Braiden Wells

1. RNA folding with a limited number of multi branch loops [35 pts].

I am assuming the term “multi branch loops” describes what is referred to in the slides as a “multiloop”, as I couldn’t find this specific term in the slides.

For this recursion, I determined that an i,j base pair followed by pairings between i,g and g+1,j would result in a multiloop. This results in at least three pairings connecting three separate structures. In order to represent this in the recursion, there are two variables. One is K, the current number of multiloops that have been found. The other is a boolean variable that is true any time a pairing is found, and becomes false any time K gets incremented, so that the next pairing can be found. K is incremented whenever a pair has been found (Boolean = true) and a split is made. Once K has reached the maximum number of allowed multiloops (k), no more multiloops are allowed by the equation. At this point, the recursion is given by the recursion show in Sf,K=k(i,j), where splits are no longer allowed. More splits would result in more multiloops at this point, and they are forbidden by the fact that k multiloops already exist.

SF,K=0(i,j) = max { St, K=0(i + 1, j – 1) + 1 [if i, j base pair]

S(i + 1, j)

S(i, j – 1)

Maxi < g < j S(i, g) + S(g + 1, j)

}

St,K=0(i,j) = max { St, K=0(i + 1, j – 1) + 1 [if i, j base pair]

S(i + 1, j)

S(i, j – 1)

Maxi < g < j Sf,K=1(i, g) + Sf,K=1(g + 1, j)

}

Sf,K=1(i,j) = max { St, K=1(i + 1, j – 1) + 1 [if i, j base pair]

S(i + 1, j)

S(i, j – 1)

Maxi < g < j S(i, g) + S(g + 1, j)

}

St,K=1(i,j) = max { St, K=1(i + 1, j – 1) + 1 [if i, j base pair]

S(i + 1, j)

S(i, j – 1)

Maxi < g < j Sf,K=2(i, g) + Sf,K=2(g + 1, j)

}

…

St,K=k-1(i,j) = max { St, K=k-1(i + 1, j – 1) + 1 [if i, j base pair]

S(i + 1, j)

S(i, j – 1)

Maxi < g < j Sf,K=k(i, g) + Sf,K=k(g + 1, j)

}

Sf,K=k(i,j) = max { Sf, K=k(i + 1, j – 1) + 1 [if i, j base pair]

S(i + 1, j)

S(i, j – 1)

}

2. Predicting pseudoknots is hard or maybe not? [35 pts].

S(i,j) = max { S(i + 1, j – 1) + 1 [if i, j base pair]

S(i + 1, j)

S(i, j – 1)

Maxi < k < j S(i, k) + S(k + 1, j)

Maxi < g < k < j S(i, k) + S(g, j)

}

The algorithm I have given is essentially the same as the original algorithm, with the key difference that the recursion has an extra case allowing pseudoknots. In this case, two recursive cases are chosen between that include overlapping base pairs, where the overlapping pairs cover the entire range from i to j. This means that the recursion will have to check (i-j)^4 possible cases, because for every k from i+2 to j - 1, g can cover every nucleotide from i+1 to k-1, and for every g from i + 1 to j – 2, k can cover every nucleotide from j-1 to g+1. This will mean that (i-j)^4 checks will be run, which, while large, is still in polynomial time.

3. Hands-on experience [30 pts]. Download and compile UNAFold and VARNA from http://mfold.rna.albany.edu/?q=DINAMelt/ software and http://varna.lri.fr respectively. Download OxyS and fhlA sequences from the course website and run UNAFold with the default parameters to predict their minimum free energy structures. Visualize the predicted structures using VARNA, and report the minimum free energies and structures. Does the predicted structure look like what we saw in class?

I could not get the unafold application to run.