A new algorithm is presented for the related problems of canonically labelling a graph or digraph and of finding its automorphism group. The automorphism group is found in the form of a set of less than \( n \) generators, where \( n \) is the number of vertices. An implementation is reported which is sufficiently conserving of time and space for it to be useful for graphs with over a thousand vertices.

1. INTRODUCTION

Let \( V \) be the finite set \( \{1, 2, \ldots, n\} \). Define \( \mathcal{G}(V) \) to be the set of all (labelled) graphs with vertex set \( V \). Let \( S_n \) be the symmetric group acting on \( V \). For \( G \in \mathcal{G}(V) \) and \( g \in S_n \), define \( G^g \in \mathcal{G}(V) \) to be the graph in which vertices \( v^g \) and \( w^g \) are adjacent exactly when \( v \) and \( w \) are adjacent in \( G \). The automorphism group of \( G \), \( \text{Aut}(G) \), is the group \( \{ g \in S_n | G^g = G \} \).

The canonical label problem is to find a map \( \text{canon}: \mathcal{G}(V) \to \mathcal{G}(V) \) such that for \( G \in \mathcal{G}(V) \) and \( g \in S_n \),

1. \( \text{canon}(G) \) is isomorphic to \( G \), and
2. \( \text{canon}(G^g) = \text{canon}(G) \).

Note that there may be many functions \( \text{canon} \) satisfying (1) and (2).

If \( G, H \in \mathcal{G}(V) \), we see that \( G \) and \( H \) are isomorphic if and only if \( \text{canon}(G) = \text{canon}(H) \).

In this paper we present a new algorithm for computing \( \text{canon}(G) \) which will also find a set of fewer than \( n \) automorphisms which generate \( \text{Aut}(G) \). With only minor modifications which we will indicate, the algorithm is equally applicable to digraphs. Undefined graph theoretic or group theoretic concepts can be found in \([1]\) or \([7]\) respectively.

2. EQUITABLE PARTITIONS

Let \( V = \{1, 2, \ldots, n\} \). A partition of \( V \) is a collection \( \pi \) of disjoint non-empty subsets of \( V \) whose union is \( V \). The elements of \( \pi \) are called its cells. An ordered partition of \( V \) is a sequence \( (C_1, C_2, \ldots, C_k) \) for which \( \{C_1, C_2, \ldots, C_k\} \) is a partition. The sets of all partitions of \( V \), and of all ordered partitions of \( V \) will be denoted by \( \Pi(V) \) and \( \Pi(V) \) respectively.
Define $\Pi^*(V) = \Pi(V) \cup \Pi(V)$. Let $\pi_1, \pi_2 \in \Pi^*(V)$. We write $\pi_1 \preceq \pi_2$ ($\pi_1$ is finer than $\pi_2$, $\pi_2$ is coarser than $\pi_1$) if every cell of $\pi_1$ is contained in some cell of $\pi_2$. If both $\pi_1 \preceq \pi_2$ and $\pi_2 \preceq \pi_1$, we write $\pi_1 \equiv \pi_2$. If $\pi \in \Pi^*(V)$, the number of cells of $\pi$ is denoted by $|\pi|$. $\pi$ is called discrete if $|\pi| = n$.

Let $\pi \in \Pi^*(V)$ and $g \in S_n$. Then $\pi^g \in \Pi^*(V)$ is formed by replacing each cell $C \in \pi$ by $C^g$. If $\pi = \pi^g$, $g$ is said to fix $\pi$. Denote by $\pi \vee g$ the finest partition of $V$ which is coarser than $\pi$ but fixed by $g$. The existence of $\pi \vee g$ follows from the fact that $(\Pi(V), \preceq)$ is a lattice [3].

Choose a fixed $G \in \mathcal{G}(V)$. If $W \subseteq V$ and $v \in V$, the number of vertices in $W$ which are adjacent to $v$ will be denoted by $d(v, W)$. Let $\pi \in \Pi^*(V)$. $\pi$ is said to be equitable (for $G$) if for any $C_1, C_2 \in \pi$ and $v_1, v_2 \in C_1$ we have $d(v_1, C_2) = d(v_2, C_2)$. For an arbitrary $\pi$, the coarsest equitable partition which is finer than $\pi$ will be denoted by $\xi(\pi)$. Similarly, $\theta(\pi)$ denotes the partition whose cells are the orbits of the subgroup of Aut($G$) which fixes $\pi$. The proof of the following lemma can be found in [3].

**LEMMA 1.** Let $\pi \in \Pi(V)$. Then

(i) $\theta(\pi) \leq \xi(\pi)$,

(ii) $\theta(\pi)$ is equitable, and

(iii) if $\pi$ is equitable, and $n - |\pi| \leq 5$, the smallest cells of $\pi$ of size $\geq 2$ are cells of $\theta(\pi)$. (Not true if $G$ is a digraph.)

Corneil proved in [2] that for any $\pi$, $\theta(\pi) = \xi(\pi)$ if $G$ is a tree. This can be generalised to uni-cyclic graphs and many others. See [3] for further details.

Algorithms for computing $\xi(\pi)$ have been used many times in graph isomorphism programs ([2], [5], [6]). For our own purposes, however, the following system appears to be more efficient. Let $\pi \in \Pi(V)$ and let $a$ be a subset of $\pi$.

**ALGORITHM 1:** Compute $\overline{\pi} = \mathcal{A}(G, \pi, a)$

1. $\overline{\pi} \leftarrow \pi$

2. If $a = \emptyset$ or $\overline{\pi}$ is discrete, stop.

   Choose any non-null subset $\beta$ of $a$.

   $a \leftarrow a \setminus \beta$, $i \leftarrow 1$

   (Suppose $\overline{\pi} = \{C_1, C_2, \ldots, C_k\}$ and $\beta = \{W_1, W_2, \ldots, W_r\}$.)

3. Partition $C_1$ into subsets $D_1, D_2, \ldots, D_s$ according to the vectors $d(v, W_1), d(v, W_2), \ldots, d(v, W_r)$ for $v \in C_1$.

   $\overline{\pi} \leftarrow \overline{\pi} \cup \{D_1, D_2, \ldots, D_s\} \setminus \{C_1\}$