PRACE IPI PAN • ICS PAS REPORTS

Józef Winkowski

Algebras of arrays a tool to deal with concurrency

287

1977

WARSZAWA

INSTYTUT PODSTAW INFORMATYKI POLSKIEJ AKADEMII NAUK
INSTITUTE OF COMPUTER SCIENCE POLISH ACADEMY OF SCIENCES
00-901 WARSAW, P.O. Box 22, POLAND

Józef Winkowski

ALGEBRAS OF ARRAYS - A TOOL TO DEAL WITH CONCURRENCY

287

Warsaw 1977

Mailing address: Józef Winkowski
Institute of Computer Science
Polish Academy of Sciences
P.O. Box 22, 00-901 Warszawa

P r i n t e d   a s   a   m a n u s c r i p t
N a   p r a w a c h   r ę k o p i s u

Nakład 700 egz. Ark. wyd. 0,55; ark. druk. 1,00. Papier offset. kl. III, 70 g, 70 x 100. Oddano do druku w marcu 1977 r. W. D. N. Zam. nr 193/0/77
F - 9

Abstract . Содержание . Streszczenie

New objects similar to words over certain alphabets are introduced. These objects, called arrays, can sometimes be composed sequentially or parallelly which gives a possibility to introduce algebras of arrays. As arrays are good mathematical models of non-sequential processes, such algebras are very handy tools to describe non-sequential systems without any loss of information on concurrency. Using them one is able to characterize sets of non-sequential processes generated by non-sequential systems as solutions of certain equations.

Алгебры расстановок - некоторое средство описания параллелизма

Вводятся новые объекты, аналогичные цепочкам над некоторыми алфавитами. Эти объекты, называемые расстановками, можно складывать последовательно или параллельно, что приводит к алгебрам расстановок. Так как расстановки являются хорошими математическими моделями параллельных процессов, эти алгебры являются очень удобным средством описания действия параллельных систем без потери информации об имеющемся параллелизме. Пользуясь ими, можно рассматривать множества параллельных процессов, порождаемых такими системами, как решения некоторых уравнений.

Algebry ustawień - narzędzie do opisu współbieżności

Wprowadzono nowe obiekty podobne do słów nad rozmaitymi alfabetami. Te obiekty, zwane ustawieniami, można składać sekwencyjnie i równolegle, co daje możliwość wprowadzenia algebr ustawień. Ponieważ ustawienia są dobrymi modelami matematycznymi procesów niesekwencyjnych, takie algebry są wygodnym środkiem opisu systemów niesekwencyjnych bez utraty informacji o współbieżności. Stosując je można charakteryzować zbiory procesów niesekwencyjnych generowanych przez systemy niesekwencyjne jako rozwiązania pewnych równań.

1. INTRODUCTION

In what follows we present new objects similar to words over certain alphabets. These objects, called arrays, are classes of isomorphic partially ordered sets labelled in a particular way. Some of them can be composed sequentially or parallelly which gives a possibility to introduce algebras of arrays. These algebras appear to be nearly strict monoidal categories.

As arrays are good mathematical models of non-sequential processes, algebras of arrays are very handy tools to describe non-sequential systems. Using them we are able to characterize sets of non-sequential processes generated by such systems as solutions of certain equations, similarly as we do in the case of sequential systems like automata.

2. ALGEBRAS OF ARRAYS

By <u>arrays</u> we mean here classes of isomorphic labelled partially ordered sets (l.p.o. sets). These l.p.o. sets are supposed to be triples $(X, \leq, l)$ belonging to a universal set U (in the sense of Sonner[7]) and consisting of a partially ordered set $(X, \leq)$ and of a mapping $l: X \longrightarrow L$ called a <u>labelling</u> such that:

(1) $l(x)=l(y)$ implies $x \leqslant y$ or $y \leqslant x$

Two l.p.o. sets $a=(X, \leqslant, l)$ and $a'=(X', \leqslant', l')$ are considered to be isomorphic if there is a bijection $f: X \longrightarrow X'$ called an isomorphism from a to a' and written as $f: a \longrightarrow a'$ such that:

$$x \leqslant y \text{ iff } f(x) \leqslant' f(y)$$
$$l(x)=l'(f(x))$$

Isomorphic l.p.o. sets a,a' belong to the same array which will be denoted [a] or [a']. Arrays with labellings having values in a set L (of labels) are said to be L-valued.

Arrays in our sense are similar to the processes considered by Mazurkiewicz [3]. The difference is that, instead of arbitrary triples $(X, \leqslant, l)$, we concentrate only on l.p.o. sets satisfying (1), and we consider arrays as classes of isomorphic l.p.o. sets.

As we said, arrays are good mathematical models of various real processes, especially non-sequential ones. Every array can be considered as a space-time in the physical sense. Its ordering reflects temporal features whereas its labelling reflects spatial ones.

In order to define the sequential and parallel compositions of arrays we introduce first two partial unary operations ∂_0, ∂_1 assigning a "source" and a "target" respectively to some of arrays.

The source $\partial_0(a)$ of an array $a=[(X, \leqslant, l)]$ is defined as the array $[(X_0, \leqslant|X_0, l|X_0)]$ with X_0 being the set of minimal elements of $(X, \leqslant)$ and $\leqslant|X_0$, $l|X_0$ being the restrictions of $\leqslant$, l to X_0, provided every element of X has a lower bound in X_0. Otherwise $\partial_0(a)$ is undefined. Similarly, the target $\partial_1(a)$ of a is defined as the array $[(X_1, \leqslant|X_1, l|X_1)]$ with X_1 being the set of maximal elements of $(X, \leqslant)$, provided every element of X has an upper bound in X_1.

The sequential composition $a_1 \cdot a_2$ is defined for arrays a_1, a_2 such that $\partial_1(a_1), \partial_0(a_2)$ are defined and identical. It consists in "glueing" a_1, a_2 by identifying every maximal element of a_1 with the minimal element of a_2 having the same label, and by extending the orderings of a_1 and a_2 to a common ordering. More formally, if $a_1 = [(X_1, \leqslant_1, l_1)]$ with the set X_0 of maximal elements of $(X_1, \leqslant_1)$, and $a_2 = [(X_2, \leqslant_2, l_2)]$ with the same set X_0 of minimal elements of $(X_2, \leqslant_2)$, and $l_1 | X_0 = l_2 | X_0$, then we define $a_1 \cdot a_2$ as $[(X, \leqslant, l)]$, where:

$$X = \{1\} \times X_1 \cup \{2\} \times (X_2 \setminus X_0)$$

$x \leqslant y$ iff there are x', y' such that

$x = (1, x')$ and $y = (1, y')$ and $x' \leqslant_1 y'$ or

$x = (2, x')$ and $y = (2, y')$ and $x' \leqslant_2 y'$ or

$x = (1, x')$ and $y = (2, y')$ and $x' \leqslant_1 z \leqslant_2 y'$ for some $z \in X_0$

$$l(x) = \begin{cases} l_1(x') & \text{for } x = (1, x') \\ l_2(x') & \text{for } x = (2, x') \end{cases}$$

The parallel composition $a_1 \times a_2$ is defined for arrays a_1, a_2 such that the sets of labels of the elements of a_1 and a_2 are disjoint. The result is defined as an array consisting of two independent parts corresponding to a_1 and a_2. More formally, if $a_1 = [(X_1, \leqslant_1, l_1)]$ and $a_2 = [(X_2, \leqslant_2, l_2)]$ with $l_1(X_1) \cap l_2(X_2) = \emptyset$ then we define $a_1 \times a_2$ as $[(X, \leqslant, l)]$, where:

$$X = \{1\} \times X_1 \cup \{2\} \times X_2$$

$x \leqslant y$ iff there are x', y' such that

$x = (1, x')$ and $y = (1, y')$ and $x' \leqslant_1 y'$ or

$x = (2, x')$ and $y = (2, y')$ and $x' \leqslant_2 y'$

$$l(x)=\begin{cases} l_1(x') \text{ for } x=(1,x') \\ l_2(x') \text{ for } x=(2,x') \end{cases}$$

There is also a particular null array $0=[(\emptyset,\emptyset,\emptyset)]$.

It is easy to check that all these operations are defined correctly. Thus, the set V(U) of arrays defined in the universal set U converts into a partial algebra:

$$A(U)=(V(U), \partial_0, \partial_1, \cdot, \bowtie, 0)$$

We call it the algebra of arrays defined in U. Its subalgebras are called algebras of arrays.

It is easy to verify that the following axioms are true in any algebra of arrays:

(A1) if $\partial_0(a)$ is defined then $\partial_0(\partial_0(a))$, $\partial_1(\partial_0(a))$ are defined and $\partial_0(\partial_0(a))=\partial_1(\partial_0(a))=\partial_0(a)$,

(A2) if $\partial_1(a)$ is defined then $\partial_0(\partial_1(a))$, $\partial_1(\partial_1(a))$ are defined and $\partial_0(\partial_1(a))=\partial_1(\partial_1(a))=\partial_1(a)$,

(A3) $a\cdot b$ is defined iff $\partial_1(a)$, $\partial_0(b)$ are defined and identical,

(A4) if $a\cdot b$ is defined then $\partial_0(a)$ is defined iff $\partial_0(a\cdot b)$ is defined, and $\partial_0(a)=\partial_0(a\cdot b)$ if defined,

(A5) if $a\cdot b$ is defined then $\partial_1(b)$ is defined iff $\partial_1(a\cdot b)$ is defined, and $\partial_1(b)=\partial_1(a\cdot b)$ if defined,

(A6) if $\partial_0(a)$ is defined then $\partial_0(a)\cdot a$ is defined and $\partial_0(a)\cdot a=a$,

(A7) if $\partial_1(a)$ is defined then $a\cdot\partial_1(a)$ is defined and $a\cdot\partial_1(a)=a$, (A8) if $\partial_1(a), \partial_0(b), \partial_1(b), \partial_0(c)$ are defined and $\partial_1(a)=\partial_0(c)$, $\partial_1(b)=\partial_0(c)$ then $a\cdot(b\cdot c)=(a\cdot b)\cdot c$,

(A9) $a\rtimes b$ is defined iff $b\rtimes a$ is defined, and $a\rtimes b=b\rtimes a$ if defined,

(A10) $(a\rtimes b)\rtimes c$ is defined iff $a\rtimes(b\rtimes c)$ is defined, and $(a\rtimes b)\rtimes c=a\rtimes(b\rtimes c)$ if defined,

(A11) there is an element 0 (null array) which is neutral for $\rtimes$, i.e., $0\rtimes a=a\rtimes 0=a$ for all a,

(A12) if $a\rtimes b$ is defined then $\partial_0(a), \partial_0(b)$ are defined iff $\partial_0(a\rtimes b)$ is defined and iff $\partial_0(a)\rtimes\partial_0(b)$ is defined, and $\partial_0(a\rtimes b)=\partial_0(a)\rtimes\partial_0(b)$ if defined,

(A13) if $a\rtimes b$ is defined then $\partial_1(a), \partial_1(b)$ are defined iff $\partial_1(a\rtimes b)$ is defined and iff $\partial_1(a)\rtimes\partial_1(b)$ is defined, and $\partial_1(a\rtimes b)=\partial_1(a)\rtimes\partial_1(b)$ if defined,

(A14) if $a\cdot a'\rtimes b\cdot b'$ is defined then $(a\rtimes b)\cdot(a'\rtimes b')$ is defined and $a\cdot a'\rtimes b\cdot b'=(a\rtimes b)\cdot(a'\rtimes b')$.

This means that algebras of arrays have nearly the same properties as strict monoidal categories (see Mac Lane[2] for definition). The only difference is that their operations are partial. For this reason, algebras fulfilling (A1) - (A14) will be called <u>partial monoidal categories</u> (p.m. categories).

We define a <u>homomorphism</u> from a p.m. category $A=(V, \partial_0, \partial_1, \cdot, \rtimes, 0)$ to a p.m. category $A'=(V', \partial_0', \partial_1', \cdot', \rtimes', 0')$ as a mapping $f:V\longrightarrow V'$ such that:

(h1) if $\partial_0(a)$ is defined then $\partial_0'(f(a))$ is defined and $f(\partial_0(a))=\partial_0'(f(a))$, (h2) if $\partial_1(a)$ is defined then $\partial'_1(f(a))$ is defined and $f(\partial_1(a))=\partial'_1(f(a))$,

(h3) if $a\cdot b$ is defined then $f(a)\cdot' f(b)$ is defined and $f(a\cdot b)=f(a)\cdot' f(b)$,

(h4) if $a\times b$ is defined then $f(a)\times' f(b)$ is defined and $f(a\times b)=f(a)\times' f(b)$,

(h5) $f(0)=0'$

In other words, f is nearly a functor that preserves the parallel composition and its neutral element.

All the p.m. categories and their homomorphisms constitute a category Pmc of p.m. categories.

Algebras of arrays are interesting p.m. categories because some of them have sets of "free" generators in the sense that mappings from such sets into underlying sets of p.m. categories can be uniquely extended to homomorphisms provided they preserve all the equalities between generators. Such "freely generated" algebras of arrays play thus a role similar to that of free monoids of words.

3. APPLICATIONS TO NON-SEQUENTIAL SYSTEMS

Having an algebra $A=(V,\partial_0,\partial_1,\cdot,\times,0)$ of arrays one can define for subsets of V:

$$\partial_0(C)=\{a\in V:\ a=\partial_0(c)\ \text{for some}\ c\in C\}$$
$$\partial_1(C)=\{a\in V:\ a=\partial_1(c)\ \text{for some}\ c\in C\}$$

$$C \cdot C' = \{a \in V:\ a = c \cdot c' \text{ for some } c \in C,\ c' \in C'\}$$

$$C \times C' = \{a \in V:\ a = c \times c' \text{ for some } c \in C,\ c' \in C'\}$$

$$0 = \{0\}$$

as it was done by Blikle [1]for sets of sequences. This gives continous operations in the lattice B(V) of subsets of V, i.e., the following conditions are satisfied:

$$\partial_0\left(\bigcup_{i \in I} C_i\right) = \bigcup_{i \in I} \partial_0(C_i), \quad \partial_1\left(\bigcup_{i \in I} C_i\right) = \bigcup_{i \in I} \partial_1(C_i)$$

for every chain $\{C_i\}_{i \in I}$ of elements of B(V), and

$$\left(\bigcup_{i \in I} C_i\right) \cdot \left(\bigcup_{i \in I} C_i'\right) = \bigcup_{i \in I} C_i \cdot C_i'$$

$$\left(\bigcup_{i \in I} C_i\right) \times \left(\bigcup_{i \in I} C_i'\right) = \bigcup_{i \in I} C_i \times C_i'$$

for every chain $\{(C_i, C_i')\}_{i \in I}$ of elements of the product lattice $B(V) \times B(V)$. Thus, every mapping $f: (B(V))^n \longrightarrow (B(V))^n$ defined using these operations is continous and, due to the well known Tarski's theorem, has fixed points. Moreover, there is the least fixed point C of f and it is given by the formula:

$$C = f(\emptyset^n) \cup f(f(\emptyset^n)) \cup \ldots$$

where $\emptyset^n = (\emptyset, \ldots, \emptyset) \in (B(V))^n$. This allows one to solve sets of equations of the form:

$$X_1 = f_1(X_1, \ldots, X_n)$$

$$\ldots$$

$$X_n = f_n(X_1, \ldots, X_n)$$

where $f_1, \ldots, f_n$ are defined using the introduced operations. For instance, the closure of a set C of arrays with respect to the parallel composition is the least solution of the equation:

$$X = C \times X \cup C$$

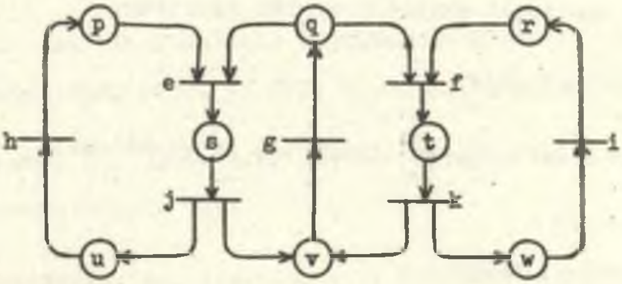
The above facts allow one to characterize algebraically sets of arrays representing non-sequential processes generated by non-sequential systems. We shall show this on the example of finite Petri nets.

Let $N = (C,E,Pre,Post)$ be a finite Petri net with a set C of conditions, a set E of events, and two binary relations: $Pre \subseteq C \times E$, indicating that certain conditions are preconditions of particular events, and $Post \subseteq E \times C$, indicating that certain conditions are postconditions of particular events (see Petri [5,6] for details).

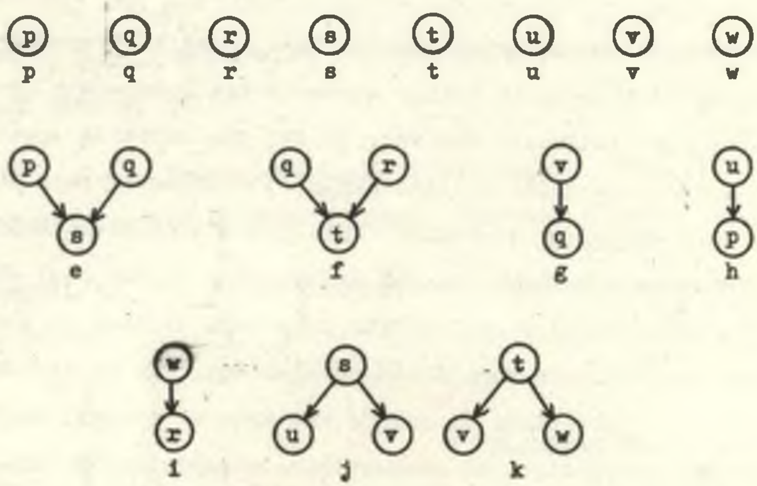
Every event e represents an elementary process that can be considered as a C-valued array consisting of minimal and maximal elements only, where the labels of minimal elements are exactly the preconditions of e, the labels of maximal elements are exactly the postconditions of e, and every minimal element is earlier than every maximal element.

Every condition c represents an elementary process that can be considered as a one-element array labelled by c.

The set P(N) of all such elementary processes corresponds to the whole net N, and it determines N completely. For example, for the following net (circles denote conditions and bars denote events):



we have the set $\{p,q,r,s,t,u,v,w,e,f,g,h,i,j,k\}$ of the following elementary processes:



The whole net N generates processes which are composed of elementary ones according to the net structure. Every of them has an initial part which is a parallel composition of elementary processes, and this part is composed sequentially with the rest of the process. Thus, the set X of generated processes is a solution of the equation:

$$X = Y \cdot X \cup Y$$

where Y is the closure of P(N) with respect to the parallel composition, i.e., the least solution of the equation:

$$Y = P(N) \times Y \cup P(N)$$

Finally, we obtain the following set of equations:

$$X = Y \cdot X \cup Y$$

$$Y = P(N) \times Y \cup P(N)$$

4. COMMENTS

The idea to characterize algebraically sets of processes generated by Petri nets is fairly known in the literature (see Paterson [4] for instance). However, in all the existing approaches Petri nets are considered as indeterministic automata capable to generate only <u>sequences</u> of states. This leads to rather large sets of equations and also to a loss of information on the real concurrency (see Petri [6]). Application of arrays instead of sequences allows one to avoid such disadvantages and, due to the introduced algebraic operations on arrays and sets of arrays, to preserve still a possibility to characterize algebraically sets of processes generated by non-sequential systems.

<u>Acknowledgments</u>. I wish to thank to Prof. A. Blikle, Dr. P. Dembiński, Dr. W. Kwasowiec, Dr. J. Leszczyłowski, Dr. J. Małuszyński, and Prof. A. Mazurkiewicz for many discussions which helped me to formulate the idea presented in the paper.

REFERENCES

1. Blikle, A., An Algebraic Approach to Programs and their Computations, Proc. of MFCS'73 Symp., High Tatras, 1973

2. Mac Lane, S., Categories for the Working Mathematician, Springer-Verlag, 1971

3. Mazurkiewicz, A., Invariants of Concurrent Programs, paper presented at INFOPOL Conf., Warsaw, March 1976

4. Paterson, J.L., Computation Sequence Sets, Journal of Comp. and System Sc., Vol 13, No 1, 1976

5. Petri, C.A., Concepts of Net Theory, Proc. of MFCS'73 Symp., High Tatras, 1973

6. Petri, C.A., Interpretations of Net Theory, GMD, Bonn, 1975, paper presented at MFCS'75 Symp., Marianske Lazne, 1975

7. Sonner, J., The Formal Definition of Categories, Math. Zeits., 80 (1962)

Received March 4, 1977

Ostatnio w serii tej ukazały się:

Nr 267 - E. Margański: Organizacja i niektóre algorytmy w nowym systemie wyszukiwania

Nr 268 - M. Kotowski: Kryteria usuwania procesów zakleszczonych

Nr 269 - R. Sawa: Brajlowski system współpracy z ODRĄ-1204

Nr 270 - W. Zawistowski: Organizacja stanowiska pracy niewidomego programisty

Nr 271 - S. Jakubowski: Optacon jako zasadnicze narzędzie pracy niewidomego programisty

Nr 272 - S. Janicki: Niejednorodne automaty stochastyczne i pewne ich własności

Nr 273 - B. Konikowska: Continuous information systems

Nr 274 - W. Skarbek: Sieci rejestrów przesuwających jako model synchronizacji procesów współbieżnych

Nr 275 - W. Lipski: Informational systems: semantic issnes related to incomplete information. Part I

Nr 276 - S. Janicki: O zbiorach generowanych przez automaty stochastyczne typu jednorodnego łańcucha Markowa

Nr 277 - S. Jakubowski: STEREOTONER - dźwiękowy aparat do czytania dla niewidomych

Nr 278 - I. Kaliszewski: Otwarty system programowania liniowego wykorzystujący trókątną postać macierzy bazowej

Nr 279 - J. Górski: On the optimal arrangement for consecutive storage of objects

Nr 280 - J. Łoś: Mathematical theory of von Neumann economic models

Nr 281 - H. Rasiowa: Algorithmic logic

Nr 282 - W. Lipski: On strings containing all aubsets as substrings

Nr 283 - K. Mańkowski: Podstawowe problemy adaptacji słabowidzących do zawodu programisty

Nr 284 - J. Tiuryn: Fixed-points and algebras with infinitely long expressions. Part I. Regular algebras

Nr 285 - T. Traczyk: On consecutive retrieval from drum-type storage

Nr 286 - J. Winkowski: Game of life as a synchronous process that can be realized asynchronously