**№5. Qidiruv algoritmlar samaradorligi**

**Reja:**

1. Ketma-ket qidiruvni samaradorligi.
2. Indeksli ketma-ket qidiruvni samaradorligi.
3. Qidiruvni mukammallashtirish usullari.
4. Mukammal qidiruv daraxti.

***Kalitli so‘zlar:*** *qidiruv, jadval, fayl, kalit, noyob kalit, ikkilamchi kalit, tashqi kalit, ichki kalit, ketma-ket qidiruv, indeksli ketma-ket qidiruv, qidiruv samaradorligi, jadvalni qayta tartiblash, transpozisiya, binar qidiruv, mukammal daraxt.*

**1. Ketma-ket qidiruvni samaradorligi**

Ixtiyoriy qidiruvning samaradorligi jadvaldagi ma’lumotlarning kalitlari bilan solishtirish soni - S bilan baxolanishi mumkin. Agar taqqoslashlar (solishtirish) soni qancha kichik bo‘lsa, qidiruv algoritmi samaradorligi shuncha yaxshi bo‘ladi.

Massivda ketma-ket qidiruvning samaradorligi quyidagicha bo‘ladi:

*C = 1 rn, C = (n + 1)/2.*

Umuman olganda ro‘yxatda xam samaradorlik yuqoridagi kabi bo‘ladi. Garchi massivda xam bog‘langan ro‘yxatda xam qidiruv samaradorligi bir xil bo‘lsada, ma’lumotlarni massiv va ro‘yxat ko‘rinishda tasvirlashning o‘ziga xos kamchilik va afzalliklari mavjud. Qidiruvning maqsadi - quyidagi jarayonlarni bajarilishidan iborat:

* *Topilgan yozuvni o‘qish;*
* *Qidirilayotgan yozuv topilmasa, uni jadvalga qo‘yish;*
* *Topilgan yozuvni o‘chirish.*

Birinchi jarayon (qidiruvning o‘zi) massiv uchun ham ro‘yxat uchun ham bir xil bo‘ladi. Ikkinchi va uchinchi jarayonda esa qidiruv ro‘yxatli tuzilmada samaraliqroq bo‘ladi (sababi massivda elementlarn siljitish lozim).

Agar k massivda elementlarni siljitishlar soni bo‘lsa, u holda k = (n + 1)/2 bo‘ladi.

**2. Indeksli ketma-ket qidiruvni samaradorligi**

Agar bo‘lishi mumkin barcha holatlar teng extimolli deb olinsa, u holda qidiruv samaradorligini quyidagicha xisoblash mumkin:

Belgilashlar kiritib olamiz: m - indeks o‘lchovi; m = n / p; p - qadam o‘lchovi

*Q = (m+1)/2 + (p+1)/2 = (n/p+1)/2 + (p+1)/2 = n/2p+p/2+1* (\*)

Q ni p bo‘yicha differensiallab uni nolga tenglashtiramiz:

*dQ/dp=(d/dp) (n/2p+p/2+1) = - n / 2 p2 + 1/2 = 0*

*Bu yerdan* 

*(\*) ifodada r o‘rniga ropt ni qo‘yib quyidagi taqqoslashlar sonini olamiz:*



Demak, indeksli ketma-ket qidiruvni samaradorligi tartibi *0 bo‘ladi.*

**3. Qidiruvni mukammallashtirish usullari**

Umuman olganda, jadvalda har bir elementni qidirish extimolligini qandaydir bir qiymat bilan izohlash mumkin. Faraz qilaylik jadvalda qidirilayotgan element mavjud. U holda qidiruv amalga oshirilayotgan barcha jadvalni diskret holatga ega tizim sifatida qarash mumkin xamda unda qidirilayotgan elementni topish extimolligi – bu tizim i-chi holati extimolligi *p(i)* deb olish mumkin.



Jadvalni diskret tizim sifatida qaraganimizda, undagi taqqoslashlar soni diskret tasodifiy miqdorlar qiymatlarini matematik kutilmasini ifodalaydi.

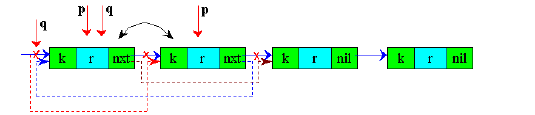
*Z=Q=1p(1)+2p(2)+3p(3)+…+np(n)*

Iloji boricha *p(1)³p(2) ³p(3) ³…³p(n)* bo‘lsa, maqsadga muvofiq bo‘ladi.

Bu shart taqqoslashlar sonini kamaytirib, samaradorlikni oshiradi. Sababi, ketma-ket qidiruv birinchi elementdan boshlanganligi uchun eng ko‘p murojaat qilinadigan elementni birinchiga qo‘yish lozim.

Qidiruv jadvalini qayta tartiblashni eng ko‘p ishlatiladigan ikkita usuli mavjud. Ularni bir bog‘lamli ro‘yxatlar misolida ko‘rib chiqamiz.

Mazkur usulni mag‘zi shundan iboratki, berilgan kalitga teng kalitli element ro‘yxatda birinchi element deb o‘zlashtiriladi, qolganlari esa suriladi.



*1-chizma. Daraxt elementi*

Keltirilgan algoritm ro‘yxat uchun ham massiv uchun xam o‘rinli. Biroq, bu algoritm massiv uchun tavsiya qilinmaydi, sababi elementlarni o‘rinlashtirishga ko‘rsatkichlarni o‘rinlashtirishdan ko‘ra ancha ko‘p vaqt talab qiladi.

Ro‘yxatni qayta tartiblash algoritmi:

*q:=nil;*

*p:=table;*

*while (p <> nil) do*

*begin*

*if key = p^.k then*

*begin*

*if q = nil*

*then 'o‘rinlashtirish shart emas'*

*search := p;*

*exit;*

*end;*

*q^.nxt := p^.nxt;*

*p^.nxt := table;*

*table := p;*

*exit;*

*end;*

*q := p;*

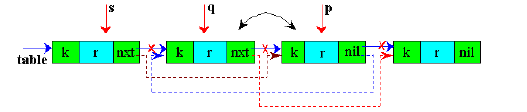
*p := p^.nxt;*

*end;*

*search := nil;*

*exit;*

***Transpozisiya usuli.*** Ushbu usulda topilgan element ro‘xatda bitta oldingi element bilan o‘rin almashtiriladi. Agarda mazkur elementga ko‘p murojaat qilinsa, bittadan oldinga surilib borib natijada ro‘yxat boshida bo‘ladi.



*2- chizma. Qo‘shni elementlarni o‘rnini almashtirish*

r – ishchi ko‘rsatkich

q – yordamchi ko‘rsatkich, r dan bitta qadam orqada bo‘ladi

s - yordamchi ko‘rsatkich, q dan ikkita qadam orqada bo‘ladi

Transpozisiya usuli algoritmi:

*s:=nil;*

*q:=nil;*

*p:=table;*

*while (p <> nil) do*

*begin*

*if key = p^.k then*

*‘transponerlaymiz*

*begin*

*if q = nil then*

*begin*

*‘o‘rinlashtirilmaydi*

*search:=p;*

*exit;*

*end;*

*q^.nxt:=p^.nxt;*

*p^.nxt:=q;*

*if s = nil then*

*table := p;*

*else*

*begin*

*s^.nxt := p;*

*end;*

*search:=p;*

*exit;*

*end; end;*

*search:=nil; exit;*

Ushbu usul nafaqat ro‘yxatda, balki massivda xam qulay (sababi faqatgina ikkita yonma-yon turgan element o‘rin almashtiriladi).

**4. Mukammal qidiruv daraxti**

Agar ajratib olingan elementlar qandaydir o‘zgarmas to‘plamni tashkil qilishsa, kelgusi qidiruvlar samaraliroq bo‘lishi uchun ularni binar daraxt ko‘rinishida ifodalash maqsadga muvofiq bo‘lishi mumkin.

Quyida keltirilgan daraxtlarda binar qidiruvni ko‘rib chiqaylik ( a)va b) chizma). Ikkala daraxt xam uchtadan elementga ega - k1, k2, k3 bo‘lib bu yerda k1<k2<k3. k3 elementni qidirish a) chizmada ikkita taqqoslashni talab qilsa, b) chizmada esa bitta. Biror bir yozuvni ajratib olish uchun zarur bo‘lgan kalitlarni taqqoslashlar soni binar qidiruv daraxtidagi ushbu yozuv bosqichiga birni qo‘shganiga teng.

Faraz qilaylik:

qidiruv argumenti key = k1 bo‘lishi extimolligi - p1,

qidiruv argumenti key = k2 bo‘lishi extimolligi – p3,

qidiruv argumenti key = k3 bo‘lishi extimolligi – p3,

key < k1 bo‘lishi extimolligi – q0,

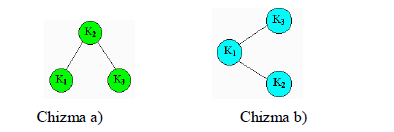
k2 > key > k1 bo‘lishi extimolligi – q1,

k3 > key > k2 bo‘lishi extimolligi – q2,

key > k3 bo‘lishi extimolligi – q3,

C1 - a) chizmadagi taqqoslashlar soni,

C2 - b) chizmadagi taqqoslashlar soni.



U holda ×r1+×r2+×r3+×q0+×q1+×q2+×q3 = 1

Biror bir qidiruvda kutilayotgan taqqoslashlar soni quyidagicha bo‘ladi:

C1 = 2×r1+1×r2+2×r3+2×q0+2×q1+2×q2+2×q3

C2 = 2×r1+3×r2+1×r3+2×q0+3×q1+3×q2+1×q3

Biror bir berilgan kalitlar va extimolliklar to‘plamida kutilayotgan taqqoslashlar sonini minimallashtiruvchi binar qidiruv daraxti mukammal deyiladi.

Garchi daraxt yaratish algoritmi ancha sermashaqqat ish bo‘lsada, biroq u yaratgan daraxtda qiduvni amalga oshirish ancha samarali bo‘ladi. Afsuski, ko‘pincha, qidiruv argumenti extimolligi oldindan aniq bo‘lmaydi.

**Nazorat savollari**

1. Topilgan elementni boshiga qo‘yish usulining transpozisiya usulidan asosiy farqlari nimalardan iborat?
2. Mazkur usullar ma’lumotlar qanday ko‘rinishda berilganda tezroq ishlaydi?
3. Ular qanday ro‘yxatlarda ishlaydi, ya’ni tartiblangan yoki ixtiyoriy?
4. Binar qidiruvning mazmun va mohiyati nimadan iborat?
5. Qanday qilib binar daraxtni chetlab o‘tish mumkin?
6. Binar qidiruvni massivda ishlatish mumkinmi?
7. Agar bo‘sh bo‘lmagan daraxtda ildiz o‘chiriladigan bo‘lsa, u holda uning o‘rniga qaysi element o‘tadi?