```
list_exists_append_lemma  THM

⊢  ∀P:(T→Prop), L,M:T List. ∃x∈(L @ M).P(x) ⟺ ∃x∈L.P(x) ∨ ∃x∈M.P(x)
| by UnivCD THEN Repeat (Analyze 0)
|
|\
| 1. T : Type
| 2. P : T→Prop
| 3. L : T List
| 4. M : T List
| 5. ∃x∈(L @ M).P(x)
| ⊢  ∃x∈L.P(x) ∨ ∃x∈M.P(x)
| | by MoveToConcl 4 THEN ListInd 3 THEN Reduce 0
| |
| |\
| | 3. T List
| | ⊢  ∀M:T List. ∃x∈M.P(x) ⇒ False ∨ ∃x∈M.P(x)  by Auto
| | |
| | 4. M : T List
| | 5. ∃x∈M.P(x)
| | ⊢  False ∨ ∃x∈M.P(x)  by Choose [2] THEN Trivial
| |
| .\
| . 4. u : T
| . 5. v : T List
| . 6. ∀M:T List. ∃x∈(v @ M).P(x) ⇒ ∃x∈v.P(x) ∨ ∃x∈M.P(x)
| . ⊢  ∀M:T List.
| . |  P(u) ∨ ∃x∈(v @ M).P(x) ⇒ P(u) ∨ ∃x∈v.P(x) ∨ ∃x∈M.P(x)
| . | by UnivCD
| . |
| . 7. M : T List
| . 8. P(u) ∨ ∃x∈(v @ M).P(x)
| . ⊢  P(u) ∨ ∃x∈v.P(x) ∨ ∃x∈M.P(x)  by Analyze -1
| . |
| . |\
| . | 8. P(u)
| . | ⊢  P(u) ∨ ∃x∈v.P(x) ∨ ∃x∈M.P(x)  by Choose [1;1] THEN Trivial
| . |
| . .\
| . . 8. ∃x∈(v @ M).P(x)
| . . ⊢  P(u) ∨ ∃x∈v.P(x) ∨ ∃x∈M.P(x)  by FwdThru 6 [-1]
| . . |
| . . 9. ∃x∈v.P(x) ∨ ∃x∈M.P(x)
| . . ⊢  P(u) ∨ ∃x∈v.P(x) ∨ ∃x∈M.P(x)  by Analyze -1
| . . |
| . . |\
| . . | 9. ∃x∈v.P(x)
| . . | ⊢  P(u) ∨ ∃x∈v.P(x) ∨ ∃x∈M.P(x)  by Choose [1;2] THEN Trivial
| . . |
| . . .\
| . . . 9. ∃x∈M.P(x)
| . . . ⊢  P(u) ∨ ∃x∈v.P(x) ∨ ∃x∈M.P(x)  by Choose [2] THEN Trivial
|
.\
. 1. T : Type
```

```
. 2. P : T→Prop
. 3. L : T List
. 4. M : T List
. 5. ∃x∈L.P(x) ∨ ∃x∈M.P(x)
. ⊢  ∃x∈(L @ M).P(x)  by MoveToConcl -2 THEN ListInd 3 THEN Reduce 0
. |
. |\
. | 3. T List
. | ⊢  ∀L:T List. ∃x∈L.P(x) ∨ False ⇒ ∃x∈(L @ nil).P(x)  by Auto
. | |
. | 4. L : T List
. | 5. ∃x∈L.P(x) ∨ False
. | ⊢  ∃x∈(L @ nil).P(x)  by Analyze -1
. | |
. | |\
. | | 5. ∃x∈L.P(x)
. | | ⊢  ∃x∈(L @ nil).P(x)
. | | | by RWH
. | | |    (LemmaC
. | | |     Thm* ∀L:T List. (L @ nil) = L
. | | |     {append_nil_right_identity})
. | | |    0
. | | |
. | | ⊢  ∃x∈L.P(x)  by Trivial
. | |
. | .\
. | . 5. False
. | . ⊢  ∃x∈(L @ nil).P(x)  by Trivial
. |
. .\
. . 3. T List
. . 4. u : T
. . 5. v : T List
. . 6. ∀L:T List. ∃x∈L.P(x) ∨ ∃x∈v.P(x) ⇒ ∃x∈(L @ v).P(x)
. . ⊢  ∀L:T List. ∃x∈L.P(x) ∨ P(u) ∨ ∃x∈v.P(x) ⇒ ∃x∈(L @ (u.v)).P(x)
. . | by UnivCD
. . |
. . 7. L : T List
. . 8. ∃x∈L.P(x) ∨ P(u) ∨ ∃x∈v.P(x)
. . ⊢  ∃x∈(L @ (u.v)).P(x)  by ListInd -2 THEN Reduce 0
. . |
. . |\
. . | 7. T List
. . | ⊢  False ∨ P(u) ∨ ∃x∈v.P(x) ⇒ P(u) ∨ ∃x∈v.P(x)  by Auto
. . | |
. . | 8. False ∨ P(u) ∨ ∃x∈v.P(x)
. . | ⊢  P(u) ∨ ∃x∈v.P(x)  by Analyze -1
. . | |
. . | |\
. . | | 8. False
. . | | ⊢  P(u) ∨ ∃x∈v.P(x)  by Trivial
. . | |
. . | .\
. . | . 8. P(u) ∨ ∃x∈v.P(x)
. . | . ⊢  P(u) ∨ ∃x∈v.P(x)  by Trivial
```

```
. . |
. . .\
. . . 8. u1 : T
. . . 9. v1 : T List
. . . 10. ∃x∈v1.P(x) ∨ P(u) ∨ ∃x∈v.P(x) ⇒ ∃x∈(v1 @ (u.v)).P(x)
. . . ⊢  P(u1) ∨ ∃x∈v1.P(x) ∨ P(u) ∨ ∃x∈v.P(x)
. . . |  ⇒
. . . |  P(u1) ∨ ∃x∈(v1 @ (u.v)).P(x)  by Auto
. . . |
. . . 11. P(u1) ∨ ∃x∈v1.P(x) ∨ P(u) ∨ ∃x∈v.P(x)
. . . ⊢  P(u1) ∨ ∃x∈(v1 @ (u.v)).P(x)  by Analyze -1
. . . |
. . . |\
. . . | 11. P(u1) ∨ ∃x∈v1.P(x)
. . . | ⊢  P(u1) ∨ ∃x∈(v1 @ (u.v)).P(x)  by Analyze -1
. . . | |
. . . | |\
. . . | | 11. P(u1)
. . . | | ⊢  P(u1) ∨ ∃x∈(v1 @ (u.v)).P(x)  by Choose [1] THEN Trivial
. . . | |
. . . | .\
. . . | . 11. ∃x∈v1.P(x)
. . . | . ⊢  P(u1) ∨ ∃x∈(v1 @ (u.v)).P(x)
. . . | . | by Choose [2] THEN Analyze -2
. . . | . |
. . . | . |\
. . . | . | .....antecedent .....
. . . | . | 10. ∃x∈v1.P(x)
. . . | . | ⊢  ∃x∈v1.P(x) ∨ P(u) ∨ ∃x∈v.P(x)
. . . | . |   by Choose [1] THEN Trivial
. . . | . |
. . . | . .\
. . . | . . 10. ∃x∈v1.P(x)
. . . | . . 11. ∃x∈(v1 @ (u.v)).P(x)
. . . | . . ⊢  ∃x∈(v1 @ (u.v)).P(x)  by Trivial
. . . |
. . . .\
. . . . 11. P(u) ∨ ∃x∈v.P(x)
. . . . ⊢  P(u1) ∨ ∃x∈(v1 @ (u.v)).P(x)  by Analyze -1
. . . . |
. . . . |\
. . . . | 11. P(u)
. . . . | ⊢  P(u1) ∨ ∃x∈(v1 @ (u.v)).P(x)
. . . . | | by Choose [2] THEN HypBackchain
. . . . | |
. . . . | ⊢  ∃x∈v1.P(x) ∨ P(u) ∨ ∃x∈v.P(x)
. . . . |   by Choose [2;1] THEN Trivial
. . . . |
. . . . .\
. . . . . 11. ∃x∈v.P(x)
. . . . . ⊢  P(u1) ∨ ∃x∈(v1 @ (u.v)).P(x)
. . . . . | by Choose [2] THEN HypBackchain
. . . . . |
. . . . . ⊢  ∃x∈v1.P(x) ∨ P(u) ∨ ∃x∈v.P(x)
. . . . .   by Choose [2;2] THEN Trivial
```

```
END
```