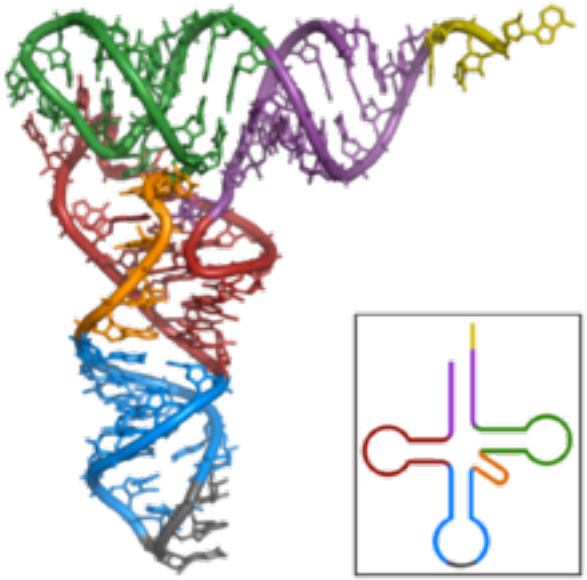
RNA secondary structure prediction

Farhat Habib

RNA

- RNA is similar to DNA chemically. It is usually only a single strand. T(hyamine) is replaced by U(racil)
- Some forms of RNA can form secondary structures by "pairing up" with itself. This can change its properties dramatically.

DNA and RNA
can pair with each other.



tRNA linear and 3D view: http://en.wikipedia.org/wiki/Transfer_RNA

Secondary structure

- Several interesting RNAs have a conserved secondary structure (resulting from base-pairing interactions)
- Sometimes, the sequence itself may not be conserved for the function to be retained
- It is important to tell what the secondary structure is going to be, for homology detection

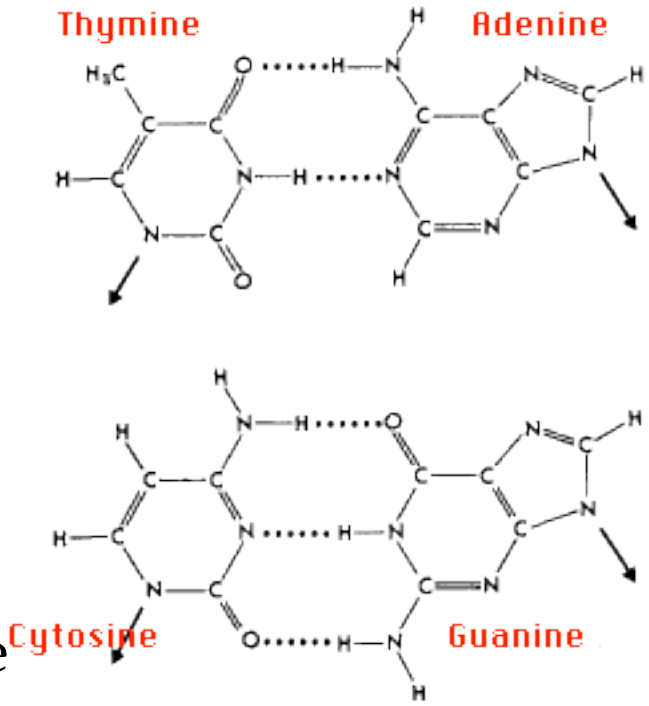
Basics of secondary structure

G-C pairing: three bonds (strong)

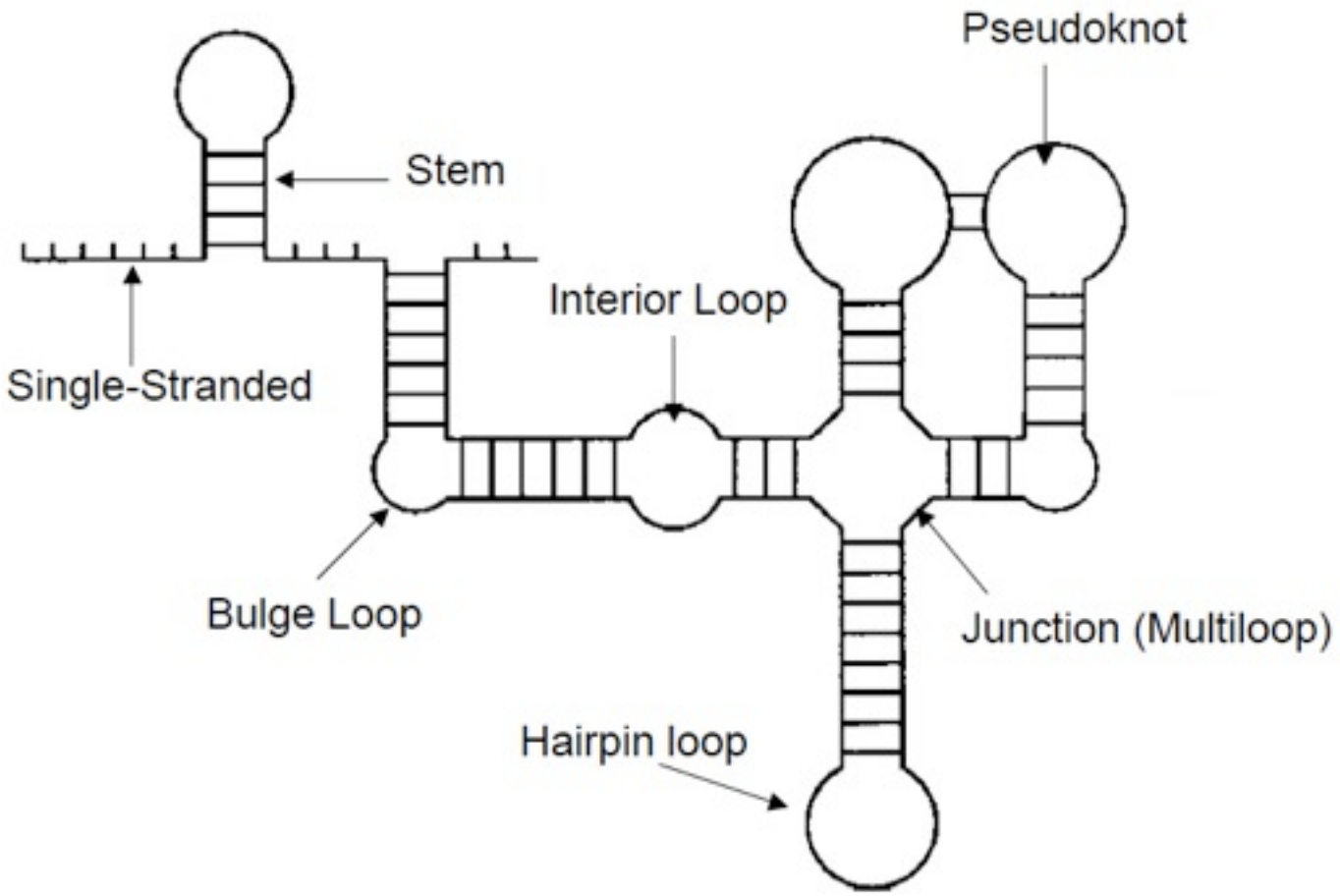
A-U pairing: two bonds (weaker)

Base pairs are approximately coplanar

Base pairs are stacked onto other base pairs (arranged side by side): “stems”



Secondary structure elements



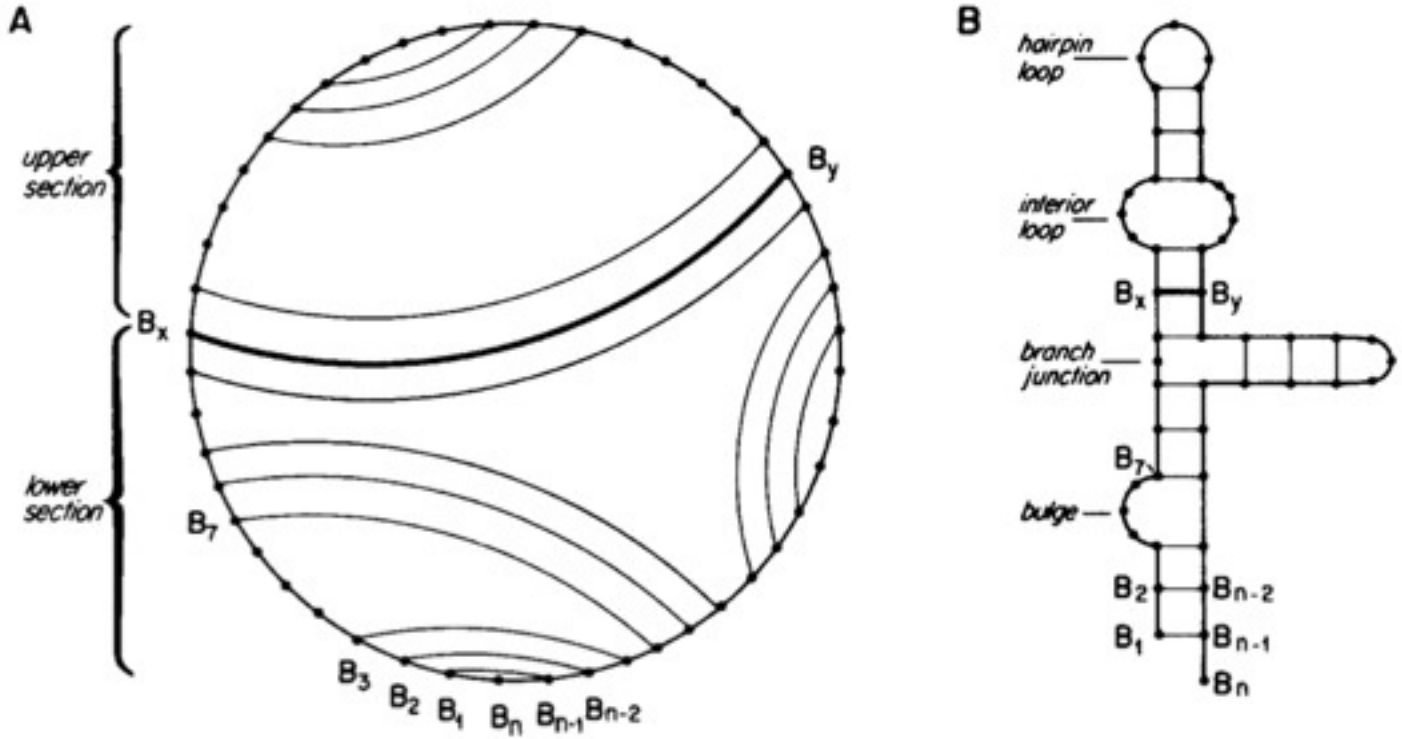
Non-canonical base pairs

- G-C and A-U are the canonical base pairs
- G-U is also possible, almost as stable

Nesting

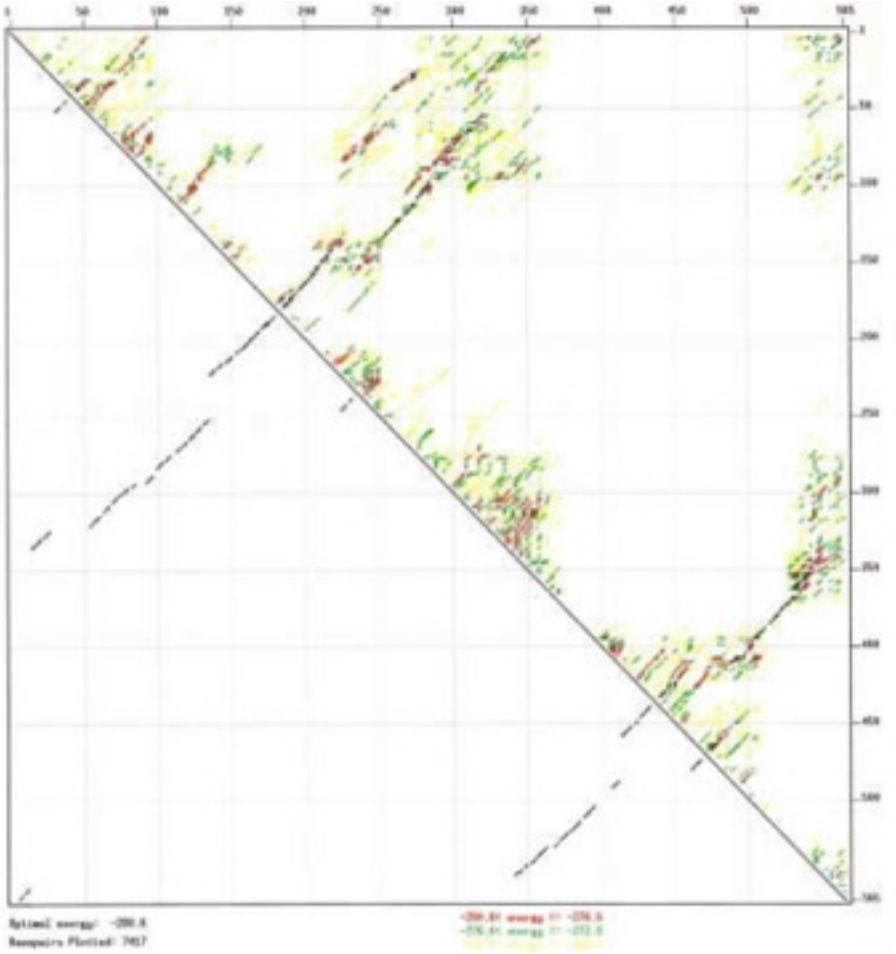
- Base pairs almost always occur in a nested fashion
- If positions i and j are paired, and positions i' and j' are paired, then these two base-pairings are said to be nested if:
  - $i < i' < j' < j$ OR $i' < i < j < j'$
- Non-nested base pairing: pseudoknot

Circle Plot


A
upper section
lower section
B_x
B_y
B_7
B_3 B_2 B_1 B_n B_{n-1} B_{n-2}
B
hairpin loop
interior loop
B_x B_y
branch junction
B_7
bulge
B_2 B_{n-2}
B_1 B_{n-1}
B_n

Dot Plot

List RNA sequence across the top.
List RNA sequence on the vertical side.
Score G-C, A-U, and G-U pairs
Long runs indicate hairpins and other structures with basepairing



Pseudoknots

- Pseudoknots are not handled by the algorithms we shall see
- Pseudoknots do occur in many important RNAs
- But the total *number* of pseudoknotted base pairs is typically relatively small

RNA Combinatorics

- The number of RNA secondary structures for the sequence *[1,n]*

$$s(0) = s(1) = s(2) = 1$$

$$s(n+1) = s(n) + \sum_{j=1}^{n-1} s(j-1)s(n-j), (n \geq 2)$$

Derived by Waterman

- Approx 1.3 b structures for n = 27

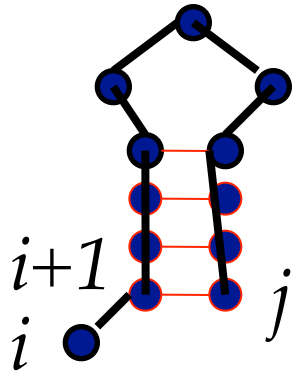
Nussinov's Algorithm

- Find the secondary structure with most base pairs.
- Recursive: finds best structure for small subsequences, and works its way outwards to larger subsequences

Nussinov's algorithm: idea

- There are only four possible ways of getting the best structure for subsequence (i,j) from the best structures of the smaller subsequences

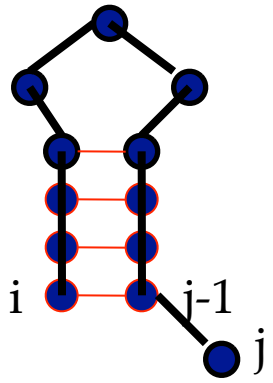
(1) Add unpaired position i onto best structure for subsequence (i+1,j)



Nussinov's algorithm: idea

- There are only four possible ways of getting the best structure for subsequence (i,j) from the best structures of the smaller subsequences

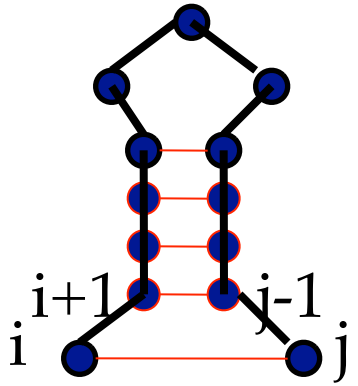
(2) Add unpaired position j onto best structure for subsequence (i,j-1)



Nussinov's algorithm: idea

- There are only four possible ways of getting the best structure for subsequence (i,j) from the best structures of the smaller subsequences

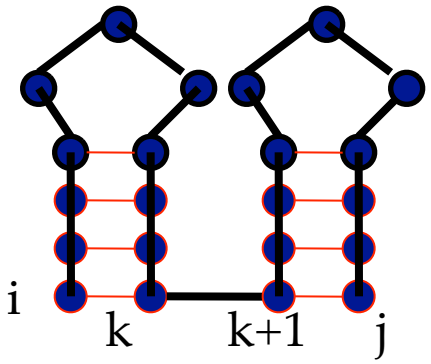
(3) Add (i,j) pair onto best structure for subsequence (i+1,j-1)



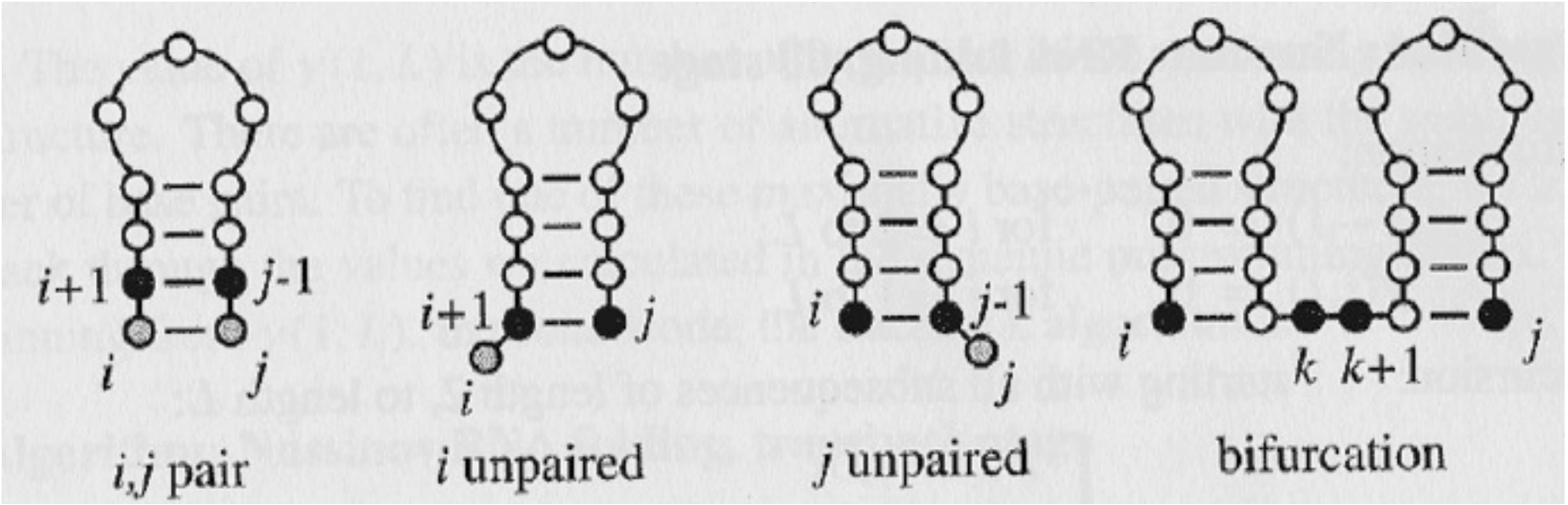
Nussinov's algorithm: idea

- There are only four possible ways of getting the best structure for subsequence (i, j) from the best structures of the smaller subsequences

(4) Combine two optimal substructures (i, k) and (k+1, j)



Summarizing



Nussinov RNA folding algorithm

- Given a sequence s of length L with symbols $s_1 \ldots s_L$. Let $\delta(i,j) = 1$ if s_i and s_j are a complementary base pair, and 0 otherwise.

- We recursively calculate scores $g(i,j)$ which are the maximal number of base pairs that can be formed for subsequence $s_i \ldots s_j$.

Recursion

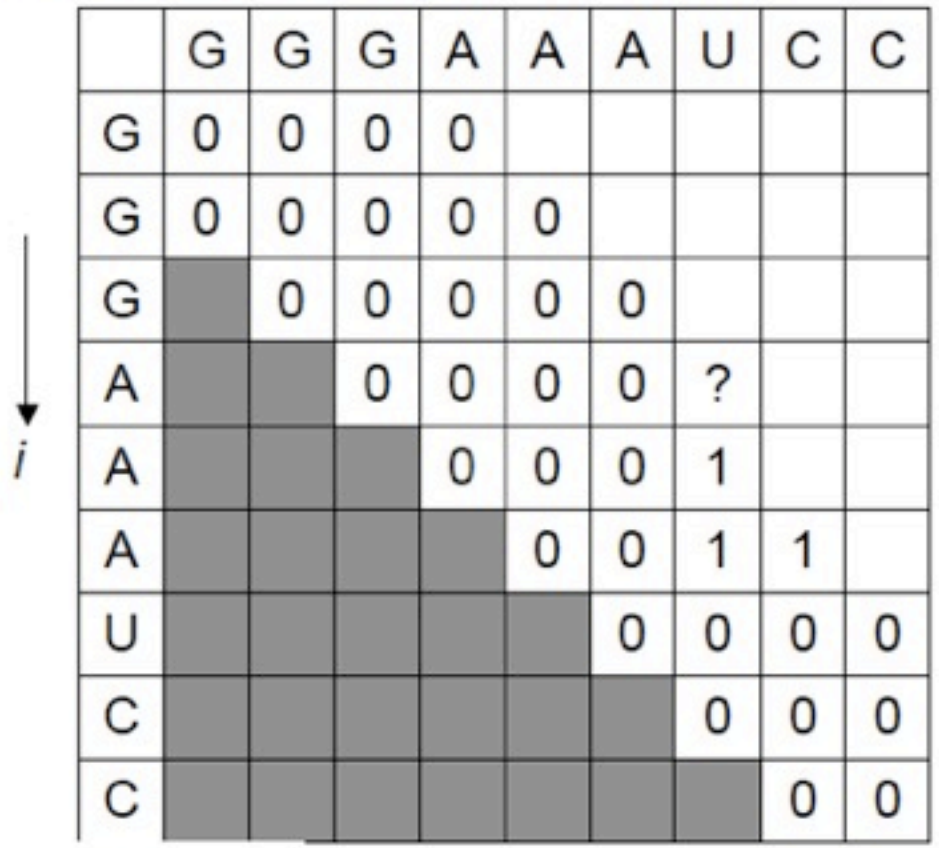
- Starting with all subsequences of length 2, to length L

$$S(i,j) = max \begin{cases} S(i+1, j-1) + w(i,j) \\ S(i+1, j) \\ S(i, j-1) \\ max_{i<k<j} S(i,k) + S(k+1, j) \end{cases}$$

- Initialization
  - $S(i,i\text{-}1) = 0$
  - $S(i,i) = 0$

Recursion

Fill up the table (DP matrix) -- diagonal by diagonal



$$S(i,j) = max \begin{cases} S(i+1, j-1) + w(i,j) & (1) \\ S(i+1, j) & (2) \\ S(i, j-1) & (3) \\ max_{i<k<j} S(i,k) + S(k+1, j) & (4) \end{cases}$$

$$w(i,j) = \begin{cases} 1 & i, j \text{ are complementary} \\ 0 & otherwise \end{cases}$$

Traceback

- As usual in sequence alignment ?
- Optimal sequence alignment is a linear path in the dynamic programming table
- Optimal secondary structure can have “bifurcations”

Traceback

Input: Matrix S and positions i, j.
Output: Secondary structure maximizing the number of base pairs.
Initial call: traceback(i = 1, j = L).

```
if i < j then
   if S(i, j) = S(i + 1, j) then                          // case (1)
      traceback(i + 1, j)
   else if S(i, j) = S(i, j – 1) then                     // case (2)
      traceback(i, j – 1)
   else if S(i, j) = S(i + 1, j – 1) + w(i, j) then       // case (3)
      print base pair (i, j)
      traceback(i + 1, j – 1)
   else for k = i + 1 to j – 1 do                         // case (4)
      if S(i, j) = S(i, k) + S(k + 1, j) then
         traceback(i, k)
         traceback(k + 1, j)
         break
end
```

Secondary structure prediction Zuker's algorithm

- Based on minimization of ΔG, the equilibrium free energy, rather than maximization of number of base pairs
- Better fit to real (experimental) ΔG
- Energy of stem is sum of "stacking" contributions from the interface between neighboring base pairs

- One way is to assign an energy to each base pair in a secondary structure. That is, there is a function *e* such that $e(r_i, r_j)$ is the energy of a base pair. The energy, *E(S)*, of the entire structure, is then given by:

$$E(S) = \sum_{i,j \in S} e(r_i, r_j)$$

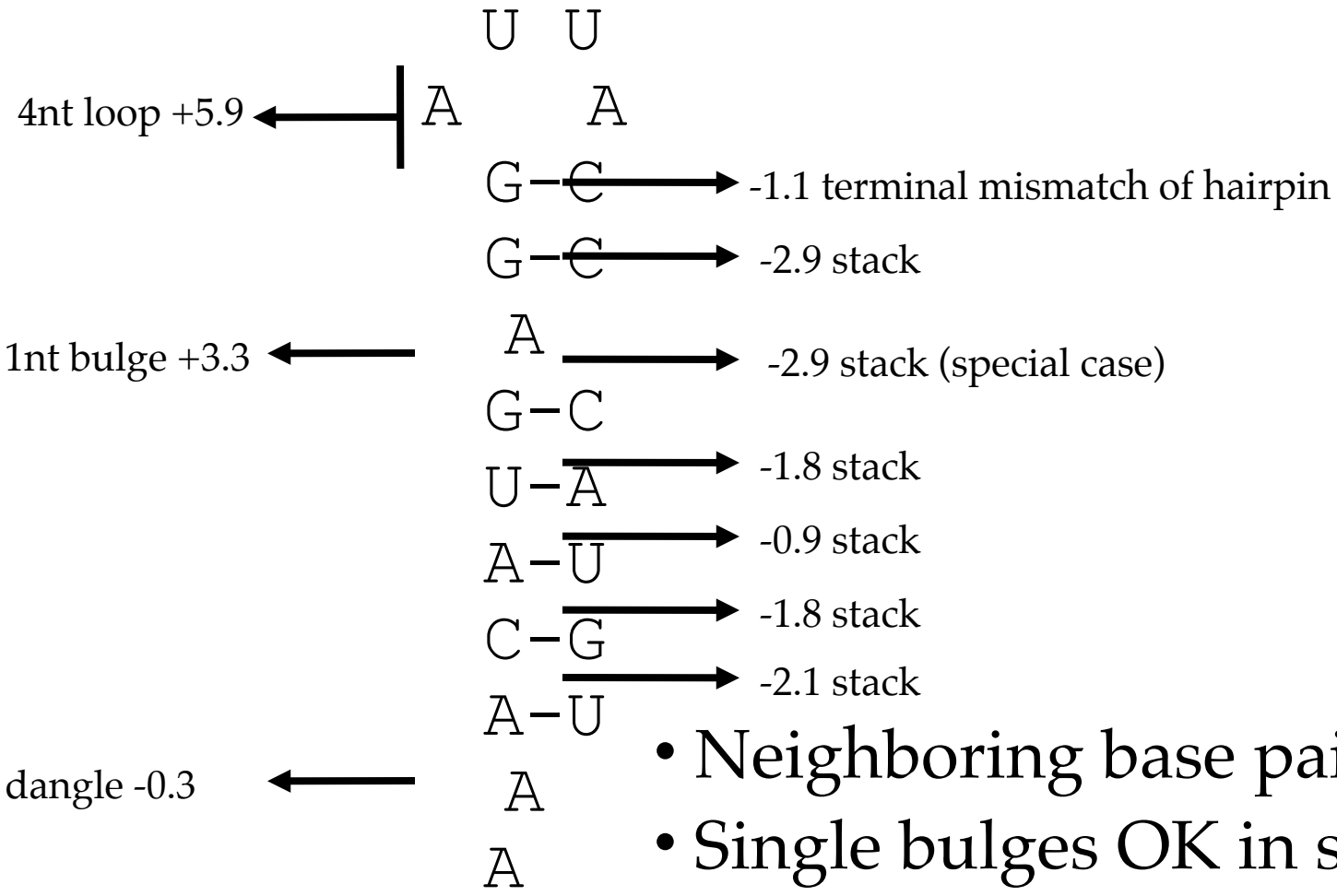
- Reasonable values of e at are -3, -2 and -1 kcal/mole for GC, AU and GU base pairs, respectively.

Base pair dependent energy rules

- Let $W=min(W_s)$, where s ranges over all secondary structures
- Energy for pairing r_i with r_j is given by $e(r_i, r_j)$
- Compute W_{ij} matrix as
-

$$W_{ij} = 0 \text{ for } j - i < 4$$

$$W_{ij} = min \begin{cases} W_{i+1,j} \\ W_{i,j-1} \\ W_{i+1,j-1} + e(i,j) \\ min_{k=i+1}^{j-1} W_{i,k} + W_{k+1,j} \end{cases}$$



- Neighboring base pairs: stack
- Single bulges OK in stacking
- Longer bulges: no stacking term
- Loop destabilisation energy
- Loop terminal mismatch energy

Energy contributions

- *eS(i,j)*: Free energy of stacked pair *(i,j)* and *(i+1,j-1)*
- *eH(i,j)*: Free energy of a loop closed by *(i,j)*: depends on length of loop, bases at *i,j*, and bases adjacent to them
- *eL(i, j, i′, j′)*: Free energy of an internal loop or bulge, with *(i, j)* and *(i′, j′)* being the bordering base pairs. Depends on bases at these positions, and adjacent unpaired bases
- *eM(i,j,i_1,j_1,…i_k,j_k)*: Free energy of a multibranch loop with *(i,j)* as the closing base pair and i_1j_1 etc as the internal base pairs

Zuker's algorithm

W(j): FE of optimal structure of *s(1..j)*

V(i,j): FE of optimal structure of *s(i..j)* assuming *i, j* form a base pair

VBI(i,j): FE of optimal structure of *s(i..j)* assuming *i, j* closes a bulge or internal loop

VM(i,j): FE of optimal structure of *s(i..j)* assuming *i, j* closes a multibranch loop

WM(i,j): used to compute *VM*

Dynamic programming recurrences

- *W(j)*: FE of optimal structure of *s(1..j)*
  - *W(0)* = 0
  - *W(j)* = min(*W(j-1)*,

s[j] is external base
(a base not in any loop)

$\min_{1 \le i < j} V(i,j) + W(i-1))$

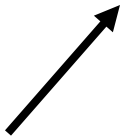
s(j) pairs with s(i),
for some i < j

Dynamic programming recurrences

$V(i,j)$: FE of optimal structure of $s(i..j)$ assuming i, j form a base pair

$V(i,j)$ = infinity if $i >= j$

$V(i,j)$ = min($eH(i,j)$,



$eS(i,j) + V(i+1,j-1)$,

$VBI(i,j)$,

$VM(i,j))$ if $i < j$

i,j is exterior base pair of a hairpin loop

Dynamic programming recurrences

V(i,j): FE of optimal structure of *s(i..j)* assuming *i, j* form a base pair

V(i,j) = infinity if *i* >= *j*

V(i,j) = min(*eH(i,j),*

eS(i,j) + *V(i+1,j-1),*

VBI(i,j),

VM(i,j)) *if* *i* < *j*

i,j is exterior pair of a stacked pair. *i+1,j-1* is therefore a pair too.

Dynamic programming recurrences

$V(i,j)$: FE of optimal structure of $s(i..j)$ assuming i, j form a base pair

$V(i,j)$ = infinity if $i >= j$

$V(i,j)$ = min($eH(i,j)$,

$eS(i,j) + V(i+1,j-1)$,

i,j is exterior pair of a bulge or interior loop → $VBI(i,j)$,

$VM(i,j))$ if $i < j$

Dynamic programming recurrences

$V(i,j)$: FE of optimal structure of $s(i..j)$ assuming i, j form a base pair

$V(i,j)$ = infinity if $i >= j$

$V(i,j)$ = min($eH(i,j)$,

$eS(i,j) + V(i+1,j-1)$,

$VBI(i,j)$,

$VM(i,j))$ if $i < j$

i,j is exterior pair of a multibranch loop

Dynamic programming recurrences

VBI(i, j): FE of optimal structure of *s(i..j)* assuming *i,j* closes a bulge or internal loop

$$VBI(i, j) = \min_{\substack{i',j' \\ i<i'<j'<j}} (eL(i, j, i', j') + V(i', j'))$$

Energy of the bulge

Slow !

Dynamic programming recurrences

VM(i,j): FE of optimal structure of *s(i..j)* assuming *i, j* closes a multibranch loop

$$VM(i,j) = \min_{\substack{k \geq 2 \\ i_1,j_1,\ldots i_k,j_k}} (eM(i,j,i_1,j_1,..,i_k,j_k) + \sum_h V(i_h,j_h))$$

Energy of the loop itself

Very slow !

Order of computation

- What order to fill the DP table in ?
  - Increasing order of *(j-i)*
  - *VBI(i, j)* and *VM(i, j)* before *V(i, j)*