Improvement of Extended Least Difference Greedy Clique-Cover Algorithm for Index Coding

Joon-Woo Lee, Jae-Won Kim, Jong-Seon No
Department of Electrical and Computer Engineering
INMC, Seoul National University
Seoul, Korea
{joonwoo3511, kjw702}@ccl.snu.ac.kr, jsno@snu.ac.kr

***Abstract*—We modify the heuristic algorithm proposed by Kwak, *et al.*, called the extended least difference greedy (ELDG) algorithm, to make it more efficient. The column-merging heuristic algorithm, which is one of the essential parts in the ELDG algorithm, is found numerically not to be efficient for reducing the index codelength; besides, the result of the column-merging heuristic algorithm can be even longer than that of the original LDG algorithm. We detect when the column-merging heuristic algorithm give worse result than the result of the original LDG algorithm. Modified slightly, the proposed column-merging heuristic algorithm becomes efficient for reducing codelength. Additionally, we add the cycle-of-p-nodes detection algorithm, which is the generalized version of the cycle-of-three-node detection algorithm in the ELDG algorithm. It is numerically shown that roughly 20% of the index codelength reduction is achieved by the proposed ELDG algorithm compared to the LDG algorithm, while the conventional ELDG algorithm shows 10% of reduction rate compared to the LDG algorithm.**

***Index Terms*—Index code, least difference greedy clique-cover algorithm**

I. Introduction

The index coding problem, suggested by Birk and Kol [1], [2], has drawn attention of research community because of its application on various situation using broadcast channel, such as satellite networks, terrestrial wireless systems, distributed storage systems, etc. The index coding problem is defined as follows.

Definition 1. *Let $R = \{r_1, r_2, \cdots, r_n\}$ be n receivers communicating with a server. The server holds a set of n binary variables $X = \{x_1, x_2, \cdots, x_n\} \in \{0,1\}^n$. Each receiver r_1 wants x_i and has prior side information of the data X, denoted by $X[N(i)]$, where $N(i)$ is the index set of side information of r_i. The purpose of the index coding problem is to minimize the number of bits for given side information index sets $N(1), N(2), \cdots, N(n)$.*

Finding the minimum index codelength is an NP-hard problem. Thus the efficient algorithms finding approximated solutions are needed. The least difference greedy (LDG) clique-cover algorithm [1], [2] is one of these algorithms. The modified version of the LDG algorithm was suggested by Kwak, *et al.*, called the extended LDG (ELDG) algorithm [3]. The ELDG algorithm includes two algorithms, the column-merging heuristic algorithm and the cycle-of-three-nodes detection algorithm. The ELDG algorithm improves performance of the LDG algorithm by 10%.

In this work, we find the inefficiency of the column-merging heuristic algorithm. The overall 10% improvement of the ELDG algorithm actually comes from only the performance of the cycle-of-three-nodes detection algorithm so that this performance offsets and hides the worsened performance of the column-merging heuristic algorithm. We suggest the counterexample on which the column-merging heuristic algorithm gives worse result than that of the original LDG algorithm. We generalize the result of [5] and suggest systematic algorithms. Specifically, we improve the column-merging heuristic by slightly modifying it. Also, we generalize the cycle-of-three-nodes detection algorithm to the cycle-of-p-nodes detection algorithm. This modification improves performance of the original LDG algorithm by 20%.

This work was supported by the National Research Foundation of Korea (NRF) through the Korea government under Grant NRF-2016R1A2B2012960.

II. Preliminary

A. Index Coding

Bar-Yossef *et al.* [4] defined the fitting matrix of a side information graph and suggested a new notion *minrank*. If a directed graph $G = (V, E)$ with n nodes does not have any self-loop, an $n \times n$ matrix A over $\mathbb{F}_2$ is said to fit the graph G if $A_{ii} = 1$ for every $i \in \{1, 2, \cdots, n\}$ and $A_{ij} = 0$ for every i, j such that $(i, j) \notin E$. $minrk_2(G)$ is defined to be $\min\{rank_2(A) | A \text{ fits } G\}$. Bar-Yossef *et al.* found that the minimum codelength of the index code whose side information graph is G equals to $minrk_2(G)$.

B. LDG Algorithm

Since getting $minrk_2(G)$ is an NP-hard problem, Birk and Kol [1], [2] suggested the LDG algorithm where the near-optimal solution can be obtained. Since the elements A_{ij} such that $i \neq j, (i, j) \in E$ is not decided in the fitting matrix of G, we write the undetermined elements of the fitting matrix as star elements, denoted by *. The distance between elements in the fitting matrix is defined as follows [2]: $d(0,0) = d(1,1) = d(*,*) = 0, d(0,*) = d(1,*) = 1$ and $d(0,1) = \infty$.

The distance between rows is defined as the sum of the distances between elements of the same column in each row. These distances mean the number of the star elements to be

filled with 0 or 1 for making the rows exactly same and the infinity distance means that we cannot make the rows same. In the LDG algorithm, the distances between all possible pairs of the rows are computed and the star elements of the pair of rows having minimum distance are filled with 0 or 1. We iterate it until the distances of all rows are infinity.

C. ELDG Algorithm

Kwak *et al.* [3] suggested the ELDG algorithm to get more close solution to the optimal one. The ELDG algorithm consists of the column-merging heuristic algorithm and the cycle-of-three-nodes detection algorithm. The column-merging heuristic algorithm computes not only the distance of the rows but the distance of the columns and compares the distance of the rows and columns simultaneously. The cycle-of-three-nodes detection algorithm finds the cycle of three nodes in the side information graph. The 'XORed rows' of all two rows are computed and the distances between the added rows and each of the rest rows are computed. Then, we select the three rows having the minimum distance. The XOR operation between the elements is defined as follows: $a \oplus^* b = a \oplus b$ if $a, b \neq *$, or * if $a = *$ or $b = *$. The addition of the rows can be defined componentwisely. With the ELDG algorithm, roughly 10% of the codelength reduction can be achieved.

III. Analysis of ELDG Algorithm

Since Kwak *et al.* [3] only analyzed overall performance and did not analyze each of two algorithms in the ELDG algorithm, we analyzed each algorithm and find the inefficiency of the first algorithm, the column-merging heuristic algorithm, in reducing codelength.

The priority in the choice of the two rows of the minimum distance was not specified in the ELDG algorithm when there are many pairs of minimum distance. So, without loss of generality, if there are many row pairs or column pairs that has the minimum distance, row pairs are first chosen rather than column pairs, and the more leftside column pair or the more topside row pair is chosen first.

We generate the fitting matrix with probability p as follows: we fix the diagonal elements as 1, and put 0 or * in the non-diagonal elements randomly, where the probability of putting * is equal to p. We increased p from 0 to 1 by 0.1 and the number of nodes is 30 and 40 in each simulation respectively. For each p we generate 100 fitting matrices and average their solution of the algorithm. The reduction rate means how much the codelength was reduced when the ELDG algorithm was used, compared with when the original LDG algorithm was used. The positive reduction rate means the codelength of the index code was reduced. For example, if the codelength the original LDG algorithm outputs in an instance of 30 nodes is 20 and the codelength the ELDG algorithm outputs in the same instance is 15, the reduction rate of the case is $(20-18)/20 \times 100 = 10\%$.

Fig. 1 shows the performance of each algorithm in the ELDG algorithm and overall performance of the ELDG algorithm. We expect notable positive values at every probability, but in simulation result of the column-merging heuristic

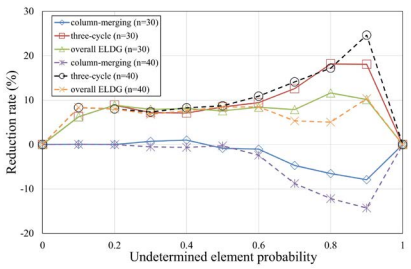

Fig. 1. Performance of each algorithm in the ELDG algorithm.

algorithm, less than 1% of reduction rate was shown when $0 < p < 0.5$, and the reduction rate was even decreased sharply when $0.5 < p < 1$. This means that the column-merging heuristic algorithm cannot improve effectively in small probability of star elements, and even worsens the performance in high probability of star elements. The overall reduction rate is indeed roughly 10% in Fig. 1, but it is easily seen that the column-merging heuristic algorithm cannot contribute to the performance of the ELDG algorithm. Hence, if we improve the column-merging algorithm, the overall performance of the ELDG algorithm will be increased.

We detect the counterexample on which the result of the column-merging heuristic algorithm is worse than the result of the originial LDG algorithm. Consider the following fitting matrix.

$$\begin{bmatrix} 1 & 0 & * & * \\ * & 1 & 0 & * \\ * & * & 1 & 0 \\ * & * & * & 1 \end{bmatrix}$$

If we perform the original LDG algorithm on this fitting matrix, we can get the 2 rank reduction. If we perform the column-merging heuristic algorithm, we can get only the 1 rank reduction. This can be the counterexample of the efficiency of the column-merging heuristic algorithm.

Also, we find that the cycle-of-three-nodes detection algorithm can be easily generalized to the similar algorithm which can detect cycles having more nodes. So, if we increase the number of nodes of cycles we will detect, we can improve the performance.

IV. Proposed ELDG Algorithm

At first, we modify the column-merging algorithm. The proposed ELDG algorithm compares the rank of the fitting matrix only merged by rows with the rank of the fitting matrix only merged by columns and select the fitting matrix having the lower rank. This can guarantee that the rank of the output fitting matrix of the proposed ELDG algorithm is less than of equal to that of the LDG algorithm. The details of the proposed column-merging heuristic algorithm are shown in Algorithm 1.

The modified algorithm is considerd to be the addition of the column version of LDG algorithm on the original LDG algorithm. Since the result of the algorithm is the smaller value between the result of the original LDG algorithm and the result

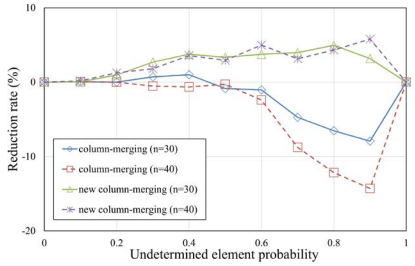

Fig. 2. Comparison of performance of the column-merging heuristic algorithm and the proposed column-merging heuristic algorithm.

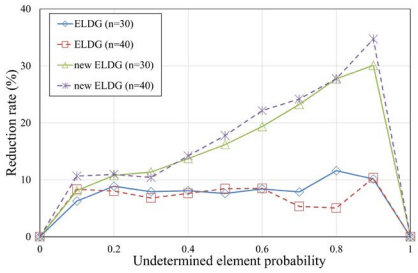

Fig. 3. Comparison of performance of the ELDG algorithm and the proposed ELDG algorithm with 30 nodes and 40 nodes.

of column version of it, the result is always less than the result of the original LDG algorithm. So this modification can guarantee the reduction of the codelength.

Next, we generalize the cycle-of-three-nodes detection algorithm to the cycle-of-p-nodes detection algorithm. This generalized algorithm XORs $p-1$ rows and computes the distances between the XORed rows and the rest rows. In the simulation, we add the cycle-of-four-nodes detection algorithm, as well as the cycle-of-three-nodes detection algorithm. The generalized algorithm is shown in Algorithm 2.

We simulate the proposed ELDG algorithm in which the column-merging heuristic algorithm is replaced by the proposed column-merging heuristic algorithm and the cycle-of-four-nodes detection algorithm is added. All values of reduction rate are drawn by comparing with the performance of the LDG algorithm. Fig. 2 shows that the first modification effectively improves the original column-merging algorithm in the ELDG algorithm. The reduction rate is positive at all probability of star elements in the proposed column merging algorithm, that is, it reduces the rank more stably than the column-merging heuristic algorithm. Especially, in high probability of star elements, we can see dramatic change in performace. Fig. 3 shows that the overall performance of the proposed the ELDG algorithm. While the ELDG algorithm takes roughly 10% reduction rate, the proposed ELDG algorithm takes roughly 20% reduction rate on average and takes 34% as maximum reduction rate.

V. Conclusion and Discussion

We found that the ELDG algorithm have some inefficiency, and we suggested the proposed ELDG algorithm. Then we showed the improved performance compared to the ELDG algorithm numerically. This result, however, was not based on theoretical analysis, so the performance improvement has to be proven mathematically for the future work.

Procedure 1 Proposed column-merging algorithm

Input: A square matrix $\mathbf{A} \in \{0,1,*\}^{n\times n}$ which represents a linear binary index coding problem.
$\mathbf{A}_1 \leftarrow \mathbf{A}, \mathbf{A}_2 \leftarrow \mathbf{A}$
repeat
Compute the distances of all pairs of rows in $\mathbf{A}_1$.
Choose two rows in $\mathbf{A}_1$ having minimum distance.
Put 0 or 1 on * of the two rows of $\mathbf{A}_1$ for making two rows same.
until distances of all pairs of rows in $\mathbf{A}_1$ are infinity.
repeat
Compute the distances of all pairs of columns in $\mathbf{A}_2$.
Choose two columns in $\mathbf{A}_2$ having minimum distance.
Put 0 or 1 on * of the two columns of $\mathbf{A}_2$ for making two columns same.
until distances of all pairs of columns in $\mathbf{A}_2$ are infinity.
return either $\mathbf{A}_1$ or $\mathbf{A}_2$ whose rank is less than the rank of the other.

Procedure 2 Cycle-of-p-nodes detection

Input: A square matrix $\mathbf{A} \in \{0,1,*\}^{n\times n}$ which represents a linear binary index coding problem.
for all in $\{(i_1, i_2, \cdots, i_{p-1}) \in \mathbb{Z}^{p-1} | 1 \le i_1 < i_2 < \cdots < i_{p-1} \le n\}$ **do**
$\mathbf{B} = \mathbf{A}_{i_1} \oplus^* \mathbf{A}_{i_2} \oplus^* \cdots \mathbf{A}_{i_{p-1}}$
repeat
Compute distance between $\mathbf{B}$ and $\mathbf{A}_j$ for all $j \in [i_{p-1}, n]$.
Choose m such that the distance between $\mathbf{B}$ and $\mathbf{A}_m$ is minimum distance among these distances.
Make $\mathbf{B}$ and $\mathbf{A}_m$ be same.
until all computed distances are 0 or ∞.
Make $\mathbf{A}_{i_1}, \mathbf{A}_{i_2}, \cdots, \mathbf{A}_{i_{p-1}}$ so that the sum is equal to $\mathbf{B}$.
end for

References

[1] Y. Birk and T. Kol, "Informed-source coding-on-demand (ISCOD) over broadcast channels." in *Proc. IEEE Conf. on Comput. Commun. (INFOCOM)*, 1998, pp.1257–1264.

[2] Y. Birk and T. Kol, "Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2825–2830, Jun. 2006.

[3] S. Kwak, J. So, and Y. Sung, "An extended least difference greedy clique-cover algorithm for index coding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2014, pp.501–505

[4] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol, "Index coding with side information," *IEEE Trans. Inf. Theory*, vol. 57, no. 3, pp. 1479–1494, Feb. 2011.

[5] J. Lee and J.-S. No, "Improvement of Extended Least Difference Greedy Clique-cover Algorithm for Index Code Optimization," in *Proc. KICS Winter General Conference*, 2018, pp.1185-1186.