Clarification on transitive verbs, binary relations and \( e(et) \) functions

FST, 20.11.2012

As we saw today, there is no well-accepted convention regarding the intuitive binary relations that transitive verbs should denote. We considered the denotation of the verb \textit{praise} in the following situation.

Mary praised Tina.
Mary praised John.
Tina praised Mary.
Mary praised herself.
Nobody else praised anybody else.

The following binary relation describes an intuitive denotation of the verb \textit{praise} in this situation:

\[ R_1 = \{ \langle \textit{mary}, \textit{tina} \rangle, \langle \textit{mary}, \textit{john} \rangle, \langle \textit{tina}, \textit{mary} \rangle, \langle \textit{mary}, \textit{mary} \rangle \} \]

In \( R_1 \) the first element in the ordered pairs corresponds to the subject of the verb \textit{praise}, and the second element corresponds to the object. However, we can think of a different relation as the denotation of the verb \textit{praise}:

\[ R_2 = \{ \langle \textit{tina}, \textit{mary} \rangle, \langle \textit{john}, \textit{mary} \rangle, \langle \textit{mary}, \textit{tina} \rangle, \langle \textit{mary}, \textit{mary} \rangle \} \]

In \( R_2 \) the first element in the ordered pairs corresponds to the object of the verb \textit{praise}, and the second element corresponds to the subject. Perhaps \( R_1 \) seems more intuitive, but note that we do not have either relation in our hierarchy of typed domains. Reason: we have two basic domains for entities and truth-values, and many other domains for functions, but no domains for binary relations. The intuitive choice between the relations \( R_1 \) and \( R_2 \) is only a matter of taste, and they are not actually used in our analysis.

What we do have in our models is \textit{characteristic functions} of binary relations – functions from entities to functions from entities to truth-values, of type \( e(et) \). In a sentence like \textit{Mary praised Tina}, we treat the verb \textit{praised} as denoting a function \( \textit{praise} \in D_{e(et)} \), which maps the object denotation, \textit{tina}, to a \textit{et} function \( \textit{praise}(\textit{tina}) \). In turn, the function \( \textit{praise}(\textit{tina}) \) maps the subject denotation \textit{mary} to the truth-value \( \textit{praise}(\textit{tina})/\textit{mary} \).

Note that it is here crucial that the \textit{first} argument of the function \textit{praise} is the denotation of the object argument \textit{Tina}. The reason for that is syntax and compositionality: we want to analyze the VP constituent \textit{praised Tina} as having a denotation of its own. In order to do that, the easiest analysis is to apply a function denotation of the verb to an entity denotation of the object.

As we claimed, the domain \( D_{e(et)} \) of \( e(et) \) functions that we assume as possible denotations of transitive verbs is isomorphic to the domain of binary relations on the domain of entities \( D_e \). Thus, for any convention we pick for thinking of the denotation of transitive verbs, \( e(et) \) functions can describe this convention adequately. We describe the situation above by uniformly using the \( e(et) \) function we defined:

\[
\begin{align*}
f : \text{tina} & \rightarrow [\text{tina} \rightarrow 0 \text{ john} \rightarrow 0 \text{ mary} \rightarrow 1] \\
\text{john} & \rightarrow [\text{tina} \rightarrow 0 \text{ john} \rightarrow 0 \text{ mary} \rightarrow 1] \\
\text{mary} & \rightarrow [\text{tina} \rightarrow 1 \text{ john} \rightarrow 0 \text{ mary} \rightarrow 1]
\end{align*}
\]

As we said, in the type-theoretical jargon, such functions are often called \textit{Curried functions} – a tribute to the logician Haskell Curry who rediscovered this method and made it famous. For some reason terms like \textit{Schönfinkeling} or \textit{Schönfinkelization}, after the logician Moses Schönfinkel who first discovered this method, have never equally caught up.