РЕАЛИЗАЦИЯ РЕШЕНИЙ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ В КОМПЬЮТЕРНЫХ МАТЕМАТИЧЕСКИХ СИСТЕМАХ (MATHCAD)

Аскерова Тамилла Ахмед кызы, Канбарова Сахиба Ибрахим кызы

Резюме

В статье рассматриваются компьютерные математические системы. Основной целью научного направления компьютерной математики является достижение решения всех видов математических вычислений с высокой точностью в автоматическом и диалоговом режиме с использованием теоретических, методических, аппаратных и программных средств. Система линейных уравнений решается в Mathcad методами Крамера, Гаусса, матричными методами. Одним из основных преимуществ системы ее применение, развитие и интеграция в образовании.

IMPLEMENTATION OF SOLUTIONS TO SYSTEMS OF LINEAR EQUATIONS IN COMPUTER MATHEMATICAL SYSTEMS (MATHCAD)

Asgerova Tamilla Ahmed, Qenberova Sahibe Ibrahim

Summary

The article deals with computer mathematical systems. The main goal of the scientific direction of computer mathematics is to achieve the solution of all types of mathematical calculations with high accuracy in automatic and interactive mode using theoretical, methodological, hardware and software tools. The system of linear equations is solved in Mathcad by Cramer, Gauss, matrix methods. One of the main advantages of the system was its application and development in education.

Rəyçi: dos. Verdiyeva G.Z.
Informatika kafedrasının 23 may 2022-ci il tarixli iclasın 05 saylı protokolu
Daxil olma tarixi 11 aprel 2022-ci il

UOT:372.0:002

ALQORİTMLƏRİN ANALİZİNİN ƏSASLARI

Babayeva Şəkər Maarif qızı
Gəncə Dövlət Universiteti
shakarbabayeva@mail.ru

Xülasə: *Əsl kompüter mütəxəssisi hesablama texnikasının müxtəlif sahələrinə aid olan əsas alqoritmlərin standart yığımı haqqında təsəvvürü olmalıdır və o, yeni alqoritmləri tərtib etməyi və onların effektivliyini analiz etməyi bacarmalıdır. Eyni məsələnin həlli üçün müxtəlif alqoritmlər qurmaq olur. Alqoritmlərin analizi bizə alqoritmin seçilməsi üçün alətdir.*

Cari məqalə alqoritmlərin analizi prosesinin əsaslarına həsr edilmişdir. Alqoritmlərin analizi istifadə olunan kompüter və proqramlaşdırma dilndən asılı deyil. Alqoritm effektivliyinin analizinin iki növü mövcuddur : vaxt effektivliyi və tutum effektivliyi. Hal hazırda əsas diqqət alqoritmin vaxt effektivliyinə yetirilir. Alqoritmlərin effektivliyi n ölçüdə giriş verilənlərin

emalında yerinə yetirəcəyi əsas (baza) əməliyyatların sayı ilə qiymətləndirilir. Alqoritmlərin analizinin əsas məqsədi alqoritmin yerinə yetirililməsi vaxtının giriş verilənlərin sonzuzluğa yaxınlaşanda artma sürəti tərtibinin təyin edilməsidir.

Açar sözlər: alqoritmlərin analizi, alqoritmin effektivliyi, vaxt effektivliyi, tutum effektivliyi**,** giriş verilənlərin ölçüsü, baza əməliyyatlar, artma sürəti**.**

Ключевые слова: анализ алгоритмов, эффективность алгоритмов, времнная эффективность, пространственная эффективность, размерность входных данных, базовая операция, порядок роста.

Key words: analysis of algorithms, algorithm efficiency, time efficiency, space efficiency, basic operation**,** order of growth**.**

Universitet təliminin əsas məqsədi qoyulmuş məsələlərin sərbəst həlli vərdişlərinin tələbələrdə yaradılmasıdır. Ona görə informatika proqramı çərçivəsində keçilən fənlərin içərisində "Alqoritmlərin hazırlanması və analizi" fənni bu məqsədə ən yaxşı cavab verir, çünki o, tələbələrdə məsələ həllinin spesifik vərdişlərini inkişaf etdirir.

Alqoritmləri nə üçün oyrənmək lazımdır? Əsl kompüter mütəxəssisi üçün bunun iki səbəbi var: praktiki və nəzəri. Praktiki nöqteyi nəzərdən onun hesablama texnikasının müxtəlif sahələrinə aid olan əsas alqoritmlərin standart yığımı haqqında təsəvvürü olmalıdır. Bundan başqa o, yeni alqoritmləri tərtib etməyi və onların effektivliyini analiz etməyi bacarmalıdır. Nəzəri nöqteyi nəzərdən alqoritmika (algorithmics) adlanan alqoritmlərin öyrənilməsi prosesi informatikanın (computer scince) əsası hesab edilir. Devid Harel (David Harel) öz "Algorithmics: the Spirit of Computing" ("Alqoritmika: hesablamaların ruhu") kitabında yazır: " Alqoritmika informatikanın bir bölməsi yox, onun əsasıdır və ürəklə demək olar ki, o, müasir elm, texnika və biznesə əhəmiyyətli dərəcədə təsir edib [[6], s.6]."

Alqoritmlərin öyrənilməsi olduqca vacibdir, çünki alqoritmlərsiz heç bir kompüter proqramını yazmaq olmaz. Kompüter proqramlarını tərtib etmək insanın peşə fəaliyyatının bütün sferalarında (və şəxsi həyatında da) əhəmiyyətli olduğu üçün, alqoritmlərin öyrənilməsi geniş əhatədə insanlar üçün çox vacib məsələyə çevrilir.

Alqoritmlərin öyrənilməsinin bir səbəbi də ondan ibarətdir ki, bu proses tələbələrin analitik düşünməsini inkişaf etdirir.

İnformatika və alqoritmika tarixi sahəsi üzrə dünyada tanınan, görkəmli alim Donald Knut yazır: " İnformatika sahəsi üzrə yaxşı təlim almış mütəxəssis alqoritmlərlə işləməyi – onları tərtib etməyi, dəyişdirməyi, başa düşməyi və analiz etməyi yaxşı bacarmalıdır. Bu biliklər tək kompüter proqramlarını yaxşı yazmağa imkan verəcək , həm də kimya, linqvistika, musiqi və s. kimi digər elmlərin öyrənilməsində qiymətsiz kömək göstərəcək, universal düşüncə aparatının əsası olacaq. Bunun səbəbini belə izah etmək olar: deyirlər ki, insan heç nəyi onu kiməsə başa salmayana kimi başa düşmür. Mən bunu belə deyərdim: insan predmeti onu kompüterə öyrədənə kimi, yəni bunu alqoritm şəklində təsvir edənə kimi, dərindən başa düşmür. Nəyinsə alqoritm şəklində formallaşdırması cəhdi onun daha dərin başa düşülməsinə gətirir, nəinki, onun ənənəvi üsulla dərk edilməsi [[7], s.9]."

Amerika ensiklopediya lüğətində (American Heritage Dictionary) "analiz (analysis)" sözü "maddi və mənəvi bir bütövün öyrənmə məqsədi ilə tərkib hissələrə ayrılması" kimi təyin olunur.

Alqoritmlərin analizi dedikdə alqoritmlərin effektivliyinin araşdırılması prosesi başa düşülür. Alqoritmlərin effektivliyi (efficiency) iki parametrlə - alqoritmin yerinə yetirilməsi vaxtı və tələb olunan yaddaş tutumu ilə qiymətləndirilir.

Bir çox məsələlərin həllində olduğu kimi iki ədədin ən böyük ortaq bölənin (ƏBOB-un) tapılması üçün də bir neçə alqoritm mövcuddur. Iki mənfi olmayan ədədlərin (*m* və *n*, *m* və *n* eyni zamanda sıfra bərabər olmaması şərt ilə) ən böyük ortaq ədədinin tapılması funksiyasını *gcd(m,n)* ilə işarə edək ("*greatest common divisor"* ingilis sözlərindən alınıb). Tərifinə görə bu funksiya *m* və n-in qalıqsız bölünənlərindən ən böyük tam ədədi (yəni ümumi bölünənlərinin ən böyüyünü) tapmalıdır. Əksəriyyət məsələlərin həlli zamanı olduğu kimi ƏBOB-un hesablanmasının da bir neçə alqoritmləri mövcuddur. Qədim yunan alimi Evklid bu məsələnin həlli alqoritmini şərh etmişdi. Evklid alqoritmi aşağıdakı bərabərliklərin rekurent hesablanmasına əsaslanır:

gcd(m,n) = gcd(m- n, n), əgər m>n (çıxma üsulu),

gcd(m,n) = gcd(n, m mod n), əgər m>n (bölmə üsulu).

Çixma üsulunda iki əddədən böyüyünü bu ədədlərin fərqi ilə əvəz edilir. Bu proses sıfırdan fərqli iki eyni ədəd alınana qədər davam etdirilir. Sonda alınmış ədəd verilmiş ədədlərin ən böyük ortaq böləni olacaq. Məsələn,

gcd(140,96) = gcd((96,44) = gcd(44,12) = gcd(32,12) = gcd(20,12) = gcd(12,8) =

= gcd(8,4) = gcd(4,4) = 4

Bölmə üsulunda (*m mod n)* ifadəsi *m-in n*-ə bölünməsi nəticəsində alınan qalıqdır. Alqoritmin icrası (*m mod n)* ifadəsi 0-a bərabər olanda tamamlanır. *gcd(m,0)=m* oduğuna görə *m*-in sonuncu qiyməti də ilkin ədədlərin ən böyük ortaq böləni olacaq (çünki həm *m* və həm *0 m*-ə bölünür).

gcd(140,96) = gcd((96,44) = gcd(44,12) = gcd(12,8) = gcd(8,4) = gcd(4,0) = 4

Misallardan gününur ki, Evklid alqoritminin bölmə üsulu daha tez icra olunur, yəni daha effektivdir.

Bu məsələnin diğər iki həll üsullarına baxaq. Bunlardan birincisi *m* və *n*-in qalıqsız bölünən ən böyük tam ədədin seçilməsinə əsaslanır. Aşkardır ki, belə ümumi bölünən iki ədədin ən kiçiyindən böyük ola bilməz. Onu *t=min{m,n} kimi* işarə etmək olar. Ona görə alqoritmin icrasını hər iki *m* və *n* ədədlərin *t*-ya qalıqsız bölünməsinin yoxlanılmasından başlamaq olar. Əgər bu doğrudur, onda *t* ədədi nəticə olacaq, əks halda *t*-nın qiymətini bir vahid azaldıb, yoxlamanı təkrar aparmaq lazımdır. Məsələn, yuxarıda baxdığımız (140,96) ədədlər cütü üçün alqoritmin icrası qalıqsız bölünmənin yoxlanması 96 ədədi üçün başlanılır, sonra – 95 üçün, daha sonra – 94 üçün və s., və *t*-in qiyməti 4 olanda alqoritmin işi tamamlanır.

Qeyd etmək lazımdir ki, Evklid alqoritmindən fərqli olaraq, əgər ilkin verilənlərdən heç olmasa biri sıfra bərabər olsa, bu alqoritm düzgün işləməyəcək. Bu misal alqoritmin ilkin parametrlərinin mümkün qiymətlər diapazonunun aşkar edilməsinin vacibliyini göstərir.

Əbob-un tapılmasının üçüncü üsulu opta məktəb kursundan tanışdı.

$140 = 2 \cdot 2 \cdot 5 \cdot 7$; $96 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3$; $\gcd(140,96) = 2 \cdot 2 = 4$.

Son üsullara aid göstərilən misallardan görünür ki, onlar Evklid alqoritmindən daha ləng işləyir. Məktəb üsulunun aşağı effektivliyini nəzərə almasaq da, bu alqoritmin psevdokodunda verilmiş ədədlərin sadə vuruqlara ayrılması tələb olunur və sadə ədədlərin siyahısı lazım olur. Bu da verilmiş ədəddən böyük olmayan sadə ədədlərin ardıcıllığını yaradılması alqoritmin

yazılmasını tələb edir. Bu alqoritm qədim Yunanıstanda yaradılıb (təqribən e.ə 200) və Eratosfen ələyi (sieve of Eratosthenes (təkcə 1-ə və özünə qalıqsız bölünən ədədlərin axtarışı üsulu) adlanır.

Əvvəlcə 2-dən n-ə qədər tam ədədlər ardıcıllığın siyahısını tərib edək. Məsələn, n=20 – dən çox olmayan sadə ədədləri seçmək üçün 2-dən 20-yə qədər ədədlər ardıcıllığının siyahısı tərtib edək:

2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Bu siyahıdan sadə ədədləri seçəcəyik. Alqoritmin birinci keçidində 2-ni saxlayıb, 2-yə bölünən bütün ədədləri (yəni 2-nin misillərini) silirik.

2 3 5 7 9 11 13 15 17 19

Sonra siyahıdan növbəti elementi (yəni 3-ü) seçirik və onun misillərini siyahıdan silirik.

2 3 5 7 11 13 17 19

Növbəti 4 ədədi 2-nin misli olduğuna görə birinci keçiddə silinib və onun üçün siyahıda keçid aparmağa eytiyac olmur (eyni olaraq bu alqoritmdə əvvəlki keçidlərdə silinmiş növbəti ədədlər üçün keçid aparmaq lazım olmur).

Siyahının növbəti elementi 5 ədədir, sonra 7 ədədidir və s., ancaq onların misilləri artıq silinib və yeni keçidlər aparmağa eytiyac olmur. Alqoritmin icrası siyahıda silinməsi mümkün olan ədədlərin qalana kimi davam edir. Alqoritmin icrasından sonra siyahıda qalan ədədər sadə ədədlərdir. Sual yaranır: hansı ən böyük ədəd üçün siyahıda hələ onun misilləri silınməmiş qalır? Suala cavab verməkdən əvvəl qeyd edək ki, əgər p – cari keçiddə misilləri silinən ədəddir, onda əvvəlki keciddə silinən onun müsilləri –

$2 \cdot p, 3 \cdot p, ..., (p-1) \cdot p$ olmalıdır və silinməni $p \cdot p$ ədədindən başlamaq lazımdır. Belə etdikdə eyni ədədin bir dəfədən çox silinməsindən qaçmaq olar. Aşkardır ki, $p \cdot p \leq n$. Buna görə p ədədi tam $\sqrt{n}$ ədədindən böyük olmamalıdır ($\sqrt{n}$ tam olmadıqda tam hissəsi götürülür). Alqoritmdə dövrün təkrarlanması şərti kimi alternativ variantda $p \cdot p \leq n$ bərabərsizliyi istifadə etmək olar.

Indi Eratosfen ələyi alqoritmini məktəbdə keçilən Əbobun axtarılması üsulu ilə birləşdirmək olar. Bu alqoritmdə giriş parametrlərin biri ya hər ikisinin 1-ə bərabər olmasını xüsusi hal kimi kənarlaşdırmaq lazımdır (1 ədədi sadə ədəd olmadığına görə).

Beləliklə, ən böyük ortaq bölənin axtarışı məsələsinin həlli üçün bir neçə müxtəlif alqoritm göstərdik. Eyni məsələnin həlli üçün müxtəlif alqoritmlər qurmaq olur. Alqoritmlərin analizi bizə alqoritmin seçilməsi üçün alətdir. Hər bir alqoritmin işinin effektivliyi müxtəlif xarakteristikalar ilə şərh olunur. Eyni məsələnin iki müxtəlif alqoritmini müqayisə etmək üçün alqoritmlərin effektivliyinin müxtəlif xarakteristikaları təyin edilmişdir. Heç vaxt müxtəlif məsələlərin alqoritmləri müqayisə olunmur, məsələn, massivlərim vurulması və massivlərin nizamlanması alqoritmləri müqayisə olunmur, ancaq massivlərin nizamlanmasının müxtəlif alqoritmləri müqayisə olunur. Alqoritmin effektivliyini analiz etməkdən əvvəl isbat etmək lazımdır ki, bu alqoritm məsələni düzgün həll edir. Əks halda alqoritmin effektivliyinin analizinin mənası olmur.

Əgər alqoritm məsələnin mümkün olan ixtiyari verilənləri üçün məsələnin tələbinə uyğun nəticə verirsə, onda bu alqoritm **düzgün** (correct) hesab edilir. Bu halda deyirlər ki, alqoritm verilən məsələni həll edir (solves). Düzgün olmayan alqoritm (bəzi verilənlər üçün) dayanmaya bilər və ya səhv nəticə verə bilər.

Alqoritmlərin analizi istifadə olunan kompüter və proqramlaşdırma dilndən asılı deyil. Biz alqoritmi psevdokodda təsvir edəcəyik. ***Psevdokod*** (pseudocode) təbii dillərin (adətən, ingilis) və proqramlaşdırma dilində istifadə olunan konstruksiyaların qarışığından ibarətdir. Alqoritmin psevdokodda təsviri təbii dilində təsvirindən daha dəqiq və yığcam olur. Mütəxəssislər indiyə qədər psevdokodun heç bir formasını standart kimi qəbul etməmişlər. Müxtəlif kitablarda psevdokodun müxtəlif "dialektləri" ilə rastlaşmaq olur. Hansısa proqramlaşdırma dilini bilən hər kəs bu dialektlərin istifadəsini başa düşə bilər.

Psevdokod – proqramlaşdırma dilinin açar sözlərini istifadə edən və onların xüsusi sintaksisinə riayət etməyən alqoritmin təsviretmə dilidir.

Alqoritm effektivliyinin analizinin iki növü mövcuddur : vaxt effektivliyi və tutum effektivliyi. ***Vaxt effektivliyi (time efficiency)*** alqoritm işinin sürətinin indikatorudur. ***Tutum (fəza) effektivliyi (space efficiency)*** alqoritmin işinə nə qədər əlavə əməli yaddaşı lazım olduğunu göstərir. EHM-lərin inkişafının ilk dövründə iki əsas göstəricilər çox vacib idi: mərkəzi prosessorun sürəti və əməli yaddaşın tutumu. Indiki dövrdə 60 ildən çox baş vermiş fasiləsiz sürətli texniki tərəqqi nəticəsində bu göstəricilər qat-qat artmışdır. Indi alqoritmin işinə əlavə lazım olan əməli yaddaş tutumuna tələbat azalmışdır. Ona görə hal hazırda əsas diqqət alqoritmin vaxt effektivliyinə yetirilir.

Bir çox alqoritmlərin yerinə yetirilməsi vaxtı giriş verilənlərin ölçülərindən asılıdır (verilənlərin sayı nə qədər çoxdur, alqoritmin işinə o qədər çox vaxt sərf edilir). Məsələn, böyük massivlərin çeşidlənməsi, böyük massivlərin vurulması alqoritmləri kifayət qədər vaxt aparır. Buna görə alqoritmlərin effektivliyinə müəyyən bir giriş verilənlərin ölçüsü ilə bağlı olan parametrin (n) funksiyasına kimi baxmaq olar. Məsələn, siyahıların emalı ilə bağlı olan məsələlərdə (ən kiçik və ya ən böyük elementin siyahıda axtarşı, siyahının nizamlanması və s.) belə parametr kimi siyahının ölçüsü götürülə bilər. Çoxhədlinin $p(x) = a_n x^n + \bullet \bullet \bullet + a_0$ qiymətinin hesablanması məsələsi üçün belə parametr kimi çoxhədlinin dərəcəsi n və ya çoxhədlinin dərəcəsindən 1 vahid çox olan çoxhədlinin əmsallarının sayını götürmək olar.

Alqoritm analizində ilkin verilənlərin rolu çox böyükdür. İlkin verilənlərdən alqoritmin analizini öyrənmək üçün aşağıdakı misala baxaq. Tutaq ki, siyahıda verilən n elementdən maksimumunu tapaq.

```
max = list[1];
for i=2 to n do
if max < list[i] then
max = list[i];
end if
end for.
```

Tutaq ki, max element siyahıda I yerdə yerləşir. Bu halda 1 mənimsətmə operatoru yerinə yetirilir. Tutaq ki, max element siyahıda axırıncı yerdə yerləşir bu halda n mənimsətmə operatoru yerinə yetirilir. Əgər max elementi siyahıda 2-ci yerdə yerləşərsə 2 mənimsətmə operatoru yerinə yetirilər. Göründüyü kimi əməliyyatların sayı ilkin verilənlərin seçilməsindən asılı olur. Belə ki, bir ilkin verilənlər üçün alqoritmi daha sürətli işləyər, digər ilkin verilənlər üçün alqoritm zəif işləyər. Ona görə də alqoritmin mürəkkəbliyinin tətbiqində ilkin verilənləri siniflərə bölmək lazımdır. N=10 götürsək bu siyahıda mümkün yerdəyişmələrin sayı 10!=3628800. Bütün yerdəyişmələr üçün alqoritmi analiz etmək çox çətin olar. Bu halda alqoritmin ilkin verilənlərini aşağıdakı kimi siniflərə bölmək olar.

I sinfə max element I yerdə yerləşən çoxluqları daxil edək.

II sinfə max element II yerdə yerləşən çoxluqları daxil edək və s.

Beləliklə bu məsələnin ilkin verilələrini 10 sinfə bölmək olar.

Bu halda hər sinifdə olan çoxluqların sayı 362880 olar. Alqoritmin mürəkkəbliyinin analizində hər sinifdən bir çoxluğu götürmək kifayətdir. Bu çoxluq üçün olan əməliyyatlarının sayı bu sinfə daxil olan digər çoxluqlar üçün də eynidir.

Müasir kompüterlərin yaddaşının tutumu çox böyük olduğuna görə alqoritmlərin analizində yaddaşın qiymətləndirilməsinə az fikir verilir. Müasir proqram təminatına daxil olan proqramların tərtib olunmasında əsas məqsəd əməliyyatların sayının minimal olması, vaxtın minimal olması götürülür.

Alqoritmlərin yerinə yetirilməsi vaxtının ölçü vahidinin seçilməsində ümumi gəbul edilmiş *vaxtın ölçülməsi vahidlərdən* (saniyə, millisaniyə və s.) istifadə etmək olardı. Ancaq belə yanaşmanın aşkar çatışmazlıqları var. Bu çatışmazlıqlar *konkret kompüterin sürətindən*, maşın kodunun qenerasiyası üçün istifadə olunan *kompilyatorun tipindən* və s. asılıdır. Bizim qarşımızda alqoritmi reallaşdıran *proqramın effektivliyi yox* alqoritmin effektivliyinin ölçülməsi məsələsi durur. Buna görə elə ölçü sistemindən istifadə etmək lazımdır ki, o, yuxarıda göstərilən *kənar faktorlardan* asılı olmasın. Mümkün yanaşmalardan biri alqoritmin hər bir yerinə yetirilən əməliyyatlarının sayını hesablamaqdır. Ancaq bu çox çətin olardı. Ona görə biz gərək baza əməliyyatlar (basic operation) adlanan alqoritmdə icra olunan əsas əməliyyatların siyahısını təyin edək və müəyyənləşdirək ki, bunlardan hansının icrasına daha çox vaxt sərf edililir və bunların neçə dəfə icra olunduqlarını hesablayaq. Adətən, əsas əməliyyatların siyahısını təyin etmək helə də çətin olmur. Bunlara daxili dövrün işində yerinə yetirilən və çox vaxt aparan əməliyyatlar aid edilir. Məsələn, bir çox çeşidləmə alqoritmlərində nizamlanan siyahının iki elementinin (açarların) müqayisəsi metodu istifadə olunur. Belə növ alqoritmlərdə əsas əməliyyat kimi *açarların müqayisəsi əməliyyatı* sayılır.

Matrislərin bir-birinə vurulması və çoxhədlinin qiymətinin hesablanması alqoritmlərində iki əsas əməliyyatlar – vurma və toplama, istifadə olunur. Bir çox kompüterlərdə iki tam ədədin vurması əməliyyatı toplama əməliyyatından daha çox vaxt aparır. Buna görə, bu alqoritmlərin analizində vurma əməliyyatı baza əməliyyatların siyahısına daxil olur. Beləliklə, alqoritmlərin effektivliyi n ölçüdə giriş verilənlərin emalında yerinə yetirəcəyi əsas əməliyyatların sayı ilə qiymətləndirilir.

Tutaq ki, c_{op} - müəyyən kompüterdə əsas əməliyyatın yerinə yetirilməsi vaxtıdır, və C(n) – alqoritmin icrası zamanı bu əməliyyatın yerinə yetirilməsi sayıdır. Onda verilən kompüterdə bu *alqoritmi reallaşdıran proqramın icrası vaxtı* təqribən bu düsturla təyin etmək olar:

$$T(n)\approx c_{op}C(n)$$

Aydındır ki, bu düsturdan çox ehtiyyatlı istifadə etmək lazımdır. C(n) sayğacın qiymətində alqoritmdə icra olunan əsas olmayan əməliyyatların sayı nəzərə alınmır. Bundan başqa adətən sayğacın qiyməti təqribi təyin etmək olur. c_{op} sabitinin qiyməti də təqribi təyin olur və onun dəqiqliyini qiymətləndirmək asan deyil. Buna baxmayaraq bu düstur sonsuz böyük olmayan və sonsuz kiçik olmayan qiymətlər üçün alqoritmin icrası vaxtının qəbul oluna bilən qiymətləndirməsini verir.

Bu düstur aşağıdakı suallara cavab verməyə imkan verir:

Verilən alqoritmin bu kompüterdə reallaşması, sürəti bu kompüterin sürətindən 10 dəfə çox olan kompüterdə neçə dəfə cəld işləyəcək? Belə gələ bilər ki, cavab aşkardır – 10 dəfə.

Və ya tutaq ki, C(n)=n(n-1)/2. Giriş verilənlərin ölçüsünü 2 dəfə artırsaq, proqramın icrası vaxtı nə qədər uzanacaq? Cavab belə olacaq – təqribən 4 dəfə uzanacaq.

Doğrudan da, kifayət qədər böyük n üçün aşağıdakı düstur doğrudur:

$C(n)=\frac{1}{2}n(n-1)=\frac{1}{2}n^2-\frac{1}{2}n\approx\frac{1}{2}n^2$

$$\frac{T(2n)}{T(n)}\approx\frac{c_{op}C(2n)}{c_{op}C(n)}\approx\frac{\frac{1}{2}(2n)^2}{\frac{1}{2}n^2}\approx 4$$

2-ci sualın cavabına görə c_{op} -nin real qiymətini bilmək lazım deyil, çünki yuxarıdakı kəsrdə o ixtisar olunur. ½ sabit vuruğu da C(n) üçün düsturda ixtisar olunur. Bu səbəblərə görə alqoritmin effektivliyinin analizində kifayət qədər böyük olan qiriş verilənlər üçün ***sabit vuruqlar nəzərə alınmır*** və əsas əməliyyatların ***artma sürətinin*** (***order of growth***) qiymətləndirilməsinə (sabit vuruğu dəqiqliyi ilə) diqqəti yönəldirlər.

Alqoritmlərin effektivliyinin analizinin əsas məqsədi alqoritmin yerinə yetirililməsi vaxtının giriş verilənlərin sonzuzluğa yaxınlaşanda artma sürəti tərtibinin təyin edilməsidir.

Alqoritmlərin effektivliyini ifadə edən onların asimptotik artma sürətinin funksiyalarının muqayisəsi üçün O, θ və Ω işarələrdən istifadə olunur. Əksəriyyət alqoritmlərin effektivliyi bir neçə siniflərə bölünür – sabit, loqarifmik, xətti, *nlogn*, kvadratik, kubik və eksponensial.

Ədəbiyyat

1. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ: М.ЦНМО, 2013.
2. Макконелл Дж. Основы современных алгоритмов. 2-е дополненное издание.М.: Техносфера, 2004.
3. Кнут Д. Искусство программирования. Тома 1, 2, 3. 3-е изд. Пер. с англ. : Уч.пос. М.: Изд. дом "Вильямс", 2003.
4. Левитин А.В. Алгоритмы. Введение в разработку и анализ. М.: Издателький дом "Вильямс" 2008 год.
5. Ахо А., Хопкрофт В., Ульман Д. Структуры данных и алгоритмы. М.: «Вильяме», 2000, -384с.
6. Harel D. Algorithmics:The Spirit of Computing, 2nd ed. Addison-Wesley, Reading, MA, 1992.
7. Knuth D.E. Selected Papes on Computer Scince. CSLI Publications and Cambridje University Press, New York, 1996.

ОСНОВЫ АНАЛИЗА АЛГОРИТМОВ

Бабаева Шакяр Маарифовна

Резюме

Настоящий компьютерный профессионал должен уметь представление о стандартном наборе основных алгоритмов, относящихся к разным областям вычислительной техники и должен уметь разрабатывать новые алгоритмы и анализировать их эффективность. Для решения одной и той же задачи может существовать несколько разных алгоритмов. Анализ алгоритмов является инструментом для выбора алгоритма.

В этой статье мы познакомимся с основными понятиями, использующимся для анализа эффективности алгоритмов. Имеются два вида эффективности алгоритмов:

временной и пространственной. В настоящее время основное внимание сосредотачивается на временной эффективности. Эффективность алгоритмов оценивается по количеству основных операций, которые должен выполнить алгоритм при обработке входных данных размера *n*. Основной целью анализа эффективности алгоритмов является определение порядка роста времени выполнения алгоритма в случае, когда размер входных данных стремится к бесконечности.

FUNDAMENTALS OF ALGORITHMS ANALYSIS

Babayeva Shakar Maarif

Summary

A true computer professional should be able to understand a standard set of basic algorithms related to different areas of computer technology and should be able to develop new algorithms and analyze their effectiveness. There can be several different algorithms for solving the same problem. Algorithms analysis is a tool for choosing an algorithm.

In this article, we will get acquainted with the basic concepts used to analyze the effectiveness of algorithms. There are two types of algorithm efficiency: temporal and spatial. Currently, the focus is on time efficiency.The efficiency of algorithms is estimated by the number of basic operations that the algorithm must perform when processing input data of size *n*. The main purpose of the analysis of the efficiency of algorithms is to determine the order of growth of the execution time of the algorithm in the case when the size of the input data tends to infinity.

Rəyçi: dosent Aslanov Ə.Ə.
İnformatika və Cəbr kafedrasının 23 may 2022 ci il tarixli iclasın 05 saylı protokolu
Daxil olma tarixi 25 may 2022-ci il