```
not_disjoint_is_member_lemma  THM

⊢  ∀eq:{T=₂}, L,M:T List.
|  ¬disjoint(eq;L;M) ⇒ (∃x:T. x(∈eq) L & x(∈eq) M)  by UnivCD
|
1. T : Type
2. eq : {T=₂}
3. L : T List
4. M : T List
5. ¬disjoint(eq;L;M)
⊢  ∃x:T. x(∈eq) L & x(∈eq) M  by DiscreteEq 2
|
3. eq ∈ T→T→𝔹
4. ∀x,y:T. eq(x,y) ⟺ x = y
5. L : T List
6. M : T List
7. ¬disjoint(eq;L;M)
⊢  ∃x:T. x(∈eq) L & x(∈eq) M  by OnHyps [-1;-1] MoveToConcl
|
5. L : T List
⊢  ∀M:T List. ¬disjoint(eq;L;M) ⇒ (∃x:T. x(∈eq) L & x(∈eq) M)
| by ListInd -1
|
|\
| ⊢  ∀M:T List. ¬disjoint(eq;nil;M) ⇒ (∃x:T. x(∈eq) nil & x(∈eq) M)
| | by Reduce 0 THEN Analyze 0
| |
| 6. M : T List
| ⊢  ¬True ⇒ (∃x:T. False & x(∈eq) M)  by ListInd -1
| |
| |\
| | ⊢  ¬True ⇒ (∃x:T. False & x(∈eq) nil)  by Reduce 0
| | |
| | 7. ¬True
| | ⊢  ∃x:T. False & False  by Analyze -1 THEN Trivial
| |
| .\
| . 7. u : T
| . 8. v : T List
| . 9. ¬True ⇒ (∃x:T. False & x(∈eq) v)
| . ⊢  ¬True ⇒ (∃x:T. False & x(∈eq) (u.v))
| . | by OnHyps [-1;0] Reduce THEN Analyze 0
| . |
| . 10. ¬True
| . ⊢  ∃x:T. False & if eq(x,u)→ true₂ else x(∈eq) v fi
| . | by Analyze -2
| . |
| . 9. ¬True
| . 10. ∃x:T. False & x(∈eq) v
| . ⊢  ∃x:T. False & if eq(x,u)→ true₂ else x(∈eq) v fi
| . | by Analyze -1 THEN Analyze -1
| . |
| . 10. x : T
| . 11. False
| . 12. x(∈eq) v
```

```
| . ⊢  ∃x:T. False & if eq(x,u)→ true₂ else x(∈eq) v fi  by Auto
|
.\
. 6. u : T
. 7. v : T List
. 8. ∀M:T List. ¬disjoint(eq;v;M) ⇒ (∃x:T. x(∈eq) v & x(∈eq) M)
. ⊢  ∀M:T List.
. |  ¬disjoint(eq;(u.v);M) ⇒ (∃x:T. x(∈eq) (u.v) & x(∈eq) M)
. | by Reduce 0 THEN Analyze 0
. |
. 9. M : T List
. ⊢  ¬(¬₂u(∈eq) M & disjoint(eq;v;M))
. |  ⇒
. |  (∃x:T. if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M)
. | by Analyze 0
. |
. 10. ¬(¬₂u(∈eq) M & disjoint(eq;v;M))
. ⊢  ∃x:T. if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M
. | by RWH (LemmaC Thm* ∀p:𝔹. ¬₂p ⟺ ¬p    {assert_of_bnot}) -1
. |
. 10. ¬(¬u(∈eq) M & disjoint(eq;v;M))
. ⊢  ∃x:T. if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M
. | by RWH
. |    (LemmaC
. |     Thm* Dec(P) ⇒ Dec(Q) ⇒ (¬(P & Q) ⟺ ¬P ∨ ¬Q)    {demorgan2})
. |    -1
. |
. 10. ¬¬u(∈eq) M ∨ ¬disjoint(eq;v;M)
. ⊢  ∃x:T. if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M
. | by Analyze -1
. |
. |\
. | 10. ¬¬u(∈eq) M
. | ⊢  ∃x:T. if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M
. | | by Witness u
. | |
. | ⊢  if eq(u,u)→ true₂ else u(∈eq) v fi & u(∈eq) M
. | | by IfThenElseCases 0
. | |
. | |\
. | | .....truecase .....
. | | 11. eq(u,u) = true₂
. | | ⊢  true₂ & u(∈eq) M  by Reduce 0
. | | |
. | | ⊢  u(∈eq) M  by Assert Stable{u(∈eq) M}
. | | |
. | | |\
. | | | .....assertion .....
. | | | ⊢  Stable{u(∈eq) M}  by ProveStable
. | | |
. | | .\
. | | . 12. Stable{u(∈eq) M}
. | | . ⊢  u(∈eq) M  by UnfoldTopAb -1
. | | . |
. | | . 12. ¬¬u(∈eq) M ⇒ u(∈eq) M
```

```
. | | . ⊢  u(∈eq) M  by HypBackchain
. | |
. | .\
. | . .....falsecase .....
. | . 11. eq(u,u) = false₂
. | . ⊢  u(∈eq) v & u(∈eq) M  by Witness4 u THEN With u (Analyze -1)
. | . |
. | . 4. T List
. | . 5. u : T
. | . 6. v : T List
. | . 7. ∀M:T List. ¬disjoint(eq;v;M) ⇒ (∃x:T. x(∈eq) v & x(∈eq) M)
. | . 8. M : T List
. | . 9. ¬¬u(∈eq) M
. | . 10. eq(u,u) = false₂
. | . 11. eq(u,u) ⟺ u = u
. | . ⊢  u(∈eq) v & u(∈eq) M  by HypSubst -2 -1 THEN Reduce -1
. | . |
. | . 11. False ⟺ u = u
. | . ⊢  u(∈eq) v & u(∈eq) M  by Analyze -1
. | . |
. | . 11. False ⇒ u = u
. | . 12. False ⇐ (u = u)
. | . ⊢  u(∈eq) v & u(∈eq) M  by Analyze -1
. |
. .\
. . 10. ¬disjoint(eq;v;M)
. . ⊢  ∃x:T. if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M
. . | by With M (Analyze -3)
. . |
. . 8. M : T List
. . 9. ¬disjoint(eq;v;M)
. . 10. ¬disjoint(eq;v;M) ⇒ (∃x:T. x(∈eq) v & x(∈eq) M)
. . ⊢  ∃x:T. if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M
. . | by Analyze -1
. . |
. . 10. ∃x:T. x(∈eq) v & x(∈eq) M
. . ⊢  ∃x:T. if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M
. . | by Analyze -1 THEN Analyze -1
. . |
. . 10. x : T
. . 11. x(∈eq) v
. . 12. x(∈eq) M
. . ⊢  ∃x:T. if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M
. . | by Witness x
. . |
. . ⊢  if eq(x,u)→ true₂ else x(∈eq) v fi & x(∈eq) M
. .   by IfThenElseCases 0 THEN Reduce 0
END
```