Πανεπιστήμιο Κρήτης, Τμήμα Επιστήμης Υπολογιστών
ΗΥ463 - Συστήματα Ανάκτησης Πληροφοριών
2006-2007 Εαρινό Εξάμηνο

3η Σειρά ασκήσεων
(Ευρετηρίαση, Αναζήτηση σε Κείμενα και Άλλα Θέματα)
(βαθμοί 12: όποιος πάρει άριστα σε όλες θα λάβει μέγιστο bonus 10%)
Ανάθεση: 24 Απριλίου
Παράδοση: 7 Μαΐου

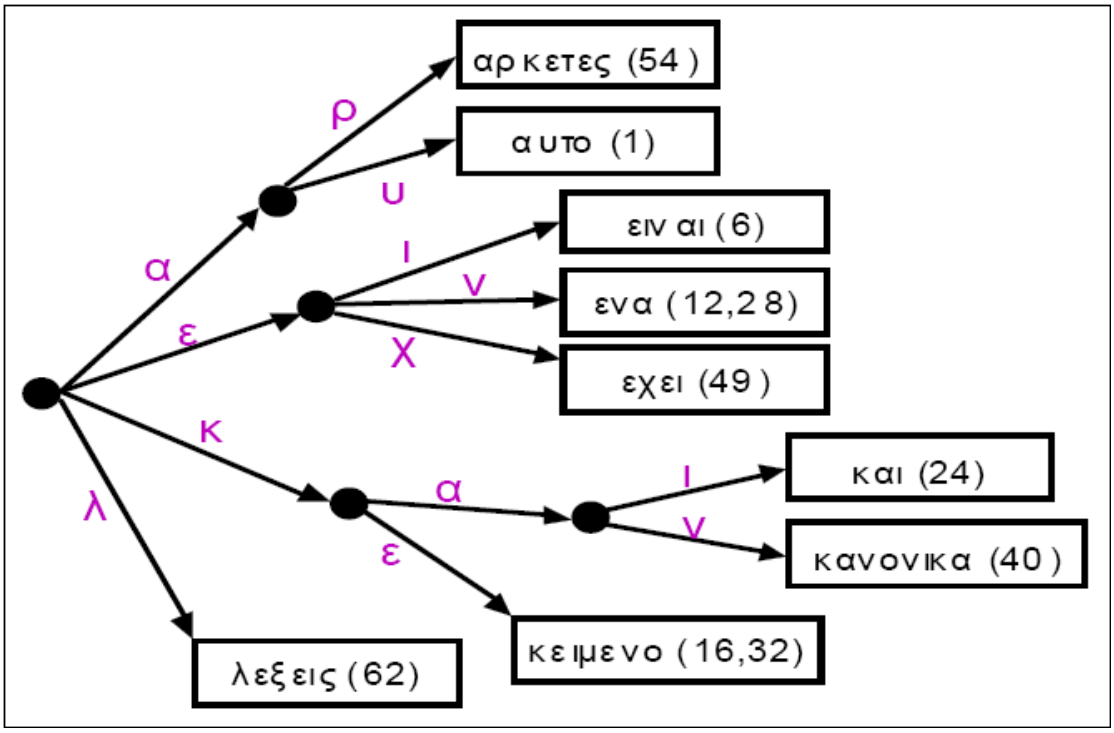
Άσκηση 1 (2.5 βαθμοί)

Θεωρείστε ένα έγγραφο με περιεχόμενο «αυτό είναι ένα κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις». Αγνοώντας τους τόνους, σχεδιάστε
(α) το trie του λεξιλογίου του παραπάνω εγγράφου,
(β) το δένδρο καταλήξεων του παραπάνω εγγράφου θεωρώντας ως σημεία ευρετηρίου (index points) τις αρχές των λέξεων, και
(γ) συμπτύξτε το παραπάνω δένδρο καταλήξεων στη μορφή ενός Patricia tree.

Λύση (από: Τσιαλιαμάνης – Αναγνωστόπουλος Πέτρος)

```
1    5     10    15    20    25    30    35    40    45    50    55    60    65
αυτό είναι ένα κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις
```

(α) Το trie του λεξιλογίου είναι



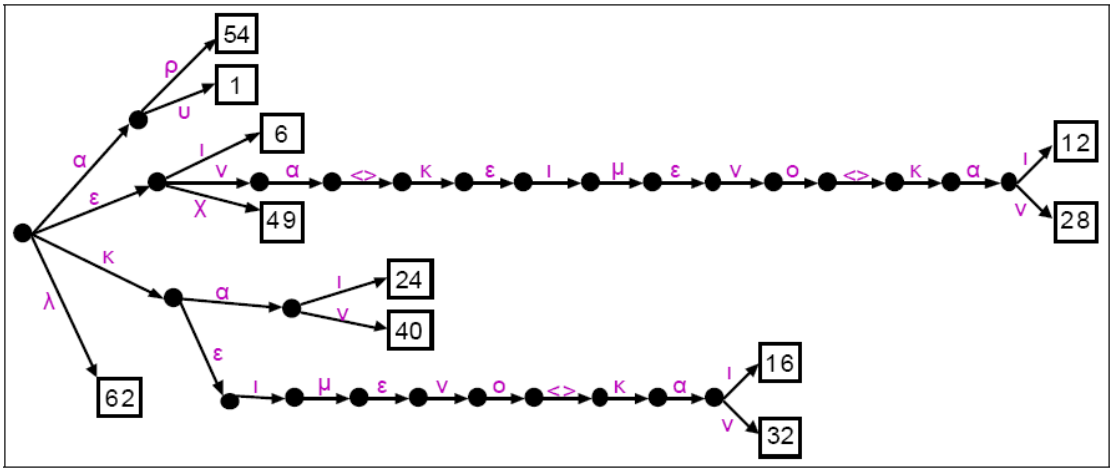
(β) Οι καταλήξεις του κειμένου είναι

αυτό είναι ένα κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις
είναι ένα κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις
ένα κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις
κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις
και ένα κείμενο κανονικά έχει αρκετές λέξεις
ένα κείμενο κανονικά έχει αρκετές λέξεις
κείμενο κανονικά έχει αρκετές λέξεις
κανονικά έχει αρκετές λέξεις
έχει αρκετές λέξεις
αρκετές λέξεις
λέξεις

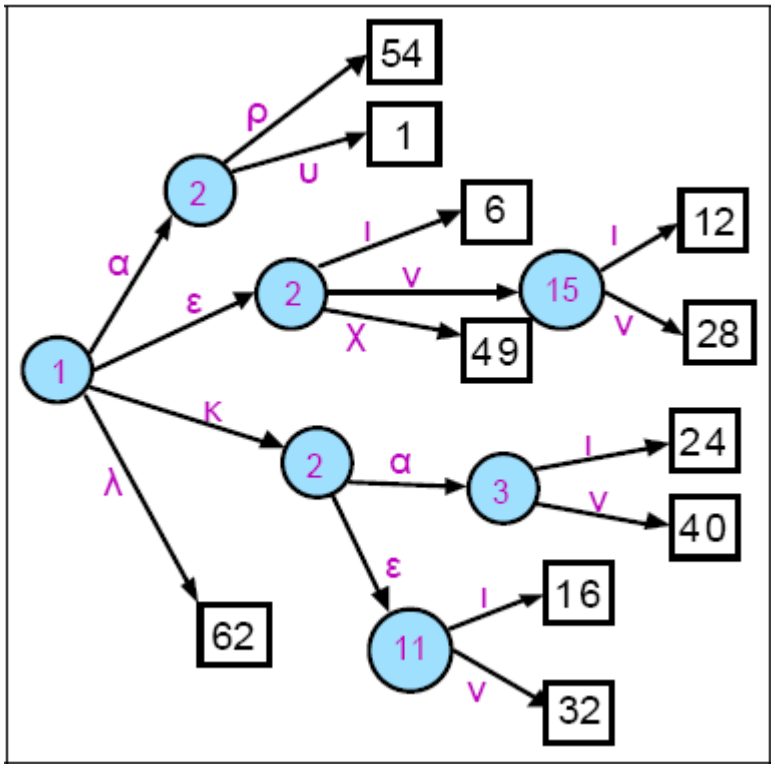
Κατόπιν ταξινομώντας τις παραπάνω καταλήξεις λεξικογραφικά έχουμε,

αρκετές λέξεις
αυτό είναι ένα κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις
είναι ένα κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις
ένα κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις
ένα κείμενο κανονικά έχει αρκετές λέξεις
έχει αρκετές λέξεις
και ένα κείμενο κανονικά έχει αρκετές λέξεις
κανονικά έχει αρκετές λέξεις
κείμενο και ένα κείμενο κανονικά έχει αρκετές λέξεις
κείμενο κανονικά έχει αρκετές λέξεις
λέξεις

Το δένδρο καταλήξεων που προκύπτει από τις ταξινομημένες καταλήξεις θεωρώντας ως σημεία ευρετηρίου (index points) τις αρχές των λέξεων είναι



(γ) Το Patricia tree που προκύπτει από το παραπάνω δένδρο καταλήξεων είναι,



Άσκηση 2 (1.0 βαθμοί)

Θεωρείστε τις συμβολοσειρές π1= «to be or not to be» και π2 = «κανονικα». Δώστε τον πίνακα προεπεξεργασίας (δηλαδή τον πίνακα «next») που θα φτιάξει ο αλγόριθμος KMP (Knuth-Morris-Pratt) για κάθε μία από τις παραπάνω συμβολοσειρές.

Λύση (από: Μαρκετάκης Γιάννης)

Για να βρούμε τον πίνακα "next" διανύουμε τους χαρακτήρες από την αρχή μέχρι το τέλος του pattern και για να συμπληρώσουμε τον πίνακα κάνουμε το εξής:
Η επόμενη θέση του πίνακα (j) είναι το μέγιστο μήκος του προθέματος στις προηγούμενες θέσεις (1, …, j-1) που είναι επίσης και επίθεμα και οι χαρακτήρες που ακολουθούν το πρόθεμα και το επίθεμα πρέπει να είναι διαφορετικοί. Παίρνουμε λοιπόν το πρώτο pattern p1= "to be or not to be"

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
p1[j]	t	o		b	e		o	r		n	o	t		t	o		b	e
next[j]	0																	

Μέχρι την θέση 11 παρατηρούμε ότι δεν μπορεί να υπάρξει κάποιο πρόθεμα που να το έχουμε ξαναδεί. Στην 12η όμως θέση έρχεται ένας χαρακτήρας "t" .Αυτός ο χαρακτήρας ήταν πρόθεμα νωρίτερα και μάλιστα ο χαρακτήρας που υπήρχε μετά είναι διαφορετικός από τον χαρακτήρα που ακολουθεί τώρα. Έτσι αποτελεί πρόθεμα και μάλιστα με μήκος 1 οπότε:

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
p1[j]	t	o		b	e		o	r		n	o	t		t	o		b	e
next[j]	0	0	0	0	0	0	0	0	0	0	0	0	1					

Στην θέση 14 συναντάμε ένα χαρακτήρα ο οποίος είναι πρόθεμα και ο επόμενος χαρακτήρας του προθέματος είναι ίδιος με τον επόμενο χαρακτήρα στην θέση 15. Επομένως το μήκος είναι 0 και προχωράμε στην επόμενη θέση. Εκεί συναντάμε τον χαρακτήρα "ο" που συνδυαζόμενος με τα προηγούμενα μας κάνει επίθεμα "το" το οποίο υπάρχει και ως πρόθεμα και μάλιστα οι χαρακτήρες που ακολουθούν του προθέματος και του επιθέματος είναι διαφορετικοί. Έτσι και εδώ το μήκος είναι όπως πριν 0. Συνεχίζοντας καταλήγουμε μέχρι να διαβάσουμε όλο το κείμενο όποτε θα έχουμε διαβάσει το επίθεμα "to be" το οποίο είναι και πρόθεμα και οι χαρακτήρες που ακολουθούν τα 2 είναι διαφορετικοί. Έτσι λοιπόν καταλήγουμε στο μέγιστο πρόθεμα με μήκος 5.

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	
$p_1[j]$	t	o		b	e		o	R		n	o	t		t	o		b	e	
next[j]	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	5

Κατόπιν παίρνουμε το δεύτερο pattern p2= "κανονικα"

j	1	2	3	4	5	6	7	8	
$p_2[j]$	κ	α	ν	ο	ν	ι	κ	A	
next[j]	0								

Μέχρι την θέση 7 είναι προφανές ότι ο πίνακας next θα έχει μέχρι εκείνη την θέση τιμή 0, αφού δεν υπάρχει πρόθεμα σε αυτές τις θέσεις που να είναι και επίθεμα.

j	1	2	3	4	5	6	7	8	
$p_2[j]$	κ	α	ν	ο	ν	ι	κ	A	
next[j]	0	0	0	0	0	0	0		

Όταν όμως συναντάμε στην 7η θέση το χαρακτήρα "κ" τότε παρατηρούμε ότι ήταν πρόθεμα νωρίτερα και ο επόμενος του και στις 2 περιπτώσεις ήταν ο χαρακτήρας "α". Επομένως σε αυτή την περίπτωση έχουμε 0.

j	1	2	3	4	5	6	7	8	
$p_2[j]$	κ	α	ν	ο	ν	I	κ	α	
next[j]	0	0	0	0	0	0	0	0	

Κατόπιν όμως τελειώνει η συμβολοσειρά και έτσι έχουμε ότι το μέγιστο μήκος προθέματος είναι 2 .Έτσι:

j	1	2	3	4	5	6	7	8	
$p_2[j]$	κ	α	ν	ο	ν	I	κ	α	
next[j]	0	0	0	0	0	0	0	0	2

Άσκηση 3 (1.0 βαθμοί)

Θεωρείστε ότι θέλουμε να βρούμε αν η συμβολοσειρά «misspell» υπάρχει στη συμβολοσειρά «misspelling» με ανοχή λάθους (Edit Distance) κ=2. Σχεδιάστε τον πίνακα που απεικονίζει τον τρόπο με τον οποίο θα λειτουργήσει ο σχετικός αλγόριθμος δυναμικού προγραμματισμού.

Λύση (από: Τσιαλιαμάνης – Αναγνωστόπουλος Πέτρος)

		m	i	s	s	p	e	l	l	i	n	g
	0	0	0	0	0	0	0	0	0	0	0	0
m	1	0	1	1	1	1	1	1	1	1	1	1
i	2	1	0	1	2	2	2	2	2	1	2	2
s	3	2	1	0	1	2	3	3	3	2	2	3
s	4	3	2	1	0	1	2	3	4	3	3	3
p	5	4	3	2	1	0	1	2	3	4	4	4
e	6	5	4	3	2	1	0	1	2	3	4	5
l	7	6	5	4	3	2	1	0	1	2	3	4
l	8	7	6	5	4	3	***2***	***1***	***0***	***1***	***2***	3

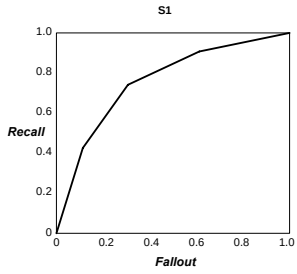
Δηλαδή οι συμβολοσειρές που μας δίνει ο αλγόριθμος δυναμικού προγραμματισμού είναι :
“misspe”
“misspel”
“misspell”
“misspelli”
“misspellin”

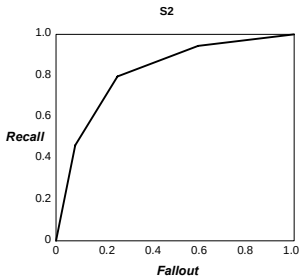
Άσκηση 4 (6.0 βαθμοί)

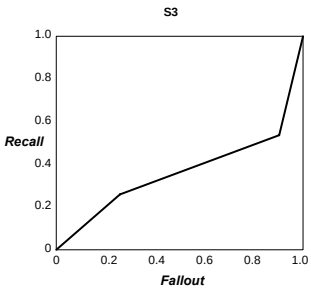
Ένας εναλλακτικός τρόπος αξιολόγησης της αποτελεσματικότητας ενός ΣΑΠ, είναι οι καμπύλες Recall-Fallout. Ορίζονται ανάλογα με τις καμπύλες Precision-Recall, μόνο που τώρα ο άξονας Χ έχει τιμές του Fallout, ενώ ο Υ τιμές του Recall.

(α)[20β] Έστω ένα σύστημα Χ το οποίο για κάθε επερώτηση που λαμβάνει επιστρέφει μια τυχαία γραμμική διάταξη των κειμένων του. Τι μορφή θα είχε η καμπύλη Recall-Fallout αυτού του συστήματος; (αυτή που θα προέκυπτε από την αξιολόγηση πολλών επερωτήσεων)

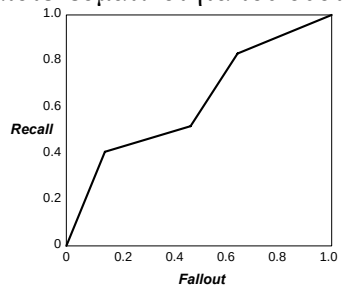
(β)[20β] Θεωρείστε τρία συστήματα με τις καμπύλες Recall-Fallout που ακολουθούν. Παρατηρώντας αυτές τις καμπύλες, ποιο σύστημα θα κρίνατε ότι προσφέρει πιο αποτελεσματική ανάκτηση πληροφορίας;







(γ) [20β] Έστω ένα ΣΑΠ του οποίου η καμπύλη Recall-Fallout έχει τη μορφή που εικονίζεται παρακάτω. Θα μπορούσατε να κάνετε κάτι για να βελτιώσετε την αποτελεσματικότητα του συστήματος; Αναπτύξτε ελεύθερα τις ιδέες σας.

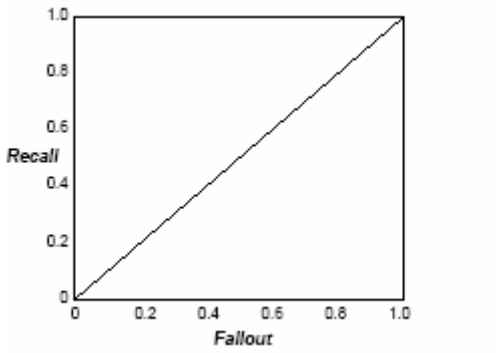


Λύση

α) Μπορούμε να θεωρήσουμε ότι ένα σύστημα επιστρέφει εκείνα τα έγγραφα των οποίων ο βαθμός ομοιότητας με την επερώτηση υπερβαίνει ένα κατώφλι.
Εναλλακτικά (αλλά ισοδύναμα για τους σκοπούς της συζήτησής μας), το σύστημα μπορεί να επιστρέφει όλα τα έγγραφα (σε φθίνουσα ως προς το βαθμό ομοιότητάς τους) σειρά και ο χρήστης να «καταναλώνει» τα έγγραφα της απάντησης αρχίζοντας από τα κορυφαία και σταματώντας μόλις συναντήσει ένα έγγραφο του οποίου ο βαθμός ομοιότητας πέσει κάτω από ένα κατώφλι.

Αν το κατώφλι είναι πολύ υψηλό τότε τίποτα δεν θα ανακτηθεί (/διαβαστεί). Σε αυτήν την περίπτωση έχουμε Recall=Fallout=0.
Αν το κατώφλι είναι πολύ χαμηλό, τότε όλα τα έγγραφα θα ανακτηθούν (/διαβαστούν). Σε αυτήν την περίπτωση έχουμε Recall=Fallout=1.
Από τα παραπάνω προκύπτει ότι όλες οι καμπύλες Recall-Fallout πρέπει να περνάνε από τα σημεία (0,0) και (1,1).

Αν ένα σύστημα X για κάθε επερώτηση που λαμβάνει επιστρέφει όλα τα έγγραφα της συλλογής σε τυχαία γραμμική διάταξη, τότε στη μέση περίπτωση το Fallout θα αυξάνει γραμμικά με το Recall, οπότε η καμπύλη του συστήματος θα είναι όπως αυτήν που φαίνεται στην Εικόνα 1.



Εικόνα 1

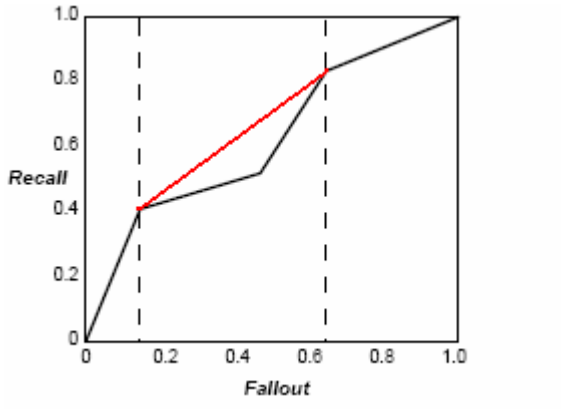
β) Θέτοντας ως κατώτατο όριο την καμπύλη της Εικόνας 1, μπορούμε να αποκλείσουμε κατευθείαν το σύστημα S3, αφού ισχύει Fallout < Recall για όλες τις τιμές.
Συγκρίνοντας τα συστήματα S1 και S2 μπορούμε να αποφασίσουμε ότι το S2 είναι καλύτερο από το S1, αφού για τις ίδιες τιμές του Fallout το S2 έχει καλύτερο Recall από το S1.

γ) Έστω α και β τα δύο άκρα του κόκκινου ευθύγραμμου τμήματος του παρακάτω διαγράμματος. Κάθε ένα από αυτά τα δύο σημεία αντιστοιχεί σε ένα βαθμό ομοιότητας, για παράδειγμα το α μπορεί να αντιστοιχεί σε βαθμό ομοιότητας 0.8, ενώ το β σε ένα βαθμό ομοιότητας 0.6.
Έστω S το σύνολο των εγγράφων μιας απάντησης που έχουν βαθμό ομοιότητας χ, τ.ω. $0.6 \leq \chi \leq 0.8$.
Από το ερώτημα (α) είδαμε ότι αν το σύστημα επιστρέφει τυχαίες διατάξεις εγγράφων, τότε η καμπύλη Recall-Fallout θα έχει τη μορφή ευθύγραμμου τμήματος.
Εκ τούτου, θα μπορούσαμε να βελτιώσουμε το σύστημα μας με τον εξής (.. εκ πρώτης όψεως παράδοξο) τρόπο:
Το σύστημα λειτουργεί όπως προηγουμένως με την μόνη διαφορά ότι τα έγγραφα που ανήκουν στο S (της εν λόγω απάντησης) δεν διατάσσονται βάσει του βαθμού συνάφειας που υπολόγισε το σύστημα αλλά τυχαία!

Προφανώς, η καμπύλη Recall-Fallout του βελτιωμένου συστήματος θα περιλάμβανε το κόκκινο ευθύγραμμο τμήμα (αντί της τεθλασμένης γραμμής που ενώνει τα σημεία α και β)

Για περισσότερα μπορείτε να δείτε το άρθρο
Stephen Robertson, On score distributions and relevance, ECIR '2007, Rome, April 2007.

.



Εικόνα 2

Άσκηση 5 (2.5 βαθμοί)

Προσπαθήστε μέσω διαδικτύου να βρείτε όσο το δυνατόν περισσότερα στοιχεία σχετικά με το μέγεθος του ελληνικού παγκόσμιου ιστού (αριθμός σελίδων, μέσο μέγεθος σελίδων, πλήθος συνδέσμων, συνολικός όγκος, και όποια άλλη χρήσιμη και αξιόπιστη μέτρηση βρείτε). Σκοπός μας είναι η σχεδίαση του ευρετηρίου μιας μηχανής αναζήτησης για τον ελληνικό ιστό, το οποίο να μπορεί να φιλοξενηθεί στο μηχάνημα που έχουμε στη διάθεση μας αυτή τη στιγμή (κύρια μνήμη: 2GBytes, σκληρός δίσκος: 150GB). Βάσει αυτών που έχουμε δει στο μάθημα, εκτιμήστε το μέγεθος που θα έχει το λεξιλόγιο και οι λίστες εμφάνισης αν αποφασίζαμε να χρησιμοποιήσουμε μια δομή ανεστραμμένου αρχείου.
Περιγράψτε όποια άλλη εναλλακτική ή συμπληρωματική δομή ευρετηρίου κρίνετε ότι μπορεί να επιταχύνει τη λειτουργία της μηχανής αναζήτησης.

Λύση

Στο [Crawling a Country: Better Strategies than BreadthFirst for Web Page Ordering] αναφέρεται ότι το μέγεθος των σελίδων στο .gr domain το 2004 ήταν περίπου 3.5 εκατομμύρια σελίδες. Επίσης, στο [http://www.optimizationweek.com/reviews/average-web-page/] αναφέρουν ότι το μέσο μέγεθος μίας ιστοσελίδας στο web (μόνο κείμενο, χωρίς εικόνες, scripts κλπ) είναι 25 KB. Τέλος, υποθέτουμε ότι ο μέσος αριθμός εξερχόμενων συνδέσμων που υπάρχουν ανά site είναι 20, το μέσο μέγεθος μιας λέξης είναι 10 χαρακτήρες και ότι υπάρχουν περίπου 150000 λέξεις στο ελληνικό λεξιλόγιο.

Ένα ανεστραμμένο ευρετήριο αποτελείται από το λεξιλόγιο και τις λίστες εμφάνισης κάθε λέξης που ανήκει στο λεξιλόγιο. Εφόσον πρόκειται να ευρετηριάσουμε όλο το ελληνικό domain, το λεξιλόγιο θα περιέχει όλες τις ελληνικές λέξεις και αρκετές αγγλικές, επομένως το μέγεθός του θα είναι περίπου $(150000+40000)\cdot 10 bytes = 1.81 MB$, έχοντας υποθέσει ότι θα υπάρχουν 40000 αγγλικές λέξεις επιπρόσθετα. Λόγω του μικρού του μεγέθους, το λεξιλόγιο θα μπορούσε για λόγους απόδοσης να βρίσκεται μόνιμα φορτωμένο στην κύρια μνήμη.

Γνωρίζοντας ότι κάθε σελίδα έχει μέγεθος 25 KB κατά μέσον όρο, θα περιέχει περίπου $25KB/10B = 2560$ λέξεις, όπου 10B είναι το μέσο μέγεθος μιας λέξης. Συνολικά, όλη η συλλογή των εγγράφων θα περιέχει περίπου $3500000 \cdot 2560 = 8960000000$ λέξεις. Οπότε το μέγεθος των λιστών εμφάνισης, αν για την αποθήκευση μιας εμφάνισης απαιτούνται 4 bytes, θα είναι $8960000000 \cdot 4\text{bytes} = 33.37GB$.

Αθροίζοντας το μέγεθος του λεξιλογίου και των λιστών εμφάνισης, το μέγεθος του ανεστραμμένου ευρετηρίου θα είναι περίπου $33.37\text{GB} + 1.81\text{MB} = 33.371GB$.

Τέλος, στο http://groogle.csd.uoc.gr/apache2-default/index.php/Indexer_DBMS-free_2007 μπορείτε να δείτε και την εκτίμηση της ομάδας του Indexer για το ίδιο θέμα.