A 798
A 112

Representation and Generation of Finite Partially Ordered Sets

E. D. Cooper

Contents

§0. Preliminaries -1-

§1. Po-diagrams -2-

§2. Po-strings -3-

§3. Representation -6-

§4. Generation -8-

§0. Preliminaries

Let R be a relation in a set X. xRy means $(x,y) \in R$. If $A,B \subseteq X$ write ARB if xRy for all $x \in A$, $y \in B$. A chain of length $n \geqslant 0$ is a sequence $(x_o,\ldots,x_n) \in X^{n+1}$ such that $x_\nu R x_{\nu+1}$, $\nu < n$; write $x_o R \ldots R x_n$. If $n = 0$ the chain reduces to an element of X; if $n = 1$ the chain is an element of R. Say R is strongly irreflexive if $x_o R \ldots R x_n = x_o$ implies $n = 0$, i.e., there are no "loops". For a strongly irreflexive relation R there is a depth function $d^R : X \longrightarrow \omega \cup \{\infty\}$ defined by

$$d^R(x) = \sup \{ a \mid \exists\, x_o R \ldots R x_a = x \}$$

and $d^R(x) = \infty$ if there are chains of arbitrary length terminating at x. Let $L_a = L_a(R)$ denote the a^{th} level, the set of elements of depth a; let ℓ_a denote the cardinality of L_a, and let d be the largest integer such that $L_a \neq \emptyset$, setting $d = \infty$ if there is an element at every level. The following facts are easy to verify.

(0.0) if $x_o R \ldots R x_a = x$ then $d^R(x) = a$ iff $d^R(x_\nu) = \nu$, $\nu \leq a$;

(0.1) card $X < \infty \Rightarrow d < \infty$;

(0.2) $a \neq b \Rightarrow L_a \cap L_b = \emptyset$;

(0.3) card $X = n < \infty \Rightarrow n = \sum_{a \leq d} \ell_a$;

(0.4) if $x \in L_a$ there is a chain $x_o R \ldots R x_a = x$ such that $x_\nu \in L_\nu$, $\nu \leq a$;

(0.5) $d^R(x) < \infty$, $xRy \Rightarrow d^R(x) < d^R(y)$.

A partially ordered set, or poset, is a pair $P = (X, \leq)$

with X a set and $\leq$ a relfexive, transitive and anti-symmetric relation in X, i.e., for all $x,y,z \in X$

$$x \leq x$$
$$x \leq y \leq z \Rightarrow x \leq z$$
$$x \leq y \leq x \Rightarrow x = y \quad .$$

P is <u>finite</u> if its underlying set X of <u>nodes</u> is finite; card $X = n \leq \infty$ is the <u>order</u> of P. A strongly reflexive relation C in X is defined by xCy if and only if

$$x < y \text{ and } (x \leq u \leq y \Rightarrow x = u \text{ or } u = y) \ .$$

C is the <u>co-cover relation</u> of P; the elements of C are co-covers.

<u>Lemma (0,6)</u> Let P be a poset with co-cover relation C. The following hold.

(i) $x_0C\ldots Cx_r$, $r \geq 1$ and x_0Cx_r imply $r = 1$;

(ii) if P is of finite order and $x < y$ then there is a chain $x = x_0C\ldots Cx_r = y$, $r \geq 1$. #

If $P' = (X', \leq')$ is also a poset a morphism from P to P' is a function $f:X \to X'$ such that $x \leq y \Rightarrow f(x) \leq f(y)$. f is an isomorphism if there is a morphism g from P' to P such that $gf = 1_X$ and $fg = 1_{X'}$. Two posets are isomorphic if there is an isomorphism between them. The relation of isomorphism is an equivalence relation.

§1. <u>Po-diagrams</u>

A <u>po-diagram</u> H = (X,C) consists of a set X and a strongly irreflexive relation C in X such that

$$x_0C\ldots Cx_r,\ r \geq 1 \text{ and } x_0Cx_r \text{ imply } r = 1 .$$

An <u>isomorphism</u> of H with a po-diagram $H' = (X',C')$ is given by a bijection $h:X \approx X'$ such that

$$xCy \Leftrightarrow h(x)C'h(y) \quad .$$

<u>Construction (1.0)</u> Let $P = (X, \leq)$ be a poset with co-cover relation C. Then $H(P) = (X,C)$ is a po-diagram.

<u>Lemma (1.2)</u> Suppose $f:P \cong P'$. Then $f:H(P) \cong H(P')$. #

<u>Construction (1.3)</u> Let $H = (X,C)$ be a po-diagram. Define a relation $\leq$ in X by $x \leq y$ if and only if

$$\text{there is a chain } x = x_0C\ldots Cx_r = y,\ r \geq 0 .$$

It is easy to prove that $P(H) = (X, \leq)$ is a poset.

<u>Lemma (1.4)</u> Suppose $h:H \cong H'$. Then $h:P(H) \cong P(H')$. #

<u>Theorem (1.5)</u> $H(P(H)) = H$ and $P(H(P)) = P$. #

§2. <u>Po-strings</u>

A concatenation $s_0s_1\ldots s_{n-1}$ of symbols is a <u>list</u> of length $n \geq 0$ with i^{th} entry s_i, $i < n$. If $n = 0$ there are no entries and the list is empty. If the entries are themselves lists, empty or not, then they shall be enclosed in parentheses to mark their beginnings and endings.

A <u>string</u> of <u>order</u> $n > 0$ is a list whose i^{th} entry is either the empty list or a list of finite ordinals of the form

$$e_i = e_{i1}\dots e_{in_i},\ n_i \geqslant 1,\ i < e_{i1} < \dots < e_{in_i} < n\ .$$

Note that e_{n-1} must be empty. Any string may be obtained from the string

$$(12\dots n-1)(23\dots n-1)\dots(n-2\ n-1)(n-1)(\ )$$

by deleting entries. In particular, the <u>empty string</u> ( )( )...( )( )( ) is obtained by deleting all entries.

Let e be a string. A strongly irreflexive relation C_e is defined in $n = \{0,1,\dots,n-1\}$ by iC_ek if and only if

$$\text{there is } j,\ 1 \leq j \leq n_i \text{ such that } e_{ij} = k\ .$$

Write d^e for the depth function of C_e. Thus $d^e(i) = 0$ means i occurs in none of the lists of e. A <u>po-string</u> is a string e with the properties

(PS1) $d^e(i) < d^e(j) \Rightarrow i < j$

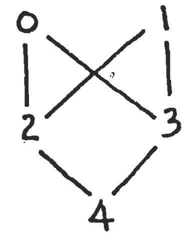
(PS2) $i_oC_e\dots C_ei_r$, $r \geqslant 1$ and $i_oC_ei_r$ imply $r = 1$.

<u>Examples (2.0)</u> (13)(2)( )( ) is a string of order 4 that fails to satisfy (PS1), and (1)(24)(3)(4)( ) is a string of order 5 which satisfies (PS1) but fails to satisfy (PS2).

<u>Construction (2.1)</u> Let e be a po-string.

Then $H(e) = (n,C_e)$ is a finite po-diagram.

<u>Example (2.2)</u> From (23)(23)(4)(4)( ) (2.1) gives



Construction (2.3) Manufacture of a string from a po-diagram $H = (X,C)$ of finite order n involves choice. It is necessary to "label" the nodes of H in the following sense. A labelling for H is an order-preserving bijection $\lambda : X \approx n$, i.e., $xCy \Rightarrow \lambda(x) < \lambda(y)$. Hence λ is an ordinal sum of bijections $\lambda_a : L_a \approx \ell_a$, $a \leq d$ such that for $x \in L_a$

$$\lambda(x) = \begin{cases} \lambda_a(x) \text{ if } a = 0 \\ \ell_0 + \ldots + \ell_{a-1} + \lambda_a(x) \text{ if } a > 0 . \end{cases}$$

Call $\lambda(x)$ the label of x by λ. The finite po-diagram H together with a choice of labelling λ is a labelled po-diagram, denoted by H_λ. Every finite po-diagram has a labelling, indeed, H has exactly $\ell_0! \ell_1! \ldots \ell_d!$ distinct labellings.

Construct a list $e = e(H_\lambda)$ of lists by taking for e_i the arrangement in ascending order from left to right of all the labels of nodes y such that $\lambda^{-1}(i)Cy$. Let $\lambda^{-1}(i) = x \in L_a$, $y \in L_b$. Then $a < b$ and so

$$\begin{aligned} (x) &= \ell_0 + \ldots + \ell_{a-1} + \lambda_a(x) \\ &< \ell_0 + \ldots + \ell_{a-1} + \ell_a \\ &\leq \ell_0 + \ldots + \ell_{b-1} \\ &\leq \ell_0 + \ldots + \ell_{b-1} + \lambda_b(y) \\ &= \lambda(y) . \end{aligned}$$

Therefore $i < \lambda(y)$. This proves e is a string. Replacing the nodes x_ν in a chain $x_0C\ldots Cx_r$ by their labels $\lambda(x_\nu)$ yields a chain $\lambda(x_0)C_e\ldots C_e\ \lambda(x_r)$. Therefore $d^C(x) \leq d^e(\lambda(x))$. The inverse replacement similarly yields $d^C(x) \geq d^e(\lambda(x))$, thus equality holds and it follows that $e(H_\lambda)$ is a po-string.

Remark (2.4) If e is a po-string then $L_0^e < \ldots < L_d^e$, where L_a^e stands for the a^{th} level of C_e. Therefore the definition of bijections $\lambda_a^e : L_a^e \approx \ell_a^e$, $a \leq d$ by the formula

$$\lambda_a^e(i) = \begin{cases} i \text{ if } a = 0 \\ i - \ell_{a-1} - \ldots - \ell_0 \text{ if } a > 0 \end{cases}$$

makes sense. We conclude that H(e) comes naturally equipped with the labelling λ^e = identity function of n.

Lemma (2.5) For any po-string e and labelled po-diagram H_λ, the following are true, (i) $e(H(e)) = e$, (ii) $H(e(H_\lambda)) \cong H$. #

Corollary (2.6) For every finite po-diagram H there is a po-string e such that $H \cong H(e)$. #

Using (1.4), (1.5) and (2.6) we have

Theorem (2.7) For every finite poset P there is a po-string e such that $P \cong P(H(e))$. #

§3. Representation ~~of finite posets~~

Construction (3.0) Consider a po-string e of order n and a permutation π of $n = \{0,1,\ldots,n-1\}$. Suppose π has the property

$$\text{(P)} \qquad d^e(\pi(i)) = d^e(i),\ i<n .$$

Then

$$\begin{aligned} iC_e k &\Rightarrow d^e(i) < d^e(k) && \text{by (0.5)} \\ &\Rightarrow d^e(\pi(i)) < d^e(\pi(k)) && \text{by (P)} \\ &\Rightarrow \pi(i) < \pi(k) && \text{by (PS1)} . \end{aligned}$$

Thus a new string e^{π} may be constructed, whose j^{th} list for $\pi(i) = j$ is the arrangement in ascending order from left to right of those values $\pi(k)$ such that $iC_e k$. Another way to put this is that the co-cover relation in e^{π} is given by

$$\pi(i) C_{e^{\pi}}\ \pi(k) \Leftrightarrow iC_e k .$$

Therefore e^{π} is a po-string. We say that π is applicable to e if (P) holds, and that e^{π} is the permute of e by π .

The set S_e of permutations which are applicable to e is a subgroup of the symmetric group S_n. If $e' \sim e$ is written when e' is a permute of e, then $\sim$ is an equivalence relation in the set of all po-strings of order n. The equivalence class of a po-string is called its permute class. It is not hard to prove that S_e is isomorphic to the direct product of symmetric groups $S_{\ell_0} \times \dots \times S_{\ell_d}$.

Lemma (3.1) Let e', e be po-strings of order n. Then $H(e) \cong H(e')$ if and only if $e' \sim e$. #

Combining (1.2), (1.5), (2.7) and (3.1) gives us

Theorem (3.2) For every finite poset P there is a unique-up-to-permutes po-string e such that $P \cong P(H(e))$. #

Lemma (3.3) There is a canonical choice of representative in the permute class of a po-string.

It will be shown below that the permute class of a po-string is totally ordered by a lexicographical ordering of the set of all strings of order n.

Theorem (3.4) For every finite poset P there is a unique po-string e such that e is first in its permute class an d $P \cong P(H(e))$. #

Remark (3.5) It can be shown that a theorem analogous to (3.4) holds in the more general context of pre-ordered sets, which are known to be the same as ~~finite~~ Alexandroff topological spaces.

§4. Generation of finite posets of order n

The first step in generating a list of representative po-strings of order n, exactly one from each permute class, is to prepare a list of all the permutations $\pi_0 \dots \pi_{n-1}$ of $n = \{0,1,\dots,n-1\}$. This may be done recursively, starting with the empty list for $n = 0$, and obtaining the list for n from the list for n-1 by inserting the figure n-1 in each of n positions of each entry in the list for n-1.

The second step is to prepare a list of all the strings of order n, arranged in their natural lexicographical order, described as follows. A string is a list $(e_0)\dots(e_{n-1})$ of lists $e_i = e_{i1}\dots e_{in_i}$ such that $i < e_{i1} < \dots < e_{in_i} < n$. Thus the possibilities for e_i correspond to the sub-lists of the list i+1 i+2 ... n-2 n-1 . These sub-lists, in turn, are in 1-1 correspondence with the binary numbers from 0 up to $2^{n-i-1}-1$. Therefore the possibilities for e_i are totally ordered in terms of numerical magnitude $\leq_i$ of corresponding binary numbers. The lexicographical ordering on the set of all strings of order n is the ordering $\leq$ given by $(e_o)\ldots(e_{n-1}) \leq (e'_o)\ldots(e'_{n-1})$ if and only if at the least index i such that $e_i \neq e'_i$ we have $e_i \leq_i e'_i$.

Example (4.0) The list of all strings of order 3 in lexicographical order.

( )( )( )
( )(2)( )
(2)( )( )
(2)(2)( )
(1)( )( )
(1)(2)( )
(12)( )( )
(12)(2)( ) .

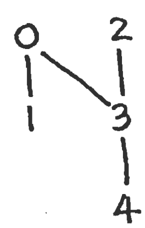
The third step is to delete from the list of all strings of order n those which are not po-strings. There are two ways a string e may fail to be a po-string, corresponding to the conditions (PS1)-(PS2). It may be that for some i and j we have $d^e(i) < d^e(j)$ but $i \geqslant j$, and it may be that for some $r > 1$ we have not only $i_o C_e \ldots C_e i_r$ but also $i_o C_e i_r$.

Algorithm (4.1) To compute d^e. Let e be a string of order n, $r = \sum_{0 \le i < n} n_i$, and let $f_k = e_{i_k j_k}$, $1 \le k \le r$ be all the entries of e, so $i_k < f_k < n-1$ for $1 \le k \le r$. For example, the 2+0+1+1+0 = 4 entries of (13)( )(3)(4)( ) are 1,3,3,4. Construct a matrix with n columns and r+1 rows as follows. Row 0 consists of zeroes. Row k+1, $0 \le k < r$ corresponds to the entry f_{k+1} and has, in column f_{k+1}, 1 plus the entry in row k at column i_{k+1}, and in all other columns is the same as row k. By this construction column 0 consists of zeroes. It is easy to see that $d^e(i)$ is the largest of the entries in column i, $0 \le i < n$. The string e fails to satisfy (PS1) if and only if the sequence $(d^e(0),\ldots,d^e(n-1))$ is not increasing (repetition allowed).

Example (4.2) Let n = 5 and e = (13)( )(3)(4)( ). Then r = 4 and the matrix is

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 2 \end{array},$$

so e fails to satisfy (PS1). This is obvious to the human being from the labelled po-diagram of e,



.

Algorithm (4.2) To verify (PS2). Let e and $f_1,\ldots,f_r$ as in (4.1). Construct a sequence of n matrices M_k such that M_k has n columns and $1 + \sum_{k \le i < n} n_i$ rows as follows. M_k is constructed exactly as in (4.1), except that all the columns from 0 up to and including column f_k are filled with zeroes, and the calculation can continue only until row $1 + \sum_{k \le i < n} n_i$. Thus (4.1) gives M_0.

It is almost easy to see that e fails to satisfy (PS2) if and only if there exists a column in one of the matrices which has both an entry 1 and an entry $d > 1$. The largest of the entries in column $i > f_k$ of M_k is the depth of node i below node f_k.

Example (4.3) Let n = 6 and e = (2)(2)(35)(4)(5)( ). Then r = 1+1+2+1+1+0 = 6, and the matrices are as follows.

	0	1	2	3	4	5
M_0	0	0	0	0	0	0
	0	0	1	0	0	0
	0	0	1	0	0	0
	0	0	1	2	0	0
	0	0	1	2	0	2
	0	0	1	2	3	2
	0	0	1	2	3	4
M_1	0	0	0	0	0	0
	0	0	1	0	0	0
	0	0	1	2	0	0
	0	0	1	2	0	2
	0	0	1	2	3	2
	0	0	1	2	3	4

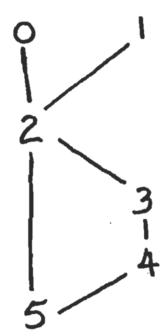
$$M_2 \quad \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 1 & 2 & 3 \end{matrix}$$

$$M_3 \quad \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 \end{matrix}$$

$$M_4 \quad \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{matrix}$$

$$M_5 \quad \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \; .$$

Since column 5 of M_2 has entries 1 and 3, e fails to satisfy (PS2). The human being sees this immediately from the labelled po-diagram of e,



.

Thus the construction of a sequence of matrices suffices to determine whether a string is a po-string, and those strings which are not po-strings are now deleted from the master string list, ~~of all strings.~~ yielding the master po-string lis[t]

The fourth and last step is to construct the <u>output list</u> of representative po-strings. Put the empty string ( )...( ) at the top of the list. Proceed down the master po-string list, accepting or rejecting po-strings according to the following rule. Let e be the po-string in question, and let $\pi_0 \dots \pi_{n-1}$ be a permutation of $n = \{0,1,\dots,n-1\}$. Then π is applicable to e if and only if $d^e(\pi_i) = d^e(i)$, $0 \leq i < n$. Proceed down the list of permutations prepared in the first step, computing e^π, when π is applicable to e, according to (3.0). Stop this calculation of permutes e^π of e as soon as $e^\pi < e$ in the lexicographical order, and reject e. If no permute of e is earlier than e in the lexicographical order, then e must be first in its permute class, and accept e. The list of accepted po-strings is the output list, and is essentially a list of finite posets of order n, one in each isomorphism class of finite posets of order n.

Remark (4.4) This listing places all finite posets of order n later than all finite posets of order n-1, therefore the set of isomorphism classes of finite posets is totally ordered.

Computation (4.5) In less than 20 minutes a 360 PL/1 program gave all representatives of posets with 6 nodes, yielding 1, 2, 5, 16, 63, and 318 for 1, 2, 3, 4, 5, and 6 nodes, respectively. A112

Acknowledgement (4.6) Robert Siegelbaum wrote the program, and is a most helpful collaborator.

Acknowledgement (4.7) Barry Mitchell offered numerous criticisms and corrections of a first draft of this paper.