Ternary Graphical Codes *

Dieter Jungnickel
Lehrstuhl für Angewandte Mathematik II
Universität Augsburg
D-86135 Augsburg
Germany

Scott A. Vanstone
Department of Combinatorics and Optimization
University of Waterloo
Waterloo, Ont., N2L 3G1
Canada

ABSTRACT. It is well-known that the set of all circulations of a connected digraph G on p vertices with q edges forms a ternary linear code $\mathbf{C} = \mathbf{C}_E(G)$ with parameters $[q, q-p+1, g]$, where g is the girth of G. Such codes were first studied by Hakimi and Bredeson [8] in 1969 who were concerned with the problems of augmenting $\mathbf{C}$ to a larger $[q, k, g]$-code and of efficiently decoding such codes; their treatment is similar to their previous work on binary codes [4, 7]. Recently, we have obtained considerable progress in the binary case by generalizing Hakimi's and Bredeson's construction method to obtain better augmenting codes and by giving a much more efficient decoding algorithm. In the present paper, we shall investigate how far our methods can be adapted to obtain corresponding progress in the ternary case. In particular, we will correct an oversight in a graph theoretic lemma of Bredeson and Hakimi which also affects their decoding algorithms and discuss the possibility of alternative decoding procedures based on a connection to an apparently difficult optimization problem.

*This paper was written while the first author was visiting the Department of Combinatorics and Optimization of the University of Waterloo as an Adjunct Professor. He would like to thank his colleagues there for their hospitality. The second author acknowledges the support of National Science and Engineering Research Council of Canada given under grant # 0GP0009258.

1 Introduction

In this paper we explore the relationship between directed graphs and ternary codes. A considerable amount of work on the connections between graph theory and coding theory has been done previously, cf. [9, 6, 4, 7, 8, 1, 2, 20, 21, 23-26, 13, 12, 14, 16, 17]. Most previous work concerns the binary case: As is well-known, the set of all even subgraphs of a connected graph G on p vertices with q edges forms a binary linear code $\mathbf{C} = \mathbf{C}_E(G)$ with parameters $[q, q-p+1, g]$, where g is the girth of G; here an *even subgraph* of G is a spanning subgraph in which each vertex has even degree. [1] We shall hereafter refer to $\mathbf{C}_E(G)$ as the *even graphical code* of G; these codes have first been systematically studied by Bredeson and Hakimi [6, 4, 7]. These authors focused on two main questions, namely the possibility of augmenting an even graphical code to a code of larger dimension while keeping the minimum distance unchanged and of efficiently decoding both the original and the augmented codes. Subsequently, Hakimi and Bredeson [8] did a similar study of the ternary case, replacing G by a connected (but not necessarily strongly connected) digraph and the even subgraphs of G by the (ternary) circulations on that digraph. Recently, we [17] have obtained considerable progress in the binary case by generalizing Hakimi's and Bredeson's construction method to obtain better augmenting codes and by giving a much more efficient decoding algorithm. In the present paper, we shall investigate how far our methods can be adapted to obtain corresponding progress in the ternary case. This turns out to work quite nicely for the augmentation problem, but seems to be rather difficult as far as decoding the circulation code is concerned, where we will only be able to obtain a connection to an apparently difficult optimization problem. Decoding augmented codes then turns out to be easy again, given a decoder for the circulation codes. Finally, we will also point out an oversight in a graph theoretic lemma of Bredeson and Hakimi [4] which also affects their decoding algorithms given in [7, 8] and present corrected versions of their lemma and their decoding procedure for the ternary case.

The basic result on ternary graphical codes [8] is as follows. Here we denote the ternary vector space of all circulations of a digraph $G = (V, E)$ by $\mathbf{C}(G)$; alternatively, after choosing a fixed labelling of the arcs of G and identifying functions $f: E \to \mathbf{Z}_3$ with ternary vectors correspondingly, $\mathbf{C}(G)$ may be viewed as the kernel of the incidence matrix M of G, i.e., as the orthogonal complement of the row space of M; see, for instance, [3]. For notational convenience, we will write the elements of $\mathbf{Z}_3$ as 0, 1 and -1. We shall also use the notation $|G|$ for the underlying graph of G, which is

[1] For background and definition from graph theory and combinatorial optimization, the reader is referred to Bondy and Murty [3], Harary [10], Jungnickel [11], Lovász and Plummer [18] and Papadimitriou and Steiglitz [22], and for coding theory to MacWilliams and Sloane [19] and van Lint [27].

obtained by ignoring the direction of the arcs of G.

Proposition 1.1. *Let G be a connected digraph with q edges on p vertices, and let g be the girth of $|G|$. Then $\mathbf{C}(G)$ is a ternary $[q, q-p+1, g]$-code.* □

We note that the assertion on the dimension of $\mathbf{C}(G)$ is well-known; the simple proof concerning the minimum weight can be found in [8]. We shall call this code the *circulation code* of G, since its elements may be most naturally viewed as "circulations". Of course, one needs to bear in mind that we do consider ternary functions so that a circulation in our code is, in general, not a circulation when considered as an integral function. To give the reader some feeling for this situation, we will now give a direct proof for the following simple result concerning the relationship between the circulation code $\mathbf{C}(G)$ and the binary code $\mathbf{C}_E(|G|)$ of the underlying graph.

Lemma 1.2. *Let G be an arbitrary digraph. Then any element of the binary code $\mathbf{C}_E(|G|)$ can be obtained from a suitable circulation $\mathbf{c}$ in the ternary code $\mathbf{C}(G)$ by ignoring signs, i.e., by replacing $\mathbf{c}$ with its support $|\mathbf{c}|$ in $|G|$.*

Proof: Consider any element of $\mathbf{C}_E(|G|)$, viewed as an even subgraph H of $|G|$; hence H splits into the disjoint union of Eulerian subgraphs on its connected components. It suffices to consider one such component and to show that – viewed as a binary vector – it may be obtained as the image $|\mathbf{c}|$ of some circulation $\mathbf{c}$; to simplify notation, we may as well assume that H is itself connected. Let us choose a Eulerian cycle C for H and orient it arbitrarily. Now a standard argument gives the desired circulation $\mathbf{c}$ on G: Put $\mathbf{c}(e) = 1$ if the orientation of an arc $e \in C$ agrees with the orientation of e in G, and $\mathbf{c}(e) = -1$ otherwise (and, of course, $\mathbf{c}(e) = 0$ for all arcs e not occurring in C). □

The reader should note that any *integral* circulation has as its support a Eulerian subgraph; in this case, taking supports would just map the integral circulation space of G on the even graphical code of $|G|$. This is, however, no longer true in the *ternary* case: Assume, for instance, that G contains two vertices s and t which may are joined by three edge-disjoint paths; then we can easily define a circulation on the support of these three paths which clearly is not an even subgraph of $|G|$. Nevertheless, one may in a certain sense think of $\mathbf{C}(G)$ as obtained by suitably "orienting" $\mathbf{C}_E(|G|)$. To be more precise, let us fix a spanning tree T for $|G|$; as is well-known, each edge $e \notin T$ defines a unique cycle of $|G|$. The (characteristic vectors of) these $q - p + 1$ cycles form a basis for the cycle space of $|G|$. Recording them as the rows of a matrix $B(|G|)$ thus gives a generator matrix for the code $\mathbf{C}_E(|G|)$ which is called the *fundamental circuit matrix* of G in [7]. Associating with each of these fundamental cycles of $|G|$ a circulation as described in the proof of Lemma 1.2 then gives us a generator matrix $B(G)$

for the ternary code $\mathbf{C}(G)$ which may be thought of as an orientation of $B(|G|)$. This explains how the binary code $\mathbf{C}_E(|G|)$ gives rise to a large number of ternary codes with the same parameters, corresponding to the possible orientations of G to a digraph. Note, however, that these codes are not essentially different, since they are all monomially equivalent.

2 Augmenting Circulation Codes

In this section we will consider the problem of augmenting the circulation code of a connected digraph $G = (V, E)$. This turns out to be similar to the study of the possible augmentations of the binary code $\mathbf{C}_E(|G|)$ of the underlying graph by adjoining further subgraphs as discussed in our paper [17]. Note that studying the "odd degree patterns" of subgraphs (which was crucial in the binary case) is no longer useful, since the odd degree pattern of the sum of two ternary vectors is in general different from the binary sum of the odd degree patterns of the two underlying subgraphs involved. Accordingly, augmentation requires ternary codes as auxiliary codes instead of even binary codes, and the proofs need a different line of argumentation. For the convenience of the reader, we will first summarize what can be done in the binary case.

Result 2.1: Consider an even graphical code $\mathbf{C} = \mathbf{C}_E(|G|)$ with parameters $[q, q-p+1, g]$ based on the connected graph $|G|$. Then $\mathbf{C}$ may be extended to a graphical code $\mathbf{C}^*$ with parameters $[q, q-p+1+k, g]$ provided that there exists an even binary $[p, k, 2g]$-code $\mathbf{O}$. One may obtain such a code $\mathbf{C}^*$ by adjoining to $\mathbf{C}$ any k linearly independent subgraphs $S_1, \ldots, S_k$ of $|G|$ with odd degree patterns forming a basis for $\mathbf{O}$ as further generators. □

Result 2.2: Consider an even graphical code $\mathbf{C} = \mathbf{C}_E(|G|)$ with parameters $[q, q-p+1, g]$ based on the connected graph $|G|$, and assume that $V_1, \ldots, V_c$ is a partition of the vertex set V into independent sets with cardinalities $p_1, \ldots, p_c$, respectively. Then $\mathbf{C}$ may be extended to a graphical code $\mathbf{C}^*$ with parameters $[q, (q-p+1)+(k_1+\cdots+k_c), g]$ provided that there exist even binary $[p_i, k_i, g]$-codes $\mathbf{O}_i$ for $i = 1, \ldots, c$. One may obtain such a code $\mathbf{C}^*$ by adjoining to $\mathbf{C}$ (for $i = 1, \ldots, c$) arbitrary sets of k_i linearly independent subgraphs $S_1^{(i)}, \ldots, S_{k_i}^{(i)}$ of $|G|$ with odd degree patterns contained in V_i and forming a basis for $\mathbf{O}_i$ (when considered as binary vectors of length p_i indexed with the vertices in V_i) as further generators. □

In order to obtain analogous results in the ternary case, one has to replace the notion of the parity of a vertex in a subgraph with that of the "excess" of a ternary function (or vector) at a vertex which we will now define in analogy to a standard notion from the theory of network flows. Given an arbitrary ternary vector $\mathbf{x}$, we call the scalar product of the kth row of the incidence matrix M of G with $\mathbf{x}$ the *excess* of $\mathbf{x}$ at the corresponding vertex

v_k. Thus x is a circulation if and only if it has excess 0 at each vertex of G. Again, we stress that all computations are done modulo 3, and hence the only possible non-zero excesses are ± 1. We first prove the following analog of Result 2.2.

Theorem 2.3. *Let G be a connected digraph of girth g with q edges on p vertices and assume the existence of a ternary $[p-1, k, 2g-1]$-code $\mathbf{A}$. Then the circulation code $\mathbf{C} = \mathbf{C}(G)$ with parameters $[q, q-p+1, g]$ may be extended to a ternary $[q, q-p+1+k, g]$-code $\mathbf{C}^*$.*

Proof: We shall use the auxiliary code $\mathbf{A}$ to produce a ternary $[q, k, g]$-code $\mathbf{C}'$ such that $\mathbf{C} + \mathbf{C}'$ results in the desired code $\mathbf{C}^*$. To this end, we first select a fixed vertex s of G and conduct a BFS-search on $|G|$ with start vertex s; let us denote the resulting spanning tree of $|G|$ by T. Given any vertex v in G, we have a unique path P_v in T joining s with v which we will consider as directed from s to v. With each such path, we associate a ternary vector $\mathbf{p}_v$ as follows: The e-coordinate $p_v(e) \neq 0$ if and only if e occurs is P_v; moreover, $p_v(e) = +1$ if e is a forward edge in P_v, i.e. if the orientation of e in P_v agrees with that of e in G, and $p_v(e) = -1$ if e is a backward edge in P_v. Note that we may choose a bijection α between the the set of vertices of G different from s and the coordinates of $\mathbf{A}$; in this way, we avoid the possibility of using the trivial path P_s in what follows. Given any non-zero vector $\mathbf{a} \in \mathbf{A}$, consider the vertices v which correspond to coordinates $\alpha(v)$ with a-entry $a(v) \neq 0$ and put $\mathbf{c}_{v,\mathbf{a}} = \mathbf{p}_v$ if $a(v) = 1$ and $\mathbf{c}_{v,\mathbf{a}} = -\mathbf{p}_v$ if $a(v) = -1$. Finally, denote by $\mathbf{c}_\mathbf{a}$ the sum of all the vectors $\mathbf{c}_{v,\mathbf{a}}$ just defined. In this way, any non-zero $\mathbf{a} \in \mathbf{A}$ gives rise to a ternary vector $\mathbf{c}_\mathbf{a}$ of length q; we choose $\mathbf{C}'$ as the vector space consisting all these vectors $\mathbf{c}_\mathbf{a}$ together with $\mathbf{c}_\mathbf{0} = \mathbf{0}$. Intuitively, we may view each element $\mathbf{c}_\mathbf{a}$ of $\mathbf{C}'$ as an "oriented sum" of a system $\mathbf{P}_\mathbf{a}$ of $w_\mathbf{a}$ paths in T, where $w_\mathbf{a}$ denotes the weight of a. In order to see that $\mathbf{C}^*$ is indeed a $[q, q-p+1+k, g]$-code it therefore suffices to show that the distance between any oriented sum of m paths in T and any circulation $\mathbf{c}$ always is at least $\lceil m/2 \rceil$ and then to use the hypothesis that $\mathbf{A}$ has minimum weight $2g-1$. To see this, we use the notion of excess defined above. Note first that each vector $\mathbf{p}_v$ has excess ± 1 at both s and v and excess 0 for any internal vertex. Since the excess of a linear combination of ternary vectors at a vertex v obviously equals the corresponding linear combination of the individual excesses at v, any oriented sum s of m paths P_v in T has excess ± 1 at each of the associated m end vertices v. Hence one needs to change the values of at least $\lceil m/2 \rceil$ coordinates of s to make s into a circulation which proves the assertion. □

We remark that it is indeed not possible to prove Theorem 2.3 by arguing about the number of odd vertices of an oriented sum s of m paths in T: It is not difficult to find examples showing that the support of such a sum

may have fewer than m vertices of odd degree. It is also easy to construct examples where s has weight exactly $\lceil m/2 \rceil$, so that the construction given in the proof of Theorem 3.3 is in general best possible. However, as in the binary case, one may often do better if one has more information about the structure of G, more precisely on its chromatic number. As the construction is similar to the one given in Theorem 2.3, the details of the proof may be left to the reader.

Theorem 2.4. *Let G be a connected digraph of girth g with q edges on p vertices, and assume that $V_1, \ldots, V_c$ is a partition of the vertex set V into independent sets with cardinalities $p_1, \ldots, p_c$, respectively. Then the circulation code $\mathbf{C} = \mathbf{C}(G)$ with parameters $[q, q-p+1, g]$ may be extended to a ternary code $\mathbf{C}^*$ with parameters $[q, (q-p+1)+(k_1+\cdots+k_c), g]$ provided that there exist ternary $[p_i, k_i, g]$-codes $\mathbf{A_i}$ for $i = 1, \ldots, c-1$ and a ternary $[p_c - 1, k_c, g]$-code $\mathbf{A_c}$.*

Proof: Similar to the construction given in the proof of Theorem 2.3, we choose a fixed vertex $s \in V_c$ and construct a BFS-tree T for $|G|$ with root s. We now use the auxiliary codes $\mathbf{A_i}$ exactly as in the proof of Theorem 2.3 to construct oriented sums of paths in T resulting in ternary $[q, k_i, g]$-codes $\mathbf{C_i}$ for $i = 1, \ldots, c$; in the case $i = c$, we identify the coordinate positions of $\mathbf{A_c}$ with the $p_c - 1$ vertices different from s in V_c (again in order to avoid the possibility of using the trivial path P_s). Now define $\mathbf{C}^*$ to be the code $\mathbf{C}+\mathbf{C_1}+\cdots+\mathbf{C_c}$. We again use the fact that any oriented sum s of m paths P_v in T has excess ± 1 at each of the associated m end vertices v. Note that any circulation and any non-zero vector in one the codes $\mathbf{C_i}$ have distance at least g, since the corresponding end vertices v form an independent set; hence changing their excesses to 0 requires at least g coordinate changes. Any linear combination of non-zero vectors from at least two of the codes $\mathbf{C_i}$ is an oriented sum of at least $2g$ paths in T and thus also has distance at least g from any circulation. □

We have shown in [17], that Results 2.1 and 2.2 generalize the construction methods of Hakimi and Bredeson [7] for binary codes, even though this is not immediately obvious. In a similar way, one can also show that Theorem 2.4 generalizes the corresponding ternary construction given in [8]; no ternary analog of Result 2.1 was stated there explicitly, though the possibility of such a result was mentioned. In fact the results of [8] cover exactly the special case of our constructions where the auxiliary ternary codes all have to be chosen as circulation codes. Hence, as in the binary case, we will in general be able to obtain far better augmenting codes. We also note that Hakimi and Bredeson have not supplied any proof for their constructions but just stated that an "appropriate modification of reasoning" for the binary case should be used. We feel that it is after all not totally obvious that this appropriate modification consists in replacing the

use of odd vertex patterns with that of "excess patterns" and that at least some sort of a hint might have been provided to the reader.

3 Decoding Augmented Circulation Codes

In this section, we shall consider the problem of decoding a general graphical code $\mathbf{C}^*$ based on a graph G provided that we have an efficient decoding algorithms for both the circulation code and also the auxiliary ternary code(s) used in the constructions of Section 2. We shall now show that the method proposed in [8] can be adapted to handle our more general augmentations. Since we want to be able to correct up to t errors using $\mathbf{C}^*$, we must first recognize whether or not a received word $\mathbf{r}$ actually belongs to a code word in $\mathbf{C} = \mathbf{C}(G)$. In view of the proofs of Theorems 2.3 and 2.4, it seems advisable to try and use the excess pattern of $\mathbf{r}$. In what follows, we shall denote the e-coordinate of a ternary vector $\mathbf{x}$ by $x(e)$. We begin with an algorithm decoding a code constructed as in Theorem 2.3.

Algorithm 3.1: Consider a ternary graphical code $\mathbf{C}^*$ with parameters $[q, q-p+1+k, g]$ obtained by using the ternary $[p-1, k, 2g-1]$-code $\mathbf{A}$ as auxiliary code from the code $\mathbf{C} = \mathbf{C}(G)$ with parameters $[q, q-p+1, g]$ based on the connected digraph G, where $g \geq 2t+1$. Let $\mathbf{r}$ be a word received and assume that at most t errors have occurred, i.e. $\mathbf{r}$ has the form $\mathbf{r} = \mathbf{c} + \mathbf{c_a} + \mathbf{f}$ for some arbitrary circulation $\mathbf{c}$, some unknown vector $\mathbf{c_a} \in \mathbf{C}'$ associated with the auxiliary vector $\mathbf{a} \in \mathbf{A}$ and some unknown error vector $\mathbf{f}$ with support $|\mathbf{f}|$ consisting of at most t edges. In what follows, α denotes the bijection introduced in the proof of Theorem 2.3.

(1) Compute the excess of $\mathbf{r}$ at all vertices.

(2) Define a ternary vector $\mathbf{a}'$ by choosing $a'(\alpha(v))$ as minus the excess of $\mathbf{r}$ at v.

(3) Using the code $\mathbf{A}$, decode $\mathbf{a}'$ into a code word $\mathbf{a} = \mathbf{a}' + \mathbf{f}' \in \mathbf{A}$ and compute the associated vector $\mathbf{c_a} \in \mathbf{C}'$.

(4) Put $\mathbf{s} = \mathbf{r} - \mathbf{c_a}$ and decode $\mathbf{s}$ into a circulation $\mathbf{c} = \mathbf{s} - \mathbf{f} \in \mathbf{C}$.

(5) Output $\mathbf{x} = \mathbf{c} + \mathbf{c_a}$.

Theorem 3.2. *Algorithm 3.1 correctly decodes the ternary graphical code* $\mathbf{C}^*$ *in* $O(q) + O(\mathbf{C}) + O(\mathbf{A})$ *steps, where* $O(\mathbf{X})$ *denotes the complexity of decoding* $\mathbf{X}$.

Proof: We first show that the proposed decoding procedure is correct. Note that the excess pattern of a vector $\mathbf{x} = \mathbf{c} + \mathbf{c_a} \in \mathbf{C}^*$ can be used to reconstruct the auxiliary vector $\mathbf{a}$ involved as follows: Any path $\mathbf{p}_v$ in the

BFS-tree T has excess -1 at v (and $+1$ at s), whereas the corresponding "reversed" path $-\mathbf{p}_v$ has excess $+1$ at v (and -1 at s). Hence the $\alpha(v)$-coordinate of $\mathbf{a}$ can be recovered (since $v \neq s$) as minus the excess of $\mathbf{x}$ at v.

By hypothesis, the error pattern $|\mathbf{f}|$ consists of at most t edges which obviously can corrupt at most $2t$ excesses. Hence the received excess pattern involves at most $g-1$ vertices which have incorrect excess, i.e., their excess for $\mathbf{r}$ is different from their excess for $\mathbf{c}+\mathbf{c_a}$. The previous arguments now show that the vector $\mathbf{a}'$ defined in Step (2) differs from the correct auxiliary vector $\mathbf{a}$ in at most $g-1$ coordinates. Since $\mathbf{A}$ is a $(g-1)$-error correcting code, we may indeed decode $\mathbf{a}'$ correctly. Following this, we compute the associated vector $\mathbf{c_a} \in \mathbf{C}'$ and subtract it from the received word $\mathbf{r}$. Note that $\mathbf{s} = \mathbf{r} - \mathbf{c_a} = \mathbf{c} + \mathbf{f}$, and thus $\mathbf{s}$ belongs to a circulation $\mathbf{c}$ corrupted by an error vector of weight $\leq t$. By Proposition 1.1, we may decode $\mathbf{s}$ uniquely, and thus Step (5) will give the correct result $\mathbf{c} + \mathbf{c_a} = \mathbf{r} - \mathbf{f}$.

The assertion on the complexity is clear, since we may compute the excesses of all vertices in $O(q)$ steps if we use adjacency lists: The excess of a vector $\mathbf{u}$ at v is the sum of the values $u(e)$ over all arcs with tail v minus the sum of the values $u(e)$ over all arcs with head v. □

Of course, the quality of Algorithm 3.2 depends on both the quality of the decoding algorithm available for decoding the auxiliary code $\mathbf{A}$ used in constructing $\mathbf{C}^*$ and also on the decoding algorithm for the circulation code $\mathbf{C}$ itself. The latter problem will be discussed in Section 4. Regarding the choice of $\mathbf{A}$, we might for instance use a smaller circulation code or a ternary BCH-code.

We now give an analogous procedure for decoding ternary codes constructed as in Theorem 2.4.

Algorithm 3.3: Consider a ternary graphical code $\mathbf{C}^*$ (obtained as in Theorem 2.4) with parameters $[q, (q-p+1)+(k_1+\cdots+k_c), g]$ by using ternary $[p_i, k_i, g]$-codes $\mathbf{A_i}$ for $i = 1, \ldots, c-1$ and a ternary $[p_c-1, k_c, g]$-code $\mathbf{A_c}$ as auxiliary codes for the circulation code $\mathbf{C} = \mathbf{C}(G)$ with parameters $[q, q-p+1, g]$ based on the connected digraph G. Put $t = \lfloor (g-1)/2 \rfloor$, let $\mathbf{r}$ be a word received and assume that at most t errors have occurred. Thus $\mathbf{r}$ has the form

$$\mathbf{r} = \mathbf{c} + \mathbf{c_{a_1}} + \cdots + \mathbf{c_{a_c}} + \mathbf{f}$$

for some arbitrary circulation $\mathbf{c}$ on G, some unknown vectors $\mathbf{c_{a_i}} \in \mathbf{C_i}$ associated with the auxiliary vector $\mathbf{a_i} \in \mathbf{A_i}$ $(i = 1, \ldots, c)$ and some unknown error vector $\mathbf{f}$ with support $|\mathbf{f}|$ consisting of at most t edges.

(1) Compute the excess of $\mathbf{r}$ at all vertices.

(2) For $i = 1, \ldots, c$ define a ternary vector $\mathbf{a_i'}$ by choosing $a_i'(\alpha(v))$ as minus the excess of $\mathbf{r}$ at v for all $v \in V_i$, $v \neq s$.

(3) Using the code $\mathbf{A_i}$, decode $\mathbf{a_i'}$ into the correct auxiliary vector $\mathbf{a_i} \in \mathbf{A_i}$ and compute the associated vectors $\mathbf{c_{a_i}} \in \mathbf{C_i}$ (for $i = 1, \ldots, c$).

(4) Put $\mathbf{s} = \mathbf{r} - (\mathbf{c_{a_1}} + \cdots + \mathbf{c_{a_c}})$ and decode $\mathbf{s}$ into a circulation $\mathbf{c} = \mathbf{s} - \mathbf{f} \in \mathbf{C}$.

(5) Output $\mathbf{x} = \mathbf{c} + \mathbf{c_{a_1}} + \cdots + \mathbf{c_{a_c}}$.

One then obtains the following result by arguing as in the proof of Theorem 3.3; the details are left to the reader.

Theorem 3.4. *Algorithm 3.4 correctly decodes the ternary graphical code* $\mathbf{C^*}$ *with complexity* $O(q) + O(\mathbf{C}) + O(\mathbf{A_1}) + \cdots + O(\mathbf{A_c})$. □

4 Decoding Circulation Codes

In this section we discuss the problem of efficiently decoding a circulation code $\mathbf{C} = \mathbf{C}(G)$ based on a connected digraph with p vertices, q edges and girth g. The only known efficient algorithm for this problem is given by Bredeson and Hakimi [4] and rests on a lemma guaranteeing the existence of certain collections of cutsets. As is clear from the inductive proof given in [4] and the application in [8], the authors only allow fundamental cutsets, i.e. cutsets of the form $C = C(X, Y)$ which consist of all edges joining the two parts X and Y of a partition $V = X \cup Y$ of the vertex set. As we shall point out, their result is only partially correct. We shall now give the following corrected version.

Lemma 4.1. *Let* $|G|$ *be a connected graph with* p *vertices,* q *edges and girth* g*, and let* e *be an edge of* $|G|$ *which is contained in some cycle. Then there exist* $g-1$ *fundamental cutsets* $B_1, \ldots, B_{g-1}$ *which pairwise intersect in* e *only. It is possible to test whether or not an edge* e *is contained in a cycle and (in the positive case) to compute a collection of cutsets as described above in* $O(q)$ *steps.*

Proof: Let u and v denote the end vertices of e and consider the graph $H = |G| \backslash e$. As e is contained in a cycle, H is still connected. Since $|G|$ has girth g, we have $d(u, v) \geq g - 1$ in H. For $i = 1, \ldots, g - 1$, let A_i consist of all edges of H which join two vertices which have distances $i - 1$ and i from u in H, respectively. Thus A_i is the fundamental cutset of H belonging to the partition $V = X_i \cup Y_i$, where X_i consists of all vertices at distance at most $i - 1$ from u. Clearly, the A_i are pairwise disjoint. Hence the sets $B_i = A_i \cup \{e\}$ pairwise intersect in e only; as $|G|$ has girth g, each of these edge sets separates u from v in $|G|$ and hence is a cutset in $|G|$ (in fact a fundamental cutset of $|G|$, belonging to the same partition $V = X_i \cup Y_i$). Moreover, these cutsets can be computed in $O(q)$ steps by conducting a BFS in H with root vertex u. Finally, note that e is in a cycle

of $|G|$ if and only if H is still connected; thus it is easily possible to check the hypothesis of the lemma while simultaneously attempting to construct the desired fundamental cutsets. □

Lemma 4.1 was claimed in [4] without the assumption that the edge e under consideration is contained in some cycle. The argument given there clearly is incomplete, since the case where the removal of e disconnects $|G|$ was not considered. In fact, it is in general even incorrect: For instance, let $|G|$ consist of a single cycle C of length g together with a pendant edge e; then it is easily seen that the largest collection of fundamental cutsets pairwise intersecting in e only has size $\lfloor g/2 \rfloor + 1$. Also, our proof of the corrected result is more constructive, since we explicitly exhibit the desired collection of cut sets.

Using Lemma 4.1, Hakimi and Bredeson [4, 8] essentially obtained majority logic decoding algorithms for both the binary and ternary cases. In view of the remarks above, their algorithms require some correction, though. We now describe a corresponding decoding procedure for a circulation code $\mathbf{C} = \mathbf{C}(G)$. To this end, we need to associate a ternary vector $\mathbf{c}$ with each fundamental cutset $C = C(X, Y)$ of the underlying graph $|G|$ as follows. Note first that we may consider C as ordered by selecting an ordering of the two parts X and Y; let us say that we consider the edges of C to be oriented from X to Y. Now the e-coordinate $c(e) \neq 0$ if and only if e occurs is C; moreover, $c(e) = +1$ if e is a forward edge in c, i.e., if the orientation of e in C agrees with that of e in G, and $c(e) = -1$ if e is a backward edge in C.

Algorithm 4.2: Consider a circulation code $\mathbf{C}(G)$ with parameters $[q, q - p + 1, g]$ based on the connected digraph G, where $g \geq 2t + 1$. Let $\mathbf{x}$ be a word received and assume that at most t errors have occurred, i.e. $\mathbf{x}$ is of the form $\mathbf{x} = \mathbf{c} + \mathbf{f}$ for some unknown circulation $\mathbf{c}$ on G and some unknown error vector $\mathbf{f}$ with a support S consisting of at most t edges.

For every arc e, perform the following steps:

(1) Determine whether or not e is contained in a cycle of $|G|$ and (in the positive case) compute a collection of fundamental cutsets $B_1(e), \ldots, B_{g-1}(e)$ of $|G|$ pairwise intersecting in e only.

(2) If e is not contained in a cycle of $|G|$, put $c(e) = 0$; otherwise, perform Steps (3) and (4).

(3) For $i = 1, \ldots, g - 1$, choose the orientation of the cutset $B_i(e)$ such that e is a forward arc in $B_i(e)$, let $\mathbf{b}_i(e)$ be the associated ternary vector and define a_i to be the inner product of $\mathbf{b}_i(e)$ with $\mathbf{x}$.

(4) Let $f(e)$ be the symbol occurring most often among the a_i, and put $c(e) = x(e) - f(e)$. (In case of a tie, choose $f(e) = 0$.)

Theorem 4.3. *Algorithm 4.2 correctly decodes* $\mathbf{C}(G)$ *in* $O(q^2)$ *steps.*

Proof: If an arc e is not contained in a cycle of $|G|$, it obviously cannot occur in the support of any circulation associated with a fundamental cycle of $|G|$. As noted at the end of Section 1, $\mathbf{C} = \mathbf{C}(G)$ has a basis consisting of such circulations, and therefore e cannot be in the support of any word in $\mathbf{C}$. This shows that the corresponding coordinate is correctly decoded to $c(e) = 0$ in Step (2). Now consider any arc e which is contained in a cycle of $|G|$; then the cutsets in Step (1) exist by Lemma 4.1. It is well-known that (the ternary vector associated with) any oriented cutset is orthogonal to each element of $\mathbf{C}$; in other words, the circulation space $\mathbf{C}$ of G is the orthogonal complement of the bond space generated by the ternary vectors associated with the fundamental cuts, see e.g. [3]. Hence the inner product ai defined in Step (3) is really just the inner product of $\mathbf{b}_i(e)$ with the error vector $\mathbf{f}$. Assume first that the e-coordinate is correct. Since at most t errors have occurred, the corresponding arcs involve at most t of the cutsets $B_i(e)$, and thus $\mathbf{f}$ can have inner product $\neq 0$ with at most t of the vectors $\mathbf{b}_i(e)$. Hence $\mathbf{f}$ is orthogonal to at least $g - 1 - t \geq t$ of these vectors, and thus we correctly put $f(e) = 0$ in Step (4). Finally, let the e-coordinate be incorrect. Then there are at most $t - 1$ further errors, and the corresponding arcs can involve at most $t - 1$ of the cutsets $B_i(e)$. For these cutsets, the inner product of $\mathbf{b}_i(e)$ with $\mathbf{f}$ might turn out to be 0; but for the remaining cutsets, the single error in the e-component will result in the inner product $a_i = f(e) \neq 0$, since the e-coordinate of each of the vectors $\mathbf{b}_i(e)$ is $+1$, because the orientation of $B_i(e)$ was chosen to agree with that of e. Again, we see that the definition of $c(e)$ in Step (4) is correct, since $a_i = f(e)$ occurs at least $(g-1) - (t-1) \geq t+1$ times. This proves the correctness of Algorithm 4.2, and it only remains to consider the complexity. According to Lemma 4.1, for a given edge e, Step (1) can be performed in $O(q)$ steps. Using adjacency lists, the inner products in Step (3) can also be computed in $O(q)$ steps, and the same clearly holds for Steps (2) and (4). Since we have q edges, we get an overall complexity of $O(q^2)$. □

We remark that Algorithm 4.2 obviously may be performed in parallel in linear time with a linear number of processors. In the binary case, it turned out to be possible to find a considerably more efficient (sequential) decoding algorithm via techniques from combinatorial optimization. This approach was first suggested by Ntafos and Hakimi [20] and later also by Solé [23]; the following explicit algorithm was given in [17]. Here it is always assumed that $\mathbf{C}$ is a t-error correcting code, i.e. $t \geq (g-1)/2$.

Algorithm 4.4: Consider an even graphical code $\mathbf{C}_E(|G|)$ with parameters $[q, q-p+1, g]$ based on the connected graph $|G|$, where $g \geq 2t+1$. Let X be a word received and assume that at most t errors have occurred, i.e. X

is subgraph of $|G|$ of the form $X = C + S$ for some unknown even subgraph C of $|G|$ and some unknown subgraph S consisting of at most t edges.

(1) Find the odd degree pattern W of X by computing the degrees of all vertices in X; write $|W| = 2w$.

(2) Compute the distance $d(x, y)$ between x and y in $|G|$ for every pair $\{x, y\}$ of vertices in W.

(3) Form the complete graph K on W.

(4) Find a minimum weight perfect matching $M = \{x_i y_i : i = 1, \ldots, w\}$ of K with respect to the weight function d computed in (2).

(5) Determine a path P_i of length $d(x_i, y_i)$ between x_i and y_i in $|G|$ for $i = 1, \ldots, w$.

(6) Let S be the symmetric difference of the paths $P_1, \ldots, P_w$ computed in (5).

(7) Output $C = X + S$.

It is not difficult to show that the preceding algorithm correctly decodes $\mathbf{C}_E(|G|)$ in $O(tq + t^3)$ steps. Note that the crucial stage in this decoding algorithm uses the odd degree pattern W of the subgraph X received to find the error subgraph E, which just is the spanning forest of least weight in $|G|$ which has the same vertices of odd degree as X. In combinatorial optimization, the problem of finding a spanning forest of least weight with $2t$ prescribed vertices of odd degree is usually called the *t-join problem* [18]; solving this problem via an application of the CPP is in fact the standard approach. Note also that in our case the necessary w-join is uniquely determined from the $2w$ vertices of odd degree in W.

The preceding remarks suggest to try and use a combinatorial optimization approach also in the ternary case. As we shall now explain, this seems to be a quite difficult problem. In what follows, we shall denote the e-coordinate of a ternary vector $\mathbf{x}$ by $x(e)$. As noted before, odd degree patterns in the binary case are replaced by excess patterns in the ternary case. Indeed, we have seen in Section 3 that the excess pattern is fundamental for decoding augmented circulation codes.

Thus let $\mathbf{x}$ be once more a word received and assume that at most t errors have occurred, i.e. $\mathbf{x}$ is of the form $\mathbf{x} = \mathbf{c} + \mathbf{f}$ for some unknown circulation $\mathbf{c}$ on G and some unknown error vector $\mathbf{f}$ with a support S consisting of at most t edges, where $g \geq 2t + 1$. We first point out that it suffices to determine the support S of $\mathbf{f}$: Note that S is a forest and that all leafs of this forest have excess $\neq 0$. Choose a leave u and let e be the unique arc incident with u. Then there is a unique way of correcting the

e-coordinate of $\mathbf{x}$ such that the excess in u becomes 0, removing e from the error subgraph S. Of course, we also have to adjust the excess of the other end vertex v of e accordingly. Continuing in this way, we can inductively correct $\mathbf{x}$ to the associated circulation $\mathbf{c}$.

We are left with the problem of determining the error forest S. In the binary case, this was simple to do via the CPP-approach described in Algorithm 4.4, since S just was the w-join of the $2w$ vertices of odd degree. Unfortunately, things are not as simple in the ternary case. We again have an even number, say $2w$, of vertices with excess $\neq 0$, but they now split into w vertices with excess $+1$ and w vertices with excess -1, since the sum of all excesses is easily seen to be 0 for any ternary vector $\mathbf{x}$. Note that any error edge e contributes $+1$ to the excess at one of its end vertices, say u, and -1 to the excess at the other vertex v. If u is a leaf and we correct $x(e)$ accordingly (to obtain excess 0 in u), we add $+1$ to the excess at v. Thus correcting edges along a path P in S starting at a leaf "moves" excess from its start vertex to its end vertex. Of course, the positive excess at a "plus-leaf" has to end up to neutralize a negative excess somewhere, if we want to correct $\mathbf{x}$ to a circulation. The problems stem from the fact, that there are two fundamentally different ways in which this may happen: Either the path P taking care of the plus-leaf u ends at some minus-leaf v, or it ends at an interior vertex z of S where it comes together with two other paths starting at plus-leafs, in which case these three excesses $+1$ cancel to excess 0, since we consider ternary functions. Thus our forest S contains $2w$ vertices, half of which have sign $+$, while the other have sign $-$, in such a way that the discrepancy between the number of plus-vertices and the number of minus-vertices in each component tree of S is a multiple of 3. Moreover, since the error pattern is unique in view of the assumption $g \geq 2t + 1$, we require the smallest forest with these properties. Hence we could also decode the circulation code $\mathbf{C}(G)$ if we had an efficient algorithm for the following apparently difficult problem from combinatorial optimization.

Problem 4.5 ("signed Steiner forest"): Let G be a connected digraph with girth g and consider a set of $2w$ vertices half of which have sign $+$, while the other have sign $-$. Find a forest S of smallest cardinality such that the discrepancy between the number of plus-vertices and the number of minus-vertices in each component tree of S is a multiple of 3.

Problem 4.5 combines features that are on one hand reminiscent of the Steiner network problem (finding a smallest forest joining up given vertices) and on the other hand of a variation of bipartite matching (either matching plus-vertices with minus-vertices, or "killing" vertices with the same sign in groups of three). To our knowledge, this problem has not been considered before; this is not surprising, as the divisibility condition involved does not seem to be a "natural" condition in optimization. We expect Problem

4.5 to be difficult, since the Steiner network problem is known to be NP-complete [5] and since the divisibility condition destroys the pure bipartite matching structure one might hope for. For our purposes, it would of course be sufficient to solve Problem 4.5 for digraphs which have large girth in comparison to w, since we have the restriction $g \geq 2w+1$; if necessary, we could even add the hypothesis that a forest with the required properties and cardinality at most $(g-1)/2$ exists. Of course, under these very severe restrictions, Algorithm 4.2 provides an efficient way of solving Problem 4.5. Nevertheless, we hope that there is a simpler, more direct optimization approach to Problem 4.5 at least under the assumption that $g \geq 2w+1$. Unfortunately, we have not been able up to now to find such an approach.

We finally note that one may also consider q-ary codes associated with digraphs, as was first noted by Bobrow and Hakimi [2]; we shall use our techniques to deal with this problem in a forthcoming note [15].

References

[1] L.S. Bobrow, Decoding augmented cut set codes, *IEEE Trans. Inform. Th.* **17** (1971), 218–220.

[2] L.S. Bobrow and S.L. Hakimi, Graph theoretic q-ary codes, *IEEE Trans. Inform. Th.* **17** (1971), 215–218.

[3] J.A. Bondy and U.S.R. Murty, *Graph theory with applications*, North Holland, Amsterdam, 1976.

[4] J.G. Bredeson and S.L. Hakimi, Decoding of graph theoretic codes, *IEEE Trans. Inform. Th.* **13** (1967), 348–349.

[5] M.R. Garey and D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, Freeman, New York, 1979.

[6] S.L. Hakimi, Recent progress and new problems in applied graph theory, *Proc. IEEE Region Six Annual Conf.*, (1966), 635–643.

[7] S.L. Hakimi and J.G. Bredeson, Graph theoretic error-correcting codes, *IEEE Trans. Inform. Th.* **14** (1968), 584–591.

[8] S.L. Hakimi and J.G. Bredeson, Ternary graph theoretic error-correcting codes, *IEEE Trans. Inform. Th.* **15** (1969), 435–436.

[9] S.L. Hakimi and H. Frank, Cut set matrices and linear codes, *IEEE Trans. Inform. Th.* **11** (1965), 457–458.

[10] F. Harary, *Graph theory*, Addison Wesley, Reading, Mass, 1969.

[11] D. Jungnickel, *Graphen, Netzwerke und Algorithmen (3rd ed.)*, B.I. Wissenschaftsverlag, Mannheim, 1994.

[12] D. Jungnickel, M.J. de Resmini and S.A. Vanstone, Codes based on complete graphs, *Designs, Codes and Cryptography* **8** (1996), 159–165.

[13] D. Jungnickel and S.A. Vanstone, An application of coding theory to a problem in graphical enumeration, *Archiv Math.* **65** (1995), 461–464.

[14] D. Jungnickel and S.A. Vanstone, Graphical codes: A tutorial, *Bulletin ICA* **18** (1996), 45–64.

[15] D. Jungnickel and S.A. Vanstone, q-ary graphical codes, Submitted

[16] D. Jungnickel and S.A. Vanstone, An application of difference sets to a problem concerning graphical codes, *J. Stat. Planning Inf.*, to appear

[17] D. Jungnickel and S.A. Vanstone, Graphical codes revisited, *IEEE Trans. Inform. Th.* **43** (1997), 136–146.

[18] L. Lovász and M.D. Plummer, *Matching theory*, North Holland, Amsterdam, 1986.

[19] F.J. MacWilliams and N.J.A. Sloane, *The theory of error-correcting codes*, North Holland, Amsterdam, 1977.

[20] S.C. Ntafos and S.L. Hakimi, On the complexity of some coding problems, *IEEE Trans. Inform. Th.* **27** (1981), 794–796.

[21] H. Oral, Constructing self-dual codes using graphs, *J. Comb. Th. (B)* **52** (1991), 250–258.

[22] C.H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: Algorithms and complexity*, Prentice Hall, Englewood Cliffs, N.J., 1982.

[23] P. Solé, Covering codes and combinatorial optimization, *Springer Lecture Notes in Computer Science* **539** (1991), 426–433.

[24] P. Solé and T. Zaslavsky, The covering radius of the cycle code of a graph, *Discr. Appl. Math.* **45** (1993), 63–70.

[25] P. Solé and T. Zaslavsky, A coding appoach to signed graphs, *SIAM J. Discr. Math.* **7** (1994), 544–553.

[26] P. Solé and T. Zaslavsky, Maximality of the cycle code of a graph, *Discr. Math.* **128** (1994), 401–405.

[27] J.H. van Lint, *Introduction to coding theory*, Springer, New York, 1982.