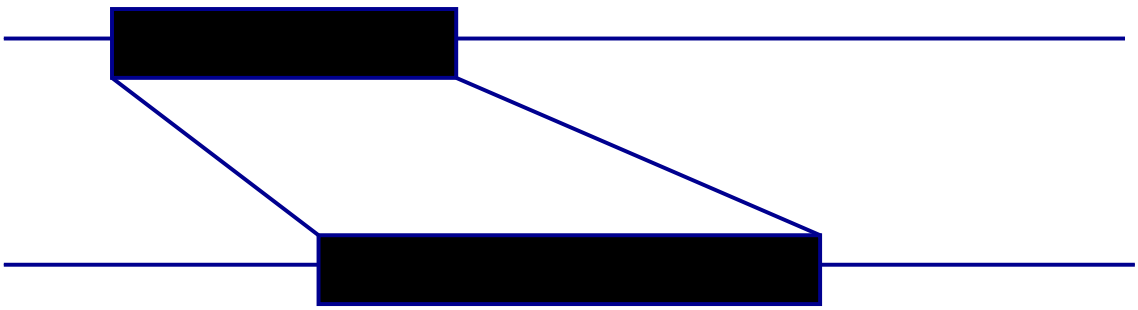
Local Alignment



```
...AT-CTA--TC...
...ATTCCAGATG...
```

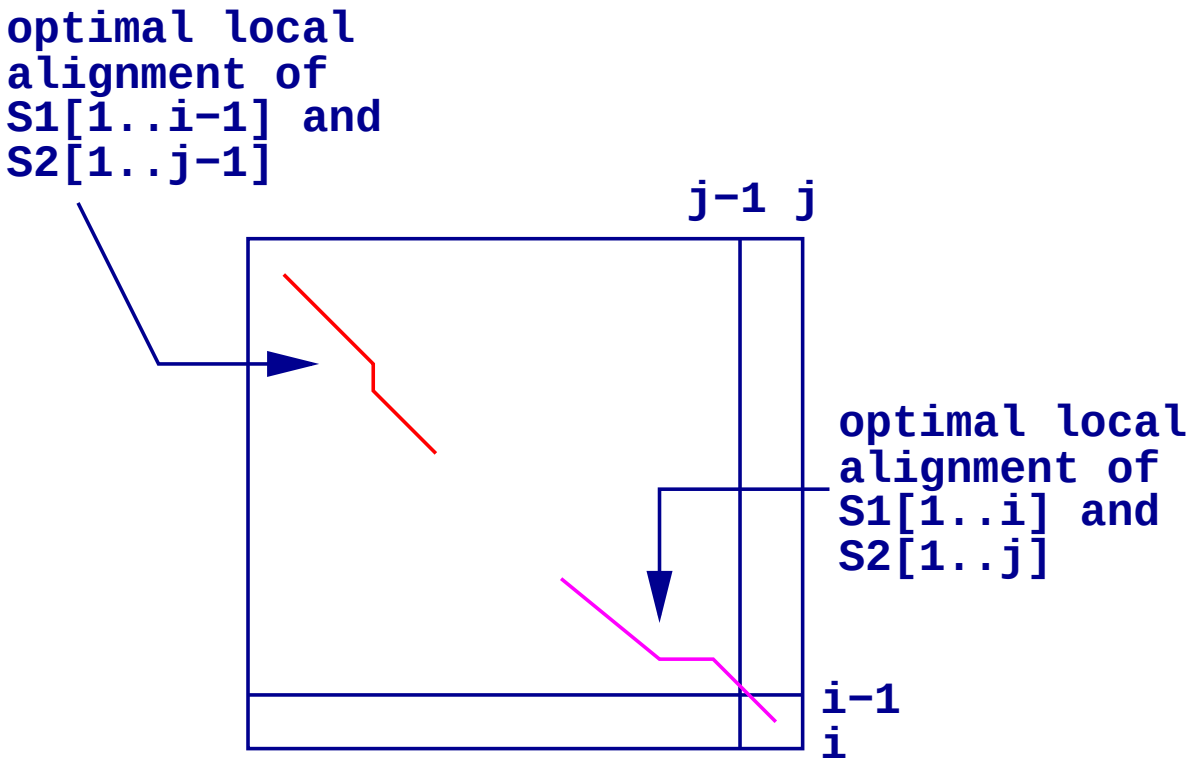
Global Alignment:

D(i,j)=optimal global alignment score of S1[1..i] and S2[1..j].

The optimal alignment is an extension of one of three shorter global alignments.

Local Alignment

H(i,j)=optimal local alignment score of S1[1..i] and S2[1..j] ?


optimal local
alignment of
S1[1..i−1] and
S2[1..j−1]
j−1 j
optimal local
alignment of
S1[1..i] and
S2[1..j]
i−1
i

H(i,j)=optimal local alignment score of S1[1..i] and S2[1..j] does not allow a dynamic programming approach.

BUT

Since gap costs are positive no optimal local alignment ends with a gap.

Every optimal local alignment ends with matching two characters (non negative score)

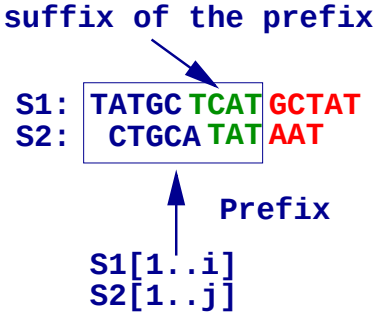
What about:
H(i,j)=optimal local alignment that ends with matching S1(i) and S2(j)?

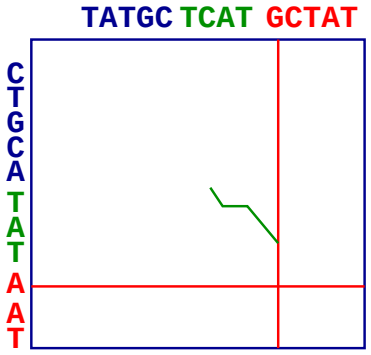
H(i,j)=optimal local alignment score of S1[1..i] and S2[1..j] does not allow a dynamic programming approach.

BUT

What about:
H(i,j)=maximal score of all local alignment, that contain S1(i) and S2(j) as there right most characters.

Optimal suffix alignment score of S1[1..i] and S2[1..j]





...assume linear gap costs

4 types of H(i,j)−alignments

(i)	...ATCGC T ...TTCCT A	H(i−1,j−1)+s(T,A)
(ii)	...ATCGCT − ...−TTCCT A	H(i,j−1)+g(1)
(iii)	...ATCGC− T ...TTCCTA −	H(i−1,j)+g(1)
(iv)	Empty	0

Note that $\begin{matrix} T \\ A \end{matrix}$ is a type (i) extension of the empty alignment

SMITH WATERMAN ALGORITHM

Recurrence relation for local alignments

$$H(i,j) = \max \begin{cases} H(i-1,j-1) + s(S1(i),S2(j)) \\ H(i,j-1) + s(-,S2(i)) \\ H(i-1,j) + s(S1(i),-) \\ 0 \end{cases}$$

S(i,j) = optimal global alignment score of S1[1..i] and S2[1..j].

INITIALIZATION

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
V	1	0							
I	2	0							
N	3	0							
T	4	0							
N	5	0							
E	6	0							
R	7	0							

Tabular calculation

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	*	*	*	*	*	*	*	*
V	1	*	*	*	*	*	*	*	*
I	2	*	*	*	*	*	*	*	*
N	3	*	*	*	?				
T	4	*							
N	5	*							
E	6	*							
R	7	*							

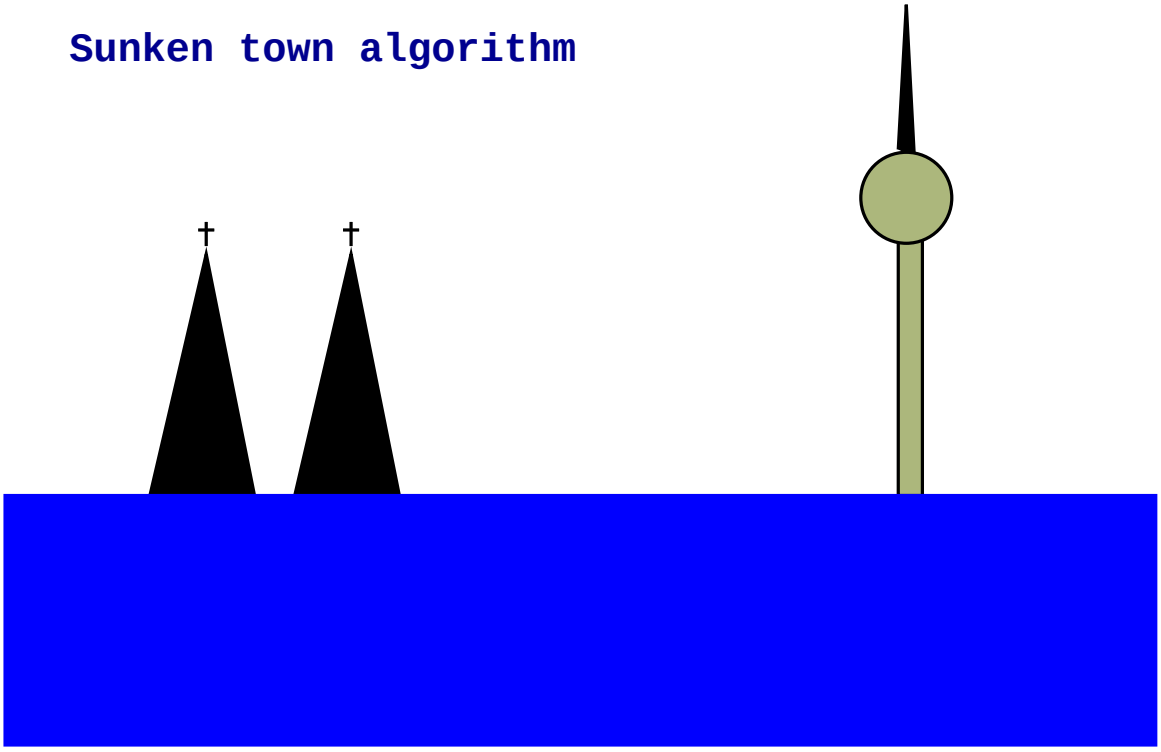
The number of colored spots depends on the lengths of the sequences.

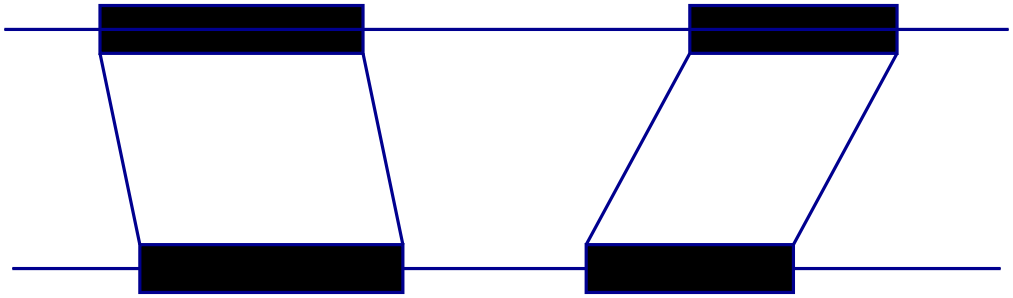
Traceback

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	*	*	*	*	*	*	*	*
V	1	*	0	*	*	*	*	*	*
I	2	*	*	*	*	*	*	*	*
N	3	*	*	*	*	*	*	*	*
T	4	*	*	*	*	*	*	*	*
N	5	*	*	*	*	*	*	*	*
E	6	*	*	*	*	*	max	*	*
R	7	*	*	*	*	*	*	*	*

The number of colored spots depends on the length of the sequences.

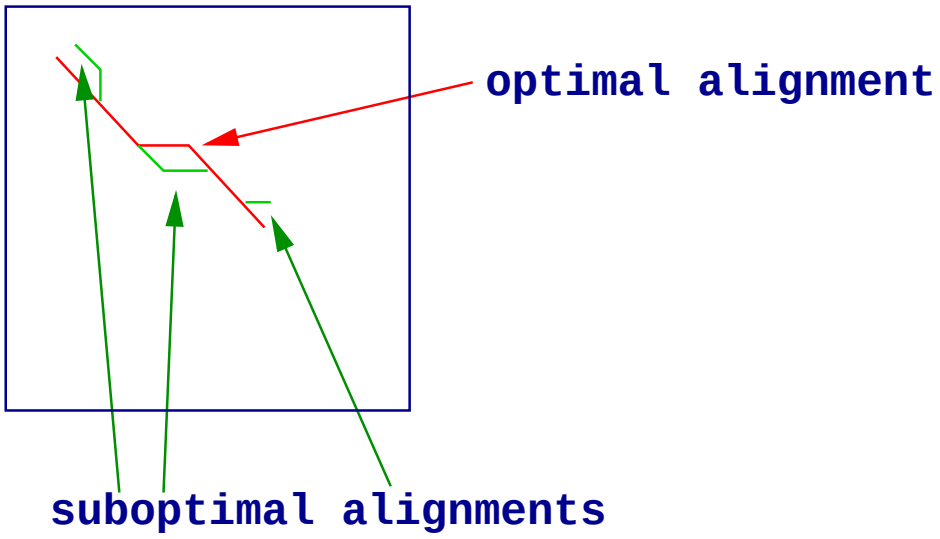
Sunken town algorithm





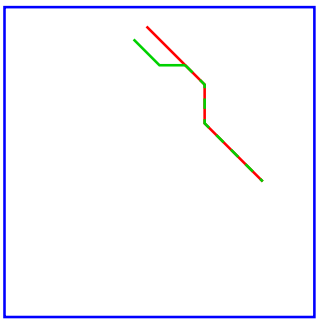
More than one conserved region...

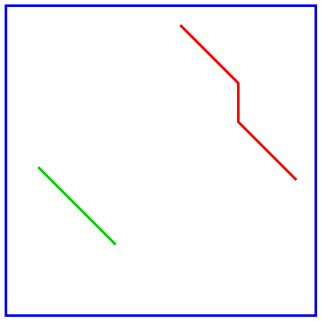
We have found the TV-tower, how can we find the cathedral?



Suboptimal alignments are often slight variants of the optimal alignment.

High score alignments occur in clumps.





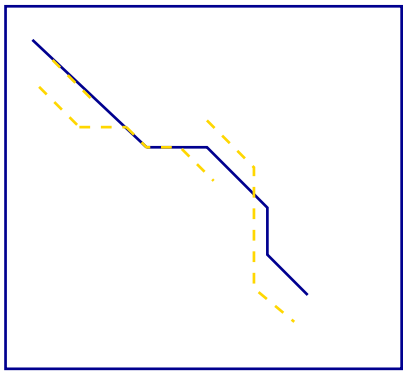
The same conserved region

Different conserved regions

We need a formal definition of what we consider to be a completely different local alignment.

Waterman and Eggert (1987)

For every local alignment one can define a clump of nearby alignments. This clump consists of all alignments that share at least one pair of matched characters with the reference alignment.



Waterman and Eggert algorithm

(1) Let C={} C: Clumps

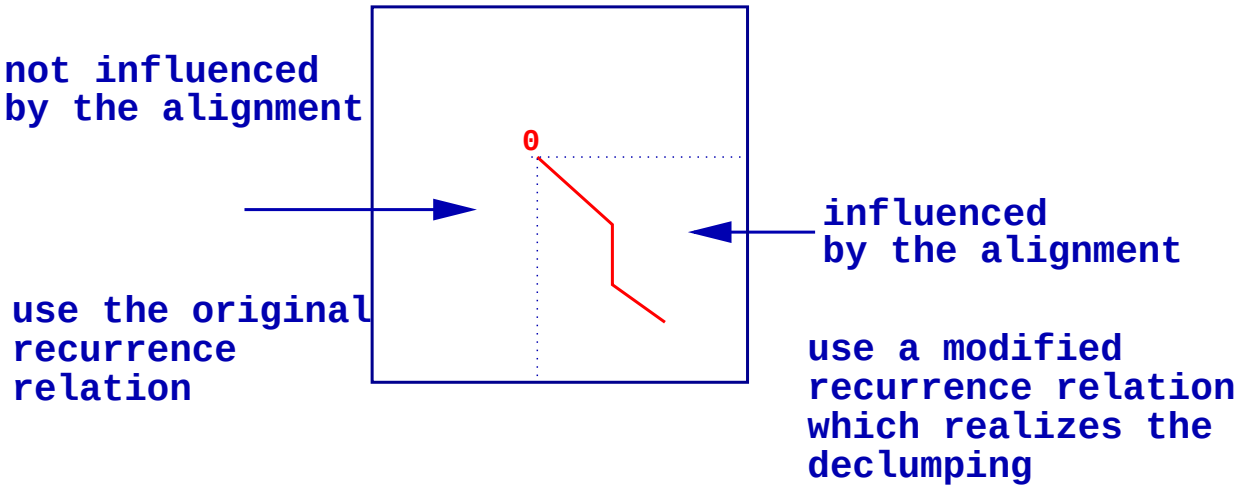
(2) Calculate the optimal local alignment that is not in C. (How?)

(3) Calculate the clump C0 associated to this alignment.

(4) Let C= C u C0

(5) Go to step (2) unless a given number of iterations is exceeded.

Declumping


not influenced
by the alignment
0
influenced
by the alignment
use the original
recurrence
relation
use a modified
recurrence relation
which realizes the
declumping

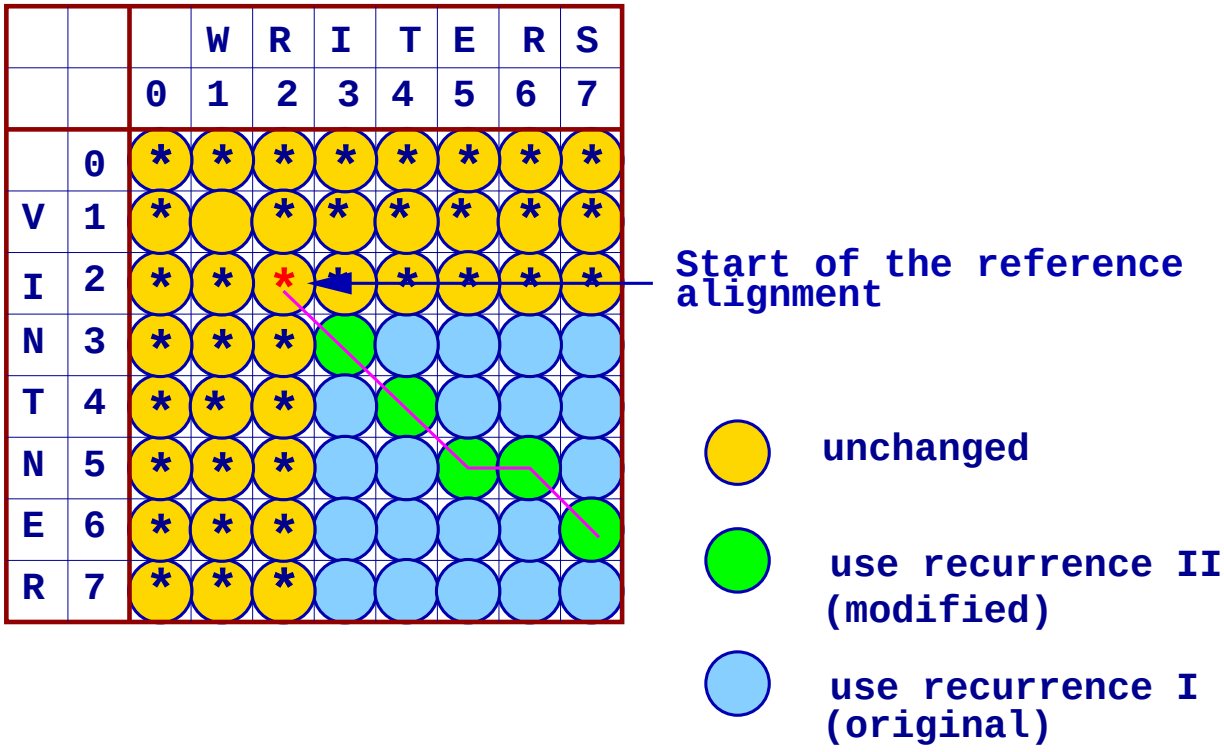
Recurrence relation I

$$H(i,j) = \max \begin{cases} H(i-1,j-1)+s(S1(i),S2(j)) \\ H(i,j-1)+s(-,S2(i)) \\ H(i-1,j)+s(S1(i),-) \\ 0 \end{cases}$$

Recurrence relation II

$$H(i,j) = \max \begin{cases} H(i,j-1)+s(-,S2(i)) \\ H(i-1,j)+s(S1(i),-) \\ 0 \end{cases}$$

The match is not alowed !


W R I T E R S
0 1 2 3 4 5 6 7
V 0
I 1
N 2
T 3
N 4
E 5
R 6
7
Start of the reference alignment
unchanged
use recurrence II (modified)
use recurrence I (original)

Semiglobal alignment

Fit one sequence into another

Detect Overlapp (Shotgun sequencing)

Endgaps do not correspond to insertions or deletions ... and they should not be penalized.

INITIALIZATION

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
V	1	0							
I	2	0							
N	3	0							
T	4	0							
N	5	0							
E	6	0							
R	7	0							

Traceback

			W	R	I	T	E	R	S
		0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
V	1	0	*	*	*	*	*	*	*
I	2	0	*	*	*	*	*	*	*
N	3	0	*	*	*	*	*	*	*
T	4	0	*	*	*	*	*	*	*
N	5	0	*	*	*	*	*	*	*
E	6	0	*	*	*	*	*	*	*
R	7	0	*	*	*	*	*	*	*

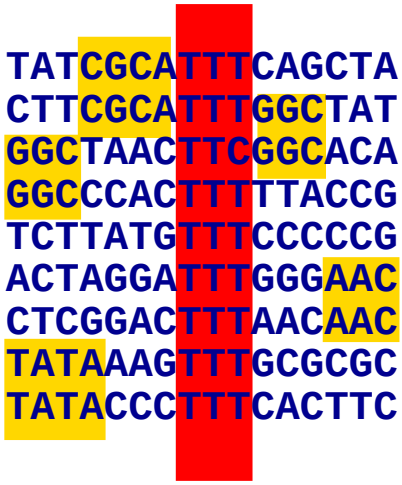
Start the trace back at the maximum of the ○ cells.

Global alignmnet: Closely related sequences

Local alignment: Remote sequences
Database searches
Closely related sequences
(Why not?)

Semi–global alignment: Shot gun sequencing
Fragment assembly

Part of many multiple alignment algorithms



Two homologous sequences whisper ...
a full multiple alignment shouts out loud.

A. Lesk

Profiles

```
T A T A A T
– A T A C T
T A T A A A
C A G A A T
T G T A – T
T – A C G T
G C T A A T
```

```
A:  0 4 1 6 4 1
C:  1 1 0 1 1 0
G:  1 1 1 0 1 0
T:  4 0 5 0 0 6
```

Given a new sequence ....

... does it fit into the profile?

A:	0	4	1	6	4	1
C:	1	1	0	1	1	0
G:	1	1	1	0	1	0
T:	4	0	5	0	0	6

calculate relative frequencies per column f(i,x)

Align a new sequence to the profile

The letters in this sequence are aligned against entire columns of the profile.

Position specific score:

$$S(Ci,y)= \sum_{x} f(i,x)s(x,y)$$

Where the sum is over all characters x in the alphabet, and y is a character in the new sequence.

The score of the profile alignment is the sum over all position specific scores.

Or we use a probabilistic model for each column:

p(i,j)=estimated frequency of letter i in column j.

q(i)=frequency of letter i in a background model.

Score: $S(C,x)=\log \frac{p(x,j)}{q(x)}$

Problem: Often not enough data for a good estimation of p(i,j) available.

Use pseudo counts, regularize with a score matrix, or use a PRIOR (best!).

Dirichlet priors are normally used.

Profiles can be aligned to sequences by standard dynamic programming algorithms ...

... one just has to replace the general scores by position dependent scores derived from the profile.

It is even possible to align profiles with profiles ...

$$S(C1,C2)=\sum_{i,j} w1(i,j)w2(i,j)s(i,j)$$

...less clear for other probabilistic scores.

Definition: Multiple alignment:

Given k>2 sequences:
Chosen spaces can be inserted inside or at the end of any sequence, such that all sequences have identical length, say l. Then the strings are arrayed in a k by l table. Columns consisting only of spaces are not allowed.

The sum of pairs score:

Given a score system for pairwise alignment ...
... gaps are treated like any other character.

The sum of pairs score of one column is the sum over all possible pairwise comparisons in this column.

$$S(C)=\sum_{i<j} S(C(i),C(j))$$

Example: match =1, mismatch=0, space=−1

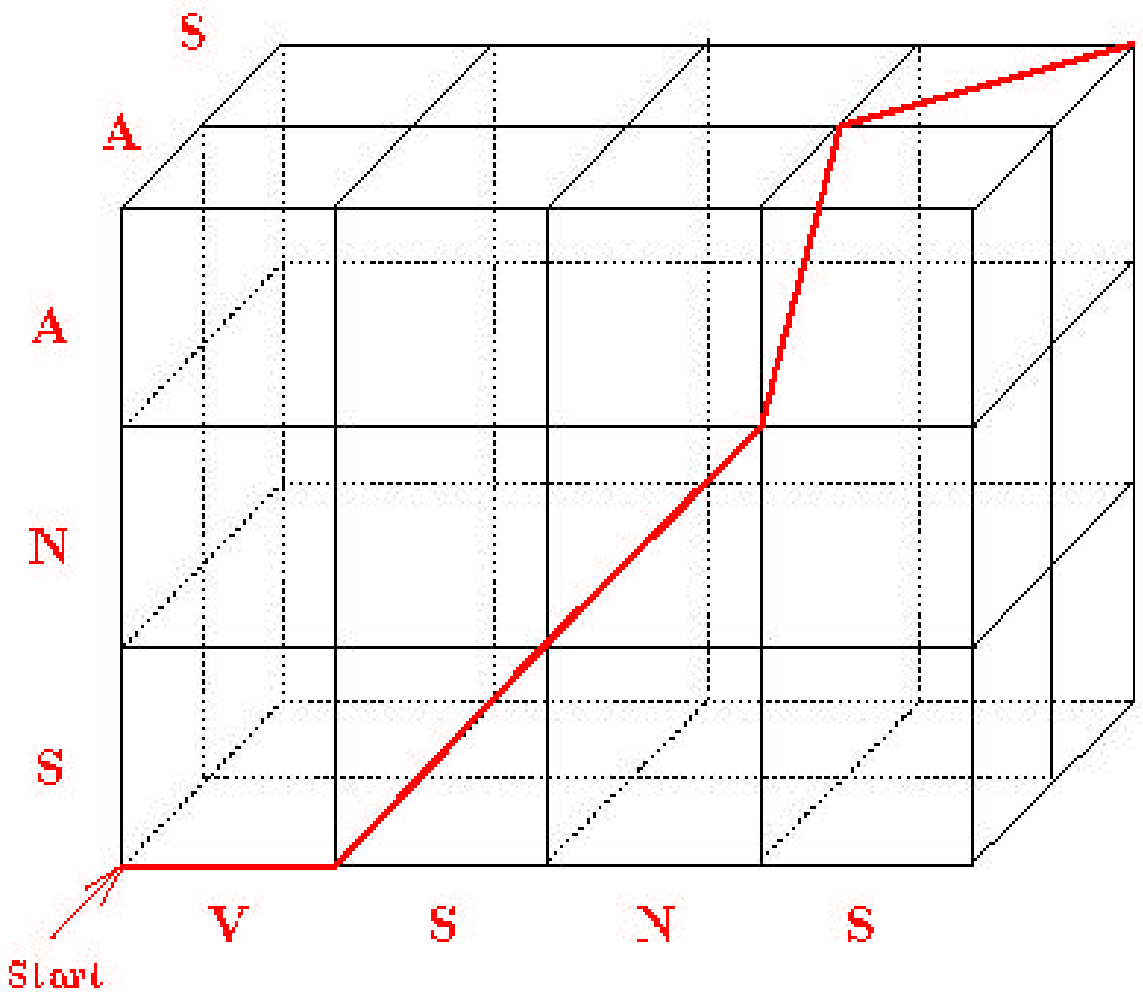
C= T / A / − / T

Score=0−1+1−1+0−1=−2.

The score of the alignment is the sum of the scores of the columns.

Three Protein Alignment
(Murata, Richardson & Sussman)

Figure 1: Alignment Path for 3 Sequences.



V S N – S
– S N A –
– – – A S

Time complexity exponential in the number of sequences:

$O(g(n1,...,nk)^k)$

NP–complete problem (Wang & Jiang 1994)

....forget it ...

... we had enough dynamic programming anyway.

Instead:
Construct multiple alignment from pairwise alignments.