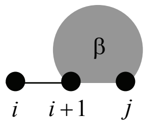
3.6.2 Parameterization and Conditioning

Let $\beta(i,j)$ be the maximum number of base pairs in a pseudoknot-free folding of subsequence $S[i \dots j]$. As the following recursive solution shows, consideration of all prefixes of S like in alignment or exon assembly would not be sufficient here. We want to calculate the best structure for subsequence $S[i \dots j]$ from the previously calculated best structure for smaller subsequences.

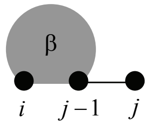
3.6.3 Recursive Solution and Bellman Principle

The computation of $\beta(i,j)$ distinguishes the following cases.

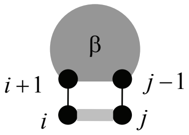
1. Add an unpaired base i to the best structure for smaller subsequence $S[i+1 \dots j]$.



2. Add an unpaired base j to the best structure for smaller subsequence $S[i \dots j-1]$.

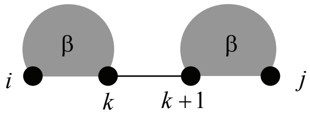


3. Add a base pair (i,j) with score $\delta(i,j)$ to the best structure for smaller subsequence $S[i+1 \dots j-1]$.



4. Combine two best structures for smaller subsequences $S[i \dots k]$ and $S[k+1 \dots j]$.

In case 1, $\beta(i,j)$ obviously equals $\beta(i+1,j)$. In case 2, $\beta(i,j)$ obviously equals $\beta(i,j-1)$. In case 3, binding of base i to base j contributes a score $\delta(i,j)$ to $\beta(i,j)$. Note that this score can either be defined as a constant or as a free



energy value reflecting how strong the bond between this specific base pair is. In case 4, we construct a so-called *bifurcation* with two best structures on each side. Summarizing all cases, we obtain the following equations called the *Nussinov algorithm* [60], for $1 \leq i \leq n$ and $i < j \leq n$:

$$\begin{aligned} \beta(i,i) &= 0 \qquad \text{for } i = 1, \ldots, n \\ \beta(i,i-1) &= 0 \qquad \text{for } i = 2, \ldots, n \\ \beta(i,j) &= \max \begin{cases} \beta(i+1,j) \\ \beta(i,j-1) \\ \beta(i+1,j-1) + \delta(i,j) \\ \max_{i<k<j} \left\{ \beta(i,k) + \beta(k+1,j) \right\}. \end{cases} \end{aligned} \tag{3.15}$$

Note that the additive term $\delta(i,j)$ can rule out pairings that are not admissible. For example, set

$$\delta(i,j) = \begin{cases} 1 & \text{if } (i,j) = (\mathrm{A},\mathrm{U}) \text{ or } (\mathrm{C},\mathrm{G}) \\ 0 & \text{else.} \end{cases}$$

In practice, a matrix will be filled along the diagonals and the solution can be recovered through a traceback step. Figure 3.2 visualizes how a matrix entry is computed recursively. Note that only the upper (or lower) half of the matrix needs to be filled. Therefore, after initialization the recursion runs from smaller to longer subsequences as follows:

```
for $l = 1$ to $n$ do
  for $i = 1$ to $n + 1 - l$ do
    $j = i + l$
    compute $\beta(i,j)$
  end for
end for
```

3.6.4 Number of Different Subcalls and Overall Complexity

There are $O(n^2)$ terms to be computed, each requiring calling of $O(n)$ already computed terms for the case of bifurcation. Thus overall complexity is $O(n^3)$ time and $O(n^2)$ space.

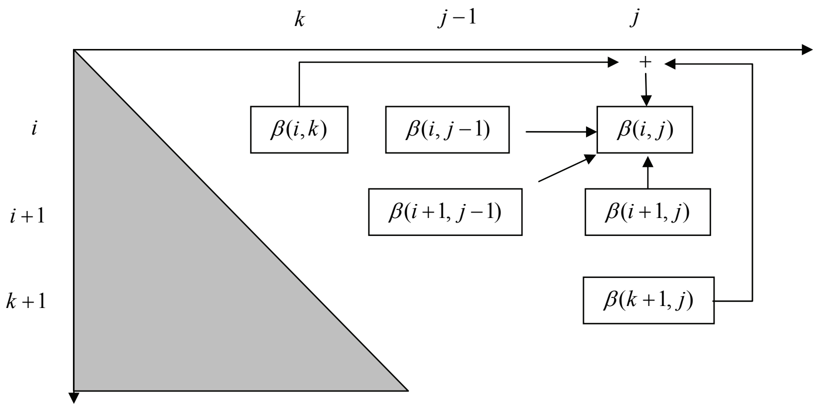


Fig. 3.2. Calculation of a matrix entry in the *Nussinov* algorithm

3.6.5 Variation: Free Energy Minimization

The base pair maximization approach neglects two main factors which in nature strongly drive RNA folding. First, loop sizes (entropic terms) are not taken into account. This is critical, as long unstructured loop regions destabilize and are therefore unlikely to form in a structure which aims for the state of minimum free energy. Second, *helical stacking* is ignored in the Nussinov algorithm. The enthalpic term which stacked base pairs (i, j) and $(i+1, j-1)$ contribute to stabilize a structure varies considerably depending on which bases are involved. For example, experimental work showed that helical stacking of pairs (G,U) and (U,G) has free energy of around -0.2 kcal/mol, whereas helical stacking of (U,G) and (G,U) is assigned free energy of around -1.5 kcal/mol. Note that helical stacking of pairs (i, j) and $(i + 1, j - 1)$ is not a symmetric relation.

In order to take into account global folding motifs such as stems, hairpin loops, bulges, internal loops, and multiloops (Sects. 1.6 and 2.9) we need to describe them in a formal notion. Thinking of loop regions, the most promising way to identify a structure element is by its *closing base pair*, i.e. the bond with largest distance. Let R be an RNA structure over a sequence S. We say a base k is *accessible* from a closing base pair $(i, j) \in R$ if there are no other base pairs $(i', j') \in R$ such that $i < i' < k < j' < j$. Informally stated, this means that if we want to reach an accessible base from the closing base pair, there is no other base pair in the way. Similarly, a base pair $(k, l) \in R$ is *accessible* from a closing base pair (i, j) if both k and l are. We call (k, l) an *interior base pair*. Now, let us define a structure element by its closing bond and interior base pairs as follows (Fig. 3.3):

- A loop with closing base pair $(i, j) \in R$ and one interior base pair $(i + 1, j - 1) \in R$ is called a *stem*.

- A loop with closing base pair $(i, j) \in R$ and no interior base pairs is called a *hairpin loop*.
- A loop with a closing base pair $(i, j) \in R$ and one interior base pair $(i', j') \in R$ with $(i' - i) + (j - j') > 2$ is called an *internal loop*.
- An internal loop is called a *bulge loop*, if $j' = j - 1$ or $i' = i + 1$.
- A loop with interior base pairs $(i_1, j_1) \ldots (i_k, i_k) \in R$ together with a closing base pair $(i, j) \in R$ is called a *k-multiloop*.

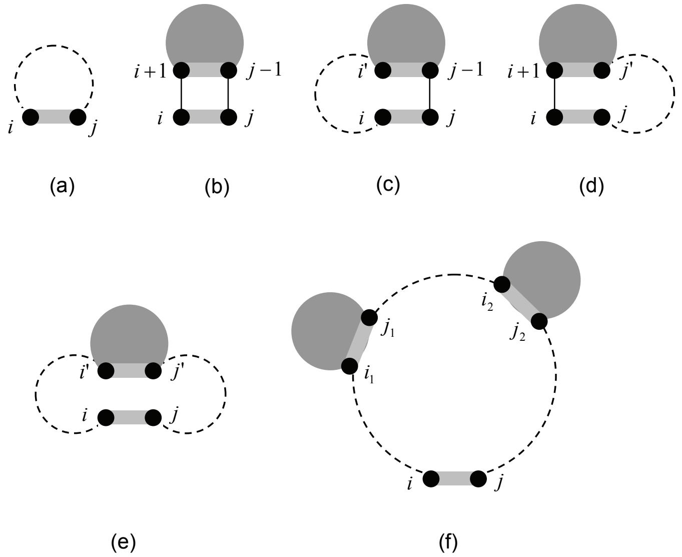


Fig. 3.3. Structure elements defined by their closing bond (i, j): **(a)** hairpin loop; **(b)** stem; **(c)** bulge to the left; **(d)** bulge to the right; **(e)** internal loop; **(f)** multiloop with $k = 2$ (*hydrogen bonding is indicated by gray area*)

For each of the structure elements, an energy function is defined which depends on the closing bond (i, j) and interior base pairs:

- $eS(i, j, i + 1, j - 1)$ is the energy of a *stem* closed by (i, j).
- $eH(i, j)$ is the energy of a *hairpin loop* closed by (i, j).
- $eL(i, j, i', j')$ is the energy of an *internal loop* or *bulge loop* closed by (i, j).
- $eM(i, i_1, j_1, \ldots, i_k, j_k, j)$ is the energy of a *k-multiloop* closed by (i, j) with interior base pairs $(i_1, j_1) \ldots (i_k, j_k)$. To make the energy computation tractable and avoid exponential runtime, the following *k-multiloop* energy simplification is commonly made:

$$eM(i, i_1, j_1, \ldots, i_k, j_k, j) = a + bk + ck' \tag{3.16}$$

with $a, b, c =$ constants and $k' =$ number of unpaired bases in the loop.

We fix the notation that every secondary structure element identified by its closing bond (i, j) contributes an energy value $E^R_{i,j}$ (calculated from the underlying energy function) to the overall free energy $E(R)$. The free energy of an RNA structure R can now be calculated as the additive sum:

$$E(R) = \sum_{(i,j)\in R} E^R_{i,j}. \tag{3.17}$$

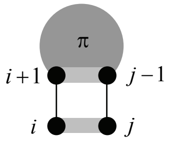
It is now clear that computing the minimum free energy structure for an RNA sequence is much more sophisticated than the *Nussinov* approach we discussed before. We simply maximized the number of base pairs (and possibly produced long unstructured loop regions). Here, we take the sum over individual energy contributions from structure elements such as stems and loops. Note that this free energy minimization approach fixes a certain *energy model*, namely the sum over structure element energy values. Therefore, it remains only an approximation of RNA folding. It allows an elegant framework for computational methods, however there are surely more complicated folding processes and factors hidden somewhere in the complex RNA world.

Zuker algorithm

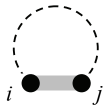
The structure elements and corresponding energy model described above are the scaffold for a clever dynamic programming algorithm, the so-called *Zuker algorithm* [81]. It takes an RNA sequence $S = S[1 \dots n]$ as input and computes the structure R with minimum free energy according to the underlying energy model. In the algorithm, three functions must be optimized simultaneously. In practice, these correspond to matrices which we need to fill. First, we define $\epsilon(i)$ to hold the minimum free energy of a structure on subsequence $S[1 \dots i]$. Second, we denote $\pi(i, j)$ to hold the minimum free energy of a structure on subsequence $S[i \dots j]$ *with i and j paired.* Third, another conditioned function is needed to account for multiloops. We define $\mu(i, j)$ to hold the minimum free energy of a structure on subsequence $S[i \dots j]$ that is *part of a multiloop.*

We start with the description of $\pi(i, j)$, which demands that bases i and j form a closing base pair for the structure element on $S[i \dots j]$ with minimum free energy. The following four cases have to be distinguished:

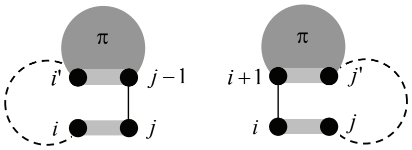
1. Base pair (i, j) closes a *stem.* We need to add the stacking energy $eS(i, j, i+1, j-1)$ and minimum free energy of a structure on smaller subsequence $S[i+1 \dots j-1]$ *closed by base pair* $(i+1, j-1)$.



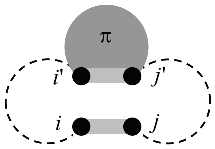
2. Base pair (i, j) closes a *hairpin loop*. We need to add the hairpin loop energy $eH(i, j)$.



3. Base pair (i, j) closes a *bulge* or *internal loop*. We need to add the loop energy $eL(i, j, i', j')$ and minimum free energy of a structure on smaller subsequence $S[i' \dots j']$ closed by base pair (i', j').

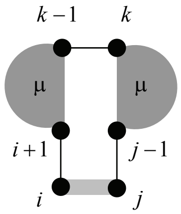


bulges



interior loop

4. Base pair (i, j) closes a *multiloop*. We need to decompose the multiloop into two smaller subsequences $S[i+1 \dots k-1]$ and $S[k \dots j-1]$ (described in more detail below) and add the offset penalty a.

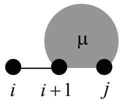


If bases i and j cannot form an admissible base pair, we set $\pi(i,j) = \infty$. The four cases lead to the following recurrence for all i, j with $1 \leq i < j \leq n$:

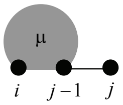
$$\begin{aligned}\pi(i,j) &= \min\left\{E(R) \mid R \text{ structure for } S[i\ldots j] \wedge (i,j) \in R\right\}\\ &= \min\begin{cases} eS(i,j,i+1,j-1) + \pi(i+1,j-1)\\ eH(i,j)\\ \min\limits_{\substack{i<i'<j'<i\\ i'-i+j-j'>2}} \left\{eL(i,j,i',j') + \pi(i',j')\right\}\\ \min\limits_{i+1<k\leq j-1} \left\{\mu(i+1,k-1) + \mu(k,j-1) + a\right\}. \end{cases}\end{aligned} \tag{3.18}$$

We initialize $\pi(i,i-1) = \pi(i,i) = \infty$, as these base pairs cannot form in a structure. Now, let us take a further look at case 4, the multiloop calculation. We already treated the closing base pair (i,j) in the $\pi(i,j)$ calculation. Searching for the multiloop structure with minimum free energy amongst all possible interior base pairs would lead to exponential runtime. Therefore, we use a simple trick to make the computation feasible. We recursively cut the two parts of a multiloop down to the interior base pairs by moving one base at a time or by adding another bifurcation. We also have to include penalties according to the underlying energy simplification (3.16). The following four cases have to be distinguished for $\mu(i,j)$:

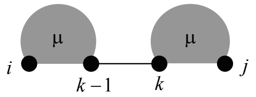
1. We move to base $i+1$, add a penalty c for an unpaired base, and the minimum free energy of a structure on smaller subsequence $S[i+1\ldots j]$ that is part of a multiloop.

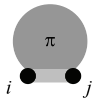


2. We move to base $j-1$, add a penalty c for an unpaired base, and the minimum free energy of a structure on smaller subsequence $S[i\ldots j-1]$ that is part of a multiloop.



3. We perform another multiloop bifurcation at base k and add the minimum free energy of two structures on smaller subsequences $S[i\ldots k-1]$ and $S[k\ldots j]$ that are part of a multiloop.
4. We discover an interior base pair (i,j) and obtain the minimum free energy of a structure on subsequence $S[i\ldots j]$ closed by (i,j).





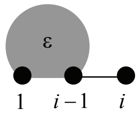
The four cases lead to the following recurrence for all i, j with $1 \leq i < j \leq n$:

$$\begin{aligned}\mu(i,j) &= \min\{E(R)|\ R \text{ structure for } S[i\ldots j] \text{ that is part of a multiloop}\}\\ &= \min\begin{cases}\mu(i+1,j)+c\\ \mu(i,j-1)+c\\ \min\limits_{i<k\leq j}\{\mu(i,k-1)+\mu(k,j)\}\\ \pi(i,j)+b.\end{cases}\end{aligned} \tag{3.19}$$

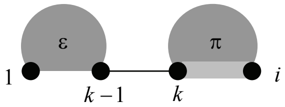
In order to ensure that we really produce at least two interior base pairs in a multiloop, we must initialize as follows: $\mu(i,i) = \infty$.

So far we are able to produce structure elements with a spanning closing base pair. However, there is no way to arrange structures in a consecutive fashion. Therefore, a third function $\epsilon(i)$ is introduced in the *Zuker* algorithm, which holds the minimum free energy of a structure on subsequence $S[1\ldots i]$. The computation of $\epsilon(i)$ distinguishes the following two cases:

1. Add an unpaired base i to the best structure for smaller subsequence $S[1 \ldots i-1]$.



2. Base i is paired to some base k. Add the minimum free energy of structures on smaller subsequences $S[1 \ldots k-1]$ and $S[k \ldots i]$ closed by (k, i).



Taking together both cases, we obtain the following formula for $\epsilon(i)$ with $1 \leq i \leq n$:

$$\begin{aligned}\epsilon(i) &= \min\left\{E(R) \mid R \text{ structure for } S[1 \ldots i]\right\} \\ &= \min \begin{cases} \epsilon(i-1) \\ \min\limits_{1 \leq k \leq i} \left\{\epsilon(k-1) + \pi(k,i)\right\} . \end{cases}\end{aligned} \tag{3.20}$$

We have to initialize as follows: $\epsilon(0) = 0$.

Now we have completely described the *Zuker* algorithm for computing an RNA structure with minimum free energy in the three equations (3.17), (3.18), and (3.19). In practice, three matrices $\epsilon(i)$, $\pi(i,j)$, and $\mu(i,j)$ are filled using the principle of dynamic programming for all i, j with $1 \leq i < j \leq n$. When all entries are computed, $\epsilon(n)$ contains the minimum free energy for an RNA sequence $S = S[1 \ldots n]$ and the corresponding best structure can be recovered by a traceback path.

Let us analyze the time and space requirements. Obviously, the entries for matrix $\epsilon(i)$ can be computed in $O(n^2)$ time and $O(n)$ space. It takes $O(n^3)$ time and $O(n^2)$ space to fill matrix $\mu(i,j)$. The most critical is matrix $\pi(i,j)$. Take a look at the case for bulges and internal loops. The minimum calculation requires $O(n^4)$ time and $O(n^2)$ space, as we have to minimize over two positions i' and j'. However, with certain internal loop length restrictions, the *Zuker* algorithm requires $O(n^3)$ time and $O(n^2)$ space to find the structure with minimum free energy.

It is easy to see that the *Zuker* algorithm excludes pseudoknots. The structure with lowest free energy for the subsequence $S[i \ldots j]$ is only allowed to contain non-crossing interactions within this interval. Otherwise, the dynamic