rel_mng_2_iff THM

```
⊢  ∀r1,r2:rel(), ds,daa:Collection(dec()), da:Collection(SimpleType)
|   , de:sig(), rho:Decl, e:{[[de]] rho}, s1,s2:{[[ds]] rho}
|   , a:[[da]] rho, tr:trace_env([[daa]] rho).
|   trace_consistent_rel(rho;daa;tr.proj;r1)
|   ⇒
|   trace_consistent_rel(rho;daa;tr.proj;r2)
|   ⇒
|   tc(r1;ds;da;de)
|   ⇒
|   tc(r2;ds;da;de)
|   ⇒
|   r1.name = r2.name
|   ⇒
|   ||r1.args|| = ||r2.args||
|   ⇒
|   (∀i:ℕ.
|    i<||r1.args||
|    ⇒
|    [[r1.args[i]]] 1of(e) s1 a tr = [[r2.args[i]]] 1of(e) s1 s2 a tr)
|   ⇒
|   ([[r1]] rho ds da de e s1 a tr
|    ⟺
|    rel_mng_2(r2; rho; ds; da; de; e; s1; s2; a; tr))
| by UnivCD THENA Auto
THEN Try (Unfold ‘decl‘ 0 THEN Complete Auto)
|

|    ...
|         THEN ExRepD THEN Trivial)

|    THEN Try (Complete (BackThruLemma ‘rel_arg_typing‘ THEN Auto

|    ...

|    ...

|    ....
|
|\
| 1. r1 : rel()
| 2. r2 : rel()
| 3. ds : Collection(dec())
| 4. daa : Collection(dec())
| 5. da : Collection(SimpleType)
| 6. de : sig()
| 7. rho : Decl
| 8. e : {[[de]] rho}
| 9. s1 : {[[ds]] rho}
| 10. s2 : {[[ds]] rho}
| 11. a : [[da]] rho
| 12. tr : trace_env([[daa]] rho)
| 13. trace_consistent_rel(rho;daa;tr.proj;r1)
| 14. trace_consistent_rel(rho;daa;tr.proj;r2)
```

```
| 15. tc(r1;ds;da;de)
| 16. tc(r2;ds;da;de)
| 17. r1.name = r2.name
| 18. ||r1.args|| = ||r2.args||
| 19. ∀i:ℕ.
| |   i<||r1.args||
| |   ⇒
| |   [[r1.args[i]]] 1of(e) s1 a tr = [[r2.args[i]]] 1of(e) s1 s2 a tr
| ⊢  [[r1]] rho ds da de e s1 a tr
| |  ⟺
| |  rel_mng_2(r2; rho; ds; da; de; e; s1; s2; a; tr)
| | by All (Unfolds ‘‘rel_mng_2 rel_mng‘‘)
| |
| 1. r1 : rel()
| 2. r2 : rel()
| 3. ds : Collection(dec())
| 4. daa : Collection(dec())
| 5. da : Collection(SimpleType)
| 6. de : sig()
| 7. rho : Decl
| 8. e : {[[de]] rho}
| 9. s1 : {[[ds]] rho}
| 10. s2 : {[[ds]] rho}
| 11. a : [[da]] rho
| 12. tr : trace_env([[daa]] rho)
| 13. trace_consistent_rel(rho;daa;tr.proj;r1)
| 14. trace_consistent_rel(rho;daa;tr.proj;r2)
| 15. tc(r1;ds;da;de)
| 16. tc(r2;ds;da;de)
| 17. r1.name = r2.name
| 18. ||r1.args|| = ||r2.args||
| 19. ∀i:ℕ.
| |   i<||r1.args||
| |   ⇒
| |   [[r1.args[i]]] 1of(e) s1 a tr = [[r2.args[i]]] 1of(e) s1 s2 a tr
| ⊢  list_accum(x,t.x
| |                ([[t]] 1of(e) s1 a tr);[[r1.name]] rho ... ;...)
| |  ⟺
| |  list_accum(x,t.x
| |                ([[t]] 1of(e) s1 s2 a tr);[[...]] rho 2of(e) ;...)
| | by MoveToConcl (-1)
| THEN RepeatFor 4 (MoveToConcl (-3))
|
| |    THEN MoveToConcl (-1)
|
| |    THEN Unfolds ‘‘tc trace_consistent_rel‘‘ 0 THEN Reduce 0
|
| |    THEN GenConcl r1.args = l1 THENA Auto THEN Thin (-1)
|
| |    THEN GenConcl r2.args = l2 THENA Auto THEN Thin (-1)
|
| |    THEN RevHypSubstSq (-3) 0 THEN Thin (-3)
|
| |    THEN let i = 8 in let j = 9 in D i
| THEN All Reduce
```

```
|
| |    THEN Unfold `sig_mng` j
| THEN Unfold `record` j
|
| |    THEN Unfold `decl_type` j
| THEN Reduce j
|
| |    THEN Unfold `st_list_mng` j
| THEN D 1 THEN D 1
|
| |    THEN ((Fold `relname_eq` 0) ORELSE (Fold `relname_other` 0))
|
| |    THEN Unfold `relname_mng` 0
| THEN All Reduce
|
| |    THEN Thin 2 THEN Thin 2
| THEN D 0 THENA Auto'
| .
| |
| |\
| | 1. x : SimpleType
| | 2. ds : Collection(dec())
| | 3. daa : Collection(dec())
| | 4. da : Collection(SimpleType)
| | 5. de : sig()
| | 6. rho : Decl
| | 7. e1 : {1of([[de]] rho)}
| | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | 9. s1 : {[[ds]] rho}
| | 10. s2 : {[[ds]] rho}
| | 11. a : [[da]] rho
| | 12. tr : trace_env([[daa]] rho)
| | 13. l1 : Term List
| | 14. l2 : Term List
| | 15. ||l1|| = ||l2||
| | ⊢  (∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i]))
| | |   ⇒
| | |   (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| | |   ⇒
| | |   ||l1|| = 2
| | |   & x ∈ term_types(ds;da;de;l1[0])
| | |     & x ∈ term_types(ds;da;de;l1[1])
| | |   ⇒
| | |   ||l2|| = 2
| | |   & x ∈ term_types(ds;da;de;l2[0])
| | |     & x ∈ term_types(ds;da;de;l2[1])
| | |   ⇒
| | |   (∀i:ℕ.
| | |    i<||l1|| ⇒ [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| | |   ⇒
| | |   (list_accum(x,t.x([[t]] e1 s1 a tr);λx@0,y. x@0 = y;l1)
| | |    ⟺
| | |    list_accum(x,t.x([[t]] e1 s1 s2 a tr);λx@0,y. x@0 = y;l2))
| | | by ...
| | |
```

```
| | 1. x : SimpleType
| | 2. ds : Collection(dec())
| | 3. daa : Collection(dec())
| | 4. da : Collection(SimpleType)
| | 5. de : sig()
| | 6. rho : Decl
| | 7. e1 : {1of([[de]] rho)}
| | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | 9. s1 : {[[ds]] rho}
| | 10. s2 : {[[ds]] rho}
| | 11. a : [[da]] rho
| | 12. tr : trace_env([[daa]] rho)
| | 13. l1 : Term List
| | 14. l2 : Term List
| | 15. ||l1|| = ||l2||
| | 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | 18. ||l1|| = 2
| | 19. x ∈ term_types(ds;da;de;l1[0])
| | 20. x ∈ term_types(ds;da;de;l1[1])
| | 21. ||l2|| = 2
| | 22. x ∈ term_types(ds;da;de;l2[0])
| | 23. x ∈ term_types(ds;da;de;l2[1])
| | ⊢  (∀i:ℕ.
| | |   i<||l1|| ⇒ [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| | |  ⇒
| | |  (list_accum(x,t.x([[t]] e1 s1 a tr);λx@0,y. x@0 = y;l1)
| | |   ⟺
| | |   list_accum(x,t.x([[t]] e1 s1 s2 a tr);λx@0,y. x@0 = y;l2))
| | | by D 0
| | |.
| | |
| | |\
| | | 1. x : SimpleType
| | | 2. ds : Collection(dec())
| | | 3. daa : Collection(dec())
| | | 4. da : Collection(SimpleType)
| | | 5. de : sig()
| | | 6. rho : Decl
| | | 7. e1 : {1of([[de]] rho)}
| | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | | 9. s1 : {[[ds]] rho}
| | | 10. s2 : {[[ds]] rho}
| | | 11. a : [[da]] rho
| | | 12. tr : trace_env([[daa]] rho)
| | | 13. l1 : Term List
| | | 14. l2 : Term List
| | | 15. ||l1|| = ||l2||
| | | 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | | 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | | 18. ||l1|| = 2
| | | 19. x ∈ term_types(ds;da;de;l1[0])
| | | 20. x ∈ term_types(ds;da;de;l1[1])
| | | 21. ||l2|| = 2
| | | 22. x ∈ term_types(ds;da;de;l2[0])
```

```
| | | 23. x ∈ term_types(ds;da;de;l2[1])
| | | 24. ∀i:ℕ.
| | | |   i<||l1|| ⇒ [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr
| | | ⊢  list_accum(x,t.x([[t]] e1 s1 a tr);λx@0,y. x@0 = y;l1)
| | | |  ⟺
| | | |  list_accum(x,t.x([[t]] e1 s1 s2 a tr);λx@0,y. x@0 = y;l2)
| | | | by InstLemma ‘list_2_decomp‘ [Term;l1] THENA Auto
| | |
| | | |    THEN HypSubstSq (-1) 0 THEN Reduce 0 THEN Thin (-1)
| | |
| | | |    THEN InstLemma ‘list_2_decomp‘ [Term;l2] THENA Auto
| | |
| | | |    THEN HypSubstSq (-1) 0 THEN Reduce 0 THEN Thin (-1)
| | | |.
| | | |
| | | 1. x : SimpleType
| | | 2. ds : Collection(dec())
| | | 3. daa : Collection(dec())
| | | 4. da : Collection(SimpleType)
| | | 5. de : sig()
| | | 6. rho : Decl
| | | 7. e1 : {1of([[de]] rho)}
| | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | | 9. s1 : {[[ds]] rho}
| | | 10. s2 : {[[ds]] rho}
| | | 11. a : [[da]] rho
| | | 12. tr : trace_env([[daa]] rho)
| | | 13. l1 : Term List
| | | 14. l2 : Term List
| | | 15. ||l1|| = ||l2||
| | | 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | | 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | | 18. ||l1|| = 2
| | | 19. x ∈ term_types(ds;da;de;l1[0])
| | | 20. x ∈ term_types(ds;da;de;l1[1])
| | | 21. ||l2|| = 2
| | | 22. x ∈ term_types(ds;da;de;l2[0])
| | | 23. x ∈ term_types(ds;da;de;l2[1])
| | | 24. ∀i:ℕ.
| | | |   i<||l1|| ⇒ [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr
| | | ⊢  [[l1[0]]] e1 s1 a tr = [[l1[1]]] e1 s1 a tr
| | | |  ⟺
| | | |  [[l2[0]]] e1 s1 s2 a tr = [[l2[1]]] e1 s1 s2 a tr  by D 0
| | | |
| | | |\
| | | | 1. x : SimpleType
| | | | 2. ds : Collection(dec())
| | | | 3. daa : Collection(dec())
| | | | 4. da : Collection(SimpleType)
| | | | 5. de : sig()
| | | | 6. rho : Decl
| | | | 7. e1 : {1of([[de]] rho)}
| | | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | | | 9. s1 : {[[ds]] rho}
| | | | 10. s2 : {[[ds]] rho}
```

```
| | | | 11. a : [[da]] rho
| | | | 12. tr : trace_env([[daa]] rho)
| | | | 13. l1 : Term List
| | | | 14. l2 : Term List
| | | | 15. ||l1|| = ||l2||
| | | | 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | | | 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | | | 18. ||l1|| = 2
| | | | 19. x ∈ term_types(ds;da;de;l1[0])
| | | | 20. x ∈ term_types(ds;da;de;l1[1])
| | | | 21. ||l2|| = 2
| | | | 22. x ∈ term_types(ds;da;de;l2[0])
| | | | 23. x ∈ term_types(ds;da;de;l2[1])
| | | | 24. ∀i:ℕ.
| | | | |   i<||l1|| ⇒ [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr
| | | | ⊢  [[l1[0]]] e1 s1 a tr = [[l1[1]]] e1 s1 a tr
| | | | |  ⇒
| | | | |  [[l2[0]]] e1 s1 s2 a tr = [[l2[1]]] e1 s1 s2 a tr  by D 0
| | | | |
| | | | |\
| | | | | 1. x : SimpleType
| | | | | 2. ds : Collection(dec())
| | | | | 3. daa : Collection(dec())
| | | | | 4. da : Collection(SimpleType)
| | | | | 5. de : sig()
| | | | | 6. rho : Decl
| | | | | 7. e1 : {1of([[de]] rho)}
| | | | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | | | | 9. s1 : {[[ds]] rho}
| | | | | 10. s2 : {[[ds]] rho}
| | | | | 11. a : [[da]] rho
| | | | | 12. tr : trace_env([[daa]] rho)
| | | | | 13. l1 : Term List
| | | | | 14. l2 : Term List
| | | | | 15. ||l1|| = ||l2||
| | | | | 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | | | | 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | | | | 18. ||l1|| = 2
| | | | | 19. x ∈ term_types(ds;da;de;l1[0])
| | | | | 20. x ∈ term_types(ds;da;de;l1[1])
| | | | | 21. ||l2|| = 2
| | | | | 22. x ∈ term_types(ds;da;de;l2[0])
| | | | | 23. x ∈ term_types(ds;da;de;l2[1])
| | | | | 24. ∀i:ℕ.
| | | | |     i<||l1||
| | | | |     ⇒
| | | | |     [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr
| | | | | 25. [[l1[0]]] e1 s1 a tr = [[l1[1]]] e1 s1 a tr
| | | | | ⊢  [[l2[0]]] e1 s1 s2 a tr = [[l2[1]]] e1 s1 s2 a tr
| | | | |   by InstHyp [0] (-2) THENA (Complete Auto’)
| | | | | THEN InstHyp [
| | | | |      1] (-3) THENA (Complete Auto’)
| | | | |
| | | | |      THEN AllHyps (\i. Unfold ‘rel_arg_typ‘ i THEN Reduce i)
| | | | |
```

```
| | | | | | THEN HypSubst (-2) (-3)
| | | | | | THENM HypSubst (-1) (-3)
| | | | | |
| | | | | |      THEN Auto ·
| | | | | |
| | | | | .\
| | | | | . .....wf .....
| | | | | . 1. x : SimpleType
| | | | | . 2. ds : Collection(dec())
| | | | | . 3. daa : Collection(dec())
| | | | | . 4. da : Collection(SimpleType)
| | | | | . 5. de : sig()
| | | | | . 6. rho : Decl
| | | | | . 7. e1 : {1of([[de]] rho)}
| | | | | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | | | | . 9. s1 : {[[ds]] rho}
| | | | | . 10. s2 : {[[ds]] rho}
| | | | | . 11. a : [[da]] rho
| | | | | . 12. tr : trace_env([[daa]] rho)
| | | | | . 13. l1 : Term List
| | | | | . 14. l2 : Term List
| | | | | . 15. ||l1|| = ||l2||
| | | | | . 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | | | | . 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | | | | . 18. ||l1|| = 2
| | | | | . 19. x ∈ term_types(ds;da;de;l1[0])
| | | | | . 20. x ∈ term_types(ds;da;de;l1[1])
| | | | | . 21. ||l2|| = 2
| | | | | . 22. x ∈ term_types(ds;da;de;l2[0])
| | | | | . 23. x ∈ term_types(ds;da;de;l2[1])
| | | | | . 24. ∀i:ℕ.
| | | | | .     i<||l1||
| | | | | .     ⇒
| | | | | .     [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr
| | | | | . ⊢  ([[l1[0]]] e1 s1 a tr = [[l1[1]]] e1 s1 a tr) ∈ Prop
| | | | | .   by ...
| | | | |
| | | | .\
| | | | . 1. x : SimpleType
| | | | . 2. ds : Collection(dec())
| | | | . 3. daa : Collection(dec())
| | | | . 4. da : Collection(SimpleType)
| | | | . 5. de : sig()
| | | | . 6. rho : Decl
| | | | . 7. e1 : {1of([[de]] rho)}
| | | | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | | | . 9. s1 : {[[ds]] rho}
| | | | . 10. s2 : {[[ds]] rho}
| | | | . 11. a : [[da]] rho
| | | | . 12. tr : trace_env([[daa]] rho)
| | | | . 13. l1 : Term List
| | | | . 14. l2 : Term List
| | | | . 15. ||l1|| = ||l2||
| | | | . 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | | | . 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
```

```
| | | . 18. ||l1|| = 2
| | | . 19. x ∈ term_types(ds;da;de;l1[0])
| | | . 20. x ∈ term_types(ds;da;de;l1[1])
| | | . 21. ||l2|| = 2
| | | . 22. x ∈ term_types(ds;da;de;l2[0])
| | | . 23. x ∈ term_types(ds;da;de;l2[1])
| | | . 24. ∀i:ℕ.
| | | . |   i<||l1|| ⇒ [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr
| | | . ⊢  ([[l1[0]]] e1 s1 a tr
| | | . |   =
| | | . |   [[l1[1]]] e1 s1 a tr) ⇐ ([[l2[0]]] e1 s1 s2 a tr
| | | . |                             =
| | | . |                             [[l2[1]]] e1 s1 s2 a tr)
| | | . | by D 0
| | | . |
| | | . |\
| | | . | 1. x : SimpleType
| | | . | 2. ds : Collection(dec())
| | | . | 3. daa : Collection(dec())
| | | . | 4. da : Collection(SimpleType)
| | | . | 5. de : sig()
| | | . | 6. rho : Decl
| | | . | 7. e1 : {1of([[de]] rho)}
| | | . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | | . | 9. s1 : {[[ds]] rho}
| | | . | 10. s2 : {[[ds]] rho}
| | | . | 11. a : [[da]] rho
| | | . | 12. tr : trace_env([[daa]] rho)
| | | . | 13. l1 : Term List
| | | . | 14. l2 : Term List
| | | . | 15. ||l1|| = ||l2||
| | | . | 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | | . | 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | | . | 18. ||l1|| = 2
| | | . | 19. x ∈ term_types(ds;da;de;l1[0])
| | | . | 20. x ∈ term_types(ds;da;de;l1[1])
| | | . | 21. ||l2|| = 2
| | | . | 22. x ∈ term_types(ds;da;de;l2[0])
| | | . | 23. x ∈ term_types(ds;da;de;l2[1])
| | | . | 24. ∀i:ℕ.
| | | . |     i<||l1||
| | | . |     ⇒
| | | . |     [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr
| | | . | 25. [[l2[0]]] e1 s1 s2 a tr = [[l2[1]]] e1 s1 s2 a tr
| | | . | ⊢  [[l1[0]]] e1 s1 a tr = [[l1[1]]] e1 s1 a tr
| | | . |   by InstHyp [0] (-2) THENA (Complete Auto’)
| | | . | THEN InstHyp [
| | | . |      1] (-3) THENA (Complete Auto’)
| | | . |
| | | . |      THEN AllHyps (\i. Unfold ‘rel_arg_typ‘ i THEN Reduce i)
| | | . |
| | | . | THEN HypSubst (-2) (-3)
| | | . | THENM HypSubst (-1) (-3)

| | | . |
| | | . |      THEN Auto ·
```

```
| | | . |
| | | . .\
| | | . . .....wf .....
| | | . . 1. x : SimpleType
| | | . . 2. ds : Collection(dec())
| | | . . 3. daa : Collection(dec())
| | | . . 4. da : Collection(SimpleType)
| | | . . 5. de : sig()
| | | . . 6. rho : Decl
| | | . . 7. e1 : {1of([[de]] rho)}
| | | . . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | | . . 9. s1 : {[[ds]] rho}
| | | . . 10. s2 : {[[ds]] rho}
| | | . . 11. a : [[da]] rho
| | | . . 12. tr : trace_env([[daa]] rho)
| | | . . 13. l1 : Term List
| | | . . 14. l2 : Term List
| | | . . 15. ||l1|| = ||l2||
| | | . . 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | | . . 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | | . . 18. ||l1|| = 2
| | | . . 19. x ∈ term_types(ds;da;de;l1[0])
| | | . . 20. x ∈ term_types(ds;da;de;l1[1])
| | | . . 21. ||l2|| = 2
| | | . . 22. x ∈ term_types(ds;da;de;l2[0])
| | | . . 23. x ∈ term_types(ds;da;de;l2[1])
| | | . . 24. ∀i:ℕ.
| | | . .     i<||l1||
| | | . .     ⇒
| | | . .     [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr
| | | . . ⊢  ([[l2[0]]] e1 s1 s2 a tr = [[l2[1]]] e1 s1 s2 a tr)
| | | . .    ∈ Prop
| | | . .   by ...
| | |
| | .\
| | . .....wf .....
| | . 1. x : SimpleType
| | . 2. ds : Collection(dec())
| | . 3. daa : Collection(dec())
| | . 4. da : Collection(SimpleType)
| | . 5. de : sig()
| | . 6. rho : Decl
| | . 7. e1 : {1of([[de]] rho)}
| | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | . 9. s1 : {[[ds]] rho}
| | . 10. s2 : {[[ds]] rho}
| | . 11. a : [[da]] rho
| | . 12. tr : trace_env([[daa]] rho)
| | . 13. l1 : Term List
| | . 14. l2 : Term List
| | . 15. ||l1|| = ||l2||
| | . 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | . 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | . 18. ||l1|| = 2
| | . 19. x ∈ term_types(ds;da;de;l1[0])
```

```
| | . 20. x ∈ term_types(ds;da;de;l1[1])
| | . 21. ||l2|| = 2
| | . 22. x ∈ term_types(ds;da;de;l2[0])
| | . 23. x ∈ term_types(ds;da;de;l2[1])
| | . ⊢  (∀i:ℕ.
| | . |   i<||l1|| ⇒ [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| | . |  ∈ Prop  by Unfold ‘rel_arg_typ‘ 0 THEN Reduce 0 THEN Auto
| | . |
| | . |\
| | . | .....subterm .....
| | . | 1. x : SimpleType
| | . | 2. ds : Collection(dec())
| | . | 3. daa : Collection(dec())
| | . | 4. da : Collection(SimpleType)
| | . | 5. de : sig()
| | . | 6. rho : Decl
| | . | 7. e1 : {1of([[de]] rho)}
| | . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | . | 9. s1 : {[[ds]] rho}
| | . | 10. {[[ds]] rho}
| | . | 11. a : [[da]] rho
| | . | 12. tr : trace_env([[daa]] rho)
| | . | 13. l1 : Term List
| | . | 14. l2 : Term List
| | . | 15. ||l1|| = ||l2||
| | . | 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | . | 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | . | 18. ||l1|| = 2
| | . | 19. x ∈ term_types(ds;da;de;l1[0])
| | . | 20. x ∈ term_types(ds;da;de;l1[1])
| | . | 21. ||l2|| = 2
| | . | 22. x ∈ term_types(ds;da;de;l2[0])
| | . | 23. x ∈ term_types(ds;da;de;l2[1])
| | . | 24. i : ℕ
| | . | 25. i<||l1||
| | . | ⊢  [[l1[i]]] e1 s1 a tr ∈ [[x]] rho
| | . | | by BackThruLemma ‘term_typing‘ THEN Auto
| | . | |
| | . | |\
| | . | | 1. x : SimpleType
| | . | | 2. ds : Collection(dec())
| | . | | 3. daa : Collection(dec())
| | . | | 4. da : Collection(SimpleType)
| | . | | 5. de : sig()
| | . | | 6. rho : Decl
| | . | | 7. {1of([[de]] rho)}
| | . | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | . | | 9. {[[ds]] rho}
| | . | | 10. {[[ds]] rho}
| | . | | 11. [[da]] rho
| | . | | 12. tr : trace_env([[daa]] rho)
| | . | | 13. l1 : Term List
| | . | | 14. l2 : Term List
| | . | | 15. ||l1|| = ||l2||
| | . | | 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
```

```
| | . | | 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | . | | 18. ||l1|| = 2
| | . | | 19. x ∈ term_types(ds;da;de;l1[0])
| | . | | 20. x ∈ term_types(ds;da;de;l1[1])
| | . | | 21. ||l2|| = 2
| | . | | 22. x ∈ term_types(ds;da;de;l2[0])
| | . | | 23. x ∈ term_types(ds;da;de;l2[1])
| | . | | 24. i : ℕ
| | . | | 25. i<||l1||
| | . | | ⊢  trace_consistent(rho;daa;tr.proj;l1[i])  by EasyHyp
| | . | |
| | . | .\
| | . | . 1. x : SimpleType
| | . | . 2. ds : Collection(dec())
| | . | . 3. daa : Collection(dec())
| | . | . 4. da : Collection(SimpleType)
| | . | . 5. de : sig()
| | . | . 6. rho : Decl
| | . | . 7. {1of([[de]] rho)}
| | . | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | . | . 9. {[[ds]] rho}
| | . | . 10. {[[ds]] rho}
| | . | . 11. [[da]] rho
| | . | . 12. tr : trace_env([[daa]] rho)
| | . | . 13. l1 : Term List
| | . | . 14. l2 : Term List
| | . | . 15. ||l1|| = ||l2||
| | . | . 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | . | . 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | . | . 18. ||l1|| = 2
| | . | . 19. x ∈ term_types(ds;da;de;l1[0])
| | . | . 20. x ∈ term_types(ds;da;de;l1[1])
| | . | . 21. ||l2|| = 2
| | . | . 22. x ∈ term_types(ds;da;de;l2[0])
| | . | . 23. x ∈ term_types(ds;da;de;l2[1])
| | . | . 24. i : ℕ
| | . | . 25. i<||l1||
| | . | . ⊢  x ∈ term_types(ds;da;de;l1[i])
| | . | .    by CaseNat 0 ‘i’ THEN Auto
| | . | . THEN CaseNat 1 ‘i’ THEN Auto’
| | . |
| | . .\
| | . . .....subterm .....
| | . . 1. x : SimpleType
| | . . 2. ds : Collection(dec())
| | . . 3. daa : Collection(dec())
| | . . 4. da : Collection(SimpleType)
| | . . 5. de : sig()
| | . . 6. rho : Decl
| | . . 7. e1 : {1of([[de]] rho)}
| | . . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | . . 9. s1 : {[[ds]] rho}
| | . . 10. s2 : {[[ds]] rho}
| | . . 11. a : [[da]] rho
| | . . 12. tr : trace_env([[daa]] rho)
```

```
| | . . 13. l1 : Term List
| | . . 14. l2 : Term List
| | . . 15. ||l1|| = ||l2||
| | . . 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | . . 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | . . 18. ||l1|| = 2
| | . . 19. x ∈ term_types(ds;da;de;l1[0])
| | . . 20. x ∈ term_types(ds;da;de;l1[1])
| | . . 21. ||l2|| = 2
| | . . 22. x ∈ term_types(ds;da;de;l2[0])
| | . . 23. x ∈ term_types(ds;da;de;l2[1])
| | . . 24. i : ℕ
| | . . 25. i<||l1||
| | . . ⊢  [[l2[i]]] e1 s1 s2 a tr ∈ [[x]] rho
| | . . | by BackThruLemma ‘term_typing2‘ THEN Auto
| | . . |
| | . . |\
| | . . | 1. x : SimpleType
| | . . | 2. ds : Collection(dec())
| | . . | 3. daa : Collection(dec())
| | . . | 4. da : Collection(SimpleType)
| | . . | 5. de : sig()
| | . . | 6. rho : Decl
| | . . | 7. {1of([[de]] rho)}
| | . . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | . . | 9. {[[ds]] rho}
| | . . | 10. {[[ds]] rho}
| | . . | 11. [[da]] rho
| | . . | 12. tr : trace_env([[daa]] rho)
| | . . | 13. l1 : Term List
| | . . | 14. l2 : Term List
| | . . | 15. ||l1|| = ||l2||
| | . . | 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | . . | 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | . . | 18. ||l1|| = 2
| | . . | 19. x ∈ term_types(ds;da;de;l1[0])
| | . . | 20. x ∈ term_types(ds;da;de;l1[1])
| | . . | 21. ||l2|| = 2
| | . . | 22. x ∈ term_types(ds;da;de;l2[0])
| | . . | 23. x ∈ term_types(ds;da;de;l2[1])
| | . . | 24. i : ℕ
| | . . | 25. i<||l1||
| | . . | ⊢  trace_consistent(rho;daa;tr.proj;l2[i])  by EasyHyp
| | . . |
| | . . .\
| | . . . 1. x : SimpleType
| | . . . 2. ds : Collection(dec())
| | . . . 3. daa : Collection(dec())
| | . . . 4. da : Collection(SimpleType)
| | . . . 5. de : sig()
| | . . . 6. rho : Decl
| | . . . 7. {1of([[de]] rho)}
| | . . . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| | . . . 9. {[[ds]] rho}
| | . . . 10. {[[ds]] rho}
```

```
| | . . . 11. [[da]] rho
| | . . . 12. tr : trace_env([[daa]] rho)
| | . . . 13. l1 : Term List
| | . . . 14. l2 : Term List
| | . . . 15. ||l1|| = ||l2||
| | . . . 16. ∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i])
| | . . . 17. ∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i])
| | . . . 18. ||l1|| = 2
| | . . . 19. x ∈ term_types(ds;da;de;l1[0])
| | . . . 20. x ∈ term_types(ds;da;de;l1[1])
| | . . . 21. ||l2|| = 2
| | . . . 22. x ∈ term_types(ds;da;de;l2[0])
| | . . . 23. x ∈ term_types(ds;da;de;l2[1])
| | . . . 24. i : ℕ
| | . . . 25. i<||l1||
| | . . . ⊢  x ∈ term_types(ds;da;de;l2[i])
| | . . .   by CaseNat 0 ‘i’ THEN Auto
| | . . . THEN CaseNat 1 ‘i’ THEN Auto’
| |
| .\
| . 1. y : Label
| . 2. ds : Collection(dec())
| . 3. daa : Collection(dec())
| . 4. da : Collection(SimpleType)
| . 5. de : sig()
| . 6. rho : Decl
| . 7. e1 : {1of([[de]] rho)}
| . 8. e2 : l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . 9. s1 : {[[ds]] rho}
| . 10. s2 : {[[ds]] rho}
| . 11. a : [[da]] rho
| . 12. tr : trace_env([[daa]] rho)
| . 13. l1 : Term List
| . 14. l2 : Term List
| . 15. ||l1|| = ||l2||
| . ⊢  (∀i:ℕ||l1||. trace_consistent(rho;daa;tr.proj;l1[i]))
| . |  ⇒
| . |  (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . |  ⇒
| . |  ||de.rel(y)|| = ||l1||
| . |  & (∀i:ℕ.
| . |     i<||l1|| ⇒ (de.rel(y))[i] ∈ term_types(ds;da;de;l1[i]))
| . |  ⇒
| . |  ||de.rel(y)|| = ||l2||
| . |  & (∀i:ℕ.
| . |     i<||l2|| ⇒ (de.rel(y))[i] ∈ term_types(ds;da;de;l2[i]))
| . |  ⇒
| . |  (∀i:ℕ.
| . |   i<||l1|| ⇒ [[l1[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . |  ⇒
| . |  (list_accum(x,t.x([[t]] e1 s1 a tr);e2.y;l1)
| . |   ⟺
| . |   list_accum(x,t.x([[t]] e1 s1 s2 a tr);e2.y;l2))
| . | by Decide ||de.rel(y)|| = ||l1|| THENA Auto
| .
```

```
| . |    ....
| . THEN GenConcl e2.y = f
| . |     THENA (Auto THEN Unfold `r_select` 0 THEN Auto)
| .
| . |    THEN Thin (-1) THEN MoveToConcl (-1)
| . THEN MoveToConcl (-1)
| .
| . |    THEN Unfold `rel_arg_typ` 0 THEN Reduce 0
| . THEN GenConcl
| . |    de.rel(y) = ls THENA Auto
| . THEN Thin (-1)
| .
| . |    THEN RepeatFor 3 (MoveToConcl (-1))
| . THEN ListInd (-1)
| .
| . |    THEN D 0 THENA Auto
| . THEN ListInd (-1)
| .
| . |    THEN Reduce 0 THEN D 0 THENA Auto'
| .
| . |    THENL [Id; Auto'; Auto'; Id]
| . THEN D 0 THENA Auto
| .

| . |    THEN ListInd (-1) THEN Reduce 0
| .
| . |    THEN D 0 THENA Complete Auto'
| . THEN Try (AssertBY False
| . |     (Complete Auto') THEN Auto)
| .
| . |    THEN Repeat (D 0 THENA Complete Auto')
| .
| . |    THEN Try (Complete Auto)
| . .
| . |
| . 1. Label
| . 2. ds : Collection(dec())
| . 3. daa : Collection(dec())
| . 4. da : Collection(SimpleType)
| . 5. de : sig()
| . 6. rho : Decl
| . 7. e1 : {1of([[de]] rho)}
| . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . 9. s1 : {[[ds]] rho}
| . 10. s2 : {[[ds]] rho}
| . 11. a : [[da]] rho
| . 12. tr : trace_env([[daa]] rho)
| . 13. Term List
| . 14. u : Term
| . 15. v : Term List
| . 16. ∀l2:Term List.
| . |   ||v|| = ||l2||
| . |   ⇒
| . |   (∀ls:SimpleType List.
| . |    ||ls|| = ||v||
| . |    ⇒
```

```
| . |     (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . |      (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . |      ⇒
| . |      (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . |      ⇒
| . |      ||ls|| = ||v||
| . |      & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . |      ⇒
| . |      ||ls|| = ||l2||
| . |      & (∀i:ℕ. i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . |      ⇒
| . |      (∀i:ℕ.
| . |       i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . |      ⇒
| . |      (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . |       ⟺
| . |       list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . 17. Term List
| . 18. u1 : Term
| . 19. v1 : Term List
| . 20. ||[u / v]|| = ||v1||
| . |   ⇒
| . |   (∀ls:SimpleType List.
| . |    ||ls|| = ||[u / v]||
| . |    ⇒
| . |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . |     (∀i:ℕ||[u / v]||.
| . |      trace_consistent(rho;daa;tr.proj;[u / v][i]))
| . |     ⇒
| . |     (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . |     ⇒
| . |     ||ls|| = ||[u / v]||
| . |     & (∀i:ℕ.
| . |         i<||[u / v]|| ⇒ ls[i] ∈ term_types(ds;da;de;[u / v][i]))
| . |     ⇒
| . |     ||ls|| = ||v1||
| . |     & (∀i:ℕ. i<||v1|| ⇒ ls[i] ∈ term_types(ds;da;de;v1[i]))
| . |     ⇒
| . |     (∀i:ℕ.
| . |      i<||[u / v]||
| . |      ⇒
| . |      [[[u / v][i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . |     ⇒
| . |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;[u / v])
| . |      ⟺
| . |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))))
| . 21. ||v||+1 = ||v1||+1
| . 22. SimpleType List
| . 23. u2 : SimpleType
| . 24. v2 : SimpleType List
| . 25. ||v2|| = ||v||+1
| . |   ⇒
| . |   (∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . |    (∀i:ℕ(||v||+1). trace_consistent(rho;daa;tr.proj;[u / v][i]))
| . |    ⇒
```

```
| . |     (∀i:ℕ(||v1||+1).
| . |      trace_consistent(rho;daa;tr.proj;[u1 / v1][i]))
| . |     ⇒
| . |     ||v2|| = ||v||+1
| . |     & (∀i:ℕ.
| . |        i<||v||+1 ⇒ v2[i] ∈ term_types(ds;da;de;[u / v][i]))
| . |     ⇒
| . |     ||v2|| = ||v1||+1
| . |     & (∀i:ℕ.
| . |        i<||v1||+1 ⇒ v2[i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . |     ⇒
| . |     (∀i:ℕ.
| . |      i<||v||+1
| . |      ⇒
| . |      [[[u / v][i]]] e1 s1 a tr = [[[u1 / v1][i]]] e1 s1 s2 a tr)
| . |     ⇒
| . |     (list_accum(x,t.x([[t]] e1 s1 a tr);f([[u]] e1 s1 a tr);v)
| . |      ⟺
| . |      list_accum(x,t.x
| . |                      ([[t]] e1 s1 s2 a tr);...;v1)))
| . 26. ||v2||+1 = ||v||+1
| . 27. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . 28. ∀i:ℕ(||v||+1). trace_consistent(rho;daa;tr.proj;[u / v][i])
| . 29. ∀i:ℕ(||v1||+1). trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . 30. ||v2||+1 = ||v||+1
| . |   & (∀i:ℕ.
| . |      i<||v||+1
| . |      ⇒
| . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . 31. ||v2||+1 = ||v1||+1
| . |   & (∀i:ℕ.
| . |      i<||v1||+1
| . |      ⇒
| . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . ⊢  (∀i:ℕ.
| . |   i<||v||+1
| . |   ⇒
| . |   [[[u / v][i]]] e1 s1 a tr = [[[u1 / v1][i]]] e1 s1 s2 a tr)
| . |  ⇒
| . |  (list_accum(x,t.x([[t]] e1 s1 a tr);f([[u]] e1 s1 a tr);v)
| . |   ⟺
| . |   list_accum(x,t.x
| . |                   ([[t]] e1 s1 s2 a tr);...;v1))
| . | by Thin (-7) THEN Thin (-11) THEN D 0 THENA Auto·
| . |
| . |\
| . | 1. Label
| . | 2. ds : Collection(dec())
| . | 3. daa : Collection(dec())
| . | 4. da : Collection(SimpleType)
| . | 5. de : sig()
| . | 6. rho : Decl
| . | 7. e1 : {1of([[de]] rho)}
| . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | 9. s1 : {[[ds]] rho}
```

```
| . | 10. s2 : {[[ds]] rho}
| . | 11. a : [[da]] rho
| . | 12. tr : trace_env([[daa]] rho)
| . | 13. Term List
| . | 14. u : Term
| . | 15. v : Term List
| . | 16. ∀l2:Term List.
| . | |   ||v|| = ||l2||
| . | |   ⇒
| . | |   (∀ls:SimpleType List.
| . | |    ||ls|| = ||v||
| . | |    ⇒
| . | |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | |     ⇒
| . | |     (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | |     ⇒
| . | |     ||ls|| = ||v||
| . | |     & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | |     ⇒
| . | |     ||ls|| = ||l2||
| . | |     & (∀i:ℕ. i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | |     ⇒
| . | |     (∀i:ℕ.
| . | |      i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | |     ⇒
| . | |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | |      ⟺
| . | |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | 17. Term List
| . | 18. u1 : Term
| . | 19. v1 : Term List
| . | 20. ||v||+1 = ||v1||+1
| . | 21. SimpleType List
| . | 22. u2 : SimpleType
| . | 23. v2 : SimpleType List
| . | 24. ||v2||+1 = ||v||+1
| . | 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | 26. ∀i:ℕ(||v||+1). trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | 27. ∀i:ℕ(||v1||+1).
| . | |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | 28. ||v2||+1 = ||v||+1
| . | |   & (∀i:ℕ.
| . | |      i<||v||+1
| . | |      ⇒
| . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | 29. ||v2||+1 = ||v1||+1
| . | |   & (∀i:ℕ.
| . | |      i<||v1||+1
| . | |      ⇒
| . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | 30. ∀i:ℕ.
| . | |   i<||v||+1
| . | |   ⇒
| . | |   [[[u / v][i]]] e1 s1 a tr = [[[u1 / v1][i]]] e1 s1 s2 a tr
```

```
| . | ⊢  list_accum(x,t.x([[t]] e1 s1 a tr);f([[u]] e1 s1 a tr);v)
| . | |  ⟺
| . | |  list_accum(x,t.x
| . | |                ([[t]] e1 s1 s2 a tr);...;v1)
| . | | by Assert (∀i:ℕ.
| . | |            i<||v||
| . | |            ⇒
| . | |            [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | |    & (∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | |    & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | |
| . | |\
| . | | .....assertion .....
| . | | 1. Label
| . | | 2. ds : Collection(dec())
| . | | 3. daa : Collection(dec())
| . | | 4. da : Collection(SimpleType)
| . | | 5. de : sig()
| . | | 6. rho : Decl
| . | | 7. e1 : {1of([[de]] rho)}
| . | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | | 9. s1 : {[[ds]] rho}
| . | | 10. s2 : {[[ds]] rho}
| . | | 11. a : [[da]] rho
| . | | 12. tr : trace_env([[daa]] rho)
| . | | 13. Term List
| . | | 14. u : Term
| . | | 15. v : Term List
| . | | 16. ∀l2:Term List.
| . | | |   ||v|| = ||l2||
| . | | |   ⇒
| . | | |   (∀ls:SimpleType List.
| . | | |    ||ls|| = ||v||
| . | | |    ⇒
| . | | |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | | |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | | |     ⇒
| . | | |     (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | | |     ⇒
| . | | |     ||ls|| = ||v||
| . | | |     & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | | |     ⇒
| . | | |     ||ls|| = ||l2||
| . | | |     & (∀i:ℕ. i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | | |     ⇒
| . | | |     (∀i:ℕ.
| . | | |      i<||v||
| . | | |      ⇒
| . | | |      [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | | |     ⇒
| . | | |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | | |      ⟺
| . | | |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | | 17. Term List
| . | | 18. u1 : Term
```

```
| . | | 19. v1 : Term List
| . | | 20. ||v||+1 = ||v1||+1
| . | | 21. SimpleType List
| . | | 22. u2 : SimpleType
| . | | 23. v2 : SimpleType List
| . | | 24. ||v2||+1 = ||v||+1
| . | | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | | 26. ∀i:ℕ(||v||+1).
| . | | |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | | 27. ∀i:ℕ(||v1||+1).
| . | | |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | | 28. ||v2||+1 = ||v||+1
| . | | |   & (∀i:ℕ.
| . | | |      i<||v||+1
| . | | |      ⇒
| . | | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | | 29. ||v2||+1 = ||v1||+1
| . | | |   & (∀i:ℕ.
| . | | |      i<||v1||+1
| . | | |      ⇒
| . | | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | | 30. ∀i:ℕ.
| . | | |   i<||v||+1
| . | | |   ⇒
| . | | |   [[[u / v][i]]] e1 s1 a tr = [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | | ⊢  (∀i:ℕ.
| . | | |   i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | | |  & (∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | | |  & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | | | by Auto
| . | | |
| . | | |\
| . | | | 1. Label
| . | | | 2. ds : Collection(dec())
| . | | | 3. daa : Collection(dec())
| . | | | 4. da : Collection(SimpleType)
| . | | | 5. de : sig()
| . | | | 6. rho : Decl
| . | | | 7. e1 : {1of([[de]] rho)}
| . | | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | | | 9. s1 : {[[ds]] rho}
| . | | | 10. s2 : {[[ds]] rho}
| . | | | 11. a : [[da]] rho
| . | | | 12. tr : trace_env([[daa]] rho)
| . | | | 13. Term List
| . | | | 14. u : Term
| . | | | 15. v : Term List
| . | | | 16. ∀l2:Term List.
| . | | | |   ||v|| = ||l2||
| . | | | |   ⇒
| . | | | |   (∀ls:SimpleType List.
| . | | | |    ||ls|| = ||v||
| . | | | |    ⇒
| . | | | |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | | | |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
```

```
| . | | | |      ⇒
| . | | | |      (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | | | |      ⇒
| . | | | |      ||ls|| = ||v||
| . | | | |      & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | | | |      ⇒
| . | | | |      ||ls|| = ||l2||
| . | | | |      & (∀i:ℕ.
| . | | | |         i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | | | |      ⇒
| . | | | |      (∀i:ℕ.
| . | | | |       i<||v||
| . | | | |       ⇒
| . | | | |       [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | | | |      ⇒
| . | | | |      (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | | | |       ⟺
| . | | | |       list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | | | 17. Term List
| . | | | 18. u1 : Term
| . | | | 19. v1 : Term List
| . | | | 20. ||v||+1 = ||v1||+1
| . | | | 21. SimpleType List
| . | | | 22. u2 : SimpleType
| . | | | 23. v2 : SimpleType List
| . | | | 24. ||v2||+1 = ||v||+1
| . | | | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | | | 26. ∀i:ℕ(||v||+1).
| . | | | |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | | | 27. ∀i:ℕ(||v1||+1).
| . | | | |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | | | 28. ||v2||+1 = ||v||+1
| . | | | |   & (∀i:ℕ.
| . | | | |      i<||v||+1
| . | | | |      ⇒
| . | | | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | | | 29. ||v2||+1 = ||v1||+1
| . | | | |   & (∀i:ℕ.
| . | | | |      i<||v1||+1
| . | | | |      ⇒
| . | | | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | | | 30. ∀i:ℕ.
| . | | | |   i<||v||+1
| . | | | |   ⇒
| . | | | |   [[[u / v][i]]] e1 s1 a tr
| . | | | |   =
| . | | | |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | | | 31. i : ℕ
| . | | | 32. i<||v||
| . | | | ⊢  [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | | | | by InstHyp [i+1] (-3) THENA Auto'
| . | | | |
| . | | | 1. Label
| . | | | 2. ds : Collection(dec())
| . | | | 3. daa : Collection(dec())
```

```
| . | | | 4. da : Collection(SimpleType)
| . | | | 5. de : sig()
| . | | | 6. rho : Decl
| . | | | 7. e1 : {1of([[de]] rho)}
| . | | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | | | 9. s1 : {[[ds]] rho}
| . | | | 10. s2 : {[[ds]] rho}
| . | | | 11. a : [[da]] rho
| . | | | 12. tr : trace_env([[daa]] rho)
| . | | | 13. Term List
| . | | | 14. u : Term
| . | | | 15. v : Term List
| . | | | 16. ∀l2:Term List.
| . | | |     ||v|| = ||l2||
| . | | |     ⇒
| . | | |     (∀ls:SimpleType List.
| . | | |      ||ls|| = ||v||
| . | | |      ⇒
| . | | |      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | | |       (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | | |       ⇒
| . | | |       (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | | |       ⇒
| . | | |       ||ls|| = ||v||
| . | | |       & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | | |       ⇒
| . | | |       ||ls|| = ||l2||
| . | | |       & (∀i:ℕ.
| . | | |          i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | | |       ⇒
| . | | |       (∀i:ℕ.
| . | | |        i<||v||
| . | | |        ⇒
| . | | |        [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | | |       ⇒
| . | | |       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | | |        ⟺
| . | | |        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | | | 17. Term List
| . | | | 18. u1 : Term
| . | | | 19. v1 : Term List
| . | | | 20. ||v||+1 = ||v1||+1
| . | | | 21. SimpleType List
| . | | | 22. u2 : SimpleType
| . | | | 23. v2 : SimpleType List
| . | | | 24. ||v2||+1 = ||v||+1
| . | | | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | | | 26. ∀i:ℕ(||v||+1).
| . | | |     trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | | | 27. ∀i:ℕ(||v1||+1).
| . | | |     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | | | 28. ||v2||+1 = ||v||+1
| . | | |     & (∀i:ℕ.
| . | | |        i<||v||+1
| . | | |        ⇒
```

```
| . | | |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | | | 29. ||v2||+1 = ||v1||+1
| . | | |     & (∀i:ℕ.
| . | | |        i<||v1||+1
| . | | |        ⇒
| . | | |        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | | | 30. ∀i:ℕ.
| . | | |     i<||v||+1
| . | | |     ⇒
| . | | |     [[[u / v][i]]] e1 s1 a tr
| . | | |     =
| . | | |     [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | | | 31. i : ℕ
| . | | | 32. i<||v||
| . | | | 33. [[[u / v][(i+1)]]] e1 s1 a tr
| . | | |     =
| . | | |     [[[u1 / v1][(i+1)]]] e1 s1 s2 a tr
| . | | | ⊢  [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | | |   by Subst [u2 / v2][(i+1)] ∼ v2[i] (-1)
| . | | | THENM Subst
| . | | |      [u1 / v1][(i+1)] ∼ v1[i] (-1)
| . | | | THENM Subst [u / v][(i
| . | | |      +1)] ∼ v[i] (-1)
| . | | | THEN Auto
| . | | | THEN Try (
| . | | |      RWW "select_cons_tl" 0 THEN Auto)
| . | | |
| . | | |\
| . | | | 1. Label
| . | | | 2. ds : Collection(dec())
| . | | | 3. daa : Collection(dec())
| . | | | 4. da : Collection(SimpleType)
| . | | | 5. de : sig()
| . | | | 6. rho : Decl
| . | | | 7. e1 : {1of([[de]] rho)}
| . | | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | | | 9. s1 : {[[ds]] rho}
| . | | | 10. s2 : {[[ds]] rho}
| . | | | 11. a : [[da]] rho
| . | | | 12. tr : trace_env([[daa]] rho)
| . | | | 13. Term List
| . | | | 14. u : Term
| . | | | 15. v : Term List
| . | | | 16. ∀l2:Term List.
| . | | | |   ||v|| = ||l2||
| . | | | |   ⇒
| . | | | |   (∀ls:SimpleType List.
| . | | | |    ||ls|| = ||v||
| . | | | |    ⇒
| . | | | |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | | | |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | | | |     ⇒
| . | | | |     (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | | | |     ⇒
| . | | | |     ||ls|| = ||v||
```

```
| . | | | |      & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | | | |      ⇒
| . | | | |      ||ls|| = ||l2||
| . | | | |      & (∀i:ℕ.
| . | | | |         i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | | | |      ⇒
| . | | | |      (∀i:ℕ.
| . | | | |       i<||v||
| . | | | |       ⇒
| . | | | |       [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | | | |      ⇒
| . | | | |      (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | | | |       ⟺
| . | | | |       list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | | | 17. Term List
| . | | | 18. u1 : Term
| . | | | 19. v1 : Term List
| . | | | 20. ||v||+1 = ||v1||+1
| . | | | 21. SimpleType List
| . | | | 22. u2 : SimpleType
| . | | | 23. v2 : SimpleType List
| . | | | 24. ||v2||+1 = ||v||+1
| . | | | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | | | 26. ∀i:ℕ(||v||+1).
| . | | | |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | | | 27. ∀i:ℕ(||v1||+1).
| . | | | |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | | | 28. ||v2||+1 = ||v||+1
| . | | | |   & (∀i:ℕ.
| . | | | |      i<||v||+1
| . | | | |      ⇒
| . | | | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | | | 29. ||v2||+1 = ||v1||+1
| . | | | |   & (∀i:ℕ.
| . | | | |      i<||v1||+1
| . | | | |      ⇒
| . | | | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | | | 30. ∀i:ℕ.
| . | | | |   i<||v||+1
| . | | | |   ⇒
| . | | | |   [[[u / v][i]]] e1 s1 a tr
| . | | | |   =
| . | | | |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | | | 31. i : ℕ
| . | | | 32. i<||v1||
| . | | | ⊢  v2[i] ∈ term_types(ds;da;de;v1[i])
| . | | | | by D (-4) THEN
| . | | | InstHyp [i+1] (-4) THENA Auto'
| . | | | |
| . | | | 1. Label
| . | | | 2. ds : Collection(dec())
| . | | | 3. daa : Collection(dec())
| . | | | 4. da : Collection(SimpleType)
| . | | | 5. de : sig()
| . | | | 6. rho : Decl
```

```
| . | | | 7. e1 : {1of([[de]] rho)}
| . | | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | | | 9. s1 : {[[ds]] rho}
| . | | | 10. s2 : {[[ds]] rho}
| . | | | 11. a : [[da]] rho
| . | | | 12. tr : trace_env([[daa]] rho)
| . | | | 13. Term List
| . | | | 14. u : Term
| . | | | 15. v : Term List
| . | | | 16. ∀l2:Term List.
| . | | | |   ||v|| = ||l2||
| . | | | |   ⇒
| . | | | |   (∀ls:SimpleType List.
| . | | | |    ||ls|| = ||v||
| . | | | |    ⇒
| . | | | |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | | | |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | | | |     ⇒
| . | | | |     (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | | | |     ⇒
| . | | | |     ||ls|| = ||v||
| . | | | |     & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | | | |     ⇒
| . | | | |     ||ls|| = ||l2||
| . | | | |     & (∀i:ℕ.
| . | | | |        i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | | | |     ⇒
| . | | | |     (∀i:ℕ.
| . | | | |      i<||v||
| . | | | |      ⇒
| . | | | |      [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | | | |     ⇒
| . | | | |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | | | |      ⟺
| . | | | |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | | | 17. Term List
| . | | | 18. u1 : Term
| . | | | 19. v1 : Term List
| . | | | 20. ||v||+1 = ||v1||+1
| . | | | 21. SimpleType List
| . | | | 22. u2 : SimpleType
| . | | | 23. v2 : SimpleType List
| . | | | 24. ||v2||+1 = ||v||+1
| . | | | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | | | 26. ∀i:ℕ(||v||+1).
| . | | | |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | | | 27. ∀i:ℕ(||v1||+1).
| . | | | |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | | | 28. ||v2||+1 = ||v||+1
| . | | | |   & (∀i:ℕ.
| . | | | |      i<||v||+1
| . | | | |      ⇒
| . | | | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | | | 29. ||v2||+1 = ||v1||+1
| . | | | 30. ∀i:ℕ.
```

```
| . | | | |   i<||v1||+1
| . | | | |   ⇒
| . | | | |   [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i])
| . | | | 31. ∀i:ℕ.
| . | | | |   i<||v||+1
| . | | | |   ⇒
| . | | | |   [[[u / v][i]]] e1 s1 a tr
| . | | | |   =
| . | | | |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | | | 32. i : ℕ
| . | | | 33. i<||v1||
| . | | | 34. [u2 / v2][(i+1)] ∈ term_types(ds;da;de;[u1 / v1][(i+1)])
| . | | | ⊢  v2[i] ∈ term_types(ds;da;de;v1[i])
| . | | | | by RWO "select_cons_tl" (-1) THEN Auto
| . | | | |
| . | | | 1. Label
| . | | | 2. ds : Collection(dec())
| . | | | 3. daa : Collection(dec())
| . | | | 4. da : Collection(SimpleType)
| . | | | 5. de : sig()
| . | | | 6. rho : Decl
| . | | | 7. e1 : {1of([[de]] rho)}
| . | | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | | | 9. s1 : {[[ds]] rho}
| . | | | 10. s2 : {[[ds]] rho}
| . | | | 11. a : [[da]] rho
| . | | | 12. tr : trace_env([[daa]] rho)
| . | | | 13. Term List
| . | | | 14. u : Term
| . | | | 15. v : Term List
| . | | | 16. ∀l2:Term List.
| . | | |     ||v|| = ||l2||
| . | | |     ⇒
| . | | |     (∀ls:SimpleType List.
| . | | |      ||ls|| = ||v||
| . | | |      ⇒
| . | | |      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | | |       (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | | |       ⇒
| . | | |       (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | | |       ⇒
| . | | |       ||ls|| = ||v||
| . | | |       & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | | |       ⇒
| . | | |       ||ls|| = ||l2||
| . | | |       & (∀i:ℕ.
| . | | |           i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | | |       ⇒
| . | | |       (∀i:ℕ.
| . | | |        i<||v||
| . | | |        ⇒
| . | | |        [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | | |       ⇒
| . | | |       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | | |        ⟺
```

```
| . | | |        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | | | 17. Term List
| . | | | 18. u1 : Term
| . | | | 19. v1 : Term List
| . | | | 20. ||v||+1 = ||v1||+1
| . | | | 21. SimpleType List
| . | | | 22. u2 : SimpleType
| . | | | 23. v2 : SimpleType List
| . | | | 24. ||v2||+1 = ||v||+1
| . | | | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | | | 26. ∀i:ℕ(||v||+1).
| . | | |     trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | | | 27. ∀i:ℕ(||v1||+1).
| . | | |     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | | | 28. ||v2||+1 = ||v||+1
| . | | |     & (∀i:ℕ.
| . | | |        i<||v||+1
| . | | |        ⇒
| . | | |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | | | 29. ||v2||+1 = ||v1||+1
| . | | | 30. ∀i:ℕ.
| . | | |     i<||v1||+1
| . | | |     ⇒
| . | | |     [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i])
| . | | | 31. ∀i:ℕ.
| . | | |     i<||v||+1
| . | | |     ⇒
| . | | |     [[[u / v][i]]] e1 s1 a tr
| . | | |     =
| . | | |     [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | | | 32. i : ℕ
| . | | | 33. i<||v1||
| . | | | 34. v2[(i+1-1)] ∈ term_types(ds;da;de;v1[(i+1-1)])
| . | | | ⊢  v2[i] ∈ term_types(ds;da;de;v1[i])
| . | | |   by ArithSimp (-1) THEN Auto
| . | | |
| . | | .\
| . | | . 1. Label
| . | | . 2. ds : Collection(dec())
| . | | . 3. daa : Collection(dec())
| . | | . 4. da : Collection(SimpleType)
| . | | . 5. de : sig()
| . | | . 6. rho : Decl
| . | | . 7. e1 : {1of([[de]] rho)}
| . | | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | | . 9. s1 : {[[ds]] rho}
| . | | . 10. s2 : {[[ds]] rho}
| . | | . 11. a : [[da]] rho
| . | | . 12. tr : trace_env([[daa]] rho)
| . | | . 13. Term List
| . | | . 14. u : Term
| . | | . 15. v : Term List
| . | | . 16. ∀l2:Term List.
| . | | . |   ||v|| = ||l2||
| . | | . |   ⇒
```

```
| . | | . |   (∀ls:SimpleType List.
| . | | . |    ||ls|| = ||v||
| . | | . |    ⇒
| . | | . |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | | . |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | | . |     ⇒
| . | | . |     (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | | . |     ⇒
| . | | . |     ||ls|| = ||v||
| . | | . |     & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | | . |     ⇒
| . | | . |     ||ls|| = ||l2||
| . | | . |     & (∀i:ℕ.
| . | | . |        i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | | . |     ⇒
| . | | . |     (∀i:ℕ.
| . | | . |      i<||v||
| . | | . |      ⇒
| . | | . |      [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | | . |     ⇒
| . | | . |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | | . |      ⟺
| . | | . |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | | . 17. Term List
| . | | . 18. u1 : Term
| . | | . 19. v1 : Term List
| . | | . 20. ||v||+1 = ||v1||+1
| . | | . 21. SimpleType List
| . | | . 22. u2 : SimpleType
| . | | . 23. v2 : SimpleType List
| . | | . 24. ||v2||+1 = ||v||+1
| . | | . 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | | . 26. ∀i:ℕ(||v||+1).
| . | | . |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | | . 27. ∀i:ℕ(||v1||+1).
| . | | . |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | | . 28. ||v2||+1 = ||v||+1
| . | | . |   & (∀i:ℕ.
| . | | . |      i<||v||+1
| . | | . |      ⇒
| . | | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | | . 29. ||v2||+1 = ||v1||+1
| . | | . |   & (∀i:ℕ.
| . | | . |      i<||v1||+1
| . | | . |      ⇒
| . | | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | | . 30. ∀i:ℕ.
| . | | . |   i<||v||+1
| . | | . |   ⇒
| . | | . |   [[[u / v][i]]] e1 s1 a tr
| . | | . |   =
| . | | . |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | | . 31. i : ℕ
| . | | . 32. i<||v||
| . | | . ⊢  v2[i] ∈ term_types(ds;da;de;v[i])
```

```
| . | | . | by D (-5) THEN InstHyp [i+1] (-5) THENA Auto'·
| . | | . |
| . | | . 1. Label
| . | | . 2. ds : Collection(dec())
| . | | . 3. daa : Collection(dec())
| . | | . 4. da : Collection(SimpleType)
| . | | . 5. de : sig()
| . | | . 6. rho : Decl
| . | | . 7. e1 : {1of([[de]] rho)}
| . | | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | | . 9. s1 : {[[ds]] rho}
| . | | . 10. s2 : {[[ds]] rho}
| . | | . 11. a : [[da]] rho
| . | | . 12. tr : trace_env([[daa]] rho)
| . | | . 13. Term List
| . | | . 14. u : Term
| . | | . 15. v : Term List
| . | | . 16. ∀l2:Term List.
| . | | . |    ||v|| = ||l2||
| . | | . |    ⇒
| . | | . |    (∀ls:SimpleType List.
| . | | . |     ||ls|| = ||v||
| . | | . |     ⇒
| . | | . |     (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | | . |      (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | | . |      ⇒
| . | | . |      (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | | . |      ⇒
| . | | . |      ||ls|| = ||v||
| . | | . |      & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | | . |      ⇒
| . | | . |      ||ls|| = ||l2||
| . | | . |      & (∀i:ℕ.
| . | | . |         i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | | . |      ⇒
| . | | . |      (∀i:ℕ.
| . | | . |       i<||v||
| . | | . |       ⇒
| . | | . |       [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | | . |      ⇒
| . | | . |      (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | | . |       ⟺
| . | | . |       list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | | . 17. Term List
| . | | . 18. u1 : Term
| . | | . 19. v1 : Term List
| . | | . 20. ||v||+1 = ||v1||+1
| . | | . 21. SimpleType List
| . | | . 22. u2 : SimpleType
| . | | . 23. v2 : SimpleType List
| . | | . 24. ||v2||+1 = ||v||+1
| . | | . 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | | . 26. ∀i:ℕ(||v||+1).
| . | | . |    trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | | . 27. ∀i:ℕ(||v1||+1).
```

```
| . | | . |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | | . 28. ||v2||+1 = ||v||+1
| . | | . 29. ∀i:ℕ.
| . | | . |   i<||v||+1
| . | | . |   ⇒
| . | | . |   [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i])
| . | | . 30. ||v2||+1 = ||v1||+1
| . | | . |   & (∀i:ℕ.
| . | | . |      i<||v1||+1
| . | | . |      ⇒
| . | | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | | . 31. ∀i:ℕ.
| . | | . |   i<||v||+1
| . | | . |   ⇒
| . | | . |   [[[u / v][i]]] e1 s1 a tr
| . | | . |   =
| . | | . |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | | . 32. i : ℕ
| . | | . 33. i<||v||
| . | | . 34. [u2 / v2][(i+1)] ∈ term_types(ds;da;de;[u / v][(i+1)])
| . | | . ⊢  v2[i] ∈ term_types(ds;da;de;v[i])
| . | | . | by RWO "select_cons_tl" (-1) THEN Auto
| . | | . |
| . | | . 1. Label
| . | | . 2. ds : Collection(dec())
| . | | . 3. daa : Collection(dec())
| . | | . 4. da : Collection(SimpleType)
| . | | . 5. de : sig()
| . | | . 6. rho : Decl
| . | | . 7. e1 : {1of([[de]] rho)}
| . | | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | | . 9. s1 : {[[ds]] rho}
| . | | . 10. s2 : {[[ds]] rho}
| . | | . 11. a : [[da]] rho
| . | | . 12. tr : trace_env([[daa]] rho)
| . | | . 13. Term List
| . | | . 14. u : Term
| . | | . 15. v : Term List
| . | | . 16. ∀l2:Term List.
| . | | .     ||v|| = ||l2||
| . | | .     ⇒
| . | | .     (∀ls:SimpleType List.
| . | | .      ||ls|| = ||v||
| . | | .      ⇒
| . | | .      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | | .       (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | | .       ⇒
| . | | .       (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | | .       ⇒
| . | | .       ||ls|| = ||v||
| . | | .       & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | | .       ⇒
| . | | .       ||ls|| = ||l2||
| . | | .       & (∀i:ℕ.
| . | | .          i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
```

```
| . | | .        ⇒
| . | | .        (∀i:ℕ.
| . | | .         i<||v||
| . | | .         ⇒
| . | | .         [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | | .        ⇒
| . | | .        (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | | .         ⟺
| . | | .         list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | | . 17. Term List
| . | | . 18. u1 : Term
| . | | . 19. v1 : Term List
| . | | . 20. ||v||+1 = ||v1||+1
| . | | . 21. SimpleType List
| . | | . 22. u2 : SimpleType
| . | | . 23. v2 : SimpleType List
| . | | . 24. ||v2||+1 = ||v||+1
| . | | . 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | | . 26. ∀i:ℕ(||v||+1).
| . | | .     trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | | . 27. ∀i:ℕ(||v1||+1).
| . | | .     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | | . 28. ||v2||+1 = ||v||+1
| . | | . 29. ∀i:ℕ.
| . | | .     i<||v||+1
| . | | .     ⇒
| . | | .     [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i])
| . | | . 30. ||v2||+1 = ||v1||+1
| . | | .     & (∀i:ℕ.
| . | | .        i<||v1||+1
| . | | .        ⇒
| . | | .        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | | . 31. ∀i:ℕ.
| . | | .     i<||v||+1
| . | | .     ⇒
| . | | .     [[[u / v][i]]] e1 s1 a tr
| . | | .     =
| . | | .     [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | | . 32. i : ℕ
| . | | . 33. i<||v||
| . | | . 34. v2[(i+1-1)] ∈ term_types(ds;da;de;v[(i+1-1)])
| . | | . ⊢  v2[i] ∈ term_types(ds;da;de;v[i])
| . | | .   by ArithSimp (-1) THEN Auto

| . | |
| . | .\
| . | . 1. Label
| . | . 2. ds : Collection(dec())
| . | . 3. daa : Collection(dec())
| . | . 4. da : Collection(SimpleType)
| . | . 5. de : sig()
| . | . 6. rho : Decl
| . | . 7. e1 : {1of([[de]] rho)}
| . | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . 9. s1 : {[[ds]] rho}
| . | . 10. s2 : {[[ds]] rho}
```

```
| . | . 11. a : [[da]] rho
| . | . 12. tr : trace_env([[daa]] rho)
| . | . 13. Term List
| . | . 14. u : Term
| . | . 15. v : Term List
| . | . 16. ∀l2:Term List.
| . | . |   ||v|| = ||l2||
| . | . |   ⇒
| . | . |   (∀ls:SimpleType List.
| . | . |    ||ls|| = ||v||
| . | . |    ⇒
| . | . |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . |     ⇒
| . | . |     (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . |     ⇒
| . | . |     ||ls|| = ||v||
| . | . |     & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . |     ⇒
| . | . |     ||ls|| = ||l2||
| . | . |     & (∀i:ℕ. i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . |     ⇒
| . | . |     (∀i:ℕ.
| . | . |      i<||v||
| . | . |      ⇒
| . | . |      [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . |     ⇒
| . | . |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . |      ⟺
| . | . |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . 17. Term List
| . | . 18. u1 : Term
| . | . 19. v1 : Term List
| . | . 20. ||v||+1 = ||v1||+1
| . | . 21. SimpleType List
| . | . 22. u2 : SimpleType
| . | . 23. v2 : SimpleType List
| . | . 24. ||v2||+1 = ||v||+1
| . | . 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . 26. ∀i:ℕ(||v||+1).
| . | . |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . 27. ∀i:ℕ(||v1||+1).
| . | . |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . 28. ||v2||+1 = ||v||+1
| . | . |   & (∀i:ℕ.
| . | . |      i<||v||+1
| . | . |      ⇒
| . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . 29. ||v2||+1 = ||v1||+1
| . | . |   & (∀i:ℕ.
| . | . |      i<||v1||+1
| . | . |      ⇒
| . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . 30. ∀i:ℕ.
| . | . |   i<||v||+1
```

```
| . | . |    ⇒
| . | . |    [[[u / v][i]]] e1 s1 a tr = [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . 31. (∀i:ℕ.
| . | . |     i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . |    & (∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . |    & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . ⊢  list_accum(x,t.x([[t]] e1 s1 a tr);f([[u]] e1 s1 a tr);v)
| . | . |  ⟺
| . | . |  list_accum(x,t.x
| . | . |                 ([[t]] e1 s1 s2 a tr);...;v1)
| . | . | by D (-1) THEN D (-2) THEN AllHyps (\i. InstHyp [v1;v2
| . | . |     ...
| . | . THEN InstHyp [0
| . | . |     ....
| . | . |
| . | . 1. Label
| . | . 2. ds : Collection(dec())
| . | . 3. daa : Collection(dec())
| . | . 4. da : Collection(SimpleType)
| . | . 5. de : sig()
| . | . 6. rho : Decl
| . | . 7. e1 : {1of([[de]] rho)}
| . | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . 9. s1 : {[[ds]] rho}
| . | . 10. s2 : {[[ds]] rho}
| . | . 11. a : [[da]] rho
| . | . 12. tr : trace_env([[daa]] rho)
| . | . 13. Term List
| . | . 14. u : Term
| . | . 15. v : Term List
| . | . 16. ∀l2:Term List.
| . | . |    ||v|| = ||l2||
| . | . |    ⇒
| . | . |    (∀ls:SimpleType List.
| . | . |     ||ls|| = ||v||
| . | . |     ⇒
| . | . |     (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . |      (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . |      ⇒
| . | . |      (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . |      ⇒
| . | . |      ||ls|| = ||v||
| . | . |      & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . |      ⇒
| . | . |      ||ls|| = ||l2||
| . | . |      & (∀i:ℕ. i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . |      ⇒
| . | . |      (∀i:ℕ.
| . | . |       i<||v||
| . | . |       ⇒
| . | . |       [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . |      ⇒
| . | . |      (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . |       ⟺
| . | . |       list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
```

```
| . | . 17. Term List
| . | . 18. u1 : Term
| . | . 19. v1 : Term List
| . | . 20. ||v||+1 = ||v1||+1
| . | . 21. SimpleType List
| . | . 22. u2 : SimpleType
| . | . 23. v2 : SimpleType List
| . | . 24. ||v2||+1 = ||v||+1
| . | . 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . 26. ∀i:ℕ(||v||+1).
| . | . |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . 27. ∀i:ℕ(||v1||+1).
| . | . |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . 28. ||v2||+1 = ||v||+1
| . | . |   & (∀i:ℕ.
| . | . |      i<||v||+1
| . | . |      ⇒
| . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . 29. ||v2||+1 = ||v1||+1
| . | . |   & (∀i:ℕ.
| . | . |      i<||v1||+1
| . | . |      ⇒
| . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . 30. ∀i:ℕ.
| . | . |   i<||v||+1
| . | . |   ⇒
| . | . |   [[[u / v][i]]] e1 s1 a tr = [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . 31. ∀i:ℕ.
| . | . |   i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . 34. ||v2||+1 = ||v1||+1
| . | . |   & (∀i:ℕ.
| . | . |      i<||v1||+1
| . | . |      ⇒
| . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . 35. ||v2||+1 = ||v||+1
| . | . |   & (∀i:ℕ.
| . | . |      i<||v||+1
| . | . |      ⇒
| . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . |   (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . |   ⇒
| . | . |   (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . |   ⇒
| . | . |   ||v2|| = ||v||
| . | . |   & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . |   ⇒
| . | . |   ||v2|| = ||v1||
| . | . |   & (∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . |   ⇒
| . | . |   (∀i:ℕ.
| . | . |    i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . |   ⇒
```

```
| . | . |   (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . |    ⟺
| . | . |    list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . ⊢  list_accum(x,t.x([[t]] e1 s1 a tr);f([[u]] e1 s1 a tr);v)
| . | . |  ⟺
| . | . |  list_accum(x,t.x
| . | . |                 ([[t]] e1 s1 s2 a tr);...;v1)
| . | . | by At Prop (HypSubst (-1) 0)·
| . | . |
| . | . |\
| . | . | 1. Label
| . | . | 2. ds : Collection(dec())
| . | . | 3. daa : Collection(dec())
| . | . | 4. da : Collection(SimpleType)
| . | . | 5. de : sig()
| . | . | 6. rho : Decl
| . | . | 7. e1 : {1of([[de]] rho)}
| . | . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . | 9. s1 : {[[ds]] rho}
| . | . | 10. s2 : {[[ds]] rho}
| . | . | 11. a : [[da]] rho
| . | . | 12. tr : trace_env([[daa]] rho)
| . | . | 13. Term List
| . | . | 14. u : Term
| . | . | 15. v : Term List
| . | . | 16. ∀l2:Term List.
| . | . | |   ||v|| = ||l2||
| . | . | |   ⇒
| . | . | |   (∀ls:SimpleType List.
| . | . | |    ||ls|| = ||v||
| . | . | |    ⇒
| . | . | |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . | |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | |     ⇒
| . | . | |     (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . | |     ⇒
| . | . | |     ||ls|| = ||v||
| . | . | |     & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . | |     ⇒
| . | . | |     ||ls|| = ||l2||
| . | . | |     & (∀i:ℕ.
| . | . | |        i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . | |     ⇒
| . | . | |     (∀i:ℕ.
| . | . | |      i<||v||
| . | . | |      ⇒
| . | . | |      [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . | |     ⇒
| . | . | |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | |      ⟺
| . | . | |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . | 17. Term List
| . | . | 18. u1 : Term
| . | . | 19. v1 : Term List
```

```
| . | . | 20. ||v||+1 = ||v1||+1
| . | . | 21. SimpleType List
| . | . | 22. u2 : SimpleType
| . | . | 23. v2 : SimpleType List
| . | . | 24. ||v2||+1 = ||v||+1
| . | . | 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . | 26. ∀i:ℕ(||v||+1).
| . | . | |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . | 27. ∀i:ℕ(||v1||+1).
| . | . | |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . | 28. ||v2||+1 = ||v||+1
| . | . | |   & (∀i:ℕ.
| . | . | |      i<||v||+1
| . | . | |      ⇒
| . | . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | 29. ||v2||+1 = ||v1||+1
| . | . | |   & (∀i:ℕ.
| . | . | |      i<||v1||+1
| . | . | |      ⇒
| . | . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | 30. ∀i:ℕ.
| . | . | |   i<||v||+1
| . | . | |   ⇒
| . | . | |   [[[u / v][i]]] e1 s1 a tr
| . | . | |   =
| . | . | |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . | 31. ∀i:ℕ.
| . | . | |   i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . | 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . | 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . | 34. ||v2||+1 = ||v1||+1
| . | . | |   & (∀i:ℕ.
| . | . | |      i<||v1||+1
| . | . | |      ⇒
| . | . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | 35. ||v2||+1 = ||v||+1
| . | . | |   & (∀i:ℕ.
| . | . | |      i<||v||+1
| . | . | |      ⇒
| . | . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . | |   (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | |   ⇒
| . | . | |   (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . | |   ⇒
| . | . | |   ||v2|| = ||v||
| . | . | |   & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . | |   ⇒
| . | . | |   ||v2|| = ||v1||
| . | . | |   & (∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . | |   ⇒
| . | . | |   (∀i:ℕ.
| . | . | |    i<||v||
| . | . | |    ⇒
| . | . | |    [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
```

```
| . | . | |   ⇒
| . | . | |   (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | |    ⟺
| . | . | |    list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . | 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . | ⊢  list_accum(x,t.x
| . | . | |                ([[t]] e1 s1 a tr);...;v)
| . | . | |  ⟺
| . | . | |  list_accum(x,t.x
| . | . | |                ([[t]] e1 s1 s2 a tr);f
| . | . | |                                      (...);v1)
| . | . | | by SimpleInstHyp f([[u1]] e1 s1 s2 a tr) (-2) ·
| . | . | |

| . | . | |\
| . | . | | .....wf .....
| . | . | | 1. Label
| . | . | | 2. ds : Collection(dec())
| . | . | | 3. daa : Collection(dec())
| . | . | | 4. da : Collection(SimpleType)
| . | . | | 5. de : sig()
| . | . | | 6. rho : Decl
| . | . | | 7. e1 : {1of([[de]] rho)}
| . | . | | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . | | 9. s1 : {[[ds]] rho}
| . | . | | 10. s2 : {[[ds]] rho}
| . | . | | 11. a : [[da]] rho
| . | . | | 12. tr : trace_env([[daa]] rho)
| . | . | | 13. Term List
| . | . | | 14. u : Term
| . | . | | 15. v : Term List
| . | . | | 16. ∀l2:Term List.
| . | . | |     ||v|| = ||l2||
| . | . | |     ⇒
| . | . | |     (∀ls:SimpleType List.
| . | . | |      ||ls|| = ||v||
| . | . | |      ⇒
| . | . | |      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . | |       (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | |       ⇒
| . | . | |       (∀i:ℕ||l2||.
| . | . | |        trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . | |       ⇒
| . | . | |       ||ls|| = ||v||
| . | . | |       & (∀i:ℕ.
| . | . | |          i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . | |       ⇒
| . | . | |       ||ls|| = ||l2||
| . | . | |       & (∀i:ℕ.
| . | . | |          i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . | |       ⇒
| . | . | |       (∀i:ℕ.
| . | . | |        i<||v||
| . | . | |        ⇒
| . | . | |        [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . | |       ⇒
```

```
| . | . | |       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | |        ⟺
| . | . | |        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . | | 17. Term List
| . | . | | 18. u1 : Term
| . | . | | 19. v1 : Term List
| . | . | | 20. ||v||+1 = ||v1||+1
| . | . | | 21. SimpleType List
| . | . | | 22. u2 : SimpleType
| . | . | | 23. v2 : SimpleType List
| . | . | | 24. ||v2||+1 = ||v||+1
| . | . | | 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . | | 26. ∀i:ℕ(||v||+1).
| . | . | |     trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . | | 27. ∀i:ℕ(||v1||+1).
| . | . | |     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . | | 28. ||v2||+1 = ||v||+1
| . | . | |     & (∀i:ℕ.
| . | . | |        i<||v||+1
| . | . | |        ⇒
| . | . | |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | | 29. ||v2||+1 = ||v1||+1
| . | . | |     & (∀i:ℕ.
| . | . | |        i<||v1||+1
| . | . | |        ⇒
| . | . | |        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | | 30. ∀i:ℕ.
| . | . | |     i<||v||+1
| . | . | |     ⇒
| . | . | |     [[[u / v][i]]] e1 s1 a tr
| . | . | |     =
| . | . | |     [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . | | 31. ∀i:ℕ.
| . | . | |     i<||v||
| . | . | |     ⇒
| . | . | |     [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . | | 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . | | 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . | | 34. ||v2||+1 = ||v1||+1
| . | . | |     & (∀i:ℕ.
| . | . | |        i<||v1||+1
| . | . | |        ⇒
| . | . | |        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | | 35. ||v2||+1 = ||v||+1
| . | . | |     & (∀i:ℕ.
| . | . | |        i<||v||+1
| . | . | |        ⇒
| . | . | |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | | 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . | |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | |     ⇒
| . | . | |     (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . | |     ⇒
| . | . | |     ||v2|| = ||v||
| . | . | |     & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
```

```
| . | . | |     ⇒
| . | . | |     ||v2|| = ||v1||
| . | . | |     & (∀i:ℕ.
| . | . | |        i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . | |     ⇒
| . | . | |     (∀i:ℕ.
| . | . | |      i<||v||
| . | . | |      ⇒
| . | . | |      [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . | |     ⇒
| . | . | |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | |      ⟺
| . | . | |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . | | 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . | | ⊢  f([[u1]] e1 s1 s2 a tr)
| . | . | |    ∈ reduce(λs,m. [[s]] rho→m;Prop;v2)  by Auto
| . | . | |
| . | . | .\
| . | . | . 1. Label
| . | . | . 2. ds : Collection(dec())
| . | . | . 3. daa : Collection(dec())
| . | . | . 4. da : Collection(SimpleType)
| . | . | . 5. de : sig()
| . | . | . 6. rho : Decl
| . | . | . 7. e1 : {1of([[de]] rho)}
| . | . | . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . | . 9. s1 : {[[ds]] rho}
| . | . | . 10. s2 : {[[ds]] rho}
| . | . | . 11. a : [[da]] rho
| . | . | . 12. tr : trace_env([[daa]] rho)
| . | . | . 13. Term List
| . | . | . 14. u : Term
| . | . | . 15. v : Term List
| . | . | . 16. ∀l2:Term List.
| . | . | . |   ||v|| = ||l2||
| . | . | . |   ⇒
| . | . | . |   (∀ls:SimpleType List.
| . | . | . |    ||ls|| = ||v||
| . | . | . |    ⇒
| . | . | . |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . | . |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | . |     ⇒
| . | . | . |     (∀i:ℕ||l2||.
| . | . | . |      trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . | . |     ⇒
| . | . | . |     ||ls|| = ||v||
| . | . | . |     & (∀i:ℕ.
| . | . | . |        i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . | . |     ⇒
| . | . | . |     ||ls|| = ||l2||
| . | . | . |     & (∀i:ℕ.
| . | . | . |        i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . | . |     ⇒
| . | . | . |     (∀i:ℕ.
| . | . | . |      i<||v||
```

```
| . | . | . |       ⇒
| . | . | . |       [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . | . |      ⇒
| . | . | . |      (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | . |       ⟺
| . | . | . |       list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . | . 17. Term List
| . | . | . 18. u1 : Term
| . | . | . 19. v1 : Term List
| . | . | . 20. ||v||+1 = ||v1||+1
| . | . | . 21. SimpleType List
| . | . | . 22. u2 : SimpleType
| . | . | . 23. v2 : SimpleType List
| . | . | . 24. ||v2||+1 = ||v||+1
| . | . | . 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . | . 26. ∀i:ℕ(||v||+1).
| . | . | . |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . | . 27. ∀i:ℕ(||v1||+1).
| . | . | . |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . | . 28. ||v2||+1 = ||v||+1
| . | . | . |   & (∀i:ℕ.
| . | . | . |      i<||v||+1
| . | . | . |      ⇒
| . | . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | . 29. ||v2||+1 = ||v1||+1
| . | . | . |   & (∀i:ℕ.
| . | . | . |      i<||v1||+1
| . | . | . |      ⇒
| . | . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | . 30. ∀i:ℕ.
| . | . | . |   i<||v||+1
| . | . | . |   ⇒
| . | . | . |   [[[u / v][i]]] e1 s1 a tr
| . | . | . |   =
| . | . | . |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . | . 31. ∀i:ℕ.
| . | . | . |   i<||v||
| . | . | . |   ⇒
| . | . | . |   [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . | . 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . | . 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . | . 34. ||v2||+1 = ||v1||+1
| . | . | . |   & (∀i:ℕ.
| . | . | . |      i<||v1||+1
| . | . | . |      ⇒
| . | . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | . 35. ||v2||+1 = ||v||+1
| . | . | . |   & (∀i:ℕ.
| . | . | . |      i<||v||+1
| . | . | . |      ⇒
| . | . | . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | . 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . | . |   (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | . |   ⇒
| . | . | . |   (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
```

```
| . | . | . |   ⇒
| . | . | . |   ||v2|| = ||v||
| . | . | . |   & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . | . |   ⇒
| . | . | . |   ||v2|| = ||v1||
| . | . | . |   & (∀i:ℕ.
| . | . | . |      i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . | . |   ⇒
| . | . | . |   (∀i:ℕ.
| . | . | . |    i<||v||
| . | . | . |    ⇒
| . | . | . |    [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . | . |   ⇒
| . | . | . |   (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | . |    ⟺
| . | . | . |    list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . | . . 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . | . . 38. (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | . |   ⇒
| . | . | . |   (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . | . |   ⇒
| . | . | . |   ||v2|| = ||v||
| . | . | . |   & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . | . |   ⇒
| . | . | . |   ||v2|| = ||v1||
| . | . | . |   & (∀i:ℕ.
| . | . | . |      i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . | . |   ⇒
| . | . | . |   (∀i:ℕ.
| . | . | . |    i<||v||
| . | . | . |    ⇒
| . | . | . |    [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . | . |   ⇒
| . | . | . |   (list_accum(x,t.x
| . | . | . |                   ([[t]] e1 s1 a tr);f
| . | . | . |                                     (...);v)
| . | . | . |    ⟺
| . | . | . |    list_accum(x,t.x
| . | . | . |                   ([[t]] e1 s1 s2 a tr);f
| . | . | . |                                        (...);v1))
| . | . | . ⊢  list_accum(x,t.x
| . | . | . |                 ([[t]] e1 s1 a tr);...;v)
| . | . | . |  ⟺
| . | . | . |  list_accum(x,t.x
| . | . | . |                 ([[t]] e1 s1 s2 a tr);f
| . | . | . |                                      (...);v1)
| . | . | . | by D (-1) THENM D (-1)
| . | . | . |
| . | . | . |\
| . | . | . | .....antecedent .....
| . | . | . | 1. Label
| . | . | . | 2. ds : Collection(dec())
| . | . | . | 3. daa : Collection(dec())
| . | . | . | 4. da : Collection(SimpleType)
| . | . | . | 5. de : sig()
```

```
| . | . | . | 6. rho : Decl
| . | . | . | 7. e1 : {1of([[de]] rho)}
| . | . | . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . | . | 9. s1 : {[[ds]] rho}
| . | . | . | 10. s2 : {[[ds]] rho}
| . | . | . | 11. a : [[da]] rho
| . | . | . | 12. tr : trace_env([[daa]] rho)
| . | . | . | 13. Term List
| . | . | . | 14. u : Term
| . | . | . | 15. v : Term List
| . | . | . | 16. ∀l2:Term List.
| . | . | .       ||v|| = ||l2||
| . | . | .       ⇒
| . | . | .       (∀ls:SimpleType List.
| . | . | .        ||ls|| = ||v||
| . | . | .        ⇒
| . | . | .        (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . | .         (∀i:ℕ||v||.
| . | . | .          trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | .         ⇒
| . | . | .         (∀i:ℕ||l2||.
| . | . | .          trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . | .         ⇒
| . | . | .         ||ls|| = ||v||
| . | . | .         & (∀i:ℕ.
| . | . | .             i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . | .         ⇒
| . | . | .         ||ls|| = ||l2||
| . | . | .         & (∀i:ℕ.
| . | . | .             i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . | .         ⇒
| . | . | .         (∀i:ℕ.
| . | . | .          i<||v||
| . | . | .          ⇒
| . | . | .          [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . | .         ⇒
| . | . | .         (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | .          ⟺
| . | . | .          list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . | . | 17. Term List
| . | . | . | 18. u1 : Term
| . | . | . | 19. v1 : Term List
| . | . | . | 20. ||v||+1 = ||v1||+1
| . | . | . | 21. SimpleType List
| . | . | . | 22. u2 : SimpleType
| . | . | . | 23. v2 : SimpleType List
| . | . | . | 24. ||v2||+1 = ||v||+1
| . | . | . | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . | . | 26. ∀i:ℕ(||v||+1).
| . | . | .       trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . | . | 27. ∀i:ℕ(||v1||+1).
| . | . | .       trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . | . | 28. ||v2||+1 = ||v||+1
| . | . | .       & (∀i:ℕ.
| . | . | .           i<||v||+1
```

```
| . | . | . |         ⇒
| . | . | . |         [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | . | 29. ||v2||+1 = ||v1||+1
| . | . | . |     & (∀i:ℕ.
| . | . | . |         i<||v1||+1
| . | . | . |         ⇒
| . | . | . |         [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | . | 30. ∀i:ℕ.
| . | . | . |     i<||v||+1
| . | . | . |     ⇒
| . | . | . |     [[[u / v][i]]] e1 s1 a tr
| . | . | . |     =
| . | . | . |     [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . | . | 31. ∀i:ℕ.
| . | . | . |     i<||v||
| . | . | . |     ⇒
| . | . | . |     [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . | . | 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . | . | 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . | . | 34. ||v2||+1 = ||v1||+1
| . | . | . |     & (∀i:ℕ.
| . | . | . |         i<||v1||+1
| . | . | . |         ⇒
| . | . | . |         [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | . | 35. ||v2||+1 = ||v||+1
| . | . | . |     & (∀i:ℕ.
| . | . | . |         i<||v||+1
| . | . | . |         ⇒
| . | . | . |         [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | . | 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . | . |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | . |     ⇒
| . | . | . |     (∀i:ℕ||v1||.
| . | . | . |      trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . | . |     ⇒
| . | . | . |     ||v2|| = ||v||
| . | . | . |     & (∀i:ℕ.
| . | . | . |         i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . | . |     ⇒
| . | . | . |     ||v2|| = ||v1||
| . | . | . |     & (∀i:ℕ.
| . | . | . |         i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . | . |     ⇒
| . | . | . |     (∀i:ℕ.
| . | . | . |      i<||v||
| . | . | . |      ⇒
| . | . | . |      [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . | . |     ⇒
| . | . | . |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | . |      ⟺
| . | . | . |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . | . | 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . | . | ⊢  ∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i])
| . | . | . |   by Auto THEN
| . | . | . | AllHyps (\h. Complete (InstHyp [i+1
```

```
| . | . | . |      ] h THENA Auto'
| . | . | . | THEN Subst [u / v][(i+1)] ~ v[i]
| . | . | . |       (-1)
| . | . | . | THEN Try Trivial
| . | . | . | THEN Try (BackThruLemma `
| . | . | . |      select_cons_tl_sq` THEN Auto)))·
| . | . | . |
| . | . | . |\
| . | . | . | .....antecedent .....
| . | . | . | 1. Label
| . | . | . | 2. ds : Collection(dec())
| . | . | . | 3. daa : Collection(dec())
| . | . | . | 4. da : Collection(SimpleType)
| . | . | . | 5. de : sig()
| . | . | . | 6. rho : Decl
| . | . | . | 7. e1 : {1of([[de]] rho)}
| . | . | . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . | . | 9. s1 : {[[ds]] rho}
| . | . | . | 10. s2 : {[[ds]] rho}
| . | . | . | 11. a : [[da]] rho
| . | . | . | 12. tr : trace_env([[daa]] rho)
| . | . | . | 13. Term List
| . | . | . | 14. u : Term
| . | . | . | 15. v : Term List
| . | . | . | 16. ∀l2:Term List.
| . | . | . |     ||v|| = ||l2||
| . | . | . |     ⇒
| . | . | . |     (∀ls:SimpleType List.
| . | . | . |      ||ls|| = ||v||
| . | . | . |      ⇒
| . | . | . |      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . | . |       (∀i:ℕ||v||.
| . | . | . |        trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | . |       ⇒
| . | . | . |       (∀i:ℕ||l2||.
| . | . | . |        trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . | . |       ⇒
| . | . | . |       ||ls|| = ||v||
| . | . | . |       & (∀i:ℕ.
| . | . | . |          i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . | . |       ⇒
| . | . | . |       ||ls|| = ||l2||
| . | . | . |       & (∀i:ℕ.
| . | . | . |          i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . | . |       ⇒
| . | . | . |       (∀i:ℕ.
| . | . | . |        i<||v||
| . | . | . |        ⇒
| . | . | . |        [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . | . |       ⇒
| . | . | . |       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | . |        ⟺
| . | . | . |        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . | . | 17. Term List
| . | . | . | 18. u1 : Term
```

```
| . | . | . | 19. v1 : Term List
| . | . | . | 20. ||v||+1 = ||v1||+1
| . | . | . | 21. SimpleType List
| . | . | . | 22. u2 : SimpleType
| . | . | . | 23. v2 : SimpleType List
| . | . | . | 24. ||v2||+1 = ||v||+1
| . | . | . | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . | . | 26. ∀i:ℕ(||v||+1).
| . | . | . |     trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . | . | 27. ∀i:ℕ(||v1||+1).
| . | . | . |     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . | . | 28. ||v2||+1 = ||v||+1
| . | . | . |     & (∀i:ℕ.
| . | . | . |        i<||v||+1
| . | . | . |        ⇒
| . | . | . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | . | 29. ||v2||+1 = ||v1||+1
| . | . | . |     & (∀i:ℕ.
| . | . | . |        i<||v1||+1
| . | . | . |        ⇒
| . | . | . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | . | 30. ∀i:ℕ.
| . | . | . |     i<||v||+1
| . | . | . |     ⇒
| . | . | . |     [[[u / v][i]]] e1 s1 a tr
| . | . | . |     =
| . | . | . |     [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . | . | 31. ∀i:ℕ.
| . | . | . |     i<||v||
| . | . | . |     ⇒
| . | . | . |     [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . | . | 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . | . | 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . | . | 34. ||v2||+1 = ||v1||+1
| . | . | . |     & (∀i:ℕ.
| . | . | . |        i<||v1||+1
| . | . | . |        ⇒
| . | . | . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | . | 35. ||v2||+1 = ||v||+1
| . | . | . |     & (∀i:ℕ.
| . | . | . |        i<||v||+1
| . | . | . |        ⇒
| . | . | . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | . | 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . | . |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | . |     ⇒
| . | . | . |     (∀i:ℕ||v1||.
| . | . | . |      trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . | . |     ⇒
| . | . | . |     ||v2|| = ||v||
| . | . | . |     & (∀i:ℕ.
| . | . | . |        i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . | . |     ⇒
| . | . | . |     ||v2|| = ||v1||
| . | . | . |     & (∀i:ℕ.
```

```
| . | . | . |         i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . | . |       ⇒
| . | . | . |       (∀i:ℕ.
| . | . | . |        i<||v||
| . | . | . |        ⇒
| . | . | . |        [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . | . |       ⇒
| . | . | . |       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | . |        ⟺
| . | . | . |        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . | . | 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . | . | ⊢  ∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i])
| . | . | . |   by Auto THEN
| . | . | . | AllHyps (\h. Complete (InstHyp [i+1
| . | . | . |      ] h THENA Auto'
| . | . | . | THEN Subst [u1 / v1][(i
| . | . | . |      +1)] ~ v1[i] (-1)
| . | . | . | THEN Try Trivial

| . | . | . |
| . | . | . |      THEN Try (BackThruLemma `
| . | . | . |      select_cons_tl_sq` THEN Auto)))· ·
| . | . | . |
| . | . | . .\
| . | . | . . 1. Label
| . | . | . . 2. ds : Collection(dec())
| . | . | . . 3. daa : Collection(dec())
| . | . | . . 4. da : Collection(SimpleType)
| . | . | . . 5. de : sig()
| . | . | . . 6. rho : Decl
| . | . | . . 7. e1 : {1of([[de]] rho)}
| . | . | . . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . | . . 9. s1 : {[[ds]] rho}
| . | . | . . 10. s2 : {[[ds]] rho}
| . | . | . . 11. a : [[da]] rho
| . | . | . . 12. tr : trace_env([[daa]] rho)
| . | . | . . 13. Term List
| . | . | . . 14. u : Term
| . | . | . . 15. v : Term List
| . | . | . . 16. ∀l2:Term List.
| . | . | . .     ||v|| = ||l2||
| . | . | . .     ⇒
| . | . | . .     (∀ls:SimpleType List.
| . | . | . .      ||ls|| = ||v||
| . | . | . .      ⇒
| . | . | . .      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . | . .       (∀i:ℕ||v||.
| . | . | . .        trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | . .       ⇒
| . | . | . .       (∀i:ℕ||l2||.
| . | . | . .        trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . | . .       ⇒
| . | . | . .       ||ls|| = ||v||
| . | . | . .       & (∀i:ℕ.
| . | . | . .          i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . | . .       ⇒
```

```
| . | . | . .       ||ls|| = ||l2||
| . | . | . .       & (∀i:ℕ.
| . | . | . .          i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . | . .       ⇒
| . | . | . .       (∀i:ℕ.
| . | . | . .        i<||v||
| . | . | . .        ⇒
| . | . | . .        [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . | . .       ⇒
| . | . | . .       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | . .        ⟺
| . | . | . .        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . | . . 17. Term List
| . | . | . . 18. u1 : Term
| . | . | . . 19. v1 : Term List
| . | . | . . 20. ||v||+1 = ||v1||+1
| . | . | . . 21. SimpleType List
| . | . | . . 22. u2 : SimpleType
| . | . | . . 23. v2 : SimpleType List
| . | . | . . 24. ||v2||+1 = ||v||+1
| . | . | . . 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . | . . 26. ∀i:ℕ(||v||+1).
| . | . | . .     trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . | . . 27. ∀i:ℕ(||v1||+1).
| . | . | . .     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . | . . 28. ||v2||+1 = ||v||+1
| . | . | . .     & (∀i:ℕ.
| . | . | . .        i<||v||+1
| . | . | . .        ⇒
| . | . | . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | . . 29. ||v2||+1 = ||v1||+1
| . | . | . .     & (∀i:ℕ.
| . | . | . .        i<||v1||+1
| . | . | . .        ⇒
| . | . | . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | . . 30. ∀i:ℕ.
| . | . | . .     i<||v||+1
| . | . | . .     ⇒
| . | . | . .     [[[u / v][i]]] e1 s1 a tr
| . | . | . .     =
| . | . | . .     [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . | . . 31. ∀i:ℕ.
| . | . | . .     i<||v||
| . | . | . .     ⇒
| . | . | . .     [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . | . . 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . | . . 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . | . . 34. ||v2||+1 = ||v1||+1
| . | . | . .     & (∀i:ℕ.
| . | . | . .        i<||v1||+1
| . | . | . .        ⇒
| . | . | . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . | . . 35. ||v2||+1 = ||v||+1
| . | . | . .     & (∀i:ℕ.
| . | . | . .        i<||v||+1
```

```
| . | . | . .        ⇒
| . | . | . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . | . . 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . | . .     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . | . .     ⇒
| . | . | . .     (∀i:ℕ||v1||.
| . | . | . .      trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . | . .     ⇒
| . | . | . .     ||v2|| = ||v||
| . | . | . .     & (∀i:ℕ.
| . | . | . .        i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . | . .     ⇒
| . | . | . .     ||v2|| = ||v1||
| . | . | . .     & (∀i:ℕ.
| . | . | . .        i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . | . .     ⇒
| . | . | . .     (∀i:ℕ.
| . | . | . .      i<||v||
| . | . | . .      ⇒
| . | . | . .      [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . | . .     ⇒
| . | . | . .     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . | . .      ⟺
| . | . | . .      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . | . . 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . | . . 38. ||v2|| = ||v||
| . | . | . .     & (∀i:ℕ.
| . | . | . .        i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . | . .     ⇒
| . | . | . .     ||v2|| = ||v1||
| . | . | . .     & (∀i:ℕ.
| . | . | . .        i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . | . .     ⇒
| . | . | . .     (∀i:ℕ.
| . | . | . .      i<||v||
| . | . | . .      ⇒
| . | . | . .      [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . | . .     ⇒
| . | . | . .     (list_accum(x,t.x
| . | . | . .                     ([[t]] e1 s1 a tr);f
| . | . | . .                                        (...);v)
| . | . | . .      ⟺
| . | . | . .      list_accum(x,t.x
| . | . | . .                     ([[t]] e1 s1 s2 a tr);f
| . | . | . .                                           (...);v1))
| . | . | . . ⊢  list_accum(x,t.x
| . | . | . .                   ([[t]] e1 s1 a tr);f
| . | . | . .                                      (...);v)
| . | . | . .    ⟺
| . | . | . .    list_accum(x,t.x
| . | . | . .                   ([[t]] e1 s1 s2 a tr);f
| . | . | . .                                         (...);v1)
| . | . | . .   by ...
| . | . | . . THEN Try Trivial
| . | . |
```

```
| . | . .\
| . | . . .....wf .....
| . | . . 1. Label
| . | . . 2. ds : Collection(dec())
| . | . . 3. daa : Collection(dec())
| . | . . 4. da : Collection(SimpleType)
| . | . . 5. de : sig()
| . | . . 6. rho : Decl
| . | . . 7. e1 : {1of([[de]] rho)}
| . | . . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . . 9. s1 : {[[ds]] rho}
| . | . . 10. s2 : {[[ds]] rho}
| . | . . 11. a : [[da]] rho
| . | . . 12. tr : trace_env([[daa]] rho)
| . | . . 13. Term List
| . | . . 14. u : Term
| . | . . 15. v : Term List
| . | . . 16. ∀l2:Term List.
| . | . . |   ||v|| = ||l2||
| . | . . |   ⇒
| . | . . |   (∀ls:SimpleType List.
| . | . . |    ||ls|| = ||v||
| . | . . |    ⇒
| . | . . |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . . |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . |     ⇒
| . | . . |     (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . . |     ⇒
| . | . . |     ||ls|| = ||v||
| . | . . |     & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . . |     ⇒
| . | . . |     ||ls|| = ||l2||
| . | . . |     & (∀i:ℕ.
| . | . . |        i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . . |     ⇒
| . | . . |     (∀i:ℕ.
| . | . . |      i<||v||
| . | . . |      ⇒
| . | . . |      [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . . |     ⇒
| . | . . |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . |      ⟺
| . | . . |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . . 17. Term List
| . | . . 18. u1 : Term
| . | . . 19. v1 : Term List
| . | . . 20. ||v||+1 = ||v1||+1
| . | . . 21. SimpleType List
| . | . . 22. u2 : SimpleType
| . | . . 23. v2 : SimpleType List
| . | . . 24. ||v2||+1 = ||v||+1
| . | . . 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . . 26. ∀i:ℕ(||v||+1).

| . | . . |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . . 27. ∀i:ℕ(||v1||+1).
```

```
| . | . . |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . . 28. ||v2||+1 = ||v||+1
| . | . . |   & (∀i:ℕ.
| . | . . |      i<||v||+1
| . | . . |      ⇒
| . | . . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . 29. ||v2||+1 = ||v1||+1
| . | . . |   & (∀i:ℕ.
| . | . . |      i<||v1||+1
| . | . . |      ⇒
| . | . . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . 30. ∀i:ℕ.
| . | . . |   i<||v||+1
| . | . . |   ⇒
| . | . . |   [[[u / v][i]]] e1 s1 a tr
| . | . . |   =
| . | . . |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . . 31. ∀i:ℕ.
| . | . . |   i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . . 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . . 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . . 34. ||v2||+1 = ||v1||+1
| . | . . |   & (∀i:ℕ.
| . | . . |      i<||v1||+1
| . | . . |      ⇒
| . | . . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . 35. ||v2||+1 = ||v||+1
| . | . . |   & (∀i:ℕ.
| . | . . |      i<||v||+1
| . | . . |      ⇒
| . | . . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . . |   (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . |   ⇒
| . | . . |   (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . . |   ⇒
| . | . . |   ||v2|| = ||v||
| . | . . |   & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . . |   ⇒
| . | . . |   ||v2|| = ||v1||
| . | . . |   & (∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . . |   ⇒
| . | . . |   (∀i:ℕ.
| . | . . |    i<||v||
| . | . . |    ⇒
| . | . . |    [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . . |   ⇒
| . | . . |   (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . |    ⟺
| . | . . |    list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . . 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . . 38. z : [[u2]] rho
| . | . . ⊢  (list_accum(x,t.x([[t]] e1 s1 a tr);f(z);v)
| . | . . |   ⟺
| . | . . |   list_accum(x,t.x
```

```
| . | . . |                  ([[t]] e1 s1 s2 a tr);f
| . | . . |                                            (...);v1))
| . | . . |  ∈ Prop  by Analyze
| . | . . |
| . | . . |\
| . | . . | .....subterm .....
| . | . . | 1. Label
| . | . . | 2. ds : Collection(dec())
| . | . . | 3. daa : Collection(dec())
| . | . . | 4. da : Collection(SimpleType)
| . | . . | 5. de : sig()
| . | . . | 6. rho : Decl
| . | . . | 7. e1 : {1of([[de]] rho)}
| . | . . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . . | 9. s1 : {[[ds]] rho}
| . | . . | 10. s2 : {[[ds]] rho}
| . | . . | 11. a : [[da]] rho
| . | . . | 12. tr : trace_env([[daa]] rho)
| . | . . | 13. Term List
| . | . . | 14. u : Term
| . | . . | 15. v : Term List
| . | . . | 16. ∀l2:Term List.
| . | . . | |   ||v|| = ||l2||
| . | . . | |   ⇒
| . | . . | |   (∀ls:SimpleType List.
| . | . . | |    ||ls|| = ||v||
| . | . . | |    ⇒
| . | . . | |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . . | |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . | |     ⇒
| . | . . | |     (∀i:ℕ||l2||.
| . | . . | |      trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . . | |     ⇒
| . | . . | |     ||ls|| = ||v||
| . | . . | |     & (∀i:ℕ.
| . | . . | |        i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . . | |     ⇒
| . | . . | |     ||ls|| = ||l2||
| . | . . | |     & (∀i:ℕ.
| . | . . | |        i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . . | |     ⇒
| . | . . | |     (∀i:ℕ.
| . | . . | |      i<||v||
| . | . . | |      ⇒
| . | . . | |      [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . . | |     ⇒
| . | . . | |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . | |      ⟺
| . | . . | |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . . | 17. Term List
| . | . . | 18. u1 : Term
| . | . . | 19. v1 : Term List
| . | . . | 20. ||v||+1 = ||v1||+1
| . | . . | 21. SimpleType List
| . | . . | 22. u2 : SimpleType
```

```
| . | . . | 23. v2 : SimpleType List
| . | . . | 24. ||v2||+1 = ||v||+1
| . | . . | 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . . | 26. ∀i:ℕ(||v||+1).
| . | . . | |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . . | 27. ∀i:ℕ(||v1||+1).
| . | . . | |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . . | 28. ||v2||+1 = ||v||+1
| . | . . | |   & (∀i:ℕ.
| . | . . | |      i<||v||+1
| . | . . | |      ⇒
| . | . . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . | 29. ||v2||+1 = ||v1||+1
| . | . . | |   & (∀i:ℕ.
| . | . . | |      i<||v1||+1
| . | . . | |      ⇒
| . | . . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . | 30. ∀i:ℕ.
| . | . . | |   i<||v||+1
| . | . . | |   ⇒
| . | . . | |   [[[u / v][i]]] e1 s1 a tr
| . | . . | |   =
| . | . . | |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . . | 31. ∀i:ℕ.
| . | . . | |   i<||v||
| . | . . | |   ⇒
| . | . . | |   [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . . | 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . . | 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . . | 34. ||v2||+1 = ||v1||+1
| . | . . | |   & (∀i:ℕ.
| . | . . | |      i<||v1||+1
| . | . . | |      ⇒
| . | . . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . | 35. ||v2||+1 = ||v||+1
| . | . . | |   & (∀i:ℕ.
| . | . . | |      i<||v||+1
| . | . . | |      ⇒
| . | . . | |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . | 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . . | |   (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . | |   ⇒
| . | . . | |   (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . . | |   ⇒
| . | . . | |   ||v2|| = ||v||
| . | . . | |   & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . . | |   ⇒
| . | . . | |   ||v2|| = ||v1||
| . | . . | |   & (∀i:ℕ.
| . | . . | |      i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . . | |   ⇒
| . | . . | |   (∀i:ℕ.
| . | . . | |    i<||v||
| . | . . | |    ⇒
| . | . . | |    [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
```

```
| . | . . | |   ⇒
| . | . . | |   (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . | |    ⟺
| . | . . | |    list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . . | 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . . | 38. z : [[u2]] rho
| . | . . | ⊢  list_accum(x,t.x([[t]] e1 s1 a tr);f(z);v) ∈ Prop
| . | . . | | by ...
| . | . . | THEN Try (D 0 THEN Try Trivial THEN Auto')·
| . | . . | |
| . | . . | 1. Label
| . | . . | 2. ds : Collection(dec())
| . | . . | 3. daa : Collection(dec())
| . | . . | 4. da : Collection(SimpleType)
| . | . . | 5. de : sig()
| . | . . | 6. rho : Decl
| . | . . | 7. e1 : {1of([[de]] rho)}
| . | . . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . . | 9. s1 : {[[ds]] rho}
| . | . . | 10. s2 : {[[ds]] rho}
| . | . . | 11. a : [[da]] rho
| . | . . | 12. tr : trace_env([[daa]] rho)
| . | . . | 13. Term List
| . | . . | 14. u : Term
| . | . . | 15. v : Term List
| . | . . | 16. ∀l2:Term List.
| . | . . |     ||v|| = ||l2||
| . | . . |     ⇒
| . | . . |     (∀ls:SimpleType List.
| . | . . |      ||ls|| = ||v||
| . | . . |      ⇒
| . | . . |      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . . |       (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . |       ⇒
| . | . . |       (∀i:ℕ||l2||.
| . | . . |        trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . . |       ⇒
| . | . . |       ||ls|| = ||v||
| . | . . |       & (∀i:ℕ.
| . | . . |          i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . . |       ⇒
| . | . . |       ||ls|| = ||l2||
| . | . . |       & (∀i:ℕ.
| . | . . |          i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . . |       ⇒
| . | . . |       (∀i:ℕ.
| . | . . |        i<||v||
| . | . . |        ⇒
| . | . . |        [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . . |       ⇒
| . | . . |       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . |        ⟺
| . | . . |        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . . | 17. Term List
| . | . . | 18. u1 : Term
```

```
| . | . . | 19. v1 : Term List
| . | . . | 20. ||v||+1 = ||v1||+1
| . | . . | 21. SimpleType List
| . | . . | 22. u2 : SimpleType
| . | . . | 23. v2 : SimpleType List
| . | . . | 24. ||v2||+1 = ||v||+1
| . | . . | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . . | 26. ∀i:ℕ(||v||+1).
| . | . . |     trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . . | 27. ∀i:ℕ(||v1||+1).
| . | . . |     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . . | 28. ||v2||+1 = ||v||+1
| . | . . |     & (∀i:ℕ.
| . | . . |        i<||v||+1
| . | . . |        ⇒
| . | . . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . | 29. ||v2||+1 = ||v1||+1
| . | . . |     & (∀i:ℕ.
| . | . . |        i<||v1||+1
| . | . . |        ⇒
| . | . . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . | 30. ∀i:ℕ.
| . | . . |     i<||v||+1
| . | . . |     ⇒
| . | . . |     [[[u / v][i]]] e1 s1 a tr
| . | . . |     =
| . | . . |     [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . . | 31. ∀i:ℕ.
| . | . . |     i<||v||
| . | . . |     ⇒
| . | . . |     [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . . | 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . . | 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . . | 34. ||v2||+1 = ||v1||+1
| . | . . |     & (∀i:ℕ.
| . | . . |        i<||v1||+1
| . | . . |        ⇒
| . | . . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . | 35. ||v2||+1 = ||v||+1
| . | . . |     & (∀i:ℕ.
| . | . . |        i<||v||+1
| . | . . |        ⇒
| . | . . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . | 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . . |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . |     ⇒
| . | . . |     (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . . |     ⇒
| . | . . |     ||v2|| = ||v||
| . | . . |     & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . . |     ⇒
| . | . . |     ||v2|| = ||v1||
| . | . . |     & (∀i:ℕ.
| . | . . |        i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . . |     ⇒
```

```
| . | . . |     (∀i:ℕ.
| . | . . |      i<||v||
| . | . . |      ⇒
| . | . . |      [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . . |     ⇒
| . | . . |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . |      ⟺
| . | . . |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . . | 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . . | 38. [[u2]] rho
| . | . . | 39. i : ℕ||v||
| . | . . | ⊢  trace_consistent(rho;daa;tr.proj;v[i])
| . | . . |   by AllHyps (\h. InstHyp [i+1] h THENA Auto'
| . | . . |
| . | . . |          THEN Subst [u / v][(i+1)] ∼ v[i] (-1)
| . | . . |
| . | . . |      ...
| . | . . |       THEN Trivial)
| . | . . |
| . | . . .\
| . | . . . .....subterm .....
| . | . . . 1. Label
| . | . . . 2. ds : Collection(dec())
| . | . . . 3. daa : Collection(dec())
| . | . . . 4. da : Collection(SimpleType)
| . | . . . 5. de : sig()
| . | . . . 6. rho : Decl
| . | . . . 7. e1 : {1of([[de]] rho)}
| . | . . . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . . . 9. s1 : {[[ds]] rho}
| . | . . . 10. s2 : {[[ds]] rho}
| . | . . . 11. a : [[da]] rho
| . | . . . 12. tr : trace_env([[daa]] rho)
| . | . . . 13. Term List
| . | . . . 14. u : Term
| . | . . . 15. v : Term List
| . | . . . 16. ∀l2:Term List.
| . | . . . |   ||v|| = ||l2||
| . | . . . |   ⇒
| . | . . . |   (∀ls:SimpleType List.
| . | . . . |    ||ls|| = ||v||
| . | . . . |    ⇒
| . | . . . |    (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . . . |     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . . |     ⇒
| . | . . . |     (∀i:ℕ||l2||.
| . | . . . |      trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . . . |     ⇒
| . | . . . |     ||ls|| = ||v||
| . | . . . |     & (∀i:ℕ.
| . | . . . |        i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . . . |     ⇒
| . | . . . |     ||ls|| = ||l2||
| . | . . . |     & (∀i:ℕ.
| . | . . . |        i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
```

```
| . | . . . |     ⇒
| . | . . . |     (∀i:ℕ.
| . | . . . |      i<||v||
| . | . . . |      ⇒
| . | . . . |      [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . . . |     ⇒
| . | . . . |     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . . |      ⟺
| . | . . . |      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . . . 17. Term List
| . | . . . 18. u1 : Term
| . | . . . 19. v1 : Term List
| . | . . . 20. ||v||+1 = ||v1||+1
| . | . . . 21. SimpleType List
| . | . . . 22. u2 : SimpleType
| . | . . . 23. v2 : SimpleType List
| . | . . . 24. ||v2||+1 = ||v||+1
| . | . . . 25. f : [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . . . 26. ∀i:ℕ(||v||+1).
| . | . . . |   trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . . . 27. ∀i:ℕ(||v1||+1).
| . | . . . |   trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . . . 28. ||v2||+1 = ||v||+1
| . | . . . |   & (∀i:ℕ.
| . | . . . |      i<||v||+1
| . | . . . |      ⇒
| . | . . . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . . 29. ||v2||+1 = ||v1||+1
| . | . . . |   & (∀i:ℕ.
| . | . . . |      i<||v1||+1
| . | . . . |      ⇒
| . | . . . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . . 30. ∀i:ℕ.
| . | . . . |   i<||v||+1
| . | . . . |   ⇒
| . | . . . |   [[[u / v][i]]] e1 s1 a tr
| . | . . . |   =
| . | . . . |   [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . . . 31. ∀i:ℕ.
| . | . . . |   i<||v||
| . | . . . |   ⇒
| . | . . . |   [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . . . 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . . . 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . . . 34. ||v2||+1 = ||v1||+1
| . | . . . |   & (∀i:ℕ.
| . | . . . |      i<||v1||+1
| . | . . . |      ⇒
| . | . . . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . . 35. ||v2||+1 = ||v||+1
| . | . . . |   & (∀i:ℕ.
| . | . . . |      i<||v||+1
| . | . . . |      ⇒
| . | . . . |      [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . . 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
```

```
| . | . . . |   (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . . |   ⇒
| . | . . . |   (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . . . |   ⇒
| . | . . . |   ||v2|| = ||v||
| . | . . . |   & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . . . |   ⇒
| . | . . . |   ||v2|| = ||v1||
| . | . . . |   & (∀i:ℕ.
| . | . . . |      i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . . . |   ⇒
| . | . . . |   (∀i:ℕ.
| . | . . . |    i<||v||
| . | . . . |    ⇒
| . | . . . |    [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . . . |   ⇒
| . | . . . |   (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . . |    ⟺
| . | . . . |    list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . . . 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . . . 38. [[u2]] rho
| . | . . . ⊢  list_accum(x,t.x
| . | . . . |                  ([[t]] e1 s1 s2 a tr);f
| . | . . . |                                        (...);v1)
| . | . . . |  ∈ Prop
| . | . . . | by ...
| . | . . . THEN Try (D 0 THEN Try Trivial THEN Auto')·
| . | . . . |
| . | . . . 1. Label
| . | . . . 2. ds : Collection(dec())
| . | . . . 3. daa : Collection(dec())
| . | . . . 4. da : Collection(SimpleType)
| . | . . . 5. de : sig()
| . | . . . 6. rho : Decl
| . | . . . 7. e1 : {1of([[de]] rho)}
| . | . . . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | . . . 9. s1 : {[[ds]] rho}
| . | . . . 10. s2 : {[[ds]] rho}
| . | . . . 11. a : [[da]] rho
| . | . . . 12. tr : trace_env([[daa]] rho)
| . | . . . 13. Term List
| . | . . . 14. u : Term
| . | . . . 15. v : Term List
| . | . . . 16. ∀l2:Term List.
| . | . . .     ||v|| = ||l2||
| . | . . .     ⇒
| . | . . .     (∀ls:SimpleType List.
| . | . . .      ||ls|| = ||v||
| . | . . .      ⇒
| . | . . .      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . | . . .       (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . .       ⇒
| . | . . .       (∀i:ℕ||l2||.
| . | . . .        trace_consistent(rho;daa;tr.proj;l2[i]))
| . | . . .       ⇒
```

```
| . | . . .       ||ls|| = ||v||
| . | . . .       & (∀i:ℕ.
| . | . . .          i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . | . . .       ⇒
| . | . . .       ||ls|| = ||l2||
| . | . . .       & (∀i:ℕ.
| . | . . .          i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . | . . .       ⇒
| . | . . .       (∀i:ℕ.
| . | . . .        i<||v||
| . | . . .        ⇒
| . | . . .        [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . | . . .       ⇒
| . | . . .       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . .        ⟺
| . | . . .        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | . . . 17. Term List
| . | . . . 18. u1 : Term
| . | . . . 19. v1 : Term List
| . | . . . 20. ||v||+1 = ||v1||+1
| . | . . . 21. SimpleType List
| . | . . . 22. u2 : SimpleType
| . | . . . 23. v2 : SimpleType List
| . | . . . 24. ||v2||+1 = ||v||+1
| . | . . . 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | . . . 26. ∀i:ℕ(||v||+1).
| . | . . .     trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | . . . 27. ∀i:ℕ(||v1||+1).
| . | . . .     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | . . . 28. ||v2||+1 = ||v||+1
| . | . . .     & (∀i:ℕ.
| . | . . .        i<||v||+1
| . | . . .        ⇒
| . | . . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . . 29. ||v2||+1 = ||v1||+1
| . | . . .     & (∀i:ℕ.
| . | . . .        i<||v1||+1
| . | . . .        ⇒
| . | . . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . . 30. ∀i:ℕ.
| . | . . .     i<||v||+1
| . | . . .     ⇒
| . | . . .     [[[u / v][i]]] e1 s1 a tr
| . | . . .     =
| . | . . .     [[[u1 / v1][i]]] e1 s1 s2 a tr
| . | . . . 31. ∀i:ℕ.
| . | . . .     i<||v||
| . | . . .     ⇒
| . | . . .     [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr
| . | . . . 32. ∀i:ℕ. i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i])
| . | . . . 33. ∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i])
| . | . . . 34. ||v2||+1 = ||v1||+1
| . | . . .     & (∀i:ℕ.
| . | . . .        i<||v1||+1
| . | . . .        ⇒
```

```
| . | . . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | . . . 35. ||v2||+1 = ||v||+1
| . | . . .     & (∀i:ℕ.
| . | . . .        i<||v||+1
| . | . . .        ⇒
| . | . . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | . . . 36. ∀f:reduce(λs,m. [[s]] rho→m;Prop;v2).
| . | . . .     (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . | . . .     ⇒
| . | . . .     (∀i:ℕ||v1||. trace_consistent(rho;daa;tr.proj;v1[i]))
| . | . . .     ⇒
| . | . . .     ||v2|| = ||v||
| . | . . .     & (∀i:ℕ. i<||v|| ⇒ v2[i] ∈ term_types(ds;da;de;v[i]))
| . | . . .     ⇒
| . | . . .     ||v2|| = ||v1||
| . | . . .     & (∀i:ℕ.
| . | . . .        i<||v1|| ⇒ v2[i] ∈ term_types(ds;da;de;v1[i]))
| . | . . .     ⇒
| . | . . .     (∀i:ℕ.
| . | . . .      i<||v||
| . | . . .      ⇒
| . | . . .      [[v[i]]] e1 s1 a tr = [[v1[i]]] e1 s1 s2 a tr)
| . | . . .     ⇒
| . | . . .     (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . | . . .      ⟺
| . | . . .      list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;v1))
| . | . . . 37. [[u]] e1 s1 a tr = [[u1]] e1 s1 s2 a tr
| . | . . . 38. [[u2]] rho
| . | . . . 39. i : ℕ||v1||
| . | . . . ⊢  trace_consistent(rho;daa;tr.proj;v1[i])
| . | . . .   by AllHyps (\h. InstHyp [i+1] h THENA Auto'
| . | . . .
| . | . . .          THEN Subst [u1 / v1][(i+1)] ~ v1[i] (-1)
| . | . . .
| . | . . .      ...
| . | . . .       THEN Trivial)
| . |
| . |\
| . | 1. Label
| . | 2. ds : Collection(dec())
| . | 3. daa : Collection(dec())
| . | 4. da : Collection(SimpleType)
| . | 5. de : sig()
| . | 6. rho : Decl
| . | 7. e1 : {1of([[de]] rho)}
| . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | 9. s1 : {[[ds]] rho}
| . | 10. s2 : {[[ds]] rho}
| . | 11. a : [[da]] rho
| . | 12. tr : trace_env([[daa]] rho)
| . | 13. Term List
| . | 14. u : Term
| . | 15. v : Term List
| . | 16. ∀l2:Term List.
| . |     ||v|| = ||l2||
```

```
| . |      ⇒
| . |      (∀ls:SimpleType List.
| . |       ||ls|| = ||v||
| . |       ⇒
| . |       (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . |        (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . |        ⇒
| . |        (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . |        ⇒
| . |        ||ls|| = ||v||
| . |        & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . |        ⇒
| . |        ||ls|| = ||l2||
| . |        & (∀i:ℕ. i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . |        ⇒
| . |        (∀i:ℕ.
| . |         i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . |        ⇒
| . |        (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . |         ⟺
| . |         list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | 17. Term List
| . | 18. u1 : Term
| . | 19. v1 : Term List
| . | 20. ||v||+1 = ||v1||+1
| . | 21. SimpleType List
| . | 22. u2 : SimpleType
| . | 23. v2 : SimpleType List
| . | 24. ||v2||+1 = ||v||+1
| . | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | 26. ∀i:ℕ(||v||+1). trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | 27. ∀i:ℕ(||v1||+1).
| . |     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | 28. ||v2||+1 = ||v||+1
| . |     & (∀i:ℕ.
| . |        i<||v||+1
| . |        ⇒
| . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | 29. ||v2||+1 = ||v1||+1
| . |     & (∀i:ℕ.
| . |        i<||v1||+1
| . |        ⇒
| . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | 30. i : ℕ
| . | 31. i<||v||+1
| . | ⊢  i<||[u2 / v2]||  by Auto
| . |
| . |\
| . | .....subterm .....
| . | 1. Label
| . | 2. ds : Collection(dec())
| . | 3. daa : Collection(dec())
| . | 4. da : Collection(SimpleType)
| . | 5. de : sig()
| . | 6. rho : Decl
```

```
| . | 7. e1 : {1of([[de]] rho)}
| . | 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . | 9. s1 : {[[ds]] rho}
| . | 10. s2 : {[[ds]] rho}
| . | 11. a : [[da]] rho
| . | 12. tr : trace_env([[daa]] rho)
| . | 13. Term List
| . | 14. u : Term
| . | 15. v : Term List
| . | 16. ∀l2:Term List.
| . |     ||v|| = ||l2||
| . |     ⇒
| . |     (∀ls:SimpleType List.
| . |      ||ls|| = ||v||
| . |      ⇒
| . |      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . |       (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . |       ⇒
| . |       (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . |       ⇒
| . |       ||ls|| = ||v||
| . |       & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . |       ⇒
| . |       ||ls|| = ||l2||
| . |       & (∀i:ℕ. i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . |       ⇒
| . |       (∀i:ℕ.
| . |        i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . |       ⇒
| . |       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . |        ⟺
| . |        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . | 17. Term List
| . | 18. u1 : Term
| . | 19. v1 : Term List
| . | 20. ||v||+1 = ||v1||+1
| . | 21. SimpleType List
| . | 22. u2 : SimpleType
| . | 23. v2 : SimpleType List
| . | 24. ||v2||+1 = ||v||+1
| . | 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . | 26. ∀i:ℕ(||v||+1). trace_consistent(rho;daa;tr.proj;[u / v][i])
| . | 27. ∀i:ℕ(||v1||+1).
| . |     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . | 28. ||v2||+1 = ||v||+1
| . |     & (∀i:ℕ.
| . |        i<||v||+1
| . |        ⇒
| . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . | 29. ||v2||+1 = ||v1||+1
| . |     & (∀i:ℕ.
| . |        i<||v1||+1
| . |        ⇒
| . |        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . | 30. i : ℕ
```

```
| . | 31. i<||v||+1
| . | ⊢  [[[u / v][i]]] e1 s1 a tr ∈ [[[u2 / v2][i]]] rho
| . |   by ExRepD THEN Using [‘de’,de] (BackThruLemma ‘term_typing‘)
| . |
| . |      THENA Auto THEN Try (Reduce 0 THEN Auto’)
| . | THEN EasyHyp·
| . |
| . .\
| . . .....subterm .....
| . . 1. Label
| . . 2. ds : Collection(dec())
| . . 3. daa : Collection(dec())
| . . 4. da : Collection(SimpleType)
| . . 5. de : sig()
| . . 6. rho : Decl
| . . 7. e1 : {1of([[de]] rho)}
| . . 8. l:Label→reduce(λs,m. [[s]] rho→m;Prop;de.rel(l))
| . . 9. s1 : {[[ds]] rho}
| . . 10. s2 : {[[ds]] rho}
| . . 11. a : [[da]] rho
| . . 12. tr : trace_env([[daa]] rho)
| . . 13. Term List
| . . 14. u : Term
| . . 15. v : Term List
| . . 16. ∀l2:Term List.
| . .     ||v|| = ||l2||
| . .     ⇒
| . .     (∀ls:SimpleType List.
| . .      ||ls|| = ||v||
| . .      ⇒
| . .      (∀f:reduce(λs,m. [[s]] rho→m;Prop;ls).
| . .       (∀i:ℕ||v||. trace_consistent(rho;daa;tr.proj;v[i]))
| . .       ⇒
| . .       (∀i:ℕ||l2||. trace_consistent(rho;daa;tr.proj;l2[i]))
| . .       ⇒
| . .       ||ls|| = ||v||
| . .       & (∀i:ℕ. i<||v|| ⇒ ls[i] ∈ term_types(ds;da;de;v[i]))
| . .       ⇒
| . .       ||ls|| = ||l2||
| . .       & (∀i:ℕ. i<||l2|| ⇒ ls[i] ∈ term_types(ds;da;de;l2[i]))
| . .       ⇒
| . .       (∀i:ℕ.
| . .        i<||v|| ⇒ [[v[i]]] e1 s1 a tr = [[l2[i]]] e1 s1 s2 a tr)
| . .       ⇒
| . .       (list_accum(x,t.x([[t]] e1 s1 a tr);f;v)
| . .        ⟺
| . .        list_accum(x,t.x([[t]] e1 s1 s2 a tr);f;l2))))
| . . 17. Term List
| . . 18. u1 : Term
| . . 19. v1 : Term List
| . . 20. ||v||+1 = ||v1||+1
| . . 21. SimpleType List
| . . 22. u2 : SimpleType
| . . 23. v2 : SimpleType List
| . . 24. ||v2||+1 = ||v||+1
```

```
| . . 25. [[u2]] rho→reduce(λs,m. [[s]] rho→m;Prop;v2)
| . . 26. ∀i:ℕ(||v||+1). trace_consistent(rho;daa;tr.proj;[u / v][i])
| . . 27. ∀i:ℕ(||v1||+1).
| . .     trace_consistent(rho;daa;tr.proj;[u1 / v1][i])
| . . 28. ||v2||+1 = ||v||+1
| . .     & (∀i:ℕ.
| . .        i<||v||+1
| . .        ⇒
| . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u / v][i]))
| . . 29. ||v2||+1 = ||v1||+1
| . .     & (∀i:ℕ.
| . .        i<||v1||+1
| . .        ⇒
| . .        [u2 / v2][i] ∈ term_types(ds;da;de;[u1 / v1][i]))
| . . 30. i : ℕ
| . . 31. i<||v||+1
| . . ⊢  [[[u1 / v1][i]]] e1 s1 s2 a tr ∈ [[[u2 / v2][i]]] rho
| . .   by ExRepD THEN Using [‘de’,de] (BackThruLemma ‘term_typing2‘)
| . .
| . . THENA Auto THEN Try (Reduce 0 THEN Auto’)
| . . THEN EasyHyp·
|
|\
| .....set predicate .....
| 1. r1 : rel()
| 2. r2 : rel()
| 3. ds : Collection(dec())
| 4. daa : Collection(dec())
| 5. da : Collection(SimpleType)
| 6. de : sig()
| 7. rho : Decl
| 8. {[[de]] rho}
| 9. {[[ds]] rho}
| 10. {[[ds]] rho}
| 11. [[da]] rho
| 12. tr : trace_env([[daa]] rho)
| 13. trace_consistent_rel(rho;daa;tr.proj;r1)
| 14. trace_consistent_rel(rho;daa;tr.proj;r2)
| 15. tc(r1;ds;da;de)
| 16. tc(r2;ds;da;de)
| 17. r1.name = r2.name
| ⊢  0≤||r1.args||  by Auto
|
.\
. .....set predicate .....
. 1. r1 : rel()
. 2. r2 : rel()
. 3. ds : Collection(dec())
. 4. daa : Collection(dec())
. 5. da : Collection(SimpleType)
. 6. de : sig()
. 7. rho : Decl
. 8. {[[de]] rho}
. 9. {[[ds]] rho}
. 10. {[[ds]] rho}
```

```
. 11. [[da]] rho
. 12. tr : trace_env([[daa]] rho)
. 13. trace_consistent_rel(rho;daa;tr.proj;r1)
. 14. trace_consistent_rel(rho;daa;tr.proj;r2)
. 15. tc(r1;ds;da;de)
. 16. tc(r2;ds;da;de)
. 17. r1.name = r2.name
. ⊢  0≤||r2.args||  by Auto
END
```