```
dfs_induction2  THM

⊢  For any graph
|  ∀the_obj:GraphObject(the_graph), P:(V→Traversal→Traversal→Prop)
|  , s:Traversal, i:V.
|  (∀s:Traversal, i:V. (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil))
|  ⇒
|  (∀s1,s2,s3:Traversal, i,j:V.
|   i-the_graph->j
|   ⇒
|   paren(V;s2)
|   ⇒
|   paren(V;s3) ⇒ P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2))
|  ⇒
|  (∀s1,s2:Traversal, i:V.
|   ¬(inl(i) ∈ s1)
|   ⇒
|   ¬(inr(i) ∈ s1)
|   ⇒
|   P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])]))
|  ⇒
|  (∃s':Traversal.
|   P(i,s,s')
|   & l_disjoint(V+V;s';s) & no_repeats(V+V;s')
|   & paren(V;s')
|   & dfs(the_obj;s;i) = (s' @ s))
| by Auto
|    THEN
|    (Inst Thm: graphobj_DASH_properties [the_graph;the_obj])
|    THEN
|    (Analyze -1)
|    THEN
|    (Analyze -1)
|    THEN
|    (Inst Thm: dfs_induction
|     [the_graph
|     ;the_obj
|     ;λ2i,s1,s2.
|      P(i,s1,s2)
|      & l_disjoint(Vertices(the_graph)+Vertices(the_graph);s2;s1)
|      & no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
|      & paren(Vertices(the_graph);s2)
|     ;s
|     ;i])
|    THEN
|    (AllHyps
|     (RWO
|      Thm* ∀E:(T→T→𝔹).
|           (∀x,y:T. E(x,y) ⟺ x = y)
|           ⇒
|           (∀i:T, s:(T+T) List.
|            member-paren(x,y.E(x,y);i;s)
|            ⟺
|            (inl(i) ∈ s) ∨ (inr(i) ∈ s))
|      {assert_DASH_member_DASH_paren}))
```

```
|    THEN
|    EasyHyp
|
|\
| 1. the_graph : Graph
| 2. the_obj : GraphObject(the_graph)
| 3. P :
|    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
|    Prop
| 4. traversal(the_graph)
| 5. Vertices(the_graph)
| 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
|    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
|    i-the_graph->j
|    ⇒
|    paren(Vertices(the_graph);s2)
|    ⇒
|    paren(Vertices(the_graph);s3)
|    ⇒
|    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
|    ¬(inl(i) ∈ s1)
|    ⇒
|    ¬(inr(i) ∈ s1)
|    ⇒
|    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| 10. ∀T:Type, s:T, x:Vertices(the_graph)
|     , f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
|     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     no_repeats(Vertices(the_graph);L)
|     & (∀y:Vertices(the_graph). (y ∈ L))
|     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| 12. s1 : traversal(the_graph)
| 13. i1 : Vertices(the_graph)
| 14. (inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1)
| ⊢  l_disjoint(Vertices(the_graph)+Vertices(the_graph);nil;s1)
|   by ObviousConcl
|
|\
| 1. the_graph : Graph
| 2. the_obj : GraphObject(the_graph)
| 3. P :
|    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
|    Prop
| 4. traversal(the_graph)
| 5. Vertices(the_graph)
| 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
|    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
```

```
|    i-the_graph->j
|    ⇒
|    paren(Vertices(the_graph);s2)
|    ⇒
|    paren(Vertices(the_graph);s3)
|    ⇒
|    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
|    ¬(inl(i) ∈ s1)
|    ⇒
|    ¬(inr(i) ∈ s1)
|    ⇒
|    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| 10. ∀T:Type, s:T, x:Vertices(the_graph)
|     , f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
|     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     no_repeats(Vertices(the_graph);L)
|     & (∀y:Vertices(the_graph). (y ∈ L))
|     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| 12. s1 : traversal(the_graph)
| 13. i1 : Vertices(the_graph)
| 14. (inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1)
| ⊢  no_repeats(Vertices(the_graph)+Vertices(the_graph);nil)
|   by ObviousConcl
|
|\
| 1. the_graph : Graph
| 2. the_obj : GraphObject(the_graph)
| 3. P :
|    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
|    Prop
| 4. traversal(the_graph)
| 5. Vertices(the_graph)
| 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
|    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
|    i-the_graph->j
|    ⇒
|    paren(Vertices(the_graph);s2)
|    ⇒
|    paren(Vertices(the_graph);s3)
|    ⇒
|    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
|    ¬(inl(i) ∈ s1)
|    ⇒
|    ¬(inr(i) ∈ s1)
|    ⇒
|    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
```

```
| 10. ∀T:Type, s:T, x:Vertices(the_graph)
|     , f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
|     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     no_repeats(Vertices(the_graph);L)
|     & (∀y:Vertices(the_graph). (y ∈ L))
|     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| 12. s1 : traversal(the_graph)
| 13. i1 : Vertices(the_graph)
| 14. (inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1)
| ⊢  paren(Vertices(the_graph);nil)  by ObviousConcl
|
|\
| 1. the_graph : Graph
| 2. the_obj : GraphObject(the_graph)
| 3. P :
|    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
|    Prop
| 4. traversal(the_graph)
| 5. Vertices(the_graph)
| 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
|    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
|    i-the_graph->j
|    ⇒
|    paren(Vertices(the_graph);s2)
|    ⇒
|    paren(Vertices(the_graph);s3)
|    ⇒
|    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
|    ¬(inl(i) ∈ s1)
|    ⇒
|    ¬(inr(i) ∈ s1)
|    ⇒
|    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| 10. ∀T:Type, s:T, x:Vertices(the_graph)
|     , f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
|     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     no_repeats(Vertices(the_graph);L)
|     & (∀y:Vertices(the_graph). (y ∈ L))
|     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| 12. s1 : traversal(the_graph)
| 13. s2 : traversal(the_graph)
| 14. s3 : traversal(the_graph)
| 15. i1 : Vertices(the_graph)
| 16. j : Vertices(the_graph)
```

```
| 17. i1-the_graph->j
| 18. P(j,s1,s2)
| 19. l_disjoint(Vertices(the_graph)+Vertices(the_graph);s2;s1)
| 20. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| 21. paren(Vertices(the_graph);s2)
| 22. P(i1,s2 @ s1,s3)
| 23. l_disjoint(Vertices(the_graph)+Vertices(the_graph);s3;s2 @ s1)
| 24. no_repeats(Vertices(the_graph)+Vertices(the_graph);s3)
| 25. paren(Vertices(the_graph);s3)
| ⊢  l_disjoint(Vertices(the_graph)+Vertices(the_graph);s3 @ s2;s1)
|   by (RWO
|       Thm* ∀a,b,c:T List.
|            l_disjoint(T;c;a @ b)
|            ⟺
|            l_disjoint(T;c;a) & l_disjoint(T;c;b)
|       {l_disjoint_append2}
|       -3)
|      THEN
|      (RWO
|       Thm* ∀a,b,c:T List.
|            l_disjoint(T;a @ b;c)
|            ⟺
|            l_disjoint(T;a;c) & l_disjoint(T;b;c)
|       {l_disjoint_append}
|       0)
|
|\
| 1. the_graph : Graph
| 2. the_obj : GraphObject(the_graph)
| 3. P :
|    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
|    Prop
| 4. traversal(the_graph)
| 5. Vertices(the_graph)
| 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
|    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
|    i-the_graph->j
|    ⇒
|    paren(Vertices(the_graph);s2)
|    ⇒
|    paren(Vertices(the_graph);s3)
|    ⇒
|    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
|    ¬(inl(i) ∈ s1)
|    ⇒
|    ¬(inr(i) ∈ s1)
|    ⇒
|    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| 10. ∀T:Type, s:T, x:Vertices(the_graph)
|     , f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
```

```
|     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     no_repeats(Vertices(the_graph);L)
|     & (∀y:Vertices(the_graph). (y ∈ L))
|     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| 12. s1 : traversal(the_graph)
| 13. s2 : traversal(the_graph)
| 14. s3 : traversal(the_graph)
| 15. i1 : Vertices(the_graph)
| 16. j : Vertices(the_graph)
| 17. i1-the_graph->j
| 18. P(j,s1,s2)
| 19. l_disjoint(Vertices(the_graph)+Vertices(the_graph);s2;s1)
| 20. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| 21. paren(Vertices(the_graph);s2)
| 22. P(i1,s2 @ s1,s3)
| 23. l_disjoint(Vertices(the_graph)+Vertices(the_graph);s3;s2 @ s1)
| 24. no_repeats(Vertices(the_graph)+Vertices(the_graph);s3)
| 25. paren(Vertices(the_graph);s3)
| ⊢  no_repeats(Vertices(the_graph)+Vertices(the_graph);s3 @ s2)
|   by (RWO
|       Thm* ∀a,b,c:T List.
|            l_disjoint(T;c;a @ b)
|            ⟺
|            l_disjoint(T;c;a) & l_disjoint(T;c;b)
|       {l_disjoint_append2}
|       -3)
|      THEN
|      (RWO
|       Thm* ∀l1,l2:T List.
|            no_repeats(T;l1 @ l2)
|            ⟺
|            l_disjoint(T;l1;l2) & no_repeats(T;l1) & no_repeats(T;l2)
|       {no_repeats_append_iff}
|       0)
|
|\
| 1. the_graph : Graph
| 2. the_obj : GraphObject(the_graph)
| 3. P :
|    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
|    Prop
| 4. traversal(the_graph)
| 5. Vertices(the_graph)
| 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
|    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
|    i-the_graph->j
|    ⇒
|    paren(Vertices(the_graph);s2)
|    ⇒
|    paren(Vertices(the_graph);s3)
|    ⇒
|    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
```

```
| 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
|    ¬(inl(i) ∈ s1)
|    ⇒
|    ¬(inr(i) ∈ s1)
|    ⇒
|    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| 10. ∀T:Type, s:T, x:Vertices(the_graph)
|     , f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
|     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     no_repeats(Vertices(the_graph);L)
|     & (∀y:Vertices(the_graph). (y ∈ L))
|     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| 12. s1 : traversal(the_graph)
| 13. s2 : traversal(the_graph)
| 14. s3 : traversal(the_graph)
| 15. i1 : Vertices(the_graph)
| 16. j : Vertices(the_graph)
| 17. i1-the_graph->j
| 18. P(j,s1,s2)
| 19. l_disjoint(Vertices(the_graph)+Vertices(the_graph);s2;s1)
| 20. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| 21. paren(Vertices(the_graph);s2)
| 22. P(i1,s2 @ s1,s3)
| 23. l_disjoint(Vertices(the_graph)+Vertices(the_graph);s3;s2 @ s1)
| 24. no_repeats(Vertices(the_graph)+Vertices(the_graph);s3)
| 25. paren(Vertices(the_graph);s3)
| ⊢  paren(Vertices(the_graph);s3 @ s2)
|   by BackThru
|      Thm* ∀s',s3:(T+T) List.
|           paren(T;s') ⇒ paren(T;s3) ⇒ paren(T;s3 @ s')
|      {append_paren}
|
|\
| 1. the_graph : Graph
| 2. the_obj : GraphObject(the_graph)
| 3. P :
|    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
|    Prop
| 4. traversal(the_graph)
| 5. Vertices(the_graph)
| 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
|    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
|    i-the_graph->j
|    ⇒
|    paren(Vertices(the_graph);s2)
|    ⇒
|    paren(Vertices(the_graph);s3)
|    ⇒
|    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
```

```
| 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
|    ¬(inl(i) ∈ s1)
|    ⇒
|    ¬(inr(i) ∈ s1)
|    ⇒
|    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| 10. ∀T:Type, s:T, x:Vertices(the_graph)
|     , f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
|     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     no_repeats(Vertices(the_graph);L)
|     & (∀y:Vertices(the_graph). (y ∈ L))
|     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)

| 12. s1 : traversal(the_graph)
| 13. s2 : traversal(the_graph)
| 14. i1 : Vertices(the_graph)
| 15. ¬((inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1))
| 16. P(i1,[inr(i1) / s1],s2)
| 17. l_disjoint(Vertices(the_graph)+
|     Vertices(the_graph);s2;[inr(i1) / s1])
| 18. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| 19. paren(Vertices(the_graph);s2)
| ⊢  P(i1,s1,[inl(i1) / (s2 @ [inr(i1)])])
|   by BackThruSomeHyp
|      THEN
|      (AllHyps (λh.(ParallelOp h) THEN (ObviousFrom [-1])))
|
|\
| 1. the_graph : Graph
| 2. the_obj : GraphObject(the_graph)
| 3. P :
|    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
|    Prop
| 4. traversal(the_graph)
| 5. Vertices(the_graph)
| 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
|    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
|    i-the_graph->j
|    ⇒
|    paren(Vertices(the_graph);s2)
|    ⇒
|    paren(Vertices(the_graph);s3)
|    ⇒
|    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
|    ¬(inl(i) ∈ s1)
|    ⇒
|    ¬(inr(i) ∈ s1)
|    ⇒
|    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
```

```
| 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| 10. ∀T:Type, s:T, x:Vertices(the_graph)
|     , f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
|     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
|     ∃L:Vertices(the_graph) List.
|     no_repeats(Vertices(the_graph);L)
|     & (∀y:Vertices(the_graph). (y ∈ L))
|     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| 12. s1 : traversal(the_graph)
| 13. s2 : traversal(the_graph)
| 14. i1 : Vertices(the_graph)
| 15. ¬((inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1))
| 16. P(i1,[inr(i1) / s1],s2)
| 17. l_disjoint(Vertices(the_graph)+
|     Vertices(the_graph);s2;[inr(i1) / s1])
| 18. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| 19. paren(Vertices(the_graph);s2)
| ⊢  l_disjoint(Vertices(the_graph)+
|    Vertices(the_graph);[inl(i1) / (s2 @ [inr(i1)])];s1)
|   by ((RWO
|        Thm* ∀a,b:T List, t:T.
|             l_disjoint(T;b;[t / a]) ⟺ ¬(t ∈ b) & l_disjoint(T;b;a)
|        {l_disjoint_cons2}
|        -3)
|       THEN
|       (RWO
|        Thm* ∀a,b:T List, t:T.
|             l_disjoint(T;[t / a];b) ⟺ ¬(t ∈ b) & l_disjoint(T;a;b)
|        {l_disjoint_cons}
|        0)
|       THEN
|       (RWO
|        Thm* ∀a,b,c:T List.
|             l_disjoint(T;a @ b;c)
|             ⟺
|             l_disjoint(T;a;c) & l_disjoint(T;b;c)
|        {l_disjoint_append}
|        0)
|       THEN
|       (RWO
|        Thm* ∀a,b:T List, t:T.
|             l_disjoint(T;[t / a];b) ⟺ ¬(t ∈ b) & l_disjoint(T;a;b)
|        {l_disjoint_cons}
|        0))
|      THEN
|      (ObviousFrom [-6])
|
|\
| 1. the_graph : Graph
| 2. the_obj : GraphObject(the_graph)
| 3. P :
| |  Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
```

```
| |  Prop
| 4. traversal(the_graph)
| 5. Vertices(the_graph)
| 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
| |  (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
| |  i-the_graph->j
| |  ⇒
| |  paren(Vertices(the_graph);s2)
| |  ⇒
| |  paren(Vertices(the_graph);s3)
| |  ⇒
| |  P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
| |  ¬(inl(i) ∈ s1)
| |  ⇒
| |  ¬(inr(i) ∈ s1)
| |  ⇒
| |  P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| 10. ∀T:Type, s:T, x:Vertices(the_graph)
| |   , f:(T→Vertices(the_graph)→T).
| |   ∃L:Vertices(the_graph) List.
| |   (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
| |   & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
| |   ∃L:Vertices(the_graph) List.
| |   no_repeats(Vertices(the_graph);L)
| |   & (∀y:Vertices(the_graph). (y ∈ L))
| |   & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| 12. s1 : traversal(the_graph)
| 13. s2 : traversal(the_graph)
| 14. i1 : Vertices(the_graph)
| 15. ¬((inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1))
| 16. P(i1,[inr(i1) / s1],s2)
| 17. l_disjoint(Vertices(the_graph)+
| |   Vertices(the_graph);s2;[inr(i1) / s1])
| 18. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| 19. paren(Vertices(the_graph);s2)
| ⊢  no_repeats(Vertices(the_graph)+
| |  Vertices(the_graph);[inl(i1) / (s2 @ [inr(i1)])])
| | by (RWO
| |     Thm* ∀l:T List, x:T.
| |          no_repeats(T;[x / l]) ⟺ no_repeats(T;l) & ¬(x ∈ l)
| |     {no_repeats_cons}
| |     0)
| |    THEN
| |    (Try
| |     (RWO
| |      Thm* ∀l1,l2:T List.
| |           no_repeats(T;l1 @ l2)
| |           ⟺
| |           l_disjoint(T;l1;l2)
| |           & no_repeats(T;l1)
| |           & no_repeats(T;l2)
```

```
| |      {no_repeats_append_iff}
| |      0))
| |    THEN
| |    (Try
| |     (RWO
| |      Thm* ∀x:T, l1,l2:T List. (x ∈ l1 @ l2) ⟺ (x ∈ l1) ∨ (x ∈ l2)
| |      {member_append}
| |      0))
| |    THEN
| |    (Try
| |     (RWO Thm* ∀a,x:T. (x ∈ [a]) ⟺ x = a    {member_singleton}
| |      0))
| |
| |\
| | 1. the_graph : Graph
| | 2. the_obj : GraphObject(the_graph)
| | 3. P :
| |    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)
| |    →Prop
| | 4. traversal(the_graph)
| | 5. Vertices(the_graph)
| | 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
| |    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| | 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
| |    i-the_graph->j
| |    ⇒
| |    paren(Vertices(the_graph);s2)
| |    ⇒
| |    paren(Vertices(the_graph);s3)
| |    ⇒
| |    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| | 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
| |    ¬(inl(i) ∈ s1)
| |    ⇒
| |    ¬(inr(i) ∈ s1)
| |    ⇒
| |    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| | 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| | 10. ∀T:Type, s:T, x:Vertices(the_graph)
| |     , f:(T→Vertices(the_graph)→T).
| |     ∃L:Vertices(the_graph) List.
| |     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
| |     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| | 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
| |     ∃L:Vertices(the_graph) List.
| |     no_repeats(Vertices(the_graph);L)
| |     & (∀y:Vertices(the_graph). (y ∈ L))
| |     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| | 12. s1 : traversal(the_graph)
| | 13. s2 : traversal(the_graph)
| | 14. i1 : Vertices(the_graph)
| | 15. ¬((inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1))
| | 16. P(i1,[inr(i1) / s1],s2)
| | 17. l_disjoint(Vertices(the_graph)+
| |     Vertices(the_graph);s2;[inr(i1) / s1])
```

```
| | 18. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| | 19. paren(Vertices(the_graph);s2)
| | ⊢  l_disjoint(Vertices(the_graph)+
| |    Vertices(the_graph);s2;[inr(i1)])
| |   by (All
| |       (RWO
| |        Thm* ∀a,b:T List, t:T.
| |             l_disjoint(T;b;[t / a])
| |             ⟺
| |             ¬(t ∈ b) & l_disjoint(T;b;a)
| |        {l_disjoint_cons2}))
| |      THEN
| |      ObviousConcl
| |
| |\
| | 1. the_graph : Graph
| | 2. the_obj : GraphObject(the_graph)
| | 3. P :
| |    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)
| |    →Prop
| | 4. traversal(the_graph)
| | 5. Vertices(the_graph)
| | 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
| |    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| | 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
| |    i-the_graph->j
| |    ⇒
| |    paren(Vertices(the_graph);s2)
| |    ⇒
| |    paren(Vertices(the_graph);s3)
| |    ⇒
| |    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| | 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
| |    ¬(inl(i) ∈ s1)
| |    ⇒
| |    ¬(inr(i) ∈ s1)
| |    ⇒
| |    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| | 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| | 10. ∀T:Type, s:T, x:Vertices(the_graph)
| |     , f:(T→Vertices(the_graph)→T).
| |     ∃L:Vertices(the_graph) List.
| |     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
| |     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| | 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
| |     ∃L:Vertices(the_graph) List.
| |     no_repeats(Vertices(the_graph);L)
| |     & (∀y:Vertices(the_graph). (y ∈ L))
| |     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| | 12. s1 : traversal(the_graph)
| | 13. s2 : traversal(the_graph)
| | 14. i1 : Vertices(the_graph)
| | 15. ¬((inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1))
| | 16. P(i1,[inr(i1) / s1],s2)
| | 17. l_disjoint(Vertices(the_graph)+
```

```
| |     Vertices(the_graph);s2;[inr(i1) / s1])
| | 18. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| | 19. paren(Vertices(the_graph);s2)
| | ⊢  no_repeats(Vertices(the_graph)+Vertices(the_graph);[inr(i1)])
| |   by ObviousConcl
| |
| .\
| . 1. the_graph : Graph
| . 2. the_obj : GraphObject(the_graph)
| . 3. P :
| . |  Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)
| . |  →Prop
| . 4. traversal(the_graph)
| . 5. Vertices(the_graph)
| . 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
| . |  (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| . 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
| . |  i-the_graph->j
| . |  ⇒
| . |  paren(Vertices(the_graph);s2)
| . |  ⇒
| . |  paren(Vertices(the_graph);s3)
| . |  ⇒
| . |  P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| . 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
| . |  ¬(inl(i) ∈ s1)
| . |  ⇒
| . |  ¬(inr(i) ∈ s1)
| . |  ⇒
| . |  P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| . 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| . 10. ∀T:Type, s:T, x:Vertices(the_graph)
| . |   , f:(T→Vertices(the_graph)→T).
| . |   ∃L:Vertices(the_graph) List.
| . |   (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
| . |   & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| . 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
| . |   ∃L:Vertices(the_graph) List.
| . |   no_repeats(Vertices(the_graph);L)
| . |   & (∀y:Vertices(the_graph). (y ∈ L))
| . |   & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| . 12. s1 : traversal(the_graph)
| . 13. s2 : traversal(the_graph)
| . 14. i1 : Vertices(the_graph)
| . 15. ¬((inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1))
| . 16. P(i1,[inr(i1) / s1],s2)
| . 17. l_disjoint(Vertices(the_graph)+
| . |   Vertices(the_graph);s2;[inr(i1) / s1])
| . 18. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| . 19. paren(Vertices(the_graph);s2)
| . ⊢  ¬((inl(i1) ∈ s2) ∨ inl(i1) = inr(i1))
| . | by (ParallelOp -5) THEN SplitOrHyps
| . |
| . |\
| . | 1. the_graph : Graph
```

```
| . | 2. the_obj : GraphObject(the_graph)
| . | 3. P :
| . |    Vertices(the_graph)→traversal(the_graph)→
| . |    traversal(the_graph)→Prop
| . | 4. traversal(the_graph)
| . | 5. Vertices(the_graph)
| . | 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
| . |    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| . | 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
| . |    i-the_graph->j
| . |    ⇒
| . |    paren(Vertices(the_graph);s2)
| . |    ⇒
| . |    paren(Vertices(the_graph);s3)
| . |    ⇒
| . |    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| . | 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
| . |    ¬(inl(i) ∈ s1)
| . |    ⇒
| . |    ¬(inr(i) ∈ s1)
| . |    ⇒
| . |    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| . | 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| . | 10. ∀T:Type, s:T, x:Vertices(the_graph)
| . |     , f:(T→Vertices(the_graph)→T).
| . |     ∃L:Vertices(the_graph) List.
| . |     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
| . |     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| . | 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
| . |     ∃L:Vertices(the_graph) List.
| . |     no_repeats(Vertices(the_graph);L)
| . |     & (∀y:Vertices(the_graph). (y ∈ L))
| . |     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| . | 12. s1 : traversal(the_graph)
| . | 13. s2 : traversal(the_graph)
| . | 14. i1 : Vertices(the_graph)
| . | 15. P(i1,[inr(i1) / s1],s2)
| . | 16. l_disjoint(Vertices(the_graph)+
| . |     Vertices(the_graph);s2;[inr(i1) / s1])
| . | 17. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| . | 18. paren(Vertices(the_graph);s2)
| . | 19. (inl(i1) ∈ s2)
| . | ⊢  (inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1)
| . |   by (FwdThru
| . |       Thm* ∀s':(T+T) List.
| . |            paren(T;s') ⇒ (∀i:T. (inl(i) ∈ s') ⇒ (inr(i) ∈ s'))
| . |       {paren_balance1}
| . |       [-1])
| . |      THEN
| . |      (RWO
| . |       Thm* ∀a,b:T List, t:T.
| . |            l_disjoint(T;b;[t / a])
| . |            ⟺
| . |            ¬(t ∈ b) & l_disjoint(T;b;a)
| . |       {l_disjoint_cons2}
```

```
| . |       -5)
| . |
| . .\
| . . 1. the_graph : Graph
| . . 2. the_obj : GraphObject(the_graph)
| . . 3. P :
| . .    Vertices(the_graph)→traversal(the_graph)→
| . .    traversal(the_graph)→Prop
| . . 4. traversal(the_graph)
| . . 5. Vertices(the_graph)
| . . 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
| . .    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
| . . 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
| . .    i-the_graph->j
| . .    ⇒
| . .    paren(Vertices(the_graph);s2)
| . .    ⇒
| . .    paren(Vertices(the_graph);s3)
| . .    ⇒
| . .    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
| . . 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
| . .    ¬(inl(i) ∈ s1)
| . .    ⇒
| . .    ¬(inr(i) ∈ s1)
| . .    ⇒
| . .    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
| . . 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
| . . 10. ∀T:Type, s:T, x:Vertices(the_graph)
| . .     , f:(T→Vertices(the_graph)→T).
| . .     ∃L:Vertices(the_graph) List.
| . .     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
| . .     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
| . . 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
| . .     ∃L:Vertices(the_graph) List.
| . .     no_repeats(Vertices(the_graph);L)
| . .     & (∀y:Vertices(the_graph). (y ∈ L))
| . .     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
| . . 12. s1 : traversal(the_graph)
| . . 13. s2 : traversal(the_graph)
| . . 14. i1 : Vertices(the_graph)
| . . 15. P(i1,[inr(i1) / s1],s2)
| . . 16. l_disjoint(Vertices(the_graph)+
| . .     Vertices(the_graph);s2;[inr(i1) / s1])
| . . 17. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
| . . 18. paren(Vertices(the_graph);s2)
| . . 19. inl(i1) = inr(i1)
| . . ⊢  (inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1)  by ObviousFrom [-1]
|
.\
. 1. the_graph : Graph
. 2. the_obj : GraphObject(the_graph)
. 3. P :
.    Vertices(the_graph)→traversal(the_graph)→traversal(the_graph)→
.    Prop
. 4. traversal(the_graph)
```

```
. 5. Vertices(the_graph)
. 6. ∀s:traversal(the_graph), i:Vertices(the_graph).
.    (inl(i) ∈ s) ∨ (inr(i) ∈ s) ⇒ P(i,s,nil)
. 7. ∀s1,s2,s3:traversal(the_graph), i,j:Vertices(the_graph).
.    i-the_graph->j
.    ⇒
.    paren(Vertices(the_graph);s2)
.    ⇒
.    paren(Vertices(the_graph);s3)
.    ⇒
.    P(j,s1,s2) ⇒ P(i,s2 @ s1,s3) ⇒ P(i,s1,s3 @ s2)
. 8. ∀s1,s2:traversal(the_graph), i:Vertices(the_graph).
.    ¬(inl(i) ∈ s1)
.    ⇒
.    ¬(inr(i) ∈ s1)
.    ⇒
.    P(i,[inr(i) / s1],s2) ⇒ P(i,s1,[inl(i) / (s2 @ [inr(i)])])
. 9. ∀x,y:Vertices(the_graph). the_obj.eq(x,y) ⟺ x = y
. 10. ∀T:Type, s:T, x:Vertices(the_graph)
.     , f:(T→Vertices(the_graph)→T).
.     ∃L:Vertices(the_graph) List.
.     (∀y:Vertices(the_graph). x-the_graph->y ⟺ (y ∈ L))
.     & the_obj.eacc(f,s,x) = list_accum(s',x'.f(s',x');s;L)
. 11. ∀T:Type, s:T, f:(T→Vertices(the_graph)→T).
.     ∃L:Vertices(the_graph) List.
.     no_repeats(Vertices(the_graph);L)
.     & (∀y:Vertices(the_graph). (y ∈ L))
.     & the_obj.vacc(f,s) = list_accum(s',x'.f(s',x');s;L)
. 12. s1 : traversal(the_graph)
. 13. s2 : traversal(the_graph)
. 14. i1 : Vertices(the_graph)
. 15. ¬((inl(i1) ∈ s1) ∨ (inr(i1) ∈ s1))
. 16. P(i1,[inr(i1) / s1],s2)
. 17. l_disjoint(Vertices(the_graph)+
.     Vertices(the_graph);s2;[inr(i1) / s1])
. 18. no_repeats(Vertices(the_graph)+Vertices(the_graph);s2)
. 19. paren(Vertices(the_graph);s2)
. ⊢  paren(Vertices(the_graph);[inl(i1) / (s2 @ [inr(i1)])])
.   by (((RecUnfold `paren` 0) THEN OrRight) THEN OrLeft)
.      THEN
.      (InstConcl [i1;s2])
.      THEN
.      (Reduce 0)
END
```