Ad Hoc Networks 37 (2016) 460–474

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Distributed multi-objective cross-layer optimization with joint hyperlink and transmission mode scheduling in network coding-based wireless networks☆

Jain-Shing Liu [a], Jichiang Tsai [b,*]

[a] Department of Computer Science and Information Engineering, Providence University, Taichung, Taiwan, ROC
[b] Department of Electrical Engineering and Graduate Institute of Communication Engineering, National Chung Hsing University, Taichung, Taiwan, ROC

ARTICLE INFO

Article history:
Received 18 April 2015
Revised 12 September 2015
Accepted 17 September 2015
Available online 30 September 2015

Keywords:
Multi-objective
Cross-layer optimization
Distributed algorithm
Joint scheduling
Network coding

ABSTRACT

In this work, we address a cross-layer multi-objective optimization problem of maximizing network lifetime and optimizing aggregate system utility with intra-flow network coding, solved in a distributed manner. Based on the network utility maximization (NUM) framework, we resolve this problem to accommodate routing, scheduling, and stream control from different layers in the coded networks. Specially, we consider that there are two scheduling primitives, namely hyperlink and transmission mode, to be concurrently activated for the multi-objective optimization. Given the constraints with respect to these primitives, the optimization problem is specifically formulated as a quadratically constrained quadratic programming (QCQP) problem that is NP-hard in general, and its scheduling subproblem even when reduced to account for only one of these primitives is a maximum weighted independent set (MWIS) problem that is NP-hard already. To alleviate this complex problem in a distributed manner, we resort to alternate convex search (ACS) and primal decomposition (PD) to approximate the optimal results by using biconvex programming model and subgradient-based algorithm that can iteratively approach to the optimal solution. For the wireless multihop networks, wherein an optimal solution could be practically approximated as its validity would be out-of-date soon in the error-prone wireless environment, our simulation results show that the distributed method can fulfill our requirements, and can make a good trade-off on the heterogeneous objectives with well computational efficiency.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Recent proliferation of wireless services has created large scale demands for transmission of traffic requiring stringent throughput guarantees, and the system performance of such networks is typically a function of the amount of data collected by individual stations and delivered to a set of sinks through multi-hop routing. However, these stations usually operate with small batteries that are difficult to replace in typical scenarios, and thus minimizing their energy consumptions and maximizing the network lifetime continuously intensify the interest of researchers in the development of energy-efficient wireless transmission schemes. Further, a trade-off inevitably arises in simultaneously maximizing the network lifetime and the application performance. For this challenge, a cross-layer optimization scheme is usually adopted by the related works because it can coordinate resources allocated to different layers to achieve globally

☆ This work was supported by the National Science Council Taiwan, under grant NSC101-2221-E-126-001-MY3.

* Corresponding author. Tel.: +886 422840688; fax: +886 422851410.
E-mail addresses: chhliu@pu.edu.tw (J.-S. Liu), jichiangt@nchu.edu.tw (J. Tsai).

http://dx.doi.org/10.1016/j.adhoc.2015.09.007
1570-8705/© 2015 Elsevier B.V. All rights reserved.

optimal performance for various objective functions. Nevertheless, due to the nature of wireless multihop transmission, distributed algorithms for, e.g., routing and scheduling, are more practical than the cross-layer optimization counterparts that would be usually operated in a centralized manner. Thus, how to preserve theoretical benefits from a centralized optimization method while adapting to the distributed computation environment motivates our research to develop decentralized approaches based on mathematical programming models.

As another perspective of research, Ahlswede et al. introduce in their seminal work [1] that by allowing intermediate nodes to perform coding operations in addition to pure packet forwarding, network coding (NC) can achieve the maximum multicast rate and thus can improve the overall network throughput. Following that, Li. et al. [2] show the fact that linear network coding suffices to achieve the maximum rate, and further Ho et al. [3] show that random linear codes can be used to achieve the linear network code rate asymptotically. More explicitly, the wireless broadcast nature enables a wireless station to broadcast data to all neighboring nodes. Each station or node can then overhear packets from the multicast source or any neighboring nodes, and act as a *router* or *forwarder* to forward data to multicast destinations. Nevertheless, caused by lossy wireless channel, different nodes could overhear the packets from the same router but might lose different packets, thus requiring the router to retransmit the lost packets. To resolve this issue, the authors in [4,5] exhibit intra-flow network coding for loss recovery of multicast traffic, and show that the number of retransmissions required for loss recovery is significant reduced.

Now, given the capability of network coding, it still remains significant challenges for a cross-layer optimization in wireless multihop networks even with centralized approaches. For example, it had been readily shown in [6–8] that the general problem of interference-free scheduling for multihop wireless networks is NP-hard to a centralized algorithm even without network coding. Here, by adopting intra-flow NC to extend the capability of routing while addressing the other problems arising from different layers, we face an even more challenging scheduling subproblem in the coded networks wherein each hyperlink (or hyperarc in [9]) has different probability to be activated and each transmission mode (or network component in [10]) has different probability to be selected for operation among the various trade-offs between throughput and lifetime utilities in a joint objective function, which should be determined simultaneously for both hyperlink and transmission mode, and all have to be done in a distributed manner consistently.

Related works. In the following, we review related works in four categories: joint cross-layer optimization, scheduling design, resource allocation on NC, and other related works.

1) Joint cross-layer optimization. For wireless sensor networks (WSNs) without NC, the authors in [11] study the problem of joint routing, link scheduling and power control to support high data rates and propose an algorithm to minimize the total average energy consumption in such networks. In [12], the authors consider a joint optimal design of physical, MAC, and routing layers to maximize the lifetime of WSNs. Specifically, they use TDMA as their MAC to formulate the optimization problem as a mixed integer convex problem, which can be solved with standard techniques such as interior point methods. In addition, the authors in [13] develop a unifying framework to understand the trade-off between the application layer performance and the lifetime of a WSN in which nodes can adopt their source rates so that the network operates at an optimal set of source rates that can jointly maximize the network utility and lifetime. In addition, many other cross-layer solutions for WSNs can be found in the survey paper [14].

2) Scheduling design. As a seminal work on scheduling, Tassiulas and Ephremides [15] obtained a link scheduling policy that attains the maximum possible throughput in presence of arbitrary scheduling constraints, by scheduling in each time slot an independent set (in the link interference or conflict graph) that has the maximum aggregate queue length. Afterward, the maximum weighted independent set (MWIS) problem of finding the independent set with the maximum weight involved is known to be a bottleneck of the wireless utility maximization problem [16]. As noted in [17], the scheduling-relevant formulations often suffer from two shortcomings: 1) the optimization problem could be intractable when the network size is large (i.e., it is NP-hard), and 2) the optimization problem could be amenable to centralized implementation only.

3) Resource allocation on NC. By randomly mixing a sequence of native packets in the same multicast session together, the empirical study [4] also shows that intra-flow or inter-session NC can be MAC-independent and have the practical benefits of mixing packets with low complexity. Given that, for an unicast traffic problem with intra-flow NC, the work [18] proposes a rate control scheme to control the forwarding data rate and improve NC efficiency under a fixed physical bit rate. A similar optimization problem in [19] has also been investigated for unicast traffic but further extended to solve the problem of adapting the transmission bit rate of each node. Apart from the above, certain aspects about subgraph selection for intra-flow multicast NC have been already revealed and summarized in [9].

4) Other related works. Complementing the approaches categorized above, there are other related works still worth mentioned here. For example, the problem of achieving min-cost multicast in networks has been studied [20], and the rate control problem for the multicast flows had been addressed [21], all by means of network coding. As a very useful tool, game theory is also utilized to accommodate the framework of network coding to achieve the maximum multicast in WSNs. Specifically, the work in [22] shows that a generalized butterfly network can be analyzed as a two-source unicast coded network, and its robustness had been investigated by game theory with the desired solution to reach equilibrium. In addition, the authors in [10] jointly consider links, routes, and network components with a nonlinear cubic game, and constitute an unconstrained optimization problem to be sequently solved with a fictitious play (FP) technique. Recently, in [23] we propose a cross-layer optimization formulation to jointly maximize two different performance utilities in wireless multihop networks with network coding. However, the previous work only considers for transmission modes, in contrast to the distributed approach presented here that accounts for both hyperlink and transmission mode. In fact, these metrics (transmission mode and hyperlink) could be formulated as two players in a matrix game, but a programming model for the relevant problem could be usually solved by a centralized approach, and would be time-consuming due to its NP nature, as shown in our previous work [24].

Thus, instead of considering centralized approaches, here we take into account the advances of intra-flow network coding for multicast sessions, and propose a cross-layer formulation of general network utility maximization (NUM) that can accommodate routing, scheduling and stream control from different layers in such coded wireless networks for maximizing network lifetime and system performance, realized in a distributed manner. In fact, the problem of maximizing the twin objectives of network lifetime and system performance that could conflict with each other results in a multi-criterion or vector optimization problem shown in Section 4.7 of [25]. A common approach to such a problem is to introduce a system parameter ($\lambda \in [0, 1]$) that can control the desired trade-off between the conflicting metrics with a weighted sum, as shown in [13]. Besides, in this work, we concurrently consider hyperlink scheduling and transmission mode scheduling in addition to many other constraints to be involved, while requiring no centralized optimization scheme to converge to the optimal solutions that may involve exponential number of subproblems required by a NP problem. Specifically, for the resulted quadratically constrained quadratic programming (QCQP) problem that is NP-hard in general,[1] we conduct a distributed algorithm based on alternate convex search (ACS) [26] and primal decomposition (PD) [27,28] to decompose the biconvex programming model to its subproblems resolved by subgradient-based algorithms, combined to iteratively approach the solution of this complex routing, stream control, and scheduling problem. In the environment of wireless multihop network, where nodes may be mobile and channels are usually error-prone and time-varying, our method can obtain a reasonable trade-off approximating the optimal within a reasonable time budget, which may be more preferred than the optimal solution that could be only obtained by a global optimization algorithm, e.g., GOP [29], supported by perfect parameters and calculated in a centralized manner to evaluate $2^{|I|}$ nonlinear subproblems in the worst case. Alternatively, block-relaxation based methods, such as ACS just noted, are usually used to alleviate intractable problems due to nonconvex and combinatorial nature by dividing the variable set involved into disjoint blocks and iteratively optimizing only the variables of an active block while leaving those of the other blocks to be fixed, which have been successfully applied to resolve some well-known problems, e.g., nonnegative matrix factorization (NMF) problem [30] and bilinear matrix inequalities (BMIs) problem [31]. However, these methods are mainly centralized, and implementing them for the scheduling constraints in question distributively is not clearly known. Similarly, although primal decomposition based methods, which may be thought of as an extension of the former, have been successfully applied to solve more recent optimization problems for communication and signal processing [27,28,32], these problems are typically convex and did not involve the non-convexity of quadratic scheduling constraints to be resolved here in a distributed manner. Thus, we elaborate in this work a distributed approach to resolve the non-convexity by means of an idea like ACS and PD, resulting in a simple and efficient method which also accounts for the maximum weighted independent set (MWIS) problem to be involved. Finally, when compared with the other related works, our work has its own characteristics as summarized as follows:

- Unlike the conventional optimization approaches, e.g., [11–13,33,34], that may or may not consider multiple objectives, but all pay no attention to network coding, our work contributes on a cross-layer optimization formulation that can actually account for such a coding scheme on wireless multihop networks.
- When compared with the related works with network coding just surveyed, e.g., [4,9,10,18–21,23], which contribute valuable results on the various studies with or without transmission modes, our work explicitly dedicates to a cross-layer optimization framework with two scheduling primitives, namely hyperlink and transmission mode, specific to the coded network while leaving its routing decision to be realized through our various cross-layer constraints accommodating also other design criteria to be involved, which leads to an optimal trade-off among the diverse objectives across multiple layers.

The remainder of this paper is organized as follows. In Section 2, we first introduce hyperlink and transmission mode considered in the network coding-based wireless networks. Then, we propose our joint lifetime-utility cross-layer optimization with concurrently scheduling on hyperlink and transmission mode in Section 3. Following that, we extend the cross-layer multi-objective programming model to a distributed algorithm in Section 4. To know its efficiency, the optimization framework along with the distributed algorithm resulted is examined numerically in Section 5. Finally, conclusions are drawn in Section 6.

2. Network coding-based network model

Consider a wireless multicast network as a directed hypergraph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{L}$ is the set of hyperlinks. In this context, a hyperlink $(i, J) \in \mathcal{L}$ for wireless network coding represents a one-hop broadcast transmission, wherein $i \in \mathcal{N}$ is the transmitter and $J \subseteq \mathcal{N}$ is a set of receivers with the same multicast address that can receive the packets from i owing to the broadcast nature of the wireless channel. When J contains only one node j, the hypergraph resulted would be reduced to a conventional graph model.

For ease of exposition, we consider a well-known butterfly wireless network with network coding shown in Fig. 1(a) as our example,[2] whose objective is to multicast packets

[1] The NP-hardness of QCQP can be well seen from the fact that any two constraints $x(x-1) \le 0$ and $x(x-1) \ge 0$ are equivalent to the constraint $x(x-1) = 0$, which is in turn equivalent to the constraint $x \in \{0, 1\}$. That is, any 0–1 integer program can be formulated as a quadratically constrained quadratic program. Because 0–1 integer programming is NP-hard in general, QCQP is also NP-hard.

[2] Some examples based on the classical wireless butterfly network had also been given in the literature for different purposes. For instance, a similar example in [35] was exhibited in bit level rather than packet level demonstrated here.

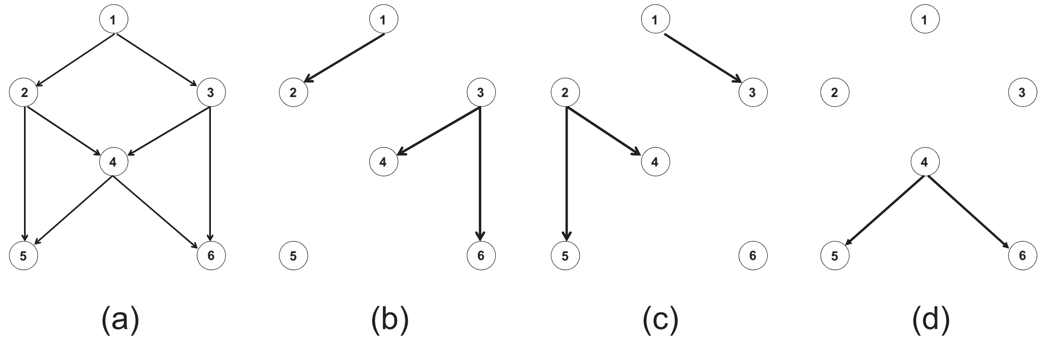

Fig. 1. Butterfly wireless network example: (a) original network, (b) transmission mode 1: {(1,{2}),(3,{4,6})}, (c) transmission mode 2: {(1,{3}),(2,{4,5})}, and (d) transmission mode 3: {(4,{5,6})}.

Table 1
Scheduling results for the network coding-based wireless network in Fig. 1(a).

Time slot	1	2	3	4	5	6	7	8
Schedule	$1\overrightarrow{p_1}3$	$3\overrightarrow{p_1}4$ $3\overrightarrow{p_1}6$ $1\overrightarrow{p_2}2$	$2\overrightarrow{p_2}4$ $2\overrightarrow{p_2}5$ $1\overrightarrow{p_3}3$	$4\overrightarrow{p_1 \otimes p_2}5$ $4\overrightarrow{p_1 \otimes p_2}6$	$3\overrightarrow{p_3}4$ $3\overrightarrow{p_3}6$ $1\overrightarrow{p_4}2$	$2\overrightarrow{p_4}4$ $2\overrightarrow{p_4}5$ $1\overrightarrow{p_5}3$	$4\overrightarrow{p_3 \otimes p_4}5$ $4\overrightarrow{p_3 \otimes p_4}6$	$3\overrightarrow{p_5}4$ $3\overrightarrow{p_5}6$ $1\overrightarrow{p_6}2$
Transmission mode		1	2	3	1	2	3	1

originating at source node 1 to both destination nodes 5 and 6. A classical collision model is assumed in this example for simplicity. That is, a collision will occur if multiple transmissions reach a node in the same time slot in spite of the distances or SINR values resulted from different sources destined to the node. Thus, for solving this multicast problem, we need to schedule conflict-free transmissions that can avoid performance loss. Specifically, given a sequence of packets, p_1, p_2,..., to be transmitted, a possible solution or schedule is exemplified in Table 1, where $a\overrightarrow{p_i}b$ denotes the transmission of packet i from node a to node b, and $p_i \otimes p_j$ denotes the XOR operation of two packets p_i and p_j to form a coded packet. Given the five hyperlinks, (1, {2}), (1, {3}), (2, {4, 5}), (3, {4, 6}), and (4, {5, 6}) for illustration, in time slot 2, we have the scheduling result that node 3 can multicast p_1 to nodes 4 and 6 while node 1 can transmit p_2 to node 2 at the same time without inference. Thus, the set of hyperlinks that can be simultaneously activated, i.e., {(1,{2}),(3, {4, 6})}, is therefore called a transmission mode and Fig. 1(b) explicitly shows this mode along with the others in (c) and (d). Clearly, when compared with the traditional routing, which requires a period of four time slots for transmitting two packets, the network coding scheme requires only three time slots, each involving a transmission mode, to complete the transmission. More explicitly, the destination node 6 (resp. 5) can obtain p_1 (resp. p_2) at slot 2 (resp. slot 3) and then use $p_1 \otimes (p_1 \otimes p_2)$ (resp. $p_2 \otimes (p_1 \otimes p_2)$) to obtain p_2 (resp. p_1) at slot 4. In other words, the gain of network coding is realized by a simple form of (random) linear network coding consisting of node 4 performing the XOR operation $p_{2i-1} \otimes p_{2i}$ at time slot $3i+1, i = 1, 2, \ldots,$ and using the wireless multicast advantage to send the coded packet to node 5 and node 6 in a single transmission, which can be further decoded with the aid of obtained packets (or say, remedies in [9]).

Note that the classical collision channel mode just exemplified is only used for ease of exposition. In the following work, it is extended to the more realistic SINR-based channel model with the relevant definition given as follows:

Definition 1. $\xi \subset \mathcal{L}$ is a set of links that can be concurrently activated without violating the minimum SINR for communication, ζ. That is, all the receivers of the concurrent links in ξ must have their SINR values higher than the requirement ζ. If ξ can satisfy this constraint, it is called a *transmission mode*.

Remark. In the previous works, e.g. [23,36,37], transmission modes are conventionally considered to have different activity probabilities while hyperlinks (or links) in such a mode are commonly assumed to be deterministic from the mode's viewpoint; that is, when a transmission mode is scheduled, all of its hyperlinks (or links) will be simultaneously activated no matter what. It had been considered in [10,24] that hyperlinks (or links) can have their own activity probabilities summing to one, independent of transmission modes (or network component sets), resulting in a general payoff on a min-max zero-sum game between hyperlinks and transmission modes in a network-coded wireless network. This definition extends the conventional scheduling problem to take into account the fact that these scheduling metrics can have different activity probabilities so that the network can be optimized by considering these scheduling metrics at the same time. However, such a generalization leads to a nonlinear and even nonconvex optimization problem which, in previous works, can be only resolved in a centralized manner.

3. Joint lifetime-utility cross-layer optimization with hyperlink and transmission mode scheduling

After introducing the hyperlink and transmission mode based on network coding, in the sequel we aim to solve a joint lifetime and utility optimization problem with a specific focus on solving its subproblem of hyperlink scheduling and transmission mode scheduling tailored for the multi-objective optimization. For this aim, we introduce in the following different relevant components to complete our programming model specific to the optimization problem.

3.1. Network coding-based transmission model

It is emphasized that, in this work, rather than only formulating the scheduling problem as exemplified in Section 2, our aim is actually to complete a cross-layer optimization scheme, which also accounts for routing and stream control in the network layer and the transport layer. Hence, we should introduce the other variables with respect to these layers, in addition to those corresponding to the hyperlink scheduling and transmission mode scheduling to be introduced.

To start with, we consider a set of multicast sessions to be transmitted for the transport layer in the coded network. Herein, a multicast session is denoted by its source node $s \in S \subset \mathcal{N}$ multicasting packets to its destination node set $\mathcal{T}_s$. Further, for the network layer, we let f_{iJj}^{st} denote the information flow rate from source s to destination node $t \in \mathcal{T}_s$ over hyperlink (i, J) and intended to node $j \in J$. Then, for a multicast session where source s wants to transmit on a rate of x_s to its set of destination nodes $\mathcal{T}_s$, we have the flow conservation law as

$$\sum_{\{J|(i,J)\in\mathcal{L}\}} \sum_{j\in J} f_{iJj}^{st} - \sum_{j\in\mathcal{N}} \sum_{\{I|(j,I)\in\mathcal{L},i\in I\}} f_{jIi}^{st} = x_{i,s}, \forall i \in \mathcal{N}, \forall s \in \mathcal{S}, \forall t \in \mathcal{T}_s \tag{1}$$

$$\text{where } x_{i,s} = \begin{cases} x_s, & \text{if node } i \text{ is the source of session } s \\ -x_s, & \text{if node } i \text{ is the sink of session } s \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Obviously, the session rate and then the flow rate in the upper layers should be realized by the hyperlink capacity for the MAC layer and the data rate for the physical layer. Thus, we proceed to establish the relationship between the upper layers and the lower layers. Specifically, with network coding, an intermediate node can generate output data by performing, e.g., (random) linear coding on the received data packets as exemplified above. Therefore, network coding is often thought of as a generalized routing by allowing the information to be modified instead of direct packet relaying. By doing so, flows with different destinations in a multicast session are allowed to share the network capacity. Here, denoting by g_{iJ}^{s} the physical flow rate from source s to the set of destination nodes $\mathcal{T}_s$ over (i, J), we can specify the constraint that, with network coding, the sum of flow rate on hyperlink (i, J) should not exceed the physical rate, as follows:

$$\sum_{j\in J} f_{iJj}^{st} \le g_{iJ}^{s}, \forall (i,J) \in \mathcal{L}, \forall s \in \mathcal{S}, \forall t \in \mathcal{T}_s \tag{3}$$

3.2. Concurrently hyperlink and mode scheduling

In the following, we introduce the hyperlink and transmission mode scheduling subproblem. In particular, unlike the previous works [12,23,38], which usually focus on processing a set of transmission modes [23], a set of wireless network realizations [38], or a set of time slots [12] for achieving their specific performance metrics, we consider in this work a decision making problem with multiple specific objectives, and extend the previous to model both hyperlink and transmission mode as the metrics to be concurrently satisfied in the utility maximization. Moreover, beyond certain centralized approaches based on programming techniques or fictitious play, e.g., in [10], our programming model can accommodate multiple utilities and can be further extended to be distributed algorithms, which are more practical in wireless networks. To show this, we afterward refer to the hyperlinks by $\{l_1, l_2,\ldots, l_m\}$, and the transmission modes by $\{\xi_1, \xi_2,\ldots \xi_n\}$ in a fixed order, respectively. Here, in terms of matrix game, in which two players are row player (or Player I) and column player (or Player II), and a game between them is determined by a $m \times n$ matrix A, we could consider hyperlink as row player, equipped with m pure strategies corresponding to $\bar{M} = \{1, \ldots, m\}$ rows, and transmission mode as column player, equipped with n pure strategies corresponding to $\bar{N} = \{1, \ldots, n\}$ columns. Given that, the set of mixed strategies for Player I or hyperlink can be defined as

$$P = \left\{ p = (p_1, \ldots, p_m) | p_u \ge 0, \forall u = 1, \ldots, m, \sum_{u=1}^{m} p_u = 1 \right\} \tag{4}$$

Similarly, the set of mixed strategies for Player II or transmission mode can be defined as

$$Q = \left\{ q = (q_1, \ldots, q_n) | q_v \ge 0, \forall v = 1, \ldots, n, \sum_{v=1}^{n} q_v = 1 \right\} \tag{5}$$

That is, the player I's move is to choose a row vector $p = (p_1, \ldots, p_m)$ with p_u, $1 \le u \le m$, nonnegative and summing to one. The player II's move is to choose a column vector $q = (q_1, \ldots, q_n)$ with $q_v, 1 \le v \le n$, likewise nonnegative and summing to one. The players make their moves independently and the game is concluded by the column player paying the expected number pAq^T to the row player. Inspired by the game, we resolve our scheduling subproblem in the cross-layer optimization by weighting each performance metric differently in the objective function, and specifying our scheduling constraints with the payoff matrices to be introduced. Thus, the objective value can be indirectly affected by the payoffs for a smoother tradeoff between the performance metrics, and the resulted mixed strategies or probabilities can be used for the scheduling. To this end, we would derive the constraints with respect to the scheduling subproblem involved first. Specifically, we denote by $r_{u,v}$ instead of r_{iJ}^{k} to be the transmission rate of a hyperlink $l_u = (i, J)$ scheduled by p_u and q_v, and denote by g_u^s instead of $g_{i,J}^{s}$ the physical flow rate of session s over $l_u = (i, J)$, wherever the hyperlink (i, J) is not required to specify its i and J and can be simply denoted by l_u with an index u in $\bar{M}$ that corresponds to $\{l_1, l_2, \ldots, l_m\}$, and k can be replaced by v in $\bar{N}$ that corresponds to $\{\xi_1, \xi_2,\ldots \xi_n\}$. With these concise notations, we can formulate the constraint that the physical flow rate accounting for all sessions $s \in S$ on a hyperlink l_u should be upper bounded by the physical capacity $r_{u,v}$ scheduled over this hyperlink by all transmission modes q_v, when p_u is given, as follows:

$$\sum_{s\in\mathcal{S}} g_u^s \le \sum_{v=1}^{n} p_u r_{u,v} q_v,\ u \in \bar{M} \tag{6}$$

In addition, the lifetime of node i can be represented by considering its initial energy consumed by all its hyperlinks activated by both hyperlink and transmission mode scheduling. That is,

$$T_i = \frac{\mathcal{E}_i}{\sum_{\{u:tr(u)=i\}} \sum_{v=1}^{n} p_u e_{u,v} q_v} \tag{7}$$

where $tr(u)$ is the transmitter of hyperlink l_u, $\mathcal{E}_i$ denotes the initial energy of node i, and $e_{u,v}$ represents the average energy spent by l_u when scheduled to be active by p_u and q_v.

Now, the lifetime in (7) might be utilized as an objective to be optimized directly. However, it could be more convenient to represent it by the relevant constraints in our programming model. Specifically, by exchanging the left-hand side of (7) and the denominator in its right-hand side as $\sum_{\{u:tr(u)=i\}} \sum_{v=1}^{n} p_u e_{u,v} q_v = \frac{\mathcal{E}_i}{T_i}$, defining the inverse lifetime as $\varrho_i = \frac{1}{T_i}$, and estimating such a lifetime by using inequality instead of equality, we have the following lifetime constraint:

$$\sum_{\{u:tr(u)=i\}} \sum_{v=1}^{n} p_u e_{u,v} q_v \le \varrho_i \mathcal{E}_i,\ \forall i \in \mathcal{N} \tag{8}$$

3.3. Cross-layer variables and weighted multi-objective function

Now, from a cross-layer viewpoint, we can think of session rates x_s as stream control variables, flow rates f_{iJj}^{st} as routing variables, and hyperlink strategies p_u and transmission mode strategies q_v as scheduling variables. In terms of these variables, we choose to use the weighted sum approach well-known in the literature [39] to accommodate the multiple objectives to be involved. Specifically, by using $\lambda = (\lambda_1, \lambda_2)$ as the trade-off weight, we can formulate the weighted sum as our objective with a proper choice of $\lambda_1 + \lambda_2 = 1$ when only two metrics involved, and if more than two metrics involve, the corresponding objective can be similarly obtained.

As noted before, our aim is to concurrently optimize lifetime and throughput. The two metrics, however, are conflict in nature, and we should particularly define different utilities to reach a suitable compromise between the heterogenous metrics. Specifically, for the metric of lifetime, we define utility U_i of a node i as a function of its lifetime T_i and the network lifetime T_N as $\min_{i\in\mathcal{N}} T_i$. In addition, we let $U_i(T_i) = \varrho_i^{\gamma}$ (where $\varrho_i = \frac{1}{T_i}$, as defined previously) to represent node i's normalized power dissipation with respect to its initial energy $\mathcal{E}_i$. When properly designed, U_i is a strictly convex and increasing function with, e.g., $\gamma \ge 2$, and maximizing the lifetime is equivalent to minimizing the penalty of U_i. More specifically, we assume $\gamma = 2$ that is also considered in the previous work [13], and proceed to solve the sum of aggregated utility maximization problem, max $\sum_{i\in\mathcal{N}} -U_i$ or min $\sum_{i\in\mathcal{N}} U_i$. Similarly, for the metric of session data rate, we adopt the well-known *log* function for fair resource allocation as our throughput utility, and consider to solve its utility maximization problem, max $\sum_{s\in\mathcal{S}} U_s$.

3.4. Cross-layer programming model

Consequently, by taking into account the objectives to be combined along with the constraints just introduced, we can formulate the joint lifetime and utility maximization with network coding and multiple payoffs (**JLUP**) as follows:

$$
\begin{array}{llll}
\textbf{maximize} & \mu(\lambda) = \lambda_1 \dfrac{\sum_{s\in\mathcal{S}} \log(x_s)}{W_1} - \lambda_2 \dfrac{\sum_{i\in\mathcal{N}} \varrho_i^2}{W_2} & & (a) \\
\textbf{subject to} & \sum_{\{J|(i,J)\in\mathcal{L}\}} \sum_{j\in J} f_{iJj}^{st} - \sum_{j\in\mathcal{N}} \sum_{\{i|(j,I)\in\mathcal{L},i\in I\}} f_{jIi}^{st} \ge x_{i,s}, & \forall i \in \mathcal{N}, \forall s \in \mathcal{S}, \forall t \in \mathcal{T}_s & (b) \\
& \sum_{\{u:tr(u)=i\}} \sum_{v=1}^{n} p_u e_{u,v} q_v \le \varrho_i \mathcal{E}_i, & \forall i \in \mathcal{N} & (c) \\
& \sum_{j\in J} f_{iJj}^{st} \le g_{iJ}^{s} & \forall (i,J) \in \mathcal{L}, \forall s \in \mathcal{S}, \forall t \in \mathcal{T}_s & (d) \\
& \sum_{s\in\mathcal{S}} g_u^s \le \sum_{v=1}^{m} p_u r_{u,v} q_v, & \forall u \in \bar{M} & (e) \\
& \sum_{u=1}^{m} p_u = 1 & & (f) \\
& \sum_{v=1}^{n} q_v = 1 & & (g) \\
& 0 \le p_u \le 1, & \forall u \in \bar{M} & (h) \\
& 0 \le q_v \le 1, & \forall v \in \bar{N} & (i)
\end{array}
\tag{9}
$$

where W_1 and W_2 are used to balance the possibly very different quantities resulted from the heterogeneous utility functions so that λ_1 and λ_2 can have their values in the same scale for the tradeoff. Apart from the above, the constraints contribute themselves as follows. (9(b)) represents the flow conservation law, showing that the total output rate should be equal to or greater than the corresponding input rate except that for sinks. (9(c)) is the lifetime constraint, saying that the energy spent by node i in its lifetime should be equal to or less than its total (or initial) energy. (9(d)–(e)) show the flow-sharing properties of network coding. (9(f)–(g)) denote the scheduling constraints, and (9(h)–(i)) simply represent the validity constraints upon the scheduling variables.

4. Distributed algorithm

In this section, we extend the programming model just introduced to a distributed algorithm. Specifically, by means of decomposition theory that provides the mathematical richness to construct an analytical framework for the design of distributed control of network [27], we introduce a systematic approach on the design of distributed algorithm utilizing the capability of network utility maximization formulation revealed by the mathematical programming to approximate the optimal results for the multihop intra-flow NC multicast optimization problem.

4.1. Two level decomposition based on alternate convex search and primal decomposition

As shown in (9(c), (e)), even without quadratic terms in the objective function, which could be still thought of as in its reduced quadric form, JLUP actually involves quadratic constraints and could be classified as a quadratically constrained quadratic programming (QCQP) problem that is NP-hard in general. To solve such a hard problem, one may resort to an optimization tool such as global optimization algorithm (GOP) [29], which would needs $2^{|I|}$ nonlinear subproblems to be solved to obtain a new lower bound to the problem in each iteration, where $|I|$ denotes the set of the connected variables involved. Clearly, the time complexity that a global optimization may involve could hardly be offered by a distributed algorithm. Thus, on the one hand, instead of directly using an exact global optimization approach, we adopt the concept of alternate convex search (ACS) [26], which is a minimization method to find partial optimum and is a special case of block-relaxation methods where the variable set is divided into disjoint blocks [40]. On the other hand, by surveying the literature for convex optimization, we find that the decomposition theory is well studied to provide a variety of decomposition approaches suitable for different types of problems, among which primal decomposition (PD) is considered more suitable for the problems with coupling variables [27].

For our JLUP, we develop a new distributed algorithm inspired by both ACS and PD. Specifically, we can see that, although defining $\varrho_i = 1/T_i$ resolves the nonlinear problem caused by the raw form of lifetime T_i, the nonlinear (or even further, nonconvex) terms $p_u r_{u,v} q_v$ and $p_u e_{u,v} q_v$ still remains to be cumbersome. In fact, if $[r_{u,v}]$ and $[e_{u,v}]$ are all positive semidefinite, the QCQP becomes a convex optimization problem. This subclass of problems is solvable in polynomial time by using, e.g., the second order cone programming method [41]. However, in this work, the matrices are not necessary positive semidefinite and finding the global optimal solution to a general QCQP is NP-hard. Nevertheless, the quadratic constraints involved are biconvex, and we could solve it approximately to approach the optimal solution in a distributed manner. That is, if q_v (or p_u) are fixed, the variables will be decoupled with the others, which transforms the QCQP to a convex program. Thus, we can apply ACS to our problem by considering only two blocks of variables defined by the convex subproblems that are activated in cycles, and then decompose JLUP into two levels of optimization problem based on the primal decomposition theory to approximate the optimum, which follows the decomposition ideas introduced by Benders [42], Geoffrion [43], and Floudas and Visweswaran [29] that are not restricted to be applied to convex optimization only. More explicitly, if we choose p_u to be coupling variables, the two level decomposition of JLUP can be represented as follows:

- Master primal problem:

$$\begin{aligned}&\mathbf{MPP}: \mathbf{max}\,\mu^*(\mathbf{p})\\ &\text{subject to } 0 \le p_u \le 1, \forall u \in \bar{M}, \text{ and } \sum_{u=1}^{m} p_u = 1\end{aligned} \tag{10}$$

- Low level problem:

$$\begin{array}{llll}
\mathbf{LLP}: & \mathbf{maximize}\ \mu(\lambda) & & (a)\\
\mathbf{subject\ to} & \displaystyle\sum_{\{J|(i,J)\in\mathcal{L}\}}\sum_{j\in J} f_{iJj}^{st} - \sum_{j\in\mathcal{N}}\sum_{\{I|(j,I)\in\mathcal{L}, i\in I\}} f_{jIi}^{st} \ge x_{i,s}, & \forall i\in\mathcal{N}-\{t\}, \forall s\in\mathcal{S}, \forall t\in\mathcal{T}_s & (b)\\
& \displaystyle\sum_{\{u:tr(u)=i\}}\sum_{v=1}^{n}\widetilde{e_{u,v}}q_v \le \varrho_i\mathcal{E}_i, & \forall i\in\mathcal{N}-\{t\} & (c)\\
& \displaystyle\sum_{j\in J} f_{iJj}^{st} \le g_{iJ}^{s} & \forall (i,J)\in\mathcal{L}, \forall s\in\mathcal{S}, \forall t\in\mathcal{T}_s & (d)\\
& \displaystyle\sum_{s\in\mathcal{S}} g_u^s \le \sum_{v=1}^{n}\widetilde{r_{u,v}}q_v, & \forall u\in\bar{M} & (e)\\
& \displaystyle\sum_{v=1}^{n} q_v = 1 & & (f)\\
& 0\le q_v\le 1, & \forall v\in\bar{N} & (g)
\end{array} \tag{11}$$

where $\widetilde{e_{u,v}} = p_u e_{u,v}$, $\widetilde{r_{u,v}} = p_u r_{u,v}$, and $\lambda = \lambda_1 + \lambda_2 = 1$ with $\lambda_1, \lambda_2 \in (0, 1)$.

As shown above, the master primal problem **MPP** is designed to respond in the duty of updating the coupling variables p_u while the low level optimization problem **LLP** is in charge of achieving optimum of the other variables. Now, given the initial values of p_u the distributed algorithm can be implemented in the order of first using **LLP** to obtain the optimal variables of the lower level problem, and then using **MPP** to update the coupling variables based on the optimal variables just resulted. In terms of the methodologies involved, **LLP** and **MPP** can be said to resemble the two steps in ACS whose variables are now decoupled by p_u according to PD, for iteratively approaching the partial optimum and thus approximating the optimal results of this NP problem.

Remark. As shown above, our formulation complies with the nonconvex NLP (nonlinear programming) problem in [29] wherein the objective function and equality constraints are biconvex while inequality constraints are biaffine, as noted in [26]. The GOP algorithm [29] for such a problem can terminate in a finite number of steps for any given $\epsilon > 0$ at the global (ϵ-)optimum (see Theorem 4.11 and Corollary 4.12 in [26]). That is, our QCQP problem satisfying the above can have a global maximum. In addition, as noted in [26] (with its notations, f, z, x, and y shown below), ACS resolves a biconvex optimization problem by iteratively solving the optimization problem $\min\{f(x, y_i)\}$ for fixed y_i and setting $x_{i+1} = x^*$ if an optimal x^* is found, and then solving the optimization problem $\min\{f(x_{i+1}, y)\}$ for fixed x_{i+1} and setting $y_{i+1} = y^*$ if an optimal y^* is found. Suppose that the sequence generated, $\{z_i\}_{i\in\mathbb{N}}$, is contained in a compact set, and for each $z^* = (x^*, y^*)$ of the sequence $\{z_i\}_{i\in\mathbb{N}}$, the optimal solution of $y = y^*$ and $x = x^*$ obtained by the ACS iterates shown above is unique. Then, all accumulation points z^* which lies in the interior by using ACS are stationary points of the objective function f (see Corollary 4.10 in [26]). However, as also noted in the literature, although an accumulation point z^* might be a partial optimum, it does not have to be a global optimum for the given biconvex optimization problem even if z^* is stationary. Thus, it is noted in the above that according to ACS, the partial optimum of **LLP** and **MPP** can be iteratively generated to *approximate* the global optimum for this NP problem. It is also evident in the numerical experiments of Section 5 that the approximating results based on ACS as well as primal decomposition (PD) are actually close to the optimum.

4.2. Partial Lagrangian reformulation for LLP

After the two level decomposition, **LLP** resulted is still complex and would not be easily solved even with a centralized approach. That is, a centralized approach directly applying ACS is not enough that only divides the variable set into disjoint blocks and in every step allows the variables of an active block to be optimized while keeping those of the other blocks to be fixed. This is because the scheduling subproblem imbedded in **LLP**, or *LLP scheduling* for short, corresponds to a maximum weight independent set problem, which is NP-hard in general, even though p_u are fixed and given. When considering its distributed implementation, we face the challenge not only caused by the maximization problem on the lifetime and utility but also the convergence problem on the scheduling. To see this, we relax the first two sets of constraints in (11) to form the partial Lagrangian as follows:

$$L(\boldsymbol{\varrho}, \boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{\kappa}, \boldsymbol{\psi}) = \frac{\lambda_1}{W_1}\sum_{s\in\mathcal{S}}\log(x_s) - \frac{\lambda_2}{W_2}\sum_{i\in\mathcal{N}}\varrho_i^2 + \sum_{i\in\mathcal{N}, s\in\mathcal{S}, t\in\mathcal{T}_s}\kappa_i^{st}\left(\sum_{\{J|(i,J)\in\mathcal{L}\}}\sum_{j\in J} f_{iJj}^{st} - \sum_{j\in\mathcal{N}}\sum_{\{i|(j,I)\in\mathcal{L}, i\in I\}} f_{jIi}^{st} - x_{i,s}\right) + \sum_{i\in\mathcal{N}}\psi_i\left(\varrho_i\mathcal{E}_i - \sum_{\{u,tr(u)=i\}}\sum_{v=1}^{n}\widetilde{e_{u,v}}q_v\right) \tag{12}$$

where $\{\kappa_i^{st}\}, \forall i\in\mathcal{N}-\{t\}, \forall s\in\mathcal{S}, \forall t\in\mathcal{T}_s$ and $\{\psi_i\}, \forall i\in\mathcal{N}-\{t\}, \delta_v, \forall v\in\bar{N}$, are the Lagrange multipliers corresponding to (11(b)) and (11(c)), respectively.

To explain the reason why the Lagrangian approach would be derived here, we turn back to note the fact that given p_u as constants, **LLP** has linear objective function and constraints. According to the optimization theory [25], the Slater constraint qualification conditions are always satisfied for a problem with a concave objective function and linear constraints, and thus **LLP** preserves the strong duality. Given that, for solving the lower level optimization problem, we can develop a primal-dual algorithm that updates the primal and the dual variables simultaneously and moves together toward the optimal points asymptotically. Moreover, to resolve the LLP scheduling subproblem embedded, we should also develop a distributed algorithm that can find the optimal transmission probability for each transmission mode and can seamlessly cooperate with the primal-dual algorithm to be derived, rather than only finding the subgradients with respect to the primal and dual variables involved. For this aim, we will develop a log-sum-exp approximation in the sequel, as a part of the overall primal-dual distributed algorithm for **LLP**. To start with, we continue the derivation of partial Lagrangian just obtained in (12), which can be further solved by finding the saddle points of $L(\varrho, x, f, g, q, \kappa, \psi)$ via the following dual problem (**DP**)

$$\mathbf{DP}: \min_{\boldsymbol{\kappa},\boldsymbol{\psi}\succeq 0}\Bigg(\max_{\boldsymbol{\varrho},\mathbf{x}\succeq 0}\Bigg[\frac{\lambda_1}{W_1}\sum_{s\in\mathcal{S}}\log(x_s) + \sum_{i\in\mathcal{N}}\left(-\frac{\lambda_2}{W_2}\varrho_i^2 + \psi_i\varrho_i\mathcal{E}_i\right) - \sum_{i\in\mathcal{N}, s\in\mathcal{S}, t\in\mathcal{T}_s}\kappa_i^{st}x_{i,s}\Bigg] + \max_{\mathbf{f},\mathbf{g},\mathbf{q}\succeq 0}\Bigg[\sum_{(i,J)\in\mathcal{L}, s\in\mathcal{S}, t\in\mathcal{T}_s, j\in J} f_{iJj}^{st}(\kappa_i^{st} - \kappa_j^{st}) - \sum_{i\in\mathcal{N}}\psi_i\left(\sum_{\{u,tr(u)=i\}}\sum_{v=1}^{n}\widetilde{e_{u,v}}q_v\right)\Bigg]\Bigg)$$
$$\text{subject to } (11(\text{d})) - (11(\text{g})) \tag{13}$$

Given the formulation, we can then solve it successively, and when proceeding, we can identify the scheduling subproblem (**SP**) involved, as follows:

$$\mathbf{SP}: \max_{\mathbf{f},\mathbf{g},\mathbf{q}\geq 0} \sum_{(i,J)\in\mathcal{L},s\in\mathcal{S},t\in\mathcal{T}_s,j\in J} f_{iJj}^{st}(\kappa_i^{st}-\kappa_j^{st}) - \sum_{i\in\mathcal{N}} \psi_i \left(\sum_{\{u,tr(u)=i\}} \sum_{v=1}^{n} \widetilde{e_{u,v}} q_v \right)$$
$$\text{subject to } (11\text{–d}) - (11\text{–g}) \tag{14}$$

SP is a linear programming problem on **f**, **g**, and **q**, wherein an optimal solution is the extreme point solution. We can solve this problem on **f** and **g** by taking into account the constraints (11(d),(e)) to simplify the constrained optimization, leading to the maximization problem (**MP**) on **q** as follows:

$$\mathbf{MP}: \max_{\mathbf{q}\geq 0} \sum_{v=1}^{n} q_v \sum_{u=1}^{m} \left(\widetilde{r_{u,v}}\omega_u - \widetilde{e_{u,v}}\psi_{tr(u)}\right)$$
$$\text{subject to } \sum_{v=1}^{n} q_v = 1 \tag{15}$$

where ω_u denotes the maximum differential backlog with respect to hyperlink l_u and will be discussed later. Given that, $\sum_{v=1}^{n}\left(\widetilde{r_{u,v}}\omega_v - \widetilde{e_{u,v}}\psi_{tr(u)}\right)$ can be sought of as the weight of the set of transmission modes $\Xi = \{\xi_1, \ldots, \xi_n\}$ with respect to $q = (q_1, \ldots, q_n)$ for scheduling, which corresponds to a maximum weight independent set problem. As a result, **MP** is NP-hard and difficult to be approximated even in a centralized way [44]. Nevertheless, we can still resolve it with a distributed approach based on the log-sum-exp approximation method in [17]. Specifically, the maximization can be approximated by the log-sum-exp function with a positive constant β,

$$\max_{\mathbf{q}\geq 0} \sum_{u=1}^{m} (\widetilde{r_{u,v}}\omega_v - \widetilde{e_{u,v}}\psi_{tr(u)}) \approx \frac{1}{\beta}\log\left(\sum_{v=1}^{n}\exp\left(\beta\sum_{u=1}^{m}\widetilde{r_{u,v}}\omega_v - \widetilde{e_{u,v}}\psi_{tr(u)}\right)\right) \tag{16}$$

Given that, we are now led to solve an approximated version of **MP**, off by an entropy term $-\frac{1}{\beta}\sum_{v=1}^{n} q_v \log q_v$, as shown as follows.

$$\mathbf{AMP}: \max_{\mathbf{q}\geq 0} \sum_{v=1}^{n} q_v \sum_{u=1}^{m} (\widetilde{r_{u,v}}\omega_v - \widetilde{e_{u,v}}\psi_{tr(u)}) - \frac{1}{\beta}\sum_{v=1}^{n} q_v \log q_v$$
$$\text{subject to } \sum_{v=1}^{n} q_v = 1 \tag{17}$$

Finally, the optimal solution to **AMP** can be obtained by

$$q_v^* = \frac{\exp\left(\beta\sum_{u=1}^{m}\left(\widetilde{r_{u,v}}\omega_u - \widetilde{e_{u,v}}\psi_{tr(u)}\right)\right)}{\sum_{v'=1}^{n}\exp\left(\beta\sum_{u=1}^{m}\left(\widetilde{r_{u,v'}}\omega_u - \widetilde{e_{u,v'}}\psi_{tr(u)}\right)\right)}, \quad \forall v \tag{18}$$

4.3. Back-pressure scheduling algorithm

We now turn back to explain the maximum differential backlog mentioned in Section 4.2. In this work, we develop a back-pressure approach inspired by the seminal work given in [15] based on the concept of using queue-length difference to equalize differential backlog. Our algorithm, however, is specified to drive the scheduling for each session, and give its results to support the intra-flow network coding and hyperlink scheduling to be followed. Specifically, with the back-pressure algorithm, each node maintains a separate queue for each destination, and for each of its hyperlinks $l_u = (i, J)$, node $i = tr(u)$ decides its target session by

$$s_u = \arg\max_{s\in S}\sum_{t\in\mathcal{T}_s}\left[\kappa_i^{st} - \kappa_j^{st}\right]_+ \tag{19}$$

and the maximum differential backlog over $l_u = (i, J)$ by

$$\omega_u = \max_{s\in\mathcal{S}}\sum_{t\in\mathcal{T}_s}\max_{j\in J}\left[\kappa_i^{s_ut} - \kappa_j^{s_ut}\right]_+ \tag{20}$$

In other words, for each hyperlink $l_u = (i, J)$, a session s with the maximum sum of queue-length differences between node $i = tr(u)$ and all its neighbors $j \in J$ over all the session sinks $t \in \mathcal{T}_s$ will be scheduled to transmit by node $i = tr(u)$ on its hyperlink $l_u = (i, J)$, and the amount to be transmitted is the maximum sum resulted from the session s. In this sense, the distributed scheduling algorithm uses back-pressure in an effort to equalize aggregate differential backlog so that each of the per-destination queues can be stable [15].

4.4. Intra-flow network coding and hyperlink scheduling

As the back-pressure scheduling implies, our distributed optimization is not required to have a predefined set of routes. That is, according to [9], our information flow rates (or routing variables) f_{iJj}^{st} with respect to $l_u = (i, J)$ are dynamically decided by the hyperlink scheduling:

$$f_{iJj}^{st} = \begin{cases} \sum_{v=1}^{n} q_v\widetilde{r_{u,v}}, & \text{if } s = s_u,\ j = \arg\max_{j\in J}\left(\kappa_i^{st} - \kappa_j^{st}\right), t \in \mathcal{T}_s, \\ & \quad\text{and } \kappa_i^{st} - \kappa_j^{st} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

That is to say, for each of its hyperlinks $l_u = (i, J)$, node $i = tr(u)$ will find a specific neighbor $j = \arg\max_{j\in J}(u_i^{s_ut} - u_j^{s_ut})$ with the maximum differential backlog among all the sink nodes $j \in J$ in session s_u, and gives this neighbor all the transmission rate $\sum_{v=1}^{n} q_v\widetilde{r_{u,v}}$.

With the intra-flow network coding, it can be realized that when a hyperlink $l_u = (i, J)$ gets an opportunity to transmit (resulted from the hyperlink scheduling), if $\omega_u > 0$, node $i = tr(u)$ will perform a random linear network coding on the packets of session s_u with destination $t \in \mathcal{T}_{s_{ij}}$ satisfying $u_i^{s_ut} - u_j^{s_ut} > 0$. As indicated in [3], the random linear network coding under consideration can be obtained by random linear combination with coefficients chosen from a finite field F with sufficient large field sizes. Given that, with a rate of $\sum_{v=1}^{n} q_v\widetilde{r_{u,v}}$, node $i = tr(u)$ can transmit the coded packets over hyperlink $l_u = (i, J)$ to $j = \arg\max_{j\in J}(u_i^{s_ut} - u_j^{s_ut})$. Otherwise, it will transmit NULL bits.

4.5. Low level update

Given all the above, we can now proceed to solve **LLP** effectively. In particular, given the optimal q_v^* in (18) for the transmission mode scheduling subproblem, solving convex **LLP** can thus resort to a primal-dual algorithm, which would update the primal and the dual variables simultaneously, moving together toward the optimal points asymptotically. For our problem, a primal-dual algorithm can be given by the following:

$$\begin{cases} \dot{\varrho}_i = \alpha_1 \left[\frac{\partial L(\boldsymbol{\varrho}, \boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{\kappa}, \boldsymbol{\psi})}{\partial \varrho_i} \right]^+_{\varrho_i}, & \forall i \in \mathcal{N} \\ \dot{x}_s = \alpha_2 \left[\frac{\partial L(\boldsymbol{\varrho}, \boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{\kappa}, \boldsymbol{\psi})}{\partial x_s} \right]^+_{x_s}, & \forall s \in \mathcal{S} \\ \dot{\kappa}_i^{st} = \alpha_3 \left[-\frac{\partial L(\boldsymbol{\varrho}, \boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{\kappa}, \boldsymbol{\psi})}{\partial \kappa_i^{st}} \right]^+_{\kappa_i^{st}}, & \forall i \in \mathcal{N} - \{t\}, \forall s \in \mathcal{S}, \\ & \forall t \in \mathcal{T}_s \\ \dot{\kappa}_t^{st} = \kappa_t^{st} = 0, & \forall s \in \mathcal{S}, \forall t \in \mathcal{T}_s \\ \dot{\psi}_i = \alpha_4 \left[-\frac{\partial L(\boldsymbol{\varrho}, \boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{\kappa}, \boldsymbol{\psi})}{\partial \psi_i} \right]^+_{\psi_i}, & \forall i \in \mathcal{N} - \{t\} \\ \dot{\psi}_t = \psi_t = 0, & \forall t \in \mathcal{T}_s \end{cases} \tag{22}$$

where $\alpha_i, i = 1, 2, 3, 4,$ are positive constants, and $[b]_a^+$ is $\max(0, b)$ if $a \leq 0$, and b if $a > 0$.

As shown in above, this algorithm requires a time-scale separation assumption that the stationary distribution involved can be converged instantaneously, which may not be easily implemented. Thus, we are demand by ourself to develop a stochastic counterpart to get rid of the cumbersome. To this end, we first show that according to the partial Lagrangian reformulation given in Section 4.2, each node in the primal-dual algorithm is conducted to iteratively decide by itself the optimal primal and dual variables to be adopted for the joint optimization. Specifically, given all variables in the current iteration, the subgradients of $L(\varrho, x, f, g, q, \kappa, \psi)$ with respect to the primal variables ϱ_i and x_s will be $\psi_i \mathcal{E}_i - 2\frac{\lambda_2}{W_2}\varrho_i$ and $\frac{\lambda_1}{x_s W_1} - \sum_{t \in \mathcal{T}_s} \kappa_s^{st}$, respectively. Similarly, the subgradient of $-L(\boldsymbol{\varrho}, \boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{\kappa}, \boldsymbol{\psi})$ with respect to the dual variables, κ_i^{st} and ψ_i, can be obtained by $x_{i,s} - (\sum_{\{J|(i,J) \in \mathcal{L}\}} \sum_{j \in J} f_{iJj}^{st} - \sum_{j \in \mathcal{N}} \sum_{\{i|(j,I) \in \mathcal{L}, i \in I\}} f_{jIi}^{st})$ and $\sum_{\{u,tr(u)=i\}} \sum_{v=1}^{n} \widetilde{e_{u,v}} q_v - \varrho_i \mathcal{E}_i$, respectively, if i is not a destination node $t \in \mathcal{T}_s$. Given that, we consider $\varrho, x, f, g, q, \kappa, \psi$ to be updated at time t_τ, $\tau = 1, 2, \ldots,$ with $t_0 = 0$ to get rid of the assumption. That is, by taking τ as the argument for each of these parameters, we can formulate the stochastic primal-dual subgradient distributed algorithm by solving the partial differential of $L(\varrho, x, f, g, q, \kappa, \psi)$ with respect to the primal and dual variables, and obtain their subgradients to update these variables in the $\tau + 1$th iteration by

$$\begin{cases} \varrho_i(\tau+1) = \left[\varrho_i(\tau) + \epsilon_1(\tau) \left(\psi_i \mathcal{E}_i - 2\frac{\lambda_2}{W_2} \varrho_i(\tau) \right) \right]^+, & \forall i \in \mathcal{N} \\ x_s(\tau+1) = \left[x_s(\tau) + \epsilon_2(\tau) \left(\frac{\lambda_1}{x_s W_1} - \sum_{t \in \mathcal{T}_s} \kappa_s^{st}(\tau) \right) \right]^+, & \forall s \in \mathcal{S} \\ \kappa_i^{st}(\tau+1) = \left[\kappa_i^{st}(\tau) + \epsilon_3(\tau) \left(x_{i,s}(\tau) - \left(\sum_{\{J|(i,J) \in \mathcal{L}\}} \sum_{j \in J} f_{iJj}^{st}(\tau) - \sum_{j \in \mathcal{N}} \sum_{\{i|(j,I) \in \mathcal{L}, i \in I\}} f_{jIi}^{st}(\tau) \right) \right) \right]^+, & \\ & \forall i \in \mathcal{N} - \{t\}, \forall s \in \mathcal{S}, \forall t \in \mathcal{T}_s \\ \kappa_t^{st}(\tau+1) = \kappa_t^{st}(\tau) = 0, & \forall s \in \mathcal{S}, \forall t \in \mathcal{T}_s \\ \psi_i(\tau+1) = \left[\psi_i(\tau) + \epsilon_4(\tau) \left(\sum_{\{u,tr(u)=i\}} \sum_{v=1}^{n} \widetilde{e_{u,v}} q_v(\tau) - \varrho_i(\tau) \mathcal{E}_i \right) \right]^+, & \forall i \in \mathcal{N} - \{t\} \\ \psi_t(\tau+1) = \psi_t(\tau) = 0, & \forall t \in \mathcal{T}_s \end{cases} \tag{23}$$

where $\epsilon_i(\tau), i = 1, 2, 3, 4,$ are the step sizes, and $[\cdot]^+$ is the projection operator defined as $\max(\cdot, 0)$.

4.6. High level update

As shown in above, $\boldsymbol{f}$ is obtained by the back-pressure algorithm introduced in Section 4.3 and explicitly updated through (21) in the following subsection, and the other variables, ϱ, x, q, g, κ, ψ, are resulted from the primal-dual algorithm just given in (23) under the assumption that $\mathbf{p}$ is fixed. Naturally, the question of how to adjust $\mathbf{p}$ arises. For this, we note that the objective function $\mu^*(\mathbf{p})$ is the optimal objective value of **LLP** for a given coupling vector of variables $\mathbf{p}$, where the corresponding primal and dual variables are $(\boldsymbol{\varrho}^*, \mathbf{x}^*, \mathbf{f}^*, \mathbf{q}^*, \mathbf{g}^*)$ and $(\boldsymbol{\kappa}^*, \boldsymbol{\psi}^*)$, respectively. In other words, $\mu^*(\mathbf{p}) = L(\boldsymbol{\varrho}^*, \boldsymbol{x}^*, \boldsymbol{f}^*, \boldsymbol{q}^*, \boldsymbol{g}^*, \boldsymbol{\kappa}^*, \boldsymbol{\psi}^*)$. To take into account the possibility of $\mu^*(\mathbf{p})$ may not be differentiable, we adopt a subgradient approach to generate a sequential of feasible points of $\mathbf{p}$ for the solution.

To find the solution, we note that while the coupling vector of variables $\mathbf{p}$ is so complexly coupled with the others among several constraints, **MPP** can be represented by using a similar formulation for **LLP**. That is, by regarding all the other variables as constants based on ACS, we can resolve the high level optimization **MPP** through the following subgradient method:

$$p_u(\eta+1) = \left[p_u(\eta) + \epsilon_5(\eta) \frac{dL(\boldsymbol{\varrho}, \boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{\kappa}, \boldsymbol{\psi})}{dp_u} \right]^{\mathcal{C}} \tag{24}$$

In above, η is the time index for **MMP** while τ is that for **LLP**, $\epsilon_5(\eta)$ is the step size, and $[\cdot]^{\mathcal{C}}$ denotes the projection onto a set $\mathcal{C}$ [28], and here

$$\mathcal{C} = \left\{ \mathbf{p} | \mathbf{p} \succeq 0 \text{ and } \sum_{u=1}^{m} p_u = 1 \right\} \tag{25}$$

represents the feasible solution set of this problem. Then, by solving $\frac{dL(\boldsymbol{\varrho}, \boldsymbol{x}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{q}, \boldsymbol{\kappa}, \boldsymbol{\psi})}{dp_u}$, we have $\sum_{v=1}^{n} (q_v r_{u,v} \omega_u - q_v e_{u,v} \psi_{tr(u)})$. Finally, we can update p_u in the $\eta + 1$th iteration more explicitly by

$$p_u(\eta+1) = \left[\left[p_u(\eta) + \epsilon_5(\eta) \left(\sum_{v=1}^{n} (q_v^*(\eta) r_{u,v} \omega_u(\eta) - q_v^*(\eta) e_{u,v} \psi_{tr(u)}(\eta)) \right) \right]^+ \right]^{\mathcal{C}} \tag{26}$$

where $r_{u,v}$ and $e_{u,v}$ are given as its inputs, and $\omega_u(\eta)$ and $\psi_{tr(u)}(\eta)$ for l_u are the solutions resulted from **LLP** and collected at the beginning of the ηth iteration of **MMP**.

4.7. Summary of the two-level distributed algorithm

To realize the proposed distributed algorithm, any node i is treated as an entity that can process, store, and communicate information. In particular, each hyperlink $l_u = (i, J)$ can be delegated to its sending node i or $tr(u)$ so that all computations with respect to the link can be charged on this node. One the one hand, every node is required to keep track of primal and dual variables, $\varrho_i(\tau), x_s(\tau), \kappa_i^{st}(\tau)$, and $\psi_i(\tau)$ for the low level. On the other hand, since the projection for $p_u(\eta)$ in (26) requires $q_v^*, \forall v$, to proceed, a *head node* may be needed to collect such parameters for the high level. Nevertheless, by resorting to the Markov approximation method in [17] along with the assumption that each node can overhear the others, q_v^* could be independently estimated for each hyperlink l_u through its sending node $tr(u)$ by using, e.g., the mini-slot implementation in [45] to reduce its processing time, while $\omega_u(\eta)$ and $\psi_{tr(u)}(\eta)$ are kept track by every node involved. For the communication overhead, it is worth noting that all the updates required at both low level and high level can be done by exchanging the parameters locally computed and stored in addition to the broadcasts required on each node. Further, if a float data structure is implemented, each primal or dual variable may take up only several bytes, and the communication overhead on these information exchanges is quite small when compared with the session data transmission. In fact, the exchanges need not be implemented as individual packets, and the overhead can be minimized as long as these variables or parameters can be conveyed through a filed in data or acknowledgement packets of the session flow. For reference, a possible implementation of the distributed algorithm is summarized in Algorithm 1.

Algorithm 1 Two-level distributed algorithm.

```
(Initialization:)
given a positive k_τ, set τ = 0, η = 0, and ϱ_i(0), x_s(0), κ_i^st(0), ψ_i(0), f_iJj^st(0), p_u(0), respectively,
  to some non-negative value for ∀i ∈ N, ∀s ∈ S, ∀t ∈ T_s, ∀u ∈ M̄.
for each η = 1, 2, ... do
  (Low-level update:)
  each node receives p_u(η) from each hyperlink u ∈ M̄, and it
  for each τ = 1, 2, ..., k_τ do
    • estimates q_v^*(τ) in (18) with the Markov approximation metho [17];
    • updates the primal and dual variables ϱ_i(τ+1), x_s(τ+1), κ_i^st(τ+1), ψ_i(τ+1) with (23),
      according to if it is a session source, a session destination, or a pure relay node;
    • calculates f_iJj^st(τ+1) with (21) for the next iteration;
  end for
  at the end of k_τ of η, each node updates and broadcasts ω_u(η) in (20) to all hyperlinks;
  (High-level update:)
  for each hyperlink l_u (represented by its sending node tr(u)) do
    • calculates p_u(η+1) with (24) and broadcasts the result to all nodes;
    • goes back to low-level update;
  end for
end for
```

5. Numerical results

In this section, we report on numerical results for our cross-layer distributed optimization. As shown in Fig. 1(a), a well-known wireless butterfly network with network coding in the literature [9,10,38] is adopted here as our simulation environment. However, unlike the example given in Section 2 which only aims to demonstrate a possible contention-free hyperlink scheduling given a set of transmission modes, in the numerical experiment, we instead consider six hyperlinks, (1,{2}), (1, {3}), (1, {2,3}), (2, {4,5}), (3, {4,6}), and (4,{5,6}) and five transmission modes {(1,{2}),(3, {4,6})}, {(1,{2}),(4,{5,6})}, {(1, {3}),(2, {4,5})}, {(1, {3}),(4, {5,6})}, and {(1, {2,3}),(4,{5,6})}, that comply with those given in [10]. Specifically, to focus on the optimization framework itself, in the numerical experiment we do not consider a particular physical layer and its energy consumption. Instead, to accommodate very different possible scenarios, we assume a normalized situation wherein each hyperlink l_u has one unit capacity and each node i has initial energy $\mathcal{E}_i = 1$ while each hyperlink has random energy consumption $e_{u,v} \in (0, 1)$ when it is scheduled by p_u and q_v to preserve the randomization suggested in the framework. In addition, we use the variable step sizes that satisfy $\epsilon_i(t) > 0$, $\sum_{t=1}^{\infty} \epsilon_i(t) = \infty$, and $\sum_{t=1}^{\infty} \epsilon_i^2(t) < \infty, t \in \{\tau, \eta\}, i \in \{1, 2, 3, 4, 5\}, \forall \tau, \eta \in \mathbb{N}$, which are usually considered for a subgradient distributed algorithm to ensure convergence. Specifically, we adopt in this experiment $\epsilon_i(t) = \frac{1}{t+c}, c > 0$. Given that, a multicast session $s = 1$ is conducted with source node 1 transmitting packets to its sink nodes 5 and 6, as the traffic demand from transport layer.

5.1. Number of low level iteration

We start in this subsection to examine how the number of low level iteration can impact the distributed optimization on behalf of the theoretical validity and numerical feasibility of our programming models. As shown in Sections 4.5 and 4.6, we have the flexibility to fix the coupling variable $\mathbf{p}$ for the primal-dual stochastic algorithm in **LLP** to be converged in a reasonable period of time, which then feedbacks the other variables to determine $\mathbf{p}$ in the next iteration. Thus, the number of iteration for the primal-dual stochastic algorithm should be determined to alleviate the NP problem involved within a reasonable time budget. To this end, we fix $\lambda = (\lambda_1 = 0.8, \lambda_2 = 0.2)$, let the step size $\epsilon_i(t)$ be $\frac{1}{t+c}, c = 150$, and vary the number of low level iteration among 5, 20, 100, 500, and 1000, to see its impacts upon the distributed optimization. In particular, to confirm that the two major metrics, i.e., session throughput and network lifetime obtained through (23) of the primal-dual stochastic algorithm, can properly converge, we also measure the actual values of these two metrics in the simulation study, and conduct a

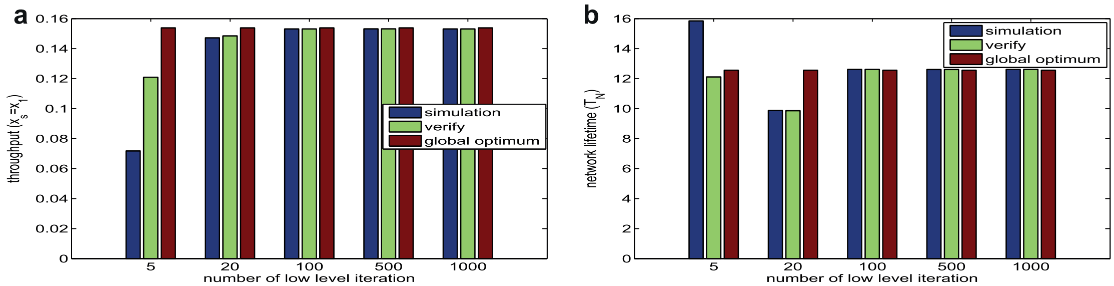

Fig. 2. System performances under different numbers of low level iteration: (a) session throughput x_s(or x_1), and (b) network lifetime T_N.

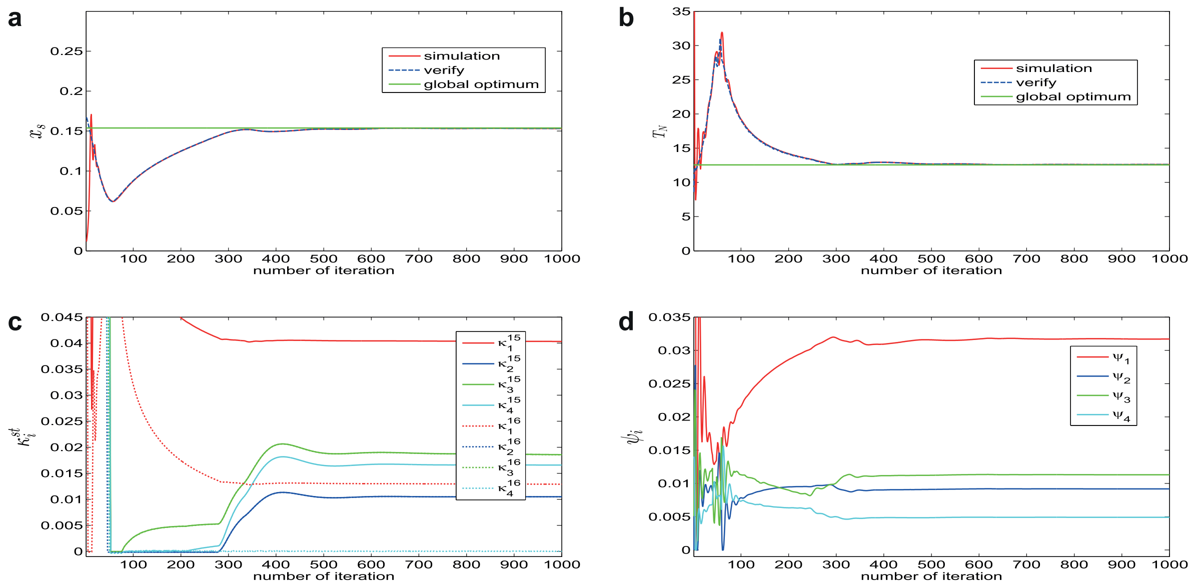

Fig. 3. Convergence behavior of optimization variables within 1000 iterations: (a) session data rate or throughput x_s, (b) network lifetime T_N (c) dual variables or Lagrange multipliers κ_i^{st}, and (d) dual variables or Lagrange multipliers ψ_i.

centralized programming tool of MATLAB to obtain the optimal results as the comparison basis for theoretical verification.

5.2. Convergence behavior of primal and dual variables

The results are now summarized in Fig. 2, in which *simulation* denotes the results of our distributed optimization, *verify* denotes the measurements on the two metrics in the simulation study, and *global optimum* denotes the centralized optimization results obtained by the MATLAB centralized programming tool. More specifically, the performance metrics, x_s and T_N, labeled by *simulation* represent the results obtained at the end of iterations (23) in the algorithm under simulation, while those labeled by *verify* are obtained by directly computing these metrics with the data rates and energy consumptions measured at the end of these iterations in the simulation. As readily shown in this figure, five low level iterations are not enough for the computed values converging to the corresponding measurements while 20 iterations may converge better but still cannot approach the optimal results. On the other hand, using more than or equal to 100 iterations can lead to the converged values closely approaching the global optimum, and thus, we adopt 100 low level iterations as our baseline for the following and a number more than 100 when converging to the corresponding global optima is not easily achieved in certain cases of the experiment. Next, we adopt the parameter examined in above that $k_\tau = 100$, i.e., each high level iteration comprises 100 low level iterations, and fix $\lambda_1 = 0.8, \lambda_2 = 0.2$, and $\epsilon_i(t) = \frac{1}{t+c}$ with $c = 150$ as before to exhibit a typical dynamic of updating primal and dual variables in the scenario. As shown in Fig. 3(a) and (b), the primal variables x_s and $T_N = \max\{T_i = 1/\varrho_i, \forall i \in \mathcal{N}\}$ obtained through the distributed optimization well converge to the global optima around 500 iterations even though we terminate the experiment at 1000 iterations. In addition, the results from our method (denoted by *simulation*) are shown to fast converge to the measured counterparts (denoted by *verify*), within the first several ten iterations, in this case. Similarly, in Fig. 3(c) and (d), it can be seen that the dual variables κ_i^{st} and ψ_i are also gradually stable, and the stability can be more easily observed after 500 iterations as mentioned for the above.

5.3. Impact of weighting factor

Finally, it may be reminded that the objective function in question is obviously a weighted sum of multiple types

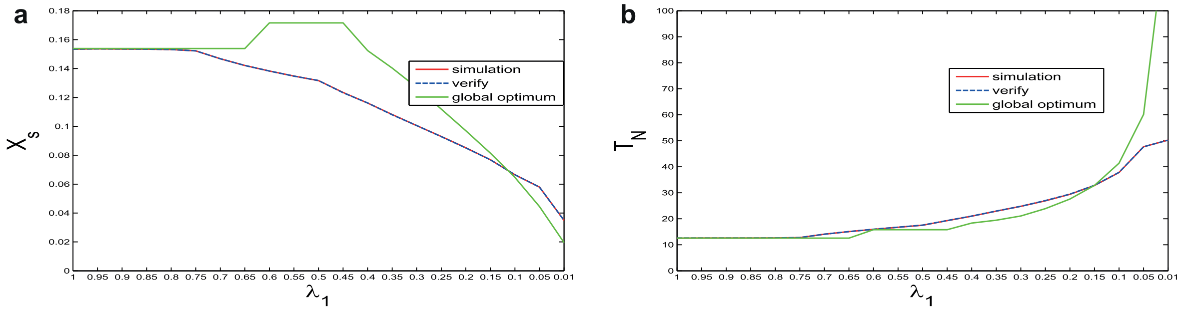

Fig. 4. Performance comparison for different trade-off weights λ: (a) session data rate x_s, and (b) network lifetime T_N.

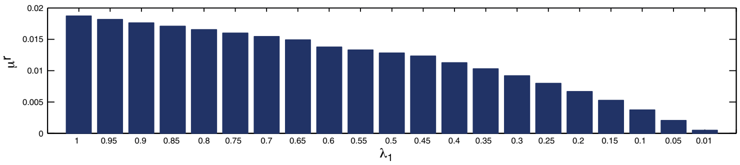

Fig. 5. Utility on different trade-off weights λ.

of utilities, and in this case throughput and lifetime are the two heterogeneous utilities motivating our study, and serve as an example to exhibit the possibility of multi-objective optimization to be implemented in a distributed manner. In the current model, the trade-off of these two metrics in is particularly reflected on the trade-off weight $\lambda = (\lambda_1, \lambda_2)$. For simplicity, we use the component λ_1 decreased from 1 to 0.05 with a decrement step size of 0.05 and finally reaching 0.01 to represent the overall weight λ since $\lambda_1 + \lambda_2 = 1$. In addition, for examining the effects of this weight on the trade-off, we let $\eta = 1000$, $k_\tau = 100$, and $\epsilon_i(t) = \frac{1}{t+c}$ with c = 150 as before for $1 \geq \lambda_1 \geq 0.1$ while $\eta = 5000$, $k_\tau = 1000$, and $\epsilon_i(t) = \frac{1}{t+c}$ with c = 2000 for $\lambda_1 \in \{0.05, 0.01\}$. The latter is so done because small value of λ_1 will cause the difficulty of converging to the theoretical optimal of lifetime, and increasing the iteration number and decreasing the step size can help alleviating the numerical difficulty.

The results are now summarized in Fig. 4. As shown there, by properly adjusting the weight λ along with the other parameters properly tuned for the shown range, the distributed optimization exhibits to have a smooth trade-off effect on the two performance metrics. Specifically, as shown in this figure, the data rate decreases and the lifetime increases with respect to λ_1 decreased gradually, with a final jump in T_N at $\lambda_1 = 0.01$ where almost the whole weight is dedicated to the lifetime, as expected. However, the trend on the results of centralized optimization may be non-monotonic and would be further noted. To see this easily with positive utility, we represent JLUP by negating the signs in the objective of maximization to be an equivalent minimization problem as $\min\{\mu^r(\lambda) = \mu_s(\lambda) + \mu_l(\lambda)\}$ with $\mu_s(\lambda) \triangleq -\lambda_1 \frac{\sum_{s\in\mathcal{S}} \log(x_s)}{W_1}$ and $\mu_l(\lambda) \triangleq \lambda_2 \frac{\sum_{i\in\mathcal{N}} \varrho_i^2}{W_2}$. As shown in Fig. 5, the utility μ^r at $\lambda_1 = 1$ obtained by the centralized optimization gives $\mu_s(=0.018718) + \mu_l(=0) = 0.018718$, which is actually higher than any other, e.g., that at $\lambda_1 = 0.6$ giving $\mu_s(=0.010576) + \mu_l(=0.003197) = 0.013773$, and monotonically decreases as λ_1 decreases. In general, this decrement leads to decreasing x_s and increasing T_N, as expected. However, the monotonicity is mainly reflected on the utility rather than the metrics. This trend is due to the utility μ^r rather than a raw weighted sum, e.g., $\lambda_1 x_s + \lambda_2 T_N$, to be optimized on the metrics directly. Such a raw weighted sum cannot be considered here as the network lifetime $T_N \triangleq \min_{i\in\mathcal{N}} T_i$ involved is already a minimization problem to be resolved at the same time. Thus, we consider a well-known method, $\min \sum_{i\in\mathcal{N}} \varrho_i^2$, to approximate this lifetime. Given that and general $[r_{u,v}]$ and $[e_{u,v}]$, our experiment reveals the possibility that a higher x_s, T_N, or both could be obtained by the optimization tool with a lower λ_1 and still yields a decreasing utility. That is, the results are optimal with respect to the trade-off utility and monotonically decrease as the weight decreases, but without the assumption that each node has the same condition on data, energy, or both, a monotonic variation on the performance metrics is not expected in the utility optimization.

Given that, the results from our optimization, denoted by *simulation*, perfectly match the actual measurements, denoted by *verify*, and closely approach the theoretical optimal obtained by the centralized optimization tool, denoted by *global optimum*. That is, the updates of optimization variables can reflect the actual values in time and the distributed optimization can approximate the optimal results properly. Nevertheless, at λ_1 approaching 0 where the whole weight is almost dedicated to the lifetime, the centralized optimization tool will report a nearly empty throughput and a nearly infinity lifetime while our distributed optimization may reflect the situation by increasing the iteration number and decreasing the step size. However, it could be noted that giving much more weight on lifetime and thus seriously sacrificing throughput is not a reasonable strategy in any network environment. Thus, given the reasonable range from 1 to 0.1, it can be shown that the degree of our results approaching the global optima, represented by $1 - \frac{|\text{simulation}-\text{global optimum}|}{\text{global optimum}}$, in Fig. 4 would be 99.78% and 99.76% at most, 76.18% and 82.32% at least, and 89.21% and 92.74% on average, for the data rate (x_s) and the network lifetime (T_N), respectively. In addition, by referring back to Fig. 3, we can see that x_s and T_N approach or converge fast to their optima (around 300 iterations) even though we show 1000 iterations for plotting. Moreover, as one of the merits resulted from an iterative algorithm like ours, the network is not necessary to wait for the algorithm to converge; instead, the results in each iteration could be applied at the moment it is obtained to couple with the network dynamic. These features are notable because a nonconvex QPQC problem is known to be NP-hard as mentioned before, and our problem is not only a QPQC problem in a general sense but also involves a MWIS subproblem which is already NP and difficult to be approximated even in a centralized way [44]. The results for such a hard problem actually reveal that our distributed optimization can perform well enough within the limited time constraint, especially in the environment of wireless multihop networks. In such an environment, nodes may be mobile and channel are usually error-prone and time-varying, and our method prevents obtaining an optimal solution with a global optimization tool that may be impractical for it is usually time-consumed and out-of-date soon.

6. Conclusion

In this work, we have introduced a mathematical programming model for the cross-layer multi-objective optimization problem of maximizing network lifetime and optimizing aggregate system utility with intra-flow network coding, solved in a distributed manner. In particular, by resorting to alternate convex search (ACS) and primal decomposition (PD), we have resolved the joint optimization modeled as a quadratically constrained quadratic programming (QCQP) problem that is NP-hard in general, wherein its scheduling subproblem on both hyperlink and transmission mode is a maximum weighted independent set (MWIS) problem and is NP-hard already. The resulted distributed optimization has been shown to be able to approximate the optimal results with a biconvex programming model, and the subgradient-based algorithms developed can iteratively approach the optimal solution of this complex problem in the spirit of decentralization. Our numerical results have readily exhibited the correctness of this programming model, and the fact that the proposed method can fulfill the desired requirement with a good trade-off between the heterogeneous objectives and have well computational efficiency.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, R.W. Yeung, Network information flow, IEEE Trans. Inf. Theory 46 (2000) 1204–1216.
[2] S.-Y. R. Li, R.W. Yeung, N. Cai, Linear network coding, IEEE Trans. Inf. Theory 49 (2003) 371–381.
[3] T. Ho, M. Medard, J. Shi, M. Effros, D.R. Karger. On randomized network coding, in: Proceedings of 41st Allerton Annual Conference Communication, October 2003.
[4] S. Chachulski, M. Cagalj, S. Bidokhti, J. Jubaux, Trading structure for randomness in wireless opportunistic routing, Proceedings of ACM SIGCOMM (2007) 169–180.
[5] J. Park, M. Gerla, D. Lun, Y. Yi, M. Medard, CodeCast: a network-coding-based ad hoc multicast protocol, IEEE Wireless Commun. 13 (5) (2006) 76–81.
[6] K. Jain, J. Padhye, V.N. Padmanabhan, L. Qiu, Impact of interference on multi-hop wireless network performance, Proceedings of MobiCom (2003) 66–80.
[7] A. Ephremides, T.V. Truong, Scheduling broadcasts in multihop radio networks, IEEE Trans. Commun. 38 (4) (1990) 456–460.
[8] E. Arikan, Some complexity results about packet networks, IEEE Trans. Inf. Theory 30 (4) (1984) 681–685.
[9] T. Ho, D.S. Lun, Network Coding: An Introduction, Cambridge University Press, 2008.
[10] E. Karami, S. Glisic, Joint optimization of scheduling and routing in multicast wireless ad hoc networks using soft graph coloring and nonlinear cubic games, IEEE Trans. Vehic. Technol. 60 (7) (2011) 3350–3359.
[11] R.L. Cruz, A.V. Santhanam, Optimal routing, link scheduling and power control in multihop wireless networks, Proceedings of IEEE INFOCOM 2003 1 (2003) 702–711.
[12] R. Madan, S. Cui, S. Lal, A. Goldsmith, Cross-layer design for lifetime maximization in interference-limited wireless sensor networks, IEEE Trans. Wireless Commun. 5 (11) (2006) 3142–3152.
[13] H. Nama, M. Chiang, N. Mandayam, Utility-lifetime trade-off in self-regulating wireless sensor networks: a cross-layer design approach, Proceedings of IEEE International Conference on Communications 8 (2006) 3511–3561.
[14] L.D.P. Mendes, J.J.P.C. Rodrigues, Review: a survey on cross-layer solutions for wireless sensor networks, J. Netw. Comput. Appl. 34 (2) (2011) 523–534.
[15] L. Tassiulas, A. Ephremides, Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks, IEEE Trans. Autom. Control 37 (12) (1992) 1936–1949.
[16] X. Lin, N.B. Shroff, R. Srikant, A tutorial on cross-layer optimization in wireless networks, IEEE J. Selected Areas Commun. 24 (8) (2006) 1452–1463.
[17] M. Chen, S. Liew, Z. Shao, C. Kai, Markov approximation for combinatorial network optimization, Proceedings of IEEE INFOCOM 2010 (2010) 1–9.
[18] X. Zhang, B. Li, Optimized multipath network coding in lossy wireless networks, IEEE J. Selected Areas Commun. 27 (5) (2009) 622–634.
[19] B. Radunovic, C. Gkantsidis, P. Key, P. Rodriguez. An optimization framework for opportunistic multipath routing in wireless mesh networks, in: Proceedings of IEEE INFOCOM 2008, April 2008.
[20] S. Bhadra, S. Shakkottai, P. Gupta, Min-cost selfish multicast with network coding, IEEE Trans. Inf. Theory 52 (2006) 5077–5087.
[21] L. Chen, T. Ho, S.T. Low, M. Chiang, J.C. Doyle, Optimization based rate control for multicast with network coding, Proceedings of IEEE INFOCOM 2007 (2007) 1163–1171.
[22] J. Price, T. Javidi, Network coding games with unicast flows, IEEE J. Selected Areas Commun. 26 (7) (2008) 1302–1316.
[23] J. Liu, C.-H. R. Lin, Cross-layer optimization for performance trade-off in network code-based wireless multi-hop networks, Comput. Commun. 52 (1) (2014) 89–101.
[24] J. Liu, Joint lifetime-utility cross-layer optimization for network coding-based wireless multi-hop networks with matrix game and multiple payoffs, IEICE Trans. Commun. E97-B (8) (2014) 1638–1646.
[25] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[26] J. Gorski, F. Pfeuffer, K. Klamroth, Biconvex sets and optimization with biconvex functions: a survey and extensions, Math. Methods Oper. Res. 66 (3) (2007) 373–407.
[27] D.P. Palomar, M. Chiang, A tutorial on decomposition methods for network utility maximization, IEEE J. Selected Areas Commun. 24 (8) (2006) 1439–1451.
[28] D.P. Palomar, Convex primal decomposition for multicarrier linear mimo transceivers, IEEE Trans. Signal Process. 53 (12) (2005) 4661–4674.
[29] C. Floudas, V. Visweswaran, A global optimization algorithm (GOP) for certain classes of nonconvex NLPS: I. Theory, Compute. Chem. Eng. 14 (12) (1990) 1397–1417.
[30] P. Paatero, U. Tapper, Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values, Environmetics 5 (1994) 111–126.
[31] L.E. Ghaoui, V. Balakrishnan, Synthesis of fixed-structure controllers via numerical optimization, Proceedings of the 33rd IEEE Conference on Decision and Control 3 (1994) 2678–2683.
[32] J. Zou, H. Xiong, L. Song, Z. He, T. Chen, Prioritized flow optimization with generalized routing for scalable multirate multicasting, IEEE International Conference on Communications (2009) 1–6.
[33] H. Nama, M. Chiang, N. Mandayam, Utility-lifetime trade-off in self-regulating wireless sensor networks, Proceedings of IEEE ICC (2006) 3511–3516.
[34] W. Liu, K. Xu, P. Zhou, Y. Ding, W. Cheng, A joint utility-lifetime optimization algorithm for cooperative mimo sensor networks, Proceedings of IEEE WCNC (2008) 1067–1072.
[35] Y.E. Sagduyu, Medium Access Control and Network Coding for Wireless Information Flows, Ph.D. thesis, University of Maryland, College Park, 2007.

[36] J. Tang, G. Xue, W. Zhang, Cross-layer design for end-to-end throughput and fairness enhancement in multi-channel wireless mesh networks, IEEE Trans. Wireless Commun. 6 (10) (2007) 3482–3486.
[37] J. Tang, R. Hincapie, G. Xue, W. Zhang, R. Bustamante, Fair bandwidth allocation in wireless mesh networks with cognitive radios, IEEE Trans. Vehic. Technol. 59 (3) (2010) 1487–1496.
[38] Y.E. Sagduyu, A. Ephremides, On joint MAC and network coding in wireless ad hoc networks, IEEE Trans. Inf. Theory 53 (10) (2007) 3697–3713.
[39] G.P. Liu, J.B. Yang, J.F. Whidborne, Multiobjective Optimisation and Control, Research Studies Press, 2013.
[40] J. de Leeuw, Block relaxation algorithms in statistics, Information Systems and Data Analysis (1994) 308–325.
[41] M. Lobo, L. Vandenberghe, S. Boyd, H. Lebret, Applications of second-order cone programming, Linear Algebra Its Appl. 284 (1998) 193–228.
[42] J. Benders, Partitioning procedures for solving mixed-variables programming problems, Numer. Math. 4 (1962) 238–252.
[43] A. Geoffrion, Generalized benders decomposition, J. Optim. Theory Appl. 10 (4) (1972) 237–260.
[44] V. Vazirani, Approximation Algorithms, Springer, 2001.
[45] J. Ni, B. Tan, R. Srikant, Q-CSMA: queue-length-based CSMA/CA algorithms for achieving maximum throughput and low delay in wireless networks, IEEE/ACM Trans. Netw. 20 (3) (2012) 825–836.

Jain-Shing Liu was born in Taipei, Taiwan, in 1970. He received the Ph.D. degree in the Department of Computer and Information Science, National Chiao-Tung University, Hsinchu, Taiwan. He currently is a professor of the Department of Computer Science and Information Engineering, Providence University, Taichung Taiwan, ROC. His research interests include designs and performance analyses of wireless communication protocols, wireless local area networks, wireless sensor networks, and personal communication networks. He is a member of IEEE and IEICE over a decade, and in the duration, he received the best paper award in the 10th Mobile Computing Workshop 2004, and was included in Marquis Who's Who in Science and Engineering, 10th Anniversary Edition. So far, he was the first author of around 50 journal and conference papers. In addition, he had served as a reviewer for many international journals, and also served as a (Technical) Program Committee member for many international conferences such as IEEE International Workshop on Wireless Network Algorithms (WiNA 2008), Fifth Annual International ICST Wireless Internet Conference (WiCON 2010) Networking Track, International Congress on Computer Applications and Computational Science (CACS 2010), and National Computer Symposium (NCS 2011), International Conference on Computer and Communication Devices (ICCCD 2014), as most recently served as the conference program chair of International Conference on Computer and Communication Devices (ICCCD 2015).

Jichiang Tsai received his B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan in 1991. Then he started his graduate study at the same university, and received the Ph.D. degree in electrical engineering in 1999. He served as a postdoctoral research fellow in the Institute of Information Science, Academia Sinica, Taipei, Taiwan from 1999 to 2001. In 2002, he joined the Department of Electrical Engineering, National Chung Hsing University, Taichung, Taiwan, as an assistant professor, and then was promoted to associate professor in 2005. His current research interests include parallel and distributed systems, dependable computing, embedded systems, mobile computing and computer networks.