A5641,2

Scan

Ramanath + Walsh

2 sgs

f91 REPRINT #0

S641
S642

Enumeration and Generation of a Class of Regular Digraphs

M. V. S. Ramanath
T. R. Walsh
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF WESTERN ONTARIO
ONTARIO, CANADA N6A 5B7

ABSTRACT

We study the class of directed graphs that have indegree = outdegree = 2 at every vertex. These digraphs can be decomposed uniquely into "alternating cycles"; we use this decomposition to present efficient techniques for counting and generating them. The number (up to isomorphism) of these digraphs and the number of connected ones on up to 20 vertices have been computed and are presented.

1. INTRODUCTION

We present techniques for enumerating and generating the class of regular digraphs whose vertices all have indegree = outdegree = 2. These graphs have arisen in a variety of contexts but appear not to have been the objects of focused attention. For example, the teleprinter diagrams (or Good diagrams) on a binary alphabet (see [1]) belong to this class; the Cartesian product of two directed cycles [6] and, more generally, the Cayley color digraphs [7] of a finite group with two generators are also members of this class. The central distinguishing feature of these digraphs, as noted in [4], is that they can be uniquely decomposed into "alternating cycles" in linear time.

In the present article, we use the alternating cycle decomposition to develop enumeration formulae for these digraphs and an algorithm to generate them efficiently; isomorphic copies can be discarded on the fly without having to compare them with all previously generated members.

Journal of Graph Theory, Vol. 11, No. 4, 471–479 (1987)
 CCC 0364-9024/87/040471-09$04.00

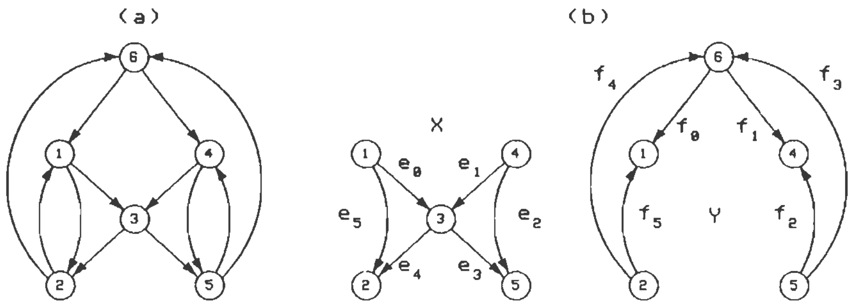
2. PRELIMINARIES

We will briefly summarize the terminology and a few results from [4]. We use $[m \,..\, n]$ to denote the empty set if $m > n$ and the set $\{m, m + 1, \ldots, n\}$ otherwise. Suppose $G = (V, A)$ is a finite digraph and $X \subseteq A$ (we allow loops but not parallel arcs). X is called an *alternating cycle* (ac) iff its elements (arcs) can be ordered as $e_0, e_1, \ldots, e_{2r-1}$ such that, for i in $[0 \,..\, 2r - 1]$, e_i and $e_{i \oplus 1}$ have a common end-vertex [start-vertex] if i is even [odd] (where $\oplus$ denotes addition mod $2r$). Such an ordering is called an *alternating ordering*. The common end-vertices (for even i) are called *exit vertices* and the common start-vertices (for odd i) are called *entry vertices* of X. An arc e_i of X is called *clockwise* [*anticlockwise*] if i is even [odd]. If a vertex is both an exit vertex and an entry vertex of X, it is called a *saturated vertex*. If X has no saturated vertices, we say it is a *simple ac*. If X contains all the vertices of G we say that it is a *spanning ac*.

If G has indegree = outdegree = 2 at every vertex, we say that G is a *2-diregular digraph* (2-dd). A 2-dd can be uniquely decomposed into acs in linear time [4]. Figure 1 shows an example of a 2-dd composed of 2 acs X and Y, each with six arcs; their exit, entry, and saturated vertices are also shown. Figure 2 shows a 2-dd with three acs, all of which are simple but only one is spanning. The acs of Figure 1 are neither spanning nor simple.

Suppose the acs of G are $X_1, \ldots, X_k$ and $|X_i| = 2m_i$ for i in $[1 \,..\, k]$, and $m_i \leq m_{i+1}$. Then, the k-tuple $(2m_1, \ldots, 2m_k)$ is called the *ac-profile* of G. Clearly, if two 2-dds are isomorphic, they must have identical ac-profiles, but the converse may not hold.

Suppose G is a 2-dd. Let $G' = (V', A')$ be the digraph obtained by splitting each vertex w of V into two vertices w' and w'' such that all incoming [outgo-



(c)

	X	Y
Exit vertices	2,5,3	1,4,6
Entry vertices	1,4,3	2,5,6
Saturated vertices	3	6

FIGURE 1.

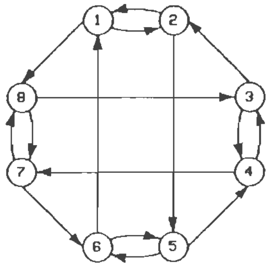


FIGURE 2.

ing] arcs of w become incoming [outgoing] arcs of w' [w'']. Let b be the bijection from $\{w'' \mid w \text{ in } V\}$ to $\{w' \mid w \text{ in } V\}$ defined by $b(w'') = w'$. Clearly G' is merely a collection of vertex disjoint simple acs with $|V'| = 2|V|$ and $|A'| = |A|$. In other words, we have

Proposition 2.1. Suppose $G = (V, A)$ is a digraph such that $A = \{X_1, \ldots, X_k\}$ ($k \geq 1$) is a collection of vertex-disjoint simple acs with $|X_i| = 2m_i$ and $m_i \leq m_{i+1}$. Let $P = \{v \mid v$ is an entry vertex of some $X_i\}$ and $Q = \{v \mid v$ is an exit vertex of some $X_i\}$. Then

(a) every bijection b from P to Q yields a (unique) 2-dd by identifying w and $b(w)$ for each w in P, and
(b) every 2-dd with ac-profile $(2m_1, \ldots, 2m_k)$ can be obtained as in (a) from at least one such bijection.

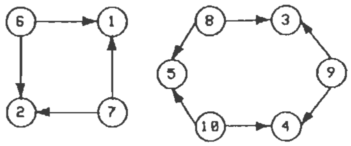
Proof. Follows since the ac decomposition is unique [4]. ∎

Suppose G, V, A, X_i, m_i, P, and Q are as in the above proposition. Let $n = \Sigma m_i$. Assume that the vertices of P are numbered from $n + 1$ to $2n$ and those of Q from 1 to n. Let M_n be the set of all bijections from P to Q. We define an equivalence relation $\sim$ and a partial order $\preccurlyeq$ on M_n as follows: $b \sim b'$ iff b and b' yield isomorphic 2-dds; $b \preccurlyeq b'$ iff $b \sim b'$ and $(b(n + 1), \ldots, b(2n)) \leq (b'(n + 1), \ldots, b'(2n))$ in the usual lexicographic ordering of tuples. Each equivalence class of $\sim$ is totally ordered in M_n and hence has a unique minimal element. We now have the following corollary of Proposition 2.1:

Corollary. There is a one-to-one correspondence between the minimal elements of $(M_n, \preccurlyeq)$ and the isomorphism classes of 2-dds with ac-profile $(2m_1, \ldots, 2m_k)$.

Figure 3 illustrates the 2-dds obtained from the minimal bijections for the ac-profile (4, 6). We now present a criterion for minimality of a bijection.

Proposition 2.2. b is minimal iff $b \preccurlyeq fbf^{-1}$ for every automorphism f of G.



	Minimal bijections				
	6	7	8	9	10
b1	1	2	3	4	5
b2	1	2	3	5	4
b3	1	2	4	5	3
b4	1	3	2	4	5
b5	1	3	2	5	4
b6	1	3	4	5	2
b7	1	3	5	4	2
b8	3	4	1	2	5
b9	3	4	1	5	2

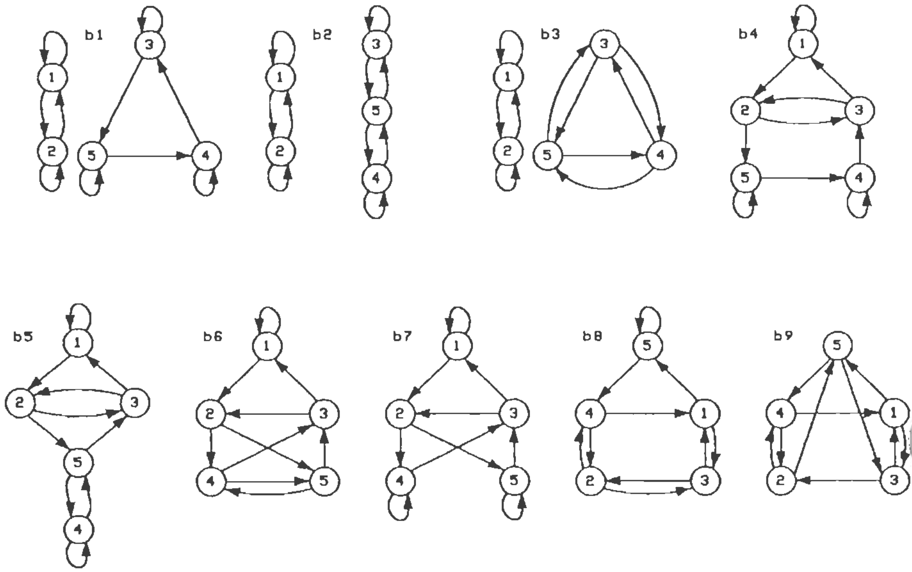


FIGURE 3.

Proof. b is minimal iff $b \preccurlyeq b'$ for every b' in the equivalence class of b. The result now follows since b' and b yield isomorphic 2-dds iff there is an automorphism f of G such that $b' = fbf^{-1}$. ∎

3. ENUMERATION

In this section we derive a formula for the number of nonisomorphic 2-dds with a given ac-profile; the numbers obtained from this formula can then be used to get the number of nonisomorphic connected 2-dds using the methods of [2, pp. 90–93].

By Proposition 2.1 it suffices to consider the simple acs into which any 2-dd can be split and the bijections from the set of entry vertices to the set of exit vertices. Hence, we need only count the number of equivalence classes of the equivalence relation $\sim$. By Burnside's lemma, [2, p. 39] this number is given by

$$\frac{\sum_{f \in F} \text{\# of bijections fixed by } f}{|F|} \tag{1}$$

where F denotes the set of automorphisms of the set of (simple) acs under consideration.

An automorphism f divides the sets of entry and exit vertices independently into disjoint cycles; let us call them *entry cycles* and *exit cycles,* respectively. If f fixes the bijection b, then for any entry vertex, we have $fb(x) = bf(x)$; thus, if x lies on an entry cycle of length c, the images of all vertices on that cycle under b are determined by $b(x)$ and they form an exit cycle of length c. From this, we deduce part (a) of

Proposition 3.1. Let f be an automorphism (of a set of acs) such that exactly m of its entry cycles are of length c. Then,

(a) f fixes some bijection $\Rightarrow$ exactly m of the exit cycles of f are of length c, and
(b) the number of independent choices of images under a bijection fixed by f of the vertices in the entry cycles of length c is $cm(cm - c)(cm - 2c) \ldots [cm - (m - 1)c] = c^m m!$

Proof. After choosing an image for one of these entry vertices, there are c fewer exit vertices left for the next choice.

Now suppose f has i_k entry cycles and j_k exit cycles of length k for $k = 1, 2, \ldots$. Following [2, pp. 35–38] we define the *cycle type* of f to be the monomial

$$y_1^{i_1} y_2^{i_2} \cdots z_1^{j_1} z_2^{j_2} \cdots \tag{2}$$

and the *cycle index* of the set of acs by the formula

$$\frac{\sum_{f \in F} \text{cycle type of } f}{|F|} \tag{3}$$

By comparing formulae (1) and (3) we find that the number of nonisomorphic 2-dds with a given ac-profile is found by replacing each cycle type in the cycle index of the corresponding set of acs with the number of bijections fixed by any automorphism with that cycle type. This latter number is found by applying Proposition 3.1 to each set of cycles of a given length and multiplying over all the cycle lengths. Combining these results, we have:

Proposition 3.2. The number of nonisomorphic 2-dds with a given ac-profile is found by computing the cycle index of the corresponding set of acs and then making the following substitutions into each monomial:

(a) each factor $(y_c z_c)^m$ is replaced by $c^m m!$;
(b) each factor y_c^m without a mate of the form z_c^m, or vice versa, is replaced by 0.

We now find the cycle index of an arbitrary set of simple acs, starting with a single ac. We recall that an automorphism must preserve entry and exit vertices; thus the group of automorphisms of an ac with $2n$ arcs is just the dihedral group D_n (rather than D_{2n}) whose cycle index is given by [2, pp. 36–37]. The formulae given there are easily modified to accommodate our definition of cycle type to yield

$$Z(D_n) = \frac{1}{2n}\left(R + \sum_{r|n} \varphi\left(\frac{n}{r}\right) y^r_{n/r} z^r_{n/r}\right) \tag{4}$$

$$\text{where } R = \begin{cases} n y_1 z_1 y_2^{(n-1)/2} z_2^{(n-1)/2} & \text{if } n \text{ is odd} \\ \frac{n}{2}(y_1^2 z_2 + z_1^2 y_2) y_2^{(n-2)/2} z_2^{(n-2)/2} & \text{otherwise,} \end{cases}$$

and φ is the Euler totient function.

Next we compute the cycle index for a set of k acs, each with $2n$ arcs. It follows from the results of [2, pp. 178–182] that the group of automorphisms of this set of acs is the Wreath product of the automorphism group of a single ac about the full symmetric group S_k representing the permutations of the k acs induced by these automorphisms. The cycle index of S_k is given in [2, p. 36]:

$$Z(S_k) = \frac{1}{n!} \sum_{j_1+2j_2+3j_3\cdots + kj_k=k} \frac{k!}{1^{j_1}2^{j_2}\cdots k^{j_k} j_1! j_2! \cdots j_k!} x_1^{j_1} x_2^{j_2} \cdots x_k^{j_k}. \tag{5}$$

The cycle index of the above-mentioned Wreath product is denoted by $Z(S_k)[Z(D_n)]$ and is found by replacing each x_i in (5) above by (4) after first "inflating" (4) by multiplying all its cycle lengths by i.

Finally, we compute the cycle index for the general set of acs: assume we have k_i acs each with $2n_i$ arcs for each i in $[1..t]$. Since any automorphism takes each ac into another with the same number of arcs, the cycle index for this set is just the product:

$$\prod_{i=1}^{t} Z(S_{k_i})[S(D_{n_i})]. \tag{6}$$

Applying Proposition 3.2 to the cycle index for a general set of acs [given by formula (6)], we found the number of nonisomorphic 2-dds for each ac-profile with up to 20 vertices. The number of connected ones was also computed; we omit the details, since the methods of [2, pp. 90–93] for counting connected graphs by the number of edges as well as vertices are easily generalized from one extra variable (the number of edges) to an indefinite number (the ac-profile). The results are given in full in [5] and are summarized in Table 1 of the present article.

The computations were done on a DEC VAX 8600 and took 42 seconds of CPU time to run. The code is approximately 400 lines of FORTRAN and uses about 2 megabytes of memory.

4. GENERATION

Our generation procedure uses a backtracking procedure [3] for generating permutations in lexicographic order; this algorithm generates permutations b by trying all possible values for $b(i)$ and then calling itself recursively to try all values for $b(i + 1)$. The generation can now be summarized as follows:

1. Use a backtracking procedure to generate the $n!$ bijections (from $[n + 1 \,..\, 2n]$ to $[1 \,..\, n]$) in lexicographic order; let b be the initial segment of a bijection constructed so far.

TABLE 1.

Vertices	No. 2-dds	No. Connected 2-dds
2	1	1
3	3	3
4	8	7
5	27	24
6	131	117
7	711	663
8	5055	4824
9	41607	40367
10	389759	381554
11	4065605	4001849
12	46612528	46043780
13	581713045	576018785
14	7846380548	7783281188
15	113718755478	112953364381
16	1762208816647	1752128923245
17	29073392136390	28930230194371
18	508777045979418	506596534953769
19	9412803430912738	9377358316123127
20	183565340753506398	182952980397576097

2. As each alternative is tried for $b(i)$ in Step 1, determine whether b can be eliminated right away by checking some easy criteria for nonminimality; if so, try the next alternative for $b(i)$, backtracking if necessary.

3. When a bijection b not eliminated in Step 2 is found, check if there is an automorphism f such $fbf^{-1} < b_i$ if not, output b as a minimal bijection.

We have generated all minimal bijections for 2-dds with up to 11 vertices. The numbers obtained by enumeration and generation confirm one another; since both numbers were obtained by computer programming, such confirmation is very valuable. The actual CPU times (on a DEC VAX 11/780) are given in [5] and a brief extract, for $n = 11$, is given in Table I. The last column shows the number of minimal bijections output per second on the average.

We have found a series of five criteria for nonminimality that can be used in Step 2: the details are provided in [5]. Clearly, Step 3 is likely to be very time consuming, especially when the number of automorphisms is large; hence, the more bijections that can be eliminated in Step 2, the faster the generation is likely to proceed. Columns 5 and 6 of Table 2 (the entries are rounded to the nearest integer) demonstrate the dramatic effectiveness of these criteria: for example, for the ac profile (4, 4, 4, 4, 6), there are, on the average, around 22914 nonminimal bijections for each minimal one; of these, all but 23 are eliminated in Step 2. Even so, approximately 90% of the time is spent in Step 3 of the algorithm. The program is approximately 1200 lines of Pascal source code.

TABLE 2.

n	ac-Profile	No. Minimal Bijections	No. Reaching Step 3	$\frac{n!}{\text{Col.3}}$	$\frac{\text{Col.4}}{\text{Col.3}}$	CPU time (h:min:s)	No./s
11	4 4 4 4 6	1742	40653	22914	23	0:05:44	5
11	4 6 6 6	9800	75901	4073	8	0:08:49	19
11	4 4 6 8	30594	223251	1305	7	0:28:02	18
11	6 8 8	55745	308199	716	6	0:29:36	31
11	4 4 4 10	12704	130383	3142	10	0:15:57	13
11	6 6 10	63780	239203	626	4	0:23:38	45
11	4 8 10	128818	540251	310	4	0:55:12	39
11	4 6 12	150390	660195	265	4	1:11:05	35
11	10 12	334728	649209	119	2	0:45:22	123
11	4 4 14	96765	498145	413	5	0:55:21	29
11	8 14	357759	792155	112	2	0:59:47	100
11	6 16	441186	1428876	90	3	2:02:10	60
11	4 18	565269	1861461	71	3	2:53:40	54
11	22	1816325	4226331	22	2	4:44:37	106

References

[1] S. Even, *Graph Algorithms*. Computer Science Press, Rockville, MD (1979) pp. 8–10.

[2] F. Harary and E. M. Palmer, *Graphical Enumeration*. Academic Press, New York (1973).

[3] R. W. Irving, Permutation backtracking in lexicographic order. *Comp. J.* **27**(4) (1984) 373–375.

[4] M. V. S. Ramanath, Factors in a class of regular digraphs. *J. Graph Theory* **9**(1) (1985) 161–175.

[5] M. V. S. Ramanath and T. R. Walsh, Enumeration and generation of a class of regular digraphs. Technical Report #147, University of Western Ontario, Canada (1986).

[6] W. T. Trotter and P. Erdös, When the Cartesian product of directed cycles is Hamiltonian. *J. Graph Theory* **2** (1978) 137–142.

[7] A. T. White, *Graphs, Groups and Surfaces*. American Elsevier, New York (1973) pp. 21–33.