Introduction to

Information Retrieval

ΜΥΕ003-ΠΛΕ70: Ανάκτηση Πληροφορίας

Διδάσκουσα: Ευαγγελία Πιτουρά

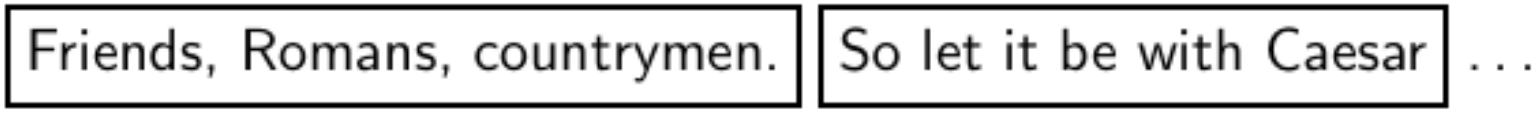
Διάλεξη 3: Δομές για Λεξικά. Ανάκτηση Ανεκτική στα Σφάλματα.

Επανάληψη προηγούμενης διάλεξης

1. Προ-επεξεργασία εγγράφων της συλλογής για την κατασκευή του αντεστραμμένου ευρετηρίου
2. Πιο γρήγορες λίστες καταχώρησης με λίστες παράλειψης
3. Υποστήριξη ερωτημάτων φράσεων (phrase queries) και θέσης (positional queries)

1. Προσδιορισμός Λεξιλογίου όρων

1. Συλλέγουμε τα έγγραφα για τα οποία θα κατασκευαστεί το ευρετήριο



2. "Tokenize" το κείμενο, αποτέλεσμα: μια λίστα από tokens:

3. Γλωσσική επεξεργασία ώστε να παραχθεί μια λίστα από κανονικοποιημένα tokens που θα είναι οι όροι που εισαχθούν στο ευρετήριο

4. Κατασκευή αντεστραμμένου ευρετηρίου

1. Προσδιορισμός Λεξιλογίου όρων

- Token – η εμφάνιση μια λέξης ή ενός όρου σε ένα έγγραφο
- Type (τύπος) – μια κλάση ισοδυναμίας από tokens
  - *Παράδειγμα: In June, the dog likes to chase the cat in the barn.*
  - 12 word tokens, 9 word types

Tokenization - Προβλήματα

- Ποια είναι τα διαχωριστικά (κενό, απόστροφος, ενωτικά, κλπ)

1. Προσδιορισμός Λεξιλογίου όρων

Από τύπους ισοδύναμων tokens σε όρους (λήμμα) που θα εισαχθούν στο ευρετήριο

Θέματα

- Λημματοποίηση (lemmatization) ή Περιστολή (stemming)
- Αριθμοί, Κεφαλαία/Μικρά, Stop words;
- Κλάσεις ισοδύναμων όρων (για συνώνυμα) κατά την επεξεργασία του ερωτήματος ή στο ευρετήριο

1. Προσδιορισμός Λεξιλογίου όρων

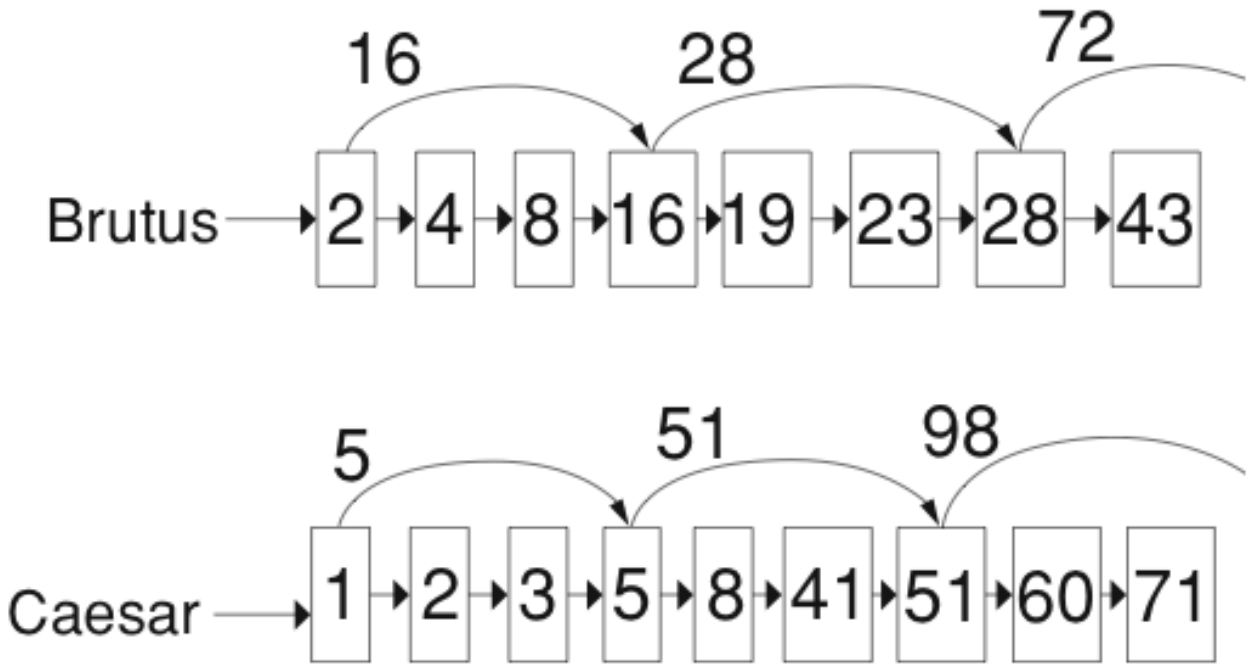
New York City is called Big Apple.

She likes big apples.

Elvis has just left the building.

It is on your left, as you enter the building.

2. Δείκτες παράλειψης



3. Ερωτήματα Φράσεων και Θέσης

- Ευρετήρια biword για ερωτήματα φράσεων
- Ευρετήρια Θέσης (positional indexes) για ερωτήματα φράσεων και εγγύτητας

3. Ερωτήματα Φράσεων και Θέσης

- Στις λίστες καταχωρήσεων σε ένα nonpositional ευρετήριο, κάθε καταχώρηση είναι μόνο ένα docID
- Στις λίστες καταχωρήσεων σε ένα positional ευρετήριο, κάθε καταχώρηση είναι ένα docID και μια λίστα από θέσεις
- Παράδειγμα ερωτήματος: *"to*$_1$ *be*$_2$ *or*$_3$ *not*$_4$ *to*$_5$ *be*$_6$*"*

TO, 993427:

‹ 1: ‹7, 18, 33, 72, 86, 231›;
2: ‹1, 17, 74, 222, 255›;
4: ‹8, 16, 190, 429, 433›;
5: ‹363, 367›;
7: ‹13, 23, 191›; . . . ›

BE, 178239:

‹ 1: ‹17, 25›;
4: ‹17, 191, 291, 430, 434›;
5: ‹14, 19, 101›; . . . ›

Τι θα δούμε σήμερα;

- Δομές δεδομένων για λεξικά
- Ανάκτηση ανεκτική (tolerant) σε σφάλματα
  - Ερωτήματα με wild-card («χαρακτήρων μπαλαντέρ»)*
  - Ορθογραφικά λάθη
  - Απόσταση μεταξύ όρων
  - Φωνητική διόρθωση

Δομές Δεδομένων για Λεξικά

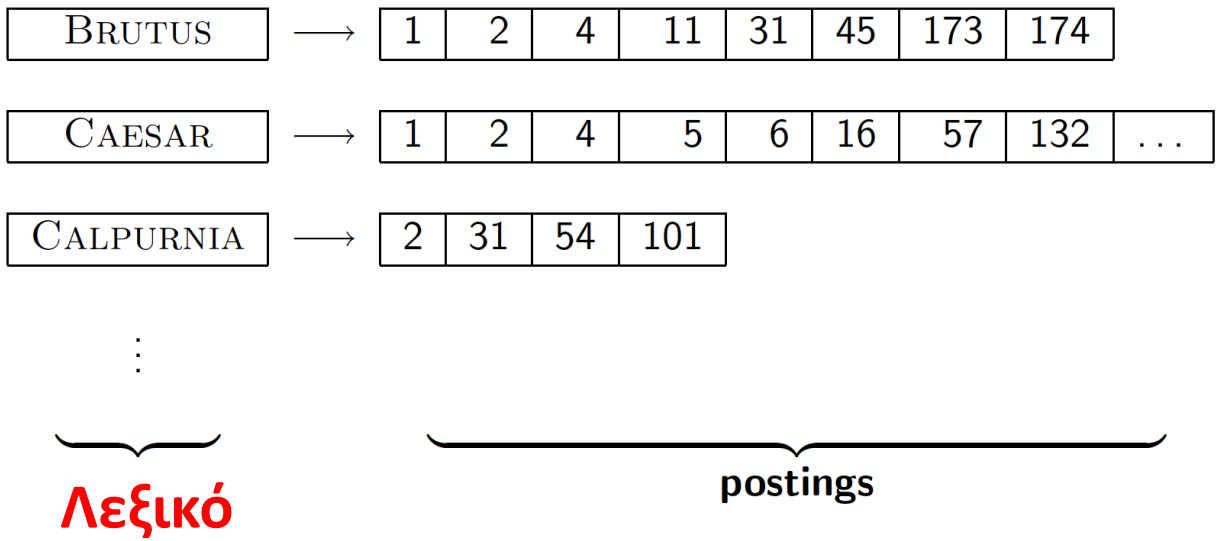
Λεξιλόγιο (vocabulary): το σύνολο των όρων

Λεξικό (dictionary): μια δομή για την αποθήκευση του λεξιλογίου

Πως αποθηκεύουμε ένα λεξικό (στη μνήμη) αποδοτικά;

Δομές Δεδομένων για Λεξικά

- Το λεξικό περιέχει: το *λεξιλόγιο όρων (λήμμα)*, για κάθε όρο: τη συχνότητα εγγράφου (document frequency) και δείκτη στη λίστα καταχωρήσεων
- Σε κάθε ερώτημα, αναζήτηση στο λεξικό αν υπάρχει ο όρος και σε ποια έγγραφα



Ποια δομή δεδομένων είναι κατάλληλη;

Μια απλοϊκή λύση

- array of struct:

term	document frequency	pointer to postings list
a	656,265	⟶
aachen	65	⟶
. . .	. . .	. . .
zulu	221	⟶

char[20] int Postings *

20 bytes 4/8 bytes 4/8 bytes

- Πως αναζητούμε έναν όρο (κλειδί, key) στο λεξικό γρήγορα κατά την εκτέλεση του ερωτήματος;

Δομές δεδομένων για το Λεξικό

- Δυο βασικές επιλογές:
  - Πίνακες Κατακερματισμού (Hashtables)
  - Δέντρα (Trees)
- Μερικά συστήματα ανάκτησης πληροφορίας χρησιμοποιούν πίνακες κατακερματισμού άλλα δέντρα

Δομές Δεδομένων για Λεξικά

Κριτήρια για την επιλογή δομής:

- Αποδοτική αναζήτηση ενός όρου (κλειδιού) στο λεξικό.
- Σχετικές συχνότητες προσπέλασης των κλειδιών (πιο γρήγορα οι συχνοί όροι;)
- Πόσοι είναι οι όροι
- Είναι στατικό ή έχουμε συχνά εισαγωγές/διαγραφές όρων ή και τροποποιήσεις; Μόνο εισαγωγές (insert only – append only)

Πίνακες Κατακερματισμού

Κάθε όρος του λεξιλογίου κατακερματίζεται σε έναν ακέραιο

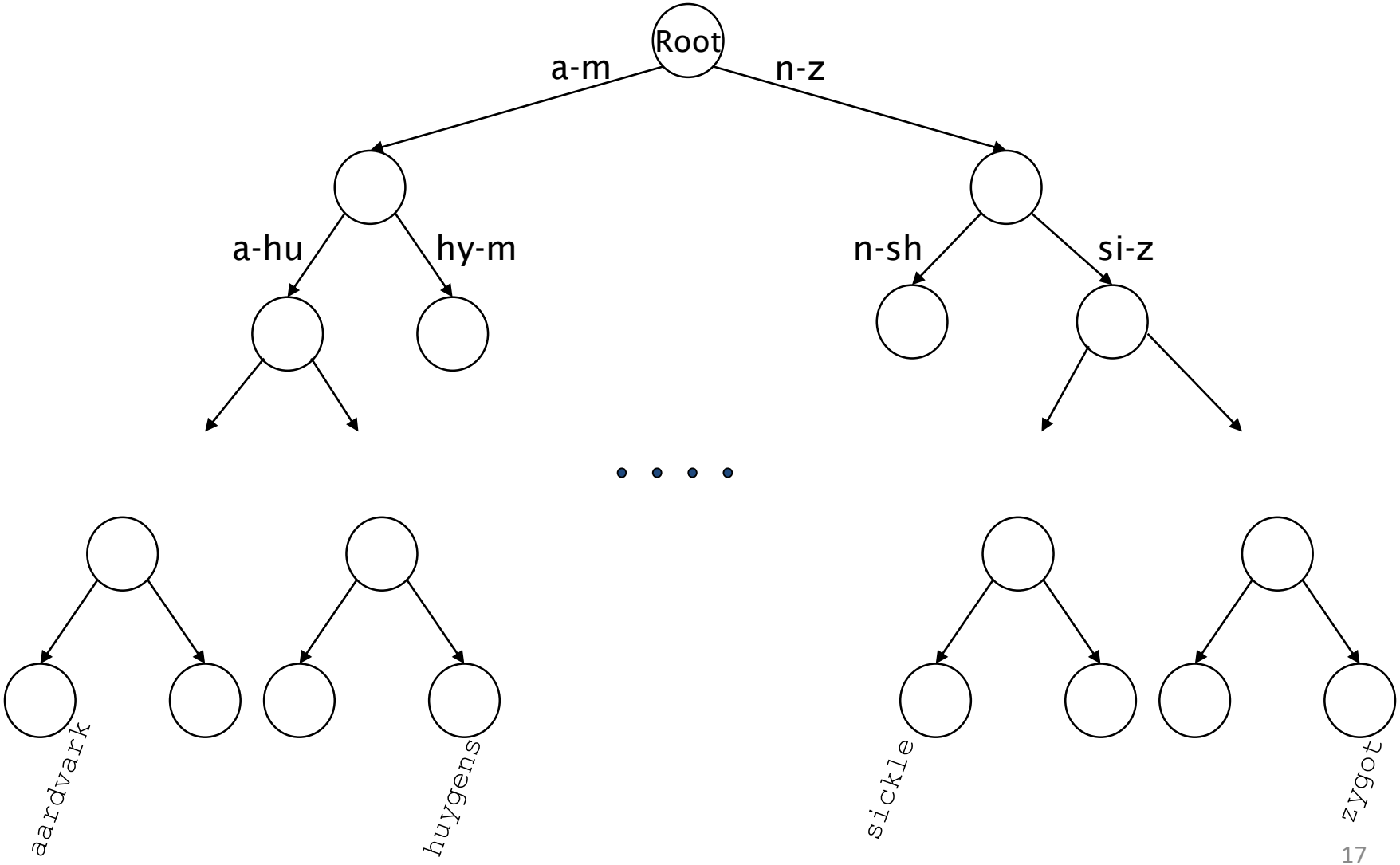
+:

- Η αναζήτηση είναι πιο γρήγορη από ένα δέντρο: O(1)

- :

- Δεν υπάρχει εύκολος τρόπος να βρεθούν μικρές παραλλαγές ενός όρου
  - judgment/judgement, *resume* vs. *résumé*
- Μη δυνατή η προθεματική αναζήτηση [ανεκτική ανάκληση]
- Αν το λεξιλόγιο μεγαλώνει συνεχώς, ανάγκη για να γίνει κατακερματισμός από την αρχή

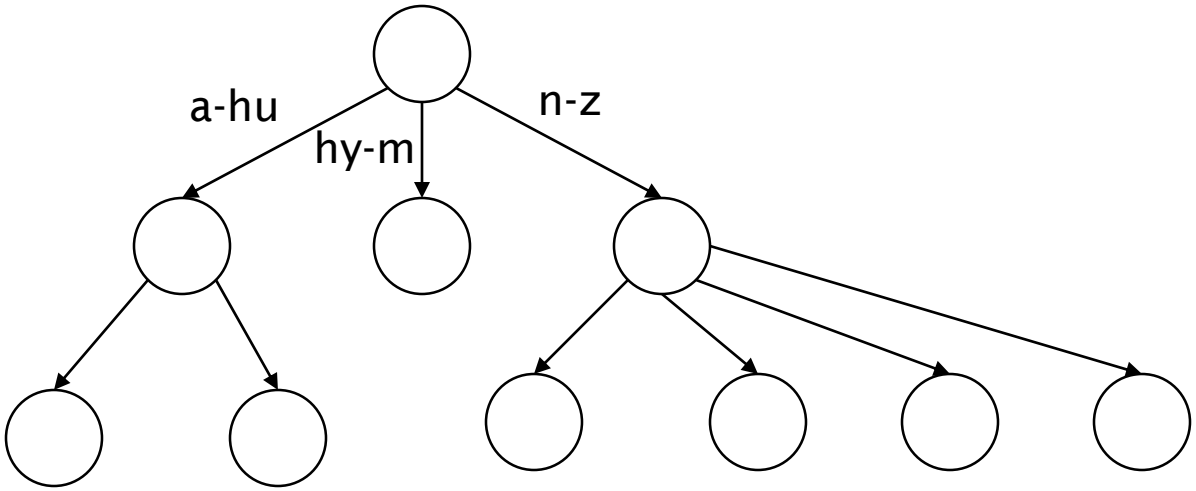
Δέντρα αναζήτησης: Δυαδικό δέντρο



Δέντρα αναζήτησης: Δυαδικό δέντρο

- O(log M), M: αριθμός των όρων (το μέγεθος του λεξικού)
  - προϋποθέτει ισοζύγιση

Δέντρα: B-δέντρα



Ορισμός: Κάθε εσωτερικός κόμβος έχει έναν αριθμό από παιδιά στο διάστημα [*a*, *b*] όπου *a*, *b* είναι κατάλληλοι φυσικοί αριθμοί, π.χ., [2, 4]

Δέντρα

- Το απλούστερο: δυαδικό δέντρο
- Το πιο συνηθισμένο: B-δέντρα
- Τα δέντρα απαιτούν ένα δεδομένο τρόπο διάταξης των χαρακτήρων (αλλά συνήθως υπάρχει ή μπορεί να οριστεί)

+:

- Λύνουν το πρόβλημα προθέματος (π.χ., όροι που αρχίζουν με *hyp*)
- Πλεονεκτούν όταν το λεξικό αποθηκεύεται στο δίσκο (τότε τα *a* και *b* καθορίζονται από το μέγεθος του block)

-:

- Πιο αργή: O(log *M*) [και αυτό απαιτεί (*ισοζυγισμένα* (*balanced)* δέντρα)
- Η επανα-ισοζύγιση (rebalancing) των δυαδικών δέντρων είναι ακριβή
  - Αλλά τα B-δέντρα καλύτερα

Λεξικά (σύνοψη)

Λεξιλόγιο (vocabulary): το σύνολο των όρων

Λεξικό (dictionary): μια δομή για την αποθήκευση του λεξιλογίου

- Δυο βασικές επιλογές:
  - Πίνακες Κατακερματισμού (Hashtables)
  - Δέντρα (Trees)

- Μερικά συστήματα ανάκτησης πληροφορίας χρησιμοποιούν πίνακες κατακερματισμού άλλα δέντρα

ΕΡΩΤΗΜΑΤΑ ΜΕ *

Ερωτήματα με wild-card (*)

(1) Δεν είμαστε σίγουροι για την ορθογραφία της λέξης (πχ Sydney? Sidney?)

(2) Υπάρχουν πολλαπλές εκδοχές της ορθογραφίας της λέξης και θέλουμε να ανακτήσουμε τα έγγραφα που περιέχουν οποιαδήποτε από αυτές (πχ color, colour)

(3) Δεν είμαστε σίγουροι αν έχει γίνει stemming

(4) Ορθογραφία ξένης λέξης (Σ*ξπ*ρ)

Ερωτήματα με wild-card (*)

Trailing wild card query: το "*" εμφανίζεται μόνο μια φορά στο τέλος (*prefix* query, *πρόθημα*)

- Π.χ., ***mon*:** Βρες όλα τα έγγραφα που περιέχουν οποιαδήποτε λέξη *αρχίζει με "mon"*
  - Εύκολο όταν το λεξικό με δυαδικό δέντρο (ή B-δέντρο):
    - ανάκτησε όλους τους όρους t στο διάστημα: ***mon ≤ t < moo***
      - Για κάθε όρο, αναζήτησε το αντεστραμμένο ευρετήριο σε ποια έγγραφα εμφανίζεται

Ερωτήματα με Wild-card (*)

Leading wild card queries: το "*" εμφανίζεται μόνο μια φορά στην αρχή (*postfix* query, *επίθημα*)

- ***Π.χ., *mon:*** Βρες όλα τα έγγραφα που περιέχουν οποιαδήποτε λέξη τελειώνει σε "mon"
- *πιο δύσκολο*
  - Διατήρησε ένα επιπρόσθετο B-tree για τους όρους ανάποδα (*backwards), πχ ο όρος demon -> nomed*
    - *Reverse B-tree*
  - Ανάκτησε όλους τους όρους t στο διάστημα: ***nom ≤ t < non***.

Ερωτήματα με Wild-card (*)

Πως μπορούμε να απαντήσουμε ερωτήσεις με ένα * στη μέση της λέξης, π.χ., ***pro*cent*** ?

Όροι με πρόθημα pro
Όροι με επίθημα cent
Διατρέχουμε τους όρους που ανήκουν στην τομή και απορρίπτουμε όσους ταιριάζουν και με το πρόθημα και με το επίθημα (αρκεί; ba*ba και όρος ba?)

Γενικά ερωτήματα με *

- * στη μέση του όρου
  - ***co*tion***
- Αναζήτησε το ***co**** AND ****tion*** σε ένα B-tree και ένα reverse και υπολόγισε την τομή των συνόλων
  - Ακριβό!
- Δύο λύσεις – και οι δύο:
  I. Έχουν ένα *ειδικό ευρετήριο*
  II. Μετατρέπουν την αρχική ερώτησης q σε μια Boolean ερώτηση Q στο ειδικό ευρετήριο έτσι ώστε η απάντηση στο Q να είναι *υπερσύνολο* της απάντησης στο q
  III. Στη συνέχεια ελέγχουν τις απαντήσεις

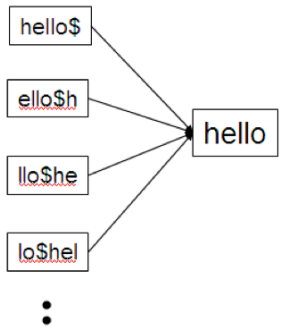
Γενικά ερωτήματα με *

- Πρώτη εναλλακτική λύση: Μετάτρεψε τις ερωτήσεις έτσι ώστε τα * να εμφανίζονται στο τέλος

Permuterm Index (ευρετήριο αντιμετατεθειμένων όρων)

Ευρετήριο αντιμετατεθειμένων όρων

Κατασκευάζουμε ένα ευρετήριο αντιμετατεθειμένων όρων (permuter index) στο οποίο όλες οι παραλλαγές που προκύπτουν από την (δεξιόστροφη) περιστροφή (rotation) του όρου συνδέονται με τον αρχικό όρο



Πχ. για τον όρο **hello -> hello$**, εισάγουμε στο *λεξικό* τα:

- **hello$, $hello, o$hell , lohel , llohe, ello$h**
  - όπου $ ένα ειδικός χαρακτήρας που σηματοδοτεί το τέλος μιας λέξης
  - Permuterm vocabulary

Περιστροφή (rotation) του όρου του ερωτήματος ώστε το * στο τέλος

Π.χ., Ερώτημα he*lo -> he*lo$ -> lo$he*

Ψάχνουμε το lo$hel* (ως prefix queries στο petmuterm ευρετήριο -- το permuterm ευρετήριο επίσης κάποιο δέντρο αναζήτησης)

Ευρετήριο αντιμετατεθειμένων όρων

Παράδειγμα

Ευρετήριο αντιμετατεθειμένων όρων για moron, man

Εισάγουμε *στο λεξικό* όλες τις περιστροφές των όρων ώστε να δείχνουν στον αρχικό όρο

moron -> moron$ -> στο λεξικό: moron$, $moron, n$moro, on$mor ron$mo oron$m

man -> man$ -> στο λεξικό: man$,man, nma, an$m

Ερώτημα m*n -> m*n$ -> n$m*

Ερώτημα: mo*n -> n$mo*

Match?

Ερώτημα: m* -> $m*

Match?

Ευρετήριο αντιμετατεθειμένων όρων

- **X*Y*Z** πως γίνεται match?
  - X*Y*Z$ -> Z$X*
  - Ψάξε Z$X* και μετά έλεγξε κάθε υποψήφιο όρο για το Y
  - Πχ fi*mo*er -> ψάξε er$fi*, έλεγξε αν και mo (π.χ., fishmonger και fillbuster)
- Στην πραγματικότητα, permuterm B-tree για τους αντιμετατεθειμένους όρους
- *Πρόβλημα: ≈ δεκαπλασιάζει το μέγεθος του λεξικού*

Εμπειρική παρατήρηση για τα Αγγλικά

Ασκήσεις

- Παράδειγμα permuterm όροι για baba, ba και αποτέλεσμα των ba*ba και ba*
- Δείξτε πως το ερώτημα s*ng δίνει και spring με χρήση permuterm index
- Στο ευρετήριο αντιμεταθειμένων όρων, κάθε αντιμεταθειμένος όρος δείχνει στους αρχικούς όρους λεξιλογίου από τους οποίους προέκυψε. Πόσοι όροι του πρωτότυπου λεξιλογίου μπορούν να υπάρχουν στη λίστα καταχωρήσεων ενός αντιμετατεθειμένου όρου;

Ευρετήρια *k*-γραμμάτων (*k*-gram indexes)

k-gram: ακολουθία *k* χαρακτήρων

- Απαρίθμησε όλα τα *k*-γράμματα που εμφανίζονται σε κάθε όρο
  - *π.χ.,* για το κείμενο "***April is the cruelest month***" έχουμε τα 2-γράμματα (*bigrams*)

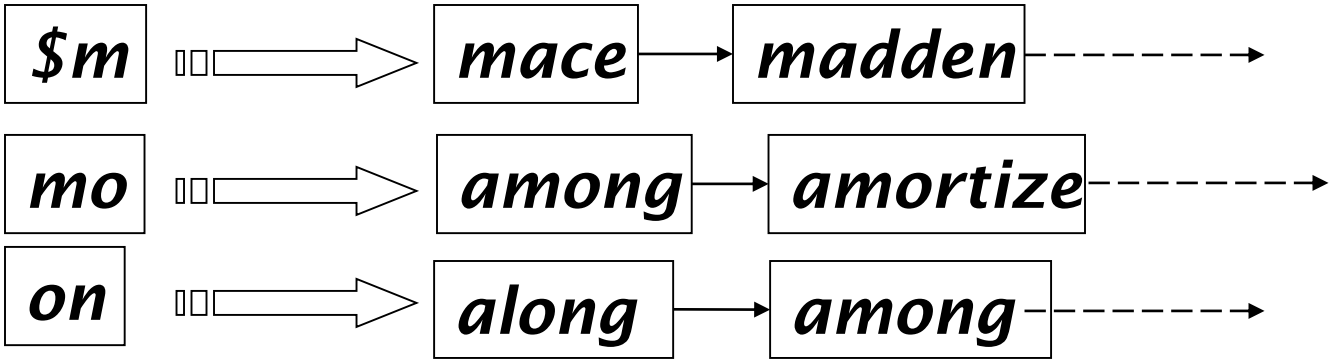
a,ap,pr,ri,il,l,i,is,s,t,th,he,e,$c,cr,ru,
ue,el,le,es,st,t$, m,mo,on,nt,h

  - Όπου $ ένα ειδικός χαρακτήρας που σηματοδοτεί το τέλος και την αρχή μιας λέξης

- Σε ένα k-gram ευρετήριο, το λεξικό περιέχει όλα τα *k*-grams που εμφανίζονται σε οποιονδήποτε όρο
- Διατήρησε ένα *δεύτερο αντεστραμμένο ευρετήριο* από τα 2-γράμματα στους *όρους του λεξικού που τα περιέχουν*

Παράδειγμα ευρετηρίου *k*-γραμμάτων

- Το ευρετήριο *k*-γραμμάτων βρίσκει τους όρους βασισμένο σε μια ερώτηση που αποτελείται από *k*-γράμματα (εδώ *k*=2).



$k = 3$

Επεξεργασία ερωτημάτων

- Ερώτημα ***mon**** τώρα γίνεται

  $m *AND* ***mo*** *AND* ***on***

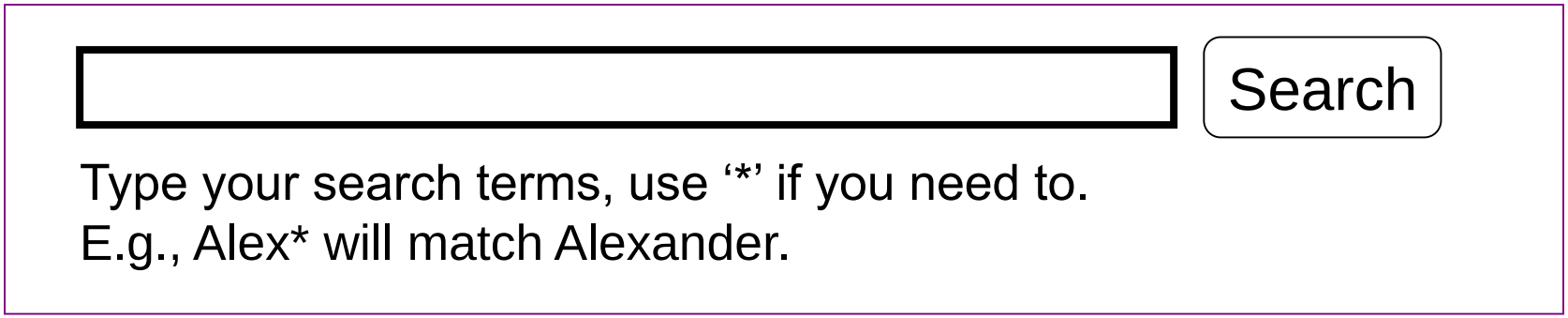
  - Βρίσκει τους όρους που ταιριάζουν μια AND εκδοχή του wildcard ερωτήματος.
- Απαιτείται βήμα μετά-φιλτραρίσματος (post-filter)
  - False positive, π.χ., moon
- Οι όροι που απομένουν αναζητούνται στο γνωστό αντεστραμμένο ευρετήριο όρων-εγγράφων

Άσκηση

- Γιατί οι λίστες καταχωρήσεων σε ένα ευρετήριο k-γραμμάτων είναι διατεταγμένες;

Επεξεργασία ερωτημάτων

- Ένα Boolean ερώτημα για κάθε όρο
- Μπορεί να οδηγήσουν σε ακριβή επεξεργασία ερωτημάτων
  - pyth* AND prog*
- Αν ενθαρρύνουμε την “τεμπελιά” οι άνθρωποι θα ανταποκριθούν!



- Ποιες μηχανές αναζήτησης επιτρέπουν τέτοια ερωτήματα; *(παλιότερα, altavista)*

ΔΙΟΡΘΩΣΗ ΟΡΘΟΓΡΑΦΙΚΩΝ ΛΑΘΩΝ

Διόρθωση ορθογραφικών λαθών

- Δύο βασικές χρήσεις
  - Διόρθωση των *εγγράφων* που ευρετηριοποιούνται
  - Διόρθωση των *ερωτημάτων* ώστε να ανακτηθούν «σωστές» απαντήσεις
- Δυο βασικές κατηγορίες:
  - *Μεμονωμένες λέξεις (isolated term)*
    - Εξέτασε κάθε λέξη μόνη της για λάθη
    - Δεν πιάνει typos που έχουν ως αποτέλεσμα σωστά γραμμένες λέξεις
      - π.χ., ***from*** $\rightarrow$ ***form***
  - Βασισμένη *σε συμφραζόμενα* (context sensitive)
    - Κοιτά τις λέξεις γύρω,
      - π.χ., ***I flew form Heathrow to Narita.***

Διόρθωση εγγράφων

- Χρήσιμη ιδιαίτερα για έγγραφα μετά από OCR
  - Αλγόριθμοι διόρθωσης ρυθμισμένοι για αυτό: rn μοιάζει με m
  - Μπορεί να χρησιμοποιούν ειδική γνώση (domain-specific)
    - Π.χ., OCR μπερδεύει το O με το D πιο συχνά από το O και το I (που είναι γειτονικά στα QWERTY πληκτρολόγιο), οπότε πιο πιθανή η ανταλλαγή τους στην πληκτρολόγηση
- Αλλά συχνά: web σελίδες αλλά και τυπωμένο υλικό έχουν typos
- Στόχος: το λεξικό να περιέχει λιγότερα ορθογραφικά λάθη
  - Αλλά συχνά δεν αλλάζουμε τα έγγραφα αλλά επεκτείνουμε την απεικόνιση ερωτήματος – εγγράφου

Διόρθωση μεμονωμένης λέξης

- Θεμελιώδης υπόθεση – *υπάρχει ένα λεξικό που μας δίνει τη σωστή ορθογραφία*
- Δυο βασικές επιλογές για αυτό το λεξικό
  - Ένα standard λεξικό όπως
    - Webster's English Dictionary
    - Ένα "industry-specific" λεξικό – hand-maintained
  - Το *λεξικό της συλλογής* (corpus)
    - Π.χ., όλες οι λέξεις στο web
    - Όλα τα ονόματα, ακρώνυμα κλπ.
    - (συμπεριλαμβανομένων και των ορθογραφικών λαθών)

Γενικά θέματα

(1) Στο ερώτημα carot πάντα επέστρεψε τα έγγραφα που περιέχουν το carot καθώς και τα έγγραφα με όλες τις διορθωμένες εκδοχές του όρου, πχ carrot and tarot.

(2) Όπως στο (1) , αλλά διορθώσεις μόνο αν το carot δεν είναι στο λεξικό

(3) Όπως στο (1), αλλά μόνο αν η αρχική ερώτηση επιστρέφει λίγα (πχ λιγότερο από 5) έγγραφα.

(4) Όταν η αρχική ερώτηση επιστρέφει λιγότερα από έναν προκαθορισμένο αριθμό από έγγραφα επιστρέφει «*spelling suggestions*" : "Did you mean carrot?" (και όχι επιπρόσθετα έγγραφα)

Γενικά θέματα

(1) Επιστρέφουμε τη λέξη (λέξεις) που είναι πιο «κοντά»

(2) Όταν ισοπαλία

(1) Την πιο συχνή (συχνές) στη συλλογή

(2) Την πιο συχνή στα ερωτήματα

- Δείτε στο http://www.netpaths.net/blog/britney-spears-spelling-variations/ (αρχικά εδώ http://www.google.com/jobs/archive/britney.html) στατιστικά για misspellings του Britney Spears

- Ένας απλός spell corrector σε Python
  http://norvig.com/spell-correct.html

Διόρθωση μεμονωμένης λέξης

Δοθέντος ενός **λεξικού** και μιας ακολουθίας χαρακτήρων Q, επέστρεψε τις λέξεις του λεξικού που είναι ***πιο κοντά*** στο Q

- Τι σημαίνει "πιο κοντά"?
- Θα εξετάσουμε δύο ορισμούς εγγύτητας:
  - Την *απόσταση διόρθωσης -- edit distance* (Levenshtein distance) και την *σταθμισμένη* εκδοχή της -- weighted edit distance
  - *Επικάλυψη (overlap) n-γραμμάτων*

Απόσταση διόρθωσης (Edit distance)

ΟΡΙΣΜΟΣ: Δοθέντων δυο αλφαριθμητικών (strings) S_1 and S_2, ο ελάχιστος αριθμός πράξεων για τη μετατροπή του ενός στο άλλο

- Συνήθως, οι πράξεις είναι σε επίπεδο χαρακτήρα
  - **Levenshtein distance:** (1) Insert – Εισαγωγή, (2) Delete - Διαγραφή και (3) Replace – Αντικατάσταση ενός χαρακτήρα (μερικές φορές, κόστος 2 (ως delete-insert)
  - **Damerau-Levenshtein distance:** + Transposition - Αντιμετάθεση ενός χαρακτήρα
- Π.χ., η απόσταση διόρθωσης από ***dof*** σε ***dog*** είναι 1
  - Από ***cat*** σε ***act*** είναι 2 (Μόνο 1 με αντιμετάθεση)
  - Από ***cat*** σε ***dog*** είναι 3.

Απόσταση Διόρθωσης (Edit distance)

Παράδειγμα

Levenshtein distance: *dog-do*: 1, *cat-cart*: 1, *cat-cut*: 1, *cat-act*: 2

Damerau-Levenshtein distance: cat-act: 1

- Γενικά υπολογίζεται με Δυναμικό Προγραμματισμό.
- Κοιτάξτε πχ το http://www.let.rug.nl/kleiweg/lev/ για παραδείγματα.

Δυναμικός προγραμματισμός

- Εκφράζουμε το πρόβλημα ως συνδυασμό υπό-προβλημάτων – η βέλτιστη λύση βασίζεται στη βέλτιστη λύση του υπό-πρόβληματος
  - Στην περίπτωση των αποστάσεων διόρθωσης – το υπό-πρόβλημα δυο προθημάτων:

    Ο βέλτιστος τρόπος από μια λέξη σε μια άλλη, βασίζεται στο βέλτιστο τρόπο από κάποιο πρόθημα της πρώτης σε πρόθημα της δεύτερης
- Έναν Πίνακα
  - Γραμμές: Γράμματα (προθήματα) της πρώτη λέξης
  - Στήλες: Γράμματα (προθήματα) της δεύτερης λέξης
  - Θέσεις του πίνακα: βέλτιστο κόστος (απόσταση)

Υπολογισμός απόστασης διόρθωσης

String s_2

String s_1

		f	a	s	t
	0	1	2	3	4
c	1	1	2	3	4
a	2	2	1	2	3
t	3	3	2	2	2
s	4	4	3	2	3

cats – fast

Κάθε στοιχείο m[i, j] του πίνακα μας δίνει το βέλτιστο κόστος (απόσταση) για να πάμε από το πρόθεμα μήκους **i** του s_1 στο πρόθεμα μήκους **j** του s_2

Δυναμικός προγραμματισμός

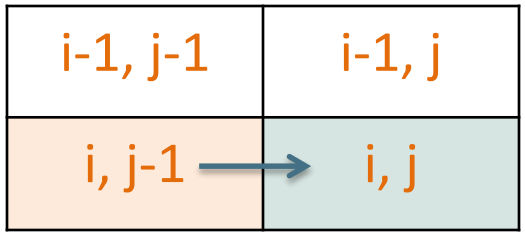
- ✓ Πως υπολογίζουμε τα στοιχεία του πίνακα;
- ✓ Επικαλυπτόμενες υπό-λύσεις:
  - ✓ Βέλτιστο κόστος m[i, j]
    - ✓ Πχ m[2, 3] ca -> fas
    - ✓ 3 διαφορετικοί τρόποι
      - ✓ m[i, j-1] από αριστερά (γραμμή)
      - ✓ m[i-1, j] από πάνω (στήλη)
      - ✓ m[i-1, j-1] (διαγώνια)

i-1, j-1	i-1, j
i, j-1	i, j

Αλγόριθμος (από αριστερά)

LEVENSHTEINDISTANCE(s_1, s_2)

```
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]}
 9            else  m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1}
10  return m[|s1|, |s2|]
```



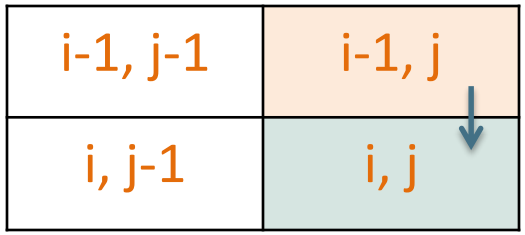
Operations: insert (cost 1), delete (cost 1), replace (cost 1), copy (cost 0)

Αλγόριθμος (από πάνω)

LEVENSHTEINDISTANCE(s_1, s_2)

```
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]}
 9            else  m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1}
10  return m[|s1|, |s2|]
```

Operations: insert (cost 1), delete (cost 1), replace (cost 1), copy (cost 0)



Υπολογισμός απόστασης διόρθωσης

```
LevenshteinDistance(s1, s2)
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min
 9            else  m[i, j] = min
10  return m[|s1|, |s2|]
```

Operations: insert (cost 1), delet (cost 0)

Αρχικοποίηση

String s_2

String s_1

		f	a	s	t
	0	1	2	3	4
c	1	1	2	3	4
a	2	2	1	2	3
t	3	3	2	2	2
s	4	4	3	2	3

Κόστος διόρθωσης για τα προθέματα

Υπολογισμός απόστασης Levenshtein

Για να υπολογίσουμε το m[i, j]

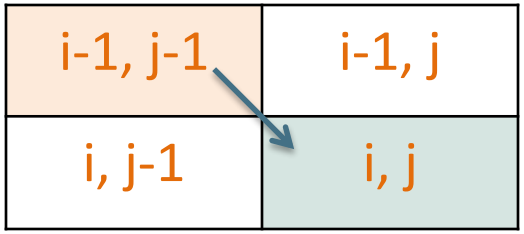
	j -1	j
i -1	Εξαρτάται από το επόμενο γράμμα	Κόστος από τον πάνω γείτονα (delete) [i-1, j]
i	κόστος από τον αριστερό γείτονα (insert) [i, j-1]	Το μικρότερο από τις 3 πιθανές για να φτάσουμε στο [i, j]

Αλγόριθμος (διαγώνια)

```
LevenshteinDistance(s1, s2)
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]}
 9            else  m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1}
10  return m[|s1|, |s2|]
```

Operations: insert (cost 1), delete (cost 1), replace (cost 1), copy (cost 0)

Αν το i-οστο στοιχείο του s_1 είναι ίδιο με το j-οστό στοιχείου του s_2

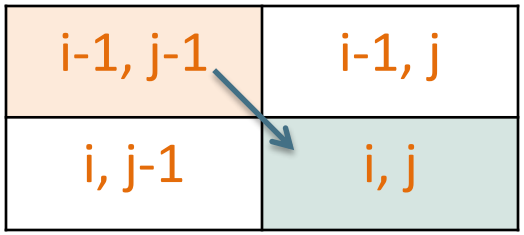


Αλγόριθμος (διαγώνια)

```
LevenshteinDistance(s1, s2)
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]}
 9            else  m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1}
10  return m[|s1|, |s2|]
```

Operations: insert (cost 1), delete (cost 1), replace (cost 1), copy (cost 0)

Αν το i-οστο στοιχείο του s_1 είναι διαφορετικό από το j-οστό στοιχείου του s_2



Υπολογισμός απόστασης Levenshtein

Κόστος από τον πάνω αριστερό γείτονα **Copy ή Replace**	Κόστος από τον πάνω γείτονα **Delete**
Κόστος από τον αριστερό γείτονα **Insert**	Το μικρότερο από τα 3 κόστη

Υπολογισμός απόστασης Levenshtein: παράδειγμα

		f		a		s		t	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
c	**1** **1**	*1* *2*	*2* *1*	**2** **2**	3 **2**	**3** **3**	4 **3**	**4** **4**	5 **4**
a	**2** **2**	**2** 3	**2** **2**	*1* *3*	*3* *1*	*3* *2*	*4* *2*	4 **3**	5 **3**
t	**3** **3**	**3** 4	**3** **3**	3 4	**2** **2**	*2* *3*	*3* *2*	*2* *3*	*4* *2*
s	**4** **4**	**4** 5	**4** **4**	4 5	**3** **3**	**2** 4	3 **2**	*3* *3*	*3* *3*

Δυναμικός προγραμματισμός

1. Βέλτιστη υπό-δομής (Optimal substructure): Η βέλτιστη λύση σε ένα πρόβλημα περιέχει τις υπό-λύσεις, δηλαδή τις βέλτιστες λύσεις σε υπό-προβλήματα

2. Επικαλυπτόμενες υπό-λύσεις (Overlapping subsolutions): Οι υπο-λύσεις υπολογίζονται ξανά και ξανά όταν υπολογίζονται οι ολικές βέλτιστες λύσεις στον brute-force αλγόριθμο.

Δυναμικός προγραμματισμός

- ✓ Στην περίπτωση των αποστάσεων διόρθωσης – το υπό-πρόβλημα δυο προθημάτων
- ✓ Οι επικαλυπτόμενες υπό-λύσεις: χρειαζόμαστε τις περισσότερες αποστάσεις 3 φορές: κίνηση δεξιά, στη διαγώνιο, κάτω

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	1 1	2 2	3 3	4 4
o	1 / 1	1 2 / 2 ?			
s	2 / 2				
l	3 / 3				
o	4 / 4				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	1	1	2	2	3	3	4	4
o	1 / 1	1 / 2	2 / 1						
s	2 / 2								
l	3 / 3								
o	4 / 4								

replace o with s

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** / **1**	**1** / 2	2 / **1**	2 / 3	2 / ?				
s	**2** / **2**								
l	**3** / **3**								
o	**4** / **4**								

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	1 1	2 2	3 3	4 4
o	1 / 1	1 2 / 2 1	2 3 / 2 2		
s	2 / 2				
l	3 / 3				
o	4 / 4				

(διαγώνιο) replace o with n, insert s
(αριστερά-γραμμή) insert n, replace o with s

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** / 2 — 2 / **1**	**2** / **2** — **2** / **2**	2 / 3 — 4 / ?	
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	1	1	2	2	3	3	4	4
o	1 1	1 2	2 1	2 2	3 2	2 3	4 2		
s	2 2								
l	3 3								
o	4 4								

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** / 2	**2** / 3	**2** / 4	4 / 5
		2 / **1**	**2** / **2**	3 / **2**	3 / ?
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** / 2	2 / **1**	**2** / **2**	3 / **2**	**4** / **3**	4 / **3**	5 / **3**
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** 1	**2** 2	**3** 3	**4** 4
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	1 2 / 3 ?			
l	**3** / **3**				
o	**4** / **4**				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**						
l	**3** **3**								
o	**4** **4**								

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** **2**	**1** 2 / 3 **1**	2 3 / 2 ?		
l	**3** **3**				
o	**4** **4**				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** / **1**	**1** / 2	2 / **1**	**2** / **2**	3 / **2**	**2** / 3	4 / **2**	4 / **3**	5 / **3**
s	**2** / **2**	**1** / 3	2 / **1**	**2** / **2**	3 / **2**				
l	**3** / **3**								
o	**4** / **4**								

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	3 3	3 ?		
l	**3** **3**								
o	**4** **4**								

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	
l	**3** / **3**				
o	**4** / **4**				

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	3 4 / 4 ?
l	**3** **3**				
o	**4** **4**				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**								
o	**4** **4**								

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** **1**	**1** 2 2 **1**	**2** 3 **2** **2**	**2** 4 3 **2**	4 5 **3** **3**
s	**2** **2**	**1** 2 3 **1**	**2** 3 **2** **2**	**3** **3** **3** **3**	**3** 4 4 **3**
l	**3** **3**	3 2 4 ?			
o	**4** **4**				

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** / **3**	3 **2** / 4 **2**			
o	**4** / **4**				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	2 3	3 ?				
o	**4** **4**								

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** / **3**	3 **2** / 4 **2**	**2** 3 / 3 **2**		
o	**4** / **4**				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	3 3	4 ?		
o	**4** **4**								

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**		
o	**4** **4**								

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** / **1**	**1** / 2	2 / **1**	**2** / **2**	3 / **3**	**2** / 3	4 / **2**	4 / **3**	5 / **3**
s	**2** / **2**	**1** / 3	2 / **1**	**2** / **2**	3 / **3**	**3** / **3**	4 / **3**	**3** / 4	4 / **3**
l	**3** / **3**	3 / 4	**2** / **2**	**2** / 3	4 / **2**	**3** / **3**	4 / **3**	4 / 4	4 / ?
o	**4** / **4**								

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** 1	**2** 2	**3** 3	**4** 4
o	**1** 1	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** 2	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** 3 / **3** **3**	**3** 4 / 4 **3**
l	**3** 3	3 **2** / 4 **2**	**2** 3 / 3 **2**	**3** 4 / **3** **3**	**4** 4 / **4** **4**
o	**4** 4				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	3 ?						

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** / **1**	**1** / 2	2 / **1**	**2** / **2**	3 / **2**	**2** / 3	4 / **2**	4 / **3**	5 / **3**
s	**2** / **2**	**1** / 3	2 / **1**	**2** / **2**	3 / **2**	**3** / **3**	3 / **3**	**3** / 4	4 / **3**
l	**3** / **3**	3 / 4	**2** / **2**	**2** / 3	3 / **2**	**3** / **3**	4 / **3**	**4** / **4**	4 / **4**
o	**4** / **4**	4 / 5	**3** / **3**						

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	3 4	3 ?				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**				

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	2 4	4 ?		

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** 1	**2** 2	**3** 3	**4** 4
o	**1** / **1**	**1** / 2 2 **1**	**2** / **2** 3 **2**	**2** / 3 4 **2**	4 / **3** 5 / **3**
s	**2** / **2**	**1** / 3 2 **1**	**2** / **2** 3 **2**	**3** / **3** 4 **3**	**3** / 4 4 / **3**
l	**3** / **3**	3 / **2** 4 **2**	**2** / 3 3 **2**	**3** / 4 **3**	**4** / **4** **4** / **4**
o	**4** / **4**	4 / **3** 5 **3**	**3** / **3** 4 **3**	**2** / 4 4 **2**	

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**	4 3	5 ?

Υπολογισμός απόστασης: παράδειγμα

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**	4 **3**	5 **3**

Υπολογισμός απόστασης: παράδειγμα

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** **1**	**1** 2 2 **1**	**2** 3 **2** **2**	**2** 4 3 **2**	4 5 **3** **3**
s	**2** **2**	**1** 2 3 **1**	**2** 3 **2** **2**	**3** **3** **3** **3**	**3** 4 4 **3**
l	**3** **3**	3 **2** 4 **2**	**2** 3 3 **2**	**3** 4 **3** **3**	**4** **4** **4** **4**
o	**4** **4**	4 **3** 5 **3**	**3** **3** 4 **3**	**2** 4 4 **2**	4 5 3 **3**

Πως μπορώ να δω τις πράξεις που οδήγησαν από OSLO σε SNOW?

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**	4 **3**	5 **3**

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**	4 **3**	5 **3**

cost	operation	input	output
1	insert	*	w

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**	4 **3**	5 **3**

cost	operation	input	output
0	(copy)	o	o
1	insert	*	w

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** **3**	3 **2** / 4 **2**	2 3 / 3 2	**3** 4 / **3** **3**	**4** **4** / **4** **4**
o	**4** **4**	4 **3** / 5 **3**	**3** **3** / 4 **3**	2 4 / 4 2	4 5 / 3 3

cost	operation	input	output
1	replace	l	n
0	(copy)	o	o
1	insert	*	w

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** **3**	3 **2** / 4 **2**	**2** 3 / 3 **2**	**3** 4 / **3** **3**	**4** **4** / **4** **4**
o	**4** **4**	4 **3** / 5 **3**	**3** **3** / 4 **3**	**2** 4 / 4 **2**	4 5 / **3** **3**

cost	operation	input	output
0	(copy)	s	s
1	replace	l	n
0	(copy)	o	o
1	insert	*	w

		s	n	o	w
	0	1 1	2 2	3 3	4 4
o	1 / 1	1 2 / 2 1	2 3 / 2 2	2 4 / 3 2	4 5 / 3 3
s	2 / 2	1 2 / 3 1	2 3 / 2 2	3 3 / 3 3	3 4 / 4 3
l	3 / 3	3 2 / 4 2	2 3 / 3 2	3 4 / 3 3	4 4 / 4 4
o	4 / 4	4 3 / 5 3	3 3 / 4 3	2 4 / 4 2	4 5 / 3 3

cost	operation	input	output
1	delete	o	*
0	(copy)	s	s
1	replace	l	n
0	(copy)	o	o
1	insert	*	w

Πως μπορώ να δω τις πράξεις που οδήγησαν από CAT σε CATCAT?

		c		a		t		c		a		t	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**	**5**	**5**	**6**	**6**
c	**1** **1**	**0** 2	2 **0**	2 **1**	3 **1**	3 **2**	4 **2**	**3** **3**	5 **3**	5 **4**	6 **4**	6 **5**	7 **5**
a	**2** **2**	2 3	**1** **1**	**0** 2	2 **0**	2 **1**	3 **1**	3 **2**	4 **2**	**3** **3**	5 **3**	5 **4**	6 **4**
t	**3** **3**	3 4	**2** **2**	2 3	**1** **1**	**0** 2	2 **0**	2 **1**	3 **1**	3 **2**	4 **2**	**3** **3**	5 **3**

		c	a	t	c	a	t
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**	**5** **5**	**6** **6**
c	**1** **1**	**0** 2 / 2 **0**	2 3 / **1** **1**	3 4 / **2** **2**	**3** 5 / **3** **3**	5 6 / **4** **4**	6 7 / **5** **5**
a	**2** **2**	2 **1** / 3 **1**	**0** 2 / 2 **0**	2 3 / **1** **1**	3 4 / **2** **2**	**3** 5 / **3** **3**	5 6 / **4** **4**
t	**3** **3**	3 **2** / 4 **2**	2 **1** / 3 **1**	**0** 2 / 2 **0**	2 3 / **1** **1**	3 4 / **2** **2**	**3** 5 / **3** **3**

cost	operation	input	output
1	insert	*	c
1	insert	*	a
1	insert	*	t
0	(copy)	c	c
0	(copy)	a	a
0	(copy)	t	t

		c	a	t	c	a	t
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**	**5** **5**	**6** **6**
c	**1** **1**	**0** 2 2 **0**	2 3 **1** **1**	3 4 **2** **2**	**3** 5 **3** **3**	5 6 **4** **4**	6 7 **5** **5**
a	**2** **2**	2 **1** 3 **1**	**0** 2 2 **0**	2 3 **1** **1**	3 4 **2** **2**	**3** 5 **3** **3**	5 6 **4** **4**
t	**3** **3**	3 **2** 4 **2**	2 **1** 3 **1**	**0** 2 2 **0**	2 3 **1** **1**	3 4 **2** **2**	**3** 5 **3** **3**

cost	operation	input	output
0	(copy)	c	c
1	insert	*	a
1	insert	*	t
1	insert	*	c
0	(copy)	a	a
0	(copy)	t	t

		c	a	t	c	a	t
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**	**5** **5**	**6** **6**
c	**1** / **1**	**0** 2 / 2 **0**	2 3 / **1** **1**	3 4 / **2** **2**	**3** 5 / **3** **3**	5 6 / **4** **4**	6 7 / **5** **5**
a	**2** / **2**	2 **1** / 3 **1**	**0** 2 / 2 **0**	2 3 / **1** **1**	3 4 / **2** **2**	**3** 5 / **3** **3**	5 6 / **4** **4**
t	**3** / **3**	3 **2** / 4 **2**	2 **1** / 3 **1**	**0** 2 / 2 **0**	2 3 / **1** **1**	3 4 / **2** **2**	**3** 5 / **3** **3**

cost	operation	input	output
0	(copy)	c	c
0	(copy)	a	a
1	insert	*	t
1	insert	*	c
1	insert	*	a
0	(copy)	t	t

			c		a		t		c		a		t	
		0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**	**5**	**5**	**6**	**6**
c	**1**	**1**	**0**	2	2	3	3	4	**3**	5	5	6	6	7
			2	**0**	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**	**5**	**5**
a	**2**	2	1		**0**	2	2	3	3	4	**3**	5	5	6
		2	3	**1**	2	**0**	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
t	**3**	3	2		2	1	**0**	2	2	3	3	4	**3**	5
		3	4	**2**	3	**1**	2	**0**	**1**	**1**	**2**	**2**	**3**	**3**

cost	operation	input	output
0	(copy)	c	c
0	(copy)	a	a
0	(copy)	t	t
1	insert	*	c
1	insert	*	a
1	insert	*	t

Σταθμισμένη απόσταση διόρθωσης

- Το βάρος μιας πράξης εξαρτάται από τον ποιο χαρακτήρα (χαρακτήρες) περιλαμβάνει
  - Στόχος να λάβει υπόψη λάθη OCR ή πληκτρολόγησης Παράδειγμα: ***m*** πιο πιθανό να πληκτρολογηθεί ως ***n*** παρά ως ***q***
  - Οπότε η αντικατάσταση του ***m*** από ***n*** έχει μικρότερη απόσταση διόρθωσης από την απόσταση του από το ***q***
  - Διατύπωση ως πιθανοτικό μοντέλο
- Προϋποθέτει ως είσοδο έναν πίνακα βαρών
- *Πως θα μετατρέψουμε το δυναμικό προγραμματισμό για να χειριστούμε τα βάρη;*

Επικάλυψη *k*-γραμμάτων

Εναλλακτικός ορισμός απόστασης: βάση των κοινών *k*-γραμμάτων

- Η ομοιότητα δυο όρων είναι ίση με τον αριθμό των κοινών *k*-γραμμάτων τους

Επικάλυψη *k*-γραμμάτων

Εναλλακτικός ορισμός απόστασης: βάση των κοινών *k*-γραμμάτων

- Η ομοιότητα δυο όρων είναι ίση με τον αριθμό των κοινών *k*-γραμμάτων τους

Παράδειγμα

3-γράμματα

november

- Τα τριγράμματα είναι *nov, ove, vem, emb, mbe, ber*.

december

- Τα τριγράμματα είναι *dec, ece, cem, emb, mbe, ber*.

Άρα 3 τριγράμματα επικαλύπτονται (από τα 6 κάθε όρου) – ομοιότητα 3

Μια δυνατότητα – συντελεστής Jaccard

- Συνήθης μέτρηση της επικάλυψης

Έστω *X* και *Y* δύο σύνολα, ο συντελεστής Jaccard (Jaccard coefficient) ορίζεται ως:

$$|X \cap Y| / |X \cup Y|$$

- Ίσος με 1 όταν τα *X* και *Y* έχουν τα ίδια στοιχεία και 0 όταν είναι ξένα
- Τα *X* and *Y* δε χρειάζεται να έχουν το ίδιο μέγεθος
- Πάντα μεταξύ του 0 και του 1
  - Το κατώφλι καθορίζει αν υπάρχει ταίριασμα, πχ., αν J.C. > 0.8, τότε ταίριασμα

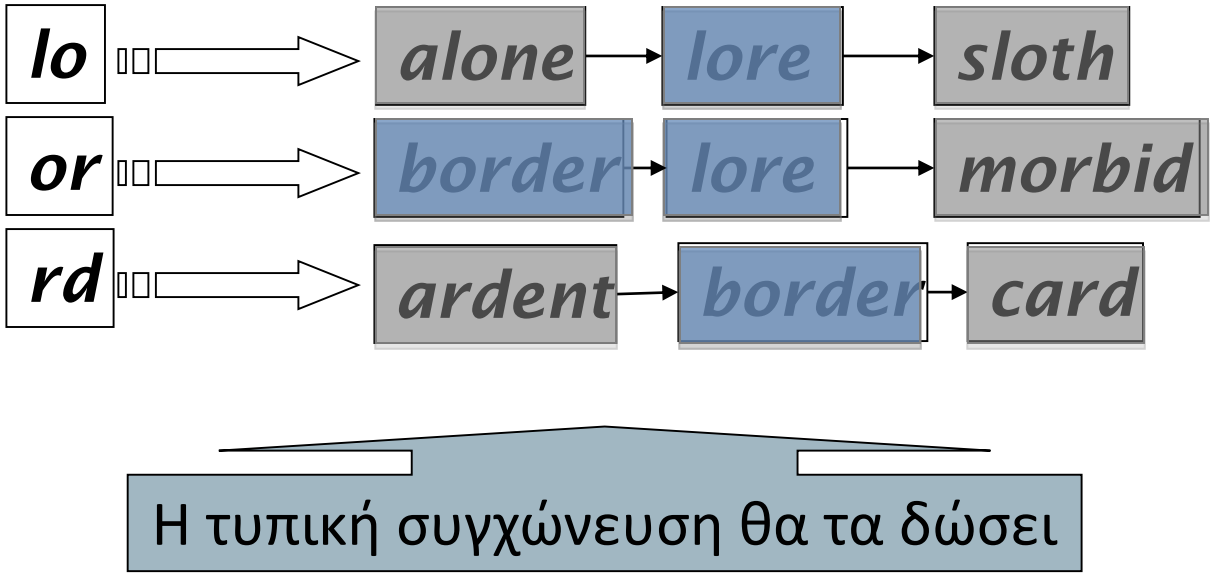
Πως το υπολογίζουμε αποδοτικά στην περίπτωση των *k*-γραμμάτων;

Επικάλυψη *k*-γραμμάτων

- Απαρίθμησε όλα το *k*-γράμματα στον όρο της ερώτησης
- Χρησιμοποίησε το ευρετήριο *k*-γραμμάτων για να ανακτήσεις όλους τους όρους του λεξικού που ταιριάζουν κάποιο από τα *k*-γράμματα του ερωτήματος
- Ανέκτησε όλους τους όρους του λεξικού που ταιριάζουν *κάποιο* (>= *κατώφλι*) αριθμό από τα *k*-γράμματα του ερωτήματος

Ταίριασμα τριγραμμάτων

- Έστω το ερώτημα ***lord*** – θέλουμε να βρούμε τις λέξεις που ταιριάζουν τουλάχιστον 2 από τα 3 διγράμματα (***lo, or, rd***)



- *Πως μπορούμε να το χρησιμοποιήσουμε ως ένα κανονικοποιημένο μέσω επικάλυψης;*

Απόσταση διόρθωσης από όλους τους όρους του λεξικού;

- Δοθέντος ενός (ανορθόγραφου) ερωτήματος, υπολογίζουμε την απόσταση διόρθωσης από όλους τους όρους του λεξικού
  - Ακριβό και αργό
- Μπορούμε να μειώσουμε τον αριθμό των υποψήφιων όρων του ευρετηρίου;
  - Μόνο λέξεις που αρχίζουν από το ίδιο γράμμα
  - Να θεωρήσουμε ότι υπάρχει * και ένα permuterm
  - Να βρούμε λέξεις με αρκετά κοινά *k*-γράμματα και να περιορίσουμε τον υπολογισμό απόστασης με αυτές

Χρήση των αποστάσεων διόρθωσης

1. ‘Η, δοθείσας μιας ερώτησης, πρώτα απαρίθμησε όλες τις ακολουθίες χαρακτήρων μέσα σε μια προκαθορισμένη (σταθμισμένη) απόσταση διόρθωσης (π.χ., 2)
2. Βρες την τομή αυτού του συνόλου με τις «σωστές» λέξεις
3. *Πρότεινε τους όρους* που βρήκες στο χρήστη
   - Ψάξε όλες τις πιθανές διορθώσεις στο αντεστραμμένο ευρετήριο και επέστρεψε όλα τα έγγραφα … αργό
   - Μπορούμε να *επιστρέψουμε τα έγγραφα* μόνο για την πιο πιθανή διόρθωση (πχ τη πιο συνηθισμένη λέξη) ή τη διόρθωση που επιλέγουν οι χρήστες πιο συχνά (hit-based correction)

- Η εναλλακτική λύση παίρνει τον έλεγχο από το χρήστη αλλά κερδίζουμε ένα γύρο διάδρασης

Διόρθωση εξαρτώμενη από το περιβάλλον

Κείμενο: ***I flew from Heathrow to Narita.***

- Θεωρείστε το ερώτημα-φράση ***"flew form Heathrow"***
- Θα θέλαμε να απαντήσουμε

  Did you mean "***flew from Heathrow***"?

Γιατί δεν υπάρχουν (αρκετά) έγγραφα που να ταιριάζουν στο αρχικό ερώτημα φράση

Διόρθωση βασισμένη στα συμφραζόμενα

- Χρειάζεται συμφραζόμενο περιβάλλον για να το πιάσει αυτό.

Πρώτη ιδέα:

1. Ανέκτησε τους όρους του λεξικού που είναι κοντά (σε σταθμισμένη απόσταση διόρθωσης) από κάθε όρο του ερωτήματος
2. Δοκίμασε όλες τις πιθανές φράσεις που προκύπτουν κρατώντας κάθε φορά μια λέξη σταθερή
   - ***flew from heathrow***
   - ***fled form heathrow***
   - ***flea form heathrow***
3. *Hit-based spelling correction*: Πρότεινε την εναλλακτική με τα περισσότερα hits

Διόρθωση βασισμένη στα συμφραζόμενα

Εναλλακτική Προσέγγιση με χρήση biwords

1. Σπάσε της φράση σε σύζευξη biwords.
2. Ψάξε τα biwords που χρειάζονται διόρθωση μόνο ενός όρου.
3. Απαρίθμησε μόνο τις φράσεις που περιέχουν «κοινά» biwords.

ΦΩΝΗΤΙΚΗ ΔΙΟΡΘΩΣΗ (SOUNDEX)

Soundex

Φωνητική διόρθωση: ερώτημα που «ακούγεται» όπως ο σωστός όρος

- Κλάση ευριστικών για την επέκταση ενός ερωτήματος σε φωνητικά (phonetic) ισοδύναμα
  - Εξαρτώνται από τη γλώσσα – κυρίως για ονόματα
    - Π.χ., ***chebyshev*** → ***tchebycheff***
- Προτάθηκε από το U.S. census … το 1918 (για ονόματα από εγκληματίες!)

Βασική ιδέα: *"phonetic hash":* όροι που «ακούγονται» ίδιοι κατακερματίζονται στην ίδια θέση

Soundex – τυπικός αλγόριθμος

- Μετέτρεψε κάθε token προς δεικτοδότηση σε μια μορφή 4-χαρακτήρων
- Το ίδιο και για τους όρους του ερωτήματος
- Κατασκεύασε ένα ανεστραμμένο ευρετήριο από αυτούς τους 4-χαρακτήρες στον αρχικό όρο και ψάξε στο ευρετήριο τις μειωμένες μορφές
  - (όταν το ερώτημα χρειάζεται φωνητικό ταίριασμα)

- http://www.creativyst.com/Doc/Articles/SoundEx1/SoundEx1.htm#Top

Soundex – τυπικός αλγόριθμος

1. Κράτησε τον πρώτο χαρακτήρα της λέξης
2. Μετάτρεψε όλες τις εμφανίσεις των παρακάτω όρων σε '0' (zero):
   'A', E', 'I', 'O', 'U', 'H', 'W', 'Y'.
3. Άλλαξε τα γράμματα σε αριθμούς ως ακολούθως:
   - B, F, P, V $\rightarrow$ 1
   - C, G, J, K, Q, S, X, Z $\rightarrow$ 2
   - D, T $\rightarrow$ 3
   - L $\rightarrow$ 4
   - M, N $\rightarrow$ 5
   - R $\rightarrow$ 6

Soundex συνέχεια

4. Σβήσε όλα τα ζεύγη συνεχόμενων αριθμών
5. Σβήσε όλα τα υπομένοντα 0
6. Πρόσθεσε 0 στο τέλος και επέστρεψε τις τέσσερις πρώτες θέσεις που θα είναι της μορφής <uppercase letter> <digit> <digit> <digit>.

Π.χ., ***Herman*** γίνεται H655.

Το ***hermann*** δίνει τον ίδιο κωδικό;

ΤΕΛΟΣ 3ου Μαθήματος

Ερωτήσεις?

Χρησιμοποιήθηκε κάποιο υλικό των:
- ✓ *Pandu Nayak and Prabhakar Raghavan, CS276:Information Retrieval and Web Search (Stanford)*
- ✓ *Hinrich Schütze and Christina Lioma, Stuttgart IIR class*