Przetwarzanie danych sieciowych

Wyszukiwarki i roboty internetowe

January 9, 2018

Plan

1. Idea robotów internetowych
2. Działanie robotów internetowych
3. Techniki przeszukiwania
4. Reprezentacja dokumentów internetowych
5. Wyszukiwanie i indeksowanie

Wstęp

Niemożliwe jest przeszukanie sieci jako wielkiej bazy danych przy obecnych ograniczeniach mocy obliczeniowej w czasie rzeczywistym. Aby umożliwić wyłuskiwanie danych najbardziej pasujących do wyszukiwanych, potrzebne są zaawansowane metody przetwarzania i eksploracji danych.

Idea i działanie robotów internetowych

Problem przeszukiwania sieci

Dokumenty html zawierają wiele znaczników nieniosących wyszukiwanych informacji.

Idea i działanie robotów internetowych

Problem przeszukiwania sieci

Dokumenty html zawierają wiele znaczników nieniosących wyszukiwanych informacji.

Fundament działania robota internetowego

Robot internetowy omija znaczniki plików html, wyłuskując z dokumentów wyłącznie treść.

Idea i działanie robotów internetowych

Problem przeszukiwania sieci

Dokumenty html zawierają wiele znaczników nieniosących wyszukiwanych informacji.

Fundament działania robota internetowego

Robot internetowy omija znaczniki plików html, wyłuskując z dokumentów wyłącznie treść.

Czy to powód ich stosowania? Nie.

Implementacja takiego rozwiązania jest łatwa, na dodatek istnieje wiele bibliotek wykonujących taką pracę za programistę.
Najpoważniejszy problem jest następujący:

Idea i działanie robotów internetowych

Problem przeszukiwania sieci

Sieć jest nieustannie aktualizowana. Nie znamy miejsc, w których zostanie zaktualizowana.

Idea i działanie robotów internetowych

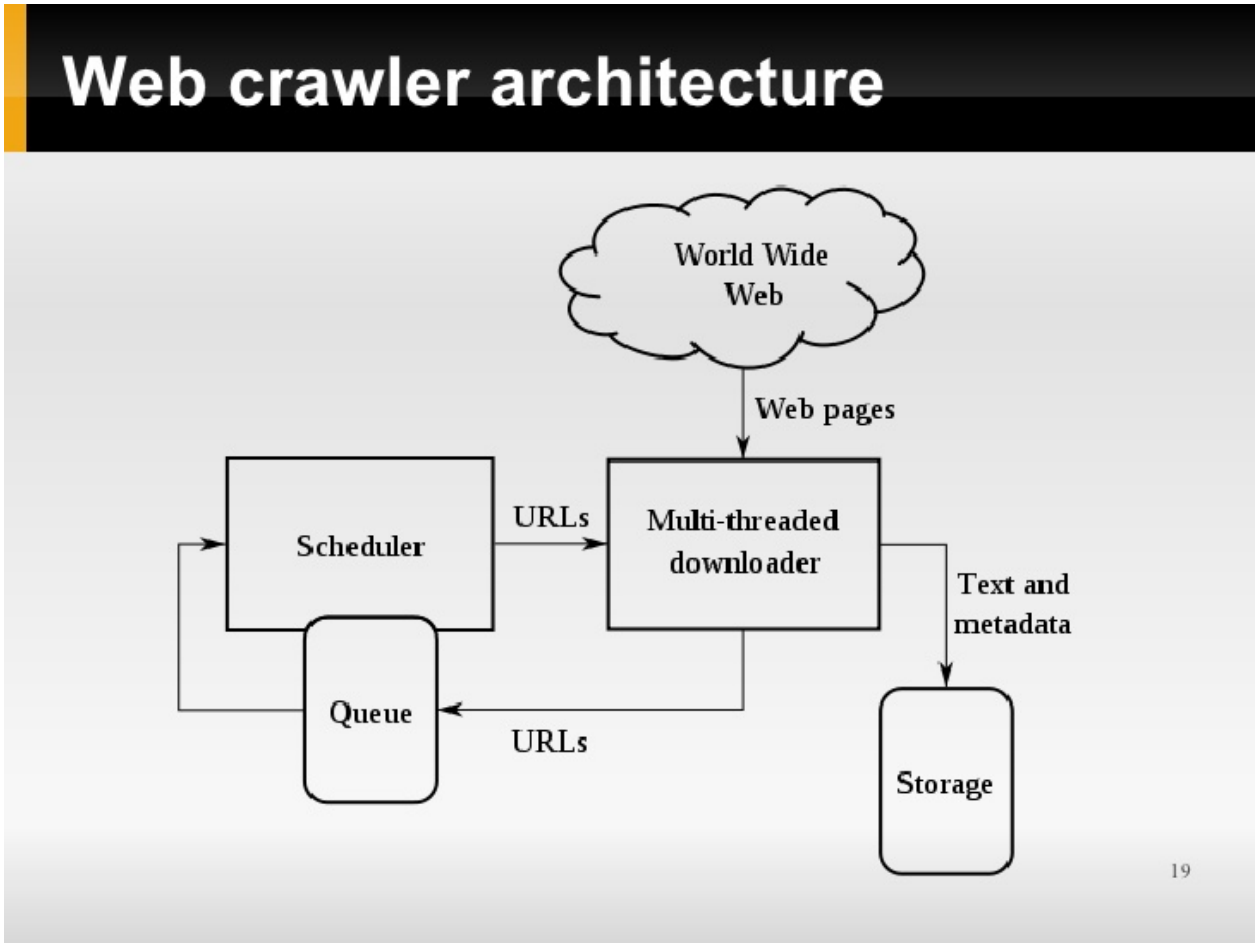
Problem przeszukiwania sieci

Sieć jest nieustannie aktualizowana. Nie znamy miejsc, w których zostanie zaktualizowana.

Powód stosowania robotów internetowych

Roboty internetowe działają współbieżnie. Wykorzystuje się sumę zgromadzonych przez nie informacji.

Uproszczony schemat działania pełzacza internetowego



Techniki i sposoby przeszukiwań

Podział ze względu na przeszukiwanie grafu połączeń:

- DFS
- BFS
- DFS z parametrem ograniczającym głębokość przeszukiwania

Techniki i sposoby przeszukiwań

Podział ze względu na topologię przeszukiwań:

- Roboty są relatywnie niedużo linków od siebie
- Roboty przeszukują odległe obszary sieci

Techniki i sposoby przeszukiwań

Podział ze względu na topologię przeszukiwań:

- Roboty są relatywnie niedużo linków od siebie
- Roboty przeszukują odległe obszary sieci

Czy powyższy podział ma sens?

Z punktu widzenia robota - nie (dlaczego?).

Techniki i sposoby przeszukiwania

Googlebot wykorzystuje dwie techniki przeszukiwania:

- deep crawl
- fresh crawl

Do ustalenia punktów startowych fresh crawl może posłużyć grupowanie (do zastosowań sieciowych najczęściej implementuje się algorytm k-means).

Techniki i sposoby przeszukiwania

Problemy przeszukiwania sieci przez roboty:

- Komunikacja między wątkami
- Robot, aby pobrać zawartość strony, musi ją "odwiedzić"- brak anonimowości / dyskretności
- tzw. mirror sites
- żadne przeszukiwanie nie odzwierciedla faktycznego stanu sieci

Przygotowanie zbioru dokumentów dla wyszukiwarki

Zawsze wykonymane operacje:

- usuwanie tzw. stopwords
- lematyzacja - morfologiczne grupowanie słów

Dokument	Nazwa dokumentu	Słowa	Termy
d1	Anthropology	114	86
d2	Art	153	105
d3	Biology	123	91
d4	Chemistry	87	58
d5	Communication	124	88

Fragment przykładowej bazy danych po zastosowaniu powyższych metod

Macierz term-dokument

Istnieją trzy zasadnicze reprezentacje macierzy term-dokument:

`Macierz binarna`

Dokument	lab	laboratory	programming	computer	program
d1	0	0	0	0	1
d2	0	0	0	0	1
d3	0	1	0	1	0
d4	0	0	0	1	1
d5	0	0	0	0	0

`Macierz częstości`

Dokument	lab	laboratory	programming	computer	program
d1	0	0	0	0	1
d2	0	0	0	0	1
d3	0	2	0	1	0
d4	0	0	0	1	2
d5	0	0	0	0	0

Macierz term-dokument

Macierz wystąpień

Dokument	lab	laboratory	programming	computer	program
d1	0	0	0	0	[71]
d2	0	0	0	0	[7]
d3	0	[65, 69]	0	[68]	0
d4	0	0	0	[26]	[30, 43]
d5	0	0	0	0	0

Inne reprezentacje term-dokument

Problem

Macierze term-dokument są rzadkie, a danych jest bardzo dużo.

Używa się również innych struktur, takich jak B-drzewa i funkcje haszujące, które ograniczają złożoność pamięciową.

Model przestrzeni wektorowej

Binarne wyszukiwanie zwraca jedynie nieuporządkowany zbiór dokumentów. Aby możliwe było indeksowanie stron według ważności, należy określić, które strony są bliższe wyszukiwanym słowom, a które dalsze.

Model przestrzeni wektorowej

Binarne wyszukiwanie zwraca jedynie nieuporządkowany zbiór dokumentów. Aby możliwe było indeksowanie stron według ważności, należy określić, które strony są bliższe wyszukiwanym słowom, a które dalsze.

Model przestrzeni wektorowej (VSM)

Niech n_{ij} - liczba wystąpnień termu t_i w dokumencie d_j. W reprezentacji binarnej dokument $d_j = (d_j^1 d_j^2 \dots d_j^m)$, gdzie:

$$d_j^i = \begin{cases} 0 & \text{dla} \quad n_{ij} = 0 \\ 1 & \text{dla} \quad n_{ij} > 0 \end{cases}$$

Reprezentacja TF - term frequency

Rozważmy dwa dokumenty - jeden o długości 1000000 różnych słów, drugi o długości 100 słów, oba zawierają wyszukiwane wyrażenie.

Reprezentacja TF - term frequency

Rozważmy dwa dokumenty - jeden o długości 1000000 różnych słów, drugi o długości 100 słów, oba zawierają wyszukiwane wyrażenie.

Problem

Używając reprezentacji binarnej dla powyższych dokumentów i standardowej odległości między wektorami (euklidesowej, Manhattan, Mahalanobisa) zostaną zindeksowane na tym samym miejscu.

Reprezentacja TF - term frequency

Aby temu zaradzić, używa się miary TF. Dla każdego termu w dokumencie obliczana jest jego lokalna miara ważności TF:

- używając liczby będącej sumą wystąpień wszystkich termów:

$$TF(t_i, d_j^i) = \begin{cases} 0 & \text{dla} \quad n_{ij} = 0 \\ \frac{n_{ij}}{\sum_{k=1}^{m}} & \text{dla} \quad n_{ij} > 0 \end{cases}$$

- używając maksymalnej liczby wystąpień termu spośród wszystkich termów:

$$TF(t_i, d_j^i) = \begin{cases} 0 & \text{dla} \quad n_{ij} = 0 \\ \frac{n_{ij}}{\max_k n_{kj}} & \text{dla} \quad n_{ij} > 0 \end{cases}$$

Reprezentacja TF - term frequency

- używając skali logarytmicznej:

$$TF(t_i, d_j^i) = \begin{cases} 0 & \text{dla} \quad n_{ij} = 0 \\ 1 + \log\left(1 + \log n_{ij}\right) & \text{dla} \quad n_{ij} > 0 \end{cases}$$

Miara IDF - inverse document frequency

Problem

Słowo, które pojawia w wielu dokumentach, jest mniej istotne przy wyszukiwaniu niż to pojawiające się rzadko.

Miara IDF termu t_i określa ważność tego termu wśród wszystkich dokumentów. Określa się ją na różne sposoby:

- jako prosty stosunek liczby wszystkich dokumentów do liczby dokumentów, w których dany term występuje:

$$IDF(t_i) = \frac{|D|}{|\{d_j : n_{ij} > 0\}|}$$

- używając skali logarytmicznej:

$$IDF(t_i) = \log \frac{1 + |D|}{|\{d_j : n_{ij} > 0\}|}$$

Reprezentacja TFIDF

TFIDF

Określamy każdą współrzędną wektora dokumentu:

$$d_j^i = TF(t_i, d_j)IDF(t_i)$$

Miara podobieństwa

Najpopularniejsze miary podobieństwa:

- norma euklidesowa

$$\|q - d_k\| = \sqrt{\sum_{i=1}^{m}(q^i - d_j^i)^2}$$

- odległość kosinusowa

$$\|q - d_k\| = qd_j = \sum_{i=1}^{m} q^i d_j^i$$

TFIDF dla przykładowych dokumentów

Wyszukiwanie i indeksowanie wektora q = (0 0 0 1 1),
po przeskalowaniu i normalizacji
q = (0 0 0 0.932 0.363)

Dokument	TFIDF (znormalizowane)	ranga cos	ranga euc
d1	0 0 0 0 1	0.363	1.129
d2	0 0 0 0 1	0.363	1.129
d3	0 0.972 0 0.234 0	0.218	1.250
d4	0 0 0 0.783 0.622	0.956	0.298
d5	0 0 0 0 1	0.363	1.129

Metody relevace feedback - sprzężenia zwrotnego

Myśl przewodnia - im częściej dokument jest wyświetlany / modyfikowany / wysoko indeksowany, tym wyżej powinien być w wyszukiwaniu innych informacji w przyszłości.

Metody relevace feedback - sprzężenia zwrotnego

Myśl przewodnia - im częściej dokument jest wyświetlany / modyfikowany / wysoko indeksowany, tym wyżej powinien być w wyszukiwaniu innych informacji w przyszłości.

Problem - współrzędne dokumentów są stałe

Rozwiązanie - można zmodyfikować wektor zapytania.

Metoda Rocchio

Niech D_+ - zbiór dokumentów istotnych, D_- - nieistotnych, q - wektor zapytania. Nowy wektor zapytania obliczany jest w następujący sposób:

$$q' = \alpha q + \beta \sum_{d_j \in D_+} d_j - \gamma \sum_{d_j \in D_-} d_j$$

Metoda Rocchio

Niech D_+ - zbiór dokumentów istotnych, D_- - nieistotnych, q - wektor zapytania. Nowy wektor zapytania obliczany jest w następujący sposób:

$$q' = \alpha q + \beta \sum_{d_j \in D_+} d_j - \gamma \sum_{d_j \in D_-} d_j$$

Problem

Ustalenie współczynników α, β, γ

Inne metody poprawiania wyszukiwania słów kluczowych

- Używanie znaczników HTML do wyszukiwania
- Wyszukiwanie ciągów zbliżonych

Ocena jakości wyszukiwania

Cel

Aby zbiory istotnych i wyszukanych dokumentów pokrywały się. Wprowadzamy oceny jakości wyszukiwania:

- dokładność (precision):

$$\frac{|D_q \cap R_q|}{|r_q|}$$

- kompletosć (recall):

$$\frac{|D_q \cap R_q|}{|D_q|}$$

Ocena jakości wyszukiwania

Problem

Dla zbiorów zwracanych przez wyszukiawrki obliczenie D_q jest w praktyce niemożliwe. Wprowadza się inne miary, obcięte do k początkowych wartości.

- dokładność(k):

$$\frac{1}{k}\sum_{i=1}^{k} r_i$$

- kompletność(k):

$$\frac{1}{|D_q|}\sum_{i=1}^{k} r_i$$

Gdzie r_i binarnie określa przynależność dokumentu d_i do zbioru istotnych dokumentów D_q.

Ranking oparty na strukturze połączeń

Dla każdego dokumetu określamy jego prestiż - funkcję liczby jego cytowań. Prestiż ma charakter rekurencyjny.

Ranking oparty na strukturze połączeń

Dla każdego dokumetu określamy jego prestiż - funkcję liczby jego cytowań. Prestiż ma charakter rekurencyjny.

Niech macierz sąsiedztwa A określa binarnie cytowanie dokumenów:

$$A(u,v) = \begin{cases} 1 & \text{dokument u cytuje dokument v} \\ 0 & \text{w przeciwnym razie} \end{cases}$$

Ranking oparty na strukturze połączeń

Dla każdego dokumetu określamy jego prestiż - funkcję liczby jego cytowań. Prestiż ma charakter rekurencyjny.

Niech macierz sąsiedztwa A określa binarnie cytowanie dokumenów:

$$A(u,v) = \begin{cases} 1 & \text{dokument u cytuje dokument v} \\ 0 & \text{w przeciwnym razie} \end{cases}$$

Definiujemu prestiż węzła u jako sumę wartości prestiżu dokumentów go cytujących:

$$p(u) = \sum_{v} A(u,v)p(v)$$

Ranking oparty na strukturze połączeń

Obliczone wartości prestiżu dla wszystkich węzłów możemy zapisać jako wektor kolumnowy P. Następnie obliczamy wektor prestiżu P':

$$P' = A^T P$$

Ranking oparty na strukturze połączeń

Obliczone wartości prestiżu dla wszystkich węzłów możemy zapisać jako wektor kolumnowy P. Następnie obliczamy wektor prestiżu P':

$$P' = A^T P$$

Okazuje się, że istnieje punkt stały λ taki, że

$$\lambda P = A^T P$$

Wybieramy wektor odpowiadający największej wartości własnej. Jego komórki przyjmuje się jako wartości prestiżu dokumentów.

Zastosowania obliczonego prestiżu dokumentów

Zastosowania:

1. algorytm PageRank
2. modyfikacja ważności dokumentów
3. wyznaczenie autorytetów

Bibliografia

 Zdravko Markov, Daniel T. Larose, Eksploracja zasobów internetowych