```
P_no_dup_composable  THM

⊢  ∀E:EventStruct.
|  (ternary) composableR(E) preserves No-dup-deliver(E)   by Auto
|
1. E : EventStruct
⊢  (ternary) composableR(E) preserves No-dup-deliver(E)
| by ...
|
1. E : EventStruct
2. x : |E| List
3. y : |E| List
4. z : |E| List
5. ∀i,j:ℕ||x||.
|  ¬is-send(E)(x[i]) ⇒
|  ¬is-send(E)(x[j]) ⇒
|  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
6. ∀i,j:ℕ||y||.
|  ¬is-send(E)(y[i]) ⇒
|  ¬is-send(E)(y[j]) ⇒
|  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
8. z = (x @ y)
9. i : ℕ||z||
10. j : ℕ||z||
11. ¬is-send(E)(z[i])
12. ¬is-send(E)(z[j])
13. z[j] =msg=(E) z[i]
14. loc(E)(z[i]) = loc(E)(z[j])
⊢  i = j  by Decide i<j THENA Auto
|
|\
| 1. E : EventStruct
| 2. x : |E| List
| 3. y : |E| List
| 4. z : |E| List
| 5. ∀i,j:ℕ||x||.
| |  ¬is-send(E)(x[i]) ⇒
| |  ¬is-send(E)(x[j]) ⇒
| |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| 6. ∀i,j:ℕ||y||.
| |  ¬is-send(E)(y[i]) ⇒
| |  ¬is-send(E)(y[j]) ⇒
| |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| 8. z = (x @ y)
| 9. i : ℕ||z||
| 10. j : ℕ||z||
| 11. ¬is-send(E)(z[i])
| 12. ¬is-send(E)(z[j])
| 13. z[j] =msg=(E) z[i]
| 14. loc(E)(z[i]) = loc(E)(z[j])
| 15. i<j
| ⊢  i = j  by Decide j<||x|| THENA Auto'
| |
```

```
| |\
| | 1. E : EventStruct
| | 2. x : |E| List
| | 3. y : |E| List
| | 4. z : |E| List
| | 5. ∀i,j:ℕ||x||.
| | |  ¬is-send(E)(x[i]) ⇒
| | |  ¬is-send(E)(x[j]) ⇒
| | |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| | 6. ∀i,j:ℕ||y||.
| | |  ¬is-send(E)(y[i]) ⇒
| | |  ¬is-send(E)(y[j]) ⇒
| | |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| | 8. z = (x @ y)
| | 9. i : ℕ||z||
| | 10. j : ℕ||z||
| | 11. ¬is-send(E)(z[i])
| | 12. ¬is-send(E)(z[j])
| | 13. z[j] =msg=(E) z[i]
| | 14. loc(E)(z[i]) = loc(E)(z[j])
| | 15. i<j
| | 16. j<||x||
| | ⊢  i = j  by Assert z[i] = x[i]
| | |
| | |\
| | | .....assertion .....
| | | 1. E : EventStruct
| | | 2. x : |E| List
| | | 3. y : |E| List
| | | 4. z : |E| List
| | | 5. ∀i,j:ℕ||x||.
| | |    ¬is-send(E)(x[i]) ⇒
| | |    ¬is-send(E)(x[j]) ⇒
| | |    (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| | | 6. ∀i,j:ℕ||y||.
| | |    ¬is-send(E)(y[i]) ⇒
| | |    ¬is-send(E)(y[j]) ⇒
| | |    (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| | | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| | | 8. z = (x @ y)
| | | 9. i : ℕ||z||
| | | 10. j : ℕ||z||
| | | 11. ¬is-send(E)(z[i])
| | | 12. ¬is-send(E)(z[j])
| | | 13. z[j] =msg=(E) z[i]
| | | 14. loc(E)(z[i]) = loc(E)(z[j])
| | | 15. i<j
| | | 16. j<||x||
| | | ⊢  z[i] = x[i]
| | |   by HypSubstList 8 0 THENA Auto' THEN
| | |
| | |      BackThruLemma `select_append_front` THENA Auto
| | |
| | |
```

```
| | .\
| | . 1. E : EventStruct
| | . 2. x : |E| List
| | . 3. y : |E| List
| | . 4. z : |E| List
| | . 5. ∀i,j:ℕ||x||.
| | . |  ¬is-send(E)(x[i]) ⇒
| | . |  ¬is-send(E)(x[j]) ⇒
| | . |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| | . 6. ∀i,j:ℕ||y||.
| | . |  ¬is-send(E)(y[i]) ⇒
| | . |  ¬is-send(E)(y[j]) ⇒
| | . |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| | . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| | . 8. z = (x @ y)
| | . 9. i : ℕ||z||
| | . 10. j : ℕ||z||
| | . 11. ¬is-send(E)(z[i])
| | . 12. ¬is-send(E)(z[j])
| | . 13. z[j] =msg=(E) z[i]
| | . 14. loc(E)(z[i]) = loc(E)(z[j])
| | . 15. i<j
| | . 16. j<||x||
| | . 17. z[i] = x[i]
| | . ⊢  i = j  by Assert z[j] = x[j]
| | . |
| | . |\
| | . | .....assertion .....
| | . | 1. E : EventStruct
| | . | 2. x : |E| List
| | . | 3. y : |E| List
| | . | 4. z : |E| List
| | . | 5. ∀i,j:ℕ||x||.
| | . |    ¬is-send(E)(x[i]) ⇒
| | . |    ¬is-send(E)(x[j]) ⇒
| | . |    (x[j] =msg=(E) x[i]) ⇒
| | . |    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| | . | 6. ∀i,j:ℕ||y||.
| | . |    ¬is-send(E)(y[i]) ⇒
| | . |    ¬is-send(E)(y[j]) ⇒
| | . |    (y[j] =msg=(E) y[i]) ⇒
| | . |    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| | . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| | . | 8. z = (x @ y)
| | . | 9. i : ℕ||z||
| | . | 10. j : ℕ||z||
| | . | 11. ¬is-send(E)(z[i])
| | . | 12. ¬is-send(E)(z[j])
| | . | 13. z[j] =msg=(E) z[i]
| | . | 14. loc(E)(z[i]) = loc(E)(z[j])
| | . | 15. i<j
| | . | 16. j<||x||
| | . | 17. z[i] = x[i]
| | . | ⊢  z[j] = x[j]
| | . |   by HypSubstList 8 0 THENA Auto' THEN
```

```
| | . |
| | . |      BackThruLemma ‘select_append_front‘ THENA Auto
| | . |
| | . |
| | . .\
| | . . 1. E : EventStruct
| | . . 2. x : |E| List
| | . . 3. y : |E| List
| | . . 4. z : |E| List
| | . . 5. ∀i,j:ℕ||x||.
| | . .    ¬is-send(E)(x[i]) ⇒
| | . .    ¬is-send(E)(x[j]) ⇒
| | . .    (x[j] =msg=(E) x[i]) ⇒
| | . .    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| | . . 6. ∀i,j:ℕ||y||.
| | . .    ¬is-send(E)(y[i]) ⇒
| | . .    ¬is-send(E)(y[j]) ⇒
| | . .    (y[j] =msg=(E) y[i]) ⇒
| | . .    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| | . . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| | . . 8. z = (x @ y)
| | . . 9. i : ℕ||z||
| | . . 10. j : ℕ||z||
| | . . 11. ¬is-send(E)(z[i])
| | . . 12. ¬is-send(E)(z[j])
| | . . 13. z[j] =msg=(E) z[i]
| | . . 14. loc(E)(z[i]) = loc(E)(z[j])
| | . . 15. i<j
| | . . 16. j<||x||
| | . . 17. z[i] = x[i]
| | . . 18. z[j] = x[j]
| | . . ⊢  i = j
| | . .   by InstHyp [i;j] 5 THEN Auto
| | . . THEN SubstFor x[i]
| | . .       0 THEN Try Trivial
| | . . THEN SubstFor x[j]
| | . .       0 THEN Try Trivial

| |
| .\
| . 1. E : EventStruct
| . 2. x : |E| List
| . 3. y : |E| List
| . 4. z : |E| List
| . 5. ∀i,j:ℕ||x||.
| . |  ¬is-send(E)(x[i]) ⇒
| . |  ¬is-send(E)(x[j]) ⇒
| . |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . 6. ∀i,j:ℕ||y||.
| . |  ¬is-send(E)(y[i]) ⇒
| . |  ¬is-send(E)(y[j]) ⇒
| . |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . 8. z = (x @ y)
| . 9. i : ℕ||z||
| . 10. j : ℕ||z||
```

```
| . 11. ¬is-send(E)(z[i])
| . 12. ¬is-send(E)(z[j])
| . 13. z[j] =msg=(E) z[i]
| . 14. loc(E)(z[i]) = loc(E)(z[j])
| . 15. i<j
| . 16. ¬j<||x||
| . ⊢  i = j
| . | by InstLemma `length_append` [|E|;x;y] THENA Auto
| .
| . |    THEN SubstFor x @ y (-1) THENA Auto
| . THEN Assert z[j]
| . |                                                     =
| . |                                                     y[(j-||x||)]
| . |
| . |
| . |\
| . | .....assertion .....
| . | 1. E : EventStruct
| . | 2. x : |E| List
| . | 3. y : |E| List
| . | 4. z : |E| List
| . | 5. ∀i,j:ℕ||x||.
| . | |  ¬is-send(E)(x[i]) ⇒
| . | |  ¬is-send(E)(x[j]) ⇒
| . | |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . | 6. ∀i,j:ℕ||y||.
| . | |  ¬is-send(E)(y[i]) ⇒
| . | |  ¬is-send(E)(y[j]) ⇒
| . | |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . | 8. z = (x @ y)
| . | 9. i : ℕ||z||
| . | 10. j : ℕ||z||
| . | 11. ¬is-send(E)(z[i])
| . | 12. ¬is-send(E)(z[j])
| . | 13. z[j] =msg=(E) z[i]
| . | 14. loc(E)(z[i]) = loc(E)(z[j])
| . | 15. i<j
| . | 16. ¬j<||x||
| . | 17. ||z|| = ||x||+||y||
| . | ⊢  z[j] = y[(j-||x||)]  by HypSubstList 8 0 THENA Auto'
| . | |
| . | 1. E : EventStruct
| . | 2. x : |E| List
| . | 3. y : |E| List
| . | 4. z : |E| List
| . | 5. ∀i,j:ℕ||x||.
| . |    ¬is-send(E)(x[i]) ⇒
| . |    ¬is-send(E)(x[j]) ⇒
| . |    (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . | 6. ∀i,j:ℕ||y||.
| . |    ¬is-send(E)(y[i]) ⇒
| . |    ¬is-send(E)(y[j]) ⇒
| . |    (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
```

```
| . | 8. z = (x @ y)
| . | 9. i : ℕ||z||
| . | 10. j : ℕ||z||
| . | 11. ¬is-send(E)(z[i])
| . | 12. ¬is-send(E)(z[j])
| . | 13. z[j] =msg=(E) z[i]
| . | 14. loc(E)(z[i]) = loc(E)(z[j])
| . | 15. i<j
| . | 16. ¬j<||x||
| . | 17. ||z|| = ||x||+||y||
| . | ⊢  (x @ y)[j] = y[(j-||x||)]
| . |   by BackThruLemma ‘select_append_back‘ THEN Auto’
| . |
| . .\
| . . 1. E : EventStruct
| . . 2. x : |E| List
| . . 3. y : |E| List
| . . 4. z : |E| List
| . . 5. ∀i,j:ℕ||x||.
| . . |  ¬is-send(E)(x[i]) ⇒
| . . |  ¬is-send(E)(x[j]) ⇒
| . . |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . 6. ∀i,j:ℕ||y||.
| . . |  ¬is-send(E)(y[i]) ⇒
| . . |  ¬is-send(E)(y[j]) ⇒
| . . |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . . 8. z = (x @ y)
| . . 9. i : ℕ||z||
| . . 10. j : ℕ||z||
| . . 11. ¬is-send(E)(z[i])
| . . 12. ¬is-send(E)(z[j])
| . . 13. z[j] =msg=(E) z[i]
| . . 14. loc(E)(z[i]) = loc(E)(z[j])
| . . 15. i<j
| . . 16. ¬j<||x||
| . . 17. ||z|| = ||x||+||y||
| . . 18. z[j] = y[(j-||x||)]
| . . ⊢  i = j  by Decide i<||x|| THENA Auto
| . . |
| . . |\
| . . | 1. E : EventStruct
| . . | 2. x : |E| List
| . . | 3. y : |E| List
| . . | 4. z : |E| List
| . . | 5. ∀i,j:ℕ||x||.
| . . | |  ¬is-send(E)(x[i]) ⇒
| . . | |  ¬is-send(E)(x[j]) ⇒
| . . | |  (x[j] =msg=(E) x[i]) ⇒
| . . | |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . | 6. ∀i,j:ℕ||y||.
| . . | |  ¬is-send(E)(y[i]) ⇒
| . . | |  ¬is-send(E)(y[j]) ⇒
| . . | |  (y[j] =msg=(E) y[i]) ⇒
| . . | |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
```

```
| . . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . . | 8. z = (x @ y)
| . . | 9. i : ℕ||z||
| . . | 10. j : ℕ||z||
| . . | 11. ¬is-send(E)(z[i])
| . . | 12. ¬is-send(E)(z[j])
| . . | 13. z[j] =msg=(E) z[i]
| . . | 14. loc(E)(z[i]) = loc(E)(z[j])
| . . | 15. i<j
| . . | 16. ¬j<||x||
| . . | 17. ||z|| = ||x||+||y||
| . . | 18. z[j] = y[(j-||x||)]
| . . | 19. i<||x||
| . . | ⊢  i = j  by Assert z[i] = x[i]
| . . | |
| . . | |\
| . . | | .....assertion .....
| . . | | 1. E : EventStruct
| . . | | 2. x : |E| List
| . . | | 3. y : |E| List
| . . | | 4. z : |E| List
| . . | | 5. ∀i,j:ℕ||x||.
| . . | |    ¬is-send(E)(x[i]) ⇒
| . . | |    ¬is-send(E)(x[j]) ⇒
| . . | |    (x[j] =msg=(E) x[i]) ⇒
| . . | |    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . | | 6. ∀i,j:ℕ||y||.
| . . | |    ¬is-send(E)(y[i]) ⇒
| . . | |    ¬is-send(E)(y[j]) ⇒
| . . | |    (y[j] =msg=(E) y[i]) ⇒
| . . | |    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . . | | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . . | | 8. z = (x @ y)
| . . | | 9. i : ℕ||z||
| . . | | 10. j : ℕ||z||
| . . | | 11. ¬is-send(E)(z[i])
| . . | | 12. ¬is-send(E)(z[j])
| . . | | 13. z[j] =msg=(E) z[i]
| . . | | 14. loc(E)(z[i]) = loc(E)(z[j])
| . . | | 15. i<j
| . . | | 16. ¬j<||x||
| . . | | 17. ||z|| = ||x||+||y||
| . . | | 18. z[j] = y[(j-||x||)]
| . . | | 19. i<||x||
| . . | | ⊢  z[i] = x[i]
| . . | |   by HypSubstList 8 0 THENA Auto' THEN
| . . | |
| . . | |      BackThruLemma `select_append_front` THENA Auto
| . . | |
| . . | |
| . . | .\
| . . | . 1. E : EventStruct
| . . | . 2. x : |E| List
| . . | . 3. y : |E| List
| . . | . 4. z : |E| List
```

```
| . . | . 5. ∀i,j:ℕ||x||.
| . . | . |  ¬is-send(E)(x[i]) ⇒
| . . | . |  ¬is-send(E)(x[j]) ⇒
| . . | . |  (x[j] =msg=(E) x[i]) ⇒
| . . | . |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . | . 6. ∀i,j:ℕ||y||.
| . . | . |  ¬is-send(E)(y[i]) ⇒
| . . | . |  ¬is-send(E)(y[j]) ⇒
| . . | . |  (y[j] =msg=(E) y[i]) ⇒
| . . | . |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . . | . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . . | . 8. z = (x @ y)
| . . | . 9. i : ℕ||z||
| . . | . 10. j : ℕ||z||
| . . | . 11. ¬is-send(E)(z[i])
| . . | . 12. ¬is-send(E)(z[j])
| . . | . 13. z[j] =msg=(E) z[i]
| . . | . 14. loc(E)(z[i]) = loc(E)(z[j])
| . . | . 15. i<j
| . . | . 16. ¬j<||x||
| . . | . 17. ||z|| = ||x||+||y||
| . . | . 18. z[j] = y[(j-||x||)]
| . . | . 19. i<||x||
| . . | . 20. z[i] = x[i]
| . . | . ⊢  i = j
| . . | . | by AllHyps (\h. Unfold `l_all` h THEN InstHyp [x[i];y[(j
| . . | . |    -||x||)]] h THEN Auto')
| . . | .
| . . | . |    THEN Try (BackThruLemma `select_member` THEN Auto')
| . . | . |
| . . | . 1. E : EventStruct
| . . | . 2. x : |E| List
| . . | . 3. y : |E| List
| . . | . 4. z : |E| List
| . . | . 5. ∀i,j:ℕ||x||.
| . . | . |  ¬is-send(E)(x[i]) ⇒
| . . | . |  ¬is-send(E)(x[j]) ⇒
| . . | . |  (x[j] =msg=(E) x[i]) ⇒
| . . | . |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . | . 6. ∀i,j:ℕ||y||.
| . . | . |  ¬is-send(E)(y[i]) ⇒
| . . | . |  ¬is-send(E)(y[j]) ⇒
| . . | . |  (y[j] =msg=(E) y[i]) ⇒
| . . | . |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . . | . 7. ∀x@0:|E|.
| . . | . |  (x@0 ∈ x) ⇒ (∀y@0:|E|. (y@0 ∈ y) ⇒ ¬(x@0 =msg=(E) y@0))
| . . | . 8. z = (x @ y)
| . . | . 9. i : ℕ||z||
| . . | . 10. j : ℕ||z||
| . . | . 11. ¬is-send(E)(z[i])
| . . | . 12. ¬is-send(E)(z[j])
| . . | . 13. z[j] =msg=(E) z[i]
| . . | . 14. loc(E)(z[i]) = loc(E)(z[j])
| . . | . 15. i<j
| . . | . 16. ¬j<||x||
```

```
| . . | . 17. ||z|| = ||x||+||y||
| . . | . 18. z[j] = y[(j-||x||)]
| . . | . 19. i<||x||
| . . | . 20. z[i] = x[i]
| . . | . 21. ¬(x[i] =msg=(E) y[(j-||x||)])
| . . | . ⊢  i = j
| . . | . | by D (-1) THEN SubstFor x[i] 0 THENA Auto
| . . | . THEN SubstFor
| . . | . |    y[(j-||x||)] 0 THEN Auto
| . . | . |
| . . | . 1. E : EventStruct
| . . | . 2. x : |E| List
| . . | . 3. y : |E| List
| . . | . 4. z : |E| List
| . . | . 5. ∀i,j:ℕ||x||.
| . . | .    ¬is-send(E)(x[i]) ⇒
| . . | .    ¬is-send(E)(x[j]) ⇒
| . . | .    (x[j] =msg=(E) x[i]) ⇒
| . . | .    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . | . 6. ∀i,j:ℕ||y||.
| . . | .    ¬is-send(E)(y[i]) ⇒
| . . | .    ¬is-send(E)(y[j]) ⇒
| . . | .    (y[j] =msg=(E) y[i]) ⇒
| . . | .    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . . | . 7. ∀x@0:|E|.
| . . | .    (x@0 ∈ x) ⇒ (∀y@0:|E|. (y@0 ∈ y) ⇒ ¬(x@0 =msg=(E) y@0))
| . . | . 8. z = (x @ y)
| . . | . 9. i : ℕ||z||
| . . | . 10. j : ℕ||z||
| . . | . 11. ¬is-send(E)(z[i])
| . . | . 12. ¬is-send(E)(z[j])
| . . | . 13. z[j] =msg=(E) z[i]
| . . | . 14. loc(E)(z[i]) = loc(E)(z[j])
| . . | . 15. i<j
| . . | . 16. ¬j<||x||
| . . | . 17. ||z|| = ||x||+||y||
| . . | . 18. z[j] = y[(j-||x||)]
| . . | . 19. i<||x||
| . . | . 20. z[i] = x[i]
| . . | . ⊢  z[i] =msg=(E) z[j]
| . . | .   by InstLemma ‘event_msg_eq_equiv‘ [E] THENA Auto
| . . | .
| . . | .      THEN UseEquiv
| . . |
| . . .\
| . . . 1. E : EventStruct
| . . . 2. x : |E| List
| . . . 3. y : |E| List
| . . . 4. z : |E| List
| . . . 5. ∀i,j:ℕ||x||.
| . . . |  ¬is-send(E)(x[i]) ⇒
| . . . |  ¬is-send(E)(x[j]) ⇒
| . . . |  (x[j] =msg=(E) x[i]) ⇒
| . . . |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . . 6. ∀i,j:ℕ||y||.
```

```
| . . . |  ¬is-send(E)(y[i]) ⇒
| . . . |  ¬is-send(E)(y[j]) ⇒
| . . . |  (y[j] =msg=(E) y[i]) ⇒
| . . . |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . . . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . . . 8. z = (x @ y)
| . . . 9. i : ℕ||z||
| . . . 10. j : ℕ||z||
| . . . 11. ¬is-send(E)(z[i])
| . . . 12. ¬is-send(E)(z[j])
| . . . 13. z[j] =msg=(E) z[i]
| . . . 14. loc(E)(z[i]) = loc(E)(z[j])
| . . . 15. i<j
| . . . 16. ¬j<||x||
| . . . 17. ||z|| = ||x||+||y||
| . . . 18. z[j] = y[(j-||x||)]
| . . . 19. ¬i<||x||
| . . . ⊢  i = j  by Assert z[i] = y[(i-||x||)]
| . . . |
| . . . |\
| . . . | .....assertion .....
| . . . | 1. E : EventStruct
| . . . | 2. x : |E| List
| . . . | 3. y : |E| List
| . . . | 4. z : |E| List
| . . . | 5. ∀i,j:ℕ||x||.
| . . . | |  ¬is-send(E)(x[i]) ⇒
| . . . | |  ¬is-send(E)(x[j]) ⇒
| . . . | |  (x[j] =msg=(E) x[i]) ⇒
| . . . | |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . . | 6. ∀i,j:ℕ||y||.
| . . . | |  ¬is-send(E)(y[i]) ⇒
| . . . | |  ¬is-send(E)(y[j]) ⇒
| . . . | |  (y[j] =msg=(E) y[i]) ⇒
| . . . | |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . . . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . . . | 8. z = (x @ y)
| . . . | 9. i : ℕ||z||
| . . . | 10. j : ℕ||z||
| . . . | 11. ¬is-send(E)(z[i])
| . . . | 12. ¬is-send(E)(z[j])
| . . . | 13. z[j] =msg=(E) z[i]
| . . . | 14. loc(E)(z[i]) = loc(E)(z[j])
| . . . | 15. i<j
| . . . | 16. ¬j<||x||
| . . . | 17. ||z|| = ||x||+||y||
| . . . | 18. z[j] = y[(j-||x||)]
| . . . | 19. ¬i<||x||
| . . . | ⊢  z[i] = y[(i-||x||)]  by HypSubstList 8 0 THENA Auto'
| . . . | |
| . . . | 1. E : EventStruct
| . . . | 2. x : |E| List
| . . . | 3. y : |E| List
| . . . | 4. z : |E| List
| . . . | 5. ∀i,j:ℕ||x||.
```

```
| . . . |    ¬is-send(E)(x[i]) ⇒
| . . . |    ¬is-send(E)(x[j]) ⇒
| . . . |    (x[j] =msg=(E) x[i]) ⇒
| . . . |    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . . | 6. ∀i,j:ℕ||y||.
| . . . |    ¬is-send(E)(y[i]) ⇒
| . . . |    ¬is-send(E)(y[j]) ⇒
| . . . |    (y[j] =msg=(E) y[i]) ⇒
| . . . |    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . . . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . . . | 8. z = (x @ y)
| . . . | 9. i : ℕ||z||
| . . . | 10. j : ℕ||z||
| . . . | 11. ¬is-send(E)(z[i])
| . . . | 12. ¬is-send(E)(z[j])
| . . . | 13. z[j] =msg=(E) z[i]
| . . . | 14. loc(E)(z[i]) = loc(E)(z[j])
| . . . | 15. i<j
| . . . | 16. ¬j<||x||
| . . . | 17. ||z|| = ||x||+||y||
| . . . | 18. z[j] = y[(j-||x||)]
| . . . | 19. ¬i<||x||
| . . . | ⊢  (x @ y)[i] = y[(i-||x||)]
| . . . |   by BackThruLemma ‘select_append_back‘ THEN Auto’
| . . . |
| . . . .\
| . . . . 1. E : EventStruct
| . . . . 2. x : |E| List
| . . . . 3. y : |E| List
| . . . . 4. z : |E| List
| . . . . 5. ∀i,j:ℕ||x||.
| . . . .    ¬is-send(E)(x[i]) ⇒
| . . . .    ¬is-send(E)(x[j]) ⇒
| . . . .    (x[j] =msg=(E) x[i]) ⇒
| . . . .    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
| . . . . 6. ∀i,j:ℕ||y||.
| . . . .    ¬is-send(E)(y[i]) ⇒
| . . . .    ¬is-send(E)(y[j]) ⇒
| . . . .    (y[j] =msg=(E) y[i]) ⇒
| . . . .    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
| . . . . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
| . . . . 8. z = (x @ y)
| . . . . 9. i : ℕ||z||
| . . . . 10. j : ℕ||z||
| . . . . 11. ¬is-send(E)(z[i])
| . . . . 12. ¬is-send(E)(z[j])
| . . . . 13. z[j] =msg=(E) z[i]
| . . . . 14. loc(E)(z[i]) = loc(E)(z[j])
| . . . . 15. i<j
| . . . . 16. ¬j<||x||
| . . . . 17. ||z|| = ||x||+||y||
| . . . . 18. z[j] = y[(j-||x||)]
| . . . . 19. ¬i<||x||
| . . . . 20. z[i] = y[(i-||x||)]
| . . . . ⊢  i = j
```

```
| . . . .   by InstHyp [i-||x||;j-||x||] 6 THEN Auto'
| . . . . THEN SubstFor
| . . . .      y[(i-||x||)] 0 THENA Auto THEN Try Trivial
| . . . .
| . . . .      THEN SubstFor y[(j-||x||)]
| . . . .       0 THENA Auto THEN Try Trivial
|
.\
. 1. E : EventStruct
. 2. x : |E| List
. 3. y : |E| List
. 4. z : |E| List
. 5. ∀i,j:ℕ||x||.
. |  ¬is-send(E)(x[i]) ⇒
. |  ¬is-send(E)(x[j]) ⇒
. |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. 6. ∀i,j:ℕ||y||.
. |  ¬is-send(E)(y[i]) ⇒
. |  ¬is-send(E)(y[j]) ⇒
. |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. 8. z = (x @ y)
. 9. i : ℕ||z||
. 10. j : ℕ||z||
. 11. ¬is-send(E)(z[i])
. 12. ¬is-send(E)(z[j])
. 13. z[j] =msg=(E) z[i]
. 14. loc(E)(z[i]) = loc(E)(z[j])
. 15. ¬i<j
. ⊢  i = j  by Decide j<i THENA Auto
. |
. |\
. | 1. E : EventStruct
. | 2. x : |E| List
. | 3. y : |E| List
. | 4. z : |E| List
. | 5. ∀i,j:ℕ||x||.
. | |  ¬is-send(E)(x[i]) ⇒
. | |  ¬is-send(E)(x[j]) ⇒
. | |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | 6. ∀i,j:ℕ||y||.
. | |  ¬is-send(E)(y[i]) ⇒
. | |  ¬is-send(E)(y[j]) ⇒
. | |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | 8. z = (x @ y)
. | 9. i : ℕ||z||
. | 10. j : ℕ||z||
. | 11. ¬is-send(E)(z[i])
. | 12. ¬is-send(E)(z[j])
. | 13. z[j] =msg=(E) z[i]
. | 14. loc(E)(z[i]) = loc(E)(z[j])
. | 15. ¬i<j
. | 16. j<i
. | ⊢  i = j  by Decide i<||x|| THENA Auto'
```

```
. | |
. | |\
. | | 1. E : EventStruct
. | | 2. x : |E| List
. | | 3. y : |E| List
. | | 4. z : |E| List
. | | 5. ∀i,j:ℕ||x||.
. | | |  ¬is-send(E)(x[i]) ⇒
. | | |  ¬is-send(E)(x[j]) ⇒
. | | |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | | 6. ∀i,j:ℕ||y||.
. | | |  ¬is-send(E)(y[i]) ⇒
. | | |  ¬is-send(E)(y[j]) ⇒
. | | |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | | 8. z = (x @ y)
. | | 9. i : ℕ||z||
. | | 10. j : ℕ||z||
. | | 11. ¬is-send(E)(z[i])
. | | 12. ¬is-send(E)(z[j])
. | | 13. z[j] =msg=(E) z[i]
. | | 14. loc(E)(z[i]) = loc(E)(z[j])
. | | 15. ¬i<j
. | | 16. j<i
. | | 17. i<||x||
. | | ⊢  i = j  by Assert z[j] = x[j]
. | | |
. | | |\
. | | | .....assertion .....
. | | | 1. E : EventStruct
. | | | 2. x : |E| List
. | | | 3. y : |E| List
. | | | 4. z : |E| List
. | | | 5. ∀i,j:ℕ||x||.
. | | |    ¬is-send(E)(x[i]) ⇒
. | | |    ¬is-send(E)(x[j]) ⇒
. | | |    (x[j] =msg=(E) x[i]) ⇒
. | | |    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | | | 6. ∀i,j:ℕ||y||.
. | | |    ¬is-send(E)(y[i]) ⇒
. | | |    ¬is-send(E)(y[j]) ⇒
. | | |    (y[j] =msg=(E) y[i]) ⇒

. | | |    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | | | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | | | 8. z = (x @ y)
. | | | 9. i : ℕ||z||
. | | | 10. j : ℕ||z||
. | | | 11. ¬is-send(E)(z[i])
. | | | 12. ¬is-send(E)(z[j])
. | | | 13. z[j] =msg=(E) z[i]
. | | | 14. loc(E)(z[i]) = loc(E)(z[j])
. | | | 15. ¬i<j
. | | | 16. j<i
. | | | 17. i<||x||
. | | | ⊢  z[j] = x[j]
```

```
. | | |   by HypSubstList 8 0 THENA Auto' THEN
. | | |
. | | |      BackThruLemma `select_append_front` THENA Auto
. | | |
. | | |
. | | .\
. | | . 1. E : EventStruct
. | | . 2. x : |E| List
. | | . 3. y : |E| List
. | | . 4. z : |E| List
. | | . 5. ∀i,j:ℕ||x||.
. | | . |  ¬is-send(E)(x[i]) ⇒
. | | . |  ¬is-send(E)(x[j]) ⇒
. | | . |  (x[j] =msg=(E) x[i]) ⇒
. | | . |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | | . 6. ∀i,j:ℕ||y||.
. | | . |  ¬is-send(E)(y[i]) ⇒
. | | . |  ¬is-send(E)(y[j]) ⇒
. | | . |  (y[j] =msg=(E) y[i]) ⇒
. | | . |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | | . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | | . 8. z = (x @ y)
. | | . 9. i : ℕ||z||
. | | . 10. j : ℕ||z||
. | | . 11. ¬is-send(E)(z[i])
. | | . 12. ¬is-send(E)(z[j])
. | | . 13. z[j] =msg=(E) z[i]
. | | . 14. loc(E)(z[i]) = loc(E)(z[j])
. | | . 15. ¬i<j
. | | . 16. j<i
. | | . 17. i<||x||
. | | . 18. z[j] = x[j]
. | | . ⊢  i = j  by Assert z[i] = x[i]
. | | . |
. | | . |\
. | | . | .....assertion .....
. | | . | 1. E : EventStruct
. | | . | 2. x : |E| List
. | | . | 3. y : |E| List
. | | . | 4. z : |E| List
. | | . | 5. ∀i,j:ℕ||x||.
. | | . |    ¬is-send(E)(x[i]) ⇒
. | | . |    ¬is-send(E)(x[j]) ⇒
. | | . |    (x[j] =msg=(E) x[i]) ⇒
. | | . |    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | | . | 6. ∀i,j:ℕ||y||.
. | | . |    ¬is-send(E)(y[i]) ⇒
. | | . |    ¬is-send(E)(y[j]) ⇒
. | | . |    (y[j] =msg=(E) y[i]) ⇒
. | | . |    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | | . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | | . | 8. z = (x @ y)
. | | . | 9. i : ℕ||z||
. | | . | 10. j : ℕ||z||
. | | . | 11. ¬is-send(E)(z[i])
```

```
. | | . | 12. ¬is-send(E)(z[j])
. | | . | 13. z[j] =msg=(E) z[i]
. | | . | 14. loc(E)(z[i]) = loc(E)(z[j])
. | | . | 15. ¬i<j
. | | . | 16. j<i
. | | . | 17. i<||x||
. | | . | 18. z[j] = x[j]
. | | . | ⊢  z[i] = x[i]
. | | . |   by HypSubstList 8 0 THENA Auto' THEN
. | | . |
. | | . |      BackThruLemma `select_append_front` THENA Auto
. | | . |
. | | . |
. | | . .\
. | | . . 1. E : EventStruct
. | | . . 2. x : |E| List
. | | . . 3. y : |E| List
. | | . . 4. z : |E| List
. | | . . 5. ∀i,j:ℕ||x||.
. | | . . |  ¬is-send(E)(x[i]) ⇒
. | | . . |  ¬is-send(E)(x[j]) ⇒
. | | . . |  (x[j] =msg=(E) x[i]) ⇒
. | | . . |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | | . . 6. ∀i,j:ℕ||y||.
. | | . . |  ¬is-send(E)(y[i]) ⇒
. | | . . |  ¬is-send(E)(y[j]) ⇒
. | | . . |  (y[j] =msg=(E) y[i]) ⇒
. | | . . |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | | . . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | | . . 8. z = (x @ y)
. | | . . 9. i : ℕ||z||
. | | . . 10. j : ℕ||z||
. | | . . 11. ¬is-send(E)(z[i])
. | | . . 12. ¬is-send(E)(z[j])
. | | . . 13. z[j] =msg=(E) z[i]
. | | . . 14. loc(E)(z[i]) = loc(E)(z[j])
. | | . . 15. ¬i<j
. | | . . 16. j<i
. | | . . 17. i<||x||
. | | . . 18. z[j] = x[j]
. | | . . 19. z[i] = x[i]
. | | . . ⊢  i = j
. | | . . | by InstHyp [j;i] 5 THEN Auto
. | | . . THEN SubstFor x[i]
. | | . . |     0 THEN Try Trivial
. | | . . THEN SubstFor x[j]
. | | . . |     0 THEN Try Trivial
. | | . . |
. | | . . 1. E : EventStruct
. | | . . 2. x : |E| List
. | | . . 3. y : |E| List
. | | . . 4. z : |E| List
. | | . . 5. ∀i,j:ℕ||x||.
. | | . .    ¬is-send(E)(x[i]) ⇒
. | | . .    ¬is-send(E)(x[j]) ⇒
```

```
. | | . .    (x[j] =msg=(E) x[i]) ⇒
. | | . .    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | | . . 6. ∀i,j:ℕ||y||.
. | | . .    ¬is-send(E)(y[i]) ⇒
. | | . .    ¬is-send(E)(y[j]) ⇒
. | | . .    (y[j] =msg=(E) y[i]) ⇒
. | | . .    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | | . . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | | . . 8. z = (x @ y)
. | | . . 9. i : ℕ||z||
. | | . . 10. j : ℕ||z||
. | | . . 11. ¬is-send(E)(z[i])
. | | . . 12. ¬is-send(E)(z[j])
. | | . . 13. z[j] =msg=(E) z[i]
. | | . . 14. loc(E)(z[i]) = loc(E)(z[j])
. | | . . 15. ¬i<j
. | | . . 16. j<i
. | | . . 17. i<||x||
. | | . . 18. z[j] = x[j]
. | | . . 19. z[i] = x[i]
. | | . . ⊢  z[i] =msg=(E) z[j]
. | | . .   by InstLemma ‘event_msg_eq_equiv‘ [E] THENA Auto
. | | . .
. | | . .      THEN UseEquiv
. | |
. | .\
. | . 1. E : EventStruct
. | . 2. x : |E| List
. | . 3. y : |E| List
. | . 4. z : |E| List
. | . 5. ∀i,j:ℕ||x||.
. | . |  ¬is-send(E)(x[i]) ⇒
. | . |  ¬is-send(E)(x[j]) ⇒
. | . |  (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . 6. ∀i,j:ℕ||y||.
. | . |  ¬is-send(E)(y[i]) ⇒
. | . |  ¬is-send(E)(y[j]) ⇒
. | . |  (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . 8. z = (x @ y)
. | . 9. i : ℕ||z||
. | . 10. j : ℕ||z||
. | . 11. ¬is-send(E)(z[i])
. | . 12. ¬is-send(E)(z[j])
. | . 13. z[j] =msg=(E) z[i]
. | . 14. loc(E)(z[i]) = loc(E)(z[j])
. | . 15. ¬i<j
. | . 16. j<i
. | . 17. ¬i<||x||
. | . ⊢  i = j
. | . | by InstLemma ‘length_append‘ [|E|;x;y] THENA Auto
. | .
. | . |    THEN SubstFor x @ y (-1) THENA Auto
. | . THEN Assert z[i]
. | . |                                                    =
```

```
. | . |                                                   y[(i
. | . |                                                   -||x||)]
. | . |
. | . |\
. | . | .....assertion .....
. | . | 1. E : EventStruct
. | . | 2. x : |E| List
. | . | 3. y : |E| List
. | . | 4. z : |E| List
. | . | 5. ∀i,j:ℕ||x||.
. | . | |  ¬is-send(E)(x[i]) ⇒
. | . | |  ¬is-send(E)(x[j]) ⇒
. | . | |  (x[j] =msg=(E) x[i]) ⇒
. | . | |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . | 6. ∀i,j:ℕ||y||.
. | . | |  ¬is-send(E)(y[i]) ⇒
. | . | |  ¬is-send(E)(y[j]) ⇒
. | . | |  (y[j] =msg=(E) y[i]) ⇒
. | . | |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . | 8. z = (x @ y)
. | . | 9. i : ℕ||z||
. | . | 10. j : ℕ||z||
. | . | 11. ¬is-send(E)(z[i])
. | . | 12. ¬is-send(E)(z[j])
. | . | 13. z[j] =msg=(E) z[i]
. | . | 14. loc(E)(z[i]) = loc(E)(z[j])
. | . | 15. ¬i<j
. | . | 16. j<i
. | . | 17. ¬i<||x||
. | . | 18. ||z|| = ||x||+||y||
. | . | ⊢  z[i] = y[(i-||x||)]  by HypSubstList 8 0 THENA Auto'
. | . | |
. | . | 1. E : EventStruct
. | . | 2. x : |E| List
. | . | 3. y : |E| List
. | . | 4. z : |E| List
. | . | 5. ∀i,j:ℕ||x||.
. | . |    ¬is-send(E)(x[i]) ⇒
. | . |    ¬is-send(E)(x[j]) ⇒
. | . |    (x[j] =msg=(E) x[i]) ⇒
. | . |    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . | 6. ∀i,j:ℕ||y||.
. | . |    ¬is-send(E)(y[i]) ⇒
. | . |    ¬is-send(E)(y[j]) ⇒
. | . |    (y[j] =msg=(E) y[i]) ⇒
. | . |    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . | 8. z = (x @ y)
. | . | 9. i : ℕ||z||
. | . | 10. j : ℕ||z||
. | . | 11. ¬is-send(E)(z[i])
. | . | 12. ¬is-send(E)(z[j])
. | . | 13. z[j] =msg=(E) z[i]
. | . | 14. loc(E)(z[i]) = loc(E)(z[j])
```

```
. | . | 15. ¬i<j
. | . | 16. j<i
. | . | 17. ¬i<||x||
. | . | 18. ||z|| = ||x||+||y||
. | . | ⊢  (x @ y)[i] = y[(i-||x||)]
. | . |   by BackThruLemma ‘select_append_back‘ THEN Auto’
. | . |
. | . .\
. | . . 1. E : EventStruct
. | . . 2. x : |E| List
. | . . 3. y : |E| List
. | . . 4. z : |E| List
. | . . 5. ∀i,j:ℕ||x||.
. | . . |  ¬is-send(E)(x[i]) ⇒
. | . . |  ¬is-send(E)(x[j]) ⇒
. | . . |  (x[j] =msg=(E) x[i]) ⇒
. | . . |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . . 6. ∀i,j:ℕ||y||.
. | . . |  ¬is-send(E)(y[i]) ⇒
. | . . |  ¬is-send(E)(y[j]) ⇒
. | . . |  (y[j] =msg=(E) y[i]) ⇒
. | . . |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . . 8. z = (x @ y)
. | . . 9. i : ℕ||z||
. | . . 10. j : ℕ||z||
. | . . 11. ¬is-send(E)(z[i])
. | . . 12. ¬is-send(E)(z[j])
. | . . 13. z[j] =msg=(E) z[i]
. | . . 14. loc(E)(z[i]) = loc(E)(z[j])
. | . . 15. ¬i<j
. | . . 16. j<i
. | . . 17. ¬i<||x||
. | . . 18. ||z|| = ||x||+||y||
. | . . 19. z[i] = y[(i-||x||)]
. | . . ⊢  i = j  by Decide j<||x|| THENA Auto
. | . . |
. | . . |\
. | . . | 1. E : EventStruct
. | . . | 2. x : |E| List
. | . . | 3. y : |E| List
. | . . | 4. z : |E| List
. | . . | 5. ∀i,j:ℕ||x||.
. | . . | |  ¬is-send(E)(x[i]) ⇒
. | . . | |  ¬is-send(E)(x[j]) ⇒
. | . . | |  (x[j] =msg=(E) x[i]) ⇒
. | . . | |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . . | 6. ∀i,j:ℕ||y||.
. | . . | |  ¬is-send(E)(y[i]) ⇒
. | . . | |  ¬is-send(E)(y[j]) ⇒
. | . . | |  (y[j] =msg=(E) y[i]) ⇒
. | . . | |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . . | 8. z = (x @ y)
. | . . | 9. i : ℕ||z||
```

```
. | . . | 10. j : ℕ||z||
. | . . | 11. ¬is-send(E)(z[i])
. | . . | 12. ¬is-send(E)(z[j])
. | . . | 13. z[j] =msg=(E) z[i]
. | . . | 14. loc(E)(z[i]) = loc(E)(z[j])
. | . . | 15. ¬i<j
. | . . | 16. j<i
. | . . | 17. ¬i<||x||
. | . . | 18. ||z|| = ||x||+||y||
. | . . | 19. z[i] = y[(i-||x||)]
. | . . | 20. j<||x||
. | . . | ⊢  i = j  by Assert z[j] = x[j]
. | . . | |
. | . . | |\
. | . . | | .....assertion .....
. | . . | | 1. E : EventStruct
. | . . | | 2. x : |E| List
. | . . | | 3. y : |E| List
. | . . | | 4. z : |E| List
. | . . | | 5. ∀i,j:ℕ||x||.
. | . . | |    ¬is-send(E)(x[i]) ⇒
. | . . | |    ¬is-send(E)(x[j]) ⇒
. | . . | |    (x[j] =msg=(E) x[i]) ⇒
. | . . | |    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . . | | 6. ∀i,j:ℕ||y||.
. | . . | |    ¬is-send(E)(y[i]) ⇒
. | . . | |    ¬is-send(E)(y[j]) ⇒
. | . . | |    (y[j] =msg=(E) y[i]) ⇒
. | . . | |    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . . | | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . . | | 8. z = (x @ y)
. | . . | | 9. i : ℕ||z||
. | . . | | 10. j : ℕ||z||
. | . . | | 11. ¬is-send(E)(z[i])
. | . . | | 12. ¬is-send(E)(z[j])
. | . . | | 13. z[j] =msg=(E) z[i]
. | . . | | 14. loc(E)(z[i]) = loc(E)(z[j])
. | . . | | 15. ¬i<j
. | . . | | 16. j<i
. | . . | | 17. ¬i<||x||
. | . . | | 18. ||z|| = ||x||+||y||
. | . . | | 19. z[i] = y[(i-||x||)]
. | . . | | 20. j<||x||
. | . . | | ⊢  z[j] = x[j]
. | . . | |   by HypSubstList 8 0 THENA Auto' THEN
. | . . | |
. | . . | |      BackThruLemma ‘select_append_front‘ THENA Auto
. | . . | |
. | . . | |
. | . . | .\
. | . . | . 1. E : EventStruct
. | . . | . 2. x : |E| List
. | . . | . 3. y : |E| List
. | . . | . 4. z : |E| List
. | . . | . 5. ∀i,j:ℕ||x||.
```

```
. | . . | . |  ¬is-send(E)(x[i]) ⇒
. | . . | . |  ¬is-send(E)(x[j]) ⇒
. | . . | . |  (x[j] =msg=(E) x[i]) ⇒
. | . . | . |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . . | . 6. ∀i,j:ℕ||y||.
. | . . | . |  ¬is-send(E)(y[i]) ⇒
. | . . | . |  ¬is-send(E)(y[j]) ⇒
. | . . | . |  (y[j] =msg=(E) y[i]) ⇒
. | . . | . |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . . | . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . . | . 8. z = (x @ y)
. | . . | . 9. i : ℕ||z||
. | . . | . 10. j : ℕ||z||
. | . . | . 11. ¬is-send(E)(z[i])
. | . . | . 12. ¬is-send(E)(z[j])
. | . . | . 13. z[j] =msg=(E) z[i]
. | . . | . 14. loc(E)(z[i]) = loc(E)(z[j])
. | . . | . 15. ¬i<j
. | . . | . 16. j<i
. | . . | . 17. ¬i<||x||
. | . . | . 18. ||z|| = ||x||+||y||
. | . . | . 19. z[i] = y[(i-||x||)]
. | . . | . 20. j<||x||
. | . . | . 21. z[j] = x[j]
. | . . | . ⊢  i = j
. | . . | . | by AllHyps (\h. Unfold ‘l_all‘ h THEN InstHyp [x[j];y[(i
. | . . | . |    -||x||)]] h THEN Auto’)
. | . . | .
. | . . | . |    THEN Try (BackThruLemma ‘select_member‘ THEN Auto’)
. | . . | . |
. | . . | . 1. E : EventStruct
. | . . | . 2. x : |E| List
. | . . | . 3. y : |E| List
. | . . | . 4. z : |E| List
. | . . | . 5. ∀i,j:ℕ||x||.
. | . . | .    ¬is-send(E)(x[i]) ⇒
. | . . | .    ¬is-send(E)(x[j]) ⇒
. | . . | .    (x[j] =msg=(E) x[i]) ⇒
. | . . | .    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . . | . 6. ∀i,j:ℕ||y||.
. | . . | .    ¬is-send(E)(y[i]) ⇒
. | . . | .    ¬is-send(E)(y[j]) ⇒
. | . . | .    (y[j] =msg=(E) y[i]) ⇒
. | . . | .    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . . | . 7. ∀x@0:|E|.
. | . . | .    (x@0 ∈ x) ⇒
. | . . | .    (∀y@0:|E|. (y@0 ∈ y) ⇒ ¬(x@0 =msg=(E) y@0))
. | . . | . 8. z = (x @ y)
. | . . | . 9. i : ℕ||z||
. | . . | . 10. j : ℕ||z||
. | . . | . 11. ¬is-send(E)(z[i])
. | . . | . 12. ¬is-send(E)(z[j])
. | . . | . 13. z[j] =msg=(E) z[i]
. | . . | . 14. loc(E)(z[i]) = loc(E)(z[j])
. | . . | . 15. ¬i<j
```

```
. | . . | . 16. j<i
. | . . | . 17. ¬i<||x||
. | . . | . 18. ||z|| = ||x||+||y||
. | . . | . 19. z[i] = y[(i-||x||)]
. | . . | . 20. j<||x||
. | . . | . 21. z[j] = x[j]
. | . . | . 22. ¬(x[j] =msg=(E) y[(i-||x||)])
. | . . | . ⊢  i = j
. | . . | .   by D (-1) THEN SubstFor x[j] 0 THENA Auto
. | . . | .
. | . . | .      THEN SubstFor y[(i-||x||)] 0 THEN Auto
. | . . |
. | . . .\
. | . . . 1. E : EventStruct
. | . . . 2. x : |E| List
. | . . . 3. y : |E| List
. | . . . 4. z : |E| List
. | . . . 5. ∀i,j:ℕ||x||.
. | . . . |  ¬is-send(E)(x[i]) ⇒
. | . . . |  ¬is-send(E)(x[j]) ⇒
. | . . . |  (x[j] =msg=(E) x[i]) ⇒
. | . . . |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . . . 6. ∀i,j:ℕ||y||.
. | . . . |  ¬is-send(E)(y[i]) ⇒
. | . . . |  ¬is-send(E)(y[j]) ⇒
. | . . . |  (y[j] =msg=(E) y[i]) ⇒
. | . . . |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . . . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . . . 8. z = (x @ y)
. | . . . 9. i : ℕ||z||
. | . . . 10. j : ℕ||z||
. | . . . 11. ¬is-send(E)(z[i])
. | . . . 12. ¬is-send(E)(z[j])
. | . . . 13. z[j] =msg=(E) z[i]
. | . . . 14. loc(E)(z[i]) = loc(E)(z[j])
. | . . . 15. ¬i<j
. | . . . 16. j<i
. | . . . 17. ¬i<||x||
. | . . . 18. ||z|| = ||x||+||y||
. | . . . 19. z[i] = y[(i-||x||)]
. | . . . 20. ¬j<||x||
. | . . . ⊢  i = j  by Assert z[j] = y[(j-||x||)]
. | . . . |
. | . . . |\
. | . . . | .....assertion .....
. | . . . | 1. E : EventStruct
. | . . . | 2. x : |E| List
. | . . . | 3. y : |E| List
. | . . . | 4. z : |E| List
. | . . . | 5. ∀i,j:ℕ||x||.
. | . . . | |  ¬is-send(E)(x[i]) ⇒
. | . . . | |  ¬is-send(E)(x[j]) ⇒
. | . . . | |  (x[j] =msg=(E) x[i]) ⇒
. | . . . | |  loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . . . | 6. ∀i,j:ℕ||y||.
```

```
. | . . . | |  ¬is-send(E)(y[i]) ⇒
. | . . . | |  ¬is-send(E)(y[j]) ⇒
. | . . . | |  (y[j] =msg=(E) y[i]) ⇒
. | . . . | |  loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . . . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . . . | 8. z = (x @ y)
. | . . . | 9. i : ℕ||z||
. | . . . | 10. j : ℕ||z||
. | . . . | 11. ¬is-send(E)(z[i])
. | . . . | 12. ¬is-send(E)(z[j])
. | . . . | 13. z[j] =msg=(E) z[i]
. | . . . | 14. loc(E)(z[i]) = loc(E)(z[j])
. | . . . | 15. ¬i<j
. | . . . | 16. j<i
. | . . . | 17. ¬i<||x||
. | . . . | 18. ||z|| = ||x||+||y||
. | . . . | 19. z[i] = y[(i-||x||)]
. | . . . | 20. ¬j<||x||
. | . . . | ⊢  z[j] = y[(j-||x||)]  by HypSubstList 8 0 THENA Auto'
. | . . . | |
. | . . . | 1. E : EventStruct
. | . . . | 2. x : |E| List
. | . . . | 3. y : |E| List
. | . . . | 4. z : |E| List
. | . . . | 5. ∀i,j:ℕ||x||.
. | . . . |    ¬is-send(E)(x[i]) ⇒
. | . . . |    ¬is-send(E)(x[j]) ⇒
. | . . . |    (x[j] =msg=(E) x[i]) ⇒
. | . . . |    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . . . | 6. ∀i,j:ℕ||y||.
. | . . . |    ¬is-send(E)(y[i]) ⇒
. | . . . |    ¬is-send(E)(y[j]) ⇒
. | . . . |    (y[j] =msg=(E) y[i]) ⇒
. | . . . |    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . . . | 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . . . | 8. z = (x @ y)
. | . . . | 9. i : ℕ||z||
. | . . . | 10. j : ℕ||z||
. | . . . | 11. ¬is-send(E)(z[i])
. | . . . | 12. ¬is-send(E)(z[j])
. | . . . | 13. z[j] =msg=(E) z[i]
. | . . . | 14. loc(E)(z[i]) = loc(E)(z[j])
. | . . . | 15. ¬i<j
. | . . . | 16. j<i
. | . . . | 17. ¬i<||x||
. | . . . | 18. ||z|| = ||x||+||y||
. | . . . | 19. z[i] = y[(i-||x||)]
. | . . . | 20. ¬j<||x||
. | . . . | ⊢  (x @ y)[j] = y[(j-||x||)]
. | . . . |   by BackThruLemma `select_append_back` THEN Auto'
. | . . . |
. | . . . .\
. | . . . . 1. E : EventStruct
. | . . . . 2. x : |E| List
. | . . . . 3. y : |E| List
```

```
. | . . . . 4. z : |E| List
. | . . . . 5. ∀i,j:ℕ||x||.
. | . . . .    ¬is-send(E)(x[i]) ⇒
. | . . . .    ¬is-send(E)(x[j]) ⇒
. | . . . .    (x[j] =msg=(E) x[i]) ⇒
. | . . . .    loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. | . . . . 6. ∀i,j:ℕ||y||.
. | . . . .    ¬is-send(E)(y[i]) ⇒
. | . . . .    ¬is-send(E)(y[j]) ⇒
. | . . . .    (y[j] =msg=(E) y[i]) ⇒
. | . . . .    loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. | . . . . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. | . . . . 8. z = (x @ y)
. | . . . . 9. i : ℕ||z||
. | . . . . 10. j : ℕ||z||
. | . . . . 11. ¬is-send(E)(z[i])
. | . . . . 12. ¬is-send(E)(z[j])
. | . . . . 13. z[j] =msg=(E) z[i]
. | . . . . 14. loc(E)(z[i]) = loc(E)(z[j])
. | . . . . 15. ¬i<j
. | . . . . 16. j<i
. | . . . . 17. ¬i<||x||
. | . . . . 18. ||z|| = ||x||+||y||
. | . . . . 19. z[i] = y[(i-||x||)]
. | . . . . 20. ¬j<||x||
. | . . . . 21. z[j] = y[(j-||x||)]
. | . . . . ⊢  i = j
. | . . . .   by InstHyp [i-||x||;j-||x||] 6 THEN Auto'
. | . . . .
. | . . . .      THEN SubstFor y[(i-||x||)]
. | . . . .       0 THENA Auto THEN Try Trivial
. | . . . . THEN SubstFor y[(j
. | . . . .      -||x||)] 0 THENA Auto THEN Try Trivial
. |
. .\
. . 1. E : EventStruct
. . 2. x : |E| List
. . 3. y : |E| List
. . 4. z : |E| List
. . 5. ∀i,j:ℕ||x||.
. .    ¬is-send(E)(x[i]) ⇒
. .    ¬is-send(E)(x[j]) ⇒
. .    (x[j] =msg=(E) x[i]) ⇒ loc(E)(x[i]) = loc(E)(x[j]) ⇒ i = j
. . 6. ∀i,j:ℕ||y||.
. .    ¬is-send(E)(y[i]) ⇒
. .    ¬is-send(E)(y[j]) ⇒
. .    (y[j] =msg=(E) y[i]) ⇒ loc(E)(y[i]) = loc(E)(y[j]) ⇒ i = j
. . 7. (∀x∈x.(∀y∈y.¬(x =msg=(E) y)))
. . 8. z = (x @ y)
. . 9. i : ℕ||z||
. . 10. j : ℕ||z||
. . 11. ¬is-send(E)(z[i])
. . 12. ¬is-send(E)(z[j])
. . 13. z[j] =msg=(E) z[i]
. . 14. loc(E)(z[i]) = loc(E)(z[j])
```

```
. . 15. ¬i<j
. . 16. ¬j<i
. . ⊢  i = j  by Auto
END
```