or taking into account that $\mathrm{sim}(s, t)$ is the largest possible alignment score between s and t,

$$\mathrm{sim}(s, t) + \mathrm{dist}(s, t) \geq \frac{M}{2}(m + n),$$

proving the theorem. ■

Combining Theorems 3.2 and 3.3, we can write

$$\mathrm{sim}(s, t) + \mathrm{dist}(s, t) = \frac{M}{2}(m + n), \tag{3.23}$$

which shows us how to compute the distance given the similarity. Thus, distance computations can be reduced to similarity computations. To compute a distance, all we need to do is select a suitable M, define scoring parameters p and g as in (3.16) and (3.17), and apply one of the algorithms for global comparison we have seen so far. The resulting similarity is converted to distance by the above formula.

For instance, in the case of the edit distance, we may choose $M = 0$ and run a similarity algorithm with match $= 0$, mismatch $= -1$, and space $= -1$. Or we may take $M = 2$ and have match $= 2$, mismatch $= 1$ and space $= 0$. Both scoring systems yield the same optimal alignments, although with different scores. But after applying formula (3.23) the distance is the same.

Before we close, a comment on the constant M is in order. It may seem suspicious that any value of M will do for Theorem 3.2. If M is a large, positive number, the scoring system may result in negative values of the space penalties $w'(k)$ for many, and possibly all, values of k. This contradicts our intuition, given that spaces should be penalized instead of rewarded. The same goes for a negative, but very large in absolute value, M. The function p this time would be negative, again contradicting our intuition. The reason for this apparent anomaly lies in the fact that we consider global comparisons only. When we change the value of M, all alignments increase or decrease on their score in a uniform manner, so that the optimal ones always remain the same. For local alignments, Equation (3.20) is not valid, and by varying M we can give preference to longer or shorter local alignments.

3.6.2 PARAMETER CHOICE IN SEQUENCE COMPARISON

In this section we present considerations concerning the choice of parameters in a scoring system and the choice of algorithm given the particular sequence comparison that must be made.

Many issues must be taken into consideration when choosing the scoring system for a particular application. This includes, in its simplest form, the scores for a match (M), for a mismatch (m), and for a space ($g < 0$).

In any scoring system it is important to assure that a match is worth more than a mismatch, so that we encourage alignments of identical characters. Another rule that is normally used is to assure that a mismatch is preferred over a pair of spaces. For instance, the following leftmost alignment should score higher than the rightmost one:

```
A  -A
C  C-.
```

The rules above translate at once into inequalities involving the scoring parameters:

$$2g < m < M.$$

Notice that if we multiply all weights by a positive constant, the optimal alignments remain the same. This property can be used to transform all weights to integers, which are generally processed with much greater speed than floating point numbers in the majority of modern computers.

Consider now the transpositions. If we have, for instance, sequences AT and TA, there are essentially two alignments that compete for the best score:

```
AT  -AT
TA  TA-.
```

(There is still a third alignment with the same score as the second one above aligning the T's.) The corresponding scores are $2m$ and $M + 2g$, respectively. Thus, if m is equal to the arithmetic mean between M and $2g$, the two preceding alignments are equivalent in terms of score. To give preference to one of them, we must choose m closer to one of the extremes of the interval $[2g, M]$.

In the same vein, it is possible to imagine other instances of pairs of short sequences and postulate which is the most desirable alignment in each case. This gives us more inequalities involving m, M, and g. For instance, when comparing ATCG to TCGA, we may prefer the first of the two following alignments:

```
ATCG-  ATCG
-TCGA  TCGA.
```

This will be reflected in the score if $4m < 3M + 2g$. The values used in Section 3.2.1 were chosen based on such criteria, among other reasons.

In practice, scoring systems more sophisticated than the simple M, m, g method are often needed. The subadditive space penalty functions mentioned in Section 3.3.3 are usually preferred. Among these, affine functions are very popular because of the quadratic running time algorithm, as opposed to cubic for general functions. Another important property, especially in comparisons of protein sequences, is the ability to distinguish among the various matchings of amino acids. A match involving amino acids with similar chemical or physical characteristics, such as size, charge, hydrophobicity, and so on, receives more points than a matching between not so similar ones. In this context, scoring systems based on identity only are in general insufficient. For protein comparison, the use of PAM matrices is commonplace.

Nevertheless, the simple identity/nonidentity method is versatile enough to include as particular cases several well-known problems in sequence comparison. One of these problems is to find the *longest common subsequence* (LCS) of a pair of sequences. This is equivalent to using $M = 1, m = g = 0$ in the basic algorithm. Thus, we can solve this problem in $O(mn)$ time and $O(\min(m, n))$ space for sequences of sizes m and n. This problem has received a great deal of attention, and several faster algorithms have been described for particular cases.

We now discuss the choice of algorithms. The decision whether to charge for end spaces and the choice of local or global methods depends heavily on the kind of application we are interested in and the results sought. If we want to compare sequences that are approximately the same length and relatively alike, a global comparison charging for all spaces is likely to be more appropriate. For instance, here we could include the case of two tRNAs of different organisms, or else two tRNAs of the same organism but carrying different amino acids.

If, on the other hand, one of the sequences is short and the other much longer, it is more advisable to charge for end spaces in the shorter sequence only. This will allow us to find all approximate occurrences of the short sequence in the long one. Such a search is useful when trying to locate relatively well-conserved structures in recently sequenced DNA.

Local comparison should be used when we have two relatively long sequences that may contain regions of high similarity. A typical case is protein sequences with similar functionality from reasonably well-separated organisms in terms of evolution. Because the proteins perform similar functions, it is probable that some high-similarity regions (active sites, motifs, functionally equivalent structures, etc.) exist, separated by unrelated regions that accumulated mutations and that do not have much influence on the protein's functionality.

If sequence comparison is done to test the hypothesis of common origin, care must be used when interpreting results. In general, optimal alignments are the most unlikely to have occurred by chance in some probabilistic sense. However, it is always advisable to compare the score obtained to what would be expected on average from completely unrelated sequences with the same characteristics as the two sequences compared. If the optimal score is well above average, this is a good indication that the similarity between the sequences is not due to chance. Even then, this result per se does not imply homology or any kind of evidence of common origin. Further experiments, based on the information the alignment gives, are in general carried out to strengthen or refute the hypothesis of common ancestry. On the other hand, if the similarity is nearly equal to what is expected by chance, it is likely that the sequences are unrelated. However, in biology there are no rules without exceptions, and cases of homologous proteins with no traces of similarity at the sequence level are known. Other pieces of evidence, such as three-dimensional structure, were used in these cases to ascertain homology.

3.6.3 STRING MATCHING AND EXACT SEQUENCE COMPARISON

Two other important problems that have relevance in computational molecular biology are the *string matching problem* and *exact sequence comparison*. In string matching we are given a string s, $|s| = n$, and a string t, $|t| = m$, and we want to find all occurrences of t in s. In other words, is t a substring of s? If it is, what are all the positions in s where we can find t? This is a classic computer science problem, and can be solved efficiently. In fact, there are algorithms that can solve this problem in time $O(n+m)$, which is much faster than the quadratic complexity of the basic algorithm for two-sequence comparison. We will not describe such algorithms here; they can be found on any good textbook