Distributed Utility Maximization for Network Coding Based Multicasting: A Critical Cut Approach

Yunnan Wu*, Mung Chiang†, and Sun-Yuan Kung†
*Microsoft Research, One Microsoft Way, Redmond, WA, 98052. yunnanwu@microsoft.com
†Dept. of Electrical Engineering, Princeton University, Princeton, NJ, 08544. {chiangm,kung}@princeton.edu

***Abstract*— A central issue in practically deploying network coding in a shared network is the adaptive and efficient allocation of network resources. This issue can be formulated as an optimization problem of maximizing the *net-utility* – the difference between a utility derived from the attainable multicast throughput and the total cost of resource provisioning.**

We develop a primal-subgradient type distributed algorithm to solve this utility maximization problem. The effectiveness of the algorithm hinges upon two key properties we discovered: (1) the set of subgradients of the multicast capacity is the convex hull of the indicator vectors for the *critical cuts*, and (2) the complexity of finding such critical cuts can be reduced by exploiting the algebraic properties of linear network coding. The extension to multiple multicast sessions is also carried out. The effectiveness of the proposed algorithm is confirmed by simulations on an Internet Service Provider's topology.

I. Introduction and Problem Formulation

We consider multicasting information from a source node s to a set of destination nodes T in a network of lossless links with bit-rate constraints, which is represented by a directed graph $G = (V, E)$ with edge capacity vector $\boldsymbol{c}$ of length-$|E|$. The capacity of link $vw \in E$ is denoted by c_{vw}. Given V, E, $\boldsymbol{c}$, s, and T, the *multicast capacity* refers to the maximum multicast throughput; this is denoted by $C_{s,T}(\boldsymbol{c})$. In [1], Ahlswede *et al.* established that the multicast capacity is equal to the minimum capacity of a cut separating s from a destination $t \in T$; i.e.,

$$C_{s,T}(\boldsymbol{c}) = \min_{t \in T} \rho_{s,t}(\boldsymbol{c}), \tag{1}$$

where $\rho_{s,t}(\boldsymbol{c})$ is the minimum s–t cut capacity in $(V, E, \boldsymbol{c})$.

An essential element needed in a practical network coding system is a distributed scheme for "properly" allocating bit-rate resources at each link for each multicast session in a shared network. From a system perspective, there are two competing considerations. On the one hand, it is desirable to maximize the utilities of end users derived from the supported end-to-end multicast throughput. On the other hand, there is incentive to economize the consumption of network resources. Following an economics approach to network design, we cast this problem as the maximization of a *net-utility* function

$$U_{\text{net}}(r, \boldsymbol{g}) \triangleq U(r) - \sum_{vw \in E} p_{vw}(g_{vw}). \tag{2}$$

Here $U(r)$ represents the raw utility when an end-to-end throughput r is provided. The cost function p_{vw} associated with link vw maps the consumed bit-rate g_{vw} to the charge. As a standard assumption, p_{vw} are nondecreasing and convex functions, and U is a nondecreasing and concave function.

The critical constraint of this maximization is that throughput r must be attainable using the resources g_{vw}. Let $\boldsymbol{g}$ be a length-$|E|$ vector collectively representing g_{vw}. For network coding based multicasting, this relation is characterized by

$$r \le C_{s,T}(\boldsymbol{g}) = \min_{t \in T} \rho_{s,t}(\boldsymbol{g}). \tag{3}$$

Since by assumption $U(r)$ is nondecreasing, given $\boldsymbol{g}$, we can always set r to be the maximum achievable throughput $C_{s,T}(\boldsymbol{g})$. With this observation, we can turn the problem into a maximization over $\boldsymbol{g}$ only:[1]

$$\begin{aligned} U^*_{\text{net}} \triangleq \max \quad & U\left(C_{s,T}(\boldsymbol{g})\right) - \sum_{vw \in E} p_{vw}(g_{vw}) \\ \text{subject to:} \quad & \mathbf{0} \le \boldsymbol{g} \le \boldsymbol{c}, \end{aligned} \tag{4}$$

Our objective is to design efficient distributed algorithms for finding an optimal solution $\boldsymbol{g}^*$ of (4), which will be used as the allocated bit-rate resources at each link. The distributed algorithms should, hopefully, incur low extra communication overhead and be adaptive to network dynamics.

A. Proposed approach

We will show that the objective function of (4) is concave. Our formulation of the problem, (4), motivates us to apply an iterative algorithm that starts with an initial assignment $\boldsymbol{g}$ and incrementally updates it along certain directions. However, one of the difficulties is that the objective function is non-differentiable, due to the non-differentiability of the function $C_{s,T}(\boldsymbol{g})$. To cope with this issue, we resort to *subgradient methods*. A subgradient is a generalization of the gradient to non-differentiable functions; for a concave function f, each subgradient at $\boldsymbol{x}$ corresponds to a linear over-estimator of f that touches f at $\boldsymbol{x}$.

The subgradient method has been applied to various formulations of network utility maximization, e.g., in [2]–[7]; a survey on network utility maximization can be found in [8]. Lun *et al.* [6] proposed a subgradient-based distributed

[1] In this paper, $\boldsymbol{a} \le \boldsymbol{b}$ is in the element-wise sense.

algorithm for minimum cost multicasting using network coding. For the utility maximization problem (4), Wu and Kung [7] presented a subgradient-based distributed algorithm, which repeatedly computes shortest paths from the source to each destination. However, in all these studies, the subgradient method is applied to the Lagrangian dual problem. In contrast, our approach in this paper represents a *primal* subgradient method. The unique challenges here lie in deriving an analytic expression of a subgradient and developing a low-complexity implementation. As shown in Sections II and III, respectively, both challenges can be met by using properties of network coding and following a primal, rather than dual, approach. The resulting distributed algorithm is presented in Section IV. The proposed primal approach is more direct than the dual approaches [6], [7]; it also extends more easily to multiple multicast sessions. In addition, the primal approach can still be used to compute a locally optimum resource allocation even when utility functions become sigmoidal and cost functions nonconvex under certain application modeling needs, whereas the dual approach may produce infeasible, oscillatory solutions. The main challenge for primal approach is that the key step of finding a critical cut (as will be shown in Subsection II-B) is more complex than the key step of finding the shortest path in dual approach [7]. Efficient complexity reduction of this step will be shown in Section III.

When particularizing the subgradient methods to (4), we need to find a subgradient for the concave function $C_{s,T}(\boldsymbol{g})$. In Sections II-B, we provide a key characterization of the set of subgradients of $C_{s,T}(\boldsymbol{g})$, proving that the set of subgradients of $C_{s,T}(\boldsymbol{g})$ is the set of convex combinations of the indicator vectors for the *s–T critical cuts*. A cut $(U^*, \overline{U^*})$ is said to be s–T critical if it is a minimum s–t^* cut for a *critical destination* t^*; a destination t^* is said to be critical if $\rho_{s,t^*}(\boldsymbol{g}) = \min_{t\in T} \rho_{s,t}(\boldsymbol{g})$.

Thus we can implement the subgradient iterations by seeking an s–T critical cut in each iteration. Finding a minimum s–t cut is a classical combinatorial optimization problem that has been well understood. In particular, the preflow-push algorithm [9] for finding a minimum s–t cut is suitable for distributed implementation. We can certainly find an s–T critical cut by finding a minimum s–t cut for each destination $t \in T$. However, as shown in Section III-B, the algebraic properties of linear network coding are helpful in reducing that complexity. Previous studies have shown the multicast capacity in an acyclic graph with unit edge capacities can be achieved with high probability by performing random linear coding over a sufficiently large finite field. Accordingly, if random mixing is done in a linear space with dimension h_0 slightly less than the capacity $C_{s,T}$, then the destinations can recover the data with high probability. We observe that if mixing is done in a linear space with dimension h slightly higher than the capacity $C_{s,T}$, then the critical destinations will have lower ranks than the noncritical destinations. To ensure that the destinations can still recover the data, we can linearly precode the original data, e.g., by letting some original data be zero. In essence, by performing random mixing in a space with dimention $h > C_{s,T}$ while using a signal space with dimension $h_0 < C_{s,T}$, the critical destinations are identified and the data recovery is feasible. As a result, we can find an s–T critical cut by first finding a critical destination t^* and then finding a minimum s–t^* cut.

II. Optimization via Subgradient Iterations

A. Review: Preliminaries on Subgradient Methods

We first briefly review some preliminaries on subgradient methods before characterizing the set of all subgradients of $C_{s,T}(\boldsymbol{g})$ in the next subsection.

Definition 1 (Subgradient, subdifferential): Given a convex function f, a vector $\boldsymbol{\xi}$ is said to be a *subgradient* of f at $\boldsymbol{x} \in \text{dom } f$ if

$$f(\boldsymbol{x}') \geq f(\boldsymbol{x}) + \boldsymbol{\xi}^T(\boldsymbol{x}' - \boldsymbol{x}), \quad \forall \boldsymbol{x}' \in \text{dom } f. \tag{5}$$

For a concave function f, a vector $\boldsymbol{\xi}$ is said to be a subgradient of f at $\boldsymbol{x}$ if $-\boldsymbol{\xi}$ is a subgradient of $-f$.

The set of all subgradients of f at $\boldsymbol{x}$ is called the *subdifferential* of f at $\boldsymbol{x}$ and denoted by $\partial f(\boldsymbol{x})$.

Lemma 1 (Subgradient calculus): For a concave function $f: \mathbb{R}^+ \mapsto \mathbb{R}$, the following properties hold.

- f is differentiable at $\boldsymbol{x}$ if and only if $\partial f(\boldsymbol{x}) = \{\nabla f(\boldsymbol{x})\}$.
- $\partial(\alpha f) = \alpha \partial f$, if $\alpha > 0$.
- $\partial(f_1 + f_2) = \{\boldsymbol{\xi}_1 + \boldsymbol{\xi}_2 | \boldsymbol{\xi}_1 \in \partial f_1,\ \boldsymbol{\xi}_2 \in \partial f_2\}$
- **pointwise minimum**: If $f = \min_{i=1,\ldots,m} f_i$ where f_i, $i = 1, \ldots, m$ are concave functions, then

$$\partial f(\boldsymbol{x}) = \text{conv}\left\{\partial f_i(\boldsymbol{x}) | i \in I(\boldsymbol{x})\right\}, \tag{6}$$

where $I(\boldsymbol{x}) \triangleq \{i | f_i(\boldsymbol{x}) = f(\boldsymbol{x}\}$ and conv$\{\}$ is the convex hull of the argument. In other words, the subdifferential of f at $\boldsymbol{x}$ is the convex hull of the subdifferentials of the "active" functions at $\boldsymbol{x}$.

In particular, if $f_i(\boldsymbol{x}) = \boldsymbol{a}_i^T \boldsymbol{x} + \boldsymbol{b}_i, i = 1, \ldots, m$, then

$$\partial f(\boldsymbol{x}) = \text{conv}\left\{\boldsymbol{a}_i | f_i(\boldsymbol{x}) = f(\boldsymbol{x})\right\}. \tag{7}$$

■

The subgradient method [10] maximizes a non-differentiable concave function in a way similar to gradient methods for differentiable functions – in each step, the variables are updated in the direction of a subgradient. However, such a direction may not be an ascent direction; instead, the subgradient method relies on a different property. If the variable takes a sufficiently small step along the direction of a subgradient, then the new point is closer to the set of optimal solutions.

Consider a generic, constrained concave maximization problem

$$\begin{aligned} \text{maximize} \quad & f(\boldsymbol{x}) \\ \text{subject to:} \quad & \boldsymbol{x} \in \mathcal{C}, \end{aligned} \tag{8}$$

where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is concave, and $\mathcal{C}$ is a closed and nonempty convex set. The subgradient method uses the iteration

$$\boldsymbol{x}^{(k+1)} = P\left[\boldsymbol{x}^{(k)} + \alpha_k \boldsymbol{\xi}^{(k)}\right], \tag{9}$$

where $\boldsymbol{x}^{(k)}$ is the k-th iterate, $\boldsymbol{\xi}^{(k)}$ is any subgradient of f at $\boldsymbol{x}^{(k)}$, $\alpha_k > 0$ is the k-th step size, and P is the projection on $\mathcal{C}$:

$$P[\boldsymbol{x}] \triangleq \arg\min_{\boldsymbol{x}' \in \mathcal{C}} \|\boldsymbol{x}' - \boldsymbol{x}\|^2. \tag{10}$$

Lemma 2 (Convergence of subgradient methods [11]): Assume $\boldsymbol{x}^*$ is a maximizer of (8) and there exists a G such that $\|\boldsymbol{\xi}^k\| \leq G$, $\forall k$. Then

$$f^* - \max_{i=1,\ldots,k} f(\boldsymbol{x}^{(i)}) \leq \frac{\|\boldsymbol{x}^{(1)} - \boldsymbol{x}^*\| + G^2 \sum_{i=1}^k \alpha_i^2}{2\sum_{i=1}^k \alpha_i}. \tag{11}$$

In particular,

- if a constant step size is used, i.e., $\alpha_k = h$, then the right hand side of (11) converges to $G^2 h/2$ as $k \to \infty$.
- if the step sizes satisfy

$$\lim_{k\to\infty} \alpha_k = 0, \quad \sum_{k=1}^{\infty} \alpha_k = \infty, \tag{12}$$

then right hand side of (11) converges to 0 as $k \to \infty$. Step sizes that satisfy this condition are called *diminishing step sizes*.

B. Subgradients of $C_{s,T}(\boldsymbol{g})$

Let a length-$|E|$ binary vector $\boldsymbol{I}_X$ be the *indicator vector* for edge set $X \subseteq E$; its e-th entry is 1 if $e \in X$, and 0 if $e \notin X$.

For $t \in T$, an s–t cut $(U, \overline{U})$ refers to a partition of the nodes $V = U + \overline{U}$ with $s \in U$, $t \in \overline{U}$. Let $\delta(U)$ denote the set of edges going from U to $\overline{U}$. The *capacity* of the cut refers to the sum capacity of $\delta(U)$. An s–t cut with minimum capacity is called a *minimum s–t cut*. The minimum s–t cut capacity is:

$$\rho_{s,t}(\boldsymbol{g}) \triangleq \min_{U:\ s\in U,\ t\in\overline{U}} \sum_{vw\in\delta(U)} g_{vw} = \min_{U:\ s\in U,\ t\in\overline{U}} \boldsymbol{I}_{\delta(U)}^T \boldsymbol{g}. \tag{13}$$

Definition 2 (Critical destination, Critical cut): A destination $t^* \in T$ is said to be a *critical destination* if $\rho_{s,t^*}(\boldsymbol{g}) = \min_{t\in T} \rho_{s,t}(\boldsymbol{g})$. The set of critical destinations is

$$T^*(\boldsymbol{g}) \triangleq \left\{ t^* \in T \,\middle|\, \rho_{s,t^*}(\boldsymbol{g}) = \min_{t\in T} \rho_{s,t}(\boldsymbol{g}) \right\}. \tag{14}$$

A cut $(U^*, \overline{U^*})$ is said to be an s–T *critical cut* if it is a minimum s–t^* cut for some critical destination $t^* \in T^*(\boldsymbol{g})$.

The name "s–T critical" comes from the observation that reducing the capacity of the cut $(U^*, \overline{U^*})$ in Definition 2 by any positive amount reduces the multicast throughput $C_{s,T}(\boldsymbol{g})$ from s to T by the same amount.

Applying the pointwise minimum rule of subgradient calculus to

$$C_{s,T}(\boldsymbol{g}) = \min_{t\in T} \rho_{s,t}(\boldsymbol{g}) = \min_{t\in T} \min_{U:\ s\in U,\ t\in\overline{U}} \boldsymbol{I}_{\delta(U)}^T \boldsymbol{g}, \tag{15}$$

we can characterize the subgradients of $C_{s,T}(\boldsymbol{g})$ in Proposition 1.

Proposition 1 (Subgradients of multicast capacity): The subdifferential of $C_{s,T}(\boldsymbol{g})$ at $\boldsymbol{g}$ is

$$\text{conv}\left\{\boldsymbol{I}_{\delta(U^*)} | (U^*, \overline{U^*}) \text{ is an } s\text{–}T \text{ critical cut in } (V, E, \boldsymbol{g})\right\} \tag{16}$$

C. Subgradient Iterations

By the assumptions on p_{vw} and U stated after (2), we have the following

Lemma 3: The objective function of (4)

$$U\left(C_{s,T}(\boldsymbol{g})\right) - \sum_{vw\in E} p_{vw}(g_{vw}) \tag{17}$$

is concave in $\boldsymbol{g}$.

Proof: The function $\rho_{s,t}(\boldsymbol{g})$ is concave, since according to the definition (13), for $\lambda_1, \lambda_2 \geq 0$

$$\rho_{s,t}(\lambda_1 \boldsymbol{g}_1 + \lambda_2 \boldsymbol{g}_2) \tag{18}$$
$$\geq \rho_{s,t}(\lambda_1 \boldsymbol{g}_1) + \rho_{s,t}(\lambda_2 \boldsymbol{g}_2) \tag{19}$$
$$= \lambda_1 \rho_{s,t}(\boldsymbol{g}_1) + \lambda_2 \rho_{s,t}(\boldsymbol{g}_2). \tag{20}$$

The pointwise minimum of a family of concave functions is also concave. Thus the lemma follows. ∎

We now look for a subgradient of (17). Let $\dot{U}(x)$ denote a subgradient of $U(x)$ at x. Let $\dot{\boldsymbol{p}}(\boldsymbol{g})$ denote a subgradient of $\sum_{vw\in E} p_{vw}(g_{vw})$ at point $\boldsymbol{g}$. This vector can be obtained by finding a subgradient for each scalar function $p_{vw}(g_{vw})$.

Proposition 2 (A Subgradient of Objective Function): A subgradient of (17) at any $\boldsymbol{g} \geq \boldsymbol{0}$ is given as follows:

$$\boldsymbol{\xi} \triangleq \dot{U}\left(C_{s,T}(\boldsymbol{g})\right) \boldsymbol{I}_{\delta(U^*)} - \dot{\boldsymbol{p}}(\boldsymbol{g}), \tag{21}$$

where $(U^*, \overline{U^*})$ is an s–T critical cut for $(V, E, \boldsymbol{g})$. This leads to the following subgradient updating rule:

$$\boldsymbol{g}^{(k+1)} = P\left[\boldsymbol{g}^{(k)} + \alpha_k \boldsymbol{\xi}^{(k)}\right]. \tag{22}$$

where $\boldsymbol{\xi}^{(k)}$ is a subgradient of (17) at the current solution $\boldsymbol{g}^{(k)}$ formed according to (21). Note that in (22), the projection is onto the Cartesian set $\{\boldsymbol{g} | \boldsymbol{0} \leq \boldsymbol{g} \leq \boldsymbol{c}\}$ and thus it decouples into finding $\min\{\max\{0, g_{vw}\}, c_{vw}\}$ for each entry g_{vw}.

Proof: We just need to show that $\dot{U}\left(C_{s,T}(\boldsymbol{g})\right) \boldsymbol{I}_{\delta(U^*)}$ is a subgradient of $U\left(C_{s,T}(\boldsymbol{g})\right)$. The following proof essentially verifies that a subgradient chain rule holds.

From the definition of subgradients

$$U(x') - U(x) \leq \dot{U}(x)(x' - x), \quad \forall x', x \geq 0 \tag{23}$$
$$C_{s,T}(\boldsymbol{g}') - C_{s,T}(\boldsymbol{g}) \leq \boldsymbol{I}_{\delta(U^*)}(\boldsymbol{g}' - \boldsymbol{g}), \quad \forall \boldsymbol{g}', \boldsymbol{g} \geq \boldsymbol{0}. \tag{24}$$

Since U is nondecreasing, we have $\dot{U}(x) \geq 0$. Substitute x and x' in (23) with $x = C_{s,T}(\boldsymbol{g})$ and $x' = C_{s,T}(\boldsymbol{g}')$, respectively. Then

$$U\left(C_{s,T}(\boldsymbol{g}')\right) - U\left(C_{s,T}(\boldsymbol{g})\right) \tag{25}$$
$$\leq \dot{U}\left(C_{s,T}(\boldsymbol{g})\right)\left(C_{s,T}(\boldsymbol{g}') - C_{s,T}(\boldsymbol{g})\right) \tag{26}$$
$$\leq \dot{U}\left(C_{s,T}(\boldsymbol{g})\right) \boldsymbol{I}_{\delta(U^*)}(\boldsymbol{g}' - \boldsymbol{g}), \quad \forall \boldsymbol{g}', \boldsymbol{g} \geq \boldsymbol{0}. \tag{27}$$

Now the above chain rule together with Propositions 1 and 2 proves Proposition 3. ■

III. Finding an s–T Critical Cut Efficiently

Finding a minimum s–t cut in graph (V, E) with edge capacities $\boldsymbol{g}$ has been well studied. For example, the preflow-push algorithm [9], [12] is a distributed algorithm for finding a minimum s–t cut. It is certainly possible to find a minimum s–t^* cut for some $t^* \in T^*$ by running the minimum s–t cut algorithm for every destination. However, it is possible to fulfill this task with lower complexity, by exploiting some algebraic properties of (random) linear network coding.

A. Review: Linear Network Coding For Acyclic Graphs With Unit Capacity Edges

Assume each edge $e \in E$ can carry one symbol from a certain finite field $\mathbb{F}$. Let y_e denote the symbol carried by edge e. Let $x_1, \ldots, x_h$, denote the *source symbols* available at the source node s. For notational consistency, we introduce h *source edges*, $s_1, \ldots, s_h$, which all end in s; the source edges $s_1, \ldots, s_h$ carry the h source symbols $x_1, \ldots, x_h$, respectively.

In a linear network coding assignment, the symbol on edge e is a linear combination of the symbols on the edges entering $\mathrm{tail}(e)$[2], namely

$$y_e = \sum_{e' \colon \mathrm{head}(e') = \mathrm{tail}(e)} w_{e,e'} y_{e'}. \tag{28}$$

We call the coefficients $\{w_{e,e'}\}$ the "mixing coefficients". By induction, y_e on any edge e is a linear combination of the source symbols, namely

$$y_e = \sum_{i=1}^{h} q_{e,i} x_i. \tag{29}$$

The vector $\boldsymbol{q}_e \triangleq [q_{e,1}, \ldots, q_{e,h}]$ is known as the *global coding vector* at edge e. It can be determined recursively as

$$\boldsymbol{q}_e = \sum_{e' \colon \mathrm{head}(e') = \mathrm{tail}(e)} w_{e,e'} \boldsymbol{q}_{e'}, \tag{30}$$

where $\boldsymbol{q}_{s_i}$ is the ith unit vector $\boldsymbol{\epsilon}_i$.

Definition 3 (Linear Network Coding Assignment): Given an acyclic graph $G = (V, E)$, a source node s, a finite field $\mathbb{F}$, and a code dimension h, a *linear network coding assignment* W refers to an assignment of mixing coefficients $w_{e,e'} \in \mathbb{F}$, one for each pair of edges (e, e') with $e \in E$, $e' \in E \cup \{s_1, \ldots, s_h\}$, and $\mathrm{head}(e') = \mathrm{tail}(e)$.

The *global coding vectors* resulting from a linear network coding assignment W, $\boldsymbol{q}_e(W)$, are the set of $|E|$ vectors $\boldsymbol{q}_e$ determined by W according to (30).

In a linear network coding assignment W, the *rank of a node* v, $\mathrm{rank}_v(W)$, refers to the rank of the span of the global coding vectors for incoming edges of v, i.e.,

$$\mathrm{rank}_v(W) \triangleq \mathrm{rank}\{\boldsymbol{q}_e(W),\ \mathrm{head}(e) = v\}. \tag{31}$$

[2]An edge e from v to w is said to have $\mathrm{tail}(e) = v$ and $\mathrm{head}(e) = w$.

Using the linear network coding result by Li, Yeung, and Cai [13], we see that

$$\mathrm{rank}_v(W) \le \min\{\rho_{s,v}, h\}, \quad \forall v \in V - \{s\}. \tag{32}$$

Now consider multicasting information from s to T using linear network coding. Using a network code assignment W, the number of distinct information symbols that can be multicast to T is

$$R(W) \triangleq \min_{t \in T} \mathrm{rank}_t(W). \tag{33}$$

Note that

$$\min_{t \in T} \mathrm{rank}_t(W) \le \min\{\min_{t \in T} \rho_{s,t}, h\} \le \min_{t \in T} \rho_{s,t} = C_{s,T}. \tag{34}$$

If a code assignment W satisfies $\min_{t \in T} \mathrm{rank}_t(W) = C_{s,T}$, then W is said to be a *capacity-achieving* code assignment, since it offers a way to multicast $C_{s,T}$ symbols while using each edge once.

It is known that random linear coding can achieve the capacity $C_{s,T}$ with high probability, for a sufficiently large field size; see, e.g., Ho *et al.* [14], Jaggi *et al.* [15]. The following particular result is from [14].

Lemma 4 (Optimality of Random Linear Coding [14]): Consider an acyclic graph $G = (V, E)$ with edges of unit capacity, a source node s, and a set of destinations T. For a code dimension $h \le C_{s,T}$ and a finite field size $|\mathbb{F}|$,

$$Pr\left[\mathrm{rank}_t(W) = h, \forall t \in T\right] \ge \left(1 - \frac{|T|}{|\mathbb{F}|}\right)^{|E|}, \tag{35}$$

where W is a random matrix with each mixing coefficient $w_{e,e'}$ chosen independently and uniformly from $\mathbb{F}$.

B. Algebraically Identifying Critical Destinations

Previous studies about random linear coding have focused on achieving the multicast capacity and hence considered using a code dimension $h \le C_{s,T}$. However, we will make use of a code dimension $h > C_{s,T}$ for the purpose of identifying the critical destinations. First, note the following easy corollary of Lemma 4.

Corollary 1: Consider an acyclic graph $G = (V, E)$ and a source node s. For any dimension $h > 0$ and finite field $\mathbb{F}$,

$$Pr\left[\mathrm{rank}_v(W) = \min\{\rho_{s,v}, h\}\right] \ge \left(1 - \frac{1}{|\mathbb{F}|}\right)^{|E|}, \quad \forall v \in V. \tag{36}$$

where W is the structured mixing matrix with each mixing coefficient $w_{e,e'}$ chosen independently and uniformly from $\mathbb{F}$. In particular, by using a sufficiently large field $\mathbb{F}$, the probability in (36) can be made arbitrarily small.

Proof: Consider each node $v \in V$. If $\rho_{s,v} \ge h$, then (36) is established upon applying Lemma 4 with $T = \{v\}$. If $\rho_{s,v} < h$, let $\boldsymbol{q}'_e$ be the subvector of $\boldsymbol{q}_e$ consisting of the first $\rho_{s,v}$ entries. Applying Lemma 4 with $T = \{v\}$ and a code dimension $\rho_{s,v}$, we see that

$$Pr\left[\mathrm{rank}\{\boldsymbol{q}'_e : \mathrm{head}(e) = v\} = \rho_{s,v},\right] \ge \left(1 - \frac{1}{|\mathbb{F}|}\right)^{|E|}.$$

Since $\text{rank}_v(W) \geq \text{rank}\{\boldsymbol{q}'_e : \text{head}(e) = v\}$, the result follows. ■

Recall that a destination t is said to be critical if $\rho_{s,t} = C_{s,T}$ and non-critical if $\rho_{s,t} > C_{s,T}$. Suppose we use random linear network coding with $h > C_{s,T}$. Then for a sufficiently large finite field $\mathbb{F}$, the rank of a critical user $t^* \in T^*$ will be close to $C_{s,T}$, whereas the rank of a non-critical user $t \in T - T^*$ will be close to $\min\{\rho_{s,t}, h\} > C_{s,T}$. This gives a method to identify the critical destinations.

There is a note on precoding worth mentioning here. With random linear coding over a sufficiently large finite field, a critical destination t^* will receive approximately $C_{s,T}$ symbols with linearly independent global coding vectors. Each symbol corresponds to a linear equation in terms of the h unknowns, $x_1, \ldots, x_h$. If $h > C_{s,T}$, there are more unknowns than the equations. How would t^* be able to recover the source information? This issue can be solved by performing *precoding*; this technique was used in [16] for robustness in a dynamic network. If the source symbols $x_1, \ldots, x_h$ are linearly coded versions of $h_0 \leq C_{s,T}$ underlying variables, then it becomes possible to recover the underlying variables. The simplest form of precoding is to set $x_{h_0+1}, \ldots, x_h$ to zero. This is sufficient for our purpose. The parameter h_0 is called the *signal space dimension* and h is called the *mixing dimension*.

We now come to implications of the above results to the implementation of our distributed utility maximization algorithm in the practical network coding system. The proposed method of identifying the critical destinations can be implemented almost "for free" in the practical network coding system [16].

To implement the proposed method of identifying the critical destinations, let the source packets in a generation be

$$\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{h_0}, \boldsymbol{x}_{h_0+1} = \boldsymbol{0}, \ldots, \boldsymbol{x}_h = \boldsymbol{0}. \tag{37}$$

Thus the payload of a packet with global coding vector $\boldsymbol{q} = [q_1, \ldots, q_h]$ will be

$$\sum_{i=1}^{h_0} q_i \boldsymbol{x}_i. \tag{38}$$

Since each payload does not involve $[q_{h_0+1}, \ldots, q_h]$, these entries of $\boldsymbol{q}$ are not used in the final decoding at each destination. These amount to the overhead incurred for testing criticality of destinations.

To leave a margin against temporary network outages, we should set h_0 such that the corresponding rate is slightly below the nominal multicast capacity. This assures that decoding will be successful at each destination with high probability. We should set h such that the corresponding rate is slightly above the multicast capacity. In this way, the critical destinations will exhibit lower ranks than the noncritical destinations.

IV. Summary of the Proposed Algorithm

We now summarize the proposed distributed algorithm. It is assumed that each node in the network can store data pertaining to its outgoing edges and perform computations on them.

For simplicity, let us assume that the algorithm is already in the steady phase. In other words, we assume that the practical network coding system is already up running. Thus we can use the rank information collected at the destinations for identifying the critical destinations.

Algorithm 1 (Primal subgradient method via critical cuts): Suppose a current solution $\boldsymbol{g}^{(k)}$ is just obtained. This vector is stored in a distributed fashion in the network. Suppose further that the source s has a coarse estimate of the range of $C_{s,T}(\boldsymbol{g}^{(k)})$, e.g., via $C_{s,T}(\boldsymbol{g}^{(k-1)})$. Then the following steps are performed.

1) Set the practical network coding system with the parameters h_0 and h at the source such that the multicast rate associated with h_0 is slightly below $C_{s,T}(\boldsymbol{g}^{(k)})$ and the multicast rate associated with h is slightly above $C_{s,T}(\boldsymbol{g}^{(k)})$; see Section III-B for details. Each destination monitors the rank information for one or more generations of data multicasting and reports the information to s.
2) Using the reported rank information, the source s determines $C_{s,T}(\boldsymbol{g}^{(k)})$ (by taking the minimum of the rank of the destinations and dividing it by the time interval), as well as a worst destination t^* which has the worst (minimum) throughput. The source s then initiates the execution of the preflow-push algorithm, to find a minimum s–t^* cut $(U^*, \overline{U^*})$. This represents an s–t^*-critical cut.
3) The source s conveys the value $\dot{U}(C_{s,T}(\boldsymbol{g}^{(k)}))$ to all nodes that have an outgoing edge (or incoming edge, depending on whether g_{vw} is stored at v or w) that goes from U^* to $\overline{U^*}$.
4) The subgradient update (22) is implemented in parallel. For each edge vw that goes from U^* to $\overline{U^*}$, set
$$g_{vw}^{(k+1)} = P_{vw}\left[g_{vw}^{(k)} + \alpha_k\left(\dot{U}(C_{s,T}(\boldsymbol{g}^{(k)})) - \dot{p}(g_{vw}^{(k)})\right)\right],$$
where $P_{vw}[x] = \min\{\max\{x, 0\}, c_{vw}\}$. For each other edge, set
$$g_{vw}^{(k+1)} = P_{vw}\left[g_{vw}^{(k)} - \alpha_k \dot{p}(g_{vw}^{(k)})\right]. \tag{39}$$

V. Extension to Multiple Multicast Sessions

Consider the scenario where there are multiple multicast sessions in the network. Label them with indices $m = 1, \ldots, M$. Denote the source and the destination set of the mth session by s_m and T_m, respectively. Assume the mth multicast session communicates using its exclusive share of resource $\boldsymbol{g}_m$. In this case the multicast rate for this session is

$$C_{s_m,T_m}(\boldsymbol{g}_m) \triangleq \min_{t \in T_m} \rho_{s_m,t}(\boldsymbol{g}_m). \tag{40}$$

We consider the following optimization, where the variables are $[\boldsymbol{g}_1; \ldots; \boldsymbol{g}_M]$

$$\begin{aligned} \text{maximize} \quad & \sum_{m=1}^{M} U_m\left(C_{s_m,T_m}(\boldsymbol{g}_m)\right)) - \sum_{vw \in E} p_{vw}\left(\sum_{m=1}^{M} g_{vw}^m\right) \\ \text{subject to:} \quad & \boldsymbol{0} \leq \boldsymbol{g}_m, \quad \forall m \\ & \boldsymbol{g}_1 + \ldots + \boldsymbol{g}_M \leq \boldsymbol{c} \end{aligned} \tag{41}$$

A subgradient of the objective function can be obtained as the stacked vector $[\boldsymbol{\xi}_1; \ldots; \boldsymbol{\xi}_M]$, where

$$\boldsymbol{\xi}_m \triangleq \dot{U}_m \left(C_{s_m,T_m}(\boldsymbol{g}_m) \right) \boldsymbol{\zeta}_m(\boldsymbol{g}_m) - \dot{\boldsymbol{p}} \left(\sum_{m=1}^{M} \boldsymbol{g}_m \right) \tag{42}$$

and $\boldsymbol{\zeta}_m(\boldsymbol{g}_m)$ is a subgradient of $C_{s_m,T_m}(\boldsymbol{g}_m)$:

$$\boldsymbol{\zeta}_m(\boldsymbol{g}_m) \in \partial C_{s_m,T_m}(\boldsymbol{g}_m). \tag{43}$$

Now the projection of the updated vector $\boldsymbol{g}^{(k+1)}$ to the feasible region of (41) becomes slightly more complicated. Since the constraints in (41) are decoupled for the edges, we need to solve the following problem for each edge $vw \in E$

$$\begin{aligned} P[\boldsymbol{g}_{vw}] \triangleq \arg\min_{\boldsymbol{g}'_{vw}} \quad & \|\boldsymbol{g}'_{vw} - \boldsymbol{g}_{vw}\|^2 \\ \text{subject to:} \quad & \mathbf{0} \leq \boldsymbol{g}'_{vw}, \\ & \mathbf{1}^T \boldsymbol{g}'_{vw} \leq c_{vw}, \end{aligned} \tag{44}$$

where $\boldsymbol{g}_{vw} \triangleq [g^1_{vw}, \ldots, g^M_{vw}]^T$, and $\mathbf{1}$ is a vector of all 1's.

VI. Simulations

The proposed algorithm has been tested on a large scale problem. The test graph (V, E) is the topology of an ISP (Exodus) backbone obtained from the Rocketfuel project at the University of Washington [17]. There are 79 nodes and 294 links. We arbitrarily placed a source node at New York, and 8 destination nodes at Oak Brook, Jersey City, Weehawken, Atlanta, Austin, San Jose, Santa Clara, and Palo Alto. The utility function, the link cost functions and link capacities are set as

$$U(r) = \ln(1 + r), \tag{45}$$

$$p_{vw}(g) = 0.005g, \quad \forall vw \in E, \tag{46}$$

$$c_{vw} = 10, \quad \forall vw \in E. \tag{47}$$

Figure 1 presents the simulation results. The straight line is the optimal net-utility U^*_{net} and the other curve corresponds to $U^{(k)}_{\text{net}}$ generated by the proposed primal subgradient algorithm for step size $h = 1.0$. The results confirm the convergence of the proposed algorithm to a small neighborhood of the global optimum when a constant step size is used, as predicted by Lemma 2.

VII. Conclusion

For network coding–based multicasting, we have proposed a net-utility maximization problem to capture the tradeoff between multicast throughput utility and resource provisioning costs. By deriving a subgradient through indicator vectors for $s - T$ critical cuts, we propose a distributed algorithm to globally solve the utility maximization problem. Further exploiting the algebraic properties of practical linear network coding, we show a low-complexity efficient implementation of the algorithm.

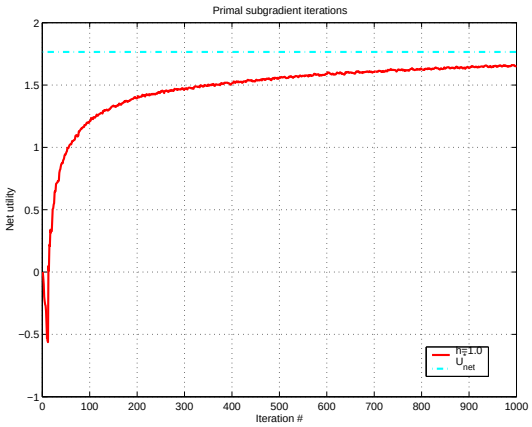

Fig. 1. Primal subgradient iterations for the large scale test case.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.

[2] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of Operations Research Society*, vol. 49, no. 3, pp. 237–252, Mar. 1998.

[3] S. Kunniyur and R. Srikant, "End-to-end congestion control: utility functions, random losses and ecn marks," *IEEE/ACM Trans. on Networking*, vol. 10, no. 5, pp. 689–702, Oct. 2003.

[4] R. J. La and V. Anantharam, "Utility-based rate control in the internet for elastic traffic," *IEEE/ACM Trans. on Networking*, vol. 9, no. 2, pp. 272–286, Apr. 2002.

[5] S. H. Low and D. E. Lapsley, "Optimization flow control, i: basic algorithm and convergence," *IEEE/ACM Trans. on Networking*, vol. 7, pp. 861–874, Dec. 1999.

[6] D. S. Lun, N. Ratnakar, R. Koetter, M. Medard, E. Ahmed, and H. Lee, "Achieving minimum-cost multicast: A decentralized approach based on network coding," in *Proc. INFOCOM*. Miami, FL: IEEE, Mar. 2005.

[7] Y. Wu and S.-Y. Kung, "Distributed utility maximization for network coding based multicasting: a shortest path approach," *IEEE J. Selected Areas in Communications*, Sept. 2005, submitted.

[8] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition," *Proceedings of the IEEE*, 2006, to appear. Short version in *Proc. Conf. Inform. Sci. and Systems*, Princeton University, March 2006.

[9] A. V. Goldberg and R. E. Tarjan, "A new approach to the maximum flow problem," in *Proc. 18th ACM Symp. Theory of Computing*, 1986, pp. 136–146.

[10] N. Z. Shor, *Minimization methods for non-differentiable functions*, ser. Springer Series in Computational Mathematics. Springer-Verlag, 1985, translated from Russian by K. C. Kiwiel and A. Ruszczyński.

[11] D. P. Bertsekas, A. Nedić, and A. E. Ozdaglar, *Convex Analysis and Optimization*. Belmont, MA: Athena Scientific, 2003.

[12] A. V. Goldberg and R. E. Tarjan, "A new approach to the maximum flow problem," *Journal of the ACM*, vol. 35, no. 4, pp. 921–940, Oct. 1988.

[13] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inform. Theory*, vol. IT-49, no. 2, pp. 371–381, Feb. 2003.

[14] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inform. Theory*, 2004, submitted. http://web.mit.edu/trace/www/.

[15] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for network code construction," *IEEE Trans. Inform. Theory*, vol. 51, pp. 1973–1982, June 2005.

[16] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. 41st Allerton Conf. Comm., Ctrl. and Comp.*, Monticello, IL, Oct. 2003.

[17] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," in *Proc. SIGCOMM*. Pittsburg, PA: ACM, Aug. 2002.