```
function LinearPartition(x)
    n ← length of x
    C ← hash()                                                    ▷ chart
    C(0, 1) ← 1                                                   ▷ axiom
    for j = 1, ...n do
        C(0, j + 1) ← C(0, j) · e^{−δ(x,j)/kT}
        C(j, j + 1) ← 1                                           ▷ action PUSH
        for all (i, j) ∈ C do
            C(i, j + 1) ← C(i, j) · e^{−δ(x,j)/kT}                ▷ action SKIP
            if (x_i, x_j) ∈{AU, UA, CG, GC, GU, UG} then
                Q_{i,j+1} ← C(i, j) · e^{−ξ(x,i,j)/kT}
                for all (k, i) ∈ C do
                    C(k, j + 1) += C(k, i) · Q_{i,j+1}            ▷ action POP
                end for
                C(0, j + 1) += C(0, i) · Q_{i,j+1}                ▷ action COMBINE
            end if
        end for
        BeamPrune(C, j + 1,beamsize)
    end for
    return C(0, n + 1)
end function

function BeamPrune(C, j, b)
    cands ← hash()                                                ▷ candidates
    for all (i, j) ∈ C do
        cands(i) ← C(0, i) · C(i, j)                              ▷ C(0, i) as prefix score
    end for
    cands ← QuickSelectTopB(cands, b)                             ▷ use quick select algorithm to select top-b by score
    for all (i, j) ∈ C do
        if i ∉ cands then
            delete (i, j) in C                                    ▷ prune out low-scoring states
        end if
    end for
end function
```