数理解析研究所講究録
第1562巻 2007年 99-103

Labelings for the Complete Bipartite Graph and its Applications

Tomoko Adachi

Department of Information Sciences, Toho University,
2-2-1 Miyama, Funabashi 274-8510, Japan
E-mail: adachi@is.sci.toho-u.ac.jp

Abstract. The desire to speed up secondary storage systems has lead to the development of redundant arrays of independent disks (RAID) which incorporate redundancy utilizing erasure codes. A 'cluttered ordering' is utilized for designing an effective writing order to a RAID system. Cohen, Colbourn and Froncek (2001) gave a cyclic construction of cluttered orderings for the complete graph by utilizing the notion of a 'wrapped ρ-labeling.' Müller, Adachi and Jimbo (2005) adapted the concept of wrapped Δ-labelings to the bipartite case instead of wrapped ρ-labelings. In this paper, we investigate more generalized cases of wrapped Δ-labelings leading to cluttered orderings for the corresponding bipartite graphs.
Keywords: ordering, RAID, disk arrays

1 Introduction

The desire to speed up secondary storage systems has lead to the development of *disk arrays* which achieve performance through disk parallelism. While performance improves with increasing numbers of disks the chance of data loss coming from *catastrophic failures*, such as head crashes and failures of the disk controller electronics, also increases. To avoid high rates of data loss in large disk arrays one includes redundant information stored on additional disks – also called *check disks* – which allows the reconstruction of the original data – stored on the so-called *information disks* – even in the presence of disk failures. These disk array architectures are known as *redundant arrays of independent disks* (RAID) (see [3]).

Hellerstein et. al. [7] introduced data redundancy in form of erasure-resilient codes. The connection between these codes and combinatorial design

theory also explored by Chee et al. [2]. For details we refer to this literature and give in Section 2 only a short summary of the coding theoretic background. Minimizing the number of disk operations when writing to consecutive disks leads to the concept of (d, f)-cluttered orderings. Cluttered orderings were introduced and studied by Cohen et. al. for Steiner triple systems (see [4]) and the complete graph (see [5, 6]). In case of the complete graph, Cohen et. al. [5] constructed for the parameters $d = 3$ and $d = 4$ optimal cluttered orderings, also called a *ladder ordering* in this case, for all complete graphs K_n, $n \in \mathbb{N}$, except for the parameters $n \in \{15, 18, 22\}$. Furthermore, they also introduce some general construction principle of cluttered orderings for the complete graph based on wrapped ρ-labelings of certain subgraphs (see [6]). Müller et. al. [8] adapted more generalized infinite families of wrapped Δ-labelings leading to cluttered orderings for the corresponding bipartite graphs. Adachi [1] gave cluttered orderings for the complete tripartite graph $K_{9,9,9}$.

2 Applications

Erasure-correcting capability of disk arrays can be achieved by employing *erasure codes*. Let $k, c \in \mathbb{N}$ and let GF(2) denote the field with two elements. Then an erasure code is defined by a linear injection $\gamma : \mathrm{GF}(2)^k \to \mathrm{GF}(2)^{k+c}$ such that an information $x \in \mathrm{GF}(2)^k$ appears unchanged in the first k bits – the so-called *information bits* – of the corresponding code vector $\gamma(x)$. The remaining c bits are referred to as *check bits* which can be computed as the parity of subsets of information bits. Each such code can be defined in terms of a $c \times (k + c)$*-parity check matrix*, $H = [C|I]$, where I denotes the $c \times c$ identity matrix and C is a $c \times k$ matrix. The codewords in the code are the vectors $y \in \mathrm{GF}(2)^{k+c}$ satisfying the equation $Hy = 0$. Note, that the first k columns of H correspond to the information bits and the last c columns to the check bits (see also Figure 1).

An unreadable bit of a code vector is called an *erasure*. It is a well known fact that a code can correct a set of t erasures iff the corresponding t columns of H are linearly independent considered as vectors over GF(2), the Galois field with two elements (see [7]). An erasure code which can correct any t erasures will be abbreviated as t-EC. In view of the RAID-application there are the following two important metrics in erasure codes. One metric is the *update penalty*, which is the number of check disks whose contents must be changed if an information disk is changed. In terms of the matrix H it can be defined as the maximum over the weights of the columns of H. It follows easily that the update penalty of an t-EC is at least t. Another metric is the *check bit overhead*, which is the ratio c/k of the number of check bits to information bits. Good erasure codes have high erasure correcting capabilities, whereas

the update penalty as well as the check bit overhead should be low. (See [7] for further details.)

Next, let $k = \ell^2$ for some $\ell \in \mathbb{N}$, so that the k information bits can be arranged in a two dimensional array. Associate to each row and each column a check bit containing the parity of that row or column, i.e., $c = 2\ell$. The so defined code is called 2-*dimensional parity code* which is also a 2-EC with update penalty 2 and check bit overhead $\frac{c}{k} = \frac{4}{c}$. In [7] is proved that this code can even correct most of the 3-erasures as well (all 3-erasures except of the so-called bad 3-erasures) and has – with respect to this even higher erasure-correcting capability – optimal check disk overhead among all such codes. As illustration we present in Figure 1 the case $\ell = 2$. For example, information disk 1 is associated to the check disks a and c.

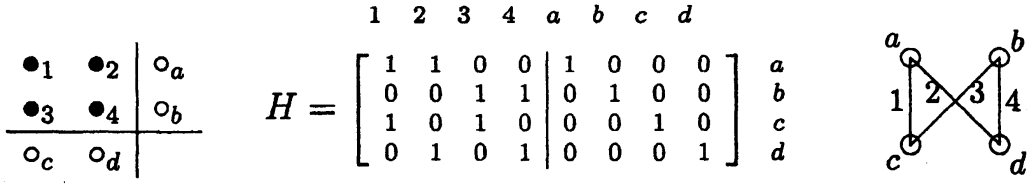

Figure 1: 2-dimensional parity code for $\ell = 2$, its parity check matrix H, and the corresponding complete bipartite graph $K_{2,2}$.

3 Cluttered Orderings

In a RAID system disk writes are expensive operations and should therefore be minimized. When writing on a single information disk one also has to recompute the parity information and change the contents of all check disks involved. This overhead is expressed by the update penalty as defined in Section 2. In many applications there are writes on a small fraction of consecutive disks – say d disks – where d is small in comparison to k, the number of information disks. In this case a write can be implemented as an efficient *read-modify-write* which can be described as follows [4]. First the d information disks are read followed by all of their associated check disks. In the case when check disks overlap, the physical read only takes place once. All of the new parity is computed and then this new parity and the new information is written back to the disks. Once again, the shared check disks are only physically written once. Therefore, to minimize the number of operations when writing to d consecutive information disks one has to minimize the number of check disks – say f – associated to the d information disks. In other words, the order of the information disks – or the order of the corresponding columns

of the parity check matrix – plays a crucial role for the efficiency of the RAID system. In terms of set systems this leads to the following definition.

Definition 1 Let $\mathcal{S} = (X, \mathcal{B})$ denote a set system and $\mathcal{B} = \{B_1, B_2, \ldots B_k\}$, $k \in \mathbb{N}$. Furthermore, let d, $1 \leq d \leq k$, be a positive integer, called *window* of $\mathcal{S}$, and π be a permutation on $\{0, 1, \cdots, k-1\}$, called *block ordering* of $\mathcal{S}$. Define $X_i^{\pi,d} := \bigcup_{j=0}^{d-1} B_{\pi(i+j)}$ for $0 \leq i \leq k-1$, where indices are considered modulo k. For some given $f \in \mathbb{N}$, π is called a *(cyclic) (d, f)-cluttered ordering* if $\max_i |X_i^{\pi,d}| \leq f$.

In view of the RAID-application one is interested in minimizing the parameter f for some given d. Let $f_{\min}(\mathcal{S}, d)$ denote the minimum value of f over all block orderings π of $\mathcal{S}$ for some fixed d and, similarly, $d_{\max}(\mathcal{S}, f)$, the maximum value of d for some fixed f. Typical questions are, for example, to find lower and upper bound for $f_{\min}(\mathcal{S}, d)$ and $d_{\max}(\mathcal{S}, f)$. Furthermore, one is interested in explicit constructions of such orderings and in asymptotic existence theorems for certain subclasses of set systems.

Cluttered orderings were introduced and studied by Colbourn et. al. for Steiner triple systems (see [4]) and the complete graph (see [5, 6]). In the latter case they give a construction of certain cluttered orderings presupposing the existence of *wrapped ρ-labelings*. Müller et. al. [8] adapted the concept of wrapped Δ-labelings to the bipartite case instead of wrapped ρ-labelings, and gave the explicit construction of several infinite families of wrapped Δ-labelings.

Here, we investigate more generalized infinite families of wrapped Δ-labelings leading to cluttered orderings for the corresponding bipartite graphs.

References

[1] Adachi, T.: Optimal ordering for the complete tripartite graph $K_{9,9,9}$. *Proceedings of the Fourth International Conference on Nonlinear Analysis and Convex Analysis*. Yokohama Publishers, 1–10, 2007.

[2] Chee, Y. M., Colbourn, C. J., Ling, A.: Asymptotically optimal erasure-resilient codes for large disk arrays. *Discrete Applied Mathematics* **102** (2000), 3–36.

[3] Chen, P., Lee, E., Gibson, G., Katz, R., Patterson, D.: RAID: High-performance, reliable secondary storage. *ACM Computing Surveys* **26** (1994), 145–185.

[4] Cohen, M., Colbourn, C.: Optimal and Pessimal Orderings of Steiner Triple Systems in Disk Arrays. *Theoretical Computer Science* **297** (2003), 103–117.

[5] Cohen, M., Colbourn, C.: Ladder orderings of pairs and RAID performance. *Discrete Applied Mathematics* **138** (2004), 35–46.

[6] Cohen, M., Colbourn, C., Froncek, D.: Cluttered orderings for the complete graph. *COCOON 2001: Lect. Notes Comp. Sci.* **2108** (2001), Springer, 420–431.

[7] Hellerstein, L., Gibson, G., Karp, R., Katz, R., Patterson, D.: Coding techniques for handling failures in large disk arrays. *Algorithmica* **12** (1994), 182–208.

[8] Müller, M., Adachi, T., Jimbo, M.: Cluttered orderings for the complete bipartite graph. *Discrete Applied Mathematics* **152** (2005), 213–228.