Type Vigilance and the Truth About Transient (Techreport)

3	
4	
5	

4	
5	

Abstract

Contents

10 11 12

17 18 19

20 21 22

23

28

37 38

> 42 43 44

47 48

49 50

40 5.4 41 5.5 5.6 5.7 45 6 Surface 46 6.1 6.2 51 Author's address: 52

Contents

1 Common Definitions 2 Simple Definitions 2 1.1 1.2 **Evaluation Language Definitions** 4 1.3 **Operational Semantics** 6 1.4 Store-Based Evaluation Language Definitions 8 Store-Based Operational Semantics 1.5 10 Operational Semantics Simulation Result 1.6 13 2 Tag Typing 14 2.1 Definition 14 2.2 Simple Typing Implies Tag Typing 15 3 Truer Transient Typing 16 3.1 Definition 16 Simple Typing Implies Truer Transient Typing 3.2 18 3.3 Tag Typing Implies Truer Transient Typing 22 Vigilance 4 23 Vigilance Logical Relation 23 4.1 4.2 Vigilance Theorem 25 Vigilance Fundamental Property for Natural with Simple Typing 4.3 26 Vigilance Fundamental Property for Transient with Truer Transient Typing 4.4 46 4.5 Vigilance Fundamental Property for Transient with Tag Typing 67 5 Contextual equivalence 68 5.1 Contextual Equivalence Logical Relation—No Store 68 5.2 Context typing 69 Contextual equivalence statement 5.3 71 Binary relation-Proofs 71 Context relation—Proofs 84 Check optimization 87 Check-elision-Proofs 89 92 Simple Translation 93 Truer Transient Translation 96

2022-11-18 03:01. Page 1 of 1-100.

1

1

54 55

56

1 Common Definitions

1.1 Simple Definitions

```
Simple language
57
                                := Nat | Int | Bool | \tau \times \tau | \tau \rightarrow \tau | *
60
                  binop ∷= sum | quotient
                                := \cdot | \Gamma, (x:\tau)
                                := \mathbb{N}
64
                                ::= \mathbb{Z}
66
                \Gamma \vdash_{\mathsf{sim}} e : \tau \mid \mathsf{typing}
67
68
                     T-VAR
69
                                                                  T-Nat
                                                                                                                  T-Int
                                                                                                                                                               T-True
                                                                                                                                                                                                                      T-FALSE
                       (x_0:\tau_0)\in\Gamma_0
70
71
                                                                  \Gamma_0 \vdash_{\mathsf{sim}} n_0 : \mathsf{Nat}
                                                                                                                  \Gamma_0 \vdash_{\mathsf{sim}} i_0 : \mathsf{Int}
                                                                                                                                                                \Gamma_0 \vdash_{\mathsf{sim}} \mathsf{True} : \mathsf{Bool}
                                                                                                                                                                                                                       \Gamma_0 \vdash_{sim} \mathsf{False} : \mathsf{Bool}
                      \Gamma_0 \vdash_{\mathsf{sim}} x_0 : \tau_0
72
                                                                                                                 T-Pair
                                                                                                                                                                                      T-Cast
73
                                                                                                                            \Gamma_0 \vdash_{\mathsf{sim}} e_0 : \tau_0
                                                                                                                                                                                                       \Gamma_0 \vdash_{\mathsf{sim}} e_0 : \tau_0
74
                                 T-LAM
                                                                                                                           \Gamma_0 \vdash_{\mathsf{sim}} e_1 : \tau_1
                                        \Gamma_0, (x_0:\tau_0) \vdash_{sim} e_0:\tau_1
                                                                                                                                                                                                              \tau_0 \sim \tau_1
                                                                                                                                                                                       \Gamma_0 \vdash_{\mathsf{sim}} \mathsf{cast} \{ \tau_1 \Leftarrow \tau_0 \} \ e_0 : \tau_1
                                 \Gamma_0 \vdash_{\mathsf{sim}} \lambda(x_0 : \tau_0). e_0 : \tau_0 \rightarrow \tau_1
                                                                                                                  \Gamma_0 \vdash_{\mathsf{sim}} \langle e_0, e_1 \rangle : \tau_0 \times \tau_1
                                                                                                                                                                                                                 T-Binop
                                                                                                                                                                                                                          \Gamma_0 \vdash_{\mathsf{sim}} e_0 : \tau_0
                     Т-Арр
                            \Gamma_0 \vdash_{\mathsf{sim}} e_0 : \tau_0 \rightarrow \tau_1
                                                                                                                                                                                                                          \Gamma_0 \vdash_{\mathsf{sim}} e_1 : \tau_1
81
                                                                                        T-FsT
                                                                                                                                                   T-SND
82
                                 \Gamma_0 \vdash_{\mathsf{sim}} e_1 : \tau_0
                                                                                            \Gamma_0 \vdash_{\mathsf{sim}} e_0 : \tau_0 \times \tau_1
                                                                                                                                                         \Gamma_0 \vdash_{\mathsf{sim}} e_0 : \tau_0 \times \tau_1
                                                                                                                                                                                                                    \Delta(binop, \tau_0, \tau_1) = \tau_2
                      \Gamma_0 \vdash_{\mathsf{sim}} \mathsf{app}\{\tau_1\} \ e_0 \ e_1 : \tau_1
                                                                                         \Gamma_0 \vdash_{\mathsf{sim}} \mathsf{fst}\{\tau_0\} \, e_0 : \tau_0
                                                                                                                                                    \Gamma_0 \vdash_{\mathsf{sim}} \mathsf{snd}\{\tau_1\} \, e_0 : \tau_1
                                                                                                                                                                                                                 \Gamma_0 \vdash_{\mathsf{sim}} binop \, e_0 \, e_1 : \tau_2
                                                                       T-IF
                                                                                       \Gamma_0 \vdash_{\mathsf{sim}} e_0 : \mathsf{Bool}
                                                                                                                                                                                T-Sub
                                                                                          \Gamma_0 \vdash_{\mathsf{sim}} e_1 : \tau_0
                                                                                                                                                                                 \Gamma_0 \vdash_{\mathsf{sim}} e_0 : \tau_0
                                                                                          \Gamma_0 \vdash_{\mathsf{sim}} e_2 : \tau_0
                                                                                                                                                                                       \tau_0 \leqslant : \tau_1
                                                                        \Gamma_0 \vdash_{\mathsf{sim}} \mathsf{if} \ e_0 \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 : \tau_0
                                                                                                                                                                                 \Gamma_0 \vdash_{\mathsf{sim}} e_0 : \tau_1
91
92
               \tau \sim \tau
93
94
95
                                                                                                                        Int \sim Int
                                                                                Nat ~ Nat
98
                \tau \leqslant :\tau
99
100
101
102
                                                                                           \tau_0 \times \tau_1 \leqslant : \tau_2 \times \tau_3
                                       Nat ≼: Int
                                                                                                                                                                                                                             \tau_0 \leqslant : \tau_0
103
104
                                                                                                                                                                                                               2022-11-18 03:01. Page 2 of 1-100.
```

```
 \begin{array}{c|cccc} 105 & \Delta: binop \times \tau \times \tau \longrightarrow \tau \\ 106 & \Delta(sum, Nat, Nat) & = Nat \\ 107 & \Delta(sum, Int, Int) & = Int \\ 109 & \Delta(quotient, Nat, Nat) = Nat \\ 110 & \Delta(quotient, Int, Int) & = Int \\ 111 & 112 & \end{array}
```

1.2 Evaluation Language Definitions

```
158
                              Evaluation Language
159
160
                                                          := n \mid i \mid \text{True} \mid \text{False} \mid \langle v, v \rangle \mid w
161
                                                           := \lambda(x:\tau). e \mid \operatorname{grd} \{\tau \leftarrow \tau\} w
                                w
162
                                                           := [] | \langle E, e \rangle | \langle v, E \rangle | \text{ fst} \{\tau\} E | \text{snd} \{\tau\} E | \text{app} \{\tau\} E e | \text{app} \{\tau\} v E | E e | v E | binop E e | binop v E | binop
                                E
163
164
                                                             | cast \{\tau \leftarrow \tau'\} E | if E then e else e | mon \{\tau \leftarrow \tau\} E | assert \tau E
165
                                Err^{\circ}
                                                         ∷= Wrong
                                Err^{\bullet} ::= DivErr | TypeErr(\tau, v)
                                                         ∷= Err° | Err•
                                Err
168
169
                                                           \equiv \operatorname{Err} |x| n |i| \lambda(x:\tau). e |\langle e, e \rangle | \operatorname{app}\{\tau\} e e | e e | \operatorname{fst}\{\tau\} e | \operatorname{snd}\{\tau\} e | \operatorname{binop} e e | \operatorname{cast}\{\tau \Leftarrow \tau'\} e
170
                                                             | if e then e else e | mon \{\tau \leftarrow \tau\} e | grd \{\tau \leftarrow \tau\} e | assert \tau e
171
                                K
                                                           := Nat | Int | Bool | * \times * | * \rightarrow * | *
172
                                                           := Nat | Int | Bool | \tau \times \tau \mid \tau \to \tau \mid *
173
174
                                binop ::= sum | quotient
175
                                                           := \mathbb{N}
176
                                                           ::= \mathbb{Z}
177
178
179
                                \sim: K \times v \longrightarrow \mathbb{B}
181
182
183
                                                                                if K_0 = \text{Nat and } v_0 \in \mathbb{N}
                              v_0 \sim K_0 = \begin{cases} \text{or } K_0 = \text{Int and } v_0 \in \mathbb{Z} \\ \text{or } K_0 = \text{Bool and } v_0 \in \mathbb{B} \\ \text{or } K_0 = * \times * \text{ and } v_0 \in \langle v, v \rangle \\ \text{or } K_0 = * \to * \text{ and } v_0 \in w \\ \text{or } K_0 = * \end{cases}
184
185
186
188
189
190
191
192
                                                                                        otherwise
194
195
196
                               \delta: binop×v×v \longrightarrow e
197
                                                                                                            i_0 + i_1
198
199
                                                                                                               if binop = sum\{\tau\}
200
                                                                                                            DivErr
201
                                                                                                             if binop = quotient\{\tau\}
                                \delta(\textit{binop}, i_0, i_1) = \left\{ \right.
202
                                                                                                              and i_1 = 0
203
                                                                                                           \lfloor i_0/i_1 \rfloor
204
205
                                                                                                            if binop = quotient\{\tau\}
207
```

209	$\sim_{pos}^{L}: \tau \times v \longrightarrow \mathbb{B}$			
210	1	1		
211	L	$v \sim_{bnd}^{L} \tau$	$v \sim_{mon}^{L} \tau$	$v \sim_{check}^{L} \tau$
212	Ν	$v \sim \lfloor \tau \rfloor$	$v \sim \lfloor \tau \rfloor$	True
213	Т	$v \sim \lfloor \tau \rfloor$	True	$v \sim \tau $
214				

```
1.3 Operational Semantics
```

```
262
                            reflexive-transitive closure of \longrightarrow_L
263
264
                \longrightarrow_L compatible closure of \hookrightarrow_L
265
266
               e \mapsto_L e
267
268
269
                \mathsf{fst}\{\tau_0\}\,v_0
                                                              \rightarrowtail_L Wrong
270
                     if v_0 \neq \langle v_1, v_2 \rangle
271
272
                fst\{\tau_0\}\langle v_0,v_1\rangle
                                                             \rightarrow_L assert \tau_0 v_0
273
274
275
                \operatorname{snd}\{\tau_0\} v_0
                                                             \rightarrow_L Wrong
276
                     if v_0 \neq \langle v_1, v_2 \rangle
277
278
279
                \operatorname{snd}\{\tau_0\}\langle v_0, v_1\rangle
                                                            \rightarrowtail_L assert \tau_0 v_1
280
281
                binop\,v_0\,v_1
                                                              \rightarrowtail_L Wrong
282
                     if \delta(binop, v_0, v_1) is undefined
283
285
                                                              \mapsto_L assert \tau_0 \, \delta(binop, v_0, v_1)
                binop v_0 v_1
286
                     if \delta(binop, v_0, v_1) is defined
287
288
289
                app\{\tau_0\} v_0 v_1
                                                              \mapsto_L assert \tau_0 (v_0 \ v_1)
290
291
                v_0 v_1
                                                              \rightarrowtail_L Wrong
292
                     if v_0 \neq w_0
293
294
295
                (\lambda(x_0:\tau_1).e_0)v_1
                                                            \mapsto_L e_0[x_0 \leftarrow v_1]
                     if v_1 \sim_{check}^L \tau_1
298
299
                                                           \rightarrowtail_L TypeErr(\tau_1, v_1)
                (\lambda(x_0:\tau_1).e_0)v_1
300
                     if \neg v_1 \sim_{check}^L \tau_1
301
302
                (\operatorname{grd}\left\{\tau_{1} \Leftarrow \tau_{2}\right\}w_{0}) \ v_{1} \ \rightarrowtail_{L} \ \operatorname{mon}\left\{\operatorname{cod}(\tau_{1}) \Leftarrow \operatorname{cod}(\tau_{2})\right\} \left(w_{0} \ (\operatorname{mon}\left\{\operatorname{dom}(\tau_{2}) \Leftarrow \operatorname{dom}(\tau_{1})\right\}v_{1})\right)
303
304
305
                \mathsf{cast} \left\{ \tau_1 \leftarrow \tau_0 \right\} v_0
                                                            \mapsto_L \mod\{\tau_1 \leftarrow \tau_0\} v_0
306
                    if v_0 \sim_{bnd}^{L} \tau_1
and v_0 \sim_{bnd}^{L} \tau_0
307
308
309
```

```
\mathsf{cast} \left\{ \tau_1 \leftarrow \tau_0 \right\} v_0
                                                                 \rightarrowtail_L TypeErr(\tau_1, v_0)
313
314
                        if \neg v_0 \sim_{bnd}^L \tau_1
315
316
                   \mathsf{cast}\left\{\tau_1 \leftarrow \tau_0\right\} v_0
                                                               \rightarrowtail_L TypeErr(\tau_0, v_0)
317
                        if \neg v_0 \sim_{hnd}^L \tau_0
318
319
320
                  \begin{aligned} & \text{mon} \left\{ \tau_1 \leftarrow \tau_2 \right\} i_0 & \rightarrowtail_L i_0 \\ & \text{if } i_0 \sim^L_{mon} \tau_1 \wedge i_0 \sim^L_{mon} \tau_2 \end{aligned}
321
322
323
324
                   \mathsf{mon}\left\{\tau_1 \leftarrow \tau_2\right\} \left\langle v_0, v_1 \right\rangle \ \rightarrowtail_L \ \left\langle \mathsf{mon}\left\{ \mathit{fst}(\tau_1) \leftarrow \mathit{fst}(\tau_2) \right\} v_0, \mathsf{mon}\left\{ \mathit{snd}(\tau_1) \leftarrow \mathit{snd}(\tau_2) \right\} v_1 \right\rangle
325
326
                   \operatorname{mon}\left\{\tau_{1} \leftarrow \tau_{2}\right\} w \qquad \qquad \rightarrowtail_{L} \operatorname{grd}\left\{\tau_{1} \leftarrow \tau_{2}\right\} w
327
                        if w \sim_{mon}^{L} \tau_1 \wedge w \sim_{mon}^{L} \tau_2
328
329
330
                  \mathsf{mon}\left\{\tau_0 \leftarrow \tau_1\right\} v_0 \qquad \qquad \rightarrowtail_L \; \mathsf{TypeErr}(\tau_0, \, v_0)
331
                        if \neg v_0 \sim_{mon}^L \tau_0
332
333
334
                                                                 \rightarrowtail_L TypeErr(\tau_1, v_0)
                   \mathsf{mon}\left\{\tau_0 \leftarrow \tau_1\right\} v_0
335
                        if \neg v_0 \sim_{mon}^L \tau_1
336
337
                   if True then e_1 else e_2 \implies_L e_1
338
339
340
                   if False then e_1 else e_2 \rightarrowtail_L e_2
341
342
343
                   assert \tau_0 v_0
                                                                         \mapsto_L v_0
                        if v_0 \sim^L_{check} \tau_0
344
345
346
                                                                       \rightarrowtail_L TypeErr(\tau_0, v_0)
                   assert \tau_0 v_0
347
                        if \neg v_0 \sim_{check}^L \tau_0
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
```

416

1.4 Store-Based Evaluation Language Definitions

```
366
              Store-Based Evaluation Language
367
368
                           := \ell \mid n \mid i \mid \text{True} \mid \text{False} \mid \langle \ell, \ell \rangle \mid \lambda(x : \tau). e
               Err° ∷= Wrong
370
               Err^{\bullet} ::= DivErr | TypeErr(\tau, v)
371
372
               Err ∷= Err° | Err•
373
                           := \operatorname{Err} |x| \ell |v| \langle e, e \rangle | \operatorname{app} \{\tau\} e e | e e | \operatorname{fst} \{\tau\} e | \operatorname{snd} \{\tau\} e | \operatorname{binop} e e | \operatorname{cast} \{\tau \leftarrow \tau'\} e
374
                            | if e then e else e | mon \{\tau \leftarrow \tau\} e | assert \tau e
                           := Nat | Int | Bool | * \times * | * \rightarrow * | *
               K
376
377
                           := Nat | Int | Bool | \tau \times \tau | \tau \rightarrow \tau | *
378
               binop ∷= sum | quotient
379
                            \in \mathbb{L} \mapsto \mathbb{V} \times option(\mathbb{T} \times \mathbb{T})
380
                            \in \mathbb{L}
381
382
                            \in \mathbb{N}
               n
383
                            \in \mathbb{Z}
               i
384
                           385
                            | cast \{\tau \leftarrow \tau'\} E | if E then e else e | mon \{\tau \leftarrow \tau\} E | assert \tau E
386
387
389
              \sim: K \times \mathbb{V} \longrightarrow \mathbb{B}
390
             v_0 \sim K_0 = \begin{cases} \text{if } K_0 = \text{Nat and } v_0 \in \mathbb{N} \\ \text{or } K_0 = \text{Int and } v_0 \in \mathbb{Z} \\ \text{or } K_0 = \text{Bool and } v_0 \in \mathbb{B} \\ \text{or } K_0 = * \times * \text{ and } v_0 \in \langle \ell, \ell \rangle \\ \text{or } K_0 = * \to * \text{ and } v_0 \in \lambda(x : \tau). e \end{cases}
391
392
393
394
396
397
398
399
                                         otherwise
401
402
403
404
              \delta: binop \times \mathbb{V} \times \mathbb{V} \longrightarrow \mathbb{E}
405
                                                  i_0 + i_1
406
407
                                                   if binop = sum\{\tau\}
408
                                                  DivErr
409
                                                  if binop = quotient\{\tau\}
              \delta(binop, i_0, i_1) = \left\{ \right.
410
                                                  and i_1 = 0
411
                                              \lfloor i_0/i_1 \rfloor
412
413
                                               if binop = quotient\{\tau\}
414
                                                    and i_1 \neq 0
415
```

$$\mathsf{pointsto}(\Sigma, \ell)$$

$$\mathsf{pointsto}(\Sigma, \ell) = \begin{cases} fst(\Sigma(\ell)) \\ & \text{if } fst(\Sigma(\ell)) \neq \ell' \\ & \text{pointsto}(\Sigma, \ell') \\ & \text{if } fst(\Sigma(\ell)) = \ell' \end{cases}$$

1.5 Store-Based Operational Semantics

```
\boxed{\longrightarrow_L^*} \text{reflexive-transitive closure of} \longrightarrow_L
```

$$\longrightarrow_L$$
 compatible closure of \hookrightarrow_L

$$\Sigma, e \hookrightarrow_L \Sigma, e$$

$$\Sigma, v \hookrightarrow_L \Sigma[\ell \mapsto (v, \mathsf{none})], \ell$$

where $loc \notin dom(\Sigma)$

481
$$\Sigma$$
, fst $\{\tau_0\}$ $\ell_0 \hookrightarrow_L \Sigma$, Wrong
482 if $\Sigma(\ell_0) \neq (\langle \ell_1, \ell_2 \rangle, _)$

$$\Sigma$$
, fst $\{\tau_0\}$ ℓ_0 \hookrightarrow_L Σ , assert τ_0 ℓ_0 if $\Sigma(\ell_0) = (\langle \ell_1, \ell_2 \rangle, _)$

$$\Sigma$$
, snd $\{\tau_0\}$ $\ell_0 \hookrightarrow_L \Sigma$, Wrong if $\Sigma(\ell_0) \neq (\langle \ell_1, \ell_2 \rangle, _)$

$$\Sigma, \operatorname{snd}\{\tau_0\} \ \ell_0 \qquad \hookrightarrow_L \quad \Sigma, \operatorname{assert} \tau_0 \ \ell_0$$
$$\operatorname{if} \ \Sigma(\ell_0) = (\langle \ell_1, \ell_2 \rangle, _)$$

$$\begin{array}{ll} \Sigma, \mathit{binop}\,\ell_0\,\ell_1 & \hookrightarrow_L \; \Sigma, \mathsf{Wrong} \\ & \text{if } \delta(\mathit{binop}, \mathsf{pointsto}(\Sigma,\ell_0), \mathsf{pointsto}(\Sigma,\ell_1)) \text{ is undefined} \end{array}$$

$$\Sigma$$
, $binop \ \ell_0 \ \ell_1 \longrightarrow_L \Sigma$, assert $\tau_0 \ \delta(binop, \mathsf{pointsto}(\Sigma, \ell_0), \mathsf{pointsto}(\Sigma, \ell_1))$ if $\delta(binop, \mathsf{pointsto}(\Sigma, \ell_0), \mathsf{pointsto}(\Sigma, \ell_1))$ is defined

$$\Sigma$$
, app $\{\tau_0\}$ ℓ_0 ℓ_1 \hookrightarrow_L Σ , assert τ_0 $(\ell_0$ $\ell_1)$

$$\begin{split} \Sigma, \ell_0 \ \ell_1 & \hookrightarrow_L \ \Sigma, \mathsf{Wrong} \\ & \text{if } \Sigma(\ell_0) = (v, _) \ \text{and} \ v \notin \lambda(x \colon \tau). \ e \cup \ell \\ & \text{or } \Sigma(\ell_0) = (\ell'_0, \mathsf{none}) \end{split}$$

$$\begin{split} & \Sigma, \ell_0 \ \ell_1 & \hookrightarrow_L \ \Sigma, e_0[x_0 \leftarrow \ell_1] \\ & \text{if } \Sigma(\ell_0) = (\lambda(x_0 \colon \tau_1).\ e_0, _) \text{ and} \\ & \text{pointsto}(\Sigma, \ell_1) \sim^L_{check} \tau_1 \end{split}$$

```
\Sigma, \ell_0 \ell_1
                                                              \hookrightarrow_L \Sigma, TypeErr(\tau_1, \ell_1)
521
522
                     if \Sigma(\ell_0) = (\lambda(x_0 : \tau_1). e_0, \_) and
523
                     \neg pointsto(\Sigma, \ell_1) \sim_{check}^{L} \tau_1
524
525
526
                \Sigma, \ell_0 \ell_1
                                                              \hookrightarrow_{I} \Sigma, mon \{cod(\tau_1) \Leftarrow cod(\tau_2)\}\ (\ell_0 \ (mon \{dom(\tau_2) \Leftarrow dom(\tau_1)\}\ \ell_1))
527
                     if \Sigma(\ell_0) = (\ell_2, some(\tau_1, \tau_2))
528
529
                \Sigma, cast \{\tau_1 \leftarrow \tau_0\} \ell_0 \longrightarrow_L \Sigma, mon \{\tau_1 \leftarrow \tau_0\} \ell_0
530
                     if pointsto(\Sigma, \ell_0) \sim_{\textit{bnd}}^{L} \tau_1
531
532
                     and pointsto(\Sigma, \ell_0) \sim_{hnd}^L \tau_0
533
534
                \Sigma, cast \{\tau_1 \leftarrow \tau_0\} \ \ell_0 \quad \hookrightarrow_L \ \Sigma, Type\mathsf{Err}(\tau_1, \ \ell_0)
535
                     if \neg pointsto(\Sigma, \ell_0) \sim_{bnd}^L \tau_1
536
537
538
                \Sigma, cast \{\tau_1 \leftarrow \tau_0\} \ \ell_0 \quad \hookrightarrow_L \ \Sigma, TypeErr(\tau_0, \ \ell_0)
539
                     if \neg pointsto(\Sigma, \ell_0) \sim_{bnd}^{L} \tau_0
540
541
542
                \Sigma, \mathsf{mon}\, \{\tau_1 \Leftarrow \tau_2\}\, \ell_0 \quad \hookrightarrow_L \ \Sigma[\ell_1 \mapsto (\ell_0, \mathsf{some}(\tau_1, \tau_2))], \ell_1
543
                     if \ell_1 \notin dom(\Sigma)
544
                     and pointsto(\Sigma, \ell_0) = v where v = i or True or False
545
                     and v \sim_{mon}^{L} \tau_1 \wedge v \sim_{mon}^{L} \tau_2
546
547
548
                \Sigma, mon \{\tau_1 \leftarrow \tau_2\} \ell_0 \longrightarrow_L \Sigma, \{\text{mon } \{fst(\tau_1) \leftarrow fst(\tau_2)\} \ell_1, \text{mon } \{snd(\tau_1) \leftarrow snd(\tau_2)\} \ell_2 \}
549
                     if \Sigma(\ell_0) = (\langle \ell_1, \ell_2 \rangle, \_)
550
551
552
                \Sigma, \mathsf{mon}\, \{\tau_1 \Leftarrow \tau_2\}\, \ell_0 \quad \hookrightarrow_L \ \Sigma[\ell_1 \mapsto (\ell_0, \mathsf{some}(\tau_1, \tau_2))], \ell_1
553
                     if \ell_1 \notin dom(\Sigma)
554
                     and pointsto(\Sigma, \ell_0) = v and v = \lambda(x_0 : \tau_1). e_0
555
                     and v \sim_{mon}^{L} \tau_1 \wedge v \sim_{mon}^{L} \tau_2
556
557
558
                \Sigma, mon \{\tau_0 \leftarrow \tau_1\} \ell_0 \hookrightarrow_L \Sigma, TypeErr(\tau_1, \ell_0)
559
                     if \neg pointsto(\Sigma, \ell_0) \sim_{mon}^{L} \tau_1
560
561
562
                \Sigma, mon \{\tau_0 \Leftarrow \tau_1\} \ell_0 \hookrightarrow_L \Sigma, Type\mathsf{Err}(\tau_0, \ell_0)
563
                     if \neg pointsto(\Sigma, \ell_0) \sim_{mon}^L \tau_0
564
565
                \Sigma, if \ell_0 then e_1 else e_2 \hookrightarrow_L \Sigma, e_1
566
567
                     if pointsto(\Sigma, \ell_0) = True
568
569
                \Sigma, if \ell_0 then e_1 else e_2 \hookrightarrow_L \Sigma, e_2
570
571
                     if pointsto(\Sigma, \ell_0) = False
572
              2022-11-18 03:01. Page 11 of 1-100.
```

```
\Sigma, if \ell_0 then e_1 else e_2 \hookrightarrow_L \Sigma, Wrong
573
                     if pointsto(\Sigma, \ell_0) \neq \ell or True or False
574
575
576
                     assert \tau_0 \ell_0 \qquad \hookrightarrow_L \quad \Sigma, \ell_0
if pointsto(\Sigma, \ell_0) \sim_{check}^L \tau_0
                \Sigma, assert \tau_0 \ell_0
577
578
579
                     assert \tau_0 \, \ell_0 \qquad \hookrightarrow_L \; \Sigma, \mathsf{TypeErr}(\tau_0, \, \ell_0) if \neg \mathsf{pointsto}(\Sigma, \ell_0) \sim^L_{check} \, \tau_0
580
                \Sigma, assert \tau_0 \ell_0
581
582
584
585
586
587
588
589
590
591
592
593
594
597
598
599
600
601
602
604
605
606
607
610
611
612
613
614
615
617
618
619
620
```

1.6 Operational Semantics Simulation Result

To compare the two semantics, we have to define a relation that compares values between the two languages. The store semantics will represent:

- (1) Guards as a linked list of pairs of types, ending at a lambda with no types.
- (2) Pairs as a pointer to the two subcomponents, with no types.
- (3) Base values as a linked list of pairs of types, ending at a base value with no types.

We capture this in the following value equivalence:

$$(\Sigma, \ell) \equiv v$$

$$\begin{split} & \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, _) \\ & (\Sigma, \ell_1) \equiv v_1 \\ & (\Sigma, \ell_1) \equiv v_1 \\ & (\Sigma, \ell_2) \equiv v_2 \\ & \ell \equiv v \end{split} \qquad \begin{aligned} & \Sigma(\ell) = (\ell', \mathsf{some}(\tau', \tau)) \\ & (\Sigma, \ell) \equiv v \\ & (\Sigma, \ell') \equiv v \\ & (\Sigma, \ell) \equiv \mathsf{grd}\left\{\tau' \Leftarrow \tau\right\} v \end{aligned} \qquad \begin{aligned} & \Sigma(\ell) = (\lambda x : \tau. e, _) \\ & (\Sigma, \ell) \equiv \lambda x : \tau. e \end{aligned}$$

Theorem 1.1 (Store and Non Store Operational Semantics are Equivalent). $e \longrightarrow_L^* e'$ and e' is irreducible iff $\forall \Sigma$. $\exists \Sigma', \ell$. $(\Sigma, e) \longrightarrow_L^* (\Sigma', \ell)$ and $(\Sigma', \ell) \equiv e'$

678 679

727 728

2 Tag Typing

2.1 Definition

```
680
                                         Simple language
681
682
                                                                                = x \mid n \mid i \mid \text{True} \mid \text{False} \mid \lambda(x:K). e \mid \langle e, e \rangle \mid \text{app}\{K\} \mid e \mid \text{fst}\{K\} \mid e \mid binopee \mid \text{cast}\{K \leftarrow K\} \mid e \mid \text{if } e \mid \text{then } e \mid \text{else } e \mid \text{fst}\{K\} \mid e \mid binopee \mid \text{cast}\{K \leftarrow K\} \mid e \mid \text{if } e \mid \text{then } e \mid \text{false} \mid h \mid e \mid h \mid 
683
                                                                                := Nat | Int | Bool | * \times * | * \rightarrow * | *
684
                                             binop ∷= sum | quotient
685
                                                                                := \cdot | \Gamma, (x:K_0)
                                                                                := \mathbb{N}
                                                                                ::= \mathbb{Z}
689
690
                                          \Gamma \vdash_{\mathsf{tag}} e : \tau \mid \mathsf{typing}
691
692
                                                       T-VAR
                                                                                                                                                                        T-Nat
                                                                                                                                                                                                                                                                                               T-Int
                                                                                                                                                                                                                                                                                                                                                                                                                T-True
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         T-False
693
                                                           (x_0:K_0)\in\Gamma_0
                                                                                                                                                                                                                                                                                                                                                                                                                \Gamma_0 \vdash_{tag} \mathsf{True} : \mathsf{Bool}
                                                        \Gamma_0 \vdash_{\mathsf{tag}} x_0 : K_0
                                                                                                                                                                        \Gamma_0 \vdash_{\mathsf{tag}} n_0 : \mathsf{Nat}
                                                                                                                                                                                                                                                                                                 \Gamma_0 \vdash_{tag} i_0 : Int
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           \Gamma_0 \vdash_{tag} \mathsf{False} : \mathsf{Bool}
695
696
                                                                                                                                                                                                                                                                                            T-Pair
                                                                                                                                                                                                                                                                                                                                                                                                                                                             T-CAST
697
                                                                                                                                                                                                                                                                                                             \Gamma_0 \vdash_{\mathsf{tag}} e_0 : K_0
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           \Gamma_0 \vdash_{\mathsf{tag}} e_0 : K_0
                                                                                      T-Lam
698
                                                                                                  \Gamma_0, (x_0:K_0) \vdash_{tag} e_0:K_1
                                                                                                                                                                                                                                                                                                             \Gamma_0 \vdash_{\mathsf{tag}} e_1 : K_1
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          K_0 \sim K_1
                                                                                                                                                                                                                                                                                            \Gamma_0 \vdash_{\mathsf{tag}} \langle e_0, e_1 \rangle : * \times *
                                                                                                                                                                                                                                                                                                                                                                                                                                                              \Gamma_0 \vdash_{\mathsf{tag}} \mathsf{cast} \{ K_1 \Leftarrow K_0 \} \ e_0 : K_1
                                                                                        \Gamma_0 \vdash_{\mathsf{tag}} \lambda(x_0 : K_0). e_0 : * \rightarrow *
701
702
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               T-Binop
703
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       \Gamma_0 \vdash_{\mathsf{tag}} e_0 : K_0
                                                 T-App
704
                                                                          \Gamma_0 \vdash_{\mathsf{tag}} e_0 : * \rightarrow *
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      \Gamma_0 \vdash_{\mathsf{tag}} e_1 : K_1
                                                                                                                                                                                                                            T-FsT
                                                                                                                                                                                                                                                                                                                                                                                  T-Snd
705
                                                                                                                                                                                                                            \Gamma_0 \vdash_{\mathsf{tag}} e_0 : * \times *
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  \Delta(\mathit{binop}, K_0, K_1) = K_2
                                                                               \Gamma_0 \vdash_{\mathsf{tag}} e_1 : K_0
                                                                                                                                                                                                                                                                                                                                                                                                 \Gamma_0 \vdash_{\mathsf{tag}} e_0 : * \times *
706
707
                                                                                                                                                                                                                          \Gamma_0 \vdash_{\mathsf{tag}} \mathsf{snd}\{K_1\} e_0 : K_1
                                                   \Gamma_0 \vdash_{\mathsf{tag}} \mathsf{app}\{K_1\} e_0 e_1 : K_1
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          \Gamma_0 \vdash_{\mathsf{tag}} binop \, e_0 \, e_1 : K_2
708
709
                                                                                                                                                                                T-IF
710
                                                                                                                                                                                                                            \Gamma_0 \vdash_{\mathsf{tag}} e_0 : \mathsf{Bool}
                                                                                                                                                                                                                                                                                                                                                                                                                                                          T-Sub
711
                                                                                                                                                                                                                                \Gamma_0 \vdash_{\mathsf{tag}} e_1 : K_0
                                                                                                                                                                                                                                                                                                                                                                                                                                                            \Gamma_0 \vdash_{\mathsf{tag}} e_0 : K_0
                                                                                                                                                                                                                                \Gamma_0 \vdash_{\mathsf{tag}} e_2 : K_0
                                                                                                                                                                                                                                                                                                                                                                                                                                                                        K_0 \leqslant : K_1
714
                                                                                                                                                                                  \Gamma_0 \vdash_{\mathsf{tag}} \mathsf{if} \ e_0 \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 : K_0
                                                                                                                                                                                                                                                                                                                                                                                                                                                           \Gamma_0 \vdash_{\mathsf{tag}} e_0 : K_1
715
```

2.2 Simple Typing Implies Tag Typing

 e^+

```
i^+
                                                  =i
b^+
                                                  = b
\langle e_1, e_2 \rangle^+
                                                 =\langle e_1^+, e_2^+ \rangle
(\lambda x : \tau. e)^+
                                                 =\lambda x: \lfloor \tau \rfloor. e^+
(\mathsf{app}\{\tau\}\,e_1\;e_2)^+
                                                = \operatorname{app}\{\lfloor \tau \rfloor\} e_1^+ e_2^+
(\operatorname{fst}\{\tau\} e)^+
                                                 = \operatorname{fst}\{\lfloor \tau \rfloor\} e^+
(\operatorname{snd}\{\tau\} e)^+
                                                = \operatorname{snd}\{\lfloor \tau \rfloor\} e^+
(binop e_1 e_2)^+ = binop e_1^+ e_2^+
(\operatorname{cast} \{\tau' \Leftarrow \tau\} e)^+ = \operatorname{cast} \{\lfloor \tau' \rfloor \Leftarrow \lfloor \tau \rfloor\} e^+
(if e_1 then e_2 else e_3)<sup>+</sup> = if e_1<sup>+</sup> then e_2<sup>+</sup> else e_3<sup>+</sup>
```

 Γ^+

$$(\Gamma, x : \tau)^+ = \Gamma^+, x : \lfloor \tau \rfloor$$

Theorem 2.1 (Simple Typing Implies Tag Typing). If $\Gamma \vdash_{sim} e : \tau \ then \ \Gamma^+ \vdash_{tag} e^+ : \lfloor \tau \rfloor$.

Proof. By induction over the typing derivation. The typing rules have a one to one correspondance, so each case follows by the induction hypothesis. \Box

782 783

828 829

3 Truer Transient Typing

3.1 Definition

```
784
                                         Simple language
785
786
                                                                                = x \mid n \mid i \mid \text{True} \mid \text{False} \mid \lambda(x:K).e \mid \langle e, e \rangle \mid \text{app}\{K\}\ e \mid \text{fst}\{K\}\ e \mid \text{snd}\{K\}\ e \mid \text{binope}\ e \mid \text{cast}\ \{K \leftrightharpoons K\}\ e \mid \text{if}\ e \text{ then } e \text{ else } e \mid \text{fst}\{K\}\ e \mid \text{binope}\ e \mid \text{cast}\ \{K \leftrightharpoons K\}\ e \mid \text{for}\ e \mid \text{for}
787
                                                                                ::=  Nat | Int | Bool | \tau \times \tau | * \rightarrow \tau | * | \bot
788
                                                                                := Nat | Int | Bool | * \times * | * \rightarrow * | *
789
                                            binop ∷= sum | quotient
                                                                                := \cdot | \Gamma, (x:K_0)
792
                                                                                := \mathbb{N}
793
                                                                                ::= \mathbb{Z}
794
795
                                          \lfloor \tau \rfloor tag of
796
                                            [Int]
                                                                                                   = Int
797
                                            [Nat]
                                                                                                   = Nat
798
799
                                            |Bool|
                                                                                                   = Bool
800
                                             \lfloor \tau \times \tau' \rfloor = * \times *
801
                                             \lfloor * \to \tau' \rfloor = * \to *
802
                                            [*]
                                                                                              = *
                                         \sqcup,\sqcap:\tau\times\tau\longrightarrow\tau
806
                                                                                                                                                                                                 if \tau = *
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     if \tau = \bot
807
                                                                                                                                                                                                  or \tau' = *
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     or \tau' = \bot
808
809
                                                                                                                                                                                                 or \lfloor \tau \rfloor \neq \lfloor \tau' \rfloor
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     or \lfloor \tau \rfloor \neq \lfloor \tau' \rfloor
810
                                                                                                                                                                                                                and \tau \neq \bot and \tau' \neq \bot
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   and \tau \neq * and \tau' \neq *
811
                                                                                                        τ
                                                                                                                                                                                                                                                                                                                                                                                                                                          τ
812
                                                                                                                                                                                                 if \tau' = \bot
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     if \tau' = *
813
814
815
                                                                                                                                                                                                 if \tau = \bot
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     if \tau = *
816
817
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     if \tau = \text{Nat} and \tau' = \text{Int}
                                                                                                                                                                                                 if \tau = \text{Nat} and \tau' = \text{Int}
818
819
                                                                                                                                                                                                 or \tau = \text{Int and } \tau' = \text{Nat}
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     or \tau = \text{Int and } \tau' = \text{Nat}
820
821
822
823
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  if \tau = \tau_1 \times \tau_2 and \tau' = \tau'_1 \times \tau'_2
                                                                                                                                                                                        if \tau = \tau_1 \times \tau_2 and \tau' = \tau'_1 \times \tau'_2
825
826
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     if \tau = * \rightarrow \tau_2 and \tau' = * \rightarrow \tau'_2
827
```

```
\Gamma \vdash_{\mathsf{tru}} e : \tau typing
833
834
                         T-Var
835
                                                                                                                                                                                                                                            T-False
                                                                           T-Nat
                                                                                                                               T-Int
                                                                                                                                                                                 T-True
                          (x_0:K_0)\in\Gamma_0
836
837
                                                                                                                                                                                 \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{True} : \mathsf{Bool}
                         \Gamma_0 \vdash_{\mathsf{tru}} x_0 : K_0
                                                                                                                                \Gamma_0 \vdash_{\mathsf{tru}} i_0 : \mathsf{Int}
                                                                                                                                                                                                                                              \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{False} : \mathsf{Bool}
                                                                           \Gamma_0 \vdash_{\mathsf{tru}} n_0 : \mathsf{Nat}
838
839
                                                                                                                T-Pair
840
                                                                                                                            \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0
                            T-LAM
                                                                                                                                                                                       T-Cast
841
                                   \Gamma_0, (x_0:K_0) \vdash_{\mathsf{tru}} e_0: \tau_1
                                                                                                                           \Gamma_0 \vdash_{\mathsf{tru}} e_1 : \tau_1
                                                                                                                                                                                                                       \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0
842
843
                                                                                                                                                                                        \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{cast} \{ K_1 \Leftarrow K_0 \} \ e_0 : K_1 \sqcap K_0 \sqcap \tau_0
                             \Gamma_0 \vdash_{\mathsf{tru}} \lambda(x_0 : K_0). e_0 : * \rightarrow \tau_1
                                                                                                                 \Gamma_0 \vdash_{\mathsf{tru}} \langle e_0, e_1 \rangle : \tau_0 \times \tau_1
844
845
                Т-Арр
                                                                                                  Т-АррВот
                               \Gamma_0 \vdash_{\mathsf{tru}} e_0 : * \rightarrow \tau_1
                                                                                                                \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \bot
846
                                                                                                                                                                       T-FsT
                                                                                                                                                                                                                                                 Т-FsтВот
847
                                     \Gamma_0 \vdash_{\mathsf{tru}} e_1 : \tau_0'
                                                                                                                \Gamma_0 \vdash_{\mathsf{tru}} e_1 : \tau_0'
                                                                                                                                                                                   \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0 \times \tau_1
                                                                                                                                                                                                                                                           \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \bot
848
                                                                                                  \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{app}\{K_1\} e_0 e_1 : \bot
                                                                                                                                                                                                                                                 \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{fst}\{K_0\} e_0 : \bot
                 \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{app}\{K_1\} \ e_0 \ e_1 : K_1 \sqcap \tau_1
                                                                                                                                                                       \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{fst}\{K_0\} \, e_0 : K_0 \sqcap \tau_0
849
850
                                                                                                                                                                                                T-Binop
851
                                                                                                                                                                                                                         \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0
                                  T-Snd
                                                                                                                       T-SndBot
852
                                                                                                                                  \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \bot
                                              \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0 {\times} \tau_1
                                                                                                                                                                                                                         \Gamma_0 \vdash_{\mathsf{tru}} e_1 : \tau_1
853
854
                                                                                                                       \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{snd}\{K_1\} \ e_0 : \bot
                                  \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{snd}\{K_1\} e_0 : K_1 \sqcap \tau_1
                                                                                                                                                                                                \Gamma_0 \vdash_{\mathsf{tru}} binop \, e_0 \, e_1 : \Delta(binop, \tau_0, \tau_1)
855
                                     T-IF
                                                                                                                                               Т-ІғВот
857
                                                             \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \mathsf{Bool}
                                                                                                                                                                    \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \bot
                                                                                                                                                                                                                                              T-Sub
858
                                                               \Gamma_0 \vdash_{\mathsf{tru}} e_1 : \tau_0
                                                                                                                                                                    \Gamma_0 \vdash_{\mathsf{tru}} e_1 : \tau_0
                                                                                                                                                                                                                                              \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0
859
                                                               \Gamma_0 \vdash_{\mathsf{tru}} e_2 : \tau_1
                                                                                                                                                                   \Gamma_0 \vdash_{\mathsf{tru}} e_2 : \tau_1
                                                                                                                                                                                                                                                     \tau_0 \leqslant : \tau_1
860
861
                                      \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{if} \ e_0 \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 : \tau_0 \sqcup \tau_1
                                                                                                                                             \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{if} \ e_0 \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 : \bot
                                                                                                                                                                                                                                              \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_1
862
                  \Delta : binop \times \tau \times \tau \longrightarrow \tau
864
                   \Delta(sum, Nat, Nat)
                                                                      = Nat
865
866
                   \Delta(sum, Int, Int)
                                                                       = Int
867
                   \Delta(quotient, Nat, Nat) = Nat
868
                   \Delta(quotient, Int, Int) = Int
869
                   \Delta(binop, \perp, \tau)
                                                                      = \bot if \tau = Nat or Int or \bot
870
871
                   \Delta(binop, \tau, \bot)
                                                                      = \bot if \tau = Nat or Int or \bot
872
                  \tau \leq \tau
873
874
875
                                       \tau_0 \leqslant : \tau_1
                                                                                                               \tau_1 \leq \tau_3
876
```

 $* \rightarrow \tau_0 \le * \rightarrow \tau_1$

 $\perp \leq \tau$

 $\tau \leq *$

 $\tau_0 \leq \tau_1$

 $\tau_0 \times \tau_1 \le \tau_2 \times \tau_3$

3.2 Simple Typing Implies Truer Transient Typing

 e^+

i ⁺	=i
b^+	= b
$\langle e_1, e_2 \rangle^+$	$=\langle e_1^+,e_2^+\rangle$
$(\lambda x:\tau.e)^+$	$= \lambda x : \lfloor \tau \rfloor . e^+$
$(app\{\tau\}e_1e_2)^+$	$= \operatorname{app}\{\lfloor \tau \rfloor\} \ e_1^+ \ e_2^+$
$(\operatorname{fst}\{\tau\} e)^+$	$= \operatorname{fst}\{\lfloor \tau \rfloor\} e^+$
$(\operatorname{snd}\{\tau\} e)^+$	$= \operatorname{snd}\{\lfloor \tau \rfloor\} e^+$
$(binop e_1 e_2)^+$	$=binope_1^+e_2^+$
$(cast\{\tau' \Leftarrow \tau\}\; e)^+$	$= \operatorname{cast} \left\{ \lfloor \tau' \rfloor \Leftarrow \lfloor \tau \rfloor \right\} e^{+}$
(if e_1 then e_2 else e_3) ⁺	= if e_1^+ then e_2^+ else e_3^+

 Γ^+

$$(\Gamma, x : \tau)^+ = \Gamma^+, x : \lfloor \tau \rfloor$$

The following proofs will use the fact honest transient types with \sqcup and \sqcap form a lattice ordered by \leq .

Lemma 3.1 (Lattice join idempotent). $\tau \sqcup \tau = \tau$

PROOF. By induction on the structure of τ , in each case following immediately from the definition of \Box .

Lemma 3.2 (Lattice join absorption). $\tau_0 \sqcup (\tau_0 \sqcap \tau_1) = \tau_0$

PROOF. By induction on the structure of τ_0 ; in each case by induction on the structure of τ_1 , in each case following immediately from the definitions of \square and \square and the prior lemma.

Lemma 3.3 (Lattice meet idempotent). $\tau \sqcap \tau = \tau$

PROOF. By induction on the structure of τ , in each case following immediately from the definition of \Box .

Lemma 3.4 (Lattice meet absorption). $\tau_0 \sqcap (\tau_0 \sqcup \tau_1) = \tau_0$

PROOF. By induction on the structure of τ_0 ; in each case by induction on the structure of τ_1 , in each case following immediately from the definitions of \square and \square and the prior lemma.

Lemma 3.5 (Lattice ordering implies \leq). If $\tau = \tau \sqcap \tau'$, then $\tau \leq \tau'$.

PROOF. We proceed by induction on the structure of the definition of $\tau \sqcap \tau'$:

2022-11-18 03:01. Page 18 of 1-100.

```
\perp Since \tau = \tau \sqcap \tau, \tau = \perp; it is immediate that \tau_0 \leq \tau_1.
937
938
          \tau This case occurs if \tau' = *; consequently it is immediate that \tau \leq \tau'.
939
          \tau' In this case, the hypothesis ensures that \tau = \tau', so \tau \le \tau' by reflexivity.
940
          Nat In this case, \tau must be Nat and \tau' must be Int. By definition, Nat \leq Int.
941
942
          \tau In this case, \tau = \tau'; it is immediate that \tau \leq \tau'.
943
          \tau_1 \sqcap \tau_1' \times \tau_2 \sqcap \tau_2' In this case, by the hypothesis, \tau_1 = \tau_1 \sqcap \tau_1' and \tau_2 = \tau_2 \sqcap \tau_2', so by induction \tau_1 \leq \tau_1' and \tau_2 \leq \tau_2'. Then
944
                      it is immediate from the definition of the lattice ordering that \tau_1 \times \tau_2 \le \tau_1' \times \tau_2'.
945
          * \to \tau_2 \sqcap \tau_2' In this case, \tau_2 = \tau_2 \sqcap \tau_2' by the hypothesis, so \tau_2 \le \tau_2' by induction; hence it is immediate from the definition
946
                     of the lattice ordering that * \rightarrow tau_2 \le * \rightarrow \tau'_2.
947
948
949
950
              Lemma 3.6 (Lattice ordering is implied by \leq). If \tau \leq \tau', then \tau = (\tau \sqcap \tau').
951
952
953
954
```

PROOF. We proceed by induction on the structure of the definition of \leq , with the cases of \leq : inlined:

Nat \leq : Int This is immediate by the definition of \sqcap .

955

956

957

958

959 960

961

962

963 964 965

966

967 968

969

970

971 972

974

975

976

977 978

979

980

981

982 983

984

985

986

987 988 $\tau_0 \times \tau_1 \leqslant \tau_2 \times \tau_3$ This is subsumed by the case $\tau_0 \times \tau_1 \leq \tau_2 \times \tau_3$ below.

 $\tau_0 \to \tau_1 \leqslant \tau_2 \to \tau_3$ Because we are considering the lattice of honest transient types, $\tau_0 = \tau_2 = *$, and this is subsumed by the case $* \rightarrow \tau_1 \le * \rightarrow \tau_3$ below.

 $\tau_0 \le \tau_0$ This is immediate by the definition of \Box .

 $\tau_0 \times \tau_1 \le \tau_2 \times \tau_3$ This rule requires that $\tau_0 \le \tau_2$ and $\tau_1 \le \tau_3$; hence, by induction $\tau_0 = \tau_0 \sqcap \tau_2$ and $\tau_1 = \tau_1 \sqcap \tau_3$. This is then immediate by the definition of \sqcap .

 $* \to \tau_1 \le * \to \tau_3$ This rule requires that $\tau_0 \le \tau_1$, and so by induction $\tau_0 = \tau_0 \sqcap \tau_1$; this is then immediate by the definition

 $\perp \leq \tau$ This is immediate by the definition of \sqcap .

 $\tau \leq *$ This is immediate by the definition of \sqcap .

THEOREM 3.7 (SIMPLE TYPING IMPLIES TRUER TRANSIENT TYPING).

```
If \Gamma \vdash_{\mathsf{sim}} e : \tau \text{ then } \Gamma^+ \vdash_{\mathsf{tru}} e^+ : \tau' \text{ where } \tau' \leq \lfloor \tau \rfloor.
```

PROOF. Proceed by induction on the simple typing derivation:

T-Var By the definition of lowering, if $x : \tau \in \Gamma$, then $x : |\tau| \in \Gamma^+$, so T-Var applies and $|\tau|$ is precisely the τ' such that $\Gamma^+ \vdash e^+ : \tau' \text{ and } \tau' \leq \lfloor \tau \rfloor.$

T-Nat, T-Int, T-True, T-False For each base type literal, a corresponding rule exists in the honest transient type system, which ascribes the same time (which is also equal to, and hence below in the lattice, the original simple type).

T-Lam Consider arbitrary Γ_0 , x_0 , τ_0 , e_0 , τ_1 , such that $\Gamma_0 \vdash \lambda(x_0 : \tau_0)$. $e_0 : \tau_0 \to \tau_1$. Then by induction we know that $(\Gamma_0, (x_0) : \tau_0)^+ \vdash e_0^+ : \tau_1'$, for some $\tau_1' \leq \lfloor \tau_1 \rfloor$. Note that $(\Gamma_0, (x_0 : \tau_0))^+ = \Gamma_0^+, x_0 : \lfloor \tau_0 \rfloor$ by definition, and similarly that $(\lambda x_0 : \tau_0. e_0)^+ = \lambda(x_0 : \lfloor \tau_0 \rfloor). e_0^+$ by definition. Then T-Lam applies s.t. $\Gamma_0^+ \vdash \lambda(x_0 : K_0). e_0^+ : * \to \tau_1'$. Note that $\lfloor \tau_0 \rightarrow \tau_1 \rfloor = * \rightarrow * \le * \rightarrow *$ by the definition of lattice ordering, completing the proof.

T-Pair Consider arbitrary Γ_0 , e_0 , e_1 , τ_0 , τ_1 , s.t. $\Gamma_0 \vdash e : \tau$ by simple typing rule T-Pair if $e = \langle e_0, e_1 \rangle$ and $\tau = \tau_0 \times \tau_1$. Then by induction, there exist some τ_0' and τ_1' , s.t. $\Gamma_0^+ \vdash e_0^+ : \tau_0', \Gamma_0^+ : e_1^+ : \tau_1', \tau_0' \leq \lfloor \tau_0 \rfloor$, and $\tau_1' \leq \lfloor \tau_1 \rfloor$. Then instantiate 2022-11-18 03:01. Page 19 of 1-100.

991

1040

```
T-Cast Consider arbitrary \Gamma_0, e_0, \tau_0, \tau_1, s.t. \Gamma_0 \vdash e : \tau by simple typing rule T-Cast if e = \text{cast} \{\tau_0 \Leftarrow \tau_1\} e_0 and \tau = \tau_1.
992
                       Then by induction, \Gamma_0^+ \vdash e_0^+ : \tau_0' for some \tau_0' s.t. \tau_0' \leq \lfloor \tau_0 \rfloor. Instantiate \tau' by \lfloor \tau_1 \rfloor \cap \lfloor \tau_0 \rfloor \cap \tau_0'; then it is clear that
993
                       the honest transient typing rule T-Cast applies, since by definition e^+ = \text{cast}\{\lfloor \tau_0 \rfloor \Leftarrow \lfloor \tau_1 \rfloor\} e_0^+. It remains to be
994
995
                       shown that \lfloor \tau_1 \rfloor \sqcap \lfloor \tau_0 \rfloor \sqcap \tau_0' \leq \lfloor \tau_1 \rfloor; this follows immediately from the properties of the lattice meet operation.
           T-App Consider arbitrary \Gamma_0, e_0, \tau_0, \tau_1 s.t. \Gamma_0 \vdash e : \tau by simple typing rule T-App if e = app\{\tau_1\} e_0 e_1 and \tau = \tau_1. Then
997
                       by induction, \Gamma_0^+ \vdash e_0^+ : \tau_l for some \tau_l \leq \lfloor \tau_0 \to \tau_1 \rfloor = * \to *, and \Gamma_0^+ \vdash e_1^+ : \tau_l' for some \tau_0' \leq \lfloor \tau_0 \rfloor. By inspection
                       of \leq, note that \tau_l must be either \perp or * \to \tau_l' for some \tau_l'. Note that e^+ = \operatorname{app}\{\lfloor \tau_1 \rfloor\} e_0^+ e_1^+, and so in the former
1000
                       case T-AppBot syntactically applies and in the latter T-App; consider each case:
1001
                       \tau_l = \bot: Instantiate \tau' = \bot; then it is clear that \Gamma_0' \vdash e' : \tau' by T-AppBot. Then \bot \le \lfloor \tau_1 \rfloor is immediate by the
1002
                               definition of lattice ordering.
1003
                       \tau_l = * \to \tau_l': Instantiate \tau' = \lfloor \tau_1 \rfloor \sqcap \tau_l'; then it is clear that \Gamma_0' \vdash e' : \tau' by T-App, so what remains to be shown is
1004
1005
                               that \lfloor \tau_1 \rfloor \sqcap \tau_l \leq \lfloor \tau_1 \rfloor; this is immediate by the definition of meet on a lattice.
1006
           T-Fst Consider arbitrary \Gamma_0, e_0, \tau_0, \tau_1, s.t. \Gamma_0 + e : \tau by simple typing rule T-Fst with premise \Gamma_0 + e_0 : \tau_0 \times \tau_1 if
1007
                       e = \text{fst}\{\tau_0\}\ e_0 \text{ and } \tau = \tau_0. Then, by induction, \Gamma_0' \vdash e : \tau_p' \text{ s.t. } \tau_p' \le \lfloor \tau_0 \times \tau_1 \rfloor = *\times *. By inspection on \le, note that
1008
                       \tau_p' must be either \bot or \tau_{p_0'} \times \tau_{p_1'} for some \tau_{p_0} and \tau_{p_1}. Since e^+ = \text{fst}\{\lfloor \tau_0 \rfloor\} e_0^+, the rule T-FstBot applies in the
1009
1010
                       former case, and similarly T-Fst applies in the latter. Consider each of these cases:
                       \tau_p' = \bot: Instantiate \tau' = \bot; \Gamma_0^+ \vdash e^+ : \tau' by T-FstBot, and \bot \le \lfloor \tau_0 \rfloor follows immediately from the definition of
                               lattice ordering.
1013
                       \tau_p' = \tau_{p_0'} \times \tau_{p_1'}: Instantiate \tau' with \lfloor \tau_0 \rfloor \sqcap \tau_{p_0'}. Then \Gamma_0^+ \vdash e^+ : \tau' by T-Fst, and \tau' \leq \lfloor \tau_0 \rfloor by the definition of
1014
1015
1016
           T-Snd Consider arbitrary \Gamma_0, e_0, \tau_0, \tau_1, s.t. \Gamma_0 \vdash e : \tau by simple typing rule T-Snd with premise \Gamma_0 \vdash e_0 : \tau_0 \times \tau_1 if
1017
                       e = \operatorname{snd}\{\tau_1\} e_0 and \tau = \tau_1. Then, by induction, \Gamma_0' \vdash e : \tau_p' s.t. \tau_p' \le \lfloor \tau_0 \times \tau_1 \rfloor = * \times *. By inspection on \le, note
1018
                       that \tau'_p must be either \perp or \tau_{p'_0} \times \tau_{p'_1} for some \tau_{p_0} and \tau_{p_1}. Since e^+ = \text{snd}\{\lfloor \tau_1 \rfloor\} e^+_0, the rule T-SndBot applies
1020
                       in the former case, and similarly T-Snd applies in the latter. Consider each of these cases:
1021
                       \tau_p' = \bot: Instantiate \tau' = \bot; \Gamma_0^+ \vdash e^+ : \tau' by T-SndBot, and \bot \le \lfloor \tau_1 \rfloor follows immediately from the definition of
1022
1023
                       \tau_p' = \tau_{p_0'} \times \tau_{p_1'}: Instantiate \tau' with \lfloor \tau_1 \rfloor \cap \tau_{p_1'}. Then \Gamma_0^+ \vdash e^+ : \tau' by T-Snd, and \tau' \leq \lfloor \tau_1 \rfloor by the the definition of
1026
           T-Binop Consider arbitrary \Gamma_0, binop, e_0, e_1, \tau_0, \tau_1, and \tau_2, s.t. \Gamma_0 \vdash e : \tau by simple typing rule T-Binop with premise
1027
                       \Delta(binop, \tau_0, \tau_1) = \tau_2 if e = binop \, e_0 \, e_1 and \tau = \tau_2. By induction, note that \Gamma_0^+ \vdash e_0^+ : \tau_0' for some \tau_0' \leq \lfloor \tau_0 \rfloor, and
1028
                       \Gamma_0^+ \vdash e_1^+ : \tau_1' for some \tau_1' \leq \lfloor \tau_1 \rfloor. Note that for the simple typing \Delta(binop, \tau_0, \tau_1) to be defined, \tau_0 and \tau_1 must
1029
1030
                       each be either Nat or Int; consequently, by inspection of the lattice order, \tau'_0 and \tau'_1 must each be Nat, Int, or \bot.
1031
                       Then by inspection, in any such case, \Delta(binop, \tau'_0, \tau'_1) is defined and \leq \Delta(binop, \tau_0, \tau_1) = \tau_2. Then instantiate \tau'
                       with \lfloor \tau_2 \rfloor \sqcap \Delta(binop, \tau'_0, \tau'_1); since e^+ = binop \, e_0 \, e_1, the rule S-Binop applies, and by the definition of meet on a
1033
1034
                       lattice, |\tau_2| \leq \tau'.
1035
           T-If Consider arbitrary \Gamma_0, e_0, e_1, e_2, \tau_0, s.t. \Gamma_0 + if e_1 then e_2 else e_3: \tau_0 by the T-If simple typing rule. Let
1036
                       e = \text{if } e_1 \text{ then } e_2 \text{ else } e_3 \text{ and } \tau = \tau_0. Then by induction, there exist some \tau_b' \leq \lfloor \mathsf{Bool} \rfloor = \mathsf{Bool}, \tau_0' \leq \lfloor \tau_0 \rfloor, and
1037
                       \tau_1' \leq \lfloor \tau_0 \rfloor, s.t. \Gamma_0^+ \vdash e_0^+ : \tau_b', \Gamma_0^+ \vdash e_1^+ : \tau_0', and \Gamma_0^+ : \vdash e_2^+ : \tau_1'. Notice that \tau_b' may be only \bot or Bool, by the definition
```

 $\tau' = \tau_0 \times \tau_1$; it is clear that the honest transient typing rule T-Pair applies, since $(\langle e_0, e_1 \rangle)^+ = \langle e_0^+, e_1^+ \rangle$, and it is

immediate by the definition of \leq that $\tau' \leq \lfloor \tau_0 \times \tau_1 \rfloor = * \times *$.

```
Type Vigilance and the Truth About Transient (Techreport)
                                         of lattice ordering. Since e^+ = \text{if } e_0^+ then e_1^+ else e_2^+, in the former case the rule T-IfBot applies; in the latter the
1041
1042
                                         rule T-If applies. Consider each of these cases:
1043
                                         \tau_h' = \bot: By T-IfBot, \Gamma_0^+ \vdash e^+ : \bot, so instantiate \tau' = \bot. Notice then that \bot \le \lfloor \tau \rfloor by lattice ordering, so the proof
                                                      is completed.
1045
                                         \tau_b' = \text{Bool: By T-If, } \Gamma_0^+ \vdash e^+ : \tau_0' \sqcup \tau_1'. \text{ Instantiate } \tau' \text{ by } \tau_0' \sqcup \tau_1'; \text{ then we must show that } \tau' \leq \lfloor \tau \rfloor. \text{ Since } \tau_0' \sqcup \tau_1' = \tau_1' = \tau_1' \sqcup \tau_1' = \tau_
1046
1047
                                                      \tau_0' \leq \lfloor \tau_0 \rfloor and \tau_1' \leq \lfloor \tau_0 \rfloor, \lfloor \tau_0 \rfloor is an upper bound of \tau_0' and \tau_1'. By the definition of join on a lattice, \tau_0' \sqcup \tau_1' is
1048
                                                      less-than-or-equal-to any other upper bound of \tau_0 and \tau_1, so this is shown.
                    T-Sub Consider arbitrary \Gamma_0, e_0, \tau_1, \tau_0, s.t. \Gamma_0 \vdash e : \tau by simple typing rule T-Sub with premise \tau_0 \leqslant \tau_1 if e = e_0 and
                                         \tau = \tau_1. By induction, \Gamma_0 \vdash e^+ : \tau_0' for some \tau_0' \le \lfloor \tau_0 \rfloor. Then instantiate \tau' = \tau_0'. It is immediate that \Gamma_0 \vdash e^+ : \tau'; it
1051
1052
                                         remains to be shown that \tau' \leq \lfloor \tau_1 \rfloor. Since \tau_0 \leqslant \tau_1, \tau_0 \leq \tau_1. By Lemma 3.8, \lfloor \tau_0 \rfloor \leq \lfloor \tau_1 \rfloor. Then by Lemma 3.9,
1053
                                         \tau' = \tau'_0 \le \lfloor \tau_0 \rfloor \le \lfloor \tau_1 \rfloor so \tau' \le \lfloor \tau_1 \rfloor.
1054
1055
                                                                                                                                                                                                                                                                                                                                                  1056
                           Lemma 3.8 (Lattice ordering is preserved by Tag-Of). If \tau_0 \leq \tau_1, then \lfloor \tau_0 \rfloor \leq \lfloor \tau_1 \rfloor.
1057
1058
                           PROOF. By cases on the structure of the definition of \leq; in each case the lemma is immediate.
                                                                                                                                                                                                                                                                                                                                                  1059
1060
                           Lemma 3.9 (Lattice ordering is transitive). If \tau \leq \tau' and \tau' \leq \tau'', then \tau \leq \tau''.
1061
                           PROOF. By induction on the structure of the definition of \tau \leq \tau' (generalized with respect to \tau''), with the cases of
1065
                    Nat \leq: Int: Since by assumption Int \leq \tau'', it is clear by inspection that \tau'' must be either Int or *; in either case
1066
                                         Nat \leqslant: \tau'' is immediate.
1067
                    \tau_0 \times \tau_1 \leqslant \tau_2 \times \tau_3: This is subsumed by the case \tau_0 \times \tau_1 \leq \tau_2 \times \tau_3 below.
1068
                    \tau_0 \to \tau_1 \leqslant \tau_2 \to \tau_3: Because we are considering the lattice of honest transient types, \tau_0 = \tau_2 = *, and this is subsumed
1069
1070
                                         by the case * \rightarrow \tau_1 \le * \rightarrow \tau_3 below.
                   \tau \leq \tau: Since by assumption \tau' \leq \tau'', \tau = \tau' \leq \tau_2.
1072
                    \tau_0 \times \tau_1 \le \tau_2 \times \tau_3: Since by assumption \tau' = \tau_2 \times \tau_3 \le \tau'', it is clear that \tau'' must be either * or \tau_0'' \times \tau_1'' for some \tau_0'', \tau_1'' s.t.
1073
                                         \tau_2 \leq \tau_0^{\prime\prime} and \tau_3 \leq \tau_1^{\prime\prime}. If \tau^{\prime\prime} is *, the lemma follows immediately. Otherwise, note that this rule requires that
1074
1075
                                         \tau_0 \le \tau_2 and \tau_1 \le \tau_3; hence, by induction, \tau_0 \le \tau_0'' and \tau_1 \le \tau_1'', and therefore \tau \le \tau''.
1076
                    *\to \tau_1 \le *\to \tau_3: Since by assumption \tau' = *\to \tau_3 \le \tau'', it is clear that \tau'' must be either * or *\to \tau_1'' for some \tau_1'' s.t.
                                         \tau_3 \leq \tau_1''. If \tau'' is *, the lemma follows immediately. Otherwise, note that this rule requires that \tau_1 \leq \tau_3; hence,
1078
                                         by induction, \tau_1 \leq \tau_1'', and therefore \tau \leq \tau''.
1079
1080
                    \perp \leq \tau \quad \tau = \perp \leq \tau'' is immediate by the definition of lattice ordering.
1081
                    \tau \leq * Since by assumption \tau' = * \leq \tau'', \tau'' must be *, and so the lemma follows immediately.
1082
                                                                                                                                                                                                                                                                                                                                                  1083
```

3.3 Tag Typing Implies Truer Transient Typing

Theorem 3.10 (Tag Typing Implies Truer Transient Typing). If $\Gamma \vdash_{\mathsf{tag}} e : K \ then \ \exists \tau \leq K \ such \ that \ \Gamma \vdash_{\mathsf{tru}} e : \tau.$

PROOF. By induction over the tag typing derivation.

$$\frac{\text{T-Var}}{(x_0:K_0) \in \Gamma_0} = \frac{\text{T-Nat}}{\Gamma_0 \vdash_{\text{tag}} x_0:K_0} = \frac{\text{T-Int}}{\Gamma_0 \vdash_{\text{tag}} i_0:\text{Int}} = \frac{\text{T-True}}{\Gamma_0 \vdash_{\text{tag}} \text{True}:\text{Bool}} = \frac{\text{T-False}}{\Gamma_0 \vdash_{\text{tag}} \text{False}:\text{Bool}} = \frac{\text{T-False}}{\Gamma_0 \vdash_{\text{t$$

These cases are immediate by applying the corresponding truer typing rule and from premises.

These cases follows by the induction hypothesis and the corresponding rule.

These cases follow by induction and their corresponding typing rule, with the caveat that if the truer type of the premise is \bot , the corresponding bot rule must be used.

T-Cast
$$\Gamma_0 \vdash_{\mathsf{tag}} e_0 : K_0$$

$$K_0 \sim K_1$$

$$\Gamma_0 \vdash_{\mathsf{tag}} \mathsf{cast} \{K_1 \Leftarrow K_0\} \ e_0 : K_1$$

This case follows by induction and applying the bnd rule in truer, noting truer doesn't require any relationships between the type of what's underneath and the tags on the bnds.

T-BINOP
$$\Gamma_0 \vdash_{\mathsf{tag}} e_0 : K_0$$

$$\Gamma_0 \vdash_{\mathsf{tag}} e_1 : K_1$$

$$\Delta(\mathit{binop}, K_0, K_1) = K_2$$

$$\overline{\Gamma_0 \vdash_{\mathsf{tag}} \mathit{binop} e_0 e_1 : K_2}$$

This case follows by induction, noting that if either of the truer types corresponding to K_0 or K_1 are \bot , then the result type is \bot . If the truer types are different, ie one is Nat and the other Int, we apply subsumption to get both at Int, and then can apply the binop rule. Otherwise, we directly apply the binop rule.

4 Vigilance

4.1 Vigilance Logical Relation

$$[\![\Gamma \vdash_t e : \tau]\!]_V^L \triangleq \ \forall (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^L[\![\Gamma]\!] \text{ where } \Sigma : (k, \Psi). \ (k, \Psi, \Sigma, \gamma(e)) \in \mathcal{E}^L[\![\tau]\!]$$

$$\begin{split} \mathcal{G}^L \llbracket \Gamma, x : \tau \rrbracket &\triangleq \{ (k, \Psi, \Sigma, \gamma [x \mapsto \ell]) \mid (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^L \llbracket \Gamma \rrbracket \\ & \wedge \ell \in dom(\Psi) \wedge \ell \notin dom(\gamma) \\ & \wedge (k, \Psi, \Sigma, \ell) \in \mathcal{V}^L \llbracket \tau \rrbracket_k \} \end{split}$$

$$\mathcal{G}^L[\![\bullet]\!] \triangleq \{(k, \Psi, \Sigma, \emptyset)\}$$

$$\vdash \Sigma \triangleq \ \forall \ell \in dom(\Sigma). \ \Sigma(\ell) = ((\ell', some(\tau', \tau)) \land \tau' \sim pointsto(\Sigma, \ell) \land \tau \sim pointsto(\Sigma, \ell))$$

$$\land \neg * \times * \sim pointsto(\Sigma, \ell))$$

$$\vee \Sigma(\ell) = (v, \text{none}) \text{ where } v \notin \mathbb{L}$$

$$\begin{split} \Sigma: (k, \Psi) \triangleq & \ dom(\Sigma) = dom(\Psi) \ \land \ \vdash \Sigma \ \land \ \forall j < k, \ell \in dom(\Sigma). ((j, \Psi, \Sigma, \ell) \in \mathcal{VH}^L[\![\Psi(\ell)]\!] \\ & \ \land (\Sigma(\ell) = (\ell', \mathsf{some}(\tau, \tau')) \Rightarrow \Psi(\ell) = [\tau, \tau', \Psi(\ell')] \ \land \Psi(\ell') = [\tau'', \ldots] \ \land \tau'' <: \tau') \\ & \ \land (\Sigma(\ell) = (v, \mathsf{none}) \land v \notin \mathbb{L} \Rightarrow \exists \tau. \Psi(\ell) = [\tau])) \end{split}$$

This is an unfolded version of the definition in the paper. We break up the definition there for ease of explanation, and unfold here for ease of use.

$$(j, \Psi) \supseteq (k, \Psi) \triangleq j \leq k \land \forall \ell \in dom(\Psi). \ \Psi'(\ell) = \Psi(\ell)$$

$$\mathcal{E}\mathcal{H}^{L}[\![\overline{\tau}]\!] \triangleq \{(k, \Psi, \Sigma, e) \mid \forall j \leq k. \ \forall \Sigma' \supseteq \Sigma, e'. \ (\Sigma, e) \longrightarrow_{L}^{j} (\Sigma', e') \land \mathsf{irred}(e') \\ \Rightarrow (e' = \mathsf{Err}^{\bullet} \lor (\exists (k - j, \Psi') \supseteq (k, \Psi). \ \Sigma' : (k - j, \Psi') \land (k - j, \Psi', \Sigma', e') \in \mathcal{VH}^{L}[\![\overline{\tau}]\!]))\}$$

$$\mathcal{VH}^{L}\llbracket \operatorname{Int}, \tau_{2}, \dots \tau_{n} \rrbracket \triangleq \{ (k, \Psi, \Sigma, \ell) \mid \forall \tau \in [\operatorname{Int}, \tau_{2}, \dots \tau_{n}]. \ (k, \Psi, \Sigma, \ell) \in \mathcal{V}^{L}\llbracket \tau \rrbracket \}$$

$$\mathcal{VH}^{L}[\![\mathsf{Nat},\tau_{2},\ldots\tau_{n}]\!] \triangleq \{(k,\Psi,\Sigma,\ell) \mid \forall \tau \in [\mathsf{Nat},\tau_{2},\ldots\tau_{n}]. \ (k,\Psi,\Sigma,\ell) \in \mathcal{V}^{L}[\![\tau]\!]\}$$

$$\mathcal{VH}^L[\![\mathsf{Bool},\tau_2,\ldots\tau_n]\!] \triangleq \{(k,\Psi,\Sigma,\ell) \mid \forall \tau \in [\mathsf{Bool},\tau_2,\ldots\tau_n]. \ (k,\Psi,\Sigma,\ell) \in \mathcal{V}^L[\![\tau]\!]\}$$

2022-11-18 03:01. Page 23 of 1-100.

```
1198
                 \mathcal{VH}^L\llbracket \tau_1' \times \tau_1'', \tau_2, \dots \tau_n \rrbracket \triangleq \{(k, \Psi, \Sigma, \ell) \mid \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \_)
1199
1200
                                                                                                                 \wedge (k, \Psi, \Sigma, \ell_1) \in \mathcal{VH}^L \llbracket \tau_1', fst(\tau_2), \dots fst(\tau_n) \rrbracket
1201
                                                                                                                 \wedge (k, \Psi, \Sigma, \ell_2) \in \mathcal{VH}^L \llbracket \tau_1'', snd(\tau_2), \dots snd(\tau_n) \rrbracket \}
1202
1203
1204
1205
1206
                \mathcal{VH}^{L}\llbracket \tau_{1}' \to \tau_{1}'', \tau_{2}, \dots \tau_{n} \rrbracket \triangleq \{(k, \Psi, \Sigma, \ell) \mid \forall (j, \Psi') \supseteq (k, \Psi), \Sigma' \supseteq \Sigma \text{ where } \Sigma' : (j, \Psi').
1207
1208
                                                                                                                         \forall \tau_0 \text{ where } cod(\tau_1'') \leqslant \tau_0 \forall \ell_v \text{ where } (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^L[\![\tau_1']\!].
1209
1210
                                                                                                                         (j, \Psi', \Sigma', \mathsf{app}\{\tau_0\} \ell \ell_v) \in \mathcal{EH}^L \llbracket [\tau_0, cod(\tau_2), \dots cod(\tau_n)] \rrbracket \}
1211
1212
                 \mathcal{VH}^{L}[\![*,\tau_2,\ldots\tau_n]\!] \triangleq \{(k,\Psi,\Sigma,\ell) \mid (k-1,\Psi,\Sigma,\ell) \in \mathcal{VH}^{L}[\![\mathsf{Int},\tau_2,\ldots\tau_n]\!]
1213
1214
                                                                                                       (k-1, \Psi, \Sigma, \ell) \in \mathcal{VH}^L \llbracket \mathsf{Bool}, \tau_2, \dots \tau_n \rrbracket
1215
                                                                                                    \forall (k-1, \Psi, \Sigma, \ell) \in \mathcal{VH}^L \llbracket * \times *, \tau_2, \dots, \tau_n \rrbracket
1216
1217
                                                                                                    \forall (k-1, \Psi, \Sigma, \ell) \in \mathcal{VH}^L \llbracket * \rightarrow *, \tau_2, \dots, \tau_n \rrbracket \}
1218
1219
1220
1222
1223
                 \mathcal{E}^{L}[\![\tau]\!] \triangleq \{(k, \Psi, \Sigma, e) \mid \forall j \leq k. \ \forall \Sigma' \supseteq \Sigma, e'. \ (\Sigma, e) \longrightarrow_{L}^{j} (\Sigma', e') \land \mathsf{irred}(e')
1224
1225
                                                                            \Rightarrow (e' = \operatorname{Err}^{\bullet} \vee (\exists (k-i, \Psi') \supseteq (k, \Psi), \Sigma' : (k-i, \Psi') \wedge (k-i, \Psi', \Sigma', e') \in \mathcal{V}^{L}[[\tau]]))\}
1226
1227
1228
1229
1230
                 \mathcal{V}^L[\![\mathsf{Int}]\!] \triangleq \{(k, \Psi, \Sigma, \ell \mid \mathsf{pointsto}(\Sigma, \ell) \in \mathbb{Z}\}
1231
1232
1233
                 \mathcal{V}^L[\![\mathsf{Nat}]\!] \triangleq \{(k, \Psi, \Sigma, \ell \mid \mathsf{pointsto}(\Sigma, \ell) \in \mathbb{N}\}
                 \mathcal{V}^L[\![\mathsf{Bool}]\!] \triangleq \{(k, \Psi, \Sigma, \ell \mid \mathsf{pointsto}(\Sigma, \ell) \in \mathbb{B}\}
1236
1237
1238
                 \mathcal{V}^L\llbracket\tau_1\times\tau_2\rrbracket\triangleq\{(k,\Psi,\Sigma,\ell)\mid\Sigma(\ell)=(\langle\ell_1,\ell_2\rangle,\_)\wedge(k,\Psi,\Sigma,\ell_1)\in\mathcal{V}^L\llbracket\tau_1\rrbracket\wedge(k,\Psi,\Sigma,\ell_2)\in\mathcal{V}^L\llbracket\tau_2\rrbracket\}
1239
1240
1241
1242
1243
                 \mathcal{V}^L\llbracket \tau_1 \to \tau_2 \rrbracket \triangleq \{(k, \Psi, \Sigma, \ell) \mid \forall (j, \Psi') \supseteq (k, \Psi). \ \forall \Sigma' \supseteq \Sigma \text{ where } \Sigma' : (j, \Psi').
1244
                                                                                            \forall \ell_v \text{ where } (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^L \llbracket \tau_1 \rrbracket . \forall \tau_o. \text{ where } \tau_2 \leqslant : \tau_0
1245
1246
                                                                                            (j, \Psi', \Sigma', \operatorname{app}\{\tau_o\} \ell \ell_v) \in \mathcal{E}^L \llbracket \tau_0 \rrbracket \}
1247
```

2022-11-18 03:01. Page 24 of 1-100.

$$\begin{split} \mathcal{V}^L[\![*]\!] &\triangleq \{(k, \Psi, \Sigma, \ell) \mid (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^L[\![\mathsf{Int}]\!] \\ &\qquad \qquad (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^L[\![\mathsf{Bool}]\!] \\ &\qquad \qquad \vee (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^L[\![* \times *]\!] \\ &\qquad \qquad \vee (k-1, \Psi\ell\Sigma, \ell) \in \mathcal{V}^L[\![* \to *]\!] \} \end{split}$$

4.2 Vigilance Theorem

 $\Gamma \vdash_t e : \tau \text{ then } [\![\Gamma \vdash_t e : \sigma]\!]_V^L$

Vigilance Fundamental Property for Natural with Simple Typing In this subsection, we use $\Gamma \vdash e : \tau$ to mean $\Gamma \vdash_{sim} e : \tau$. 4.3.1 Lemmas Used Without Mention $\text{Lemma 4.1 (Stepping to Error Implies Expression Relation)}. \ \ If (\Sigma, e) \longrightarrow_N^j (\Sigma', \mathsf{Err}^\bullet) \ \ then \ (k, \Psi, \Sigma, e) \in \mathcal{E}^N[\![\tau]\!]$ PROOF. If k < j, then we're done because the condition in the expression relation is vacuously true. Otherwise, we can use j as our steps, Σ' as our ending value log, and Err^{\bullet} as our irreducible expression, and we satisfy the condition in the expression relation. Lemma 4.2 (Stepping to Error Implies Expression History Relation). If $(\Sigma, e) \longrightarrow_N^j (\Sigma', \operatorname{Err}^{\bullet})$ then $(k, \Psi, \Sigma, e) \in$ $\mathcal{EH}^{N}[\![\overline{\tau}]\!]$ PROOF. Similar to the previous proof. Lemma 4.3 (Anti-Reduction - Head Expansion - Expression Relation Commutes With Steps). If $(k, \Psi', \Sigma', e') \in$ $\mathcal{E}^{N}[\![\tau]\!] \ \ and \ (\Sigma,e) \longrightarrow_{N}^{j} (\Sigma',e') \ \ and \ \Sigma': (k,\Psi') \ \ then \ (k+j,\Psi,\Sigma,e) \in \mathcal{E}^{N}[\![\tau]\!]$ PROOF. Unfolding the expression relation in our hypothesis, there exists (Σ'', e'') , j' such that $(\Sigma', e') \longrightarrow_N^{j'} (\Sigma'', e'')$ and (Σ''', e'') is irreducible. Either $e'' = \operatorname{Err}^{\bullet}$, in which case $(\Sigma, e) \longrightarrow_{N}^{j+j'} (\Sigma'', \operatorname{Err}^{\bullet})$, so we're done. Otherwise, there is a $(k-j', \Psi'') \supseteq (k, \Psi')$ such that $\Sigma'' : (k-j', \Psi'')$, and $(k-j', \Psi'', \Sigma'', e'') \in \mathcal{V}^{N}[\![\tau]\!]$. Using this information, we can show $(k+j, \Psi, \Sigma, e) \in \mathcal{E}^{N}[\![\tau]\!]$ by noting $(\Sigma, e) \longrightarrow_{N}^{j+j'} (\Sigma'', e'')$. Lemma 4.4 (Anti-Reduction - Head Expansion - Expression History Commutes With Steps). If $(k, \Psi', \Sigma', e') \in$ $\mathcal{EH}^N[\![\bar{\tau}]\!] \ and \ (\Sigma,e) \longrightarrow_N^j (\Sigma',e') \ and \ \Sigma': (k,\Psi') \ then \ (k+j,\Psi,\Sigma,e) \in \mathcal{EH}^N[\![\bar{\tau}]\!]$ PROOF. Similar to the previous proof. Lemma 4.5 (The Operational Semantics Preserves Well Formed Value Logs). If $\vdash \Sigma$ and $(\Sigma, e) \longrightarrow_N^* (\Sigma', e')$ then $\vdash \Sigma'$. PROOF. The proof is immediate by inspection of the Operational Semantics. Lemma 4.6 (Not Enough Steps Implies any Expression Relation). If $(\Sigma,e) \longrightarrow_N^k (\Sigma',e')$ and (Σ',e') is not irreducible, then $\forall j \leq k$. $(j, \Psi, \Sigma, e) \in \mathcal{E}^N[\![\tau]\!]$ and $(j, \Psi, \Sigma, e) \in \mathcal{EH}^N[\![\tau]\!]$.

PROOF. Both conclusions are immediate, since the implications in the relations are vacuously true.

Lemma 4.7 (The Operational Semantics Only Grows Stores). If $(\Sigma, e) \longrightarrow_N^* (\Sigma', e')$ then $\Sigma' \supseteq \Sigma$.

PROOF. This is a corollary of Lemma 4.8.

Lemmas Used With Mention

Lemma 4.8 (The Operational Semantics Produces Value Log Extensions). If $(\Sigma, e) \longrightarrow_N^* (\Sigma', e')$, then $\exists \overline{\ell} \subseteq \mathbb{R}$ $dom(\Sigma')$ such that $\overline{\ell \notin dom(\Sigma)}$ and $\Sigma' = \Sigma \overline{[\ell \mapsto (v, _)]}$.

2022-11-18 03:01. Page 26 of 1-100.

П

PROOF. By inspection of the Operational Semantics, no steps modify the value stored in the value log, meaning $\Sigma' \supseteq \Sigma$.

And also by the inspection of the Operational Semantics, there is exactly one rule to allocate new entries in the value \log , meaning $\Sigma' \setminus \Sigma$ is a suitable choice for $\overline{[\ell \mapsto (v,_)]}$.

Lemma 4.9 (Steps are Preserved in Future Value Logs). If $(\Sigma, e) \longrightarrow_N^j (\Sigma', e')$ and $\overline{\ell \notin dom(\Sigma')}$ then $(\Sigma[\ell \mapsto (v, _)], e) \longrightarrow_N^j (\Sigma'[\ell \mapsto (v, _)], e')$.

PROOF. Since all of the added locations are not in Σ' , and therefore also not in Σ , no rule that will lookup a label in the derivation tree for $(\Sigma, e) \longrightarrow_N^j (\Sigma', e')$ will find a different value or type.

The only remaining notable reduction steps are those that allocate a new label and value entry, but since $\overline{\ell \notin dom(\Sigma')}$, we can allocate the same entry unchanged.

Lemma 4.10 (Subtyping Preserves Logical Relations). $\forall \Sigma, k, \Psi, \tau, \tau'$. where $\Sigma : (k, \Psi)$ and $\tau \leqslant \tau'$.

```
(1) If (k, \Psi, \Sigma, e) \in \mathcal{E}^N \llbracket \tau \rrbracket then (k, \Psi, \Sigma, e) \in \mathcal{E}^N \llbracket \tau' \rrbracket
```

(2) If
$$(k, \Psi, \Sigma, \ell) \in \mathcal{V}^N \llbracket \tau \rrbracket$$
 then $(k, \Psi, \Sigma, \ell) \in \mathcal{V}^N \llbracket \tau' \rrbracket$

(3) If
$$(k, \Psi, \Sigma, e) \in \mathcal{EH}^N[\![\tau, \overline{\tau}]\!]$$
 then $(k, \Psi, \Sigma, e) \in \mathcal{EH}^N[\![\tau', \overline{\tau}]\!]$

(4) If
$$(k, \Psi, \Sigma, \ell) \in \mathcal{VH}^N[\![\tau, \overline{\tau}]\!]$$
 then $(k, \Psi, \Sigma, \ell) \in \mathcal{VH}^N[\![\tau', \overline{\tau}]\!]$

Proof. Proceed by mutual induction on k and τ :

- k = 0: Both 1 and 3 are immediate if $e \neq \ell$.
 - If $e = \ell$ then 1 and 3 follow immediately from 2 and 4.

2 and 4 follow identically in the k = 0 case as they do in the k > 0 case, but the function case is vacuously true.

• k > 0:

 (1) Unfolding our hypothesis, there is some (Σ',e') , j such that $(\Sigma,e)\longrightarrow_N^j (\Sigma',e')$.

If $e' = \text{Err}^{\bullet}$ then we're done.

Otherwise, there is some $(k - j, \Psi') \supseteq (k, \Psi')$ such that $\Sigma' : (k - j, \Psi')$ and $(k - j, \Psi', \Sigma', e') \in \mathcal{V}^N[\![\tau]\!]$. We now have two obligations:

a)
$$(k - j, \Psi', \Sigma', e') \in \mathcal{V}^N[\![\tau']\!]$$
.

b)
$$\Sigma' : (k - j, \Psi')$$
.

For a) by IH 2) (not necessarily smaller by type or index), we have $(k - j, \Psi', \Sigma', e') \in \mathcal{V}^N[\![\tau']\!]$, which is what we wanted to show.

For b), this is immediate from the premise.

- (2) Case split on $\tau \leqslant \tau'$:
 - i) $\tau \leqslant : \tau$: immediate.
 - ii) Nat \leq : Int: immediate because $\mathbb{N} \subseteq \mathbb{Z}$.
 - iii) $\tau_1 \times \tau_2 \leqslant : \tau_1' \times \tau_2'$, with $\tau_1 \leqslant : \tau_1'$ and $\tau_2 \leqslant : \tau_2'$:

We want to show $(k, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\![\tau']\!]$.

Unfolding our hypothesis, we get that $\Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, _)$.

We want to show $(k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^N[\![\tau_1']\!]$ and $(k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^N[\![\tau_2']\!]$

We can apply IH 2) (smaller by type) to both of these judgements to get $(k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^N[\![\tau_1']\!]$ and

```
(k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^N \llbracket \tau_2' \rrbracket.
1405
1406
                                               This is sufficient to show (k, \Psi, \Sigma, \Sigma(\ell)) \in \mathcal{V}^N \llbracket \tau' \rrbracket.
1407
                                        iv) \tau_1 \to \tau_2 \leqslant : \tau'_1 \to \tau'_2, with \tau'_1 \leqslant : \tau_1 and \tau_2 \leqslant : \tau'_2:
1408
                                               We want to show (k, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\![\tau']\!].
1409
                                               Let (i, \Psi') \supseteq (k, \Psi) and \Sigma' \supseteq \Sigma such that \Sigma' : (i, \Psi').
1410
1411
                                               Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N \llbracket \tau_1' \rrbracket.
1412
                                               Let \tau_0 \geqslant : \tau_2'.
1413
                                               We want to show (j, \Psi', \Sigma', \operatorname{app}\{\tau_0\} \ell \ell_v) \in \mathcal{E}^N \llbracket \tau_0 \rrbracket.
                                               From IH 2) (smaller by type) applied to the facts that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N[[\tau'_1]] and that \tau'_1 \leqslant \tau_1 gives
1416
                                               us (j + 1, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N [\![\tau_1]\!].
1417
                                               Then, we can apply our hypothesis about \Sigma(\ell) (noting that \tau_0 \geqslant : \tau_2' \geqslant : \tau_2) to get (j, \Psi', \Sigma', \mathsf{app}\{\tau_0\} \ell \ell_v) \in
1418
                                               \mathcal{E}^N \llbracket \tau_0 \rrbracket, which is what we wanted to prove.
1419
                             (3) Unfolding our hypothesis, we get that there are some (\Sigma', e'), j such that (\Sigma, e) \longrightarrow_N^j (\Sigma', e') and (\Sigma', e')
1420
1421
                                    are irreducible.
1422
                                    If e' = \text{Err}^{\bullet}, then we're done.
1423
                                    Otherwise, there is some (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{VH}^N[\tau, \overline{\tau}],
1424
                                    which means \exists \ell \in dom(\Sigma') such that e' = \ell.
1425
1426
                                    Then by IH 4) (not necessarily smaller by type or index) with \tau \leq \tau', we get (k-j, \Psi', \Sigma', \ell) \in \mathcal{VH}^N[\![\tau', \overline{\tau}]\!],
                                    which is what we wanted to show.
                             (4) We want to show (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^N[\tau', \overline{\tau}].
                                    We case split on \tau \leqslant : \tau':
1431
                                          i) \tau = \tau': immediate by premise.
1432
                                         ii) Nat ≤: Int:
1433
                                               by our premise, we already get that \forall \tau_o \in \overline{\tau}, (k, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\![\tau_o]\!].
1434
                                               Therefore, it suffices to show (k, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\![ \text{Int} ]\!] given (k, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\![ \text{Nat} ]\!] which is imme-
1436
                                               diate since \mathbb{N} \subset \mathbb{Z}.
1437
                                        iii) \tau_1 \times \tau_2 \leqslant \tau_1' \times \tau_2 with \tau_1 \leqslant \tau_1' and \tau_2 \leqslant \tau_2':
1438
                                               by our premise, we get that \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \underline{\ }) and (k, \Psi, \Sigma, \ell_1) \in \mathcal{VH}^N[\![\tau_1, \mathit{fst}(\overline{\tau})]\!] and (k, \Psi, \Sigma, \ell_2) \in \mathcal{VH}^N[\![\tau_1, \mathit{fst}(\overline{\tau})]\!]
1439
                                               \mathcal{VH}^N \llbracket \tau_2, snd(\overline{\tau}) \rrbracket.
                                               We can apply IH 4) (smaller by type) to both to get (k, \Psi, \Sigma, \ell_1) \in \mathcal{VH}^N[\![\tau'_1, \mathit{fst}(\overline{\tau})]\!] and (k, \Psi, \Sigma, \ell_2) \in \mathcal{VH}^N[\![\tau'_1, \mathit{fst}(\overline{\tau})]\!]
1442
                                               \mathcal{VH}^N[\![\tau_2', snd(\overline{\tau})]\!], which is what we wanted to show.
1443
                                        iv) \tau_1 \to \tau_2 \leqslant : \tau_1' \to \tau_2' with \tau_1' \leqslant : \tau_1 and \tau_2 \leqslant : \tau_2':
1444
                                               unfolding what we want to show, let \Sigma' \supseteq \Sigma, (j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (j, \Psi').
1445
1446
                                               Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N \llbracket \tau_1' \rrbracket.
1447
                                               Let \tau_0 \leqslant : \tau_2'.
                                               We want to show (j, \Psi', \Sigma', \operatorname{app}\{\tau_0\} \ell \ell_v) \in \mathcal{EH}^N \llbracket \tau_0, \operatorname{cod}(\overline{\tau}) \rrbracket.
1449
1450
1451
                                               By IH 2) (smaller by type), we get that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N[\![\tau_1]\!].
1452
                                               We can then apply the fact that (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^N[\![\tau, \overline{\tau}]\!] to get (j, \Psi', \Sigma', \mathsf{app}\{\tau_0\} \ell \ell_v) \in \mathcal{EH}^N[\![\tau_0, cod(\overline{\tau})]\!],
1453
                                               which is what we wanted to show.
```

1508

2022-11-18 03:01. Page 29 of 1-100.

```
1458
                Lemma 4.11 (RV-Monotonicity). If \Sigma: (k, \Psi) and 0 \le j \le k and \Sigma' \supseteq \Sigma and (k-j, \Psi') \supseteq (k, \Psi) and \Sigma': (k-j, \Psi')
1459
            and (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^N[\![\overline{\tau}]\!] then (k - j, \Psi', \Sigma', \ell) \in \mathcal{VH}^N[\![\overline{\tau}]\!]
1460
1461
                PROOF. We want to show (k - j, \Psi', \Sigma', \ell) \mathcal{VH}^N[\![\overline{\tau}]\!].
1462
           Let \tau be the head of \overline{\tau} so that \overline{\tau} = [\tau, \ldots].
1463
1464
            We proceed by induction over k and \tau:
1465
                    • k = 0: The function and dynamic cases are vacuously true, and the rest follow as in the other case.
                    • k > 0:
1468
                            i) \tau = \text{Int: immediate because } \Sigma(\ell) = \Sigma'(\ell).
1469
                           ii) \tau = \text{Nat}: same as previous case.
1470
                          iii) \tau = Bool: same as previous case.
1471
1472
                          iv) \tau = \tau_1 \times \tau_2: then \Sigma'(\ell) = (\langle \ell_1, \ell_2 \rangle, ).
1473
                                 We want to show (k-j, \Psi', \Sigma', \ell_1) \in \mathcal{VH}^L[\tau_1, \overline{fst(\tau)}] and (k-j, \Psi', \Sigma', \ell_2) \in \mathcal{VH}^L[\tau_2, \overline{snd(\tau)}].
1474
                                 We have (k, \Psi, \Sigma, \ell_1) \in \mathcal{VH}^L[\tau_1, \overline{fst(\tau)}] and (k, \Psi, \Sigma, \ell_2) \in \mathcal{VH}^L[\tau_2, \overline{snd(\tau)}].
1475
                                 Both follow by IH (smaller by type).
1476
                           v) \tau = \tau_1 \rightarrow \tau_2:
1477
1478
                                 Let (j', Psi'') \supseteq (k - j, \Psi') and \Sigma'' \supseteq \Sigma' such that \Sigma'' : (j', \Psi').
                                 Let \ell_v \in dom(\Sigma'') such that (j', \Psi'', \Sigma'', \ell_v) \in \mathcal{V}^N[\![\tau_1]\!].
                                 Let \tau_0 \geqslant : \tau_2.
1481
                                 We want to show (j', \Psi'', \Sigma'', \operatorname{app}\{\tau_0\} \ell \ell_v) \in \mathcal{E}^N \llbracket \tau_0 \rrbracket.
1482
1483
                                 Since (j', \Psi'') \supseteq (k, \Psi) and \Sigma'' \supseteq \Sigma, we can apply our premise to finish the case.
1484
                          vi) \tau = *: note by downward closure, \Sigma' : (k - j - 1, \Psi').
1485
                                 Then we want to show (k-j-1,\Psi',\Sigma',\ell)\in\mathcal{V}^N[\![\ln t]\!] or (k-j-1,\Psi',\Sigma',\ell)\in\mathcal{V}^N[\![*\times\times]\!] or (k-j-1,\Psi',\Sigma',\ell)\in\mathcal{V}^N[\![*\times\times]\!]
1486
                                 1, \Psi', \Sigma', \ell) \in \mathcal{V}^N \llbracket * \rightarrow * \rrbracket.
1488
                                 We know (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\![ \text{Int} ]\!] or (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\![ * \times * ]\!] or (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\![ * \to * ]\!].
1489
                                 The case follows by the IH (smaller by index).
1490
1491
                                                                                                                                                                                                              1492
                Lemma 4.12 (Extensions Preserve Value Log Typing). If \Sigma: (k, \Psi) and 0 \le j \le k and \Sigma' \supseteq \Sigma and (k-j, \Psi') \supseteq (k, \Psi)
1493
            and \Sigma': (k-j, \Psi') and \overline{\ell \notin dom(\Sigma')} and \Sigma[\ell \mapsto (v, \underline{\hspace{0.1cm}})]: (k, \Psi[\ell \mapsto \overline{\tau}]) then \Sigma'[\ell \mapsto (v, \underline{\hspace{0.1cm}})]: (k-j, \Psi'[\ell \mapsto \overline{\tau}]).
1494
1495
                PROOF. Note that all of the conditions in \Sigma'[\overline{\ell} \mapsto (v, \underline{\ })] : (k - j, \Psi'[\overline{\ell} \mapsto \overline{\tau}]) besides those concerning the history
1496
            relation are immediate from the hypotheses.
1497
1498
1499
           Let \Sigma'' = \Sigma' \overline{[\ell \mapsto (v, )]} and let \Psi'' = \Psi' \overline{[\ell \mapsto \overline{\tau}]}.
1500
            We want to show \forall j' < k - j, and \forall \ell \in dom(\Sigma''), (j', \Psi'', \Sigma'', \ell) \in \mathcal{VH}^N \llbracket \Psi''(\ell) \rrbracket.
1501
            Note by downward closure, \Sigma'':(j',\Psi''). If \ell\in dom(\Sigma'), then we can apply Lemma 4.11 with the fact that
1502
1503
            (i', \Psi'') \supseteq (k - i, \Psi') and \Sigma'' \supseteq \Sigma'.
1504
            If \ell \notin dom(\Sigma'), then \ell \in \overline{\ell}.
1505
            Then we can apply Lemma 4.11 with the fact that (j', \Psi'') \supseteq (k, \Psi[\ell \mapsto \overline{\ell}]) and \Sigma'' \supseteq \Sigma[\ell \mapsto (v, )] to get (j', \Psi'', \Sigma'', \ell) \in \Gamma
1506
            \mathcal{VH}^N \llbracket \Psi''(\ell) \rrbracket, which is what we wanted to show.
1507
```

```
Lemma 4.13 (Later Than Preserved By Lower Steps). If (j, \Psi') \supseteq (k, \Psi) and j' \leq j then (j - j', \Psi') \supseteq (k - j', \Psi).
1509
1510
             PROOF. Unfolding the world extension definition, we need to show j - j' \le k - j' and \forall \ell \in dom(\Psi), \Psi'(\ell) = \Psi(\ell).
1511
1512
          For the first condition, since j \le k and j' \le j, j - j' \le k - j'.
1513
          For the second condition, we can unfold the hypothesis to get the statement we need.
                                                                                                                                                                               1514
1515
             Lemma 4.14 (RE-Monotonicity). If \Sigma:(k,\Psi) and 0 \le j \le k and \Sigma' \supseteq \Sigma and (k-j,\Psi') \supseteq (k,\Psi) and \Sigma':(k-j,\Psi')
1516
          and (k, \Psi, \Sigma, e) \in \mathcal{EH}^N[\![\overline{\tau}]\!] then (k - j, \Psi', \Sigma', e) \in \mathcal{EH}^N[\![\overline{\tau}]\!].
1517
1518
             PROOF. Unfolding the relation in our hypothesis, we get that there is some (\Sigma'', e'), j' such that (\Sigma, e) \longrightarrow_N^{j'} (\Sigma'', e').
          If e' = \text{Err}^{\bullet} then we're done.
1520
1521
          Otherwise, there is some (k - j', \Psi'') \supseteq (k, \Psi) such that \Sigma'' : (k - j', \Psi'') and (k - j', \Psi'', \Sigma'', e') \in \mathcal{VH}^N[\![\tau]\!].
1522
1523
          By Lemma 4.8, \Sigma'' = \Sigma \overline{[\ell \mapsto (v, \_)]}.
1524
          By the fact that \Sigma'': (k-j', \Psi'') this also means \Psi'' = \Psi \overline{[\ell \mapsto \overline{\tau}]}.
1525
1526
          We also know from \Sigma' \supseteq \Sigma that \Sigma' = \Sigma \overline{[\ell' \mapsto (v', \_)]}.
1527
          And from \Sigma': (k-i, \Psi') that \Psi' = \Psi[\ell' \mapsto \overline{\tau'}].
1528
          By alpha renaming, we can assume that \overline{\ell' \notin dom(\Sigma'')}.
1529
          Then by Lemma 4.9, we get that (\Sigma', e) \longrightarrow_N^{j'} (\Sigma'' [\ell' \mapsto (v', \_)], e').
1530
1531
          Now, unfolding the expression relation in what we want to show, we have two obligations:
1533
                a) \Sigma''[\overline{\ell' \mapsto (v', \_)}] : (k - j - j', \Psi''[\ell' \mapsto \overline{\tau'}]).
1534
                b) (k-j-j',\Psi''[\ell'\mapsto\overline{\tau'}],\Sigma''[\ell'\mapsto(v',)],e')\in\mathcal{VH}^N[\![\overline{\tau}]\!].
1535
1536
              For a) we can apply Lemma 4.12. We have a number of obligations:
1537
1538
                 i) \Sigma : (k - j, \Psi): immediate by downward closure.
1539
                ii) \Sigma'' \supseteq \Sigma: immediate.
1540
               iii) (k - j - j', \Psi'') \supseteq (k - j, \Psi): by Lemma 4.13.
1541
1542
               iv) \Sigma'': (k-j-j', \Psi'')i: immediate by downward closure.
1543
                v) \overline{\ell' \notin dom(\Sigma'')}: assumed above by alpha renaming.
               vi) \Sigma[\ell' \mapsto (v', \underline{\hspace{0.1cm}})] : (k - j, \Psi[\ell' \mapsto \overline{t'}]): this is exactly \Sigma' : (k - j, \Psi').
1546
              For b), we can apply Lemma 4.11 with the fact proven in a).
1547
1548
              Lemma 4.15 (E-V-Monotonicity). If \Sigma:(k,\Psi) and 0 \le j \le k and \Sigma' \supseteq \Sigma and (k-j,\Psi') \supseteq (k,\Psi) and \Sigma':(k-j,\Psi')
1549
1550
          then
1551
               (1) If (k, \Psi, \Sigma, e) \in \mathcal{E}^N[\![\tau]\!] then (k - j, \Psi', \Sigma', e) \in \mathcal{E}^N[\![\tau]\!]
1552
               (2) If (k, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\![\tau]\!] then (k - j, \Psi', \Sigma', \ell) \in \mathcal{V}^N[\![\tau]\!]
1553
1554
             PROOF. Proceed by simultaneous induction on k and \tau:
1555
1556
                 • k = 0: 1) follows immediately from 2).
1557
                     Proceeds similarly to the other case, but function and dynamic cases are vacuously true.
                 • k > 0:
```

```
1) Unfolding the expression relation in our hypothesis, we get that there is some (\Sigma'', e'), j' such that
1561
1562
                             (\Sigma, e) \longrightarrow_N^{j'} (\Sigma'', e').
1563
                              If e' = \text{Err}^{\bullet} then we're done.
                              Otherwise, there is some (k-j', \Psi'') \supseteq (k, \Psi) such that \Sigma'' : (k-j', \Psi'') and (k-j', \Psi'', \Sigma'', e') \in \mathcal{V}^N[\![\tau]\!].
1565
1566
1567
                              By Lemma 4.8, \Sigma'' = \Sigma \overline{[\ell \mapsto (v, \_)]}.
1568
                              By the fact that \Sigma'': (k-j', \Psi'') this also means \Psi'' = \Psi[\ell \mapsto \overline{\ell}].
                              We also know from \Sigma' \supseteq \Sigma that \Sigma' = \Sigma \overline{[\ell' \mapsto (v', \_)]}, and from \Sigma' : (k - j, \Psi') that \Psi' = \Psi \overline{[\ell' \mapsto \overline{\tau'}]}.
                              By alpha renaming, we can assume that \overline{\ell' \notin dom(\Sigma'')}.
1572
                              Then by Lemma 4.9, we get that (\Sigma', e) \longrightarrow_N^{j'} (\Sigma'' [\ell' \mapsto (v', \_)], e').
1573
1574
                              Now, unfolding the expression relation in what we want to show, we have two obligations:
1575
                                  a) \Sigma''[\ell' \mapsto (v',)] : (k-j-j', \Psi''[\ell' \mapsto \overline{\tau'}]).
1576
1577
                                  b) (k - j - j', \Psi''[\underline{\ell' \mapsto \overline{\iota'}}], \Sigma''[\underline{\ell' \mapsto (v', \_)}], e') \in \mathcal{V}^N[\![\tau]\!]
1578
                              For a) we can apply Lemma 4.12. We have a number of obligations:
1579
                                   i) \Sigma : (k - j, \Psi): immediate by downward closure.
1580
1581
                                  ii) \Sigma'' \supseteq \Sigma: immediate.
1582
                                 iii) (k - j - j', \Psi'') \supseteq (k - j, \Psi): by Lemma 4.13.
                                 iv) \Sigma'': (k-j-j', \Psi'')i: immediate by downward closure.
                                  v) \ell' \notin dom(\Sigma''): assumed above by alpha renaming.
                                 vi) \Sigma[\ell' \mapsto (v', \underline{\hspace{0.1cm}})] : (k - j, \Psi[\ell' \mapsto \overline{\tau'}]): this is exactly \Sigma' : (k - j, \Psi').
1587
                              For b), we can apply the IH 2) (not necessarily smaller by type or index) with the fact proven in a).
1588
1589
                         2) We want to show that (k - j, \Psi', \Sigma', \ell) \in \mathcal{V}^N[\![\tau]\!].
                              We case split on \tau:
                                   i) \tau = \text{Nat: then } \Sigma(\ell) = (n, \_) where n \in \mathbb{N}, so the case is immediate.
1592
1593
1594
                                  ii) \tau = tint: same as above.
1595
                                 iii) \tau = Bool: same as above.
1599
                                 iv) \tau = \tau_1 \times \tau_2: then \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \_).
1600
                                       Unfolding our hypothesis gives us (k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^N[\tau_1] and (k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^N[\tau_2].
1601
                                       Applying IH 2) (smaller by type) to both gives us (k-j, \Psi', \Sigma', \ell_1) \in \mathcal{V}^N[\tau_1] and (k-j, \Psi', \Sigma', \ell_2) \in
1602
                                       V^N \llbracket \tau_2 \rrbracket, which is sufficient to complete the case.
1603
1604
                                  v) \tau = \tau_1 \to \tau_2: Let \Sigma'' \supseteq \Sigma' and (j', \Psi'') \supseteq (k - j, \Psi') such that \Sigma'' : (j', \Psi'').
1605
                                       Let \ell_v \in dom(\Sigma'') such that (j', \Psi'', \Sigma'', \ell_v) \in \mathcal{V}^N[\![\tau_1]\!]
1606
                                       Let \tau_0 \geqslant : \tau_2.
1607
                                       We want to show (j', \Psi'', \Sigma'', \operatorname{app}\{\tau_0\} \ell \ell_v) \in \mathcal{E}^N \llbracket \tau_0 \rrbracket.
1608
```

Since \supseteq and \supseteq are both transitive, we have $\Sigma'' \supseteq \Sigma$, and $(j', \Psi'') \supseteq (k, \Psi)$.

Therefore we can apply the hypothesis to complete the case.

1609

1610

```
vi) \tau = *: we want to show (k-1, \Psi', \Sigma', \ell) \in \mathcal{V}^N[[] or \mathcal{V}^N[[] Bool[] or \mathcal{V}^N[[] * \times *] or \mathcal{V}^N[[] * \to *].
1614
                                         This follows from IH 2) (smaller by index).
1615
                                                                                                                                                                                                      1616
1617
                                                                                                (1) (k+1, \Psi, \Sigma, \text{assert } \tau_0 e) \in \mathcal{E}^N \llbracket \tau \rrbracket \text{ iff } (k, \Psi, \Sigma, e) \in \mathcal{E}^N \llbracket \tau \rrbracket.
                Lemma 4.16 (Check is a No Op in Natural).
1618
                 (2) (k+1, \Psi, \Sigma, \text{assert } \tau_0 e) \in \mathcal{EH}^V \llbracket \overline{\tau} \rrbracket \text{ iff } (k, \Psi, \Sigma, e) \in \mathcal{EH}^V \llbracket \overline{\tau} \rrbracket.
1619
1620
               Proof. By the operational semantics, (\Sigma, \operatorname{assert} \tau_0 e) \longrightarrow_N (\Sigma, e), so the statement is immediate.
1621
                                                                                                                                                                                                      (1) (k+1, \Psi, \Sigma, app\{\tau_0\} e_1 e_2) \in \mathcal{E}^N[\tau] iff (k, \Psi, \Sigma, e_1 e_2) \in
                Lemma 4.17 (App Annotations Don't Matter in Natural).
1624
                 (2) (k+1, \Psi, \Sigma, \operatorname{app}\{\tau_0\} e_1 e_2) \in \mathcal{EH}^V[[\overline{\tau}]] iff (k, \Psi, \Sigma, e_1 e_2) \in \mathcal{EH}^V[[\overline{\tau}]].
1626
1627
                PROOF. By the operational semantics, (\Sigma, app\{\tau_0\} e_1 e_2) \longrightarrow_N (\Sigma, assert \tau_0 e_1 e_2).
1628
           We can apply Lemma 4.16 to complete the proof.
                                                                                                                                                                                                      1629
1630
               Lemma 4.18 (Pairs of Semantically Well Typed Terms are Semantically Well Typed). If(k, \Psi, \Sigma, e_1) \in \mathcal{E}^N[\![\tau_1]\!]
1631
1632
           and (k, \Psi, \Sigma, e_2) \in \mathcal{E}^N \llbracket \tau_2 \rrbracket then (k, \Psi, \Sigma, \langle e_1, e_2 \rangle) \in \mathcal{E}^N \llbracket \tau_1 \times \tau_2 \rrbracket.
1633
1634
               PROOF. Unfolding the expression relation in our hypothesis about e_1, we get that there are (\Sigma, e'_1), j such that
1635
           (\Sigma, e_1) \longrightarrow_{\mathcal{N}}^{j} (\Sigma, e'_1) and (\Sigma', e'_1) is irreducible.
           If e'_1 = \operatorname{Err}^{\bullet}, then were done because the entire application steps to an error.
           Otherwise, there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi) and (k-j, \Psi', \Sigma', e'_1) \in \mathcal{V}^N \llbracket \tau_1 \rrbracket.
1638
1639
           This means e'_1 = \ell_1 for some \ell_1 \in dom(\Sigma').
1640
1641
           With this and by the OS, we get (\Sigma, \langle e_1, e_2 \rangle) \longrightarrow_N^j (\Sigma', \langle loc_1, e_2 \rangle).
1642
1643
           We can apply Lemma 4.15 to our hypothesis about e_2 to get (k-j, \Psi', \Sigma', e_2) \in \mathcal{E}^N[\![\tau_2]\!].
1644
1645
           Unfolding the expression relation, we get that there are (\Sigma',e_2'),j' such that (\Sigma',e_2)\longrightarrow_N^{j'}(\Sigma',e_2') and (\Sigma'',e_2') is
1646
           irreducible.
1647
           If e_2' = \text{Err}^{\bullet}, then were done because the entire application steps to an error.
1648
           Otherwise, there is a (k-j-j',\Psi'') \supseteq (k-j,\Psi') such that \Sigma'': (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e_2') \in \mathcal{V}^N[\![\tau_2]\!],
           which means e_2' = \ell_2 for some \ell_2 \in dom(\Sigma'').
1652
           Putting everything together we get (\Sigma, \langle e_1, e_2 \rangle) \longrightarrow_N^{j'} (\Sigma'', \langle \ell_1, \ell_2 \rangle), with \Sigma'' : (k - j - j', \Psi'').
1653
1654
           Note by OS, (\Sigma'', \langle \ell_1, \ell_2 \rangle) \longrightarrow_N (\Sigma''[\ell' \mapsto (\langle \ell_1, \ell_2 \rangle, \_)]) where \ell' \notin dom(\Sigma'').
1655
1656
           We firstly need \Sigma''[\ell' \mapsto (\langle \ell_1, \ell_2 \rangle, \_)] : (k - j - j' - 1, \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)]).
1657
           Note the only interesting part of this statement is that \forall k' < k - j - j' - 1. (k', \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)], \Sigma''[\ell' \mapsto \Psi''(\ell_2)]
1658
1659
           (\langle \ell_1, \ell_2 \rangle, \_)], \ell') \in \mathcal{VH}^N \llbracket \Psi''(\ell_1) \times \Psi''(\ell_2) \rrbracket.
1660
           This is immediate from the fact that \Sigma'':(k',\Psi'') from downward closure, and therefore that (k',\Psi'',\Sigma'',\ell_1)\in
1661
           \mathcal{VH}^{N}[\![\Psi''(\ell_1)]\!] and (k', \Psi'', \Sigma'', \ell_2) \in \mathcal{VH}^{N}[\![\Psi''(\ell_2)]\!]
1662
```

1699

1700

1701

1702 1703

1704 1705

1706 1707

1708

1709

1710

1711 1712

1713

1714

1715 1716

```
We know that (k-j, \Psi', \Sigma', \ell_1') \in \mathcal{V}^N[\![\tau_1]\!] and (k-j-j', \Psi'', \Sigma'', \ell_2) \in \mathcal{V}^N[\![\tau_2]\!], and Lemma 4.15 with down-
1665
1666
           ward closure and the store typing judgement above.
1667
           From these facts we get that (k-j-j'-1,\Psi''[\ell'\mapsto\Psi''(\ell_1)\times\Psi''(\ell_2)],\Sigma''[\ell'\mapsto(\langle\ell_1,\ell_2\rangle,\_)],\ell_1)\in\mathcal{V}^N[\![\tau_1]\!] and
1668
           (k-j-j'-1,\Psi''[\ell'\mapsto \Psi''(\ell_1)\times \Psi''(\ell_2)],\Sigma''[\ell'\mapsto \langle \ell_1,\ell_2\rangle],\ell_2)\in \mathcal{V}^N[\![\tau_2]\!].
1669
           This is sufficient to show (k-j-j'-1,\Psi''[\ell'\mapsto \Psi''(\ell_1)\times \Psi''(\ell_2)],\Sigma''[\ell'\mapsto (\langle \ell_1,\ell_2\rangle,\_)],\langle \ell_1,\ell_2\rangle)\in \mathcal{V}^N[\![\tau_1\times\tau_2]\!],
1670
1671
           which is what we wanted to prove.
1672
               Lemma 4.19 (Pairs of History Related Terms are History Related). If (k, \Psi, \Sigma, e_1) \in \mathcal{EH}^N[\![fst(\overline{\tau})]\!] and
1673
           (k, \Psi, \Sigma, e_2) \in \mathcal{EH}^N \llbracket \operatorname{snd}(\overline{\tau}) \rrbracket \text{ then } (k, \Psi, \Sigma, \langle e_1, e_2 \rangle) \in \mathcal{EH}^N \llbracket \overline{\tau} \rrbracket.
1674
               PROOF. Unfolding the erroring expression relation in our hypothesis about e_1, we get that there are (\Sigma, e'_1), j such
1676
1677
           that (\Sigma, e_1) \longrightarrow_N^j (\Sigma, e'_1) and (\Sigma', e'_1) is irreducible.
1678
           If e'_1 = \text{Err}^{\bullet}, then were done because the entire application steps to an error.
1679
           Otherwise, there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi) and (k-j, \Psi', \Sigma', e'_1) \in \mathcal{VH}^N[\![fst(\overline{\tau})]\!].
1680
           This means e'_1 = \ell_1 for some \ell_1 \in dom(\Sigma').
1681
1682
1683
           With this and by the OS, we get (\Sigma, \langle e_1, e_2 \rangle) \longrightarrow_N^j (\Sigma', \langle loc_1, e_2 \rangle).
1684
1685
           We can apply Lemma 4.14 to our hypothesis about e_2 to get (k - j, \Psi', \Sigma', e_2) \in \mathcal{EH}^N \llbracket snd(\overline{\tau}) \rrbracket.
1686
           Unfolding the erroring expression relation, we get that there are (\Sigma',e_2'), j' such that (\Sigma',e_2) \longrightarrow_N^{j'} (\Sigma',e_2') and (\Sigma'',e_2')
           is irreducible.
1689
           If e_2' = \text{Err}^{\bullet}, then were done because the entire application steps to an error.
1690
           Otherwise, there is a (k-j-j',\Psi'') \supseteq (k-j,\Psi') such that \Sigma'': (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e_2') \in
1691
           \mathcal{VH}^N[\![snd(\overline{\tau})]\!], which means e_2' = \ell_2 for some \ell_2 \in dom(\Sigma'').
1692
1693
1694
           Putting everything together we get (\Sigma, \langle e_1, e_2 \rangle) \longrightarrow_N^{j'} (\Sigma'', \langle \ell_1, \ell_2 \rangle), with \Sigma'' : (k - j - j', \Psi'').
1695
           Note by OS, (\Sigma'', \langle \ell_1, \ell_2 \rangle) \longrightarrow_N (\Sigma''[\ell' \mapsto (\langle \ell_1, \ell_2 \rangle, \_)]) where \ell' \notin dom(\Sigma'').
1696
1697
```

Note the only interesting part of this statement is that $\forall k' < k - j - j' - 1$. $(k', \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)], \Sigma''[\ell' \mapsto \Psi''(\ell_2)]$

This is immediate from the fact that $\Sigma'':(k',\Psi'')$ from downward closure, and therefore that $(k',\Psi'',\Sigma'',\ell_1)\in$

We know that $(k-j, \Psi', \Sigma', \ell'_1) \in \mathcal{VH}^N[\![fst(\overline{\tau})]\!]$ and $(k-j-j', \Psi'', \Sigma'', \ell_2) \in \mathcal{VH}^N[\![snd(\overline{\tau})]\!]$, and Lemma 4.11

From these facts we get that $(k-j-j'-1,\Psi''[\ell'\mapsto\Psi''(\ell_1)\times\Psi''(\ell_2)],\Sigma''[\ell'\mapsto(\langle\ell_1,\ell_2\rangle,_)],\ell_1)\in\mathcal{VH}^N[\![fst(\bar{\tau})]\!]$

This is sufficient to show $(k-j-j'-1,\Psi''[\ell'\mapsto\Psi''(\ell_1)\times\Psi''(\ell_2)],\Sigma''[\ell'\mapsto(\langle\ell_1,\ell_2\rangle,_)],\langle\ell_1,\ell_2\rangle)\in\mathcal{VH}^N[\![\bar{\tau}]\!]$, which

Lemma 4.20 (Applications of Semantically Well Typed Terms are Semantically Well Typed). If $(k, \Psi, \Sigma, e_f) \in$

We firstly need $\Sigma''[\ell' \mapsto (\langle \ell_1, \ell_2 \rangle,)] : (k - j - j' - 1, \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)]).$

and $(k - j - j' - 1, \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)], \Sigma''[\ell' \mapsto \langle \ell_1, \ell_2 \rangle], \ell_2) \in \mathcal{VH}^N[\![snd(\overline{\tau})]\!].$

 $\mathcal{E}^{N}[\tau \to \tau']$ and $(k, \Psi, \Sigma, e) \in \mathcal{E}^{N}[\tau]$ then $\forall \tau_0 \ge \tau', (k, \Psi, \Sigma, app\{\tau_0\} e_f e) \in \mathcal{E}^{N}[\tau_0]$.

 $(\langle \ell_1, \ell_2 \rangle, _)], \ell') \in \mathcal{VH}^N \llbracket \Psi''(\ell_1) \times \Psi''(\ell_2) \rrbracket.$

is what we wanted to prove.

2022-11-18 03:01. Page 33 of 1-100.

 $\mathcal{VH}^N\llbracket \Psi''(\ell_1) \rrbracket$ and $(k', \Psi'', \Sigma'', \ell_2) \in \mathcal{VH}^N\llbracket \Psi''(\ell_2) \rrbracket$.

with downward closure and the store typing judgement above.

```
PROOF. Unfolding the expression relation in our hypothesis about e_f, we get that there are (\Sigma', e_f'), j such that
1718
           (\Sigma,e_f) \longrightarrow_N^j (\Sigma',e_f') \text{ and } (\Sigma',e_f') \text{ is irreducible.}
1719
           If e'_f = \operatorname{Err}^{\bullet}, then we're done because the entire application steps to an error.
1720
           Otherwise, there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e'_f) \in \mathcal{V}^N[\![\tau \to \tau']\!].
1721
1722
           This means e'_f = \ell_f for some \ell_f \in dom(\Sigma').
1723
1724
           Using this, we know from the OS that (\Sigma, \operatorname{app}\{\tau_0\} e_f e) \longrightarrow_N^j (\Sigma', \operatorname{app}\{\tau_0\} \ell_f e).
1725
1726
           We can apply Lemma 4.15 with \Sigma': (k-j, \Psi') to our hypothesis about e to get (k-j, \Psi', \Sigma', e) \in \mathcal{E}^N[\![\tau]\!].
           Unfolding the expression relation, we get that there are (\Sigma'', e'), j' such that (\Sigma', e) \longrightarrow_N^{j'} (\Sigma'', e') where (\Sigma'', e') is
           irreducible.
1730
           If e' = \text{Err}^{\bullet} than we're done, because the whole application errors.
1731
1732
           Otherwise, there exists (k-j-j',\Psi'') \supseteq (k-j,\Psi') such that \Sigma'': (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e') \in \mathcal{V}^N[\![\tau]\!].
1733
           This means e' = \ell for some \ell \in dom(\Sigma'').
1734
1735
          Putting what we have together, by the OS, (\Sigma, \operatorname{app}\{\tau_0\} e_f e) \longrightarrow_N^{j+j'} (\Sigma'', (\operatorname{app}\{\tau_0\} \ell_f \ell)).
We have (k-j, \Psi', \Sigma', \ell_f) \in \mathcal{V}^N[\![\tau \to \tau']\!] and (k-j-j', \Psi'') \supseteq (k-j, \Psi') and \Sigma'' \supseteq \Sigma' and \Sigma'' : (k-j-j', \Psi'')
1736
1737
1738
           and \tau_0 \geqslant : \tau'.
1739
           We can combine these to get (k - j - j', \Psi'', \Sigma'', app\{\tau_0\} \ell_f \ell) \in \mathcal{E}^N[\![\tau_0]\!]
1740
1741
           This is sufficient to complete the proof.
                                                                                                                                                                                              COROLLARY 4.21. If (k, \Psi, \Sigma, \ell) \in \mathcal{E}^N[\![*]\!] and \Sigma(\ell) = w and (k, \Psi, \Sigma, e) \in \mathcal{E}^N[\![*]\!] then (k - 1, \Psi, \Sigma, \mathsf{app}\{*\} w e) \in \mathcal{E}^N[\![*]\!]
1743
1744
           \mathcal{E}^{N}[\![*]\!].
1745
               Lemma 4.22 (Applications of History Related Terms are History Related). If (k, \Psi, \Sigma, e_f) \in \mathcal{EH}^N[\![\tau, \overline{\tau}]\!] and
1746
1747
           (k, \Psi, \Sigma, e) \in \mathcal{E}^N \llbracket dom(\tau) \rrbracket then \forall \tau_0 \geqslant : cod(tau), (k, \Psi, \Sigma, app\{\tau_0\} e_f e) \in \mathcal{EH}^N \llbracket \tau_0, cod(\overline{\tau}) \rrbracket.
1748
1749
              Proof. Unfolding the erroring expression relation in our hypothesis about e_f, we get that there are (\Sigma', e_f'), j such
1750
           that (\Sigma, e_f) \longrightarrow_N^j (\Sigma', e_f') and (\Sigma', e_f') is irreducible.
1751
           If e_f' = \operatorname{Err}^{\bullet}, then we're done because the entire application steps to an error.
1752
           Otherwise, there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e'_f) \in \mathcal{VH}^N[[\tau, \overline{\tau}]].
1753
1754
           This means e'_f = \ell_f for some \ell_f \in dom(\Sigma').
1756
           Using this, we know from the OS that (\Sigma, \operatorname{app}\{\tau_0\} e_f e) \longrightarrow_N^j (\Sigma', \operatorname{app}\{\tau_0\} \ell_f e).
1757
1758
1759
           We can apply Lemma 4.15 with \Sigma': (k-j, \Psi') to our hypothesis about e to get (k-j, \Psi', \Sigma', e) \in \mathcal{E}^N \llbracket dom(\tau) \rrbracket.
1760
           Unfolding the expression relation, we get that there are (\Sigma'', e'), j' such that (\Sigma', e) \longrightarrow_N^{j'} (\Sigma'', e') where (\Sigma'', e') is
1762
           irreducible.
1763
           If e' = \text{Err}^{\bullet} than we're done, because the whole application errors.
1764
           Otherwise, there exists (k-j-j',\Psi'') \supseteq (k-j,\Psi') such that \Sigma'' : (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e') \in \mathcal{V}^N[\![\tau]\!].
1765
           This means e' = \ell for some \ell \in dom(\Sigma'').
1766
```

```
Putting what we have together, by the OS, (\Sigma, \operatorname{app}\{\tau_0\} e_f e) \longrightarrow_N^{j+j'} (\Sigma'', (\operatorname{app}\{\tau_0\} \ell_f \ell)).
1769
            We have (k-j, \Psi', \Sigma', \ell_f) \in \mathcal{V}^N[\tau \to \tau'] and (k-j-j', \Psi'') \supseteq (k-j, \Psi') and \Sigma'' \supseteq \Sigma' and \Sigma'' : (k-j-j', \Psi'')
1770
1771
            and \tau_0 \geqslant : \tau'.
1772
            We can combine these to get (k - j - j', \Psi'', \Sigma'', app\{\tau_0\} \ell_f \ell) \in \mathcal{EH}^N[\![\tau_0, cod(\overline{\tau})]\!].
1773
1774
            This is sufficient to complete the proof.
                                                                                                                                                                                                                   1775
1776
                 COROLLARY 4.23. If (k, \Psi, \Sigma, e_f) \in \mathcal{EH}^N[\![*, \overline{\tau}]\!] and (k-1, \Psi, \Sigma, e) \in \mathcal{E}^N[\![*]\!] then (k-1, \Psi, \Sigma, \operatorname{app}\{\tau_0\} e_f e) \in \mathcal{EH}^N[\![*]\!]
1777
            \mathcal{EH}^N[\![*,cod(\overline{\tau})]\!].
1778
                                                                                                                                                             (1) If (k, \Psi, \Sigma, e) \in \mathcal{E}^N[\![\tau]\!] then
                Lemma 4.24 (Expression Relation implies Expression History Relation).
1780
                          (k, \Psi, \Sigma, e) \in \mathcal{EH}^N \llbracket \tau \rrbracket.
1781
                   (2) If (k, \Psi, \Sigma, \ell) \in \mathcal{V}^N \llbracket \tau \rrbracket then (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^N \llbracket \tau \rrbracket.
1782
1783
1784
                PROOF. Proceed by induction on k and \tau:
1785
                     • k = 0: 1) is immediate from 2).
1786
1787
1788
                             - \tau = Int: immediate.
1789
                             -\tau = \tau_1 \times \tau_2: then \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \_).
1790
                                 The case follows from the IH on \ell_1 and \ell_2.
                             - \tau = \tau_1 \rightarrow \tau_2: vacuously true.
1793
                             - \tau = *: vacuously true.
1794
                     • k > 0: 1) is immediate from 2).
1795
1796
1797
                            -\tau = Int: immediate.
                            -\tau = \tau_1 \times \tau_2: then \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \_).
                                 The case follows from the IH on \ell_1 and \ell_2.
1800
                             - \tau = \tau_1 \rightarrow \tau_2: Follows from 1) from the IH (smaller by index).
1801
1802
                             - \tau = *: Follows from 2) from the IH (smaller by index), using * × *, * \rightarrow *, or Int.
1803
                                                                                                                                                                                                                   Lemma 4.25 (Monitor Compatibility). If \Sigma:(k,\Psi), then
1806
1807
                   (1) If (k, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\tau] and \Sigma(\ell') = (\ell, some(\tau', \tau)), then (k, \Psi, \Sigma, \ell') \in \mathcal{V}^N[\tau']
1808
                   (2) If (k, \Psi, \Sigma, e) \in \mathcal{E}^N \llbracket \tau \rrbracket then (k, \Psi, \Sigma, \text{mon } \{\tau' \Leftarrow \tau\} e) \in \mathcal{E}^N \llbracket \tau' \rrbracket.
1809
                   (3) If (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^N[\![\Psi(\ell)]\!] and \Psi(\ell) = [\tau_s, \ldots] and \tau \geqslant : \tau_s and \Sigma' = \Sigma[\ell' \mapsto (\ell, \mathsf{some}(\tau', \tau))] and \Psi' = (\ell, \mathsf{some}(\tau', \tau))
1810
1811
                          [\ell' \mapsto \tau', \tau, \Psi(\ell)] \Psi and \ell' \notin dom(\Sigma) and \vdash \Sigma' then (k, \Psi', \Sigma', \ell') \in \mathcal{VH}^N \llbracket \tau', \tau, \Psi(\ell) \rrbracket
1812
                   (4) If (k, \Psi, \Sigma, e) \in \mathcal{EH}^N[\![\overline{\tau}]\!] and \overline{\tau} = [\tau, \ldots] then (k, \Psi, \Sigma, \text{mon } \{\tau' \Leftarrow \tau\} e) \in \mathcal{EH}^N[\![\tau', \tau, \overline{\tau}]\!]
1813
1814
```

PROOF. Proceed by simultaneous induction on k and τ .

- k = 0: 2) and 4) follow from 1) and 3) respectively.
 The proofs follow similarly to the other case, but any function or dynamic cases are vacuously true.
- k > 0:

1815 1816

1817

1818

```
1) Unfolding the relation in the statement we want to prove, note from our hypothesis about \Sigma, we get that
1821
1822

    Σ.

1823
                                Proceed by case analysis on \tau':
1824
                                     a) \tau' = \text{Nat: Since} \vdash \Sigma, we have pointsto(\Sigma, \ell') \sim \text{Nat.}
1825
                                          Therefore, we have pointsto(\Sigma, \ell') \in \mathbb{N}, which is sufficient to complete the case.
1826
1827
                                     b) \tau' = \text{Int: same reasoning as Nat.}
1828
                                     c) \tau' = Bool: same reasoning as Nat.
                                     d) \tau' = \tau'_1 \times \tau'_2: By the fact that \vdash \Sigma, this case is a contradiction.
                                     e) \tau' = \tau'_1 \to \tau'_2: Unfolding the value relation, let \Sigma' \supseteq \Sigma, and (j, \Psi') \supseteq (k, \Psi), such that \Sigma' : (j, \Psi').
                                          Let \ell_v such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N \llbracket dom(\tau') \rrbracket.
1833
                                          Let \tau_0 \leqslant : cod(\tau').
1834
                                          We want to show (j, \Psi', \Sigma', \operatorname{app}\{\tau_0\} \ell' \ell_v) \in \mathcal{E}^N \llbracket \tau_0 \rrbracket.
1835
                                          Note by the operational semantics, (\Sigma', \operatorname{app}\{\tau_0\} \ell' \ell_v) \longrightarrow_N^2
1836
1837
                                          (\Sigma', \operatorname{assert} \tau_0 \pmod{\{\operatorname{cod}(\tau') \Leftarrow \operatorname{cod}(\tau)\}} (\ell \pmod{\{\operatorname{dom}(\tau) \Leftarrow \operatorname{dom}(\tau')\}} \ell_v))).
1838
                                          Note by downward closure we have \Sigma' : (j - 2, \Psi').
1839
                                          Therefore it suffices to show (j-2, \Psi', \Sigma', \text{assert } \tau_0 \pmod{\{cod(\tau') \Leftarrow cod(\tau)\}} (\ell \pmod{\{dom(\tau) \Leftarrow dom(\tau')\}} \ell_v)))) \in \mathcal{C}(\tau)
1840
                                          \mathcal{E}^N \llbracket \tau_0 \rrbracket.
1841
1842
                                          Note that \tau_0 \geqslant : cod(\tau').
                                          By Lemma 4.10, it suffices to show (j-2, \Psi', \Sigma', \text{assert } \tau_0 \pmod{\{cod(\tau') \Leftarrow cod(\tau)\}} (\ell \pmod{\{dom(\tau) \Leftarrow dom(\tau')\}} \ell_v)))) \in \mathcal{L}_{\mathcal{L}_{q,q}}
                                          \mathcal{E}^N \llbracket cod(\tau') \rrbracket.
1847
1848
                                          By Lemma 4.16, it suffices to show (j-3, \Psi', \Sigma', \text{mon } \{cod(\tau') \Leftarrow cod(\tau')\} \ (\ell \ (\text{mon } \{dom(\tau) \Leftarrow dom(\tau')\} \ \ell_0))) \in \{cod(\tau'), \ell_0\} \}
1849
                                          \mathcal{E}^N \llbracket cod(\tau') \rrbracket.
1850
1851
1852
                                          By IH 2) (smaller by type), it suffices to show (j-3, \Psi', \Sigma', \ell \pmod{dom(\tau)} \Leftarrow dom(\tau') \} \ell_v)) \in
1853
                                          \mathcal{E}^N \llbracket cod(\tau') \rrbracket.
1854
1855
                                          By Lemma 4.17, it suffices to show (j-2, \Psi', \Sigma', \mathsf{app}\{cod(\tau')\} \ell (\mathsf{mon}\{dom(\tau) \Leftarrow dom(\tau')\} \ell_{v})) \in
                                          \mathcal{E}^N \llbracket cod(\tau') \rrbracket.
                                          We now have two cases:
1860
                                              i) \tau = *:
1861
1862
                                                   Then by Lemma 4.21 it suffices to show (j-1, \Psi', \Sigma', \ell) \in \mathcal{V}^N[\![*]\!] and (j-1, \Psi', \Sigma', \text{mon } \{dom(\tau) \Leftarrow dom(\tau')\} \ell_p) \in \mathcal{V}^N[\![*]\!]
1863
                                                   \mathcal{E}^N \llbracket dom(\tau') \rrbracket.
1864
                                                   Both follow by Lemma 4.15, and IH 2) (smaller by index) in the second case.
1865
                                             ii) \tau = \tau_1 \rightarrow \tau_2:
1866
1867
                                                   Then by Lemma 4.20 it suffices to show (j-2, \Psi', \Sigma', \ell) \in \mathcal{V}^N[\tau] and (j-2, \Psi', \Sigma', \text{mon } \{dom(\tau) \Leftarrow dom(\tau')\} \ell_v) \in \mathcal{V}^N[\tau]
1868
                                                   \mathcal{E}^N \llbracket dom(\tau') \rrbracket.
                                                   Both follow by Lemma 4.15, and IH 2) (smaller by index) in the second case.
```

```
f) \tau' = *: Unfolding the relation in what we want to show, we want to show (k, \Psi, \Sigma, \ell') \in \mathcal{V}^N[\![ Int ]\!]
1873
1874
                                         or \mathcal{V}^N [Bool] or \mathcal{V}^N [* \times *] or \mathcal{V}^N [* \to *].
1875
                                         In each case, we can apply IH 1) (smaller by index) to complete the case.
                          2) Unfolding the expression relation in our hypothesis, we have that there are (e', \Sigma'), j such that (e, \Sigma) \longrightarrow_N^J
1877
                               (e', \Sigma') with (e', \Sigma') irreducible.
1878
1879
                               If e' = \text{Err}^{\bullet} then we're done, because the monitor will step to an error as well.
                               Otherwise, there is (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{V}^N[\![\tau]\!].
                               This means \exists \ell \in dom(\Sigma') such that e' = \ell.
1884
                               If \neg \text{pointsto}(\Sigma', \ell) \sim \tau', then (\Sigma, \text{mon}\{\tau' \Leftarrow \tau\} e) \longrightarrow_N^j (\Sigma', \text{mon}\{\tau' \Leftarrow \tau\} \ell) \longrightarrow_N (\Sigma', \text{TypeErr}(\tau', \ell)),
1885
                               so we're done.
1886
                               Otherwise, we have pointsto(\Sigma', \ell) \sim \tau', and since pointsto(\Sigma', \ell) \sim \tau, we also have \tau \sim \tau'.
1887
1888
1889
                               We have 5 cases:
1890
                                   (a) \tau' = \text{Nat}:
1891
                                         Then (\Sigma', \text{mon } \{\text{Nat} \Leftarrow \tau\} \ell) \longrightarrow_N (\Sigma'[\ell' \mapsto (\ell, \text{some}(\text{Nat}, \tau))], \ell').
1892
                                         It suffices to show (k-j-1, \Psi'[\ell' \mapsto \mathsf{Nat}, \tau, \Psi(\ell)], \Sigma'[\ell' \mapsto (\ell, \mathsf{some}(\mathsf{Nat}, \tau))], \ell) \in \mathcal{V}^N[\![\mathsf{Nat}]\!],
1893
                                         and that \Sigma'[\ell' \mapsto (\ell, \mathsf{some}(\mathsf{Nat}, \tau))] : (k - j - 1, \Psi'[\ell' \mapsto \mathsf{Nat}, \tau, \Psi(\ell)]).
                                         The first follows from downward closure, and the fact that \Sigma'(\ell) \sim \text{Nat means } \Sigma'(\ell) = n.
                                         The second follows from IH 3) (smaller by index).
                                   (b) \tau' = Int: Essentially the same as Nat.
1899
                                   (c) \tau' = Bool: Essentially the same as Nat.
1900
                                   (d) \tau' = \tau'_1 \times \tau'_2:
1901
                                         By the fact that fst(\Sigma'(\ell)) \sim \tau_1' \times \tau_2', we have that \Sigma'(\ell) = (\langle \ell_1, \ell_2 \rangle, \_).
1902
                                         Then by the OS we have that (\Sigma', \text{mon } \{\tau' \leftarrow \tau\} \ell) \longrightarrow_N (\Sigma', (\text{mon } \{\tau'_1 \leftarrow fst(\tau)\} \ell_1, \text{mon } \{\tau'_2 \leftarrow snd(\tau)\} \ell_2)).
1904
                                         By downward closure, we get \Sigma' : (k - j - 1, \Psi').
1905
                                         By Lemma 4.18, it suffices to show (k-j-1,\Psi',\Sigma',\mathsf{mon}\,\{\tau_1' \Leftarrow \mathit{fst}(\tau)\}\,\ell_1) \in \mathcal{E}^N[\![\tau_1']\!] and (k-j-1)
1906
                                         1, \Psi', \Sigma', \text{mon } \{\tau_2' \Leftarrow snd(\tau)\} \ell_2\} \in \mathcal{E}^N[\![\tau_2']\!].
1907
                                         If \tau = \tau_1 \times \tau_2, then we have (k - j, \Psi', \Sigma', \ell_1) \in \mathcal{V}^N[\![\tau_1]\!], and (k - j, \Psi', \Sigma', \ell_2) \in \mathcal{V}^N[\![\tau_2]\!].
1910
                                         Then we just need to apply IH 2) (smaller by type) and Lemma 4.15.
1911
1912
                                         If \tau = *, then we have (k - j, \Psi', \Sigma', \langle \ell_1, \ell_2 \rangle) \in \mathcal{V}^N[\![*]\!].
1913
1914
                                         This means (k - j - 1, \Psi', \Sigma', \langle \ell_1, \ell_2 \rangle) \in \mathcal{V}^N \llbracket * \times * \rrbracket.
1915
                                         Therefore (k - j - 1, \Psi', \Sigma', \ell_1) \in \mathcal{V}^N[\![*]\!], and (k - j - 1, \Psi', \Sigma', \ell_2) \in \mathcal{V}^N[\![*]\!]
1916
                                         Then we just need to apply IH 2) (smaller by index).
1917
                                   (e) \tau' = \tau'_1 \rightarrow \tau'_2:
1918
1919
                                         By the fact that \tau \sim \tau', and by the OS, we have (\Sigma', \text{mon } \{\tau' \Leftarrow \tau\} \ell) \longrightarrow_N (\Sigma'[\ell' \mapsto (\ell, \text{some}(\tau', \tau))])
1920
                                         for \ell' \notin dom(\Sigma').
                                         Let \Sigma'' = \Sigma'[\ell' \mapsto (\ell, \mathsf{some}(\tau', \tau))], and \Psi'' = \Psi'[\ell' \mapsto [\tau', \tau, \Psi'(\ell)].
                                         We want to show \Sigma'' : (k - j - 2, \Psi'').
```

```
To start, the condition on entries in the value log is immediate.
1926
                                       Otherwise the only interesting case is the value history relation.
1927
                                       Let k' < k - j - 2.
                                       Then by downward closure, we get \Sigma' : (k', \Psi').
1929
                                       By IH 3) (smaller by index), we get (k', \Psi'', \Sigma'', \ell') \in \mathcal{VH}^N[\![\tau', \tau, \Psi(\ell)]\!], which is sufficient.
1930
1931
                                       Then we just need to apply IH 1) (smaller by index).
1932
                                 (f) \tau' = *: case spit on the shape of pointsto(\Sigma', \ell):
                                           i) pointsto(\Sigma', \ell) = i: the proof follows identically to the Nat case.
                                          ii) pointsto(\Sigma', \ell) = b: the proof follows identically to the Bool case.
                                        iii) pointsto(\Sigma', \ell) = \lambda x : _. e: then by the operational semantics, (\Sigma', mon {* \Leftarrow \tau} \ell) \longrightarrow_N
1937
                                               (\Sigma'[\ell' \mapsto (\ell, some(*, \tau))], \ell').
1938
                                               Therefore we want to show:
1939
                                               -\Sigma'[\ell' \mapsto (\ell, \mathsf{some}(*, \tau))] : (k - j - 2, \Psi'[\ell' \mapsto [*, \tau, \Psi'(\ell)]])
1940
1941
                                               -(k-j-2,\Psi'[\ell'\mapsto [*,\tau,\Psi'(\ell)]],\Sigma'[\ell'\mapsto (\ell,\mathsf{some}(*,\tau))],\ell')\in\mathcal{V}^N[\![*]\!]
1942
                                               The first condition follows from applications of IH 3) (smaller by index).
1943
                                               The second condition follows from an application of IH 1) (smaller by index).
1944
1945
                                         iv) pointsto(\Sigma', \ell) = \langle \ell_1, \ell_2 \rangle:
                                               By the operational semantics, either:
                                               -(\Sigma', mon \{* \Leftarrow \tau\} \ell) \longrightarrow_N (\Sigma', (mon \{* \Leftarrow fst(\tau)\} \ell_1, mon \{* \Leftarrow snd(\tau)\} \ell_2)) \text{ or }
                                               - (\Sigma', \mathsf{mon} \, \{* \Leftarrow \tau\} \, \ell) \longrightarrow_N (\Sigma', \mathsf{TypeErr}(\tau, \, \ell))
                                               In the case it errors, we're done.
1951
                                               Otherwise, it suffices to show (k - j - 1, \Psi', \Sigma', (\text{mon } \{* \Leftarrow \textit{fst}(\tau)\} \ell_1, \text{mon } \{* \Leftarrow \textit{snd}(\tau)\} \ell_2)) \in
1952
                                               \mathcal{E}^{N}[\![*]\!].
1953
                                               By Lemma 4.18, it suffices to show:
1954
                                               -(k-j-1,\Psi',\Sigma', mon\{* \Leftarrow fst(\tau)\} \ell_1) \in \mathcal{E}^N[\![*]\!]
1956
                                               -(k-j-1,\Psi',\Sigma', mon \{* \Leftarrow snd(\tau)\} \ell_2) \in \mathcal{E}^N[\![*]\!]
1957
                                               We can unfold our hypothesis that (k, \Psi, \Sigma, \ell) \mathcal{V}^N \llbracket \tau \rrbracket to get (k, \Psi, \Sigma, \langle \ell_1, \ell_2 \rangle) \in \mathcal{V}^N \llbracket \tau \rrbracket.
1958
                                               We now have two cases depending on whether \tau = * or \tau_1 \times \tau_2:
                                               - If \tau = *, then (k - 1, \Psi, \Sigma, \ell_1) \in \mathcal{V}^N[\![*]\!] and (k - 1, \Psi, \Sigma, \ell_2) \in \mathcal{V}^N[\![*]\!].
                                                   By Lemma 4.15, (k - j - 1, \Psi', \Sigma', \ell_1) \in \mathcal{V}^N[\![*]\!] and (k - j - 1, \Psi', \Sigma', \ell_2) \in \mathcal{V}^N[\![*]\!].
1962
                                                   Then we can apply IH 2) (smaller by index) to get what we need.
1963
                                               - If \tau = \tau_1 \times \tau_2, then (k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^N \llbracket \tau_1 \rrbracket and (k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^N \llbracket \tau_2 \rrbracket.
1964
                                                   By Lemma 4.15, (k - j - 1, \Psi', \Sigma', \ell_1) \in \mathcal{V}^N[\![\tau_1]\!] and (k - j - 1, \Psi', \Sigma', \ell_2) \in \mathcal{V}^N[\![\tau_2]\!].
1965
1966
                                                   Then we can apply IH 2) (smaller by index) to get what we need.
1967
                         3) We proceed by case analysis on \tau':
                                 (a) \tau' = \text{Nat: Since we already know } (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^V[N]\Psi(\ell), it suffices to show (k, \Psi, \Sigma, \ell') \in \mathcal{VH}^V[N]\Psi(\ell)
1969
                                       \mathcal{V}^N \llbracket \tau' \rrbracket and (k, \Psi, \Sigma, \ell') \in \mathcal{V}^N \llbracket \tau \rrbracket.
1970
1971
                                       This is immediate from \vdash \Sigma', which implies \tau' \sim \text{pointsto}(\Sigma', \ell') and \tau \sim \text{pointsto}(\Sigma', \ell').
1972
                                 (b) \tau' = \text{Int: same as the Nat case.}
                                 (c) \tau' = Bool: same as the Nat case.
                                 (d) \tau' = \tau'_1 \times \tau'_2: this case is a contradiction by the fact that \vdash \Sigma.
```

2022-11-18 03:01. Page 38 of 1-100.

```
(e) \tau' = \tau'_1 \to \tau'_2: Unfolding the relation in what we want to prove, let (j, \Psi') \supseteq (k, \Psi) and \Sigma' \supseteq \Sigma such
1978
                                          that \Sigma':(j,\Psi').
1979
                                          Let \tau_0 such that cod(\tau') \leq \tau_0.
                                          Let \ell_v such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N \llbracket dom(\tau') \rrbracket.
1981
                                          We want to show (j, \Psi', \Sigma', \operatorname{app}\{\tau_0\} \ell' \ell_v) \in \mathcal{EH}^N \llbracket \tau_0, \operatorname{cod}(tau), \operatorname{cod}(\Psi'(\ell)) \rrbracket.
1982
1983
                                          We know by the OS that (\Sigma', \operatorname{app}\{\tau_0\} \ell \ell_v) \longrightarrow_N (\Sigma', \operatorname{assert} \tau_0 (\ell \ell_v)) \longrightarrow_N
1984
                                          (\Sigma', \text{assert } \tau_0 \pmod{\{cod(\tau') \Leftarrow cod(\tau)\}} (\ell \pmod{\{dom(\tau) \Leftarrow dom(\tau')\}} \ell_v))).
                                          Note by downward closure, \Sigma' : (j - 2, \Psi').
                                          By Lemma 4.10, it suffices to show (j-2, \Psi', \Sigma', \text{assert } \tau_0 \pmod{\{cod(\tau') \Leftarrow cod(\tau)\}} (\ell \pmod{\{dom(\tau) \Leftarrow dom(\tau')\}} \ell_0)))
1989
                                          \in \mathcal{EH}^N \llbracket cod(\tau'), cod(\tau), cod(\Psi'(\ell)) \rrbracket
1990
1991
                                          By Lemma 4.16, it suffices to show (j-1, \Psi', \Sigma', mon \{cod(\tau') \Leftarrow cod(\tau)\} (\ell (mon \{dom(\tau) \Leftarrow dom(\tau')\} \ell_n))) \in
1992
1993
                                          \mathcal{EH}^N \llbracket cod(\tau'), cod(\tau), cod(\Psi'(\ell)) \rrbracket.
1994
1995
                                         By IH 4) (smaller by index), it suffices to show (j-1, \Psi', \Sigma', (\ell \pmod{dom(\tau)} \leftarrow dom(\tau'))) \in
1996
                                          \mathcal{EH}^N \llbracket cod(\Psi'(\ell)) \rrbracket.
1997
                                          We now have two cases:
                                             i) \tau = *: \text{By Lemma 4.23}, it suffices to show (j, \Psi', \Sigma', \ell) \in \mathcal{EH}^N[\![\Psi'(\ell)]\!] and (j-1, \Psi', \Sigma', \text{mon } \{* \Leftarrow \textit{dom}(\tau')\} \ell_v) \in \mathcal{EH}^N[\![\Psi'(\ell)]\!]
                                                 \mathcal{E}^N[\![*]\!] (since \Psi'(\ell) = [\tau, \ldots]).
2002
2003
2004
                                                 The first follows from the fact that (i, \Psi', \Sigma', \ell) \in \mathcal{VH}^N[\![\Psi'(\ell)]\!] by Lemma 4.11.
2005
                                                 For the second, by IH 2) (smaller by index), it suffices to show (j-1, \Psi', \Sigma', \ell_n) \in \mathcal{E}^N \llbracket dom(\tau') \rrbracket.
2008
                                                 This follows by Lemma 4.15 applied to the fact that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N \llbracket dom(\tau') \rrbracket.
2009
                                            ii) \tau = \tau_1 \rightarrow \tau_2:
2010
                                                 By Lemma 4.22, it suffices to show (j-1, \Psi', \Sigma', \ell) \in \mathcal{EH}^N[\![\Psi'(\ell)]\!] and (j-1, \Psi', \Sigma', \text{mon } \{dom(\tau) \Leftarrow dom(\tau')\} \ell_v) \in \mathcal{EH}^N[\![\Psi'(\ell)]\!]
2011
                                                  \mathcal{E}^N \llbracket dom(\tau) \rrbracket (since \Psi'(\ell) = [\tau, \ldots]).
2014
                                                 The first follows from the fact that (j-1, \Psi', \Sigma', \ell) \in \mathcal{VH}^N[\![\Psi'(\ell)]\!] by Lemma 4.11.
2015
2016
                                                  For the second, by IH 2) (smaller by index), it suffices to show (j-1, \Psi', \Sigma', \ell_v) \in \mathcal{E}^N \llbracket dom(\tau') \rrbracket.
2017
2018
                                                 This follows by Lemma 4.15 applied to the fact that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N \llbracket dom(\tau') \rrbracket.
2019
                                   (f) \tau' = *: unfolding the relation in what we want to show, the proof follows by IH 3) (smaller by index).
                          4) Unfolding the expression relation in our hypothesis, we have that there are (e', \Sigma'), j such that (e, \Sigma) \longrightarrow_N^J
2021
                                (e', \Sigma') with (e', \Sigma') irreducible.
2022
2023
                                If e' = \text{Err}^{\bullet} then we're done, because the monitor will step to an error as well.
                                Otherwise, there is (k - j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k - j, \Psi') and (k - j, \Psi', \Sigma', e') \in \mathcal{VH}^N[[\overline{\tau}]].
                                This means \exists \ell \in dom(\Sigma') such that e' = \ell, and \Psi'(\ell) = \overline{\tau}.
```

```
If \neg \text{pointsto}(\Sigma', \ell) \sim \tau', then (\Sigma, \text{mon } \{\tau' \Leftarrow \tau\} e) \longrightarrow_N^j (\Sigma', \text{mon } \{\tau' \Leftarrow \tau\} \ell) \longrightarrow_N (\Sigma', \text{TypeErr}(\tau', \ell)),
2029
2030
                               so we're done.
2031
                               Otherwise, we have pointsto(\Sigma', \ell) \sim \tau', and since pointsto(\Sigma', \ell) \sim \tau, we also have \tau \sim \tau'.
2032
2033
                               We want to show (k - j, \Psi', \Sigma', \text{mon } \{\tau' \Leftarrow \tau\} \ell) \in \mathcal{EH}^N \llbracket \tau', \tau, \Psi'(\ell) \rrbracket.
2034
2035
                               We have three cases:
                                   a) pointsto(\Sigma', \ell) = i: By OS, (\Sigma', mon {\tau' \leftarrow \tau} \ell) \longrightarrow_N (\Sigma'[\ell' \mapsto (\ell, some(\tau', \tau))], \ell').
                                        Let \Sigma'' = \Sigma'[\ell' \mapsto (\ell, some(\tau', \tau))] and \Psi'' = Psi'[\ell' \mapsto \tau', \tau, \Psi(\ell)].
                                        Unfolding the relation in what we want to show, it suffices to show \forall \tau_z \in \Psi''(\ell), (k-j-1, \Psi'', \Sigma'', \ell) \in \Psi''(\ell)
                                        \mathcal{V}^N \llbracket \tau_z \rrbracket and \Sigma'' : (k - j - 1, \Psi'').
2041
2042
                                        For the second, we can apply IH 3) (smaller by index).
2043
                                        For the first, by downward closure, by Lemma 4.11, (k - j - 1, \Psi'', \Sigma'', \ell) \in \mathcal{VH}^N[\![\Psi'(\ell)]\!].
2044
2045
                                        Then we already know (k - j - 1, \Psi'', \Sigma'', \ell) \in \mathcal{V}^N[\tau_z] when \tau_z \in \Psi'(\ell).
2046
                                        So it suffices to show (k - j - 1, \Psi'', \Sigma'', \ell) \in \mathcal{V}^N[\tau'].
2047
2048
2049
                                        If \tau' = Int, then we're done.
                                        Otherwise, \tau' = *, in which case we need to show (k - j - 2, \Psi'', \Sigma'', \ell') \in \mathcal{V}^N [Int], which is also
                                   b) pointsto(\Sigma', \ell) = b: essentially the same as the previous case.
                                   c) \Sigma'(\ell) = \langle \ell_1, \ell_2 \rangle:
2055
                                        By OS, (\Sigma', \text{mon } \{\tau' \Leftarrow \tau\} \ell) \longrightarrow_N (\Sigma', (\text{mon } \{fst(\tau') \Leftarrow fst(\tau)\} \ell_1, \text{mon } \{snd(\tau') \Leftarrow snd(\tau)\} \ell_2)).
2056
                                        Note by downward closue, \Sigma' : (k - j - 2, \Psi').
2057
                                        By Lemma 4.19, it suffices to show (k-j-2, \Psi', \Sigma', \text{mon } \{fst(\tau') \Leftarrow fst(\tau)\} \ell_1) \in \mathcal{EH}^N \llbracket fst(\tau'), fst(\Psi'(\ell)) \rrbracket
2058
                                        and (k - j - 2, \Psi', \Sigma', \text{mon } \{snd(\tau') \Leftarrow snd(\tau)\} \ell_1) \in \mathcal{EH}^N \llbracket snd(\tau'), snd(\tau), snd(\Psi'(\ell)) \rrbracket.
2060
                                        Both of these follow by unfolding the relation in the hypothesis about \ell, applying Lemma 4.14, and
2061
                                        applying IH 4) (smaller by index).
                                   d) pointsto(\Sigma', \ell) = \lambda x : _. e:
                                        By OS, (\Sigma', \text{mon } \{\tau' \Leftarrow \tau\} \ell) \longrightarrow_N (\Sigma'[\ell' \mapsto (\ell, \text{some}(\tau', \tau))], \ell'), where \ell' \notin dom(\Sigma').
                                        Then let \Sigma'' = \Sigma'[\ell' \mapsto (\ell, some(\tau', \tau))] and let \Psi'' = \Psi'[\ell' \mapsto \tau', \tau, \Psi'(\ell)].
                                        By IH 3) (smaller by index) we get (k-j-2,\Psi'',\Sigma'',\ell') \in \mathcal{VH}^N[\![\tau',\tau,\Psi'(\ell)]\!], so all that's left is to
                                        show is \Sigma'' : (k - j - 2, \Psi'').
2068
2069
2070
                                        Let k' < k - j - 2.
2071
                                        Note by downward closure, \Sigma': (k', \Psi'), so \forall \ell'' \in dom(\Sigma'), by Lemma 4.11, (k', \Psi'', \Sigma'', \ell'') \in
                                        \mathcal{VH}^N \llbracket \Psi''(\ell'') \rrbracket (note \Psi'(\ell'') = \Psi''(\ell'')).
2073
                                        So the final condition is (k', \Psi'', \Sigma'', \ell') \in \mathcal{VH}^N[\![\Psi''(\ell')]\!], which follows from IH 3) (smaller by
2074
2075
                                        index).
2076
```

4.3.3 Compatability Lemmas

```
2083
                Lemma 4.26 (T-Var compatibility). \frac{\llbracket (x : \tau) \in \Gamma \rrbracket}{\llbracket \Gamma \vdash x : \tau \rrbracket}
2085
2086
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma] such that \Sigma : (k, \Psi).
2087
            We want to show (k, \Psi, \Sigma, \gamma(x)) \in \mathcal{E}^N \llbracket \tau \rrbracket.
2088
            Since x : \tau \in \Gamma, we get that \gamma(x) = \ell.
            Since (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma], we get (k, \Psi, \Sigma, \ell) \in \mathcal{V}^N[\Gamma]
            Then we get that (k, \Psi, \Sigma, \ell) \in \mathcal{E}^N[\![\tau]\!] immediately since \ell is already a value and we have as a premise that \Sigma : (k, \Psi). \square
2092
2093
                Lemma 4.27 (T-Nat compatibility). \frac{}{\llbracket \Gamma \vdash n : \mathsf{Nat} \rrbracket}
2094
2095
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma] such that \Sigma : (k, \Psi).
2096
2097
            We want to show (k, \Psi, \Sigma, \gamma(n)) \in \mathcal{E}^N [\![ \text{Nat} ]\!].
2098
            Note \gamma(n) = n.
2099
            By the OS, we have (\Sigma, n) \longrightarrow_N (\Sigma[\ell \mapsto (n, \_)], \ell).
2100
            We get (k, \Psi, \Sigma, \ell) \in \mathcal{V}^N \llbracket \text{Nat} \rrbracket immediately because n \in \mathbb{N}.
2101
2102
            Since \mathcal{V}^N[\![\text{Nat}]\!] does not rely on \Psi or \Sigma, we have that (k, \Psi[\ell \mapsto [\text{Nat}]], \Sigma[\ell \mapsto (n, \_)], \ell) \in \mathcal{V}^N[\![\text{Nat}]\!].
2103
                                                                                                                                                                                                                 2104
2105
                Lemma 4.28 (T-Int compatibility).
                                                                                \boxed{ \llbracket \Gamma \vdash i : \mathsf{Int} \rrbracket }
2106
2107
                PROOF. Not meaningfully different from T-Int
2108
                                                                                                                                                                                                                 2109
                2110
2111
2112
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma] such that \Sigma : (k, \Psi).
2113
           We want to show (k, \Psi, \Sigma, \gamma(\mathsf{True})) \in \mathcal{E}^N[\![\mathsf{Bool}]\!]
2114
2115
            Note v(True) = True.
2116
            By the OS, we have (\Sigma, \mathsf{True}) \longrightarrow_N (\Sigma[\ell \mapsto (\mathsf{True}, \_)], \ell).
2117
            We get (k, \Psi, \Sigma, \mathsf{True}) \in \mathcal{V}^N \llbracket \mathsf{Bool} \rrbracket immediately.
2118
            Since \mathcal{V}^N[\![\mathsf{Bool}]\!] does not rely on \Psi or \Sigma, we have that (k, \Psi[\ell \mapsto [\mathsf{Bool}]], \Sigma[\ell \mapsto (\mathsf{True}, \_)], \ell) \in \mathcal{V}^N[\![\mathsf{Bool}]\!].
2119
2120
                                                                                                                                                                                                                 2121
                LEMMA 4.30 (T-FALSE COMPATIBILITY).
2122
2123
2124
                PROOF. Not meaningfully different from the previous case.
                                                                                                                                                                                                                 2125
2126
                Lemma 4.31 (T-Lam compatibility). \frac{\llbracket \Gamma_1, \ (x_1 \colon \tau_1) \vdash e_1 \colon \tau_2 \rrbracket}{\llbracket \Gamma_1 \vdash \lambda(x_1 \colon \tau_1). \ e_1 \colon \tau_1 \to \tau_2 \rrbracket}
2127
2128
2129
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma] such that \Sigma : (k, \Psi).
2130
            We want to show (k, \Psi, \Sigma, \gamma(\lambda x_1 : \tau_1. e_1)) \in \mathcal{E}^N \llbracket \tau_1 \to \tau_2 \rrbracket.
2131
2132
            2022-11-18 03:01. Page 41 of 1-100.
```

```
2133
            Note that \gamma(\lambda x_1 : \tau_1. e_1) = \lambda x_1 : \tau_1. \gamma(e_1).
2134
            Since \lambda x_1 : \tau_1 \cdot \gamma(e_1) is a value, by the OS we have (\Sigma, \lambda x_1 : \tau_1, \gamma(e_1)) \longrightarrow_N (\Sigma[\ell \mapsto (\lambda x_1 : \tau_1, \gamma(e_1), none)]), where
2135
2136
            We choose our later \Psi' to be \Psi[\ell \mapsto \tau_1 \to \tau_2].
2137
2138
           We now have two obligations:
2139
                  (1) (k-1, \Psi[\ell \mapsto \tau_1 \to \tau_2], \Sigma[\ell \mapsto (\lambda x_1 : \tau_1, \gamma(e_1), \mathsf{none}], \ell) \in \mathcal{V}^N[\![\tau_1 \to \tau_2]\!]
2140
2141
                  (2) \Sigma[\ell \mapsto (\lambda x_1 : \tau_1, \gamma(e_1), \text{none})] : (k - 1, \Psi[\ell \mapsto \tau_1 \rightarrow \tau_2])
2142
           For 1), unfolding the value relation:
2144
           Let (j, \Psi') \supseteq (k-1, \Psi[\ell \mapsto \tau_1 \to \tau_2]) and \Sigma' \supseteq \Sigma[\ell \mapsto (\lambda x_1 : \tau_1, \gamma(\ell_1), \mathsf{none})] such that \Sigma' : (j, \Psi').
           Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N \llbracket \tau_1 \rrbracket.
2146
           Let \tau_0 \geqslant : \tau_2.
2147
2148
           We want to show (j, \Psi', \Sigma', \operatorname{app}\{\tau_0\} \ell \ell_v) \in \mathcal{E}^N \llbracket \tau_0 \rrbracket.
2149
            By Lemma 4.17, it suffices to show (j-1, \Psi', \Sigma', \ell \ell_v) \in \mathcal{E}^N[\![\tau_0]\!].
2150
            By the OS, (\Sigma', \ell \ell_v) \longrightarrow_N (\Sigma', \gamma(e_1)[\ell_v/x]).
2151
            By the definition of substitution, \gamma(e_1)[\ell_v/x] = \gamma[x \mapsto \ell_v](e_1).
2152
            Note that (j-1, \Psi', \Sigma', \gamma[x \mapsto \ell_v](e_1)) \in \mathcal{G}^N[\Gamma, x : \tau_1]:
2153
2154
                    i) (j-1, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N [\![\tau_1]\!] by Lemma 4.15.
2155
2156
                   ii) \forall y \in dom(\gamma), (j-1, \Psi', \Sigma', \gamma(y)) \in \mathcal{V}^N[\Gamma(y)] by the premise about \gamma and Lemma 4.15.
2157
2158
            Therefore, we can apply the hypothesis to \gamma[x \mapsto \ell_n], \Psi', \Sigma', \text{ and } e_1 \text{ at } j-1 \text{ to get } (j-1, \Psi', \Sigma', \gamma[x \mapsto \ell_n](e_1)) \in \mathcal{E}^N[[\tau_2]].
2159
            Finally, we can apply Lemma 4.10 to get (j-1, \Psi', \Sigma', \gamma[x \mapsto \ell_n](e_1)) \in \mathcal{E}^N[\tau_0] which is what we wanted to show.
2160
2161
2162
                For 2), first note the domains are equal, since dom(\Sigma) = dom(\Psi).
2163
           Then note \vdash \Sigma[\ell \mapsto \lambda x_1 : \tau_1.\gamma(e_1)] since \vdash \Sigma.
2164
           Then let i < k - 1 and let \ell' \in dom(\Sigma[\ell \mapsto (\lambda x_1 : \tau_1, \gamma(e_1), none)]).
2165
            If \ell' \neq \ell, then we get the remaining conditions from \Sigma : (k, \Psi) and Lemma 4.11.
2166
2167
            If \ell' = \ell, then note the structural obligation on \Psi[\ell \mapsto [\tau_1 \to \tau_2]] is immediate.
2168
           We want to show (j, \Psi[\ell \mapsto \tau_1 \to \tau_2], \Sigma[\ell \mapsto (\lambda x_1 : \tau_1, \gamma(e_1), \mathsf{none})], \ell) \in \mathcal{VH}^N[\![\tau_1 \to \tau_2]\!].
           Let (j, \Psi') \supseteq (k-1, \Psi[\ell \mapsto \tau_1 \to \tau_2]) and \Sigma' \supseteq \Sigma[\ell \mapsto (\lambda x_1 : \tau_1, \gamma(e_1), \mathsf{none})] such that \Sigma' : (j, \Psi').
            Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N \llbracket \tau_1 \rrbracket.
2172
            Let \tau_0 \geqslant : \tau_2.
2173
            By inspection of the value relation, we get immediately that \Sigma'(\ell_p) \sim \tau_1, so we want to show (j, \Psi', \Sigma', \mathsf{app}\{\tau_0\} \ell \ell_p) \in
2174
            \mathcal{EH}^V \llbracket \tau_0 \rrbracket.
2175
            By Lemma 4.17, it suffices to show (j-1, \Psi', \Sigma', \ell \ell_v) \in \mathcal{EH}^V \llbracket \tau_0 \rrbracket.
2176
2177
            By the OS, (\Sigma', \ell \ell_v) \longrightarrow_N (\Sigma', \gamma(e_1)[\ell_v/x]).
2178
            By the definition of substitution, \gamma(e_1)[\ell_v/x] = \gamma[x \mapsto \ell_v](e_1).
2179
            Note that (j-1, \Psi', \Sigma', \gamma[x \mapsto \ell_v](e_1)) \in \mathcal{G}^N[\Gamma, x : \tau_1]:
2180
2181
                    i) (j-1, \Psi', \Sigma', \ell_v) \in \mathcal{V}^N [\tau_1] by Lemma 4.15.
2182
                   ii) \forall y \in dom(\gamma), (j-1, \Psi', \Sigma', \gamma(y)) \in \mathcal{V}^N[\Gamma(y)] by the premise about \gamma and Lemma 4.15.
2183
```

```
Therefore, we can apply the hypothesis to \gamma[x \mapsto \ell_v], \Psi', \Sigma', and e_1 at j-1 to get (j-1, \Psi', \Sigma', \gamma[x \mapsto \ell_v]) \in \mathcal{E}^N[\tau_2].
2185
2186
             Then we can apply Lemma 4.24 to get (j-1, \Psi', \Sigma', \gamma[x \mapsto \ell_v](e_1)) \in \mathcal{EH}^V[\![\tau_2]\!].
2187
             Finally, we can apply Lemma 4.10 to get (j-1, \Psi', \Sigma', \gamma[x \mapsto \ell_n](e_1)) \in \mathcal{EH}^V[\pi_0] which is what we wanted to show.
2188
2189
2190
2191
                Lemma 4.32 (T-Pair compatibility). \frac{\llbracket \Gamma_1 \vdash e_2 : \tau_2 \rrbracket}{\llbracket \Gamma_1 \vdash \langle e_1, e_2 \rangle : \tau_1 \times \tau_2 \rrbracket}
2192
2193
2194
                 PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\![\Gamma]\!] such that \Sigma : (k, \Psi).
2195
2196
            We want to show (k, \Psi, \Sigma, \gamma(\langle e_1, e_2 \rangle)) \in \mathcal{E}^N \llbracket \tau_1 \times \tau_2 \rrbracket.
2197
            Note \gamma(\langle e_1, e_2 \rangle) = \langle \gamma(e_1), \gamma(e_2) \rangle.
2198
             We can apply the first hypothesis to get (k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^N[\tau_1].
2199
             We can apply the second hypothesis to get (k, \Psi, \Sigma, \gamma(e_2)) \in \mathcal{E}^N[\![\tau_2]\!].
2200
             Then by Lemma 4.19, (k, \Psi, \Sigma, \langle \gamma(e_1), \gamma(e_2) \rangle) \in \mathcal{E}^N[\![\tau_1 \times \tau_2]\!], which is what we wanted to show.
2201
                                                                                                                                                                                                                            2202
                 Lemma 4.33 (T-App compatibility). \frac{\llbracket \Gamma_1 \vdash e_1 : \tau_1 \rightarrow \tau_2 \rrbracket \qquad \llbracket \Gamma_1 \vdash e_2 : \tau_1 \rrbracket}{\llbracket \Gamma_1 \vdash \mathsf{app} \{ \tau_2 \} \, e_1 \, e_2 : \tau_2 \rrbracket}
2203
2204
2205
2206
                 PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma] such that \Sigma : (k, \Psi).
2207
            We want to show (k, \Psi, \Sigma, \gamma(\mathsf{app}\{\tau_2\} e_1 e_2)) \in \mathcal{E}^N[\![\tau_2]\!].
            Note \gamma(\operatorname{app}\{\tau_2\} e_1 e_2) = \operatorname{app}\{\tau_2\} \gamma(e_1) \gamma(e_2).
2209
             By the first hypothesis we have (k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^N \llbracket \tau_1 \to \tau_2 \rrbracket.
2210
2211
             By the second hypothesis we have (k, \Psi, \Sigma, \gamma(e_2)) \in \mathcal{E}^N[\tau_1].
2212
            Then we can apply Lemma 4.20 to get (k, \Psi, \Sigma, \operatorname{app}\{\tau_2\} \gamma(e_1) \gamma(e_2)) \in \mathcal{E}^N[\![\tau_2]\!] which is what we wanted to show. \square
2213
2214
                 Lemma 4.34 (T-Fst compatibility). \frac{\llbracket \Gamma_1 \vdash e_1 : \tau_1 \times \tau_2 \rrbracket}{\llbracket \Gamma_1 \vdash \operatorname{fst} \{\tau_1\} e_1 : \tau_1 \rrbracket}
2215
2216
2217
                 PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N \llbracket \Gamma_1 \rrbracket such that \Sigma : (k, \Psi).
2218
            We want to show (k, \Psi, \Sigma, \gamma(\text{fst}\{\tau_1\} e_1)) \in \mathcal{E}^N \llbracket \tau_1 \rrbracket.
2219
2220
            Note \gamma(\text{fst}\{\tau_1\} e_1) = \text{fst}\{\tau_1\} \gamma(e_1).
            From the first hypothesis, we have (k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^N[\![\tau_1 \times \tau_2]\!].
2222
             Unfolding the expression relation, there are j, \Sigma', e'_1 such that (\Sigma, \gamma(e_1)) \longrightarrow_N^j (\Sigma'', e'_1) and e'_1 is irreducible.
2223
            If e'_1 = \operatorname{Err}^{\bullet} then we're done because the projection also steps to an error.
2224
             Otherwise, there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j\Psi') and (k-j, \Psi', \Sigma', e_1') \in \mathcal{V}^N \llbracket \tau_1 \times \tau_2 \rrbracket.
2225
2226
             Unfolding the location and value relations, we get that \Sigma'(e_1') = \langle \ell_1, \ell_2 \rangle.
2227
             By the OS, (\Sigma, \mathsf{fst}\{\tau_1\} e_1) \longrightarrow_N^j (\Sigma' \mathsf{fst}\{\tau_1\} e_1') \longrightarrow_N (\Sigma', \mathsf{assert} \ \tau_1 \ \ell_1) \longrightarrow_N (\Sigma', \ell_1).
2228
            We can apply Lemma 4.15 to the premise that (k-j, \Psi', \Sigma', \ell_1) \in \mathcal{V}^N[\![\tau_1]\!] to get (k-j-2, \Psi', \Sigma', \ell_1) \in \mathcal{V}^N[\![\tau_1]\!].
2229
2230
             Finally, we can apply Lemma 4.11 to get that \Sigma': (k-j-2, \Psi'), which is sufficient to complete the proof.
```

Lemma 4.35 (**T-Snd** compatibility). $\frac{\llbracket \Gamma_1 \vdash e_1 : \tau_1 \times \tau_2 \rrbracket}{\llbracket \Gamma_1 \vdash \operatorname{snd}\{\tau_2\} e_1 : \tau_2 \rrbracket}$

PROOF. Not meaningfully different from the previous lemma.

2022-11-18 03:01. Page 43 of 1-100.

```
\llbracket \Gamma_1 \vdash e_1 : \tau_1 \rrbracket
2237
2238
                LEMMA 4.36 (T-BINOP COMPATIBILITY). -
2239
2241
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma] such that \Sigma : (k, \Psi).
2242
2243
           We want to show (k, \Psi, \Sigma, \gamma(binop e_1 e_2)) \in \mathcal{E}^N \llbracket \tau_3 \rrbracket.
2244
           Note \gamma(binop e_1 e_2) = binop \gamma(e_1) \gamma(e_2).
2245
           By the first hypothesis applied to \gamma we have (k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^N \llbracket \tau_1 \rrbracket.
2246
           Unfolding we get there are j, \Sigma', e'_1 such that (\Sigma, \gamma(e_1)) \longrightarrow_N^j (\Sigma', e'_1) and e'_1 is irreducible.
2247
2248
           If e'_1 = \text{Err}^{\bullet} then we're done, because the whole operation errors.
           Otherwise there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e'_1) \in \mathcal{V}^N[\![\tau_1]\!].
2250
2251
2252
           Note by Lemma 4.15 and Lemma 4.11, we have (k-j, \Psi', \Sigma', \gamma) \in \mathcal{G}^N[\Gamma_1] and \Sigma' : (k-j, \Psi').
2253
           By the second hypothesis applied to \gamma we have (k - j, \Psi', \Sigma', \gamma(e_2)) \in \mathcal{E}^N[\![\tau_2]\!].
2254
2255
           Unfolding we get there are j', \Sigma'', e_2' such that (\Sigma', \gamma(e_2)) \longrightarrow_N^{j'} (\Sigma'', e_2') and e_2' is irreducible.
2256
           If e_2' = \text{Err}^{\bullet} then we're done, because the whole operation errors.
2257
           Otherwise, there is a (k-j-j',\Psi'') \supseteq (k-j,\Psi) such that \Sigma'': (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e_2') \in \mathcal{V}^N[\tau_2].
2258
2259
           From the definition of \Delta, \tau_3 = Int or Nat the cases proceed identically, so without loss of generality assume \tau_3 = Int.
2261
2262
           \tau_1 = \tau_2 = \text{Int}, and therefore \Sigma''(e_1') = i_1 and \Sigma''(e_2') = i_2.
2263
           If binop = \text{quotient} and i_2 = 0 then (\Sigma'', binop e'_1 e'_2) \longrightarrow_N (\Sigma'', \text{DivErr}), so we're done.
2264
           If binop = \text{quotient and } i_2 \neq 0, then (\Sigma'', binop e_1' e_2') \longrightarrow_N (\Sigma'', i_1/i_2) \longrightarrow_N (\Sigma''[\ell \mapsto (i_1/i_2, \text{none})], \ell).
2265
           Since i_1/i_2 \in \mathbb{Z}, we're done.
2266
2267
           If binop = \text{sum then } (\Sigma'', binop e'_1 e'_2) \longrightarrow_N (\Sigma'', i_1 + i_2) \longrightarrow_N (\Sigma''[\ell \mapsto (i_1 + i_2, \text{none})], \ell).
2268
           Since i_1 + i_2 \in \mathbb{Z}, we're done.
                                                                                                                                                                                                     2269
2270
                                                                                    \llbracket \Gamma_1 \vdash e_1 : \mathsf{Bool} \rrbracket
2271
                                                                                       \llbracket \Gamma_1 \vdash e_2 : \tau \rrbracket
2272
                                                                         \frac{ \llbracket \Gamma_1 \vdash e_3 : \tau \rrbracket }{ \llbracket \Gamma_1 \vdash \text{if } e_1 \text{ then } e_2 \text{ else } e_3 : \tau \rrbracket }
               LEMMA 4.37 (T-IF COMPATIBILITY).
2274
2276
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma] such that \Sigma : (k, \Psi).
2277
           We want to show (k, \Psi, \Sigma, \gamma(\text{if } e_1 \text{ then } e_2 \text{ else } e_3)) \in \mathcal{E}^N[\![\tau]\!].
2278
2279
           Note \gamma(\text{if } e_1 \text{ then } e_2 \text{ else } e_3) = \text{if } \gamma(e_1) \text{ then } \gamma(e_2) \text{ else } \gamma(e_3).
2280
           From the first hypothesis applied to \gamma, we know (k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^N \llbracket \text{Bool} \rrbracket.
2281
           Unfolding, we have that there is \Sigma', e'_1, j such that (\Sigma, e_1) \longrightarrow_N^j (\Sigma', e'_1) where e'_1 is irreducible.
2282
2283
           If e'_1 = \text{Err}^{\bullet} then we're done, because the entire if statement errors.
2284
           Otherwise, there is a (k - j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k - j, \Psi') and (k - j, \Psi', \Sigma', e'_1) \in \mathcal{V}^N [Bool]].
2285
           Unfolding the location and then the value relation, we get that pointsto(\Sigma', e'_1) = True or pointsto(\Sigma', e'_1) = False.
2286
```

• pointsto(Σ', e_1') = True: Note by OS, (Σ , if $\gamma(e_1)$ then $\gamma(e_2)$ else $\gamma(e_3)$) $\longrightarrow_N^j (\Sigma', \text{if } e_1' \text{ then } \gamma(e_2) \text{ else } \gamma(e_3)) \longrightarrow_N^j (\Sigma', \text{if } e_2' \text{ then } \gamma(e_2) \text{ else } \gamma(e_3))$ $(\Sigma', \gamma(e_2)).$ By Lemma 4.15 and Lemma 4.11, we have $(k-j-1,\Psi',\Sigma',\gamma)\in\mathcal{G}^N[\Gamma_1]$ and $\Sigma':(k-j-1,\Psi')$. From the second hypothesis, we get $(k-j-1,\Psi',\Sigma',\gamma(e_2))\in\mathcal{E}^N[\![\tau]\!]$, which is sufficient to complete the proof. • pointsto(Σ' , e'_1) = False: same as other case except replace e_2 with e_3 . $\llbracket \Gamma_1 \vdash e_1 : \tau_1 \rrbracket$ Lemma 4.38 (T-Cast compatibility). $\frac{\tau_1 \sim \tau_2}{\|\Gamma_1 \vdash \mathsf{cast} \ \{\tau_2 \Longleftarrow \tau_1\} \ e_1 : \tau_2\|}$ PROOF. Let $(k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma]$ such that $\Sigma : (k, \Psi)$. We want to show $(k, \Psi, \Sigma, \gamma(\text{cast } \{\tau_2 \leftarrow \tau_1\} e_1)) \in \mathcal{E}^N \llbracket \tau_2 \rrbracket$. Note $\gamma(\text{cast } \{\tau_2 \Leftarrow \tau_1\} e_1) = \text{cast } \{\tau_2 \Leftarrow \tau_1\} \gamma(e_1).$

By the operational semantics, $(\Sigma, \text{cast } \{\tau_2 \Leftarrow \tau_1\} \ \gamma(e_1)) \longrightarrow_N (\Sigma, \text{mon } \{\tau_2 \Leftarrow \tau_1\} \ e_1).$ By Lemma 4.11 and Lemma 4.15, $(k-1, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma]$ and $\Sigma : (k-1, \Psi)$.

By the hypothesis, $(k-1, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^N \llbracket \tau_1 \rrbracket$.

By Lemma 4.25, $(k-1, \Psi, \Sigma, \text{mon } \{\tau_2 \leftarrow \tau_1\} e_1) \in \mathcal{E}^N[\![\tau_2]\!]$, which is sufficient to complete the proof.

$$\text{Lemma 4.39 (T-Sub compatibility)}. \ \frac{\llbracket \Gamma_1 \vdash e_1 : \tau_1 \rrbracket}{\llbracket \Gamma_1 \vdash e_1 : \tau_2 \rrbracket}$$

PROOF. Let $(k, \Psi, \Sigma, \gamma) \in \mathcal{G}^N[\Gamma]$ such that $\Sigma : (k, \Psi)$.

We want to show $(k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^N \llbracket \tau_2 \rrbracket$.

From our hypothesis, we have $(k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^N \llbracket \tau_1 \rrbracket$.

We can apply Lemma 4.10 to finish the case.

4.3.4 Fundamental Property / Vigilance

Theorem 4.40 (Vigilance). If $\Gamma \vdash e : \tau$ then $\llbracket \Gamma \vdash e : \tau \rrbracket_V^N$

PROOF. By induction over the typing derivation, using the compatability lemmas.

4.4 Vigilance Fundamental Property for Transient with Truer Transient Typing

In this subsection, we use $\Gamma \vdash e : \tau$ to mean $\Gamma \vdash_{\mathsf{tru}} e : \tau$.

The relation needs to be extended with a case to handle \perp :

$$\mathcal{V}^L[\![\bot]\!] = \emptyset$$

We also edit the function cases of the relation to insert a tag into the annotation of the app, and produce a value in the meet of the tag and the result type:

$$\mathcal{VH}^{L}[\![* \to \tau_1'', \tau_2, \dots \tau_n]\!] = \{(k, \Psi, \Sigma, \ell) \mid \forall (j, \Psi') \supseteq (k, \Psi), \Sigma' \supseteq \Sigma \text{ where } \Sigma' : (j, \Psi'). \forall K.$$

$$\forall \ell_v \text{ where } (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^L \llbracket * \rrbracket.$$

$$(j, \Psi' \Sigma', \operatorname{app} \{\tau_0\} \ell \ell_v) \in \mathcal{EH}^L \llbracket [\tau_1'' \cap K, \operatorname{cod}(\tau_2), \dots \operatorname{cod}(\tau_n)] \rrbracket \}$$

$$\mathcal{V}^{L}[\![* \to \tau_2]\!] = \{(k, \Psi, \Sigma, w) \mid \forall (j, \Psi') \supseteq (k, \Psi). \ \forall \Sigma' \supseteq \Sigma \text{ where } \Sigma' : (j, \Psi').$$

$$\forall \ell \text{ where } (j, \Psi', \Sigma', \ell) \in \mathcal{V}^L \llbracket * \rrbracket . \forall K.$$

$$(j+1, \Psi', \Sigma', \operatorname{app}\{K\} \ w \ \ell) \in \mathcal{E}^L[\![\tau_2 \sqcap K]\!]\}$$

We also need to edit the $\Sigma : (k, \Psi)$ judgement because we no longer have or need a correspondence between the from type of a guard and the type underneath the guard:

$$\Sigma : (k, \Psi) \triangleq dom(\Sigma) = dom(\Psi) \land \vdash \Sigma \land \forall j < k, \ell \in dom(\Sigma).((j, \Psi, \Sigma, \ell) \in \mathcal{VH}^{L}[\![\Psi(\ell)]\!]$$
$$\land (\Sigma(\ell) = (\ell', some(\tau, \tau')) \Rightarrow \Psi(\ell) = [\tau, \tau', \Psi(\ell')] \land$$
$$\land (\Sigma(\ell) = (v, none) \land v \notin \mathbb{L} \Rightarrow \exists \tau. \Psi(\ell) = [\tau]))$$

4.4.1 Lemmas Used Without Mention

Lemma 4.41 (Stepping to Error Implies Expression Relation). If $(\Sigma, e) \longrightarrow_T^j (\Sigma', \operatorname{Err}^{\bullet})$ then $(k, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau \rrbracket$

PROOF. If k < j, then we're done because the condition in the expression relation is vacuously true.

Otherwise, we can use j as our steps, Σ' as our ending value log, and Err^{\bullet} as our irreducible expression, and we satisfy the condition in the expression relation.

 $\text{Lemma 4.42 (Stepping to Error Implies Expression History)}. \ \ \textit{If} \ (\Sigma, e) \longrightarrow_{T}^{j} (\Sigma', \mathsf{Err}^{\bullet}) \ \textit{then} \ (k, \Psi, \Sigma, e) \in \mathcal{EH}^{T}[\![\overline{\tau}]\!]$

PROOF. Similar to the previous proof.

Lemma 4.43 (Anti-Reduction - Head Expansion - Expression Relation Commutes With Steps). If $(k, \Psi', \Sigma', e') \in \mathcal{E}^T[\![\tau]\!]$ and $(\Sigma, e) \longrightarrow_T^j (\Sigma', e')$ and $\Sigma' : (k, \Psi')$ then $(k+j, \Psi, \Sigma, e) \in \mathcal{E}^T[\![\tau]\!]$

PROOF. Unfolding the expression relation in our hypothesis, there exists (Σ'', e'') , j' such that $(\Sigma', e') \longrightarrow_T^{j'} (\Sigma'', e'')$ and (Σ''', e'') is irreducible.

Either $e'' = \operatorname{Err}^{\bullet}$, in which case $(\Sigma, e) \longrightarrow_{T}^{j+j'} (\Sigma'', \operatorname{Err}^{\bullet})$, so we're done.

Otherwise, there is a $(k-j',\Psi'') \supseteq (k,\Psi')$ such that $\Sigma'' : (k-j',\Psi'')$, and $(k-j',\Psi'',\Sigma'',e'') \in \mathcal{V}^T[\![\tau]\!]$.

Using this information, we can show $(k+j, \Psi, \Sigma, e) \in \mathcal{E}^T[\![\tau]\!]$ by noting $(\Sigma, e) \longrightarrow_T^{j+j'} (\Sigma'', e'')$.

2022-11-18 03:01. Page 46 of 1-100.

Lemma 4.44 (Anti-Reduction - Head Expansion - Expression History Commutes With Steps). If $(k, \Psi', \Sigma', e') \in \mathcal{EH}^T[\![\bar{\tau}]\!]$ and $(\Sigma, e) \longrightarrow_T^j (\Sigma', e')$ and $\Sigma' : (k, \Psi')$ then $(k + j, \Psi, \Sigma, e) \in \mathcal{EH}^T[\![\bar{\tau}]\!]$

PROOF. Similar to the previous proof.

Lemma 4.45 (The Operational Semantics Preserves Well Formed Value Logs). If $\vdash \Sigma$ and $(\Sigma, e) \longrightarrow_T^* (\Sigma', e')$ then $\vdash \Sigma'$.

PROOF. The proof is immediate by inspection of the Operational Semantics.

LEMMA 4.46 (NOT ENOUGH STEPS IMPLIES ANY EXPRESSION RELATION). If $(\Sigma, e) \longrightarrow_T^k (\Sigma', e')$ and (Σ', e') is not irreducible, then $\forall j \leq k$. $(j, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau \rrbracket$ and $(j, \Psi, \Sigma, e) \in \mathcal{EH}^T \llbracket \tau \rrbracket$.

PROOF. Both conclusions are immediate, since the implications in the relations are vacuously true.

Lemma 4.47 (The Operational Semantics Only Grows Stores). If $(\Sigma, e) \longrightarrow_T^* (\Sigma', e')$ then $\Sigma' \supseteq \Sigma$.

PROOF. This is a corollary of Lemma 4.48.

4.4.2 Lemmas Used With Mention

Lemma 4.48 (The Operational Semantics Produces Value Log Extensions). If $(\Sigma, e) \longrightarrow_T^* (\Sigma', e')$, then $\exists \overline{\ell} \subseteq dom(\Sigma')$ such that $\overline{\ell} \notin dom(\overline{\Sigma})$ and $\Sigma' = \Sigma \overline{[\ell \mapsto (v, \underline{\ })]}$.

Proof. By inspection of the Operational Semantics, no steps modify the value stored in the value log, meaning $\Sigma' \supseteq \Sigma$.

And also by the inspection of the Operational Semantics, there is exactly one rule to allocate new entries in the value \log , meaning $\Sigma' \setminus \Sigma$ is a suitable choice for $\overline{[\ell \mapsto (v,_)]}$.

Lemma 4.49 (Steps are Preserved in Future Value Logs). If $(\Sigma, e) \longrightarrow_T^j (\Sigma', e')$ and $\overline{\ell \notin dom(\Sigma')}$ then $(\Sigma[\overline{\ell} \mapsto (v, _)], e) \longrightarrow_T^j (\Sigma'[\overline{\ell} \mapsto (v, _)], e')$.

PROOF. Since all of the added locations are not in Σ' , and therefore also not in Σ , no rule that will lookup a label in the derivation tree for $(\Sigma, e) \longrightarrow_T^j (\Sigma', e')$ will find a different value or type.

The only remaining notable reduction steps are those that allocate a new label and value entry, but since $\ell \notin dom(\Sigma')$, we can allocate the same entry unchanged.

Lemma 4.50 (Subtyping Preserves Logical Relations). $\forall \Sigma, k, \Psi, \tau, \tau'$. where $\Sigma : (k, \Psi)$ and $\tau \leqslant \tau'$.

- (1) If $(k, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau \rrbracket$ then $(k, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau' \rrbracket$
- (2) If $(k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau \rrbracket$ then $(k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau' \rrbracket$
- (3) If $(k, \Psi, \Sigma, e) \in \mathcal{EH}^T \llbracket \tau, \overline{\tau} \rrbracket$ then $(k, \Psi, \Sigma, e) \in \mathcal{EH}^T \llbracket \tau', \overline{\tau} \rrbracket$
- (4) If $(k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T \llbracket \tau, \overline{\tau} \rrbracket$ then $(k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T \llbracket \tau', \overline{\tau} \rrbracket$

PROOF. Proceed by mutual induction on k and τ :

• k = 0: Both 1 and 3 are immediate if $e \neq \ell$.

If $e = \ell$ then 1 and 3 follow immediately from 2 and 4.

2 and 4 follow identically in the k=0 case as they do in the k>0 case, but the function case is vacuously true. 2022-11-18 03:01. Page 47 of 1–100.

```
• k > 0:
2446
                          (1) Unfolding our hypothesis, there is some (\Sigma', e'), j such that (\Sigma, e) \longrightarrow_T^j (\Sigma', e').
2447
                                If e' = \text{Err}^{\bullet} then we're done.
2448
                                Otherwise, there is some (k-j, \Psi') \supseteq (k, \Psi') such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{V}^T \llbracket \tau \rrbracket.
2449
                                We now have two obligations:
2450
2451
                                     a) (k - j, \Psi', \Sigma', e') \in \mathcal{V}^T [\![\tau']\!].
2452
                                     b) \Sigma' : (k - i, \Psi').
                                For a) by IH 2) (not necessarily smaller by type or index), we have (k - j, \Psi', \Sigma', e') \in \mathcal{V}^T \llbracket \tau' \rrbracket, which is
                                what we wanted to show.
2456
2457
                                For b), this is immediate from the premise.
2458
                          (2) Case split on \tau \leqslant \tau':
2459
                                     i) \tau \leqslant \tau: immediate.
2460
2461
                                     ii) Nat \leqslant: Int: immediate because \mathbb{T} \subseteq \mathbb{Z}.
2462
                                    iii) \tau_1 \times \tau_2 \leqslant \tau_1' \times \tau_2', with \tau_1 \leqslant \tau_1' and \tau_2 \leqslant \tau_2':
2463
                                          We want to show (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau' \rrbracket.
2464
                                          Unfolding our hypothesis, we get that \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, ).
2465
2466
                                          We want to show (k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^T \llbracket \tau_1' \rrbracket and (k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^T \llbracket \tau_2' \rrbracket.
                                          We can apply IH 2) (smaller by type) to both of these judgements to get (k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^T[\![\tau_1']\!] and
                                          (k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^T \llbracket \tau_2' \rrbracket
2469
                                          This is sufficient to show (k, \Psi, \Sigma, \Sigma(\ell)) \in \mathcal{V}^T \llbracket \tau' \rrbracket.
2470
2471
                                    iv) * \rightarrow \tau_2 \leqslant : * \rightarrow \tau'_2, with \tau_2 \leqslant : \tau'_2:
2472
                                          We want to show (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau' \rrbracket.
2473
                                          Let (i, \Psi') \supseteq (k, \Psi) and \Sigma' \supseteq \Sigma such that \Sigma' : (i, \Psi').
2474
                                          Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^T \llbracket * \rrbracket.
2476
                                          Let K.
2477
                                          We want to show (j, \Psi', \Sigma', \operatorname{app}\{K\} \ell \ell_v) \in \mathcal{E}^T \llbracket \tau_2' \sqcap K \rrbracket.
2478
                                          Then, we can apply our hypothesis about \ell to get (j, \Psi', \Sigma', \operatorname{app}\{K\} \ell \ell_v) \in \mathcal{E}^T \llbracket \tau_2 \sqcap K \rrbracket.
2479
                                          Finally, we can apply IH 1) (smaller by type) to get (j, \Psi', \Sigma', \mathsf{app}\{K\} \ \ell \ \ell_v) \in \mathcal{E}^T \llbracket \tau_2' \sqcap K \rrbracket which is
                                          what we wanted to show.
2482
2483
                         (3) Unfolding our hypothesis, we get that there are some (\Sigma', e'), j such that (\Sigma, e) \longrightarrow_T^j (\Sigma', e') and (\Sigma', e')
2484
                                are irreducible.
2485
2486
                                If e' = \text{Err}^{\bullet}, then we're done.
2487
                                Otherwise, there is some (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{VH}^T[\![\tau, \overline{\tau}]\!],
2488
                                which means \exists \ell \in dom(\Sigma') such that e' = \ell.
2489
                                Then by IH 4) (not necessarily smaller by type or index) with \tau \leqslant \tau', we get (k-j, \Psi', \Sigma', \ell) \in \mathcal{VH}^T[\![\tau', \overline{\tau}]\!],
2490
2491
                                which is what we wanted to show.
2492
                         (4) Unfolding the history relation, we want to show (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T[\![\tau', \overline{\tau}]\!].
2493
                                We case split on \tau \leqslant \tau':
                                      i) \tau = \tau': immediate by premise.
```

```
ii) Nat ≤: Int:
2498
                                               by our premise, we already get that \forall \tau_o \in \overline{\tau}, (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau_o \rrbracket.
2499
                                               Therefore, it suffices to show (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \operatorname{Int} \rrbracket given (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \operatorname{Nat} \rrbracket which is immediately
2500
                                               ate since \mathbb{T} \subset \mathbb{Z}.
2501
                                       iii) \tau_1 \times \tau_2 \leqslant \tau_1' \times \tau_2 with \tau_1 \leqslant \tau_1' and \tau_2 \leqslant \tau_2':
2502
2503
                                               by our premise, we get that \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \underline{\ }) and (k, \Psi, \Sigma, \ell_1) \in \mathcal{VH}^T \llbracket \tau_1, fst(\overline{\tau}) \rrbracket and (k, \Psi, \Sigma, \ell_2) \in \mathcal{VH}^T \llbracket \tau_1, fst(\overline{\tau}) \rrbracket
2504
                                               \mathcal{VH}^T[\![\tau_2, snd(\overline{\tau})]\!].
                                               We can apply IH 4) (smaller by type) to both to get (k, \Psi, \Sigma, \ell_1) \in \mathcal{VH}^T[[\tau'_1, fst(\overline{\tau})]] and (k, \Psi, \Sigma, \ell_2) \in \mathcal{VH}^T[[\tau'_1, fst(\overline{\tau})]]
                                               \mathcal{VH}^T[\![\tau_2',snd(\overline{\tau})]\!], which is what we wanted to show.
2508
                                        iv) * \rightarrow \tau_2 \leqslant : * \rightarrow \tau'_2 with \tau_2 \leqslant : \tau'_2:
2509
                                               unfolding what we want to show, let \Sigma' \supseteq \Sigma, (j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (j, \Psi').
2510
                                               Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^T \llbracket * \rrbracket.
2511
                                               Let K.
2512
2513
                                               We want to show (j, \Psi', \Sigma', \operatorname{app}\{K\} \ell \ell_v) \in \mathcal{EH}^T \llbracket \tau' \sqcap K, \operatorname{cod}(\overline{\tau}) \rrbracket.
2514
2515
                                               We can then apply the fact that (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T[\![\tau, \overline{\tau}]\!] to get (j, \Psi', \Sigma', \mathsf{app}\{K\} \ell \ell_v) \in \mathcal{EH}^T[\![\tau \sqcap ]\!]
2516
2517
                                               K, cod(\overline{\tau}).
2518
                                               Then we can apply IH 3) (smaller by type) to get (j, \Psi', \Sigma', \mathsf{app}\{K\} \ell \ell_p) \in \mathcal{EH}^T \llbracket \tau' \sqcap K, cod(\overline{\tau}) \rrbracket,
2519
                                               which is what we wanted to show.
2521
                                                                                                                                                                                                                                  2522
                  Lemma 4.51 (RV-Monotonicity). If \Sigma: (k, \Psi) and 0 \le j \le k and \Sigma' \supseteq \Sigma and (k - j, \Psi') \supseteq (k, \Psi) and \Sigma': (k - j, \Psi')
2523
             and (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T[\![\overline{\tau}]\!] then (k - j, \Psi', \Sigma', \ell) \in \mathcal{VH}^T[\![\overline{\tau}]\!]
2524
2525
                  PROOF. We want to show (k - j, \Psi', \Sigma', \ell) \mathcal{VH}^T \llbracket \overline{\tau} \rrbracket.
2526
            Let \tau be the head of \overline{\tau} so that \overline{\tau} = [\tau, \ldots].
2527
2528
             We proceed by induction over k and \tau:
2529
                      • k = 0: The function and dynamic cases are vacuously true, and the rest follow as in the other case.
2530
                      • k > 0:
2531
                              i) \tau = \text{Int: immediate because } \Sigma(\ell) = \Sigma'(\ell).
2532
                             ii) \tau = \text{Nat}: same as previous case.
2534
                             iii) \tau = Bool: same as previous case.
2535
                             iv) \tau = \tau_1 \times \tau_2: then \Sigma'(\ell) = (\langle \ell_1, \ell_2 \rangle, \_).
2536
                                    We want to show (k - j, \Psi', \Sigma', \ell_1) \in \mathcal{VH}^L[\tau_1, \overline{fst(\tau)}] and (k - j, \Psi', \Sigma', \ell_2) \in \mathcal{VH}^L[\tau_2, \overline{snd(\tau)}].
2537
2538
                                    We have (k, \Psi, \Sigma, \ell_1) \in \mathcal{VH}^L[\tau_1, \overline{fst(\tau)}] and (k, \Psi, \Sigma, \ell_2) \in \mathcal{VH}^L[\tau_2, \overline{snd(\tau)}].
2539
                                    Both follow by IH (smaller by type).
2540
                              v) \tau = * \rightarrow \tau_2:
2541
                                    Let (j', Psi'') \supseteq (k - j, \Psi') and \Sigma'' \supseteq \Sigma' such that \Sigma''(j', \Psi').
2542
2543
                                    Let \ell_v \in dom(\Sigma'') such that (j', \Psi'', \Sigma'', \ell_v) \in \mathcal{V}^T[\![*]\!].
2544
2545
                                    We want to show (j', \Psi'', \Sigma'', \operatorname{app}\{K\} \ell \ell_v) \in \mathcal{E}^T \llbracket \tau_2 \sqcap K \rrbracket.
2546
                                    Since (j', \Psi'') \supseteq (k, \Psi) and \Sigma'' \supseteq \Sigma, we can apply our premise to finish the case.
2548
             2022-11-18 03:01. Page 49 of 1-100.
```

```
2550
                                Then we want to show (k-j-1,\Psi',\Sigma',\ell) \in \mathcal{V}^T \llbracket \operatorname{Int} \rrbracket or (k-j-1,\Psi',\Sigma',\ell) \in \mathcal{V}^T \llbracket *\times * \rrbracket or (k-j-1,\Psi',\Sigma',\ell) \in \mathcal{V}^T \llbracket *\times * \rrbracket
2551
                                1, \Psi', \Sigma', \ell) \in \mathcal{V}^T \llbracket * \rightarrow * \rrbracket.
2552
                                We know (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \operatorname{Int} \rrbracket or (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket * \times * \rrbracket or (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket * \to * \rrbracket.
2553
2554
                                The case follows by the IH (smaller by index).
2555
                                                                                                                                                                                                         2556
2557
               Lemma 4.52 (Extensions Preserve Value Log Typing). If \Sigma: (k, \Psi) and 0 \le j \le k and \Sigma' \supseteq \Sigma and (k-j, \Psi') \supseteq (k, \Psi)
2558
            and \Sigma': (k-i, \Psi') and \ell \notin dom(\Sigma') and \Sigma[\ell \mapsto (v, \cdot)]: (k, \Psi[\ell \mapsto \overline{t}]) then \Sigma'[\ell \mapsto (v, \cdot)]: (k-i, \Psi'[\ell \mapsto \overline{t}]).
2559
                PROOF. Note that all of the conditions in \Sigma' [\ell \mapsto (v, \underline{\ })] : (k - j, \Psi' [\ell \mapsto \overline{\tau}]) besides those concerning the history
2561
            relation are immediate from the hypotheses.
2562
2563
           Let \Sigma'' = \Sigma' \overline{[\ell \mapsto (v, \cdot)]} and let \Psi'' = \Psi' \overline{[\ell \mapsto \overline{\tau}]}.
2564
2565
           We want to show \forall j' < k - j, and \forall \ell \in dom(\Sigma''), (j', \Psi'', \Sigma'', \ell) \in \mathcal{VH}^T \llbracket \Psi''(\ell) \rrbracket.
2566
            Note by downward closure, \Sigma'': (j', \Psi''). If \ell \in dom(\Sigma'), then we can apply Lemma 4.51 with the fact that
2567
            (i', \Psi'') \supseteq (k - i, \Psi') and \Sigma'' \supseteq \Sigma'.
2568
2569
           If \ell \notin dom(\Sigma'), then \ell \in \overline{\ell}.
2570
           Then we can apply Lemma 4.51 with the fact that (j', \Psi'') \supseteq (k, \Psi[\ell \mapsto \overline{t}]) and \Sigma'' \supseteq \Sigma[\ell \mapsto (v, )] to get (j', \Psi'', \Sigma'', \ell) \in \Gamma
2571
            \mathcal{VH}^T \llbracket \Psi''(\ell) \rrbracket, which is what we wanted to show.
2572
2573
                Lemma 4.53 (Later Than Preserved By Lower Steps). If (j, \Psi') \supseteq (k, \Psi) and j' \leq j then (j - j', \Psi') \supseteq (k - j', \Psi).
2574
2575
               PROOF. Unfolding the world extension definition, we need to show j - j' \le k - j' and \forall \ell \in dom(\Psi), \Psi'(\ell) = \Psi(\ell).
2576
            For the first condition, since j \le k and j' \le j, j - j' \le k - j'.
2577
            For the second condition, we can unfold the hypothesis to get the statement we need.
2578
2579
               Lemma 4.54 (RE-Monotonicity). If \Sigma:(k,\Psi) and 0 \le j \le k and \Sigma' \supseteq \Sigma and (k-j,\Psi') \supseteq (k,\Psi) and \Sigma':(k-j,\Psi')
2580
            and (k, \Psi, \Sigma, e) \in \mathcal{EH}^T \llbracket \overline{\tau} \rrbracket then (k - j, \Psi', \Sigma', e) \in \mathcal{EH}^T \llbracket \overline{\tau} \rrbracket.
2581
2582
               Proof. Unfolding the relation in our hypothesis, we get that there is some (\Sigma'', e'), j' such that (\Sigma, e) \longrightarrow_T^{j'} (\Sigma'', e').
2583
            If e' = \text{Err}^{\bullet} then we're done.
2584
            Otherwise, there is some (k-j',\Psi'') \supseteq (k,\Psi) such that \Sigma'': (k-j',\Psi'') and (k-j',\Psi'',\Sigma'',e') \in \mathcal{VH}^T[\![\overline{\tau}]\!].
2585
            By Lemma 4.48, \Sigma'' = \Sigma \overline{[\ell \mapsto (v, \_)]}.
2588
            By the fact that \Sigma'': (k-j', \Psi'') this also means \Psi'' = \Psi[\ell \mapsto \overline{\ell}].
2589
           We also know from \Sigma' \supseteq \Sigma that \Sigma' = \Sigma \overline{[\ell' \mapsto (v', )]}.
2590
2591
            And from \Sigma': (k-i, \Psi') that \Psi' = \Psi[\ell' \mapsto \overline{\tau'}].
2592
            By alpha renaming, we can assume that \overline{\ell' \notin dom(\Sigma'')}.
2593
           Then by Lemma 4.49, we get that (\Sigma',e) \longrightarrow_T^{j'} (\Sigma'' \overline{[\ell' \mapsto (v',\_)]},e').
2594
2595
2596
           Now, unfolding the expression relation in what we want to show, we have two obligations:
2597
                   a) \Sigma''[\ell' \mapsto (v', \underline{\hspace{0.1cm}})] : (k - j - j', \Psi''[\ell' \mapsto \overline{\tau'}]).
2598
                  b) (k - j - j', \Psi'' \overline{[\ell' \mapsto \overline{\tau'}]}, \Sigma'' \overline{[\ell' \mapsto (v', \cdot)]}, e') \in \mathcal{VH}^T \llbracket \overline{\tau} \rrbracket.
2599
```

2022-11-18 03:01. Page 50 of 1-100.

vi) $\tau = *:$ note by downward closure, $\Sigma' : (k - j - 1, \Psi')$.

For a) we can apply Lemma 4.52. We have a number of obligations:

- i) $\Sigma : (k j, \Psi)$: immediate by downward closure.
- ii) $\Sigma'' \supseteq \Sigma$: immediate.

2601 2602

2603

2604

2605 2606

2607

2608

2612

2613 2614 2615

2616

2617 2618

2619 2620

2621

2622

2626

2627

2628 2629 2630

2632

2633

2634

2635

2639 2640

2641

2642

2643

2645

2646

2647

2648

2652

- iii) $(k-j-j', \Psi'') \supseteq (k-j, \Psi)$: by Lemma 4.53.
- iv) $\Sigma'': (k-j-j', \Psi'')$ i: immediate by downward closure.
- v) $\overline{\ell' \notin dom(\Sigma'')}$: assumed above by alpha renaming.
- vi) $\Sigma[\ell' \mapsto (v', \underline{\ })] : (k j, \Psi[\ell' \mapsto \overline{\iota'}])$: this is exactly $\Sigma' : (k j, \Psi')$.

For b), we can apply Lemma 4.51 with the fact proven in a).

Lemma 4.55 (E-V-Monotonicity). If $\Sigma:(k,\Psi)$ and $0 \le j \le k$ and $\Sigma' \supseteq \Sigma$ and $(k-j,\Psi') \supseteq (k,\Psi)$ and $\Sigma':(k-j,\Psi')$ then

- (1) If $(k, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau \rrbracket$ then $(k j, \Psi', \Sigma', e) \in \mathcal{E}^T \llbracket \tau \rrbracket$
- (2) If $(k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau \rrbracket$ then $(k j, \Psi', \Sigma', \ell) \in \mathcal{V}^T \llbracket \tau \rrbracket$

Proof. Proceed by simultaneous induction on k and τ :

• k = 0: 1) follows immediately from 2).

Proceeds similarly to the other case, but function and dynamic cases are vacuously true.

- k > 0:
 - 1) Unfolding the expression relation in our hypothesis, we get that there is some (Σ'', e') , j' such that $(\Sigma, e) \longrightarrow_T^{j'} (\Sigma'', e')$.

If $e' = Err^{\bullet}$ then we're done.

Otherwise, there is some $(k-j', \Psi'') \supseteq (k, \Psi)$ such that $\Sigma'' : (k-j', \Psi'')$ and $(k-j', \Psi'', \Sigma'', e') \in \mathcal{V}^T \llbracket \tau \rrbracket$.

By Lemma 4.48, $\Sigma'' = \Sigma \overline{[\ell \mapsto (v, _)]}$.

By the fact that $\Sigma'': (k-j', \Psi'')$ this also means $\Psi'' = \Psi \overline{[\ell \mapsto \overline{\tau}]}$.

We also know from $\Sigma' \supseteq \Sigma$ that $\Sigma' = \Sigma[\ell' \mapsto (v', _)]$, and from $\Sigma' : (k - j, \Psi')$ that $\Psi' = \Psi[\ell' \mapsto \overline{t'}]$.

By alpha renaming, we can assume that $\overline{\ell' \notin dom(\Sigma'')}$.

Then by Lemma 4.49, we get that $(\Sigma', e) \longrightarrow_T^{j'} (\Sigma'' [\ell' \mapsto (v', _)], e')$.

Now, unfolding the expression relation in what we want to show, we have two obligations:

- a) $\Sigma^{\prime\prime}\overline{[\ell^\prime\mapsto (v^\prime,_)]}:(k-j-j^\prime,\Psi^{\prime\prime}[\ell^\prime\mapsto \overline{\tau^\prime}]).$
- b) $(k j j', \Psi''[\ell' \mapsto \overline{\tau'}], \Sigma''[\ell' \mapsto (v', _)], e') \in \mathcal{V}^T[[\tau]]$

For a) we can apply Lemma 4.52. We have a number of obligations:

- i) $\Sigma : (k j, \Psi)$: immediate by downward closure.
- ii) $\Sigma'' \supseteq \Sigma$: immediate.
- iii) $(k-j-j',\Psi'') \supseteq (k-j,\Psi)$: by Lemma 4.53.
- iv) $\Sigma'' : (k j j', \Psi'')$ i: immediate by downward closure.
- v) $\overline{\ell' \notin dom(\Sigma'')}$: assumed above by alpha renaming.
- vi) $\Sigma[\ell' \mapsto (v', _)] : (k j, \Psi[\ell' \mapsto \overline{\tau'}])$: this is exactly $\Sigma' : (k j, \Psi')$.

For b), we can apply the IH 2) (not necessarily smaller by type or index) with the fact proven in a).

2022-11-18 03:01. Page 51 of 1-100.

```
2) We want to show that (k - j, \Psi', \Sigma', \ell) \in \mathcal{V}^T \llbracket \tau \rrbracket.
2653
2654
                               We case split on \tau:
2655
                                    i) \tau = \text{Nat: then } \Sigma(\ell) = (n, \_) where n \in \mathbb{T}, so the case is immediate.
                                   ii) \tau = tint: same as above.
2659
                                  iii) \tau = Bool: same as above.
                                  iv) \tau = \tau_1 \times \tau_2: then \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, ).
                                        Unfolding our hypothesis gives us (k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^T \llbracket \tau_1 \rrbracket and (k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^T \llbracket \tau_2 \rrbracket.
                                        Applying IH 2) (smaller by type) to both gives us (k-j, \Psi', \Sigma', \ell_1) \in \mathcal{V}^T[\tau_1] and (k-j, \Psi', \Sigma', \ell_2) \in \mathcal{V}^T[\tau_1]
2666
                                        \mathcal{V}^T \llbracket \tau_2 \rrbracket, which is sufficient to complete the case.
2667
                                   v) \tau = * \rightarrow \tau_2: Let \Sigma'' \supseteq \Sigma' and (j', \Psi'') \supseteq (k - j, \Psi') such that \Sigma'' : (j', \Psi'').
2668
                                        Let \ell_v \in dom(\Sigma'') such that (j', \Psi'', \Sigma'', \ell_v) \in \mathcal{V}^T[\![*]\!].
2670
2671
                                        We want to show (j', \Psi'', \Sigma'', \operatorname{app}\{K\} \ell \ell_v) \in \mathcal{E}^T \llbracket K \sqcap \tau_2 \rrbracket.
2672
                                        Since \supseteq and \supseteq are both transitive, we have \Sigma'' \supseteq \Sigma, and (i', \Psi'') \supseteq (k, \Psi).
2673
                                        Therefore we can apply the hypothesis to complete the case.
                                  vi) \tau = *: we want to show (k-1, \Psi', \Sigma', \ell) \in \mathcal{V}^T \llbracket \operatorname{Int} \rrbracket \text{ or } \mathcal{V}^T \llbracket \operatorname{Bool} \rrbracket \text{ or } \mathcal{V}^T \llbracket * \times * \rrbracket \text{ or } \mathcal{V}^T \llbracket * \to * \rrbracket.
                                        This follows from IH 2) (smaller by index).
2678
2679
               Lemma 4.56 (Bot Relation If and Only If Error). (k, \Psi, \Sigma, e) \in \mathcal{E}^T[\![\bot]\!] and (\Sigma, e) \longrightarrow_T^J (\Sigma', e') where (\Sigma', e') is
2680
2681
           irreducible and j \le k, iff e' = Err^{\bullet}.
2682
                Proof.
                                     • \Rightarrow: Unfolding our hypothesis about e in the expression relation, we get that either:
2684
                           -e' = Err^{\bullet} or
2685
                           -\exists (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{V}^T \llbracket \bot \rrbracket
                       Assume for sake of contradiction the second case holds.
                        (k-j, \Psi', \Sigma', e') \in \mathcal{V}^T[\![\bot]\!] implies (k-j, \Psi', \Sigma', \Sigma'(e')) \in \mathcal{V}^T[\![\bot]\!], which is a contradiction.
                       Therefore, e' = \text{Err}^{\bullet}.
                   • ⇐: immediate.
2691
2692
2693
               Lemma 4.57 (Tagmatch Makes Values In Relation At Meet). If K \sim \text{pointsto}(\Sigma, \ell) and (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T[\tau] then
2694
           (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket K \sqcap \tau \rrbracket
2695
               PROOF. There are three cases to consider:
2697
2698
                 (1) K \sqcap \tau = \bot: a contradiction.
2699
2700
                 (2) K \sqcap \tau = \tau: immediate by Lemma 4.55.
                 (3) K \sqcap \tau = K and \tau = *: immediate by unfolding the value relation in our hypothesis, and noting that whichever
                       type of Int, * \times * or * \rightarrow * we satisfy must be K.
```

2022-11-18 03:01. Page 52 of 1-100.

2756

2022-11-18 03:01. Page 53 of 1-100.

```
2706
               Lemma 4.58 (Check Makes Terms In Relation At Meet). If (k, \Psi, \Sigma, e) \in \mathcal{E}^T[\![\tau]\!] then (k, \Psi, \Sigma, assert Ke) \in \mathcal{E}^T[\![\tau]\!]
2707
2708
           \mathcal{E}^T \llbracket \tau \sqcap K \rrbracket.
2709
               Proof. Unfolding the expression relation in our hypothesis, we have that \exists e', \Sigma', j such that (\Sigma, e) \longrightarrow_T^j (\Sigma', e')
2710
2711
           and (\Sigma', e') is irreducible.
2712
           If e' = \text{Err}^{\bullet} then we're done.
2713
           Otherwise \exists (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{V}^T \llbracket \tau \rrbracket
2714
2715
           It suffices to show (k - j, \Psi', \Sigma', \text{assert } K e') \in \mathcal{E}^T \llbracket \tau \sqcap K \rrbracket.
2716
           By the OS, if \neg K \sim \text{pointsto}(\Sigma', e') then (\Sigma', \text{assert } K e') \longrightarrow_T (\Sigma', \text{Err}^{\bullet}) and we're done.
2717
           Otherwise, (\Sigma', \text{assert } K e') \longrightarrow_T (\Sigma', e') and K \sim \text{pointsto}(\Sigma', e').
2718
           By Lemma 4.57, we therefore get (k-j-1,\Psi',\Sigma',e') \in \mathcal{V}^T[\tau \sqcap K], which is sufficient to complete the proof.
2719
2720
                Lemma 4.59 (Tagmatch Makes Values In history relation At Meet). If K \sim \text{pointsto}(\Sigma, \ell) and (k, \Psi, \Sigma, \ell) \in
2721
2722
           \mathcal{VH}^T[\![\tau,\overline{\tau}]\!] then (k-1,\Psi,\Sigma,\ell)\in\mathcal{VH}^T[\![K\sqcap\tau,\overline{\tau}]\!]
2723
               PROOF. There are three cases to consider:
2724
2725
                 (1) K \sqcap \tau = \bot: a contradiction because K \sim \Sigma(\ell) and (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau \rrbracket.
2726
                 (2) K \sqcap \tau = \tau: immediate by Lemma 4.51.
                 (3) K \sqcap \tau = K and \tau = *: immediate by unfolding the erroring value relation in our hypothesis, and noting that
2730
                        whichever type of Int, * \times * or * \rightarrow * we satisfy must be K.
2731
2732
2733
               Lemma 4.60 (Check Makes Terms In History relation At Meet). If (k, \Psi, \Sigma, e) \in \mathcal{EH}^T \llbracket \tau, \overline{\tau} \rrbracket then (k, \Psi, \Sigma, assert Ke) \in \mathcal{EH}^T \llbracket \tau, \overline{\tau} \rrbracket
2734
2735
           \mathcal{E}\mathcal{H}^T \llbracket \tau \sqcap K, \overline{\tau} \rrbracket.
2736
2737
               PROOF. Unfolding the erroring expression relation in our hypothesis, we have that \exists e', \Sigma', j such that (\Sigma, e) \longrightarrow_T^J
2738
           (\Sigma', e') and (\Sigma', e') is irreducible.
2739
           If e' = \text{Err}^{\bullet} then we're done.
2740
           Otherwise \exists (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{VH}^V[\![T]\!]\tau, \overline{\tau}.
2742
           It suffices to show (k - j, \Psi', \Sigma', \text{assert } K e') \in \mathcal{EH}^T \llbracket \tau \sqcap K, \overline{\tau} \rrbracket.
2743
           By the OS, if \neg K \sim \text{pointsto}(\Sigma', e') then (\Sigma', \text{assert } K e') \longrightarrow_T (\Sigma', \text{Err}^{\bullet}) and we're done.
2744
           Otherwise, (\Sigma', \text{assert } K e') \longrightarrow_T (\Sigma', e') and K \sim \text{pointsto}(\Sigma', e').
2745
           By Lemma 4.59, we therefore get (k-j-1, \Psi', \Sigma', e') \in \mathcal{VH}^V \llbracket T \rrbracket \tau \sqcap K, \overline{\tau}, which is sufficient to complete the proof. \square
2746
2747
                Lemma 4.61 (Lattice Ordering Preserves Relation). If \tau \leq \tau' then
2748
2749
                 (1) If (k, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau \rrbracket then (k, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau' \rrbracket
2750
                 (2) If (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau \rrbracket then (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau' \rrbracket.
2751
2752
                                   (1) Unfolding the expression relation in our hypothesis, we have that \exists e', \Sigma', j such that (\Sigma, e) \longrightarrow_T^j
2753
                        (\Sigma', e') and (\Sigma', e') is irreducible.
2754
                        If e' = \text{Err}^{\bullet} then we're done.
2755
```

```
Otherwise \exists (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{V}^T \llbracket \tau \rrbracket.
2758
                         It suffices to show (k - j, \Psi', \Sigma', e') \in \mathcal{V}^T \llbracket \tau' \rrbracket, which follows by IH 2).
2759
                   (2) Proceed by induction over the lattice ordering:
2760
                          (a) \tau \leqslant \tau': follows from Lemma 4.50.
2761
                          (b) \tau = \tau_1 \times \tau_2, \tau' = \tau'_1 \times \tau'_2, \tau_1 \le \tau'_1, and \tau_2 \le \tau'_2:
2762
2763
                                 Then unfolding the location relation in our hypothesis, we have that \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \_).
2764
                                 We also have that (k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^T \llbracket \tau_1 \rrbracket and (k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^T \llbracket \tau_2 \rrbracket.
                                 Unfolding the relation in what we want to show, we want to show (k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^T \llbracket \tau_2 \rrbracket and (k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^T \llbracket \tau_2 \rrbracket
                                 \mathcal{V}^T \llbracket \tau_2' \rrbracket, which follows by IH 2).
2768
                          (c) \tau = * \rightarrow \tau_o, \tau' = * \rightarrow \tau'_o, \text{ and } \tau_o \le \tau'_o:
2769
                                 We want to show (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket * \to \tau_o' \rrbracket.
2770
                                 Let (j, \Psi') \supseteq (k, \Psi) and \Sigma' \supseteq \Sigma such that \Sigma' : (j, \Psi').
2771
                                 Let \ell_n \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_n) \in \mathcal{V}^T \llbracket * \rrbracket.
2772
                                 Let K.
2774
                                 We want to show (j, \Psi', \Sigma', \operatorname{app}\{K\} \ell \ell_v) \in \mathcal{E}^T \llbracket \tau'_o \sqcap K \rrbracket.
2775
                                 From our hypothesis, we get that (j, \Psi', \Sigma', app\{K\} \ell \ell_v) \in \mathcal{E}^T \llbracket \tau_o \sqcap K \rrbracket.
2776
                                 The proof follows from IH 1).
2777
2778
                          (d) \tau' = *: Proceed by case analysis on \tau:
                                     (i) \tau = \text{Nat: Immediate.}
                                    (ii) \tau = Int: Immediate.
2781
                                   (iii) \tau = \text{Bool}: Immediate.
2782
2783
                                   (iv) \tau = \tau_1 \times \tau_2: Then unfolding the location relation in our hypothesis, we have that \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \_).
2784
                                           We also have that (k, \Psi, \Sigma, \ell_1) \in \mathcal{V}^T \llbracket \tau_1 \rrbracket and (k, \Psi, \Sigma, \ell_2) \in \mathcal{V}^T \llbracket \tau_2 \rrbracket.
2785
                                           Unfolding the relation in what we want to show, we want to show (k-1, \Psi, \Sigma, \ell_1) \in \mathcal{V}^T[\![*]\!] and
2786
                                           (k-1, \Psi, \Sigma, \ell_2) \in \mathcal{V}^T[\![*]\!], which follows by IH 2) and Lemma 4.55.
2788
                                    (v) \tau = * \to \tau': We want to show (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket * \to * \rrbracket.
2789
                                           Let (i, \Psi') \supseteq (k, \Psi) and \Sigma' \supseteq \Sigma such that \Sigma' : (i, \Psi').
                                           Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^T [\![ * ]\!].
2791
                                           Let K.
                                           We want to show (j, \Psi', \Sigma', \operatorname{app}\{K\} \ell \ell_v) \in \mathcal{E}^T \llbracket K \rrbracket.
                                           From our hypothesis, we get that (j, \Psi', \Sigma', \operatorname{app}\{K\} \ell \ell_v) \in \mathcal{E}^T \llbracket \tau' \sqcap K \rrbracket.
2795
                                           By the IH 1), we get that (j, \Psi', \Sigma', app\{K\} \ell \ell_v) \in \mathcal{E}^T[\![K]\!] which is what we wanted to show.
2796
2797
2798
                Lemma 4.62 (Pairs of Semantically Well Typed Terms are Semantically Well Typed). If (k, \Psi, \Sigma, e_1) \in \mathcal{E}^T \llbracket \tau_1 \rrbracket
2799
            and (k, \Psi, \Sigma, e_2) \in \mathcal{E}^T \llbracket \tau_2 \rrbracket then (k, \Psi, \Sigma, \langle e_1, e_2 \rangle) \in \mathcal{E}^T \llbracket \tau_1 \times \tau_2 \rrbracket.
2800
2801
                PROOF. Unfolding the expression relation in our hypothesis about e_1, we get that there are (\Sigma, e'_1), j such that
2802
2803
            (\Sigma, e_1) \longrightarrow_T^J (\Sigma, e_1') and (\Sigma', e_1') is irreducible.
2804
            If e'_1 = \text{Err}^{\bullet}, then were done because the entire application steps to an error.
2805
            Otherwise, there is a (k - j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k - j, \Psi) and (k - j, \Psi', \Sigma', e'_1) \in \mathcal{V}^T \llbracket \tau_1 \rrbracket.
            This means e'_1 = \ell_1 for some \ell_1 \in dom(\Sigma').
                                                                                                                                                                    2022-11-18 03:01. Page 54 of 1-100.
```

```
2810
            With this and by the OS, we get (\Sigma, \langle e_1, e_2 \rangle) \longrightarrow_T^j (\Sigma', \langle loc_1, e_2 \rangle).
2811
2812
           We can apply Lemma 4.55 to our hypothesis about e_2 to get (k - j, \Psi', \Sigma', e_2) \in \mathcal{E}^T \llbracket \tau_2 \rrbracket.
2813
           Unfolding the expression relation, we get that there are (\Sigma', e_2'), j' such that (\Sigma', e_2) \longrightarrow_T^{j'} (\Sigma', e_2') and (\Sigma'', e_2') is
2814
2815
2816
            If e_2' = \text{Err}^{\bullet}, then were done because the entire application steps to an error.
2817
            Otherwise, there is a (k-j-j',\Psi'') \supseteq (k-j,\Psi') such that \Sigma'' : (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e_2') \in \mathcal{V}^T[[\tau_2]],
2818
            which means e_2' = \ell_2 for some \ell_2 \in dom(\Sigma'').
2819
2820
2821
           Putting everything together we get (\Sigma, \langle e_1, e_2 \rangle) \longrightarrow_T^{j'} (\Sigma'', \langle \ell_1, \ell_2 \rangle), with \Sigma'' : (k - j - j', \Psi'').
2822
            Note by OS, (\Sigma'', \langle \ell_1, \ell_2 \rangle) \longrightarrow_T (\Sigma''[\ell' \mapsto \langle \ell_1, \ell_2 \rangle]) where \ell' \notin dom(\Sigma'').
2823
2824
2825
           We firstly need \Sigma''[\ell' \mapsto (\langle \ell_1, \ell_2 \rangle, )] : (k - j - j' - 1, \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)]).
2826
            Note the only interesting part of this statement is that \forall k' < k - j - j' - 1. (k', \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)], \Sigma''[\ell' \mapsto \Psi''(\ell_2)]
2827
            (\langle \ell_1, \ell_2 \rangle, \_)], \ell') \in \mathcal{VH}^T \llbracket \Psi''(\ell_1) \times \Psi''(\ell_2) \rrbracket.
2828
           This is immediate from the fact that \Sigma'': (k', \Psi'') from downward closure, and therefore that (k', \Psi'', \Sigma'', \ell_1) \in
2829
2830
            \mathcal{VH}^T \llbracket \Psi''(\ell_1) \rrbracket and (k', \Psi'', \Sigma'', \ell_2) \in \mathcal{VH}^T \llbracket \Psi''(\ell_2) \rrbracket.
2831
           We know that (k-j, \Psi', \Sigma', \ell'_1) \in \mathcal{V}^T[\![\tau_1]\!] and (k-j-j', \Psi'', \Sigma'', \ell_2) \in \mathcal{V}^T[\![\tau_2]\!], and Lemma 4.55 with down-
2833
           ward closure and the store typing judgement above.
2834
2835
            From these facts we get that (k - j - j' - 1, \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)], \Sigma''[\ell' \mapsto (\langle \ell_1, \ell_2 \rangle, \_)], \ell_1) \in \mathcal{V}^T[\![\tau_1]\!] and
2836
            (k-j-j'-1,\Psi''[\ell'\mapsto \Psi''(\ell_1)\times \Psi''(\ell_2)],\Sigma''[\ell'\mapsto \langle \ell_1,\ell_2\rangle],\ell_2)\in \mathcal{V}^T[\![\tau_2]\!].
2837
           This is sufficient to show (k-j-j'-1,\Psi''[\ell'\mapsto \Psi''(\ell_1)\times \Psi''(\ell_2)],\Sigma''[\bar{\ell'}\mapsto (\langle \ell_1,\ell_2\rangle,\_)], \langle \ell_1,\ell_2\rangle)\in \mathcal{V}^T[\![\tau_1\times \tau_2]\!],
2838
2839
           which is what we wanted to prove.
2840
               Lemma 4.63 (Pairs of Related Terms are Related). If (k, \Psi, \Sigma, e_1) \in \mathcal{EH}^T \llbracket \mathit{fst}(\overline{\tau}) \rrbracket \ \mathit{and} \ (k, \Psi, \Sigma, e_2) \in \mathcal{EH}^T \llbracket \mathit{snd}(\overline{\tau}) \rrbracket
2841
            then (k, \Psi, \Sigma, \langle e_1, e_2 \rangle) \in \mathcal{EH}^T \llbracket \overline{\tau} \rrbracket.
2842
2843
                PROOF. Unfolding the erroring expression relation in our hypothesis about e_1, we get that there are (\Sigma, e'_1), j such
2844
            that (\Sigma, e_1) \longrightarrow_T^J (\Sigma, e_1') and (\Sigma', e_1') is irreducible.
2846
            If e'_1 = \operatorname{Err}^{\bullet}, then were done because the entire application steps to an error.
2847
            Otherwise, there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi) and (k-j, \Psi', \Sigma', e'_1) \in \mathcal{VH}^T \llbracket \mathit{fst}(\overline{\tau}) \rrbracket.
2848
            This means e'_1 = \ell_1 for some \ell_1 \in dom(\Sigma').
2849
2850
2851
           With this and by the OS, we get (\Sigma, \langle e_1, e_2 \rangle) \longrightarrow_T^j (\Sigma', \langle loc_1, e_2 \rangle).
2852
2853
           We can apply Lemma 4.54 to our hypothesis about e_2 to get (k - j, \Psi', \Sigma', e_2) \in \mathcal{EH}^T \llbracket snd(\overline{\tau}) \rrbracket.
2854
           Unfolding the erroring expression relation, we get that there are (\Sigma', e_2'), j' such that (\Sigma', e_2) \xrightarrow{-j'} (\Sigma', e_2') and (\Sigma'', e_2')
2855
2856
2857
           If e_2' = \text{Err}^{\bullet}, then were done because the entire application steps to an error.
2858
            Otherwise, there is a (k-j-j',\Psi'') \supseteq (k-j,\Psi') such that \Sigma'': (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e_2') \in \mathbb{R}
2860
            2022-11-18 03:01. Page 55 of 1-100.
```

```
\mathcal{VH}^T[\![snd(\overline{\tau})]\!], which means e_2' = \ell_2 for some \ell_2 \in dom(\Sigma'').
2861
2862
2863
           Putting everything together we get (\Sigma, \langle e_1, e_2 \rangle) \longrightarrow_T^{j'} (\Sigma'', \langle \ell_1, \ell_2 \rangle), with \Sigma'' : (k - j - j', \Psi'').
2864
           Note by OS, (\Sigma'', \langle \ell_1, \ell_2 \rangle) \longrightarrow_T (\Sigma''[\ell' \mapsto (\langle \ell_1, \ell_2 \rangle, \_)]) where \ell' \notin dom(\Sigma'').
2865
2866
2867
           We firstly need \Sigma''[\ell' \mapsto (\langle \ell_1, \ell_2 \rangle, \_)] : (k - j - j' - 1, \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)]).
2868
           Note the only interesting part of this statement is that \forall k' < k - j - j' - 1. (k', \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)], \Sigma''[\ell' \mapsto \Psi''(\ell_2)]
2869
           (\langle \ell_1, \ell_2 \rangle, )], \ell') \in \mathcal{VH}^T \llbracket \Psi''(\ell_1) \times \Psi''(\ell_2) \rrbracket.
2870
           This is immediate from the fact that \Sigma'':(k',\Psi'') from downward closure, and therefore that (k',\Psi'',\Sigma'',\ell_1)\in
2871
           \mathcal{VH}^T \llbracket \Psi''(\ell_1) \rrbracket and (k', \Psi'', \Sigma'', \ell_2) \in \mathcal{VH}^T \llbracket \Psi''(\ell_2) \rrbracket.
2874
           We know that (k-j, \Psi', \Sigma', \ell'_1) \in \mathcal{VH}^T \llbracket fst(\overline{\tau}) \rrbracket and (k-j-j', \Psi'', \Sigma'', \ell_2) \in \mathcal{VH}^T \llbracket snd(\overline{\tau}) \rrbracket, and Lemma 4.51
2875
           with downward closure and the store typing judgement above.
2876
2877
           From these facts we get that (k - j - j' - 1, \Psi''[\ell' \mapsto \Psi''(\ell_1) \times \Psi''(\ell_2)], \Sigma''[\ell' \mapsto (\langle \ell_1, \ell_2 \rangle, \_)], \ell_1) \in \mathcal{VH}^T[\![fst(\overline{t})]\!]
           and (k-j-j'-1,\Psi''[\ell'\mapsto \Psi''(\ell_1)\times \Psi''(\ell_2)],\Sigma''[\ell'\mapsto \langle \ell_1,\ell_2\rangle],\ell_2)\in \mathcal{VH}^T[\![\mathit{snd}(\overline{\tau})]\!].
           This is sufficient to show (k-j-j'-1,\Psi''[\ell'\mapsto\Psi''(\ell_1)\times\Psi''(\ell_2)],\Sigma''[\ell'\mapsto(\langle\ell_1,\ell_2\rangle,\_)],\langle\ell_1,\ell_2\rangle)\in\mathcal{VH}^T[[\bar{\tau}]], which
2880
           is what we wanted to prove.
2881
2882
2883
               Lemma 4.64 (Applications of Semantically Well Typed Terms are Semantically Well Typed). If (k, \Psi, \Sigma, e_f) \in
2884
           \mathcal{E}^T[\![* \to \tau]\!] and (k, \Psi, \Sigma, e) \in \mathcal{E}^T[\![*]\!] then \forall K, (k, \Psi, \Sigma, \mathsf{app}\{K\} \, e_f \, e) \in \mathcal{E}^T[\![\tau \sqcap K]\!]
2885
               PROOF. Unfolding the expression relation in our hypothesis about e_f, we get that there are (\Sigma', e_f'), j such that
2887
2888
           (\Sigma, e_f) \longrightarrow_T^j (\Sigma', e_f') and (\Sigma', e_f') is irreducible.
2889
           If e'_f = \text{Err}^{\bullet}, then we're done because the entire application steps to an error.
2890
           Otherwise, there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e'_f) \in \mathcal{V}^T[[* \to \tau]].
2891
2892
           This means e'_f = \ell_f for some \ell_f \in dom(\Sigma').
2893
2894
           Using this, we know from the OS that (\Sigma, \operatorname{app}\{K\} e_f e) \longrightarrow_T^J (\Sigma', \operatorname{app}\{K\} \ell_f e).
2895
2896
           We can apply Lemma 4.55 with \Sigma': (k-j, \Psi') to our hypothesis about e to get (k-j, \Psi', \Sigma', e) \in \mathcal{E}^T[\![*]\!].
2897
           Unfolding the expression relation, we get that there are (\Sigma'', e'), j' such that (\Sigma', e) \longrightarrow_T^{j'} (\Sigma'', e') where (\Sigma'', e') is
           irreducible.
           If e' = \text{Err}^{\bullet} than we're done, because the whole application errors.
2901
           Otherwise, there exists (k-j-j',\Psi'') \supseteq (k-j,\Psi') such that \Sigma'': (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e') \in \mathcal{V}^T[\![*]\!].
2902
2903
           This means e' = \ell for some \ell \in dom(\Sigma'').
2904
2905
           Putting what we have together, by the OS, (\Sigma, \operatorname{app}\{K\} e_f e) \longrightarrow_T^{j+j'} (\Sigma'', (\operatorname{app}\{K\} \ell_f \ell)).
2906
           We have (k-j, \Psi', \Sigma', \ell_f) \in \mathcal{V}^T \llbracket * \to \tau \rrbracket and (k-j-j', \Psi'') \supseteq (k-j, \Psi') and \Sigma'' \supseteq \Sigma' and \Sigma'' : (k-j-j', \Psi'').
2907
2908
           We can combine these to get (k - j - j', \Psi'', \Sigma'', \mathsf{app}\{K\} \ell_f \ell) \in \mathcal{E}^T \llbracket \tau \sqcap K \rrbracket.
2909
           This is sufficient to complete the proof.
                                                                                                                                                                                                    2910
```

```
COROLLARY 4.65. If (k, \Psi, \Sigma, \ell) \in \mathcal{E}^T[\![*]\!] and \Sigma(\ell) = w and (k, \Psi, \Sigma, e) \in \mathcal{E}^T[\![*]\!] then (k-1, \Psi, \Sigma, \mathsf{app}\{*\} w e) \in \mathcal{E}^T[\![*]\!]
2913
2914
            \mathcal{E}^T \llbracket * \rrbracket.
2915
                Lemma 4.66 (Applications of Related Terms are Related). If (k, \Psi, \Sigma, e_f) \in \mathcal{EH}^T[\![\tau, \overline{\tau}]\!] and (k, \Psi, \Sigma, e) \in \mathcal{E}^T[\![*]\!]
2917
            then \forall K, (k, \Psi, \Sigma, \mathsf{app}\{K\} e_f e) \in \mathcal{EH}^T \llbracket cod(\tau) \sqcap K, cod(\overline{\tau}) \rrbracket.
2918
2919
                Proof. Unfolding the erroring expression relation in our hypothesis about e_f, we get that there are (\Sigma', e_f'), j such
2920
           that (\Sigma, e_f) \longrightarrow_T^j (\Sigma', e_f') and (\Sigma', e_f') is irreducible.
2921
           If e'_f = \operatorname{Err}^{\bullet}, then we're done because the entire application steps to an error.
2922
            Otherwise, there is a (k - j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k - j, \Psi') and (k - j, \Psi', \Sigma', e'_f) \in \mathcal{VH}^T[[\tau, \overline{\tau}]].
2924
            This means e'_f = \ell_f for some \ell_f \in dom(\Sigma').
2925
2926
           Using this, we know from the OS that (\Sigma, \operatorname{app}\{K\} e_f e) \longrightarrow_T^j (\Sigma', \operatorname{app}\{K\} \ell_f e).
2927
2928
2929
           We can apply Lemma 4.55 with \Sigma':(k-j,\Psi') to our hypothesis about e to get (k-j,\Psi',\Sigma',e)\in\mathcal{E}^T[\![*]\!].
2930
            Unfolding the expression relation, we get that there are (\Sigma'', e'), j' such that (\Sigma', e) \longrightarrow_T^{j'} (\Sigma'', e') where (\Sigma'', e') is
2931
2932
2933
           If e' = \text{Err}^{\bullet} than we're done, because the whole application errors.
2934
            Otherwise, there exists (k-j-j',\Psi'') \supseteq (k-j,\Psi') such that \Sigma'': (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e') \in \mathcal{V}^T[\![*]\!].
           This means e' = \ell for some \ell \in dom(\Sigma'').
2937
           Putting what we have together, by the OS, (\Sigma, \operatorname{app}\{K\} e_f e) \longrightarrow_T^{j+j'} (\Sigma'', (\operatorname{app}\{K\} \ell_f \ell)).
2938
2939
            We have (k-j, \Psi', \Sigma', \ell_f) \in \mathcal{V}^T[\![* \to \tau]\!] and (k-j-j', \Psi'') \supseteq (k-j, \Psi') and \Sigma'' \supseteq \Sigma' and \Sigma'' : (k-j-j', \Psi'').
2940
            We can combine these to get (k-j-j',\Psi'',\Sigma'',\operatorname{app}\{K\}\ \ell_f\ \ell)\in\mathcal{EH}^T[\![\operatorname{cod}(\tau)\sqcap K,\operatorname{cod}(\overline{\tau})]\!]
2941
           This is sufficient to complete the proof.
2942
                COROLLARY 4.67. If (k, \Psi, \Sigma, e_f) \in \mathcal{EH}^T[\![*, \overline{\tau}]\!] and (k-1, \Psi, \Sigma, e) \in \mathcal{E}^T[\![*]\!] then (k-1, \Psi, \Sigma, \mathsf{app}\{\tau_0\} e_f e) \in \mathcal{EH}^T[\![*]\!]
2944
           \mathcal{EH}^T \llbracket *, cod(\overline{\tau}) \rrbracket.
2945
2946
                                                                                                (1) If (k + 1, \Psi, \Sigma, \text{assert } * e) \in \mathcal{E}^T \llbracket \tau \rrbracket then (k, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau \rrbracket.
                LEMMA 4.68 (DYNAMIC CHECKS ARE NOOPS).
2947
                  (2) If (k + 1, \Psi, \Sigma, \text{assert } * e) \in \mathcal{EH}^T \llbracket \overline{\tau} \rrbracket then (k, \Psi, \Sigma, e) \in \mathcal{EH}^T \llbracket \overline{\tau} \rrbracket.
                                   (1) assume there is \Sigma', e', j such that (\Sigma, e) \longrightarrow_T^j (\Sigma', e') where (\Sigma', e') is irreducible.
                        By the OS, we get that (\Sigma, \text{assert } * e) \longrightarrow_T^j (\Sigma', \text{assert } * e').
2951
2952
                        Then by OS, we have (\Sigma', \text{assert } * e') \longrightarrow_T^j (\Sigma', e').
2953
                        Therefore, we can apply our hypothesis to complete the proof.
2954
                  (2) Same as previous case, just using the history relation.
2955
                                                                                                                                                                                                           2957
                Lemma 4.69 (Monitor Compatibility). If \Sigma:(k, \Psi), then
2958
2959
                  (1) If (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau \rrbracket and \Sigma(\ell') = (\ell, \mathsf{some}(K', K)), then (k, \Psi, \Sigma, \ell') \in \mathcal{V}^T \llbracket K' \sqcap K \sqcap \tau \rrbracket
2960
                  (2) If (k, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau \sqcap K \sqcap K' \rrbracket then (k, \Psi, \Sigma, \text{mon } \{K' \Leftarrow K\} e) \in \mathcal{E}^T \llbracket \tau \sqcap K \sqcap K' \rrbracket.
```

(3) If $(k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T \llbracket \Psi(\ell) \rrbracket$ and $\Sigma' = \Sigma[\ell' \mapsto (\ell, \mathsf{some}(K', K))]$ and $\Psi' = [\ell' \mapsto K', K, \Psi(\ell)] \Psi$ and $\ell' \notin \mathcal{U}$

2022-11-18 03:01. Page 57 of 1-100.

2963 2964 $dom(\Sigma)$ and $\vdash \Sigma'$ then $(k, \Psi', \Sigma', \ell') \in \mathcal{VH}^T \llbracket K', K, \Psi(\ell) \rrbracket$

```
(4) If (k, \Psi, \Sigma, e) \in \mathcal{EH}^T[\![\overline{\tau}]\!] then (k, \Psi, \Sigma, \text{mon } \{* \Leftarrow *\} e) \in \mathcal{EH}^T[\![*, *, \overline{\tau}]\!]
2965
2966
                PROOF. Proceed by simultaneous induction on k and \tau.
2967
                    • k = 0: 2) and 4) follow from 1) and 3) respectively.
2969
                         The proofs follow similarly to the other case, but any function or dynamic cases are vacuously true.
2970
2971
                     • k > 0:
                            1) Unfolding the relation in the statement we want to prove, note from our hypothesis about \Sigma, we get that
                                 - Σ.
                                 Proceed by case analysis on \tau \sqcap K \sqcap K':
                                       i) \tau = \tau \sqcap K \sqcap K': Immediate.
2978
                                      ii) \tau \sqcap K \sqcap K' = \bot: then either K or K' is \bot, which is a contradiction since they both tagmatch
2979
                                           pointsto(\Sigma, \ell).
2980
2982
                                     iii) \tau \sqcap K \sqcap K' \leq :\tau: then \tau = \text{Int and } K \text{ or } K' = \text{Nat.}
2983
                                           Immediate because by \vdash \Sigma, Nat \sim pointsto(\Sigma, \ell).
2984
                                     iv) \tau \sqcap K \sqcap K' \neq \tau: then it must be the case that \tau = * and K or K' = * \rightarrow *.
                                           Note K or K' cannot be * \times *, by \vdash \Sigma.
                                           Unfolding the relation in our hypothesis, we have that (k-1, \Psi, \Sigma, \ell) \in \mathcal{V}^T[\![* \to *]\!].
                                           We want to show that (k, \Psi, \Sigma, \ell') \in \mathcal{V}^T [\![ * \to * ]\!].
                                           Unfolding the relation, let (j, \Psi') \supseteq (k, \Psi) and \Sigma' \supseteq \Sigma such that \Sigma' : (j, \Psi').
2991
                                           Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^T [\![ * ]\!].
2992
                                           Let K.
2993
                                           We want to show (j, \Psi', \Sigma', \mathsf{app}\{K\} \ell' \ell_v) \in \mathcal{E}^T \llbracket K \rrbracket.
2994
                                           By the OS, (\Sigma', \operatorname{app}\{K\} \ell' \ell_v) \longrightarrow_T^2 (\Sigma', \operatorname{assert} K (\operatorname{mon} \{* \Leftarrow *\} (\ell (\operatorname{mon} \{* \Leftarrow *\} \ell_v)))).
                                           By IH 2), we have (j, \Psi', \Sigma', \text{mon } \{* \Leftarrow *\} \ell_v) \in \mathcal{E}^T \llbracket * \rrbracket.
2997
                                           By Lemma 4.64, we have that (j, \Psi', \Sigma', \operatorname{app}\{K\} \ell \pmod{* \Leftarrow *} \ell_v) \in \mathcal{E}^T \llbracket K \rrbracket.
                                           Then by IH 2), we have (j, \Psi', \Sigma', \text{mon } \{* \Leftarrow *\} (\text{app}\{K\} \ell (\text{mon } \{* \Leftarrow *\} \ell_v))) \in \mathcal{E}^T \llbracket K \rrbracket.
                                           Note that (j, \Psi', \Sigma', \text{mon } \{* \Leftarrow *\} (\text{app}\{K\} \ell (\text{mon } \{* \Leftarrow *\} \ell_{v}))) \in \mathcal{E}^{T}[\![K]\!] \text{ iff } (j, \Psi', \Sigma', \text{assert } K (\text{mon } \{* \Leftarrow *\} \ell (\text{mon } \{* \Leftarrow *\} \ell_{v}))) \in \mathcal{E}^{T}[\![K]\!] \text{ in } (j, \Psi', \Sigma', \text{assert } K (\text{mon } \{* \Leftarrow *\} \ell_{v})))
                                           Therefore, this is sufficient to complete the case.
3003
                           2) Unfolding the expression relation in our hypothesis, we have that there are (e', \Sigma'), j such that (e, \Sigma) \longrightarrow_T^J
3004
                                 (e', \Sigma') with (e', \Sigma') irreducible.
3005
3006
                                 If e' = \text{Err}^{\bullet} then we're done, because the monitor will step to an error as well.
3007
                                 Otherwise, there is (k - j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k - j, \Psi') and (k - j, \Psi', \Sigma', e') \in \mathcal{V}^T \llbracket \tau \sqcap K \sqcap K' \rrbracket.
                                 This means \exists \ell \in dom(\Sigma') such that e' = \ell.
3009
3010
3011
                                 We want to show (k - j, \Psi', \Sigma', \text{mon } \{K \Leftarrow \ell\} \in \mathcal{E}^T \llbracket \tau \sqcap K \sqcap K' \rrbracket.
3012
                                 We destruct on whether \Sigma'(\ell) is a pair.
                                 If \Sigma'(\ell) = (\langle \ell_1, \ell_2 \rangle, \_), then by the OS, (\Sigma', \text{mon } \{K \Leftarrow \ell\} \longrightarrow_T (\Sigma', \langle \text{mon } \{* \Leftarrow *\} \ell_1, \text{mon } \{* \Leftarrow *\} \ell_2 \rangle).
                                 Then by Lemma 4.62, it suffices to show (k-j, \Psi', \Sigma', \text{mon } \{* \Leftarrow \ell_1\} \in \mathcal{E}^T \llbracket fst(\tau) \rrbracket \text{ and } (k-j, \Psi', \Sigma', \text{mon } \{* \Leftarrow \ell_2\} \in \mathcal{E}^T \llbracket snd(\tau) \rrbracket
3016
                                                                                                                                                                    2022-11-18 03:01. Page 58 of 1-100.
```

```
These both follow from IH 2) (smaller by index).
3017
3018
                                Otherwise, by the OS, (\Sigma', \text{mon } \{K \Leftarrow \ell\} \longrightarrow_T (\Sigma'[\ell' \mapsto (\ell, \text{some}(K', K))], \ell').
3019
                                Then by IH 3), we get \Sigma'[\ell' \mapsto (\ell, \mathsf{some}(K', K))] : (k - j - 1, \Psi'[\ell' \mapsto K', K, \Psi'(\ell)]).
                                And by IH 1), we get (k-j-1, \Psi'[\ell' \mapsto K', K, \Psi'(\ell)], \Sigma'[\ell' \mapsto (\ell, \mathsf{some}(K', K))], \ell') \in \mathcal{V}^T[\![\tau \sqcap K \sqcap K']\!].
3021
                                These two facts are sufficient to complete the case.
3022
3023
                           3) We proceed by case analysis on K' (note by the fact that \vdash \Sigma', K \sim K'):
3024
                                    (a) K' = \text{Nat}: Since we already know (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^V[N][\Psi(\ell)], it suffices to show (k, \Psi, \Sigma, \ell') \in \mathcal{VH}^V[N][\Psi(\ell)]
                                          \mathcal{V}^N \llbracket K' \rrbracket and (k, \Psi, \Sigma, \ell') \in \mathcal{V}^N \llbracket K \rrbracket.
                                          This is immediate from \vdash \Sigma', which implies K' \sim \text{pointsto}(\Sigma', \ell') and K \sim \text{pointsto}(\Sigma', \ell').
3029
                                    (b) K' = Int: same as the Nat case.
3030
                                    (c) K' = Bool: same as the Nat case.
3031
                                    (d) K' = * \times *: this case is a contradiction by the fact that \vdash \Sigma.
3032
3033
                                    (e) K' = * \rightarrow *: Since pointsto(\Sigma, \ell) \sim K' and pointsto(\Sigma, \ell) \sim K, K = * or * \rightarrow *.
3034
                                          Also, since \vdash \Sigma', we get that \Psi(\ell) = [*, \overline{\tau'}] or [* \to *, \overline{\tau'}].
3035
                                          From the fact that (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T \llbracket \Psi(\ell) \rrbracket, we get that (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T \llbracket *, \overline{\tau'} \rrbracket or (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T \llbracket *, \overline{\tau'} \rrbracket or (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T \llbracket *, \overline{\tau'} \rrbracket
3036
                                          \mathcal{VH}^T[\![* \to *, \overline{\tau'}]\!].
3037
                                          In the case of *, we can unfold and get (k-1, \Psi, \Sigma, \ell) \in \mathcal{VH}^T[[* \to *, \overline{\tau'}]].
                                          Otherwise we can get the same using Lemma 4.51.
                                          Similarly, we want to show that (k, \Psi', \Sigma', \ell') \in \mathcal{VH}^T [\![K', K, \Psi(\ell)]\!].
3041
                                          By Lemma 4.51, in the K' = * case, it suffices to show (k, \Psi', \Sigma', \ell') \in \mathcal{VH}^T \llbracket * \to *, K, \Psi(\ell) \rrbracket.
3042
3043
                                          So let (j, \Psi'') \supseteq (k, \Psi'), and let \Sigma'' \supseteq \Sigma' such that \Sigma'' : (j, \Psi'').
3044
                                          Let \ell_v \in dom(\Sigma'') such that (j, \Psi'', \Sigma'', \ell_v) \in \mathcal{V}^T \llbracket * \rrbracket.
3045
                                          Let K''.
                                          We want to show (j, \Psi'', \Sigma'', \operatorname{app}\{K''\} \ell' \ell_v) \in \mathcal{EH}^T \llbracket K'', *, \operatorname{cod}(\Psi(\ell)) \rrbracket
3048
                                          By the OS, (\Sigma'', \operatorname{app}\{K''\} \ell' \ell_v) \longrightarrow_T (\Sigma'', \operatorname{assert} K'' (\ell' \ell_v)).
3049
                                          By Lemma 4.60, it suffices to show (j-1, \Psi'', \Sigma'', \ell' \ell_v) \in \mathcal{EH}^T [\![ *, *, cod(\Psi(\ell)) ]\!].
3050
                                          By the OS, (\Sigma'', \ell' \ell_v) \longrightarrow_T (\Sigma'', \text{mon } \{* \Leftarrow *\} (\ell \text{ (mon } \{* \Leftarrow *\} \ell_v))).
3051
                                          By IH 2) (smaller by index), it suffices to show (j-2, \Psi'', \Sigma'', \ell \pmod{* \Leftarrow *} \ell_v)) \in \mathcal{EH}^T [\![*, *, cod(\Psi(\ell))]\!].
                                          By Lemma 4.68, it suffices to show (j-1, \Psi'', \Sigma'', assert *\ell (mon \{* \Leftarrow *\} \ell_v)) \in \mathcal{EH}^T [\![*, *, cod(\Psi(\ell))]\!].
                                          Then by the OS, it suffices to show (j, \Psi'', \Sigma'', \mathsf{app}\{*\} \ell (\mathsf{mon} \{* \Leftarrow *\} \ell_v)) \in \mathcal{EH}^T [\![*, *, cod(\Psi(\ell))]\!].
3055
                                          By IH 2), (j, \Psi'', \Sigma'', \text{mon } \{* \Leftarrow *\} \ell_v) \in \mathcal{V}^T \llbracket * \rrbracket.
3056
                                          Unfolding, we get that there exists some j', e'', \Sigma''' such that (\Sigma'', \text{mon } \{* \Leftarrow *\}) \longrightarrow_T^{j'} (\Sigma''', e').
3057
3058
                                          If e' = \text{Err}^{\bullet}, then we're done because the entire application errors.
3059
                                          Otherwise, we get that there exists a (j-j',\Psi''') \supseteq (j,\Psi'') such that \Sigma''' : (j-j',\Psi''') and
                                          (j-j',\Psi^{\prime\prime\prime},\Sigma^{\prime\prime\prime},e^{\prime\prime})\in\mathcal{V}^T[\![*]\!].
3061
                                          Note by the operational semantics, j' \ge 1.
3062
3063
                                          By Lemma 4.51, we get (j - j', \Psi''', \Sigma''', \ell) \in \mathcal{VH}^T[[* \to *, \overline{\tau'}]].
3064
                                          Finally we can apply this hypothesis to the fact about e'' to get that (j - j', \Psi''', \Sigma''', \text{app}\{*\} \ell e'') \in
                                          \mathcal{EH}^T \llbracket *, *, cod(\Psi(\ell)) \rrbracket, which is sufficient to complete the case.
                                    (f) K' = *: unfolding the relation in what we want to show, the proof follows by IH 3) (smaller by index).
```

```
4) Unfolding the expression relation in our hypothesis, we have that there are (e', \Sigma'), j such that (e, \Sigma) \longrightarrow_T^j
3070
                              (e', \Sigma') with (e', \Sigma') irreducible.
3071
                              If e' = \text{Err}^{\bullet} then we're done, because the monitor will step to an error as well.
                              Otherwise, there is (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{VH}^T[\![\overline{\tau}]\!].
3073
                              This means \exists \ell \in dom(\Sigma') such that e' = \ell.
3074
                              We want to show (k - j, \Psi', \Sigma', \text{mon } \{* \Leftarrow *\} \ell) \in \mathcal{EH}^T \llbracket *, *, \Psi'(\ell) \rrbracket.
                              For ii), by OS, if \Sigma'(\ell) = (\langle \ell_1, \ell_2 \rangle, \underline{\ }), then (\Sigma', \text{mon } \{* \Leftarrow \ell\} \longrightarrow_T (\Sigma', \langle \text{mon } \{* \Leftarrow *\} \ell_1, \text{mon } \{* \Leftarrow *\} \ell_2 \rangle).
                              Then by Lemma 4.63, it suffices to show (k-j-j'-1, \Psi, \Sigma, \text{mon } \{* \in \ell_1\} \in \mathcal{VH}^T \llbracket *, *, \tau \rrbracket \text{ and } (k-j-j') = 0
                              j' - 1, \Psi, \Sigma, \text{mon } \{* \Leftarrow \ell_2\} \in \mathcal{VH}^T \llbracket *, *, \tau \rrbracket.
3081
                              Both of these follow from (4) (smaller by index).
3082
3083
                              Otherwise, by the OS, (\Sigma', \text{mon } \{* \Leftarrow *\}) \longrightarrow_T (\Sigma'[\ell' \mapsto (\ell, \text{some}(*, *))], \ell').
3084
                              We can finish the proof by applying IH 3) (smaller by index).
                                                                                                                                                                                              3087
3088
                                                                                                                                                (1) If (k, \Psi, \Sigma, e) \in \mathcal{E}^T \llbracket \tau \rrbracket then
               LEMMA 4.70 (EXPRESSION RELATION IMPLIES ERRORING EXPRESSION RELATION).
3089
                       (k, \Psi, \Sigma, e) \in \mathcal{EH}^T \llbracket \tau \rrbracket.
                 (2) If (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau \rrbracket then (k, \Psi, \Sigma, \ell) \in \mathcal{VH}^T \llbracket \tau \rrbracket.
              PROOF. Proceed by induction on k and \tau:
                   • k = 0: 1) is immediate from 2).
3095
3096
3097
                          - \tau = Int: immediate.
3098
                          -\tau = \tau_1 \times \tau_2: then \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \_).
                              The case follows from the IH on \ell_1 and \ell_2.
3100
3101
                          - \tau = \tau_1 \rightarrow \tau_2: vacuously true.
3102
                          - \tau = *: vacuously true.
3103
                   • k > 0: 1) is immediate from 2).
3104
3106
                          - \tau = Int: immediate.
3107
                          - \tau = \tau_1 \times \tau_2: then \Sigma(\ell) = (\langle \ell_1, \ell_2 \rangle, \_).
3108
                              The case follows from the IH on \ell_1 and \ell_2.
3109
                          - \tau = \tau_1 \rightarrow \tau_2: Follows from 1) from the IH (smaller by index).
3110
3111
                          - τ = *: Follows from 2) from the IH (smaller by index), using * × *, * → *, or Int.
3112
                                                                                                                                                                                              3113
3114
3115
           4.4.3 Compatability Lemmas
3116
3117
              Lemma 4.71 (T-Var compatibility). \frac{(x_0:K_0) \in \Gamma}{\Gamma \vdash x_0:K_0}
```

```
PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3121
3122
            We want to show (k, \Psi, \Sigma, \gamma(x)) \in \mathcal{E}^T \llbracket \tau \rrbracket.
3123
            Since x : \tau \in \Gamma, we get that \gamma(x) = \ell.
3124
            Since (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket, we get (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \tau \rrbracket.
3125
            Then we get that (k, \Psi, \Sigma, \ell) \in \mathcal{E}^T \llbracket \tau \rrbracket immediately since \ell is already a value and we have as a premise that \Sigma : (k, \Psi). \square
3126
3127
3128
                Lemma 4.72 (T-Nat compatibility).
                                                                                  \llbracket \Gamma \vdash n_0 : \mathsf{Nat} \rrbracket
3129
3130
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3131
3132
            We want to show (k, \Psi, \Sigma, \gamma(n)) \in \mathcal{E}^T \llbracket \operatorname{Nat} \rrbracket.
3133
            Note \gamma(n) = n.
3134
            By the OS, we have (\Sigma, n) \longrightarrow_T (\Sigma[\ell \mapsto (n, none)], \ell).
3135
            We get (k, \Psi, \Sigma, \ell) \in \mathcal{V}^T \llbracket \operatorname{Nat} \rrbracket immediately because n \in \mathbb{T}.
3136
3137
            Since \mathcal{V}^T[\![\text{Nat}]\!] does not rely on \Psi or \Sigma, we have that (k, \Psi[\ell \mapsto [\text{Nat}]], \Sigma[\ell \mapsto (n, \text{none})], \ell) \in \mathcal{V}^T[\![\text{Nat}]\!].
3138
            Since \ell \mapsto \text{Nat}, we have that (k, \Psi[\ell \mapsto [\text{Nat}]], \Sigma[\ell \mapsto (n, \text{none})], \ell) \in \mathcal{V}^T[[\text{Nat}]].
3139
            Similarly we have (k, \Psi[\ell \mapsto [\mathsf{Nat}]], \Sigma[\ell \mapsto (n, \mathsf{none})], \ell) \in \mathcal{VH}^V[T] Nat.
3140
            Therefore, given we know \Sigma : (k, \Psi), we know \Sigma[\ell \mapsto (n, \text{none})] : (k, \Psi[\ell \mapsto [\text{Nat}]]).
                                                                                                                                                                                                                    3141
3142
                Lemma 4.73 (T-Int compatibility). \frac{}{ \left[\!\!\left[\Gamma \vdash i_0 : \mathsf{Int}\right]\!\!\right]}
3143
3144
3145
                PROOF. Not meaningfully different from T-Nat
                                                                                                                                                                                                                    3146
3147
                Lemma 4.74 (T-True compatibility). \frac{}{ \llbracket \Gamma \vdash \mathsf{True} : \mathsf{Bool} \rrbracket }
3148
3149
3150
                PROOF. Not meaningfully different from T-Nat
                                                                                                                                                                                                                    3151
3152
                3153
3154
                PROOF. Not meaningfully different from T-Nat
3155
                                                                                                                                                                                                                    3156
                \text{Lemma 4.76 (T-Lam compatibility). } \frac{ \left[\!\!\left[\Gamma_0,\; (x_0\!:\!K_0) \vdash e_0:\tau_1\right]\!\!\right]}{ \left[\!\!\left[\Gamma_0 \vdash \lambda(x_0\!:\!K_0).\,e_0:*\!\to\!\tau_1\right]\!\!\right]}
3157
3158
3159
3160
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3161
            We want to show (k, \Psi, \Sigma, \gamma(\lambda x_1 : K. e_1)) \in \mathcal{E}^T \llbracket * \to \tau_1 \rrbracket.
3162
3163
            Note that \gamma(\lambda x_1 : K. e_1) = \lambda x_1 : K. \gamma(e_1).
3164
            Since \lambda x_1 : K, \gamma(e_1) is a value, by the OS we have (\Sigma, \lambda x_1 : K, \gamma(e_1)) \longrightarrow_T (\Sigma[\ell \mapsto (\lambda x_1 : K, \gamma(e_1), none)]), where
3165
            \ell \notin dom(\Sigma).
3166
            We choose our later \Psi' to be \Psi[\ell \mapsto * \to *].
3167
            We now have two obligations:
3168
3169
                  (1) (k-1, \Psi[\ell \mapsto * \to *], \Sigma[\ell \mapsto (\lambda x_1 : K, \gamma(e_1), \mathsf{none})], \ell) \in \mathcal{V}^T[[* \to \tau_1]]
3170
                  (2) \Sigma[\ell \mapsto (\lambda x_1 : K. \gamma(e_1), none)] : (k-1, \Psi[\ell \mapsto * \rightarrow *])
3171
3172
            2022-11-18 03:01. Page 61 of 1-100.
```

```
For 1), we want to show (k-1, \Psi[\ell \mapsto * \to *], \Sigma[\ell \mapsto (\lambda x_1 : K, \gamma(e_1), \mathsf{none})], \lambda x_1 : K, \gamma(e_1)) \in \mathcal{V}^T[[* \to \tau_1]].
3173
3174
           Unfolding the value relation:
3175
           Let (j, \Psi') \supseteq (k-1, \Psi[\ell \mapsto * \to *]) and \Sigma' \supseteq \Sigma[\ell \mapsto (\lambda x_1 : K, \gamma(e_1), none)] such that \Sigma' : (j, \Psi').
3176
           Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^T \llbracket * \rrbracket.
3177
3178
           Let K.
3179
           We want to show (j, \Psi', \Sigma', \operatorname{app}\{K\} \ell \ell_v) \in \mathcal{E}^T \llbracket \tau_1 \sqcap K \rrbracket.
3180
           By the OS, if \neg K \sim \Sigma(\ell_v) then the application steps to an error and we're done.
3181
           Otherwise, (\Sigma', \operatorname{app}\{K\} \ell \ell_v) \longrightarrow_T (\Sigma', \operatorname{assert} K \gamma(e_1)[\ell_v/x]).
3182
           By the definition of substitution, \gamma(e_1)[\ell_n/x] = \gamma[x \mapsto \ell_n](e_1).
3183
3184
           Note that (j-2, \Psi', \Sigma', \gamma[x \mapsto \ell_v](e_1)) \in \mathcal{G}^T \llbracket \Gamma, x : K \rrbracket:
3186
                   i) (j-2, \Psi', \Sigma', \ell_v) \in \mathcal{V}^T [\![K]\!] by Lemma 4.55 and Lemma 4.57.
3187
                  ii) \forall y \in dom(y), (j-2, \Psi', \Sigma', \gamma(y)) \in \mathcal{V}^T \llbracket \Gamma(y) \rrbracket by the premise about y and Lemma 4.55.
3188
3189
3190
           Therefore, we can apply the hypothesis to \gamma[x \mapsto \ell_v], \Psi', \Sigma', and e_1 at j-2 to get (j-2, \Psi', \Sigma', \gamma[x \mapsto \ell_v](e_1)) \in \mathcal{E}^T[[\tau_1]].
3191
           Finally, we can apply Lemma 4.58 to get (j-1, \Psi', \Sigma', \text{assert } K \gamma[x \mapsto \ell_n](e_1)) \in \mathcal{E}^T \llbracket \tau_1 \sqcap K \rrbracket which is what we wanted
3192
           to show.
3193
3194
3195
           For 2), first note the domains are equal, since dom(\Sigma) = dom(\Psi).
3196
           Then note \vdash \Sigma[\ell \mapsto (\lambda x_1 : K.\gamma(e_1), \text{none}] \text{ since } \vdash \Sigma.
3197
           Then let j < k - 1 and let \ell' \in dom(\Sigma[\ell \mapsto (\lambda x_1 : K.\gamma(e_1), none)]).
3198
           If \ell' \neq \ell, then we get the remaining conditions from \Sigma : (k, \Psi) and Lemma 4.51.
3199
3200
           If \ell' = \ell, then note the structural obligation on \Psi[\ell \mapsto [* \to *]] is immediate.
3201
           We want to show (i, \Psi[\ell \mapsto * \to *], \Sigma[\ell \mapsto (\lambda x_1 : K, \gamma(e_1), none)], \ell) \in \mathcal{VH}^T[[* \to *]].
3202
           Let (j, \Psi') \supseteq (k-1, \Psi[\ell \mapsto * \to *]) and \Sigma' \supseteq \Sigma[\ell \mapsto (\lambda x_1 : K, \gamma(e_1), \_)] such that \Sigma' : (j, \Psi').
3203
3204
           Let \ell_v \in dom(\Sigma') such that (j, \Psi', \Sigma', \ell_v) \in \mathcal{V}^T \llbracket * \rrbracket.
3205
3206
           We get immediately that pointsto(\Sigma', \ell_n) ~ *, so we want to show (i, \Psi', \Sigma', app\{K\} \ell \ell_n) \in \mathcal{EH}^V[\![* \sqcap K]\!].
3207
           By the OS, if \neg K \sim \Sigma(\ell_p), then the application errors and we're done. Otherwise, (\Sigma', \operatorname{app}\{K\} \ell \ell_p) \longrightarrow_T (\Sigma', \operatorname{assert} K \gamma(e_1) [\ell_p/X]).
3208
           By the definition of substitution, \gamma(e_1)[\ell_v/x] = \gamma[x \mapsto \ell_v](e_1).
3210
           Note that (j-2, \Psi', \Sigma', \gamma[x \mapsto \ell_v](e_1)) \in \mathcal{G}^T[\Gamma, x : *]:
3211
3212
                   i) (j-2, \Psi', \Sigma', \ell_v) \in \mathcal{V}^T \llbracket K \rrbracket by Lemma 4.55 and Lemma 4.57.
3213
                  ii) \forall y \in dom(\gamma), (j-2, \Psi', \Sigma', \gamma(y)) \in \mathcal{V}^T \llbracket \Gamma(y) \rrbracket by the premise about \gamma and Lemma 4.55.
3214
3215
3216
           Therefore, we can apply the hypothesis to \gamma[x \mapsto \ell_v], \Psi', \Sigma', and e_1 at j-2 to get (j-2, \Psi', \Sigma', \gamma[x \mapsto \ell_v](e_1)) \in \mathcal{E}^T[\tau_1].
3217
           Then we can apply Lemma 4.70 to get (j-2, \Psi', \Sigma', \gamma[x \mapsto \ell_v](e_1)) \in \mathcal{EH}^V[\tau_1].
3218
           We can then apply Lemma 4.61 to get (j-2, \Psi', \Sigma', \gamma[x \mapsto \ell_v](e_1)) \in \mathcal{EH}^V[\![*]\!].
3219
           Finally, we can apply Lemma 4.58 to get (j-1,\Psi',\Sigma', assert K\gamma[x\mapsto \ell_v](e_1))\in \mathcal{EH}^V[\![*\sqcap K]\!] which is what we
3220
3221
           wanted to show.
3222
                                                                                                                                                          2022-11-18 03:01. Page 62 of 1-100.
```

2022-11-18 03:01. Page 63 of 1-100.

```
Lemma 4.77 (T-Pair compatibility). \frac{\llbracket \Gamma \vdash e_1 : \tau_1 \rrbracket}{\llbracket \Gamma \vdash \langle e_0, e_1 \rangle : \tau_0 \times \tau_1 \rrbracket}
3226
3227
3228
3229
                  PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3230
             We want to show (k, \Psi, \Sigma, \gamma(\langle e_1, e_2 \rangle)) \in \mathcal{E}^T \llbracket \tau_1 \times \tau_2 \rrbracket.
3231
3232
             Note \gamma(\langle e_1, e_2 \rangle) = \langle \gamma(e_1), \gamma(e_2) \rangle.
3233
             We can apply the first hypothesis to get (k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^T \llbracket \tau_1 \rrbracket.
3234
             We can apply the second hypothesis to get (k, \Psi, \Sigma, \gamma(e_2)) \in \mathcal{E}^T \llbracket \tau_2 \rrbracket.
3235
3236
             Then by Lemma 4.63, (k, \Psi, \Sigma, \langle \gamma(e_1), \gamma(e_2) \rangle) \in \mathcal{E}^T \llbracket \tau_1 \times \tau_2 \rrbracket, which is what we wanted to show.
                                                                                                                                                                                                                                       3237
                  Lemma 4.78 (T-Cast compatibility). \frac{\llbracket \Gamma \vdash e_0 : \tau_0 \rrbracket}{\llbracket \Gamma \vdash \mathsf{cast} \{K_1 \Leftarrow K_0\} \ e_0 : K_1 \sqcap K_0 \sqcap \tau_0 \rrbracket}
3238
3239
3240
3241
                  PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3242
3243
             We want to show (k, \Psi, \Sigma, \gamma(\text{cast } \{K_1 \Leftarrow K_0\} e_0)) \in \mathcal{E}^T \llbracket K_1 \sqcap K_0 \sqcap \tau_0 \rrbracket.
3244
             Note \gamma(\text{cast } \{K_1 \Leftarrow K_0\} e_0) = \text{cast } \{K_1 \Leftarrow K_0\} \gamma(e_0).
3245
             We can apply the first hypothesis to get (k, \Psi, \Sigma, \gamma(e_0)) \in \mathcal{E}^T \llbracket \tau_0 \rrbracket.
3246
             Unfolding the expression relation, there are j, \Sigma', e' such that (\Sigma, \gamma(e_0)) \longrightarrow_T^j (\Sigma', e') where (\Sigma', e') is irreducible.
3247
             If e' = \text{Err}^{\bullet} then we're done, because the entire boundary expression errors.
3249
             Otherwise, we know there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e') \in \mathcal{V}^T \llbracket \tau_0 \rrbracket.
3250
             This means \exists \ell \in dom(\Sigma') such that e' = \ell.
3251
             By the OS, (\Sigma, \text{cast } \{K_1 \Leftarrow K_0\} \gamma(e_0)) \longrightarrow_T^j (\Sigma', \text{cast } \{K_1 \Leftarrow K_0\} \ell) \longrightarrow_T (\Sigma', \text{mon } \{K_1 \Leftarrow K_0\} \ell).
3252
3253
             By Lemma 4.55, (k - j - 1, \Psi', \Sigma', \ell) \in \mathcal{V}^T [\![\tau_0]\!].
3254
             By Lemma 4.69, (k-j-1, \Psi', \Sigma', \text{mon } \{K_1 \Leftarrow K_0\} \ell) \in \mathcal{E}^T \llbracket K_1 \sqcap K_0 \sqcap \tau_0 \rrbracket, which is what we wanted to show.
3255
3256
3257
                 Lemma 4.79 (T-App compatibility). \frac{ [\![ \Gamma \vdash e_1 : \tau_0' ]\!]}{ [\![ \Gamma \vdash \mathsf{app} \{K_1\} e_0 \ e_1 : K_1 \sqcap \tau_1]\!]}
3258
3259
3260
3261
                  PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3262
             We want to show (k, \Psi, \Sigma, \gamma(\mathsf{app}\{K_1\} e_1 e_2)) \in \mathcal{E}^T \llbracket K_1 \sqcap \tau_1 \rrbracket.
3263
             Note \gamma(\text{app}\{K_1\} e_1 e_2) = \text{app}\{K_1\} \gamma(e_1) \gamma(e_2).
3264
3265
             By the first hypothesis we have (k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^T \llbracket * \to \tau_1 \rrbracket
3266
             By the second hypothesis we have (k, \Psi, \Sigma, \gamma(e_2)) \in \mathcal{E}^T \llbracket \tau_0' \rrbracket.
3267
             By Lemma 4.61, we have (k, \Psi, \Sigma, \gamma(e_2)) \in \mathcal{E}^T \llbracket * \rrbracket.
3268
             Then we can apply Lemma 4.64 to get (k, \Psi, \Sigma, \mathsf{app}\{K_1\} \gamma(e_1) \gamma(e_2)) \in \mathcal{E}^T \llbracket \tau_1 \sqcap K_1 \rrbracket which is what we wanted to
3269
3270
             show.
3271
3272
                 Lemma 4.80 (T-AppBot compatibility). \frac{\llbracket \Gamma \vdash e_1 : \tau_0' \rrbracket}{\llbracket \Gamma \vdash \mathsf{app}\{K_1\} \ e_0 \ e_1 : \bot \rrbracket}
3273
3274
3275
```

```
PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3277
3278
            We want to show (k, \Psi, \Sigma, \gamma(\mathsf{app}\{K_1\} e_0 e_1)) \in \mathcal{E}^T \llbracket \bot \rrbracket
3279
             By Lemma 4.56, we have that (\Sigma, e_0) \longrightarrow_T^* (\Sigma', e_0') where e_0' = \operatorname{Err}^{\bullet}, which is sufficient to complete the case.
                                                                                                                                                                                                                             3280
3281
                 Lemma 4.81 (T-Fst compatibility). \frac{\llbracket \Gamma \vdash e_0 : \tau_0 \times \tau_1 \rrbracket}{\llbracket \Gamma \vdash \operatorname{fst}\{K_0\} \ e_0 : K_0 \sqcap \tau_0 \rrbracket}
3282
3283
3284
                 PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma_1 \rrbracket such that \Sigma : (k, \Psi).
3285
            We want to show (k, \Psi, \Sigma, \gamma(\operatorname{fst}\{K_0\} e_0)) \in \mathcal{E}^T \llbracket \tau_0 \sqcap K_0 \rrbracket.
             Note \gamma(\text{fst}\{K_0\} e_1) = \text{fst}\{K_0\} \gamma(e_0).
             From the first hypothesis, we have (k, \Psi, \Sigma, \gamma(e_0)) \in \mathcal{E}^T \llbracket \tau_0 \times \tau_1 \rrbracket.
             Unfolding the expression relation, there are j, \Sigma', e'_0 such that (\Sigma, \gamma(e_0)) \longrightarrow_T^j (\Sigma'', e'_0) and e'_0 is irreducible.
3290
3291
             If e'_0 = \text{Err}^{\bullet} then we're done because the projection also steps to an error.
3292
             Otherwise, there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j\Psi') and (k-j, \Psi', \Sigma', e'_0) \in \mathcal{V}^T \llbracket \tau_0 \times \tau_1 \rrbracket.
3293
             Unfolding the location and value relations, we get that \Sigma'(e_0') = (\langle \ell_0, \ell_1 \rangle, \_).
3294
             By the OS, (\Sigma, fst\{K_0\} e_0) \longrightarrow_N^j (\Sigma' fst\{K_0\} e_0') \longrightarrow_T (\Sigma', assert K_0 \ell_0).
3295
3296
             We can apply Lemma 4.55 to the premise that (k-j, \Psi', \Sigma', \ell_0) \in \mathcal{V}^T \llbracket \tau_0 \rrbracket to get (k-j-1, \Psi', \Sigma', \ell_0) \in \mathcal{V}^T \llbracket \tau_0 \rrbracket.
3297
             Then we can apply Lemma 4.58 to get (k-j-1,\Psi',\Sigma',\text{assert }K_0\,\ell_0)\in\mathcal{E}^T[\![\tau_0\sqcap K_0]\!].
3298
             Finally, we can apply Lemma 4.51 to get that \Sigma': (k-j-1, \Psi'), which is sufficient to complete the proof.
                                                                                                                                                                                                                             3299
                 Lemma 4.82 (T-FstBot compatibility). \frac{\llbracket \Gamma \vdash e_0 : \bot \rrbracket}{\llbracket \Gamma \vdash \text{fst}\{K_0\} e_0 : \bot \rrbracket}
3301
3302
3303
                 PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3304
3305
            We want to show (k, \Psi, \Sigma, \gamma(\text{fst}\{K_0\} e_0)) \in \mathcal{E}^T \llbracket \bot \rrbracket.
3306
             By Lemma 4.56, we have that (\Sigma, e_0) \longrightarrow_T^* (\Sigma', e_0') where e_0' = \operatorname{Err}^{\bullet}, which is sufficient to complete the case.
                                                                                                                                                                                                                             3307
3308
                 Lemma 4.83 (T-Snd compatibility). \frac{\llbracket \Gamma \vdash e_0 : \tau_0 \times \tau_1 \rrbracket}{\llbracket \Gamma \vdash \operatorname{snd}\{K_1\} \ e_0 : K_1 \sqcap \tau_1 \rrbracket}
3309
3310
3311
                 PROOF. Not meaningfully different from the T-Fst case.
                                                                                                                                                                                                                             3312
                Lemma 4.84 (T-SndBot compatibility). \frac{\llbracket \Gamma \vdash e_0 : \bot \rrbracket}{\llbracket \Gamma \vdash \mathsf{snd}\{K_1\} \ e_0 : \bot \rrbracket}
3314
3315
3316
                 PROOF. Not meaningfully different from the T-FstBot case.
                                                                                                                                                                                                                             3317
3318
3319
                Lemma 4.85 (T-Binop compatibility). \frac{\llbracket \Gamma \vdash e_1 : \tau_1 \rrbracket}{\llbracket \Gamma \vdash binop \, e_0 \, e_1 : \Delta(binop, \tau_0, \tau_1) \rrbracket}
3320
3321
3322
3323
                 PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3324
            We want to show (k, \Psi, \Sigma, \gamma(binop e_0 e_1)) \in \mathcal{E}^T \llbracket K_2 \rrbracket.
3325
             Note \gamma(binop e_0 e_1) = binop \gamma(e_0) \gamma(e_1).
3326
             By the first hypothesis applied to \gamma we have (k, \Psi, \Sigma, \gamma(e_0)) \in \mathcal{E}^T \llbracket \tau_0 \rrbracket.
3327
```

2022-11-18 03:01. Page 65 of 1-100.

```
Unfolding we get there are j, \Sigma', e'_0 such that (\Sigma, \gamma(e_0)) \longrightarrow_T^j (\Sigma', e'_0) and e'_0 is irreducible.
3329
3330
           If e'_0 = \text{Err}^{\bullet} then we're done, because the whole operation errors.
3331
           Otherwise there is a (k-j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k-j, \Psi') and (k-j, \Psi', \Sigma', e'_0) \in \mathcal{V}^T \llbracket \tau_0 \rrbracket.
3332
3333
3334
           Note by Lemma 4.55 and Lemma 4.51, we have (k - j, \Psi', \Sigma', \gamma) \in \mathcal{G}^T \llbracket \Gamma_1 \rrbracket and \Sigma' : (k - j, \Psi').
3335
           By the second hypothesis applied to \gamma we have (k - j, \Psi', \Sigma', \gamma(e_1)) \in \mathcal{E}^T \llbracket \tau_1 \rrbracket.
3336
3337
           Unfolding we get there are j', \Sigma'', e'_1 such that (\Sigma', \gamma(e_1)) \longrightarrow_T^{j'} (\Sigma'', e'_1) and e'_1 is irreducible.
3338
           If e'_1 = \text{Err}^{\bullet} then we're done, because the whole operation errors.
3339
           Otherwise, there is a (k-j-j',\Psi'') \supseteq (k-j,\Psi) such that \Sigma'': (k-j-j',\Psi'') and (k-j-j',\Psi'',\Sigma'',e_1') \in \mathcal{V}^T[\![\tau_1]\!].
3340
3341
3342
           From the definition of \Delta, K_2 = \text{Int or Nat or } \bot.
3343
3344
           In the case of \bot, we're done because either \tau_0 or \tau_1 is a \bot, which is a contradiction.
3345
           Otherwise, the cases proceed identically, so without loss of generality assume K_2 = Int.
3346
           \tau_0 = \tau_1 = \text{Int}, and therefore pointsto(\Sigma'', (e'_0)) = i_0 and pointsto(\Sigma'', e'_1) = i_1.
3347
           If binop = quotient and i_1 = 0 then (\Sigma'', binop e'_0 e'_1) \longrightarrow_T (\Sigma'', DivErr), so we're done.
3348
           If binop = \text{quotient and } i_1 \neq 0, then (\Sigma'', binop e_0' e_1') \longrightarrow_T (\Sigma'', i_0/i_1) \longrightarrow_T (\Sigma''[\ell \mapsto (i_0/i_1, \text{none})], \ell).
3349
3350
           Since i_0/i_1 \in \mathbb{Z}, we're done.
3351
           If binop = \text{sum then } (\Sigma'', binop e_0' e_1') \longrightarrow_T (\Sigma'', i_0 + i_1) \longrightarrow_T (\Sigma''[\ell \mapsto (i_0 + i_1, \text{none})], \ell).
3352
           Since i_0 + i_1 \in \mathbb{Z}, we're done.
                                                                                                                                                                                            3353
3354
                                                                                     \llbracket \Gamma \vdash e_0 : \mathsf{Bool} \rrbracket
3355
3356
3357
              3358
3359
3360
              Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3361
3362
          We want to show (k, \Psi, \Sigma, \gamma(\text{if } e_0 \text{ then } e_1 \text{ else } e_2)) \in \mathcal{E}^T \llbracket \tau_0 \sqcup \tau_1 \rrbracket.
3363
           Note \gamma(\text{if } e_0 \text{ then } e_1 \text{ else } e_2) = \text{if } \gamma(e_0) \text{ then } \gamma(e_1) \text{ else } \gamma(e_2).
3364
           From the first hypothesis applied to \gamma, we know (k, \Psi, \Sigma, \gamma(e_0)) \in \mathcal{E}^T \llbracket \text{Bool} \rrbracket.
3365
           Unfolding, we have that there is \Sigma', e'_0, j such that (\Sigma, e_0) \longrightarrow_T^j (\Sigma', e'_0) where e'_0 is irreducible.
3366
3367
           If e'_0 = \text{Err}^{\bullet} then we're done, because the entire if statement errors.
3368
           Otherwise, there is a (k - j, \Psi') \supseteq (k, \Psi) such that \Sigma' : (k - j, \Psi') and (k - j, \Psi', \Sigma', e'_0) \in \mathcal{V}^T \llbracket \mathsf{Bool} \rrbracket.
3369
           Unfolding the location and then the value relation, we get that pointsto(\Sigma', e'_0) = True or pointsto(\Sigma', e'_0) = False.
3370
3371
3372
                   • pointsto(\Sigma', e_0') = True: Note by OS, (\Sigma, if \gamma(e_0) then \gamma(e_1) else \gamma(e_2)) \longrightarrow_T^j (\Sigma', \text{if } e_0' \text{ then } \gamma(e_1) \text{ else } \gamma(e_2)) \longrightarrow_T
3373
                       (\Sigma', \gamma(e_1)).
3374
3375
                       By Lemma 4.55 and Lemma 4.51, we have (k-j-1,\Psi',\Sigma',\gamma) \in \mathcal{G}^T \llbracket \Gamma_1 \rrbracket and \Sigma' : (k-j-1,\Psi').
3376
                       From the second hypothesis, we get (k - j - 1, \Psi', \Sigma', \gamma(e_1)) \in \mathcal{E}^T \llbracket \tau_0 \rrbracket.
3377
                       Finally, by Lemma 4.61, we get (k-j-1, \Psi', \Sigma', \gamma(e_1)) \in \mathcal{E}^T \llbracket \tau_0 \sqcup \tau_1 \rrbracket which is sufficient to complete the proof.
3378
                   • pointsto(\Sigma', e'_0) = False: same as other case except replace e_1 with e_2.
3379
```

```
3381
                Proof.
                                                                                                                                                                                                              3382
                                                                                                   \llbracket \Gamma \vdash e_0 : \bot \rrbracket
3383
                                                                                                   \llbracket \Gamma \vdash e_1 : \tau_0 \rrbracket
3384
               Lemma 4.87 (T-IfBot compatibility). \frac{\llbracket \Gamma \vdash e_2 : \tau_1 \rrbracket}{\llbracket \Gamma \vdash \text{if } e_0 \text{ then } e_1 \text{ else } e_2 : \bot \rrbracket}
3385
3386
3387
3388
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3389
           We want to show (k, \Psi, \Sigma, \gamma(\text{if } e_0 \text{ then } e_1 \text{ else } e_2)) \in \mathcal{E}^T[\![\bot]\!].
3390
           By Lemma 4.56, we have that (\Sigma, e_0) \longrightarrow_T^* (\Sigma', e_0') where e_0' = \mathsf{Err}^{\bullet}, which is sufficient to complete the case.
3391
                                                                                                                                                                                                              3392
                                                                                \llbracket \Gamma \vdash e_0 : \tau_0 \rrbracket
3393
               Lemma 4.88 (T-Sub compatibility). \frac{\tau_0 \leqslant : \tau_1}{\llbracket \Gamma \vdash e_0 : \tau_1 \rrbracket}
3394
3395
3396
3397
                PROOF. Let (k, \Psi, \Sigma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket such that \Sigma : (k, \Psi).
3398
           We want to show (k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^T \llbracket \tau_2 \rrbracket.
3399
            From our hypothesis, we have (k, \Psi, \Sigma, \gamma(e_1)) \in \mathcal{E}^T \llbracket \tau_1 \rrbracket.
3400
3401
            We can apply Lemma 4.50 to finish the case.
                                                                                                                                                                                                              3402
3403
            4.4.4 Transient with Truer Transient Typing is Vigilant
3404
3405
                Theorem 4.89 (Transient with Truer Transient Typing is Vigilant). If \Gamma \vdash e : \tau then \llbracket \Gamma \vdash e : \tau \rrbracket_V^T
3406
3407
                PROOF. By induction over the typing derivation, using the compatability lemmas.
                                                                                                                                                                                                              3408
3409
3410
3411
3412
3413
3414
3415
3418
3419
3420
3421
3422
3423
3424
3425
3426
3427
3428
3429
```

4.5 Vigilance Fundamental Property for Transient with Tag Typing

Theorem 4.90 (Transient is Tag Vigilant). If $\Gamma \vdash_{\mathsf{tag}} e : K \ then \ \llbracket \Gamma \vdash_{\mathsf{tag}} e : K \ \rrbracket^T$

PROOF. By Theorem 3.10, we have that there exists some $\tau \leq K$ such that $\Gamma \vdash_{\mathsf{tru}} e : \tau$.

By Theorem 4.89, we have that $\llbracket \Gamma \vdash_{\mathsf{tru}} e : \tau \rrbracket^T$.

Unfolding this result and what we want to prove, we note the only distinction is that in what we have, we get $(k, \Psi, \Sigma, \gamma(e)) \in \mathcal{E}^T \llbracket \tau \rrbracket$, and what we want to prove is $(k, \Psi, \Sigma, \gamma(e)) \in \mathcal{E}^T \llbracket K \rrbracket$.

This follows directly from Lemma 4.61.

5 Contextual equivalence

5.1 Contextual Equivalence Logical Relation—No Store

```
3488
                    DivErr ≈ DivErr
3489
                   \mathsf{TypeErr}(\tau, v) \approx \mathsf{TypeErr}(\tau', v')
3490
3491
3492
                   \llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \leq e_2 : \tau \rrbracket_C^{\mathcal{L}} \triangleq \ \forall (k, \gamma_1, \gamma_2) \in \mathcal{G}^{\mathcal{L}} \llbracket \Gamma \rrbracket. \ (k, \gamma_1(e_1), \gamma_2(e_2)) \in \mathcal{E}^{\mathcal{L}} \llbracket \tau \rrbracket
3493
3494
                   \llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \approx e_2 : \tau \rrbracket_C^{\mathcal{L}} \triangleq \llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \leq e_2 : \tau \rrbracket_C^{\mathcal{L}} \wedge \llbracket \Gamma \vdash_{\mathsf{tru}} e_2 \leq e_1 : \tau \rrbracket_C^{\mathcal{L}}
3495
3496
3497
                   \mathcal{G}^{\mathcal{L}}\llbracket \Gamma, x : \tau \rrbracket \triangleq \{ (k, \gamma_1[x \mapsto v_1], \gamma_2[x \mapsto v_2]) \mid (k, \gamma_1, \gamma_2) \in \mathcal{G}^{\mathcal{L}}\llbracket \Gamma \rrbracket
3498
3499
                                                                                                                                             \wedge (k, v_1, v_2) \in \mathcal{V}^{\mathcal{L}} \llbracket \tau \rrbracket_L \}
3500
3501
3502
                   \mathcal{G}^{\mathcal{L}}[\![\bullet]\!] \triangleq \{(k,\emptyset,\emptyset)\}
3503
3504
3505
                   \mathcal{E}^{\mathcal{L}}[\![\tau]\!] \triangleq \{(k, e_1, e_2) \mid \forall j \leq k, e_1'. \ e_1 \longrightarrow_I^j e_1' \land \mathsf{irred}_L(e_1')
3506
3507
                                                                                                      \Rightarrow \exists e_2'. e_2 \longrightarrow_I^* e_2'
3508
                                                                                                                      \wedge (e_1' \approx e_2' \in \operatorname{Err}^{\bullet} \vee (k - j, e_1', e_2') \in \mathcal{V}^{\mathcal{L}} \llbracket \tau \rrbracket) \}
3510
3511
3512
                   \mathcal{V}^{\mathcal{L}}\llbracket \operatorname{Int} \rrbracket \triangleq \{(k, v_1, v_2 \mid v_1 = v_2 \in \mathbb{Z}\}\
3513
3514
                   \mathcal{V}^{\mathcal{L}} \llbracket \operatorname{\mathsf{Nat}} \rrbracket \triangleq \{ (k, v_1, v_2 \mid v_1 = v_2 \in \mathbb{N} \}
3515
3516
3517
                   \mathcal{V}^{\mathcal{L}} \llbracket \mathsf{Bool} \rrbracket \triangleq \{ (k, v_1, v_2 \mid v_1 = v_2 \in \mathbb{B} \}
3518
3519
3520
                  \mathcal{V}^{\mathcal{L}}[\![\tau_{1} \times \tau_{2}]\!] \triangleq \{(k, \langle v_{1,1}, v_{1,2} \rangle, \langle v_{2,1}, v_{2,2} \rangle) \mid (k, v_{1,1}, v_{2,1}) \in \mathcal{V}^{L}[\![\tau_{1}]\!] \land (k, v_{2,1}, v_{2,2}) \in \mathcal{V}^{L}[\![\tau_{2}]\!]\}
3521
3522
3523
3524
3525
3526
                  \mathcal{V}^{\mathcal{L}}\llbracket\tau_1 \to \tau_2\rrbracket \triangleq \{(k, v_1, v_2) \mid \forall j \le k,
3527
                                                                                                    \forall v_1', v_2' \text{ where } (j, v_1', v_2') \in \mathcal{V}^L [\![\tau_1]\!].
3528
3529
                                                                                                     \forall K, K' \text{ where } K \cap \tau_2 = K' \cap \tau_2.
3530
3531
                                                                                                     (j, \mathsf{app}\{K\} \ v_1 \ v_1', \mathsf{app}\{K'\} \ v_2 \ v_2') \in \mathcal{E}^L [\![K \sqcap \tau_2]\!]\}
3532
3533
```

 $\mathcal{V}^{\mathcal{L}}[\![*]\!] \triangleq \{(k, \Sigma_1, \Sigma_2, \ell_1, \ell_2) \mid (k - 1, v_1, v_2) \in \mathcal{V}^{\mathcal{L}}[\![\mathsf{Int}]\!]$ $(k - 1, v_1, v_2) \in \mathcal{V}^{\mathcal{L}}[\![\mathsf{Bool}]\!]$ $\vee (k - 1, v_1, v_2) \in \mathcal{V}^{\mathcal{L}}[\![* \times *]\!]$ $\vee (k - 1, v_1, v_2) \in \mathcal{V}^{\mathcal{L}}[\![* \to *]\!]$

 $\mathcal{V}^{\mathcal{L}}[\![\bot]\!] \triangleq \emptyset$

5.2 Context typing

Truer transient contexts:

```
3590
                  T-CTX-HOLE
                                                                                            Т-Стх-Lам
                                                                                                                                                                                                                            T-CTX-PAIR-1
3591
                                                                                                             \Gamma, (x:K) \vdash_{\mathsf{tru}} E : (\Gamma' \triangleright \tau) \leadsto \tau'
                                        \Gamma' \subseteq \Gamma
                                                                                                                                                                                                                            \Gamma \vdash_{\mathsf{tru}} E : (\Gamma' \triangleright \tau) \leadsto \tau_1 \qquad \Gamma \vdash_{\mathsf{tru}} e : \tau_2
3592
                                                                                            \Gamma \vdash_{\mathsf{tru}} \lambda(x : K) . E : (\Gamma', (x : K) \triangleright \tau) \leadsto * \rightarrow \tau'
                                                                                                                                                                                                                                      \Gamma \vdash_{\mathsf{tru}} \langle E, e \rangle : (\Gamma' \triangleright \tau) \leadsto \tau_1 \times \tau_2
                   \Gamma \vdash_{\mathsf{tru}} [] : (\Gamma' \triangleright \tau) \leadsto \tau
3593
3594
                                              T-CTX-PAIR-2
3595
                                                                                          \Gamma \vdash_{\mathsf{tru}} E : (\Gamma' \triangleright \tau) \leadsto \tau_2
                                                \Gamma \vdash_{\mathsf{tru}} e : \tau_1
                                                                                                                                                                                    \Gamma \vdash_{\mathsf{tru}} E : (\Gamma' \triangleright \tau) \leadsto * \rightarrow \tau_1
3596
                                                                                                                                                                                            \Gamma \vdash_{\mathsf{tru}} \mathsf{app}\{K\} E e : (\Gamma' \triangleright \tau) \leadsto K \sqcap \tau_1
3597
                                                           \Gamma \vdash_{\mathsf{tru}} \langle e, E \rangle : (\Gamma' \triangleright \tau) \leadsto \tau_1 \times \tau_2
                                               Т-Стх-АррВот-1
                                                                                                                                                                                    T-CTX-APP-2
 3600
                                                \Gamma \vdash_{\mathsf{tru}} E : (\Gamma' \triangleright \tau) \leadsto \bot \qquad \Gamma \vdash_{\mathsf{tru}} e : \tau_2
                                                                                                                                                                                                                                  \Gamma \vdash_{\mathsf{tru}} E : (\Gamma' \triangleright \tau) \leadsto \tau_2
                                                                                                                                                                                    \Gamma \vdash_{\mathsf{tru}} e : * \rightarrow \tau_1
3601
                                                                                                                                                                                            \Gamma \vdash_{\mathsf{tru}} \mathsf{app}\{K\} \ e \ E : (\Gamma' \triangleright \tau) \leadsto K \sqcap \tau_1
                                                         \Gamma \vdash_{\mathsf{tru}} \mathsf{app}\{K\} E e : (\Gamma' \triangleright \tau) \leadsto \bot
3602
3603
                     Т-Стх-АррВот-2
                                                                                                                                            T-CTX-FST
                                                                                                                                                                                                                                                    Т-Стх-ГѕтВот
3604
                                                              \Gamma \vdash_{\mathsf{tru}} E : (\Gamma' \triangleright \tau) \leadsto \tau_2
                                                                                                                                                   \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \tau_1 \times \tau_2
                                                                                                                                                                                                                                                           \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \bot
                      \Gamma \vdash_{\mathsf{tru}} e : \bot
3605
                              \Gamma \vdash_{\mathsf{tru}} \mathsf{app}\{K\} \ e \ E : (\Gamma' \triangleright \tau) \leadsto \bot \qquad \qquad \Gamma \vdash_{\mathsf{tru}} \mathsf{fst}\{K\} \ E : (\Gamma \triangleright \tau) \leadsto K \sqcap \tau_1
                                                                                                                                                                                                                                                  \Gamma \vdash_{\mathsf{tru}} \mathsf{fst}\{K\} E : (\Gamma \triangleright \tau) \leadsto \bot
3606
3607
                                                                  T-CTX-SND
                                                                                                                                                                                                    T-CTX-SNDBOT
3608
                                                                               \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \tau_1 \times \tau_2
                                                                                                                                                                                                               \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \bot
3609
3610
                                                                                                                                                                                                     \Gamma \vdash_{\mathsf{tru}} \mathsf{snd}\{K\} E : (\Gamma \triangleright \tau) \leadsto \bot
                                                                   \Gamma \vdash_{\mathsf{tru}} \mathsf{snd}\{K\} E : (\Gamma \triangleright \tau) \leadsto K \sqcap \tau_2
3611
                                                                                                                                                                                         T-CTX-BINOP-2
                                             T-CTX-BINOP-1
3613
                                                                                                                                                                                                                                        \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \tau_2
                                                   \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \tau_1
                                                                                                                   \Gamma \vdash_{\mathsf{tru}} e : \tau_2
                                                                                                                                                                                               \Gamma \vdash_{\mathsf{tru}} e : \tau_1
3614
                                              \Gamma \vdash_{\mathsf{tru}} binop E e : (\Gamma \vdash \tau) \leadsto \Delta(binop, \tau_1, \tau_2)
                                                                                                                                                                                         \Gamma \vdash_{\mathsf{tru}} binop E e : (\Gamma \triangleright \tau) \leadsto \Delta(binop, \tau_1, \tau_2)
3615
3616
                    T-CTX-BND-1
                                                                                                                                                                     T-CTX-IF-1
3617
                                                       \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \tau'
                                                                                                                                                                     \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \mathsf{Bool} \qquad \Gamma \vdash_{\mathsf{tru}} e_1 : \tau_1
3618
3619
                     \overline{\Gamma \vdash_{\mathsf{tru}} \mathsf{cast} \{ K_2 \Leftarrow K_1 \} E : (\Gamma \vdash \tau) \rightsquigarrow K_2 \sqcap K_1 \sqcap \tau'}
                                                                                                                                                                                    \Gamma \vdash_{\mathsf{tru}} \mathsf{if} \ E \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 : (\Gamma \triangleright \tau) \leadsto \tau_1 \sqcup \tau_2
3620
3621
                                                                                                T-CTX-IFBOT-1
3622
                                                                                                 \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \bot
                                                                                                                                                                \Gamma \vdash_{\mathsf{tru}} e_1 : \tau_1 \qquad \Gamma \vdash_{\mathsf{tru}} e_2 : \tau_2
 3623
                                                                                                                       \Gamma \vdash_{\mathsf{tru}} \mathsf{if} \ E \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 : (\Gamma \triangleright \tau) \leadsto \bot
                                                                                            T-CTX-IF-2
                                                                                             \Gamma \vdash_{\mathsf{tru}} e_b : \mathsf{Bool} \qquad \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \vdash \tau) \leadsto \tau_1 \qquad \Gamma \vdash_{\mathsf{tru}} e_2 : \tau_2
 3627
3628
                                                                                                                \Gamma \vdash_{\mathsf{tru}} \mathsf{if} \ e_b \ \mathsf{then} \ E \ \mathsf{else} \ e_2 : (\Gamma \triangleright \tau) \leadsto \tau_1 \sqcup \tau_2
3629
3630
                                                                                                T-CTX-IFBOT-2
3631
                                                                                                                                             \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \tau_1 \qquad \Gamma \vdash_{\mathsf{tru}} e_2 : \tau_2
                                                                                                \Gamma \vdash_{\mathsf{tru}} e_b : \bot
 3632
                                                                                                                       \Gamma \vdash_{\mathsf{tru}} \mathsf{if} \ e_b \ \mathsf{then} \ E \ \mathsf{else} \ e_2 : (\Gamma \triangleright \tau) \leadsto \bot
3633
3634
                                                                                             T-CTX-IF-3
3635
                                                                                                                                                                                               \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \triangleright \tau) \leadsto \tau_2
                                                                                             \Gamma \vdash_{\mathsf{tru}} e_h : \mathsf{Bool}
                                                                                                                                                 \Gamma \vdash_{\mathsf{tru}} e_1 : \tau_1
3636
3637
                                                                                                                 \Gamma \vdash_{\mathsf{tru}} \mathsf{if} \ e_b \ \mathsf{then} \ e_1 \ \mathsf{else} \ E : (\Gamma \triangleright \tau) \leadsto \tau_1 \sqcup \tau_2
3638
                                                                                                T-CTX-IFBOT-3
3639
                                                                                                \Gamma \vdash_{\mathsf{tru}} e_b : \bot
                                                                                                                                                                                           \Gamma \vdash_{\mathsf{tru}} E : (\Gamma \vdash \tau) \leadsto \tau_2 2022-11-18 03:01. Page 70 of 1–100.
                                                                                                                                              \Gamma \vdash_{\mathsf{tru}} e_1 : \tau_1
                                                                                                                       \Gamma \vdash_{\mathsf{tru}} \mathsf{if} \ e_h \mathsf{then} \ e_1 \mathsf{else} \ E : (\Gamma \triangleright \tau) \leadsto \bot
```

5.3 Contextual equivalence statement

We define a logical relation for contexts:

$$\llbracket \Gamma \vdash_{\mathsf{tru}} C_1 \approx C_2 : (\Gamma' \triangleright \tau) \leadsto \tau' \rrbracket \triangleq \forall e_1, e_2. \llbracket \Gamma' \vdash_{\mathsf{tru}} e_1 \approx e_2 : \tau \rrbracket \Rightarrow \llbracket \Gamma \vdash_{\mathsf{tru}} C_1[e_1] \approx C_2[e_2] : \tau' \rrbracket$$

We define an abbreviation for the notion that an expression reduces to an eventual value without encountering an error: $e \Downarrow \triangleq \exists e'. \ e \longrightarrow_L^* e' \land (val(e'))$

Theorem 5.1 (Expression relation implies reduction equivalence). If $\llbracket \Gamma \vdash_{\text{tru}} e_1 \approx e_2 : \tau \rrbracket$, then $e_1 \Downarrow \Leftrightarrow e_2 \Downarrow$.

PROOF. By applying Lemm 5.2 in both directions.

Lemma 5.2 (Expression relation implies reduction equivalence). If $\llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \leq e_2 : \tau \rrbracket$, then $e_1 \Downarrow \Rightarrow e_2 \Downarrow$.

PROOF. Since $e_1 \downarrow$, then there exists some e'_1, k s.t. $e_1 \longrightarrow_L^k e'_1$ and e'_1 is a value and hence irreducible.

We want to show that $e_2' \Downarrow$. Instantiate the premise with $(k, \emptyset, \emptyset)$, obtaining that $(k, e_1, e_2) \in \mathcal{E}^{\mathcal{L}}[\![\tau]\!]$. Instantiate j with k and e_1' with e_1' , observing that e_1' being a value entails it is irreducible. Then e_2' from this relation is just what we need, since e_2 reduces to it, and it is syntactically a value.

The usual definition of contextual equivalence is then:

$$\Gamma \vdash_{\mathsf{tru}} e_1 \approx^{\mathsf{ctx}} e_2 : \tau \triangleq \forall C, \bullet \vdash_{\mathsf{tru}} C : (\Gamma \vdash \tau) \leadsto \tau' \Rightarrow (C[e_1] \Downarrow \Leftrightarrow C[e_2] \Downarrow)$$

Theorem 5.3 (Binary relation is sound for contextual equivalence). If $[\Gamma \vdash_{\mathsf{tru}} e_1 \approx e_2 : \tau]$, then $\Gamma \vdash_{\mathsf{tru}} e_1 \approx^{ctx} e_2 : \tau$.

PROOF. Consider an arbitrary type τ' and context C s.t. \bullet $\vdash_{\mathsf{tru}} C : (\Gamma \triangleright \tau) \leadsto \tau'$. Then we must show that $C[e_1] \Downarrow \Leftrightarrow C[e_2] \Downarrow$. By Theorem 5.1, it is sufficient to show that $\llbracket \bullet \vdash_{\mathsf{tru}} C[e_1] \approx C[e_2] : \tau' \rrbracket$.

By Theorem 5.71, $\llbracket \bullet \vdash_{\mathsf{tru}} C \approx C : (\Gamma \triangleright \tau) \leadsto \tau' \rrbracket$. Unfolding this definition and instantiating it with e_1, e_2 , and our hypothesis about them, we obtain precisely the required conclusion.

5.4 Binary relation-Proofs

5.4.1 Lemmas Used Without Mention

Lemma 5.4 (Values are in the \mathcal{E} -relation). If $(k, v, v') \in \mathcal{V}^{\mathcal{L}}[\![\tau]\!]$, then $(k, v, v') \in \mathcal{E}^{\mathcal{L}}[\![\tau]\!]$.

PROOF. Consider arbitrary j s.t. $v \longrightarrow^j v_f \land \mathsf{irred}_{\mathcal{L}}(v_f)$. Note that j must be equal to 0 since values do not reduce. Then choose v' as the e'_2 of the expression relation; it is easy to see that v' reduces to v' in some number (0) of steps. By our assumption, $(k-0,v,v') \in \mathcal{V}^{\mathcal{L}}[\![\tau]\!]$, so we are done.

Lemma 5.5 (Anti-Reduction - Head Expansion - Expression Relation Commutes With Steps). If $(k,e_1',e_2') \in \mathcal{E}^T[\![\tau]\!]$ and $e_1 \longrightarrow_T^j e_1'$ and $e_2 \longrightarrow_T^{j'} e_2'$, then $(k+j,e_1,e_2) \in \mathcal{E}^T[\![\tau]\!]$

PROOF. Consider arbitrary j', e_1'' s.t. $e_1 \longrightarrow_T^{j'} e_1''$. If $j' \leq j$, by determinism of the operational semantics, e_1'' must not be irreducible and so we are trivially done. Otherwise, assume $\mathsf{irred}_T(e_1'')$ and $j' \leq k+j$; we must show that $\exists e_2''.e_2 \longrightarrow_T^* e_2'' \land (e_1'' \approx e_2'' \in \mathsf{Err}^{\bullet} \lor (k+j-j',e_1'',e_2'') \in \mathcal{V}^T[\![\tau]\!]$.

Instantiate the hypothesis with $(k+j'-j,e_1'')$. Since $k+j'-j\leq k$ and the operational semantics are deterministic, this gives us that $\exists e_2''.e_2' \longrightarrow_T^* e_2'' \land (e_1''\approx e_2''\in \operatorname{Err}^{\bullet} \lor (k+j-j',e_1'',e_2'')\in \mathcal{V}^T[\![\tau]\!]$, from which our conclusion follows immediately.

 Lemma 5.6 (Anti-Reduction - Head Expansion - Steps Commute With Expression Relation). If $(k+j,e_1,e_2) \in \mathcal{E}^T[\![\tau]\!]$ and $e_1 \longrightarrow_T^j e_1'$ and $e_2 \longrightarrow_T^{j'} e_2'$, then $(k,e_1',e_2') \in \mathcal{E}^T[\![\tau]\!]$

PROOF. Consider arbitrary j', e_1'' s.t. $j' \leq k \land \text{irred}_T(e_1'') \land e_1' \longrightarrow_T^{j'} e_1''$.

We must show that $\exists e_2^{\prime\prime}.e_2^{\prime} \longrightarrow_T^* e_2^{\prime\prime} \land (e_1^{\prime\prime} \approx e_2^{\prime\prime} \in \operatorname{Err}^{\bullet} \lor (k-j^{\prime},e_1^{\prime\prime},e_2^{\prime\prime}) \in \mathcal{V}^T[[\tau]].$

Instantiate the hypothesis with $j+j', e_1''$. Since $j' \le k, j+j' \le k+j$. Since the operational semantics are deterministic and transitive, the other conditions apply. Then the hypothesis provides precisely the appropriate e_2'' and conditions on it and e_1'' .

We define a notion of tags extended with bottom that are compatible with the usual lattice:

$$K^{\perp} = K \mid \perp$$

$$\lfloor K^{\perp} \rfloor^{\perp} = \begin{cases} \perp & \text{if } K^{\perp} = \perp \\ \lfloor K^{\perp} \rfloor & \text{otherwise} \end{cases}$$

$$\sim^{\perp} (K^{\perp}, v) = \begin{cases} \text{False} & \text{if } K^{\perp} = \perp \\ v \sim K^{\perp} & \text{otherwise} \end{cases}$$

Lemma 5.7 (Tagof-bot is compatible with Meet). $\lfloor K_1^{\perp} \sqcap K_2^{\perp} \rfloor^{\perp} = \lfloor K_1^{\perp} \rfloor^{\perp} \sqcap \lfloor K_2^{\perp} \rfloor^{\perp}$.

PROOF. Immediate, by unfolding definitions and case analysis.

Lemma 5.8 (Relation implies tagmatch). If $(k, v, v') \in \mathcal{V}^{\mathcal{L}}[\![\tau]\!]$ and $K^{\perp} \leq \lfloor \tau \rfloor^{\perp}$, then $\sim^{\perp} (K^{\perp}, v)$.

PROOF. By case analysis on τ and K^{\perp} ; in each case this follows immediately from unfolding the definitions of \mathcal{V} and tagmatch.

5.4.2 Lemmas Used With Mention

Lemma 5.9 (Related values have matching constructors). If $(k, v, v') \in \mathcal{V}^{\mathcal{L}}[\![\tau]\!]$, then either

- v = v'
- There exist some v_1, v_2, v_1', v_2' s.t. $v = \langle v_1, v_2 \rangle$ and $v' = \langle v_1', v_2' \rangle$
- There exist some w, w' s.t. v = w and v' = w'.

PROOF. By induction on τ , unfolding the definition of V in each case.

Lemma 5.10 (Tagmatch is up to approximation). If $(k, v, v') \in \mathcal{V}^T[\![\tau]\!]$, then $\sim^{\perp} (K^{\perp}, v) \Leftrightarrow \sim^{\perp} (K^{\perp}, v')$.

PROOF. By Lemma 5.9 and inspection of the definition of $\sim^{\perp} (K^{\perp}, v)$.

Lemma 5.11 (Tagmatch respects meets). $\sim^{\perp} (K_1^{\perp} \sqcap K_2^{\perp}, v) \Leftrightarrow \sim^{\perp} (K_1^{\perp}, v) \wedge \sim^{\perp} (K_2^{\perp}, v)$.

PROOF. By case analysis on K_1^{\perp} , K_2^{\perp} ; in each case the conclusion follows immediately by unfolding.

Lemma 5.12 (Tagmatch implies values in relation at meet). If $(k, v, v') \in \mathcal{V}^T[\![\tau]\!]$ and $\sim^{\perp} (K^{\perp}, v)$, then $(k-1, v, v') \in \mathcal{V}^T[\![K^{\perp} \sqcap \tau]\!]$.

PROOF. Proceed by case analysis on K^{\perp} :

* By lattice properties, $K^{\perp} \sqcap \tau = \tau$, so this is trivial by Lemma ??.

Nat By the definition of tagmatch, v must be a natural number. By inspection, this is possible only when τ is *, Int, or Nat; in each case, $K^{\perp} \sqcap \tau = \text{Nat}$. By inspection on the relation, v always satisfied what is needed.

Int Analogous to the Nat case above.

- *** By the definition of tagmatch, v must be a pair; by inspection this is possible only if τ is * or some pair type. If the latter, $K^{\perp} \sqcap \tau = \tau$, and so the conclusion is immediate; otherwise, $K^{\perp} \sqcap \tau = *\times *$, and the conclusion is immediate from the definition of the * case of the relation.
- * \to * By the definition of tagmatch, v must be a w; by inspection this is possible only if τ is * or some function type. If the latter, $K^{\perp} \sqcap \tau = \tau$, and so the conclusion is immediate; otherwise, $K^{\perp} \sqcap \tau = *\to *$, and the conclusion is immediate from the definition of the * case of the relation.e
- ⊥ Contradiction

```
LEMMA 5.13 (\mathcal{E}-\mathcal{V}-MONOTONICITY). (1) If (k, e_1, e_2) \in \mathcal{E}^T \llbracket \tau \rrbracket and j \leq k, then (j, e_1, e_2) \in \mathcal{E}^T \llbracket \tau \rrbracket. (2) If (k, v_1, v_2) \in \mathcal{V}^T \llbracket \tau \rrbracket and j \leq k, then (j, v_1, v_2) \in \mathcal{V}^T \llbracket \tau \rrbracket.
```

PROOF. Proceed by simultaneous induction on k and τ :

- k = 0: 1) follows immediately from 2).
 Proceeds similarly to the other case, but function and dynamic cases are vacuously true.
- k > 0:
 - Unfolding the expression relation in our hypothesis, we get that there is some e'₁, j' such that e₁ → j'_T e'₁, and some e'₂ such that e₂ → *_T e'₂.
 If e'₁ = Err[•] then we're done.

Otherwise, $(k - j', e'_1, e'_2) \in \mathcal{V}^T [\![\tau]\!]$.

Now, unfolding the expression relation, we want to show $(k - j - j', e'_1, e'_2) \in \mathcal{V}^T[\tau]$.

We can apply the IH 2) with the fact proven in a).

2) We want to show that $(k - j, v_1, v_2) \in \mathcal{V}^T \llbracket \tau \rrbracket$.

We case split on τ :

- i) $\tau = \text{Nat}$: then where $n \in \mathbb{N}$, so the case is immediate.
- ii) $\tau = tint$: same as above.
- iii) τ = Bool: same as above.
- iv) $\tau = \tau_1 \times \tau_2$: Then unfolding our hypothesis gives us $v_1 = \langle v_1', v_1'' \rangle$ and $v_2 = \langle v_1', v_1'' \rangle$ with $(k, v_1', v_2') \in \mathcal{V}^T \llbracket \tau_1 \rrbracket$ and $(k, v_1'', v_2'') \in \mathcal{V}^T \llbracket \tau_2 \rrbracket$.

The case follows by applying the IH 2) to both premises.

v)
$$\tau = * \to \tau_2$$
: Let $j' \le k - j$.
Let $(j', v'_1, v'_2) \in \mathcal{V}^T [\![*]\!]$.
Let K, K' .

```
3798
                                       Since j' \le k - j \le k, we can apply the hypothesis to complete the case.
3799
                                 vi) \tau = *: we want to show (k - 1, v_1, v_2) \in \mathcal{V}^T \llbracket \text{Int} \rrbracket \text{ or } \mathcal{V}^T \llbracket \text{Bool} \rrbracket \text{ or } \mathcal{V}^T \llbracket * \times * \rrbracket \text{ or } \mathcal{V}^T \llbracket * \to * \rrbracket.
3800
                                       This follows from IH 2) (smaller by index).
3801
3802
                                                                                                                                                                                            3803
              LEMMA 5.14 (MONADIC BIND). Suppose that E_1, E_2 are any evaluation contexts (n.b. not a general context, as used else-
3804
3805
           where in these proofs), (k, e_1, e_2) \in \mathcal{E}^T[\![\tau]\!], and for all k', v_1, v_2, if k' \leq k \land (k', v_1, v_2) \in \mathcal{V}^T[\![\tau]\!] then (k', E_1[v_1], E_2[v_2]) \in \mathcal{E}^T[\![\tau]\!]
           \mathcal{E}^T \llbracket \tau' \rrbracket.
               Then (k, E_1[e_1], E_2[e_2]) \in \mathcal{E}^T [\![\tau']\!].
3809
              PROOF. Consider arbitrary j, e_1' s.t. j \leq k \wedge E_1[e_1] \longrightarrow_T^j e_1' \wedge \text{irred}_T(e_1'). Then we must show that must show that
3810
          \exists e_2'.E_2[e_2] \longrightarrow_T^* e_2' \land (e_1' \approx e_2' \in \mathsf{Err}^{\bullet} \lor (k-j, e_1', e_2') \in \mathcal{V}^T[\![\tau]\!]).
3811
              Because E_1[e_1] reaches an irreducible term in at most j steps, by our operational semantics e_1 must itself reduce to
3812
3813
           some irreducible term e_3 in some smaller number of steps j' \leq j. Then since j' \leq j \wedge e_1 \longrightarrow_T^{j'} e_3 \wedge \text{irred}_T(e_3), we can
3814
           instantiate our first assumption, obtaining that there similarly exists e_4 s.t. e_2 \longrightarrow_T^* e_4 \land (e_3 \approx e_4 \in \mathsf{Err}^{\bullet} \lor (k-j', e_3, e_4) \in \mathsf{Err}^{\bullet} \lor (k-j', e_3, e_4) \in \mathsf{Err}^{\bullet} \lor (k-j', e_3, e_4)
3815
3816
               Suppose that e_3 \approx e_4 \in \text{Err}^{\bullet}. Then by the operational semantics, E_1[e_1] and E_2[e_2] reduce to the same errors, so
3817
3818
           instantiating e'_1 and e'_2 with them proves our goal.
3819
               Otherwise, we know that (k-j', e_3, e_4) \in \mathcal{V}^T[\![\tau]\!]. We may therefore instantiate our other assumption with k-j', e_3, e_4
           and this fact, obtaining that (k - j', E_1[e_3], E_2[e_4]) \in \mathcal{E}^T[[\tau]]. We still must show that \exists e_2'. E_2[e_2] \longrightarrow_T^* e_2' \land (e_1' \approx e_2 \in \mathbb{E}^T[[\tau]])
3821
           \operatorname{Err}^{\bullet} \vee (k - j, e'_1, e'_2) \in \mathcal{V}^T \llbracket \tau \rrbracket ).
3822
3823
               Instantiate the result of our assumption with step index j - j' \le k - j' and e'_1. By determinism of the operational
3824
          semantics, E_1[e_3] \longrightarrow_T^{j-j'} e_1', so we obtain that \exists e_2'.E_2[e_4] \longrightarrow_T^* e_2' \land (e_1' \approx e_2' \in \mathsf{Err}^{\bullet} \lor (k-j'-(j-j'), e_1', e_2') \in \mathcal{V}^T[\![\tau]\!]).
3825
           Note that k - j' - (j - j') = k - j, and that since E_2[e_4] \longrightarrow_T^* e_2' and e_2 \longrightarrow_T^* e_4, then E_2[e_2] \longrightarrow_T^* e_2', so this is precisely
3826
3827
           the e_2' that we needed to show the existence of.
3828
              Lemma 5.15 (Check compatibility). If (k, v, v') \in \mathcal{E}^T \llbracket \tau \rrbracket and \tau' = K \sqcap \tau = K' \sqcap \tau, then (k, \text{assert } K v, \text{assert } K' v') \in \mathcal{E}^T \llbracket \tau \rrbracket
3829
          \mathcal{E}^T \llbracket \tau' \rrbracket.
3830
3831
              PROOF. Proceed by case analysis on K \sqcap \tau:
3832
           K \cap \tau = \tau Then it must be the case that K \sim v and K' \sim v', meaning assert Kv \longrightarrow_T v and assert K'v' \longrightarrow_T v', which
3834
                       is sufficient to complete the case.
3835
           K \sqcap \tau = \text{Nat } \text{and } \tau = \text{Int } \text{Unfolding our hypothesis, we get that } v = v' \text{ and } v \in \mathbb{Z}.
3836
                       If v \in \mathbb{N}, then assert Kv \longrightarrow_T v and assert K'v' \longrightarrow_T v', which is sufficient to complete the case.
3837
3838
                       Otherwise, assert Kv \longrightarrow_T \text{TypeErr}(\text{Nat}, v) and assert K'v' \longrightarrow_T \text{TypeErr}(\text{Nat}, v'), which is sufficient to
3839
                      complete the case.
3840
          K \sqcap \tau = \bot Then assert Kv \longrightarrow_T TypeErr(Nat, v) and assert Kv' \longrightarrow_T TypeErr(Nat, v'), which is sufficient to com-
3841
3842
                       plete the case.
3843
          K \sqcap \tau = K and \tau \neq K Then \tau = * and K = K'.
3844
                      We can unfold our hypothesis to get that (k-1, v, v') \in \mathcal{V}^T \llbracket K'' \rrbracket for some K'', which implies v' \sim v.
3845
                       By the OS, either assert Kv \longrightarrow v and v \sim K, or assert Kv \longrightarrow \mathsf{TypeErr}(K, v) and \neg v \sim K.
                      In either case, we have the corresponding property needed to complete the case.
                                                                                                                                                    2022-11-18 03:01. Page 74 of 1-100.
```

We want to show $(j', \operatorname{app}\{K\} v_1 v_1', \operatorname{app}\{K'\} v_2 v_2') \in \mathcal{E}^T \llbracket K \sqcap \tau_2 \rrbracket$.

3891

3893 3894

3895

3896

3897

3900

Lemma 5.16 (Dynamic Checks Are No-ops). If $(k+1, \text{assert} * v, \text{assert} * v') \in \mathcal{E}^T[\![\tau]\!]$, then $(k, v, v') \in \mathcal{E}^T[\![\tau]\!]$

PROOF. By the OS, assert $*v \longrightarrow v$ and assert $*v' \longrightarrow v'$.

Then by our hypothesis, $(k, v, v') \in \mathcal{V}^T \llbracket \tau \rrbracket$, which is sufficient to complete the proof.

Lemma 5.17 (Subtyping Compatibility). (1) If $(k, v_1, v_2) \in \mathcal{V}^T[\![\tau]\!]$ and $\tau \leqslant \tau'$ then $(k, v_1, v_2) \in \mathcal{V}^T[\![\tau']\!]$ (2) If $(k, e_1, e_2) \in \mathcal{E}^T[\![\tau]\!]$ and $\tau \leqslant \tau'$ then $(k, e_1, e_2) \in \mathcal{E}^T[\![\tau']\!]$.

Proof. Proceed by mutual induction on k and τ :

- k = 0: 2 is immediate if $e \neq v$.
 - If e = v then 2 follows immediately from 1.

1 follows identically in the k = 0 case as it does in the k > 0 case, but the function case is vacuously true.

- k > 0:
 - (1) Case split on $\tau \leqslant : \tau'$:
 - i) $\tau \leqslant \tau$: immediate.
 - ii) Nat \leq : Int: immediate because $\mathbb{T} \subseteq \mathbb{Z}$.
 - iii) $\tau_1 \times \tau_2 \leqslant \tau_1' \times \tau_2'$, with $\tau_1 \leqslant \tau_1'$ and $\tau_2 \leqslant \tau_2'$:

We want to show $(k, v_1, v_2) \in \mathcal{V}^T \llbracket \tau' \rrbracket$.

Unfolding our hypothesis, we get that $v_1 = \langle v_1', v_1'' \rangle$ and similarly for v_2 .

We want to show $(k, v'_1, v'_2) \in \mathcal{V}^T [\![\tau'_1]\!]$ and $(k, v''_1, v''_2) \in \mathcal{V}^T [\![\tau'_2]\!]$.

We can apply IH 1) to both of judgements in our hypothesis to get $(k, v_1', v_2') \in \mathcal{V}^T[\![\tau_1']\!]$ and $(k, v_1'', v_2'') \in \mathcal{V}^T[\![\tau_2']\!]$.

This is sufficient to show $(k, v_1, v_2) \in \mathcal{V}^T \llbracket \tau' \rrbracket$.

iv) $* \to \tau_2 \leqslant : * \to \tau'_2$, with $\tau_2 \leqslant : \tau'_2$:

We want to show $(k, v_1, v_2) \in \mathcal{V}^T \llbracket \tau' \rrbracket$.

Let
$$j \le k$$
 and $(j, v'_1, v'_2) \in \mathcal{V}^T [\![*]\!]$.

Let K.

We want to show $(j, \mathsf{app}\{K\}\,v_1\,v_1', \mathsf{app}\{K\}\,v_2\,v_2') \in \mathcal{E}^T[\![\tau_2' \sqcap K]\!].$

Then, we can apply our hypothesis about v_1, v_2 to get $(j, \operatorname{app}\{K\} v_1 v_1', \operatorname{app}\{K\} v_2 v_2') \in \mathcal{E}^T \llbracket \tau_2 \sqcap K \rrbracket$. Finally, we can apply IH 1) to get $(j, \operatorname{app}\{K\} v_1 v_1', \operatorname{app}\{K\} v_2 v_2') \in \mathcal{E}^T \llbracket \tau_2' \sqcap K \rrbracket$ which is what we wanted to show.

(2) Unfolding our hypothesis, there is some $j \le k$ and irreducible e'_1, e'_2 such that $e_1 \longrightarrow_T^j e'_1$ and $e_2 \longrightarrow_T^* e'_2$. If $e'_1, e'_2 \in \operatorname{Err}^{\bullet}$ then we're done.

Otherwise, $(k - j, e'_1, e'_2) \in \mathcal{V}^T \llbracket \tau \rrbracket$.

By IH 1), we have $(k - j, e'_1, e'_2) \in \mathcal{V}^T[\![\tau']\!]$, which is what we wanted to show.

Lemma 5.18 (Monitor Compatibility). If $(k, v, v') \in \mathcal{V}^T[\![\tau]\!]$, then $(k+1, \text{mon } \{K'_1 \Leftarrow K_1\}, \text{mon } \{K'_2 \Leftarrow K_2\} v') \in \mathcal{E}^T[\![\tau]\!]$.

PROOF. By induction on k and v:

2022-11-18 03:01. Page 75 of 1-100.

```
3901
           k = 0 By case analysis on v, v':
3902
                        i, i' By OS, mon \{K'_1 \Leftarrow K_1\} i \longrightarrow i and mon \{K'_2 \Leftarrow K_2\} i' \longrightarrow i'e, so this is immediate.
3903
                        True, True As in case i above.
3904
                        False, False As in case True above.
3905
                        \langle v_1, v_2 \rangle, \langle v_1', v_2' \rangle Since (k, v_1, v_2) \in \mathcal{V}^T[\![\tau]\!], by inspection \tau must be either \tau_1 \times \tau_2 or *:
3906
3907
                               \tau_1 \times \tau_2 Note that mon \{K_1' \in K_1\} \langle v_1, v_2 \rangle \longrightarrow \langle \text{mon } \{fst(K_1') \in fst(K_1)\} v_1, \text{mon } \{snd(K_1') \in snd(K_1)\} v_2 \rangle,
3908
                                        and similarly mon \{K_2' \Leftarrow K_2\} \langle v_1', v_2' \rangle \longrightarrow \langle \text{mon} \{fst(K_2') \Leftarrow fst(K_2)\} v_1', \text{mon} \{snd(K_2') \Leftarrow snd(K_2)\} v_2' \rangle
                                        It is therefore sufficient to show that
           3912
3913
                                        By unfolding, this is the same as showing (k, \text{mon } \{fst(K'_1) \leftarrow fst(K_1)\} v_1, \text{mon } \{fst(K'_2) \leftarrow fst(K_2)\} v'_1) \in fst(K_2)\} v'_1
3914
                                        \mathcal{E}^T \llbracket \tau_1 \rrbracket and (k, \text{mon } \{snd(K'_1) \Leftarrow snd(K_1)\} v_2, \text{mon } \{snd(K'_2) \Leftarrow snd(K_2)\} v'_2) \in \mathcal{E}^T \llbracket \tau_2 \rrbracket.
3915
                                        By Lemma 5.13, it suffices to show (k+1, \text{mon } \{fst(K_1') \Leftarrow fst(K_1)\} v_1, \text{mon } \{fst(K_2') \Leftarrow fst(K_2)\} v_1') \in
3916
3917
                                        \mathcal{E}^{T}[[\tau_{1}]] and (k+1, \text{mon } \{snd(K'_{1}) \Leftarrow snd(K_{1})\} v_{2}, \text{mon } \{snd(K'_{2}) \Leftarrow snd(K_{2})\} v'_{2}) \in \mathcal{E}^{T}[[\tau_{2}]].
3918
                                        In both cases, IH applies and hence it suffices to show (k, v_1, v_1') \in \mathcal{E}^T \llbracket \tau_1 \rrbracket and (k, v_2, v_2') \in \mathcal{E}^T \llbracket \tau_2 \rrbracket.
3919
                                        These are both obtained by unfolding our assumption.
3920
                               * Impossible, since k = 0.
3921
3922
                       w, w' Since (k, w, w') \in \mathcal{V}^T[[\tau]], by inspection \tau must be either * \to \tau' or *:
                               * \to \tau' Note that mon \{K_1' \Leftarrow K_1\} w \longrightarrow \operatorname{grd} \{K_1' \Leftarrow K_1\} w, and similarly mon \{K_2' \Leftarrow K_2\} w' \longrightarrow \operatorname{grd} \{K_2' \Leftarrow K_2\} w'.
                                        Consequently, it is sufficient to show that (k, \operatorname{grd} \{K_1' \Leftarrow K_1\} w, \operatorname{grd} \{K_2' \Leftarrow K_2\} w') \in \mathcal{E}^T[\![* \to \tau']\!].
                                        Consider arbitrary j \le k, v, v' s.t. (j, v, v') \in \mathcal{V}^T[\![*]\!], K, K'. Then we must show that
3926
                                        (j, \operatorname{\mathsf{app}}\{K\} (\operatorname{\mathsf{grd}}\{K_1' \Leftarrow K_1\} w) v, \operatorname{\mathsf{app}}\{K'\} (\operatorname{\mathsf{grd}}\{K_2' \Leftarrow K_2\} w') v') \in \mathcal{E}^T \llbracket K \sqcap \tau' \rrbracket.
3927
3928
                                        By assumption, k = 0, so j = 0. Therefore, this is vacuously true.
3929
                               * Impossible, since k = 0.
3930
                       otherwise Impossible by Lemma 5.9.
3932
           k > 0 By case analysis on v, v':
3933
                       i, i' As in k = 0 case.
3934
                       True, True As in k = 0 case.
3935
                        False, False As in k = 0 case.
                        \langle v_1, v_2 \rangle, \langle v_1', v_2' \rangle Since (k, v_1, v_2) \in \mathcal{V}^T \llbracket \tau \rrbracket, by inspection \tau must be either \tau_1 \times \tau_2 or *:
                               \tau_1 \times \tau_2 As in k = 0 case.
3939
                               * By unfolding, (k-1, w, w') \in \mathcal{V}^T [*\times *]. By an argument essentially identical to the previous case,
3940
                                        merely reducing one application of monotonicity by one is sufficient to show what is needed.
3941
                       w, w' Since (k, w, w') \in \mathcal{V}^T \llbracket \tau \rrbracket, by inspection \tau must be either * \to \tau' or *:
3942
3943
                               * \to \tau' Note that mon \{K_1' \Leftarrow K_1\} w \longrightarrow \operatorname{grd} \{K_1' \Leftarrow K_1\} w, and similarly mon \{K_2' \Leftarrow K_2\} w' \longrightarrow \operatorname{grd} \{K_2' \Leftarrow K_2\} w'.
3944
                                        Consequently, it is sufficient to show that (k, \operatorname{grd} \{K_1' \Leftarrow K_1\} w, \operatorname{grd} \{K_2' \Leftarrow K_2\} w') \in \mathcal{E}^T \llbracket * \to \tau' \rrbracket.
3945
                                        Consider arbitrary j \le k, v, v' s.t. (j, v, v') \in \mathcal{V}^T[\![*]\!], K, K' s.t. K \cap \tau' = K' \cap \tau'. Then we must show
3946
3947
3948
                                        (j, \operatorname{app}\{K\} (\operatorname{grd}\{K'_1 \Leftarrow K_1\} w) v, \operatorname{app}\{K'\} (\operatorname{grd}\{K'_2 \Leftarrow K_2\} w') v') \in \mathcal{E}^T \llbracket K \sqcap \tau' \rrbracket.
                                        By OS, it suffices to show that
                                        (j-1, \operatorname{assert} K((\operatorname{grd}\{K_1' \Leftarrow K_1\} w) v), \operatorname{assert} K'((\operatorname{grd}\{K_2' \Leftarrow K_2\} w') v')) \in \mathcal{E}^T \llbracket K \sqcap \tau' \rrbracket.
                                                                                                                                                         2022-11-18 03:01. Page 76 of 1-100.
```

```
By Lemma 5.15, it suffices to show that (j-1,(\operatorname{grd}\{K_1' \Leftarrow K_1\}\,w)\,\,v,(\operatorname{grd}\{K_2' \Leftarrow K_2\}\,w')\,\,v') \in K_1'
3953
3954
                                       \mathcal{E}^T \llbracket \tau' \rrbracket
3955
                                       By OS, it suffices to show that
                                       (j-2, mon \{cod(K'_1) \Leftarrow cod(K_1)\} w mon \{dom(K_1) \Leftarrow dom(K'_1)\} v,
                                       mon \{cod(K_2) \Leftarrow cod(K_2)\} w' mon \{dom(K_2) \Leftarrow dom(K_2')\} v'\}
                                       \in \mathcal{E}^T \llbracket \tau' \rrbracket.
                                       By IH, it suffices to show that (j-3, w \text{ mon } \{dom(K_1) \Leftarrow dom(K_1')\} v, w' \text{ mon } \{dom(K_2) \Leftarrow dom(K_2')\} v' \} \in dom(K_2') 
                                       \mathcal{E}^T \llbracket \tau' \rrbracket
                                       By Lemma 5.16, it suffices to show that
                                       (j-2, \text{assert} * w \text{ mon } \{dom(K_1) \Leftarrow dom(K_1')\} v, \text{assert} * w' \text{ mon } \{dom(K_2) \Leftarrow dom(K_2')\} v') \in
3965
                                      \mathcal{E}^T \llbracket \tau' \rrbracket.
3966
                                      By the definition of meet and OS, this is equivalent to
3967
                                       (j-1, \operatorname{app}\{*\} w \operatorname{mon} \{\operatorname{dom}(K_1) \Leftarrow \operatorname{dom}(K_1')\} v, \operatorname{app}\{*\} w' \operatorname{mon} \{\operatorname{dom}(K_2) \Leftarrow \operatorname{dom}(K_2')\} v') \in \mathcal{E}^T[\![*\sqcap] 
3968
                                       \tau'].
3970
                                      By unfolding the assumption that (k, w, w') \in \mathcal{E}^T \llbracket * \to \tau' \rrbracket, it suffices to show that
3971
                                       (j-1, mon \{dom(K_1) \Leftarrow dom(K_1')\} v, mon \{dom(K_2) \Leftarrow dom(K_2')\} v') \in \mathcal{E}^T[[*]].
3972
                                       By IH, it suffices to show that (j-2, v, v') \in \mathcal{E}^T [\![ * ]\!].
                                      By Lemma 5.13, it suffices to show that (j, v, v') \in \mathcal{E}^T[\![*]\!].
                                      This is immediate from the assumption that (i, v, v') \in \mathcal{V}^T[\![*]\!].
                              * By unfolding, (k-1, w, w') \in \mathcal{V}^T [\![ * \to * ]\!]. By an argument essentially identical to the previous case,
                                       merely reducing one application of monotonicity by one is sufficient to show what is needed.
3979
                       otherwise Impossible by Lemma 5.9.
3980
```

COROLLARY 5.19. If $(k, e_1, e_2) \in \mathcal{E}^T \llbracket \tau \rrbracket$, then $(k + 1, \text{mon } \{K'_1 \Leftarrow K_1\}, \text{mon } \{K'_2 \Leftarrow K_2\} e_2) \in \mathcal{E}^T \llbracket \tau \rrbracket$.

PROOF. Unfolding the expression relation in our hypothesis, we get that there is a j and e'_1 such that $e_1 \longrightarrow_T^j e'$ such that e' is irreducible, and an e'_2 such that $e_2 \longrightarrow_T^* e'_2$ and either they're errors, or $(k-j,e'_1,e'_2) \in \mathcal{V}^T[\![\tau]\!]$. If they're errors, then we're done because the monitors will also step to errors.

Otherwise, we have mon $\{K_1' \Leftarrow K_1\} \longrightarrow_T^j \text{mon } \{K_1' \Leftarrow K_1\}$ and mon $\{K_2' \Leftarrow K_2\} \longrightarrow_T^j \text{mon } \{K_2' \Leftarrow K_2\}$. By Lemma 5.18, we have that $(k-j, \text{mon } \{K_1' \Leftarrow K_1\}, \text{mon } \{K_2' \Leftarrow K_2\}) \in \mathcal{E}^T[\![\tau]\!]$, which is sufficient to complete the proof.

Lemma 5.20 (Boundary Compatibility). If $(k, v_1, v_2) \in \mathcal{V}^T[\![\tau]\!]$ and $\tau' = K_1' \sqcap K_1 \sqcap \tau = K_2' \sqcap K_2 \sqcap \tau$, then $(k + 1, \text{cast } \{K_1' \Leftarrow K_1\} \ v_1, \text{cast } \{K_2' \Leftarrow K_2\} \ v_2) \in \mathcal{E}^T[\![\tau']\!]$.

PROOF. By Lemma 5.10, notice that $\sim^{\perp} (\lfloor \tau' \rfloor^{\perp}, v_1) \Leftrightarrow \sim^{\perp} (\lfloor \tau' \rfloor^{\perp}, v_2)$. By Lemma 5.11 and our assumption, therefore, $\sim^{\perp} (K'_1, v_1) \wedge \sim^{\perp} (K_1, v_1) \wedge \sim^{\perp} (\lfloor \tau \rfloor^{\perp}, v_1) \Leftrightarrow \sim^{\perp} (K'_2, v_2) \wedge \sim^{\perp} (K_2, v_2) \wedge \sim^{\perp} (\lfloor \tau \rfloor^{\perp}, v_2)$. By Lemma 5.10, $\sim^{\perp} (\lfloor \tau \rfloor^{\perp}, v_1) \Leftrightarrow \sim^{\perp} (\lfloor \tau \rfloor^{\perp}, v_2)$. Consequently, $\sim^{\perp} (K'_1, v_1) \wedge \sim^{\perp} (K_1, v_1) \Leftrightarrow \sim^{\perp} (K'_2, v_2) \wedge \sim^{\perp} (K_2, v_2)$ —which is to say, either both of the values match both of their annotated tags, or both of them do not match at least one of their annotated tags.

Consider then each case: 2022-11-18 03:01. Page 77 of 1-100.

3981

3984

3985 3986

3987

3991

3992 3993

3994

3995 3996

3997

3998

4000

 $\mathcal{E}^{T} \llbracket \tau' \rrbracket$.

4005 4006

```
4007
                                       By Lemma 5.18, it is sufficient to show that (k-1, v_1, v_2) \in \mathcal{E}^T \llbracket \tau' \rrbracket.
4008
                                       By Lemma 5.12, it is sufficient to show that (k, v_1, v_2) \in \mathcal{E}^T \llbracket \tau \rrbracket, which is our assumption.
4009
4010
                   Tags do not match Inspection of the operational semantics shows that both terms step to a boundary error, and so
4011
                                       are trivially in the relation.
4012
4013
4014
                         Lemma 5.21 (Boundary Compatibility—open relation). If \llbracket \Gamma \vdash_{\text{tru}} e_1 \leq e_2 : \tau \rrbracket_C^T and \tau' = K_1' \sqcap K_1 \sqcap \tau = K_2' \sqcap K_2 \sqcap \tau,
                   then [\Gamma \vdash_{\mathsf{tru}} \mathsf{cast} \{K_1' \Leftarrow K_1'\} e_1 \leq \mathsf{cast} \{K_2' \Leftarrow K_2\} e_1 : \tau']
4016
4017
                         PROOF. Consider arbitrary (k, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma \rrbracket.
4018
                         We must show that (k, \gamma(\text{cast }\{K_1' \Leftarrow K_1\} e_1), \gamma'(\text{cast }\{K_2' \Leftarrow K_2\} e_2)) \in \mathcal{E}^T[[\tau']].
4019
                          By the definition of substitution, it suffices to show that (k, \text{cast } \{K_1' \Leftarrow K_1\} \ \gamma(e_1), \text{cast } \{K_2' \Leftarrow K_2\} \ \gamma'(e_2)) \in \mathcal{E}^T[\![\tau']\!].
4020
4021
                         Instantiate the hypothesis with (k, \gamma, \gamma'), providing that (k, \gamma(e_1), \gamma'(e_2)) \in \mathcal{E}^T \llbracket \tau \rrbracket.
4022
                         Then Lemma 5.14 applies. Consider arbitrary (k', v_1, v_2) s.t. (k', v_1, v_2) \in \mathcal{V}^T[\![\tau]\!]; we must show that (k', \text{cast } \{K'_1 \Leftarrow (v_1, v_2) \in \mathcal{V}^T[\![\tau]\!]\}
4023
                   K_1} v_1, cast \{K_2' \Leftarrow K_2\} v_2) \in \mathcal{E}^T[\![\tau]\!]. This is immediate by Lemma 5.20 and Lemma 5.13.
4024
4025
                          \text{Lemma 5.22 (Application compatibility)}. \  \, \textit{If } (k, v_f, v_f') \in \mathcal{V}^T \llbracket * \rightarrow \tau_2 \rrbracket \  \, \textit{and } (k, v_a, v_a') \in \mathcal{V}^T \llbracket \tau_1 \rrbracket \  \, \textit{and } \tau' = K \sqcap \tau_2 = 1 \text{ and } \tau' = 1 \text{ an
4026
                  K' \sqcap \tau_2, then (k, \operatorname{app}\{K\} v_f v_a, \operatorname{app}\{K'\} v_f' v_a') \in \mathcal{E}^T \llbracket \tau' \rrbracket
4027
                          PROOF. Unfolding the V relation on our first assumption and instantiating with j = k, v'_1 = v_a, v'_2 = v'_a, K = K,
4029
4030
                   K' = K' gives precisely what is to be shown.
4031
                         Lemma 5.23 (Application Compatibility—open relation). If \llbracket \Gamma \vdash_{\mathsf{tru}} e_{f1} \leq e_{f2} : * \rightarrow \tau_2 \rrbracket_C^T and \tau' = K_1 \sqcap \tau_2 = K_2 \sqcap \tau_2
4032
4033
                   and \llbracket \Gamma \vdash_{\mathsf{tru}} e_{a1} \leq e_{a2} : \tau_1 \rrbracket_C^T, then \llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{app}\{K_1\} e_{f1} e_{a1} \leq \mathsf{app}\{K_2\} e_{f2} e_{a2} : \tau' \rrbracket_C^T.
4034
4035
                         PROOF. Consider arbitrary (k, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma \rrbracket.
4036
                         We must show that (k, \gamma(app\{K_1\} e_{f_1} e_{a1}), \gamma'(app\{K_2\} e_{f_2} e_{a2})) \in \mathcal{E}^T [\![\tau']\!].
4037
                          By the definition of substitution, it suffices to show that (k, \operatorname{app}\{K_1\} \gamma(e_{f1}) \ \gamma(e_{a1}), \operatorname{app}\{K_2\} \gamma'(e_{f2}) \ \gamma'(e_{a2})) \in \mathbb{R}
4038
                   \mathcal{E}^T \llbracket \tau' \rrbracket.
4039
                          Instantiate the first hypothesis with (k, \gamma, \gamma'), providing (k, \gamma(e_{f1}), \gamma'(e_{f2})) \in \mathcal{E}^T[\![* \to \tau_2]\!]. Similarly, the second
                   provides (k, \gamma(e_{a1}), \gamma'(e_{a2})) \in \mathcal{E}^T \llbracket \tau_1 \rrbracket.
                         Then Lemma 5.14 applies. Consider arbitrary (k', v_{f1}, v_{f2}) \in \mathcal{V}^T[\![* \to \tau_2]\!] with k' \le k. Then by Lemma 5.13,
                   (k', \gamma(e_{a1}), \gamma'(e_{a2})) \in \mathcal{E}^T \llbracket \tau_1 \rrbracket, Lemma 5.14 again applies. Consider arbitrary (k'', v_{a1}, v_{a2} \in \mathcal{V}^T \llbracket \tau_1 \rrbracket) with k'' \leq k'.
4044
                  We must show that (k'', \operatorname{app}\{K_1\} v_{f1} v_{a1}, \operatorname{app}\{K_2\} v_{f2} v_{a2}) \in \mathcal{E}^T[[\tau']]; this is immmediate by Lemma 5.22.
4045
4046
                         Lemma 5.24 (Application compatibility-function is bottom). If [\Gamma \vdash_{\mathsf{tru}} e_{f1} \leq e_{f2} : \bot]_{C}^{T} then [\Gamma \vdash_{\mathsf{tru}} e_{f1} \leq e_{f2} : \bot]_{C}^{T}
4047
4048
                   \operatorname{app}\{K_1\} e_{f_1} e_{a_1} \leq \operatorname{app}\{K_2\} e_{f_2} e_{a_2} : \bot \}_C^T
4049
                          PROOF. Consider arbitrary (k, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma \rrbracket.
4050
4051
                         We must show that (k, \gamma(app\{K_1\} e_{f_1} e_{a1}), \gamma'(app\{K_2\} e_{f_2} e_{a2})) \in \mathcal{E}^T [[\tau']].
4052
                          By the definition of substitution, it suffices to show that (k, \operatorname{app}\{K_1\} \gamma(e_{f1}) \ \gamma(e_{a1}), \operatorname{app}\{K_2\} \gamma'(e_{f2}) \ \gamma'(e_{a2})) \in \mathbb{R}
4053
                   \mathcal{E}^T \llbracket \tau' \rrbracket.
4054
                          Instantiate the first hypothesis with (k, \gamma, \gamma'), providing (k, \gamma(e_{f1}), \gamma'(e_{f2})) \in \mathcal{E}^T[\![\bot]\!].
4056
                                                                                                                                                                                                                                                               2022-11-18 03:01. Page 78 of 1-100.
```

Tags match By the operational semantics, it is sufficient to show that $(k, mon \{K'_1 \Leftarrow K_1\} v_1, mon \{K'_2 \Leftarrow K_2\} v_2) \in$

 Then Lemma 5.14 applies. Consider arbitrary $(k', v_{f1}, v_{f2}) \in \mathcal{V}^T[\![\bot]\!]$ with $k' \leq k$. By unfolding of \mathcal{V} no such values can exist, so we are done.

Lemma 5.25 (FST compatibility). If $(k, v, v') \in \mathcal{V}^T \llbracket \tau_1 \times \tau_2 \rrbracket$ and $\tau' = K \sqcap \tau_1 = K' \sqcap \tau_1$, then $(k, \text{fst}\{K\} v, \text{fst}\{K'\} v') \in \mathcal{E}^T \llbracket \tau' \rrbracket$.

PROOF. Unfolding the definition of \mathcal{V} tells us that there must be some v_1, v_2, v'_1, v'_2 s.t. $v = \langle v_1, v_2 \rangle$, $v' = \langle v'_1, v'_2 \rangle$, $(k, v_1, v'_1) \in \mathcal{V}^T \llbracket \tau_1 \rrbracket$, and $(k, v_2, v'_2) \in \mathcal{V}^T \llbracket \tau_2 \rrbracket$. We must show that $(k, \operatorname{fst}\{K\} \langle v_1, v_2 \rangle, \operatorname{fst}\{K'\} \langle v'_1, v'_2 \rangle) \in \mathcal{E}^T \llbracket \tau' \rrbracket$. By the OS, it suffices to show that $(k - 1, \operatorname{assert} K v_1, \operatorname{assert} K' v'_1) \in \mathcal{E}^T \llbracket \tau' \rrbracket$.

By Lemma 5.15, it suffices to show that $(k-1, v_1, v_1') \in \mathcal{E}^T[\![\tau_1]\!]$. This is immediate by Lemma 5.13.

Lemma 5.26 (Fst compatibility—open relation). If $\llbracket \Gamma \vdash_{\mathsf{tru}} e \leq e' : \tau_1 \times \tau_2 \rrbracket_C^T$ and $\tau' = K \sqcap \tau_1 = K' \sqcap \tau_1$, then $\llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{fst}\{K\} e \leq \mathsf{fst}\{K'\} e' : \tau' \rrbracket_C^T$.

PROOF. Consider arbitrary $(k, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma \rrbracket$.

We must show that $(k, \gamma(\text{fst}\{K\} e), \gamma'(\text{fst}\{K'\} e')) \in \mathcal{E}^T \llbracket \tau' \rrbracket$.

By the definition of substitution, it suffices to show that $(k, \text{fst}\{K\} \gamma(e), \text{fst}\{K'\} \gamma'(e')) \in \mathcal{E}^T \llbracket \tau' \rrbracket$.

Instantiate the hypothesis with (k, γ, γ') , providing $(k, \gamma(e), \gamma'(e')) \in \mathcal{E}^T \llbracket \tau_1 \times \tau_2 \rrbracket$.

Then Lemma 5.14 applies. Consider arbitrary $(k', v, v') \in \mathcal{V}^T \llbracket \tau_1 \times \tau_2 \rrbracket$. We must show that $(k', \operatorname{fst}\{K\} v, \operatorname{fst}\{K'\} v') \in \mathcal{E}^T \llbracket \tau' \rrbracket$; this is immediate by Lemma 5.25.

Lemma 5.27 (Fst compatibility-pair is bottom). If $\llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \leq e_2 : \bot \rrbracket_C^T$ then $\llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{fst}\{K_1\} e_1 \leq \mathsf{fst}\{K_2\} e_2 : \bot \rrbracket_C^T$.

PROOF. By the same reasoning as Lemma 5.24.

LEMMA 5.28 (SND COMPATIBILITY).

PROOF. Nearly identical to that of Lemma 5.25.

Lemma 5.29 (FST compatibility—open relation). If $\llbracket \Gamma \vdash_{\text{tru}} e \leq e' : \tau_1 \times \tau_2 \rrbracket_C^T$ and $\tau' = K \sqcap \tau_2 = K' \sqcap \tau_2$, then $\llbracket \Gamma \vdash_{\text{tru}} \text{snd}\{K\} e \leq \text{snd}\{K'\} e' : \tau \rrbracket_C^T$.

PROOF. Nearly identical to that of Lemma 5.26, using Lemma 5.28.

Lemma 5.30 (Snd compatibility-pair is bottom). If $\llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \leq e_2 : \bot \rrbracket_C^T$ then $\llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{snd}\{K_1\} e_1 \leq \mathsf{snd}\{K_2\} e_2 : \bot \rrbracket_C^T$.

PROOF. By the same reasoning as Lemma 5.24.

5.4.3 Binary relation: Compatibility Lemmata

Lemma 5.31 (T-Var compatibility).
$$\frac{(x\!:\!K) \in \Gamma}{ \left[\!\!\left[\Gamma \vdash_{\mathsf{tru}} x \leq x \!:\! K\right]\!\!\right]_{\!\!\!\!-}^{\mathcal{L}}}$$

PROOF. Consider arbitrary $(k, \gamma, \gamma') \in \mathcal{G}^{\mathcal{L}} \llbracket \Gamma \rrbracket$.

We must show that $(k, \gamma(x), \gamma'(x)) \in \mathcal{E}^{\mathcal{L}} \llbracket K \rrbracket$.

Since $x: K \in \Gamma$, we know that there exist some values v, v' s.t. $\gamma(x) = v$ and $\gamma'(x) = v'$. Since $(k, \gamma, \gamma') \in \mathcal{G}^{\mathcal{L}}[\![\Gamma]\!]$, we know that $(k, v, v') \in \mathcal{V}^{\mathcal{L}}[\![K]\!]$. Then we get $(k, v, v') \in \mathcal{E}^{\mathcal{L}}[\![\Gamma]\!]$ immediately since v, v' are already values.

```
Lemma 5.32 (T-Nat compatibility). \frac{}{\left[\!\!\left[\Gamma \vdash_{\mathsf{tru}} n \leq n : \mathsf{Nat}\right]\!\!\right]_C^{\mathcal{L}}}
4109
4110
4111
                  Proof. Consider arbitrary (k, \gamma, \gamma') \in \mathcal{G}^{\mathcal{L}} \llbracket \Gamma \rrbracket.
4112
                  We must show (k, \gamma(n), \gamma'(n)) \in \mathcal{E}^{\mathcal{L}}[[Nat]].
4113
4114
                  Note that y(n) = n.
4115
                  Since n is already a value, it suffices to show that (k, n, n) \in \mathcal{V}^{\mathcal{L}}[\![ \text{Nat} ]\!].
4116
                  Unfolding the definition of V^{\mathcal{L}}[Nat], this is true.
                                                                                                                                                                                                                                           4117
                  Lemma 5.33 (T-Int compatibility). \frac{}{ \left[\!\!\left[\Gamma \vdash_{\mathsf{tru}} i \leq i : \mathsf{Int}\right]\!\!\right]_C^{\mathcal{L}}}
4120
4121
                  PROOF. Consider arbitrary (k, \gamma, \gamma') \in \mathcal{G}^{\mathcal{L}} \llbracket \Gamma \rrbracket.
4122
                  We must show (k, \gamma(i), \gamma'(i)) \in \mathcal{E}^{\mathcal{L}}[[Nat]].
4123
                  Note that y(i) = i.
4124
                  Since i is already a value, it suffices to show that (k, i, i) \in \mathcal{V}^{\mathcal{L}}[[Int]].
4125
4126
                  Unfolding the definition of \mathcal{V}^{\mathcal{L}}[\![ Nat ]\!], this is true.
                                                                                                                                                                                                                                           4127
                  Lemma 5.34 (T-True compatibility). \frac{}{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{True} \leq \mathsf{True} : \mathsf{Bool} \rrbracket_C^{\mathcal{L}}}
4128
4129
4130
                  PROOF. Consider arbitrary (k, \gamma, \gamma') \in \mathcal{G}^{\mathcal{L}} \llbracket \Gamma \rrbracket.
4131
                  We must show (k, \gamma(\mathsf{True}), \gamma'(\mathsf{True})) \in \mathcal{E}^{\mathcal{L}}[\![\mathsf{Bool}]\!].
4132
4133
                  Note that \gamma(\mathsf{True}) = \mathsf{True}.
4134
                   Since True is already a value, it suffices to show that (k, \text{True}, \text{True}) \in \mathcal{V}^{\mathcal{L}}[[\text{Bool}]]
4135
                  Unfolding the definition of \mathcal{V}^{\mathcal{L}}[Bool], this is true.
                                                                                                                                                                                                                                           4136
                  Lemma 5.35 (T-False compatibility). \frac{}{ \left[\!\!\left[ \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{False} \leq \mathsf{False} : \mathsf{Bool} \right]\!\!\right]_C^{\mathcal{L}}}
4137
4138
4139
4140
                   PROOF. Consider arbitrary (k, \gamma, \gamma') \in \mathcal{G}^{\mathcal{L}} \llbracket \Gamma \rrbracket.
4141
                  We must show (k, \gamma(\text{False}), \gamma'(\text{False})) \in \mathcal{E}^{\mathcal{L}} \llbracket \text{Bool} \rrbracket.
4142
                  Note that \gamma(False) = False.
4143
                  Since False is already a value, it suffices to show that (k, \mathsf{False}, \mathsf{False}) \in \mathcal{V}^{\mathcal{L}}[\![\mathsf{Bool}]\!]
                  Unfolding the definition of \mathcal{V}^{\mathcal{L}}[Bool], this is true.
                                                                                                                                                                                                                                           4146
                   \text{Lemma 5.36 (T-Lam compatibility)}. \quad \frac{ \left[\!\!\left[\Gamma_0,\; (x_0\!:\!K_0) \vdash_{\mathsf{tru}} e_0 \leq e_0' : \tau_1\right]\!\!\right]_C^{\mathcal{L}}}{ \left[\!\!\left[\Gamma_0 \vdash_{\mathsf{tru}} \lambda(x_0\!:\!K_0). \, e_0 \leq \lambda(x_0\!:\!K_0). \, e_0' : * \to \tau_1\right]\!\!\right]_C^{\mathcal{L}}} 
4147
4148
4149
4150
                  PROOF. Let (k, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma_0 \rrbracket.
4151
             We want to show (k, \gamma(\lambda x_0 : K_0. e_0), \gamma'(\lambda x_0. K_0 e_0')) \in \mathcal{E}^T [\![ * \to \tau_1 ]\!].
4152
              Note that \gamma(\lambda x_0 : K_0. e_0) = \lambda x_0 : K_0. \gamma(e_0) and similarly for the other.
4153
             We want to show (k-1, \lambda x_0 : K_0, \gamma(e_0), \lambda x_0 : K_0, \gamma(e'_0)) \in \mathcal{V}^T[[* \to \tau_1]].
4154
4155
             Unfolding the value relation:
4156
             Let j \leq k.
4157
             Let (j, v, v') \in \mathcal{V}^T \llbracket * \rrbracket.
4158
             Let K.
4159
```

We want to show $(j, \operatorname{app}\{K\} (\lambda x_0 : K_0, \gamma(e_0)) v, \operatorname{app}\{K\} (\lambda x_0 : K_0, \gamma(e_0')) v') \in \mathcal{E}^T \llbracket \tau_1 \sqcap K \rrbracket$.

By the OS, if $\neg K \sim v$ then the application steps to an error and we're done.

Otherwise, app $\{K\}$ ($\lambda x_0 : K_0, \gamma(e_0)$) $v \longrightarrow_T$ assert $K((\lambda x_0 : K_0, \gamma(e_0)) v) \longrightarrow$ assert $K(e_0)[v/x]$.

By the definition of substitution, $\gamma(e_0)[v/x] = \gamma[x \mapsto v](e_0)$.

 Note that $(j-2,\gamma[x\mapsto v](e_0),\gamma'[x\mapsto v](e_0'))\in \mathcal{G}^T[\Gamma,x:K]$ by Lemma 4.55 and Lemma 4.57.

Therefore, we can apply the hypothesis to $\gamma[x \mapsto v]$, $\gamma'[x \mapsto v']$, and e_0, e'_0 at j-2 to get $(j-2, \gamma[x \mapsto v](e_0), \gamma'[x \mapsto v']e'_0) \in \mathcal{E}^T[\tau_1]$.

Finally, we can apply Lemma 4.58 to get $(j-1, \operatorname{assert} K \gamma[x \mapsto v](e_0), \operatorname{assert} K \gamma'[x \mapsto v'](e_0')) \in \mathcal{E}^T[[\tau_1 \sqcap K]]$ which is what we wanted to show.

$$\text{Lemma 5.37 (T-Pair compatibility)}. \quad \frac{ \begin{bmatrix} \Gamma \vdash_{\mathsf{tru}} e_1 \leq e_1' : \tau_1 \end{bmatrix}_C^T}{ \begin{bmatrix} \Gamma \vdash_{\mathsf{tru}} e_2 \leq e_2' : \tau_2 \end{bmatrix}_C^T} \\ \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} e_2 \leq e_2' : \tau_2 \rrbracket_C^T}{ \llbracket \Gamma \vdash_{\mathsf{tru}} \langle e_1, e_2 \rangle \leq \langle e_1', e_2' \rangle : \tau_1 \times \tau_2 \rrbracket_C^T}$$

PROOF. Consider arbitrary $(k, \gamma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket$.

We must show $(k, \gamma(\langle e_1, e_2 \rangle), \gamma'(\langle e_1', e_2' \rangle)) \in \mathcal{E}^T \llbracket \tau_1 \times \tau_2 \rrbracket$.

Note that $\gamma(\langle e_1, e_2 \rangle) = \langle \gamma(e_1), \gamma(e_2) \rangle$, and similarly for γ', e_1', e_2' . We want to show that $(k, \langle \gamma(e_1), \gamma(e_2) \rangle, \langle \gamma'(e_1'), \gamma'(e_2') \rangle) \in \mathcal{E}^T [\tau_1 \times \tau_2]$.

Notice that by instantiating our hypothesis with (k, γ, γ') , we know that $(k, \gamma(e_1), \gamma'(e_1')) \in \mathcal{E}^T[[\tau_1]]$ and $(k, \gamma(e_2), \gamma'(e_2')) \in \mathcal{E}^T[[\tau_2]]$.

By Lemma 5.14, it suffices to show that for any $(k', v_1, v_1') \in \mathcal{V}^T[\![\tau_1]\!]$ where $k' \leq k$, $(k', \langle v_1, e_2 \rangle, \langle v_1', e_2' \rangle) \in \mathcal{E}^T[\![\tau_1 \times \tau_2]\!]$.

By Lemma 5.13, we know that $(k', \gamma(e_2), \gamma'(e_2')) \in \mathcal{E}^T[\tau_2]$. Again by Lemma 5.14, therefore, it suffices to show that for any $k'' \leq k'$ and v_2, v_2' s.t. $(k'', v_2, v_2') \in \mathcal{V}^T[\tau_2]$, $(k'', \langle v_1, v_2 \rangle, \langle v_1', v_2' \rangle) \in \mathcal{E}^T[[\tau_1 \times tau_2]]$.

Since these terms are values, it suffices to show that $(k'', \langle v_1, v_2 \rangle, \langle v_1', v_2' \rangle) \in \mathcal{V}^T \llbracket \tau_1 \times \tau_2 \rrbracket$.

Unfolding the definition of \mathcal{V} , it suffices to show that $(k'', v_1, v_1') \in \mathcal{V}^T[[\tau_1]]$ and $(k'', v_2, v_2') \in \mathcal{V}^T[[\tau_2]]$; both of these are immediate by Lemma 5.13 from our assumptions.

PROOF. Follows immediately from Lemma 5.21.

PROOF. Follows immediately from Lemma 5.23.

PROOF. Consider arbitrary $(k, \gamma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket$.

We must show $(k, \gamma(app\{K_1\} e_0 e_1), \gamma'(app\{K_1\} e_0' e_1')) \in \mathcal{E}^T[\![\bot]\!]$.

Apply the first hypothesis to get $(k, \gamma(e_0), \gamma'(e'_0)) \in \mathcal{E}^T \llbracket \bot \rrbracket$.

2022-11-18 03:01. Page 81 of 1-100.

 Note $\gamma(binop e_0 e_1) = binop \gamma(e_0) \gamma(e_1)$, and similarly for e'_0, e'_1 .

```
Unfolding, there exists some j \le k, e_2, e_3 such that \gamma(e_0) \longrightarrow_T^j e_2 and \gamma'(e_0') \longrightarrow_T^j e_3 where e_2 and e_3 are irreducible.
4213
4214
                 Either e_2 = e_3 \in \operatorname{Err}^{\bullet}, or (j, e_2, e_3) \in \mathcal{V}^T \llbracket \bot \rrbracket.
4215
                 By inversion, it must be the case that e_2 = e_3 \in \operatorname{Err}^{\bullet}, which means that by the OS, \gamma(\operatorname{app}\{K_1\} e_0 \ e_1 \longrightarrow_T^{j+1} e_2 and
4216
             \gamma'(\operatorname{app}\{K_1\} e_0' e_1') \longrightarrow_T^{j+1} e_3.
4217
4218
                 Then either, j + 1 > k, in which case we're done, and otherwise both applications step to the same error within k
4219
             steps, in which case we're done.
4220
                 \text{Lemma 5.41 (T-Fst compatibility)}. \quad \frac{ \left[\!\!\left[\Gamma_0 \vdash_{\mathsf{tru}} e_0 \leq e_0' : \tau_0 \!\times\! \tau_1\right]\!\!\right]_C^T}{ \left[\!\!\left[\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{fst}\{K_0\} e_0 \leq \mathsf{fst}\{K_0\} e_0' : K_0 \sqcap \tau_0\right]\!\!\right]_C^T}
4221
4222
4224
                 PROOF. Consider arbitrary (k, \gamma, \gamma) \in \mathcal{G}^T \llbracket \Gamma \rrbracket.
4225
                 We must show (k, \gamma(\operatorname{fst}\{K_0\} e_0), \gamma'(\operatorname{fst}\{K_1\} e_0')) \in \mathcal{E}^T \llbracket K_0 \sqcap \tau_0 \rrbracket.
4226
                 Note that \gamma(\text{fst}\{K_0\} e_0) = \text{fst}\{K_0\} \gamma(e_0) and similarly for e_0'.
4227
4228
                 Assume that there are j \leq k, e_1 such that fst\{K_0\} e_0 \longrightarrow_T^J e_1 and e_1 is irreducible.
4229
                 By the OS, it must be the case that there are irreducible e_1', e_1'' such that \mathsf{fst}\{K_0\}\ e_0 \longrightarrow^{j-2} \mathsf{fst}\{K_0\}\ e_1' \longrightarrow \mathsf{assert}\ K_0\ e_1'' \longrightarrow \mathsf{assert}\ K_0 = \mathsf{e}_1''
4230
4231
                 Unfolding our hypothesis and applying it to the reduction e_0 \longrightarrow^{j-2} e'_1, we get that there is an irreducible e'_2 such
4232
4233
             that e'_0 \longrightarrow_T^* e'_2 and (k - j + 2, e'_1, e'_2) \in \mathcal{V}^T [\![ \tau_0 \times \tau_1 ]\!].
4234
                 Unfolding the value relation, we get that both e'_1 and e'_2 are pairs.
4235
                 Therefore, we have by the OS that there exists e_2'', e_2 such that fst\{K_0\} e_0' \longrightarrow_T^* fst\{K_0\} e_2' \longrightarrow_T assert K_0 e_2'' \longrightarrow_T e_2.
                 Unfolding the fact that (k-j+2,e_1',e_2') \in \mathcal{V}^T[\![\tau_0 \times \tau_1]\!] gives us that (k-j+2,e_1'',e_2'') \in \hat{\mathcal{V}}^T[\![\tau_0]\!].
4237
                 Finally, by Lemma 5.15, we get that (k-j+2, \operatorname{assert} K_0 e_1'', \operatorname{assert} K_0 e_2'') \in \mathcal{E}^T[\![\tau_0 \sqcap K_0]\!], which is sufficient to
4238
4239
             complete the proof.
4240
                 Lemma 5.42 (T-FstBot compatibility). \frac{\llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \leq e_0' : \bot \rrbracket_C^T}{\llbracket \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{fst}\{K_0\} \, e_0 \leq \mathsf{fst}\{K_0\} \, e_0' : \bot \rrbracket_C^T}
4241
4242
4243
4244
                 PROOF. Similar reasoning to T-AppBot.
                                                                                                                                                                                                                           4245
                  \text{Lemma 5.43 (T-Snd compatibility)}. \quad \frac{ \left[\!\!\left[\Gamma_0 \vdash_{\mathsf{tru}} e_0 \leq e_0' : \tau_0 \!\times\! \tau_1\right]\!\!\right]_C^T}{ \left[\!\!\left[\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{snd}\{K_1\} e_0' \leq \mathsf{snd}\{K_1\} e_0' : K_1 \sqcap \tau_1\right]\!\!\right]_C^T} 
4246
4247
                 PROOF. Almost identical to T-FsT.
                                                                                                                                                                                                                           Lemma 5.44 (T-SndBot compatibility). \frac{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \leq e_0' : \bot \rrbracket_C^T}{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{snd}\{K_1\} \, e_0' \leq \mathsf{snd}\{K_1\} \, e_0' : \bot \rrbracket_C^T}
4251
4252
4253
4254
                 PROOF. Similar reasoning to T-APPBOT.
                                                                                                                                                                                                                           4255
                4256
4257
4258
4259
4260
                 PROOF. Let (k, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma \rrbracket.
4261
             We want to show (k, \gamma(binop e_0 e_1), \gamma(binop e_0' e_1')) \in \mathcal{E}^T \llbracket \Delta(binop \tau_0, \tau_1) \rrbracket.
```

2022-11-18 03:01. Page 83 of 1-100.

```
By the first hypothesis applied to \gamma, \gamma' we have (k, \gamma(e_0), \gamma'(e_0')) \in \mathcal{E}^T \llbracket \tau_0 \rrbracket
4265
4266
           Unfolding we get there is a j \leq k, and irreducible e_2, e_2' such that \gamma(e_0) \longrightarrow_T^j e_2 and \gamma'(e_0') \longrightarrow_T^* e_2'.
4267
           If e_2 = e_2' = \text{Err}^{\bullet} then we're done, because the whole operation errors.
4268
           Otherwise (k - j, e_2, e'_2) \in \mathcal{V}^T [\![\tau_0]\!].
4269
4270
4271
          Note by Lemma 5.13 (k - j, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma_1 \rrbracket.
4272
4273
          By the second hypothesis applied to \gamma, \gamma' and k-j, we have (k-j,\gamma(e_1),\gamma'(e_1')) \in \mathcal{E}^T[\![\tau_1]\!].
4274
           Unfolding we get there are j', and irreducible e_3, e_3' such that \gamma(e_1) \longrightarrow_T^{j'} e_3 and \gamma'(e_1') \longrightarrow_T^* e_3'.
           If e_3 = e_3' = \text{Err}^{\bullet} then we're done, because the whole operation errors.
4276
           Otherwise (k - j - j', e_3, e_3') \in \mathcal{V}^T [\![\tau_1]\!].
4277
4278
4279
4280
          From the definition of \Delta, K_2 = \text{Int or Nat or } \perp.
4281
          In the case of \bot, we're done because either \tau_0 or \tau_1 is a \bot, which is a contradiction.
4282
           Otherwise, the cases proceed identically, so without loss of generality assume K_2 = Int.
4283
           \tau_0 = \tau_1 = \text{Int}, and therefore e_2 = e_2' = i_0 and e_3 = e_3' = i_1.
4284
4285
           If binop = quotient and i_1 = 0 then binop i_0 i_1 \longrightarrow_T DivErr, so we're done.
4286
          If binop = quotient and i_1 \neq 0, then binop i_0 i_1 \longrightarrow_T (i_0/i_1).
4287
          Since i_0/i_1 \in \mathbb{Z}, we're done.
4288
          If binop = sum then binop i_0 i_1 \longrightarrow_T i_0 + i_1.
4289
4290
          Since i_0 + i_1 \in \mathbb{Z}, we're done.
                                                                                                                                                                                         4291
                                                                                                \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \leq e'_0 : \mathsf{Bool} \rrbracket_C^T
4292
                                                                                               4293
4294
                                                                    LEMMA 5.46 (T-IF COMPATIBILITY).
4296
4297
              PROOF. Let (k, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma \rrbracket.
4298
          We want to show (k, \gamma(\text{if } e_0 \text{ then } e_1 \text{ else } e_2), \gamma'(\text{if } e_0' \text{ then } e_1' \text{ else } e_2')) \in \mathcal{E}^T[\![\tau_0 \sqcup \tau_1]\!].
4299
           Note \gamma(\text{if }e_0 \text{ then }e_1 \text{ else }e_2) = \text{if } \gamma(e_0) \text{ then } \gamma(e_1) \text{ else } \gamma(e_2) \text{ and similarly for }e'_0,e'_1,e_2.
4300
           From the first hypothesis applied to \gamma, \gamma', we know (k, \gamma(e_0), \gamma'(e_0')) \in \mathcal{E}^T \llbracket \mathsf{Bool} \rrbracket.
4302
           Unfolding, we have that there is a j \leq k and irreducible e_4, e'_4 such that e_0 \longrightarrow_T^j e_4 and e'_0 \longrightarrow_T^* e'_4.
4303
           If e_4, e'_4 \in \text{Err}^{\bullet} then we're done, because the entire if statement errors.
4304
           Otherwise, (k - j, e_4, e'_4) \in \mathcal{V}^T \llbracket \text{Bool} \rrbracket.
4305
           Unfolding the location and then the value relation, we get that e_4 = e'_4 = \text{True or } e_4 = e'_4 = \text{False}.
4306
4307
4308
                  • e_4 = e_4' = \text{True}: Note by OS, if \gamma(e_0) then \gamma(e_1) else \gamma(e_2) \longrightarrow_T^j if e_4 then \gamma(e_1) else \gamma(e_2) \longrightarrow_T \gamma(e_1), and
4309
                      similarly for if \gamma'(e'_0) then \gamma'(e'_1) else \gamma(e'_2).
4310
4311
                      By Lemma 5.13, we have (k - j - 1, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma_1 \rrbracket.
4312
                      From the second hypothesis, we get (k - j - 1, \gamma(e_1), \gamma'(e_1')) \in \mathcal{E}^T \llbracket \tau_0 \rrbracket
4313
                      Finally, by Lemma 4.61, we get (k-j-1,\gamma(e_1),\gamma'(e_1')) \in \mathcal{E}^T \llbracket \tau_0 \sqcup \tau_1 \rrbracket which is sufficient to complete the proof.
4314
4315
                  • e_4 = e'_4 = False: same as other case except replace e_1 with e_2.
```

 $\llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \leq e_0' : \bot \rrbracket_C^T$ PROOF. Similar reasoning to T-APPBOT. $\llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \le e_0' : \tau_0 \rrbracket_C^T$ Lemma 5.48 (T-Sub compatibility). $\frac{\tau_0 \leqslant : \tau_1}{\left\|\Gamma_0 \vdash_{\mathsf{tru}} e_0 \leq e_0' : \tau_1\right\|_C^T}$ PROOF. Follows directly from Lemma 5.17. **Binary relation: Fundamental Property** Theorem 5.49 (Binary relation is reflexive). If $\Gamma \vdash_{\mathsf{tru}} e : \tau \text{ then } \llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e : \tau \rrbracket_{C}^{T}$ PROOF. By induction over the typing derivation, using the compatibility lemmata. **Context relation—Proofs Context relation: Compatibility Lemmata**

 $\text{Lemma 5.50 (T-CTX-Hole compatibility)}. \quad \frac{\Gamma' \subseteq \Gamma}{ \llbracket \Gamma \vdash_{\mathsf{tru}} [\] \approx [\] : (\Gamma' \triangleright \tau) \leadsto \tau \rrbracket_C^T }$

PROOF. Let e, e' such that $\llbracket \Gamma' \vdash_{\mathsf{tru}} e \approx e' : \tau \rrbracket$.

We want to show $\llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : \tau \rrbracket$.

 Note $\forall (k, \gamma, \gamma') \in \mathcal{G}^T \llbracket \Gamma \rrbracket, (k, \gamma|_{dom(\Gamma')}, \gamma'|_{dom(\Gamma')}) \in \mathcal{G}^T \llbracket \Gamma' \rrbracket.$

And note $\gamma(e) = \gamma|_{dom(\Gamma')}(e)$ and similarly for e'.

Then given such k, γ, γ' , we can apply the hypothesis to get that $(k, \gamma(e), \gamma'(e')) \in \mathcal{E}^T[\![\tau]\!]$, which is sufficient to complete the proof.

$$\text{Lemma 5.51 (T-CTX-Lam compatibility)}. \quad \frac{ \llbracket \Gamma, \ (x:K) \vdash_{\mathsf{tru}} E \approx E' : (\Gamma', \ (x:K) \trianglerighteq \tau) \leadsto \tau' \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \lambda(x:K) . E \approx \lambda(x:K) . E' : (\Gamma', \ (x:K) \trianglerighteq \tau) \leadsto * \to \tau' \rrbracket_C^T }$$

PROOF. Let e, e' such that $\llbracket \Gamma', (x:K) \vdash_{\mathsf{tru}} e \approx e' : \tau \rrbracket$.

We want to show $\llbracket \Gamma \vdash_{\mathsf{tru}} \lambda(x:K). e \approx \lambda(x:K). e' : * \rightarrow \tau' \rrbracket$.

From our hypothesis we get $[\Gamma', (x:K) \vdash_{\mathsf{tru}} E[e] \approx E[e'] : \tau']$.

Then the case follows from Lemma 5.36.

$$\text{Lemma 5.52 (T-CTX-Pair-1 compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma' \triangleright \tau) \leadsto \tau_1 \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \langle E, e \rangle \approx \langle E', e' \rangle : (\Gamma' \triangleright \tau) \leadsto \tau_1 \times \tau_2 \rrbracket_C^T }$$

PROOF. Let e, e' such that $\llbracket \Gamma' \vdash_{\mathsf{tru}} e_1 \approx e'_1 : \tau \rrbracket$.

We want to show $\llbracket \Gamma' \vdash_{\mathsf{tru}} \langle E[e_1], e \rangle \approx \langle E'[e'_1], e \rangle : \tau_1 \times \tau_2 \rrbracket$.

From our first hypothesis, we have $\llbracket \Gamma' \vdash_{\mathsf{tru}} E[e_1] \approx E'[e'_1] : \tau_1 \rrbracket$.

Then the case follows by Lemma 5.37.

$$\text{Lemma 5.53 (T-Ctx-Pair-2 compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : \tau_1 \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma' \triangleright \tau) \leadsto \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \langle e, E \rangle \approx \langle e', E' \rangle : (\Gamma' \triangleright \tau) \leadsto \tau_1 \times \tau_2 \rrbracket_C^T }$$

Proof. Analogous to T-CTX-PAIR-1.

PROOF. Let e, e' such that $\llbracket \Gamma' \vdash_{\mathsf{tru}} e_1 \approx e'_1 : * \to \tau_1 \rrbracket$.

We want to show $\llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{app}\{K\} E[e_1] e \approx \mathsf{app}\{K\} E'[e_1'] e' : K \sqcap \tau_1 \rrbracket$

By the first hypothesis, we have $\llbracket \Gamma \vdash_{\mathsf{tru}} E[e_1] \approx E'[e'_1] : * \to \tau_1 \rrbracket$.

Then the case follows by Lemma 5.22.

$$\text{Lemma 5.55 (T-CTX-AppBot-1 compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma' \triangleright \tau) \leadsto \bot \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{app}\{K\} \, E \, e \approx \mathsf{app}\{K\} \, E' \, e' : (\Gamma' \triangleright \tau) \leadsto \bot \rrbracket_C^T }$$

PROOF. Analogous to T-CTX-APP-1.

$$\text{Lemma 5.56 (T-Ctx-App-2 compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : * \to \tau_1 \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma' \triangleright \tau) \leadsto \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{app}\{K\} \ e \ E \approx \mathsf{app}\{K\} \ e' \ E' : (\Gamma' \triangleright \tau) \leadsto K \sqcap \tau_1 \rrbracket_C^T }$$

PROOF. Analogous to T-CTX-APP-1.

$$\text{Lemma 5.57 (T-CTX-AppBot-2 compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : \bot \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma' \triangleright \tau) \leadsto \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{app}\{K\} \ e \ E \approx \mathsf{app}\{K\} \ e' \ E' : (\Gamma' \triangleright \tau) \leadsto \bot \rrbracket_C^T }$$

PROOF. Analogous to T-CTX-APP-1.

$$\text{Lemma 5.58 (T-CTX-Fst compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \vdash \tau) \leadsto \tau_1 \times \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{fst}\{K\} E \approx \mathsf{fst}\{K\} \, E' : (\Gamma \vdash \tau) \leadsto K \sqcap \tau_1 \rrbracket_C^T }$$

PROOF. Let e, e' such that $\llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : \tau_1 \times \tau_2 \rrbracket$.

We want to show $[\Gamma \vdash_{\mathsf{tru}} \mathsf{fst}\{K\} E[e] \approx \mathsf{fst}\{K\} E'[e'] : K \sqcap \tau_1]$.

By the hypothesis, we get $[\Gamma \vdash_{\mathsf{tru}} E[e] \approx E'[e'] : \tau_1 \times \tau_2]$.

Then the case follows by Lemma 5.25.

$$\text{Lemma 5.59 (T-CTX-FstBot compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \vdash \tau) \rightsquigarrow \bot \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{fst}\{K\} E \approx \mathsf{fst}\{K\} E' : (\Gamma \vdash \tau) \rightsquigarrow \bot \rrbracket_C^T }$$

PROOF. Analagous to T-CTX-FST.

Lemma 5.60 (T-CTX-SND compatibility).
$$\frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \trianglerighteq \tau) \leadsto \tau_1 \times \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{snd}\{K\} E \approx \mathsf{snd}\{K\} E' : (\Gamma \trianglerighteq \tau) \leadsto K \sqcap \tau_2 \rrbracket_C^T }$$

PROOF. Analagous to T-CTX-FST.

2022-11-18 03:01. Page 85 of 1-100.

```
Lemma 5.61 (T-CTX-SndBot compatibility). \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \triangleright \tau) \leadsto \bot \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{snd}\{K\} E \approx \mathsf{snd}\{K\} E' : (\Gamma \triangleright \tau) \leadsto \bot \rrbracket_C^T }
4421
4422
4423
4424
                       PROOF. Analagous to T-CTX-FST.
                                                                                                                                                                                                                                                                                                           4425
                       \text{Lemma 5.62 (T-Ctx-Binop-1 compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \rhd \tau) \leadsto \tau_1 \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} binop E e \approx binop E' e' : (\Gamma \rhd \tau) \leadsto \Delta(binop, \tau_1, \tau_2) \rrbracket_C^T } 
4426
4427
4428
4429
                       PROOF. Let e_1, e'_1 such that \llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \approx e'_1 : \tau \rrbracket.
                       We want to show \llbracket \Gamma \vdash_{\mathsf{tru}} binop E[e_1] e \approx binop E'[e'_1] e' : \Delta(binop, \tau_1, \tau_2) \rrbracket.
4431
                        By the first hypothesis, \llbracket \Gamma \vdash_{\mathsf{tru}} E[e_1] \approx E'[e'_1] : \tau_1 \rrbracket.
4432
4433
                       Then the case follows by Lemma 5.45.
4434
                                                                                                                                                                                                                                                                                                           4435
                       Lemma 5.63 (T-CTX-BINOP-2 compatibility). \frac{\llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : \tau_1 \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \vdash_{\mathsf{T}}) \rightsquigarrow \tau_2 \rrbracket_C^T}{\llbracket \Gamma \vdash_{\mathsf{tru}} binop E e \approx binop E' e' : (\Gamma \vdash_{\mathsf{T}}) \rightsquigarrow \Delta(binop, \tau_1, \tau_2) \rrbracket_C^T}
4436
4437
4438
4439
                       PROOF. Analogous to T-CTX-BINOP-1.
                                                                                                                                                                                                                                                                                                           4440
                        \underbrace{ \left[ \!\! \left[ \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \triangleright \tau) \rightsquigarrow \tau' \right] \!\! \right]_C^T }_{\left[ \!\! \left[ \Gamma \vdash_{\mathsf{tru}} \mathsf{cast} \left\{ K_2 \Leftarrow K_1 \right\} E \approx \mathsf{cast} \left\{ K_2 \Leftarrow K_1 \right\} E' : (\Gamma \triangleright \tau) \rightsquigarrow K_2 \sqcap K_1 \sqcap \tau' \right] \!\! \right]_C^T } 
4441
4442
4443
4444
                       PROOF.
4445
                       4446
4447
4448
                       PROOF. Let e_0, e_0' such that \llbracket \Gamma \vdash_{\mathsf{tru}} e_0 \approx e_0' : \tau \rrbracket.
4449
4450
                       We want to show \llbracket \Gamma \vdash_{\mathsf{tru}} \mathsf{if} E[e_0] \mathsf{then} \ e_1 \mathsf{else} \ e_2 \approx \mathsf{if} \ E'[e'_0] \mathsf{then} \ e'_1 \mathsf{else} \ e'_2 : \tau_1 \sqcup \tau_2 \rrbracket.
4451
                        By the first hypothesis, [\Gamma \vdash_{\mathsf{tru}} E[e_0] \approx E'[e'_0] : \mathsf{Bool}].
4452
                       The case follows by Lemma 5.46.
4453
4454
                       \text{Lemma 5.66 (T-CTx-IfBot-1 compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \trianglerighteq \tau) \leadsto \bot \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \approx e_1' : \tau_1 \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} e_2 \approx e_2' : \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \text{ if } E \text{ then } e_1 \text{ else } e_2 \approx \text{ if } E' \text{ then } e_1' \text{ else } e_2' : (\Gamma \trianglerighteq \tau) \leadsto \bot \rrbracket_C^T } 
4455
4456
4457
                       Proof. Analagous to T-CTX-IF-1
4458
4459
                      4460
4461
4462
                       PROOF. Analagous to T-CTX-IF-1
4463
4464
                        \text{Lemma 5.68 (T-CTX-IfBot-2 compatibility)}. \quad \frac{\llbracket\Gamma \vdash_{\mathsf{tru}} e_b \approx e_b' : \bot \rrbracket_C^T \qquad \llbracket\Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \trianglerighteq \tau) \leadsto \tau_1 \rrbracket_C^T \qquad \llbracket\Gamma \vdash_{\mathsf{tru}} e_2 \approx e_2' : \tau_2 \rrbracket_C^T }{\llbracket\Gamma \vdash_{\mathsf{tru}} \text{ if } e_b \text{ then } E \text{ else } e_2 \approx \text{ if } e_b' \text{ then } E' \text{ else } e_2' : (\Gamma \trianglerighteq \tau) \leadsto \bot \rrbracket_C^T } 
4465
4466
4467
                       PROOF. Analagous to T-CTX-IF-1
4468
4469
                       \text{Lemma 5.69 (T-Ctx-If-3 compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} e_b \approx e_b' : \mathsf{Bool} \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \approx e_1' : \tau_1 \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \triangleright \tau) \leadsto \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \text{ if } e_b \text{ then } e_1 \text{ else } E \approx \text{ if } e_b' \text{ then } e_1' \text{ else } E' : (\Gamma \triangleright \tau) \leadsto \tau_1 \sqcup \tau_2 \rrbracket_C^T } 
4470
4471
4472
```

PROOF. Analogous to T-CTX-IF-1

 $\text{Lemma 5.70 (T-CTX-IfBot-3 compatibility)}. \quad \frac{ \llbracket \Gamma \vdash_{\mathsf{tru}} e_b \approx e_b' : \bot \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} e_1 \approx e_1' : \tau_1 \rrbracket_C^T \qquad \llbracket \Gamma \vdash_{\mathsf{tru}} E \approx E' : (\Gamma \trianglerighteq \tau) \leadsto \tau_2 \rrbracket_C^T }{ \llbracket \Gamma \vdash_{\mathsf{tru}} \text{ if } e_b \text{ then } e_1 \text{ else } E \approx \text{ if } e_b' \text{ then } e_1' \text{ else } E' : (\Gamma \trianglerighteq \tau) \leadsto \bot \rrbracket_C^T }$

PROOF. Analagous to T-CTX-IF-1

5.5.2 Context relation: Fundamental Property

Theorem 5.71 (Context relation is reflexive). If $\Gamma \vdash_{\mathsf{tru}} C : (\Gamma' \vdash_{\mathsf{tru}} C) \leadsto \tau'$, then $[\![\Gamma \vdash_{\mathsf{tru}} C \approx C : (\Gamma' \vdash_{\mathsf{tru}} \tau) \leadsto \tau']\!]$.

PROOF. By induction over the typing derivation, using the compatibility lemmata.

5.6 Check optimization

$$K \setminus \tau = \begin{cases} * & \text{if } \tau \le K \\ K & \text{otherwise} \end{cases}$$

```
\Gamma \vdash_{\mathsf{tru}} e : \tau \leadsto e \mid \mathsf{optimization}
4525
4526
4527
                                                                                                                                                                                                             T-Int
                                                                                                                                                                                                                                                                                             T-True
                                                   (x_0\!:\!K_0)\in\Gamma_0
4528
4529
                                                                                                                        \Gamma_0 \vdash_{\mathsf{tru}} n_0 : \mathsf{Nat} \leadsto n_0
                                                                                                                                                                                                 \Gamma_0 \vdash_{\mathsf{tru}} i_0 : \mathsf{Int} \leadsto i_0
                                                                                                                                                                                                                                                                                             \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{True} : \mathsf{Bool} \leadsto \mathsf{True}
4530
4531
4532
                                                                                                                                                                                                                                                                                                         \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0 \leadsto e'_0
                                                                                                                               \frac{\Gamma_{0}, (x_{0}:K_{0}) \vdash_{\mathsf{tru}} e_{0} : \tau_{1} \leadsto e'_{0}}{\Gamma_{0} \vdash_{\mathsf{tru}} \lambda(x_{0}:K_{0}). e_{0} : * \to \tau_{1} \leadsto \lambda(x_{0}:K_{0}). e'_{0}} \frac{\Gamma_{0} \vdash_{\mathsf{tru}} e_{1} : \tau_{1} \leadsto e'_{1}}{\Gamma_{0} \vdash_{\mathsf{tru}} \langle e_{0}, e_{1} \rangle : \tau_{0} \times \tau_{1} \leadsto \langle e'_{0}, e'_{1} \rangle}
4533
                             T-FALSE
                              \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{False} : \mathsf{Bool} \leadsto \mathsf{False}
4536
                                                                                     T-Cast
4537
                                                                                                                                                                         \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0 \leadsto e'_0
                                                                                      \frac{\Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0 \leadsto e_0'}{\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{cast} \{K_1 \Leftarrow K_0\} e_0 : K_1 \sqcap K_0 \sqcap \tau_0 \leadsto \mathsf{cast} \{K_1 \setminus (K_0 \sqcap \tau_0) \Leftarrow K_0 \setminus \tau_0\} e_0'}
4538
4539
4540
4541
                                           Т-Арр
4542
                                            \frac{\Gamma_0 \vdash_{\mathsf{tru}} e_0 : * \to \tau_1 \leadsto e'_0}{\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{app}\{K_1\} \ e_0 \ e_1 : K_1 \sqcap \tau_1 \leadsto \mathsf{app}\{K_1 \setminus \tau_1\} \ e'_0 \ e'_1}{\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{app}\{K_1\} \ e_0 \ e_1 : K_1 \sqcap \tau_1 \leadsto \mathsf{app}\{K_1 \setminus \tau_1\} \ e'_0 \ e'_1} \frac{\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{app}\{K_1\} \ e_0 \ e_1 : \bot \leadsto \mathsf{app}\{K_1 \setminus \bot\} \ e'_0 \ e'_1}{\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{app}\{K_1\} \ e_0 \ e_1 : \bot \leadsto \mathsf{app}\{K_1 \setminus \bot\} \ e'_0 \ e'_1}
4543
4544
4545
4546
4547
                                                           T-FsT
                                                                                                                                                                                                                                     Т-ГятВот
                                                            \frac{\Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0 \! \times \! \tau_1 \rightsquigarrow e_0'}{\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{fst}\{K_0\} \, e_0 : K_0 \sqcap \tau_0 \rightsquigarrow \mathsf{app}\{K_0 \setminus \tau_0\} \, e_0'}
                                                                                                                                                                                                                                   \frac{\Gamma_0 \vdash_{\mathsf{tru}} e_0 : \bot \leadsto e'_0}{\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{fst}\{K_0\} e_0 : \bot \leadsto \mathsf{fst}\{K_0 \setminus \bot\} e'_0}
4549
4550
4551
4552
                                                         \frac{\Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0 \times \tau_1 \leadsto e_0'}{\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{snd}\{K_1\} e_0 : K_1 \sqcap \tau_1 \leadsto \mathsf{snd}\{K_1 \setminus \tau_1\} e_0'}
                                                                                                                                                                                                                                 \frac{\Gamma_0 \vdash_{\mathsf{tru}} e_0 : \bot \leadsto e'_0}{\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{snd}\{K_1\} e_0 : \bot \leadsto \mathsf{snd}\{K_1 \setminus \bot\} e'_0}
4553
4554
4555
4556
                                                                                                                                                                                                   T-IF
                                                                                                                                                                                                                                                           \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \mathsf{Bool} \leadsto e_0'
4557
                                 T-Binop
4558
                                                                                                                                                                                                                                                               \Gamma_0 \vdash_{\mathsf{tru}} e_1 : \tau_0 \leadsto e_1'
                                                                            \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0 \leadsto e'_0
                                \frac{\Gamma_0 \vdash_{\mathsf{tru}} e_0 : \tau_0 \leadsto e'_0}{\Gamma_0 \vdash_{\mathsf{tru}} e_1 : \tau_1 \leadsto e'_1}
\frac{\Gamma_0 \vdash_{\mathsf{tru}} binop \, e_0 \, e_1 : \Delta(binop, \tau_0, \tau_1) \leadsto binop \, e'_0 \, e'_1}{\Gamma_0 \vdash_{\mathsf{tru}} binop \, e'_0 \, e'_1}
                                                                                                                                                                                                                 \Gamma_0 \vdash_{\mathsf{tru}} e_1 : \tau_0 \leadsto e_1
\Gamma_0 \vdash_{\mathsf{tru}} e_2 : \tau_1 \leadsto e_2'
4559
                                                                                                                                                                                                \overline{\Gamma_0} \vdash_{\mathsf{tru}} \mathsf{if} \ e_0 \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 : \tau_0 \sqcup \tau_1 \leadsto \mathsf{if} \ e_0' \ \mathsf{then} \ e_1' \ \mathsf{else} \ e_2'
4562
                                                                          Т-ІғВот
4563
                                                                                                                              \Gamma_0 \vdash_{\mathsf{tru}} e_0 : \bot \leadsto e'_0
4564
                                                                         4565
4566
4567
4568
4569
4570
                             Theorem 5.72 (Check-Elision Correctness). If \Gamma \vdash_{\mathsf{tru}} e : \tau \leadsto e', then \Gamma \vdash_{\mathsf{tru}} e \approx^{\mathsf{ctx}} e' : \tau.
4571
4572
4573
                             PROOF. Consider arbitrary \Gamma, e, \tau, e' s.t. \Gamma \vdash_{\mathsf{tru}} e : \tau \leadsto e'. By Lemma 5.92, \llbracket \Gamma \vdash_{\mathsf{tru}} e \approx e' : \tau \rrbracket_C^T. By Theorem 5.3,
4574
```

 $\Gamma \vdash_{\mathsf{tru}} e \approx^{\mathsf{ctx}} e' : \tau$, which is what was to be shown.

5.7 Check-elision—Proofs

 Lemma 5.73 ($K \setminus \tau$ preserves meets). $K \cap \tau = (K \setminus \tau) \cap \tau$.

Proof. Immediate by unfolding and lattice properties.

5.7.1 Check-elision: Compatibility Lemmata

$$\text{Lemma 5.74 (T-Var compatibility). } \frac{(x_0\!:\!K_0) \in \Gamma_0}{ \left[\!\!\left[\Gamma_0 \vdash_{\text{tru}} x_0 \approx x_0 : K_0\right]\!\!\right]_C^T}$$

PROOF. By unfolding and Lemma 5.31.

Lemma 5.75 (T-Nat compatibility).
$$\frac{}{\llbracket \Gamma_0 \vdash_{\mathsf{tru}} n_0 \approx n_0 : \mathsf{Nat} \rrbracket_C^T}$$

PROOF. By unfolding and Lemma 5.32.

Lemma 5.76 (T-Int compatibility).
$$\frac{}{\left[\!\!\left[\Gamma_0 \vdash_{\mathsf{tru}} i_0 \approx i_0 : \mathsf{Int}\right]\!\!\right]_C^T}$$

PROOF. By unfolding and Lemma 5.32.

Lemma 5.77 (T-True compatibility).
$$\frac{}{ \left[\!\!\left[\Gamma_{\!0} \vdash_{\mathsf{tru}} \mathsf{True} \approx \mathsf{True} : \mathsf{Bool} \right]\!\!\right]_{C}^{T}}$$

PROOF. By unfolding and Lemma 5.34.

Lemma 5.78 (T-False compatibility).
$$\frac{}{\left[\!\!\left[\Gamma_0 \vdash_{\mathsf{tru}} \mathsf{False} \approx \mathsf{False} : \mathsf{Bool}\right]\!\!\right]_C^T}$$

PROOF. By unfolding and Lemma 5.35.

$$\text{Lemma 5.79 (T-Lam compatibility).} \quad \frac{ \llbracket \Gamma_0, \ (x_0 : K_0) \vdash_{\mathsf{tru}} e_0 \approx e_0' : \tau_1 \rrbracket_C^T }{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} \lambda(x_0 : K_0). \ e_0 \approx \lambda(x_0 : K_0). \ e_0' : * \rightarrow \tau_1 \rrbracket_C^T }$$

PROOF. By unfolding and Lemma 5.36.

$$\text{Lemma 5.80 (T-Pair compatibility)}. \quad \frac{ \begin{bmatrix} \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \approx e_0' : \tau_0 \rrbracket_C^T \\ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_1 \approx e_1' : \tau_1 \rrbracket_C^T \end{bmatrix}}{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} \langle e_0, e_1 \rangle \approx \langle e_0', e_1' \rangle : \tau_0 \times \tau_1 \rrbracket_C^T}$$

PROOF. By unfolding and Lemma 5.37.

$$\underbrace{ \begin{bmatrix} \Gamma \vdash_{\mathsf{tru}} e_1 \approx e_2 : \tau \end{bmatrix}_C^T }_{ \begin{bmatrix} \Gamma \vdash_{\mathsf{tru}} \mathsf{cast} \left\{ K' \leftarrow K \right\} e_1 \approx \mathsf{cast} \left\{ K' \setminus (K \sqcap \tau) \leftarrow K \setminus \tau \right\} e_2 : K' \sqcap K \sqcap \tau \end{bmatrix}_C^T }_{ }$$

PROOF. Follows immediately from lattice properties and Lemma 5.21.

$$\begin{split} & & \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \approx e_0' : * \rightarrow \tau_1 \rrbracket_C^T \\ & & \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_1 \approx e_1' : \tau_0' \rrbracket_C^T \\ & & & \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_1 \approx e_1' : \tau_0' \rrbracket_C^T \\ & & & & \llbracket \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{app}\{K_1\} e_0 \ e_1 \approx \mathsf{app}\{K_1 \setminus \tau_1\} e_0' \ e_1' : K_1 \sqcap \tau_1 \rrbracket_C^T \end{split}$$

```
PROOF. Follows immediately from lattice properties and Lemma 5.23.
4629
                                                                                                                                                                                                                                         4630
                   \text{Lemma 5.83 (T-AppBot compatibility)}. \quad \frac{ \begin{bmatrix} \Gamma_0 \vdash_{\mathsf{tru}} e_0 \approx e_0' : \bot \end{bmatrix}_C^T }{ \begin{bmatrix} \Gamma_0 \vdash_{\mathsf{tru}} e_1 \approx e_1' : \tau_0' \end{bmatrix}_C^T } \\ \frac{ \begin{bmatrix} \Gamma_0 \vdash_{\mathsf{tru}} e_1 \approx e_1' : \tau_0' \end{bmatrix}_C^T }{ \begin{bmatrix} \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{app}\{K_1\} e_0 \ e_1 \approx \mathsf{app}\{K_1 \setminus \bot\} e_0' \ e_1' : \bot \end{bmatrix}_C^T } 
4631
4632
4633
4634
                   PROOF. Follows immediately from Lemma 5.24.
                                                                                                                                                                                                                                         4635
4636
                   \text{Lemma 5.84 (T-Fst compatibility)}. \quad \frac{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \approx e_0' : \tau_0 \times \tau_1 \rrbracket_C^T }{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{fst}\{K_0\} \ e_0 \approx \mathsf{fst}\{K_0 \setminus \tau_0\} \ e_0' : K_0 \sqcap \tau_0 \rrbracket_C^T } 
4637
                   Proof. Follows immediately from lattice properties and Lemma 5.26
                                                                                                                                                                                                                                         4640
                 Lemma 5.85 (T-FstBot compatibility). \frac{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \approx e_0' : \bot \rrbracket_C^T}{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{fst}\{K_0\} \, e_0 \approx \mathsf{fst}\{K_0 \setminus \bot\} \, e_0' : \bot \rrbracket_C^T}
4641
4642
4643
4644
                  PROOF. Follows immediately from Lemma 5.27.
                                                                                                                                                                                                                                         4645
                   \text{Lemma 5.86 (T-Snd compatibility)}. \quad \frac{ \llbracket \Gamma_0 \vdash_{\text{tru}} e_0 \approx e_0' : \tau_0 \times \tau_1 \rrbracket_C^T }{ \llbracket \Gamma_0 \vdash_{\text{tru}} \text{snd}\{K_1\} \ e_0 \approx \text{snd}\{K_1 \setminus \tau_1\} \ e_0' : K_1 \sqcap \tau_1 \rrbracket_C^T } 
4646
4647
4648
                   Proof. Follows immediately from lattice properties and Lemma 5.29.
                                                                                                                                                                                                                                         4649
                 Lemma 5.87 (T-SndBot compatibility). \frac{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \approx e_0' : \bot \rrbracket_C^T }{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{snd}\{K_1\} \, e_0 \approx \mathsf{snd}\{K_1 \setminus \bot\} \, e_0' : \bot \rrbracket_C^T }
4650
4651
4653
                   Proof. Follows immediately from Lemma 5.30.
                                                                                                                                                                                                                                         4654
                  4655
4656
4657
4658
4659
                  PROOF. By unfolding and Lemma 5.45.
                                                                                                                                                                                                                                         4660
                                                                                                                        \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \approx e_0' : \mathsf{Bool} \rrbracket_C^T
4661
                                                                                                                          \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_1 \approx e_1' : \tau_0 \rrbracket_C^T
4662
                                                                                     \frac{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_2 \approx e_2' : \tau_1 \rrbracket_C^T}{ \llbracket \Gamma_0 \vdash_{\mathsf{tru}} \mathsf{if} e_0 \mathsf{then} e_1 \mathsf{else} e_2 \approx \mathsf{if} e_0' \mathsf{then} e_1' \mathsf{else} e_2' : \tau_0 \sqcup \tau_1 \rrbracket_C^T}
4663
                   LEMMA 5.89 (T-IF COMPATIBILITY).
4664
4666
                  Proof. By unfolding and Lemma 5.46.
                                                                                                                                                                                                                                         4667
                                                                                                                              \llbracket \Gamma_0 \vdash_{\mathsf{tru}} e_0 \approx e'_0 : \bot \rrbracket_C^T
4668
                 4669
4670
4671
4672
4673
                   PROOF. By unfolding and Lemma 5.47.
                                                                                                                                                                                                                                         4674
                  4675
4676
4677
4679
                   PROOF. By unfolding and Lemma 5.48.
                                                                                                                                                                                                                                         4680
                                                                                                                                                                                        2022-11-18 03:01. Page 90 of 1-100.
```

5.7.2 Check-elision: Fundamental Property

 Theorem 5.92 (Check-elision is correct for Binary LR). If $\Gamma \vdash_{\text{tru}} e : \tau \leadsto e'$, then $\llbracket \Gamma \vdash_{\text{tru}} e \approx e' : \tau \rrbracket_C^T$

PROOF. By induction over the check-elision judgment derivation, using the compatibility lemmata.

6 Surface

```
4734
               Surface language
4735
4736
                               \coloneqq x \mid n \mid i \mid \mathsf{True} \mid \mathsf{False} \mid \lambda(x : K) \to \tau. \ e \mid \langle e, e \rangle \mid e \ e \mid \mathsf{fst} \ e \mid \mathsf{snd} \ e \mid \mathit{binop} \ e \ e \mid \mathsf{if} \ e \ \mathsf{then} \ e \ \mathsf{else} \ e
4737
                               ::=  Nat | Int | Bool | \tau \times \tau | * \rightarrow \tau | *
4738
                 binop ::= sum | quotient
4739
4740
                            := \cdot | \Gamma, (x:\tau)
4741
                         ::= N
4742
               i := \mathbb{Z}
\Delta^{-1}(binop, \tau) = \begin{cases} Int, Int & \text{if } \tau = Int \\ Nat, Nat & \text{if } \tau = Nat \end{cases}
4743
4745
4746
4747
4748
```

6.1 Simple Translation

$$\Gamma \vdash_{\mathsf{sim}} e : \tau \leadsto e'$$

$$\frac{(x:\tau) \in \Gamma}{\Gamma \vdash_{\mathsf{sim}} x:\tau \leadsto x} \qquad \qquad \frac{\Gamma \vdash_{\mathsf{sim}} n: \mathsf{Nat} \leadsto n}{\Gamma \vdash_{\mathsf{sim}} i: \mathsf{Int} \leadsto i}$$

$$\frac{\Gamma,\ (x\colon\tau) \vdash_{\mathsf{sim}} e\colon\tau'' \leadsto e'}{\Gamma \vdash_{\mathsf{sim}} \lambda(x\colon\tau) \to \tau'.\ e\colon\tau \to \tau' \leadsto \lambda(x\colon\tau).\ ([\tau'\swarrow\tau'']e')} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1\colon\tau_1 \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{sim}} e_2\colon\tau_2 \leadsto e'_2}{\Gamma \vdash_{\mathsf{sim}} \langle e_1, e_2 \rangle \colon\tau_1 \times \tau_2 \leadsto \langle e'_1, e'_2 \rangle}$$

$$\frac{\Gamma \vdash_{\mathsf{sim}} e_1 : \tau \to \tau' \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau'' \leadsto e_2'}{\Gamma \vdash_{\mathsf{sim}} e_1 e_2 : \tau' \leadsto \mathsf{app}\{\tau'\} e_1' \left([\tau \swarrow \tau''] e_2' \right)} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau'}{\Gamma \vdash_{\mathsf{sim}} e_1 e_2 : * \leadsto \mathsf{app}\{*\} \left(\mathsf{cast} \left\{ * \to * \Leftarrow * \right\} e_1 \right) e_2}$$

$$\frac{\Gamma \vdash_{\mathsf{sim}} e : \tau \times \tau' \leadsto e'}{\Gamma \vdash_{\mathsf{sim}} \mathsf{fst} \, e : \tau \leadsto \mathsf{fst}\{\tau\} \, e'} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e : \star \leadsto e'}{\Gamma \vdash_{\mathsf{sim}} \mathsf{fst} \, e : \star \leadsto \mathsf{fst}\{\star\} \, (\mathsf{cast}\, \{\star \times \star \Leftarrow \star \star\} \, e')} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e : \tau \times \tau' \leadsto e'}{\Gamma \vdash_{\mathsf{sim}} \mathsf{snd} \, e : \tau' \leadsto \mathsf{snd}\{\tau'\} \, e'}$$

$$\frac{\Gamma \vdash_{\mathsf{sim}} e : * \leadsto e'}{\Gamma \vdash_{\mathsf{sim}} \mathsf{snd} \, e : * \leadsto \mathsf{snd} \{*\} \, (\mathsf{cast} \, \{* \times * \Leftarrow *\} \, e')}$$

$$\frac{\Gamma \vdash_{\mathsf{sim}} e_1 : \tau_1 \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau_2 \leadsto e_2' \qquad \Delta(\mathit{binop}, \tau_1 \sqcup_{\leqslant} \tau_2, \tau_1 \sqcup_{\leqslant} \tau_2) = \tau' \qquad \tau_1 \leqslant : \mathsf{Int} \land \tau_2 \leqslant : \mathsf{Int}}{\Gamma \vdash_{\mathsf{sim}} \mathit{binop} \, e_1 \, e_2 : \tau' \leadsto \mathit{binop} \, e_1' \, e_2'}$$

$$\frac{\Gamma \vdash_{\mathsf{sim}} e_1 : \tau_1 \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau_2 \leadsto e_2' \qquad \tau_1 = * \lor \tau_1 \leqslant : \mathsf{Int} \qquad \tau_2 = * \lor \tau_2 \leqslant : \mathsf{Int}}{\Gamma \vdash_{\mathsf{sim}} \mathit{binop}\, e_1 \, e_2 : \tau' \leadsto \mathit{binop}\, ([\mathsf{Int} \swarrow \tau_1] e_1') \, ([\mathsf{Int} \swarrow \tau_2] e_2')}$$

$$\frac{\Gamma \vdash_{\mathsf{sim}} e_b : \mathsf{Bool} \leadsto e_b' \qquad \Gamma \vdash_{\mathsf{sim}} e_1 : \tau_1 \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau_2 \leadsto e_2'}{\Gamma \vdash_{\mathsf{sim}} \mathsf{if} \ e_b \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 : \tau_1 \sqcup_{\leqslant} \tau_2 \leadsto \mathsf{if} \ e_b' \ \mathsf{then} \ e_1' \ \mathsf{else} \ e_2'}$$

$$\begin{bmatrix} \tau \swarrow \tau' \end{bmatrix} e = \begin{cases} e & \text{if } \tau \geqslant : \tau' \\ \text{cast } \{\tau \Leftarrow \tau'\} e & \text{if } \tau \not \geqslant : \tau' \land \tau \sim \tau' \end{cases}$$

$$\tau \sqcup_{\leqslant} \tau' = \begin{cases} \tau_{1} & \text{if } \tau_{2} \leqslant : \tau_{1} \\ \tau_{2} & \text{if } \tau_{1} \leqslant : \tau_{2} \end{cases}$$

Lemma 6.1 (Typed Translation Imply Simple Typing). If $\Gamma \vdash_{\text{sim}} e : \tau \leadsto e' \ then \ \Gamma \vdash_{\text{sim}} e' : \tau' \ with \ \tau' \leq \tau$.

PROOF. Proceed by induction on the typed translation.

$$\frac{(x \colon \tau) \in \Gamma}{\Gamma \vdash_{\mathsf{sim}} x \colon \tau \leadsto x} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} n \colon \mathsf{Nat} \leadsto n}{\Gamma \vdash_{\mathsf{sim}} i \colon \mathsf{Int} \leadsto i}$$

These cases are all immediate.

2022-11-18 03:01. Page 93 of 1-100.

$$\frac{\Gamma \vdash_{\mathsf{sim}} e_1 : \tau_1 \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau_2 \leadsto e'_2}{\Gamma \vdash_{\mathsf{sim}} \langle e_1, e_2 \rangle : \tau_1 \times \tau_2 \leadsto \langle e'_1, e'_2 \rangle} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e : \tau \times \tau' \leadsto e'}{\Gamma \vdash_{\mathsf{sim}} \mathsf{st} e : \tau \leadsto \mathsf{fst} \{\tau\} e'} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e : \tau \times \tau' \leadsto e'}{\Gamma \vdash_{\mathsf{sim}} \mathsf{snd} e : \tau' \leadsto \mathsf{snd} \{\tau'\} e'}$$

These cases are all immediate by the IH applied to their premises and their corresponding typing rule in sim.

$$\frac{\Gamma,\ (x\colon\tau)\vdash_{\mathsf{sim}} e\colon\tau''\rightsquigarrow e'}{\Gamma\vdash_{\mathsf{sim}} \lambda(x\colon\tau)\to\tau'.e\colon\tau\to\tau'\rightsquigarrow\lambda(x\colon\tau).\left([\tau'\swarrow\tau'']e'\right)} \qquad \frac{\Gamma\vdash_{\mathsf{sim}} e_1\colon\tau\to\tau'\rightsquigarrow e'_1 \qquad \Gamma\vdash_{\mathsf{sim}} e_2\colon\tau''\rightsquigarrow e'_2}{\Gamma\vdash_{\mathsf{sim}} e_1\colon\tau\to\tau'\rightsquigarrow\lambda(x\colon\tau).\left([\tau'\swarrow\tau'']e'\right)}$$

These cases proceed similarly.

First we apply the IH to all premises.

Then we either, use subsumption to typecheck the body or argument respectively if the types are subtype related, or use T-Cast if they're instead compatible.

Finally, we use the corresponding typing rule to typecheck the elimination form.

$$\frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau'}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau'}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau'}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau'}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1}{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e'_1} \qquad \frac{\Gamma \vdash_{\mathsf{sim}} e_1 : * \leadsto e$$

All of these cases proceed similarly.

First, we apply the IH to all premises.

Then we typecheck the casts with T-Cast, where all compatibility constraints are either given as a premise or immediate. Finally we use the corresponding typing rule to typecheck the elimination form.

$$\frac{\Gamma \vdash_{\mathsf{sim}} e_1 : \tau_1 \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau_2 \leadsto e_2' \qquad \Delta(\mathit{binop}, \tau_1 \sqcup_{\leqslant} \tau_2, \tau_1 \sqcup_{\leqslant} \tau_2) = \tau' \qquad \tau_1 \leqslant : \mathsf{Int} \land \tau_2 \leqslant : \mathsf{Int}}{\Gamma \vdash_{\mathsf{sim}} \mathit{binop} e_1 e_2 : \tau' \leadsto \mathit{binop} e_1' e_2'}$$

By the IH, we have $\Gamma \vdash_{\mathsf{sim}} e_1' : \tau_1$.

By the IH, we have $\Gamma \vdash_{\mathsf{sim}} e_2' : \tau_2$.

Then we can use subsumption to get both $\Gamma \vdash_{\mathsf{sim}} e'_1 : \tau_1 \sqcup_{\leqslant} \tau_2$ and $\Gamma \vdash_{\mathsf{sim}} e'_2 : \tau_1 \sqcup_{\leqslant} \tau_2$.

Finally we can typecheck with T-BINOP.

$$\frac{\Gamma \vdash_{\mathsf{sim}} e_1 : \tau_1 \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau_2 \leadsto e_2' \qquad \tau_1 = \ast \lor \tau_1 \leqslant : \mathsf{Int} \qquad \tau_2 = \ast \lor \tau_2 \leqslant : \mathsf{Int}}{\Gamma \vdash_{\mathsf{sim}} \mathit{binop}\, e_1 \, e_2 : \tau' \leadsto \mathit{binop}\, ([\mathsf{Int} \swarrow \tau_1] e_1') \, ([\mathsf{Int} \swarrow \tau_2] e_2')}$$

By the IH, we have $\Gamma \vdash_{\mathsf{sim}} e_1' : \tau_1$.

By the IH, we have $\Gamma \vdash_{\mathsf{sim}} e_2' : \tau_2$.

If $\tau_1 = *$, then [Int $\checkmark \tau_1$] $e'_1 = \text{cast {Int }} \Leftarrow *$ } e'_1 , and by the IH we have $\Gamma \vdash_{\text{sim}} \text{cast {Int }} \Leftarrow *$ } $e'_1 : \text{Int.}$

Otherwise, [Int $\sqrt{\tau_1}$] $e'_1 = e'_1$.

If $\tau_2 = *$, then [Int $\checkmark \tau_2$] $e_2' = \text{cast {Int }} \Leftarrow *$ } e_2' , and by the IH we have $\Gamma \vdash_{\text{sim}} \text{cast {Int }} \Leftarrow *$ } $e_2' : \text{Int.}$

2022-11-18 03:01. Page 94 of 1-100.

Otherwise, [Int $\checkmark \tau_2$] $e'_2 = e'_2$.

Finally we can typecheck with T-BINOP and potentially T-Subsumption.

$$\frac{\Gamma \vdash_{\mathsf{sim}} e_b : \mathsf{Bool} \leadsto e_b' \qquad \Gamma \vdash_{\mathsf{sim}} e_1 : \tau_1 \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{sim}} e_2 : \tau_2 \leadsto e_2'}{\Gamma \vdash_{\mathsf{sim}} \mathsf{if} \ e_b \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 : \tau_1 \sqcup_{\leqslant} \tau_2 \leadsto \mathsf{if} \ e_b' \ \mathsf{then} \ e_1' \ \mathsf{else} \ e_2'}$$

By the IH, we have $\Gamma \vdash_{\mathsf{sim}} e'_h$: Bool.

By the IH, we have $\Gamma \vdash_{\mathsf{sim}} e_1' : \tau_1$.

By the IH, we have $\Gamma \vdash_{\mathsf{sim}} e_2' : \tau_2$.

Then by subsumption, we have $\Gamma \vdash_{\mathsf{sim}} e'_1 : \tau_1 \sqcup_{\leqslant} \tau_2$ and $\Gamma \vdash_{\mathsf{sim}} e'_2 : \tau_1 \sqcup_{\leqslant} \tau_2$.

Finally, we can typecheck with T-IF.

6.2 Truer Transient Translation

$$\tau \setminus K = \begin{cases} * & \text{if } K \le \tau \\ \tau & \text{otherwise} \end{cases}$$
$$\Gamma \vdash_{\mathsf{tru}} e \Rightarrow \tau \leadsto e'$$

$$\frac{(x:K) \in \Gamma}{\Gamma \vdash_{\mathsf{tru}} x \Rightarrow K \leadsto x}$$

$$\Gamma \vdash_{\mathsf{tru}} n \Longrightarrow \mathsf{Nat} \rightsquigarrow n$$

$$\Gamma \vdash_{\mathsf{tru}} i \Rightarrow \mathsf{Int} \rightsquigarrow i$$

$$\frac{\Gamma,\ (x{:}K) \vdash_{\mathsf{tru}} e \Leftarrow^+ \tau \leadsto e'}{\Gamma \vdash_{\mathsf{tru}} \lambda(x{:}K) \to \tau. \, e \Rightarrow * \to \tau \leadsto \lambda(x{:}K). \, e'}$$

$$\frac{\Gamma,\ (x:K) \vdash_{\mathsf{tru}} e \leftrightharpoons^+ \tau \leadsto e'}{\Gamma \vdash_{\mathsf{tru}} \lambda(x:K) \to \tau.\ e \Rightarrow * \to \tau \leadsto \lambda(x:K).\ e'} \frac{\Gamma \vdash_{\mathsf{tru}} e_1 \Rightarrow \tau_1 \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{tru}} e_2 \Rightarrow \tau_2 \leadsto e'_2}{\Gamma \vdash_{\mathsf{tru}} \langle e_1, e_2 \rangle \Rightarrow \tau_1 \times \tau_2 \leadsto \langle e'_1, e'_2 \rangle}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e_1 \Rightarrow * \rightarrow \tau \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{tru}} e_2 \Rightarrow \tau'}{\Gamma \vdash_{\mathsf{tru}} e_1 e_2 \Rightarrow \tau \leadsto \mathsf{app}\{*\} e_1 e_2}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e_1 \Rightarrow * \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{tru}} e_2 \Rightarrow \tau'}{\Gamma \vdash_{\mathsf{tru}} e_1 e_2 \Rightarrow * \leadsto \mathsf{app}\{*\} \left(\mathsf{cast} \left\{* \to * \Leftarrow *\right\} e_1\right) e_2}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e \Rightarrow \tau \times \tau' \rightsquigarrow e'}{\Gamma \vdash_{\mathsf{tru}} \mathsf{fst} e \Rightarrow \tau \rightsquigarrow \mathsf{fst}\{*\} e'}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e \Rightarrow \tau \times \tau' \leadsto e'}{\Gamma \vdash_{\mathsf{tru}} \mathsf{fst} \, e \Rightarrow \tau \leadsto \mathsf{fst}\{*\} \, e'} \qquad \frac{\Gamma \vdash_{\mathsf{tru}} e \Rightarrow * \leadsto e'}{\Gamma \vdash_{\mathsf{tru}} \mathsf{fst} \, e \Rightarrow * \leadsto \mathsf{fst}\{*\} \, (\mathsf{cast}\, \{* \times * \Leftarrow *\} \, e')} \qquad \frac{\Gamma \vdash_{\mathsf{tru}} e \Rightarrow \tau \times \tau' \leadsto e'}{\Gamma \vdash_{\mathsf{tru}} \mathsf{snd} \, e \Rightarrow \tau \leadsto \mathsf{snd}\{*\} \, e'}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e \Rightarrow \tau \times \tau' \rightsquigarrow e'}{\Gamma \vdash_{\mathsf{tru}} \mathsf{end} e \Rightarrow \tau \rightsquigarrow \mathsf{end} * \mathsf{e}'}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e \Rightarrow * \leadsto e'}{\Gamma \vdash_{\mathsf{tru}} \mathsf{snd} \, e \Rightarrow * \leadsto \mathsf{snd}\{*\} \, (\mathsf{cast} \, \{* \times * \Leftarrow *\} \, e')}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e_1 \Rightarrow \tau_1 \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{tru}} e_2 \Rightarrow \tau_2 \leadsto e_2' \qquad \Delta(\mathit{binop}, \tau_1, \tau_2) = \tau'}{\Gamma \vdash_{\mathsf{tru}} \mathit{binop}\, e_1 \, e_2 \Rightarrow \tau' \leadsto \mathit{binop}\, e_1' \, e_2'}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e_b \Rightarrow \mathsf{Bool} \leadsto e_b' \qquad \Gamma \vdash_{\mathsf{tru}} e_1 \Rightarrow \tau_1 \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{tru}} e_2 \Rightarrow \tau_2 \leadsto e_2'}{\Gamma \vdash_{\mathsf{tru}} \mathsf{if} \ e_b \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 \Rightarrow \tau_1 \sqcup \tau_2 \leadsto \mathsf{if} \ e_b' \ \mathsf{then} \ e_1' \ \mathsf{else} \ e_2'}$$

$$\Gamma \vdash_{\mathsf{tru}} e \iff \tau \leadsto e'$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e \Rightarrow \tau' \leadsto e' \qquad \tau' \leq}{\Gamma \vdash_{\mathsf{tru}} e \Longleftrightarrow \tau \leadsto e'}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e \Rightarrow \tau' \leadsto e' \qquad \tau' \leq \tau}{\Gamma \vdash_{\mathsf{tru}} e \Longleftrightarrow \tau \leadsto e'} \qquad \frac{\Gamma \vdash_{\mathsf{tru}} e \Rightarrow \tau' \leadsto e' \qquad \tau' \nleq K}{\Gamma \vdash_{\mathsf{tru}} e \Longleftrightarrow^{\Rightarrow} \kappa \leadsto \mathsf{cast} \{K \Leftarrow \lfloor \tau' \rfloor\} e'}$$

$$\Gamma \vdash_{\mathsf{tru}} e \Leftarrow^+ \tau \leadsto e'$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e \Leftarrow \tau \leadsto e'}{\Gamma \vdash_{\mathsf{tru}} e \Leftarrow^+ \tau \leadsto e'}$$

$$\frac{\neg(\exists e'. \ \Gamma \vdash_{\mathsf{tru}} e \Leftarrow \tau \leadsto e') \qquad \Gamma \vdash_{\mathsf{tru}} e \Leftarrow^{\Rightarrow} \tau \leadsto e'}{\Gamma \vdash_{\mathsf{tru}} e \Leftarrow^{+} \tau \leadsto e'}$$

$$\Gamma \vdash_{\mathsf{tru}} e \Leftarrow \tau \leadsto e'$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e_1 \Leftarrow^+ \tau_1 \rightsquigarrow e_1' \qquad \Gamma \vdash_{\mathsf{tru}} e_2 \Leftarrow^+ \tau_2 \rightsquigarrow e_2'}{\Gamma \vdash_{\mathsf{tru}} \langle e_1, e_2 \rangle \Leftarrow \tau_1 \times \tau_2 \rightsquigarrow \langle e_1', e_2' \rangle} \qquad \frac{\Gamma \vdash_{\mathsf{tru}} e \Leftarrow^+ (\tau \setminus \lfloor \tau \rfloor) \times * \rightsquigarrow e'}{\Gamma \vdash_{\mathsf{tru}} \mathsf{fst} e \Leftarrow \tau \rightsquigarrow \mathsf{fst} \{ \lfloor \tau \rfloor \} e'}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e \Leftarrow^+ *\times (\tau \setminus \lfloor \tau \rfloor) \leadsto e'}{\Gamma \vdash_{\mathsf{tru}} \mathsf{snd} e \Leftarrow \tau \leadsto \mathsf{snd}\{\lfloor \tau \rfloor\} e'} \qquad \frac{\Gamma \vdash_{\mathsf{tru}} e_b \Leftarrow^+ \mathsf{Bool} \leadsto e'_b \qquad \Gamma \vdash_{\mathsf{tru}} e_1 \Leftarrow^+ \tau \leadsto e'_1 \qquad \Gamma \vdash_{\mathsf{tru}} e_2 \Leftarrow^+ \tau \leadsto e'_2}{\Gamma \vdash_{\mathsf{tru}} \mathsf{if} e_b \mathsf{then} e_1 \mathsf{else} e_2 \Leftarrow \tau \leadsto \mathsf{if} e'_b \mathsf{then} e'_1 \mathsf{else} e'_2}$$

$$\frac{\Gamma \vdash_{\mathsf{tru}} e_1 \Leftarrow^+ \tau_1 \leadsto e_1' \qquad \Gamma \vdash_{\mathsf{tru}} e_2 \Leftarrow^+ \tau_2 \leadsto e_2' \qquad \Delta^{-1}(\mathit{binop}, \tau') = \tau_1, \tau_2}{\Gamma \vdash_{\mathsf{tru}} \mathit{binop} e_1 e_2 \Leftarrow \tau' \leadsto \mathit{binop} e_1' e_2'}$$

For the purpose of the following proof, assume the tru rules are used in each judgement.

LEMMA 6.2 (TYPED TRANSLATIONS IMPLY TRUER Transient TYPING).

- (1) If $\Gamma \vdash e \Rightarrow \tau \leadsto e'$ then $\Gamma \vdash e' : \tau'$ with $\tau' \leq \tau$.
- (2) If $\Gamma \vdash e \iff \tau \leadsto e' \text{ then } \Gamma \vdash e' : \tau' \text{ with } \tau' \leq \tau$.
- (3) If $\Gamma \vdash e \rightleftharpoons^+ \tau \rightsquigarrow e'$ then $\Gamma \vdash e' : \tau'$ with $\tau' \leq \tau$.
- (4) If $\Gamma \vdash e \leftarrow \tau \leadsto e'$ then $\Gamma \vdash e' : \tau'$ with $\tau' \leq \tau$.

PROOF. All cases proceed by induction over their respective judgement derivations.

This is well founded by the size of the term e, with the caveat that (2) will call into (1) with the same term, but (1) will then reduce the size before calling back into (2) (in the lambda case, through (3)).

Similarly, (3) will call into (2), but by the time it gets back to (3), the term will have been reduced in size in (1) (in the lambda case).

And similarly, (3) will call into (4), but by the time it gets back to (3), the term will have reduced in size.

$$\frac{(x:K) \in \Gamma}{\Gamma \vdash x \Rightarrow K \rightsquigarrow x} \qquad \frac{\Gamma \vdash n \Rightarrow \text{Nat} \rightsquigarrow n}{\Gamma \vdash n \Rightarrow \text{Int} \rightsquigarrow i}$$

All of the above cases follow immediately.

$$\frac{\Gamma \vdash e_1 \Rightarrow \tau_1 \leadsto e_1' \qquad \Gamma \vdash e_2 \Rightarrow \tau_2 \leadsto e_2'}{\Gamma \vdash \langle e_1, e_2 \rangle \Rightarrow \tau_1 \times \tau_2 \leadsto \langle e_1', e_2' \rangle}$$

Follows immediately by the induction hypotheses.

$$\frac{\Gamma \vdash e_1 \Rightarrow * \rightarrow \tau \leadsto e_1' \qquad \Gamma \vdash e_2 \Rightarrow \tau'}{\Gamma \vdash e_1 e_2 \Rightarrow \tau \leadsto \operatorname{app}\{*\} e_1 e_2} \qquad \frac{\Gamma \vdash e \Rightarrow \tau \times \tau' \leadsto e'}{\Gamma \vdash \operatorname{fst} e \Rightarrow \tau \leadsto \operatorname{fst}\{*\} e'} \qquad \frac{\Gamma \vdash e \Rightarrow \tau \times \tau' \leadsto e'}{\Gamma \vdash \operatorname{snd} e \Rightarrow \tau \leadsto \operatorname{snd}\{*\} e'}$$

All of the above cases follow similar reasoning.

We apply the induction hypothesis to each premise.

If the term being eliminated is at type \bot , then we use the corresponding \bot rule.

Otherwise we use the corresponding elimination rule with check *.

2022-11-18 03:01. Page 97 of 1-100.

$$\frac{\Gamma \vdash e_1 \Rightarrow * \leadsto e_1' \qquad \Gamma \vdash e_2 \Rightarrow \tau'}{\Gamma \vdash e_1 e_2 \Rightarrow * \leadsto \operatorname{app}\{*\} \left(\operatorname{cast}\{* \to * \Leftarrow *\} e_1\right) e_2} \qquad \frac{\Gamma \vdash e \Rightarrow * \leadsto e'}{\Gamma \vdash \operatorname{fst} e \Rightarrow * \leadsto \operatorname{fst}\{*\} \left(\operatorname{cast}\{* \times * \Leftarrow *\} e'\right)}$$

$$\frac{\Gamma \vdash e \Rightarrow * \leadsto e'}{\Gamma \vdash \mathsf{snd}\, e \Rightarrow * \leadsto \mathsf{snd}\{*\}\,(\mathsf{cast}\, \{*\times * \Leftarrow *\}\, e')}$$

All of the above cases follow similar reasoning.

The reasoning is identical to the previous case, with the note that the boundary term also sends the type below the tag corresponding to the kind of elimination form.

$$\frac{\Gamma \vdash e_1 \Rightarrow \tau_1 \leadsto e_1' \qquad \Gamma \vdash e_2 \Rightarrow \tau_2 \leadsto e_2' \qquad \Delta(\mathit{binop}, \tau_1, \tau_2) = \tau}{\Gamma \vdash \mathit{binop}\, e_1 \, e_2 \Rightarrow \tau' \leadsto \mathit{binop}\, e_1' \, e_2'}$$

From (1) we get that there is a $\tau'_1 \leq \tau_1$ such that $\Gamma \vdash e'_1 : \tau'_1$.

From (1) we get that there is a $\tau_2' \le \tau_2$ such that $\Gamma \vdash e_2' : \tau_2'$.

If $\tau_1' = \bot$ or $\tau_2' = \bot$ then we're done, because $\Delta(\textit{binop}, \tau_1', \tau_2') = \bot$.

Otherwise, $\tau_1' = \text{Int or Nat and } \tau_2' = \text{Int or Nat. If } \tau_1' \neq \tau_2'$, we can use subsumption to get both e_1' and e_2' at Int to complete the case.

Otherwise they're both at Nat or Int, which is sufficient to complete the case.

$$\frac{\Gamma \vdash e_b \Rightarrow \mathsf{Bool} \leadsto e_b' \qquad \Gamma \vdash e_1 \Rightarrow \tau_1 \leadsto e_1' \qquad \Gamma \vdash e_2 \Rightarrow \tau_2 \leadsto e_2'}{\Gamma \vdash \mathsf{if} \ e_b \ \mathsf{then} \ e_1 \ \mathsf{else} \ e_2 \Rightarrow \tau_1 \sqcup \tau_2 \leadsto \mathsf{if} \ e_b' \ \mathsf{then} \ e_1' \ \mathsf{else} \ e_2'}$$

By (1) we have $\exists \tau_b \leq \text{Bool such that } \Gamma \vdash e'_b : \tau_b$.

By (1) we have $\exists \tau_1 \leq \tau$ such that $\Gamma \vdash e'_1 : \tau_1$.

By (1) we have $\exists \tau_2 \leq \tau$ such that $\Gamma \vdash e_2' : \tau_2$.

If $\tau_b = \bot$, then we're done by the if bot rule.

Otherwise, we get by the if rule that $\Gamma \vdash \text{if } e_b'$ then e_1' else $e_2' : \tau_1 \sqcup \tau_2$, and that $\tau_1 \sqcup \tau_2 \leq \tau$ by the fact that \sqcup is a greatest lower bound.

$$\frac{\Gamma,\,(x\!:\!K) \vdash e \Leftarrow^+ \tau \leadsto e'}{\Gamma \vdash \lambda(x\!:\!K) \to \tau.\,e \Rightarrow * \to \tau \leadsto \lambda(x\!:\!K).\,e'}$$

By the lambda typing rule for truer typing, we want to show there is a $\tau' \le \tau$ such that Γ , $(x:K) \vdash e' : \tau'$. This is immediate from (3) applied to the premise.

$$\frac{\Gamma \vdash e \Rightarrow \tau' \leadsto e' \qquad \tau' \leq \tau}{\Gamma \vdash e \Longleftrightarrow \tau \leadsto e'}$$

By (1), we have there is a $\tau'' \le \tau'$ such that $\Gamma \vdash e : \tau''$.

Since \leq is transitive, this completes the case.

$$\frac{\Gamma \vdash e \Rightarrow \tau' \leadsto e' \qquad \tau' \nleq K}{\Gamma \vdash e \Longleftrightarrow^{\Rightarrow} K \leadsto \text{cast} \{K \Leftarrow \lfloor \tau' \rfloor\} e'}$$

From (1) we have $\tau'' \le \tau'$ such that $\Gamma \vdash e' : \tau''$.

We want to show there is a $tau''' \le K$ such that $\Gamma \vdash \text{cast } \{K \Leftarrow \lfloor \tau' \rfloor\} \ e' : \tau'''$.

Set $\tau''' \cap \lfloor \tau' \rfloor \cap K$ to be τ''' .

 By the boundary typing rule of truer typing, this typechecks.

The last condition is that $\tau''' \leq K$, which is immediate by the fast that \sqcap is the greatest lower bound.

$$\frac{\neg(\exists e'.\,\Gamma\vdash e \Leftarrow \tau \leadsto e') \qquad \Gamma\vdash e \Leftarrow^{\Rightarrow} \tau \leadsto e'}{\Gamma\vdash e \Leftarrow^{+} \tau \leadsto e'}$$

Immediate by (2).

$$\frac{\Gamma \vdash e \Leftarrow \tau \leadsto e'}{\Gamma \vdash e \Leftarrow^+ \tau \leadsto e'}$$

Immediate by (4).

$$\frac{\Gamma \vdash e_1 \Leftarrow^+ \tau_1 \leadsto e_1' \qquad \Gamma \vdash e_2 \Leftarrow^+ \tau_2 \leadsto e_2'}{\Gamma \vdash \langle e_1, e_2 \rangle \Leftarrow \tau_1 \times \tau_2 \leadsto \langle e_1', e_2' \rangle}$$

Immediate by (3) and induction.

$$\frac{\Gamma \vdash e \Leftarrow^+ (\tau \setminus \lfloor \tau \rfloor) \times * \leadsto e'}{\Gamma \vdash \text{fst } e \Leftarrow \tau \leadsto \text{fst}\{|\tau|\} e'}$$

By our induction hypothesis, we have that there is some $\tau' \leq (\tau \setminus \lfloor \tau \rfloor) \times *$ such that $\Gamma \vdash e' : \tau'$.

If $\tau' = \bot$, then we're done by the fst bot rule.

Otherwise, $\tau' = \tau'_1 \times \tau'_2$, and $\tau'_1 \le \tau \setminus \lfloor \tau \rfloor$.

By the fst projection typing rule, we have that $\Gamma \vdash \mathsf{fst}\{\lfloor \tau \rfloor\} \, e' : \tau'_1 \sqcap \lfloor \tau \rfloor$.

It suffices to show that $\tau'_1 \sqcap \lfloor \tau \rfloor \leq \tau$.

If $\tau \setminus \lfloor \tau \rfloor = *$, then $\lfloor \tau \rfloor \le \tau$, which means $\tau'_1 \cap \lfloor \tau \rfloor \le \lfloor \tau \rfloor \le \tau$.

Otherwise, $\tau \setminus \lfloor \tau \rfloor = \tau$, which means $\tau'_1 \le \tau$ and therefore $\tau'_1 \sqcap \lfloor \tau \rfloor \le \tau$.

$$\frac{\Gamma \vdash e \Leftarrow^+ *\times (\tau \setminus \lfloor \tau \rfloor) \rightsquigarrow e'}{\Gamma \vdash \operatorname{snd} e \Leftarrow \tau \rightsquigarrow \operatorname{snd}\{\lfloor \tau \rfloor\} e'}$$

Not meaningfully different from the previous case regarding fst .

```
5149
5150
                                                        \frac{\Gamma \vdash e_b \Leftarrow^+ \operatorname{Bool} \leadsto e_b' \qquad \Gamma \vdash e_1 \Leftarrow^+ \tau \leadsto e_1' \qquad \Gamma \vdash e_2 \Leftarrow^+ \tau \leadsto e_2'}{\Gamma \vdash \operatorname{if} e_b \operatorname{then} e_1 \operatorname{else} e_2 \Leftarrow \tau \leadsto \operatorname{if} e_b' \operatorname{then} e_1' \operatorname{else} e_2'}
5151
5152
5153
             By (3) we have \exists \tau_b \leq \text{Bool such that } \Gamma \vdash e'_b : \tau_b.
5154
             By (3) we have \exists \tau_1 \leq \tau such that \Gamma \vdash e'_1 : \tau_1.
5155
5156
             By (3) we have \exists \tau_2 \leq \tau such that \Gamma \vdash e_2' : \tau_2.
5157
             If \tau_b = \bot, then we're done by the if bot rule.
5158
             Otherwise, we get by the if rule that \Gamma \vdash \text{if } e_h' then e_1' else e_2' : \tau_1 \sqcup \tau_2, and that \tau_1 \sqcup \tau_2 \leq \tau by the fact that \sqcup is a
5159
5160
             greatest lower bound.
5161
5162
                                                      \frac{\Gamma \vdash e_1 \Leftarrow^+ \tau_1 \leadsto e_1' \qquad \Gamma \vdash e_2 \Leftarrow^+ \tau_2 \leadsto e_2' \qquad \Delta^{-1}(\mathit{binop}, \tau') = \tau_1, \tau_2}{\Gamma \vdash \mathit{binop}\, e_1 \, e_2 \Leftarrow \tau' \leadsto \mathit{binop}\, e_1' \, e_2'}
5163
5164
5165
                 By (3) we have \exists \tau_1' \leq \tau_1 such that \Gamma \vdash e_1' : \tau_1'.
5166
5167
            By (3) we have \exists \tau_2' \leq \tau_2 such that \Gamma \vdash e_2' : \tau_2'.
5168
             By the definition of \Delta^{-1}, either \tau_1 = \tau_2 = \text{Int or } \tau_1 = \tau_2 = \text{Nat.}
5169
             If \tau_1' = \bot or \tau_2' = \bot, then we're done because \Delta(binop, \tau_1', \tau_2') = \bot.
5170
             Otherwise, we have \tau_1' = Int or Nat and similarly for \tau_2'.
5171
5172
             If \tau_1' \neq \tau_2', then we can use subsumption to get both at Int and complete the case.
5173
             Otherwise, we get that both are Int or Nat, which is sufficient to complete the case.
                                                                                                                                                                                                                               5174
5175
                 THEOREM 6.3 (TYPED TRANSLATION IMPLIES TRUER Transient TYPING).
5176
             If \Gamma \vdash e \Rightarrow \tau \leadsto e' then \Gamma \vdash e : \tau' where \tau' \leq \tau.
5177
5178
5179
5180
5181
5182
5183
5184
5186
5187
5188
5189
5190
5191
5192
5193
5194
```