MAKING SUBSTITUTIONS EXPLICIT IN SASyLF

Michael Ariotti & John Tang Boyland

University of Wisconsin-Milwaukee

Wisconsin, USA

What is SASyLF? (part 1)

- **S**econd-Order **A**bstract **Sy**ntax **L**ogical **F**ramework
- Interactive proof assistant (via Eclipse plug-in)
- Developed at CMU [Aldrich et al., 2008]
- Maintained by John Tang Boyland at UW-Milwaukee
- SASyLF is open source, available on GitHub

What is SASyLF? (part II)

- Domain: programming languages and type systems
- Educational:
  - Proof code looks much as it would on paper
  - Errors are generated close to the cause
  - Minimal automation

What is SASyLF? (part III)

- Uses the Edinburgh Logical Framework (LF)
  (as does Twelf)

  [Harper, Honsell, & Plotkin, 1993]

- Proof structure resembles

  Schürmann's meta-language $\mathcal{M}_2^+$

  [PhD Thesis, 2000]

Example: STLC with Booleans

```
terminals lam dot true false if then else Bool value

syntax

t ::= x | lam x:T dot t[x] | t t |
      true | false | if t then t else t

T ::= T -> T | Bool

Gamma ::= * | Gamma, x:T
```

Example: STLC with Booleans

```
terminals lam dot true false if then else Bool value

syntax

t ::= x | lam x:T dot t[x] | t t |
        true | false | if t then t else t

T ::= T -> T | Bool

Gamma ::= * | Gamma, x:T
```

```
judgment typing: Gamma |- t : T
assumes Gamma

------------------- T-Var
Gamma, x:T |- x : T


Gamma, x:T1 |- t2[x] : T2
-------------------------------------- T-Abs
Gamma |- lam x:T1 dot t2[x] : T1 -> T2


Gamma |- t1 : T2 -> T
Gamma |- t2 : T2
--------------------- T-App
Gamma |- t1 t2 : T


-------------------- T-True
Gamma |- true : Bool


--------------------- T-False
Gamma |- false : Bool

Gamma |- t1 : Bool
Gamma |- t2 : T
Gamma |- t3 : T
---------------------------------- T-If
Gamma |- if t1 then t2 else t3: T
```

```
judgment evaluation: t -> t

t1 -> t1'
--------------- E-App1
t1 t2 -> t1' t2

t1 value
t2 -> t2'
--------------- E-App2
t1 t2 -> t1 t2'

t2 value
-------------------------------- E-AppAbs
(lam x:T dot t1[x]) t2 -> t1[t2]


----------------------------- E-IfTrue
if true then t2 else t3 -> t2


------------------------------ E-IfFalse
if false then t2 else t3 -> t3

t1 -> t1'
---------------------------------------------------- E-If
if t1 then t2 else t3 -> if t1' then t2 else t3
```

Reference

Definition

Access

```
judgment typing: Gamma |- t : T
assumes Gamma

------------------- T-Var
Gamma, x:T |- x : T


Gamma, x:T1 |- t2[x] : T2
-------------------------------------- T-Abs
Gamma |- lam x:T1 dot t2[x] : T1 -> T2


Gamma |- t1 : T2 -> T
Gamma |- t2 : T2
--------------------- T-App
Gamma |- t1 t2 : T


-------------------- T-True
Gamma |- true : Bool


--------------------- T-False
Gamma |- false : Bool

Gamma |- t1 : Bool
Gamma |- t2 : T
Gamma |- t3 : T
------------------------------------- T-If
Gamma |- if t1 then t2 else t3: T
```

```
judgment evaluation: t -> t

t1 -> t1'
---------------- E-App1
t1 t2 -> t1' t2

t1 value
t2 -> t2'
---------------- E-App2
t1 t2 -> t1 t2'

t2 value
-------------------------------- E-AppAbs
(lam x:T dot t1[x]) t2 -> t1[t2]


----------------------------- E-IfTrue
if true then t2 else t3 -> t2


------------------------------ E-IfFalse
if false then t2 else t3 -> t3

t1 -> t1'
----------------------------------------------- E-If
if t1 then t2 else t3 -> if t1' then t2 else t3
```

```
judgment typing: Gamma |- t : T
assumes Gamma

-------------------- T-Var
Gamma, x:T |- x : T


Gamma, x:T1 |- t2[x] : T2
-------------------------------------- T-Abs
Gamma |- lam x:T1 dot t2[x] : T1 -> T2


Gamma |- t1 : T2 -> T
Gamma |- t2 : T2
---------------------- T-App
Gamma |- t1 t2 : T


--------------------- T-True
Gamma |- true : Bool


---------------------- T-False
Gamma |- false : Bool

Gamma |- t1 : Bool
Gamma |- t2 : T
Gamma |- t3 : T
--------------------------------- T-If
Gamma |- if t1 then t2 else t3: T
```

```
judgment evaluation: t -> t

t1 -> t1'
--------------- E-App1
t1 t2 -> t1' t2

t1 value
t2 -> t2'
--------------- E-App2
t1 t2 -> t1 t2'

t2 value
-------------------------------- E-AppAbs
(lam x:T dot t1[x]) t2 -> t1[t2]


----------------------------- E-IfTrue
if true then t2 else t3 -> t2


------------------------------ E-IfFalse
if false then t2 else t3 -> t3

t1 -> t1'
----------------------------------------------- E-If
if t1 then t2 else t3 -> if t1' then t2 else t3
```

```
judgment typing: Gamma |- t : T
assumes Gamma

-------------------- T-Var
Gamma, x:T |- x : T


Gamma, x:T1 |- t2[x] : T2
---------------------------------------- T-Abs
Gamma |- lam x:T1 dot t2[x] : T1 -> T2


Gamma |- t1 : T2 -> T
Gamma |- t2 : T2
---------------------- T-App
Gamma |- t1 t2 : T


--------------------- T-True
Gamma |- true : Bool


---------------------- T-False
Gamma |- false : Bool

Gamma |- t1 : Bool
Gamma |- t2 : T
Gamma |- t3 : T
------------------------------------- T-If
Gamma |- if t1 then t2 else t3: T
```

```
judgment evaluation: t -> t

t1 -> t1'
--------------- E-App1
t1 t2 -> t1' t2

t1 value
t2 -> t2'
--------------- E-App2
t1 t2 -> t1 t2'

t2 value
--------------------------------- E-AppAbs
(lam x:T dot t1[x]) t2 -> t1[t2]
```

β-reduction

```
------------------------------ E-IfTrue
if true then t2 else t3 -> t2


------------------------------- E-IfFalse
if false then t2 else t3 -> t3

t1 -> t1'
------------------------------------------------- E-If
if t1 then t2 else t3 -> if t1' then t2 else t3
```

Derivation Trees

Example

$$\dfrac{\dfrac{\dfrac{}{\texttt{*, x:Bool |- true : Bool}}\ \textsc{T-True}}{\texttt{* |- (lam x:Bool dot true) : Bool -> Bool}}\ \textsc{T-Abs} \qquad \dfrac{}{\texttt{* |- false : Bool}}\ \textsc{T-False}}{\texttt{* |- (lam x:Bool dot true) false : Bool}}\ \textsc{T-App}$$

Case Analysis

$$\dfrac{\texttt{Any premises?}}{\texttt{Gamma |- t : T}}\ \textsc{T-Which?}$$

Derivation Trees

Example

$$\dfrac{\dfrac{\dfrac{}{\texttt{*, x:Bool |- true : Bool}}\,\textsc{T-True}}{\texttt{* |- (lam x:Bool dot true) : Bool -> Bool}}\,\textsc{T-Abs} \qquad \dfrac{}{\texttt{* |- false : Bool}}\,\textsc{T-False}}{\boxed{\texttt{* |- (lam x:Bool dot true) false : Bool}}}\,\textsc{T-App}$$

Case Analysis

$$\dfrac{\texttt{Any premises?}}{\texttt{Gamma |- t : T}}\,\textsc{T-Which?}$$

Derivation Trees

Example

$$\dfrac{\dfrac{\dfrac{}{\texttt{*, x:Bool |- true : Bool}}\ \textsc{T-True}}{\boxed{\texttt{* |- (lam x:Bool dot true) : Bool -> Bool}}}\ \textsc{T-Abs} \qquad \dfrac{}{\boxed{\texttt{* |- false : Bool}}}\ \textsc{T-False}}{\texttt{* |- (lam x:Bool dot true) false : Bool}}\ \textsc{T-App}$$

Case Analysis

$$\dfrac{\text{Any premises?}}{\texttt{Gamma |- t : T}}\ \textsc{T-Which?}$$

Derivation Trees

Example

$$\dfrac{\dfrac{\dfrac{}{\texttt{*, x:Bool |- true : Bool}}\,\text{T-True}}{\texttt{* |- (lam x:Bool dot true) : Bool -> Bool}}\,\text{T-Abs} \qquad \dfrac{}{\texttt{* |- false : Bool}}\,\text{T-False}}{\texttt{* |- (lam x:Bool dot true) false : Bool}}\,\text{T-App}$$

Case Analysis

$$\dfrac{\texttt{Any premises?}}{\boxed{\texttt{Gamma |- t : T}}}\,\underline{\text{T-Which?}}$$

A case for each rule possibly applied last

Pierce-like

SASyLF

THEOREM [PRESERVATION]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \rightarrow \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-TRUE: $\mathtt{t} = \mathtt{true}$ $\mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term `true` does not evaluate.

Case T-IF: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IFTRUE, E-IFFALSE, and E-IF. Each rule must be addressed in its own case.

Subcase E-IFTRUE: $\mathtt{t}_1 = \mathtt{true}$ $\mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be `true` in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be $\mathtt{T}$, from a subderivation of $\mathtt{t}$'s typing derivation for this case.

...

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  -------------------------- T-True
  _: Gamma |- true : Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------ T-If
  _: Gamma |- if t1 then t2 else t3 : T
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
    is
      proof by d2
    end case
...
```

Pierce-like

SASyLF

Inputs

Outputs

Theorem [Preservation]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \to \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-True: $\mathtt{t} = \mathtt{true}$ $\mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term true does not evaluate.

Case T-If: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IfTrue, E-IfFalse, and E-If. Each rule must be addressed in its own case.

Subcase E-IfTrue: $\mathtt{t}_1 = \mathtt{true}$ $\mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be true in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be T, from a subderivation of t's typing derivation for this case.

...

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  --------------------------- T-True
  _: Gamma |- true : Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------ T-If
  _: Gamma |- if t1 then t2 else t3 : T
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
    is
      proof by d2
    end case
...
```

Pierce-like

THEOREM [PRESERVATION]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \rightarrow \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-TRUE: $\mathtt{t} = \mathtt{true}$ $\mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term `true` does not evaluate.

Case T-IF: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IFTRUE, E-IFFALSE, and E-IF. Each rule must be addressed in its own case.

Subcase E-IFTRUE: $\mathtt{t}_1 = \mathtt{true}$ $\mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be `true` in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be $\mathtt{T}$, from a subderivation of $\mathtt{t}$'s typing derivation for this case.

...

SASyLF

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  --------------------------- T-True
  _: Gamma |- true : Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------- T-If
  _: Gamma |- if t1 then t2 else t3 : T
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
    is
      proof by d2
    end case
...
```

Pierce-like

THEOREM [PRESERVATION]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \to \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-TRUE: $\mathtt{t} = \mathtt{true}$ $\mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term `true` does not evaluate.

Case T-IF: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IFTRUE, E-IFFALSE, and E-IF. Each rule must be addressed in its own case.

Subcase E-IFTRUE: $\mathtt{t}_1 = \mathtt{true}$ $\mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be `true` in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be T, from a subderivation of t's typing derivation for this case.

...

SASyLF

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  --------------------------- T-True
  _: Gamma |- true : Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------- T-If
  _: Gamma |- if t1 then t2 else t3 : T
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
    is
      proof by d2
    end case
...
```

Pierce-like

SASyLF

Theorem [Preservation]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \rightarrow \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-True: $\mathtt{t} = \mathtt{true}$ $\quad \mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term `true` does not evaluate.

Case T-If: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\quad \Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\quad \Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\quad \Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IfTrue, E-IfFalse, and E-If. Each rule must be addressed in its own case.

Subcase E-IfTrue: $\mathtt{t}_1 = \mathtt{true}$ $\quad \mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be `true` in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be T, from a subderivation of t's typing derivation for this case.

...

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  --------------------------- T-True
  _: Gamma |- true : Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------ T-If
  _: Gamma |- if t1 then t2 else t3 : T
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
    is
      proof by d2
    end case
...
```

Pierce-like

THEOREM [PRESERVATION]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \to \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-TRUE: $\mathtt{t} = \mathtt{true}$ $\quad \mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term `true` does not evaluate.

Case T-IF: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\quad \Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\quad \Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\quad \Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IFTRUE, E-IFFALSE, and E-IF. Each rule must be addressed in its own case.

Subcase E-IFTRUE: $\mathtt{t}_1 = \mathtt{true}$ $\quad \mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be `true` in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be $\mathtt{T}$, from a subderivation of $\mathtt{t}$'s typing derivation for this case.

...

SASyLF

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  -------------------------- T-True
  _: Gamma |- true : Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------- T-If
  _: Gamma |- if t1 then t2 else t3 : T
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
    is
      proof by d2
    end case
...
```

Pierce-like

THEOREM [PRESERVATION]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \to \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-TRUE: $\mathtt{t} = \mathtt{true}$ $\quad \mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term `true` does not evaluate.

Case T-IF: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\quad \Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\quad \Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\quad \Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IFTRUE, E-IFFALSE, and E-IF. Each rule must be addressed in its own case.

Subcase E-IFTRUE: $\mathtt{t}_1 = \mathtt{true}$ $\quad \mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be `true` in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be $\mathtt{T}$, from a subderivation of $\mathtt{t}$'s typing derivation for this case.

...

SASyLF

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  -------------------------- T-True
  _: Gamma |- true : Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------- T-If
  _: Gamma |- if t1 then t2 else t3 : T
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
    is
     proof by d2
    end case
...
```

Pierce-like

THEOREM [PRESERVATION]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \to \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-TRUE: $\mathtt{t} = \mathtt{true}$ $\mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term true does not evaluate.

Case T-IF: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IFTRUE, E-IFFALSE, and E-IF. Each rule must be addressed in its own case.

Subcase E-IFTRUE: $\mathtt{t}_1 = \mathtt{true}$ $\mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be true in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be T, from a subderivation of t's typing derivation for this case.

...

SASyLF

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.    <- Need this
use induction on d
proof by case analysis on d:
case rule
  --------------------------- T-True
  _: Gamma |- true : Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T     <- Have this
  d3: Gamma |- t3 : T
  ------------------------------------- T-If
  _: Gamma |- if t1 then t2 else t3 : T
is
  proof by case analysis on e:
   case rule
     -------------------------------- E-IfTrue
     _: if true then t' else t3 -> t'
   is
     proof by d2 ?
   end case
...
```

SASyLF

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  -------------------------- T-True
  _: Gamma |- true : Bool
  where t := true
  and T := Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------- T-If
  _: Gamma |- if t1 then t2 else t3 : T
  where t := if t1 then t2 else t3
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
      where t1 := true
      and t2 := t'
    is
      proof by d2
    end case
...
```

Pierce-like

THEOREM [PRESERVATION]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \to \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-TRUE: $\mathtt{t} = \mathtt{true}$ $\quad$ $\mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term true does not evaluate.

Case T-IF: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\quad$ $\Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\quad$ $\Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\quad$ $\Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IFTRUE, E-IFFALSE, and E-IF. Each rule must be addressed in its own case.

Subcase E-IFTRUE: $\mathtt{t}_1 = \mathtt{true}$ $\quad$ $\mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be true in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be T, from a subderivation of t's typing derivation for this case.

...

SASyLF

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  -------------------------- T-True
  _: Gamma |- true : Bool
  where t := true
  and T := Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------- T-If
  _: Gamma |- if t1 then t2 else t3 : T
  where t := if t1 then t2 else t3
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
      where t1 := true
      and t2 := t'
    is
      proof by d2
    end case
...
```

Pierce-like

THEOREM [PRESERVATION]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \to \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-TRUE: $\mathtt{t} = \mathtt{true}$ $\quad$ $\mathtt{T} = \mathtt{Bool}$

This case cannot occur, because the term `true` does not evaluate.

Case T-IF: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ $\quad$ $\Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\quad$ $\Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\quad$ $\Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IFTRUE, E-IFFALSE, and E-IF. Each rule must be addressed in its own case.

Subcase E-IFTRUE: $\mathtt{t}_1 = \mathtt{true}$ $\quad$ $\mathtt{t}_2 = \mathtt{t}'$

The term $\mathtt{t}_1$ must be `true` in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be $\mathtt{T}$, from a subderivation of $\mathtt{t}$'s typing derivation for this case.

...

SASyLF

```
theorem preservation:
assumes Gamma
  forall d: Gamma |- t : T
  forall e: t -> t'
  exists Gamma |- t' : T.
use induction on d
proof by case analysis on d:
case rule
  -------------------------- T-True
  _: Gamma |- true : Bool
  where t := true
  and T := Bool
is
  proof by contradiction on e
end case
case rule
  d1: Gamma |- t1 : Bool
  d2: Gamma |- t2 : T
  d3: Gamma |- t3 : T
  ------------------------------------- T-If
  _: Gamma |- if t1 then t2 else t3 : T
  where t := if t1 then t2 else t3
is
  proof by case analysis on e:
    case rule
      -------------------------------- E-IfTrue
      _: if true then t' else t3 -> t'
      where t1 := true
      and t2 := t'
    is
      proof by d2
    end case
...
```

Pierce-like

THEOREM [PRESERVATION]: If $\Gamma \vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \to \mathtt{t}'$, then $\Gamma \vdash \mathtt{t}' : \mathtt{T}$.

Proof: By induction on a derivation of $\Gamma \vdash \mathtt{t} : \mathtt{T}$.

Case T-TRUE: $\boxed{\mathtt{t} = \mathtt{true}}$ $\boxed{\mathtt{T} = \mathtt{Bool}}$

This case cannot occur, because the term true does not evaluate.

Case T-IF: $\boxed{\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3}$ $\Gamma \vdash \mathtt{t}_1 : \mathtt{Bool}$ $\Gamma \vdash \mathtt{t}_2 : \mathtt{T}$ $\Gamma \vdash \mathtt{t}_3 : \mathtt{T}$

There are three rules by which $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$ can evaluate: E-IFTRUE, E-IFFALSE, and E-IF. Each rule must be addressed in its own case.

Subcase E-IFTRUE: $\boxed{\mathtt{t}_1 = \mathtt{true}}$ $\boxed{\mathtt{t}_2 = \mathtt{t}'}$

The term $\mathtt{t}_1$ must be true in this case, which means $\mathtt{t}'$ is the same as $\mathtt{t}_2$. Furthermore, the type of $\mathtt{t}_2$ is known to be T, from a subderivation of t's typing derivation for this case.

...

SASyLF “Where” Clauses

- In the SASyLF proof code
- Optionally required
- Verified by the system
- Verification separate from the proof system

Future Work

- Usability study
- “Where” clauses for inversions
- Auto-generate the clauses

Conclusion

- SASyLF is an educational proof assistant
- Certain substitutions arise through the course of a proof, which were previously hidden to the SASyLF user
- “Where” clauses, our latest contribution, make these substitutions explicit, and are verified by the system
- SASyLF is open source and available at

  `http://github.com/boyland/sasylf`