

Adversarial Paths

by Sven Nilsen, 2018

Based on ideas from a discussion with Adam Nemecek, in this paper I formalize what it means to make a choice $A \sim 0$ in path semantics, without being able to remember how one ended up in $A \sim 1$. The notation is designed to easily work with higher order dependencies between choices.

A choice is a type A with an associated function $A::f : A \rightarrow B$.

An adversarial path is a higher order equivalence path such that:

$$A \sim 0 : T \rightarrow A \sim 1$$

$$A \sim 1 := \lambda(g : B \rightarrow \text{bool}) = g \subseteq \bigcup x : T \{ (A::f \sim 1)'(x) \}$$

Where $A \sim 0$ consumes A as a resource, and $A \sim 1$ produces a new resource.

The rest of this paper explains the notation above.

I will derive it from the notation used in the paper “Equivalence Paths”: An equivalence path is a function $\sim f$ created from some function f . The \sim unary operator is called the “universal equivalence path”. One can think of $\sim f$ as crossing out all input-output pairs that intersect, such that for all outputs, there exist a unique input. To access all equivalence paths of a function, the transformation is controlled by manipulating the input domain constraint $\sim f\{\forall f\}$. A higher order trivial path means that $\sim f'$ depends on some quantified variable. This means that:

$$\forall f : A \rightarrow \text{bool}$$

$$f : A \rightarrow B$$

Is replaced by:

$$\forall f' : T \rightarrow A \rightarrow \text{bool}$$

Since the trivial path $f \sim 0$ of the equivalence path $\sim f\{\forall f\}$ is defined by:

$$f \sim 0 \iff \forall \sim f\{\forall f\}$$

It follows that the higher order trivial path $(f \sim 0)'$ of the higher order equivalence path $\sim f\{\forall f'\}$:

$$(f \sim 0)' \iff \forall \sim f\{\forall f'\}$$

$$(f \sim 0)' : T \rightarrow A \rightarrow \text{bool}$$

Since the existential of f constrained to $f \sim 0$ is determines $f \sim 1$:

$$\exists f \{f \sim 0\} \Leftrightarrow f \sim 1$$

It follows that the higher order existential path of f constrained to $(f \sim 0)'$ determines $(f \sim 1)'$:

$$\exists f \{(f \sim 0)'\} \Leftrightarrow (f \sim 1)'$$

In Adversarial Path Semantics, one exploits the following properties:

$$\exists f \{(f \sim 0)'\} \Leftrightarrow (f \sim 1)'$$

$$\begin{aligned} (f \sim 0)' &: T \rightarrow A \rightarrow \text{bool} \\ (f \sim 1)' &: T \rightarrow B \rightarrow \text{bool} \\ f &: A \rightarrow B \end{aligned}$$

Instead of defining an f for every A , it is associated with A , such that:

$$\exists A::f \{(A::f \sim 0)'\} \Leftrightarrow (A::f \sim 1)'$$

$$\begin{aligned} (A::f \sim 0)' &: T \rightarrow A \rightarrow \text{bool} \\ (A::f \sim 1)' &: T \rightarrow B \rightarrow \text{bool} \\ A::f &: A \rightarrow B \end{aligned}$$

A type A with an associated function $A::f$ is called a “choice”.

An adversarial choice A has the following abstract judgemental properties:

$$\begin{aligned} \forall x \{ (A::f \sim 0)'(x) : \text{unknown} \} \\ \forall x \{ (A::f \sim 1)'(x) : \text{known} \} \end{aligned}$$

This is meant as $A \sim 0$ consumes A as a resource.

The higher order existential path $(\exists A::f \{(A::f \sim 0)'\})(x) \Leftrightarrow (A::f \sim 1)'(x)$ has a sub-type $A \sim 1$:

$$\begin{aligned} A \sim 1 &:= \{(g : B \rightarrow \text{bool}) = g \subseteq \cup x : T \{ (A::f \sim 1)'(x) \} \\ A \sim 1 &: (B \rightarrow \text{bool}) \rightarrow \text{bool} \end{aligned}$$

Because $(A::f \sim 0)'$ is unknown for all inputs, this frees up the syntax $A \sim 0$ to mean something else:

$$A \sim 0 : T \rightarrow A \sim 1$$

Since A is consumed by $A \sim 0$, one can interpret it as consuming A and producing $A \sim 1$. This notation is designed to easily work with higher order dependencies between choices.

$A \sim 0$ is interpreted as making the choice itself, then feeding in some $x : T$ to obtain $A \sim 1$. Notice that this can be thought of as “going through $A \sim 0$ into $A \sim 1$ ”. One can also think of $A \sim 0$ as committing to making a choice, even though the concrete choice is delayed until the next step.