FUNCTIONS AS SET BEHAVIOR

(Essential Concepts: Conceptual & Formal)

D L Childs
<iis@umich.edu>

1. Introduction

Though the definition of *function* under classical set theory, CST, adequately services most mathematical pursuits, it is not suitable for formally modeling general systems behavior. However, using the modeling extensions provided by **extended set theory**[1], XST, it is a rather easy exercise to redefine functions, while preserving CST dependent definitions, that provide the formal muscle required to model general systems behavior[2]. This paper is for those that like exercise.

The basic notion of the term *function* suggests a process that accepts an **input** and produces an **output**. This notion can be refined by stipulating a *domain* of acceptable inputs coupled with a *codomain* of acceptable outputs. The notion can be further refined to accommodate properties like: *on*, *onto*, *one-to-one*, *many-to-one*, and *one-to-many*. The exclusion of the last property qualifies a process to be a function.

A practical consequence of defining processes in terms of **extended set processing**[3] allows building intrinsically reliable systems that can dynamically manage data access performance.

2. Processes

As in CST where functions are a restricted relation as defined by the CST image operation, the XST definition of function will be a restricted process as defined by the XST image operation.

For both conceptual convenience and mathematical compatibility, the term **process** will be defined as an abstract modeling symbol that can be algebraically manipulated and combined with other like symbols to reflect the behavior of one set as influenced by another. Not having any mathematical substance, processes do not exist in any formal set theory and thus can not be contained in sets. However, since the notation for a process will be defined in terms of legitimate sets, these sets can be represented in such a way as to denote the proper process.

Definition 2.1. Process: Two sets $\mathbf{f}$ and σ define a process $\mathbf{f}_{(\sigma)}$ iff

$$(\exists x)\big(\ \mathbf{f}_{(\sigma)}(x) \neq \varnothing\big)\ \&\ (\forall \mathbf{g})(\mathbf{g} \subseteq\!\!\!\cdot\ \mathbf{f})(\exists y)\big(\ \mathbf{g}_{(\sigma)}(y) \neq \varnothing\ \big).$$

[Where '$\subseteq\!\!\!\cdot$' means non-empty subset.]

Definition 2.2. Process Equality:

$$\mathbf{f}_{(\sigma)} = \mathbf{g}_{(\omega)} \iff (\forall x)(\ \mathbf{f}_{(\sigma)}(x) = \mathbf{g}_{(\omega)}(x)\ ).$$

Since all uses of process are only a prediction, the actual set behavior can not be realized until a process is given a set-theoretic interpretation. This can be achieved with a definition of *Application* that equates the process $\mathbf{f}_{(\sigma)}$ with an extended definition of *Image*. Similarly as to how CST functions are defined by the *Image* operation.

3. Application

The familiar notation for a function, $\mathbf{f}(\mathbf{a}) = \mathbf{b}$, actually has a set-theoretic definition in terms of *Image*. Image is defined in terms of a set of ordered pairs, $\mathbf{R}$, and a set $\mathbf{A}$ containing 'first elements' from ordered pairs in $\mathbf{R}$.

Definition 3.1. CST Image: $\mathbf{R}\big[\mathbf{A}\big] = \{y : (\exists x)(x \in \mathbf{A}\ \&\ \langle x, y\rangle \in \mathbf{R}\ \}.$

Given a set of ordered pairs, $\mathbf{f}$, such that no 'second element' of an ordered pair in $\mathbf{f}$ shares the same 'first element', then a CST function can be defined as follows.

Definition 3.2. CST Function: $\mathbf{f}(\mathbf{a}) = \mathbf{b} \iff \mathbf{f}\big[\{\mathbf{a}\}\big] = \{\mathbf{b}\}$.

Though this equivalence may be interesting, it may not appear particularly useful. However, the definition of Image is actually equivalent to the set constructed by two more primitive set operations: the *Restriction* operation followed by the *2-Domain* operation.

Definition 3.3. CST Restriction: $\mathbf{R}\,|\,\mathbf{A} = \Big\{\ \langle \mathbf{x}, \mathbf{y}\rangle \colon \big(\ \langle \mathbf{x}, \mathbf{y}\rangle \in \mathbf{R}\big)\ \&\ \ (\mathbf{x} \in \mathbf{A})\Big\}$.

Definition 3.4. CST 1-Domain: $\text{Đ}_1(\mathbf{R}) = \Big\{\ \mathbf{x} \colon (\exists \mathbf{y})\big(\ \langle \mathbf{x}, \mathbf{y}\rangle \in \mathbf{R}\ \Big\}$.

Definition 3.5. CST 2-Domain: $\text{Đ}_2(\mathbf{R}) = \Big\{\ \mathbf{y} \colon (\exists \mathbf{x})\big(\ \langle \mathbf{x}, \mathbf{y}\rangle \in \mathbf{R}\ \Big\}$.

In the above definitions $\mathbf{R}$ is a relation, a set of ordered pairs. The restriction operation produces a subset of $\mathbf{R}$ dictated by the elements available in $\mathbf{A}$. The 1-Domain operation produces a collection of all 'first elements' from ordered pairs in $\mathbf{R}$. The 2-Domain operation produces a collection of all 'second elements' from ordered pairs in $\mathbf{R}$.

A more constructive definition of CST Image can now be defined in terms of Restriction and Domain operations.

Definition 3.6. CST Image: $\mathbf{R}\big[\mathbf{A}\big] = \text{Đ}_2\big(\mathbf{R}\,|\,\mathbf{A}\big)$.

The CST function can now be defined to include a constructive definition of Image.

Definition 3.7. CST Function: $\mathbf{f}(\mathbf{a}) = \mathbf{b} \iff \mathbf{f}\big[\{\mathbf{a}\}\big] = \text{Đ}_2\big(\mathbf{f}\,|\,\{\mathbf{a}\}\big) = \{\mathbf{b}\}$.

The CST model for defining CST functions, using Relations, Image, Restriction, and Domain, works just as well for defining XST functions.

The big difference is that CST functions are sets, while XST functions are *behaviors*. A reconciliation of theses differences has to be formally defined. The definition of *Application* is the XST answer to the role of Image in CST.

Definition 3.8. Application: $\mathbf{f}_{(\sigma)}(\mathbf{x}) = \mathbf{f}\big[\mathbf{x}\big]_{\sigma}$.

Application dictates a set behavior that produces a result set, when the behavior is applied to a set. It is important to recognize that the application of a process produces a set while the process itself defines a behavior of a set, and does not define a specific set.

Note: For a set $\mathbf{Q}$, '$\mathbf{f}_{(\sigma)}(x) \in \mathbf{Q}$' may be true, while '$\mathbf{f}_{(\sigma)} \in \mathbf{Q}$' can not be. The expression '$\mathbf{f}_{(\sigma)}(x)$' defines a set-membership condition, while the expression '$\mathbf{f}_{(\sigma)}$' defines a set-behavior.

Though the definition of *Application* is deceptively similar to the CST definition of function, the difference is significant.

Definition 3.9. CST Function: $\mathbf{f}(\mathbf{a}) \in \mathbf{f}\big[\{\mathbf{a}\}\big]$.

In both cases $\mathbf{f}$ may be the same set of ordered pairs. However $\mathbf{f}_{(\sigma)}$ is a *behavior*, not a set. While $\mathbf{f}_{(\sigma)}(\mathbf{x})$ is a set as defined by $\mathbf{f}\big[\mathbf{x}\big]_{\sigma}$. Possibly the most notable difference between the CST definition of function and the XST definition of function is that CST functions take *elements-to-elements* while XST functions take *sets-to-sets*. However it will be shown that with sufficient manipulation all four of the following can be supported: *elements-to-elements*, *elements-to-sets*, *sets-to-elements*, and *sets-to-sets*

Similar to the constructive definition of CST Image, the XST Image to support the Application operation will be defined in terms of an XST version of Domain and an XST version of Restriction.

Definition 3.10. XST Image: $\mathbf{R}\big[\mathbf{A}\big]_{\langle \sigma_1, \sigma_2\rangle} = \text{Đ}_{\sigma_2}(\mathbf{R}\,|_{\sigma_1}\,\mathbf{A})$.

This is read as the σ_2-Domain of the σ_1-Restriction. Both of which need to be defined, along with supporting definitions.

4. Nested Applications

Application dictates a set behavior that produces a result set, when the behavior is applied to a set. What happens when a behavior is applied to a behavior, possible even itself?

Definition 4.1. Nested Application: $\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)}) = \Big(\mathbf{f}_{(\sigma)}(\mathbf{g})\Big)_{(\omega)} = \Big(\mathbf{f}\big[\mathbf{g}\big]_{\sigma}\Big)_{(\omega)}$.

Notice that $\mathbf{g}_{(\omega)}$ may or may not make mathematical sense, but whether or not it does is not relevant to the definition. Note also, an application applied to a process produces another process, not a result set!

Though application is well defined, sequences of applications are not. Consider the simple expression $\mathbf{f}_{(\sigma)}\mathbf{g}_{(\omega)}(x)$. Without proper bracketing or an explicitly defined bracketing convention, its meaning is ambiguous. There are two legitimate interpretations: $\mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}(x)\Big)$ and $\Big(\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)})\Big)(x)$. The differences are explicit in the following examples.

Example 4.1. *Interpretations of* $\mathbf{f}_{(\sigma)}\mathbf{g}_{(\omega)}(x)$.

$$(a)\quad \mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}(x)\Big) \;=\; \mathbf{f}\big[\mathbf{g}\big[x\big]_{\omega}\big]_{\sigma},$$
$$(b)\quad \Big(\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)})\Big)(x) \;=\; \Big(\mathbf{f}_{(\sigma)}(\mathbf{g})\Big)_{(\omega)}(x) \;=\; \Big(\mathbf{f}\big[\mathbf{g}\big]_{\sigma}\Big)\big[x\big]_{\omega}.$$

Example 4.2. *Interpretations of* $\mathbf{f}_{(\sigma)}\mathbf{g}_{(\omega)}\mathbf{h}_{(\tau)}(x)$.

$$(a)\quad \mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}\Big(\mathbf{h}_{(\tau)}(x)\Big)\Big) \;=\; \mathbf{f}\Big[\mathbf{g}\big[\mathbf{h}\big[x\big]_{\tau}\big]_{\omega}\Big]_{\sigma},$$
$$(b)\quad \mathbf{f}_{(\sigma)}\Big(\big(\mathbf{g}_{(\omega)}(\mathbf{h}_{(\tau)})\big)(x)\Big) \;=\; \mathbf{f}_{(\sigma)}\Big(\big(\mathbf{g}_{(\omega)}(\mathbf{h})\big)_{(\tau)}(x)\Big) \;=\; \mathbf{f}\Big[\Big(\mathbf{g}\big[\mathbf{h}\big]_{\omega}\Big)\big[x\big]_{\tau}\Big]_{\sigma},$$
$$(c)\quad \mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}(\mathbf{h}_{(\tau)})\Big)(x) \;=\; \mathbf{f}_{(\sigma)}\Big(\big(\mathbf{g}_{(\omega)}(\mathbf{h})\big)_{(\tau)}\Big)(x) \;=\; \mathbf{f}\Big[\mathbf{g}\big[\mathbf{h}\big]_{\omega}\Big]_{\sigma}\big[x\big]_{\tau},$$
$$(d)\quad \Big(\mathbf{f}_{(\sigma)}\big(\mathbf{g}_{(\omega)}\big)\big(\mathbf{h}_{(\tau)}\big)\Big)(x) \;=\; \Big(\big(\mathbf{f}_{(\sigma)}(\mathbf{g})\big)_{(\omega)}(\mathbf{h})\Big)_{(\tau)}(x) \;=\; \Big(\mathbf{f}\big[\mathbf{g}\big]_{\sigma}\big[\mathbf{h}\big]_{\omega}\Big)\big[x\big]_{\tau},$$
$$(e)\quad \Big(\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)})\Big)\Big(\mathbf{h}_{(\tau)}(x)\Big) \;=\; \Big(\mathbf{f}_{(\sigma)}(\mathbf{g})\Big)_{(\omega)}\Big(\mathbf{h}_{(\tau)}(x)\Big) \;=\; \mathbf{f}\big[\mathbf{g}\big]_{\sigma}\big[\mathbf{h}\big[x\big]_{\tau}\big]_{\omega}.$$

Note: Interpretations for a sequences greater than three gets large rather quickly with 14 for four and 42 for five.

It may not be immediately apparent that even the simplest nested case has more that one valid interpretation. Therefore it must be shown that there is a case where both interpretations are non-empty and not equal to each other, (see appendix A).

5. Spaces

For the notion of a process to be useful requires some association between the expression for the application of a process to the accepted values for input and output, or more precisely from the domain of accepted values to the codomain of accepted values. This can be achieved by specifying the collection, or space, of all allowable processes from some domain $\mathbf{A}$ to some codomain $\mathbf{B}$.

A $\mathcal{P}$-Space, process space, is the collection of all processes from one given set, $\mathbf{A}$, the *domain* to another set, $\mathbf{B}$, the *codomain*.

Since every process $\mathbf{f}_{(\sigma)}$ is associated with a specific domain $\mathbf{A}$ and a specific codomain $\mathbf{B}$, the respective domain and codomain of $\mathbf{f}_{(\sigma)}$ can be expressed in terms $\mathbf{f}$ and σ. Conversely, for any given domain $\mathbf{A}$ and codomain $\mathbf{B}$, a process space, $\mathcal{P}(\mathbf{A},\mathbf{B})$, defines the collection of all processes having $\mathbf{A}$ as its domain and $\mathbf{B}$ as its codomain. ($Sing(\mathbf{A})$ means that $\mathbf{A}$ is a singleton set and '$\subseteq$' means non-empty subset.)

Definition 5.1. Process Space, $\mathcal{P}$-Space: (where $\sigma = \langle \sigma_1, \sigma_2 \rangle$)

$$\mathcal{P}(\mathbf{A},\mathbf{B}) = \Big\{\mathbf{f}^{\sigma}\colon \mathbf{\Theta}_{\sigma_1}(\mathbf{f}) \subseteq \mathbf{A} \;\;\&\; \mathbf{\Theta}_{\sigma_2}(\mathbf{f}) \subseteq \mathbf{B}\;\big)\;\&\;(\forall x)\big(\;\mathbf{f}_{(\sigma)}(x) \subseteq \mathbf{B}\;\big)\Big\}.$$

Since the collection of all functions from $\mathbf{A}$ to $\mathbf{B}$ is a sub-collection of all processes from $\mathbf{A}$ to $\mathbf{B}$, a function space, $\mathcal{F}(\mathbf{A},\mathbf{B})$, from $\mathbf{A}$ to $\mathbf{B}$ can be define in terms of a process space $\mathcal{P}(\mathbf{A},\mathbf{B})$.

Definition 5.2. Function Space, $\mathcal{F}$-Space:

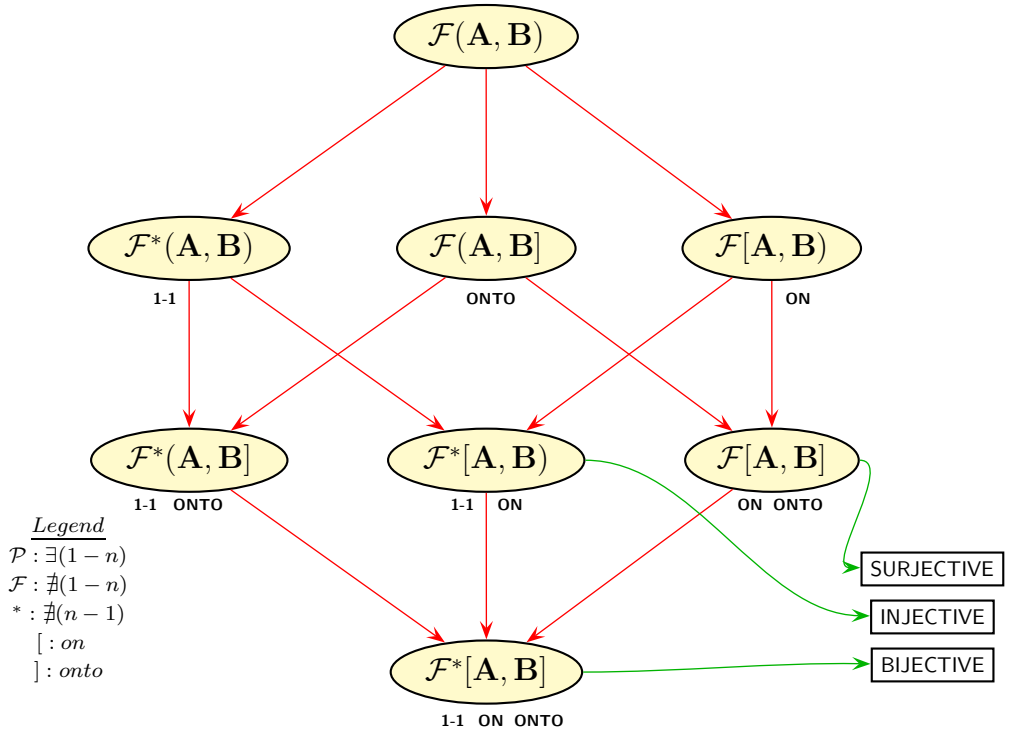
$$\mathcal{F}(\mathbf{A},\mathbf{B}) \;=\; \Big\{\, \mathbf{f}^{\sigma}\colon\ \mathbf{f} \in_{\sigma} \mathcal{P}(\mathbf{A},\mathbf{B}) \;\;\&\;\; (\forall y)\Big(\; Sing(y) \;\;\&\;\; \mathbf{f}_{(\sigma)}(y) \neq \varnothing \;\;\rightarrow\;\; Sing\big(\mathbf{f}_{(\sigma)}(y)\big) \;\Big)\Big\}.$$

The following are equivalent.

$$\mathbf{f} \in_{\sigma} \mathcal{F}(\mathbf{A},\mathbf{B}) \quad\Longleftrightarrow\quad \mathbf{f}_{(\sigma)}\colon\ \mathbf{A} \rightarrow \mathbf{B} \quad\Longleftrightarrow\quad \mathbf{A} \xrightarrow{\ \mathbf{f}_{(\sigma)}\ } \mathbf{B}$$

6. Sub-Spaces

Sub-spaces can be derived by placing restrictions on the most general form of a process space. There are domain restrictions: *on* and *onto*, and also domain associations: *many-to-one*, *one-to-one*. and *one-to-many*. Legitimate combinations of these restrictions generate 16 basic process spaces, 8 of these quality as non-empty function spaces. (see Appendix D)



The above lattice represents the the 8 non-empty process spaces that qualify as function spaces by not allowing one-to-many input/output associations.

Definition 6.1. $\mathcal{F}$-Space ON $\mathbf{A}$ *from* $\mathbf{A}$ *to* $\mathbf{B}$:

$$\mathcal{F}[\mathbf{A},\mathbf{B}) = \big\{\mathbf{f}^{\sigma}\colon\ \mathbf{f} \in_{\sigma} \mathcal{F}(\mathbf{A},\mathbf{B}) \;\;\&\;\; \text{Đ}_{\sigma_1}(\mathbf{f}) = \mathbf{A}\big\}.$$

Definition 6.2. $\mathcal{F}$-Space ONTO $\mathbf{B}$ *from* $\mathbf{A}$ *to* $\mathbf{B}$:

$$\mathcal{F}(\mathbf{A},\mathbf{B}] = \big\{\mathbf{f}^{\sigma}\colon\ \mathbf{f} \in_{\sigma} \mathcal{F}(\mathbf{A},\mathbf{B}) \;\;\&\;\; \text{Đ}_{\sigma_2}(\mathbf{f}) = \mathbf{B}\big\}.$$

Definition 6.3. $\mathcal{F}$-Space 1-1 *from* $\mathbf{A}$ *to* $\mathbf{B}$:

$$\mathcal{F}^{*}(\mathbf{A},\mathbf{B}) = \big\{\mathbf{f}^{\sigma}\colon\ \mathbf{f} \in_{\sigma} \mathcal{F}(\mathbf{A},\mathbf{B}) \;\;\&\;\; (\forall x,y)\big(\; \mathbf{f}_{(\sigma)}(x) = \mathbf{f}_{(\sigma)}(y) \neq \varnothing \;\;\rightarrow\;\; x = y \;\big)\;\big\}.$$

Consequence 6.1. *$\mathcal{F}$-Space Properties:*

$(a)\ \ \mathcal{F}[\mathbf{A},\mathbf{B}) \subseteq \mathcal{F}(\mathbf{A},\mathbf{B}),$

$(b)\ \ \mathcal{F}(\mathbf{A},\mathbf{B}] \subseteq \mathcal{F}(\mathbf{A},\mathbf{B}),$

$(c)\ \ \mathcal{F}[\mathbf{A},\mathbf{B}] \subseteq \mathcal{F}(\mathbf{A},\mathbf{B}],$

$(d)\ \ \mathcal{F}[\mathbf{A},\mathbf{B}] \subseteq \mathcal{F}[\mathbf{A},\mathbf{B}).$

Though all 8 basic function spaces have CST equivalences, three are of traditional interest.

Definition 6.4. Injective $\mathcal{F}$-Space: $\mathcal{F}^*[\mathbf{A},\mathbf{B})$.

Definition 6.5. Surjective $\mathcal{F}$-Space: $\mathcal{F}[\mathbf{A},\mathbf{B}]$.

Definition 6.6. Bijective $\mathcal{F}$-Space: $\mathcal{F}^*[\mathbf{A},\mathbf{B}]$.

With a notational refinement all 29 can be uniquely identified under five specific conditions: *on* "[", *onto* "]", *many-to-one* ">", *one-to-one* "−", and *one-to-many* "<", (see appendix E).

A more familiar notation for expressing a process (usually a function), **A** to **B** is an arrow, such as $\mathbf{f}_{(\sigma)}\colon \mathbf{A} \to \mathbf{B}$ or $\mathbf{A} \xrightarrow{\mathbf{f}_{(\sigma)}} \mathbf{B}$.

Definition 6.7. $\mathbf{f}_{(\sigma)}\colon \mathbf{A} \longrightarrow \mathbf{B} \iff \mathbf{f} \in_\sigma \mathcal{P}(\mathbf{A},\mathbf{B})$.

Definition 6.8. $\mathbf{f}_{(\sigma)}\colon \mathbf{A} \xrightarrow{\bullet} \mathbf{B} \iff \mathbf{f} \in_\sigma \mathcal{F}_{\_}[\mathbf{A},\mathbf{B}]$.

7. Set Theoretic Support

The definition of process is independent of any specific choice of supporting set theory, but one has to be chosen for any set behavior to be realized. The choice of set theory chosen for the rest of this paper follows the axioms of an extended set theory[1].

Though Image can be defined directly, its behavior is more intuitive if recognized as a two step process.

Definition 7.1. Image: $\mathbf{R}\big[\mathbf{A}\big]_{\langle\sigma_1,\sigma_2\rangle} = \mathbf{Đ}_{\sigma_2}(\mathbf{R}\,|_{\sigma_1}\,\mathbf{A})$.

This is read as the σ_2-Domain of the σ_1-Restriction. Both of which need to be formally defined, along with supporting definitions.

Definition 7.2. Ordered Pair: $\langle x,y\rangle = \{\ x^1, y^2\ \}$.

Definition 7.3. Re-Scope by Scope: $\mathbf{A}^{/\sigma/} = \big\{\mathbf{x}^w\ \ (\exists s)\big(\ \mathbf{x} \in_s \mathbf{A}\ \ \&\ \ s \in_w \sigma\ \big)\big\}$.

For example: $\{a^x, b^y, c^z\}^{/\{x^1,y^2,z^3\}/} = \{a^1, b^2, c^3\}$,

Definition 7.4. σ-Domain:

$$\mathbf{Đ}_\sigma(\mathbf{R}) = \Big\{\ \mathbf{x}^s\colon (\exists \mathbf{z}, w)\big(\ \mathbf{z} \in_w \mathbf{R}\ \ \&\ \mathbf{x} = \mathbf{z}^{/\sigma/} \neq \varnothing\ \ \&\ s = w^{/\sigma/}\ \big)\Big\}.$$

For example:

$\mathbf{Đ}_{\{A^1,C^2\}}(\{\{a^A, b^B, c^C\}\}) = \{\{a^1, c^2\}\}$,

$\mathbf{Đ}_{\langle 3,1\rangle}(\{\{a^1, b^2, c^3\}^{\{A^1,B^2,C^3\}}\}) = \{\langle c, a\rangle^{\langle C,A\rangle}\}$,

$\mathbf{Đ}_{\{3^1,1^2,y^9,v^5,v^7,R^A\}}(\{\{a^1, b^2, c^3\}^{\{x^y,w^v,z^R\}}\}) = \{\langle c, a\rangle^{\{x^9,w^5,w^7,z^A\}}\}$.

The usual CST properties associated with the Domain operation are preserved in XST.

Consequence 7.1. *Preserved Domain Properties:*

$(a)\ \mathbf{Đ}_\sigma(\mathbf{R} \cup \mathbf{R}) = \mathbf{Đ}_\sigma(\mathbf{R}) \cup \mathbf{Đ}_\sigma(\mathbf{R}),$

$(b)\ \mathbf{Đ}_\sigma(\mathbf{R} \cap \mathbf{R}) \subseteq \mathbf{Đ}_\sigma(\mathbf{R}) \cap \mathbf{Đ}_\sigma(\mathbf{R}),$

$(c)\ \mathbf{Đ}_\sigma(\mathbf{R}) \sim \mathbf{Đ}_\sigma(\mathbf{R}) \subseteq \mathbf{Đ}_\sigma(\mathbf{R} \sim \mathbf{R}),$

$(d)\ \mathbf{R} \odot \mathbf{R} \longrightarrow \mathbf{Đ}_\sigma(\mathbf{R}) \subseteq \mathbf{Đ}_\sigma(\mathbf{R})$,
$(e)\ \mathbf{Đ}_\varnothing(\mathbf{R}) = \varnothing$.

Definition 7.5. Re-Scope by Element: $\mathbf{A}^{\backslash\sigma\backslash} = \{\mathbf{x}^w\colon (\exists s)\big(\ \mathbf{x} \in_s \mathbf{A}\ \ \&\ \ w \in_s \sigma\ \big)\}$.

For example: $\{a^1, b^2, c^3\}^{\backslash\{w^1, v^2, t^3\}\backslash} = \{a^w, b^v, c^t\}$.

Definition 7.6. σ-Restriction: $\mathbf{R}\,|_{\sigma}\,\mathbf{A} = \Big\{\ \mathbf{z}^w\colon \big(\ \mathbf{z} \in_w \mathbf{R}\big)\ \&\ \big(\exists\, a, s\big)\big(a \in_s \mathbf{A}\ \&\ a^{\backslash\sigma\backslash} \odot \mathbf{z}\ \ \&\ s^{\backslash\sigma\backslash} \subseteq w\big)\Big\}$.

Given that $\mathbf{R}\big[\mathbf{A}\big]_{\langle\sigma_1,\sigma_2\rangle} = \mathbf{Đ}_{\sigma_2}(\mathbf{R}\,|_{\sigma_1}\,\mathbf{A})$, the usual CST properties associated with the Image operation are preserved in XST (see Appendix C).

8. Functions & Sets

In CST $\mathbf{f}(a) = b$ implies that $\mathbf{f}$ is a function and (using $\mathbf{f}\big[\{a\}\big]$ for CST Image) that if $x \in \mathbf{f}\big[\{a\}\big]$ and $y \in \mathbf{f}\big[\{a\}\big]$, then $x = y = b$. Application in XST has a similar reliance on Image.

Definition 8.1. Application: $\mathbf{f}_{(\sigma)}(x) = \mathbf{f}\big[x\big]_\sigma$.

Given the above definition for instantiating a behavior through application, a more set oriented definition of function can be provided by the following.

Definition 8.2. Function: $\mathbf{f}_{(\sigma)}$ is a function $\iff (\forall y)\Big(\ Sing(y)\ \ \&\ \ \mathbf{f}\big[y\big]_\sigma \neq \varnothing\ \rightarrow\ \ Sing\Big(\mathbf{f}\big[y\big]_\sigma\Big)\ \Big)$.

Consequence 8.1. *Function Properties:*

$(a)\ \big(\mathbf{f} \cup \mathbf{g}\big)_{(\sigma)}(x) = \mathbf{f}_{(\sigma)}(x) \cup \mathbf{g}_{(\sigma)}(x)$,
$(b)\ \big(\mathbf{f} \cap \mathbf{g}\big)_{(\sigma)}(x) \subseteq \mathbf{f}_{(\sigma)}(x) \cap \mathbf{g}_{(\sigma)}(x)$,
$(c)\ \mathbf{f}_{(\sigma)}(x) \sim \mathbf{g}_{(\sigma)}(x) \subseteq \big(\mathbf{f} \sim \mathbf{g}\big)_{(\sigma)}(x)$.

Example 8.1.

Given: $\mathbf{f} = \Big\{\ <a, x>^{<A,Z>},\ <b, y>^{<B,Y>},\ <c, x>^{<A,Z>}\ \Big\}$,

$(a)\ \ \mathbf{f}_{(\sigma)}\big(\{<a>^{<A>}\}\big) = \mathbf{f}\big[\{<a>^{<A>}\}\big]_\sigma = \{<x>^{<Z>}\},\ \ \&\ \ \sigma = <<1>, <2>>$,

$(b)\ \ \mathbf{f}_{(\tau)}\big(\{<x>^{<Z>}\}\big) = \mathbf{f}\big[\{<x>^{<Z>}\}\big]_\tau = \{<a>^{<A>},\ <c>^{<A>}\},\ \ \&\ \ \tau = <<2>, <1>>$,

In the above example $\mathbf{Đ}_{\sigma_1}(\mathbf{f}) = \{<a>^{<A>}, <b>^{<B>}, <c>^{<A>}\}$ and $\mathbf{Đ}_{\sigma_2}(\mathbf{f}) = \{\ <x>^{<Z>}, <y>^{<Y>}\}$. In this example, $\mathbf{f}_{(\sigma)}$ behaves as a function from $\mathbf{Đ}_{\sigma_1}(\mathbf{f})$ to $\mathbf{Đ}_{\sigma_2}(\mathbf{f})$ with $\mathbf{f}_{(\tau)}$ behaving as its inverse, but not as a function.

When functions are defined as subsets of a Cartesian product, the ability to formalize the notion of self-application, in the sense that $\mathbf{f}\big[\mathbf{f}\big] \neq \varnothing$, becomes difficult to express. Since functions under XST are not defined as subsets of a Cartesian product, the concept of self-application, in the sense of a set acting on itself, is somewhat easier to capture. (see Appendix B)

9. Tuples & Products

The classical formulation of n-tuples, as nested sets, presents challenges when n-tuples are used as operands[5]. The following definitions replace these old challenges with new ones.

Definition 9.1. n-Tuple: $tup(x) = n \iff x = \{\ x_1^1, x_2^2, .., x_n^n\ \}$.

This definition confines tuples to be finite, which is in keeping with the traditional use and intent of tuples.

Definition 9.2. Tuple Concatenation: $x \cdot y \;=\; z \iff$
$(\exists n, m)\Big(tup(x) = n \;\&\; tup(y) = m \;\&\; z = \{\; x_1^1, x_2^2, .., x_n^n, y_1^{n+1}, y_2^{n+2}, .., y_m^{n+m} \;\}\Big)$

EXAMPLE: $\langle a, b, c, d\rangle \cdot \langle w, x, y, z\rangle \;=\; \langle a, b, c, d, w, x, y, z\rangle$.
Note: $tup(x) = n \;\&\; tup(y) = m \;\longrightarrow\; tup(x \cdot y) = n + m$.

Definition 9.3. XST Cross Product: $\mathbf{A} \otimes \mathbf{B} = \{(x \cdot y)^{(s \cdot t)} : (x \in_s \mathbf{A} \;\&\; y \in_t \mathbf{B})\}$.

Theorem 9.4. $\mathbf{A} \otimes \mathbf{B} \otimes \mathbf{C} \;=\; \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}) \;=\; (\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C}$.

Definition 9.5. Tag: $\mathbf{A}^{(a)} = \{\{x^a\}^{\{s^a\}} : x \in_s \mathbf{A}\}$. (for $s \neq \varnothing$)

Definition 9.6. Tag: $\mathbf{A}^{(a)} = \{\{x^a\} : x \in_s \mathbf{A}\}$. (for $s = \varnothing$)

For backward compatibility with graph-based function definitions, CST Cartesian product can be defined so as to preserve all its CST properties.

Definition 9.7. CST Cartesian Product: $\mathbf{A} \times \mathbf{B} \;=\; \mathbf{A}^{(1)} \otimes \mathbf{B}^{(2)}$.

Definition 9.8. σ-Value: $\mathcal{V}_\sigma(\mathbf{x}) = \mathbf{b} \iff (\forall y)(\; \langle y\rangle \in_{\langle\sigma\rangle} \mathbf{x} \to y = \mathbf{b}\;)$.

Definition 9.9. Value: $\mathcal{V}(\mathbf{x}) = \mathbf{b} \iff (\forall y)(\; \langle y\rangle \in \mathbf{x} \to y = \mathbf{b}\;)$.

Example 9.1. *Square root function.*

$\sqrt{\sqrt{16}} = \{\; \langle 2\rangle^{\langle +\rangle},\; \langle -2\rangle^{\langle -\rangle},\; \langle 2i\rangle^{\langle i\rangle},\; \langle -2i\rangle^{\langle -i\rangle}\;\}$

$\mathcal{V}_+(\sqrt{\sqrt{16}}) = 2$

$\mathcal{V}_-(\sqrt{\sqrt{16}}) = -2$

$\mathcal{V}_i(\sqrt{\sqrt{16}}) = 2i$

$\mathcal{V}_{-i}(\sqrt{\sqrt{16}}) = -2i$

All CST element-based functions can be represented by XST set-based functions.

Theorem 9.10. *Given* $\mathbf{f} \subseteq \mathbf{A} \times \mathbf{B}$) *and* $\sigma = \langle\langle 1\rangle, \langle 2\rangle\rangle$, $\mathbf{f}(x) = \mathcal{V}(\mathbf{f}_{(\sigma)}(\{\langle x\rangle\}))$.

10. Relative Product

Relative Product has more personality than other operations in that given the same two operands the resultant set can have many forms. Though, in CST the operation is rather bland matching the range elements of the first operand with the domain elements of the second operand and producing a pair of the domain element of the first with the range element of the second.

For example, in CST: $\{\langle a, b\rangle\}/\{\langle b, c\rangle\} = \{\langle a, c\rangle\}$. Following are some element combinations that are potentially more interesting:

1) $\langle a, b\rangle \;\&\langle b, c\rangle \;\longrightarrow\; \langle a, c\rangle$,
2) $\langle a, b\rangle \;\&\langle b, c\rangle \;\longrightarrow\; \langle a, b, c\rangle$,
3) $\langle a, b\rangle \;\&\langle a, c\rangle \;\longrightarrow\; \langle a, b, c\rangle$,
4) $\langle a, b\rangle \;\&\langle a, c\rangle \;\longrightarrow\; \langle b, c\rangle$,
5) $\langle a, c\rangle \;\&\langle b, c\rangle \;\longrightarrow\; \langle a, b, c\rangle$,
6) $\langle a, c\rangle \;\&\langle b, c\rangle \;\longrightarrow\; \langle a, b\rangle$,
7) $\langle a, b, c\rangle \;\&\langle x, y, c, b\rangle \;\longrightarrow\; \langle b, c, a, y, b, c, x, x\rangle$,
8) $\langle a, b, c, w, v\rangle \;\&\langle a, b, c, x, y, z\rangle \;\longrightarrow\; \langle a, b, c, w, v, x, y, z\rangle$.

All of the above are producible with the following definition.

Definition 10.1. Relative Product:

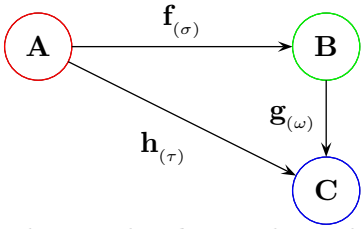
$$\mathbf{F}\Big/_{\langle\sigma_1,\sigma_2\rangle}^{\langle\omega_1,\omega_2\rangle}\mathbf{G} = \Big\{\mathbf{z}^{\tau}\colon(\exists x,s,y,t)(\ x\in_s\mathbf{F}\ \&\ y\in_t\mathbf{G}\ \ \&\ \ x^{/\sigma_2/}=y^{/\omega_1/}\ \ \&\ \ s^{/\sigma_2/}=t^{/\omega_1/}\ \ \&\ \ \mathbf{z}=x^{/\sigma_1/}\cup y^{/\omega_2/}\ \ \&\ \ \tau=s^{/\sigma_1/}\cup t^{/\omega_2/})\ \Big\}.$$

For $\mathbf{f}\Big/_{\sigma}^{\omega}\mathbf{g}$ the following values for σ and ω support the corresponding mappings above.

1) $\sigma=\langle\{1^1\},\ \{2^1\}\rangle\ \ \&\ \ \omega=\langle\{1^1\},\ \{2^2\}\rangle$,
2) $\sigma=\langle\{1^1\},\ \{2^1\}\rangle\ \ \&\ \ \omega=\langle\{1^1\},\ \{1^2,2^3\}\rangle$,
3) $\sigma=\langle\{1^1,2^2\},\ \{1^1\}\rangle\ \ \&\ \ \omega=\langle\{1^1\},\ \{2^3\}\rangle$,
4) $\sigma=\langle\{2^1\},\ \{1^1\}\rangle\ \ \&\ \ \omega=\langle\{1^1\},\ \{2^2\}\rangle$,
5) $\sigma=\langle\{1^1\},\ \{2^1\}\rangle\ \ \&\ \ \omega=\langle\{2^1\},\ \{1^2,2^3\}\rangle$,
6) $\sigma=\langle\{1^1\},\ \{2^1\}\rangle\ \ \&\ \ \omega=\langle\{2^1\},\ \{1^2\}\rangle$,
7) $\sigma=\langle\{2^1,3^2,1^3\},\{2^1,3^2\}\rangle\ \ \&\ \ \omega=\langle\{4^1,3^2\}\{2^4,4^5,3^6,1^7,1^8\}\rangle$,
8) $\sigma=\langle\{1^1,2^2,3^3,4^4,5^5\},\ \{1^1,2^2,3^3\}\rangle\ \ \&\ \ \omega=\langle\{1^1,2^2,3^3\},\ \{4^6,5^7,6^8\}\rangle$.

11. Composition

Composition is an act of aggregating the interactive resultant behavior of multiple processs into a single process. Besides its categorical relevance for studying equivalent system behaviors, it can be used constructively to produce alternative computer programs by eliminating the intermediate operations indirectly contributing to resultant behavior. The execution of operation, **h**, in the diagram below is equivalent to executing **f** followed by an execution of **g**.



Two problems arise. It is not obvious that **h** can always be defined in terms of a given **f** and **g**, and even if it can be, it may not be of any value since processs are abstract mathematical objects with no authority to execute on a computer,

It needs to be shown that given any two processes that are compositable there is always a constructible composition and that the resultant process is definable in terms of appropriate sets.

Definition 11.1. Composition: $\mathbf{g}_{(\omega)}\circ\ \mathbf{f}_{(\sigma)} = \Big(\ \mathbf{f}\Big/_{\langle\sigma_1,\ \sigma_2\rangle}^{\langle\omega_1,\ \omega_2\rangle}\mathbf{g}\ \Big)_{(\langle\sigma_1,\ \omega_2\rangle)}$.

Theorem 11.2. *For* $\mathbf{f}\in_\sigma\mathcal{F}[\mathbf{A},\mathbf{B})$, $\mathbf{g}\in_\omega\mathcal{F}[\mathbf{B},\mathbf{C})$ *and if* $\mathbf{g}_{(\omega)}\circ\ \mathbf{f}_{(\sigma)}$ *exists, there exists* **h** *and* τ *such that* $\mathbf{h}_{(\tau)}=\mathbf{g}_{(\omega)}\circ\ \mathbf{f}_{(\sigma)}$ *and* $\mathbf{h}\in_\tau\mathcal{F}[\mathbf{A},\mathbf{C})$.

Proof: 1) $\mathbf{f}\in_\sigma\mathcal{F}[\mathbf{A},\mathbf{B})\implies\mathbf{Đ}_{\sigma_1}(\mathbf{f})=\mathbf{A}\ \ \&\ \ \mathbf{Đ}_{\sigma_2}(\mathbf{f})\subseteq\!\!\!\cdot\ \mathbf{B}$

2) $\mathbf{g}\in_\omega\mathcal{F}[\mathbf{B},\mathbf{C})\implies\mathbf{Đ}_{\omega_1}(\mathbf{g})=\mathbf{B}\ \ \&\ \ \mathbf{Đ}_{\omega_2}(\mathbf{g})\subseteq\!\!\!\cdot\ \mathbf{C}$

3) Let $\mathbf{h}=\mathbf{f}\Big/_{\langle\sigma_1,\sigma_2\rangle}^{\langle\omega_1,\omega_2\rangle}\mathbf{g}$

4) For $\tau=\langle\sigma_1,\omega_2\rangle$, $\mathbf{h}=\mathbf{f}\Big/_{\langle\tau_1,\sigma_2\rangle}^{\langle\omega_1,\tau_2\rangle}\mathbf{g}$

5) $\mathbf{g}_{(\omega)}\circ\ \mathbf{f}_{(\sigma)}\ =\ \Big(\ \mathbf{f}\Big/_{\langle\tau_1,\ \sigma_2\rangle}^{\langle\omega_1,\ \tau_2\rangle}\mathbf{g}\ \Big)_{(\tau)}\ =\ \mathbf{h}_{(\tau)}$

6) $\mathbf{Đ}_{\tau_1}(\mathbf{h})=\mathbf{Đ}_{\sigma_1}(\mathbf{f})=\mathbf{A}$ and $\mathbf{Đ}_{\tau_2}(\mathbf{h})=\mathbf{Đ}_{\omega_2}(\mathbf{g})\subseteq\!\!\!\cdot\ \mathbf{C}\ \rightarrow\ \mathbf{h}\in_\tau\mathcal{F}[\mathbf{A},\mathbf{C})$.

12. Summary

The above material defines a **process** to be any mechanism that accepts well-defined inputs and produces well-defined outputs. Every computer program accepts inputs and produces outputs, but not always with the expected results. These programs are disqualified from being processes by not having well-defined inputs, not having well-defined outputs, or by not having a formal mechanism for transforming inputs to outputs. This disqualification is not surprising, since computer programs process data representations modeled as physical objects, not as well-defined mathematical operands.

Since all data representations have a mathematical identity (in terms of structured-sets as supported by XST), all data representations can be managed as mathematical operands. Thus the data management component of any computer program can qualify as a **process**.

Since the composition theorem (11.2) asserts *that given any two processes that are compositable there is always a constructible composition and that the resultant process is definable in terms of appropriate sets*, it is not only possible to establish reliable data management behavior, but to also optimize the performance of that behavior.

References

[1] Blass, A.; Childs, D L: *Axioms and Models for an Extended Set Theory*, A Formal Foundation for Unified Modeling of Mathematical Objects - 2011 http://www.math.lsa.umich.edu/%7Eablass/XST_Axioms.pdf

[2] Childs, D L:: *Why Not Sets?* Why are sets not more widely used in modeling the behavior and assisting the development of computing systems? - 2010 http://xsp.xegesis.org/WNSETS.pdf

[3] Childs, D L:: *XSP: Extended Set Processing*, Mathematically Managing Data Representations - 2015 http://xsp.xegesis.org/ExcSum.pdf

[4] Childs, D L:: *Set Processing vs. Record Processing Performance*, Dynamic Data Restructuring vs. Prestructured Data Storage - 2015 http://xsp.xegesis.org/SPvsRP.pdf

[5] Skolem, Thoralf: *Two Remarks on Set Theory,* Mathematica Scandinavica 5 (1957), p.43-46. http://xsp.xegesis.org/T_Skolem.pdf

Appendix A. Nested Application

Though application is well defined, sequences of applications are not. Consider the simple expression $\mathbf{f}_{(\sigma)}\mathbf{g}_{(\omega)}(x)$. Without proper bracketing or an explicitly defined bracketing convention, its meaning is ambiguous. There are two legitimate interpretations: $\mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}(x)\Big)$ and $\Big(\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)})\Big)(x)$. The differences are explicit in the following examples.

Example A.1. *Interpretations of* $\mathbf{f}_{(\sigma)}\mathbf{g}_{(\omega)}(x)$.

$$(a)\quad \mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}(x)\Big) \;=\; \mathbf{f}\big[\mathbf{g}\big[x\big]_{\omega}\big]_{\sigma},$$
$$(b)\quad \Big(\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)})\Big)(x) \;=\; \Big(\mathbf{f}_{(\sigma)}(\mathbf{g})\Big)_{(\omega)}(x) \;=\; \Big(\mathbf{f}\big[\mathbf{g}\big]_{\sigma}\Big)\big[x\big]_{\omega}.$$

It may not be immediately apparent that even the simplest case has more that one valid interpretation. Therefore it must be shown that there is a case where both interpretations are non-empty and not equal to each other,

Example A.2. *Present a case such that:*

$$\mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}(\mathbf{h})\Big) \neq \varnothing,\quad \Big(\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)})\Big)(\mathbf{h}) \neq \varnothing \quad \textit{and} \quad \mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}(\mathbf{h})\Big) \neq \Big(\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)})\Big)(\mathbf{h}).$$

Let:

$$\mathbf{f} \;=\; \{\; \langle y, z\rangle^{\langle\varnothing,\varnothing\rangle},\; \langle a, x, b, k\rangle^{\langle\varnothing,\varnothing,\varnothing,\varnothing\rangle}\;\}$$
$$\mathbf{g} \;=\; \{\; \langle x, y\rangle^{\langle\varnothing,\varnothing\rangle},\; \langle a, b\rangle^{\langle\varnothing,\varnothing\rangle}\;\},$$
$$\mathbf{p} \;=\; \{\; \langle x, k\rangle^{\langle\varnothing,\varnothing\rangle}\;\},$$

$\mathbf{h} \;=\; \{\ \langle x\rangle^{\langle\varnothing\rangle}\ \}$
$\sigma \;=\; \langle\langle 1,3\rangle,\langle 2,4\rangle\rangle,$
$\omega \;=\; \langle\langle 1\rangle,\langle 2\rangle\rangle.$

$\text{Ð}_{\sigma_1}(\mathbf{f}) = \{\ \langle y\rangle^{\langle\varnothing\rangle},\ \langle a,b\rangle^{\langle\varnothing,\varnothing\rangle}\ \},$
$\text{Ð}_{\sigma_2}(\mathbf{f}) = \{\ \langle z\rangle^{\langle\varnothing\rangle},\ \langle x,k\rangle^{\langle\varnothing,\varnothing\rangle}\ \},$
$\text{Ð}_{\omega_1}(\mathbf{g}) = \{\ \langle x\rangle^{\langle\varnothing\rangle},\ \langle a\rangle^{\langle\varnothing\rangle}\ \},$
$\text{Ð}_{\omega_2}(\mathbf{g}) = \{\ \langle y\rangle^{\langle\varnothing\rangle},\ \langle b\rangle^{\langle\varnothing\rangle}\ \}.$

Then:

$\mathbf{f}_{(\sigma)}\big(\{\langle y\rangle^{\langle\varnothing\rangle}\}\big) \;=\; \{\langle z\rangle^{\langle\varnothing\rangle}\}$
$\mathbf{f}_{(\sigma)}(\mathbf{g}) \;=\; \{\langle x,k\rangle^{\langle\varnothing,\varnothing\rangle}\}$
$\mathbf{g}_{(\omega)}(\mathbf{h}) \;=\; \{\langle y\rangle^{\langle\varnothing\rangle}\}$
$\mathbf{p}_{(\omega)}(\mathbf{h}) \;=\; \{\langle k\rangle^{\langle\varnothing\rangle}\}$ *and by substitution,*
$\mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}(\mathbf{h})\Big) \;=\; \mathbf{f}_{(\sigma)}\big(\{\langle y\rangle^{\langle\varnothing\rangle}\}\big) \;=\; \{\langle z\rangle^{\langle\varnothing\rangle}\}$ *and*
$\Big(\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)})\Big)(\mathbf{h}) \;=\; \mathbf{p}_{(\omega)}(\mathbf{h}) \;=\; \{\langle k\rangle^{\langle\varnothing\rangle}\}.$

Therefore, for $k \neq z$, $\mathbf{f}_{(\sigma)}\Big(\mathbf{g}_{(\omega)}(\mathbf{h})\Big) \neq \Big(\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)})\Big)(\mathbf{h})$.

Appendix B. Self Application

When functions are defined as subsets of a Cartesian product, the ability to formalize the notion of self-application, in the sense that $\mathbf{f}[\mathbf{f}] \neq \varnothing$, becomes difficult to express. Since functions under XST are not defined as subsets of a Cartesian product, the concept of self-application, in the sense of a set acting on itself, is somewhat easier to capture.

A simple, non-trivial example of self-application, difficult to express when functions are modeled as objects instead of as processes, can be generated from examining the functions from a set of cardinality 2 to itself.

Let $\mathbf{A} = \{\langle a\rangle,\langle b\rangle\}$, with $\mathbf{g}_1 = \{\langle a,a\rangle,\langle b,b\rangle\}$, $\mathbf{g}_2 = \{\langle a,a\rangle,\langle b,a\rangle\}$, $\mathbf{g}_3 = \{\langle a,b\rangle,\langle b,a\rangle\}$, and $\mathbf{g}_4 = \{\langle a,b\rangle,\langle b,b\rangle\}$, then for $\sigma = \langle\langle 1\rangle,\langle 2\rangle\rangle$:

$\mathbf{g}_{1(\sigma)}\Big(\{\langle a\rangle\}\Big) = \{\langle a\rangle\}, \quad \mathbf{g}_{1(\sigma)}\Big(\{\langle b\rangle\}\Big) = \{\langle b\rangle\},$
$\mathbf{g}_{2(\sigma)}\Big(\{\langle a\rangle\}\Big) = \{\langle a\rangle\}, \quad \mathbf{g}_{2(\sigma)}\Big(\{\langle b\rangle\}\Big) = \{\langle a\rangle\},$
$\mathbf{g}_{3(\sigma)}\Big(\{\langle a\rangle\}\Big) = \{\langle b\rangle\}, \quad \mathbf{g}_{3(\sigma)}\Big(\{\langle b\rangle\}\Big) = \{\langle a\rangle\},$ and
$\mathbf{g}_{4(\sigma)}\Big(\{\langle a\rangle\}\Big) = \{\langle b\rangle\}, \quad \mathbf{g}_{4(\sigma)}\Big(\{\langle b\rangle\}\Big) = \{\langle b\rangle\}.$

Let $\mathbf{f} = \{\langle a,a,a,b,b\rangle,\langle b,b,a,a,b\rangle\}$ and $\omega = \langle\langle 1\rangle,\langle 1,3,4,5,2\rangle\rangle$, then

a) $\mathbf{f}_{(\sigma)} = \mathbf{g}_{1(\sigma)}$,

b) $\mathbf{f}_{(\omega)}(\mathbf{f}_{(\sigma)}) = \mathbf{g}_{2(\sigma)}$,

c) $\big(\mathbf{f}_{(\omega)}(\mathbf{f}_{(\omega)})\big)(\mathbf{f}_{(\sigma)}) = \mathbf{g}_{3(\sigma)}$,

d) $\Big(\mathbf{f}_{(\omega)}\big(\mathbf{f}_{(\omega)}(\mathbf{f}_{(\omega)})\big)\Big)(\mathbf{f}_{(\sigma)}) = \mathbf{g}_{4(\sigma)}$,

Other equalities:

$\mathbf{f}_{(\sigma)}\big(\{\langle a\rangle\}\big) = \{\langle a\rangle\}, \quad \mathbf{f}_{(\sigma)}\big(\{\langle b\rangle\}\big) = \{\langle b\rangle\}, \quad \mathbf{f}_{(\sigma)} = \mathbf{I}_{\mathbf{A}}.$

Notice that nothing in the definition of a function requires the resultant behavior to be functional.

In fact, $\mathbf{f}_{(\tau)}$ in example [8.1] demonstrates the case where it is not. Following is a validation of the equations above. Recall the following definitions:

Definition B.1. Process Equivalence: $\mathbf{f}_{(\sigma)} = \mathbf{g}_{(\gamma)} \iff (\forall x)\big(\, \mathbf{f}\big[x\big]_{\sigma} = \mathbf{g}\big[x\big]_{\gamma} \,\big)$.

The above equivalence does not assert a unique mathematical identity for set membership, but does assert a unique mathematical identity for a process.

Consequence B.1. $\mathbf{f}_{(\sigma)} = \mathbf{g}_{(\gamma)} \longrightarrow \mathbf{Đ}_{\sigma_1}(\mathbf{f}) = \mathbf{Đ}_{\gamma_1}(\mathbf{g}) \quad \& \quad \mathbf{Đ}_{\sigma_2}(\mathbf{f}) = \mathbf{Đ}_{\gamma_2}(\mathbf{g})$.

Consequence B.2. $\mathbf{f}_{(\sigma)} = \mathbf{g}_{(\gamma)} \quad \& \quad \mathbf{g}_{(\gamma)} = \mathbf{h}_{(\tau)} \longrightarrow \mathbf{f}_{(\sigma)} = \mathbf{h}_{(\tau)}$.

Consequence B.3. *Negative Properties:*

(*a*) $\mathbf{f}_{(\sigma)} = \mathbf{g}_{(\sigma)} \not\Longrightarrow \mathbf{f} = \mathbf{g}$.

Ex: $\mathbf{f} = \{< a, b, x >\},\ \mathbf{g} = \{< a, b, w >\},\ \sigma =<< 1 >, < 2 >> .$

(*b*) $\mathbf{f}_{(\sigma)} = \mathbf{f}_{(\gamma)} \not\Longrightarrow \sigma = \gamma$.

Ex: $\mathbf{f} = \{< a, b >, < b, a >\},\ \sigma =<< 1 >, < 2 >>,\ \gamma =<< 2 >, < 1 >> .$

(*c*) $\mathbf{f}_{(\sigma)} = \mathbf{f}_{(\gamma)} \ \& \ \mathbf{g} \Subset \mathbf{f} \not\Longrightarrow \mathbf{g}_{(\sigma)} = \mathbf{g}_{(\gamma)}$.

Ex: $\mathbf{f} = \{< a, b >, < b, a >\},\ \mathbf{g} = \{< a, b >\},\ \sigma =<< 1 >, < 2 >>,\ \gamma =<< 2 >, < 1 >> .$

(*d*) $\mathbf{f}_{(\sigma)} = \mathbf{g}_{(\gamma)} \ \& \ \mathbf{f} \in_{\sigma} \mathbf{A} \not\Longrightarrow \mathbf{g} \in_{\gamma} \mathbf{A}$.

Ex: $\mathbf{A} = \{\ \{< a, b >\}^{<<1>,<2>>}\},\ \mathbf{B} = \{\ \{< b, a >\}^{<<2>,<1>>}\},\ \mathbf{f} \in_{\sigma} \mathbf{A},\ \&\ \mathbf{g} \in_{\gamma} \mathbf{B}.$

Definition B.2. Application: $\mathbf{f}_{(\sigma)}(\mathbf{x}) = \mathbf{f}\big[\mathbf{x}\big]_{\sigma} = \mathbf{Đ}_{\sigma_2}(\mathbf{f}\,|_{\sigma_1}\mathbf{x})$.

Definition B.3. Nested Application: $\mathbf{f}_{(\sigma)}(\mathbf{g}_{(\omega)}) = \Big(\mathbf{f}_{(\sigma)}(\mathbf{g})\Big)_{(\omega)} = \Big(\mathbf{f}\big[\mathbf{g}\big]_{\sigma}\Big)_{(\omega)} = \mathbf{Đ}_{\sigma_2}(\mathbf{f}\,|_{\sigma_1}\mathbf{g})_{(\omega)}$.

For $\mathbf{f} = \{\langle a, a, a, b, b\rangle, \langle b, b, a, a, b\rangle\},\ \sigma = \langle\langle 1\rangle, \langle 2\rangle\rangle,\ \ \omega = \langle\langle 1\rangle, \langle 1, 3, 4, 5, 2\rangle\rangle$ and $\mathbf{x} = \{\langle a\rangle\}$ or $\mathbf{x} = \{\langle b\rangle\}$.

a) $\mathbf{f}_{(\sigma)}\Big(\{\langle a\rangle\}\Big) = \mathbf{Đ}_{\sigma_2}(\mathbf{f}\,|_{\sigma_1}\{\langle a\rangle\}\ )$
$= \mathbf{Đ}_{\langle 2\rangle}(\{\langle a, a, a, b, b\rangle, \langle b, b, a, a, b\rangle\}|_{\langle 1\rangle}\{\langle a\rangle\}\ )$
$= \mathbf{Đ}_{\langle 2\rangle}(\{\langle a, a, a, b, b\rangle\})$
$= \{\langle a\rangle\}$

b) $\mathbf{f}_{(\sigma)}\Big(\{\langle b\rangle\}\Big) = \mathbf{Đ}_{\sigma_2}(\mathbf{f}\,|_{\sigma_1}\{\langle b\rangle\}\ )$
$= \mathbf{Đ}_{\langle 2\rangle}(\{\langle a, a, a, b, b\rangle, \langle b, b, a, a, b\rangle\}|_{\langle 1\rangle}\{\langle b\rangle\}\ )$
$= \mathbf{Đ}_{\langle 2\rangle}(\{\langle b, b, a, a, b\rangle\})$
$= \{\langle b\rangle\}$

c) $\mathbf{f}_{(\omega)}\Big(\{\langle a\rangle\}\Big) = \mathbf{Đ}_{\omega_2}(\mathbf{f}\,|_{\omega_1}\{\langle a\rangle\}\ )$
$= \mathbf{Đ}_{\langle 1,3,4,5,2\rangle}(\{\langle a, a, a, b, b\rangle, \langle b, b, a, a, b\rangle\}|_{\langle 1\rangle}\{\langle a\rangle\}\ )$
$= \mathbf{Đ}_{\langle 1,3,4,5,2\rangle}(\{\langle a, a, a, b, b\rangle\})$
$= \{\langle a, a, b, b, a\rangle\}$

d) $\mathbf{f}_{(\omega)}\Big(\{\langle b\rangle\}\Big) = \mathbf{Đ}_{\omega_2}(\mathbf{f}\,|_{\omega_1}\{\langle b\rangle\}\ )$
$= \mathbf{Đ}_{\langle 1,3,4,5,2\rangle}(\{\langle a, a, a, b, b\rangle, \langle b, b, a, a, b\rangle\}|_{\langle 1\rangle}\{\langle b\rangle\}\ )$
$= \mathbf{Đ}_{\langle 1,3,4,5,2\rangle}(\{\langle b, b, a, a, b\rangle\})$
$= \{\langle b, a, a, b, b\rangle\}$

B.1. **Derivations.**

a) $\mathbf{f}_{(\sigma)} = \mathbf{g}_{1(\sigma)}$,
$\mathbf{f}_{(\sigma)}\Big(\{\langle a\rangle\}\Big) = \mathbf{Đ}_{\sigma_2}(\mathbf{f}\,|_{\sigma_1}\{\langle a\rangle\}\ )$

$$\mathbf{f}_{(\sigma)}\Big(\{\langle a\rangle\}\Big) = \mathbf{Ð}_{\sigma_2}(\{\langle a,a,a,b,b\rangle,\langle b,b,a,a,b\rangle\} \mid_{\sigma_1} \{\langle a\rangle\}\ )$$
$$\mathbf{f}_{(\sigma)}\Big(\{\langle a\rangle\}\Big) = \mathbf{Ð}_{\sigma_2}(\ \{\langle a,a,a,b,b\rangle\}\ \ )$$
$$\mathbf{f}_{(\sigma)}\Big(\{\langle a\rangle\}\Big) = \{\langle a\rangle\} = \mathbf{g}_{1(\sigma)}\Big(\{\langle a\rangle\}\Big)$$
similarly, $\mathbf{f}_{(\sigma)}\Big(\{\langle b\rangle\}\Big) = \{\langle b\rangle\} = \mathbf{g}_{1(\sigma)}\Big(\{\langle b\rangle\}\Big)$

b) $\mathbf{f}_{(\omega)}(\mathbf{f}_{(\sigma)}) = \Big(\mathbf{f}\big[\mathbf{f}\big]_{\omega}\Big)_{(\sigma)} = \mathbf{Ð}_{\omega_2}(\mathbf{f} \mid_{\omega_1} \mathbf{f}\ )_{(\sigma)}$
$= \mathbf{Ð}_{\omega_2}(\langle a,a,a,b,b\rangle,\langle b,b,a,a,b\rangle \mid_{\omega_1} \langle a,a,a,b,b\rangle,\langle b,b,a,a,b\rangle\ )_{(\sigma)}$
$= \mathbf{Ð}_{\omega_2}(\ \langle a,a,a,b,b\rangle,\langle b,b,a,a,b\rangle\ )_{(\sigma)}$
$= (\ \langle a,a,b,b,a\rangle,\langle b,a,a,b,b\rangle\ )_{(\sigma)}$
$= (\ \langle a,a\rangle,\langle b,a\rangle\ )_{(\sigma)}$
$= \mathbf{g}_{2(\sigma)},$

c) $\big(\mathbf{f}_{(\omega)}(\mathbf{f}_{(\omega)})\big)(\mathbf{f}_{(\sigma)}) = \mathbf{Ð}_{\omega_2}(\ \langle a,a,a,b,b\rangle,\langle b,b,a,a,b\rangle\ )_{(\sigma)}$
$= (\ \langle a,b,b,a,a\rangle,\langle b,a,b,b,a\rangle\ )_{(\sigma)}$
$= (\ \langle a,b\rangle,\langle b,a\rangle\ )_{(\sigma)}$
$= \mathbf{g}_{3(\sigma)},$

d) $\Big(\mathbf{f}_{(\omega)}\big(\mathbf{f}_{(\omega)}(\mathbf{f}_{(\omega)})\big)\Big)(\mathbf{f}_{(\sigma)}) = \mathbf{Ð}_{\omega_2}(\ \langle a,a,a,b,b\rangle,\langle b,b,a,a,b\rangle\ )_{(\sigma)}$
$= (\ \langle a,b,a,a,b\rangle,\langle b,b,b,a,a\rangle\ )_{(\sigma)}$
$= (\ \langle a,b\rangle,\langle b,b\rangle\ )_{(\sigma)}$
$= \mathbf{g}_{4(\sigma)}.$

Appendix C. Supplement Material

Given that $\mathbf{R}\big[\mathbf{A}\big]_{\langle\sigma_1,\sigma_2\rangle} = \mathbf{Ð}_{\sigma_2}(\mathbf{R} \mid_{\sigma_1} \mathbf{A})$, the usual CST properties associated with the Image operation are preserved in XST.

Consequence C.1. *Preserved Image Properties:*

(a) $\mathbf{Q}\big[\mathbf{A}\cup\mathbf{B}\big]_{\sigma} = \mathbf{Q}\big[\mathbf{A}\big]_{\sigma}\cup\mathbf{Q}\big[\mathbf{B}\big]_{\sigma}$,
(b) $\mathbf{Q}\big[\mathbf{A}\cap\mathbf{B}\big]_{\sigma} \subseteq \mathbf{Q}\big[\mathbf{A}\big]_{\sigma}\cap\mathbf{Q}\big[\mathbf{B}\big]_{\sigma}$,
(c) $\mathbf{Q}\big[\mathbf{A}\big]_{\sigma} \sim \mathbf{Q}\big[\mathbf{B}\big]_{\sigma} \subseteq \mathbf{Q}\big[\mathbf{A}\sim\mathbf{B}\big]_{\sigma}$,
(d) $\mathbf{A}\subseteq\mathbf{B} \to \mathbf{Q}\big[\mathbf{A}\big]_{\sigma} \subseteq \mathbf{Q}\big[\mathbf{B}\big]_{\sigma}$,
(e) $\mathbf{Q}\big[\mathbf{Ð}_{\sigma}(\mathbf{Q})\cap\mathbf{A}\big]_{\langle\sigma,\gamma\rangle} = \mathbf{Q}\big[\mathbf{A}\big]_{\langle\sigma,\gamma\rangle}$,
(f) $\mathbf{Q}\big[\mathbf{A}\big]_{\langle\sigma,\gamma\rangle} = \mathbf{Ð}_{\gamma}(\mathbf{Q} \mid_{\sigma} \mathbf{A})$.
(g) $\mathbf{Q}\big[\varnothing\big]_{\sigma} = \varnothing, \quad \varnothing\big[\mathbf{A}\big]_{\sigma} = \varnothing, \quad \mathbf{Q}\big[\mathbf{A}\big]_{\varnothing} = \varnothing$,
(h) $\mathbf{Ð}_{\sigma}(\mathbf{Q})\cap\mathbf{A} = \varnothing \to \mathbf{Q}\big[\mathbf{A}\big]_{\langle\sigma,\gamma\rangle} = \varnothing$,
(i) $(\mathbf{Q}\cup\mathbf{R})\big[\mathbf{A}\big]_{\sigma} = \mathbf{Q}\big[\mathbf{A}\big]_{\sigma} \cup \mathbf{R}\big[\mathbf{A}\big]_{\sigma}$,
(j) $(\mathbf{Q}\cap\mathbf{R})\big[\mathbf{A}\big]_{\sigma} \subseteq \mathbf{Q}\big[\mathbf{A}\big]_{\sigma}\cap\mathbf{R}\big[\mathbf{A}\big]_{\sigma}$,
(k) $\mathbf{Q}\big[\mathbf{A}\big]_{\sigma} \sim \mathbf{R}\big[\mathbf{A}\big]_{\sigma} \subseteq (\mathbf{Q}\sim\mathbf{R})\big[\mathbf{A}\big]_{\sigma}$
(l) $\mathbf{Q} \subseteq \mathbf{R} \to \mathbf{Q}\big[\mathbf{A}\big]_{\sigma} \subseteq \mathbf{R}\big[\mathbf{A}\big]_{\sigma}$.

Appendix D. Basic Process & Function Spaces

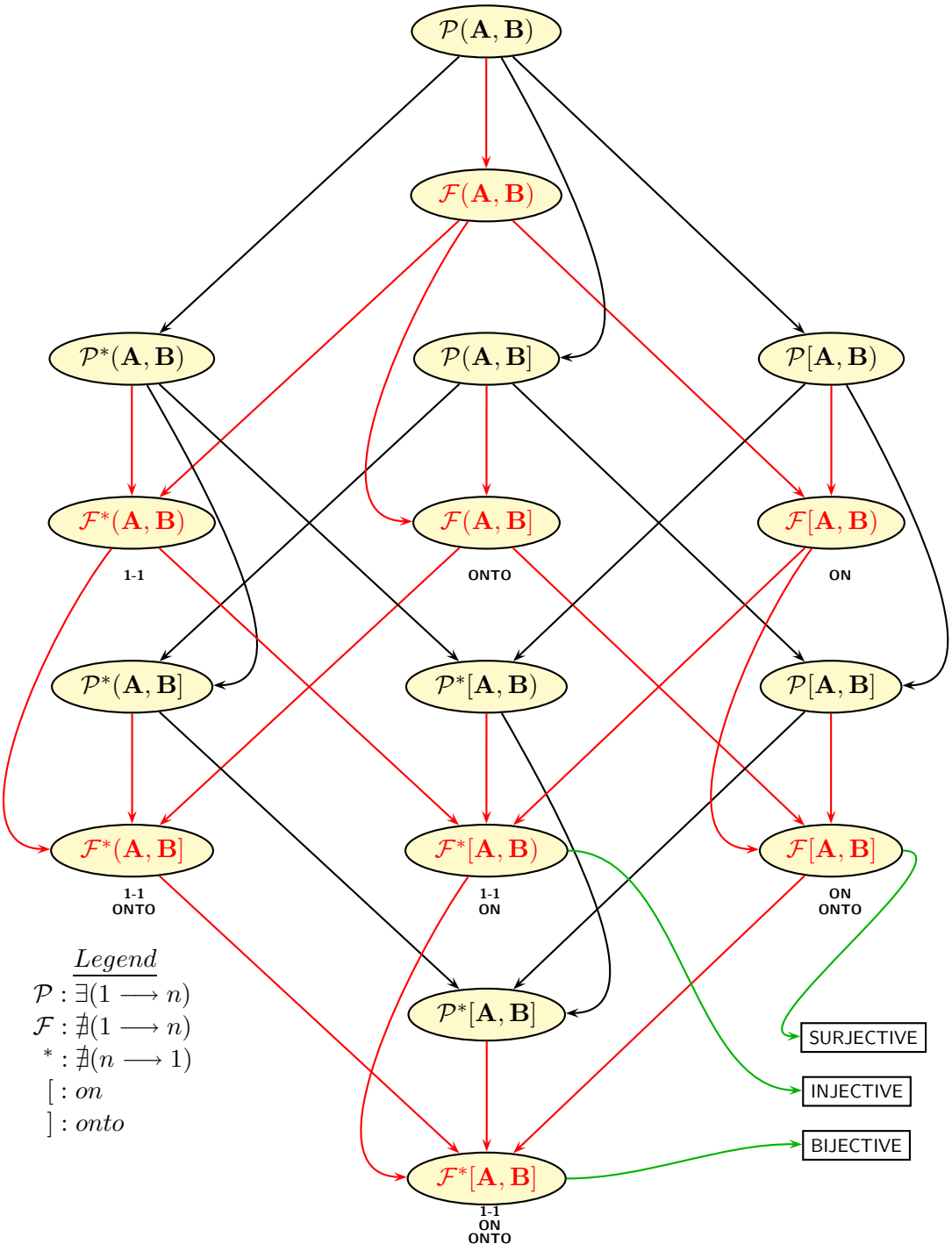
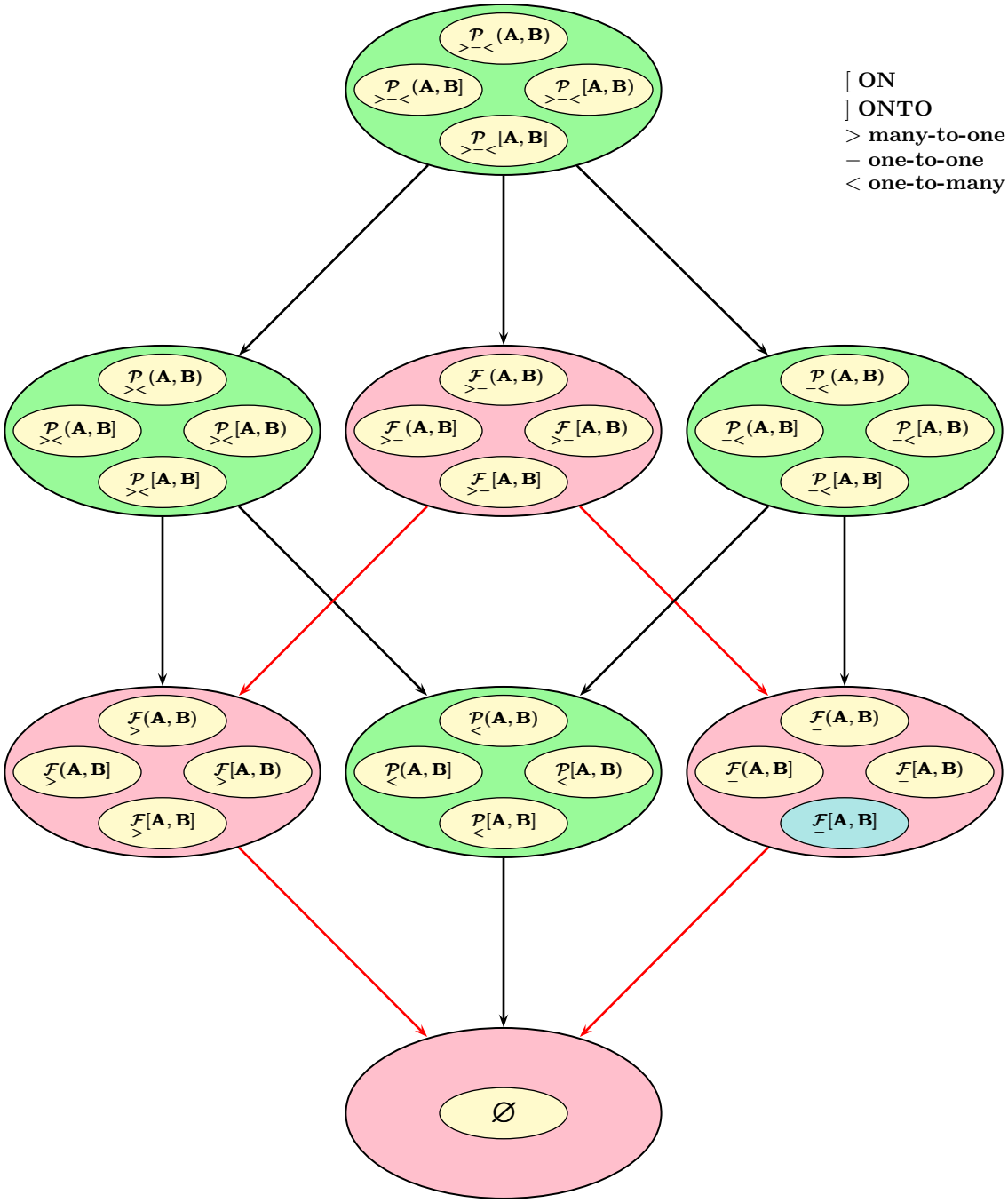


FIGURE 1 : *Process & Function Spaces* (16 *of* 29)

The above represents the lattice of process spaces with basic refinments imposed. For a detailed graphic of the 29 refinements of process spaces, see Appendix E.

Appendix E. Totally Refined Process Spaces



Refined Spaces: Process (29), Non-Empty Function (12).