

Recap

- Free variables
- Semantics of the λ -calculus: The three rewrite rules

Substitution

- Notation: $[E/x]D$
 - "Substitute E for x in D"
 - E and D are expressions; x is an identifier
 - This notation not part of λ -calculus, just used to define substitution rules
- Three rules:
 - S1: If D is a single identifier:
 - Rule: $[E/X]x = x$
 - Rule: $[E/X]y = y$ if $y \neq x$
 - S2: If D is a function definition, $(\lambda y . C)$
 - Three cases:
 - Case 1: y is the formal parameter ($x = y$)
 - It doesn't matter what the formal parameter is, so don't change it.
 - Rule: $[E/x](\lambda x . C) = (\lambda x . C)$
 - Case 2: y is not the formal parameter, but y is free in E
 - Example:
 - $[y + 1/x](\lambda y . x + y)$
 - Here, y is free in $y + 1$ but bound in $(\lambda y . x + y)$
 - What happens if we just go ahead and substitute?
 - $(\lambda y . y + 1 + y)$
 - The substitution changed the meaning of the expression!
 - The name y got bound by accident.
 - Need to first replace the y in the expression with something else
 - Can replace $(\lambda y . x + y)$ with $(\lambda z . x + z)$ without changing the meaning.
 - Rule: $[E/x](\lambda y . C) = \lambda z . [E/x][z/y]C$ if $y \neq x$ and y is free in E and Z is not free in E
 - Apply it to the example:
 - $[y + 1/x](\lambda y . x + y) =$
 - $(\lambda z . [y + 1/x][z/y](x + y)) =$
 - $(\lambda z . [y + 1/x](x + z)) =$
 - $(\lambda z . y + 1 + z)$
 - Case 3: y is not the formal parameter, but is not free in E
 - Simple
 - Rule: $[E/x](\lambda y . C) = \lambda y . [E/x]C$
 - S3: If D is a function application, $(C B)$
 - First do substitution both in the function definition C and its argument B.
 - Then apply the resulting function to the resulting argument expression.
 - Rule: $[E/x](C B) = ([E/x]C [E/x]B)$