Structure-Preserving Functions

by Sven Nilsen, 2017

Of all functions of type `bool → bool`, there are only two structure-preserving functions:

not := if(false, true)
id := if(true, false)

if := \(a, b) = \(x : bool) = if x { a } else { b }

These functions are called structure-preserving because when using in a path, the path set is always non-empty. All structure-preserving functions have inverses.

In order to be a structure preserving function, it must map from one type to another that contains the same or more number number of elements.

$f : A \rightarrow B \wedge |A| <= |B|$

However, this is not sufficient to define a structure-preserving function. A more accurate way is to think of the type `B` having some implicit sub-type that constrains it to those values returned by `f`. This constrained type must have the same number of elements as the input type:

$f : A \rightarrow B \wedge |A| == |f(A)|$

Another way of writing is using the existential path:

$f : A \rightarrow B \wedge |A| == |\exists f|$

This is a shorthand version for:

$f : A \rightarrow B \wedge |A| == |[\exists f]\ true|$

Therefore, one has the following for any function of type `A → B`:

$|f(A)| == |\exists f|$
$f(A) \subseteq B$

$f : A \rightarrow B$

To construct all structure-preserving functions of type `A → A`, one can use the `id` function, take the partial function pairs `(x, id(x))` and then rearrange the outputs:

$(x_i, id(x_j)) \qquad \neg\exists\ k: (\neg= i)\ \{\ (x_k, id(x_j))\ \}$

This means there exists `|A|!` number of structure-preserving functions of type `A → A`.

If the existential path of a function is `$true_1$`, then the length of the existential path is equal to the length of the output type. The same is also true in the other direction:

$$(∃f <=> true_1) <=> (|∃f| == |B|)$$

$$f : A → B$$

For example:

$$f : bool × bool → bool × bool ∧ |bool × bool| == |f(bool × bool)|$$
$$|bool × bool| == |∃f|$$
$$|bool| · |bool| == |∃f|$$
$$2 · 2 == |∃f|$$
$$4 == |∃f|$$
$$∃f <=> true_1$$

Because `|bool × bool| == 4` and `( ∃f <=> $true_1$ ) <=> ( |∃f| == |bool × bool| )`.

There exists `4! = 4 · 3 · 2 · 1 = 24` number of structure preserving functions
of type `bool × bool → bool × bool`.

For functions of type `A → B` there is a way to count the number of structure-preserving functions:

$$|f| == |A|! · bin(|B|, |A|) == |B|! / (|B| - |A|)!$$

$$f : A → B ∧ |A| == |f(A)|$$

For example, for functions of type `bool → bool × bool`, there exists `12` structure-preserving functions:

$$|f| == |bool|! · bin(|bool × bool|, |bool|)$$
$$|f| == 2! · bin(4, 2)$$
$$|f| == 2 · 6$$
$$|f| == 12$$

$$f : bool → bool × bool ∧ |bool| == |f(bool)|$$

The reason is that there are 2 structure-preserving functions of type `bool → bool`. For each of these functions one can pick 2 values of type `bool × bool` to use as output. The values can not be rearranged per function of type `bool → bool`, because it would be the same as counting the same functions more than once.

From this way of counting structure-preserving functions of different types, there is an algorithm to construct all structure-preserving functions of type `A → B`:

1. Construct all structure-reserving functions of type `A → A`.
2. Find all ordered maps from `B` to `A`.
3. Replace outputs of 1) with 2) using the order of `A` to look up the value of `B`.