Raport asupra practicii: 25.06-06.07.2018

Sarageaua Ana

Cuprins

Capitolul 1

Introducere

Implementarea algoritmului de sortare rapidă (quick sort).

Quicksort este un celebru algoritm de sortare, dezvoltat de C. A. R. Hoare și care, în medie, efectuează O(n logn) comparații. De obicei, în practică, quicksort este mai rapid decât ceilalți algoritmi de sortare de complexitate O(n logn) deoarece bucla sa interioară are implementări eficiente pe majoritatea arhitecturilor și, în plus, în majoritatea implementărilor practice se pot lua, la proiectare, decizii ce ajută la evitarea cazului când complexitatea algoritmului este de $O(n^2)$.

Quicksort efectuează sortarea bazându-se pe o strategie divide et impera. Astfel, el împarte lista de sortat în două subliste mai ușor de sortat. Pașii algoritmului sunt:

Se alege un element al listei, denumit pivot.

Se reordonează lista astfel încât toate elementele mai mici decât pivotul să fie plasate înaintea pivotului și toate elementele mai mari să fie după pivot. După această partiționare, pivotul se află în poziția sa finală.

Se sortează recursiv sublista de elemente mai mici decât pivotul și sublista de elemente mai mari decât pivotul.

O listă de dimensiune 0 sau 1 este considerată sortată.

Folosind metoda divide et impera problema iniţială va fi descompusă în subprobleme, astfel:

Pas1. Se rearanjează vectorul, determinându-se poziţia în care va fi mutat pivotul (m).

Pas2. Problema iniţială (sortarea vectorului iniţial) se descompune în două subprobleme prin descompunerea vectorului în doi subvectori: vectorului din stânga pivotului şi vectorul din dreapta pivotului, care vor fi sortaţi prin aceeaşi metodă. Aceşti subvectori, la rândul lor, vor fi şi ei rearanjaţi şi împărţiţi de pivot în doi subvectori etc.

Pas3. Procesul de descompunere în subprobleme ca continua până când, prin descompunerea vectorului în subvectori, se vor obţine vectori care conţin un singur element.

Subprogramele specifice algoritmului divide et impera vor avea următoarea semnificaţie:

În subprogramul divizeaza() se va rearanja vectorul şi se va determina poziţia pivotului xm, care va fi folosită pentru divizarea vectorului în doi subvectori :[xs, xm-1] şi [xm+1, xd].

Subprogramul combina() nu mai este necesar, deoarece combinarea soluţiilor se face prin rearanjarea elementelor în vector.

În subprogramul divizeaza() vectorul se parcurge de la ambele capete către poziţia în care trebuie mutat pivotul.

Se vor folosi doi indici: i – pentru parcurgerea vectorului de la începutul lui către poziţia pivotului (i se va incrementa) şi j – pentru parcurgerea vectorului de la sfârşitul lui către poziţia pivotului (j se va decrementa). Cei doi indici vor fi iniţializaţi cu capetele vectorului (i=s, respectiv j=d) şi se vor deplasa până când se întâlnesc, adică atât timp cât i¡j.

În momentul în care cei doi indici s-au întâlnit înseamnă că operaţiile de rearanjare a vectorului s-au terminat şi pivotul a fost adus în poziţia corespunzătoare lui în vectorul sortat. Această poziţie este i (sau j) şi va fi poziţia m de divizare a vectorului.

Voi prezenta în continuare exemple unde se vor utiliza două versiuni pentru subprogramul divizează().

Versiunea 1.

3 4 1 5 2

Se folosesc variabilele logice: pi, pentru parcurgerea cu indicele i, şi pj, pentru parcurgerea cu indicele j. Ele au valoarea: 1 – se parcurge vectorul cu acel indice, şi 0 – nu se 10 parcurge vectorul cu acel indice; cele două valori sunt complementare.

Cei doi indici i şi j sunt iniţializaţi cu extremităţile vectorului(i=1, j=5) şi parcurgerea începe cu indicele j (pi=0, pj=1)

Se compară elementul din poziţia i(4) cu elementul din poziţia j(3). Deoarece 4 este mai mare decât 3, cele două valori se interschimbă, şi se schimbă şi modul de parcurgere (pi=0;pj=1 – avansează inicele j).

Se compară elementul din poziţia i(3) cu elementul din poziţia j(5). Deoarece 3 este mai mic decât 5, cele două valori nu se interschimbă, şi se schimbă modul de parcurgere (pi=0;pj=1 – avansează inicele j).

Se compară elementul din poziţia i(3) cu elementul din poziţia j(1). Deoarece 3 este mai mare decât 1, cele două valori se interschimbă, şi se schimbă modul de parcurgere (pi=0;pj=1 – avansează inicele i).Cei doi indici fiind egali, algoritmul se termină.

Versiunea 2.

3 4 1 5 2

Iniţial, cei doi indici i şi j sunt iniţializaţi cu extremităţile vectorului(i=1, j=5) şi pivotul are valoarea 3.

Elementul din poziţia i(3) nu este mai mic decât pivotul; indicele i nu avansează (i=1). Elementul din poziţia j(2) nu este mai mare decât pivotul; indicele j nu avansează(j=5). Valorile din poziţiile i şi j se interschimbă.

Elementul din poziţia i(2) este mai mic decât pivotul; indicele i avansează până la primul element mai mare decât pivotul (i=2). Elementul din poziţia j(3) nu este mai mare decât pivotul; indicele j nu avansează(j=5). Valorile din poziţiile i şi j se interschimbă.

Elementul din poziţia i(3) nu este mai mic decât pivotul; indicele i nu avansează (i=2). Elementul din poziţia j(4) este mai mare decât pivotul; indicele j avansează până la primul element mai mic decât pivotul (j=3). Valorile din poziţiile i şi j se interschimbă.

Elementul din poziţia i(1) este mai mic decât pivotul; indicele i avansează până la primul element mai mare decât pivotul (i=4). Elementul din poziţia j(3) nu este mai mare decât pivotul; indicele j nu avansează(j=3), algoritmul se termină.

Raportul asupra practicii efectuate zilnic intre datele 25.06-06.07.2018.

Capitolul 2

Activități planificate

1. Luni, 25.06.2018
   Aducerea la cunoștință a obiectivelor și cerințelor practicii de producție
2. Marți, 26.06.2018
   Configurarea sistemelor software pe calculatoare.
3. Miercuri, 27.06.2018
   Studierea modului de lucru cu Git. Interfețe grafice de lucru cu Git (SmartGit).
4. Joi, 28.06.2018
   Studierea și practicarea LaTeX
5. Vineri, 29.06.2018
   Inițierea unei lucrări (descrierea unui algoritm, a unei teme agreate cu prof. coordonator)
6. Luni, 02.07.2018
   Lucrul asupra lucrării
7. Marți, 03.07.2018
   Lucrul asupra lucrării
8. Miercuri, 04.07.2018
   Prezentarea lucrărllor
9. Joi, 05.07.2018
   Prezentarea lucrărilor
10. Vineri, 06.07.2018
    Notarea finală a activității

Capitolul 3

25.06.2018

Am desfăţurat următoarele activităţi:

- Am identificat sursele pentru MikTeX, Git, SmartGit și BitBucket.
  - Am identificat sursele pentru MikTeX, Git, SmartGit și BitBucket.
  - Am instalat, configurat pe calculatorul de lucru aplicațiile necesare:
    * MikTeX
    * SmartGit
    * Bitbucket
  - Am instalat, configurat pe calculatorul de lucru aplicațiile necesare:
    * MikTeX
    * SmartGit
    * Bitbucket

Capitolul 4

26.06.2018

Studierea obiectivelor și cerințelor față de practica de producție. Clarificarea situațiilor incerte.

Capitolul 5

27.06.2018

Am studiat modul de lucru cu Git și interfața grafică de lucru cu Git (SmartGit).

Git este un sistem revision control care rulează pe majoritatea platformelor, inclusiv Linux, POSIX, Windows și OS X. Ca și Mercurial, Git este un sistem distribuit și nu întreține o bază de date comună. Este folosit în echipe de dezvoltare mari, în care membrii echipei acționează oarecum independent și sunt răspândiți pe o arie geografică mare.

Git este dezvoltat și întreținut de Junio Hamano, fiind publicat sub licență GPL și este considerat software liber.

Dezvoltarea Git a început după ce mai mulți developeri ai nucleului Linux au ales să renunțe la sistemul de revision control proprietar BitKeeper. Posibilitatea de a utiliza BitKeeper gratuit a fost retrasă după ce titularul drepturilor de autor a afirmat că Andrew Tridgell a încălcat licența BitKeeper prin acțiunile sale de inginerie inversă. La conferința Linux.Conf.Au 2005, Tridgell a demonstrat în timpul discursului său că procesul de inginerie inversă pe care l-a folosit a fost pur și simplu o sesiune telnet pe portul corespunzător al serverului BitKeeper și rularea comenzii help pe server.

Controversa a dus la o renunțarea rapidă la sistemul BitKeeper care a fost înlocuit cu un nou sistem intitulat Git construit special pentru scopul de revision control în cadrul proiectului Linux kernel. Dezvoltarea noului sistem a fost începută de Linus Torvalds în 3 aprilie 2005 pentru a fi anunțat câteva zile mai târziu (aprilie 6) pe lista de email a proiectului Linux kernel[28]. O zi mai târziu, noul sistem a început să fie folosit pentru dezvoltarea actuală de cod pentru proiectul Git. Primele operații merge a avut loc pe data de 18 aprilie. În data de 16 iunie, versiunea 2.6.12 Linux kernel a fost pusă în Git care continuă și în ziua de azi să fie sistemul revision control folosit de proiectul Linux kernel.

Tot în această perioadă, și tot cu scopul de a înlocui BitKeeper, a fost creat sistemul Mercurial.

Capitolul 6

28.06.2018

Am studiat și am practicat Latex.

LaTeX este un sistem de preparare a documentului, care permite tipărirea în format electronic cu ajutorul limbajului de programare TeX.

LaTeX a fost creat de Leslie Lamport în 1984 la SRI International și în timp a devenit principala metodă pentru programarea în TeX. Datorită capacităților de a programa în amănunt orice aspect care ține de publicarea unui material (articol, carte, tratat, broșură), LaTeX este folosit în general în mediu academic de către matematicieni, ingineri, etc, dar și în mediul comercial, datorită costurilor reduse de utilizare (LaTex și TeX sunt gratuite; TeX este eliberat de către creatorul său, Donald Knuth, în domeniu public). LaTeX permite programarea aspectelor necesare în desktop publishing, inclusiv tabele, figuri și imagini, referințe încrucișate, bibliografie și note bibliografice.

Din punct de vedere al limbajului de programare, LaTeX este un limbaj de programare de nivel-înalt, util în a accede la toate resursele limbajului TeX. Deoarece TeX este un limbaj de programare de nivel scăzut s-a dovedit a fi destul de dificil de utilizat de către utilizatorii comuni, motiv pentru care LaTeX a fost construit special pentru a permite oricărui utilizator să beneficieze de puterea limbajului TeX.

Versiunea curentă este (LaTeX2e). LaTeX, ca și TeX, este un program liber.

Capitolul 7

29.06.2018

Am inițiat o lucrare scrisă în Latex.

Capitolul 8

30.06.2018

Am continuat lucrul asupra temei alese.
Implementarea algoritmului de sortare rapidă (quick sort).

Capitolul 9

02.07.2018

Am continuat lucrul asupra temei și am terminat .

Capitolul 10

03.07.2018

Am continuat lucrul asupra temei și am terminat . Prezentarea proiectului.

Capitolul 11

04.07.2018

Prezentarea proiectului.

Capitolul 12

05.07.2018

Prezentarea proiectului.

Capitolul 13

06.07.2018

Notarea finală a activității. Practica 2018.

Capitolul 14

Concluzii

Am invățat să lucrez cu Latex ,Git și BitBucket. Aici pot fi prezentate succint lucrurile învățate (câte ceva despre git, latex, bitbucket, etc - se pot adăuga secțiuni pentru fiecare subiect).

Și acum să cităm unele dintre referințele noastre [1] și [2]. La fel putem să citățm și alte cărți și surse online cum ar fi [3, 4]. Alte exemple sunt în mostra / modelul unei lucrări de licență de pe site-ul facultății.

Bibliografie

[1] J. Doe, *The Book without Title*. Dummy Publisher, 2100.

[2] G. H.J., *A Simplified Introduction to LaTeX*. 2004.

[3] P. C.E, *An Introduction to Python and LaTeX (draft)*. 2010.

[4] C. Strom, *3D game programming for kids: create interactive worlds with JavaScript*. Pragmatic Bookshelf, 2013.