Wyszukiwanie informacji w sieci, Lista 1 **03.03.2007**

(zadania na ćw. w tygodniu 5.03–11.03.2007)

1. [2] Budujemy zapytania z termów oraz spójników logicznych *and*, *or* i *not*. Czy dla takich zapytań zawsze można wyznaczyć odpowiedź w czasie liniowym ze względu na sumę długości list adresów wszystkich termów występujących w zapytaniu i rozmiar wyniku? (w tym zadaniu rozmiar zapytania uznajemy za wielkość stałą)

   Zacznij od przykładu:

   (Brutus OR Caesar) AND NOT (Anthony OR Cleopatra)

   Wskazówka. Skorzystaj z praw rachunku zdań.

2. [1] Używając praw rachunku zdań przepisz zapytanie z powyższego przykładu z koniunkcyjnej postaci normalnej do dysjunkcyjnej postaci normalnej. W której postaci mniej czasu zajmie wyznaczenie odpowiedzi na zapytanie?

   Czy można stąd wyciągnąć jakieś ogólne wnioski na temat przewagi postaci koniunkcyjnej/dysjunkcyjnej?

3. [2] Załóżmy, że w indeksie odwróconym zindeksowano w sumie N dokumentów. Przyjmijmy też, że dla dwóch list adresów (odpowiadających termom bądź koniunkcji termów) o długościach n_1 i n_2, wartość oczekiwana długości części wspólnej tych list wynosi $\frac{n_1 \cdot n_2}{N}$.

   Niech (a_1 and a_2 and $\ldots$ and a_p) będzie zapytaniem, dla którego odpowiedzią są adresy dokumentów zawierających wszystkie termy ze zbioru $\{a_1, \ldots, a_p\}$. Rozważmy algorytm realizujący powyższe zapytanie w następujący sposób:

   (a) Niech L_i to lista dokumentów termu a_i.
   (b) $L \leftarrow \{L_1, \ldots, L_p\}$.
   (c) Powtarzaj $p-1$ razy:
      i. wybierz dwa różne elemeny $L' \neq L''$; $L', L'' \in L$;
      ii. $\bar{L} \leftarrow L' \cap L''$
      iii. $L \leftarrow L - \{L', L''\} \cup \{\bar{L}\}$.

   Podaj sposób wyboru L', L'' w każdym kroku algorytmu, tak aby czas **oczekiwany** realizacji całego algorytmu był najmniejszy.

   Sprawdź, czy Twoja metoda zawsze gwarantuje najkrótszy czas działania powyższej procedury.

4. [1] W trakcie tworzenia indeksu odwróconego wykonywane są m.in. operacje lematyzacji (lemmatization), tokenizacji (tokenization), normalizacji (normalization), usuwania *stop words*.

   Uporządkuj te operacje według kolejności, w jakiej są wykonywane, opisz ich rolę i na przykładzie zdania (lub kilku zdań) w języku polskim zilustruj te etapy (Twój przykład powinien ilustrować jak najwięcej problemów związanych z rozważanymi zagadnieniami).

   Podaj też znaczenie terminu stemming oraz wskaż, do którego z wymienionych wyżej etapów tworzenia indeksu on należy.

   Na koniec zaproponuj lub podaj przyjęte w literaturze tłumaczenia terminów angielskich występujących w tym zadaniu.

5. [1] Niech T oznacza zbiór wszystkich dokumentów, które indeksujemy, a $T_{pos}(Q)$ zbiór dokumentów, które powinny być zwrócone w odpowiedzi na zapytanie Q oraz $T_{neg}(Q) = T - T_{pos}(Q)$. Następnie, $T_A(Q)$ oznacza zbiór dokumentów zwracanych przez wyszukiwarkę A na zapytanie Q. Zdefiniujmy dwa pojęcia:

- precision: $|T_A(Q) \cap T_{pos}(Q)|/|T_A(Q)|$
- recall: $|T_A(Q) \cap T_{pos}(Q)|/|T_{pos}(Q)|$.

Opisz znaczenie tych pojęć "swoimi słowami", w języku naturalnym oraz zaproponuj (lub podaj przyjęte w literaturze) polskie terminy dla tych pojęć.

Następnie wskaż, które z poniższych zdań są prawdziwe. Odpowiedź uzasadnij.

(a) Stemming nigdy nie powoduje zmniejszenia precyzji.

(b) Stemming nigdy nie powoduje zmniejszenia recall.

(c) Stemming zwiększa rozmiar słownika.

(d) Stemming należy wykonywać w procesie indeksowania dokumentów, ale nie dla treści zapytania.

6. [1] Poniższe pary słów zostały zredukowane do tej samej formy przez stemmer Portera:

   (a) abandon/abandonment

   (b) absorbency/absorbent

   (c) marketing/markets

   (d) university/universe

   (e) volume/volumes

   Które z par Twoim zdaniem nie powinny być sprowadzone do tej samej formy? Podaj 3 przykłady par słów w języku polskim, dla których "typowy stemmer języka polskiego" spowoduje takie kłopotliwe sprowadzenie obu elementów do tej samej formy.

7. [1] Podaj algorytm wyznaczania odpowiedzi na zapytanie o frazę $term_1\ term_2 \ldots term_n$. Twój algorytm powinien minimalizować oczekiwany czas działania. Podaj intuicyjne uzasadnienie optymalności algorytmu.

   Uwaga. Przyjmij, że w pamięci komputera możemy przechowywać w tym samym czasie co najwyżej dwie listy adresów odpowiadające termom/frazom (oraz listę "wynikową").

   [1] Podaj formalne uzasadnienie optymalności Twojego algorytmu przy założeniu, że dla list adresów L_1 i L_2 odpowiadających frazom f_1 i f_2, wartość oczekiwana rozmiaru zbioru dokumentów zawierających frazę $f_1 f_2$ wynosi $|L_1| \cdot |L_2|/N$, gdzie N to liczba zindeksowanych przez wyszukiwarkę dokumentów. (pomińmy milczeniem kwestię czy takie założenie "pasuje" do rzeczywistości...)

8. (a) [2] W. Pugh wprowadził w pracy "Skip lists: A probabilistic alternative to balanced trees" hierarchiczne *skip lists*, które umożliwiają dostęp do elementów ze zbioru uporządkowanego, oraz operacje dodawania elementów, w oczekiwanym czasie $O(\log n)$. Zaprezentuj ten algorytm i odpowiednią strukturę danych (link do artykułu na stronie wykładu).

   (b) [2] Wykaż, że ten algorytm rzeczywiście wykonuje operacje w oczekiwanym czasie $O(\log n)$.

   Uwaga. Punkty za (b) nie są wliczane do sumy punktów do zdobycia na ćwiczeniach.