1 Rezumat

În această lucrare voi prezenta etapele proiectării și implementării unui sistem distribuit care poate procesa grafuri foarte mari. Modelul folosit pentru proiectare este cel propus in 2010 de Google În lucrarea *A System for Large-Scale Graph Processing* [1]. Necesitatea unui astfel de sistem apare în contextul actual odată cu nevoia de a procesa un volum foarte mare de date pentru aplicații din ce În ce mai uzuale. Foarte multe probleme practice implică modelarea prin grafuri și aplicarea de algoritmi pe grafurile rezultate, două exemple canonice sunt graful rețelei web și rețelele sociale unde grafurile pot ajunge la miliarde sau chiar trilioane [20] de noduri. Un exemplu de algoritm aplicat pe grafuri de dimensiuni mari este *Page rank* (eng.) proiectat tot de *Google* pentru calculul relevanței unei pagini web În vederea oferirii unor rezultate cât mai relevante returnate de motorul de căutare *Google* și a furnizării unei experiențe cât mai plăcute utilizatorului. Acest exemplu este deosebit de important deoarece motorul de căutare *Google* este o aplicație folosită zilnic de un număr foarte mare de utilizatori.

Modelul propus de Google se bazează pe procesarea de tip *Bulk synchronous parallel* (eng.) în care execuția unui algoritm este împărțită În mai mulți super pași(*superstep* (eng.)) care presupun 1)calcul, 2)comunicare și 3)sincronizare, într-un super pas de obicei mai multe mașini/procese/fire de execuție aplică o serie de calcule, își trimit mesaje și se sincronizează cu ajutorul unei bariere. Pentru a aplica BSP pe grafuri Pregel folosește modelul ,,*Gândește ca un nod*" în care pentru a ajunge la un anumit rezultat aplicăm o funcție în fiecare super pas pentru fiecare nod din graf, acestea devenind entități de sine stătătoare care pot comunica unele cu altele și pot ,,colabora" pentru obținerea unor rezultate, acest model presupune limitarea viziunii la nivel de nod din graf și este ideal pentru implementarea de algoritmi iterativi care nu ajung neapărat la soluția optimă, ci converg la aceasta într-un număr predefinit de super pași sau prin atingerea unui criteriu de convergență.

Cu ajutorul acestui model putem implementa orice algoritm pentru grafuri într-o manieră distribuită și cu ajutorul unui framework (eng.) putem ascunde detaliile de implementare utilizatorului final. Tot ce trebuie să facă utilizatorul este să încapsuleze într-o clasă descrierea algoritmului dorit și să apeleze funcțiile puse la dispoziție printr-un API (eng.), apelarea funcțiilor vor conduce la un întreg proces de execuție distribuită. În afară de execuția unui algoritm API-ul mai pune la dispoziție funcții care furnizează date despre starea sistemului, istoricul execuțiilor, metrici despre execuția joburilor, etc.

Modul de execuție al joburilor este *Batch processing (eng.)* similar cu *Apache Giraph* [4] în care pentru obținerea unor rezultate utilizatorul trebuie să execute unul sau mai multe joburi înlănțuite, pe un anumit set de date.

În afară de implementarea propriu zisă a sistemului am mai inclus și componente auxiliare specifice unui sistem distribuit care fac mult mai atractivă folosirea acestuia și oferă avantajul automatizării unor sarcini repetitive. Cele două componente sunt un site web pentru a face mult mai ușoară apelarea funcțiilor puse la dispoziție de API și o componentă de monitorizare care se ocupă cu gestionarea(stocarea și furnizarea) metricilor. Cu ajutorul acestui site web se pot încărca clase ce încapsulează un algoritm pentru folosirea lor ulterioară, se pot executa și monitoriza joburi și se pot obține date despre execuția joburilor, de exemplu metrici de timp. Cea mai importantă parte a interfeței cu utilizatorul este urmărirea în timp real a execuției unui job prin folosirea unei sub pagini în care sunt primite notificări în legătură cu execuția unui super pas, terminarea jobului și includerea unei adrese către o aplicație web care facilitează lucrul cu baza de date *NoSQL* MongoDB unde se pot vizualiza rezultatele parțiale. Tot la acest capitol am inclus și trimiterea de e-mail în legătură cu starea jobului, la fiecare schimbare a statusului execuției unui job utilizatorul este anunțat prin e-mail.

Am ales această aplicație deoarece pot evidenția un număr mare de cunoștințe dobândite, iar implementarea sistemului poate evidenția un număr mare de tehnologii și concepte specifice în ingineria programării.