```
l_member_decomp2  THM

⊢  ∀T:Type.
|  (∀x,y:T. Dec(x = y))
|  ⇒
|  (∀s:T List, z:T.
|   (z ∈ s) ⇒ (∃s1,s2:T List. s = (s1 @ [z / s2]) & ¬(z ∈ s1)))
| by InductionOnList
|
|\
| 1. T : Type
| 2. ∀x,y:T. Dec(x = y)
| 3. T List
| 4. z : T
| 5. (z ∈ nil)
| ⊢  ∃s1,s2:T List. nil = (s1 @ [z / s2]) & ¬(z ∈ s1)
|   by ObviousContradiction [-1]
|
.\
. 1. T : Type
. 2. ∀x,y:T. Dec(x = y)
. 3. T List
. 4. u : T
. 5. v : T List
. 6. ∀z:T. (z ∈ v) ⇒ (∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1))
. 7. z : T
. 8. (z ∈ [u / v])
. ⊢  ∃s1,s2:T List. [u / v] = (s1 @ [z / s2]) & ¬(z ∈ s1)
. | by (InstHyp [z;u] 2) THEN (Analyze -1)
. |
. |\
. | 1. T : Type
. | 2. ∀x,y:T. Dec(x = y)
. | 3. T List
. | 4. u : T
. | 5. v : T List
. | 6. ∀z:T.
. | |  (z ∈ v) ⇒ (∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1))
. | 7. z : T
. | 8. (z ∈ [u / v])
. | 9. z = u
. | ⊢  ∃s1,s2:T List. [u / v] = (s1 @ [z / s2]) & ¬(z ∈ s1)
. | | by (InstConcl [nil;v]) THEN (Reduce 0)
. | |
. | |\
. | | 1. T : Type
. | | 2. ∀x,y:T. Dec(x = y)
. | | 3. T List
. | | 4. u : T
. | | 5. v : T List
. | | 6. ∀z:T.
. | |    (z ∈ v) ⇒ (∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1))
. | | 7. z : T
. | | 8. (z ∈ [u / v])
. | | 9. z = u
```

```
. | | ⊢  [u / v] = [z / v]  by Analyze
. | |
. | .\
. | . 1. T : Type
. | . 2. ∀x,y:T. Dec(x = y)
. | . 3. T List
. | . 4. u : T
. | . 5. v : T List
. | . 6. ∀z:T.
. | .    (z ∈ v) ⇒ (∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1))
. | . 7. z : T
. | . 8. (z ∈ [u / v])
. | . 9. z = u
. | . ⊢  ¬(z ∈ nil)  by ObviousConcl
. |
. .\
. . 1. T : Type
. . 2. ∀x,y:T. Dec(x = y)
. . 3. T List
. . 4. u : T
. . 5. v : T List
. . 6. ∀z:T.
. . |  (z ∈ v) ⇒ (∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1))
. . 7. z : T
. . 8. (z ∈ [u / v])
. . 9. ¬z = u
. . ⊢  ∃s1,s2:T List. [u / v] = (s1 @ [z / s2]) & ¬(z ∈ s1)
. . | by InstHyp [z] -4
. . |
. . |\
. . | .....antecedent .....
. . | 1. T : Type
. . | 2. ∀x,y:T. Dec(x = y)
. . | 3. T List
. . | 4. u : T
. . | 5. v : T List
. . | 6. ∀z:T.
. . |    (z ∈ v) ⇒ (∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1))
. . | 7. z : T
. . | 8. (z ∈ [u / v])
. . | 9. ¬z = u
. . | ⊢  (z ∈ v)
. . |   by (RWO
. . |       Thm* ∀l:T List, a,x:T. (x ∈ [a / l]) ⟺ x = a ∨ (x ∈ l)
. . |       {cons_member}
. . |       -2)
. . |      THEN
. . |      (Analyze -2)
. . |
. . .\
. . . 1. T : Type
. . . 2. ∀x,y:T. Dec(x = y)
. . . 3. T List
. . . 4. u : T
. . . 5. v : T List
```

```
. . . 6. ∀z:T.
. . . |  (z ∈ v) ⇒ (∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1))
. . . 7. z : T
. . . 8. (z ∈ [u / v])
. . . 9. ¬z = u
. . . 10. ∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1)
. . . ⊢  ∃s1,s2:T List. [u / v] = (s1 @ [z / s2]) & ¬(z ∈ s1)
. . . | by ExRepD THEN (InstConcl [[u / s1];s2]) THEN (Reduce 0)
. . . |
. . . |\
. . . | 1. T : Type
. . . | 2. ∀x,y:T. Dec(x = y)
. . . | 3. T List
. . . | 4. u : T
. . . | 5. v : T List
. . . | 6. ∀z:T.
. . . |    (z ∈ v) ⇒ (∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1))
. . . | 7. z : T
. . . | 8. (z ∈ [u / v])
. . . | 9. ¬z = u
. . . | 10. s1 : T List
. . . | 11. s2 : T List
. . . | 12. v = (s1 @ [z / s2])
. . . | 13. ¬(z ∈ s1)
. . . | ⊢  [u / v] = [u / (s1 @ [z / s2])]  by Analyze
. . . |
. . . .\
. . . . 1. T : Type
. . . . 2. ∀x,y:T. Dec(x = y)
. . . . 3. T List
. . . . 4. u : T
. . . . 5. v : T List
. . . . 6. ∀z:T.
. . . .    (z ∈ v) ⇒ (∃s1,s2:T List. v = (s1 @ [z / s2]) & ¬(z ∈ s1))
. . . . 7. z : T
. . . . 8. (z ∈ [u / v])
. . . . 9. ¬z = u
. . . . 10. s1 : T List
. . . . 11. s2 : T List
. . . . 12. v = (s1 @ [z / s2])
. . . . 13. ¬(z ∈ s1)
. . . . ⊢  ¬(z ∈ [u / s1])
. . . .   by (Analyze 0)
. . . .      THEN
. . . .      (RWO
. . . .       Thm* ∀l:T List, a,x:T. (x ∈ [a / l]) ⟺ x = a ∨ (x ∈ l)
. . . .       {cons_member}
. . . .       -1)
. . . .      THEN
. . . .      (Analyze -1)
END
```