Categorical Unification

M.Ángeles Galán García
magalan@cs.umu.se

Dpt. of Computing Science
Umeå University
Sweden

Overview

• Use of categorical methods in logic programming
• Generalised terms using compositions of monads
• Techniques for monad compositions
• Visual Representations
• Similarities between powersets of terms

Some Relations

Kleisli Category

Categories

Nat. Transf.

Monad

Functors

Motivations

"Knowledge advances by steps, and not by leaps"

(Thomas B. Macaulay)

Why?

• In classical logic, the unification algorithm can be specified in terms of category theory
• A substitution of variables is a mapping \(\sigma: X \to T_0 Y\)
• Given two substitutions \(\sigma_1: X \to T_0 Y, \sigma_2: Y \to T_0 Z\),
  its composition \(\sigma_2 \circ \sigma_1: X \to T_0 Z\)
  cannot be the composition of the mappings

Example

Consider the substitutions

\(\sigma_1 = [x/f(y,g(a,z))], y/b, z/h(y,c)]\)
\(\sigma_2 = [x/a, y/g(z,f(b,z)), z/(g(a,z),f(a,b))]\)

Just replacing variables in terms by term, we obtain the composition:

\(\sigma_1 \circ \sigma_2 = [x/a,y/g(\sigma_1(z),f(b,\sigma_1(z))),z/h(g(a,\sigma_1(z)),f(a,b))]\)
In general

Given two mappings \( \sigma_1 : X \rightarrow TY \) \( \sigma_2 : Y \rightarrow TZ \), we are seeking a composite one \( \sigma_1 \circ \sigma_2 : X \rightarrow TZ \).

A possibility is to consider the following chain of compositions:
\[
X \xrightarrow{\sigma_1} TY \xrightarrow{T \tau} TZ
\]

\( \tau, \mu \) come from the properties of \( T \), that is a monad.

Generalising

What would happen if what we want is to substitute a variable by a set of terms?
\[
[x/(t_1, t_3, t_6), y/(t_2, t_3)]
\]

In this case, a variable substitution would be
\[
\tau \mu : X \rightarrow P_o T \mu Y
\]

where \( P \) denotes the powerset functor. As in the previous case, this composition can be defined if \( P_o T \mu Y \) is a monad.

Monads

"For the things we have to learn before we can do them,
we learn by doing them"

(Aristotle)

Monad

A monad in a category \( C \) is a triple \( F = (F, \eta, \mu) \), where

\( F : C \rightarrow C \) is a (covariant) functor,
\( \eta : \text{id} \rightarrow F \) and \( \mu : F o F \rightarrow F \) are natural transformations such that:
\[
\mu x o F \mu x = \mu x o \mu x \quad \text{and} \quad \mu x o F \eta x = \mu x o \eta x = \text{id} x
\]

Kleisli Category

The Kleisli category, \( C_\Phi \), of a monad \( \Phi = (\Phi, \eta, \mu) \) in \( C \) is the following category \( C_\Phi \) over \( C \):

(i) \( \text{Ob}(C_\Phi) = \text{Ob}(C) \),
(ii) \( \text{hom}_{C_\Phi}(X, Y) = \text{hom}_C(X, \Phi Y) \),
(iii) \( (\eta C) C_\Phi = \eta C_\Phi \),
(iv) The composition of \( f : X \rightarrow \Phi Y \) with \( g : Y \rightarrow \Phi Z \) in \( C_\Phi \), is given by \( \Phi g o f \) (in \( C \))
The Composisions of Monads

"In union there is strength"  
(Aesop)

Monad Compositions

Consider the monads \( P = (\mathbb{P}, \eta, \mu) \) and \( T = (\mathbb{P}^*, \eta^*, \mu^*) \). To have a monad structure for the composition we need to define

\[ \eta^* : \text{id} \rightarrow PT \quad \text{and} \quad \mu^* : PT PT \rightarrow PT \]

In order to define \( \mu^* \) we will use \( \mu, \mu' \) and a particular natural transformation \( \sigma \), called swapper.

\[ \eta^* = \eta \circ \eta^* \quad \mu^* = \mu \circ \mu^* \circ P_{\tau TX} \]

\[ \sigma = \mu^* \circ \mu^* \circ P_{\tau TX} \]

Further, define \( \eta_X : X \rightarrow L_{id}X \) by \( \eta_X(x)(x') = \begin{cases} 1 & \text{if } x = x' \\ 0 & \text{otherwise} \end{cases} \)

The multiplication \( \mu_X : L_{id}L_{id}X \rightarrow L_{id}X \) is defined as

\[ \mu_X(A)(x) = \bigvee_{A' \in L_{id}A} A(x) \wedge A(A') \]

As a result we have that \( L_{id} = (L_{id}, \eta, \mu) \) is a monad

Covariant powerset functor \( L_{id} \):

\[ L_{id}X = \mathcal{P}X = \{ A : X \rightarrow \{0,1\} \} \]

and for a morphism \( f : X \rightarrow Y \) in \( \text{Set} \),

\[ L_{id}f(A)(y) = A(f^{-1}(y)) \]

It is easy to check that this functor generalises the direct image functor in the crisp case

The Term Functor

Let \( \Omega = \prod_{i=1}^n \Omega_i \) be an operator domain, where each \( \Omega_i \) is intended to contain operators of arity \( n \).

The definition of the term functor \( T_\Omega \text{Set} \rightarrow \text{Set} \) is given as \( T_\Omega(X) = \bigcup_{m_1, \ldots, m_n} T^n_\Omega(X) \), where

1. \( T^n_\Omega(X) = X \)
2. \( T^{n+1}_\Omega(X) = \{ (\omega, (m_1, \ldots, m_n)) \mid \omega \in \Omega, m_1, \ldots, m_n \in T^n_\Omega(X) \} \)

Notation \( (\omega, (m_1, \ldots, m_n)) \) for the term \( \omega(x_1, \ldots, x_n) \).
LT Composition

First problem: Define a multiplication \( LT \times LT \rightarrow LT \) (?)

Swapper \( \sigma_X : T^2LX \rightarrow LX \) defined as follows:

For the base case consider \( \sigma_X : 1LX = \text{id}_{LX} \).

For \( i = (m, \omega, (l_i)_{i \in \mathbb{N}}) \in T^2LX, n > 0, l_i \in T^2LX, \beta_i < \alpha, \) define

\[
\sigma_X(i)(m', \omega', (l_i)_{i \in \mathbb{N}}) = \begin{cases} 
\sigma_X(l_i)(m_i) & \text{if } m = m' \text{ and } \omega = \omega' \\
0 & \text{otherwise}
\end{cases}
\]

Note that in the case of \( \alpha > 0, \) for \( L = 2 \) we have

\[
\sigma_X(i) = \{(m, \omega, (l_i)_{i \in \mathbb{N}}) \mid m_i \in \sigma_X(l_i)\}
\]

LT Monad

Natural transformations \( \eta^{LT} : \text{id} \rightarrow LT \) and \( \mu^{LT} : LT \times LT \rightarrow LT \)

are defined as follows:

\[
\eta^{LT}(x) = \eta^{T}(x) \\
\mu^{LT}(R)(m) = \bigvee_{r \in LT} R(r) \land \sigma_{TX}(r)(m)
\]

Further, for \( R \in LT \times TX, \alpha > 0, \) and \( m \in TX, \) note that

\[
\mu^{LT}(R)(m) = \bigvee_{r \in LT \times TX} R(r) \land \sigma_{TX}(r)(m)
\]

With this definition \( (L \text{id} \circ T, \eta^{LT}, \mu^{LT}) \) is a monad

Monad Compositions: Theorem-1

Let \( \Phi : (\Phi_1, \Phi_2) \) and \( \Phi' : (\Phi'_1, \Phi'_2) \) be monads/multisets

\[
\Phi \circ \Phi' : (\Phi_1 \circ \Phi'_1, \Phi_2 \circ \Phi'_2)
\]

\[
\Phi' \circ \Phi : (\Phi'_1 \circ \Phi_1, \Phi'_2 \circ \Phi_2)
\]

\[
(\Phi \circ \Phi')_1 = \Phi_1 \circ \Phi'_1 \\
(\Phi \circ \Phi')_2 = \Phi_2 \circ \Phi'_2
\]

Then \( \Phi \circ \Phi' \) is a monad.

Monad Compositions: Theorem (Reverse)

We have a monad \( \Phi \circ \Phi' \).

\( \Phi \circ \Phi' \) is a monad if and only if for all \( x \in \Phi_1 \) and \( y \in \Phi_2 \),

\[
(\Phi_1 \circ \Phi'_1)_{\phi_1} \circ (\Phi_2 \circ \Phi'_2)_{\phi_2} = (\Phi_1 \circ \Phi'_1)_{\phi_1} \circ (\Phi_2 \circ \Phi'_2)_{\phi_2}
\]

Condition for \( (\Phi_1, \Phi'_1) \) and \( (\Phi_2, \Phi'_2) \) are satisfied.

In addition:

Condition for \( x(\phi_1) \) and \( y(\phi_2) \) if and only if

\[
(\Phi_1 \circ \Phi'_1)_{\phi_1} \circ (\Phi_2 \circ \Phi'_2)_{\phi_2} = (\Phi_1 \circ \Phi'_1)_{\phi_1} \circ (\Phi_2 \circ \Phi'_2)_{\phi_2}
\]

Visual Representations

“All things are difficult before they are easy”

(John Norley)
With equations...

\[
\begin{align*}
\sigma^* \circ \mu^* &= \psi^* \circ (\mu \circ \psi) \\
\sigma \circ \mu &= (\sigma \circ \psi) \circ (\mu \circ \psi) \\
\sigma \circ \mu &= (\sigma \circ \psi) \circ (\mu \circ \psi)
\end{align*}
\]

Visually...

\[
\begin{align*}
\sigma^* \circ \mu^* &= \\
\sigma \circ \mu &= (\sigma \circ \psi) \circ (\mu \circ \psi)
\end{align*}
\]

Visual Representations

\[
\begin{align*}
F \xrightarrow{\tau} G &= \\
F' \xrightarrow{1} F &= \\
F' \xrightarrow{\tau \cdot \tau} F' &=
\end{align*}
\]

Composition of Natural Transformations

\[
\begin{align*}
F' \xrightarrow{\tau} F &= \\
F' \xrightarrow{\eta} F &=
\end{align*}
\]

Interchange Law

\[
(F' \circ \tau \circ F) \circ (\sigma \circ \tau) = (\sigma' \circ \tau) \circ (\tau' \circ \tau)
\]

Monad (visually)

Let \((F, \eta, \mu)\) be a monad:

\[
\begin{align*}
\text{Associativity:} & \quad \mu \circ F \eta = \mu \circ \eta \\
\text{Identities:} & \quad \mu \circ F \eta = \mu \circ \eta
\end{align*}
\]
Summary

"The roots of education are bitter, but the fruit is sweet"
(Aristotle)

Collaboration

M.A. Galán

P. Eklund
Dept. Computer Science. Åbo Akademi University, Finland.

J. Medina, M. Ojeda-Aciego, A. Valverde
Dept. Matemática Aplicada. Universidad de Málaga, Spain.

Unification

Classical Logic — Term Monad

Coequalizer

Many-valued Logic — Generalised Terms

Fuzzy Unification

Publications
