```
term_mng_equal  THM

⊢  ∀ds,da:Collection(dec()), st:Collection(SimpleType), de:sig()
|  , rho:Decl, e1:{1of([[de]] rho)}, s1,s2:{[[ds]] rho}, a:[[st]] rho
|  , tr:trace_env([[da]] rho), u:Term, t:SimpleType.
|  trace_consistent(rho;da;tr.proj;u) ⇒
|  (∀x:Label. (x ∈ term_vars(u)) ⇒ s1.x = s2.x) ⇒
|  t ∈ term_types(ds;st;de;u) ⇒ [[u]] e1 s1 a tr = [[u]] e1 s2 a tr
| by RepeatFor 11 (D 0 THENA (Auto THEN Fold `decl` 0 THEN Auto))
|
1. ds : Collection(dec())
2. da : Collection(dec())
3. st : Collection(SimpleType)
4. de : sig()
5. rho : Decl
6. e1 : {1of([[de]] rho)}
7. s1 : {[[ds]] rho}
8. s2 : {[[ds]] rho}
9. a : [[st]] rho
10. tr : trace_env([[da]] rho)
11. u : Term
⊢  ∀t:SimpleType.
|  trace_consistent(rho;da;tr.proj;u) ⇒
|  (∀x:Label. (x ∈ term_vars(u)) ⇒ s1.x = s2.x) ⇒
|  t ∈ term_types(ds;st;de;u) ⇒ [[u]] e1 s1 a tr = [[u]] e1 s2 a tr
| by  TermInd (-1) THEN Reduce 0

|    THEN Try (Auto THEN Try (Fold `decl` 0 THEN Auto)

|         THEN ((InstHyp [x1] (-2)) ORELSE (InstHyp [x
|    ] (-2))) THEN Auto

|         THEN Try (RWW "member_singleton" 0 THEN Auto)

|         THEN RWW "member_dec_lookup" (-2) THENA Auto

|         THEN HypSubst (-1) 0 THEN Auto THEN Try (Fold `member` 0)

|         THEN Try (BackThruLemma `decls_mng_subtype` THEN Auto))

|    THEN Try (Auto THEN Try (Fold `decl` 0 THEN Auto)

|             THEN RW ColMemberC (-1)
|        THEN HypSubstSq (-1) 0
|
|        THEN Try (Unfold `r_select` 0)

|             THEN AllHyps (\i. Unfold `sig_mng` i THEN

|                               Unfold `record` i THEN

|                               Unfold `decl_type` i THEN Reduce i)

|             THEN Auto)
THEN Try ( Auto
```

```
|               THEN Try (BackThruLemma `sts_mng_subtype` THEN Auto)
|               THEN Try (Fold `decl` 0 THEN Auto))
|
|
|
.
|
|\
| 1. ds : Collection(dec())
| 2. da : Collection(dec())
| 3. st : Collection(SimpleType)
| 4. de : sig()
| 5. rho : Decl
| 6. e1 : l:Label→[[de.fun(l)]] rho
| 7. s1 : {[[ds]] rho}
| 8. s2 : {[[ds]] rho}
| 9. a : [[st]] rho
| 10. tr : trace_env([[da]] rho)
| 11. Term
| 12. u1 : Term→Type
| 13. u:{v:Term| u1(v) }→∀t:SimpleType.
|     trace_consistent(rho;da;tr.proj;u) ⇒
|     (∀x:Label. (x ∈ term_vars(u)) ⇒ s1.x = s2.x) ⇒
|     t ∈ term_types(ds;st;de;u) ⇒
|     [[u]] e1 s1 a tr = [[u]] e1 s2 a tr
| 14. y1 : Label
| 15. t : SimpleType
| 16. trace_consistent(rho;da;tr.proj;trace(y1))
| 17. ∀x@0:Label. (x@0 ∈ nil) ⇒ s1.x@0 = s2.x@0
| 18. t = lbl_pr(<Trace, y1>)
| ⊢  tr.y1 = tr.y1
|   by Fold `member` 0 THEN BackThruLemma `tproj_w_f2` THEN Auto
|
.\
. 1. ds : Collection(dec())
. 2. da : Collection(dec())
. 3. st : Collection(SimpleType)
. 4. de : sig()
. 5. rho : Decl
. 6. e1 : {1of([[de]] rho)}
. 7. s1 : {[[ds]] rho}
. 8. s2 : {[[ds]] rho}
. 9. a : [[st]] rho
. 10. tr : trace_env([[da]] rho)
. 11. Term
. 12. u1 : Term→Type
. 13. u:{v:Term| u1(v) }→∀t:SimpleType.
. |   trace_consistent(rho;da;tr.proj;u) ⇒
. |   (∀x:Label. (x ∈ term_vars(u)) ⇒ s1.x = s2.x) ⇒
. |   t ∈ term_types(ds;st;de;u) ⇒
. |   [[u]] e1 s1 a tr = [[u]] e1 s2 a tr
. 14. y1 : {v:Term| u1(v) }
. 15. y2 : {v:Term| u1(v) }
```

```
. 16. t : SimpleType
. 17. trace_consistent(rho;da;tr.proj;y1 y2)
. 18. ∀x:Label. (x ∈ term_vars(y1) @ term_vars(y2)) ⇒ s1.x = s2.x
. 19. t ∈ st_app(term_types(ds;st;de;y1);term_types(ds;st;de;y2))
. ⊢  [[y1]] e1 s1 a tr([[y2]] e1 s1 a tr)
. |  =
. |  [[y1]] e1 s2 a tr([[y2]] e1 s2 a tr)
. | by RWW "member_st_app" (-1) THENA Auto THEN ExRepD
.
. |       THEN AllHyps (InstHyp [y1;a1→t]) THENA Auto
.
. |       THEN Try (BackThruSomeHyp THEN RWW "member_append" 0
.
. |                 THENA Auto THEN SimpConcl)
. |
. |\
. | .....antecedent .....
. | 1. ds : Collection(dec())
. | 2. da : Collection(dec())
. | 3. st : Collection(SimpleType)
. | 4. de : sig()
. | 5. rho : Decl
. | 6. e1 : {1of([[de]] rho)}
. | 7. s1 : {[[ds]] rho}
. | 8. s2 : {[[ds]] rho}
. | 9. a : [[st]] rho
. | 10. tr : trace_env([[da]] rho)
. | 11. Term
. | 12. u1 : Term→Type
. | 13. u:{v:Term| u1(v) }→∀t:SimpleType.
. |     trace_consistent(rho;da;tr.proj;u) ⇒
. |     (∀x:Label. (x ∈ term_vars(u)) ⇒ s1.x = s2.x) ⇒
. |     t ∈ term_types(ds;st;de;u) ⇒
. |     [[u]] e1 s1 a tr = [[u]] e1 s2 a tr
. | 14. y1 : {v:Term| u1(v) }
. | 15. y2 : {v:Term| u1(v) }
. | 16. t : SimpleType
. | 17. trace_consistent(rho;da;tr.proj;y1 y2)
. | 18. ∀x:Label. (x ∈ term_vars(y1) @ term_vars(y2)) ⇒ s1.x = s2.x
. | 19. a1 : SimpleType
. | 20. a1 ∈ term_types(ds;st;de;y2)
. | 21. a1→t ∈ term_types(ds;st;de;y1)
. | ⊢  trace_consistent(rho;da;tr.proj;y1)
. |   by AllHyps (\h. RepeatFor 2 (ParallelOp h) THEN ParallelOp (-1)
. |
. | THEN Reduce 0 THEN RW assert_pushdownC 0 THENA Auto
. |
. |      THEN Complete SimpConcl)

. |
. .\
. . 1. ds : Collection(dec())
. . 2. da : Collection(dec())
. . 3. st : Collection(SimpleType)
. . 4. de : sig()
. . 5. rho : Decl
```

```
. . 6. e1 : {1of([[de]] rho)}
. . 7. s1 : {[[ds]] rho}
. . 8. s2 : {[[ds]] rho}
. . 9. a : [[st]] rho
. . 10. tr : trace_env([[da]] rho)
. . 11. Term
. . 12. u1 : Term→Type
. . 13. u:{v:Term| u1(v) }→∀t:SimpleType.
. . |   trace_consistent(rho;da;tr.proj;u) ⇒
. . |   (∀x:Label. (x ∈ term_vars(u)) ⇒ s1.x = s2.x) ⇒
. . |   t ∈ term_types(ds;st;de;u) ⇒
. . |   [[u]] e1 s1 a tr = [[u]] e1 s2 a tr
. . 14. y1 : {v:Term| u1(v) }
. . 15. y2 : {v:Term| u1(v) }
. . 16. t : SimpleType
. . 17. trace_consistent(rho;da;tr.proj;y1 y2)
. . 18. ∀x:Label. (x ∈ term_vars(y1) @ term_vars(y2)) ⇒ s1.x = s2.x
. . 19. a1 : SimpleType
. . 20. a1 ∈ term_types(ds;st;de;y2)
. . 21. a1→t ∈ term_types(ds;st;de;y1)
. . 22. [[y1]] e1 s1 a tr = [[y1]] e1 s2 a tr
. . ⊢  [[y1]] e1 s1 a tr([[y2]] e1 s1 a tr)
. . |  =
. . |  [[y1]] e1 s2 a tr([[y2]] e1 s2 a tr)
. . | by AllHyps (InstHyp [y2;a1]) THENA Auto
. .
. . |       THEN Try (BackThruSomeHyp THEN RWW "member_append" 0
. .
. . |                 THENA Auto THEN SimpConcl)
. . |
. . |\
. . | .....antecedent .....
. . | 1. ds : Collection(dec())
. . | 2. da : Collection(dec())
. . | 3. st : Collection(SimpleType)
. . | 4. de : sig()
. . | 5. rho : Decl
. . | 6. e1 : {1of([[de]] rho)}
. . | 7. s1 : {[[ds]] rho}
. . | 8. s2 : {[[ds]] rho}
. . | 9. a : [[st]] rho
. . | 10. tr : trace_env([[da]] rho)
. . | 11. Term
. . | 12. u1 : Term→Type
. . | 13. u:{v:Term| u1(v) }→∀t:SimpleType.
. . |     trace_consistent(rho;da;tr.proj;u) ⇒
. . |     (∀x:Label. (x ∈ term_vars(u)) ⇒ s1.x = s2.x) ⇒
. . |     t ∈ term_types(ds;st;de;u) ⇒
. . |     [[u]] e1 s1 a tr = [[u]] e1 s2 a tr
. . | 14. y1 : {v:Term| u1(v) }
. . | 15. y2 : {v:Term| u1(v) }
. . | 16. t : SimpleType
. . | 17. trace_consistent(rho;da;tr.proj;y1 y2)
. . | 18. ∀x:Label. (x ∈ term_vars(y1) @ term_vars(y2)) ⇒ s1.x = s2.x
. . | 19. a1 : SimpleType
```

```
. . | 20. a1 ∈ term_types(ds;st;de;y2)
. . | 21. a1→t ∈ term_types(ds;st;de;y1)
. . | 22. [[y1]] e1 s1 a tr = [[y1]] e1 s2 a tr
. . | ⊢  trace_consistent(rho;da;tr.proj;y2)
. . |   by ...
. . | THEN Reduce 0 THEN RW assert_pushdownC 0 THENA Auto
. . |
. . |      THEN Complete SimpConcl)
. . |
. . .\
. . . 1. ds : Collection(dec())
. . . 2. da : Collection(dec())
. . . 3. st : Collection(SimpleType)
. . . 4. de : sig()
. . . 5. rho : Decl
. . . 6. e1 : {1of([[de]] rho)}
. . . 7. s1 : {[[ds]] rho}
. . . 8. s2 : {[[ds]] rho}
. . . 9. a : [[st]] rho
. . . 10. tr : trace_env([[da]] rho)
. . . 11. Term
. . . 12. u1 : Term→Type
. . . 13. u:{v:Term| u1(v) }→∀t:SimpleType.
. . .     trace_consistent(rho;da;tr.proj;u) ⇒
. . .     (∀x:Label. (x ∈ term_vars(u)) ⇒ s1.x = s2.x) ⇒
. . .     t ∈ term_types(ds;st;de;u) ⇒
. . .     [[u]] e1 s1 a tr = [[u]] e1 s2 a tr
. . . 14. y1 : {v:Term| u1(v) }
. . . 15. y2 : {v:Term| u1(v) }
. . . 16. t : SimpleType
. . . 17. trace_consistent(rho;da;tr.proj;y1 y2)
. . . 18. ∀x:Label. (x ∈ term_vars(y1) @ term_vars(y2)) ⇒ s1.x = s2.x
. . . 19. a1 : SimpleType
. . . 20. a1 ∈ term_types(ds;st;de;y2)
. . . 21. a1→t ∈ term_types(ds;st;de;y1)
. . . 22. [[y1]] e1 s1 a tr = [[y1]] e1 s2 a tr
. . . 23. [[y2]] e1 s1 a tr = [[y2]] e1 s2 a tr
. . . ⊢  [[y1]] e1 s1 a tr([[y2]] e1 s1 a tr)

. . .    =
. . .    [[y1]] e1 s2 a tr([[y2]] e1 s2 a tr)
. . .   by Reduce (-2) THEN AppHyps (-1) (-2)
END
```