# Mathematical Structuralism, S06

Amir Tabatabai

January 8, 2021

## 1 Category Theory (continued)

### 1.1 Functors and Natural Transformations

**Example 1.1.** (*Exponentiation*) Let $A$ be a fixed set. Define the assignment $(-)^A : \mathbf{Set} \to \mathbf{Set}$, mapping a set $B$ to $B^A = \{f : A \to B\}$ and a function $f : B \to C$ to a function $f^A : B^A \to C^A$ defined by $f^A(g) = fg$. Then, $(-)^A$ is a functor, generalizing the finite power functor $A \mapsto A^n$ generated by the iteration of the product functor. Similarly, it is possible to define the functor $A^{(-)} : \mathbf{Set}^{op} \to \mathbf{Set}$, mapping a set $B$ to $A^B = \{f : B \to A\}$ and a map $f : B \to C$ to a function $A^f : A^C \to A^B$ defined by $A^f(g) = gf$. Then, $A^{(-)}$ is a functor, generalizing the functor $\bar{P} = 2^{(-)}$. Any combination of the product, the sum, and the functor $(-)^A$, for different fixed sets $A$ such as $F(X) = A_0 \times X^{N_0} + A_1 \times X^{N_1} + \cdots + A_k \times X^{N_k}$ is a polynomial functor. The notion of polynomial functor, though, is more general than this.

**Remark 1.2.** *(Algebras)* Algebras are sets equipped with some operations that have some properties. For instance, a monoid is a set $M$ with an element $e$ and a binary operation such that the latter is associative and the former is the identity element for the latter. The operational data (not the properties) can be stored in one function $a : F_m(M) \to M$, where $F_m(X) = 1 + X^2$ is a functor, storing the type of the algebra and $a(0, *) = e$ and $a(1, m, n) = mn$, storing the operations. By type we mean the number and the arity of the operations (in the monoid case it is one nullary and one binary operations). Some examples may be helpful here. A group $(G, e, (-)^{-1}, \cdot)$ is a set $G$ with a function $a : F_g(G) \to G$, where $F_g(X) = 1 + X + X^2$, $a(0, *) = e$, $a(1, m) = m^{-1}$ and $a(2, m, n) = mn$; the basic structure of natural numbers, i.e., $(\mathbb{N}, s, 0)$ is a function $a : F_i(\mathbb{N}) \to \mathbb{N}$, where $F_i(X) = 1 + X$, $a(0, *) = 0$ and $a(1, n) = s(n) = n + 1$ and the structure $(\mathbb{W}, s_0, s_1, \epsilon)$ of binary strings can be described by a function $a : F_s(\mathbb{W}) \to \mathbb{W}$, where $F_s(X) = 1 + X + X$,

$a(0, *) = \epsilon$, $a(1, w) = s_0(w) = w0$ and $a(2, w) = s_1(w) = w1$. To have a general notion of algebra, we use a functor $F : \mathbf{Set} \to \mathbf{Set}$ to formalize the type of the algebra and then by an $F$-algebra, (an algebra of type $F$), we mean a function $a : F(A) \to A$. This also suggest a generalization for homomorphisms. Generally, a homomorphism is a function that preserves all the operations in the type of the algebra. With our generalization here, an $F$-algebra homomorphism from the $F$-algebra $a_A : F(A) \to A$ to the $F$-algebra $a_B : F(B) \to B$ is a function $f : A \to B$ such that

$$\begin{array}{ccc} F(A) & \xrightarrow{a_A} & A \\ {\scriptstyle F(f)}\downarrow & & \downarrow{\scriptstyle f} \\ F(B) & \xrightarrow[a_B]{} & B \end{array}$$

It is easy to check that in the familiar cases it really captures the notion of homomorphism.

**Example 1.3.** (*Forgetful Functors*) Sometimes, we have a category and we will forget some of the structures that the objects posses and the maps preserve, to think somewhat loosely about the same data that we originally had. Let us provide three examples of such phenomenon. First, the forgetful assignment mapping any group $G$ and any homomorphism $f : G \to H$ in $\mathbf{Grp}$ to themselves in $\mathbf{Set}$, forgetting that there is the group structure there, is a functor. For the second example, take the two forgetful functors from $\mathbf{Set}^{\to}$ to $\mathbf{Set}$, forgetting that a variable set actually varies, by making two snapshots of a variable set in the two possible moments. More precisely, for any $i \in \{0, 1\}$, define $p_i : \mathbf{Set}^{\to} \to \mathbf{Set}$, by mapping any $f : A_0 \to A_1$ to $A_i$ and any $\alpha : f \to g$ to $\alpha_i : A_i \to B_i$, where $f : A_0 \to A_1$ and $g : B_0 \to B_1$. Both $p_0$ and $p_1$ are functors. Finally, as the third example, define $V : \mathbf{Quiv} \to \mathbf{Set}$, by mapping any quiver to its set of elements and any quiver morphism to its underlying function on vertices. This $V$ is a functor. We can do the same thing to define the edge functor $E$.

**Example 1.4.** *(Free Functors)* In some cases, we want to put a structure on an object in a free way, meaning we want it to be free from any unexpected relations. For instance, let $X$ be a set. Then, $F(X)$ as the set of all finite sequences of the elements of $X$ (including the empty sequence) with concatenation is a free-monoid constructed from $X$. It is a monoid, since concatenation is associative and the empty sequence is an identity. It is free because we add all possible products of the elements of $X$, and there is no non-trivial relation on the elements of $F(X)$, except what the monoid structure dictates. This assignment $F$ gives rise to a functor $\mathbf{Set} \to \mathbf{Mon}$, mapping any set $X$ to the monoid $F(X)$ and any map $f : X \to Y$ to the homomorphisms $F(f) : F(X) \to F(Y)$ such that $F(f)(\sigma) = f(\sigma_0) \cdots f(\sigma_n)$, for any finite sequence $\sigma = \sigma_0\sigma_1 \cdots \sigma_n$.

**Example 1.5.** Let $\mathcal{C}$ be a category. Then, the identity functor $id_{\mathcal{C}} : \mathcal{C} \to \mathcal{C}$ mapping any object and morphism to itself is a functor. Moreover, if $A$ is a fixed object in $\mathcal{C}$, the constant assignment $c_A : \mathcal{C} \to \mathcal{C}$, mapping all objects to $A$ and all morphisms to identity is another functor.

**Example 1.6.** Let $\mathcal{C}$ be a groupoid. Then, the inverse assignment $inv : \mathcal{C} \to \mathcal{C}^{op}$, defined by $inv(A) = A$ and $inv(f) : B \to A$ as $inv(f) = f^{-1}$, for $f : A \to B$, is a functor.

**Example 1.7.** Let $\mathcal{C}$ be a locally small category. The assignment $Hom_{\mathcal{C}} : \mathcal{C}^{op} \times \mathcal{C} \to \mathbf{Set}$, defined by $Hom_{\mathcal{C}}(A, B) = \{f : A \to B \mid f \in Mor(\mathcal{C})\}$ and $Hom_{\mathcal{C}}(g, h) : Hom_{\mathcal{C}}(A, B) \to Hom_{\mathcal{C}}(C, D)$ as $Hom_{\mathcal{C}}(g, h)(f) = hfg$, for any $f : A \to B$, $g : C \to A$ and $h : B \to D$, is a functor. This functor captures the whole structure of the category $\mathcal{C}$.

**Example 1.8.** Let $\mathcal{C}$ be a locally small category. For any object $A$ in $\mathcal{C}$, there is a canonical functor $Hom_{\mathcal{C}}(A, -) : \mathcal{C} \to \mathbf{Set}$, capturing the behavior of the maps above $A$. It is defined by $B \mapsto Hom_{\mathcal{C}}(A, B)$ and $Hom(A, f) : Hom_{\mathcal{C}}(A, B) \to Hom_{\mathcal{C}}(A, C)$ as $Hom_{\mathcal{C}}(A, f)(g) = fg$, for any $f : B \to C$. Similarly, there is a canonical functor $y_A = Hom_{\mathcal{C}}(-, A) : \mathcal{C}^{op} \to \mathbf{Set}$, capturing the behavior of the maps below $A$. It is defined by $y_A(B) = Hom_{\mathcal{C}}(B, A)$ and $y_A(f) : Hom_{\mathcal{C}}(C, A) \to Hom_{\mathcal{C}}(B, A)$ as $y_A(f)(g) = gf$, for any $f : B \to C$. These functors are the localized version of the concrete representation we have introduced for the small categories, mapping an object $A$ to $A_* = \{g : C \to A \mid g \in Mor(\mathcal{C})$ and $f : A \to B$ to $f_* : A_* \to B_*$ by $f_*(g) = fg$. The current act of localization has no point except to handle the size issue that in a locally small category the collection $A_*$ is not necessarily a set.

**Example 1.9.** Let $\mathcal{C}$ be a category and $f : A \to B$ be a morphism. The assignment mapping an object $g : X \to A$ in $\mathcal{C}/A$ to the object $fg : X \to B$ in $\mathcal{C}/B$ and mapping to themselves is a functor from $\mathcal{C}/A$ to $\mathcal{C}/B$. We denote this functor by $f_* : \mathcal{C}/A \to \mathcal{C}/B$.

**Example 1.10.** Let $\mathcal{C}$, $\mathcal{D}$ and $\mathcal{E}$ be some categories and $F : \mathcal{D} \to \mathcal{E}$ and $G : \mathcal{C} \to \mathcal{D}$ be two functors. Then, the composition $FG : \mathcal{C} \to \mathcal{E}$ with the canonical definition is also a functor.

Note that all small categories with functors as morphisms constitute a category. We denote this category by **Cat**.

**Example 1.11.** Let $\mathcal{C}$ be a small category. Then, the assignment mapping an object $A$ to the category $\mathcal{C}/A$ and morphism $f : A \to B$ to the functor $f_* : C/A \to C/B$ is a functor from $\mathcal{C}$ to **Cat**.

**Example 1.12.** (*Baby Schemes*) Let $\mathcal{R}$ be the category of all subsets $R$ of $\mathbb{C}$, including 1 and closed under addition and multiplication with morphisms as the functions that preserve the element 1 and these two operations. Let $I(\vec{x}) = I(x_0, \ldots, x_n)$ be a set of equations between polynomials in variables $x_0 \ldots, x_n$ with coefficients in $\mathbb{Z}$. For instance, we can take $I(x_0, x_1) = \{x_0^2 + x_1^2 = 1\}$. Define the assignment $V_I : \mathcal{R} \to \mathbf{Set}$ by mapping $R$ to $V_I(R) = \{\vec{r} \in R^{n+1} \mid \text{all equations in } I(\vec{x}) \text{ hold for } \vec{x} = \vec{r}\}$ and any $f : R \to S$ to the function $V_I(f) : V_I(R) \to V_I(S)$ defined by $V_I(f)(\vec{x}) = (f(x_0), \ldots, f(x_n))$. The function $V_I(f)$ is well-defined, because when $\vec{r}$ is the root for an equation, then so is $f(\vec{r})$, simply because $f$ preserves 1, addition and multiplication. This assignment is clearly a functor. It is reasonable to think of $V_I$ as a method to keep track of all the possible realizations (models) of the set of equations in all possible worlds. It is the semantical way to capture the syntactic data $I(\vec{x})$.

**Remark 1.13.** Note that $V_I$ is not a faithful semantical apparatus. For instance, for the different sets of equations $I(x) = \{x = 0\}$ and $J(x) = \{x^2 = 0\}$, we have $V_I(R) = V_J(R) = \{0\}$.