

# Indexarea documentelor folosind paradigma Map-Reduce

Responsabil de temă: Elena Apostol

Data publicării: 11/09/2012

Termenul de predare: 23/09/2012

**Observatie:** Metoda **main** va trebui sa se afle intr-o clasa cu numele **Main** si **nu** va fi inclusa in vreun pachet.

## 1. Introducere. Cerințele temei

Pentru a facilita obținerea rapidă și precisă a informațiilor existente în paginile web, motoarele de căutare indexează (colectează, parsează și stochează) datele înainte să efectueze căutări. Procesul se mai numește și Web indexing atunci când se referă la indexarea paginilor existente în Internet.

Se poate face o căutare simplă, după un cuvânt cheie într-un document neindexat, se poate face indexarea unui document în timp ce se face o căutare în el pentru prima dată, sau se pot indexa în prealabil toate documentele. Pentru optimizarea timpului de căutare, motoarele de căutare cele mai populare fac indexarea totală a textului existent online.

### Cerințele temei

În această temă se cere scrierea unui program paralel în Java care să facă indexarea unui set de documente primit ca input și apoi căutare după cuvinte cheie în documentele indexate. În urma procesului de indexare se determină numărul de apariții al fiecărui cuvânt existent într-o pagină web sau document, obținându-se o listă de perechi (cuvânt, număr de apariții). Programul trebuie să permită efectuarea de căutări după cuvinte cheie într-un set de documente și afișarea documentelor cu gradul maxim de relevanță. Căutările se pot face după unul, două sau mai multe cuvinte cheie.

Pentru paralelizarea indexării se va folosi paradigma replicated workers (vezi laborator 5) și modelul MapReduce, ideea generală fiind: fiecare document se fragmentează în părți de dimensiune fixă, care vor fi procesate în paralel, pentru fiecare parte rezultând câte un vector parțial conținând termenii cu frecvența cea mai mare în fragmentul respectiv. Apoi vectorii sunt combinați și va rezulta un vector ce caracterizează întregul document. Notăm lungimea finală a vectorului cu  $N$  (adică păstrăm cele mai frecvente  $N$

cuvinte). Pentru ca un document sa fie returnat ca relevant intr-o anumita cautare, cuvantul/cuvintele cheie dupa care se face cautarea trebuie sa se afle in vectorul care contine termenii cu frecventele cele mai mari de aparitie.

## 2. Implementare

### 2.1 Paradigma Map-Reduce - Prezentare generala

Pentru rezolvarea acestei probleme se va folosi un model replicated-workers, asemanator cu modelul MapReduce folosit de inginerii de la Google pentru procesarea unor seturi mari de documente in sisteme paralele si distribuite. [Acest articol](#) prezinta modelul MapReduce folosit de Google si o parte dintre aplicatiile lui (mai importante pentru intelegerea modelului sunt primele 4 pagini).

MapReduce este un model de programare paralela (si implementarea asociata) pentru procesarea si generarea unor seturi imense de date folosind sute sau mii de procesoare. Modelul permite paralelizarea si distributia automata a taskurilor. Paradigma MapReduce se bazeaza pe existenta a doua functii: map si reduce. Map primeste ca input o functie f si o lista si returneaza o noua lista aplicand functia f fiecarui element din lista. Reduce combina rezultatele obtinute anterior.

Mecanismul MapReduce functioneaza in modul urmator:

- Utilizatorul cere procesarea unui set de documente; aceasta cerere este adresata unui proces (fir de executie) master.
- Master-ul imparte documentele in fragmente de dimensiuni fixe care vor fi asignate unor procese (fire de executie) worker; un worker va executa pentru un fragment de fisier o operatie numita "map", care va genera niste rezultate parțiale avand forma unor perechi de tip (cheie, valoare).
- Dupa ce operatiile "map" au fost executate, master-ul asigneaza worker-ilor task-uri de tip "reduce", care combina rezultatele parțiale.

### 2.2 Map-Reduce - Cerinte pentru problema propusa

Dandu-se un set de ND documente si un set de NC cuvinte de cautat in documentele respective, sa se determine primele X documente cu relevanta maxima pentru cautare.

Prin document cu relevanta maxima intr-o cautare dupa un set de cuvinte cheie se intelege documentul in care cuvintele date au frecventa cea mai mare de aparitie

**Frecventa de aparitie** a unui cuvant la nivel de document:

$$\text{frecventa} = (\text{nr\_aparitii\_cuvant} / \text{nr\_total\_cuvinte}) * 100 [\%]$$

**Pentru cerinta descrisa mai sus se considera:**

- **[MAP]:** Impartirea fisierelor se va face in fragmente de cate aproximativ D octeti (cu exceptia ultimului fragment, care poate fi mai scurt).

**Observatie:** Poate aparea problema ca fragmentul prelucrat de un worker sa se termine sau sa inceapa in mijlocul unui cuvant. Se va adopta urmatoarea conventie: daca fragmentul incepe in mijlocul unui cuvant, worker-ul va "sari peste" acel cuvant; daca fragmentul se termina in mijlocul unui cuvant, worker-ul va prelucra si acel cuvant. In acest mod, cuvintele care sunt "la granita" dintre doua fragmente vor fi prelucrate tot timpul de worker-ul ce se ocupa de fragmentul care este in fisier inaintea cuvantului respectiv.

Astfel un task de tip "map":

- poate primi ca input : numele fisierului, pozitia de inceput din fisier, si pozitia de sfarsit
- intoarce ca output: perechi de tip (cheie, valoare), in cazul problemei de fata: (nume document, vector partial). Vectorul partial va contine setul de cuvinte impreuna cu frecventele acestora.
- **[REDUCE]:** In cazul problemei propuse, operatia "reduce" se face in 2 etape:
  - In **prima** operatie de "reduce" : se vor aduna frecventele cuvintelor din vectorii obtinuti pentru fiecare parte dintr-un document, si se pastreaza primele N cuvinte cu cele mai mari frecvente.
  - **A doua** operatie de tip "reduce": se determina documentele care contin in vectorul de cuvinte cu frecvente maxime de aparitie, toate cele NC cuvinte cheie primite (cuvintele dupa care se face cautarea). Este necesar ca toate cuvintele cheie sa apara in vectorul de cuvinte cu frecv maxime pentru toate cele X (sau mai putine - in cazul in care nu exista cel putin X documente cu proprietatea de mai sus) documente ce vor fi afisate ca rezultat in final.

### 2.3 Observatii Generale:

- Avand la dispozitie memoria partajata intre thread-uri, rezultatele operatiilor "map" vor fi tinute in memorie; in mod normal ele s-ar fi scris si pe disc.
- Ca mod de executie, puteti folosi (desi nu este obligatoriu) obiectele de tip "thread pool" care sunt deja implementate in Java (vezi interfata *ExecutorService*); astfel, un thread worker poate prelucra mai multe task-uri.
- Pentru simplificare puteti utiliza mai multe thread pool-uri – de ex. unul pentru operatiile de tip "map", si unul pentru operatiile de tip "reduce".
- Cuvintele pot fi de orice dimensiune si pot contine doar litere
- Compararea intre cuvinte nu va fi case sensitive (veti transforma toate cuvintele in cuvinte cu litere mici inainte de a le prelucra).
- Afisati frecventele cu 2 zecimale, obtinute prin trunchiere (nu prin rotunjire).

### 3. Formatul datelor de intrare/ieșire.

Programul va avea ca argumente în linia de comandă: NT (numărul de thread-uri worker), numele unui fișier de intrare și numele unui fișier de ieșire (în această ordine).

**Observatie:** Se vor porni NT thread-uri de tip Map, respectiv NT thread-uri de tip Reduce.

Fisierul ce conține datele de intrare are următorul format:

- *pe linia I:* numărul NC de cuvinte cheie de căutat
- *pe linia II:* cele NC cuvinte cheie despartite prin spațiu
- *pe linia III:* dimensiunea D (în octeți) a fragmentelor în care se vor împărți fișierele
- *pe linia IV:* numărul N (cele mai frecvente N cuvinte reținute pentru fiecare document)
- *pe linia V:* numărul X reprezentând câte documente vreau să primesc ca răspuns (ex. vreau să mi se returneze primele X documente cele mai relevante pentru căutare)
- *pe linia VI:* numărul ND de documente de tip text de indexat și în care se va face căutarea
- *pe următoarele ND linii:* numele celor ND documente (câte unul pe linie)

În fișierul de ieșire, pentru setul de cuvinte cheie de căutat se vor afișa numele primelor X documente cu relevanță maximă pentru căutarea făcută -- fiecare nume pe câte un rând, în ordinea apariției lor în fișierul de intrare -- împreună cu frecvențele cuvintelor căutate.

**Formatul fișierului de ieșire este următorul:**

Rezultate pentru: (cuvant\_1, cuvant\_2, ..., cuvant\_NC)

DOCUMENT\_1 (frecventa\_1, frecventa\_2, ..., frecventa\_NC)

...

DOCUMENT\_X (frecventa\_1, frecventa\_2, ..., frecventa\_NC)

, unde frecventa\_i este frecvența cuvântului cuvant\_i

### 4. Teste.

Pentru a verifica funcționalitatea temei puteți folosi aceste teste .

### 5. Resurse (opțional)

[The Anatomy of a Large-Scale Hypertextual Web Search Engine](#)

[Alt articol introductiv despre MapReduce](#)

[MapReduce on Wikipedia](#)