Eksploracja zasobów internetowych

Wykład 2

Pobieranie i przetwarzanie treści stron WWW

mgr inż. Maciej Kopczyński

Białystok 2014

Wstęp

Jedną z funkcji silników wyszukiwania danych, a właściwie ich modułów crawlerów, jest pobieranie danych ze stron WWW.

Pobierane dane są przetwarzane, zapisywane i wykorzystywane do zwracania wyników wyszukiwania przez kolejne moduły silnika wyszukiwania.

Crawler musi zawierać niezawodny parser języka HTML, gdyż wiele stron jest niezgodnych z jakimikolwiek standardami.

Rodzaje danych

Wyróżniane są dwa rodzaje danych zawierających treść dokumentów:

- *structured* – zawierające połączenia typu klucz – wartość,
- *unstructured* – nie zawierające tego typu połączeń, z reguły bez jakichkolwiek cech.

Rodzaje danych – structured

Dane typu *structured* zawierają określone atrybuty opisujące ich zawartość, np. bazy danych czy katalogi biblioteczne.

Przykładowo, wybranie stron WWW wydziałów pobranych ze strony głównej uczelni CCSU, które posiadają pracownię komputerową w przypadku wykorzystania języka SQL mogłoby wyglądać następująco:

```
SELECT * FROM Departments WHERE facilities=
”computer lab”
```

Rodzaj danych – unstructured

Dane typu *unstructured* nie zawierają żadnych cech opisujących ich zawartość. Jest to tekst, który jest zrozumiały jedynie dla człowieka. Dobrymi przykładami tego typu danych są książki czy gazety.

A jak to wygląda w przypadku sieci WWW ?

Sieć WWW składa się z obydwu typów danych, jednak przeważają dane typu *unstructured*.

Rodzaje danych – podsumowanie

Dane typu *structured* na pewno lepiej nadają się do przetwarzania za pomocą komputerów, jednak największym problemem jest przetworzenie danych typu *unstructured* do typu *structured*.

Tego typu konwersja musi być wykonywana przez człowieka lub przynajmniej przez niego nadzorowana.

Przykład:

- tworzenie katalogów bibliotecznych,
- tworzenie *Topic Directories*.

Technika IR

Ze względu na to że konwersja sieci WWW do danych typu *structured* jest bardzo problematyczna, powstała technika wyszukiwania danych znana jako *Information Retrieval* (*IR*).

Ideą *IR* jest wykorzystanie prostej zależności boolowskiej - jeśli dokument zawiera określone słowa kluczowe, to dokument jest istotny.
Jednak sformułowanie zapytania ze słów kluczowych zgodnego z intencją użytkownika należy do szukającego.

Technika IR

Technikę *IR* można podsumować jako wyszukiwanie istotnych danych za pomocą nieistotnych słów kluczowych.

Technika *IR* sprawdza się bardzo dobrze w przypadku danych bibliograficznych, przetwarzania czasopism oraz w sieciach WWW.

Dzięki temu, że strony WWW składają się ze znaczników opisujących ich treść, istnieje możliwość wzbogacenia techniki *IR* o sortowanie wyników pod względem istotności.

Zapytania

Dużym problemem w definiowaniu zapytań przez użytkownika jest rozbieżność pomiędzy tym, co użytkownik chce odnaleźć a tym, w jaki sposób definiowane jest zapytanie.

Jedną z metod radzenia sobie z problemem jest generowanie słów-podpowiedzi.

Statystyki pokazują, że średnia długość zapytania to 2 – 3 słowa.

Budowa crawlera

Moduł silnika wyszukiwania odpowiedzialny za przetwarzanie treści stron WWW składa się z:

- parsera języka HTML,
- indeksera odwiedzonych stron,
- algorytmów analizy danych:
  - magazynu przetworzonych danych,
  - konwertera języka naturalnego, w którym stworzona została strona.

Problemy w analizie treści

Parser treści strony HTML silnika wyszukiwarki jest narażony na wiele problemów:

- niezgodny ze standardami kod HTML,
- błędy w treści strony (np. literówki),
- obrazki zrozumiałe dla człowieka, ale nie dla komputera,
- różne formy i odmiany tych samych wyrazów,
- różne znaczenie tych samych wyrazów zależne od kontekstu.

Konwersja treści

Pobrane dane z treścią strony poddawane są procesowi obróbki:

- usunięcie znaków interpunkcyjnych,
- zamiana wszystkich znaków na duże bądź małe litery,
- przekształcenie słów do postaci podstawowej (ang. *stemming*),
- usunięcie słów nie wnoszących nic do treści strony, tj. przyimków, zaimków, itp. (ang. *stopwords*).

W wyniku obróbki danych, dokumenty są reprezentowane w postaci tzw. *termów*.

Cele przetwarzania tekstu

Przetworzenie treści zawartych na stronach WWW pozwala przede wszystkim na:

- usunięcie zbędnych słów,
- zmniejszenie zajętości pamięciowej,
- standaryzację zawartości dokumentów,
- łatwiejszą analizę danych (pozyskiwanie wiedzy, wyszukiwanie),
- zmniejszenie kosztu czasowego późniejszego przetwarzania danych.

Dane przykładowe

Strona internetowa *www.artsci.ccsu.edu*:

Dane przykładowe

Przykładowy zbiór danych ze strony *www.artsci.ccsu.edu*:

Document ID	Document name	Words	Terms
d_1	Anthropology	114	86
d_2	Art	153	105
d_3	Biology	123	91
d_4	Chemistry	87	58
d_5	Communication	124	88
d_6	Computer Science	101	77
d_7	Criminal Justice	85	60
d_8	Economics	107	76
d_9	English	116	80
d_{10}	Geography	95	68
d_{11}	History	108	78
d_{12}	Mathematics	89	66
d_{13}	Modern Languages	110	75
d_{14}	Music	137	91
d_{15}	Philosophy	85	54
d_{16}	Physics	130	100
d_{17}	Political Science	120	86
d_{18}	Psychology	96	60
d_{19}	Sociology	99	66
d_{20}	Theatre	116	80
Total number of words/terms		2195	1545
Number of different words/terms		744	671

Przechowywanie danych

Pobrane ze stron WWW oraz przetworzone dane muszą być gdzieś zapisane. Klasyczne mechanizmy baz danych są nieprzystosowane do tego typu zastosowań.

Termy określające treść stron WWW są przechowywane w odpowiednich strukturach danych przyspieszających wykonywanie na nich dalszych obliczeń.

Cechy optymalnej struktury

Charakterystyka odpowiednio dobranej struktury danych do przechowywania termów:

- szybkość wyszukiwania zadanych termów,
- minimalna ilość zajmowanej pamięci,
- możliwość zapisania ilości wystąpień termów,
- możliwość lokalizowania miejsc wystąpień termów w ciągu opisującym treść strony WWW,
- przejrzystość logiczna struktury.

Technika odwróconego indeksu

Wyszukiwanie za pomocą słów kluczowych w przypadku małych kolekcji dokumentów, takich jak strony opisujące wydziały na uczelni *CCSU*, może być implementowane jako bezpośrednie przeszukiwanie tekstu.

Jednak w przypadku dużych kolekcji dokumentów, rozwiązanie tego typu nie będzie się sprawdzało ze względu na duży koszt obliczeniowy.

Jaki szacunkowo będzie to koszt?
Kwadratowy – ilość dokumentów * ilość termów w dokumencie

Technika odwróconego indeksu

Struktury danych do przechowywania termów wykorzystują technikę odwróconego indeksu (*inverted index*). Kluczami w takich strukturach są termy, zaś wartościami dokumenty.

Co daje nam takie rozwiązanie?

Rozwiązanie to pozwala przyspieszyć przetwarzanie zapytań związanych ze słowami kluczowymi, gdyż dokumentów jest znacznie więcej niż wszystkich dostępnych termów.

Macierz typu boolowskiego

W macierzy term-dokument typu boolowskiego połączenia pomiędzy termami i dokumentami, które te termy zawierają jest realizowana poprzez przyporządkowanie jednej z dwóch wartości logicznych dla każdej komórki stworzonej macierzy.

Wiersze tej macierzy stanowią identyfikatory dokumentów, zaś kolumny reprezentują pobrane termy.

Macierz typu boolowskiego

Przykładowa macierz term-dokument typu boolowskiego:

DID	lab	laboratory	programming	computer	program
d_1	0	0	0	0	1
d_2	0	0	0	0	1
d_3	0	1	0	1	0
d_4	0	0	0	1	1
d_5	0	0	0	0	0
d_6	0	0	1	1	1
d_7	0	0	0	0	1
d_8	0	0	0	0	1
d_9	0	0	0	0	0
d_{10}	0	0	0	0	0
d_{11}	0	0	0	0	0
d_{12}	0	0	0	1	0
d_{13}	0	0	0	0	0
d_{14}	1	0	0	1	1
d_{15}	0	0	0	0	1
d_{16}	0	0	0	0	1
d_{17}	0	0	0	0	1
d_{18}	0	0	0	0	0
d_{19}	0	0	0	0	1
d_{20}	0	0	0	0	0

Macierz typu boolowskiego

Zalety macierzy typu boolowskiego:

- prostota przechowywania danych,
- łatwość wyszukiwania informacji.

Wady macierzy typu boolowskiego:

- brak możliwości dalszej obróbki danych,
- brak możliwości wyszukiwania słów leżących w określonych odległościach względem siebie.

Macierz typu ilościowego

W macierzy term-dokument typu ilościowego połączenia pomiędzy termami i dokumentami, które te termy zawierają jest realizowana poprzez zapisanie ilości wystąpień danego termu w ramach pojedynczego dokumentu.

Wiersze tej macierzy stanowią identyfikatory dokumentów, zaś kolumny reprezentują termy.

Macierz typu ilościowego

Zalety macierzy typu ilościowego:

- prostota przechowywania danych,
- łatwość wyszukiwania informacji,
- możliwość dalszej obróbki danych (jednak ograniczona!).

Wady macierzy typu ilościowego:

- brak możliwości wyszukiwania słów leżących w określonych odległościach względem siebie.

Macierz typu pozycyjnego

W macierzy term-dokument typu pozycyjnego połączenia pomiędzy termami i dokumentami, które te termy zawierają jest realizowana poprzez zapisanie pozycji wystąpienia danego termu w ciągu wszystkich termów tworzących dany dokument.

Wiersze tej macierzy stanowią identyfikatory dokumentów, zaś kolumny reprezentują termy.

Macierz typu pozycyjnego

Przykładowa macierz term-dokument typu pozycyjnego:

DID	lab	laboratory	programming	computer	program
d_1	0	0	0	0	[71]
d_2	0	0	0	0	[7]
d_3	0	[65,69]	0	[68]	0
d_4	0	0	0	[26]	[30,43]
d_5	0	0	0	0	0
d_6	0	0	[40,42]	[1,3,7,13,26,34]	[11,18,61]
d_7	0	0	0	0	[9,42]
d_8	0	0	0	0	[57]
d_9	0	0	0	0	0
d_{10}	0	0	0	0	0
d_{11}	0	0	0	0	0
d_{12}	0	0	0	[17]	0
d_{13}	0	0	0	0	0
d_{14}	[42]	0	0	[41]	[71]
d_{15}	0	0	0	0	[37,38]
d_{16}	0	0	0	0	[81]
d_{17}	0	0	0	0	[68]
d_{18}	0	0	0	0	0
d_{19}	0	0	0	0	[51]
d_{20}	0	0	0	0	0

Macierz typu pozycyjnego

Zalety macierzy typu pozycyjnego:

- łatwość wyszukiwania informacji,
- możliwość dalszej obróbki danych,
- możliwość wyszukiwania słów leżących w określonych odległościach względem siebie.

Wady macierzy typu pozycyjnego:

- trudniejsza implementacja struktury,
- większy narzut obliczeniowy związany z przetwarzaniem danych.

Implementacja struktur

Opisane struktury mogą być bezpośrednio zaimplementowane jako macierze dwuwymiarowe, jednak w przypadku dużych ilości danych rozwiązanie to będzie bardzo wymagające pamięciowo.

Tak więc pożądana implementacja struktur powinna być zrealizowana z wykorzystaniem tablic haszujących lub B-drzew. Dla struktur pozycyjnych pojedynczym elementem może być np.:

- *lab* → d_{14}:42
- *laboratory* → d_3:65,69
- *computer* → d_3:68; d_4:26; d_6:1,3,7,13,26,34; d_{12}:17; d_{14}:41

Implementacja struktur

W przypadku implementacji struktur należy zwrócić uwagę na dwie bardzo istotne kwestie:

- koszt czasowy oraz pamięciowy utworzenia struktury danych opisującej indeks,
- koszt czasowy aktualizacji stworzonego wcześniej indeksu.

Przykładowe rozmiary danych:

- tekst wszystkich książek z *U.S. Library of Congress* – ok. 20 TB,
- tekst wszystkich stron pobranych przez *Yahoo!* do 2005 r. - 200 TB.

Indeksowanie danych

W konkursie na indeksowanie dokumentów *Text Retrieval Conference 2004*, jedno z rozwiązań potrafiło stworzyć indeks 25 000 000 dokumentów zajmujących ok. 426 GB z wykorzystaniem 6 komputerów w przeciągu 6 godzin.

Zakładając liniową zależność czasu przetwarzania do wzrostu rozmiaru danych, można oszacować, że indeksowanie danych zebranych przez *Yahoo!* zajmuje ok. 3000 h.

Indeksowanie danych

Jaki powinien być maksymalny czas tworzenia indeksu zebranych po raz kolejny stron?

Czas tworzenia indeksu na bazie zebranych dokumentów powinien być krótszy niż średni czas zmiany zawartości na statystycznej stronie WWW.

Indeksy stron webowych powinny być budowane oraz aktualizowane bardzo szybko .

Indeksowanie danych

Należy również pamiętać o maksymalnym czasie oczekiwania na zwrócenie wyników zgodnych z zapytaniem (ang. *query time*).

Statystyki pokazują, że typowy użytkownik jest w stanie oczekiwać na wyniki wyszukiwarki maksymalnie kilka sekund, zaś najbardziej pożądany jest czas nie przekraczający 1 sekundy.

Dobry mechanizm indeksowania musi być kompromisem pomiędzy kosztem pamięciowym oraz kosztem obliczeniowym.

Dziękuję za uwagę!