```
equal_appends_case1  THM

⊢  ∀T:Type, x1,z,x2,x3:T List.
|  ||x1||≤||z||
|  ⇒
|  (x1 @ x2) = (z @ x3)
|  ⇒
|  (∃z':T List. z = (x1 @ z') & x2 = (z' @ x3))
| by InductionOnList THEN (Reduce 0)
|
|\
| 1. T : Type
| 2. T List
| ⊢  ∀z,x2,x3:T List.
|    0≤||z|| ⇒ x2 = (z @ x3) ⇒ (∃z':T List. z = z' & x2 = (z' @ x3))
|   by Auto THEN Obvious
|
.\
. 1. T : Type
. 2. T List
. 3. u : T
. 4. v : T List
. 5. ∀z,x2,x3:T List.
. |  ||v||≤||z||
. |  ⇒
. |  (v @ x2) = (z @ x3)
. |  ⇒
. |  (∃z':T List. z = (v @ z') & x2 = (z' @ x3))
. ⊢  ∀z,x2,x3:T List.
. |  ||v||+1≤||z||
. |  ⇒
. |  [u / (v @ x2)] = (z @ x3)
. |  ⇒
. |  (∃z':T List. z = [u / (v @ z')] & x2 = (z' @ x3))
. | by InductionOnList THEN (Reduce 0)
. |
. |\
. | 1. T : Type
. | 2. T List
. | 3. u : T
. | 4. v : T List
. | 5. ∀z,x2,x3:T List.
. |    ||v||≤||z||
. |    ⇒
. |    (v @ x2) = (z @ x3)
. |    ⇒
. |    (∃z':T List. z = (v @ z') & x2 = (z' @ x3))
. | 6. T List
. | ⊢  ∀x2,x3:T List.
. |    ||v||+1≤0
. |    ⇒
. |    [u / (v @ x2)] = x3
. |    ⇒
. |    (∃z':T List. nil = [u / (v @ z')] & x2 = (z' @ x3))
. |   by Obvious
```

```
. |
. .\
. . 1. T : Type
. . 2. T List
. . 3. u : T
. . 4. v : T List
. . 5. ∀z,x2,x3:T List.
. . |  ||v||≤||z||
. . |  ⇒
. . |  (v @ x2) = (z @ x3)
. . |  ⇒
. . |  (∃z':T List. z = (v @ z') & x2 = (z' @ x3))
. . 6. T List
. . 7. u1 : T
. . 8. v1 : T List
. . 9. ∀x2,x3:T List.
. . |  ||v||+1≤||v1||
. . |  ⇒
. . |  [u / (v @ x2)] = (v1 @ x3)
. . |  ⇒
. . |  (∃z':T List. v1 = [u / (v @ z')] & x2 = (z' @ x3))
. . ⊢  ∀x2,x3:T List.
. . |  ||v||+1≤||v1||+1
. . |  ⇒
. . |  [u / (v @ x2)] = [u1 / (v1 @ x3)]
. . |  ⇒
. . |  (∃z':T List. [u1 / v1] = [u / (v @ z')] & x2 = (z' @ x3))
. . | by Auto THEN (InstHyp [v1;x2;x3] 5)
. . |
. . |\
. . | .....antecedent .....
. . | 1. T : Type
. . | 2. T List
. . | 3. u : T
. . | 4. v : T List
. . | 5. ∀z,x2,x3:T List.
. . |    ||v||≤||z||
. . |    ⇒
. . |    (v @ x2) = (z @ x3)
. . |    ⇒
. . |    (∃z':T List. z = (v @ z') & x2 = (z' @ x3))
. . | 6. T List
. . | 7. u1 : T
. . | 8. v1 : T List
. . | 9. ∀x2,x3:T List.
. . |    ||v||+1≤||v1||
. . |    ⇒
. . |    [u / (v @ x2)] = (v1 @ x3)
. . |    ⇒
. . |    (∃z':T List. v1 = [u / (v @ z')] & x2 = (z' @ x3))
. . | 10. x2 : T List
. . | 11. x3 : T List
. . | 12. ||v||+1≤||v1||+1
. . | 13. [u / (v @ x2)] = [u1 / (v1 @ x3)]
. . | ⊢  (v @ x2) = (v1 @ x3)  by Obvious
```

```
. . |
. . .\
. . . 1. T : Type
. . . 2. T List
. . . 3. u : T
. . . 4. v : T List
. . . 5. ∀z,x2,x3:T List.
. . . |  ||v||≤||z||
. . . |  ⇒
. . . |  (v @ x2) = (z @ x3)
. . . |  ⇒
. . . |  (∃z':T List. z = (v @ z') & x2 = (z' @ x3))
. . . 6. T List
. . . 7. u1 : T
. . . 8. v1 : T List
. . . 9. ∀x2,x3:T List.
. . . |  ||v||+1≤||v1||
. . . |  ⇒
. . . |  [u / (v @ x2)] = (v1 @ x3)
. . . |  ⇒
. . . |  (∃z':T List. v1 = [u / (v @ z')] & x2 = (z' @ x3))
. . . 10. x2 : T List
. . . 11. x3 : T List
. . . 12. ||v||+1≤||v1||+1
. . . 13. [u / (v @ x2)] = [u1 / (v1 @ x3)]
. . . 14. ∃z':T List. v1 = (v @ z') & x2 = (z' @ x3)
. . . ⊢  ∃z':T List. [u1 / v1] = [u / (v @ z')] & x2 = (z' @ x3)
. . . | by ExRepD THEN (HypSubst -1 0)
. . . |
. . . 1. T : Type
. . . 2. T List
. . . 3. u : T
. . . 4. v : T List
. . . 5. ∀z,x2,x3:T List.
. . .    ||v||≤||z||
. . .    ⇒
. . .    (v @ x2) = (z @ x3)
. . .    ⇒
. . .    (∃z':T List. z = (v @ z') & x2 = (z' @ x3))
. . . 6. T List
. . . 7. u1 : T
. . . 8. v1 : T List
. . . 9. ∀x2,x3:T List.
. . .    ||v||+1≤||v1||
. . .    ⇒
. . .    [u / (v @ x2)] = (v1 @ x3)
. . .    ⇒
. . .    (∃z':T List. v1 = [u / (v @ z')] & x2 = (z' @ x3))
. . . 10. x2 : T List
. . . 11. x3 : T List
. . . 12. ||v||+1≤||v1||+1
. . . 13. [u / (v @ x2)] = [u1 / (v1 @ x3)]
. . . 14. z' : T List
. . . 15. v1 = (v @ z')
. . . 16. x2 = (z' @ x3)
```

```
. . . ⊢  ∃z'@0:T List.
. . .     [u1 / v1] = [u / (v @ z'@0)] & (z' @ x3) = (z'@0 @ x3)
. . .    by (InstConcl [z']) THEN Analyze THEN Obvious
END
```