Ascending-Bid Auction for Unequal-Erasure-Protected Network Coding

Apirath Limmanee and Werner Henkel
Jacobs University
EECS, TrSys, Campus Ring 1
28759 Bremen, Germany
Email: {a.limmanee, w.henkel}@jacobs-university.de

***Abstract*—According to our previous paper, global encoding kernels (GEKs) describing linear network codes have different levels of built-in unequal-erasure-protecting (UEP) capability. This creates conflicts among sink nodes in a multicast session since a particular assignment of GEKs favors one sink node over another. This paper proposes a resolution to the conflict by means of a simple ascending-bid auction scheme which has three obvious benefits. First, it frees all the nodes from the complicated optimization algorithms that might otherwise be used. Second, it provides the source node with some revenue. Lastly, it allows the richer sink nodes to receive the data with better quality.**

I. Introduction

It was proven by Ahlswede et al. [1] that the maximum data flow from the source to each destination in a multicast session could only be guaranteed if network coding was applied. Later, Li et al. [2] proved that a simple class of network coding, called linear network coding, sufficed to achieve the maximum flow.

Although recent works on network coding consider errors and erasures in networks [3], [8], [9], [10], earlier papers model networks as graphs in which each edge represents an erasure-free channel with a unit capacity [5], [6], [7]. In this paper, however, each edge represents the data transmission rate of one symbol per unit time in a binary erasure channel (BEC), i.e., the edge capacity is reduced from 1 to $1-p$, if p denotes the erasure probability.

Scalable video and image data, such as Scalable Video Coding (SVC) recently standardized by JVT as an extension of H.264/AVC, consists of several layers of data [4]. Upper layers represent fine details added to lower ones. Scalable data therefore asks for unequal erasure protection (UEP), sometimes mentioned as unequal loss protection (ULP), such that the parts of data with higher priority are better protected against erasures. UEP in network codes are studied in [3] and [10].

Erasures in a coded network create a problem regarding the assignment of network codes that maximizes the satisfaction of the individual receivers in a multicast session. However, as will be shown in Section 2, when transmitted data consists of data with different priorities, an assigned set of network codes ends up favoring one receiver at the expense of others. Moreover, although all receivers enjoy similar satisfaction by receiving data with high decoding probability, the amount of "money" that they are willing to pay differs. Therefore, we start our economic analysis in Section 3 and derive the satisfaction that a receiver is expected to gain. As Section 4 shows, the satisfaction relates to how the global encoding functions (GEKs) describing linear network codes are assigned to the edges. Every receiver will compete for the rights to choose the GEK assignment to maximize its satisfaction. The competition is carried out by means of an ascending-bid auction discussed in Section 5, in which the detailed implementation is also given. After that, Section 6 concludes this work.

II. The Edge-Disjoint Path Model with Binary Erasure Channels and UEP Conflict

Figure 1 shows network coding in a network that is usually called the butterfly network, where A aims to multicast two binary symbols b_1 and b_2 to D and E. We can see that the node F encodes b_1 and b_2 together to achieve the multicast rate of 2 bits per unit time, if each edge represents the capacity of one bit per unit time, D receives b_1 from the path ABD and can recover b_2 from the symbol $b_1 \oplus b_2$ from $ACFGD$, whereas E receives b_2 from the path ACE and recovers b_1 from the symbol $b_1 \oplus b_2$ from $ABFGE$. Had network coding not been in place, only either b_1 or b_2 would have been able to pass the bottleneck FG in one unit time, i.e., one receiver would have been unable to use one of its possible transmission paths.

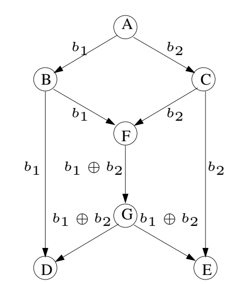

Fig. 1. Network coding in a butterfly network

By means of network coding, all receivers can use all of their possible paths at the same time. In general cases, the multicast rate of ω requires ω non-intersecting paths from the source to each sink, which are called edge-disjoint paths by Jaggi et al. [6], although the paths destined to different receivers may share some edges.

If each edge is modeled as a binary erasure channel (BEC), the erasure probability of an edge-disjoint path can be computed from erasure probabilities of all the edges from the source to the sink. The loss of a symbol in each edge-disjoint path does not affect the recovery in another. For example, the loss of b_1 at the edge BF does not affect the recovery of b_2 at D via $ACFGD$.

Let us consider Fig. 1 again and see what will happen if one symbol is received at each sink whereas another one is erased. At D, if $b_1 \oplus b_2$ is erased, D still obtains b_1. But if b_1 is erased, D obtains neither b_1 nor b_2. This means, for D, b_1 is better protected than b_2. On the other hand, for E, b_2 is better protected. In case b_1 and b_2 are equally important, the network coding is fair. However, if b_1 is more important than b_2 and the erasure probability of each symbol is the same, D is in favor because its more important symbol is better protected [3]. Thus, to achieve fairness and optimality, the network encoding function of each edge-disjoint path should be carefully chosen.

III. Marginal Values of Scalable Data and the Receivers' Satisfaction

Definition 1: For any $\mathcal{S} = \{s_1, s_2, ..., s_\omega\}$, representing a set of scalable data with progressively increasing quality from s_1 to s_ω, the i^{th} element s_i can be mapped into a prefix vector $\mathbf{P}_i = [m_1, m_2, ..., m_i]$ of a scalable message $\mathbf{M}= [m_1, m_2, ..., m_\omega]$, which is an ω-dimensional row vector of a finite field $\mathbb{F}$. [3]

$$s_i \longmapsto [m_1, m_2, ..., m_i] \tag{1}$$

Definition 2: A symbol m_k belonging to the scalable message $\mathbf{M} = [m_1, m_2, ..., m_\omega]$ has a dependency level of $\lambda(m_k) = j$ if its significance depends on the successful recovery of the symbols having the dependency level of $j-1$ but not on those with larger dependency level. A symbol of which significance does not depend on any symbol has the dependency level of 1. [3]

Definition 3: A functional vector $\mathbf{V_k}(\mathcal{S}) = [v_k(s_1), v_k(s_2), ..., v_k(s_\omega)]$, $k = 1, 2, ..., N$, where N is the number of sink nodes in the network-coded multicast, is called the cumulative value vector assigned by the sink node k to the set of scalable data $\mathcal{S}$ described in Definition 1 if each element $v_k(s_i)$, $1 \leq i \leq \omega$, is a non-negative real number representing the private value that the sink node k assigns to the scalable data s_i.

Definition 4: A functional vector $\Delta\mathbf{V_k}(\mathcal{S})$ is called the marginal value vector assigned by the sink node k to the set of scalable data $\mathcal{S}$ described in Definition 1 if

$$\begin{aligned} \Delta\mathbf{V_k}(\mathcal{S}) &= [\Delta_1, \Delta_2, ..., \Delta_\omega] \qquad (2)\\ &= [v_k(s_1) - v_k(s_0), v_k(s_2) - v_k(s_1), ... \\ &\quad ..., v_k(s_\omega) - v_k(s_{\omega-1})], \qquad (3) \end{aligned}$$

where $v_k(s_0) = 0$ and $v_k(s_i)$, $1 \leq i \leq \omega$, represents the element in the cumulative value vector $\mathbf{V_k}(\mathcal{S})$ defined in Definition 3.

Definition 5: The set of scalable data $\mathcal{S}$ is said to be ordered if and only if the two following conditions are fulfilled.

$$\lambda(m_u) \geq \lambda(m_v), 1 \leq v < u \leq \omega \tag{4}$$

$$\Delta_{i-1} \geq \Delta_i, 1 < i \leq \omega \tag{5}$$

where $\lambda(m_j)$ and Δ_i denote the dependency level of the symbol m_j and the i^{th} element in the marginal value vector of $\mathcal{S}$, respectively. A message $\mathbf{M}$ corresponding to an ordered set $\mathcal{S}$ of scalable data is called an ordered scalable message.

From definitions 1 to 5, we can relate the expected value of received scalable data at a sink i to the probability that each symbol in the scalable message is successfully decoded. To simplify our analysis, we consider an ordered scalable message $\mathbf{M}= [m_1, m_2, ..., m_\omega]$, in which each element m_u has a dependency level of u, i.e., every symbol depends on all of its predecessors. In this case, the expected quality value of received scalable data at i can be written as

$$\begin{aligned} E[V_i^{\mathcal{S}}] &= \sum_{j=1}^{\omega} \left[\prod_{l=1}^{j} \varrho_{i,l}\right] \cdot \Delta_j \qquad (6)\\ &= \sum_{j=1}^{\omega} \rho_{i,j} \cdot \Delta_j, \end{aligned}$$

where $\varrho_{i,l}$ and $\rho_{i,j}$ represent the probabilities that the symbol m_l and the prefix $\mathbf{P}_j$ are successfully decoded at the sink i, respectively. Note that the sink i's overall quality judgment about the received data may differ from this expected value. It can be any non-linear function $\mathcal{I}_i = f(\rho_{i,j}, \Delta_j)$, $j = 1, 2, ..., \omega$. In any case, it is assumed that the general impression, or at least its estimate, can be quantitatively represented by a positive real number so that the sink i has a basis for its bidding decision. In the next section, we show how $\rho_{i,j}$ relates to the sink i's choice of network codes.

IV. UEP Levels of Global Encoding Kernels for Linear Network Codes

For a network that employs linear network coding, each of its edges in the graphical model, such as Fig. 1, is used to transmit the linear combination of the source symbols. This linear combination can either be represented locally as a linear function of symbols from adjacent edges, which is called "the local encoding mapping", or globally as a linear function of source symbols, which is called "the global encoding mapping" [5]. The global encoding mapping is described by a vector called "the global encoding kernel (GEK)," which is introduced in Definition 6.

Definition 6: Let $\mathbb{F}$ be a finite field, ω a positive integer, and the ω-dimensional, $\mathbb{F}$-valued row vector $\mathbf{M}$ the message generated by the source node $\mathcal{S}$. A function $f_e(\mathbf{M})$ of the edge e is said to be a linear global encoding mapping if there exists an ω-dimensional $\mathbb{F}$-valued column vector $\mathbf{f}_e$ such that

$$f_e(\mathbf{M}) = \mathbf{M} \cdot \mathbf{f}_e. \tag{7}$$

$\mathbf{f}_e$ is called the global encoding kernel [5].

We only consider the problem of assigning, given some local constraints, a suitable global encoding kernel for each edge, since the local encoding mapping can be easily derived thereafter.

To relate the $\rho_{i,j}$ to the network codes, we firstly define the UEP level of a global encoding mapping as follows.

Definition 7: For a scalable message $\mathbf{M}$, which is an ω-dimensional row vector of a finite field $\mathbb{F}$, a global encoding mapping $\gamma_i(\mathbf{M})$ is of the i^{th} UEP level, $0 < i \leq \omega$, if there exists an ω-dimensional, $\mathbb{F}$-valued column vector $\mathbf{C}_i = [c_1, c_2, ..., c_i, 0, 0, ..., 0]^T$, $c_i \neq 0$, such that

$$\gamma_i(\mathbf{M}) = \mathbf{M} \cdot \mathbf{C}_i = \sum_{j=1}^{i} c_j \cdot m_j. \quad (8)$$

$\mathbf{C}_i$ is then called an i^{th}-UEP-level global encoding kernel [3].

Now, let us consider an ordered scalable message $\mathbf{M} = [m_1, m_2, m_3]$, where m_1, m_2, and m_3 are binary symbols of the first, second, and third dependency level, respectively. According to Definition 7, there exist seven possible GEKs, one of the first UEP level, two of the second level, and four of the third level, as shown in Table I.

TABLE I
GEKS, THEIR UEP LEVELS, AND THE RESULTING NETWORK-CODED SYMBOLS

UEP Levels	GEKs	Resulting Network-Coded Symbols
1	$[1\ 0\ 0]^T$	m_1
2	$[0\ 1\ 0]^T$ $[1\ 1\ 0]^T$	m_2 $m_1 + m_2$
3	$[0\ 0\ 1]^T$ $[1\ 0\ 1]^T$ $[0\ 1\ 1]^T$ $[1\ 1\ 1]^T$	m_3 $m_1 + m_3$ $m_2 + m_3$ $m_1 + m_2 + m_3$

From Table I, the prefix $\mathbf{P}_2 = [m_1, m_2]$ can be recovered either from any two symbols from the levels 1 and 2 or from any three symbols from the level 3. For an arbitrary scalable message $\mathbf{M}= [m_1, m_2, ..., m_\omega]$, we can state as a general rule that, in order to recover the prefix $\mathbf{P}_i$, we need either i network coded-symbols belonging to i linearly independent GEKs of which UEP levels do not exceed i, or more than i symbols in case some UEP levels of GEKs exceed i. [3]

Reconsidering (6), assuming that the k^{th} edge-disjoint path of the sink i is used to transmit the k^{th} network-coded symbol to the sink i, the parameter $\rho_{i,j}$, which is the probability that the prefix $\mathbf{P}_j$ is recovered at the sink i, can be written as

$$\rho_{i,j} = \prod_{k=1}^{\omega} \mu_{j,k} \cdot [1 - P_{e,ik}], \quad (9)$$

where $P_{e,ik}$ denotes the erasure probability of the path k used to transmit the k^{th} network-coded symbol to the sink i. $\mu_{j,k} = 1$ if the k^{th} network-coded symbol is needed to recover the prefix $\mathbf{P}_j$. Otherwise, $\mu_{j,k} = 0$ [3].

Equations (6) and (9) show that, by changing the GEKs allocated to the edges, the expected value of received data, or any other function $\mathcal{I}_i = f(\rho_{i,j}, \Delta_j)$, $j = 1, 2, ..., \omega$ representing the satisfaction of the receiver i, varies. Therefore, if the receiver i is allowed to assign a GEK to an edge or a set of edges, it can do so in such a way that its satisfaction increases. The more GEKs a node i is allowed to choose, the better its satisfaction. This means there is a marginal value associated with each increase in the number of such permissions, which is defined as follows.

Definition 8: A functional vector $\mathbf{\Theta}_i$ is called the marginal value vector assigned by the sink node i to the GEK allocation permissions if

$$\begin{aligned} \mathbf{\Theta}_i &= [\theta_1, \theta_2, ..., \theta_\zeta] \quad (10) \\ &= [\mathcal{I}_{bi1} - \mathcal{I}_{wi1}, \mathcal{I}_{bi2} - \mathcal{I}_{wi2}, ... \\ &\quad ..., \mathcal{I}_{bi\zeta} - \mathcal{I}_{wi\zeta}], \quad (11) \end{aligned}$$

where ζ is the number of possible permissions. $\mathcal{I}_{bij}$ is a non-negative real number reflecting the maximum satisfaction of the sink node i if it is allowed to allocate j GEKs, whereas $\mathcal{I}_{wij}$ is the one when it is only allowed to allocate $j - 1$ GEKs before one worst-case GEK is assigned to it in the j^{th} allocation. Thus, the difference between $\mathcal{I}_{bij}$ and $\mathcal{I}_{wij}$ reflects how important it is for i to receive the j^{th} allocation permission.

According to (6) and (9), the best option for the sink i is to make $\rho_{i,j}$ large for small j to satisfy the UEP requirements and optimize the scalable data quality. To do so, it first sorts the erasure probability $P_{e,ik}$ such that $P_{e,ik} \leq P_{e,ir}$ for any $1 \leq k < r \leq \omega$, and, after that, allocates a q^{th} level GEK to the path with the index $k = q$ [3].

However, as earlier discussed in Section 3, this simple allocation scheme may not be possible due to conflicts among sink nodes as well as linear independence and dependence constraints of GEKs. Linear independence constraints ensure that the maximum information flow is achieved at each node, whereas linear dependence represents topological constrains, i.e., the GEK of any outgoing edge of the node i must be linearly dependent on the GEKs of incoming edges. The conflict resolution by means of auction is discussed in detail in the next section.

V. ASCENDING-BID AUCTION ALGORITHMS FOR GEK ALLOCATION

We assume that the source node is the auctioneer and the sinks are participants who bid for the rights to choose the GEK allocation. Since we will not give a monopoly to any single sink node, the allocation problem is formulated as a multiple-item auction. For this kind of auction, the dynamic counterpart of Vickrey's effective static design [12] has recently been proposed by Ausubel [13], whose idea will be used in our auction as follows. The source node calls a price, bidders respond with quantities, and the process iterates with increasing price until demand is not greater than supply.

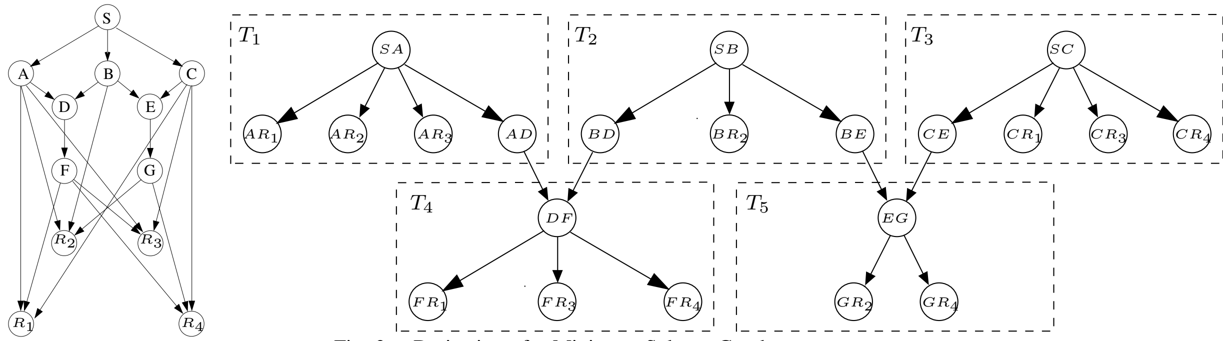

Fig. 2. Derivation of a Minimum Subtree Graph

Although the supply is the total number of GEK allotments, it does not equal the total number of edges, because some edges are forced by the topology to have the same GEK. Thus, before the auction begins, the source node has to derive the minimum subtree graph of the network. An example is shown in Fig. 2, in which the network on the left is transformed into a line graph on the right such that each node in the right graph corresponds to an edge in the left one. Then, the nodes in the right graph can be grouped into five subtrees, each of which is surrounded by dashed lines, such that the members in each subtree are forced by the topology to have the same GEK. In this case, the problem of assigning GEKs to twenty-six edges is reduced to that of assigning them to six subtrees.

Let $\mathcal{T}$ represent the set of all subtrees. The source wishes to allocate $|\mathcal{T}|$ GEKs to $|\mathcal{T}|$ subtrees, not on its own but by means of a $|\mathcal{T}|$-item auction. The sink i who wins x_i items is allowed to choose the allocation of x_i GEKs to x_i subtrees in order to maximize its satisfaction. The source then distributes the data accordingly and informs the intermediate coding nodes about the local encoding functions they have to use.

According to Ausubel's idea, a bidder's payment is not the product of its final quantity and the final price. Rather, at the price ψ, the auctioneer sees if the aggregate demand x_{-i} of bidder i's competitiors is less than the supply $|\mathcal{T}|$. If so, $|\mathcal{T}| - x_{-i}$ items are clinched and awarded to i with the price ψ. Since the winner's payment depends on its competitors bids and not its own bids, every participant has incentive to reveal truthfully its value for the item. [13]

Our auction, however, requires some modification since each clinched item and the resulting GEK allocation of a winning sink affects the demand of others. This is best explained by an example.

TABLE II
PREFERENCE ORDER OF GEK ALLOCATION

Sink	Preference Order		
R_1	T_4	T_3	T_1
R_2	T_2	T_1	T_5
R_3	T_1	T_4	T_3
R_4	T_5	T_4	T_3

Suppose that the source node S in Fig. 2 would like to multicast an ordered scalable message $\mathbf{M} = [m_1, m_2, m_3]$, of which each element m_i has a dependency level of i, to four sink nodes R_1, R_2, R_3, and R_4. We assume that Table 2 reflects the preference order of GEK allocation, e.g., if R_1 is allowed to allocate only one GEK, it will choose the best one and assign it to T_4. If two allotments are allowed, R_1 will assign two GEKs to T_4 and T_3. The preference order relates to the erasure probability of each edge-disjoint path discussed in earlier sections. Each sink's main preference is to allocate the GEKs with lower UEP levels to the paths with lower erasure probability. This means, according to Table 2, the path from the source that reaches R_1 via T_4 has lower erasure probability than those reaching R_1 via T_3 and T_1.

Let each receiver's estimate of its marginal values θ_1, θ_2, and θ_3, as defined in Definition 8 be shown below.

$\mathbf{\Theta}$	R_1	R_2	R_3	R_4
θ_1	123	75	85	45
θ_2	113	5	65	25
θ_3	40	3	7	5

Now, let the auction start with the initial price of 10. At this price, R_1 is happy to buy three allotments, while R_2 will buy only one, since the price exceeds the second marginal value. Accordingly, the response from each receiver is shown in the first row of Table 3. Since nobody cliches anything at this point, the source node raises the price. When the price reaches 25, R_4 has no profit to be gained from the second allotment, and therefore changes its response, as shown in the second row of Table 3.

TABLE III
THE SINKS' RESPONSES TO THE INCREASING PRICE

Sinks	R_1	R_2	R_3	R_4
Responses to 10	3	1	2	2
Responses to 25	3	1	2	1
Responses to 25+ϵ	3	1	1	1
Responses to 40	2	1	1	1

From R_1's perspective, the demands of all other bidders are four, while five allotments are available. If other sinks bid monotonically, R_1 is now guaranteed to win at least one allotment. Thus, according to our rule, R_1 clinches one allotment at the price of 25. It then chooses the first-level GEK to allocate to T_4. This allocation affects the marginal values of every sink except R_1. They are updated as follows.

$\mathbf{\Theta}$	R_1	R_2	R_3	R_4
θ_1	123	75	85	45
θ_2	113	4	6	3
θ_3	40	2	-	-

Due to the allocation of T_4, R_3 and R_4's last entries are removed since they now have only (T_1, T_3) and (T_5, T_3), respectively, to bid for. At the next announced price 25+ϵ, R_3's response changes, as shown in the third row of Table 3.

Since R_1 is now guaranteed to win two items, it is allowed to allocate one GEK to T_3. After that, the marginal values are updated again, as follows.

$\mathbf{\Theta}$	R_1	R_2	R_3	R_4
θ_1	123	71	62	42
θ_2	113	3	-	-
θ_3	40	1	-	-

At a price of 40, the demand equals the supply, as shown in the fourth row of Table 3, and the market clears. Each of R_2, R_3, and R_4 clinches one object at this price and assigns a GEK to T_2, T_1, and T_5, respectively.

The algorithm implementing the auction at the source node is shown as follows.

Problem: Allocate a GEK $\mathbf{f}_t \in \mathbb{F}^\omega$ to each $t \in \mathcal{T}$ such that the sets of linear independence and dependence constraints are satisfied.
1. Initialize the number of available items $N_T = |\mathcal{T}|$, the cumulative clinches $C = 0$, the cumulative quantity $X = 0$, the individual current clinches $\gamma_i = 0$ and individual cumulative clinches $\Gamma_i = 0$ for the node i, $i = 1, 2, ..., n$. Set the current price ψ as the initial price ψ_0. Broadcast linear dependence and independence constraints to all sink nodes.
2. At an appropriate time $t + \Delta t_i$, where Δt_i is the time offset calculated from the distance between the source and the sink i, send the current price ψ to the sink i and wait for response.
3. Upon receiving the quantity x_i from every sink i, update X, Γ_i, and γ_i as follows.

$$X = \sum_{i=1}^{n} x_i \tag{12}$$

$$x_{-i} = X - x_i \tag{13}$$

$$Y_i = \Gamma_i \tag{14}$$

$$Z = C \tag{15}$$

$$\Gamma_i = \begin{cases} |\mathcal{T}| - x_{-i} & \text{if } |\mathcal{T}| > x_{-i} \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

$$\gamma_i = \Gamma_i - Y_i \tag{17}$$

$$C = \sum_{i=1}^{n} \Gamma_i \tag{18}$$

4. From 3, if $C - Z > 0$, go to 5. Otherwise, go to 10.
5. Find the index $j \in \mathcal{K}$, $\mathcal{K} = \{\forall k | \gamma_k \neq 0\}$ such that $x_j \geq x_i$, for every $i \neq j$, $i \in \mathcal{K}$. If more than one x_j are found, randomly select one of them.
6. Inform the sink node j about the individual current clinches γ_j. Wait for response.
7. If the response is negative, let $x_j = x_j - 1$ and go back to 3. In case the affirmative response, together with the GEK $\mathbf{f}_r$ that j chooses to allocate to the subtree $r \in \mathcal{T}$, is received, check if the allocation violates the linear dependence or independence constraints. If so, randomly allocate a GEK satisfying the constraints to r. If not, allocate $\mathbf{f}_r$ to r.
8. Broadcast the index of the allocated subtree in 7 and its GEK to all nodes.
9. Let $Z = Z + 1$, $\mathcal{K} = \mathcal{K} - \{j\}$, and $N_T = N_T - 1$. If $N_T > 0$, go back to 4. Otherwise, the algorithm ends.
10. Update the price such that $\psi = \psi + \Delta\psi$. Go back to 2.

In Step 3, the cumulative clinches Γ_i for the sink i is computed as the difference between the supply $|\mathcal{T}|$ and the aggregate demand x_{-i} of its opponents. Y_i is simply a variable used to store the previous Γ_i such that, after Γ_i is updated in (16), the individual clinches γ_i at the current price can be computed from (17). In a similar manner, the variable Z in (15) is used to count the previous cumulative clinches C, which will be updated in (18).

In Step 4, if the updated cumulative clinches C is greater than the previous clinches Z, we continue with Steps 5 to 8, which allow one GEK to be allocated. According to Steps 5 and 6, when more than one sink node clinches some objects at a specific price, the nodes which bid for higher quantity are allowed to choose the allocation prior to those bidding for lower quantity.

Step 9 increases Z by 1 and checks whether there are still some items available. If there is, we return to Step 4 to compare the increased Z with C. If C is still greater than Z, we repeat Steps 5 to 9. Otherwise, no more item is clinched at this price and the price is raised in Step 10.

VI. CONCLUSIONS

The proposed algorithm running at the source has a complexity that increases linearly with the number of sinks n. Since, at every time an item is clinched, each sink node needs to update the marginal values of GEK allocation permissions described in Definition 8, the computational complexity at each sink grows linearly with the number of subtrees $|\mathcal{T}|$.

The distributed nature of the auctioning algorithm offers one clear advantage over a centralized optimization algorithm: If a sink ends up buying an item at an inappropriate price, it has done something wrong in the estimation of marginal values, thus having only itself to blame. However, if the same thing occurs with the algorithm centralized at the source, the source would easily be accused of being unfair.

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, Jul. 2000.

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

[3] A. Limmanee, W. Henkel, "UEP Network Coding for Scalable Data," 5^{th} *Int. Symp. on Turbo Codes and Related Topics*, Sep. 2008.

[4] S.-K. Chang, K.-C. Yang, and J.-S. Wang "Unequal-Protected LT Code for Layered Video Streaming," *IEEE Int. Conf. on Communications*, May 2008.

[5] R.W. Yeung, S.-Y.R. Li, N. Cai, and Z. Zhang, "Network Coding Theory," *Foundation and Trends in Communications and Information Theory*, vol. 2, nos. 4 and 5, pp. 241-381, 2005.

[6] S. Jaggi, P. Sanders, P.A. Chou, M. Effros, S. Egner, K. Jain, L. Tolhuizen, "Polynomial Time Algorithms for Multicast Network Code Construction," *IEEE Trans. Inform. Theory*, vol. 51, no. 6, pp. 1973-1982, Jun. 2005.

[7] C. Fragouli, E. Soljanin, A. Shokrollahi, "Network Coding as a Coloring Problem," in *Proc. Conf. Information Sciences and Systems.* Princeton, NJ, Mar. 2004.

[8] R.W. Yeung, and N. Cai, "Network Error Correction, Part I: Basic Concepts and Upper Bounds," *Communications in Information and Systems*, vol. 6, no. 1, pp. 19-36, 2006.

[9] R. Koetter, and F.R. Kschischang, "Coding for Errors and Erasures in Random Network Coding," *IEEE Trans. Inform. Theory*, vol. 54, no. 8, pp. 3579-3591, Aug. 2008.

[10] Y. Lin, B. Li, and B. Liang, "Differentiated Data Persistence with Priority Random Linear Codes," 27^{th} *Int. Conf. on Distributed Computing Systems*, Jun. 2007.

[11] A.P. Lerner, *The Economics of Control*, Augustus M. Kelley Publishers, NY, 1970.

[12] W. Vickrey, "Counterspeculation, Auctions, and Competitive Sealed Tenders," *Journal of Finance*, vol. 16, no. 1, pp. 8-37, Mar. 1961.

[13] L.M. Ausubel, "An Efficient Ascending-bid Auction for Multiple Objects," *The American Economic Review*, vol. 94, no. 5, pp. 1452-1475, Dec. 2004.