ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Εισαγωγή στη Βιοπληροφορική

Ενότητα 7: Αλγόριθμοι εναλλακτικών προσεγγιστικών ταιριασμάτων βιολογικών δεδομένων

Μακρής Χρήστος, Τσακαλίδης Αθανάσιος, Περδικούρη Αικατερίνη

Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ και Πληροφορικής

Σκοποί ενότητας

- Σκοπός της ενότητας είναι η παρουσίαση εναλλακτικοί αλγόριθμοι προσεγγιστικού ταιριάσματος βιολογικών δεδομένων

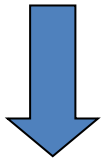
Βασικές Βιβλιογραφικές Πηγές στις οποίες βασίζονται οι διαφάνειες

- Costas S. I Costas S. Iliopoulos, Christos Makris, Yannis Panagis, Katerina Perdikuri, Evangelos Theodoridis, Athanasios K. Tsakalidis: The Weighted Suffix Tree: An Efficient Data Structure for Handling Molecular Weighted Sequences and its Applications. Fundam. Informaticae 71(2-3): 259-277 (2006)

- Maxime Crochemore, Costas S. Iliopoulos, Christos Makris, Wojciech Rytter, Athanasios K. Tsakalidis, T. Tsichlas: Approximate String Matching with Gaps. Nord. J. Comput. 9(1): 54-65 (2002)

Motivation

- DNA sequencing processes large chains into subsequences of ~500 characters long
- Assembling all pieces, produces a single sequence but…
  - At some positions we have uncertainty
  - Uncertainty: NOT '*' each character appears with some probability



Weighted Sequence

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}
A	1	0	0	0	0.5	0	0	0.5	0	0	0
C	0	1	0	0	0.5	0	1	0.3	0	0	0
G	0	0	0	0	0	0	0	0	0	0	0
T	0	0	1	1	0	1	0	0.2	1	1	1

Definitions

- ✓ Word *w* : a sequence of zero or more characters from an alphabet *Σ*.
- ✓ w = w[1]w[2]…w[n] or w[1..n]
- ✓ Subword *u* = w[*i*..*i*+*p*-1]. If *i*=1, *u* is a prefix. If *i*+*p*-1 = *n*, *u* is a suffix.

- ✓ Repeat: At least two equal subwords *u*

Definitions (cont'd)

Repetition: *At least two consecutive equal subwords*

u_1 = w[*i*..*i*+*p*-1] = u_2 = w[*i*+*p*..*i*+2**p*-1]=…

Example: w= abaabab

abaaba

aa

abab

Cover *u*: A repeated subword that covers the entire sequence (allowing catenations and overlaps)

Weighted words

Weighted word $w = w_1 w_2 \ldots w_n$.

– $w_i = \{(\sigma_1, p_i(\sigma_1)), (\sigma_2, p_i(\sigma_2)), \ldots\}$

– $\sigma \in \Sigma$ and

Example: $\sum_{\forall \sigma} p_i = 1$

Σ = {A, C, G, T}

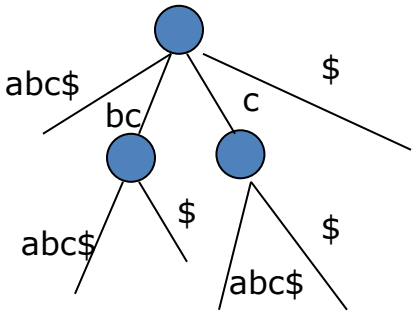
1	2	3	4	5	6	7	8	9	10	11
A	C	T	T	(A, 0.5) (C, 0.5) (G, 0) (T, 0)	T	C	(A, 0.5) (C, 0.3) (G, 0) (T, 0.2)	T	T	T

Q: Which subwords occur with probability ≥ 1/4?

A: ACTTATCATTT (0.25), ACTTCTCATTT(0.25) ATTT (0.5), CTTT(0.3) **and all their subwords** (but not ACTTATCCTTT)

Suffix trees

- ✓ Suffix tree $T(S)$ of a sequence S, $|S| = n$ is the *compact trie* of all the suffixes of $S\$$, $\$ \notin \Sigma$.
  - Leaf v is labeled with integer i if stores $S[i..n]$
  - At internal node v
    - LL(v) = list of suffixes at its descendants (*leaf-list*)
    - L(v) = the string spelled from root to v (*path label*)
  - Can be built in $O(n)$ time and space

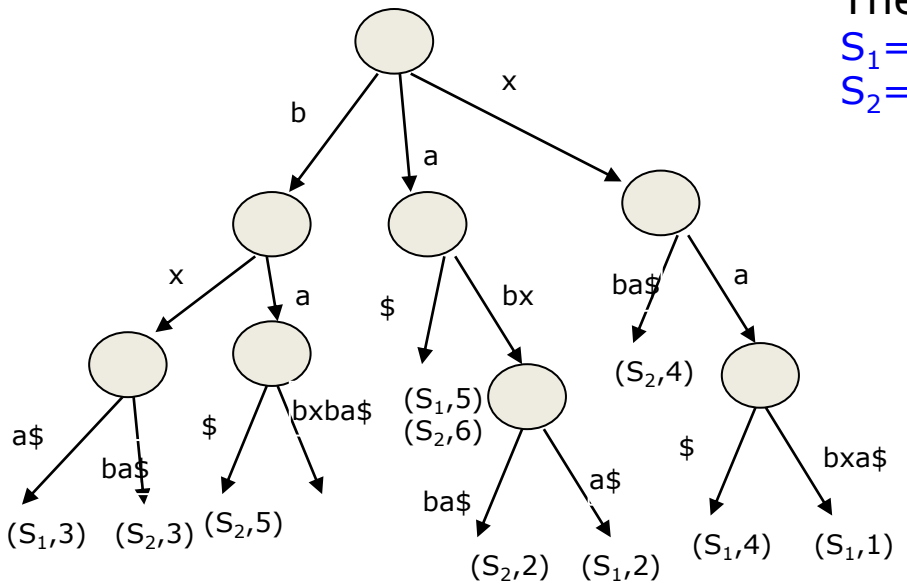


Suffix tree for bcabc$

Suffix trees (cont'd)

- ✓ Generalised Suffix Tree (GST)
  - Multistring Suffix Tree for *S*1, *S*2,…, *Sm*
  - Leaves can store labels for several strings
  - Can be built in $O(|S_1|+|S_2|+\cdots+|S_m|)$ time and space

The GST for
S_1=xabxa$
S_2=babxba$



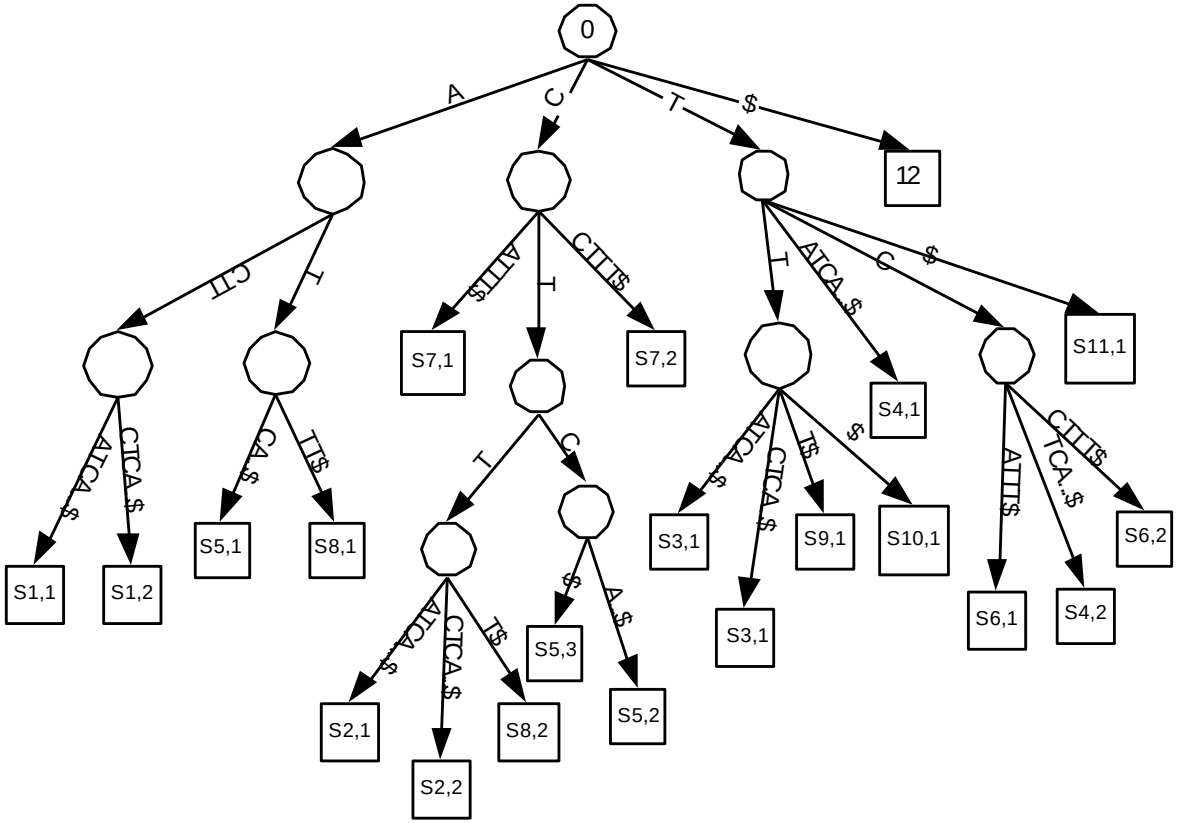
Weighted Suffix Tree

- ✓ The generalised suffix tree for all the subwords of a weighted sequence S, $|S| = n$, where $\Pr(S) \geq 1/k$, k a fixed parametre.
  - Leaf v labeled with a pair (i,j), for the subword $S_{i,j}$ (the j-th subword starting at position i)

1	2	3	4	5	6	7	8	9	10	11
A	C	T	T	(A, 0.5) (C, 0.5) (G, 0) (T, 0)	T	C	(A, 0.5) (C, 0.3) (G, 0) (T, 0.2)	T	T	T

$S_{1,1}$ = ACTTATCATTT\$, $S_{1,2}$ = ACTTCTCATTT\$, ... $S_{8,1}$= ATTT\$

An example


0
A
C
T
$
12
CT
T
ATC$
T
CTC$
ATCA$
CTCA$
CA$
T$
S7,1
S7,2
S1,1
S1,2
S5,1
S8,1
T
C
ATCA$
CTCA$
T$
$
A$
S2,1
S2,2
S8,2
S5,3
S5,2
T
ATCA$
C
$
S11,1
S4,1
ATCA$
CTCA$
T$
$
S3,1
S9,1
S10,1
S3,1
ATT$
TCA..$
CTT$
S6,2
S6,1
S4,2

Applications (1/4)

- ✓ Pattern Matching in weighted sequences, with Pr > $1/k$
- ✓ Build tree for S. Then as in ordinary suffix tree:
- ✓ Solid pattern P, P is spelled from the root of the tree. Stops at internal node. Report all leaves if necessary.
- ✓ Weighted pattern P, Break P into solid subwords and proceed as with solid patterns.

- ✓ Time: $O(m)$, $O(n)$ preprocessing $|S| = n$, $|P| = m$

Applications (2/4)

- ✓ Repeats in weighted sequences with Pr > 1/k for each.
  - Build WST for S with parameter 1/k.
  - Traverse the WST, in DFS. At the return step to an internal node v, build leaf-lists LL(v) from descendants.
  - LL(v)'s contents are positions where string Path-label(v) is repeated.

- ✓ Time: O(n+a) $|S| = n$, a = answer size

Applications (3/4)

- ✓ Longest Common Substring in weighted sequences with Pr > $1/k$
  - Build *Generalised Weighted Suffix Tree* for S_1, S_2.
  - Each internal node = a common substring
  - Find longest path label

- ✓ Time: $O(|S_1|+|S_2|)$.

Applications (4/4)

- ✓ Haplotype inference
- ✓ Indeterminate strings
- ✓ Degenerate strings

Computational Molecular Biology Goals

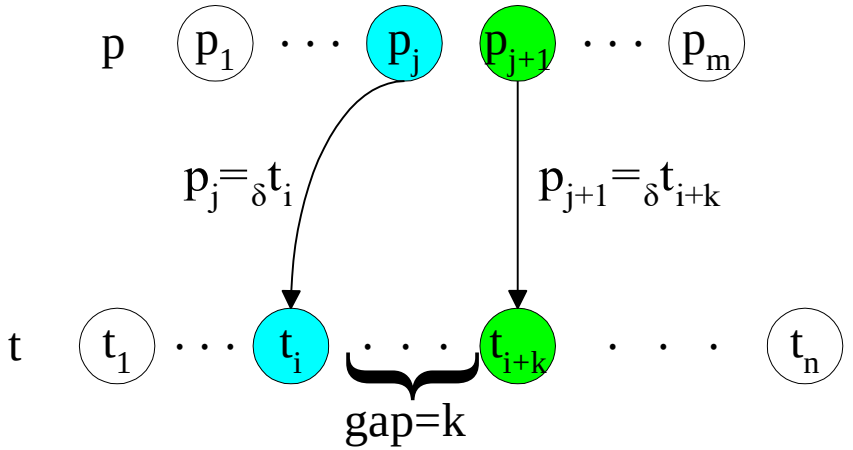
- Finding regularities in nucleic or protein sequences
- Finding features that are common to such sequences

Gene Expression and Regulation

- Match “structured patterns”
- Infer “structured patterns”

Approximate Matching

String Matching with Gaps: The occurrences of the symbols of pattern *p* do not appear successively but have gaps.



Definitions

✓Σ: Alphabet Σ*:set of all strings over Σ

✓Assume a, $b \in \Sigma$ and p (pattern), t (text) are strings over Σ.

✓Assume that $g_i = j_{i+1} - j_i - 1$ is the gap between the occurrences of symbols p_{i+1} and p_i that occur at positions j_{i+1} and j_i in text t.

1. $p = p_1, p_2, \ldots, p_m, (|p|=m)$
2. $a =_\delta b$ iff $|a-b| \leq \delta$
3. $p =_\delta t$ iff $p_i =_\delta t_i$ $1 \leq i \leq n$ (δ-approximate)
4. $p =_\gamma t$ iff $|p|=|t|$ and $\sum_{1 \leq i \leq |p|} |p_i - t_i| < \gamma$ (γ-approximate)

δ-approximate string matching with α-bounded gaps

✓*Problem:* We want to bound the gap between the *δ-occurrences of* p_i *and* p_{i+1} *in text t by* α.

✓*Basic Idea:* Compute the *δ-occurrences* of continuously increasing prefixes of *p* in *t*.

δ-approximate string matching with α-bounded gaps (the algorithm)

The basic structure is the $(m+1)\times(n+1)$ matrix D ($m=|p|$ & $n=|t|$):

$$D_{i,j}=\begin{cases} j, if\ (t_j =_\delta p_i)\ and\ (j - D_{i-1,j-1} \le \alpha+1)\ and\ (D_{i-1,j-1} > 0) \\ D_{i,j-1}, if\ (t_j \neq_\delta p_i)\ and\ (j - D_{i,j-1} < \alpha+1) \\ 0, otherwise \end{cases}$$

$$D_{0,0}=1,\ D_{i,0}=0,\ D_{0,j}=j$$

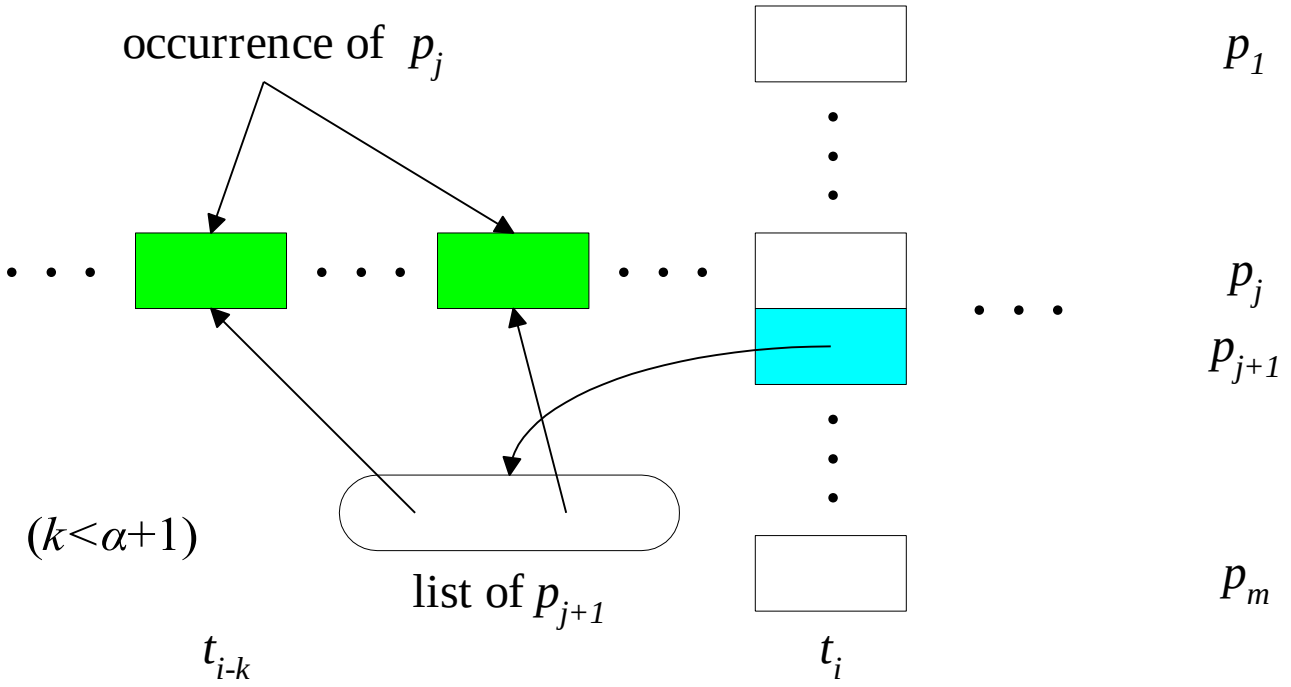
Example: t=acaeccaceaeeacbe (n=15) p=ace (m=3) (α=1, δ=1)

1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	1	3	3	0	6	6	0	9	9	0	12	12	0	0
0	0	2	2	0	5	5	7	7	0	0	0	0	13	14	14
0	0	0	0	4	0	0	0	8	0	0	0	0	0	0	15

(δ,γ)-approximate string matching with α-bounded gaps

Use matrix *D* combined with *min-FIFO queue* to keep track of the occurrences of the pattern symbols.

For each p_i we maintain a list (as we construct the matrix *D* column by column) that keeps all the occurrences of p_{i-1} for which the invariant of the bounded gap is not violated. We also need a matrix *C* with the costs of the occurrences.



Complexities

- ✓ For δ-approximate α-bounded gaps $O(mn)$ time complexity and $O(mn)$ space ($O(m)$ if we notice that for the computation of column i we only need column $i-1$).

- ✓ For (δ,γ)-approximate α-bounded gaps $O(mn)$ time complexity and $O(mn+m\alpha)$ space.

α-strict bounded gaps and unbounded gaps

✓*α-strict bounded gaps:* The gaps in this version are strictly of length α.

✓*Solution:* Rearrange text t so that symbols α far away become adjacent. The use a standard algorithm for δ-approximate matching (without gaps) is sufficient. Space and time complexity is $O(n)$.

unbounded gaps: The gaps in this version are unbounded. (we seek only one occurrence)

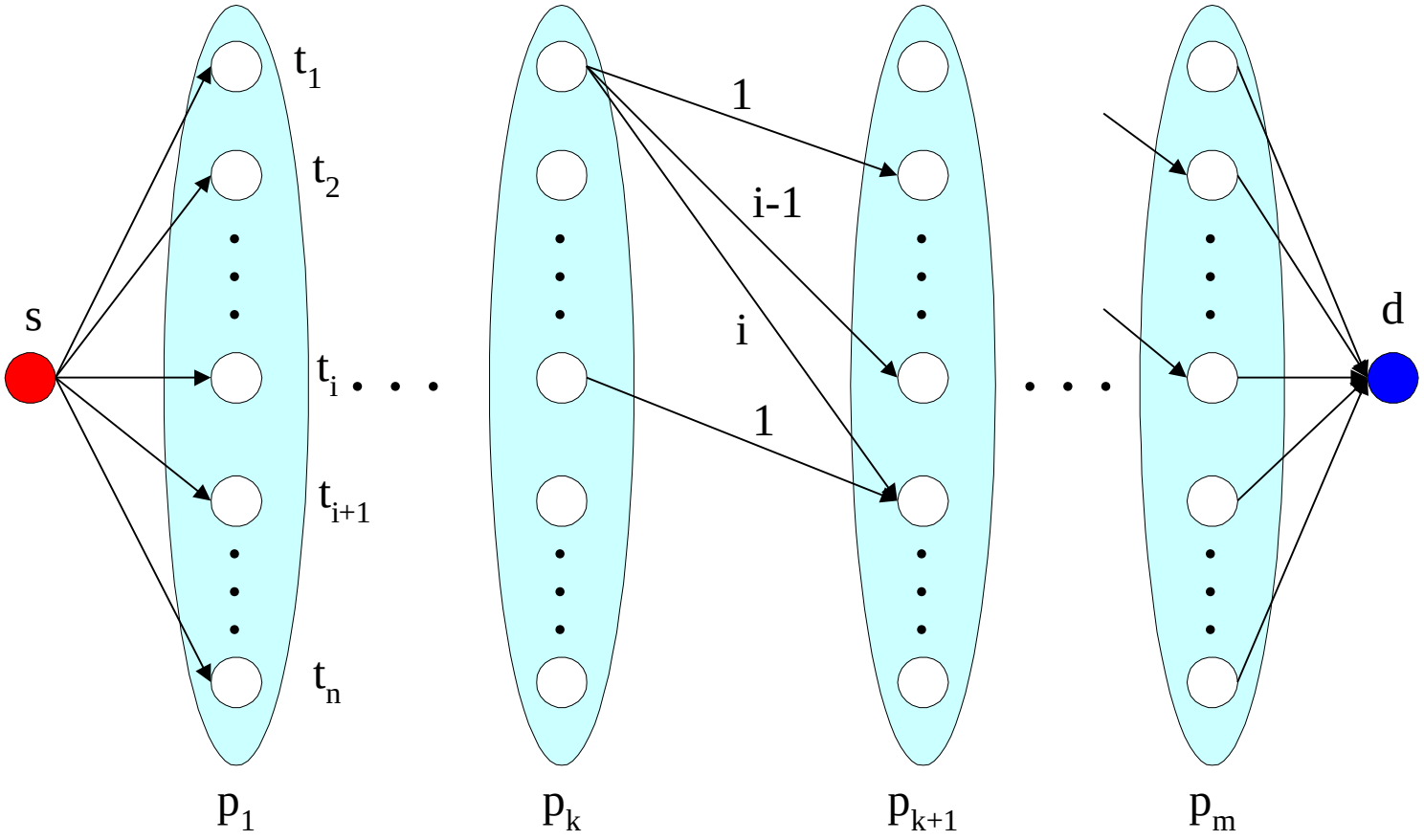
Solution: Just scan from left to right the string (time and space complexity is $O(n)$). If we want (δ,γ)-approximate matching then we have to resort to the algorithm for α-bounded gaps setting $\alpha=n+1$ or $\alpha=\infty$ (time and space complexity is $O(nm)$).

δ-occurrence minimizing total difference of gaps

We seek a δ-occurrence of p in t minimizing $\sum_{1\leq i\leq m-2} G_i$, where $Gi=|g_i-g_{i+1}|$. We reduce this minimization problem to the shortest path problem on a graph:

1. Construct graph $H=(V,E)$. The set of nodes V is constructed by creating nodes $v_{i,j}$ ($1\leq i\leq m$, $1\leq j\leq n$) whenever $p_i=_\delta t_j$. An edge exists between $v_{i,j}$ and $v_{i',j'}$ if $i'=i+1$ and $j'>j$. This edge has weight equal to $j'-j-1$. These edges encode the occurrences of the pattern p in t. Link node s to all nodes $v_{1,j}$ and node d to all nodes $v_{m,j}$.
2. By contracting two nodes connected by an edge in a single node we get the graph H' that encodes the differences of consecutive gaps. The shortest path from s to d gives us the appropriate occurrence of p in t.

δ-occurrence minimizing total difference of gaps (an example)



The time and space complexity of this algorithm is $O(n^2m)$.

δ-occurrence with ε-bounded difference gaps

✓*Problem:* We seek a *δ*-occurrence of *p* in *t* such that $G_i = |g_i - g_{i+1}| < \varepsilon$.

✓*Solution:* Make use of graph *H´* with the difference that we need not find the shortest path but just to find a path from *s* to *d (after removing all the edges with weight* .

The time and the space complexity is equal to $O(n^2m)$.

δ-occurrence of a set of strings with Δ-bounded gaps

✓*Problem:* Assume $w_1, \ldots, w_m \in \Sigma^*$. We wish to find δ-occurrences of w_i (without gaps) where the gaps between consecutive occurrences of strings w_i *and* w_{i+1} *are bounded by* Δ.

✓*Solution:* Define $p=w_1w_2\ldots w_m$. Then we abstract each w_i as a single character and continue as in α-bounded gaps with the construction of matrix D. The space and time complexity is $O(n(|w_1|+|w_2|+\ldots+|w_m|))$.

Τέλος Ενότητας

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Πατρών**» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημειώματα

Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση 1.0.

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.