An Analysis of Controllable Processes with Uncertainties

Rudolf F. Albrecht

University of Innsbruck, Institute for Informatics

Rudolf.Albrecht@uibk.ac.at

Abstract

Presented is an abstract model of controllable processes on partial ordered times. Emphasized is the definition of structures, of variables and their control, of uncertainty, and the study of partial ordered times and their mutual relationships. This is applied to describe evolutionary processes in general spaces, modeling physical processes by taking causality and operational time lag into account.

Keywords: Mathematical systems theory, theory of processes on partial ordered time, processes with uncertainties, control theory, evolutionary processes.

1. Introduction

To show the occurrence of processes on partial ordered times and to clarify the problem classes and concepts we are dealing with, we consider the following examples:

Example 1: Let $(A, <_A) = \{a<b<c<d\}$ and $(B, <_B) = \{1<2<3<4<5\}$ be two strictly ordered independent processes, for example operation sequences on a computer, driven by independent clocks "a" and "b", respectively. For process intercommunication, at 2 a message is to send to a, which a is waiting for. This establishes an order relation $2 \prec a$. By another intercommunication may be defined $c \prec 5$. Then on $A \cup B$ a partial ordering $<_A \cup <_B \cup \prec$ is defined as "global" time. A planned intercommunication $4 \prec a$, $c \prec 2$ causes a circle and does not create a global time. Process A waits for process B and B waits for A (a "dead lock"). This is all well investigated in the theory of multi-tasking and parallel processing computers.

Example 2: We consider the class $\mathcal{F}$ of functions f: $[0, 5] \rightarrow \mathbf{R}$, $\mathbf{R}$ the real numbers, $[0, 5] \subset \mathbf{R}$ a closed interval, in particular the subclass of polynomials $\mathcal{Pol}(n)$ of degree n. $\mathcal{Pol}(2, [0, 5])$ is described by $y = ax^2 + bx + c$, x is a variable on $[0, 5]$, $a \in \mathbf{R}\backslash\{0\}$, $b \in \mathbf{R}$, $c \in \mathbf{R}$ are variables, y is a variable on $\mathbf{R}$, depending on "object" variable x and on "control" variable (a,b,c), both are independent. $([0, 5], <)$ represents a "time".
1. Initial value problem: Present time instant is $t = 0$, posed is $y(0) = 0, y'(0) = 1$. There are infinite many solutions $c = 0, b = 1$, variable a is uncertain, i.e. domain $\mathbf{R}\backslash\{0\}$ is known, but no value is determined. Posing $y(0) = 0, y'(0) = 1, y''(0) = 2$, this determines

one solution $y = x^2$+x. The derivatives imply information about objects in the "nearest" future $t > 0$; however, this problem representation is not understood to be anticipatory.
2. Anticipatory representations contain information (function- and derivative values or functions of these, or domain restrictions) about objects in a "farer" future $0<t\leq 5$. Posing $y(0) = 0$, $y(4) = 0$, yields $c = 0$ and infinite many solutions. Adding a third condition, (max $y(x)$) = 4 solutions are $y = ax(x-4)$ for $a = -1$ and $a \geq 4/5$, i.e. infinite many. Adding a fourth condition $y(5) < 0$, we find $y = x(4-x)$. For this example, the anticipatory representation can be replaced by the initial value problem $y(0) = 0$, $y'(0) = 4$, $y''(0) = -2$. Mixed initial- and boundary value problems for linear differential equations are extensively treated in text books.
3. For discrete time $\ldots t_{n-1}, t_n, t_{n+1}, \ldots$ a representation $x_{n+1} = F(\ldots x_n, x_{n+1})$, $\ldots x_n$ known, x_{n+1} the successor of x_n, is not anticipatory but a fix point equation for x_{n+1}, which may have no, one, or many solutions. For example: For real numbers, the equation $x_{n+1} = 5+x_n+x_{n+1}$ has no solution if $5+x_n \neq 0$, and infinite many solutions for $5+x_n = 0$. If it is sure that a fix point exists, one can try to solve the fix point equation $x = F(\ldots x_n, x)$ approximately by iterations $(x^{(m+1)} = F(\ldots x^{(m)}))_{m=0,1,2,3,\ldots}$. This sequence need not converge, or may converge but not to the fix point x wanted. Examples for real functions: For equation $x = -x$, the fix point is 0, it cannot be reached by iterations starting with $x^{(0)} \neq 0$. For equation $x = x^2$, fix points are 0 and 1, starting with $x^{(0)} \neq 1$, iterations either diverge or converge to 0, but not to 1.

2. Structures

For any two sets I, S and any function $\iota: I \rightarrow S$ we use the following notations: ι is an indexing, I is a set of indices (e.g. names, addresses, co-ordinates), S is a set of objects, $s_{[i]} =_{\text{def}} \iota(i) = s(i)$, $s_i =_{\text{def}} (i, s_{[i]})$, $(s_i)_{i\in I} =_{\text{def}} \{(i, s_{[i]}) \mid i\in I\} \subset I\times S$ denotes a "family" (or parameterized set). $(s_i)_{i\in I}$ represents the function ι and is also denoted by s_I with $s_{[I]} = \iota(I)$ if ι is unmistakably specified. $s_\varnothing = \varnothing$. In case $I = \{i\}$ we write also s_i for $s_{\{i\}}$. If the set S is itself indexed by k, we write s_{ki} for an element of S_k. Notice, without reference to an indexing function ι, notations like s_i, si, is, etc. can denote elements of a set. Then distinct elements need distinct names. However, in a family for $s_i \neq s_j$, $s_{[i]} = s_{[j]}$ is possible. The reciprocal $\iota^{-1}(s_{[i]})$ of $s_{[i]}$ is a subset of I and in general not a singleton.
Let there be given a family $(S_i)_{i\in I}$ of sets $S_{[i]}$ with $I \neq \varnothing$, $S_{[i]} \neq \varnothing$. We define: $\boldsymbol{S} =_{\text{def}} \bigcup_{i\in I} S_i$, for any $J \subseteq I$ the (in general unordered) set product $\prod_{i\in J} S_i =_{\text{def}} \{(s_i)_{i\in J} \mid s_i\in S_i, i\in J\}$, $\boldsymbol{S}^* =_{\text{def}} \bigcup_{J\subseteq I} \prod_{i\in J} S_i$. $\prod$ differs from the Cartesian product $\times$, which is ordered and finite. If all $S_{[i]}$ equal the set S we write S^I for $\prod_{i\in I} S_i$. Any $R \subseteq \boldsymbol{S}^*$ defines a "structure" (or "general relation") on $\boldsymbol{S}$. Classically, a relation is defined as a subset of one Cartesian product.

The basic selection- and composition operations on sets, like subset selection $\subseteq$, intersection $\cap$, union $\cup$, difference \, symmetric difference Δ, apply to structures R of $\boldsymbol{S^*}$ as sets of families, and to families as sets of pairs $(i, s_{[i]})$. For illustration, we consider singleton relations $R = \{r_U\} \subset \boldsymbol{S^*}$, $r_U = (r_i)_{i\in U}$, $R' = \{r'_V\} \subset \boldsymbol{S^*}$, $r'_V = (r'_j)_{j\in V}$. Denotation $\wedge$ means "and", denotation $\vee$ means "or".

(1) For $R \neq R'$ holds: $R'' =_{\text{def}} R \cup R' = \{r_U, r'_V\}$, $R \subseteq R''$, $R \cap R' = \varnothing$.

Considering the families r_U, r'_V, examples for operations are:

(2.1) Selection by subsets of indices: For any $K \subseteq I$ the "projection of r_U onto K" is $pr(K; r_U) =_{\text{def}} (r_i)_{i\in U\cap K}$ (compare with "call by name").

(2.2) Selection by subsets of objects: For $T \subset \bigcup_{i\in I} S_{[i]}$ the "selection by T out of r_U" is

$sel(T; r_U) =_{\text{def}} (r_i)_{i\in W} =_{\text{def}} \{r_j \mid r_j \in r_U \wedge r_{[j]} \in T\}$ (compare with "call by value").

(3.1) Intersection of families: $r_U \cap r'_V = (r''_j)_{j\in W} =_{\text{def}} \{r''_j \mid r''_j = r_j = r'_j,\ j\in U\cap V\}$.

(3.2) The union of families: $r_U \cup r'_V$ yields a family if r_U and r'_V coincide on $U\cap V$. Then $(r''_j)_{j\in U\cup V} =_{\text{def}} \{ r''_j \mid r''_j = r_j$ for $j\in U\setminus V$, $r''_j = r'_j$ for $j\in V\setminus U$, $r''_j = r_j = r'_j$ for $j\in U\cap V\}$. If functions $(\sigma_i : (r_i, r'_i) \to (r''_i))_{i\in U\cap V}$ are given, a family on $U\cup V$ can be obtained. We assume σ_i = identity for $r_i = r'_i$.

These operations can be extended to families $(R_j)_{j\in J}$ of general structures $R_j \subseteq \boldsymbol{S^*}$ by application to all component families of the R_j. For any $(R_j)_{j\in J}$ with cardinality $J>1$ we name functions $K: (R_j)_{j\in J} \xrightarrow{\text{onto}} R$, $R \subseteq S^*$, "concatenations" (infix notation κ). In particular, if for any J–tupel $((r_{ji})_{i\in I(j)})_{j\in J}$, $(r_{ji})_{i\in I(j)} \in R_j$, a commutative group operation $\Sigma_i\,(r_{ji})_{j\in J} = r_i$ exists, $K((r_{ji})_{i\in I(j)})_{j\in J})$ may be defined as $(r_i)_{i\in I}$, $I = \bigcup_{j\in J} I(j)$. K is then a commutative group operation. Examples for Σ are for sets: union $\cup$ or intersection $\cap$, for lattices: join $\vee$ or meet $\wedge$, for additive groups: $+$. Reverse to concatenation κ is a partition π of a family s_W into part families. The objects of the sets $S_{[i]}$, $i\in I$, and the indices can themselves be parameterized structures of other objects and other indices, which are then on lower structural hierarchical level than the previous ones. For example, a set $\{M_{[i]} \mid i\in I\}$ of sets $M_{[i]}$ or a family $((m_{ji})_{j\in J(i)})_{i\in I}$ of families $(m_{ji})_{j\in J(i)}$ are of higher hierarchical level than their constituents. The union $\bigcup_{i\in I} M_{[i]}$ and a concatenation $\boldsymbol{K}(((m_{ji})_{j\in J(i)})_{i\in I}) = (m_{ji})_{ji\in U}$, $U =_{\text{def}} \bigcup_{i\in I} J(i)\times\{i\}$, reduce the hierarchical level.

3. Variables and their Control

We consider a non-empty family $(r_p)_{p\in P}$ of non-empty structures (i.e. general relations) $r_{[p]}$. For all p let be $r_p = K_p(c, v_{[p]})$, $K_{[p]}$ being a concatenation, which can depend on $(c, v_{[p]})$, and let c be a structure which is independent of all $v_{[p]}$. c can be empty. To facilitate the representation of $R = \{r_{[p]} \mid p\in P\}$ we define as new objects the variable var v on variability domain $V = \{v_{[p]} \mid p\in P\}$ with respect to R, written var v

: $(V; R)$, variable var r = var κ $(c$, var $v)$ on R, and var $\kappa : K = \{\kappa_{[p]} \mid p \in P\}$, the domain of admitted concatenations to yield elements of R. We make the variables "controllable" by associating to var x, $x \in \{r,\kappa,v\}$, a function val with val: $(p$, var $x) \mapsto x_{[p]}$. The val-functions are named "control-" or "assignment" functions. P or indirectly any set Q with a given function $\alpha: Q \to P = \alpha(Q)$, is a set of control- / assignment parameters. For assignment according parameter p we write also val(var x): $p \mapsto x_{[p]}$, or var x :=(p) $x_{[p]}$. The variable domains R, K, V are sets of "states" or "instances" of var r, var κ, var v, respectively. p itself can be the result of an assignment to a variable var $p : P$. If a "reset" function : $x_p \mapsto$ var x is known, i.e. if a variable, its domain of definition and an assignment function val are known of which $x_{[p]}$ is an instance, application of val: $(q$, var $x) \mapsto (x_q)$ results in a substitution (re-assignment) x_q for x_p.

In general, let there be given a set $\tilde{X} \subseteq X =_{\text{def}} \{x_{[p]} \mid p \in P\}$. $\tilde{X}$ determines a maximal set $\tilde{P} \subseteq P$ such that $\tilde{X} = \{x_{[p]} \mid p \in \tilde{P}\}$. We write $(\tilde{P}$, var $x) = val^{-1}(\tilde{X})$ with respect to the function val: $P \times \{\text{var } x\} \to X$. val^{-1} is a homomorphism of $pow\, X$ onto pow $(P \times \{\text{var } x\})$. pow means power set.

The variability domain of a variable can be structured by functions or general relations and is also named "type" of the variable. Variables and control parameters can be composite. We consider var v = (var $v_l)_{l \in L}$, var $v_l : V_l = \{v_{l[p]} \mid p \in P(l)\}$, var $v : V = \{(v_{l[p(l)]})_{l \in L} \mid p = (p(l))_{l \in L} \in P\}$ with $V \subseteq \prod_{l \in L} V_l$, $P \subseteq \prod_{l \in L} P(l)$. Let be $L = L' \cup L''$ with $L' \neq \emptyset$, $L'' \neq \emptyset$, $L' \cap L'' = \emptyset$. Let var v be partially assigned with control parameter p' $= (p(l))_{l \in L'} \in P' =_{\text{def}} pr(L'; P)$. Then var $v^* =_{\text{def}} (v_{l[p(l)]})_{l \in L'}$ κ (var $v_l)_{l \in L''}$. The control parameter set for (var $v_l)_{l \in L''}$ is $P'' =_{\text{def}} pr(L''; ext(p'; P)$, i.e. the projection onto L'' of the extension of the parameter family p' into P. Notice, in general $P'' \subset \prod_{l \in L''} P(l)$.

Visualization is shown in Figure 1. $L = \{1,2\}$, $p(1)$ given, var $p(2) : P''$.

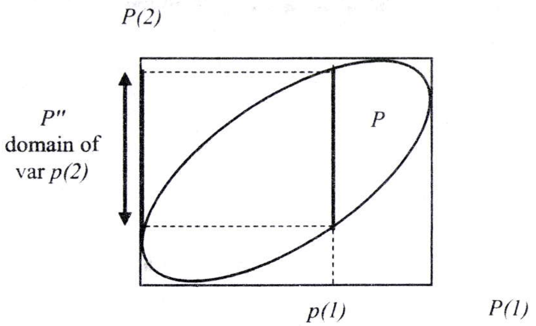

Figure 1

The terminology is: var v^* is a partial variable, (the assignment to, the knowledge about) var v^* is incomplete, *val*(var v^*) is indeterminate, however its domain (range) is determined, the assumption, *val*(var v^*) = $\tilde{v}$ for expected $\tilde{v} \in V^*$ is uncertain.

The domain of a variable can contain variables, which are then of lower variability level with respect to this variable. In this way hierarchies of variables and their controls can be defined.

For a function variable var y = var f(var x), or written var f: var $x \rightarrow$ var y, we assume var $f:\{f_p : X_p \rightarrow Y_p = f_p(X_p) \mid p \in P\}$. We have var $x_p : X_p = \{x_{pq} \mid q \in Q(p)\}$, var $y_p : Y_p$. The assignments are: var $f :=(p)\, f_p$, which determines X_p and Y_p, after that var $x_p :=(q)$ x_{pq}, then var $y_p := f_p\,(x_{pq})$ with pq or x_{pq} as control parameters. var y depends on var x, var y is of higher variability level than var x.

Example 3: Relational database, questionnaire. Let the scheme for var r be

number	*first name*	*middle initial*	*family name*	*date of birth*	*date of death*
var p	var v_1	var v_2	var v_3	var v_4	var v_5

The fixed structure c is the *text* and the *frame*, var $r_{var\,p} = c\ \kappa\,(\text{var } v_{l,\,var\,p})_{l \in \{1,2,3,4,5\}}$, concatenation κ is insertion into the scheme. The domains of the variables (types) are denoted in the first line. For example, *first name* = set of all English prenames. A variable of type *date* is structured: (var *day*, var *month,* var *year*). A relation of var v_5 with var v_4 is: *year of birth* ≤ present year ≤ *year of death* < *year of birth* + 150, if the person is alive. A partial assignment is for example:

var p := 5, var $v_{5,1}$:= *William*, var $v_{5,2}$:= *J.* , var $v_{5,3}$:= *Miller*, var $v_{5,4}$:= 28.7.1968. Control parameters are omitted. Entry into the scheme as line 5 is the concatenation κ.

number	*first name*	*middle initial*	*family name*	*date of birth*	*date of death*
5	*William*	*J.*	*Miller*	28.7.1968.	var v_5

The knowledge about *William J. Miller* is incomplete, however the range of var v_5 is partly known: Present year 2006 ≤ *year of death* < 2118. If *Miller* is now seriously sick, adaptation to a more realistic bound is: *year of death* < 2048. Surviving 2010 is rather certain, 2030 is not so certain, 2040 is rather uncertain, the uncertainty is graded. In course of time, entries and the scheme can be changed, variables and their domains can be deleted, and others can be added.

4. Time

We model "time" by a set $(T, <)$ of time instances /-points t as elements and with a partial, irreflexive, asymmetric, transitive order relation $<$. We also write $< \subset T \times T$. Any subset $U \subseteq T$ is assumed to have the induced ordering. In particular, a subset $C \subseteq$

T is totally ordered, if for any $t', t'' \in C$ either $t' = t''$, or $t' < t''$, or $t'' < t'$, mutually exclusive. Then C is a "chain" in T. Any chain C is element of T^C. For $t \in T$, $\{(t)\}$ is a chain. T is the union of all chains in $(T, <)$, thus $T \subseteq \bigcup_{C \text{ chain in } T} T^C$, i.e. T is a relation according our definition. Two chains C', C'' are of the same "ordering type" if there exists an isomorphism $\mu: C' \leftrightarrow C''$, this means μ is a 1 to 1 mapping and from $c' < c'^*$ in C' follows $\mu(c') < \mu(c'^*)$, from $c'' < c''^*$ in C'' follows $\mu^{-1}(c'') < \mu^{-1}(c''^*)$. For $U \subset T$ and $V \subset T$, a chain in $U \cup V$ which is not a chain in U and not a chain in V is a "connector" of U and V.

Notations used in the following are: $\bigwedge$ ("for all"), $\bigvee$ ("it exists"), $\neg$ ("not"), $\wedge$ ("and"), $\vee$ ("or"), $=_{\text{def}}$ ("is defined by"), $\prec$ in a formula stands for $<$ or $\leq$, mutually exclusive in this formula.

By $<$ on T (partial) order relations on *pow* T can be defined. For subsets U, V, W of T examples are:

(1) $U \prec_{\wedge(\vee)} V =_{\text{def}} \bigwedge u \in U(\bigvee v \in V(u \prec v))$, from $U \prec_{\wedge(\vee)} V$, $V \prec_{\wedge(\vee)} W$ follows $U \prec_{\wedge(\vee)} W$.

(2) $U \prec_{(\vee)\wedge} V =_{\text{def}} \bigwedge v \in V(\bigvee u \in U(u \prec v))$, from $U \prec_{(\vee)\wedge} V$, $V \prec_{(\vee)\wedge} W$ follows $U \prec_{(\vee)\wedge} W$.

(3) $U \prec_{\wedge(\vee)\wedge} V =_{\text{def}} (U \prec_{\wedge(\vee)} V$ and $U \prec_{(\vee)\wedge} V)$, also denoted by $U \prec V$.
Notice: From $U \leq V$ and $V \leq U$ need not follow $U = V$. Example: U = the interval $[0,1]$ of real numbers, V = the interval $[0,1]$ of rational numbers.

A particular case is: $U <_{\wedge\wedge} V =_{\text{def}} \bigwedge(u,v) \in U \times V\, (u < v)$.

(4) $(U \lhd V) =_{\text{def}} (\neg \bigvee (u,v) \in U \times V\, (v \leq u))$. In general, $\lhd$ is not associative. However, from $U \lhd W$, $V \subset W$ and $V \lhd W \backslash V$ follows $U \cup V \lhd W \backslash V$. If $U <_{\wedge\wedge} V$ then $U \lhd V$.

For $U = \{u\}$, $V = \{v\}$, $W = \{w\}$, these orderings reduce to $\prec$. By order relations on *pow* T order relations on *pow(pow* T*)* can be defined, and so on.

If T is finite then to each subset U of T exists a set U_{min} of minimal elements, i.e. $U_{min} \lhd U \backslash U_{min}$. We consider a recursive procedure (A) which is basic in the theory of algorithms and of evolutionary systems:

We use $(\mathbf{N}, <)$, the strictly ordered natural numbers, as "algorithmic" or "evolution" time, a time $(T, <)$ and as initial data $n = 1$, $T^{(1)} =_{\text{def}} T$, $A_1 =_{\text{def}} \varnothing$, and do recursively

(A1): select a non empty set $\Delta_n A \subseteq T^{(n)} \subseteq T$ of minimal elements of $T^{(n)}$,

(A2): concatenate $(A_n, \Delta_n A)$ to $A_{n+1} =_{\text{def}} A_n \cup \Delta_n A$ with the ordering induced by $<$,

(A3): concatenate $(T^{(n)}, \Delta_n A)$ to $T^{(n+1)} =_{\text{def}} T^{(n)} \setminus \Delta_n A$ with the ordering induced by $<$,

(A4): if $T^{(n+1)} \neq \varnothing$ then replace n by $n + 1$, go to (A1), else denote n by n^* and terminate (A).

The selection of $\Delta_n A$ may be controlled and may be subject to conditions. The procedure always terminates by exhausting the finite set T. We have $\bigcup_{i \leq n} \Delta_i A \lhd \bigcap_{i \leq n+1} T^i$, $\bigcup_{i \leq n} \Delta_i A \cap \bigcap_{i \leq n+1} T^i = \varnothing$. If $c = (t_i)_{i=1,2,\ldots m}$ is a maximal chain in T, then $m \leq n^* \leq card\, T$ (cardinality of T). The set $N_{global} = \{1,2,\ldots n^*\}$ or any isomorphic set can serve as global time for the process $(\Delta_n A)_{n\leq n^*}$. $\Delta_n A \mapsto n$, $n \in N_{global}$, is a homomorphism, the elements of $\Delta_n A$ are independent with respect to N_{global}. A "shortest" global time is c. In topological terms, $\{T^{(n)} \mid n\in N_{global}\}$ forms a filter base with limit $\varnothing$, $\{A_n \mid n\in N_{global}\}$ forms an ideal base with limit T. Generalizations of the procedure (A) are treated in Section 6.:

Example 4: $T = \{t_i \mid i = 1,2,3,\ldots 9\}$. Let be: $G = \{(t_1 < t_4), (t_2 < t_4), (t_4 < t_6), (t_4 < t_5) (t_7 < t_8), (t_7 < t_9)\}$. By associability, $H = \{(t_2 < t_5), (t_2 < t_6), (t_1 < t_5), (t_1 < t_6)\}$. The set of chains is $C = \{(t_3)\}\cup G\cup H\cup\{(t < t_4 < t_5), (t_2 < t_4 < t_6), (t_1 < t_4 < t_5) (t_1 < t_4 < t_6)\} \subset T\cup(T\times T)\cup(T\times T\times T)$. One possible decomposition is: $\Delta_1 A = \{t_2, t_3\}$, $\Delta_2 A = \{t_1, t_7\}$, $\Delta_3 A = \{t_4, t_9\}$, $\Delta_4 A = \{t_5, t_8\}$, $\Delta_5 A = \{t_6\}$, a maximal chain is $\{(t_1 < t_4 < t_6)\}$ with length 3, *card* $T = 9$. For example, $\Delta_1 A \cup \Delta_2 A \Rightarrow (T_3 = \{t_4, t_5, t_6, t_8, t_9\})$, the connector set of $\Delta_1 A$ with $\Delta_2 A \cup \Delta_3 A \cup \Delta_4 A \cup \Delta_5 A$ is $\{(t_2, t_4), (t_2, t_5), (t_2, t_6), (t_2, t_4, t_5), (t_2, t_4, t_6)\}$. See Figure 2.

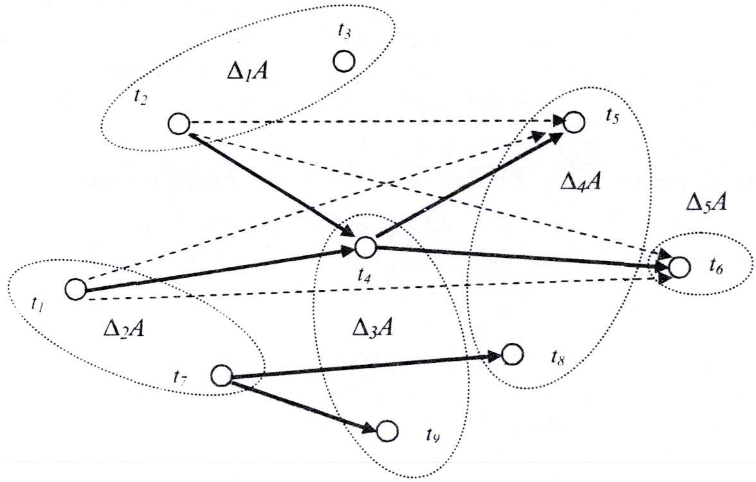

Figure 2

5. Processes and Process Homomorphisms

Let there be given: A time $(T, <)$, a subset $U \subseteq T$, a non-empty set S, an indexing ι: $U \to S$ with $t \mapsto s_{[t]}$, $s_{[t]} \in S$. Frequently, $S \subseteq pow\ Z$, Z a non-empty set. Then $s_U = (s_t)_{t\in U}$ is a "process on U" with "state" $s_{[t]}$ at t. The ordering $<$ on T induces an ordering on $s_U : t < t' \Leftrightarrow$ (if and only if) $s_t < s'_{t'}$. All concepts defined by the ordering of T are transferable to processes. Let $s_V = (s'_t)_{t\in V}$ be another process. If $U\cap V \neq \varnothing$ then s_U and s_V are "concurrent" on $U\cap V$. If $U\cap V = \varnothing$ then s_U and s_V are named independent ("parallel"). If a time $(T, <)$ and a set S are given, the set of all processes on S over T is $\boldsymbol{P} = \bigcup_{U \subseteq T} S^U$. Each subset of $\boldsymbol{P}$ is also named process. Two processes s_U, s'_V are time isomorphic if a $<$-isomorphism $\varphi : U \leftrightarrow V$ exists for them. If in addition for all $u \in U$ holds $s_{[u]} = s'_{[\varphi(u)]}$, then the processes are time invariant with respect to φ. If S is ordered itself, $(S, <_S)$, and if the indexing ι is a $\leq_S$ – homomorphism, i.e. for $t' < t''$ holds $\iota(t') \leq_S \iota(t'')$, then $\iota(T)$ is a time.

We consider a function $f\colon X = (x_t)_{t \in U} \xrightarrow{\text{onto}} Y = (y_t)_{t \in V}, x_t \mapsto y'_{t'}$. In general, time $(U, <_U)$ is independent of time $(V, <_V)$. Let $\mathcal{X}$ be the domain of the $x_{[t]}$, and $\mathcal{Y}$ be the domain of the $y_{[t]}$. We say X, Y are "input-" and "output processes" respectively to the "processor" f. Functional dependency is denoted by $\rightarrow, \mapsto$. We further assume, f is a $\leq$–homomorphism, i.e. for $x_t < x'_{t'}$ holds $f(x_t) \leq f(x'_{t'})$, subscripts of $<$ omitted. The set extension F of f is a homomorphism with respect to set inclusion $\subseteq$. For (y_{t^*}) we consider the reciprocal image $\overset{-1}{f}(y_{t^*}) = (x_t)_{t \in U^*}$. In case F is independent of U, F maps U^* onto $\{t^*\}$ and $(x_{[t]})_{t \in U^*}$ onto $(y^*_{t^*})$. Let U and V be sub-times of a time $(T, <)$ with the induced ordering. To model physical reality, where a cause is not later or simultaneous with its effect and all operations take time, we assume, the "time delay condition" $\bigwedge x \in X(\neg(f(x) \leq x)$ holds, in our notation: $x \lhd f(x)$. This excludes mathematical identity for f but includes operational time delay $x < f(x)$. If $\bigwedge x \in X(x < f(x))$, then $X < Y$ (according our definition of $<$), because f is surjective. Under these assumptions, the "processing time" of f is $U \cup V \subseteq T$. We denote $X_{[U]} =_{\text{def}} \{x_{[t]} \mid t \in U\}$, $Y_{[V]} =_{\text{def}} \{y_{[t]} \mid t \in V\}$. Output data may be used as input data at later time. For example, if the objects are sets, if $y'_{[t']} \subseteq y_{[t']}$ and $x''_{[t'']} \subseteq x_{[t'']}$ with $t' < t''$, then $x''_{[t'']} = y'_{[t']}$ is possible. By causality, input X^* exists not exclusively originating from the output, by intention, output Y^* exists not exclusively used for input. We say, the processes X^*, Y^* are "external", i.e. their states are given, observable, measurable, available, accessible for other applications.

Example 5: For real numbers let be $Y = \{y_{t+2} = x_{t+2} =_{\text{def}} (x_{[t+1]} \times x_{[t]})_{t+2} \mid t \in \mathbf{N}\}$, $X = Y \cup X^*$, $X^* = \{x_1, x_2\}$ given, $Y^* = \varnothing$. f is multiplication $\times$. This is physically an infinite operation without an external result, only changing "internal states". $x_{[t+1]}$ and $x_{[t]}$ have to be memorized. However, if the output process is external then $Y = Y^*$. For $x_{[1]} = x_{[2]} = 1$ the process is stationary. This example is covered by classical automata theory.

Example 6: $(T, <) = (\mathbf{R}, <)$, let be $X^* = (t^2_t)_{t \in [0, t^*]}$, $0 < t^* \in T$, $Y^* = ((\int_0^t \tau^2 \, d\tau)_{\varphi(t)})_{t \in [0, t^*]}$, φ is a $<$ - isomorphism with $t < \varphi(t)$, expressing the operational time delay. This example is not covered by classical automata theory.

An object $x_t \in X$ may be uncertain, but known to be an element of $\hat{X} \subset X$, i.e. var x_t : $\hat{X}$. Then the functional result y_s is undetermined but an element of $F(\hat{X}) \subset Y$, i.e. var $y_s = F(\text{var } x_t)$ varies on $F(\hat{X})$. In addition, F can be uncertain.

Example 7: Let be $T \subseteq (\mathbf{R}_+, <)\times(\mathbf{R}_+, <)$, $\mathbf{R}_+$ denotes the set of non-negative real numbers. A partial ordering $<<$ on T is given by $((t', t'') << (s', s'')) \Leftrightarrow ((t' < s') \wedge (t'' \leq s''))$ excl. or $((t' \leq s') \wedge (t'' < s''))$. For example, to $X = (x_t)_{t\in T}$, $x_{[t]} \in \mathbf{R}_+$, let correspond $Y = ((s(t) = t'_{[t]} \times t''_{[t]}, \varphi(x_{[t]}))_t)_{t\in T}$, $\varphi(x_{[t]}) \in \mathbf{R}_+$. From $t << t^*$ follows $s(t) < s(t^*)$. If var t: [a, b]×[c, d], a ≤ b, c ≤ d, then var s: [a×c, b×d] (this is an example for interval arithmetic on **R**, the time delay is neglected).

Under certain assumptions and for a universal time, processors can be concatenated to a composite processor (or "network"): For illustration, let be F: $x_u \to y_v$; $x'_{u'}$, $x''_{u''}$, $y'''_{v'''}$ a partition of y_v; $\tilde{x}'$, $\tilde{x}''$ admissible given input parts. For given concatenations κ', κ'' let be $x'_{w'} =_{\text{def}} x'_{u'}\ \kappa'\ \tilde{x}'$, $x''_{w''} =_{\text{def}} x''_{u''}\ \kappa''\ \tilde{x}''$. G: $x'_{w'} \to y'_{z'}$, H: $x''_{w''} \to y''_{z''}$. Figure 3 is a visualization.

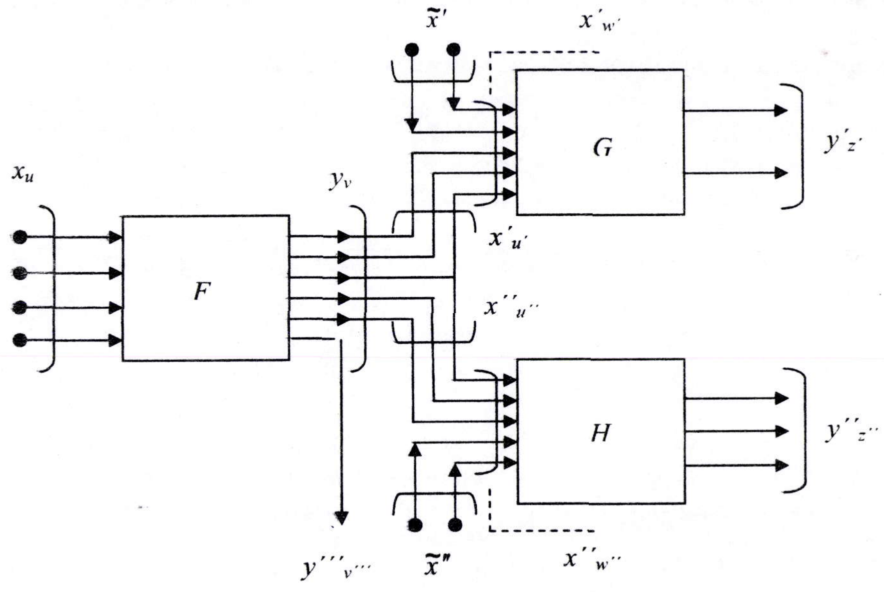

Figure 3

For the composed processor the input processes are x_u, $\tilde{x}'$, $\tilde{x}''$, the output processes are $y'''_{v'''}, y'_{z'}, y''_{z''}$. The relation "part of the output of a processor F is a part of the input of a processor G" is a connector $C(F,G)$. If not empty, it generates an irreflexive, transitive ordering $F \prec G$ ("G depends on F") on the set of processors composed. $\prec$ is compatible with $\lhd$ for the processes. In reality, $C(F,G)$ represents itself a processor ("channel"), copying output of F as time lagged input to G. The assumptions we made are: x_u is in the domain of F; $x'_{w'}$ is in the domain of G; $x''_{w''}$ is in the domain of H ("domain condition"). For example, if this is violated by input arriving too early or because input parts have to be sequenced for processing, a connector needs synchronizing devices like memories or transmission delays.

Starting with a finite set $\{F_i \mid i\in I\}$ of time independent processors F_i on level $n = 0$ ("atoms"), compositions of composite processors on level n generate processors on level $n+1$ (hierarchy of compositions). A physical object which represented a processor, may be reused after its processing time w at "later" time w', $w \cap w' = \varnothing$. Use may change the physical object and the processor.

Example 6: $(T, <) = (\mathbf{N}, <)$, $x_{[n]}, y_{[n]}$ non-negative integers. For $n = 1,2,3,\ldots$ let be $x_{n+1} = (x_{[n]} - 1)_{n+1}$ while $x_{[n]} \geq 1$, $y_{\varphi(n+1)} = (y_{[\varphi(n)]} + n^2 x_{[n]})_{\varphi(n+1)}$, $\varphi(n) = n^2$, with initial values: $x_{[1]} = 5$, $y_{[1]} = 0$. We find $x = (5_1,4_2,3_3,2_4,1_5)$, $y = (0_1,5_4,21_9,48_{16},80_{25},105_{36})$.

Varieties of times, processes, processors, connectors and networks can be described by controlled variables. We revisit Section 2 and apply the previous reasoning on time behavior of objects.

We consider a function variable var y = var f(var x), var $f : \{f_p : X_p \rightarrow Y_p = f_p(X_p) \mid p\in P\}$, var $x_p : X_p = \{x_{pq} \mid q\in Q(p)\}$, var $y_p : Y_p$. The assignments are: var f :=$(p)\, f_p$, which determines X_p and Y_p, after that var x_p :=$(q)\, x_{pq}$, then var y_p : =$(pq)\, y_{pq} = f_p(x_{pq})$ with pq or x_{pq} as control parameters. We assume, $(S, <_S)$, $(T, <_T)$ are given times, $S(pq) \subseteq S$, $T(pq) \subseteq T$ are sub-times with the induced orderings. Now let the objects x_{pq}, y_{pq} be processes, $x_{pq} = (x_{pqs})_{s\in S(pq)}$, $y_{pq} = (y_{pqt})_{t\in T(pq)} = f_p((x_{pqs})_{s\in S(pq)})$. To process x_{pq} the process y_{pq} is assigned but not states x_{pqs} to states y_{pqt}. However, if a point wise function $h_p : (x_{pqs})_{s\in S(pq)} \rightarrow (y_{pqt})_{t\in T(pq)}$ is given, i.e. $x_{pqs} \mapsto y_{pqt}$, the set extension H_p of h_p maps processes onto processes. By causality, $(s(pq)\in S(pq)) \lhd (t(pq)\in T(pq))$.

Let us assume: $P = \{p\}$, thus omitting index p in the following, $S =_{\text{def}} \bigcup_{q\in Q} S(q)$, for $q \neq q'$ to have $S(q)\cap S(q') = \varnothing$, $\wedge q\in Q(H: (x_{qs})_{s\in S(q)} \mapsto (y_{qt(q)}))$, thus $S(q) \mapsto \{t(q)\}$, and $T = \{t(q) \mid q\in Q\}$. H is a $\subseteq$ - set homomorphism. In addition, H implies a $\leq$-homomorphism H_{time}: $S \rightarrow T$. Time $(T, <_T)$ is a coarsening of time $(S, <_S)$. The reciprocal image $\overset{-1}{H}((y_t)_{t\in T^*})$ to a process $(y_t)_{t\in T^*}$, $T^* \subseteq T$, is a family of processes, $((x_{qs})_{s\in S(q)}) \mid S(q) = \overset{-1}{H}_{time}(t))_{t\in T^*}$. For given t, by causality, $\{t\} \lhd \overset{-1}{H}_{time}(t)$. An example is a compiler, transforming a series of statements in a higher order programming language into a series of series of instructions in a lower level language. If for all $q,q'\in Q$ a $<$-isomorphism $\sigma(q,q')$: $S(q) \rightarrow S(q')$ exists, i.e. a 1 to 1 mapping preserving $<$, and for all $s \in S(q)$ $x_{[q,s]} = x_{[q',\sigma(s)]}$, then $(x_{\text{var } q,s})_{s\in S(\text{var } q)}$ with var q: Q is a time invariant representation with respect to $\{S(q) \mid q\in Q\}$.

6. Evolutionary Systems

Let there be given a time $(T, <)$. Any subsets U, V of T have the ordering induced by $<$. Concatenation $U \kappa V$ (also written $\kappa(U, V)$) is defined by $U \cup V$ with ordering induced by $<$. We consider a set $U \in pow\, T$ and make the following assumptions:

(A1): $\overline{U} \subseteq T \setminus U$ and $\overline{U}$ is a maximal set such that $U \lhd \overline{U}$,

(A2): $\Delta U \subseteq (U \cup \overline{U})$ such that $\Delta\tilde{U} =_{def} \Delta U \setminus U \neq \varnothing$,

(A3): $\Delta\tilde{U} \lhd (\overline{U} \setminus \Delta\tilde{U})$.

From these assumptions follows: $U \cap \overline{U} = \varnothing$ (by definition of $\lhd$), thus $U \cap \Delta\tilde{U} = \varnothing$,

(C1): $\Delta U \lhd (\overline{U} \setminus \Delta U)$, $(\Delta U \cup U) \lhd (\overline{U} \setminus \Delta U)$, (proof: by (A2), (A3), and for any X,Y,Z: $(X \lhd Y \wedge Z \subset Y) \Rightarrow X \lhd Y \setminus Z$, $(X \lhd Z \wedge Y \lhd Z) \Rightarrow (X \cup Y \lhd Z)$),

(C2): $\bigwedge(u,v) \in (U \times \Delta\tilde{U})(\neg \bigvee t \in T \setminus \Delta\tilde{U}\,(u<t<v)$ (proof: $u<t \wedge t \notin \Delta\tilde{U} \Rightarrow t \in \overline{U} \setminus \Delta\tilde{U}$ (A1), $t<v \Rightarrow \neg(\Delta\tilde{U} \lhd \overline{U} \setminus \Delta\tilde{U})$), this contradicts (A3)).

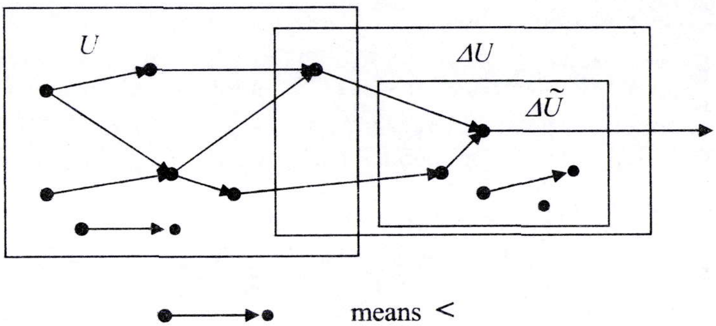

Figure 4

Figure 4 shows an illustration. Applied to chains $C(U \cup \Delta\tilde{U})$ on $U \cup \Delta\tilde{U}$ with $C(U) =_{def} C(U \cup \Delta\tilde{U}) \cap U$, $C(\Delta\tilde{U}) =_{def} C(U \cup \Delta\tilde{U}) \cap \Delta\tilde{U}$, we have the following cases, visualized in Figure 5:

(a) if max $C(U)$ and min $C(\Delta\tilde{U})$ exist: max $C(U) <$ min $C(\Delta\tilde{U})$, (jump);

(b) if max $C(U)$ and not min $C(\Delta\tilde{U})$ exists: $\{\max C(U)\} <_{\wedge\wedge} C(\Delta\tilde{U})$, (Dedekind cut);

(c) if not max $C(U)$ and min $C(\Delta\tilde{U})$ exists: $C(U) <_{\wedge\wedge} \{\min C(\Delta\tilde{U})\}$, (Dedekind cut);

(d) if not max $C(U)$ and not min $C(\Delta\tilde{U})$ exist: $C(U) <_{\wedge\wedge} C(\Delta\tilde{U})$, (gap).

In all cases there exists no time point t between $C(U)$ and $C(\Delta\tilde{U})$. If this holds for all chains in $U \cup \Delta\tilde{U}$ we say U and $\Delta\tilde{U}$ are "adherent" or neighboring, $U \cup \Delta\tilde{U}$ is a successor of U. A visualization is shown in Figure 5. If all cuts of all chains in $(T, <)$ are Dedekind cuts, we say $(T, <)$ is continuous.

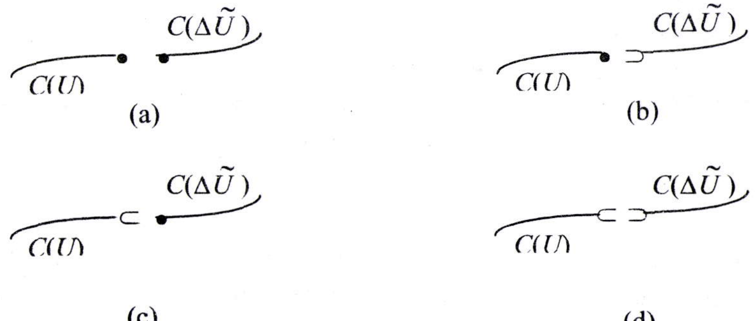

Figure 5

Starting with an $U_{initial}$ and assuming the procedure can be recursively repeated, the family of indexed successors $((U \cup \Delta\tilde{U})_m)_{m\in M}$ or increments $(\Delta\tilde{U}_m)_{m\in M}$ creates a well-ordered ("algorithmic", "evolution-") time $(M, \prec)$.

We are going to define processes on the time structures studied. Let there be given a non-empty set X of objects and a time $(T, <)$, a time process variable $\text{var}\ w : \mathcal{W} \subseteq \text{pow}\ X$ and an object process variable $\text{var}(x_W) =_{\text{def}} \text{var}\ x_{\text{var}W} =_{\text{def}} (\text{var}\ x_t)_{t\in\text{var}W}$ with domain $\mathfrak{X}(\text{var}\ w) \subseteq X^{\text{var}\ w}$. For simplicity sake, we restricted the general case $\prod_{t\in\text{var}w} X_t$ with distinct sets $X_{[t]}, X_{[t']}$ for $t\neq t'$, to $X^{\text{var}\ w}$. Admitted are only concatenations $\text{var}(x_W)\ \kappa\ \text{var}(x'_{W''}) = \text{var}(x''_{W\cup W''})$, $W\cap W' = \varnothing$, $\text{var}\ x''_t = \text{var}\ x_t$ on $\text{var}\ W$, $\text{var}\ x''_t = \text{var}\ x'_{t'}$ on $\text{var}\ W'$, i.e. without overlap. We apply this to the evolution equation

$\text{var}^{(3)}(x_{U\ \kappa\ \Delta U}) = \text{var}^{(1)}(x_U)\ \kappa\ \text{var}^{(2)}(\Delta x_{\Delta U})$, upper indices indicate the variable level.

We assume: $\text{var}\ U$, $\text{var}\ \Delta U$ are subject to the conditions (A1, A2, A3), $\text{var}\ \Delta U = \text{var}\ \Delta\tilde{U}$. By our restrictions, κ is independent of $(\text{var}(x_U), \text{var}(\Delta x_{\Delta U}))$. Omitting possible control parameters p for $\text{var}(x_U)$, $q(p)$ for $\text{var}(\Delta x_{\Delta U})$, let be assigned $\text{var}^{(1)}(x_U) := x_U \in \mathfrak{X}(\text{var}(x_U))$. This restricts the domain $\Delta\mathfrak{X}(\text{var}(\Delta x_{\Delta U}), \text{var}(x_U))$ of $\text{var}(\Delta x_{\Delta U})$ to $\Delta\mathfrak{X}(\text{var}(\Delta x_{\Delta U}), x_U)$. The result is

$\text{var}^{(2)}(x_{U\ \kappa\ \Delta U}) = (x_U)\ \kappa\ \text{var}^{(1)}(\Delta x_{\Delta U})$, $\text{var}^{(1)}(\Delta x_{\Delta U})$ is a control variable for $\text{var}^{(2)}(x_{U\ \kappa\ \Delta U})$. If $\Delta x_{\Delta U}$ is assigned to $\text{var}^{(1)}(\Delta x_{\Delta U})$, we have $\text{var}^{(1)}(x_{U\ \kappa\ \Delta U}) := (x_U)\ \kappa\ (\Delta x_{\Delta U}) = x_{U\ \kappa\ \Delta U}$. Let there be given an initial object process $(x'_U)_0$, and let continuations be possible. We obtain by recursion for an evolution time $(0 \prec \text{succ}\ 0 \prec \text{succ}(\text{succ}\ 0) \prec \ldots \prec \text{m})$

$(x_U)_{\text{succ m}} = (\ldots(((x'_U)_0\ \kappa\ (\Delta x_{\Delta U})_0)_{\text{succ}\ 0})\ \kappa\ (\Delta x_{\Delta U})_{\text{succ}\ 0})_{\text{succ}(\text{succ}\ 0)} \ldots) \ldots \kappa\ (\Delta x_{\Delta U})_{\text{m}} = (x'_U)_0\ \kappa\ \mathop{K}_{\mu=0}^{\text{m}} (\Delta x_{\Delta U})_\mu$ because κ is associative. According the assumptions made, κ is independent of $((\Delta x_{\Delta U})_\mu, (\Delta x_{\Delta U})_{\text{succ}\ \mu})$. If a further continuation starts with an initial process $(x'_U)_{\text{succ m}}$ to be concatenated with $(x_U)_{\text{succ m}}$ and so forth, we have a process of ("external") initial objects and the ("internal") object process controlled by $\text{var}(\Delta x_{\Delta U})$.

If a commutative group operation $\sigma : X \times X \to X$ is defined (see Section 2), two overlapping evolutionary object processes can be concatenated to one object process. If $T \times X$ is a metric space, for continuous and differentiable functions $T \to X$, limits in the evolution equation can be defined.

Example 8: We consider the partial ordered space $T \subseteq (\mathbf{R}_+, <) \times (\mathbf{R}_+, <)$ of *Example 7*, functions $T \to \mathbf{R}$ and a chain C with time points (r, r'). In general, r, r' are independent. If $r' = \gamma(r)$, we can describe C by $(r, \gamma(r))_{r \in U}$. For the ordering $<<$, γ has to be monotone. We choose $<$ induced by $(\mathbf{R}_+, <)$, set $t = r$ and consider the chain $(t, t))_{t \in [0,1)} \cup (t, 0)_{t \in [1,2)}$. With chosen initial process $(x'_0 = 0,\ x'_1 = -1)$ and given derivatives $\dot{x} = 2t$ for $[0 \le t < 1)$, $\dot{x} = -e^{-t+1}$ for $[1 \le t < 2)$, we find $x(t) = 0 + \int_0^t 2\tau \, d\tau = t^2$ for $[0 \le t < 1)$, $x(t) = -1 - \int_1^t e^{-\tau+1} d\tau = -1 + (e^{-t+1} - 1)$ for $[1 \le t < 2)$. For $t \to 1$: $x(t) \to 1$, $\dot{x}(t) \to 2$; $x(1) = -1$, $\dot{x}(1) = -1$. Neither $x(t)$ nor $\dot{x}(t)$ is continuous at $t = 1$. $[0 \le t < 2)$ is a continuous set. Let us assume, t is undetermined and varies in intervals: $[t - 0.1, t + 0.1)$ for $0.1 \le t < 1.9$, $[0, t + 0.1)$ for $0 \le t < 0.1$, $[t - 0.1, 2)$ for $1.9 \le t < 2$. Then for $t \in [0, 0.1)$, var $x(t)$: $[0, (t + 0.1)^2]$; for $t \in [0.1, 0.9)$, var $x(t)$: $[(t - 0.1)^2, (t + 0.1)^2]$; for $t \in [0.9, 1.1)$, var $x(t)$: $[-2 + e^{-t+0.9}, 1)$; for $t \in [1.1, 1.9)$, var $x(t)$: $[-2 + e^{-t+0.9}, -2 + e^{-t+1.1}]$; for $t \in [1.9, 2)$, var $x(t)$: $(-2 + e^{-1}, -2 + e^{-t+1.1}]$.

Up to now, we disregarded general concatenations $\kappa((x'_t)_{t \in U}, (x''_t)_{t \in V}) = ((x_t)_{t \in W}$, also causality and the operational time delay. We use the notations x_U, $x'_{U'}$, y_V for past object process, past initial process, past control process, respectively. $\mathrm{var}(\Delta x_{\Delta U}) = \mathrm{var}\ \Delta x_{\mathrm{var}\Delta U}$, $\mathrm{var}(\Delta y_{\Delta U}) = \mathrm{var}\ \Delta y_{\mathrm{var}\ \Delta V}$, $\mathrm{var}\ (\Delta x'_{\Delta U'}) = \mathrm{var}\ \Delta x'_{\mathrm{var}\Delta U'}$ are process continuation variables, subject to assumptions (A1,A2,A3). Omitting the prefix var, a general formulation of the incremental equations is in case

(X): The incremental control process $(\Delta y_{\Delta U})$ and the incremental initial process $(\Delta x'_{\Delta U'})$ are given, the controlled incremental object process $\Delta x_{\Delta U}$ is to determine. We consider $(\Delta x_{\Delta U}) = f(x_U, y_V, (\Delta y_{\Delta V}))$, with assumptions: $V \lhd \Delta V$, $U \lhd \Delta U$, $\varnothing \neq \Delta V(\min) \subseteq \Delta V$, $\Delta V(\min)$ the set of $\lhd$-minimal elements in ΔV, $(V, \Delta V(\min))$ are adherent. Let f be a time continuous $\lhd$- homomorphism, then $(\Delta x_{\Delta U(\min)}) = f(x_U, y_V, (\Delta y_{\Delta V(\min)}))$, $(U, \Delta U(\min))$ are adherent. By causality: $\wedge \tilde{v} \in \Delta V(\Delta y_{\tilde{v}} \lhd f(x_U, y_V, (\Delta y_{\tilde{v}})))$. Renaming $(\Delta x''_{\Delta U''}) =_{\mathrm{def}} (\Delta x_{\Delta U})$, we set $(\Delta x_{\Delta U}) = (\Delta x'_{\Delta U'})\ \kappa_x\ (\Delta x''_{\Delta U''})$ if concatenation κ_x is defined. Processing time is $\Delta V \cup \Delta U' \cup \Delta U''$. In case of anticipatory systems we would have processes $\bar{x}_{\bar{U}}$, $\bar{y}_{\bar{V}}$ with $U \lhd \bar{U}$, $V \lhd \bar{V}$ in the arguments of f which are not adherent x_U, y_V.

(Y): The incremental object process $(\Delta x_{\Delta U})$ is given, an incremental control process Δy is to determine which would be a control of it when later to reproduce. In general, there can exist many control processes leading to the same object process. We consider one of them, $(\Delta y_{\Delta V}) = g(x_U, y_V, (\Delta x_{\Delta U}))$. Then assumptions and reasoning analogue to case (X) can be applied.

In a physical representation of f and g, the memorized history $x_U, x'_{U'}, y_V$ (or parts of them) represent the "states" of the "processors" f and g, influencing their functional behavior. By definition, succ $x_U = x_U \ \kappa_x \ \Delta x_{\Delta U}$, succ $y_V = y_V \ \kappa_y \ \Delta y_{\Delta V}$, κ_x and κ_y concatenations of x-, y-families. If f, g are strict homomorphisms with respect to the concatenations κ_x, κ_y, then $(\Delta x_{\Delta U}) \ \kappa_x$ succ $(\Delta x_{\Delta U}) = (\Delta x'_{\Delta U'}) \ \kappa_x$ succ $(\Delta x'_{\Delta U'}) \ \kappa_x \ f(x_U \ \kappa_x \ (\Delta x_{\Delta U}), y_V \ \kappa_y \ (\Delta y_{\Delta V}), (\Delta y_{\Delta V}) \ \kappa_y$ succ $(\Delta y_{\Delta V}))$. Analogously, $(\Delta y_{\Delta V}) \ \kappa_y$ succ $(\Delta y_{\Delta V}) = g(x_U \ \kappa_x \ (\Delta x_{\Delta U}), y_V \ \kappa_y \ (\Delta y_{\Delta V}), (\Delta x_{\Delta U}) \ \kappa_x$ succ $(\Delta x_{\Delta U}))$. By recursion, integral representations are obtainable.

The operational time delay is assumed to be describable by a time isomorphism φ: $\Delta U \rightarrow \varphi(\Delta U) \subset T$, $\Delta U < \varphi(\Delta U)$, $\varphi(\Delta U) \cup \varphi(\text{succ } \Delta U) = \varphi(\Delta U \cup \text{succ } \Delta U)$, and ψ: $\Delta V \rightarrow \psi(\Delta V) \subset T$, $\Delta V < \psi(\Delta V)$, $\psi(\Delta V) \cup \psi(\text{succ } \Delta V) = \psi(\Delta V \cup \text{succ } \Delta V)$.

7. Conclusion

The article is based on classical set theory and analysis as presented for example in [2,3,4] or in any other equivalent text books. The concept of families, general set products and relations, concatenations, order relations among sets of subsets of a partial ordered set, variables and their control, procedure (A), composition of processors, are already considered in [1]. Seemingly, processes on general partial ordered times, the importance of the adherence condition, the assumptions A1, A2, A3 which imply the well-ordering theorem, the use of relation $\triangleleft$, are not treated in the literature (compare for example [5,6,7]). [7] is on our line, has more details and less general concepts. For modeling of physical systems, causality, operational time delay and uncertainty are regarded.

References

[1] Albrecht R. F.: "On mathematical systems theory". In: R. Albrecht (ed.): Systems: Theory and Practice, Springer, Vienna-New York, pp. 33-86, (1998)

[2] Alexandroff P.S.: Einführung in die Mengenlehre und die Theorie der reellen Funktionen, VEB Deutscher Verlag der Wissenschaften, Berlin 1956

[3] Bourbaki N.: Théorie des Ensembles, Hermann et Compagnie, Paris, 1954

[4] Gähler W.: Grundstrukturen der Analysis, Vol. I and II, Birkhäuser Verlag, Basel-Stuttgart, 1977, 1978

[5] Pichler F.: Mathematische Systemtheorie, Walter de Gruyter, Berlin-New York, 1975

[6] Mesarovic M.D., Y.Takahara: Abstract Systemstheory, Springer Verlag, Berlin, Heidelberg, New York, London, Paris, Tokyo, 1989

[7] Wunsch G.:. Systemtheorie-Prinzipien und Systemklassen, in Handbuch der Systemtheorie, Oldenburg Verlag, München-Wien, 1986, pp.1-35