Algorithm 10.1 Upward pass of variable elimination in clique tree

```
Procedure CTree-SP-Upward (
    Φ,    // Set of factors
    T,    // Clique tree over Φ
    α,    // Initial assignment of factors to cliques
    C_r   // Some selected root clique
)
1   Initialize-Cliques
2   while C_r is not ready
3     Let C_i be a ready clique
4     δ_{i→p_r(i)}(S_{i,p_r(i)}) ← SP-Message(i, p_r(i))
5   β_r ← ψ_r · ∏_{k∈Nb_{C_r}} δ_{k→r}
6   return β_r

Procedure Initialize-Cliques (

)
1   for each clique C_i
2     ψ_i(C_i) ← ∏_{φ_j : α(φ_j)=i} φ_j
3

Procedure SP-Message (
    i,    // sending clique
    j     // receiving clique
)
1   ψ(C_i) ← ψ_i · ∏_{k∈(Nb_i−{j})} δ_{k→i}
2   τ(S_{i,j}) ← ∑_{C_i−S_{i,j}} ψ(C_i)
3   return τ(S_{i,j})
```

10.2.1.3 Correctness

We now prove that this algorithm, when applied to a clique tree that satisfies the family preservation and running intersection property, computes the desired expressions over the messages and the cliques.

In our algorithm, a variable X is eliminated only when a message is sent from $\boldsymbol{C}_i$ to a neighboring $\boldsymbol{C}_j$ such that $X \in \boldsymbol{C}_i$ and $X \notin \boldsymbol{C}_j$. We first prove the following result:

Proposition 10.2 *Assume that X is eliminated when a message is sent from $\boldsymbol{C}_i$ to $\boldsymbol{C}_j$. Then X does not appear anywhere in the tree on the $\boldsymbol{C}_j$ side of the edge $(i$–$j)$.*

Proof The proof is a simple consequence of the running intersection property. Assume by contradiction that X appears in some other clique $\boldsymbol{C}_k$ that is on the $\boldsymbol{C}_j$ side of the tree. Then $\boldsymbol{C}_j$ is on the path from $\boldsymbol{C}_i$ to $\boldsymbol{C}_k$. But we know that X appears in both $\boldsymbol{C}_i$ and $\boldsymbol{C}_k$ but not in $\boldsymbol{C}_j$, violating the running intersection property. ∎