```
total_back_listify  THM

⊢  ∀S:ActionSet(Alph), sL:S.car*.
|  Fin(Alph) ⇒
|  Fin(S.car) ⇒
|  (∃TBL:S.car*.
|   ∀s:S.car.
|   mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| by RepD
|
1. Alph : Type
2. S : ActionSet(Alph)
3. sL : S.car*
4. Fin(Alph)
5. Fin(S.car)
⊢  ∃TBL:S.car*.
|  ∀s:S.car. mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| by Assert
|    (∀n:ℕ.
|     ∃TBL:S.car*.
|     (∀s:S.car.
|      mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
|     ∨ ||TBL|| = n
|       & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
|       & (∀s:S.car.
|          mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
|       & (∃AL:S.car*.
|          (∀s:S.car.
|           mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
|          & (∀s:S.car.
|             mem_f(S.car;s;sL) ⇒
|             mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
|          & (∀s:S.car, a:Alph.
|             mem_f(S.car;S.act(a,s);TBL) ⇒
|             mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))))
|
|\
| .....assertion .....
| ⊢  ∀n:ℕ.
| |  ∃TBL:S.car*.
| |  (∀s:S.car.
| |   mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| |  ∨ ||TBL|| = n
| |    & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
| |    & (∀s:S.car.
| |       mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| |    & (∃AL:S.car*.
| |       (∀s:S.car.
| |        mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| |       & (∀s:S.car.
| |          mem_f(S.car;s;sL) ⇒
| |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
| |       & (∀s:S.car, a:Alph.
| |          mem_f(S.car;S.act(a,s);TBL) ⇒
| |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL)))
```

```
| | by (Analyze 0) THEN (NatInd -1)
| |
| |\
| | .....basecase .....
| | ⊢  ∃TBL:S.car*.
| | |  (∀s:S.car.
| | |   mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| | |  ∨ ||TBL|| = 0
| | |    & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
| | |    & (∀s:S.car.
| | |       mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| | |    & (∃AL:S.car*.
| | |       (∀s:S.car.
| | |        mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| | |       & (∀s:S.car.
| | |          mem_f(S.car;s;sL) ⇒
| | |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
| | |       & (∀s:S.car, a:Alph.
| | |          mem_f(S.car;S.act(a,s);TBL) ⇒
| | |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL)))
| | | by (InstConcl [nil]) THEN (Reduce 0) THEN (Sel 2 (Analyze 0))
| | |
| | ⊢  ∃AL:S.car*.
| | |  (∀s:S.car.
| | |   mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| | |  & (∀s:S.car. mem_f(S.car;s;sL) ⇒ False ∨ mem_f(S.car;s;AL))
| | |  & (∀s:S.car, a:Alph. False ⇒ False ∨ mem_f(S.car;s;AL))
| | | by InstConcl [sL]
| | |
| | |\
| | | 6. s : S.car
| | | 7. mem_f(S.car;s;sL)
| | | ⊢  ∃w:Alph*. mem_f(S.car;(S:w←s);sL)
| | |   by (InstConcl [nil]) THEN (RecUnfold `maction` 0)
| | |      THEN
| | |      (Reduce 0)
| | |
| | .\
| | . 6. s : S.car
| | . 7. mem_f(S.car;s;sL)
| | . ⊢  False ∨ mem_f(S.car;s;sL)  by Sel 2 (Analyze 0)
| |
| .\
| . .....upcase .....
| . 6. n : ℤ
| . 7. 0<n
| . 8. ∃TBL:S.car*.
| . |  (∀s:S.car.
| . |   mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . |  ∨ ||TBL|| = n-1
| . |    & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
| . |    & (∀s:S.car.
| . |       mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . |    & (∃AL:S.car*.
| . |       (∀s:S.car.
```

```
| . |        mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . |       & (∀s:S.car.
| . |          mem_f(S.car;s;sL) ⇒
| . |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
| . |       & (∀s:S.car, a:Alph.
| . |          mem_f(S.car;S.act(a,s);TBL) ⇒
| . |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL)))
| . ⊢  ∃TBL:S.car*.
| . |  (∀s:S.car.
| . |   mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . |  ∨ ||TBL|| = n
| . |    & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
| . |    & (∀s:S.car.
| . |       mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . |    & (∃AL:S.car*.
| . |       (∀s:S.car.
| . |        mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . |       & (∀s:S.car.
| . |          mem_f(S.car;s;sL) ⇒
| . |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
| . |       & (∀s:S.car, a:Alph.
| . |          mem_f(S.car;S.act(a,s);TBL) ⇒
| . |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL)))
| . | by Analyze -1 THEN Analyze -1
| . |
| . |\
| . | 8. TBL : S.car*
| . | 9. ∀s:S.car.
| . |    mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . | ⊢
| . |   by (InstConcl [TBL]) THEN (Sel 1 (Analyze 0))
| . |      THEN
| . |      (NthHyp -1)
| . |
| . .\
| . . 8. TBL : S.car*
| . . 9. ||TBL|| = n-1
| . . |  & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
| . . |  & (∀s:S.car.
| . . |     mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . |  & (∃AL:S.car*.
| . . |     (∀s:S.car.
| . . |      mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . |     & (∀s:S.car.
| . . |        mem_f(S.car;s;sL) ⇒
| . . |        mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
| . . |     & (∀s:S.car, a:Alph.
| . . |        mem_f(S.car;S.act(a,s);TBL) ⇒
| . . |        mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL)))
| . . ⊢   by ExRepD THEN ListInd 12 THEN Reduce 0 THEN RepD
| . . |
| . . |\
| . . | 9. ||TBL|| = n-1
| . . | 10. ∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j]
| . . | 11. ∀s:S.car.
```

```
| . . | |   mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . . | 12. S.car*
| . . | 13. ∀s:S.car. False ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . . | 14. ∀s:S.car. mem_f(S.car;s;sL) ⇒ mem_f(S.car;s;TBL) ∨ False
| . . | 15. ∀s:S.car, a:Alph.
| . . | |   mem_f(S.car;S.act(a,s);TBL) ⇒ mem_f(S.car;s;TBL) ∨ False
| . . | ⊢   by (InstConcl [TBL]) THEN (Sel 1 (Analyze 0))
| . . | |
| . . | |\
| . . | | 16. s : S.car
| . . | | 17. mem_f(S.car;s;TBL)
| . . | | ⊢  ∃w:Alph*. mem_f(S.car;(S:w←s);sL)  by BackThru 11
| . . | |
| . . | .\
| . . | . 16. s : S.car
| . . | . 17. ∃w:Alph*. mem_f(S.car;(S:w←s);sL)
| . . | . ⊢  mem_f(S.car;s;TBL)
| . . | . | by Analyze -1
| . . | . |    THEN
| . . | . |    Assert
| . . | . |    (∀i:ℕ. i≤||w|| ⇒ mem_f(S.car;(S:nth_tl(i;w)←s);TBL))
| . . | . |
| . . | . |\
| . . | . | .....assertion .....
| . . | . | 17. w : Alph*
| . . | . | 18. mem_f(S.car;(S:w←s);sL)
| . . | . | ⊢  ∀i:ℕ. i≤||w|| ⇒ mem_f(S.car;(S:nth_tl(i;w)←s);TBL)
| . . | . | | by (Analyze 0) THEN (NatInd -1) THEN RepD
| . . | . | |
| . . | . | |\
| . . | . | | 19. 0≤||w||
| . . | . | | ⊢  mem_f(S.car;(S:nth_tl(0;w)←s);TBL)
| . . | . | |   by (Reduce 0 THEN FwdThru 14 [18]) THEN (Analyze -1)
| . . | . | |
| . . | . | .\
| . . | . | . 19. i : ℤ
| . . | . | . 20. 0<i
| . . | . | . 21. i-1≤||w|| ⇒ mem_f(S.car;(S:nth_tl(i-1;w)←s);TBL)
| . . | . | . 22. i≤||w||
| . . | . | . ⊢  mem_f(S.car;(S:nth_tl(i;w)←s);TBL)
| . . | . | . | by (Analyze -2) THEN (RecUnfold `maction` -1)
| . . | . | . |    THEN
| . . | . | . |    (MoveToConcl -1)
| . . | . | . |    THEN
| . . | . | . |    SplitOnConclITE
| . . | . | . |    THEN
| . . | . | . |    (Analyze 0)
| . . | . | . |
| . . | . | . |\
| . . | . | . | 21. i≤||w||
| . . | . | . | 22. nth_tl(i-1;w) = nil
| . . | . | . | 23. mem_f(S.car;s;TBL)
| . . | . | . | ⊢
| . . | . | . |   by ((Assert (||nth_tl(i-1;w)|| = ||nil||))
| . . | . | . |       THEN
```

```
| . . | . | . |       (Reduce -1))
| . . | . | . |      THENA
| . . | . | . |      Analyze
| . . | . | . |
| . . | . | . .\
| . . | . | . . 21. i≤||w||
| . . | . | . . 22. ¬nth_tl(i-1;w) = nil
| . . | . | . . 23. mem_f(S.car;S.act
| . . | . | . . |               (hd(nth_tl(i-1;w))
| . . | . | . . |               ,(S:tl(nth_tl(i-1;w))←s));TBL)
| . . | . | . . ⊢
| . . | . | . . | by (RWH
| . . | . | . . |     (LemmaC
| . . | . | . . |      Thm* ∀y:ℕ, as:T*.
| . . | . | . . |           tl(nth_tl(y;as)) = nth_tl(y+1;as)
| . . | . | . . |      {nth_tl_lem0})
| . . | . | . . |     -1)
| . . | . | . . |    THEN
| . . | . | . . |    (ArithSimp -1)
| . . | . | . . |
| . . | . | . . 23. mem_f(S.car;S.act
| . . | . | . .                (hd(nth_tl(-1+i;w))
| . . | . | . .                ,(S:nth_tl(i;w)←s));TBL)
| . . | . | . . ⊢   by (FwdThru 15 [-1]) THEN (Analyze -1)
| . . | . |
| . . | . .\
| . . | . . 17. w : Alph*
| . . | . . 18. mem_f(S.car;(S:w←s);sL)
| . . | . . 19. ∀i:ℕ. i≤||w|| ⇒ mem_f(S.car;(S:nth_tl(i;w)←s);TBL)
| . . | . . ⊢   by (InstHyp [||w||] -1) THEN (Reduce -1)
| . . | . . |
| . . | . . 20. mem_f(S.car;(S:nth_tl(||w||;w)←s);TBL)
| . . | . . ⊢
| . . | . .   by Assert (nth_tl(||w||;w) = nil)
| . . | . .      THENL
| . . | . .      [BackThru
| . . | . .       Thm* ∀L:T*. ||L||≤0 ⇒ L = nil    {zero_nil}
| . . | . .      ;(RWH (HypC -1) -2) THEN (RecUnfold `maction` -2)
| . . | . .       THEN
| . . | . .       (Reduce -2)]
| . . |
| . . .\
| . . . 9. ||TBL|| = n-1
| . . . 10. ∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j]
| . . . 11. ∀s:S.car.
| . . . |   mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . . . 12. S.car*
| . . . 13. u : S.car
| . . . 14. v : S.car*
| . . . 15. (∀s:S.car.
| . . . |    mem_f(S.car;s;v) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . . |   ⇒
| . . . |
| . . . |   (∀s:S.car.
| . . . |    mem_f(S.car;s;sL) ⇒
```

```
| . . . |    mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;v))
| . . . |   ⇒
| . . . |
| . . . |   (∀s:S.car, a:Alph.
| . . . |    mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . |    mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;v))
| . . . |   ⇒
| . . . |
| . . . |   (∃TBL:S.car*.
| . . . |    (∀s:S.car.
| . . . |     mem_f(S.car;s;TBL)
| . . . |      ⟺
| . . . |     (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . . |    ∨ ||TBL|| = n
| . . . |      & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
| . . . |      & (∀s:S.car.
| . . . |         mem_f(S.car;s;TBL) ⇒
| . . . |         (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . . |      & (∃AL:S.car*.
| . . . |         (∀s:S.car.
| . . . |          mem_f(S.car;s;AL) ⇒
| . . . |          (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . . |         & (∀s:S.car.
| . . . |            mem_f(S.car;s;sL) ⇒
| . . . |            mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
| . . . |         & (∀s:S.car, a:Alph.
| . . . |            mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . |            mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))))
| . . . 16. ∀s:S.car.
| . . . |   u = s ∨ mem_f(S.car;s;v) ⇒
| . . . |   (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . . . 17. ∀s:S.car.
| . . . |   mem_f(S.car;s;sL) ⇒
| . . . |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . 18. ∀s:S.car, a:Alph.
| . . . |   mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . ⊢
| . . . | by (Inst
| . . . |     Thm* ∀s:S, l:S*. Fin(S) ⇒ Dec(mem_f(S;s;l))
| . . . |     {mem_f_dec}
| . . . |     [S.car;u;TBL])
| . . . |    THEN
| . . . |    (Analyze -1)
| . . . |
| . . . |\
| . . . | 19. mem_f(S.car;u;TBL)
| . . . | ⊢   by Analyze 15
| . . . | |
| . . . | |\
| . . . | | 15. ∀s:S.car.
| . . . | |     u = s ∨ mem_f(S.car;s;v) ⇒
| . . . | |     (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . . . | | 16. ∀s:S.car.
| . . . | |     mem_f(S.car;s;sL) ⇒
```

```
| . . . | |     mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . | | 17. ∀s:S.car, a:Alph.
| . . . | |     mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . | |     mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . | | 18. mem_f(S.car;u;TBL)
| . . . | | 19. s : S.car
| . . . | | 20. mem_f(S.car;s;v)
| . . . | | ⊢  ∃w:Alph*. mem_f(S.car;(S:w←s);sL)
| . . . | |   by (BackThru 15) THEN (Sel 2 (Analyze 0))
| . . . | |
| . . . | .\
| . . . | . 15. ∀s:S.car.
| . . . | . |   u = s ∨ mem_f(S.car;s;v) ⇒
| . . . | . |   (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . . . | . 16. ∀s:S.car.
| . . . | . |   mem_f(S.car;s;sL) ⇒
| . . . | . |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . | . 17. ∀s:S.car, a:Alph.
| . . . | . |   mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . | . |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . | . 18. mem_f(S.car;u;TBL)
| . . . | . 19. (∀s:S.car.
| . . . | . |    mem_f(S.car;s;sL) ⇒
| . . . | . |    mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;v))
| . . . | . |   ⇒
| . . . | . |
| . . . | . |   (∀s:S.car, a:Alph.
| . . . | . |    mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . | . |    mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;v))
| . . . | . |   ⇒
| . . . | . |
| . . . | . |   (∃TBL:S.car*.
| . . . | . |    (∀s:S.car.
| . . . | . |     mem_f(S.car;s;TBL)
| . . . | . |     ⟺
| . . . | . |     (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . . | . |    ∨ ||TBL|| = n
| . . . | . |      & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
| . . . | . |      & (∀s:S.car.
| . . . | . |         mem_f(S.car;s;TBL) ⇒
| . . . | . |         (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . . | . |      & (∃AL:S.car*.
| . . . | . |         (∀s:S.car.
| . . . | . |          mem_f(S.car;s;AL) ⇒
| . . . | . |          (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . . | . |         & (∀s:S.car.
| . . . | . |            mem_f(S.car;s;sL) ⇒
| . . . | . |            mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
| . . . | . |         & (∀s:S.car, a:Alph.
| . . . | . |            mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . | . |            mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))))
| . . . | . ⊢   by (Analyze -1) THEN (Analyze -1) THEN (NthHyp -1)
| . . . | . |
| . . . | . |\
| . . . | . | 19. s : S.car
```

```
| . . . | . | 20. mem_f(S.car;s;sL)
| . . . | . | ⊢  mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;v)
| . . . | . | | by (FwdThru 16 [-1]) THEN GenExRepD
| . . . | . | |    THEN
| . . . | . | |    (Try ProveProp)
| . . . | . | |
| . . . | . | 21. u = s
| . . . | . | ⊢   by (Sel 1 (Analyze 0)) THEN (RWH (HypC -1) 18)
| . . . | . |
| . . . | . .\
| . . . | . . 19. s : S.car
| . . . | . . 20. a : Alph
| . . . | . . 21. mem_f(S.car;S.act(a,s);TBL)
| . . . | . . ⊢  mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;v)
| . . . | . . | by (FwdThru 17 [-1]) THEN GenExRepD
| . . . | . . |    THEN
| . . . | . . |    (Try ProveProp)
| . . . | . . |
| . . . | . . 22. u = s
| . . . | . . ⊢   by (Sel 1 (Analyze 0)) THEN (RWH (HypC -1) 18)
| . . . |
| . . . .\
| . . . . 19. ¬mem_f(S.car;u;TBL)
| . . . . ⊢
| . . . . | by (Thin 15 THEN Thin 12 THEN InstConcl [u.TBL])
| . . . . |    THEN
| . . . . |    (Sel 2 (Analyze 0))
| . . . . |    THEN
| . . . . |    (Reduce 0)
| . . . . |
| . . . . |\
| . . . . | 12. u : S.car
| . . . . | 13. v : S.car*
| . . . . | 14. ∀s:S.car.
| . . . . | |   u = s ∨ mem_f(S.car;s;v) ⇒
| . . . . | |   (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . . . . | 15. ∀s:S.car.
| . . . . | |   mem_f(S.car;s;sL) ⇒
| . . . . | |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . . | 16. ∀s:S.car, a:Alph.
| . . . . | |   mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . . | |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . . | 17. ¬mem_f(S.car;u;TBL)
| . . . . | 18. i : ℕ(||TBL||+1)
| . . . . | 19. j : ℕi
| . . . . | ⊢  ¬(u.TBL)[i] = (u.TBL)[j]  by Decide (j = 0)
| . . . . | |
| . . . . | |\
| . . . . | | 20. j = 0
| . . . . | | ⊢
| . . . . | | | by (RWH (HypC -1) 0) THEN (Reduce 0)
| . . . . | | |    THEN
| . . . . | | |    (Unfold `select` 0)
| . . . . | | |    THEN
| . . . . | | |    (RecCaseSplit `nth_tl`)
```

```
| . . . . | | |
| . . . . | | 21. 0<i
| . . . . | | ⊢  ¬hd(nth_tl(i-1;TBL)) = u
| . . . . | | | by (Fold `select` 0 THEN Analyze 0) THEN (Analyze 17)
| . . . . | | |
| . . . . | | 17. i : ℕ(||TBL||+1)
| . . . . | | 18. j : ℕi
| . . . . | | 19. j = 0
| . . . . | | 20. 0<i
| . . . . | | 21. TBL[(i-1)] = u
| . . . . | | ⊢  mem_f(S.car;u;TBL)
| . . . . | |   by (RWH (RevHypC -1) 0)
| . . . . | |      THEN
| . . . . | |      (BackThru
| . . . . | |       Thm* ∀L:T*, i:ℕ||L||. mem_f(T;L[i];L)
| . . . . | |       {select_mem})
| . . . . | |
| . . . . | .\
| . . . . | . 20. ¬j = 0
| . . . . | . ⊢   by (Unfold `select` 0) THEN (RecCaseSplit `nth_tl`)
| . . . . | . |
| . . . . | . 21. 0<i
| . . . . | . ⊢  ¬hd(nth_tl(i-1;TBL)) = hd(nth_tl(j;u.TBL))
| . . . . | . | by RWH (AddrC [2] (FoldC `select`)) 0
| . . . . | . |    THEN
| . . . . | . |    RecCaseSplit `nth_tl`
| . . . . | . |
| . . . . | . 22. 0<j
| . . . . | . ⊢  ¬TBL[(i-1)] = hd(nth_tl(j-1;TBL))
| . . . . | .   by Fold `select` 0 THEN BackThru 10
| . . . . |
| . . . . |\
| . . . . | 12. u : S.car
| . . . . | 13. v : S.car*
| . . . . | 14. ∀s:S.car.
| . . . . | |   u = s ∨ mem_f(S.car;s;v) ⇒
| . . . . | |   (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . . . . | 15. ∀s:S.car.
| . . . . | |   mem_f(S.car;s;sL) ⇒
| . . . . | |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . . | 16. ∀s:S.car, a:Alph.
| . . . . | |   mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . . | |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . . | 17. ¬mem_f(S.car;u;TBL)
| . . . . | 18. s : S.car
| . . . . | 19. u = s ∨ mem_f(S.car;s;TBL)
| . . . . | ⊢  ∃w:Alph*. mem_f(S.car;(S:w←s);sL)
| . . . . | | by Analyze -1 THENL [BackThru 14;BackThru 11]
| . . . . | |
| . . . . | 19. u = s
| . . . . | ⊢  u = s ∨ mem_f(S.car;s;v)
| . . . . |   by (Sel 1 (Analyze 0)) THEN (NthHyp -1)
| . . . . |
| . . . . .\
| . . . . . 12. u : S.car
```

```
| . . . . . 13. v : S.car*
| . . . . . 14. ∀s:S.car.
| . . . . . |   u = s ∨ mem_f(S.car;s;v) ⇒
| . . . . . |   (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
| . . . . . 15. ∀s:S.car.
| . . . . . |   mem_f(S.car;s;sL) ⇒
| . . . . . |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . . . 16. ∀s:S.car, a:Alph.
| . . . . . |   mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . . . |   mem_f(S.car;s;TBL) ∨ u = s ∨ mem_f(S.car;s;v)
| . . . . . 17. ¬mem_f(S.car;u;TBL)
| . . . . . ⊢  ∃AL:S.car*.
| . . . . . |  (∀s:S.car.
| . . . . . |   mem_f(S.car;s;AL) ⇒
| . . . . . |   (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
| . . . . . |  & (∀s:S.car.
| . . . . . |     mem_f(S.car;s;sL) ⇒
| . . . . . |     u = s ∨ mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
| . . . . . |  & (∀s:S.car, a:Alph.
| . . . . . |     u = S.act(a,s) ∨ mem_f(S.car;S.act(a,s);TBL) ⇒
| . . . . . |     u = s ∨ mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
| . . . . . | by (Inst
| . . . . . |     Thm* ∀S:ActionSet(Alph), s:S.car.
| . . . . . |          Fin(Alph) ⇒
| . . . . . |          Fin(S.car) ⇒
| . . . . . |          (∃BL:S.car*.
| . . . . . |           ∀t:S.car.
| . . . . . |           mem_f(S.car;t;BL)
| . . . . . |           ⟺
| . . . . . |           (∃a:Alph. S.act(a,t) = s))
| . . . . . |     {back_listify}
| . . . . . |     [Alph;S;u])
| . . . . . |    THEN
| . . . . . |    (Analyze -1)
| . . . . . |
| . . . . . 18. BL : S.car*
| . . . . . 19. ∀t:S.car.
| . . . . . |   mem_f(S.car;t;BL) ⟺ (∃a:Alph. S.act(a,t) = u)
| . . . . . ⊢
| . . . . . | by (InstConcl [BL @ v])
| . . . . . |    THEN
| . . . . . |    (RWH
| . . . . . |     (LemmaC
| . . . . . |      Thm* ∀L1,L2:T*, t:T.
| . . . . . |           mem_f(T;t;L1 @ L2)
| . . . . . |           ⟺
| . . . . . |           mem_f(T;t;L1) ∨ mem_f(T;t;L2)
| . . . . . |      {mem_append})
| . . . . . |     0)
| . . . . . |    THEN
| . . . . . |    GenExRepD
| . . . . . |
| . . . . . |\
| . . . . . | 20. s : S.car
| . . . . . | 21. mem_f(S.car;s;BL)
```

```
| . . . . . | ⊢  ∃w:Alph*. mem_f(S.car;(S:w←s);sL)
| . . . . . | | by (FwdThru 19 [21]) THEN (Analyze -1)
| . . . . . | |
| . . . . . | 22. a : Alph
| . . . . . | 23. S.act(a,s) = u
| . . . . . | ⊢   by (InstHyp [S.act(a,s)] 14) THEN (Analyze -1)
| . . . . . | |
| . . . . . | |\
| . . . . . | | .....antecedent .....
| . . . . . | | ⊢  u = S.act(a,s) ∨ mem_f(S.car;S.act(a,s);v)
| . . . . . | |   by Sel 1 (Analyze 0)
| . . . . . | |
| . . . . . | .\
| . . . . . | . 24. w : Alph*
| . . . . . | . 25. mem_f(S.car;(S:w←S.act(a,s));sL)
| . . . . . | . ⊢
| . . . . . | . | by (InstConcl [w @ [a]])
| . . . . . | . |    THEN
| . . . . . | . |    (RWH
| . . . . . | . |     (LemmaC
| . . . . . | . |      Thm* ∀S:ActionSet(T), s:S.car, tl1,tl2:T*.
| . . . . . | . |           (S:tl1 @ tl2←s) = (S:tl1←(S:tl2←s))
| . . . . . | . |      {action_append})
| . . . . . | . |     0)
| . . . . . | . |
| . . . . . | . ⊢  mem_f(S.car;(S:w←(S:[a]←s));sL)
| . . . . . | .   by RepeatFor 2
| . . . . . | .      (RWH (AddrC [2;3] (RecUnfoldC `maction`)) 0
| . . . . . | .       THEN
| . . . . . | .       Reduce 0)
| . . . . . | .      THEN
| . . . . . | .      NthHyp -1
| . . . . . |
| . . . . . |\
| . . . . . | 20. s : S.car
| . . . . . | 21. mem_f(S.car;s;v)
| . . . . . | ⊢  ∃w:Alph*. mem_f(S.car;(S:w←s);sL)
| . . . . . |   by (BackThru 14) THEN (Sel 2 (Analyze 0))
| . . . . . |
| . . . . . |\
| . . . . . | 20. s : S.car
| . . . . . | 21. mem_f(S.car;s;sL)
| . . . . . | ⊢  u = s ∨ mem_f(S.car;s;TBL)
| . . . . . |    ∨ mem_f(S.car;s;BL)
| . . . . . |    ∨ mem_f(S.car;s;v)
| . . . . . |   by (FwdThru 15 [-1]) THEN ProveProp
| . . . . . |
| . . . . . |\
| . . . . . | 20. s : S.car
| . . . . . | 21. a : Alph
| . . . . . | 22. u = S.act(a,s)
| . . . . . | ⊢  u = s ∨ mem_f(S.car;s;TBL)
| . . . . . | |  ∨ mem_f(S.car;s;BL)
| . . . . . | |  ∨ mem_f(S.car;s;v)
| . . . . . | | by (InstHyp [s] 19) THEN (Analyze -1)
```

```
| . . . . . | |    THEN
| . . . . . | |    (Analyze -1)
| . . . . . | |
| . . . . . | |\
| . . . . . | | .....antecedent .....
| . . . . . | | 23. mem_f(S.car;s;BL) ⇒ (∃a:Alph. S.act(a,s) = u)
| . . . . . | | ⊢  ∃a:Alph. S.act(a,s) = u  by InstConcl [a]
| . . . . . | |
| . . . . . | .\
| . . . . . | . 23. mem_f(S.car;s;BL) ⇒ (∃a:Alph. S.act(a,s) = u)
| . . . . . | . 24. mem_f(S.car;s;BL)
| . . . . . | . ⊢   by Sel 2 (Analyze 0)
| . . . . . |
| . . . . . .\
| . . . . . . 20. s : S.car
| . . . . . . 21. a : Alph
| . . . . . . 22. mem_f(S.car;S.act(a,s);TBL)
| . . . . . . ⊢  u = s ∨ mem_f(S.car;s;TBL)
| . . . . . .    ∨ mem_f(S.car;s;BL)
| . . . . . .    ∨ mem_f(S.car;s;v)
| . . . . . .   by (FwdThru 16 [-1]) THEN ProveProp
|
.\
. 6. ∀n:ℕ.
. |  ∃TBL:S.car*.
. |  (∀s:S.car.
. |   mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
. |  ∨ ||TBL|| = n
. |    & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
. |    & (∀s:S.car.
. |       mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
. |    & (∃AL:S.car*.
. |       (∀s:S.car.
. |        mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
. |       & (∀s:S.car.
. |          mem_f(S.car;s;sL) ⇒
. |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
. |       & (∀s:S.car, a:Alph.
. |          mem_f(S.car;S.act(a,s);TBL) ⇒
. |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL)))
. ⊢
. | by (Analyze -2
. |     THEN
. |     RWH
. |     (LemmaC
. |      Thm* (∃f:(A→B). Bij(A; B; f)) ⟺ (A ∼ B)
. |      {bij_iff_1_1_corr})
. |     -2)
. |    THEN
. |    (Analyze -2)
. |    THEN
. |    (Analyze -2)
. |
. 5. n : ℕ
. 6. f : ℕn→S.car
```

```
. 7. g : S.car→ℕn
. 8. InvFuns(ℕn; S.car; f; g)
. 9. ∀n:ℕ.
. |  ∃TBL:S.car*.
. |  (∀s:S.car.
. |   mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
. |  ∨ ||TBL|| = n
. |    & (∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j])
. |    & (∀s:S.car.
. |       mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
. |    & (∃AL:S.car*.
. |       (∀s:S.car.
. |        mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL)))
. |       & (∀s:S.car.
. |          mem_f(S.car;s;sL) ⇒
. |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL))
. |       & (∀s:S.car, a:Alph.
. |          mem_f(S.car;S.act(a,s);TBL) ⇒
. |          mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL)))
. ⊢   by Decide (n = 0)
. |
. |\
. | 10. n = 0
. | ⊢   by (InstConcl [nil]) THEN (Analyze 0)
. | |
. | 11. s : S.car
. | ⊢  mem_f(S.car;s;nil) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
. |   by Assert (g(s) ∈ ℕ0)
. |
. .\
. . 10. ¬n = 0
. . ⊢
. . | by (InstHyp [n+1] -2) THEN (Analyze -1) THEN (Analyze -1)
. . |    THEN
. . |    ExRepD
. . |
. . |\
. . | 11. TBL : S.car*
. . | 12. ∀s:S.car.
. . |     mem_f(S.car;s;TBL) ⟺ (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
. . | ⊢   by (InstConcl [TBL]) THEN (NthHyp -1)
. . |

. . .\
. . . 11. TBL : S.car*
. . . 12. ||TBL|| = n+1
. . . 13. ∀i:ℕ||TBL||, j:ℕi. ¬TBL[i] = TBL[j]
. . . 14. ∀s:S.car.
. . . |   mem_f(S.car;s;TBL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
. . . 15. AL : S.car*
. . . 16. ∀s:S.car.
. . . |   mem_f(S.car;s;AL) ⇒ (∃w:Alph*. mem_f(S.car;(S:w←s);sL))
. . . 17. ∀s:S.car.
. . . |   mem_f(S.car;s;sL) ⇒ mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL)
. . . 18. ∀s:S.car, a:Alph.
. . . |   mem_f(S.car;S.act(a,s);TBL) ⇒
```

```
. . . |   mem_f(S.car;s;TBL) ∨ mem_f(S.car;s;AL)
. . . ⊢
. . . | by (Inst
. . . |     Thm* ∀n:{1...}, f:(ℕ(n+1)→ℕn).
. . . |          ∃i:ℕ(n+1), j:ℕi. f(i) = f(j)
. . . |     {phole_aux}
. . . |     [n;λx.g(TBL[x])])
. . . |    THEN
. . . |    ExRepD
. . . |    THEN
. . . |    (Reduce -1)
. . . |
. . . 19. i : ℕ(n+1)
. . . 20. j : ℕi
. . . 21. g(TBL[i]) = g(TBL[j])
. . . ⊢   by (InstHyp [i;j] 13) THEN (Analyze -1)
. . . |
. . . ⊢  TBL[i] = TBL[j]
. . . | by Assert (f(g(TBL[i])) = f(g(TBL[j]))) THENA Analyze
. . . |
. . . 22. f(g(TBL[i])) = f(g(TBL[j]))
. . . ⊢
. . .   by (RWH add_composeC -1 THEN Analyze 8 THEN RWH (HypC 9) -1)
. . .      THEN
. . .      (Reduce -1)
END
```