Recenzie articol

Parallel and Distributed Collaborative Filtering: A Survey

EFTHALIA KARYDI and KONSTANTINOS MARGARITIS, University of Macedonia

IEREMIAȘ Viorel | Gr. 244-2

Servicii Web şi tehnologii Middleware

Cuprins

[1. Abstract 2](#_Toc40818433)

[2. Structura 2](#_Toc40818434)

[3. Continut 3](#_Toc40818435)

[3.1. Sectiunea 1 - Introduction 3](#_Toc40818436)

[3.2. Sectiunea 2 – Related work 4](#_Toc40818437)

[3.3. Sectiunea 3 – Distributed implementations 5](#_Toc40818438)

[3.4. Sectiunea 4 – Parallel implementations 7](#_Toc40818439)

[3.5. Sectiunea 5 – Platform-based recommendations 9](#_Toc40818440)

[3.6. Sectiunea 6 – Heterogenous implementations 10](#_Toc40818441)

[3.7. Sectiunea 7 – Discussion and conclusions 10](#_Toc40818442)

[4. Surse bibliografice 11](#_Toc40818443)

# Abstract

Articolul este o lucrare de tip studiu de caz, care analizeaza folosirea tehnicilor specifice sistemelor paralele si distribuite in implementarea sistemelor de recomandare, in mod specific a sistemelor de tip collaborative filtering.

Obiectivul autorilor este in primul rand de a oferi o prezentare cuprinzatoare a evolutiei tehnicilor aplicate in domeniu. Pe de alta parte, se doreste conturarea directiilor de cercetare viitoare prin evidentierea principalelor lipsurilor actuale, care necesita dezvoltare ulterioara.

Motivatia sta in faptul ca, desi este studii de caz in literatura de specialitate referitoare la sistemele de recomandare, acestea nu abordeaza problema in context paralel sau distribuit.

Premisa de la care se pleaza este ca performanta sistemelor de recomandare, adica acuratetea recomandarii, este dependenta de cantitatea de informatie folosita. Cu alte cuvinte, cu cat sunt mai multe date disponibile si procesate, cu atat recomandarile sunt mai precise. Provocarile pe care le implica volume din ce in ce mai mari de date determina necesitatea proiectarii sistemelor de recomandare ca sisteme paralele sau distribuite. Insa, printre beneficiile proiectarii unui sistem de recomandare in acest fel nu se numara doar calitatea superioara a recomandarilor, ci si caracteristici non-functionale imbunatatite (performanta, eficienta, disponibilitate, elasticitare, etc.).

# Structura

Introducerea articolului prezinta motivatia lucrarii si implicit premisa de la care se pleaca. Pornind de la nevoia folosirii unor volume din ce in ce mai mari de date in implementarea sistemelor de recomadare, autorii prezinta evolutia utilizarii tehnicilor de procesare distribuita si paralela in proiectarea acestora si contureaza directiilor viitoare de dezvoltare in acest domeniu.

Tot in introducere se prezinta conceptele principale ale celor doua arii principale: sisteme paralele si distribuite si sisteme de recomadare. Pentru sistemele paralele si distrbuite, se prezinta caracteristicile principale ale diferitelor arhitecturi. Pentru sistemele de recomadare, se prezinta o taxonomie bazata pe algoritmul folosit.

Sectiunea a doua detaliaza lucrarile din literatura de specialitate care realizeaza studii de caz asupra sistemelor de recomandare. Dupa cum s-a mentionat, niciunul nu abordeaza tema in context paralel sau distribuit. Sectiunea a treia prezinta implementari distribuite ale sistemelor de recomandare, in timp de sectiunea a patra priveste implementarile paralele. Tot aici se face o clasificare in trei categorii: sisteme care functioneaza intr-un mediu cu memorie distribuita, sisteme care functioneaza intr-un mediu cu memorie comuna, si sisteme care folosesc acceleratoare GPU.

Sectiunile 5 si 6 se concentreaza pe metode hibride care inglobeaza mai multe dintre metodele prezentate in sectiunile anterioare. Ultima sectiune prezinta concluziile studiului de caz.

# Continut

## Sectiunea 1 - Introduction

**Sisteme paralele si distribuite**

Un sistem distribuit (*distributed memory system*) este o colectie de calculatoare conectate impreuna intr-o retea, executand simultan o sarcina. Comunicarea si sincronizarea intre noduri se face prin schimb de mesaje peste retea (protocoale de tip MPI). Cele mai frecvente arhictecturi sunt *client-server* si *peer-to-peer* (P2P). Din exterior, un computer cluster este vazut ca o resursa de calcul singulara, desi este compus din mai multe noduri, care pot fi identice (*homogenous cluster*) sau diferite (*heterogenous cluster*).

Un sistem paralel este o colectie de procesoare cu un spatiu de memorie comun (*shared memory system*) in care comunicarea intre procesoare este implicita, iar sincronizare este realizata prin acces diferential la memorie, operatii atomice si mecanisme de sincronizare precum bariere. Sistemele paralele sunt deseori accelerate folosind *GPUs* – co-procesoare care folosesc un model de programare de tip *data parallel*.

Ambele abordari se bucura de un numar mare de implementari, pentru diferite limbaje de programare. Cel mai adesea insa, cerintele sistemului care se implementeaza impun combinarea tehnicilor si framework-urilor pentru a obtine solutia cea mai buna.

**Sisteme de recomandare**

Sistemele de recomandare sunt programe care fac recomandari de obiecte unor utilizator. Scopul sistemelor de recomandare este de a oferi recomandari personalizate utilizatorilor, care sa fie interesante si utile in acelasi timp.

Sistemele de tip *collaborative filtering* (sau *user-based*) exploateaza faptul ca utilizatori cu un set similari de alegeri in trecut probabil o sa faca alegeri asemenatoare in continuare. Sistemele de tip *content-based* calculeaza similatitea intre obiecte luand in calcul caracteristici specifice. *Demographic recommenders* folosesc indici demografici despre utilizatori pentru a forma categorii. Sistemele *knowledge-based* folosesc informatii specifice domeniului care cuantifica cat de util este un obiect pentru un utilizator. Sistemele *community-based* ofera recomandari pe baza preferintelor unor grupuri de utilizatori.

Sistemele de tip *collaborative filtering* se clasifica de asemenea in:

* *memory-based* –cunoscute si ca *neighbordhood-based,* sunt sisteme in care setul de date este folosit pentru a extrage un grup de utilizatori care sunt cei mai apropiati de utilizatorul pentru care se face recomandarea, utilizand o metrica de similaritate; in final, se face recomandarea luand in calcul preferintele utilizatorilor din grupul calculate anterior
* *model-based* – setul de date este folosit pentru a antrena un model; ulterior acel model este folosit pentru a face predictii; obiectivul modelului este de a recunoaste tiparele de comportament ale utilizatorilor in setul de date; diferiti algoritmi de *machine-learning* si *data-mining* sunt folositi pentru a crea modelul

Evaluarea sistemelor de recomandare este de asemenea o problema de interes. Recomandarile pot sa tinteasca obiecte pe care utilizatorul este foarte probabil sa le foloseasca (*trustful recommendation*), sau obiecte noi pe care sa le propuna (*novel recommendation*). Asadar, evaluarea trebuie sa ia in considerare scopul principal pentru care a fost proiectat sistemul.

## Sectiunea 2 – Related work

Sectiunea face o trecere in revista a lucrarilor din literatura de specialitate care realizeaza studii de caz asupra sistemelor de recomandare.

**Studii de caz generale**

Prezinta lucrarile care:

* construiesc taxonomii generale ale sistemelor de recomandare; taxonomiile se bazeaza pe elementul central dupa care se face recomandarea (similaritatea intre utilizatori, similaritatea intre obiecte, metode hibride) sau dupa tehnica de data mining utilizata.
* prezinta evolutia abordarilor folosite in domeniul sistemelor de recomandare

**Studii de caz privind metodele utilizate**

Lucrarile din aceasta categorie prezinta diferitele metode folosite in implementarea sistemelor de recomandare. Scopul lor este de:

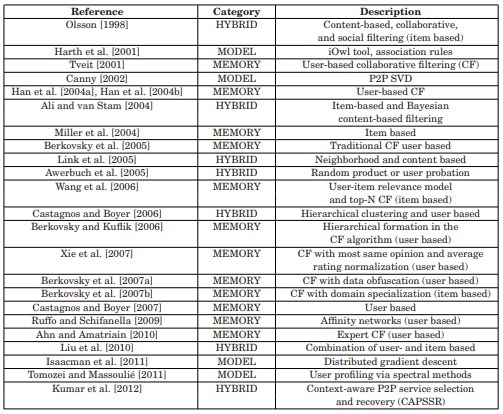
* a scoate in evidenta multitudinea de variante hibride cu care se poate experimenta
* a compara performanta diferitelor abordari
* a compara performanta diferitelor surse pentru seturi de date
* a compara caracteristicile non-functionale ale sistemelor *model-based* versus *memory-based*

**Studii de caz asupra aplicativitatii**

Articolele din aceasta categorie realizeaza taxonomii ale sistemelor de recomandare in functie de aria in care sunt utilizate.

## Sectiunea 3 – Distributed implementations

Sectiunea prezinta implementarile existente ale sistemelor de recomandare de tip *collaborative filtering* care folosesc tehnici specifice sistemelor distribuite si evidentiaza evolutia abordarilor si principalele puncte de referinta. Tabelul de mai jos rezuma implementarile prezentate in aceasta sectiune.



**Sisteme de recomandare collaborative filtering de tip memory–based**

Initial, sistemele de recomandare, atat cele *memory-based*, cat si cele *model-based* au fost implementate traditional. Prima implementare distribuita este Han et al. 2004a; Han et al. 2004b, folosind o arhitectura P2P pentru un sistem *memory-based*, *user-based* de *collaborative filtering.* Sistemul foloseste un *hash table* distribuit. Pentru a genera o recomandare, se creeaza un set de date local alcatuit si noduri care au cel putin un obiect in comun si se stabileste recomandarea pe baza preferintelor membrilor din acest set de date.

Implementarile ulterioare sunt mai sofisticate. Wang et al. [2006] calculeaza similaritatea intre obiecte folosit *log-based user profiles*. Reteaua are de asemenea o arhitectura P2P, in care fiecare nod stocheaza local tabele de similaritate intre obiecte.

Ruffo and Schifanella [2009] descriu o implmentare P2P in care nu se utilizeaza profilurile utilizatorilor, ci retele de afinitate intre utilizatori. Retelele de afinitate sunt generate potrivit fisierelor pe care le au in comun utilizatorii. Ahn and Amatriain [2010] propun un sistem de *expert collaborative filtering* distribuit in care evaluarile utilizatorilor sunt inlocuite de evaluari alcatuite de experti in domeniu, preluate de la Metacritic.

**Sisteme de recomandare collaborative filtering de tip model–based**

Primul sistem de recomandare distribuit *model-based* este introdus de Canny [2002]. Este un sistem cu model SVD care foloseste o arhitectura P2P, axata pe securitatea datelor utilizatorilor. Recomandarile sunt generate de un model agregat care integreaza preferintele utilizatorului

Alte abordari, de tip *user profiling*, utilizeaza algoritmi de *dimensionality reduction* pentru a crea un vector care codifica profilul utilizatorului. Pe baza aceastora o valoare de similaritate este calculata intre nodul curent si celelalte noduri din retea la care este conectat.

**Sisteme de recomandare collaborative filtering hibride**

Ali and van Stam [2004] propun o arhitectura *client-server* in care corelatiile intre obiecte sunt calculate pe partea serverului. Aceastea sunt folosite la nivelul clientului pentru a genera recomandari fara evaluarea niciunui model.

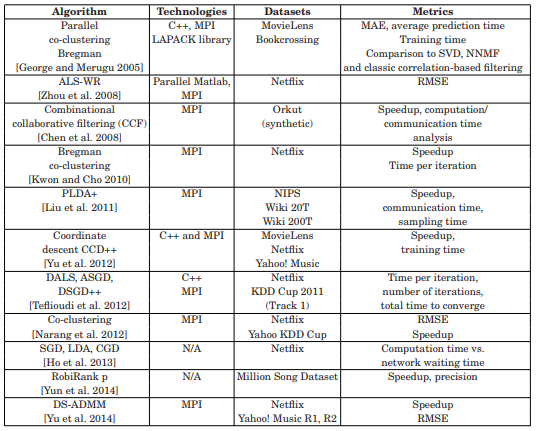
Castagnos and Boyer [2006] prezinta o abordare in care se combina *user-base collaborative filtering* si un algoritm ierarhic de clustering. Componenta server creeaza comunitati virtuale. Recomandarile sunt facute determinand pe partea clientului grupul de care este cel mai aproape.

## Sectiunea 4 – Parallel implementations

Sectiunea prezinta implementarile existente ale sistemelor de recomandare de tip *collaborative filtering* care folosesc tehnici specifice sistemelor paralele si evidentiaza evolutia abordarilor si principalele puncte de referinta

**Implementari cu memorie distribuita**

Tabelul de mai jos listeaza implementarile paralele care folosesc un model de memorie distribuita. Toate folosesc o abordare *model-based*, niciuna nu opteaza pentru o abordare *memory-based*. Ca algoritm, cel mai adesea se utilizeaza clustering in aceste implementari.

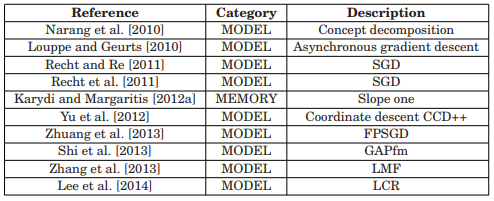


[Banerjee et al. 2004] propun o solutie bazata pe algoritmul Bregman de co-clustering. Algoritmul este implementat paralel de George and Merugu [2005] si Kwon and Cho [2010]. O abordare asemanatoare apare si in Narang et al. [2012].

Un algoritm distribuit LDA (Latent Dirichlet Allocation) este descris in Liu et al. [2011] si este implementat folosing MPI. *Pipeline processing*, *word bundling*, si *priority-based scheduling* se numara printre tehnicile folosite pentru a reduce timpul necesar pentru comunicare si numarul de mesaje schimbate.

**Implementari cu memorie comuna**

Tabelul de mai jos listeaza implementarile paralele care folosesc un model de memorie comuna. Din nou se observa preferinta pentru sistemele *model-based*, o singura implementare folosind o abordare *memory-based*.

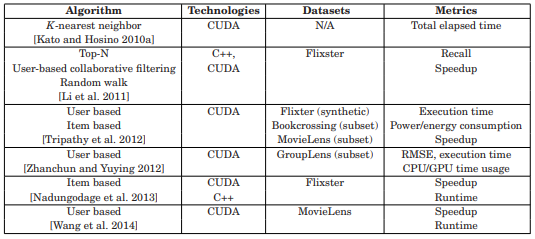


Narang et al. [2010] prezinta o implementare paralela bazata pe tehnica descompunerii conceptelor pentru aproximari matriceale. Tehnica realizeaza clustering folosind algoritmul *k-means* si apoi rezolva o problema de tip *least squares*. Implementarea are 4 niveluri *multithreaded*, urmate de nivelul de predictie. Implementarea foloseste *Posix threads* si este evaluata pe setul de date Nextlix. Louppe and Geurts [2010] implementeaza *parallel gradient descent* intr-un context cu memorie comuna.

**Implementari cu accelerare GPU**

Dezvoltarea accelerata a dispozitivelor GPU a creat o oportunitate pentru dezvoltarea de noi solutii de calcul paralel. Implementari din diverse domenii din *Computer Science* beneficiaza de o performanta mult imbunatatita in momentul in care se utilizeaza accelerare pe baza de GPU. Tabelul de mai jos prezinta implementarile discutate in lucrarea analizata.

Se observa ca toate implementarile folosesc CUDA ca API pentru platforma de calcul paralel. De asemenea, in multe cazuri, algoritmul trebuie adaptat pentru a folosi eficient paradigma *data-driven* care reprezinta caracteristica definitorie a dispozitivelor GPU. Kato and Hosino [2010a] abordeaza generarea de recomandari folosind algoritmul de clasificare *k-nearest neighbor*, insa problema calcularii distantelor este subdivizata in calcule la nivel de zone. Fiecare GPU proceseaza o zona. Fiecare zona este divizata de asemenea in seturi de randuri, fiecare fiind alocat unui set de *thread­*-uriastfel incat un *thread* proceseaza un rand dintr-o zona.



Din categoria sistemelor de recomandare *model-based*, una din primele implementari paralele pe GPU de SVD (Single Value Decomposition) este descrisa de Lahabar and Narayanan [2009]. Implementarea pastreaza datele necesare in memoria interna a GPU pentru a evita transferul de date intre CPU si GPU.

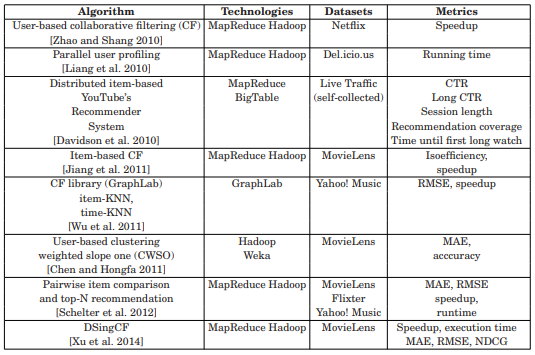
Se observa din nou, ca in cazul tuturor implementarilor paralele, o preferinta pentru abordarile *model-based*. Algoritmii cel mai adesea implementati sunt SGD (Stochastic gradient descent), SVD (Single value decomposition) si co-clustering. Comunicarea intre noduri in cazul sistemelor cu memorie distribuita este prin MPI pentru toate implementarile. Propunerile care folosesc memorie comuna sunt intr-un numar mai mic sim ai recente.

## Sectiunea 5 – Platform-based recommendations

Sectiunea prezinta implementarile de sisteme de *collaborative-filtering* realizate cu ajutorul framworkurilor specializate pentru calcul paralel si aplicatii de tip *big data.*

Tabelul urmator sintetizeaza solutiile de tip *memory-based.* Cu o exceptie, toate recurg la un model *MapReduce* implementat in *Hadoop*. Procesul include de regula o faza de *map* in care se genereaza o lista de recomandari pentru utilizator, si o faza de *reduce* in care se alege numarul necesar de recomandari.

Preferinta pentru *Hadoop* este prezenta si in cazul implementarilor *model-based,* desi algoritmul este diferit. Procesul este controlat de un nod *master* care atribuie dinamic date unor *workers*, avand grija in acelasi timp sa evite scrierile simulatane in aceeasi zona de memorie.



## Sectiunea 6 – Heterogenous implementations

Implementarile hibride inglobeaza solutiile care combina mai multe modele, tehnici si tehonologii. Majoritatea celor prezentate in studiul de caz sunt construite folosing MPI, OpenMP sau POSIX threads. In practica se observa insa ca rareori sistemele utilizeaza mai multe metode de paralelizare, un *use-case* mai des intalnit fiind combinarea modelelor *shared-memory* cu modele *distributed-memory*.

## Sectiunea 7 – Discussion and conclusions

Concluzia principala este ca sistemele clasice, fie *memory-based*, fie *model-based* sunt mai frecvente decat mai nou-aparutele sisteme hibride. Din motive evidente, nu exista implementari *memory-based* pe arhitecturi *distributed memory*. Insa chiar si in cazul arhitecturilor *shared-memory*, exista o singura implementare *memory-based*.

In ciuda diversitatii implementarilor existente, se concluzioneaza de asemenea ca este necesara continuarea cercetarii in aceasta directie, pentru a beneficia in mod eficient de avantajele pe care le ofera platformele distribuite si paralele. O directie care se sugereaza este dezvoltarea de sisteme hibride pe mai multe niveluri. Dezvoltarea de care s-au bucurat framework-urilor de paralelizare reprezinta un plus.

In materie de evaluare, chiar daca metricile sunt in general bine definite, studiile sunt realizate pe seturi de date diferite, ceea ce face corelarea rezultatelor foarte dificila.

# Surse bibliografice

Fiind un studiu de caz, lucrarea are un numar mult mai mare de surse bibliografice. Lucrarile referentiate sunt toate specifice domeniului sistemelor de recomandare, si complementar sistemelor distribuite si paralele. Majoritatea sunt lucrari relativ noi, publicate intre 2000 si 2015.