DTU

Formalizing multiparty session types

Frederik Krogsdal Jacobsen

Technical University of Denmark

The electrical engineering to logic pipeline

Electrical engineer

↓ I would not drive the car I just designed :-(

Apply formal methods

↓ I wish these tools were more helpful :-(

Do a PhD in logic

↓ :-)?

Breaking it down

Formalizing
multiparty
session
types

Breaking it down

Formalizing	encoding into higher order logic
multiparty	
session	
types	

Breaking it down

Formalizing	encoding into higher order logic
multiparty	several agents
session	
types	

Breaking it down

Formalizing	encoding into higher order logic
multiparty	several agents
session	having conversations
types	

Breaking it down

Formalizing	encoding into higher order logic
multiparty	several agents
session	having conversations
types	following a structured protocol

Encoding into higher order logic

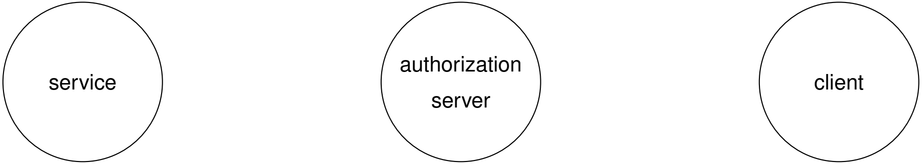
```
| ‹subst_type (Rec s) t x = Rec (subst_type s t x)›
| ‹subst_type (RecFVariable s) t x = (if s = t then x else RecFVariable s)›
| ‹subst_type (RecBVariable s) _ _ = RecBVariable s›

subsection ‹Session subtyping›

lemma [mono]: ‹x ≤ y ⟹
       (∃mpb mpa r. xa = Internal r mpa ∧ xb = Internal r mpb ∧ fmrel_on_fset (fmdom mpb) (λ(s, s') (t, t'). x t s ∧ x s' t') mpa mpb) ∨ xa = Terminated ∧ xb = Terminated ∨ (∃s t. xa = R
       (∃mpb mpa r. xa = Internal r mpa ∧ xb = Internal r mpb ∧ fmrel_on_fset (fmdom mpb) (λ(s, s') (t, t'). y t s ∧ y s' t') mpa mpb) ∨ xa = Terminated ∧ xb = Terminated ∨ (∃s t. xa = R
proof
  assume ‹x ≤ y› ‹(∃mpb mpa r. xa = Internal r mpa ∧ xb = Internal r mpb ∧ fmrel_on_fset (fmdom mpb) (λ(s, s') (t, t'). x t s ∧ x s' t') mpa mpb) ∨ xa = Terminated ∧ xb = Terminated ∨ (∃
  then show ‹(∃mpb mpa r. xa = Internal r mpa ∧ xb = Internal r mpb ∧ fmrel_on_fset (fmdom mpb) (λ(s, s') (t, t'). y t s ∧ y s' t') mpa mpb) ∨ xa = Terminated ∧ xb = Terminated ∨ (∃s t.
    by (smt (z3) ‹x ≤ y› case_prod_unfold dual_order.refl fmrel_on_fset_mono le_funD le_funI predicate2D)
qed

lemma [mono]: ‹x ≤ y ⟹ ((λ(s, s') (t, t'). x s t ∧ x s' t') a b) ⟶ ((λ(s, s') (t, t'). y s t ∧ y s' t') a b)›
  by blast

coinductive subtype :: ‹('r, 'm, 't) type ⇒ ('r, 'm, 't) type ⇒ bool› where
  SubExternal: ‹⟦fmrel_on_fset (fmdom mpa) (λ(s,s') (t,t'). subtype s t ∧ subtype s' t') mpa mpb⟧ ⟹ subtype (External r mpa) (External r mpb)›
| SubInternal: ‹⟦fmrel_on_fset (fmdom mpb) (λ(s,s') (t,t'). subtype t s ∧ subtype s' t') mpa mpb⟧ ⟹ subtype (Internal r mpa) (Internal r mpb)›
| SubEnd: ‹subtype Terminated Terminated›
| SubRecL: ‹subtype (opening_type s (Rec s)) t ⟹  subtype (Rec s) t›
| SubRecR: ‹subtype s (opening_type t (Rec t)) ⟹ subtype s (Rec t)›

subsection ‹Typing contexts›

type_synonym ('r, 'm, 'p, 't) proc_type_ctx = ‹('p, ('r, 'm, 't) type list) fmap›
type_synonym ('v, 'r, 'c, 's, 'm, 't) chan_type_ctx = ‹(('v, 'r, 'c, 's) channel, ('r, 'm, 't) type) fmap›

definition chan_not_in_ctx :: ‹('v, 'r, 'c :: countable, 's :: countable, 'm, 't) chan_type_ctx ⇒ 's ⇒ bool› where
  ‹chan_not_in_ctx Γ s = fmpred (λc _. fresh_session_channel c s) Γ›

definition unique_chan_in_ctx :: ‹('v, 'r, 'c :: countable, 's :: countable, 'm, 't) chan_type_ctx ⇒ 's ⇒ bool› where
  ‹unique_chan_in_ctx Γ s = fmpred (λc _. ∃r. c = FRole s r) Γ›

definition sub_chan_ctx :: ‹('v, 'r, 'c :: countable, 's :: countable, 'm, 't) chan_type_ctx ⇒ ('v, 'r, 'c :: countable, 's :: countable, 'm, 't) chan_type_ctx ⇒ bool› where
  ‹sub_chan_ctx Γ Γ' = fmrel subtype Γ Γ'›

subsection ‹Typing context reductions›

datatype ('r, 's, 'm, 't) transition_label =
  TLExternal 's 'r 'r 'm ‹('r, 'm, 't) type›
| TLInternal 's 'r 'r 'm ‹('r, 'm, 't) type›
| TLComm 's 'r 'r 'm

inductive transition_type_ctx :: ‹('v, 'r, 'c :: countable, 's :: countable, 'm, 't) chan_type_ctx ⇒ ('r, 's, 'm, 't) transition_label ⇒ ('v, 'r, 'c, 's, 'm, 't) chan_type_ctx ⇒ bool› w
  ΓInternal: ‹⟦Γ = fmap_of_list [(FRole s p, Internal q mps)]; Γ' = fmap_of_list [(FRole s p, S')]; fmlookup mps m = Some (S, S')⟧ ⟹ transition_type_ctx Γ (TLInternal s p q m S) Γ'›
| ΓExternal: ‹⟦Γ = fmap_of_list [(FRole s p, External q mps)]; Γ' = fmap_of_list [(FRole s p, S')]; fmlookup mps m = Some (S, S')⟧ ⟹ transition_type_ctx Γ (TLExternal s p q m S) Γ'›
| ΓComm: ‹⟦transition_type_ctx Γ1 (TLInternal s p q m S) Γ1'; transition_type_ctx Γ2 (TLExternal s q p m T) Γ2'; subtype S T⟧ ⟹ transition_type_ctx (Γ1 ++f Γ2) (TLComm s p q m) (Γ1' ++f Γ
| ΓRec: ‹transition_type_ctx (fmupd c (opening_type S (Rec S)) Γ) α Γ' ⟹ transition_type_ctx (fmupd c (Rec S) Γ) α Γ'›
| ΓCong: ‹transition_type_ctx Γ α Γ' ⟹ transition_type_ctx (fmupd c S Γ) α (fmupd c S Γ')›

definition ‹has_transition Γ α ≡ ∃Γ'. transition_type_ctx Γ α Γ'›

definition ‹reduction_ctx Γ Γ' ≡ ∃s p q m. transition_type_ctx Γ (TLComm s p q m) Γ'›
```

Encoding into higher order logic

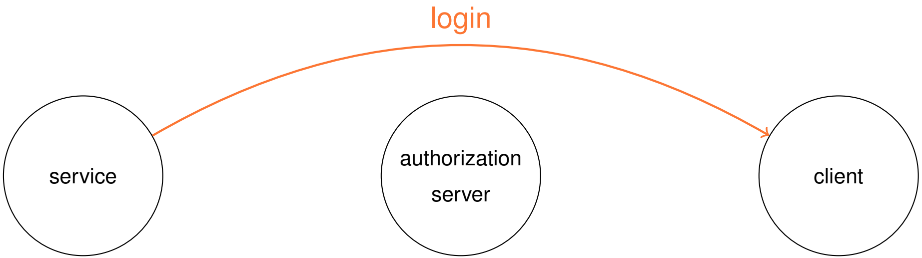
Issues:

- Barendregt convention
- Having to provide actual algorithms
- What works in text does not necessarily work in Isabelle

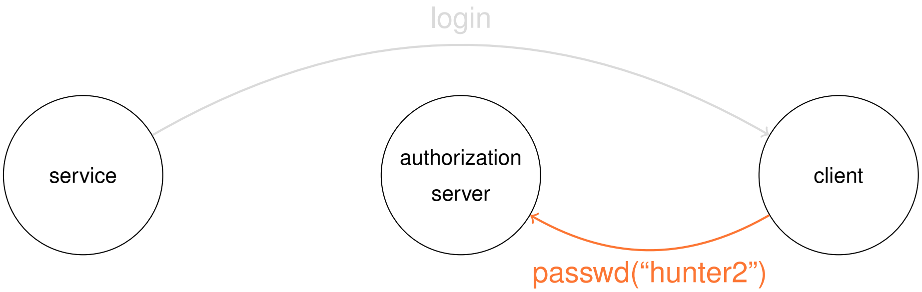
several agents



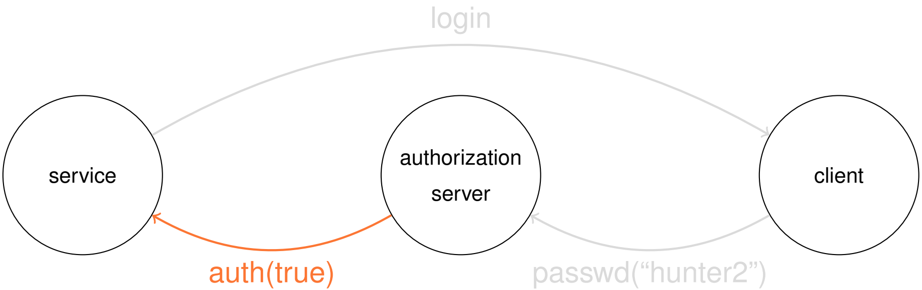
several agents having conversations



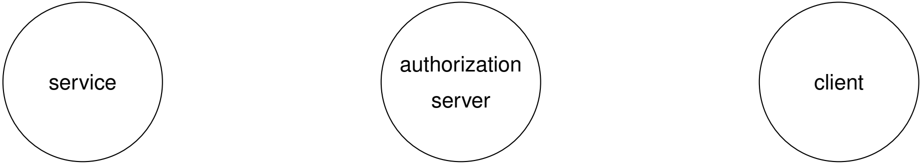
several agents having conversations



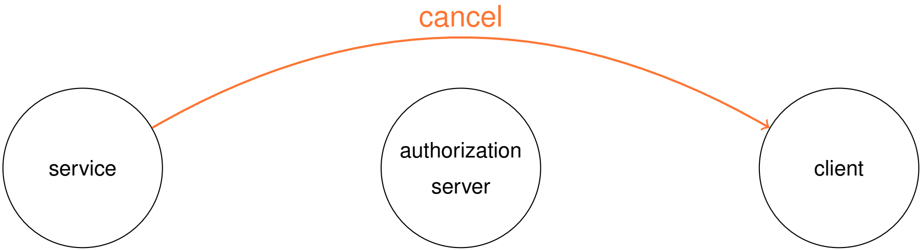
several agents having conversations



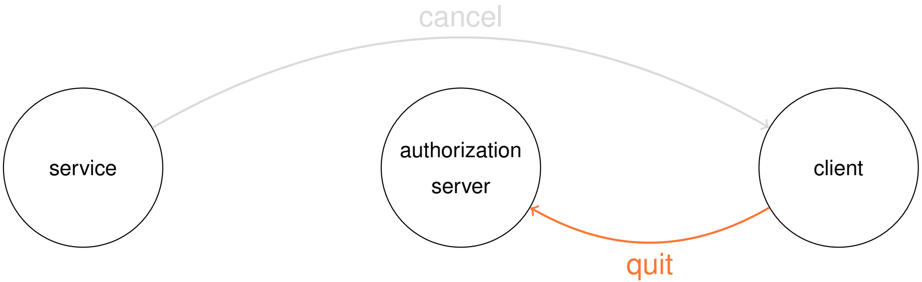
several agents having conversations



several agents having conversations



several agents having conversations



following a structured protocol

$$c \oplus \begin{cases} \mathrm{login}.a\&\mathrm{auth}(\mathrm{Bool}) \\ \mathrm{cancel} \end{cases}$$

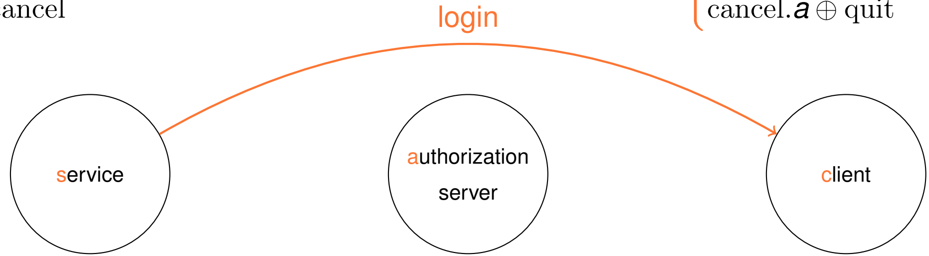
$$s\& \begin{cases} \mathrm{login}.a \oplus \mathrm{passwd}(\mathrm{String}) \\ \mathrm{cancel}.a \oplus \mathrm{quit} \end{cases}$$

$$c\& \begin{cases} \mathrm{passwd}(\mathrm{String}).s\&\mathrm{auth}(\mathrm{Bool}) \\ \mathrm{quit} \end{cases}$$

following a structured protocol

$$c \oplus \begin{cases} \mathrm{login}.a \& \mathrm{auth}(\mathrm{Bool}) \\ \mathrm{cancel} \end{cases}$$

$$s \& \begin{cases} \mathrm{login}.a \oplus \mathrm{passwd}(\mathrm{String}) \\ \mathrm{cancel}.a \oplus \mathrm{quit} \end{cases}$$

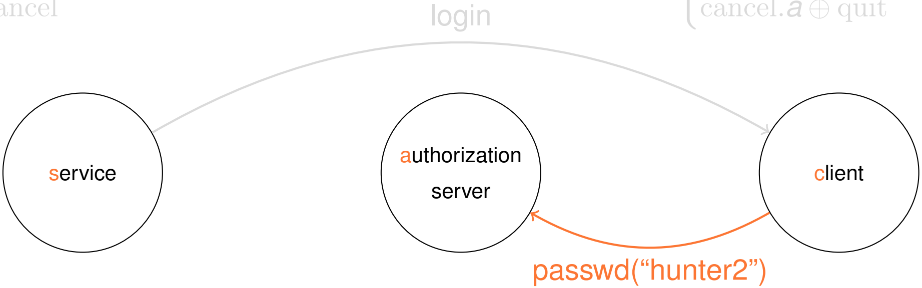


$$c \& \begin{cases} \mathrm{passwd}(\mathrm{String}).s \& \mathrm{auth}(\mathrm{Bool}) \\ \mathrm{quit} \end{cases}$$

following a structured protocol

$$c \oplus \begin{cases} \mathrm{login}.a \& \mathrm{auth}(\mathrm{Bool}) \\ \mathrm{cancel} \end{cases}$$

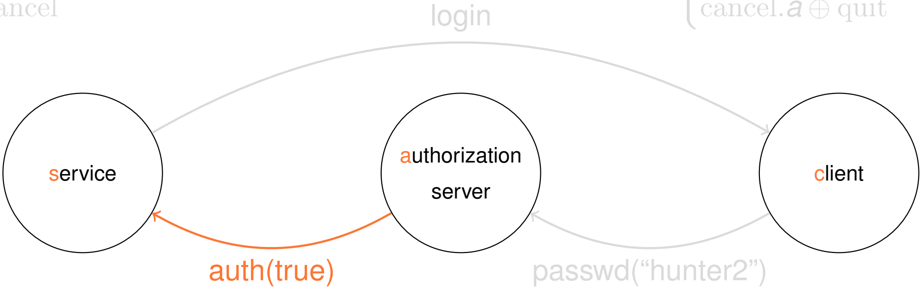
$$s \& \begin{cases} \mathrm{login}.a \oplus \mathrm{passwd}(\mathrm{String}) \\ \mathrm{cancel}.a \oplus \mathrm{quit} \end{cases}$$



$$c \& \begin{cases} \mathrm{passwd}(\mathrm{String}).s \& \mathrm{auth}(\mathrm{Bool}) \\ \mathrm{quit} \end{cases}$$

following a structured protocol

$$c \oplus \begin{cases} \mathrm{login}.a \& \mathrm{auth}(\mathrm{Bool}) \\ \mathrm{cancel} \end{cases} \qquad s \& \begin{cases} \mathrm{login}.a \oplus \mathrm{passwd}(\mathrm{String}) \\ \mathrm{cancel}.a \oplus \mathrm{quit} \end{cases}$$



$$c \& \begin{cases} \mathrm{passwd}(\mathrm{String}).s \& \mathrm{auth}(\mathrm{Bool}) \\ \mathrm{quit} \end{cases}$$

following a structured protocol

$$c \oplus \begin{cases} \mathrm{login}.a\&\mathrm{auth}(\mathrm{Bool}) \\ \mathrm{cancel} \end{cases}$$

$$s\& \begin{cases} \mathrm{login}.a \oplus \mathrm{passwd}(\mathrm{String}) \\ \mathrm{cancel}.a \oplus \mathrm{quit} \end{cases}$$

service

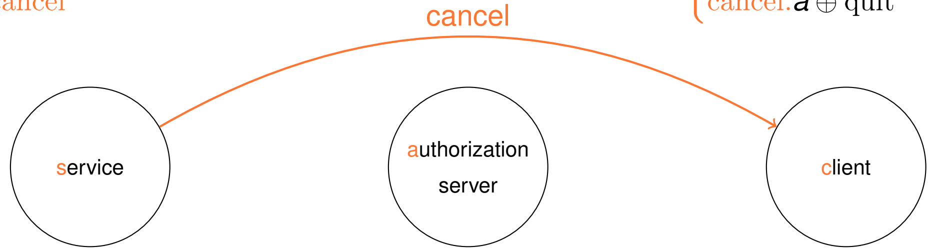
authorization server

client

$$c\& \begin{cases} \mathrm{passwd}(\mathrm{String}).s\&\mathrm{auth}(\mathrm{Bool}) \\ \mathrm{quit} \end{cases}$$

following a structured protocol

$$c \oplus \begin{cases} \mathrm{login}.a\&\mathrm{auth}(\mathrm{Bool}) \\ \mathrm{cancel} \end{cases} \qquad s\& \begin{cases} \mathrm{login}.a \oplus \mathrm{passwd}(\mathrm{String}) \\ \mathrm{cancel}.a \oplus \mathrm{quit} \end{cases}$$

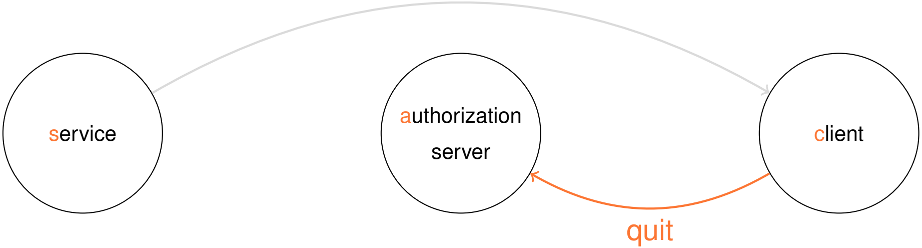


$$c\& \begin{cases} \mathrm{passwd}(\mathrm{String}).s\&\mathrm{auth}(\mathrm{Bool}) \\ \mathrm{quit} \end{cases}$$

following a structured protocol

$$c \oplus \begin{cases} \mathrm{login}.a\&\mathrm{auth}(\mathrm{Bool}) \\ \mathrm{cancel} \end{cases}$$

$$s\& \begin{cases} \mathrm{login}.a \oplus \mathrm{passwd}(\mathrm{String}) \\ \mathrm{cancel}.a \oplus \mathrm{quit} \end{cases}$$



$$c\& \begin{cases} \mathrm{passwd}(\mathrm{String}).s\&\mathrm{auth}(\mathrm{Bool}) \\ \mathrm{quit} \end{cases}$$

following a structured protocol

$$c \oplus \begin{cases} \mathrm{login}.a \& \mathrm{auth}(\mathrm{Bool}) \\ \mathrm{cancel} \end{cases}$$

$$s \& \begin{cases} \mathrm{login}.a \oplus \mathrm{passwd}(\mathrm{String}) \\ \mathrm{cancel}.a \oplus \mathrm{quit} \end{cases}$$

$$c \& \begin{cases} \mathrm{passwd}(\mathrm{String}).s \& \mathrm{auth}(\mathrm{Bool}) \\ \mathrm{quit} \end{cases}$$

multiparty session types

$$\frac{\Theta(X) = S_1, \ldots, S_n}{\Theta \vdash X : S_1, \ldots, S_n}\;[\text{T-}X] \qquad \frac{S \leqslant S'}{c : S \vdash c : S'}\;[\text{T-Sub}] \qquad \frac{\forall i \in 1..n \quad c_i : S_i \vdash c_i : \mathbf{end}}{\mathrm{end}(c_1 : S_1, \ldots, c_n : S_n)}\;[\text{T-end}]$$

$$\frac{\mathrm{end}(\Gamma)}{\Theta \cdot \Gamma \vdash \mathbf{0}}\;[\text{T-}\mathbf{0}] \qquad \frac{\Theta, X : S_1, \ldots, S_n \cdot x_1 : S_1, \ldots, x_n : S_n \vdash P \qquad \Theta, X : S_1, \ldots, S_n \cdot \Gamma \vdash Q}{\Theta \cdot \Gamma \vdash \mathbf{def}\; X(x_1 : S_1, \ldots, x_n : S_n) = P \;\mathbf{in}\; Q}\;[\text{T-}\mathbf{def}]$$

$$\frac{\Theta \vdash X : S_1, \ldots, S_n \quad \mathrm{end}(\Gamma_0) \quad \forall i \in 1..n \quad \Gamma_i \vdash c_i : S_i}{\Theta \cdot \Gamma_0, \Gamma_1, \ldots, \Gamma_n \vdash X\langle c_1, \ldots, c_n \rangle}\;[\text{T-}X]$$

$$\frac{\Gamma_1 \vdash c : \mathbf{q} \&_{i \in I} \mathtt{m}_i(S_i).S_i' \quad \forall i \in I \quad \Theta \cdot \Gamma, y_i : S_i, c : S_i' \vdash P_i}{\Theta \cdot \Gamma, \Gamma_1 \vdash c[\mathbf{q}] \sum_{i \in I} \mathtt{m}_i(y_i).P_i}\;[\text{T-}\&]$$

$$\frac{\Gamma_1 \vdash c : \mathbf{q} \oplus \mathtt{m}(S).S' \quad \Gamma_2 \vdash d : S \quad \Theta \cdot \Gamma, c : S' \vdash P}{\Theta \cdot \Gamma, \Gamma_1, \Gamma_2 \vdash c[\mathbf{q}] \oplus \mathtt{m}\langle d \rangle.P}\;[\text{T-}\oplus] \qquad \frac{\Theta \cdot \Gamma_1 \vdash P_1 \qquad \Theta \cdot \Gamma_2 \vdash P_2}{\Theta \cdot \Gamma_1, \Gamma_2 \vdash P_1 \mid P_2}\;[\text{T-}|]$$

$$\frac{\Gamma' = \{s[\mathbf{p}] : S_{\mathbf{p}}\}_{\mathbf{p} \in I} \quad \varphi(\Gamma') \quad s \notin \Gamma \quad \Theta \cdot \Gamma, \Gamma' \vdash P}{\Theta \cdot \Gamma \vdash (\nu s : \Gamma')\, P}\;[\text{T-}\nu]$$

where φ is a typing context property

Scalas & Yoshida, POPL'19, https://doi.org/10.1145/3290343

multiparty session types

$$\frac{\Theta(X) = S_1, \ldots, S_n}{\Theta \vdash X{:}S_1, \ldots, S_n}\ [\text{T-}X] \qquad \boxed{\frac{S \leqslant S'}{c{:}S \vdash c{:}S'}\ [\text{T-Sub}]} \qquad \frac{\forall i \in 1..n \quad c_i{:}S_i \vdash c_i{:}\mathbf{end}}{\mathrm{end}(c_1{:}S_1, \ldots, c_n{:}S_n)}\ [\text{T-end}]$$

$$\frac{\mathrm{end}(\Gamma)}{\Theta \cdot \Gamma \vdash \mathbf{0}}\ [\text{T-}\mathbf{0}] \qquad \frac{\Theta, X{:}S_1, \ldots, S_n \cdot x_1{:}S_1, \ldots, x_n{:}S_n \vdash P \qquad \Theta, X{:}S_1, \ldots, S_n \cdot \Gamma \vdash Q}{\Theta \cdot \Gamma \vdash \mathbf{def}\ X(x_1{:}S_1, \ldots, x_n{:}S_n) = P\ \mathbf{in}\ Q}\ [\text{T-}\mathbf{def}]$$

$$\frac{\Theta \vdash X{:}S_1, \ldots, S_n \quad \mathrm{end}(\Gamma_0) \quad \forall i \in 1..n \quad \Gamma_i \vdash c_i{:}S_i}{\Theta \cdot \Gamma_0, \Gamma_1, \ldots, \Gamma_n \vdash X\langle c_1, \ldots, c_n\rangle}\ [\text{T-}X]$$

$$\frac{\Gamma_1 \vdash c{:}\mathbf{q}\&_{i\in I}\mathtt{m}_i(S_i).S_i' \quad \forall i\in I \quad \Theta \cdot \Gamma, y_i{:}S_i, c{:}S_i' \vdash P_i}{\Theta \cdot \Gamma, \Gamma_1 \vdash c[\mathbf{q}]\sum_{i\in I}\mathtt{m}_i(y_i).P_i}\ [\text{T-}\&]$$

$$\frac{\Gamma_1 \vdash c{:}\mathbf{q}\oplus\mathtt{m}(S).S' \quad \Gamma_2 \vdash d{:}S \quad \Theta \cdot \Gamma, c{:}S' \vdash P}{\Theta \cdot \Gamma, \Gamma_1, \Gamma_2 \vdash c[\mathbf{q}]\oplus\mathtt{m}\langle d\rangle.P}\ [\text{T-}\oplus] \qquad \frac{\Theta \cdot \Gamma_1 \vdash P_1 \qquad \Theta \cdot \Gamma_2 \vdash P_2}{\Theta \cdot \Gamma_1, \Gamma_2 \vdash P_1 \mid P_2}\ [\text{T-}|]$$

$$\frac{\Gamma' = \{s[\mathbf{p}]{:}S_{\mathbf{p}}\}_{\mathbf{p}\in I} \quad \varphi(\Gamma') \quad s \notin \Gamma \quad \Theta \cdot \Gamma, \Gamma' \vdash P}{\Theta \cdot \Gamma \vdash (\nu s{:}\Gamma')\,P}\ [\text{T-}\nu]$$

where φ is a typing context property

Scalas & Yoshida, POPL'19, https://doi.org/10.1145/3290343

multiparty session types

$$\frac{\Theta(X) = S_1, \ldots, S_n}{\Theta \vdash X{:}S_1, \ldots, S_n}\ [\text{T-X}] \qquad \frac{S \leqslant S'}{c{:}S \vdash c{:}S'}\ [\text{T-Sub}] \qquad \frac{\forall i \in 1..n \quad c_i{:}S_i \vdash c_i{:}\mathbf{end}}{\mathrm{end}(c_1{:}S_1, \ldots, c_n{:}S_n)}\ [\text{T-end}]$$

$$\frac{\mathrm{end}(\Gamma)}{\Theta \cdot \Gamma \vdash \mathbf{0}}\ [\text{T-}\mathbf{0}] \qquad \frac{\Theta, X{:}S_1, \ldots, S_n \cdot x_1{:}S_1, \ldots, x_n{:}S_n \vdash P \qquad \Theta, X{:}S_1, \ldots, S_n \cdot \Gamma \vdash Q}{\Theta \cdot \Gamma \vdash \mathbf{def}\ X(x_1{:}S_1, \ldots, x_n{:}S_n) = P\ \mathbf{in}\ Q}\ [\text{T-}\mathbf{def}]$$

$$\frac{\Theta \vdash X{:}S_1, \ldots, S_n \quad \mathrm{end}(\Gamma_0) \quad \forall i \in 1..n \quad \Gamma_i \vdash c_i{:}S_i}{\Theta \cdot \Gamma_0, \Gamma_1, \ldots, \Gamma_n \vdash X\langle c_1, \ldots, c_n \rangle}\ [\text{T-X}]$$

$$\frac{\Gamma_1 \vdash c{:}\mathbf{q}\&_{i\in I}\mathtt{m}_i(S_i).S'_i \quad \forall i \in I \quad \Theta \cdot \Gamma, y_i{:}S_i, c{:}S'_i \vdash P_i}{\Theta \cdot \Gamma, \Gamma_1 \vdash c[\mathbf{q}]\sum_{i\in I}\mathtt{m}_i(y_i).P_i}\ [\text{T-}\&]$$

$$\frac{\Gamma_1 \vdash c{:}\mathbf{q}\oplus\mathtt{m}(S).S' \quad \Gamma_2 \vdash d{:}S \quad \Theta \cdot \Gamma, c{:}S' \vdash P}{\Theta \cdot \Gamma, \Gamma_1, \Gamma_2 \vdash c[\mathbf{q}]\oplus\mathtt{m}\langle d \rangle.P}\ [\text{T-}\oplus] \qquad \frac{\Theta \cdot \Gamma_1 \vdash P_1 \qquad \Theta \cdot \Gamma_2 \vdash P_2}{\Theta \cdot \Gamma_1, \Gamma_2 \vdash P_1 \mid P_2}\ [\text{T-}|]$$

$$\frac{\Gamma' = \{s[\mathbf{p}]{:}S_{\mathbf{p}}\}_{\mathbf{p}\in I} \quad \boxed{\varphi(\Gamma')} \quad s \notin \Gamma \quad \Theta \cdot \Gamma, \Gamma' \vdash P}{\Theta \cdot \Gamma \vdash (\nu s{:}\Gamma')\,P}\ [\text{T-}\nu]$$

where φ is a typing context property

Scalas & Yoshida, POPL'19, https://doi.org/10.1145/3290343