



Εισαγωγή στη Βιοπληροφορική



Βασική βιβλιογραφική πηγή

- Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Τεχνικές Ανάλυσης και Σύγκρισης Ακολουθιών Βιολογικών Δεδομένων

- Προσεγγιστική Εύρεση Προτύπου - Approximate Pattern Matching
- Στοίχιση Ακολουθιών - Multiple Sequence Alignment
- Εφαρμογές σε Προβλήματα Μοριακής Βιολογίας

Παρατηρήσεις

- Η ολική στοίχιση καλείται Needleman-Wunsch alignment (1970)
- Η τοπική στοίχιση καλείται Smith-Waterman alignment (1981)
- Smith-Waterman can find regions with high similarity by simply performing a trace-back from any cell (i,j) backwards and locate a pair with similarity $v(i,j)$.

Βασικοί Ορισμοί (α)

- **Απόσταση Μετασχηματισμού - Edit Distance:** για 2 συμβολοσειρές ορίζουμε το ελάχιστο πλήθος των πράξεων μετασχηματισμού που απαιτούνται για να μετασχηματίσουμε την πρώτη συμβολοσειρά στη δεύτερη. Οι βασικές πράξεις μετασχηματισμού είναι η **ένθεση**, **διαγραφή** και **αντικατάσταση** συμβόλων.
- Παράδειγμα: S_1 : vintner και S_2 : writers
- $\text{edit-distance}(S_1 \rightarrow S_2) = 5$

Βασικοί Ορισμοί (β)

- **Ακολουθία Μετασχηματισμού - Edit Transcript:** για το μετασχηματισμό μιας συμβολοσειράς ορίζεται ως η ακολουθία των πράξεων μετασχηματισμού που απαιτούνται για να μετασχηματίσουμε την πρώτη συμβολοσειρά στη δεύτερη. Οι βασικές πράξεις μετασχηματισμού αναπαρίστανται ως εξής:
 - **ένθεση: I**
 - **διαγραφή: D**
 - **αντικατάσταση: R**
 - **ταίριασμα: M**
- Παράδειγμα: S_1 : vintner και S_2 : writers
- $\text{edit-distance}(S_1 \rightarrow S_2) = \text{RIMDMDMMI}$

Στοίχιση Ακολουθιών

- **Στοίχιση Ακολουθιών- Sequence Alignment:** τοποθετούμε τη μια ακολουθία κάτω από την άλλη έτσι ώστε οι κοινοί χαρακτήρες να τοποθετούνται στις ίδιες θέσεις.

-	V	I	N	T	N	E	R	-
		⇓		⇓		⇓	⇓	
W	R	I	-	T	-	E	R	S

Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Στοίχιση Ακολουθιών επιτρέποντας κενά

- Στοίχιση δυο ακολουθιών με την εισαγωγή 7 κενών χαρακτήρων σε 4 θέσεις, που μεταφράζεται ως μετάλλαξη της ακολουθίας του DNA στις αντίστοιχες θέσεις.

<i>c</i>	<i>t</i>	<i>t</i>	<i>t</i>	<i>a</i>	<i>a</i>	<i>c</i>	-	-	<i>a</i>	-	<i>a</i>	<i>c</i>
<i>c</i>	-	-	-	<i>c</i>	<i>a</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>a</i>	<i>t</i>	-	<i>c</i>

Υπολογισμός score-function στοίχισης ακολουθιών

1	2	3	4	5	6	7
<i>g</i>	<i>a</i>	<i>g</i>	-	<i>t</i>	<i>c</i>	<i>t</i>
<i>g</i>	<i>a</i>	<i>c</i>	<i>c</i>	<i>t</i>	<i>c</i>	-

S	a	c	g	t	-
a	1	-1	-2	0	-1
c		3	-2	-1	0
g			0	-4	-2
t				3	-1
-					0

■ **Score-function** = $0+1-2+0+3+3-1=4$

Η Μέθοδος του Δυναμικού Προγραμματισμού

■ Δυναμικός Προγραμματισμός:

Έστω 2 ακολουθίες S_1 και S_2 , θα συμβολίζουμε ως $D(i,j)$ την απόσταση μετασχηματισμού μεταξύ των προθεμάτων $S_1[1..i]$ και $S_2[1..j]$, δηλαδή τον ελάχιστο αριθμό πράξεων μετασχηματισμού που απαιτούνται για να μετασχηματίσουμε τους i πρώτους χαρακτήρες της ακολουθίας S_1 στους j πρώτους χαρακτήρες της ακολουθίας S_2 .

■ Χρήση 3 βασικών τεχνικών:

- σχέση αναδρομής- recurrence relation,
- χρήση πίνακα- tabular computation,
- σχέση οπισθοχώρησης- traceback.

Παράδειγμα Πίνακα Δυναμικού Πρ/σμου

<i>D(i,j)</i>			<i>w</i>	<i>r</i>	<i>i</i>	<i>t</i>	<i>e</i>	<i>r</i>	<i>s</i>
		0	1	2	3	4	5	6	7
	0	0	1	2	3	4	5	6	7
<u>v</u>	1	1	1	2	3	4	5	6	7
<u>i</u>	2	2	2	2	2	3	4	5	6
<u>n</u>	3	3	3	3	3	3	4	5	6
<u>t</u>	4	4	4	4	4	*			
<u>e</u>	5	5							
<u>r</u>	6	6							
s	7	7							

Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Η Σχέση Αναδρομής

■ Σχέση Αναδρομής:

$$D(i,j)=\min[D(i-1,j)+1,D(i,j-1)+1,D(i-1,j-1)+t(i,j)]$$

- $D(i,j-1)+1$: πρέπει να ενθέσουμε το χαρακτήρα $S_2[j]$
- $D(i-1,j)+1$: πρέπει να διαγράψουμε το χαρακτήρα $S_1[i]$,
- $D(i-1,j-1)+1$: για να μετασχηματίσουμε το χαρακτήρα $S_1[i]$ στο χαρακτήρα $S_2[j]$ πρέπει να αντικαταστήσουμε το χαρακτήρα $S_1[i]$, με το χαρακτήρα $S_2[j]$,
- $D(i-1,j-1)$: έχουμε ταίριασμα

Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Παράδειγμα: σχέση αναδρομής

$D(i,j)$			<i>w</i>	<i>r</i>	<i>i</i>	<i>t</i>	<i>e</i>	<i>r</i>	<i>s</i>
		0	1	2	3	4	5	6	7
	0	0	1	2	3	4	5	6	7
<u>v</u>	1	1	1	2	3	4	5	6	7
<u>i</u>	2	2	2	2	2	3	4	5	6
<u>n</u>	3	3	3	3	3	3	4	5	6
<u>t</u>	4	4	4	4	4	*			
<u>e</u>	5	5							
<u>r</u>	6	6							
<u>s</u>	7	7							



$D(4,4) = D(3,3) = 3$, αφού $S_1(4) = S_2(4) = t$.

Η Σχέση Οπισθοχώρησης

■ Σχέση Οπισθοχώρησης:

- από την (i,j) θέση προς την $(i,j-1)$ αν $D(i,j) = D(i,j-1) + 1$ (ένθεση χαρακτήρα)
- από την (i,j) θέση προς την $(i-1,j)$ αν $D(i,j) = D(i-1,j) + 1$ (διαγραφή χαρακτήρα)
- από την (i,j) θέση προς την $(i-1,j-1)$ αν $D(i,j) = D(i-1,j-1) + t(i,j)$ (αντικατάσταση χαρακτήρα ή ταίριασμα)

Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Προσθήκη δεικτών οπισθοχώρησης

D(i,j)			w	r	i	t	e	r	s
		0	1	2	3	4	5	6	7
	0	0	? 1	? 2	? 3	? 4	? 5	? 6	? 7
v	1	?1	? 1	? ? 2	? ? 3	? ? 4	? ? 5	? ? 6	? ? 7
i	2	?2	? ?2	? 2	? 2	? 3	? 4	? 5	? 6
n	3	?3	? ?3	? ?3	? ?3	? 3	? ?4	? ?5	? ? 6
t	4	?4	? ?4	? ?4	? ?4	? 3	? ?4	? ?5	? ? 6
e	5	?5	? ?5	? ?5	? ?5	?4	? 4	? ?5	? ? 6
r	6	?6	? ?6	? ?6	? ?6	?5	? 4	? ?5	? ? 6
s	7	?7	? ?7	? 6	? ? ?7	?6	?5	? 4	? 5

Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Ερμηνεία δεικτών οπισθοχώρησης

D(i,j)			w	r	i	t	e	r	s
		0	1	2	3	4	5	6	7
	0	0	? 1	? 2	? 3	? 4	? 5	? 6	? 7
v	1	? 1	? 1	? ? 2	? ? 3	? ? 4	? ? 5	? ? 6	? ? 7
i	2	? 2	? ? 2	? 2	? 2	? 3	? 4	? 5	? 6
n	3	? 3	? ? 3	? ? 3	? ? 3	? 3	? ? 4	? ? 5	? ? 6
t	4	? 4	? ? 4	? ? 4	? ? 4	? 3	? ? 4	? ? 5	? ? 6
e	5	? 5	? ? 5	? ? 5	? ? 5	? 4	? 4	? ? 5	? ? 6
r	6	? 6	? ? 6	? ? 6	? ? 6	? 5	? 4	? ? 5	? ? 6
s	7	? 7	? ? 7	? 6	? ? ? 7	? 6	? 5	? 4	? 5

V	I	N	T	N	E	R	-
W	R	I	T	-	E	R	S

Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Πολυπλοκότητα της μεθόδου Δυναμικού Προγραμματισμού

- Αρχικοποίηση: $O(n) + O(m)$
- Σχέση Αναδρομής: $O(n*m)$
- Δείκτες Οπισθοχώρησης: $O(n+m)$
- Πολυπλοκότητα: $O(n^2)$
- Ισοδυναμία με πρόβλημα της θεωρίας γραφημάτων, όπου κάθε κόμβος έχει ετικέτα ένα ζεύγος (i,j)

Βασικοί Ορισμοί (γ)

- **Ζυγισμένη Απόσταση Μετασχηματισμού - Weighted Edit Distance:** το ελάχιστο πλήθος των πράξεων μετασχηματισμού που απαιτούνται για να μετασχηματίσουμε την πρώτη συμβολοσειρά στη δεύτερη. Κάθε πράξη μετασχηματισμού έχει συγκεκριμένο κόστος - βάρος. Έστω ότι οι βασικές πράξεις μετασχηματισμού έχουν τα ακόλουθα βάρη:
 - ένθεση ή διαγραφή: d
 - αντικατάσταση: r
 - ταίριασμα: m .
- Παράδειγμα: S_1 : vintner και S_2 : writers
- $\text{weighted edit-distance}(S_1 \rightarrow S_2) = \mathbf{r+4d+4m}$.

Η Σχέση Αναδρομής με βάρη

■ Σχέση Αναδρομής:

$$D(i,j)=\min[D(i-1,j)+d,D(i,j-1)+d,D(i-1,j-1)+t(i,j)],$$

■ όπου:

- $t(i,j) = e$, αν $S_1(i)=S_2(j)$,
- $t(i,j)=r$, αν $S_1(i)\neq S_2(j)$ και
- $D(i,0)=i*d$ και $D(0,j)=j*d$.

Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Δυναμικός Προγραμματισμός & Ομοιότητα Ακολουθιών βάσει αλφαβήτου

- **Σχέση Αναδρομής για την ομοιότητα ακολουθιών:**

$$V(i,j) = \max[V(i-1,j-1) + s(S_1(i), S_2(j)), V(i-1,j) + s(S_1(i), _), V(i,j-1) + s(_, S_2(j))],$$

- **όπου:**

- $s(x,y)$: η τιμή στοίχισης του χαρακτήρα x με τον y

- $V(0,j) = \sum s(_, S_2(k)), 1 \leq k \leq j$ και $V(i,0) = \sum s(S_1(k), _), 1 \leq k \leq i.$

ΕΠΕΚΤΑΣΕΙΣ

- Ζυγισμένη Απόσταση Μετασχηματισμού βάσει Αλφαβήτου - Weighted Edit Distance.
- Σε εφαρμογές Μοριακής Βιολογίας τα βάρη αντικατάστασης χαρακτήρων αποθηκεύονται σε Πίνακες Αντικατάστασης - Substitution Matrix: PAM και BLOSUM (proteins)
- Καλύτερη οπτική η αντιμετώπιση σαν alignment και η ενσωμάτωση του score.
- Κατά κανόνα match είναι θετικό, οτιδήποτε άλλο 0 ή αρνητικό
- Chapter6_edit_distance_application

ΕΠΕΚΤΑΣΕΙΣ

- Longest Common Subsequence (match weight 1, otherwise 0)
- End-space free variant that encourages one string to align in the interior of the other, or the suffix of one to align with the prefix of the other (initial conditions 0, everything is countable in the last row and the last column) – shotgun sequence assembly
- Approximate occurrence of P in T (the optimal alignment of P to a substring of T has distance δ from the optimal alignment) (initial conditions 0). $V(0,j)=0$.
 - locate a cell (m,j) with value greater than δ .
 - traverse backpointers from (m,j) to $(0,k)$.
 - occurrence in $T[k,j]$

Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

1. Map longest common subsequence to longest increasing subsequence

Έστω οι δύο ακολουθίες $S1$ και $S2$ των οποίων θέλουμε να βρούμε τη μέγιστη κοινή υποακολουθία. Έστω ότι η $S2$ είναι μικρότερη σε μέγεθος από την $S1$. Για κάθε χαρακτήρα x του αλφαβήτου που εμφανίζεται τουλάχιστον μία φορά στην $S1$, δημιουργήστε μια λίστα από τις θέσεις όπου ο χαρακτήρας x εμφανίζεται στη συμβολοσειρά $S2$. γράψτε αυτή τη λίστα ανάποδα.

Το πρόβλημα longest common subsequence (εύρεση μέγιστης κοινής υποακολουθίας των $S1$ και $S2$) ανάγεται σε πρόβλημα εύρεσης της μεγαλύτερης αύξουσας υποακολουθίας στην λίστα.

2. Compute the Longest increasing subsequence algorithm

Ξεκινώντας από τα αριστερά, εξετάστε κάθε διαδοχικό αριθμό και τοποθετήστε τον στο τέλος της πρώτης (αριστερότερης) φθίνουσας υποακολουθίας που μπορεί να επεκτείνει. Εάν δεν υπάρχουν φθίνουσες υποακολουθίες που μπορούν να επεκταθούν τότε ξεκινήστε μια νέα (φθίνουσα) υποακολουθία δεξιά από όλες τις υπάρχουσες φθίνουσες υποακολουθίες.

Ενδιαφέρον paper: Maxime Crochemore, Ely Porat: Fast computation of a longest increasing subsequence and application. Inf. Comput. 208(9): 1054-1059 (2010)

Το Πρόβλημα Τοπικής Στοίχισης Επιθέματος- Local Suffix Alignment Problem

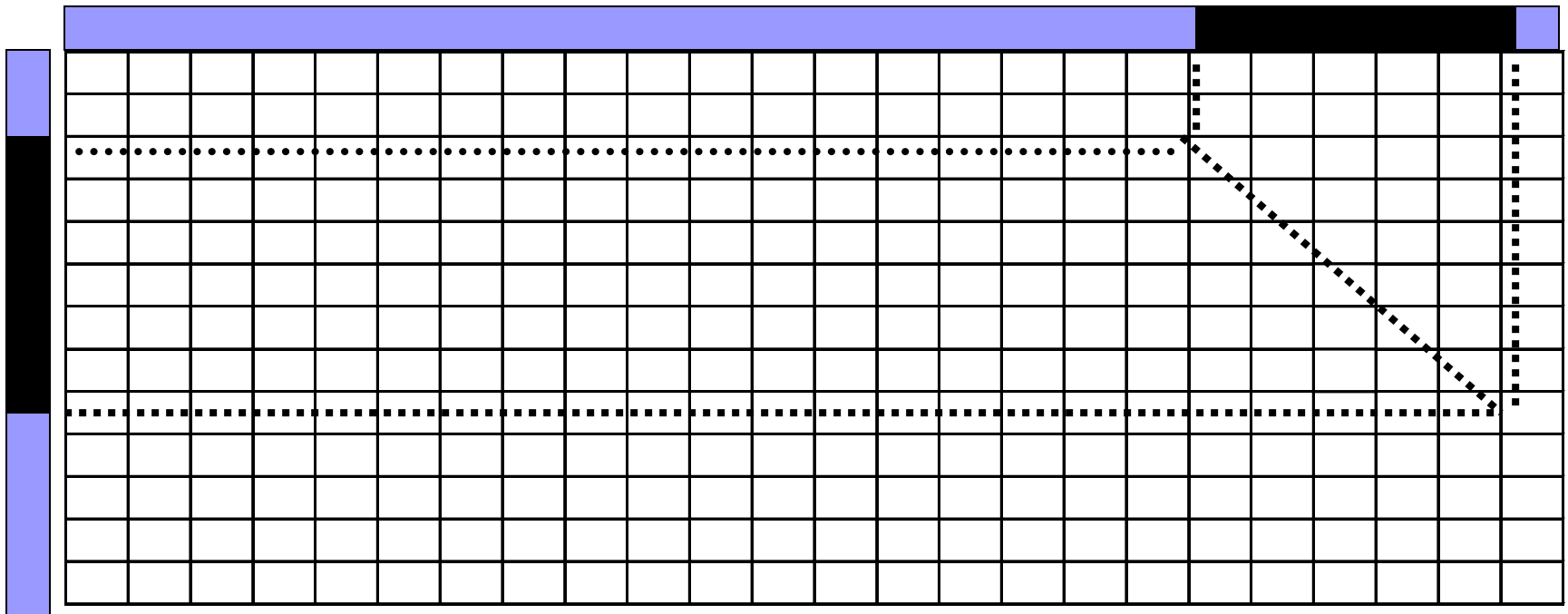
- **Local suffix alignment problem:** για δυο ακολουθίες S_1 και S_2 εντόπισε ένα επίθεμα α του $S_1[1..i]$ (με την πιθανότητα να είναι κενό) και ένα επίθεμα β του $S_2[1..j]$ (πιθανόν κενό) τέτοια ώστε το $V(\alpha, \beta)$ να έχει τη μέγιστη τιμή από όλα τα άλλα δυνατά ζεύγη επιθεμάτων των $S_1[1..i]$ και $S_2[1..j]$. Συμβολίζουμε ως $u(i, j)$ τη βέλτιστη τοπική στοίχιση επιθεμάτων για τις τιμές i και j ($i < n$ και $j < m$).
- Αρχικές συνθήκες $v(i, 0) = 0$ και $v(0, j) = 0$ καθώς μπορούμε να επιλέξουμε κάθε άδειο επίθεμα.
- $$u(i, j) = \max[0, u(i-1, j-1) + s(S_1(i), S_2(j)), u(i-1, j) + s(S_1(i), _), u(i, j-1) + s(_, S_2(j))]$$

Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Locally Similar Strings

T: 

S: 



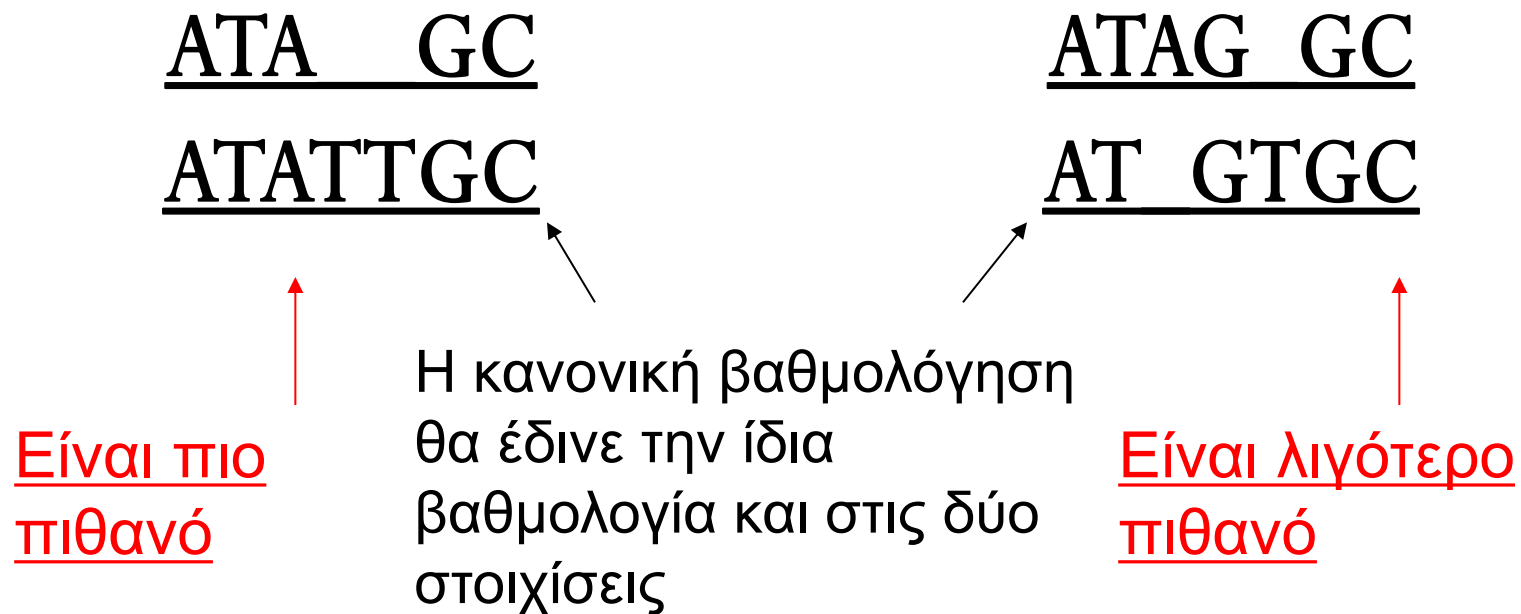
Πηγή Dan Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, 2010.

Στοίχιση Ακολουθιών με κενά

- **Έννοια κενού:** συνεχόμενα spaces, θέλουμε να ελέγχουμε την κατανομή των κενών.
- **Εισαγωγή Κενών:** Για να συμπεριλάβουμε το κόστος που η εισαγωγή κενών εισάγει στη στοίχιση 2 ακολουθιών, μπορούμε σε μια απλή προσέγγιση να θεωρήσουμε ότι κάθε κενό συνεισφέρει ένα σταθερό βάρος W_g , ανεξάρτητα από το μήκος του.
- τιμή στοίχισης που περιέχει “k” κενά:
$$\sum_{i=1}^l s(S'_1(i), S'_2(i)) - kW_g$$
- μία καλύτερη προσέγγιση είναι η χρησιμοποίηση μίας συνάρτησης του μήκους του κενού. Τότε μπορούμε να γεμίσουμε ένα πίνακα: $V(i,j) = \max[E(i,j), F(i,j), G(i,j)]$

Συγγενικές ποινές κενού

- Στη φύση, μια σειρά k προσθαιρέσεων εμφανίζεται συχνά ως ένα μόνο συμβάν αντί για μια σειρά k συμβάντων που αφορούν μεμονωμένα νουκλεοτίδια:



Λαμβάνοντας υπόψη τα κενά

- Κενά – συνεχόμενη ακολουθία (κενών) διαστημάτων σε μία από τις γραμμές
- Η βαθμολογία για ένα κενό μήκους x είναι:
 $-(\rho + \sigma x)$
όπου $\rho > 0$ είναι η ποινή για την εισαγωγή του κενού:
ποινή ανοίγματος κενού
το ρ θα είναι μεγάλο σε σχέση με το σ :
ποινή επέκτασης κοινού
επειδή δεν θέλουμε να έχουμε πολύ μεγάλη ποινή για την επέκταση του κενού.

Συγγενικές ποινές κοινού

- Ποινές κενού:

- ☐ - ρ - σ όταν υπάρχει 1 προσθαφαίρεση
- ☐ - ρ -2 σ όταν υπάρχουν 2 προσθαφαιρέσεις
- ☐ - ρ -3 σ όταν υπάρχουν 3 προσθαφαιρέσεις, κλπ.
- ☐ - ρ - x · σ (-άνοιγμα κενού - x επεκτάσεις κενού)

- Επιβάλλονται σχετικά μειωμένες ποινές (σε σύγκριση με την απλοϊκή βαθμολόγηση) σε ακολουθίες οριζόντιων και κάθετων ακμών

Στοίχιση Ακολουθιών με κενά

- **Έννοια κενού:** συνεχόμενα spaces, θέλουμε να ελέγχουμε την κατανομή των κενών.
- **Εισαγωγή Κενών:** Για να συμπεριλάβουμε το κόστος που η εισαγωγή κενών εισάγει στη στοίχιση 2 ακολουθιών, μπορούμε σε μια απλή προσέγγιση να θεωρήσουμε ότι κάθε κενό συνεισφέρει ένα σταθερό βάρος W_g , ανεξάρτητα από το μήκος του.
- τιμή στοίχισης που περιέχει “k” κενά:
$$\sum_{i=1}^l s(S'_1(i), S'_2(i)) - kW_g$$
- μία καλύτερη προσέγγιση είναι η χρησιμοποίηση μίας συνάρτησης του μήκους του κενού $w(x)$. Τότε μπορούμε να γεμίσουμε ένα πίνακα: $V(i,j) = \max[E(i,j), F(i,j), G(i,j)]$

Στοιχισή με arbitrary gap weights

$$V(i,j) = \max[E(i,j), F(i,j), G(i,j)]$$

i



$$G(i,j) = V(i-1, j-1) + \text{cost}(i \rightarrow j)$$

i

$$E(i,j) = \max_K V(i,k) - w(j-k) \quad (0 \leq k \leq j-1)$$

i



i

$$F(i,j) = \max_l V(l,j) - w(i-l) \quad (0 \leq l \leq i-1)$$

i



i

- $V(i,j)=\max\{E(i,j), F(i,j), G(i,j)\}$
- $V(i,0)=-w(i)$
- $V(0,j)=-w(j)$
- $E(i,0)=-w(i)$
- $F(0,j)=-w(j)$
- $G(0,0)=0$
- Assuming that $|S_1|=n$ and $|S_2|=m$ the recurrences can be evaluated in $O(nm^2+n^2m)$