Ko‘p o‘lchovli massiv uchun belgilangan funksiyalarni tadqiq qilish va ularning matematik xususiyatlari hamda o‘ziga xosligi

Gulbodom Oybek qizi Norqulova
BXU

Annotatsiya: Ushbu maqolada ko‘p o‘lchovli massivlar uchun belgilangan funksiyalarning matematik xususiyatlari, ularning dasturlashda qo‘llanilishi va didaktik jihatdan o‘rganish metodikasi tadqiq qilingan. Ko‘p o‘lchovli massivlar zamonaviy dasturlash tillarida ma’lumotlarni saqlash va qayta ishlashning muhim vositasi hisoblanadi. Maqolada ikki o‘lchovli va yuqori o‘lchovli massivlar uchun asosiy operatsiyalar, ularning xotira tuzilmasi, indekslash tizimlari va matematik transformatsiyalar tahlil qilingan. Tadqiqot natijalarida massiv funksiyalarining vaqt va xotira murakkabligi baholangan, optimal algoritmlash yondashuvlari tavsiya etilgan hamda ta’lim jarayonida ushbu mavzuni samarali o‘qitish uchun didaktik prinsiplar ishlab chiqilgan.

Kalit so‘zlar: ko‘p o‘lchovli massiv, matritsa operatsiyalari, indekslash algoritmlari, xotira tuzilmasi, vaqt murakkabligi, ikki o‘lchovli massiv, tensorlar, massiv funksiyalari, algoritm samaradorligi, ma’lumotlar strukturasi, massivlarni qayta ishlash

Research on functions defined for multidimensional arrays and their mathematical properties and uniqueness

Gulbodom Oybek qizi Norqulova
BIU

Abstract: This article studies the mathematical properties of functions defined for multidimensional arrays, their application in programming, and didactic learning methods. Multidimensional arrays are an important means of storing and processing data in modern programming languages. The article analyzes the basic operations for two-dimensional and higher-dimensional arrays, their memory structure, indexing systems, and mathematical transformations. The research results estimate the time and memory complexity of array functions, recommend optimal algorithmic approaches, and develop didactic principles for effective teaching of this topic in the educational process.

Keywords: multidimensional array, matrix operations, indexing algorithms, memory structure, time complexity, two-dimensional array, tensors, array functions, algorithm efficiency, data structure, array processing

Kirish. Zamonaviy axborot texnologiyalari va dasturlash sohasida ma’lumotlar bilan ishlash asosiy vazifalardan biri hisoblanadi. Ko‘p o‘lchovli massivlar katta hajmdagi strukturalashgan ma’lumotlarni saqlash va qayta ishlash uchun eng samarali vositalardan biridir. Ular sun’iy intellekt, tasvirlarni qayta ishlash, ilmiy hisoblashlar, statistik tahlil va boshqa ko‘plab sohalarda keng qo‘llaniladi.

Ko‘p o‘lchovli massivlarning nazariy asoslari chiziqli algebra, diskret matematika va hisoblash nazariyasi bilan chambarchas bog‘liq. Ular matematik jihatdan vektorlar, matritsalar va yuqori tartibli tensorlar sifatida qaralishi mumkin. Dasturlash nuqtai nazaridan esa, ko‘p o‘lchovli massivlar murakkab ma’lumotlar strukturalarini ifodalash va ular ustida turli amallarni bajarish imkonini beradi.

Hozirgi kunda Python, Java, C++, MATLAB va boshqa dasturlash tillarida ko‘p o‘lchovli massivlar bilan ishlash uchun keng imkoniyatlar mavjud. Har bir til o‘z ichiga olgan kutubxonalar va funksiyalar orqali massivlar ustida operatsiyalarni amalga oshiradi. Biroq, ushbu funksiyalarning ichki ishlash mexanizmlari, matematik asoslari va samaradorligi ko‘pincha yetarlicha tushunilmaydi.

Ta’lim sohasida ko‘p o‘lchovli massivlarni o‘rgatish jarayonida bir qator didaktik muammolar mavjud. Talabalar uchun ko‘p o‘lchovli tuzilmalarni vizualizatsiya qilish, indekslash tizimini tushunish va murakkab operatsiyalarni amalga oshirish qiyin bo‘lishi mumkin. Shuning uchun, mavzuni bosqichma-bosqich, amaliy misollar bilan birga o‘rgatish zarurati paydo bo‘ladi.

Ushbu tadqiqotning maqsadi ko‘p o‘lchovli massivlar uchun belgilangan funksiyalarning matematik xususiyatlarini o‘rganish, ularning samaradorligini baholash va ta’lim jarayonida qo‘llash uchun didaktik yondashuvlarni ishlab chiqishdan iborat. Tadqiqot ob’ekti sifatida ikki o‘lchovli va yuqori o‘lchovli massivlar, hamda ular ustida bajariladigan asosiy operatsiyalar tanlab olingan.

Asosiy qism

Ko‘p o‘lchovli massivlarning nazariy asoslari

Ko‘p o‘lchovli massiv matematik jihatdan n-o‘lchovli fazodagi diskret nuqtalar to‘plami sifatida qaralishi mumkin. Eng oddiy holat - bir o‘lchovli massiv vektorga, ikki o‘lchovli massiv esa matritsaga mos keladi. Umumiy holda, n-o‘lchovli massiv n-tartibli tensor deb ataladi.

Ikki o‘lchovli massiv A quyidagi ko‘rinishda ifodalanadi:

A = [a(i,j)], bu yerda i = 0, 1, ..., m-1 va j = 0, 1, ..., n-1

Bu yerda m - qatorlar soni, n - ustunlar soni hisoblanadi. Massivning umumiy elementlar soni m × n ga teng.

Ko‘p o‘lchovli massivlarning asosiy xususiyatlari:

- O‘lchovlilik: Massivning nechta indeks bilan aniqlanishi
- Hajmi: Har bir o‘lcham bo‘yicha elementlar soni
- Umumiy razmer: Barcha elementlar sonining ko‘paytmasi
- Xotira hajmi: Massivni saqlash uchun zarur xotira miqdori

Xotira tuzilmasi va indekslash tizimlari

Ko‘p o‘lchovli massivlar xotirada ikki asosiy usulda saqlanadi:

1. Qatorma-qator (Row-major order) Bu usulda massiv elementlari birinchi qator, keyin ikkinchi qator va hokazo tartibida saqlanadi. C, C++, Python tillarida qo‘llaniladi.

Ikki o‘lchovli massiv A[m][n] uchun element A[i][j] ning xotiradagi manzili: address(A[i][j]) = base_address + (i × n + j) × element_size

2. Ustunma-ustun (Column-major order) Elementlar birinchi ustun, keyin ikkinchi ustun tartibida joylashadi. FORTRAN, MATLAB tillarida ishlatiladi.

Element manzili: address(A[i][j]) = base_address + (j × m + i) × element_size

Xotira tuzilmasini tushunish dastur samaradorligini oshirish uchun muhimdir. Kesh xotiradan samarali foydalanish uchun ma’lumotlarga xotirada joylashgan tartibda murojaat qilish tavsiya etiladi.

Asosiy massiv funksiyalari va ularning matematik xususiyatlari

1. Yaratish va initsializatsiya

Ko‘p o‘lchovli massivni yaratish operatsiyasining vaqt murakkabligi O(m×n) bo‘lib, bu yerda m va n massiv o‘lchamlari. Boshlang‘ich qiymatlarni berish ham xuddi shunday murakkablikka ega.

Python misoli:

python

import numpy as npA = np.zeros((m, n)) # *Nollar bilan to‘ldirish*B = np.ones((m, n)) # *Birlar bilan to‘ldirish*C = np.full((m, n), k) # *k qiymati bilan to‘ldirish*

2. Elementlarga murojaat

Bitta elementga murojaat qilish O(1) vaqt murakkabligiga ega. Bu massivlarning asosiy afzalliklaridan biridir.

A[i][j] orqali elementga bevosita murojaat qilish mumkin.

3. Transpozitsiya

Matritsani transpozitsiya qilish A^T operatsiyasi matematik jihatdan: (A^T)[i][j] = A[j][i]

Vaqt murakkabligi: O(m×n) Xotira murakkabligi: O(m×n) - yangi massiv yaratilganda

Transpozitsiyaning muhim xususiyatlari:

- (A^T)^T = A
- (A + B)^T = A^T + B^T

- (kA)^T = kA^T
- (AB)^T = B^T A^T

4. Qo‘shish va ayirish

Ikki o‘lchovli massivlarni qo‘shish va ayirish elementma-element amalga oshiriladi: C[i][j] = A[i][j] ± B[i][j]

Vaqt murakkabligi: O(m×n)

Bu operatsiyalar kommutativ va assotsiativdir:

- A + B = B + A
- (A + B) + C = A + (B + C)

5. Skalyarga ko‘paytirish

Massivni skalyar songa ko‘paytirish: B[i][j] = k × A[i][j]

Vaqt murakkabligi: O(m×n)

Xususiyatlari:

- k(A + B) = kA + kB
- (k + l)A = kA + lA
- k(lA) = (kl)A

6. Matritsalarni ko‘paytirish

A[m×n] va B[n×p] matritsalar ko‘paytmasi C[m×p]: C[i][j] = Σ(k=0 to n-1) A[i][k] × B[k][j]

Standart algoritm murakkabligi: O(m×n×p)

Optimallashtirilgan algoritmlar (masalan, Strassen algoritmi) murakkablikni O(n^2.807) gacha kamaytiradi.

Matritsalarni ko‘paytirishning xususiyatlari:

- Assotsiativlik: (AB)C = A(BC)
- Distributivlik: A(B + C) = AB + AC
- Kommutativ emas: AB ≠ BA (umumiy holda)

7. Qidiruv operatsiyalari

Massivda berilgan elementni qidirish:

- Chiziqli qidiruv: O(m×n)
- Tartiblangan massivda ikkilik qidiruv: O(log(m×n))

Minimal/maksimal element topish: O(m×n)

8. Saralash

Ko‘p o‘lchovli massivni saralash turli mezonga ko‘ra amalga oshirilishi mumkin:

- Barcha elementlarni bir o‘lchovli massiv sifatida saralash: O(mn log(mn))
- Qatorlarni yoki ustunlarni alohida saralash: O(m × n log n) yoki O(n × m log m)

9. Subregio bilan ishlash

Massivning ma’lum qismini (slice) ajratib olish: A[i1:i2, j1:j2]

Bu operatsiya ko‘pincha O(1) yoki O(k) vaqtda amalga oshiriladi, bu yerda k - yangi massiv hajmi.

10. Transformatsiya operatsiyalari

Massiv shaklini o‘zgartirish (reshape): A[m×n] → B[p×q], bu yerda m×n = p×q

Vaqt murakkabligi ko‘pincha O(1), chunki xotirada ma’lumotlar o‘zgarmaydi, faqat indekslash tizimi o‘zgaradi.

Yuqori o‘lchovli massivlar

Uch va undan yuqori o‘lchovli massivlar tensorlar deb ataladi. Ular chuqur o‘rganish, tasvirlarni qayta ishlash (rang kanallari), video tahlil va fizik simulyatsiyalarda muhim rol o‘ynaydi.

Uch o‘lchovli massiv A[l×m×n] xotirada quyidagicha saqlanadi: address(A[i][j][k]) = base_address + (i×m×n + j×n + k) × element_size

Yuqori o‘lchovli massivlar bilan ishlashda:

- Xotira sarfi tez ortadi: O(n^d), bu yerda d - o‘lchovlar soni
- Vizualizatsiya qiyinlashadi
- Hisoblash murakkabligi oshadi

Maxsus massiv turlari va ularning xususiyatlari

1. Simmetrik matritsalar

A[i][j] = A[j][i] bo‘lgan matritsalar. Faqat yarmi saqlanadi, xotirani tejaydi.

2. Diagonal matritsalar

i ≠ j bo‘lganda A[i][j] = 0. Faqat diagonal elementlar saqlanadi.

3. Siyrak matritsalar

Ko‘p elementlari nolga teng massivlar. Maxsus tuzilmalarda saqlanadi (CSR, CSC formatlar).

4. Trikamali matritsalar

Faqat asosiy diagonal va uning atrofidagi elementlar noldan farqli.

Optimallash strategiyalari

Ko‘p o‘lchovli massivlar bilan samarali ishlash uchun:

1. Vektorizatsiya SIMD (Single Instruction Multiple Data) texnologiyasidan foydalanish orqali bir vaqtning o‘zida bir nechta elementlarni qayta ishlash.

2. Parallellashtirish Katta massivlarni qismlarga bo‘lib, parallel oqimlarda qayta ishlash.

3. Kesh optimizatsiyasi Ma’lumotlarga xotirada joylashgan tartibda murojaat qilish orqali kesh xotiradan samarali foydalanish.

4. Bloklash usuli Matritsalarni kichik bloklarga bo‘lib, ular ustida operatsiyalarni bajarish.

5. Xotira boshqaruvi Foydalanilmayotgan massivlarni o‘z vaqtida xotiradan tozalash.

Didaktik yondashuv

Ko‘p o‘lchovli massivlarni o‘qitishda quyidagi prinsiplarga amal qilish kerak:

1. Bosqichma-bosqich o‘rgatish

- Bir o‘lchovli massivlardan boshlash
- Ikki o‘lchovli matritsalarga o‘tish
- Yuqori o‘lchovli tensorlarni o‘rganish

2. Vizualizatsiya Ikki va uch o‘lchovli massivlarni grafiklar orqali ko‘rsatish talabalar uchun tushunishni osonlashtiradi.

3. Amaliy mashqlar

- Oddiy masalalardan murakkabga o‘tish
- Hayotiy misollar bilan ishlash (rasmlar, jadvallar)
- Dasturlash tili va kutubxonalaridan foydalanish

4. Matematik asoslarni tushuntirish Operatsiyalarning matematik mantiqini chiziqli algebra orqali tushuntirish.

5. Xatolarni tahlil qilish Tez-tez uchraydigan xatolar va ularning oldini olish usullari:

- Indeksdan chiqib ketish (Index out of bounds)
- Noto‘g‘ri o‘lchamlar bilan ishlash
- Xotira to‘lib ketishi

6. Murakkablikni baholash Algoritm samaradorligini baholashni o‘rgatish.

Amaliy qo‘llanmalar

Ko‘p o‘lchovli massivlar quyidagi sohalarda qo‘llaniladi:

1. Raqamli tasvirlarni qayta ishlash Rangli tasvir [tinglik×kenglik×3] o‘lchovli massiv sifatida saqlanadi, bu yerda 3 - RGB kanallari.

2. Ma’lumotlar tahlili Jadval ko‘rinishidagi ma’lumotlar ikki o‘lchovli massivda saqlanadi.

3. Sun’iy neyron tarmoqlari Og‘irliklar matritsalari va aktivatsiya tensorlari.

4. Ilmiy hisoblashlar Chiziqli tenglamalar tizimini yechish, diferensial tenglamalarni raqamli yechish.

5. O‘yinlar O‘yin maydoni ikki o‘lchovli massiv sifatida ifodalanadi.

6. Statistik tahlil Ko‘p o‘zgaruvchili ma’lumotlar bilan ishlash.

Dasturlash tillari va kutubxonalar

Turli dasturlash tillarida ko‘p o‘lchovli massivlar bilan ishlash xususiyatlari:

Python (NumPy)

- Yuqori darajadagi abstraktsiya
- Ko‘plab tayyor funksiyalar
- Samarali C/Fortran bilan integratsiya

MATLAB

- Matritsalar bilan ishlash uchun maxsus yaratilgan
- Kuchli vizualizatsiya vositalari
- Matematik operatsiyalarni oson amalga oshirish

Java

- Ko‘p o‘lchovli massivlar massivlar massivi sifatida amalga oshiriladi
- Qat’iy tip tekshiruvi
- Ob’ektga yo‘naltirilgan yondashuv

C/C++

- Past darajali nazorat
- Yuqori samaradorlik
- Qo‘lda xotirani boshqarish zarurati

Xatoliklar va ularning oldini olish

Ko‘p o‘lchovli massivlar bilan ishlashda tez-tez uchraydigan muammolar:

1. Xotira to‘lib ketishi Yechim: Katta massivlarni qismlarga bo‘lib qayta ishlash, generator funksiyalaridan foydalanish.

2. Sekin ishlash Yechim: Algoritmlami optimallash, vektorizatsiya, parallellashtirish.

3. O‘lchamlar mos kelmasligi Yechim: Operatsiyadan oldin massiv o‘lchamlarini tekshirish.

4. Indekslash xatolari Yechim: Chegaralarni tekshirish, xavfsiz indekslash usullaridan foydalanish.

Xulosa. Ko‘p o‘lchovli massivlar zamonaviy dasturlashning ajralmas qismi bo‘lib, turli sohalarda keng qo‘llaniladi. Ushbu tadqiqot natijasida quyidagi xulosalarga kelamiz:

1. Nazariy asos: Ko‘p o‘lchovli massivlar matematik jihatdan vektorlar, matritsalar va tensorlar bilan chambarchas bog‘liq. Ularning nazariy asoslari chiziqli algebra va diskret matematikaga tayanadi.

2. Xotira tuzilmasi: Massivlarning xotirada saqlanish tartibi (qatorma-qator yoki ustunma-ustun) dastur samaradorligiga sezilarli ta’sir ko‘rsatadi. Kesh xotiradan samarali foydalanish uchun ma’lumotlarga xotiradagi joylashuv tartibida murojaat qilish muhimdir.

3. Operatsiyalar murakkabligi: Asosiy operatsiyalarning vaqt murakkabligi aniq baholangan. Oddiy operatsiyalar (qo‘shish, ayirish) O(m×n), murakkab operatsiyalar (ko‘paytirish) O(m×n×p) murakkablikka ega. Bu bilim algoritmlarni optimallashtirishda yordam beradi.

4. Optimallash imkoniyatlari: Vektorizatsiya, parallellashtirish, bloklash va kesh optimizatsiyasi orqali samaradorlikni sezilarli darajada oshirish mumkin. Zamonaviy kutubxonalar (NumPy, MATLAB) bu optimallashtirishlarni ichki amalga oshiradi.

5. Didaktik jihatdan**:** Ko‘p o‘lchovli massivlarni o‘qitishda bosqichma-bosqich yondashuv, vizualizatsiya va amaliy mashqlar tavsiya etiladi. Matematik asoslarni tushuntirish bilan birga amaliy dasturlash ko‘nikmalarini rivojlantirish zarur.

6. Amaliy ahamiyat: Tasvirlarni qayta ishlash, sun’iy intellekt, ma’lumotlar tahlili, ilmiy hisoblashlar va boshqa ko‘plab sohalarda ko‘p o‘lchovli massivlar asosiy vosita hisoblanadi.

7. Til va kutubxonalar: Har bir dasturlash tili o‘ziga xos afzalliklarga ega. Python/NumPy qulaylik va samaradorlik, MATLAB matematik hisoblashlar, C/C++ maksimal nazorat va tezlik uchun mos keladi.

Tadqiqot ko‘rsatdiki, ko‘p o‘lchovli massivlarni chuqur tushunish uchun nazariy bilim, amaliy ko‘nikmalar va samaradorlik tamoyillarini birlashtirish zarur. Bu bilimlar asosida murakkab masalalarni samarali yechish, optimal algoritmlar yaratish va talabalarni to‘g‘ri yo‘naltirish mumkin.

Kelajakda tadqiqotni davom ettirishning muhim yo‘nalishlari:

- Kvant hisoblash muhitida ko‘p o‘lchovli strukturalar bilan ishlash
- GPU va TPU kabi maxsus protsessorlarda optimallash usullari
- Yangi dasturlash paradigmalari (funksional dasturlash) kontekstida massivlar bilan ishlash
- Virtual va kengaytirilgan reallik uchun yuqori o‘lchovli ma’lumotlar strukturalari

Foydalanilgan adabiyotlar

1. Cormen T.H., Leiserson C.E., Rivest R.L., Stein C. Introduction to Algorithms, 3rd Edition. MIT Press, 2009. 1312 p.

2. Press W.H., Teukolsky S.A., Vetterling W.T., Flannery B.P. Numerical Recipes: The Art of Scientific Computing, 3rd Edition. Cambridge University Press, 2007. 1235 p.

3. Strang G. Introduction to Linear Algebra, 5th Edition. Wellesley-Cambridge Press, 2016. 584 p.

4. Goodfellow I., Bengio Y., Courville A. Deep Learning. MIT Press, 2016. 775 p.

5. Van Rossum G., Drake F.L. The Python Language Reference Manual. Network Theory Ltd, 2011. 151 p.

6. Harris C.R., Millman K.J., van der Walt S.J. et al. Array programming with NumPy // Nature. 2020. Vol. 585. P. 357-362.

7. Kolter J.Z., Johnson M.J. Convex Optimization for Machine Learning. Cambridge University Press, 2020. 234 p.

8. Ахо А., Хопкрофт Дж., Ульман Дж. Структуры данных и алгоритмы. М.: Вильямс, 2000. 384 с.

9. Кнут Д. Искусство программирования, том 1. Основные алгоритмы. М.: Вильямс, 2006. 720 с.

10. Вирт Н. Алгоритмы и структуры данных. СПб.: Невский Диалект, 2001. 352 с.

11. Sedgewick R., Wayne K. Algorithms, 4th Edition. Addison-Wesley Professional, 2011. 976 p.

12. Skiena S.S. The Algorithm Design Manual, 3rd Edition. Springer, 2020. 793 p.

13. Golub G.H., Van Loan C.F. Matrix Computations, 4th Edition. Johns Hopkins University Press, 2013. 756 p.

14. Trefethen L.N., Bau D. Numerical Linear Algebra. SIAM, 1997. 361 p.

15. McKinney W. Python for Data Analysis, 2nd Edition. O‘Reilly Media, 2017. 544 p.