```
dfl-traversal-consr  THM

⊢  For any graph
|  ∀L:V List, i:V, s:Traversal.
|  (∀j:V. (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i)
|  ⇒
|  L-->*i
|  ⇒
|  dfl-traversal(the_graph;L;s)
|  ⇒
|  dfl-traversal(the_graph;L;[inr(i) / s])
| by Unfold `dfl-traversal` 0
|
|\
| 1. the_graph : Graph
| 2. L : Vertices(the_graph) List
| 3. i : Vertices(the_graph)
| 4. s : traversal(the_graph)
| 5. ∀j:Vertices(the_graph).
| |  (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
| 6. L-the_graph->*i
| 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| |  s = (s1 @ [inr(i)] @ s2)
| |  ⇒
| |  (∀j:Vertices(the_graph).
| |   (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
| 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
| 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| |  (∀j:Vertices(the_graph).
| |   i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
| |  ⇒
| |  s = (s1 @ [inl(i)] @ s2) ⇒ L-the_graph->*i
| 10. i@0 : Vertices(the_graph)
| 11. s1 : traversal(the_graph)
| 12. s2 : traversal(the_graph)
| 13. [inr(i) / s] = (s1 @ [inr(i@0)] @ s2)
| 14. j : Vertices(the_graph)
| 15. (inr(j) ∈ s2)
| 16. ¬(inl(j) ∈ s2)
| ⊢  j-the_graph->*i@0  by (Analyze -6) THEN (All Reduce)
| |
| |\
| | 1. the_graph : Graph
| | 2. L : Vertices(the_graph) List
| | 3. i : Vertices(the_graph)
| | 4. s : traversal(the_graph)
| | 5. ∀j:Vertices(the_graph).
| | |  (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
| | 6. L-the_graph->*i
| | 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| | |  s = (s1 @ [inr(i) / s2])
| | |  ⇒
| | |  (∀j:Vertices(the_graph).
| | |   (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
| | 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
```

```
| | 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| | |  (∀j:Vertices(the_graph).
| | |   i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
| | |  ⇒
| | |  s = (s1 @ [inl(i) / s2]) ⇒ L-the_graph->*i
| | 10. i@0 : Vertices(the_graph)
| | 11. s2 : traversal(the_graph)
| | 12. [inr(i) / s] = [inr(i@0) / s2]
| | 13. j : Vertices(the_graph)
| | 14. (inr(j) ∈ s2)
| | 15. ¬(inl(j) ∈ s2)
| | ⊢  j-the_graph->*i@0
| | | by (SplitCons -4) THEN (Subst (i@0 = i) 0)
| | |    THEN
| | |    (Try (BackThruSomeHyp THEN (HypSubst -1 0)))
| | |
| | .....equality .....
| | 1. the_graph : Graph
| | 2. L : Vertices(the_graph) List
| | 3. i : Vertices(the_graph)
| | 4. s : traversal(the_graph)
| | 5. ∀j:Vertices(the_graph).
| |    (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
| | 6. L-the_graph->*i
| | 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| |    s = (s1 @ [inr(i) / s2])
| |    ⇒
| |    (∀j:Vertices(the_graph).
| |     (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
| | 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
| | 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| |    (∀j:Vertices(the_graph).
| |     i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
| |    ⇒
| |    s = (s1 @ [inl(i) / s2]) ⇒ L-the_graph->*i
| | 10. i@0 : Vertices(the_graph)
| | 11. s2 : traversal(the_graph)
| | 12. j : Vertices(the_graph)
| | 13. (inr(j) ∈ s2)
| | 14. ¬(inl(j) ∈ s2)
| | 15. inr(i) = inr(i@0)
| | 16. s = s2
| | ⊢  i@0 = i  by ObviousFrom [-2]
| |
| .\
| . 1. the_graph : Graph
| . 2. L : Vertices(the_graph) List
| . 3. i : Vertices(the_graph)
| . 4. s : traversal(the_graph)
| . 5. ∀j:Vertices(the_graph).
| . |  (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
| . 6. L-the_graph->*i
| . 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| . |  s = (s1 @ [inr(i) / s2])
| . |  ⇒
```

```
| . |  (∀j:Vertices(the_graph).
| . |   (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
| . 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
| . 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| . |  (∀j:Vertices(the_graph).
| . |   i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
| . |  ⇒
| . |  s = (s1 @ [inl(i) / s2]) ⇒ L-the_graph->*i
| . 10. i@0 : Vertices(the_graph)
| . 11. u : Vertices(the_graph)+Vertices(the_graph)
| . 12. v : (Vertices(the_graph)+Vertices(the_graph)) List
| . 13. s2 : traversal(the_graph)
| . 14. [inr(i) / s] = [u / (v @ [inr(i@0) / s2])]
| . 15. j : Vertices(the_graph)
| . 16. (inr(j) ∈ s2)
| . 17. ¬(inl(j) ∈ s2)
| . ⊢  j-the_graph->*i@0  by InstHyp [i@0;v;s2;j] 7
| . |
| . .....antecedent .....
| . 1. the_graph : Graph
| . 2. L : Vertices(the_graph) List
| . 3. i : Vertices(the_graph)
| . 4. s : traversal(the_graph)
| . 5. ∀j:Vertices(the_graph).
| .    (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
| . 6. L-the_graph->*i
| . 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| .    s = (s1 @ [inr(i) / s2])
| .    ⇒
| .    (∀j:Vertices(the_graph).
| .     (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
| . 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
| . 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| .    (∀j:Vertices(the_graph).
| .     i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
| .    ⇒
| .    s = (s1 @ [inl(i) / s2]) ⇒ L-the_graph->*i
| . 10. i@0 : Vertices(the_graph)
| . 11. u : Vertices(the_graph)+Vertices(the_graph)
| . 12. v : (Vertices(the_graph)+Vertices(the_graph)) List
| . 13. s2 : traversal(the_graph)
| . 14. [inr(i) / s] = [u / (v @ [inr(i@0) / s2])]
| . 15. j : Vertices(the_graph)
| . 16. (inr(j) ∈ s2)
| . 17. ¬(inl(j) ∈ s2)
| . ⊢  s = (v @ [inr(i@0) / s2])  by SplitCons -4
|
|\
| 1. the_graph : Graph
| 2. L : Vertices(the_graph) List
| 3. i : Vertices(the_graph)
| 4. s : traversal(the_graph)
| 5. ∀j:Vertices(the_graph).
| |  (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
| 6. L-the_graph->*i
```

```
| 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| |  s = (s1 @ [inr(i)] @ s2)
| |  ⇒
| |  (∀j:Vertices(the_graph).
| |   (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
| 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
| 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
| |  (∀j:Vertices(the_graph).
| |   i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
| |  ⇒
| |  s = (s1 @ [inl(i)] @ s2) ⇒ L-the_graph->*i
| 10. j : Vertices(the_graph)
| 11. (inr(j) ∈ [inr(i) / s])
| ⊢  L-the_graph->*j
| | by (RWO
| |     Thm* ∀l:T List, a,x:T. (x ∈ [a / l]) ⟺ x = a ∨ (x ∈ l)
| |     {cons_member}
| |     -1)
| |    THEN
| |    (Analyze -1)
| |    THEN
| |    EasyHyp
| |    THEN
| |    (Subst (j = i) 0)
| |
| .....equality .....
| 1. the_graph : Graph
| 2. L : Vertices(the_graph) List
| 3. i : Vertices(the_graph)
| 4. s : traversal(the_graph)
| 5. ∀j:Vertices(the_graph).
|    (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
| 6. L-the_graph->*i
| 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
|    s = (s1 @ [inr(i)] @ s2)
|    ⇒
|    (∀j:Vertices(the_graph).
|     (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
| 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
| 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
|    (∀j:Vertices(the_graph).
|     i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
|    ⇒
|    s = (s1 @ [inl(i)] @ s2) ⇒ L-the_graph->*i
| 10. j : Vertices(the_graph)
| 11. inr(j) = inr(i)
| ⊢  j = i  by ObviousFrom [-1]
|
.\
. 1. the_graph : Graph
. 2. L : Vertices(the_graph) List
. 3. i : Vertices(the_graph)
. 4. s : traversal(the_graph)
. 5. ∀j:Vertices(the_graph).
. |  (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
```

```
. 6. L-the_graph->*i
. 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
. |  s = (s1 @ [inr(i)] @ s2)
. |  ⇒
. |  (∀j:Vertices(the_graph).
. |   (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
. 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
. 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
. |  (∀j:Vertices(the_graph).
. |   i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
. |  ⇒
. |  s = (s1 @ [inl(i)] @ s2) ⇒ L-the_graph->*i
. 10. i@0 : Vertices(the_graph)
. 11. s1 : traversal(the_graph)
. 12. s2 : traversal(the_graph)
. 13. ∀j:Vertices(the_graph).
. |   i@0-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j)
. 14. [inr(i) / s] = (s1 @ [inl(i@0)] @ s2)
. ⊢  L-the_graph->*i@0
. | by (Analyze -4) THEN (All Reduce) THEN (SplitCons -1)
. |
. |\
. | 1. the_graph : Graph
. | 2. L : Vertices(the_graph) List
. | 3. i : Vertices(the_graph)
. | 4. s : traversal(the_graph)
. | 5. ∀j:Vertices(the_graph).
. |    (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
. | 6. L-the_graph->*i
. | 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
. |    s = (s1 @ [inr(i) / s2])
. |    ⇒
. |    (∀j:Vertices(the_graph).
. |     (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
. | 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
. | 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
. |    (∀j:Vertices(the_graph).
. |     i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
. |    ⇒
. |    s = (s1 @ [inl(i) / s2]) ⇒ L-the_graph->*i
. | 10. i@0 : Vertices(the_graph)
. | 11. s2 : traversal(the_graph)
. | 12. ∀j:Vertices(the_graph).
. |     i@0-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j)
. | 13. inr(i) = inl(i@0)
. | 14. s = s2
. | ⊢  L-the_graph->*i@0  by ObviousContradiction [-2]
. |
. .\
. . 1. the_graph : Graph
. . 2. L : Vertices(the_graph) List
. . 3. i : Vertices(the_graph)
. . 4. s : traversal(the_graph)
. . 5. ∀j:Vertices(the_graph).
. .    (inr(j) ∈ s) ⇒ ¬(inl(j) ∈ s) ⇒ j-the_graph->*i
```

```
. . 6. L-the_graph->*i
. . 7. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
. .    s = (s1 @ [inr(i) / s2])
. .    ⇒
. .    (∀j:Vertices(the_graph).
. .     (inr(j) ∈ s2) ⇒ ¬(inl(j) ∈ s2) ⇒ j-the_graph->*i)
. . 8. ∀j:Vertices(the_graph). (inr(j) ∈ s) ⇒ L-the_graph->*j
. . 9. ∀i:Vertices(the_graph), s1,s2:traversal(the_graph).
. .    (∀j:Vertices(the_graph).
. .     i-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j))
. .    ⇒
. .    s = (s1 @ [inl(i) / s2]) ⇒ L-the_graph->*i
. . 10. i@0 : Vertices(the_graph)
. . 11. u : Vertices(the_graph)+Vertices(the_graph)
. . 12. v : (Vertices(the_graph)+Vertices(the_graph)) List
. . 13. s2 : traversal(the_graph)
. . 14. ∀j:Vertices(the_graph).
. .     i@0-the_graph->*j ⇒ ¬non-trivial-loop(the_graph;j)
. . 15. inr(i) = u
. . 16. s = (v @ [inl(i@0) / s2])
. . ⊢  L-the_graph->*i@0  by AutoInst [i@0;v;s2]
END
```