Algebraic Effects and Effect Handlers

Takahiro Sanada

RIMS, Kyoto University, https://www.kurims.kyoto-u.ac.jp/~tsanada/, tsanada@kurims.kyoto-u.ac.jp,

Introduction

The mathematical treatment of a program is important to verify that it does not contain bugs or to prove some good properties. Although a program takes arguments and returns the result of the computation, it cannot simply be regarded as a mathematical function. This is because, in the process of computation, a program may perform I/O, read or write memory, throw errors, or loop indefinitely. Such behavior is called computational effects. Monads and algebras are known as a concept to capture computational effects.
An algebra consists of a carrier set, operations and equations. For example, the set $\mathbb{N}$ of natural numbers has an algebraic structure $(\mathbb{N}, +^{\mathbb{N}}, 0^{\mathbb{N}})$ determined by operations $+$ of arity 2 and 0 of arity 0, and equations $x + y = y + x$ and $0 + x = x$. The set of programs which may perform computational effects can be seen as an algebra. For example, consider the set P of programs which can read a memory cell of type of boolean value. Consider the operation on P is $\underline{\text{get}}\colon P^2 \to P$. The program $r = \underline{\text{get}}(p, q) \in P$ for $p, q \in P$ means that r reads the memory cell and executes the program p if its value is true, otherwise it executes the program q. The equations on P are $\underline{\text{get}}(\underline{\text{get}}(p, q), \underline{\text{get}}(r, s)) = \underline{\text{get}}(p, s)$ and $\underline{\text{get}}(p, p) = p$, which means that the value in the memory cell does not change during the execution of the program.

	Natural numbers	Programs
Carrier set	$\mathbb{N}$	P
Operation(s)	$+^{\mathbb{N}}\colon \mathbb{N}^2 \to \mathbb{N}$ $0^{\mathbb{N}}\colon \mathbb{N}^0 \to \mathbb{N}$	$\underline{\text{get}}^P\colon P^2 \to P$
Equation(s)	$x +^{\mathbb{N}} y = y +^{\mathbb{N}} x$ $0 +^{\mathbb{N}} x = x$	$\underline{\text{get}}^P(p, p) = p$ $\underline{\text{get}}^P(\underline{\text{get}}^P(p, q), \underline{\text{get}}^P(r, s))$ $= \underline{\text{get}}^P(p, s)$

The viewpoint of programs as algebra allows us to discuss various properties about programs in terms of category theory. Error handlers can be interpreted as homomorphisms between algebras, and the notion of handler can be defined for other effects.

Algebraic Theory

A signature $\Sigma = (\Sigma, \mathrm{ar})$ consists of

- a set Σ of operation symbols and
- a function $\mathrm{ar}\colon \Sigma \to \mathbb{N}$, which assigns the arity $\mathrm{ar}(\underline{\text{op}})$ for each $\underline{\text{op}} \in \Sigma$.

For a signature Σ and a set X, the set of Σ-terms $\mathit{Term}_\Sigma(X)$ generated by X is defined as the smallest set such that

- $X \subseteq \mathit{Term}_\Sigma(X)$ and
- for any $\underline{\text{op}} \in \Sigma$ and $t_1, \dots, t_{\mathrm{ar}(\underline{\text{op}})} \in \mathit{Term}_\Sigma(X)$, $\underline{\text{op}}(t_1, \dots, t_{\mathrm{ar}(\underline{\text{op}})}) \in \mathit{Term}_\Sigma(X)$.

An equation is a pair (ℓ, r) of Σ-terms $\ell, r \in \mathit{Term}_\Sigma(V)$. We sometime write an equation (ℓ, r) as $V \vdash \ell = r$.

An algebraic theory $\mathfrak{T}$ is a pair $(\Sigma, \mathcal{E})$ of a signature Σ and a set of equations $\mathcal{E} = \{V_i \vdash \ell_i = r_i\}_{i \in I}$.

Example: Algebraic Theory of Commutative Monoid

An algebraic theory of commutative monoid is $\mathfrak{T}_{\text{cmon}} = (\Sigma, \mathcal{E})$ where

- $\Sigma = \{+, e\}$, $\mathrm{ar}(+) = 2$, $\mathrm{ar}(e) = 0$ and
- $\mathcal{E} = \left\{ \begin{array}{l} \{x, y\} \vdash x + y = y + x, \\ \{x, y, z\} \vdash (x + y) + z = x + (y + z), \\ \{x\} \vdash e + x = x \end{array} \right\}.$

Example: Algebraic Theory of Read Only State

An algebraic theory of read only state is $\mathfrak{T}_{\text{read}} = (\Sigma, \mathcal{E})$ where

- $\Sigma = \{\underline{\text{get}}\}$, $\mathrm{ar}(\underline{\text{get}}) = 2$ and
- $\mathcal{E} = \left\{ \begin{array}{l} \{x, y, z, w\} \vdash \underline{\text{get}}(\underline{\text{get}}(x, y), \underline{\text{get}}(z, w)) = \underline{\text{get}}(x, w) \\ \{x\} \vdash \underline{\text{get}}(x, x) = x \end{array} \right\}$

Model of Algebraic Theory

Let $\mathfrak{T} = (\Sigma, \mathcal{E})$ be an algebraic theory. A $\mathfrak{T}$-model $\mathcal{A}$ consists of

- a carrier set A, and
- for each $\underline{\text{op}} \in \Sigma$, its interpretation $\underline{\text{op}}^{\mathcal{A}}\colon A^{\mathrm{ar}(\underline{\text{op}})} \to A$

satisfying all equations in $\mathcal{E}$, that is

- for each $(V \vdash \ell = r) \in \mathcal{E}$ and $s\colon V \to A$, $\bar{s}(\ell) = \bar{s}(r)$ holds

where for a function $s\colon V \to A$ and a term $m \in \mathit{Term}_\Sigma(V)$, $\bar{s}(m) \in A$ is defined as follows:

- $\bar{s}(v) = s(v)$
- $\bar{s}(\underline{\text{op}}(m_1, \dots, m_n)) = \underline{\text{op}}^{\mathcal{A}}(\bar{s}(m_1), \dots, \bar{s}(m_n))$

Example: Models of Theory of Commutative Monoid

The following triples are $\mathfrak{T}_{\text{cmon}}$-models.

- $(\mathbb{N}, +^{\mathbb{N}}, 0)$
- $(\mathbb{N}, \times^{\mathbb{N}}, 1)$
- $(\mathcal{P}X, \cup, \varnothing)$ for a set X where $\mathcal{P}X$ is the power set of X.

Homomorphism and Category of Models

Let $\mathfrak{T}$ be $(\Sigma, \mathcal{E})$ and $\mathcal{A}, \mathcal{A}'$ be $\mathfrak{T}$-models. A homomorphism $\phi\colon \mathcal{A} \to \mathcal{A}'$ is a function $\phi\colon A \to A'$ between their carrier sets such that for each $\underline{\text{op}} \in \Sigma$, $\phi(\underline{\text{op}}^{\mathcal{A}}(a_1, \dots, a_n)) = \underline{\text{op}}^{\mathcal{A}'}(\phi(a_1), \dots, \phi(a_n))$ holds where $n = \mathrm{ar}(\underline{\text{op}})$, that is the following diagram commutes:

$$\begin{tikzcd} A^n \arrow[r, "\phi^n"] \arrow[d, "\underline{\text{op}}^{\mathcal{A}}"'] & A'^n \arrow[d, "\underline{\text{op}}^{\mathcal{A}'}"] \\ A \arrow[r, "\phi"] & A' \end{tikzcd} \tag{1}$$

We can check that all models of $\mathfrak{T}$ and homomorphisms forms a category. We write the category of models as $\mathrm{Mod}(\mathfrak{T})$.

Free Model

Given a theory $\mathfrak{T} = (\Sigma, \mathcal{E})$ and a set X, the free $\mathfrak{T}$-model generated by X is $\mathit{Term}_\Sigma(X)/{\sim_{\mathcal{E}}}$ where $\sim_{\mathcal{E}}$ is the equivalence relation on $\mathit{Term}_\Sigma(X)$ generated by the following rules:

- for each equation $(V \vdash \ell = r) \in \mathcal{E}$ and substitution $s\colon V \to \mathit{Term}_\Sigma(X)$, $\bar{s}(\ell) \sim_{\mathcal{E}} \bar{s}(r)$, and
- $\underline{\text{op}}(t_1, \dots, t_n) \sim_{\mathcal{E}} \underline{\text{op}}(t'_1, \dots, t'_n)$ if $t_k \sim_{\mathcal{E}} t'_k$ for every $1 \le k \le n$.

We write the free model $\mathit{Term}_\Sigma(X)/{\sim_{\mathcal{E}}}$ as $\mathcal{F}X$. We can make $\mathcal{F}$ a functor $\mathrm{Set} \to \mathrm{Mod}(\mathfrak{T})$. The forgetful functor $\mathcal{U}\colon \mathrm{Mod}(\mathfrak{T}) \to \mathrm{Set}$ sends models to its carrier sets. Then there is an adjunction $\mathcal{F} \dashv \mathcal{U}$:

$$\begin{tikzcd} \mathrm{Set} \arrow[r, bend left, "\mathcal{F}"] \arrow[r, phantom, "\perp"] & \mathrm{Mod}(\mathfrak{T}). \arrow[l, bend left, "\mathcal{U}"] \end{tikzcd}$$

The unit of this adjunction is $\eta\colon X \to \mathcal{U}\mathcal{F}X$. The universal property of the free algebra is described as follows:

$$\begin{tikzcd} X \arrow[r, "\eta"] \arrow[dr, "f"'] & \mathcal{U}\mathcal{F}X \arrow[d, dashed, "\mathcal{U}\phi"] & \mathcal{F}X \arrow[d, dashed, "\phi"] \\ & \mathcal{U}\mathcal{A} & \mathcal{A} \end{tikzcd}$$

Programming Language

Syntax

Type	A, B	$::=$	$0 \mid \texttt{bool} \mid \texttt{int} \mid A \to B$
Value	V, W	$::=$	$x \mid \mathrm{true} \mid \mathrm{false} \mid n \mid \lambda x.M$
Computation	M, N	$::=$	$\mathsf{ret}\ V \mid \mathsf{let}\ x \leftarrow N\ \mathsf{in}\ M \mid \underline{\text{op}}$
Environment	Γ	$::=$	$x_1 : A_1, \dots, x_n : A_n$

Typing

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A} \quad \frac{c : \tau}{\Gamma \vdash c : \tau} \quad \frac{\Gamma, x : A \vdash^{\mathrm{c}} M : B}{\Gamma \vdash \lambda x.M : A \to B}$$

$$\frac{\Gamma \vdash V : A}{\Gamma \vdash^{\mathrm{c}} \mathsf{ret}\ V : A} \quad \frac{\Gamma \vdash^{\mathrm{c}} N : A \quad \Gamma, x : A \vdash^{\mathrm{c}} M : B}{\Gamma \vdash^{\mathrm{c}} \mathsf{let}\ x \leftarrow N\ \mathsf{in}\ M : B}$$

$$\frac{\underline{\text{op}} \in \Sigma \quad \mathrm{ar}(\underline{\text{op}}) = A}{\Gamma \vdash^{\mathrm{c}} \underline{\text{op}} : A}$$

Semantics of the Language

Let $\mathcal{T}$ be the monad $\mathcal{U} \circ \mathcal{F}$. We write the unit, multiplication, and strength of $\mathcal{T}$ as $\eta_X\colon X \to \mathcal{T}X$, $\mu_X\colon \mathcal{T}\mathcal{T}X \to \mathcal{T}X$, and $\mathrm{st}_{A,B}\colon A \times \mathcal{T}B \to \mathcal{T}(A \times B)$, respectively.

Type

$$[\![\texttt{bool}]\!] = \{\mathsf{t}, \mathsf{f}\}, \quad [\![\texttt{int}]\!] = \mathbb{N},$$
$$[\![A \times B]\!] = [\![A]\!] \times [\![B]\!], \quad [\![A \to B]\!] = (\mathcal{T}[\![B]\!])^{[\![A]\!]}$$

Environment $[\![\Gamma]\!] = [\![A_1]\!] \times \cdots \times [\![A_n]\!]$

Value $[\![\Gamma \vdash V : A]\!]\colon [\![\Gamma]\!] \to [\![A]\!]$

$$[\![\mathrm{true}]\!] = \mathsf{t}, \quad [\![\mathrm{false}]\!] = \mathsf{f}, \quad [\![n]\!] = n$$
$$[\![\Gamma \vdash x : A]\!] = \pi_x, \quad [\![\Gamma \vdash \lambda x.M : A \to B]\!] = \Lambda[\![M]\!]$$

Computation $[\![\Gamma \vdash^{\mathrm{c}} M : A]\!]\colon [\![\Gamma]\!] \to \mathcal{T}[\![A]\!]$

$$[\![\Gamma \vdash^{\mathrm{c}} \mathsf{ret}\ V : A]\!] = \eta_{[\![A]\!]} \circ [\![V]\!]$$
$$[\![\Gamma \vdash^{\mathrm{c}} \mathsf{let}\ x \leftarrow N\ \mathsf{in}\ M : B]\!] = \mu_{[\![B]\!]} \circ \mathcal{T}[\![M]\!] \circ \mathrm{st}_{[\![\Gamma]\!],[\![A]\!]} \circ \langle \mathrm{id}, [\![N]\!] \rangle$$
$$[\![\Gamma \vdash^{\mathrm{c}} \underline{\text{op}} : A]\!] = \underline{\text{op}}(\lambda a.a)$$

Error Handler

Consider the signature $\Sigma_{\mathrm{e}} = \{\underline{\mathrm{e}}\}$ where the arity of $\underline{\mathrm{e}}$ is 0. The monad $\mathcal{T}_{\mathrm{e}}$ associated the theory $\mathfrak{T}_{\mathrm{e}} = (\Sigma_{\mathrm{e}}, \varnothing)$ is $\mathcal{T}_{\mathrm{e}}X = X \sqcup \{\perp\}$.
For example the following program throw an error before executing M. Hence, M is not executed.

$$\mathsf{let}\ x \leftarrow \underline{\mathrm{e}}\ \mathsf{in}\ M$$

Error handler provide a way to recover errors. For example, the following program in the handler throw an error, and it is caught by the handler. Then, the program N is executed to recover the error.

$$\mathsf{handle}\ (\mathsf{let}\ x \leftarrow \underline{\mathrm{e}}\ \mathsf{in}\ M)\ \mathsf{with}\ \{\underline{\mathrm{e}} \mapsto N\}$$

From the algebraic point of view, the handler defines a structure of $\mathfrak{T}_{\mathrm{e}}$-model $\mathcal{A}$ in some set A. Therefore, by the universality of the free $\mathfrak{T}_{\mathrm{e}}$-model, there exists a unique homomorphism $\phi\colon \mathcal{F}_{\mathrm{e}}A \to \mathcal{A}$ which makes the following diagram commutes:

$$\begin{tikzcd} A \arrow[r, "\eta_A"] \arrow[dr, "\mathrm{id}_A"'] & \mathcal{T}_{\mathrm{e}}A \arrow[d, dashed, "\mathcal{U}\phi"] & \mathcal{F}_{\mathrm{e}}A \arrow[d, dashed, "\phi"] \\ & A = \mathcal{U}\mathcal{A} & \mathcal{A}. \end{tikzcd}$$

The morphism $\mathcal{U}\phi$ can be regarded as the interpretation of the error handler:

$$[\![\mathsf{handle}\ M\ \mathsf{with}\ \{\underline{\mathrm{e}} \mapsto N\}]\!]s = (\mathcal{U}\phi)([\![M]\!]s)$$

where $s \in [\![\Gamma]\!]$.

Effect Handler

Effect handler is generalisation of error handler. Similar to error handlers, the interpretation of effect handlers is given by the homomorphism obtained by the universality of free algebra. The general form of effect handler is as follows:

$$\begin{array}{l} \mathsf{handle}\ M\ \mathsf{with}\{ \\ \quad \mathsf{ret}\ x \mapsto L \\ \} \cup \{ \\ \quad \underline{\text{op}}, k \mapsto N_{\text{op}} \\ \}_{\text{op} \in \Sigma} \end{array}$$

The term L determines the morphism $A \to A'$, and The set of terms $\{N_{\text{op}}\}_{\text{op} \in \Sigma}$ determines an algebraic structure $\mathcal{A}'$ of $\mathfrak{T} = (\Sigma, \varnothing)$ with a carrier set A'.

$$\begin{tikzcd} A \arrow[r, "\eta_A"] \arrow[dr, "{[\![L]\!]}"'] & \mathcal{T}A \arrow[d, dashed, "\mathcal{U}\phi"] & \mathcal{F}A \arrow[d, dashed, "\phi"] \\ & A' = \mathcal{U}\mathcal{A}' & \mathcal{A}'. \end{tikzcd}$$

The morphism $\mathcal{U}\phi$ obtained by the universality of free $\mathfrak{T}$-model $\mathcal{F}A$ is the interpretation of the handler.

$$\begin{aligned} &[\![\mathsf{handle}\ M\ \mathsf{with}\{\mathsf{ret}\ x \mapsto L\} \cup \{\underline{\text{op}}(x), k \mapsto N_{\text{op}}\}_{\text{op} \in \Sigma}]\!] \\ &\quad = (\mathcal{U}\phi)([\![M]\!]) \end{aligned}$$

An example of effect handler for the theory of read only state is

$$\{x \mapsto \mathsf{ret}\ \lambda s.x\} \cup \{\underline{\text{get}}, k \mapsto \mathsf{ret}\ \lambda s.ks\}.$$

References:
1. Plotkin, G. and Power, J.: Notions of Computation Determine Monads. FoSSaCS 2002. *Foundations of Software Sciences and Computation Structures*, pp 342–356. 2002.
2. Bauer, A. and Pretner, M.: An effect system for algebraic effects and handlers. *Log. Methods Comput. Sci.* **10**(4), 2014.