Solving Systems of Equations Whose Variables are Organizational Forms

Spyros Vassilakis
University of Pittsburgh
4T18 Forbes Quad.
Pittsburgh, PA 15260

Running Title: Equations in Trees

Spyros Vassilakis
University of Pittsburgh
4T18 Forbes Quad.
Pittsburgh, PA 15260

* I would like to thank M. Ali Khan for encouraging me to write this note up and Lauree Graham for her technical support. The usual caveat applies.

1. Introduction

It is by now commonplace to state that the architecture of an organization is economically important; Williamson (1985), Geanokoplos and Milgrom (1988), and Stiglitz (1991), among others, have provided several convincing arguments to this effect. It is also commonplace to represent (informally) an organizational form by a tree (or a flowchart) that indicates the chains of command and the flows of information; see, for example, the pictures in Ch. 11 of Williamson (1985) and in pp. 75-77 in Chandler (1962). What is not done, to the best of my knowledge, is to model the choice of organizational form as the choice of a tree.

The space of trees is not a well understood, or even well-defined, object, and modelling the choice of a tree as some kind of optimization or equilibrium problem would require, at some point, the ability to solve equations on this space. The objective of this note if to advertise a theory of equations in trees developed in the 70s and 80s by computer scientists.

The note does not claim originality, but answers the following (hopefully important) questions. What is the set of all organizational forms that are possible in a given setting? What kind of structure should this set have to allow description of complex organizational forms in terms of simple recipes for their construction? Is there a simple criterion that distinguishes between meaningless recipes and recipes that (uniquely) describe some organizational form?

It is well beyond the scope of this note, however, to answer two related questions: given an interpretation of the symbols labelling the notes of the tree representing on organization form, what are the properties of the resulting organization? Which interpretations of these symbols are economically interesting? There is, however, some discussion of these questions in section 2, and a relevant theorem in section 5.

The note is organized as follows: section 2 presents an example, due to Sah and Stiglitz (1986), to motivate the constructions that follow. Section 3 shows how to describe organizational forms by (finite

or infinite) trees, and how to represent trees by partial functions. In the same section, the space of all organizational forms in a given environment is endowed with a structure that allows for a smooth representation of "combinations" of organizational forms by an operation that behaves like composition of functions. Section 4 defines systems of equations in organizational forms, and solutions of such systems. The space of forms is endowed with an order structure that allows us to solve such systems by an application of an order-theoretic fixed point theorem; there is also an algorithm for obtaining solutions inductively. Section 5 looks very briefly at semantics of organizational forms. Section 6 concludes with a short guide to related literature.

2. An Example

In this section I illustrate informally, in terms of an example due to Sah and Stiglitz (1986), the representation of organizational forms by trees, inductive and recursive definitions of organizational forms, and solutions of fixed-point equations whose arguments are organizational forms.

Consider a population of projects that can be either good or bad. An organization samples this population and decides whether to accept or reject the projects sampled. Sah and Stiglitz consider two particular organizational forms. A two-member hierarchy consists of two managers. Each manager recommends acceptance of a project with probability z. If the first manager to look at the project rejects it, the hierarchy also rejects it; otherwise, the project is evaluated by the second manager, whose decision is final. To represent this organizational form as a tree, let $p\colon I^3 \to I$ be defined by

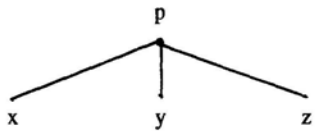
$$p(x, y, z) = zx + (1-z)y, \quad I = [0,1] \tag{1}$$

We interpret p as follows: if a manager recommends rejection, the organization will accept the project with probability y and reject it with probability 1-y. If a manager recommends acceptance, the

organization will accept the project with probability x and reject with probability 1-x. Then, the two member hierarchy H_2: [o,1] → [o, 1] is defined by

$$H_2(z) = p(z, o, z) = z^2. \tag{2}$$

We now adopt the following notational convention. The tree



or, equivalently, the term pxyz, will be taken to represent the function p: I^3 → I, (not the value of p at (x, y, z)). Similarly, the term pzoz represents the function from I to I that maps z to p(z, o, z) = z^2. Then, the two-member hierarchy is defined by

$$H_2 = pzoz \tag{3}$$

Similarly, a two-member polyarchy (an organizational form that rejects a project only if both managers recommend rejection), can be written as

$$\Pi_2 = p1zz \tag{4}$$

$$\text{or, } \Pi_2(z) = z + (1 - z)z. \tag{5}$$

We interpret $H_2(z)$ and $\Pi_2(z)$ as the probabilities that the two-member hierarchy, respectively polyarchy, will accept a project given that each manager recommends acceptance with probability z. It is not difficult to see that the (n + 1) member hierachy and polyarchy are given by, respectively,

$$H_{n+1} = pH_noz \tag{6}$$

$$\Pi_{n+1} = p1\Pi_nz \tag{7}$$

We interpret (7), for example, as follows: In the n+1- member polyarchy a project is accepted if the first manager to look at it recommends acceptance; if, however, he recommends rejection, the project is evaluated by the n-member polyarchy.

All the definitions of organizational forms up to now were either direct (3 and 4) or inductive (6 and 7). To define hierarchies and polyarchies with no restriction on the number of managers, we will need to use recursive definitions. The hierarchy form H is defined by

$$H = pH_n oz \tag{8}$$

and the polyarchy form Π by

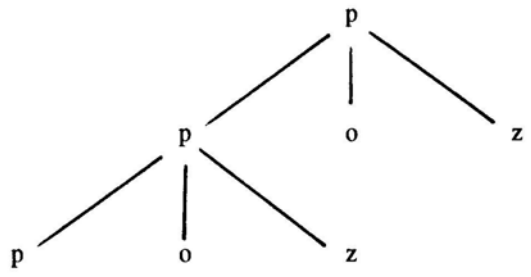
$$\Pi = p1\Pi z \tag{9}$$

We interpret (8), for example, as follows: if a manager recommends rejection, H rejects the project otherwise the project is evaluated again by the hierarchy H. We will shortly see that the solution of (8) is the limit of the solution of (6) as $n \to \infty$, and similarly for Π.

Expressions (8) and (9) are fixed-point equations, but on which space? Here we have a choice. We can choose to treat the symbols p, z, o, 1 as just symbols without any particular meaning, and consider equations (8) and (9) as defined on the space of all trees whose nodes are labelled by these symbols. The solutions of (8) and (9), then, are going to be infinite trees (obtained as limits of the finite trees generated by iterating (8) and (9).) For example, by iterating (8), we get a sequence of deeper and deeper trees

$$pHoz,\ p(pHoz)oz, \tag{10}$$

whose "limit" is the infinite tree that represents the general hierarchy form:



(11)

This is a solution at the syntactic level. (note that the infinite tree (11) does not contain any instance of H, as it should). We can now interpret the symbols p, o, z in (11) to obtain a solution of the semantic level, i.e., a function H: I → I defined by

$$H(z) = \lim_{n \to \infty} H_n(z) = \lim_{n \to \infty} z^n = \begin{cases} o & \text{if } z < 1 \\ 1 & \text{if } z = 1 \end{cases} \tag{12}$$

On the other hand, we could choose to first interprest p, z, o, and then solve the recursive equation (8), now mapping the space I^I (of functions from I to I) into itself. The equation can be written as

$$H(z) = (1-z)o + zH(z) = zH(z) \tag{13}$$

and it has two solutions: H = o, and H(1) = 1, H(z) = o if z < 1. We select the second solution, because it is the limit of the sequence of hierarchies with finitely many members.

Solving at the syntactic level has two (interrelated) advantages. The syntactic solution is a tree that describes the way decisions are made in an organization; this description is independent of the meaning of the symbols that label the nodes of the tree. Hence, we can be precise when we say, for example, that two organizations that perform very different functions have the same architecture. Furthermore, the semantic solution (i.e., eq. (12)) supresses all the information contained in the syntactic solution: an equation like (12) describes the "input-output" behavior of an organization, not the process by which the output is obtained from the input. In other words, the semantic solution can be recovered from the syntactic solution (see also section 5), but not vice-versa.

How can we make sure that equations like (8) and (9) have (unique) solutions? To answer this, we will have to specify the space on which such equations are defined, and in which solutions live. This is done in the next section.

3. The Space of Organizational Forms

In section 2, the symbol p was always interpreted as a function mapping Q^3 into Q, for some set Q, while o, 1 were always interpreted as elements of Q. We say that p has rank three and o, 1 have rank zero. In a general setting, for each natural number $n > o$ we will have a set $\sum_n$ of operation symbols of rank n to label trees with.

In the Sah-Stiglitz case

$$\Sigma_o = \{o, 1\}$$

$$\Sigma_3 = \{p\}$$

$$\Sigma_n = \phi \qquad n \neq o, 3.$$

The set $\sum = \bigcup_{n=o}^{\infty} \sum_n$ describes the operations that are available to build organizational forms with.

The trees in section 2 were labelled either with elements of Σ or with variables like z, H and Π. In equations like (8) and (9), we solve for H or Π, while z is treated like a parameter. We are going to assume, without loss of generality, that all variables belong to a countable set $X = \{x_1, x_2, \ldots\}$; to distinguish variables from operation sysmbols, we assume that X, Σ are disjoint in each particular equation we will distinguish those x_i's that are "unknowns" from those that are parameters. For example, equations (8) and (9) will now be written as follows

$$x_1 = px_1ox_2 \tag{8'}$$

$$x_1 = p1x_1x_2 \tag{9'}$$

In both equations, x_1 is an unknown and x_2 is a parameter.

Having settled this notational point, we now identify each tree labelled with elements of $\Sigma \cup X$ with a partial function mapping strings of natural numbers into $\Sigma \cup X$. Let ω be the set of natural numbers 1, 2, ...; a string u is a finite sequence $u_1 \ldots u_n$, $u_i \in \omega$ for all i = i, .., n; the set of strings is denoted by ω^*;

the empty string is denoted by Λ. Note, then, that the trees in equations (8′) and (9′) can be represented, respectively, by two partial functions $t_H, t_\Pi \quad \omega^* \rightarrow \Sigma \cup X$ defined by $t_H(\Lambda) = p, t_H(1) = x_1, t_H(2) = o, t_H(3) = x_2\, t_H(v) =$ undefined for all other $v \in \omega^*$; $t_\Pi(\Lambda) = p$, $t_\Pi(1) = 1, t_\Pi(2) = x_1, t_\Pi(3) = x_2$, $t_\Pi(v) =$ undefined for all other $\nu \in \omega^*$. The infinite tree in equation (11) is given by H: $\omega^* \rightarrow \Sigma \cup X$, where

$$H(v) = \begin{cases} p & v = \Lambda, 1, 11, 111, \ldots \\ o & v = 2, 12, 112, 1112, \ldots \\ z & v = 3, 13, 113, 1113, \ldots \end{cases}$$

In general, if t: $\omega^* \rightarrow \Sigma \cup X$, $\nu \in \omega^*$, $i \in \omega$, and if $t(\nu)$, $t(\nu i)$ are defined, we say that νi is the i-th immediate success or node of node ν. Note that the rank of $t(\nu)$ has to be larger than or equal to i for this to make sense, because a symbol of rank n can have at most n immediate successors. Hence our first definition

Definition 1 A tree in the environment described by Σ (a Σ- tree) is a partial function t: $\omega^* \rightarrow \Sigma \cup X$ such that if $\nu \in \omega^*$, $i \in \omega$, and if t (νi) is defined, then there is some $n \geq i$ such that $t(v) \in \Sigma_n$. (in particular, $t(\nu)$ is defined).

The set of all trees will be denoted by $CT_\Sigma(X)$; the set of all trees that map into $\Sigma \cup X_k$, where $X_k = \{x_1, \ldots, x_k\}$ will be denoted by $CT_\Sigma(X_k)$; if t is a tree, dom (t) will stand for its domain of definition. We now impose additional structure on $CT_\Sigma(X)$ that will allow us to compose trees correctly and to distinguish between variables and parameters in equations like (8') or (9').

Definition 2: t is a morphism with source 1 and target k, *t: 1 → k, iff* $t \in CT_\Sigma(X_k)$. Hence, t: $1 \rightarrow k$ if t is a tree with one root whose terminal nodes, if any, are labelled with elements of $X_k \cup \Sigma_o$.

Definition 3: t is a morphism with source $n \geq o$ and target $k \geq o$, $t: n \to k$, if $t = (t_1, \ldots, t_n)$ and $t_i: 1 \to k$ for each $i = 1, \ldots, n$. We denote by $CT_\Sigma(n, k)$ the set of these morphisms.

In particular, if $n = o$, there is a u nique morphism $o \to k$ for each k (the empty tree), and if $k = o$, then a morphism $t: 1 \to o$ is a tree whose terminal nodes are labelled only with elements of Σ_o (not of X).

Note that for each $k \geq 1$, $CT_\Sigma(1, k)$ contains k morphisms $\tilde{x}_i: 1 \to k$ defined by the partial functions $\tilde{x}_i: \omega^* \to \Sigma \cup X$, where $\tilde{x}_i(v) =$ undefined *if* $v \neq \bigwedge$, $\tilde{x}_i(\bigwedge) = x_i$.

Having attached to each tree a source and a target, we show how to compose two trees with compatible sources and targets. To motivate the definition, let $t: 1 \to 2$ by given by $t = px_1ox_2$ and $t': 2 \to 3$ be given by $t' = \langle t'_1, t'_2, \rangle$ $t'_i: 1 \to 3$, and $t'_1 = fx_1, x_2, x_3$; $t'_2 = gx_2, x_3, x_1$.

Then tt′ is obtained by attaching t'_1 to the leaf of t labelled x_1 and t'_2 to the leaf of t labelled x_2, to end up with the composite tree $tt' = p(fx_1, x_2, x_3)o(gx_2, x_3, x_1$. The next definition captures this process of composition.

Definition 4: If $t: n \to k$ and $t': k \to q$ are morphisms, then their composite $tt': n \to q$ is the n-tuple $(s_1, \ldots, s_n)$, $s_i: 1 \to q$, defined by

$$s_i(w) = \begin{cases} t_i(w) & \text{if } t_i(w) \in \sum_n \text{ for some } n \geq o \\ t'_j(v) & \text{if } w = uv \text{ and } t_i(u) = x_j \ 1 \leq j \leq k \\ \text{undefined} & \text{otherwise.} \end{cases}$$

It is not difficult to show that

Fact 1: For each k, the morphism $(x_1, \ldots, x_k): k \to k$ behaves as an identity with respect to composition i.e.

$(x_1 \ldots x_k)t = t$, *if* t: $k \to q$
$t(x_1 \ldots x_k) = t$, *if* t: $n \to k$.

This result motivates

Definition 5: For each $k \geq o$, id_k: $k \to k$ is defined to be the morphism $(x_1, \ldots, x_k)$.

4. Systems of Equations in Trees

We will now use this structure to define formally a system of equations, and a solution of such a system. We will represent equations line (8') and (9") by their right-hand side only. The right-hand side of (8') is t_H: $1 \to 2$, where $t_H = px_1ox_2$. To separate unknowns from parameters, we write t_H: $1 \to 1 + 1$, or, in general, t: $1 \to 1 + p$: this means that t represents the equation $x_1 = t$ and not $x_i = t$, $i = 2, \ldots, 1 + k$. A system of n equations in n unknowns and k parameters is, then, a morphism α: $n \to n + k$. If $\alpha = \prec \alpha_1 \ldots \alpha_n \succ$, $\alpha_i : 1 \to n + k$ then α represents the system

$$x_1 = \alpha_1, x_2 = \alpha_2 \ldots x_n = \alpha_n$$

Definition 6: A system of n equations in n unknowns and k parameters is a morphism α: $n \to n + k$. Note that the definition requires the number of unknowns to be equal to the number of equations.

To see what a solution of α is, recall that the solution of t_H was the tree H in (11), and that H: $1 \to 1$ (or in general, $n \to k$), is a morphism that satisfies $t_H(H, id_1 = H$, i.e., $pHox_2 = H$ (recall the definition of composition and of id_1) Hence, in general

Definition 7: A solution of α: $n \to n + k$ is a morphism ß: $n \to k$ such that $\alpha(ß, id_k) = ß$.

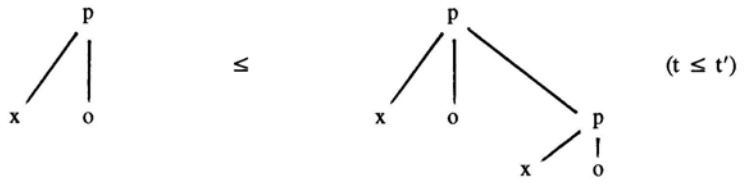
This equation can be written as $\psi_\alpha(\beta) = \beta$, where $\psi_\alpha: CT_\Sigma(n, k) \to CT_\Sigma(n, k), \psi_\alpha(\beta) = \alpha(\beta, id_k)$. To solve this fixed point equation we endow each CT_Σ (n, k) with an order structure and then show that ψ_α has continuity properties that guarantee the existence of a least fixed point.

Definition 8: If t, t′: $1 \to p$ then $t \leq t'$ if

(a) dom (t) $\subseteq$ dom (t′)

(b) $t(\eta) = t'(\eta)$ for each $\eta \in$ dom (t).

(Recall that t, t′ are partial functions and dom (t) is the domain of definition of t). An example might clarify this defintion.



because dom (t) = {Λ, 1, 2}, dom (t′) = Λ, 1, 2, 3, 31, 32, and

$$t(\Lambda) = p = t'(\lambda),\ t(1) = x = t'(1),\ t(2) = o = t'(2).$$

We order CT_Σ (n, k) componentwise, and we obtain

Fact 2: CT_Σ (n, k) is a poset with

(a) a least element $\perp$, the totally undefined tree, and

(b) least upper bounds of increasing sequences

Proof: (a) is immediate. We prove (b) for n = 1. Let $\prec t_i \succ$ be an increasing sequence in $CT_\Sigma(1, k)$ and let t be defined by $dom(t) = \bigcup_{i=1}^{\infty} dom(t_i)$ and $t(\nu) = t_i(\nu)$ if $\nu \in dom(t_i)$. Then t is the least upper bound of the sequence $\prec t_i \succ$, and we write $t = \bigsqcup_{i=1}^{\infty} t_i$.

A poset with properties (a) and (b) is called a complete partial order (cpo).

We will now show that the function ψ: $CT_\Sigma(1, 1) \to CT_\Sigma(1, 1)$ that defines the hierarchy form (8') has some convenient continuity properties. By (8'), ψ is defined, for each $\beta \in CT_\Sigma(1, 1)$, by

$$\psi(\beta)(\nu) = \begin{cases} p & \text{if } \nu = \Lambda \\ \beta(w) & \text{if } \nu = 1w \\ o & \text{if } \nu = 2 \\ x_2 & \text{if } \nu = 3 \\ \textit{undefined} & \textit{otherwise} \end{cases}$$

Fact 3: ψ preserves least upper bounds of increasing sequences.

Proof: If $\prec t_i \succ$ is an increasing sequence and $t = \bigsqcup t_i$, we want to show that $\psi(t) = \bigsqcup \psi(t_i)$.

$$dom \bigsqcup_{i=1}^{\infty} \psi(t_i) = \bigcup_{i=1}^{\infty} dom\psi(t_i) = \bigcup_{i=1}^{\infty} \{1w: w \in dom(t_i)\} \cup \{\Lambda, 2, 3\} = \{1w: w \in dom\ (t)\} \cup \{\Lambda, 2, 3\} = dom\psi(t)$$

The first equality follows from the proof of 1b; the second and the fourth from the definition of ψ; and the third from the definition of $\sqcup$; furthermore if $\nu \in$ dom $\psi(t)$ and $\nu = \lambda, 2, 3$ then obviously $\psi(t)(\nu) = \psi(t_i)(\nu) = \bigsqcup_{i=1}^{\infty} \psi(t_i)(\nu)$. If $\nu = 1w$, $w \in dom(t)$, then $w \in dom(t_i)$ for *some* i, and

$$\psi(t)(\nu) = t(w) = t_i(w) = \psi(t_i)(\nu) = \bigsqcup_{j=1}^{\infty} \psi(t_j)(\nu).$$

The first and the third equalities follow from the definition of ψ; the second and the fourth from the definitions of t and $\bigsqcup$.

Hence $\psi(t) = \bigsqcup_{j=1}^{\infty} \psi(t_j)$.

This result motivates

Definition 9: Let A be a cpo. Any function f: A → A that preserves least upper bounds of increasing sequences is called Scott continuous.

Scott continuous functions have the property we need to solve equations, as shown by

Fact 4: If f: A → A is Scott continuous, it has a least fixed point m(f), that is equal to $\bigsqcup_{n=o}^{\infty} f^n(\perp)$.

Proof: $f(m(f)) = f(\bigsqcup f^n(\perp)) = \bigsqcup f^{n+1}(\perp) = \bigsqcup f^n(\perp) = m(f)$.

To see that m(f) is the least fixed point, let f(a) = a. Then, $\perp \leq a$ and by induction $f^n(\perp) \leq a$ for all $n \geq o$, i.e., $m(f) \leq a$.

Combining theorems 3 and 4, we obtain

Corollary 1: The recipe ψ for the construction of the hierarchy form has a least fixed point.

The least fixed point $m(\psi)$ of ψ can be explicitly described by

$$m(\psi)(\nu) = \begin{cases} p & \text{if } \nu = \Lambda, 1, 11, 111, .. \\ o & \text{if } \nu = 2, 12, 112, 1112, .. \\ x_1 & \text{if } \nu = 3, 13, 113, 1113, .. \\ \textit{undefined} & \textit{otherwise} \end{cases}$$

Note that this is the same tree as that in equation (11)

Abstracting from this example, we will need to show that each morphism α: n → n + k defines a Scott continuous ψ_α: CT_Σ (n, k) → CT_Σ (n, k), where $\psi_\alpha(\beta) = \alpha(\beta, id_k)$ (see definition 7)

This is the content of

Fact 5: ψ_α is Scott continuous for each morphism α: $n \to n + k$.

Proof: Wagner et.al. (1976), theorem 4.5 and proposition 5.4.

5. Semantics of Organizational Forms

As long as the operation symbols in Σ remain uninterpreted, a Σ-tree describes the structure of an organization form but not the "behavior" of the organization with such a structure. For example, (11) describes the structure of a hierarchy but it provides no information about the probability that a hierarchy will accept a project as a function of the probability that a manager will accept a project. Hence the need for semantics

Definition 9: An interpretation (Q, δ) of Σ is a cpo Q and, for each $\sigma \in \Sigma_n$, a Scott continuous $\delta(\sigma)$: $Q^n \to Q$; if $n = o$, $\delta(\sigma)$ is an element of Q.

The main result of semantics in this context is

Fact 6: Each interpretation (Q, δ) of Σ has a unique extension to an interpretation $(Q, \hat{\delta})$ of $CT_\Sigma(X)$ that satisfies

(a) if t: $1 \to k$ then $\hat{\delta}(t)$: $Q^k \to Q$ is Scott continuous (note the inversion)

(b) if t: $n \to k$, $t = (t_1 \quad t_n)$, then $\hat{\delta}(t)$: $Q^k \to Q^n$ equals $(\hat{\delta}(t_1), \quad ,\delta(t_n))$

(c) if x_i: $1 \to k$ then $\hat{\delta}(x_i)$: $Q^k \to Q$ is the i-th projection

(d) $\hat{\delta}$ preserves composition

(e) $\hat{\delta}(\perp) = \perp$

(f) $\hat{\delta}$ is Scott-continuous

Proof: Wagner et.al. (1976), theorem 4.5.

This theorem allows us to find the properties of an organizational form defined by a recipe ψ. (The least fixed point t: n → k of ψ is uniquely interpreted as a function $\hat{\delta}(t)$: $Q^k \to Q^n$ that represents the "input-output" behavior of the organization described by ψ).

Not all interpretations of Σ are interesting; for example, Q might be a one-element set. The development of economically important semantics is well beyond the scope of this note.

6. Conclusion

A wide class of fixed-point equations in trees have (unique) least fixed points. The result was obtained by endowing the space of trees with two structures. First we turned the space of trees into a category with objects the natural numbers and morphisms tuples of trees. Then we endowed the sets of morphisms with a partial order, turning them into cpo's. Finally we showed that each fixed point equation was a Scott-continuous function mapping some set of morphisms into itself. An order-theoretic fixed point theorem guaranteed existence of least fixed points and provided an algorithm for their construction.

Similar results can be obtained by endowing the set of trees with a metric: the same class of equations are strict contractions with respect to this metric, and an application of Banach's fixed point theorem yields existence and uniqueness of solutions: see Bloom (1983).

More general organizational forms can be obtained if we allow for operations interpreted as functions defined on products of different sets, rather than products of the same set (see definition 10). The same theory of equations goes through with minimal notational changes: see Wagner et. al. (1985), who also solve systems of inequalities in trees. Still more general organizational forms can be specified by the method of sketches, developed in Barr and Wells (1990).

Finally, a general theory of specifications at the syntactic level and their meaning at the semantic level has been recently developed by Goguen and Burstall (1991) under the name of "institutions".

References

1. M. Barr and C. Wells (1990) Category Theory and Computing Science. Prentice Hall, New York.

2. S. Bloom (1983) All solutions of a System of Recursion Equations in Infinite Trees and Other Contraction Theories. Journal of Computer and System Sciences 27, 225-255.

3. A. Chandler (1962): Strategy and Structure. MIT Press, Cambridge, MA.

4. J. Geanokoplos and P. Milgrom: A theory of Hierarchy Based on Limited Managerial Attention. Forthcoming in J of Jap and Int Econ.

5. J. Goguen and R. Burstall (1991) Institutions: Abstract Model Theory for Specification and Programming. Forthcoming in J of the Assoc of Comp Mach.

6. Sah, R. and J. Stiglitz (1986) The Architecture of Economic Systems: Hierarchies and Polyarchies. AER, 76: 716 - 27.

7. J. Stiglitz (1991) Sumposium on Organizations and Economics. J of Econ Pers, 5(2): 15-24.

8. E. Wagner, J. Wright, J. Goguen and J. Thatcher (1976) Some Fundamentals of Order - Algebraic Semantics. Lecture Notes in Computer Science 45: 153-168 , Springer, Berlin.

9. E. Wagner, S. Bloom, J. Thatcher (1985) Why Algebraic Theories? In, M. Nivat and J. Reynolds (eds.), Algebraic Methods in Semantics, Cambridge Univ. Press.

10. Williamson, O (1985) Economic Institutions of Capitalism, The Free Press, New York.