

MA'LUMOTLARNI QIDIRISH VA SARALASHNING MURAKKABLIGI

Tojimamatov Israiljon Nurmatovich

*Farg'ona davlat universiteti amaliy matematika va
informatika kafedrası katta o'qituvchisi*

isik80@mail.ru

Jo'raboyeva Surayyo Iqboljon qizi

Farg'ona davlat universiteti talabasi

Annotatsiya: Ushbu maqolada ma'lumotlarni qidirish va saralash algoritmlarining murakkabligi tahlil qilinadi. Algoritmik murakkablik, xususan, vaqt va xotira murakkabligi, qidiruv va saralash jarayonlarida katta ahamiyatga ega bo'lib, katta hajmdagi ma'lumotlarni qayta ishlash samaradorligini belgilaydi. Maqolada Linear Search, Binary Search, Bubble Sort, Merge Sort va Quick Sort kabi asosiy algoritmlar muhokama qilinib, ularning eng yaxshi, eng yomon va o'rtacha holatdagi samaradorliklari tahlil qilinadi. Shuningdek, algoritm tanlash mezonlari va ma'lumotlarning hajmi, tartiblilik darajasi kabi omillarni hisobga olish muhimligi ta'kidlanadi. Maqola qidiruv va saralash jarayonlarining samaradorligini oshirish uchun optimal yechimlarni topishga yordam beradi va zamonaviy hisoblash tizimlari uchun amaliy tavsiyalar beradi.

Kalit so'zlar: Linear Search, Binary Search, Bubble Sort, Merge Sort, Quick Sort

Annotation: This article analyzes the complexity of data retrieval and sorting algorithms. Algorithmic complexity, in particular, time and memory complexity, is of great importance in search and sorting processes and determines the efficiency of processing large volumes of data. The article discusses the main algorithms such as Linear Search, Binary Search, Bubble Sort, Merge Sort, and Quick Sort, and analyzes their best, worst, and average performance. It is also emphasized the importance of taking into account factors such as algorithm selection criteria and data size and order level. The article helps to find optimal solutions for increasing the efficiency of search and sorting processes and provides practical recommendations for modern computing systems.

Key words: Linear Search, Binary Search, Bubble Sort, Merge Sort, Quick Sort

Аннотация: В данной статье анализируется сложность алгоритмов поиска и сортировки данных. Алгоритмическая сложность, в частности сложность времени и памяти, имеет большое значение в процессах поиска и сортировки и определяет эффективность обработки больших объемов данных. В статье обсуждаются основные алгоритмы, такие как линейный поиск, двоичный поиск, пузырьковая сортировка, сортировка слиянием и быстрая сортировка, а также анализируется их лучшая, худшая и средняя производительность. Также подчеркивается важность учета таких факторов, как критерии выбора

алгоритма, размер данных и уровень порядка. Статья помогает найти оптимальные решения для повышения эффективности процессов поиска и сортировки, а также дает практические рекомендации для современных вычислительных систем.

Ключевые слова: *Линейный поиск, двоичный поиск, пузырьковая сортировка, сортировка слиянием, быстрая сортировка*

Algoritmlar informatika fanining asosi bo'lib, katta hajmdagi ma'lumotlarni qayta ishlashni osonlashtiradi. Ayniqsa, qidiruv va saralash algoritmlari tezkor va samarali hisoblash tizimlari uchun juda muhimdir. Ushbu maqola qidiruv va saralash algoritmlarining murakkabligi, ularning tahlil usullari, va optimal yechimlarni topish muammolari haqida so'z yuritadi.

Qidiruv algoritmlari – ma'lum bir qiymatni yoki elementni ma'lumotlar to'plamida topishga mo'ljallangan. Masalan, Binary Search, Linear Search va boshqalar.

Algoritmik murakkablikni tahlil qilishda asosiy e'tibor vaqt va xotira talablariga qaratiladi. Big-O notatsiyasi yordamida algoritmlarning eng yaxshi, eng yomon va o'rtacha holatdagi vaqt murakkabliklari ifodalanadi. Bu esa algoritmlarni taqqoslash va ularni aniq talablar uchun tanlash imkonini beradi. Qidiruv va saralash algoritmlari murakkabligini tahlil qilishda quyidagi omillarni hisobga olish muhim:

1. Eng yaxshi holat (Best-case complexity): Algoritm eng qulay sharoitda ishlaganda vaqt murakkabligi.

2. Eng yomon holat (Worst-case complexity): Algoritm eng murakkab sharoitda ishlaganda vaqt murakkabligi.

3. O'rtacha holat (Average-case complexity): Odatda tasodifiy vaziyatda yoki o'rtacha sharoitda algoritmning ishlash samaradorligi.

Misol uchun, Linear Search algoritmi eng yaxshi holatda (agar qidirilayotgan element birinchi o'rinda joylashgan bo'lsa) $O(1)$ vaqtni oladi. Ammo eng yomon holatda butun massivni qidirib chiqishi kerak, bu esa $O(n)$ vaqtni talab qiladi. Shu sababli Linear Search katta hajmdagi ma'lumotlar uchun mos emas.

Exponential Search: Agar qidiruv massiv tartiblangan bo'lsa, Exponential Search katta massivlarda qidiruvni tezlatishi mumkin. Bu algoritm Binary Search bilan birga ishlatiladi va eng yaxshi holatda $O(\log n)$ murakkablikni ta'minlaydi.

Fibonacci Search: Bu algoritm ham Binary Search kabi tartiblangan massivlar uchun mo'ljallangan. Lekin Fibonacci sonlari yordamida massivni bo'lib boradi va shu tariqa $O(\log n)$ murakkablikka ega bo'ladi. Bu algoritm raqamli qurilmalar uchun samarador hisoblanadi.

Saralash algoritmlari – ma'lumotlarni ma'lum bir tartib bo'yicha qayta joylashtirish uchun ishlatiladi. Misol tariqasida Bubble Sort, Merge Sort, Quick Sort, va boshqa algoritmlarni keltirish mumkin.

Algoritmik murakkablik

Algoritmlarning samaradorligi big-O notatsiyasi orqali tahlil qilinadi. Qidiruv va saralash algoritmlarining murakkabligini o'rganishda vaqt va xotira murakkabligi katta rol o'ynaydi. Masalan:

1. Linear Search: $O(n)$ vaqt murakkabligiga ega, bu esa katta hajmdagi ma'lumotlar uchun unchalik samarali emas.

2. Binary Search: $O(\log n)$ vaqt murakkabligi bilan ancha tezkor, ammo bu algoritm tartiblangan ma'lumotlar bilan ishlaydi.

Saralash algoritmlari esa ancha murakkabroq tahlil talab qiladi. Masalan:

Bubble Sort: $O(n^2)$ vaqt murakkabligi bilan kichik hajmdagi ma'lumotlar uchun mo'ljallangan.

Merge Sort va Quick Sort: $O(n \log n)$ murakkablikka ega bo'lib, ular tez-tez katta ma'lumotlar uchun qo'llanadi.

Insertion Sort: Kichik hajmdagi tartibsiz massivlar uchun juda samarali hisoblanadi. Uning eng yaxshi holati $O(n)$, lekin umumiy murakkabligi $O(n^2)$. Kichik massivlar va qisman tartiblangan massivlar uchun ishlatilishi mumkin.

Heap Sort: $O(n \log n)$ murakkablikka ega bo'lib, xotiradan samarali foydalanadi. Eng katta elementni yoki eng kichik elementni birinchi joyga olib kelishda bu algoritm juda tezdir, ammo amalda tez-tez ishlatilmaydi, chunki Merge Sort yoki Quick Sort ko'proq mos keladi.

Counting Sort, Radix Sort, va Bucket Sort: Ushbu algoritmlar sonlar yoki maxsus ma'lumot turlari uchun aniqroq ishlaydi. Masalan, Counting Sort $O(n + k)$ murakkablik bilan ishlaydi, bu yerda k ma'lumotdagi eng katta qiymatdir. Ushbu algoritmlar yirik ma'lumotlarda samarador, lekin ba'zi cheklavlarga ega.

Algoritmnin tanlashda ma'lumotlarning hajmi va xarakteristikasini hisobga olish muhim. Katta hajmdagi ma'lumotlar bilan ishlashda vaqt va xotira murakkabligini hisoblash algoritm samaradorligini oshiradi. Quyidagi tavsiyalarni keltirish mumkin:

1. Kichik hajmdagi ma'lumotlar uchun: Oddiy saralash algoritmlari (Bubble Sort yoki Insertion Sort) amaliyotda samarali ishlaydi. Ular katta resurs talab qilmaydi va kichik hajmdagi yoki qisman tartiblangan ma'lumotlar bilan samarali ishlaydi.

2. Katta hajmdagi tartiblangan ma'lumotlar uchun: Binary Search eng tezkor qidiruv usuli hisoblanadi, chunki u har bir qadamda ma'lumotlar massivini yarmiga qisqartiradi.

3. O'zgaruvchan hajmdagi va yuqori tezlik talab qilinadigan ma'lumotlar uchun: Quick Sort yoki Merge Sort ma'qul. Quick Sort tez ishlashi bilan mashhur, lekin eng yomon holatda $O(n^2)$ ga tushishi mumkin. Merge Sort esa har doim $O(n \log n)$ murakkablikni kafolatlaydi va katta hajmdagi ma'lumotlarda yaxshi ishlaydi.

Xulosa qilib aytganda, qidiruv va saralash algoritmlari katta hajmdagi ma'lumotlarni qayta ishlashni osonlashtiradi. Ushbu algoritmlarning murakkabligini tahlil qilish orqali biz samarali algoritm tanlash va hisoblash resurslarini tejash imkoniyatiga ega bo'lamiz. Algoritm tanlashda hajm, vaqt va xotira murakkabligini hisobga olish muhimdir.

Texnologiya rivojlanib borar ekan, yangi qidiruv va saralash algoritmlari ham ixtiro qilinib, ularning samaradorligi yaxshilanishi kutiladi.

FOYDALANILGAN ADABIYOTLAR:

1. METANIT.COM
2. Geeksforgeeks
3. **Heineman, G. T., Pollice, G., & Selkow, S.** (2008). *Algorithms in a Nutshell*. O'Reilly Media.
4. **Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C.** (2009). *Introduction to Algorithms* (3rd ed.). MIT Press.
5. **Skiena, S. S.** (2008). *The Algorithm Design Manual* (2nd ed.). Springer.

