CS313K: Logic, Sets, and Functions

J Strother Moore
Department of Computer Sciences
University of Texas at Austin

(Lecture 18)

Last Time...

I promised to give you an example of why "`x` and `y` are permutations of each other if every object occurs the same number of times in each of them" is *not* formalized by:

γ: `(`$\forall$ `e : ((hm e x)=(hm e y))` $\longrightarrow$ `(perm x y))`

But First... "For every e, if e occurs 0 times in x, then x is empty."

$(\forall$ `e` : [(`hm e x`) $=$ `0` $\rightarrow$ (`endp x`)])

versus

"If, for every e, e occurs 0 times in x, then x is empty."

[$\forall$ `e` : (`hm e x`) $=$ `0`] $\rightarrow$ (`endp x`)

Last Time...

I promised to give you an example of why "x and y are permutations of each other if every object occurs the same number of times in each of them" is *not* formalized by:

γ: `(`$\forall$ `e : ((hm e x)=(hm e y))` $\longrightarrow$ `(perm x y))`

The statement: "`x` and `y` are permutations of each other if every object occurs the same number of times in each of them" is a true statement.

So γ doesn't formalize it if I can show you an `x` and `y` that make γ false.

γ: `(`$\forall$ `e : ((hm e x)=(hm e y))` $\longrightarrow$ `(perm x y))`

γ: `(`$\forall$ `e : ((hm e x)=(hm e y))` $\longrightarrow$ `(perm x y))`

Let

`x : '(0)`

`y : '(1)`

γ: `(`$\forall$ `e : ((hm e '(0))=(hm e '(1)))`$\longrightarrow$`(perm '(0) '(1)))`

γ: `(∀ e : ((hm e '(0))=(hm e '(1)))→nil)`

γ: (∀ e : ((hm e '(0))=(hm e '(1)))→nil)

↔

[((hm e '(0))=(hm e '(1)))→nil)/{e ← 0}

∧

((hm e '(0))=(hm e '(1)))→nil)/{e ← 1}

∧

((hm e '(0))=(hm e '(1)))→nil)/{e ← 2}

∧

...]

γ: `(∀ e : ((hm e '(0))=(hm e '(1)))→nil)`

$\leftrightarrow$

`[((hm 0 '(0))=(hm 0 '(1)))→nil`

$\wedge$

`((hm 1 '(0))=(hm 1 '(1)))→nil`

$\wedge$

`((hm 2 '(0))=(hm 2 '(1)))→nil`

$\wedge$

`...]`

γ: (∀ e : ((hm e '(0))=(hm e '(1)))→nil)

↔

[((1=0)→nil)

∧

((hm 1 '(0))=(hm 1 '(1)))→nil

∧

((hm 2 '(0))=(hm 2 '(1)))→nil

∧

...]

γ: (∀ e : ((hm e '(0))=(hm e '(1)))→nil)

↔

[nil→nil

∧

((hm 1 '(0))=(hm 1 '(1)))→nil

∧

((hm 2 '(0))=(hm 2 '(1)))→nil

∧

...]

γ: (∀ e : ((hm e '(0))=(hm e '(1)))→nil)

↔

[t

∧

((hm 1 '(0))=(hm 1 '(1)))→nil

∧

((hm 2 '(0))=(hm 2 '(1)))→nil

∧

...]

γ: ($\forall$ e : ((hm e '(0))=(hm e '(1)))$\longrightarrow$nil)

$\leftrightarrow$

[t

$\wedge$

(0=1)$\longrightarrow$nil

$\wedge$

((hm 2 '(0))=(hm 2 '(1)))$\longrightarrow$nil

$\wedge$

...]

γ: (∀ e : ((hm e '(0))=(hm e '(1)))→nil)

↔

[t

∧

nil→nil

∧

((hm 2 '(0))=(hm 2 '(1)))→nil

∧

...]

γ: `(`$\forall$` e : ((hm e '(0))=(hm e '(1)))`$\rightarrow$`nil)`

$\leftrightarrow$

`[t`

$\wedge$

`t`

$\wedge$

`((hm 2 '(0))=(hm 2 '(1)))`$\rightarrow$`nil`

$\wedge$

`...]`

γ: `(`$\forall$ `e : ((hm e '(0))=(hm e '(1)))`$\longrightarrow$`nil)`

$\leftrightarrow$

`[t`

$\wedge$

`t`

$\wedge$

`(0=0)`$\longrightarrow$`nil`

$\wedge$

`...]`

γ: (∀ e : ((hm e '(0))=(hm e '(1)))→nil)

↔

[t

∧

t

∧

t→nil

∧

...]

γ: (∀ e : ((hm e '(0))=(hm e '(1)))→nil)

$\leftrightarrow$

[t

$\wedge$

t

$\wedge$

nil

$\wedge$

...]

γ: `(∀ e : ((hm e '(0))=(hm e '(1)))→nil)`

$\leftrightarrow$

`nil`

So γ doesn't have the same meaning as "`x` and `y` are permutations of each other if every object occurs the same number of times in each of them.".