Elaborating F_i^+ to record calculus λ^{rec}

Snow

March 14, 2023

Changes in λ^{rec} 0) Simplify the system, drop polymorphism, and use simple labels
1) Type soundness proved
2) Use list for record types, so the time cost of concatenation can be moved to compilation.

1 Syntax of λ^{rec}

$ttyp,\ At,\ Bt,\ Ct$	::=		types
	|	$\top$	top type
	|	$\bot$	bottom type
	|	$\mathbb{B}$	base type
	|	$At \to Bt$	function types
	|	$\{ll : At, Bt\}$	record

$texp,\ t,\ te,\ tp,\ tv,\ tu$	::=		target term
	|	x	variable
	|	b	base value
	|	$\lambda x.t$	abstractions
	|	$\mathsf{fix}\, x.t$	fixpoint
	|	$t_1\ t_2$	applications
	|	$\{\}$	empty record
	|	$\{ll \Rightarrow t_1', t_2\}$	
	|	$t_1.ll$	projection
	|	$t_1; t_2$	concatenation

Notes 0) Multi-field records are nullable. 1) The whole program evaluates to a record form. 3) l stands for a string.

$\boxed{\textbf{value}\ tv}$ *(Values)*

$$\text{VALUE-UNIT}\ \frac{}{\textbf{value}\ \{\}} \qquad \text{VALUE-LIT}\ \frac{}{\textbf{value}\ b} \qquad \text{VALUE-ABS}\ \frac{}{\textbf{value}\ \lambda x.t} \qquad \text{VALUE-MERGE}\ \frac{\textbf{value}\ tv_1 \qquad \textbf{value}\ tv_2}{\textbf{value}\ \{ll \Rightarrow tv_1, tv_2\}}$$

$\boxed{t \to t'}$ *(Record Calculus Structrual Reduction)*

$$\text{TS-PROJ1}\ \frac{t \to t'}{t.ll \to t'.ll} \qquad \text{TS-APPL}\ \frac{t \to t'}{t\ t_2 \to t'\ t_2} \qquad \text{TS-APPR}\ \frac{\textbf{value}\ tv \qquad t \to t'}{tv\ t \to tv\ t'} \qquad \text{TS-MERGEL}\ \frac{t \to t'}{t; t_2 \to t'; t_2} \qquad \text{TS-MERGER}\ \frac{\textbf{value}\ tv \qquad t \to t'}{tv; t \to tv; t'}$$

$$\text{TS-RCDHEAD}\ \frac{t \to t'}{\{ll \Rightarrow t, t_2\} \to \{ll \Rightarrow t', t_2\}} \qquad \text{TS-RCDTAIL}\ \frac{\textbf{value}\ tv \qquad t \to t'}{\{ll \Rightarrow tv, t\} \to \{ll \Rightarrow tv, t'\}} \qquad \text{TS-MERGEEMPTY}\ \frac{\textbf{value}\ tv}{\{\}; tv \to tv}$$

TS-MergeRcd
$$\frac{\begin{array}{c}\mathbf{value}\ tv_1\\ \mathbf{value}\ tv_2 \qquad \mathbf{value}\ tv_3\end{array}}{(\{ll \Rightarrow tv_1, tv_2\}); tv_3 \to \{ll \Rightarrow tv_1, (tv_2; tv_3)\}}$$

TS-ProjRcd
$$\frac{\mathbf{value}\ tv \qquad \mathbf{lookup}\ ll\ tv \Rightarrow t}{tv.ll \to t}$$

TS-AppAbs
$$\frac{}{(\lambda x.t)\ tv \to t[x \mapsto tv]}$$

TS-Fixpoint
$$\frac{}{\mathsf{fix}\, x.t \to t[x \mapsto \mathsf{fix}\, x.t]}$$

$\boxed{At \,\&\, Bt \Rightarrow Ct}$ *(Concatenate types)*

CT-Nil
$$\frac{}{\top \,\&\, Bt \Rightarrow Bt}$$

CT-Rcd
$$\frac{\begin{array}{c}\mathbf{lookup}\ ll\ Bt_2 \Rightarrow At \ \vee\ \mathbf{lookup}\ ll\ Bt_2 = / >\\ Bt_1 \,\&\, Bt_2 \Rightarrow Ct\end{array}}{(\{ll : At, Bt_1\}) \,\&\, Bt_2 \Rightarrow \{ll : At, Ct\}}$$

$\boxed{\mathbf{recTyp}\ At}$ *(Valid record types)*

RT-Nil
$$\frac{}{\mathbf{recTyp}\ \top}$$

RT-Rcd
$$\frac{\mathbf{recTyp}\ Bt}{\mathbf{recTyp}\ \{ll : At, Bt\}}$$

$\boxed{\vdash At}$ *(Type wellformedness)*

WF-Nil
$$\frac{}{\vdash \top}$$

WF-Bot
$$\frac{}{\vdash \bot}$$

WF-Base
$$\frac{}{\vdash \mathbb{B}}$$

WF-Rcd
$$\frac{\begin{array}{c}\vdash At \qquad \vdash Bt \qquad \mathbf{recTyp}\ Bt\\ \mathbf{lookup}\ ll\ Bt \Rightarrow At' \ \vee\ \mathbf{lookup}\ ll\ Bt = / >\\ At \approx At'\end{array}}{\vdash \{ll : At, Bt\}}$$

WF-Arrow
$$\frac{\vdash At \qquad \vdash Bt}{\vdash At \to Bt}$$

$\boxed{At \,\&\, Bt \Rightarrow Ct}$ *(Concatenate types)*

CT-Nil
$$\frac{}{\top \,\&\, Bt \Rightarrow Bt}$$

CT-Rcd
$$\frac{\begin{array}{c}\mathbf{lookup}\ ll\ Bt_2 \Rightarrow At \ \vee\ \mathbf{lookup}\ ll\ Bt_2 = / >\\ Bt_1 \,\&\, Bt_2 \Rightarrow Ct\end{array}}{(\{ll : At, Bt_1\}) \,\&\, Bt_2 \Rightarrow \{ll : At, Ct\}}$$

$\boxed{\Gamma \vdash t : At}$ *(Target typing)*

TTyping-Base
$$\frac{\vdash \Gamma}{\Gamma \vdash b : \mathbb{B}}$$

TTyping-Var
$$\frac{\vdash \Gamma \qquad x : At \in \Gamma}{\Gamma \vdash x : At}$$

TTyping-Abs
$$\frac{\Gamma, x : At \vdash t : Bt}{\Gamma \vdash \lambda x.t : At \to Bt}$$

TTyping-Fix
$$\frac{\Gamma, x : Bt \vdash t : At \qquad At \approx Bt}{\Gamma \vdash \mathsf{fix}\, x.t : At}$$

TTyping-App
$$\frac{\begin{array}{c}\Gamma \vdash t_1 : At \to Bt\\ \Gamma \vdash t_2 : At' \qquad At \approx At'\end{array}}{\Gamma \vdash t_1\ t_2 : Bt}$$

TTyping-RcdNil
$$\frac{\vdash \Gamma}{\Gamma \vdash \{\} : \top}$$

TTyping-RcdCons
$$\frac{\begin{array}{c}\mathbf{recTyp}\ Bt\\ \mathbf{lookup}\ ll\ Bt \Rightarrow At' \ \vee\ \mathbf{lookup}\ ll\ Bt = / >\\ At \approx At'\\ \Gamma \vdash t_1 : At \qquad \Gamma \vdash t_2 : Bt\end{array}}{\Gamma \vdash (\{ll \Rightarrow t_1, t_2\}) : (\{ll : At, Bt\})}$$

TTyping-RcdProj
$$\frac{\begin{array}{c}\Gamma \vdash t : At\\ \mathbf{lookup}\ ll\ At \Rightarrow Bt\end{array}}{\Gamma \vdash t.ll : Bt}$$

TTyping-RcdMerge
$$\frac{\begin{array}{c}\mathbf{recTyp}\ At\\ \mathbf{recTyp}\ Bt \qquad \Gamma \vdash t_1 : At\\ \Gamma \vdash t_2 : Bt \qquad At \,\&\, Bt \Rightarrow Ct\end{array}}{\Gamma \vdash t_1; t_2 : Ct}$$

$\boxed{\Gamma \vdash t \Leftarrow At}$ *(Target flex typing)*

$$\textsc{TFTyping-Orig}\quad \frac{\Gamma \vdash t : At}{\Gamma \vdash t \Leftarrow At}$$

$$\textsc{TFTyping-Top}\quad \frac{\Gamma \vdash t : At}{\Gamma \vdash t \Leftarrow \top}$$

$$\textsc{TFTyping-Part}\quad \frac{\begin{array}{c}\Gamma \vdash t : At \\ \textbf{lookup}\ ll\ At \Rightarrow Bt\end{array}}{\Gamma \vdash t \Leftarrow \{ll : Bt\}}$$

$$\textsc{TFTyping-Cons}\quad \frac{\begin{array}{c}\Gamma \vdash t \Leftarrow \{ll : At\} \\ \Gamma \vdash t \Leftarrow Bt\end{array}}{\Gamma \vdash t \Leftarrow \{ll : At, Bt\}}$$

$$\textsc{TFTyping-Sim}\quad \frac{\Gamma \vdash t \Leftarrow At \qquad At \approx Bt}{\Gamma \vdash t \Leftarrow Bt}$$

2 Syntax of F_i^+

typ, A, B, C	::=		types
	|	$\top$	top type
	|	$\bot$	bottom type
	|	$\mathbb{B}$	base type
	|	$A \to B$	function types
	|	$A \,\&\, B$	intersection
	|	$\{l : A\}$	record

exp, e	::=		expressions
	|	$\top$	top value
	|	b	base literal
	|	x	variables
	|	$\lambda x.\, e : A \to B$	abstractions
	|	$\mathsf{fix}\ x.\, e : A$	fixpoint
	|	$e_1\ e_2$	applications
	|	$e_1 \,,,\, e_2$	merge
	|	$e : A$	annotation
	|	$\{l = e\}$	record
	|	$e.l$	projection

Notes 0) Fixpoint typing rule updated.
1) Type translation (from source type to string) modeled.
2) Current focus: converting source type to target type.

$\boxed{A \approx B}$ *(Types with the same index)*

$$\textsc{EI-trans}\ \frac{A \approx B \qquad B \approx C}{A \approx C} \qquad \textsc{EI-symm}\ \frac{B \approx A}{A \approx B} \qquad \textsc{EI-arrow}\ \frac{A_1 \approx A_2 \qquad B_1 \approx B_2}{A_1 \to B_1 \approx A_2 \to B_2} \qquad \textsc{EI-rcd}\ \frac{A \approx B}{\{l : A\} \approx \{l : B\}} \qquad \textsc{EI-and}\ \frac{A \approx A'}{A \,\&\, B \approx A' \,\&\, B}$$

$$\textsc{EI-comm}\ \frac{}{A \,\&\, B \approx B \,\&\, A} \qquad \textsc{EI-assoc}\ \frac{}{A \,\&\, (B \,\&\, C) \approx (A \,\&\, B) \,\&\, C} \qquad \textsc{EI-top}\ \frac{\rceil A \lceil}{A \,\&\, B \approx B} \qquad \textsc{EI-dup}\ \frac{A_1 \approx A_2}{(A_1 \,\&\, A_2) \,\&\, B \approx A_1 \,\&\, B}$$

2.1 Elaboration rules

1) All top-like values are translated into empty lists.

$$\textsc{Ela-Top}\ \frac{}{\Gamma \vdash \top \Rightarrow \top \rightsquigarrow \{\}} \qquad \textsc{Ela-TopAbs}\ \frac{\rceil B \lceil}{\Gamma \vdash \lambda x.\, e : A \to B \Rightarrow A \to B \rightsquigarrow \{\}} \qquad \textsc{Ela-TopRcd}\ \frac{\Gamma \vdash e \Rightarrow A \rightsquigarrow t \qquad \rceil A \lceil}{\Gamma \vdash \{l = e\} \Rightarrow \{l : A\} \rightsquigarrow \{\}}$$

2) All basic values (that are not merges) are translated into single-field records.

$$\textsc{Ela-Base}\ \frac{}{\Gamma \vdash b \Rightarrow \mathbb{B} \rightsquigarrow \{|\mathbb{B}| \Rightarrow b\}} \qquad \textsc{Ela-Abs}\ \frac{\Gamma, x : A \vdash e \Leftarrow B \rightsquigarrow t}{\Gamma \vdash \lambda x.\, e : A \to B \Rightarrow A \to B \rightsquigarrow \{|\overrightarrow{B}| \Rightarrow \lambda x.t\}}$$

$$\textsc{Ela-Rcd}\ \frac{\Gamma \vdash e \Rightarrow A \rightsquigarrow t}{\Gamma \vdash \{l = e\} \Rightarrow \{l : A\} \rightsquigarrow \{\{l : |A|\} \Rightarrow t\}}$$

3) Variables are directly used (unwrap the thunk) and merged terms are simply concatenated, with the expectation that $\cdot \vdash e \Rightarrow A \rightsquigarrow t$ implies $t \to^* \{|A| \Rightarrow t'\}$ (or a multi-field record) (i.e. the variable will be substituted by a term of this exact type).

$$\textsc{Ela-Var}\ \frac{x : A \in \Gamma}{\Gamma \vdash x \Rightarrow A \rightsquigarrow x} \qquad \textsc{Ela-Merge}\ \frac{\Gamma \vdash e_1 \Rightarrow A \rightsquigarrow t_1 \quad \Gamma \vdash e_2 \Rightarrow B \rightsquigarrow t_2 \quad A * B}{\Gamma \vdash e_1 \,,, \, e_2 \Rightarrow A \,\&\, B \rightsquigarrow t_1; t_2}$$

4) Projection, application, and type application make use of the property that every translated subterms are records. Application is lazy.

$$\textsc{Ela-Proj}\ \frac{\Gamma \vdash e \Rightarrow A \rightsquigarrow t_1 \quad t_1 : A; \{l\} \rightsquigarrow t_2 : B}{\Gamma \vdash e.l \Rightarrow B \rightsquigarrow t_2} \qquad \textsc{Ela-App}\ \frac{\Gamma \vdash e_1 \Rightarrow A \rightsquigarrow t_1 \quad \Gamma \vdash e_2 \Rightarrow B' \rightsquigarrow t_2 \quad t_1 : A; t_2 : B' \rightsquigarrow t_3 : C}{\Gamma \vdash e_1\, e_2 \Rightarrow C \rightsquigarrow t_3}$$

5) Annotated expressions, like fixpoints, rely on the subsumption rule to insert coercions. The coerced expression always evaluates to a record.

$$\textsc{Ela-Anno}\ \frac{\Gamma \vdash e \Leftarrow A \rightsquigarrow t}{\Gamma \vdash e : A \Rightarrow A \rightsquigarrow t} \qquad \textsc{Ela-Fix}\ \frac{\Gamma, x : A \vdash e \Leftarrow A \rightsquigarrow t}{\Gamma \vdash \mathsf{fix}\ x.\, e : A \Rightarrow A \rightsquigarrow \mathsf{fix}\, x.t} \qquad \textsc{Ela-Sub}\ \frac{\Gamma \vdash e \Rightarrow A \rightsquigarrow t_1 \quad t_1 : A <: B \rightsquigarrow t_2}{\Gamma \vdash e \Leftarrow B \rightsquigarrow t_2}$$

2.2 Coercions

$\boxed{t_1 : A <: B \rightsquigarrow t_2}$ *(Coercive subtyping)*

$$\textsc{S-Top}\ \frac{B\ \mathsf{Ordinary} \quad \rceil B \lceil}{t : A <: B \rightsquigarrow \{\}} \qquad \textsc{S-Bot}\ \frac{B\ \mathsf{Ordinary} \quad \neg\rceil B \lceil}{t : \bot <: B \rightsquigarrow \{|B| \Rightarrow \mathsf{fix}\, x.x\}} \qquad \textsc{S-Base}\ \frac{}{t : \mathbb{B} <: \mathbb{B} \rightsquigarrow \{|\mathbb{B}| \Rightarrow t.|\mathbb{B}|\}}$$

$$\textsc{S-Arrow}\ \frac{B_2\ \mathsf{Ordinary} \quad \neg\rceil B_2 \lceil \quad x : B_1 <: A_1 \rightsquigarrow t_1 \ \wedge\ (t.|A_1 \to A_2|)\ t_1 : A_2 <: B_2 \rightsquigarrow t_2}{t : A_1 \to A_2 <: B_1 \to B_2 \rightsquigarrow \{|B_1 \to B_2| \Rightarrow \lambda x.t_2\}} \qquad \textsc{S-Rcd}\ \frac{B\ \mathsf{Ordinary} \quad \neg\rceil B \lceil \quad t.|\{l : A\}| : A <: B \rightsquigarrow t_2}{t : \{l : A\} <: \{l : B\} \rightsquigarrow \{|\{l : B\}| \Rightarrow t_2\}}$$

$$\textsc{S-AndL}\ \frac{C\ \mathsf{Ordinary} \quad t : A <: C \rightsquigarrow t'}{t : A \,\&\, B <: C \rightsquigarrow t'} \qquad \textsc{S-AndR}\ \frac{C\ \mathsf{Ordinary} \quad t : B <: C \rightsquigarrow t'}{t : A \,\&\, B <: C \rightsquigarrow t'} \qquad \textsc{S-Split}\ \frac{B_1 \lhd B \rhd B_2 \quad t : A <: B_1 \rightsquigarrow t_1 \quad t : A <: B_2 \rightsquigarrow t_2 \quad t_1 : B_1 \rhd B \lhd t_2 : B_2 \rightsquigarrow t_3}{t : A <: B \rightsquigarrow t_3}$$

$\boxed{t_1 : A \rhd C \lhd t_2 : B \rightsquigarrow t}$ *(Coercive merging)*

$$\textsc{M-And}\ \frac{}{t_1 : A \rhd A \,\&\, B \lhd t_2 : B \rightsquigarrow t_1; t_2} \qquad \textsc{M-Arrow}\ \frac{(t_1.|A \to B_1|)\ x : B_1 \rhd B \lhd (t_2.|A \to B_2|)\ x : B_2 \rightsquigarrow t}{t_1 : A \to B_1 \rhd A \to B \lhd t_2 : A \to B_2 \rightsquigarrow \{|A \to B| \Rightarrow \lambda x.t\}}$$

$$\textsc{M-Rcd}\ \frac{t_1.|\{l : A_1\}| : A_1 \rhd A \lhd t_2.|\{l : A_2\}| : A_2 \rightsquigarrow t}{t_1 : \{l : A_1\} \rhd \{l : A\} \lhd t_2 : \{l : A_2\} \rightsquigarrow \{|\{l : A\}| \Rightarrow t\}}$$

$\boxed{t_1 : A; t_2 : B \rightsquigarrow t_3 : C}$ *(Distributive application)*

$$\text{A-Top}\quad \frac{\rceil A \lceil}{t_1 : A; t_2 : B \rightsquigarrow \{\} : \top}$$

$$\text{A-Arrow}\quad \frac{t_2 : C <: A \rightsquigarrow t_3}{t_1 : A \to B; t_2 : C \rightsquigarrow (t_1.|A \to B|)\ t_3 : B}$$

$$\text{A-And}\quad \frac{t : A; t_2 : C \rightsquigarrow t_3 : A' \qquad t : B; t_2 : C \rightsquigarrow t_4 : B'}{t_1 : A \,\&\, B; t_2 : C \rightsquigarrow t_3; t_4 : A' \,\&\, B'}$$

2.3 Auxiliary definitions

$$\frac{A_{k_1} < A_{k_2} < \cdots < A_{k_m} \qquad \neg \rceil A_k \lceil}{|A_1 \& A_2 \& \cdots \& A_n| = |A_{k_1}| \& |A_{k_2}| \& \cdots \& |A_{k_m}|}$$

$\boxed{\rceil A \lceil}$ *(Top-like Types)*

$$\text{TL-top}\quad \frac{}{\rceil \top \lceil} \qquad \text{TL-and}\quad \frac{\rceil A \lceil \qquad \rceil B \lceil}{\rceil A \,\&\, B \lceil} \qquad \text{TL-arr}\quad \frac{\rceil B \lceil}{\rceil A \to B \lceil} \qquad \text{TL-rcd}\quad \frac{\rceil B \lceil}{\rceil \{l : B\} \lceil}$$

$\boxed{A\ \mathsf{Ordinary}}$ *(Ordinary Types)*

$$\text{O-top}\quad \frac{}{\top\ \mathsf{Ordinary}} \qquad \text{O-int}\quad \frac{}{\mathbb{B}\ \mathsf{Ordinary}} \qquad \text{O-arrow}\quad \frac{B\ \mathsf{Ordinary}}{A \to B\ \mathsf{Ordinary}} \qquad \text{O-rcd}\quad \frac{B\ \mathsf{Ordinary}}{\{l : B\}\ \mathsf{Ordinary}}$$

$\boxed{A * B}$ *(Type Disjointness (Algorithmic))*

$$\text{D-topL}\quad \frac{}{\top * A} \qquad \text{D-topR}\quad \frac{}{A * \top} \qquad \text{D-andL}\quad \frac{A_1 * B \qquad A_2 * B}{A_1 \,\&\, A_2 * B} \qquad \text{D-andR}\quad \frac{A * B_1 \qquad A * B_2}{A * B_1 \,\&\, B_2}$$

$$\text{D-BaseArr}\quad \frac{}{\mathbb{B} * A_1 \to A_2} \qquad \text{D-ArrBase}\quad \frac{}{A_1 \to A_2 * \mathbb{B}}$$

$$\text{D-ArrArr}\quad \frac{A_2 * B_2}{A_1 \to A_2 * B_1 \to B_2} \qquad \text{D-rcdEq}\quad \frac{A * B}{\{l : A\} * \{l : B\}} \qquad \text{D-rcdNeq}\quad \frac{l_1 \neq l_2}{\{l_1 : A\} * \{l_2 : B\}}$$

$$\text{D-BaseRcd}\quad \frac{}{\mathbb{B} * \{l : A\}} \qquad \text{D-RcdBase}\quad \frac{}{\{l : A\} * \mathbb{B}}$$

$$\text{D-ArrRcd}\quad \frac{}{A_1 \to A_2 * \{l : A\}} \qquad \text{D-RcdArr}\quad \frac{}{\{l : A\} * A_1 \to A_2}$$

$\boxed{B \lhd A \rhd C}$ *(Split a Type into Two)*

$$\text{Sp-and}\quad \frac{}{(A) \lhd (A \,\&\, B) \rhd (B)} \qquad \text{Sp-arrow}\quad \frac{B_1 \lhd B \rhd B_2}{(A \to B_1) \lhd (A \to B) \rhd (A \to B_2)} \qquad \text{Sp-rcd}\quad \frac{B_1 \lhd B \rhd B_2}{\{l : B_1\} \lhd \{l : B\} \rhd \{l : B_2\}}$$

$\boxed{At <: Bt}$ *(Target Subtyping)*

$$\text{TS-top}\quad \frac{}{At <: \top} \qquad \text{TS-refl}\quad \frac{}{At <: At} \qquad \text{TS-arrow}\quad \frac{At_1 <: At_2 \qquad Bt_1 <: Bt_2}{At_1 \to Bt_1 <: At_2 \to Bt_2}$$

$$\text{TS-rcd}\quad \frac{\mathbf{recTyp}\ Ct' \qquad \mathbf{lookup}\ ll\ Ct \Rightarrow Bt \qquad Bt <: At \qquad Ct <: Ct'}{Ct <: \{ll : At, Ct'\}}$$

$\boxed{At \approx Bt}$ *(Types with the same index)*

$$\text{TEI-REFL}\quad \frac{}{At \approx At}$$

$$\text{TEI-ARROW}\quad \frac{At_1 \approx At_2 \qquad Bt_1 \approx Bt_2}{At_1 \to Bt_1 \approx At_2 \to Bt_2}$$

$$\text{TEI-RCD}\quad \frac{\begin{array}{c}\textbf{recTyp}\ Ct \\ \textbf{lookup}\ ll\ Ct = /> \ \vee\ (\textbf{lookup}\ ll\ Ct \Rightarrow At' \ \wedge\ At' \approx At) \\ At \approx Bt \qquad Ct \approx Ct'\end{array}}{\{ll : At, Ct\} \approx \{ll : Bt, Ct'\}}$$

$$\text{TEI-COMM}\quad \frac{\begin{array}{c}\textbf{recTyp}\ Ct \\ \textbf{lookup}\ ll_1\ Ct = /> \ \vee\ (\textbf{lookup}\ ll_1\ Ct \Rightarrow At' \ \wedge\ At' \approx At) \\ \textbf{lookup}\ ll_2\ Ct = /> \ \vee\ (\textbf{lookup}\ ll_2\ Ct \Rightarrow Bt' \ \wedge\ Bt' \approx Bt) \\ ll_1 \neq ll_2 \\ \{ll_2 : Bt, (\{ll_1 : At, Ct\})\} \approx Ct'\end{array}}{\{ll_1 : At, (\{ll_2 : Bt, Ct\})\} \approx Ct'}$$

$$\text{TEI-DUP}\quad \frac{\begin{array}{c}\textbf{recTyp}\ Ct \\ \textbf{lookup}\ ll\ Ct = /> \ \vee\ (\textbf{lookup}\ ll\ Ct \Rightarrow At' \ \wedge\ At' \approx At) \\ At \approx Bt \\ \{ll : Bt, (\{ll : At, Ct\})\} \approx Ct'\end{array}}{\{ll : At, (\{ll : Bt, Ct\})\} \approx Ct'}$$

```
1) type soundness of the target calculus

2) connecting the source to the target

- type translation function (from source type to string):
intersection types are flattened, their
toplike parts are filtered, and all the conjuncts are sorted (but no deduplication).

- a relation that characterizes types that corresponds to the same
string.

3) I need a function that converts source types to target types, to prove that
elaboration generates well-typed terms. There are 3 problems:
- Source types and target types are not one-to-one mapped. The typing rule for
toplike lambda functions does not split types but the subsumption rule does
(via subtyping)
\x:Int . x : Top&Top ~> { }
(\x:Int . x : Int) : Int->Top&Top ~> { ->|Top&Top| => \ x . ... }
```

Lemma eqIndTyp_sound_complete: $A \approx B$ iff $|A| = |B|$.

Lemma disjoint_type_no_eqInd: if $A \approx B$ and $A * B$ then there is a contradiction.

Lemma translation from source type to target type is a function

Lemma cosub_well_typed: if $t_1 : A <: B \rightsquigarrow t_2$ and $G|| \vdash t1 <= |[A]|$ then $G|| \vdash t2 : |[B]|$.

Theorem elaboration_well_typed: if $\Gamma \vdash e\ \mathit{dirflag}\ A \rightsquigarrow t$ then $|[G]|| \vdash t : |[A]|$.