\Omega_k}$, then obtain $\hat{Z}[n]$;

Step 4: compute $X^*[n]$ and $Z^*[n]$ according to (14);

Step 5: calculate the optimal results $f^*[n]$ of the primary problem using (15);

Step 6: if $f^*[n]$ satisfies the stopping criterion, then break; otherwise, proceed to **Step 7**;

Step 7: compute the current subgradient $g[n]$ using (10);

Step 8: compute the current $d[n]$ using (9);

Step 9: calculate $\lambda[n+1]$ using (8), sets $n=n+1$ and return to **Step 1**.

End while

Output: $f^*[n]$

The stopping criterion of the algorithm can be designed based on the difference between the optimal results of the primary problem in two successive iterations. For example, the stopping criterion can be designed as $\left| f^*[n] - f^*[n-1] \right| \leq \varepsilon$.

Regarding the solution of the sub-problem 1 in (6), we can model it as the classical shortest path problem, which can be solved through a traditional asynchronous distributed Bellman-Ford algorithm [19]. Whereas, for the sub-problem 2 in (6), we can model it as a simple linear programming problem and a solution can be obtained by randomly sampling K maximal stable sets, as proposed in Section 4.2 of [14]. We would like to point out an important aspect of the present contribution is that the proposed primal-dual RDS algorithm is independent of the solutions of both the sub-problem 1 and the sub-problem 2. That is, any solution to the sub-problem 1 and/or the sub-problem 2 can easily be integrated into the proposed algorithm for improving the optimization performance. In this paper, we only adopt the proposed solution to the sub-problem 2 in [14] as an example for the illustrations.

Remark* 3*: Firstly, as mentioned previously, the calculation of both equations (8) and (10) can be implemented in a fully distributed manner. And the solutions to both the sub-problem 1 and the sub-problem 2 can also be implemented in a distributed manner as explained in [14]. Therefore, the proposed RDS based algorithm can also be implemented in a distributed fashion. Secondly, through observing the recursive rules as shown in (14), we can find that the proposed algorithm only makes use of the results obtained by the previous iteration. It indicates that the proposed algorithm only needs to store the results obtained during the previous iteration. This implementation procedure can save the memory required for storage significantly and carry out online computations easily. Thirdly, based on the step size defined by (11), the proposed algorithm can provide a parameter for controlling the accuracy of the optimization results, which cannot be achieved by the traditional subgradient algorithms. Finally, since the deflected coefficient can be chosen randomly at the beginning, we do not need to set this parameter by some difficult trial and error methods as adopted by the traditional subgradient algorithms.

5. Illustrative Results

To illustrate the advantages of the proposed algorithm, we developed a simulation package in MATLAB. To find a solution for (3) using a centralized algorithm and a solution for the sub-problem 2 in (6) as proposed by [14], we used CVX, a package for specifying and solving convex programs [20]. For each simulation, we used a random network topology model in which all nodes were randomly and uniformly distributed within a square area with a unit density. The side length of the square area is set to the square root of $|\mathcal{N}|$. Any two nodes are viewed as reachable when the distance between them is less than a certain communication radius r. We used multicast scenarios with one session and two sink nodes in the simulations. We also used an auxiliary interference model to model a conflict graph. We selected the left-most node as the source node and the right-most multiple nodes as sink nodes for each multicast scenario within a randomly generated network topology.

5.1 Performance comparison

For illustrating a performance comparison of different schemes, we carried out simulations for the following four algorithms: i) a fully centralized algorithm presented in [21] indicating the approximate global optimization results; ii) the *Greedy Maximal Weighted Independent Set* (GMWIS) based algorithm proposed in [13]; iii) the *K-Random Maximal Stable-sets Subgradient* (*K*-RMSS) based traditional subgradient algorithm proposed by us recently in [14], and iv) the *K-Random Maximal Stable-sets Deflected Subgradient* (*K*-RMSDS) algorithm proposed in this paper, in which we solve the sub-problem 2 by randomly sampling K maximal stable sets, as proposed in [14].

For a fair comparison, both the *K*-RMSS and the GMWIS adopt the following rule for updating the step size: $\delta[n]= \alpha/(\beta+n)$. A detailed information on this rule can be found in [13]. As presented in **Table 2**, the *K*-RMSDS algorithm need to set the following two iteration parameters: a predefined accuracy requirement ε and the deflected coefficient φ. Moreover, both the communication radius r and the iteration parameters always use the same values as listed in the following **Table 3**.

Table 3. Simulation parameters

Parameters	α	β	φ	ε	r
GMWIS	0.6	1	-	0.001	1.6
K-RMSS	0.6	1	-		
K-RMSDS	-	-	0.6		

5.1.1 Optimality

Since the size of the feasible search space increases exponentially with the number of sink nodes for the centralized algorithm proposed in [21], we set the number of sink nodes to 2 (i.e. $|T|=2$) for the convenience of comparing with the approximate global optimization results. However, the number of nodes is set to vary from 6 to 20 in simulations. For each multicast scenario with a fixed number of nodes, we produced 100 random network topologies for simulations and compare the average values. Moreover, for each simulation scenario, both the *K*-RMSS and the *K*-RMSDS algorithms use the same *K* value, which is listed in **Table 4**.

Table 4. *K* values

Number of Nodes, i.e. $\lvert\mathcal{N}\rvert$	6	8	10	12	14	16	18	20
K values	5	10	15	20	30	40	50	60

Finally, by computing the average values over 100 random network topologies for the four algorithms mentioned above, we obtain the final results, which are shown in **Fig. 1**.

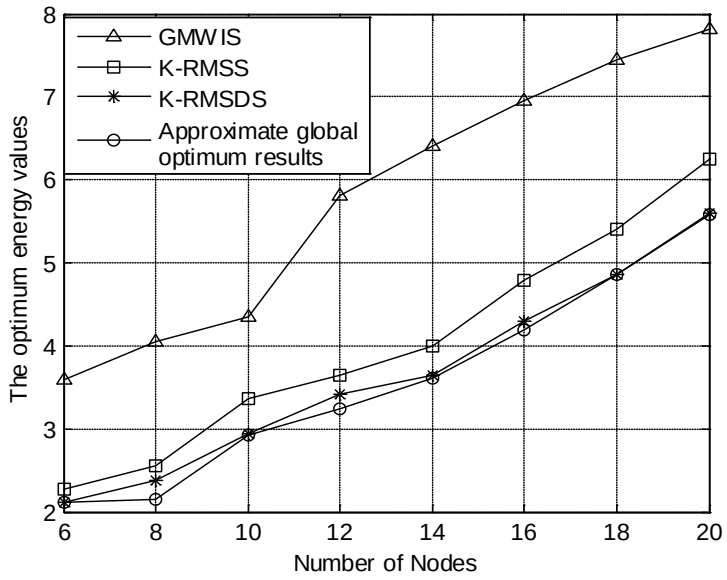


Fig. 1. Performance comparisons over 100 random network topologies

We illustrate a performance comparison of various algorithms in **Fig. 1.** This figure shows that both the *K*-RMSS and the *K*-RMSDS algorithms outperform the GMWIS algorithm significantly. It can be explained by the search space size adopted by different algorithms: the GWMIN algorithm uses only one MSS while the other two algorithms use *K* MSSs by random sampling algorithm for obtaining the optimization results during each iteration. This means that the searching space in the *K*-RMSS and the *K*-RMSDS algorithms is enlarged *K* times compared to the GMWIS algorithm, which thus results in a better optimization performance. Another observation that can be made from the **Fig. 1** is that the *K*-RMSDS outperforms the *K*-RMSS significantly. This is because the *K*-RMSDS algorithm uses the previously computed subgradient information in each iteration through the computation of the deflected subgradient. This results in a searching direction in each iteration of the *K*-RMSDS algorithm that is much more accurate than traditional subgradient algorithms. Additionally, the proposed *K*-RMSDS algorithm can control the precision of the optimization solution for the dual problem by changing the precision parameter ε. Hence, this algorithm can improve the optimization results without increasing the size of search space. Finally, **Fig. 1** shows that the performance of the *K*-RMSDS algorithm is identical to the global optimization result for most scenarios. Accordingly, it is expected that the *K*-RMSDS algorithm can approximate the global optimization search results by choosing a large values for *K* and setting a smaller value for ε.

5.1.2 The number of sink nodes

In order to study the effect of the number of sink nodes on both the *K*-RMSS and the *K*-RMSDS algorithms, we also carried out simulations for multicast scenarios with 20 nodes over 100 random network topologies. In this case, we keep all other simulation parameters similar to the ones considered for the result in Fig.1, except that we vary the number of sink nodes. By computing the average values over 100 random network topologies for the *K*-RMSS and the *K*-RMSDS algorithms, we obtain the average optimum energy consumption values, which are specified in the following table.

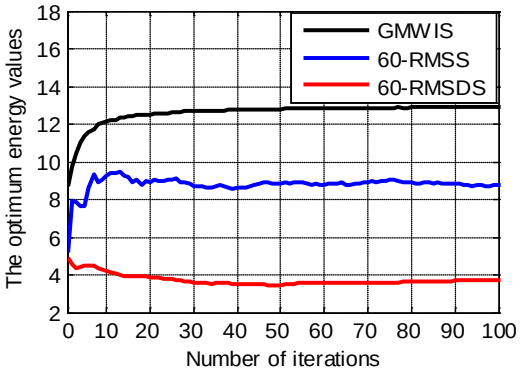
Table 5. The average optimum energy values over 100 random topologies

Number of sink nodes, i.e.|T|	6	7	8	9	10
60-RMSS	7.99	8.01	8.67	8.37	8.46
60-RMSDS	5.19	4.79	4.92	5.14	5.24
Performance gain	35%	40%	43%	39%	38%

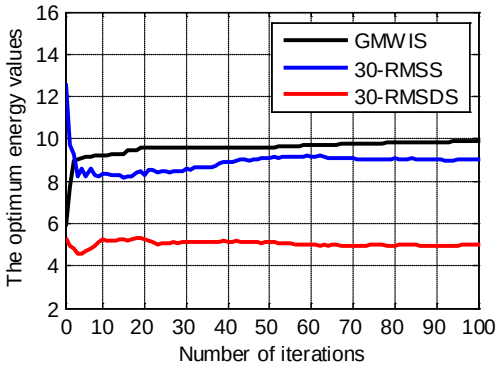
The values in **Table 5** show that the *K*-RMSDS scheme can provide a performance gain of more than 35% compared with the *K*-RMSS scheme. This indicates that the *K*-RMSDS scheme can outperform the *K*-RMSS scheme significantly for the multicast scenarios with variable group size. In addition, for both the *K*-RMSS and the *K*-RMSDS, we have noted that increasing the number of sink nodes has no significant effect on the optimum results. This observation can be explained as follows: independent of the number of sink nodes, for improving the multicast efficiency, the optimization algorithm will select as many nodes as possible for taking part in the NC operations. This results in almost all of the nodes having to carry out the NC operations for load balancing. Therefore, for multicast scenarios with a fixed number of total nodes, the variable number of sink nodes has no significant impact on the optimal values obtained by the algorithms.

5.1.3 Convergence and complexity

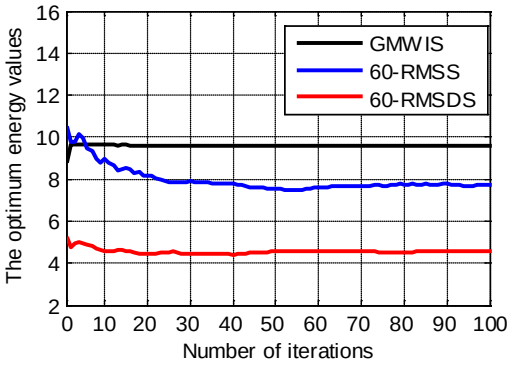
Next, we compare the convergence performance of different schemes. For this purpose, without loss generality, we choose multicast scenarios with 20 nodes. In the following figures, we present four snapshots of typical convergence performances in simulations with four random topologies using different values of |T| and *K*.



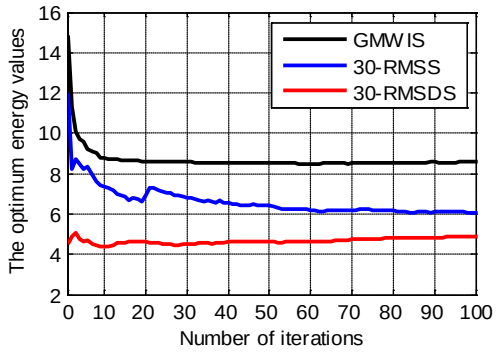
a. |T|=10，*K*=60



b. |T|=10，*K*=30



c. |T|=5，*K*=60



d. |T|=5，*K*=30

Fig. 2. Convergence comparisons

From the four illustrations presented in **Fig. 2**, we can see that the performance of both the *K*-RMSS and the *K*-RMSDS algorithms are always better than the GWMIN algorithm. Similar to the analysis in Section 5.1.1 and 5.1.2, it indicates that the same results hold for the multicast scenarios with different group sizes. Moreover, these illustrations show that the *K*-RMSDS can outperform *K*-RMSS significantly for all of the four cases.

The illustrations show that the convergence rate of the *K*-RMSDS scheme is almost same as that of the GWMIS scheme, and is much faster than that of the *K*-RMSS scheme. As shown in **Fig. 2. d**, both the GWMIS and the *K*-RMSDS can converge within 20 iterations. However, the *K*-RMSS scheme needs about 55 iterations for the convergence in this case, which has a much slower convergence performance than the other two schemes.

Finally, to have a fair comparison of the efficiencies between the *K*-RMSS and the *K*-RMSDS, we recorded the running times of 500 iterations for each simulation and then computed the average running time for one iteration. The resulting values are specified in **Table 6**. The simulations have been carried out under the following software and hardware environments: CVX (ver.2.0), MATLAB (ver.7.11.0), Windows 10 OS, 2.93GHz Intel 2 Duo

CPU and 4G RAM.

Table 6. Comparisons on the average running time for one iteration

Total number of nodes, i.e. $\|\mathcal{N}\|$		6	12	16	20
Number of sink nodes, i.e.$\|T\|$		3	6	8	10
K-RMSS	*K*=60	0.80s	1.09s	1.61s	2.53s
	K=30	0.74	1.00s	1.45	2.12s
K-RMSDS	*K*=60	0.74s	1.08s	1.62s	2.49s
	K=30	0.72	1.00s	1.43	2.12s

Table 6 shows that the average running time of *K*-RMSDS with *K*=60 or *K*=30 is approximately the same as that of the *K*-RMSS algorithm. Furthermore, from **Fig. 2**, we can observe that the *K*-RMSDS can converge in less number of iterations compared to the *K*-RMSS algorithm. Therefore, both observations indicate that the convergence performance of the *K*-RMSDS scheme is much better than that of the *K*-RMSS scheme. Finally, we would like to point out that, as shown in **Table 6,** the average running time decreases with a decrease of the value of *K* for both schemes. It means that, for both the *K*-RMSS and the *K*-RMSDS, the parameter *K* can provide a tradeoff between the optimization results and the efficiency.

5.2 Stability Analysis of the *K*-RMSDS algorithm

For the traditional subgradient algorithms, the iteration parameters have to be predefined carefully through trial and error methods for obtaining a robust performance. Otherwise, the algorithm will be quite unstable and it cannot converge in some cases. For the GMWIS and the *K*-RMSS algorithms, the parameters α and β are the parameters that need to be chosen by trial and error methods. However, to achieve a good convergence and accuracy by the proposed *K*-RMSDS algorithm, the parameters *K* and ε can be pre-set easily according to practical scenarios, and the deflected coefficient φ can be set randomly. To validate the algorithm stability, we chose random network topologies with 15 nodes and two sink nodes for the simulation studies. For comparison, the value of α is uniformly sampled within the interval (0,1) and β is set to 1 for each simulation in case of both the GMWIS and the *K*-RMSS algorithms. For the *K*-RMSDS algorithm, we set ε=0.0001 and set the deflected coefficient φ as a uniformly distributed random variable in (0,1). For both the *K*-RMSS and the *K*-RMSDS algorithms, the value of *K* is set to 40 for a fair comparison. We also carried out 100 simulations for each algorithm and computed the average and variance of the convergence results and the resulting values are specified in **Table 7**.

Table 7. Comparisons over 100 random network topologies with 15 nodes

Algorithm	GMWIS	40-RMSS	40-RMSDS
Average Optimum Results	7.03	6.49	5.86
Variances	2.63	0.13	0.003
Global Optimum Result		5.58	

According to **Table 7**, we can note that the variance of the GMWIS algorithm is very large. This leads to the observation that we need to choose the iteration parameters of the algorithm carefully for achieving a good convergence and accuracy performance. In addition, we can see that the variance of the *K*-RMSS is much less than that of the GMWIS, but much larger than that of the *K*-RMSDS. Furthermore, the average value of the *K*-RMSS is much higher than the global optimization result. It indicates that, for achieving the best optimization performance by

the *K*-RMSS, we have to choose iteration parameters carefully in advance by some trial and error methods. However, for the *K*-RMSDS algorithm values shown in this table, we can find that the variance of the algorithm is quite small and the average value is very close to the global optimization result. It means that, the *K*-RMSDS algorithm is robust against the choice of values for the parameters φ, *K* and ε.

Finally, for studying the influence of the parameter *K* on the algorithm, we carried out simulations by increasing the value of *K* for the simulation scenarios described above. The results are shown in **Fig. 3**.

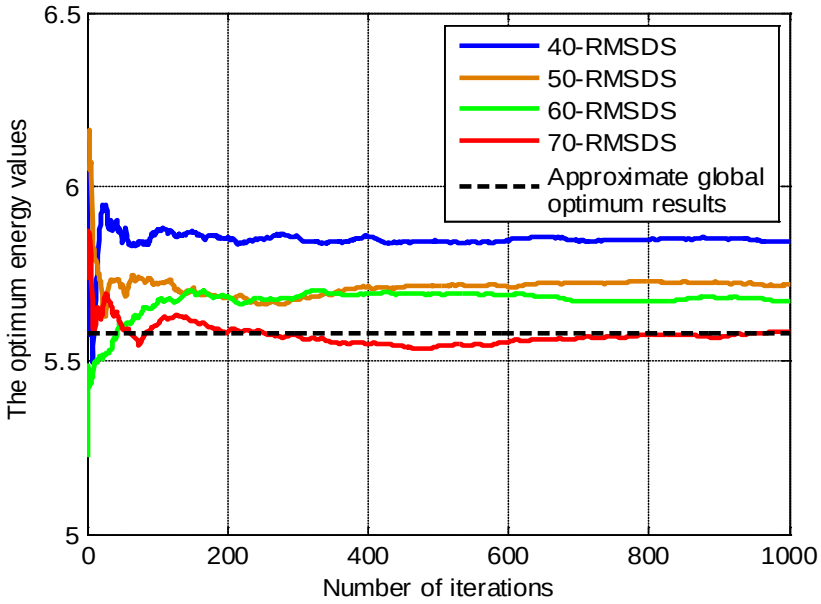


Fig. 3. Performance comparisons for *K*-RMSDS with different *K*

From this figure, we can see that the performance of the *K*-RMSDS improves significantly with an increase in the value of the parameter *K*. Specially, the performance of the *K*-RMSDS can almost achieve the approximate global optimization result with *K*=70. This is because the search space increases with an increase in the value of the parameter *K*. Moreover, from **Fig. 3**, we can observe that the rate of convergence of the algorithm remains constant independent of the value of the random deflected coefficient. That is, for this scenario, the algorithm can always converge within about 200 iterations for different choice of value for *K*. This observation also leads us to conclude that the proposed random optimization algorithm has a stable and robust performance.

6. Conclusions

The optimization of energy consumption for real-time multicast in wireless multi-hop networks is a challenging problem. For solving this optimization problem, using an optimization framework of joint NC and scheduling via conflict graph models, we proposed a *K*-Random Maximal Stable-sets Deflected Subgradient (*K*-RMSDS) algorithm. Moreover, we derived a corresponding set of recursive formulas suitable for practical applications. The illustrative results showed that the proposed algorithm not only outperforms the existing

algorithms, but also has better performance in terms of stability and robustness. Although the studies in this paper have shown that the proposed algorithm has an excellent performance, many topics still need further studies. For example, a study on the choice of parameter K. A distributed method for implementation of the proposed algorithm for practical systems without a loss of accuracy.

Finally, we would like to point out some potential interesting applications of the proposed RDS based primal-dual algorithm:

1. The architecture of the proposed RDS algorithm is general. Hence, it can be applied in any primal-dual solution for a similar optimization framework with joint NC and scheduling. Therefore, by applying the proposed RDS based primal-dual algorithm, it is expected to result in an improved performance for solving the similar optimization problems such as those with inter-session NC [22][23], with packet scheduling [24], and with strict delay constraints [25][26] etc. The extended applications mentioned above are worthy of deeper study in the future.

2. It is also interesting to study how to apply the proposed algorithm for solving a similar optimization problem with more accurate interference models (IMs). Recent studies have shown that the optimization framework with more accurate IMs, such as the Signal-to-Interference-and-Noise-Ratio (SINR) based probability IM proposed in [16], can perform more efficiently than those with conflict-free based protocol IMs adopted in this paper. We thus expect that a significant performance gain can be achieved when applying the proposed RDS based algorithm approach for solving the similar optimization problem with accurate IMs.

Appendix: Derivation of the recursive primary recovery formulas

First, Sherali and Choi [17] have proven that, when choosing a random deflected coefficient φ ($0<\varphi<1$), the convex coefficients $\tau_l[n\text{-}1]$ adopted in Eq. (13) can be computed by

$$\tau_n[n]=\frac{1-\varphi}{1-\varphi^n},\quad \tau_l[n]=\frac{(1-\varphi)\varphi^{n-l}}{1-\varphi^n}\quad \forall l=1,...,n,\forall n \tag{16}$$

Now considering the following ratio for $\forall l=1,...,n,\forall n$

$$\begin{aligned}\frac{\tau_l[n]}{\tau_l[n-1]}&=\frac{(1-\varphi)\varphi^{n-l}}{1-\varphi^n}\times\frac{1-\varphi^{n-1}}{(1-\varphi)\varphi^{n-1-l}}\\&=\frac{1-\varphi^{n-1}}{(1-\varphi^n)\varphi^{-1}}\\&=\frac{\varphi(1-\varphi^{n-1})}{(1-\varphi^n)}\\&=\tau'[n-1]\end{aligned} \tag{17}$$

By observing (17), we can find that this ratio is independent of the parameter l, which is denoted by $\tau'[n\text{-}1]$ here. Therefore, we can derive a recursive algorithm for computing $X^*[n]$ in the primary recovery algorithm as follows

$$
\begin{aligned}
X^*[n] &= \sum_{l=1}^{n} \tau_l[n]\hat{X}[l]] \\
&= \sum_{l=1}^{n-1} \frac{\tau_l[n]}{\tau_l[n-1]} \tau_l[n-1]\hat{X}[l]] + \tau_n[n]\hat{X}[n] \\
&= \sum_{l=1}^{n-1} \tau'[n-1]\tau_l[n-1]\hat{X}[l]] + \tau_n[n]\hat{X}[n] \\
&= \tau'[n-1]\sum_{l=1}^{n-1} \tau_l[n-1]\hat{X}[l]] + \tau_n[n]\hat{X}[n] \\
&= \tau'[n-1]\hat{X}^*[n-1] + \tau_n[n]\hat{X}[n]
\end{aligned} \tag{18}
$$

Then, substituting (16) and (17) into (18), we can obtain the recursive formula corresponding to $X^*[n]$ as follows

$$
X^*[n] = \frac{\varphi\left(1-\varphi^{n-1}\right)}{1-\varphi^n} X^*[n-1] + \frac{1-\varphi}{1-\varphi^n}\hat{X}[n] \tag{19}
$$

Similarly, the recursive formula corresponding to $Z^*[n]$ can be obtained by

$$
Z^*[n] = \frac{\varphi\left(1-\varphi^{n-1}\right)}{1-\varphi^n} Z^*[n-1] + \frac{1-\varphi}{1-\varphi^n}\hat{Z}[n] \tag{20}
$$

This completes the derivation.

References

[1] P. E. Kourouthanassis and C. K. Georgiadis,, "Analysing m-commerce research: technology, applications and research themes," *Int. J. Mobile Communications*, Vol. 12, No. 1, 2014. Article (CrossRef Link)

[2] 3GPP TS 36.440 v12.0.0: "LTE; General aspects and principles for interfaces supporting Multimedia Broadcast Multicast Service (MBMS) within E-UTRAN," Sept., 2014.

[3] G. P. Fettweis, "The Tactile Internet: Applications and Challenges," *Vehicular Technology Magazine, IEEE*, vol. 9, no. 1, pp. 64-70, 2014. Article (CrossRef Link)

[4] R. Ahlswede, N. Cai, S. Y. Li, and R. Yeung, "Network Information Flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000. Article (CrossRef Link)

[5] T. Ho, R. Koetter, M. Medard, M. Effros, and et al., "A Random Linear Network Coding Approach to Multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413-4430, Oct. 2006. Article (CrossRef Link)

[6] D. Lun, N. Ratnakar, M. Medard, R. Koetter, and et al., "Minimum-Cost Multicast over Coded Packet Networks," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2608-2623, Jun. 2006. Article (CrossRef Link)

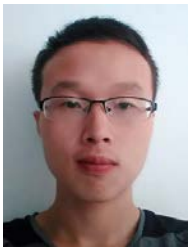
[7] Y. Wu, P. A. Chou, and S. Kung, "Minimum-Energy Multicast in Mobile Ad Hoc Networks Using Network Coding," *IEEE Trans. Comm.*, vol. 53, no. 11, pp. 1906-1918, Nov. 2005. Article (CrossRef Link)

[8] Y. Wu and S.Y. Kung, "Distributed Utility Maximization for Network Coding Based Multicasting: A Shortest Path Approach," *IEEE Journal on Sel. Areas in Comm.*, vol. 24, no. 8, pp. 1475–1488, Aug. 2006. Article (CrossRef Link)

[9] J. Jaramillo, R. Srikant, and L. Ying, "Scheduling for Optimal Rate Allocation in Ad Hoc Networks With Heterogeneous Delay Constraints," *IEEE Journal on Sel. Areas in Comm.*, Vol. 29, No.5, pp. 979-987. May 2011. Article (CrossRef Link)

[10] J. Jaramillo, and R. Srikant, "Optimal Scheduling for Fair Resource Allocation in Ad Hoc Networks with Elastic and Inelastic Traffic," *IEEE/ACM Trans. Networking*, vol. 19, no. 4, pp. 1125-1136, Aug. 2011. Article (CrossRef Link)
[11] Xiong, N.; Huang,X. X. B.; Cheng, H. J. and Wan, Z, "Energy-Efficient Algorithm for Broadcasting in Ad Hoc Wireless Sensor Networks," *Sensors*, 13(4), pp. 4922-4946, 2013. Article (CrossRef Link)
[12] Rajawat, K.; Giannakis, G.B. "Joint Scheduling and Network Coding for Multicast in Delay-Constrained Wireless Networks," *IEEE Trans. Sign. Processing*, 59(12), 6186-6196, 2011. Article (CrossRef Link)
[13] Traskov, D.; Heinlmaier, M.; Medard, M.; Koetter, R. "Scheduling for Network-coded multicast," *IEEE/ACM Trans. Networking*, 20(99), 1479-1488, 2012. Article (CrossRef Link)
[14] G.Tan, Y.Zhu, Y.Li, L.Zhang and J.Xu, "Minimum-Energy Wireless Real-Time Multicast by Joint Network Coding and Scheduling Optimization," *Mathematical Problems in Engineering*, vol. 2015, Article ID 390457, 9 pages, 2015. Article (CrossRef Link)
[15] M. Mohandespour, M. Govindarasu, and Z. Wang, "Rate, Energy, and Delay Tradeoffs in Wireless Multicast: Network Coding versus Routing," *IEEE Transactions on Mobile Computing*, vol. 15, no. 4, pp. 952-963, 2016. Article (CrossRef Link)
[16] T. Wiese, M. Riemensberger, and W. Utschick, "Scheduling for Network-Coded Multicast With Interference," *IEEE Transactions on Signal Processing*, vol. 64, no. 9, pp. 2245-2254, 2016. Article (CrossRef Link)
[17] Sherali H, and Choi G. "Recovery of primal solutions when using subgradient optimization methods to solve Lagrangian duals of linear programs," *Operations Research Letters*, 19(3): 105-113, 1996. Article (CrossRef Link)
[18] T. Larsson, M. Patriksson, and A. Stroemberg, "Ergodic, Primal Convergence in Dual Subgradient Schemes for Convex Programming," *Math. Program.*, vol. 86, no. 2, pp. 283–312, Nov. 1999. Article (CrossRef Link)
[19] D. Bertsekas and R. Gallager, Data networks (2nd ed.), Upper Saddle River, NJ: Prentice-Hall, 1992.
[20] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming," *http://cvxr.com/cvx*, Sep. 2013.
[21] Heinlmaier M, Traskov D, Koetter R, et al., "Scheduling for Network-coded multicast: A Distributed Approach ," in *Proc. of 2009 IEEE GLOBECOM*, Hawaii, USA, 1-6, 2009. Article (CrossRef Link)
[22] M. Heinlmaier, D. Lun, D. Traskov, and M. Medard, "Wireless Inter-Session Network Coding-An Approach Using Virtual Multicasts," in *Proc. of 2011 IEEE International Conference on Communications (ICC)*, pp. 1-5, Kyoto, Japan, Jun. 2011. Article (CrossRef Link)
[23] M. Esmaeilzadeh, and N. Aboutorab, "Inter-session Network Coding for Transmitting Multiple Layered Streams over Single-hop Wireless Networks," in *Proc. of IEEE Information Theory Workshop (ITW)*, pp. 401-405, Hobart, Australia, Nov. 2014. Article (CrossRef Link)
[24] N. Moghadam, and H. Li, "Queue Stability Analysis in Network Coded Wireless Multicast Network," *IEEE Communications Letters*, vol. 20, no. 5, pp. 950-953, 2016. Article (CrossRef Link)
[25] H. Li, B. Li, T. T. Tran and et al., "Transmission Schemes for Multicasting Hard Deadline Constrained Prioritized Data in Wireless Multimedia Streaming," *IEEE Trans. Wireless Comm.*, vol. 15, no. 3, pp. 1631-1641, 2016. Article (CrossRef Link)
[26] S. Zhou, and L. Ying, "On Delay Constrained Multicast Capacity of Large-Scale Mobile Ad Hoc Networks," *IEEE Trans. Inf. Theory*, vol. 61, no. 10, pp. 5643-5655, 2015. Article (CrossRef Link)

Guoping Tan is a full Prof. with the college of computer and information engineering at Hohai University, Nanjing, China. He received the PhD degree in Computer Science from Saarland University, Germany, in 2009. He was a senior engineer in ZTE Corporation China from 2000 to 2003. From Sep.2003 to Aug. 2005, he was a PhD candidate with the National Mobile Communication Research Lab (NCRL) at Southeast University, Nanjing, China. From Sep.2005 to Apr.2009, he was a researcher in Telecommunications Laboratory, Saarland University, Germany under DAAD scholarship. He is an IEEE member since 2009. His current research interests mainly include mobile Ad-hoc networks, wireless multimedia communications and signal processing for wireless communications. E-mail: gptan@hhu.edu.cn



Jianjun Liu is a graduate student , with the college of computer and information engineering at Hohai University, Nanjing, China. He received the bachelor degree in Hohai University, China, in 2015. His undergraduate major is communication engineering. His current research interests mainly include channel estimation, wireless multimedia communications and signal processing for wireless communications. E-mail：liujianjun93@hotmail.com

Yueheng Li received the B.Sc and M.Sc degrees in electronic engineering from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1993 and 1998 respectively, and the Ph.D degree in communications and information engineering from Southeast University, Nanjing, China, in 2001. He is currently a professor at the College of Computer and Information, Hohai University. His current research interests include signal processing in communications, compact MIMO capacity analysis, and channel coding.