PERIODICA POLYTECHNICA SER. MECH. ENG. VOL. 36, NOS. 3-4, PP. 299-316 (1992)

DATA STRUCTURES AND PROCEDURES FOR A POLYHEDRON ALGORITHM[1]

I. PROK

Department of Geometry
Faculty of Mechanical Engineering
Technical University of Budapest

Received: November 16, 1992

Abstract

In this paper we describe the data structures and the procedures of a program, which is based on the algorithms of [5,6]. Knowing the incidence structure of a polyhedron, the program finds all the essentially different facet pairings. The transformations, pairing the facets generate a space group, for which the polyhedron is a fundamental domain. The program also creates the defining relations of the group. Thus, we obtain discrete groups of certain combinatorial spaces. We have still to examine which groups can be realised in spaces of constant curvature (or in other simply connected spaces). Finally, we mention some results: Examining the 4-simplex, our program disproves Zhuk's conjecture concerning the number of essentially different facet pairings of d-simplices [11]. The classification of 3-simplex tilings has also been completed [7]. We have found the fundamental tilings of the Euclidean space with marked cubes and the corresponding crystallographic groups [8].

Introduction

We start from a d-dimensional polyhedron P given by its finite flag structure $\mathbf{F}$. In Section 1 we define the basic concepts connecting with $\mathbf{F}$, describe its data structure and a basic procedure on it.

In Section 2 we define the isomorphisms (combinatorial isometries) of the $(d-1)$-facets, the automorphisms (combinatorial symmetries) of a facet, and the isomorphism classes of the facets. In order to reduce our discussion, we make an assumption: the facets of P form only one isomorphism class. This assumption can be made without loss of generality, because the classes must be treated independently in the same manner. Then we describe the data structure of a class and describe the procedure that creates it from $\mathbf{F}$.

Considering a space group for which P is a fundamental domain, it can be generated by its transformations that map P onto the neighbouring polyhedron along facets. These transformations and their inverses determine a pairing of the facets of P identifying the facets in pairs. Conversely,

[1]Supported by Hungarian Nat. Found.for Sci. Research OTKA No. 1615 (1991).

if we give a system of identifying generators, it determines a group $\mathcal{G}$. Let $P^{\mathcal{G}}$ denote the $\mathcal{G}$-images of P. Then $\mathcal{G}$ acts simply transitively on $P^{\mathcal{G}}$, and the factor space $P^{\mathcal{G}}/\mathcal{G}$ (orbit space) is the same as P equipped with identifications (Section 5).

In Section 3 we introduce the facet-mappings as certain permutations of the flags. Their group contains the automorphisms of P as a subgroup (Section 4). Moreover, the involutive facet-mappings determine the systems of identifying generators. Thus, the equivalence of the generator systems at the automorphism group can be checked easily. The data structure of a facet-mapping is much more concise than a permutation of the flags. In Section 4 we describe the procedure that creates the automorphisms of P as facet-mappings. Section 5 contains a procedure for finding all the non-equivalent involutive facet-mappings (or rather all the essentially different generator systems.)

Finally, in Section 6 we describe the procedure, which carries out the Poincaré algorithm starting from an involutive facet- mapping (a generator system) to determine the defining relations of the generated group $\mathcal{G}$. Finding the transformations surrounding an edge, we obtain a cycle transformation. These will define relations with arbitrary natural exponents. The exponent determines the order of the 'rotation' subgroup of the edge stabilizer. In Section 7 we mention some results.

In order to reduce the length of our discussion, we make some agreements for our procedures.

- If a variable has already been declared, then we think the pointer that points to it. For example, the flag **f** means the pointer that points to **f** whenever the flag structure has already been created. In our figures this pointer is denoted by $\uparrow$ **f**.
- Referring to a chain of data, we think the pointer which points to the first element of the chain. We can obtain an element e_{1i} of a chain c_1 as a result of (linear) searching. Moreover, going parallel in another chain c_2 we can obtain its element e_{2i} corresponding to e_{1i}. In our procedures we do not detail these searchings.
- A vector is declared as an array indexed from 0 to $(d-1)$.

1. The Flag Structure of a Polyhedron (Isomorphisms of Face Systems)

Let a d-dimensional polyhedron P be given. Let $F^{(i)}$ denote the set of the i-faces $(i = 0, 1, \ldots, (d-1))$. We use the notations V, E and F for the sets $F^{(0)}$, $F^{(d-2)}$ and $F^{(d-1)}$. Moreover, we call their elements vertices, edges and facets, respectively.

Definition 1.1. We define a *flag* of P as an ordered d-tuple of incident 0-face, 1-face, ..., $(d-1)$-face. So the *flag structure* $\mathbf{F}$ of P is a non-empty subset of the direct product $F^{(0)} \times F^{(1)} \times \cdots \times F^{(d-1)}$. Let $\mathbf{f} = (f^{(0)}, f^{(1)}, \ldots, f^{(d-1)})$ be a flag and let us consider its i_0-, i_1-, ..., i_m-face components where $i_0, i_1, \ldots, i_m$ is a subsequence in $0, 1, \ldots, (d-1)$ (it may be empty). We say that $(f^{(i_0)}, f^{(i_1)}, \ldots, f^{(i_m)})$ is an $(i_0, i_1, \ldots, i_m)$ *face system* (or subflag) of P. •

Definition 1.2. Let a face system $(f^{(i_0)}, f^{(i_1)}, \ldots, f^{(i_m)})$ be given. The flags $\mathbf{f} = (\ldots, f^{(i_0)}, \ldots, f^{(i_1)}, \ldots, \ldots, f^{(i_m)}, \ldots)$ that contain the faces of the system in their components form the set $\mathbf{F}_{(f^{(i_0)}, f^{(i_1)}, \ldots, f^{(i_m)})} \subset \mathbf{F}$, which is called the flags of the face system. In particular, if $v \in V = F^{(0)}$, $e \in E = F^{(d-2)}$ and $f \in F = F^{(d-1)}$, then $\mathbf{F}_v$, $\mathbf{F}_e$ and $\mathbf{F}_f$ are the set of the flags of the vertex v, the edge e and the facet f, respectively. (The set $\mathbf{F}_\emptyset$ contains the flags of the empty system: $\mathbf{F}_\emptyset = \mathbf{F}$.) •

Definition 1.3. The flag structure of the polyhedron P is *strictly connected* iff the following two properties are fulfilled.

1. Each flag in $\mathbf{F}$ has exactly one i-adjacent flag for every i. Two flags are i-adjacent iff their i-face components are different but their other components coincide.
2. If $\mathbf{g}$ and $\mathbf{h}$ are different flags in $\mathbf{F}$, there exists a finite sequence of flags
$$\mathbf{g} = \mathbf{f}_1, \quad \mathbf{f}_2, \quad \ldots, \quad \mathbf{f}_{n-1}, \quad \mathbf{f}_n = \mathbf{h}$$
so that each $\mathbf{f}_j$ $(1 \le j \le n)$ has the common face components of $\mathbf{g}$ and $\mathbf{h}$ (in other words each element of the sequence is a flag of the face system given by the common components of $\mathbf{g}$ and $\mathbf{h}$) furthermore $\mathbf{f}_{k+1}$ is i-adjacent to $\mathbf{f}_k$ $(1 \le k < n)$ for certain i. •

In the following we assume that the flag structure of P is strictly connected.

Definition 1.4. Let $(g^{(i_0)}, g^{(i_1)}, \ldots, g^{(i_m)})$ and $(h^{(i_0)}, h^{(i_1)}, \ldots, h^{(i_m)})$ be $(i_0, i_1, \ldots, i_m)$ face systems and let $\mathbf{G}$ and $\mathbf{H}$ denote the flags of these systems, respectively. Furthermore, let $j_0, j_1, \ldots, j_n$ be the complement sequence of $i_0, i_1, \ldots, i_m$. We say that these *systems are isomorphic* if there exists a bijective map $\psi : \mathbf{G} \to \mathbf{H}$ preserving the j_0-, j_1-, ..., j_n-adjacencies. We call ψ an isomorphism between these systems. Moreover, the bijective map $\omega : \mathbf{G} \to \mathbf{G}$ is an automorphism of the face system $(g^{(i_0)}, g^{(i_1)}, \ldots, g^{(i_m)})$ if ω preserves the j_0-, j_1-, ..., j_n-adjacencies. Since P is strictly connected, ψ (and ω) is given by arbitrary corresponding flags (by any flag and its image). •

Thus, we can speak about the isomorphism classes of the $(i_0, i_1, \ldots, i_m)$ face systems and the automorphism group of a face system. The main things will be the isomorphism classes of facets and the automorphism group of the whole polyhedron.

Remember that the polyhedron P is given by its flag structure. Thus the input data of our program are the flags of P. We create the following two data structures of them. First we give the i-faces of the polyhedron.

Data 1.1. An i-face is a record, which contains two fields. The first field is the mark of the i-face and the second one is a pointer to the mark of the next i-face to form a chain of i-faces. Moreover, we need a head for the i-face chains. It is a vector of the pointers, which point to the chain of i-faces ($i = 0, 1, \ldots, (d-1)$). (*Fig. 1.1*) •

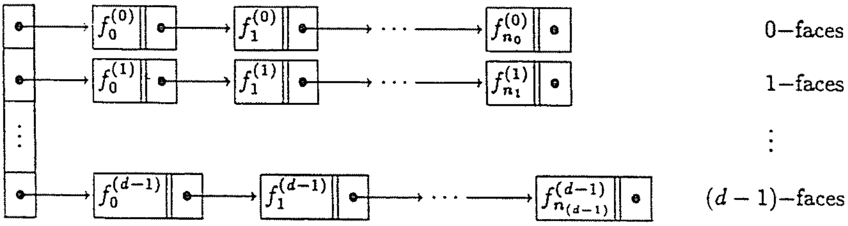

Fig. 1.1. The data structure of the faces

Looking for the i-adjacencies to each flag we prepare the structure of the flags.

Data 1.2. A flag is a record composed of three fields. The first field is a vector of pointers. Its i-th pointer points to the i-face of the flag. The second field is also a pointer vector. Its i-th component points to the i-adjacent of the flag. The third field is a pointer to form a chain of the flags. (*Fig. 1.2*) •

The flag sequences and the vector sequences play important roles in our discussion. Therefore we describe their structure.

Data 1.3. An element of a flag sequence (flag chain) is a record consisting of two fields. The first one is a pointer that points to the flag (Data 3.2), and the second one is also a pointer to form the chain of the flag sequence. (*Fig. 1.3*) •

Data 1.4. An element of a vector sequence (vector chain) is a record with two fields. The first one is a vector of integer numbers, and the second one is a pointer to form a chain of the vector sequence. (*Fig.1.3*) •

Procedure 1.1. There are two input data of this procedure. The first one is a flag $\mathbf{f}_0$ (or rather its pointer), and the second one is a subsequence

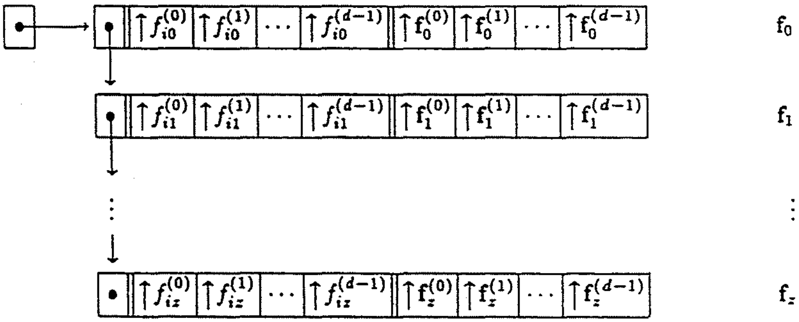

Fig. 1.2. The data structure of the flags

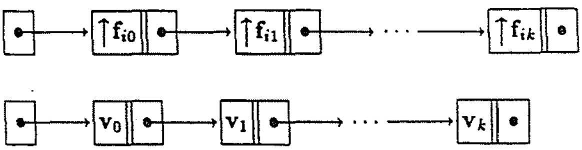

Fig. 1.3. The data structure of the flag sequence and a vector sequence

$i_0, i_1, \ldots, i_m$ from $0, 1, \ldots, (d-1)$ described by a vector with 'true' Boolean values in its i_0-th, i_1-th, $\ldots$, i_m-th components and 'false' values in the other components. The face components $f^{(i_0)}, f^{(i_1)}, \ldots, f^{(i_m)}$ of $\mathbf{f}_0$ form a face system in P. The output data will be a complete flag sequence of the system, and a vector sequence characterizing the incidence structure of the system.

The procedure works according to the following recursive algorithm. Let $j_0, j_1, \ldots, j_n$ be the complement sequence of $i_0, i_1, \ldots, i_m$ with respect to $0, 1, \ldots, (d-1)$. Let the first element of the flag sequence be $\mathbf{f}_0$, and the first vector in the vector sequence be the zero vector $\mathbf{0}$.

If the subsequence $\mathbf{f}_0, \mathbf{f}_1, \ldots, \mathbf{f}_s$ has already been given, then we get $\mathbf{f}_{s+1}$ in the following manner. Starting with $\mathbf{f}_s$, we go back in the subsequence. Let $\mathbf{f}_t$ denote the flag next in turn. We examine whether the j_0-, j_1-, $\ldots$, j_n-adjacent flags $\mathbf{f}_t^{(j_0)}, \mathbf{f}_t^{(j_1)}, \ldots, \mathbf{f}_t^{(j_n)}$ of $\mathbf{f}_t$ have already occurred in the subsequence. If $\mathbf{f}_t^{(j_l)}$ is the first flag, which has not occurred yet, in the above order, then $\mathbf{f}_{s+1} = \mathbf{f}_t^{(j_l)}$. If $t = 0$ and there is no suitable adjacent flag of $\mathbf{f}_0$, then the last element of the flag sequence is $\mathbf{f}_s$.

If the vector sequence $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_s$ has already been done and we have already found the new flag $\mathbf{f}_{s+1}$, we define the components of the vector $\mathbf{v}_{s+1}$ in the following way. We consider the i-face component $f^{(i)}_{s+1}$ of $\mathbf{f}_{s+1}$ and we examine whether it occurs as an i-face component of any flag $\mathbf{f}_t$ $(i = 0, 1, \ldots, (d-1);\quad t = 0, 1, \ldots, s)$. If so, then let the i-th component $\mathbf{v}^i_{s+1}$ of $\mathbf{v}_{s+1}$ be equal to $\mathbf{v}^i_t$, else let $\mathbf{v}^i_{s+1} = \max(\mathbf{v}^i_0, \mathbf{v}^i_1, \ldots \mathbf{v}^i_s) + 1$. •

This procedure ends because P is strictly connected (Def.1.3). Moreover, the following theorem holds (its proof can be found in [5]).

Theorem 1.1. Let $g^{(i_0)}, g^{(i_1)}, \ldots, g^{(i_m)}$ and $h^{(i_0)}, h^{(i_1)}, \ldots, h^{(i_m)}$ be $(i_0, i_1, \ldots, i_m)$ face systems, moreover, the set of their flags be $\mathbf{G}$ and $\mathbf{H}$, respectively. Let $\mathbf{g}_0 \in \mathbf{G}$ and $\mathbf{h}_0 \in \mathbf{H}$ be arbitrary flags. Using Proc.1.1 we obtain the flag sequences $\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_s$ and $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_s$, furthermore, the vector sequences $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_s$ and $\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_s$. *If the vector sequences* $\mathbf{x}_t$ *and* $\mathbf{y}_t$ *are identical, then the face systems* $g^{(i_k)}$ *and* $h^{(i_k)}$ *are isomorphic, and an isomorphism between their flags is* $\psi : \mathbf{G} \to \mathbf{H}$ *where* $\mathbf{g}^{\psi}_t = \mathbf{h}_t$ $(t = 0, 1, \ldots, s)$. •

We remark that Proc.1.1 is suitable to check the second property of Def.1.3. Namely, if $\mathbf{g}$ and $\mathbf{h}$ are different flags with common i_0-, i_1-, …, i_m-face components, starting with $\mathbf{g}$ and with the subsequence $i_0, i_1, \ldots, i_m$, Proc.1.1 yields the flag sequence $\mathbf{f}_0, \mathbf{f}_1, \ldots, \mathbf{f}_s$, where $\mathbf{f}_0 = \mathbf{g}$ (and yields a vector sequence). The flag structure of P is not strictly connected iff there is a pair $\mathbf{g}$, $\mathbf{h}$ of the flags, for which the above flag sequence $(\mathbf{f}_t)$ does not contain $\mathbf{h}$. (We have assumed that the first property of Def.1.3 is fulfilled.)

2. Isomorphism Class of Facets

By the general Def.1.4 we particularly define the isomorphism of two facets of the polyhedron.

Definition 2.1. Let $\mathbf{G}$ and $\mathbf{H}$ denote the flags of the facets g and h, respectively. We say that *the facets g and h are isomorphic* iff there is a bijective map $\varphi : \mathbf{G} \to \mathbf{H}$ preserving the 0-, 1-, …, $(d-2)$-adjacencies (φ is an isomorphism between the facets). The isomorphic facets form an *isomorphism class* of the facets. Moreover, the bijective map $\beta : \mathbf{G} \to \mathbf{G}$ is an automorphism of g iff β preserves the 0-, 1-, …, $(d-2)$-adjacencies. •

Referring to the Introduction we assume that the facets of P form only one isomorphism class denoted by c.

Let $f_0, f_1, \ldots, f_r$ and $\mathbf{F}_0, \mathbf{F}_1, \ldots, \mathbf{F}_r$ denote the facets belonging to c, and their flags, respectively. Thus, there are so-called indicating isomorphisms $\varphi_i : \mathbf{F}_0 \to \mathbf{F}_i$ $(i \in \{0, 1, \ldots, r\})$. Moreover, let $\mathcal{B} = \{\beta_0, \beta_1, \ldots, \beta_s\}$

(where $\beta_0 = 1$ is the identity element) be the automorphism group of f_0 ($\beta_j : \mathbf{F}_0 \to \mathbf{F}_0$ where $j \in \{0, 1, \ldots, s\}$).

Let $\varphi_{ij} : \mathbf{F}_i \to \mathbf{F}_j$ be any isomorphism between the facets f_i and f_j, and let us consider the automorphism $\beta_l = \varphi_i \varphi_{ij} \varphi_j^{-1}$ ($\beta_l : \mathbf{F}_0 \to \mathbf{F}_i \to \mathbf{F}_j \to \mathbf{F}_0$) of f_0. Thus, $\varphi_{ij} = \varphi_i^{-1} \beta_l \varphi_j$. It means that an arbitrary isomorphism φ_{ij} between two facets of c can be described by the indicating isomorphisms φ_i, φ_j and an automorphism β_l from $\mathcal{B}$. Especially, if $\beta : \mathbf{F}_k \to \mathbf{F}_k$ an arbitrary automorphism of the facet f_k, there exists an automorphism $\beta_j \in \mathcal{B}$ for which $\beta = \varphi_k^{-1} \beta_j \varphi_k$.

Thus, for the data structure of c we have to describe the indicating isomorphisms and the automorphism group of a distinguished facet in c. We will fix a flag sequence of this facet. Then we will create the flag sequences of the other facets in c. These sequences will be the indicating isomorphisms if the k-th element of the fixed sequence is mapped to the k-th element of the sequence of an arbitrary facet by an isomorphism for each index k. The automorphisms will be suitable permutations of the fixed sequence, similarly, and we will obtain the automorphism group of the distinguished facet as a permutation group.

Data 2.1. An isomorphism is a record consisting of two fields. The first one is a pointer to form a chain of the isomorphisms, and the second one is also a pointer that points to the flag sequence. The data structure of an automorphism is identical with an isomorphism. (*Fig.2.1*) •

Data 2.2. An isomorphism class is a record consisting of three fields. The first one is a pointer to form a chain of the classes. (This chain contains only the element c now.) The second one is a pointer that points to the chain of the automorphisms. The third one is also a pointer that points to the chain of the indicating isomorphisms. (The first automorphism and the first isomorphism of the chains have identical flag sequences of the distinguished facet.) (*Fig.2.1*) •

Using Proc.1.1 in Proc.2.1–2.2 we assume that its input subsequence is $\{d-1\}$, which contains only one element.

Procedure 2.1. The input data are a flag sequence $(\mathbf{f}_t)$ of a facet f with the corresponding vector sequence $(\mathbf{u}_t)$ created by Proc.1.1, and a facet g. The output is a chain of the isomorphisms (Data 2.1) between f and g. The isomorphism φ is given by its flag sequence $(\mathbf{g}_{\varphi t})$, where $\mathbf{f}_i \mapsto \mathbf{g}_{\varphi i}$ for each index i. The chain is empty iff f and g are not isomorphic.

Starting from a flag $\mathbf{g}_0$ of g Proc.1.1 creates the flag sequence $(\mathbf{g}_t)$ and a vector sequence. The facets f and g are not isomorphic, if the lengths of $(\mathbf{f}_t)$ and $(\mathbf{g}_t)$ are not identical, trivially, the procedure ends. (If $d \leq 3$, we can speak iff .)

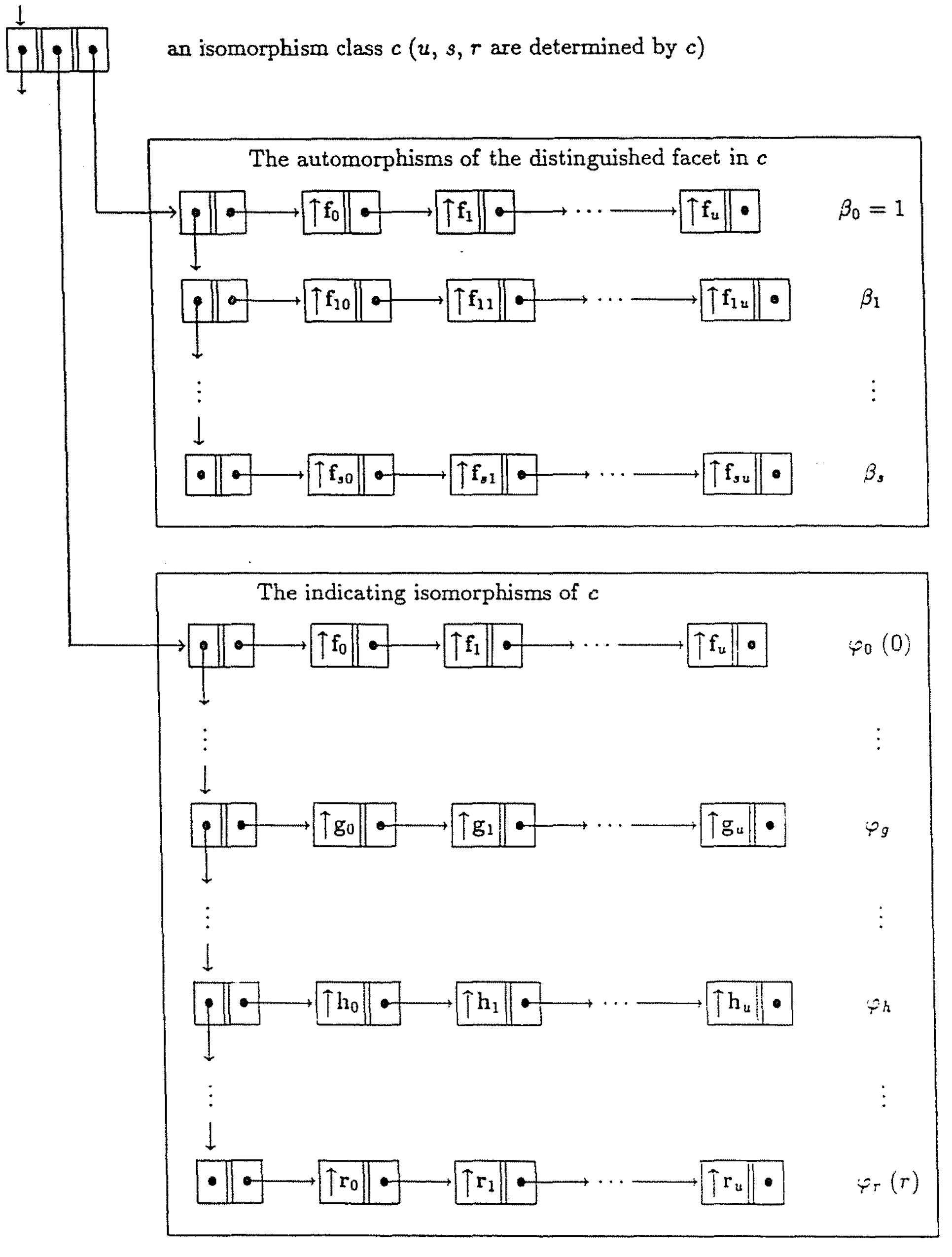

Fig. 2.1. Isomorphism classes

We consider each element $\mathbf{g}_x$ in turn in $(\mathbf{g}_t)$. Starting from $\mathbf{g}_{x0} = \mathbf{g}_x$ Proc.1.1 creates the flag sequence $(\mathbf{g}_{xt})$ and the vector sequence $(\mathbf{v}_t)$. If $(\mathbf{u}_t)$ and $(\mathbf{v}_t)$ are identical, we make an isomorphism from $(\mathbf{g}_{xt})$ and attach it to the existing ones (Theorem 1.1). (If $d \leq 3$, then each $(\mathbf{g}_{xt})$ gives an isomorphism.) •

Procedure 2.2 This procedure creates the isomorphism class c.

We choose a facet f as the distinguished facet of c. Let $\mathbf{f}_0$ denote a flag of f. Starting from $\mathbf{f}_0$ Proc.1.1 creates the flag sequence $(\mathbf{f}_t)$ and the vector sequence $(\mathbf{u}_t)$.

Starting from $(\mathbf{f}_t)$, $(\mathbf{u}_t)$, f, Proc.2.1 creates a chain of the automorphisms of f (Data 2.2). The flag sequence of the first (the identity) automorphism is $(\mathbf{f}_t)$.

The flag sequence of the first isomorphism is $(\mathbf{f}_t)$, too. We consider each facet $g \neq f$ in turn. Starting from $(\mathbf{f}_t)$, $(\mathbf{v}_t)$, g, Proc.2.1 creates the isomorphisms between f and g. We choose such an isomorphism, and attach it to the other one (Data 2.2).

Having constructed c, we fix their ordering. •

If there is a facet h that is not isomorphic with f; then there are several isomorphism classes. Starting from h instead of f, and examining the remaining facets, we obtain a new class with the distinguished facet h, similarly to c. Then we fix the ordering of the classes, too.

Now we fix the following notations. Let f be the distinguished facet, g and h be arbitrary facets of c. Their flags are $\mathbf{F}$, $\mathbf{G}$, $\mathbf{H}$, respectively. The indicating isomorphisms and their flag sequences denoted by $\varphi_0 : \mathbf{F} \rightarrow \mathbf{F}$, $\varphi_g : \mathbf{F} \rightarrow \mathbf{G}$, $\varphi_h : \mathbf{F} \rightarrow \mathbf{H}$, and $(\mathbf{f}_t)$, $(\mathbf{g}_t)$, $(\mathbf{h}_t)$, respectively. The identity automorphism is $\beta_0 = \varphi_0$. Two arbitrary automorphisms are $\beta_l, \beta_m : \mathbf{F} \rightarrow \mathbf{F}$, and their flag sequences are $(\mathbf{f}_{lt}), (\mathbf{f}_{mt})$.

Procedure 2.3. The input data are the class c, an automorphism β_l and two indicating isomorphisms φ_g and φ_h. The output will be the flag sequence $(\mathbf{h}_{lt})$ as a permutation of $(\mathbf{h}_t)$ for which $\mathbf{g}_t^{\varphi_g^{-1}\beta_l\varphi_h} = \mathbf{h}_{lt}$ holds. Thus, we obtain the isomorphism $\varphi_g^{-1}\beta_l\varphi_h : \mathbf{G} \rightarrow \mathbf{H}$.

Let $(\mathbf{f}_t)$ and $(\mathbf{f}_{lt})$ denote the flag sequences of the identity automorphism β_0 and the given automorphism β_l, respectively. We consider each flag $\mathbf{g}_t$ in turn in $(\mathbf{g}_t)$. Then, $\mathbf{f}_{lt} = \mathbf{g}^{\varphi_g{}^{-1}\beta_l}$ in $(\mathbf{f}_{lt})$. If $\mathbf{f}_{lt} = \mathbf{f}_v$ in $(\mathbf{f}_t)$, then $\mathbf{h}_v = \mathbf{g}_t{}^{\varphi_g{}^{-1}\beta_l\varphi_h}$, thus $\mathbf{h}_{lt} = \mathbf{h}_v$. •

Procedure 2.4. The input data are the isomorphism class c, and two flags $\mathbf{g} \in \mathbf{G}$, $\mathbf{h} \in \mathbf{H}$ determining an isomorphism φ between the facets g and h of c. The output data are the indicating isomorphisms φ_g, φ_h and the automorphism β_l with $\varphi_g{}^{-1}\beta_l\varphi_h = \varphi$.

In c we can find the indicating isomorphisms φ_g and φ_h containing $\mathbf{g} = \mathbf{g}_v$ and $\mathbf{h} = \mathbf{h}_w$ in their flag sequences $(\mathbf{g}_t)$ and $(\mathbf{h}_t)$, respectively. Then, $\mathbf{g} = \mathbf{f}_v^{\varphi_g}$ and $\mathbf{h} = \mathbf{f}_w^{\varphi_h}$. Thus, we have to find the automorphism β_l for which $\mathbf{f}_v^{\beta_l} = \mathbf{f}_w$, namely $\mathbf{f}_{lv} = \mathbf{f}_w$ in its flag sequence $(\mathbf{f}_{lt})$. •

The product of two isomorphisms $\varphi_i^{-1}\beta_l\varphi_j : \mathbf{F}_i \to \mathbf{F}_j$ and $\varphi_j^{-1}\beta_m\varphi_k : \mathbf{F}_j \to \mathbf{F}_k$ is $\varphi_i^{-1}\beta_l\beta_m\varphi_k : \mathbf{F}_i \to \mathbf{F}_k$. ($\mathbf{F}_i, \mathbf{F}_j, \mathbf{F}_k$ denote the flags of the facets f_i, f_j, f_k that belong to c.) In order to create this product we have to know the product of the automorphisms β_l and β_m.

Procedure 2.5. The input data are the isomorphism class c and two automorphisms β_l and β_m in it. The output is an automorphism β_n for which $\beta_n = \beta_l\beta_m$.

We can find the flag $\mathbf{f}_{l0} = \mathbf{f}_0^{\beta_l}$ in $(\mathbf{f}_t)$: $\mathbf{f}_{l0} = \mathbf{f}_v$. Then $\mathbf{f}_{mv} = \mathbf{f}_v^{\beta_m} = \mathbf{f}_{l0}^{\beta_l\beta_m}$. Thus, we have to find the automorphism β_n containing $\mathbf{f}_{mv} = \mathbf{f}_{n0}$ in its flag sequence $(\mathbf{f}_{nt})$. •

The inverse of an isomorphism $\varphi_g^{-1}\beta_l\varphi_h : \mathbf{G} \to \mathbf{H}$ is $\varphi_h^{-1}\beta_l^{-1}\varphi_g : \mathbf{H} \to \mathbf{G}$. In order to create this product we have to know the inverse of automorphisms β_l.

Procedure 2.6. The input are the isomorphism class c and an automorphism β_l in it. The output is the automorphism β_m with $\beta_m = \beta_l^{-1}$.

If we find $\mathbf{f}_{lv} = \mathbf{f}_0$ in $(\mathbf{f}_{lt})$, then $\mathbf{f}_0^{\beta_l^{-1}} = \mathbf{f}_v$. Thus we have to find β_m containing $\mathbf{f}_v = \mathbf{f}_{m0}$ in its flag sequence $(\mathbf{f}_{mt})$. •

3. Facet-Mappings

Definition 3.1. A bijective map of the flag structure onto itself $\delta : \mathbf{F} \to \mathbf{F}$ is called a facet-mapping if it preserves the 0-, 1-, ..., $(d-2)$-adjacencies. Thus, the restriction of a facet-mapping to the flags of any facet is an isomorphism. Let $\mathcal{D}$ denote the group of the facet-mappings of P. •

Let us consider a facet-mapping δ. Let $f_0, f_1, \dots, f_r$ and $\mathbf{F}_0, \mathbf{F}_1, \dots, \mathbf{F}_r$ denote the facets and their flags that belong to the isomorphism class c in this fixed ordering (Data 2.2; Proc.2.2). The distinguished facet is f_0. The indicating isomorphisms are $\varphi_0, \varphi_1, \dots, \varphi_r$ ($\varphi_i : \mathbf{F}_0 \to \mathbf{F}_i$; $i = 0, 1, \dots, r$), and the automorphisms of f_0 are $\beta_0, \beta_1, \dots, \beta_s$. Then $\delta|_{\mathbf{F}_k} = \varphi_k^{-1}\beta_{j_k}\varphi_{i_k} : \mathbf{F}_k \to \mathbf{F}_{i_k}$ for certain β_{j_k} $(k = 0, 1, \dots, r)$. Thus, δ creates the permutation $\varphi_{i_0}, \varphi_{i_1}, \dots, \varphi_{i_r}$ of the indicating isomorphisms $\varphi_0, \varphi_1, \dots, \varphi_r$ and the variation $\beta_{j_0}, \beta_{j_1}, \dots, \beta_{j_r}$ of the automorphisms $\beta_0, \beta_1, \dots, \beta_s$. (An ordered pair $(\varphi_{i_k}, \beta_{j_k})$ will be called a facet-isomorphism in Data 3.1) We shall use the notation

$$\begin{bmatrix} \varphi_{i_0} & \varphi_{i_1} & \cdots & \varphi_{i_r} \\ \beta_{j_0} & \beta_{j_1} & \cdots & \beta_{j_r} \end{bmatrix}$$

for the restriction $\delta|_{\mathbf{F}_0 \cup \mathbf{F}_1 \cup \ldots \cup \mathbf{F}_r}$ (which will be called a class-automorphism in Data 3.2) to the flags of the facets belonging to the class. In order to give a facet-mapping, we have to make a permutation of the isomorphisms and a variation of the automorphisms in c. This is a concise description of the flag permutation that is given by a facet-mapping.

We consider the facet-mappings δ_1 and δ_2,

$$\delta_{1c} = \begin{bmatrix} \varphi_{i_0} & \varphi_{i_1} & \cdots & \varphi_{i_r} \\ \beta_{j_0} & \beta_{j_1} & \cdots & \beta_{j_r} \end{bmatrix}_c \quad \text{and} \quad \delta_{2c} = \begin{bmatrix} \varphi_{x_0} & \varphi_{x_1} & \cdots & \varphi_{x_r} \\ \beta_{y_0} & \beta_{y_1} & \cdots & \beta_{y_r} \end{bmatrix}_c.$$

We look for their product

$$\zeta_c = \begin{bmatrix} \varphi_{v_0} & \varphi_{v_1} & \cdots & \varphi_{v_r} \\ \beta_{w_0} & \beta_{w_1} & \cdots & \beta_{w_r} \end{bmatrix}_c.$$

(The index c shows that the permutation matrices are given independently in each class.) We can see that $\delta_1|_{\mathbf{F}_k} = \varphi_k^{-1}\beta_{j_k}\varphi_{i_k} : \mathbf{F}_k \to \mathbf{F}_{i_k}$ and $\delta_2|_{\mathbf{F}_{i_k}} = \varphi_{i_k}^{-1}\beta_{y_{i_k}}\varphi_{x_{i_k}} : \mathbf{F}_{i_k} \to \mathbf{F}_{x_{i_k}}$. Thus, $\delta_1\delta_2|_{\mathbf{F}_k} = \varphi_k^{-1}\beta_{j_k}\beta_{y_{i_k}}\varphi_{x_{i_k}} : \mathbf{F}_k \to \mathbf{F}_{x_{i_k}}$. Therefore $\varphi_{v_k} = \varphi_{x_{i_k}}$ and $\beta_{w_k} = \beta_{j_k}\beta_{y_{i_k}}$. Using this matrix form we obtain that

$$\zeta_c = \delta_{1c}\delta_{2c} = \begin{bmatrix} \varphi_{x_{i_0}} & \varphi_{x_{i_1}} & \cdots & \varphi_{x_{i_r}} \\ \beta_{j_0}\beta_{y_{i_0}} & \beta_{j_1}\beta_{y_{i_1}} & \cdots & \beta_{j_r}\beta_{y_{i_r}} \end{bmatrix}_c.$$

It is easy to see that the identity element ε and the inverse of an element δ_1 in matrix form are

$$\varepsilon_c = \begin{bmatrix} \varphi_0 & \varphi_1 & \cdots & \varphi_r \\ \beta_0 & \beta_0 & \cdots & \beta_0 \end{bmatrix}_c \qquad \text{and} \qquad \delta_{1c}^{-1} = \begin{bmatrix} \varphi_{p_0} & \varphi_{p_1} & \cdots & \varphi_{p_r} \\ \beta_{q_0} & \beta_{q_1} & \cdots & \beta_{q_r} \end{bmatrix}_c,$$

where $\varphi_{p_{i_k}} = \varphi_k$ and $\beta_{q_{i_k}} = \beta_{j_k}^{-1}$.

A facet-mapping δ_1 is involutive (identical with its inverse) iff $\varphi_{i_{i_k}} = \varphi_k$ and $\beta_{j_{i_k}} = \beta_{j_k}^{-1}$ for all $k \in \{0, 1, \ldots, r\}$ and in each class. The first property shows that the permutation of the indicating isomorphisms is involutive (in other words, it makes a pairing), and the second one shows the connection of the automorphisms belonging to the paired isomorphisms.

Now we define the data structure of a facet-mapping.

Data 3.1. A facet-isomorphism is a record containing three fields. The first and the second fields are pointers. The first one points to an isomorphism

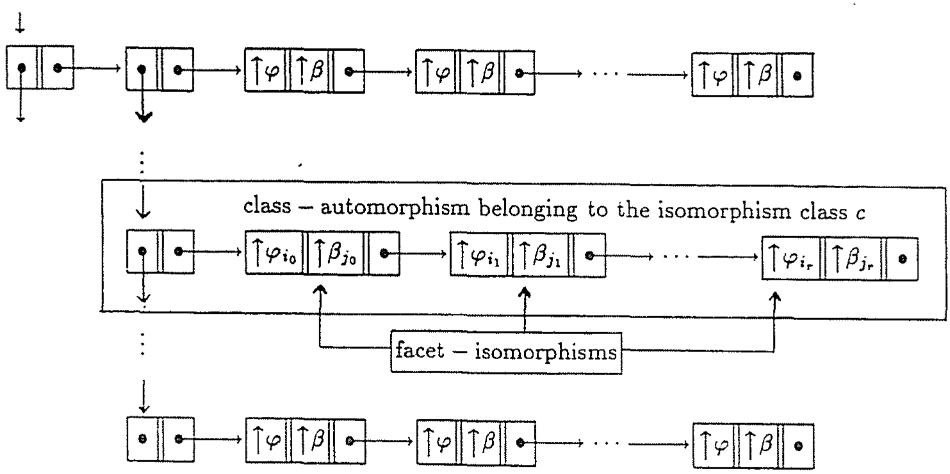

Fig. 3.1. A facet-mapping

(Data 2.1), and the second one points to an automorphism (Data 2.2). The third field is also a pointer to form a chain of the facet-isomorphisms in a class. (Knowing the place of a facet-isomorphism in the chain, we obtain the map as written above.) (*Fig.3.1*) •

Data 3.2. A class-automorphism is a record containing two fields. The first field is a pointer to form a chain of the class-automorphisms. (This chain contains only one element belonging to c now.) The second one is also a pointer to the chain of the facet-isomorphisms. (The isomorphism class referred to is given by the place in the chain, because the ordering of the isomorphism classes is fixed by Proc.2.2.) (*Fig. 3.1*) •

Data 3.3. A facet-mapping is a record containing two fields. The first one is a pointer to form a chain of the facet-mappings. The second one is also a pointer to the chain of the class-automorphisms. (*Fig.3.1*) •

Procedure 3.1. The input data are two facet-mappings δ_1, δ_2 and the output is their product $\zeta = \delta_1\delta_2$. (We shall use the notation above.)

Let us consider the isomorphism class c and the corresponding class-automorphisms δ_{1c} and δ_{2c}. We create their product as follows.

We consider each facet-isomorphism $(\varphi_{i_k}, \beta_{j_k})$; (it is the k-th one) in turn in δ_{1c}. If φ_{i_k} is the t-th indicating isomorphism of c ($t = i_k$), then we consider the facet isomorphism $(\varphi_{x_t}, \beta_{y_t})$ of δ_{2c}. Proc.2.5 creates the product $\beta_{j_k}\beta_{y_t}$. Thus, we obtain the k-th facet-isomorphism $(\varphi_{v_k}, \beta_{w_k})$ of the product ζ_c, where $\varphi_{v_k} = \varphi_{x_t}$ and $\beta_{w_k} = \beta_{j_k}\beta_{y_t}$. •

Procedure 3.2. The input is a facet-mapping δ_1 and the output is its inverse δ_1^{-1}.

Let us consider the isomorphism class c and the corresponding class-automorphism δ_{1c}. We create its inverse as follows.

We consider each indicating isomorphism φ_k in c. It can be found in the t-th facet-isomorphism $(\varphi_{i_t}, \beta_{j_t};\ \varphi_{i_t} = \varphi_k,\ i_t = k)$ of δ_{1c}. Proc.2.6 creates the inverse of β_{j_t}. Thus, we obtain the k-th facet-isomorphism $(\varphi_{p_k}, \beta_{q_k})$ of δ_{1c}^{-1}, where $\varphi_{p_k} = \varphi_t$ and $\beta_{q_k} = \beta_{j_t}^{-1}$. •

Procedure 3.3. The input is a facet-mapping δ_1. The output data are two flag sequences (Data 1.3.) $\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_z$ and $\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_z$, where $\mathbf{g}_t^{\delta_1} = \mathbf{h}_t$ for each $t \in \{0, 1, \ldots, z\}$.

At the beginning the flag sequences $\mathbf{g}_t$ and $\mathbf{h}_t$ are empty. We consider the isomorphism class c and the corresponding class-automorphism δ_{1c}. We take each indicating isomorphism φ_k of c and the corresponding facet-isomorphism $(\varphi_{i_k}, \beta_{j_k})$ of δ_{1c} in turn. Let $(\mathbf{g}_{kv})$ denote the flag sequence of φ_k. Starting from c, β_{j_k}, φ_k and φ_{i_k}, Proc.2.3 creates the flag sequence $(\mathbf{h}_{kv})$. Thus, $\mathbf{g}_{kv}^{\varphi_k^{-1}\beta_{j_k}\varphi_{i_k}} = \mathbf{h}_{kv}$. Therefore we attach $(\mathbf{g}_{kv})$ to $(\mathbf{g}_t)$, and attach $(\mathbf{h}_{kv})$ to $(\mathbf{h}_t)$. •

Procedure 3.4. The input data are two flag sequences $(\mathbf{g}_t)$ and $(\mathbf{h}_t)$ as permutations of the flags of P defining a facet-mapping δ_1 ($\mathbf{g}_t^{\delta_1} = \mathbf{h}_t$, $t = 0, 1, \ldots, z$). The output is δ_1 using the data structures Data 3.1–3.3.

We consider each indicating isomorphism φ_k in turn in c. Let $\mathbf{f}$ denote an element of the flag seqence of φ_k. If $\mathbf{f} = \mathbf{g}_v$ in $(\mathbf{g}_t)$, then starting from c, $\mathbf{g}_v$ and $\mathbf{h}_v$ Proc.2.4 gives the indicating isomorphisms φ_k, φ_{i_k} and the automorphism β_{j_k} ($\varphi_k^{-1}\beta_{j_k}\varphi_{i_k} = \delta_1|_{\mathbf{D}}$, where $\mathbf{D}$ is the domain of φ_k). Thus we obtain the k-th facet isomorphism $(\varphi_{i_k}, \beta_{j_k})$ of δ_{1c}. •

4. Automorphisms

Definition 4.1. A bijective map $\alpha : \mathbf{F} \to \mathbf{F}$ is called an *automorphism* of the polyhedron P if it preserves all of the adjacencies. Let $\mathcal{A}$ denote the automorphism group of P. It is easy to see that $\mathcal{A} \le \mathcal{D}$. •

Definition 4.2. Let $\delta_1, \delta_2 \in \mathcal{D}$ be facet-mappings. We say that δ_1 and δ_2 are *equivalent* (or essentially non different) if there exists an (equivariant) automorphism $\alpha \in \mathcal{A}$ of P so that $(\mathbf{f}^\alpha)^{\delta_2} = (\mathbf{f}^{\delta_1})^\alpha$ for any flag $\mathbf{f} \in \mathbf{F}$, i.e. $\delta_2 = \alpha^{-1}\delta_1\alpha$. •

Procedure 4.1. This procedure creates the automorphisms of P as facet-mappings and makes their chain. Using Proc.1.1 in this procedure, we assume that the input subsequence is $\emptyset$ (it is empty).

At the beginning the chain is empty. We choose a flag $\mathbf{g}$ of P (in Data 1.2). Starting from $\mathbf{g}_0 = \mathbf{g}$, Proc.1.1 creates the flag sequence $(\mathbf{g}_t)$, which contains all flags of P, and the vector sequence $(\mathbf{u}_t)$.

Then we consider each element $\mathbf{g}_v$ of the flag sequence $(\mathbf{g}_t)$ in turn. Starting from $\mathbf{h}_0 = \mathbf{g}_v$, Proc.1.1 creates the flag sequence $(\mathbf{h}_t)$ as a permutation of $(\mathbf{g}_t)$ and the vector sequence $(\mathbf{v}_t)$. If the vector sequences $(\mathbf{u}_t)$ and $(\mathbf{v}_t)$ are identical, then the map by $\mathbf{g}_t \mapsto \mathbf{h}_t$ is an automorphism α of P. Starting from $(\mathbf{g}_t)$ and $(\mathbf{h}_t)$, Proc.3.4 creates α as a facet-mapping (Data 3.1–3.3).

If an automorphism of P as a facet-mapping has been made, we link it to the others by a pointer. Finally, we obtain each element of the automorphism group of P. •

Procedure 4.2. The input data are two facet-mappings δ_1 and δ_2. The output is a Boolean value, which shows whether the facet-mappings are equivalent.

We consider each automorphism α of P in turn. Using Proc.3.1 and Proc.3.2, we obtain the product facet-mapping $\alpha^{-1}\delta_1\alpha$. If there exists an automorphism for which this product is equal to δ_2, then the procedure ends with the value 'True', else the procedure gives 'False'. •

5. Generator Systems and Facet-Identifications

If the polyhedron P is a fundamental domain of a space group, then the transformations, mapping P onto its neighbours along its facets, form a generator system of the group. Let $\tilde{\varphi}$ be such a transformation. $\tilde{\varphi}$ maps P onto its neighbour along its facet f mapping the facet f^{-1} of P onto f. At $\tilde{\varphi}$ the flags of f^{-1} are mapped onto the flags of f by an isomorphism φ. The transformation $\tilde{\varphi}$ determines the isomorphism φ. Similarly, the transformation $\tilde{\varphi}^{-1}$ maps P onto its neighbour along its facet f^{-1} mapping f onto f^{-1}. The mapping of the flags is given by φ^{-1}. If $f = f^{-1}$ namely, if $\tilde{\varphi}$ maps P onto its neighbour along its facet f mapping f onto itself, then $\tilde{\varphi}^{-1} = \tilde{\varphi}$ and $\varphi^{-1} = \varphi$ (φ is an involutive automorphism of f or its identity).

Thus, a generator system can be determined combinatorially as follows. First, we make a pairing of the facets of P so that the following two properties are fulfilled.

1. If two facets are in the same pair, then these facets are isomorphic. (A facet may be paired with itself.)
2. Every facet occurs in exactly one pair.

Second, to each pair $[f^{-1}, f]$ of facets we correspond a pair of isomorphisms as follows. Let $\mathbf{F}^{-1}$ and $\mathbf{F}$ denote the flags of the facets, respectively. We choose an isomorphism $\varphi : \mathbf{F}^{-1} \to \mathbf{F}$ and make the pair $[\varphi, \varphi^{-1}]$. If a pair $[f, f]$ occurs, then the corresponding pair is $[\varphi, \varphi]$ with an arbitrary involutive automorphism $\varphi : \mathbf{F} \to \mathbf{F}$ of f.

The map whose restrictions to the flags of the facets are the above isomorphisms is an involutive facet-mapping (Def.3.1), so-called *facet-identification*. Thus, a generator system determines a facet-identification (or facet pairing). Conversely, it is obvious that a facet-identification determines a generator system if we form its restriction to the flags of each facet.

Definition 5.1. *Two generator systems are equivalent* iff the corresponding facet-identifications are equivalent (Def.4.2). •

Procedure 5.1. This procedure creates exactly one element from each equivalence class of the facet-identifications, and makes their chain.

At the beginning the chain is empty. We create the facet-identifications in turn. The first one comes simply to the chain. Then using Proc.4.2, we examine whether there is an equivalent element of the chain with the facet-identification next in turn. If not, then we attach it to the chain.

In order to enumerate the facet-identifications, we consider the involutive permutations of the indicating isomorphisms $\varphi_0, \varphi_1, \ldots, \varphi_r$ in turn in c. (It is sufficient to consider all non-equivalent involutive permutations. Two permutations π_1 and π_2 are equivalent iff there is an automorphism α of P for which $\pi_1 = \pi^{-1}(\alpha)\pi_2\pi(\alpha)$, where $\pi(\alpha)$ denotes the permutation of the isomorphisms in the facet-mapping α. Moreover, we can apply the following observation. If two facet-mappings δ_1 and δ_2 are equivalent, then $\pi\ (\delta_1)$ and $\pi\ (\delta_2)$ are also equivalent.)

Let us consider a fixed involutive permutation $(\varphi_{i_0}, \varphi_{i_1}, \ldots, \varphi_{i_r})$ determining a pairing $[\varphi_{i_k}, \varphi_k]$ of the isomorphisms of c. We construct the variations $\beta_{j_0}, \beta_{j_1}, \ldots, \beta_{j_s}$ of the automorphisms $\beta_0, \beta_1, \ldots, \beta_s$ as follows. We make the pairs $[\beta_j, \beta_j^{-1}]$, and add them independently in turn to each pair $[\varphi_{i_k}, \varphi_k]$. Thus we obtain the facet-isomorphisms $(\varphi_{i_k}, \beta_{j_k})$ and $(\varphi_k, \beta_{j_k}^{-1})$ from $[\varphi_{i_k}, \varphi_k]$ and $[\beta_{j_k}, \beta_{j_k}^{-1}]$. If a pair $[\varphi_{i_k}, \varphi_{i_k}]$ contains identical isomorphisms ($i_k = k$), then we can add to it only the pairs $[\beta_j, \beta_j]$ where $\beta_j^{-1} = \beta_j$. Thus, we obtain only one facet-isomorphism $(\varphi_{i_k}, \beta_{j_k})$. •

6. The Poincaré Algorithm

If the polyhedron P and a generator system are given, then we look for the defining relations to determine the space group. For this purpose we shall use the Poincaré algorithm [3,5,9]. This algorithm is based on the following ideas.

Let $\tilde{\Phi}$ denote the set of the elements of a generator system ($|\tilde{\Phi}| = |F|$). $\tilde{\varphi}_i \in \tilde{\Phi}$ maps the facet $f_i^{-1} = f_j$ onto f_i, and maps P onto the adjacent polyhedron $P^{\tilde{\varphi}_i}$ along f_i. Moreover, $\tilde{\varphi}_j = \tilde{\varphi}_i^{-1}$ maps $f_j^{-1} = f_i$ onto f_j, and maps P onto the adjacent polyhedron $P^{\tilde{\varphi}_j}$ along f_j. We choose an edge e (a

$(d-2)$-face) of P. Let us consider the images of P (at the space group) that are incident with e. These polyhedra are adjacent along facets. These facets are incident with e. Let us go round e considered. The generator $\tilde{\varphi}$ maps P its adjacent polyhedron $P^{\tilde{\varphi}_1}$ along f_1. The facet $f_2^{\tilde{\varphi}_1}$ of $P^{\tilde{\varphi}_1}$ is incident with e, and it is not identical with f_1. The transformation $\tilde{\varphi}_1^{-1}\tilde{\varphi}_2\tilde{\varphi}_1$ maps $P^{\tilde{\varphi}_1}$ onto its adjacent polyhedron $P^{\tilde{\varphi}_2\tilde{\varphi}_1}$ along $f_2^{\tilde{\varphi}_1}$, and so on. Finally, we obtain the polyhedron $P^{\tilde{\varphi}_n\tilde{\varphi}_{n-1}\cdots\tilde{\varphi}_1}$ that is identical with $P = P^1$, and the relation $\tilde{\varphi}_n\tilde{\varphi}_{n-1}\cdots\tilde{\varphi}_1 = 1$. Starting with an edge, which has not mapped onto e yet, we get a new relation, and so on. The combinatorial algorithm works in the following (inverse) manner (see [5] for more details).

Procedure 6.1. The input is a facet-identification δ. This procedure prints the generators that are determined by δ, and prints the cycle transformations, which will be the defining relations with certain natural exponents.

Starting from δ, Proc.3.3 creates the flag sequences $(\mathbf{g}_t)$ and $(\mathbf{h}_t)$ $(\mathbf{g}_t^\delta = \mathbf{h}_t)$.

We consider a flag $\mathbf{f}$ of each facet f of P in turn. If $\mathbf{f} = \mathbf{g}_j$, then we print the combinatorial generator

$$\varphi_i : \mathbf{g}_j \mapsto \mathbf{h}_j.$$

(The isomorphism φ_i is the restriction of δ to the flags of f).

We choose an edge e_1 from the set E (of the $(d-2)$-faces). Let $\mathbf{f}_1$ be an arbitrary flag of e_1. We create the flag sequence $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n$ in the following manner. If $\mathbf{f}_v = \mathbf{g}_t$ in $(\mathbf{g}_t)$ then $\mathbf{f}_{v+1}$ is the facet adjacent ($(d-1)$-adjacent) flag of $\mathbf{h}_t$ $(\mathbf{h}_t = \mathbf{f}_v^\delta)$. The last flag $\mathbf{f}_n$ is given if $\mathbf{f}_{n+1} = \mathbf{f}_1$. Let $f_1, f_2, \ldots, f_n$ and $e_1, e_2, \ldots, e_n$ denote the facet components and the edge components of the flags $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n$ respectively. Then the procedure prints the cycle transformation

$$\sigma_1 = (\varphi_1\varphi_2\cdots\varphi_n),$$

where φ_u is the restriction of δ to the flags of f_u.

Now we consider the set $E \setminus \{e_1, e_2, \ldots, e_n\}$ and choose an edge from it. Starting with this edge, we obtain the second cycle transformation σ_2, and so on. When the difference set $E \setminus \{\textit{the occurred edges}\}$ is empty, then the procedure ends, we have obtained each cycle transformation. •

Considering the equations $\sigma_i^{k_i} = 1$ with arbitrary natural exponents k_i, we obtain the defining relations of a space group Γ. If the restriction of δ to the flags of some facets $f_v, f_w, \ldots, f_z$ are involutive automorphisms, then the relations $\varphi_v^2 = 1,\ \varphi_w^2 = 1,\ \ldots,\ \varphi_z^2 = 1$ must be added [1,3].

In general, Γ is not yet a discrete group of any space of constant curvature. This will depend on the exponents k_i. However, a combinatorial simply connected space, denoted by P^Γ, can be given. The group Γ acts on this space, and P is a fundamental domain of Γ [4,5,8].

7. Results of the Implementation

Let SI_d denote the number of the essentially different facet pairings (combinatorial d-simplex tilings). It is trivial that $\mathrm{SI}_1 = 2 = 2^1$. It may be checked that $\mathrm{SI}_2 = 8 = 2^3$ (triangle tilings). The 3-dimensional case (the tetrahedron) was examined by I. K. ZHUK, and the results were published in [10,11]. He found that $\mathrm{SI}_3 = 64 = 2^6$. Our program also verified this result in 1988 [6]. Zhuk examined the realizations of these cases in the Euclidean and hyperbolic space. The complete enumeration in the spaces of constant curvatutre was accomplished by Emil MOLNÁR [7].

Knowing the results $\mathrm{SI}_1 = 2$, $\mathrm{SI}_2 = 8$, $\mathrm{SI}_3 = 64$, Zhuk conjectured that $\mathrm{SI}_d = 2^d \mathrm{SI}_{d-1}$ where $\mathrm{SI}_0 := 1$ and $1 \leq d$ [11]. So SI_4 would be $1024 = 2^{10}$. Howeover, our implemented program has given the result $\mathrm{SI}_4 = 4096 = 2^{12}$ disproving Zhuk's conjecture. But SI_d still seems to be a power of 2. Now we are working to find the correct formula for SI_d if it is possible.

We examined the cube of the Euclidean 3-space by our program. This examination is based on the consideration that there are four solids surrounding each edge in the cubic tilings. Thus, modifying Proc.5.1, we look for the essentially different involutive face-mappings that give only cycle transformations with word length either 1 or 2 or 4 by Proc.6.1. We found 298 such essentially different face pairings. Consequently, there are 298 fundamental tilings with marked cubes. Each tiling determines a crystallographic group, which occurs among the 219 non-isomorphic groups enumerated in [2], by its generator system and defining relations. We have found this group for each tiling, and obtained that there are 130 crystallographic groups with cubic fundamental domain [8].

References

1. COXETER, H. S. M. — MOSER, W. O. J.: Generators and Relations for Discrete Groups. 4th edition, Ergeb. der Math., Neue Folge, Bd. 14., Springer-Verlag, Berlin — Heidelberg — New York, 1980.
2. HENRY, N. F. M. — LONSDALE, K.: *Symmetry groups. International Tables for X-Ray Crystallography.* Vol. I. Kynoch Press, Birmingham, 1969. New edition: Vol. A. edited by Theo Hahn, Reidel, Dordrecht, 1983.
3. MASKIT, B.: On Poincaré Theorem for Fundamental Polygons. *Adv. in Math.* Vol. 7 (1971), pp. 219–230.

4. MOLNÁR, E.: Minimal Presentation of the 10 Compact Euclidean Space Forms by Fundamental Domains. *Studia Sci. Math. Hung.* Vol. 22 (1987), pp. 19–51.
5. MOLNÁR, E.: Polyhedron Complexes with Simply Transitive Group Actions and their Realizations. *Acta Math. Hung.* Vol. 59 (1-2) (1992), pp. 175–216.
6. MOLNÁR, E.: – PROK, I.: A Polyhedron Algorithm for Finding Space Groups. *Proc. of Third Int. Conf. on Engineering Graphics and Descriptive Geometry*, Vienna 1988, Vol. 2, pp. 37–44.
7. MOLNÁR, E.: – PROK, I.: Classification of Solid Transitive Simplex Tilings in Simply Connected 3-spaces. I. The combinatorial description by figures and tables, results in spaces of constant curvature. Preprint 92–003 Univ. Bielefeld SFB 343 'Diskrete Strukturen in der Mathematik' *Colloquia Math. Soc. J. Bolyai 63. Intuitive Geometry,* Szeged (Hungary) 1991, North-Holland Co. Amsterdam – Oxford – New York, pp.
8. PROK, I.: The Euclidean Space Has 298 Fundamental Tilings with Marked Cubes by 130 Space Groups. *Colloquia Math. Soc. J. Bolyai 63. Intuitive Geometry,* Szeged (Hungary) 1991, North-Holland Co. Amsterdam – Oxford – New York, pp.
9. VINBERG, E. B. — SHVARTSMAN, O. V.: Discrete Transformation Groups of Spaces of Constant Curvature. In: *Geometriya* 2, *VINITI, Itogi Nauki i Tekhniki, Sovr. Probl. Math. Fund. Napr.* Vol 29 (1988), pp. 147–259 (in Russian).
10. ZHUK, I. K.: Fundamental Tetrahedra in Euclidean and Lobachevsky Spaces. *Soviet Math. Dokl.* Vol 27 (1983), pp. 540–543.
11. ZHUK, I. K. : Regular Decompositions of Spaces of Constant Curvature by Congruent Tetrahedra I. (1980) II. (1983). *Preprints of Math. Inst. of AN Belorussian SSR*, Minsk (in Russian).

Address:

István PROK
Department of Geometry
Faculty of Mechanical Engineering
Technical University of Budapest
H–1521 Budapest, Hungary