```
append_paren  THM

⊢  ∀T:Type, s',s3:(T+T) List.
|  paren(T;s') ⇒ paren(T;s3) ⇒ paren(T;s3 @ s')
| by Auto THEN (Analyze 2)
|     THEN
|     (Try
|      (RWO Thm* ∀as:T List. (as @ nil) = as    {append_back_nil} 0))
|     THEN
|     (Analyze 4)
|     THEN
|     (Try ((Reduce 0) THEN Trivial))
|
1. T : Type
2. u : T+T
3. v : (T+T) List
4. u1 : T+T
5. v1 : (T+T) List
6. paren(T;[u / v])
7. paren(T;[u1 / v1])
⊢  paren(T;[u1 / v1] @ [u / v])
  by (((RecUnfold `paren` 0) THEN OrRight) THEN OrRight)
     THEN
     (InstConcl [[u1 / v1];[u / v]])
     THEN
     (Reduce 0)
END
```