Algorithm 13.2 Max-product message computation for MAP

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

13.3 Max-Product in Clique Trees

We now extend the ideas used in the MAP variable elimination algorithm to the case of clique trees. As for the case of sum-product, the benefit of the clique tree algorithm is that it uses dynamic programming to compute an entire set of marginals simultaneously. For sum-product, we used clique trees to compute the *sum-marginals* over each of the cliques in our tree. Here, we compute a set of *max-marginals* over each of those cliques.

At this point, one might ask why we want to compute an entire set of max-marginals simultaneously. After all, if our only task is to compute a single MAP assignment, the variable elimination algorithm provides us with a method for doing so. There are two reasons for considering this extension.

First, a set of max-marginals can be a useful indicator for how confident we are in particular components of the MAP assignment. Assume, for example, that our variables are binary-valued, and that the max-marginal for X_1 has $\mathit{MaxMarg}(x_1^1) = 3$ and $\mathit{MaxMarg}(x_1^0) = 2.95$, whereas the max-marginal for X_2 has $\mathit{MaxMarg}(x_2^1) = 3$ and $\mathit{MaxMarg}(x_2^0) = 1$. In this case, we know that there is an alternative joint assignment whose probability is very close to the optimum, in which X_1 takes a different value; by contrast, the best alternative assignment in which X_2 takes a different value has a much lower probability. Note that, without knowing the partition function, we cannot determine the actual magnitude of these differences in terms of probability. But we can determine the relative difference between the change in X_1 and the change in X_2.

pseudo-max-marginal

Second, in many cases, an exact solution to the MAP problem via a variable elimination procedure is intractable. In this case, we can use message passing procedures in cluster graphs, similar to the clique tree procedure, to compute *approximate* max-marginals. These *pseudo-max-marginals* can be used for selecting an assignment; while this assignment is not generally the MAP assignment, we can nevertheless provide some guarantees in certain cases. As before, our task has two parts: computing the max-marginals and decoding them to extract a MAP assignment. We describe each of those steps in turn.

13.3.1 Computing Max-Marginals

max-product belief propagation

In the same way that we used dynamic programming to modify the sum-product variable elimination algorithm to the case of clique trees, we can also modify the max-product algorithm to define a *max-product belief propagation* algorithm in clique trees. The resulting algorithm executes precisely the same initialization and overall message scheduling as in the sum-product