MTA SZTAKI Közlemények 33/1985 p. 95-112

NETS, POLYCATEGORIES AND SEMANTICS OF PARALLEL PROGRAMS

Y.P. Velinov

Center of Applied Mathematics; Higher Institute of Mechanical and Electrical Engineering; 1000 Sofia, p.box 384, Bulgaria

The purpose of the present paper is to establish some connections between (a modified variant of) Petry nets and Polycategories and on this base to spread the category approach to the semantics of programs (flow diagrams) proposed by ADJ group [JCS] and Goguen [HCT] over a wider class of parallel programs.

The reader which is not accustomed with the category approach to programming is invited to consult [HCT] or [JCS], where the ideas not concerned with the parallelism can be easily followed.

1. POLYGRAHS AND λ-POLYCATEGORIES

1.1. A *polygraph* (or a net) $\mathbb{G}$ is defined to be a system presented by the following string of data

$$<\mathbb{O}, \mathbb{A}, +, \bullet, \mathrm{dom}, \mathrm{cod}>$$

where

- $\mathbb{O} \doteq \mathrm{Ob}(\mathbb{G})$ is a set of elements called *objects*; $\mathbb{A} \doteq \mathrm{Ar}(\mathbb{G})$ is a set of elements called *(poly)arrows*; $\bullet \in \mathbb{O}$;

- dom,cod and + are three mappings: dom $\mathbb{A} \to \mathbb{O}$ puts in correspondence to each arrow an object called its *initial object*; cod: $\mathbb{A} \to \mathbb{O}$ puts in correspondence to each arrow an object called its *final object*; $+: \mathbb{O} \times \mathbb{O} \to \mathbb{O}$.

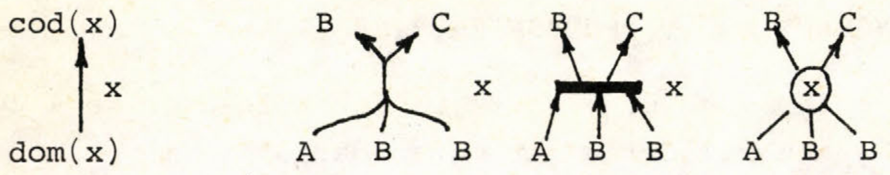
These data are such that:

PO. $<\mathbb{O},+,\bullet>$ is a monoid with operation + and neutral element $\bullet$.

The objects of a polygraph usually will be denoted by A, B,C,u,v,w and the arrows by x,y,z,f,g,h.

The presented structure is similar to the Petry net structure [PNT]. If the monoid of objects of a polygraph is taken to be a free monoid over some set $\mathbb{V}$ of "places" and arrows are taken to be "transitions" the only significant difference will be that instead of sets string of places are put in correspondence to the transitions by the "input" and "output" functions dom and cod.

Situations in a polygraph can be graphically illustrated representing objects by their names situated on a sheet of paper and arrows by "drawn branching arrows" (sometimes with boxes or circules on them) which lead from names of objects to names of objects. If several names of objects are situated at different beginnings (ends) of a drawn arrow, they form the name of the initial (final) object of the arrow by a standard agreement of composing them from the left to the right.



$$dom(x) = A + B + B, \ cod(x) = B + C$$

An ordered triple of objects will be called a *connector*.

Connector will be denoted by $\perp$ or $\top$. A connector <A,B,C> will be denoted by $A\underline{B}C$ also. If $\perp = <A,B,C>$ is a connector then $|\perp|$ will denote the object A+B+C and $L(\perp)$, $C(\perp)$, $R(\perp)$ will denote its first, second and third component respectively. $C(\perp)$ will be called the *center* of the connector. The set of all connectors of a polygraph $\mathbb{G}$ will be denoted by $Con(\mathbb{G})$.

An operation ϕ: Con($\mathbb{G}$)xCon($\mathbb{G}$) $\rightarrow$ Con($\mathbb{G}$) defined by the correspondence

$$<<A_1,A_2,A_3>, <B_1,B_2,B_3>> \mapsto <A_1+B_1, B_2, B_3+A_3>$$

will be used in the following exposition.

The sign "+" for the monoidal operation will often be omitted.

1.2 A λ -*polycategory* $\mathbb{P}$ is defined to be a system presented by the string of data

$$< \mathbb{O}, \mathbb{A} +, \bullet, \mathrm{dom}, \mathrm{cod}, \gamma, I>$$

where $< \mathbb{O}, \mathbb{A},+,\bullet,\mathrm{dom},\mathrm{cod}> \doteq \mathrm{Gr}(\mathbb{P})$ is a polygraph (the underlying polygraph of the polycategory), $I: \mathbb{O} \rightarrow \mathbb{A}$, $I:A \mapsto I_A$ is a mapping that puts in correspondence to each object a selected arrow, called *identity* arrow and

$$\gamma: \mathbb{A}\mathrm{x}\mathrm{Con}(\mathrm{Gr}(\mathbb{P}))\mathrm{x}\, \mathbb{A} \rightarrow \mathbb{A}$$
$$\gamma: <x,\perp,y> \mapsto x \overline{/\perp/}\, y$$

is a partial mapping, called *composition law for the arrows*. These data fulfil the following axioms:

PY 1. (Existence of composites)

$$\exists z(x\overline{/\perp/}\,y = z) \Leftrightarrow \mathrm{dom}(x)= |\perp| \;\&\; C(\perp)= \mathrm{cod}(y).$$

PY 2. (dom and cod of a composite)

$$x\overline{/\perp/}\,y = z \;\leftrightarrow\; \begin{cases} \mathrm{dom}(z) = |\perp\phi\,\mathrm{dom}(y)| \\ \mathrm{cod}(z) = \mathrm{cod}(x) \end{cases}$$

PY 3. (Partial commutativity)

$$(x/\overline{A_1\underline{A_2}A_3A_4A_5}/y)/\overline{A_1\mathrm{dom}(y)A_3\underline{A_4}A_5}/z =$$

$$= (x/\overline{A_1A_2A_3\underline{A_4}A_5}/z)/\overline{A_1\underline{A_2}A_3\mathrm{dom}(z)A_5}/y$$

if the described composites exist.

PY 4. (Associativity)

$$x/\overline{\ \perp\ }/(y/\overline{\ \top\ }/z) = (x/\overline{\ \perp\ }/y)/\overline{\perp\bullet\top}/z$$

if the described composites exist.

PY 5. (Unit law)

For any object A and arrows x,y : $\mathrm{dom}(I_A) = \mathrm{cod}(I_A) = A$ and $I_A/\overline{\underline{A}}/x=x$, $y/\overline{\underline{A}}/I_A=y$ if the described composites exist.

Now to shorten the notation we can introduce two derivative operations - "o" and "+":

$$x \circ y = x/\underline{\mathrm{dom}(x)}/y,$$
$$x + y = (I_{\mathrm{cod}(x)+\mathrm{cod}(y)}/\overline{\underline{\mathrm{cod}(x)}\mathrm{cod}(y)}/x)/\overline{\mathrm{dom}(x)\underline{\mathrm{cod}(y)}}/y$$

The main interpretation of the presented structure, which we will need here is the λ-polycategory $\mathbb{P}$fn. It is defined as follows:

- Ob($\mathbb{P}$fn) contains all the sets (in some universe) distingwished to within the associativity of Cartesian product and the product with the set $\{\phi\}$;
- Ar($\mathbb{P}$fn) contains all the functions;
- the monoidal operation is the product and the neutral element of this operation is $\{\phi\}$;
- the composition law for the arrows is the superposition specified by the connectors.

A λ-polycategory is said to be a λ-polycategory with forks iff for each object A there is an arrow $\nabla_A : A \to A+A$ such that

$$(I_{AAA}\overline{/\underline{AA}A/}\nabla_A)\overline{/\underline{AA}/}\nabla_A = (I_{AAA}\overline{/A\underline{AA}/}\nabla_A)\overline{/\underline{AA}/}\nabla_A \doteq \nabla_A^3$$

More details about polygraphs, polycategories and λ-polycategories can be found in [P,CFP,P.EP,PSMC I, PSMC II, PR,CAP]. In particular λ-polycategories can be viewed as strict monoidal categories [CWM,MFC] in the frame of which a new composition law

$$\langle x, \langle A,B,C\rangle, y\rangle \mapsto x\overline{/ABC/}y \doteq x\ o(I_A+y+I_C)$$

is introduced [PSMC I, PSMC II, PR].

1.3. Let $\mathbb{G} = \langle \mathbb{V}^*, \mathbb{A}, +, \bullet, \mathrm{dom}, \mathrm{cod}\rangle$ be a polygraph with a free monoid of objects $\langle \mathbb{V}^*, +, \bullet\rangle$ over a set of nodes $\mathbb{V}$. Notice that in such a situation the objects can be viewed as words over $\mathbb{V}$ and a connector $\perp$ specifies a participation of the word $C(\perp)$ in $|\perp|$.

A sequence of arrow and connectors in $\mathbb{G}$ of the form

$$p = \langle \perp_1, x_1, \perp_2, x_2, \ldots \perp_k, x_k, \ldots, \perp_m, x_m, \ldots \perp_n, x_n, \perp_{n+1}\rangle$$

is called a path in $\mathbb{G}$ iff

- the center of $\perp_i$ is $\mathrm{cod}(x_i)$, $(i=1,2,\ldots,n)$;
- $|\perp_{i+1}| = |\perp_i \blacklozenge \mathrm{dom}(x_i)|$, $(i=1,2,\ldots,n)$;
- $\perp_{n+1} = \langle \bullet, |\perp_n \blacklozenge \mathrm{dom}(x_n)|, \bullet\rangle$.

The objects $|\perp_1|$ and $|\perp_{n+1}|$ are called the *end* and the *beginning* of the path and denoted by $\mathrm{cod}^*(p)$ and $\mathrm{dom}^*(p)$ respectively.

In degenerated cases, when a path does not contain arrows, it should be in the form $<\underline{uu \ldots u}, \underline{u}>$, $u \in V^*$. Such paths are called fork paths (or diagonals). The paths of the form $<\underline{u},\underline{u}>$ are called identity paths. They will be denoted by p_u also. Paths of the form $p_x = <\underline{cod(x)}, x, \underline{dom(x)}>$ and degenerated paths will be called *elementary paths*.

Each path p in $\mathfrak{G}$ proposes a sequence of words

$$|\perp_1|, |\perp_2|, \ldots, |\perp_{n+1}|$$

such that $|\perp_i|$ is obtained from $|\perp_{i-1}|$ by substitution of words. So each path proposes a sequence of substitutions also.

Let p be a path in $\mathfrak{G}$. An arrow x_m is *directly connected.* with a connector $\perp_k$ in p iff the participation $\perp_m$ of $cod(x_m)$ in $|\perp_m|$ is a conveyed (by substitutions of p) participation of $cod(x_m)$ in $|\perp_k|$; an arrow x_m is directly connected with another arrow x_k in p iff it is directly connected with $\perp_{k+1}$ and the center of $\perp_m$ is a conveyed participation of such a subword of $|\perp_{k+1}|$ that intersects with the center of $\perp_k \blacklozenge dom(x_k)$. If x_1 and x_m are directly connected with $\perp_k$ in p the connection with x_m *is to the left* of the connection with x_m iff the predecessors of $\perp_1$ and $\perp_m$ in $|\perp_k|$ do not intersect and the first is situated to the left of the second.

A path in $\mathfrak{G}$ is called canonical iff it satisfies the following conditions:

- each arrow situated between two arrows directly connected with a connector is directly connected with the connector also;

- if two arrows x_1 and x_m are directly connected with a connector and the connection with the first is to the left of the connection with the second then the first is situated in the path to the left of the second;

- there are not two connected forks in the path.

An operation "transposition" can be applied to any two adjacent unconnected arrows x_{m-1} and x_m in a path

$$p = \langle \perp_1, x_1, \perp_2, x_2, \ldots, \perp_{m-1}, x_{m-1}, \perp_m, x_m, \ldots, \perp_n, x_n, \perp_{n+1} \rangle$$

to transform it to another path

$$q = \langle \perp_1, x_1, \perp_2, x_2, \ldots \top_{m-1}, x_m, \top_m, x_{m-1}, \ldots, \perp_n, x_n, \perp_{n+1} \rangle$$

where $\top_{m-1}$ is obtained from $\perp_{m-1}$ removing its center over the predecessor of the center of $\perp_m$ and $\top_m$ is obtained from $\perp_{m-1}$ substituting the predecessor of the center of $\perp_m$ by $\mathrm{dom}(x_m)$.

The operation is convertable. It is easy to spread it over any two arrows in a path which are not adjacent but fulfil the condition that there are no arrows between them, the second one is connected with.

Similarly, an operation "coalescence of forks" can be introduced. It includes applying of transpositions so to put all connected forks in neighbourhood, replacement of each group of k forks by ∇^{k+1} and the corresponding modifications of connectors.

PROPOSITION 1. Each path in a polygraph $\mathfrak{G}$ with a free monoid of objects can be transformed to a unique canonical path applying finitely many times the operations transposition and coalescence of forks.

Let $p_1 = \langle \perp_{1,1}, x_{1,1}, \perp_{1,2}, x_{1,2}, \ldots, \perp_{1,k}, x_{1,k}, \perp_{1,k+1} \rangle$ and $p_2 = \langle \perp_{2,1}, x_{2,1}, \perp_{2,2}, x_{2,2}, \ldots, \perp_{2,\ell}, x_{2,\ell}, \perp_{2,\ell+1} \rangle$ be two paths in a polygraph $\mathfrak{G}$ and $\top = u\ \underline{\mathrm{cod}^*(p_2)}\ v$ be a connector such that $|\top| = \mathrm{dom}^*(p_1)$. A *composition* of p_1 and p_2 by $\top$ is a path

$$p_1 \overline{/\top/}' p_2 = \langle \top_1, x_{1,1}, \top_2, x_{1,2}, \ldots, \top_k, x_{1,k}, \top_{k+1}, x_{2,1}, \top_{k+2}, x_{2,2},$$
$$\ldots, \top_{k+l}, x_{2,\ell}, \top_{k+\ell+1}\rangle,$$

where $\top_i = \perp_{1,i}$ for $i=1,2,\ldots,k$, $\top_{k+j} = \top\!\bullet\!\perp_{2j}$ for $j=1,2,\ldots,l$ and $|\top_{k+\ell+1}| = |\top\!\bullet\!\perp_{2,\ell+1}|$. A *canonical composition* of p_1 and p_2 by $\top$ is a path $p_1 \overline{/\top/} p_2$ obtained from $p_1 \overline{/\top/}' p_2$ after transforming it into canonical form.

Each path in a polygraph $\mathcal{G}$ with a free monoid of objects can be represented as a composition of elementary paths.

The λ*-polycategory of paths* over a polygraph $\mathcal{G}$ with a free monoid of objects is represented by the string

$$\mathcal{G}^* = \langle \mathbb{V}^*, \mathbb{P}, +, \bullet, \mathrm{dom}^*, \mathrm{cod}^*, \gamma, I\rangle$$

where $\mathbb{P}$ is the set of all canonical paths in $\mathcal{G}$, dom^* and cod^* put in correspondence the beginning and the end to each path respectively, γ is the canonical composition of paths and $I: \mathbb{V}^* \to \mathbb{P}$, $I: u \mapsto p_u$. One can easy see that it is a λ-polycategory with forks.

Let us denote by $U\mathcal{P}$ the polygraph obtained from a polycategory $\mathcal{P}$ after forgetting the composition law for the arrows and by $\mathrm{In}\,\mathcal{G} \doteq \langle \mathrm{In}\,\mathcal{G}_{\mathbb{O}}, \mathrm{In}\,\mathcal{G}_{\mathbb{A}}\rangle$ the inclusion morphism $\mathcal{G} \to U\mathcal{G}^*$ determined by the correspondences $\mathrm{In}\,\mathcal{G}_{\mathbb{O}}: v \mapsto v$, $\mathrm{In}\,\mathcal{G}_{\mathbb{A}}: x \mapsto p_x$.

PROPOSITION 2: $\mathcal{G}^*$ is a free λ-polycategory with forks over the polygraph $\mathcal{G}$ with free monoid of objects in the sense that for each λ-polycategory $\mathcal{P}$ and each polygraph morphism $\mathcal{F} = \langle \mathcal{F}_{\mathbb{O}}, \mathcal{F}_{\mathbb{A}}\rangle$ $\mathcal{F}: \mathcal{G} \to U\mathcal{P}$ there exists a unique functor (λ-polycategory morphism) $\mathcal{F}^*: \mathcal{G}^* \to \mathcal{P}$ such that the following diagram commutes

$$\begin{tikzcd}
\mathcal{G} \arrow[r, dash, "\mathrm{In}\,\mathcal{G}"] \arrow[dr, dash, "\mathcal{F}"'] & U\mathcal{G}^* \arrow[d, dash, "U\mathcal{F}^*"] & \mathcal{G}^* \arrow[d, "\mathcal{F}^*"] \\
 & U\mathcal{P} & \mathcal{P}
\end{tikzcd}$$

2. PARALLEL PROGRAMS

2.1. Let $G = \langle \mathbb{V}^*, \mathbb{A}, +, \bullet, \mathrm{dom}, \mathrm{cod} \rangle$ be a polygraph with a free monoid of objects $\langle \mathbb{V}^*, +, \bullet \rangle$ over a set of nodes $V = \{v_1, v_2, \dots\}$

A sequence $s \doteq x_1, x_2, \dots, x_m$ of arrows such that $\mathrm{cod}(x_i)$ and $\mathrm{dom}(x_{i+1})$, $(i=1,2,\dots,m-1)$ contain common nodes will be called a *string* of arrows.

An object $u \in \mathbb{V}^*$ is in *conformity* with another object $v \in \mathbb{V}^*$ iff for any two strings of arrows s_1, s_2 which lead from v to u the stituation of the codomains of the last arrows of s_1 and s_2 in u (to the left or to the right with possible overlapping) is the same as the situation of the domains of the first arrows of s_1 and s_2.

Two arrows x_1 and x_2 such that $\mathrm{dom}(x_1)$ and $\mathrm{dom}(x_2)$ contain common nodes will be called alternative arrows.

A polygraph G with two selected objects - a *start object*, u_s and an *end object*, u_e will be called a *program scheme* iff it satisfies the following conditions:

(i) - $\mathbb{A}$ and $\mathbb{V}$ are finite sets;

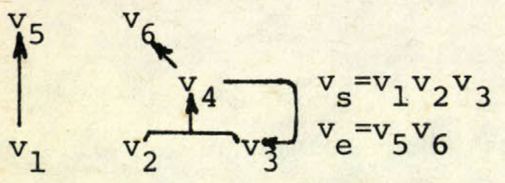
(ii) - if the domains and/or codomains of two arrows contain common nodes their participations form a subword of any of the domains or codomains under consideration;

(iii) - the end object is in conformity with the start object;

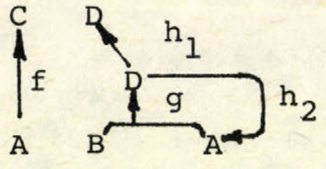
(iv) - any two strings of arrows which begin in the start object and finish with arrows with common nodes in codomains branch out through alternative arrows or one is contained in the other;

(v) - all the nodes which take part in the domain of some arrow but do not take part in the codomain of any arrow take part in the start object; there is no arrow such that a node which takes part in the end object takes part in its domain also.

A simple example of a (parallel) program scheme, represented in diagram form is shown on the next figure:



(a) a program scheme $\mathcal{G}$



(b) a program in the shape $\mathcal{G}$

Let $\mathcal{C}$ be any subcategory with forks of the polycategory Pfn such that all the functions in it are computable. By U$\mathcal{C}$ as usual we denote the corresponding polygraph obtained after forgetting the composition law for the arrows in $\mathcal{C}$.

A (deterministic) *program* in the shape of the program scheme $\mathcal{G}$ (or an interpretation of $\mathcal{G}$) is a functor $P: \mathcal{G} \to U\mathcal{C}$ such that the P-images of any two alternative arrows are functions whose ranges of definition (in the corresponding to the common nodes particular domains) are disjoint.

2.2. Following the ideas similar to that used in the Theory of Petry nets a concept of data flow through a program scheme $\mathcal{G}$ can be introduced as a token game:

- at the beginning (step 0) all the nodes in the start object are marked with active tokens;

- at the step t each arrow, which is not alternative with any other, with marked domain such that at least one of its nodes is marked with an active token can be activated to convert the tokens in its domain to passive and to give raise to active tokens in all nodes in its codomain; in alternative situations just one arrow can be selected to act as described;

- the data flow can be stopped at some step or it can stop naturally if there is no possibility for it to be continued.

Given an object u which is in conformity with the start object a data flow is a *data flow to u* if when it stops all the nodes in u are marked with active tokens and there are no other active tokens.

PROPOSITION 3: There is no data flow in a program scheme such that:

(i) - a node is supplied with an active token by two different arrows simultaneously;

(ii) - a node marked with an active token is supplied with an active token again before its dead.

Proof: Suppose the opposite for a node. In both cases following back the movement of the active tokens two strings s_1 and s_2 can be formed such that: their arrows have been activated; they begin from the start object; their last arrows have the node in common. They cannot branch out through alternative arrows. So one of them contains the other. In the case (i) inclusion is not possible (we want the node to be marked with an active tokens simultaneously). So s_1 and s_2 coincide. In the case (ii) because of the inclusion the token should be made passive before being forced to be active again.

An element of a set $A_1 x A_2 x \ldots x A_m$ will be called a *string of data* also. Each string of data $<a_1, a_2, \ldots, a_m>$ has $a_1, a_2, \ldots, a_m$ as components.

Given a string of data from the start set a *calculation according to a data flow* can be defined as follows:

- at the step 0 a string of data from the start object is available;

- at the step t a function is calculated for the available data iff it corresponds to an arrow which is activated in the data flow and the available data are in the domain of the function; in this case new avaliable data appear in its codomain according to (ii) the available data for the step t can be arranged in appropriate string of data.

A data flow (to u) is *adequate* to a string of data from the start set iff all the functions of the corresponding calculation can be calculated (in particular from two alternative arrows the arrow which correspondsto a function defined for the available data should be activated). A data flow (to u) is *punctual* for a string of data from the start set iff it is adequate to the string of data and the calculation can not be continued according to another adequite data flow (to u) containing the data flow under consideration.

Given a program P and an object u which is in conformity with the start object a *main calculation to u* over some string of data from the start set according P is a calculation according to a punctual (to the string of data) data flow to u in P.

For each string of data from the start set there is just one punctual data flow in P, at most one punctual data flow to u in P and so at most one main calculation to u.

A *result* of a calculation (to u) according to a given program P for the given string of data from the start set is a string of data received in the end set (the set corresponding to u) according to the main calculation to the end object (to u) if there is any and otherwise there is no result.

Propositin 3 allows us to state:

PROPOSITION 4: There is just one result of a calculation, if any, for a given string of data from the start set according to a program P.

The *function defined by calculation* according to a program is a function from the start set to the end set of the program which puts in correspondence to each string of data the result of the calculation over it. The *operational semantics* for programs supplies a program with the function defined by calculation as meaning.

2.3. The considerations implemented above intuitively suggest paths in a program scheme as formal descriptions of data flows.

Let P: $\mathcal{G} \to U\,\mathcal{C}$ be a program. According to Proposition 2 P has a unique extension to a functor P*: $\mathcal{G}^* \to \mathcal{C}$ from the λ-polycategory with forks of paths over $\mathcal{G}$. The set of the arrows in $\mathcal{G}$ is partially ordered by a relation "=":

$$p \subseteq q \quad \text{iff } p \text{ is an initial part of } q.$$

Given a program P: $\mathcal{G} \to U\mathcal{C}$ the relation $<P^*(v_s), P^*(v_e), \delta>$ where

$$\delta = \left\{ <a, [P^*(q)](a)> \;\middle|\; \begin{array}{l} q \text{ is a maximal (related to } \subseteq) \text{ path in } \mathcal{G} \\ \text{such that } P^*(q) \text{ is defined for a, } dom^*(q)= \\ =u_s, cod^*(q)=u_e \end{array} \right\}$$

will be called the *conceptual meaning* of P. The semantics which supplies each program with its conceptual meaning will be called *conceptual semantic*. (We use the term "conceptual" to distinguish between conceptual semantic, operational semantic and denotational semantic, the latter usually connected with fix-point constructions.)

PROPOSITION 5: Conceptual and operational semantics coincide.

Proof: Each path q from the start object to an object u which is in conformity with it describes a data flow. Looking on it from the right to the left it can be considered as a sequence of objects and arrows activated in consequent steps. According to the superposition rules in $\mathbb{P}$fn if $P^*(q)$ is defined for a $\mathbb{P}^*(u_s)$ then the calculation according these data flow gives as a result $[P^*(q)](a)$ in $P^*(u_e)$. (These statements can be proved more strictly by induction.)

Each (finite) data flow to an object u which is in conformity with the start object can be described as a (canonical) path q from the start object to u such that if the data flow is adequate to a string of data a then $P^*(q)$ is defined for a and the result of the calculation according to the data flow over it is $[P^*(q)](a)$.

To prove this statement we shall use induction on the number of steps in a data flow.

Suppose that all data flows of t steps fulfil the statement. Consider a data flow of t+1 steps to an object u which is in conformity with the start object. Let at the step t+1 the arrows $x_1, x_2, \ldots, x_n$ have been activated. Their codomains take part in u and do not overlap. If the order of situation of codomains coincides with the chosen order of arrows the object u is of the form

$$u = u_1 \mathrm{cod}(x_1) u_2 \mathrm{cod}(x_2) \ldots u_n \mathrm{cod}(x_n) u_{n+1}.$$

All the nodes in it are active and the nodes in codomains have been activated after the step t+1. This leads us to a conclusion that the object

$$w = u_1 \mathrm{dom}(x_1) u_2 \mathrm{dom}(x_2) \ldots u_n \mathrm{dom}(x_n) u_{n+1}$$

is in conformity with the start object (suppose the opposite and u will not be in conformity with the start object) and all the nodes in it have been activated after the step t. Restricting the data flow under consideration a data flow to the object w of t steps can be obtained. By induction hipothesis there is a path p fulfilling the statement for this data flow.

Consider the path

$$\begin{aligned} q = q' \circ p = <&(u_1\underline{cod(x_1)}u_2cod(x_2)\ \dots\ u_ncod(x_n)u_{n+1}),x_1,\\ &(u_1dom(x_1)u_2\underline{cod(x_2)}\ \dots\ u_ncod(x_n)u_{n+1}),x_2,\\ &\dots\dots\dots\dots\dots\dots\dots,\\ &(u_1dom(x_1)u_2dom(x_2)\ \dots\ u_ndom(x_n)u_{n+1})> \circ p. \end{aligned}$$

Since $P^*(q) = P^*(q') \circ P^*(p)$, according to the rules of superposition in Pfn the evaluatin of $P^*(q)$ for the string of data a can be done continuing the evaluation of $P^*(p)$ in the manner described by q'. So the result of executing of t+1(th) step of the calculation according to the given data flow on the result of the calculation till the step t will be obtained. That is just the result of the calculation according to the given data flow.

Moreover, if a data flow is punctual for a given string of data then the corresponding path is a maximal path defined for it (suppose the opposite and the data flow could be continued to another data flow to the same object adequate to the given string of data) and vice verse.

These statements related to the end object prove the proposition.

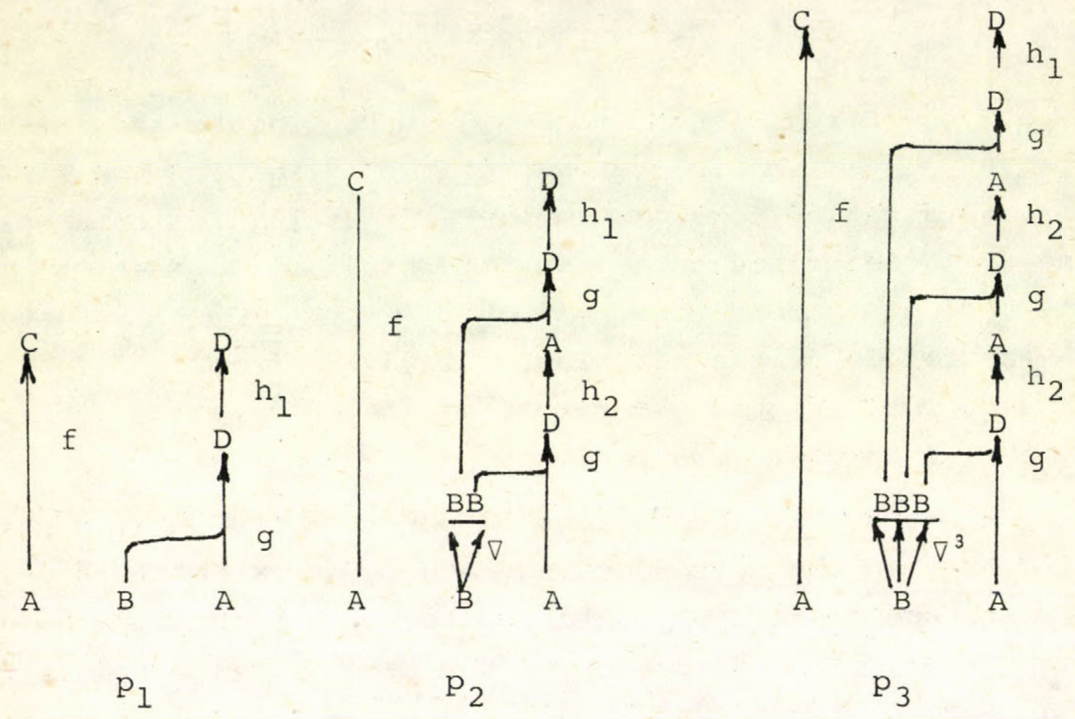
COROLLARY: The conceptual meaning of a program is a function.

The conceptual meaning of a program $P: \mathcal{G} \rightarrow U\mathcal{C}$ can also be defined as a relation $<P^*(u_s),P^*(u_e), \delta>$ where

$$\delta = U\{P^*(q)\,|\,dom^*(q) = u_s,\ cod^*(q) = u_e\}$$

The limitations of the objects which could be end objects insure the maximality of the paths. So the semantic is the same. But such a definition supplies other objects with relations instead of functions as meaning.

As an example, on the next figure the interpretations of three paths to $u_e = v_1 v_2$ for the program scheme and the program presented above are shown in diagram form.



3. CONCLUDING REMARKS

There are many other problems, like the problem of termination or the problem of equivalence which have not been discussed in the present paper. The reader could try to consider them following the papers [HCT, JCS].

The presented constructions are not perfect in several directions. First of all, there are two many restrictions on a polygraph to be a program scheme. Some of them can be avoided if more complicated polycategories (we shall need projections and transpositions [P]) are used. Moreover, it is desirable to find out "external" characteristics of some notions as "domain of definition" or "meaning of a program" instead of the elementwise used here. Such characteristics will give possibility to involve other polycategories on the place of $\mathbb{P}$fn.

REFERENCES

JCS - J.A.Goguen, J.W.Thatcher, E.G.Vagner, J.B.Wright, "A Junction between Computer Science and Category Theory, I,II" IBM Watson Res. Reports, 1973.

HCT - J.A.Goguen, "On Homomorphisms, Correctness, Termination, Unfoldments, and Equivalence of Flow Diagram Program", JCSS 8, 1974.

P.EP - Y.Velinov, "Polycategories. Elementary Properties" University Annual - Applied Mathematics, tome 17, book 1, Technica, Sofia, 1981.

PSMC I - Y.Velinov, "Polycategories and Strict Monoidal Categories I", University Annual - Applied Mathematics, tome 17, book 4, Technica, Sofia, 1981.

PSMC II - Y.Velinov, "Polycategories and Strict Monoidal Categories II", University Annual - Applied Mathematics, Technica, Sofia, 1984.

P - Y.Velinov, "Polycategories", Proc. "Second Symposium N-ary Structures", Varna, 1983.

CFP - Y.Velinov, "A Construction of Free λ-polycategories", Proc. "Second Symposium N-ary Structures", Varna, 1983.

PR - Y.Velinov, "Polycategories and Recursiveness", Proc. "First Symposium N-ary Structures", Skopje, 1982.

CAP - Y.Velinov, "Combinatorial Arrows in a Polycategory", Proc. "First Symposium N-ary Structures", Skopje, 1982.

Hálók, polikategóriák és a párhuzamos programok szemantikája

Y.P. Velinov

Összefoglaló

A szerző bizonyos összefüggésekre mutat rá, amelyek a Petri hálók és polikategóriák között van. Ezen összefüggések alapján a programok szemantikájának kategória-elméleti tárgyalását kiterjeszti a párhuzamos programokra.

Сети, поликатегории и семантики параллельных программ

И. П. Велинов

Р е з ю м е

В статье устанавливается связь между сетями Петри и поликатегориями. На основе этой связи расширяется категорический подход к семантике программ на более широкий класс параллельных программ.