INTERPOLĀCIJAS MEKLĒŠANA
INTERPOLATION SEARCH

Autori: **Dzintars Apeināns**, **Vasilijs Steklovs,** e-pasts: 2009a02@inbox.lv, vasilij_s@inbox.lv, +37128327720, +37129927757
Zinātniskā darba vadītājs: **Pēteris Grabusts, Dr.sc.ing.,** e-pasts: peteris.grabusts@rta.lv
Rēzeknes Tehnoloģiju akadēmija, Atbrīvošanas aleja 115, Rēzekne, Latvija

Abstract. *Searching for information may take some time because of slow searching algorithm work. Some of searching algorithm work comparing needed value with all array's value of step by step. In this work will be looking about interpolation search which predict looking value place in array. Main idea is to compare searching algorithm by time which it take to find correct value and will make conclusions which of compared algrithm work faster with different size of array.*

Keywords: *Algorithm, interpolation, search*

Interpolācijas meklēšana

Interpolācija meklēšana (dažreiz saukta par ekstrapolāciju meklēšana vai "paredzēšanas" meklēšanu) ir algoritms, kurš meklē konkrēto atslēgas vērtību indeksētā masīvā. Algoritms strādā paralēli, līdzīgi kā cilvēks pārlūkojot telefona grāmatu meklējot konkrētu vērtību. Katrā meklēšanas solī tiek aprēķināts kāds varētu būt atlikušais meklēšanas apgabals, balstoties uz atslēgas vērtībām pie meklēšanas apgabala robežām un vērtību indeksā. Parasti tiek izmantota lineārā interpolācija. Meklējamā vērtība faktiski tiek konstatēta paredzot tās pozīciju salīdzinot ar atslēgas vērtību. Ja salīdzinātā vērtība ar salīdzināmo vērtību nav vienādas, tad atkarība no salīdzinājuma, atlikušais meklēšanas apgabals tiek samazināts līdz daļai, pirms vai pēc aprēķinātā stāvokļa. Dotā metode strādās tikai tad, ja aprēķinātā masīva izmēra starpība starp atslēgas vērtībām ir jūtama[1].

Salīdzinājumam, binārā meklēšana, meklēšanas apgabalu sadala uz pusēm, sākotnēju veicot meklēšanu tikai vienā meklēšanas apgabala pusē, ja vērtība dotajā pusē netiek atrasta notiek meklēšana otrajā meklēšanas apgabala pusē. Atlikušais meklēšanas apgabals ir samazināts līdz daļai, pirms vai pēc tam, paredzamās pozīcijas, kur var atrasties meklējamā vērtība.

Interpolācijas meklēšanu vidēji veic:

$$\log(\log N) \tag{1}$$

salīdzinājumu, kur:

log - matemātiskā funkcija logaritms,
N - ir elementu skaits masīvā starp kuriem jāmeklē.

Sliktākajā gadījumā tā var nonākt līdz:

$$O(N) \tag{2}$$

salīdzinājumiem, kur:
O - matemātiskā notācija priekš salīdzināšanas asimptotiskās uzvedības funkcijā,
N - ir elementu skaits masīvā starp kuriem jāmeklē

Pielāgošanās dažādām datu kopām

Kad atslēgas vērtības tiek kārtotas datu kopā, tās vienmērīgi tiek sadalītas numuros, lineāro interpolāciju ir vienkārši īstenot un atrast indekss ļoti tuvu meklējamai vērtībai.

No otras puses, uz telefona grāmatas vērtībām sakārtoti pēc nosaukuma, tad vienkāršā pieeja interpolācijas meklēšanai neattiecas. Tie paši principi joprojām var attiekties, lai gan: var

http://dx.doi.org/10.17770/het2016.20.3500

novērtēt meklējamās vērtības pozīcijas telefona grāmata izmantojot relatīvās frekvences burtiem nosaukumos un izmantot šo kā meklēšanas apgabala sākumu.

Daži interpolācijas meklēšanas implementāciju varianti var nedarboties, kā paredzēts, kad tiek palaista vienlīdzīga vērtība atslēgas vērtībai kura jau eksistē. Vienkāršākā implementācija interpolācijas meklēšanas ne vienmēr izvēlēties pirmo (vai pēdējā) elements šādai realizēšanai[1].

"Book-based" meklēšana

Nosaukumi telefona grāmatā, savā veidā nenosaka acīmredzamu sakārtošanu pēc kārtas, neskaitot vienotu sadalījumu(piemēram nosaukums #1, nosaukums #2 u.c.) un vēl vairāk, tas ir labi zināms, ka daži vārdi ir daudz biežāk sastopami nekā citi. Līdzīgi ar vārdnīcām, kur ir daudz vairāk vārdi, kas sākas vienādi, bet vārds beidzas ar dažādiem burtiem nekā iepriekšējais vārds[2].

Veiktspēja

Ja mēs neizdarīsim nekādus pieņēmumus par atslēgu izplatīšanu masīvā, interpolācija meklēšana ir:

(3)

$$O(N)$$

kur: O - matemātiskā notācija priekš salīdzināšanas asimptotiskās uzvedības funkcijā,

N - ir elementu skaits masīvā starp kuriem jāmeklē,jo masīvs var būt eksponenciālo sadalījumu atslēga. Tomēr, ja atslēgas ir vienoti sadalītas, interpolācijas meklēšana ir:

(4)

$$O(\log(\log N))$$

kur: O - matemātiskā notācija priekš salīdzināšanas asimptotiskās uzvedības funkcijā,
log - matemātiskā funkcija logaritms,
N - ir elementu skaits masīvā starp kuriem jāmeklē[1][3].

Tāpat kā lielākā daļa datorzinātnēs, interpolācija meklēšana ietver kompromisu. Mēs palielinām aprēķinu skaitu iesaistītu katrā solī, cerot samazināt soļu daudzumu.

Tas var būt noderīgi, kad jums ir neindeksēts, sakārtots masīvs uz cietā diska, un jums vajag atrast atslēgas vērtību.

Piemēram, ja jūs zināt, ka atslēgas sadalījums ir eksponenciāls, jūs varat aprēķināt vidējo elementu kā:

(5)

$$mid = low + ((key - \log(arr[low])) * (high - low)) / (\log(arr[high]) - \log(arr[low]))$$

kur: mid - masīva izmēra vidējā pozīcija, veseli skaitļi,
low - masīva izmēra mazākā pozīcija, veseli skaitļi,
high - masīva izmēra lielākā pozīcija, veseli skaitļi,
arr - mainīgā masīva nosaukums,
log - matemātiskā funkcija logaritms [4].

Interpolācijas meklēšanas salīdzināšana ar citiem meklēšanas algoritmiem

Izvēloties meklēšanas algoritmus ar kuriem tiktu salīdzināts interpolācijas meklēšanas algoritms, tika izvēlēti lineārās meklēšanas un binārās meklēšanas algoritmi. Lineārais meklēšanas algoritms tika izvēlēts tāpēc, ka tas ir vienkāršākais no meklēšanas algoritmiem. Binārās meklēšanas algoritms tika izvēlēts tāpēc, ka pēc darbības principa tas ir līdzīgs interpolācijas meklēšanas algoritmam. Lai veiktu salīdzināšanu tika īstenota C++ programmēšanas valodas programma ar visiem nosauktajiem meklēšanas algoritmiem. Īstenojot interpolācijas meklēšanas algoritmu tika izskatīts tā pseido kods[5].

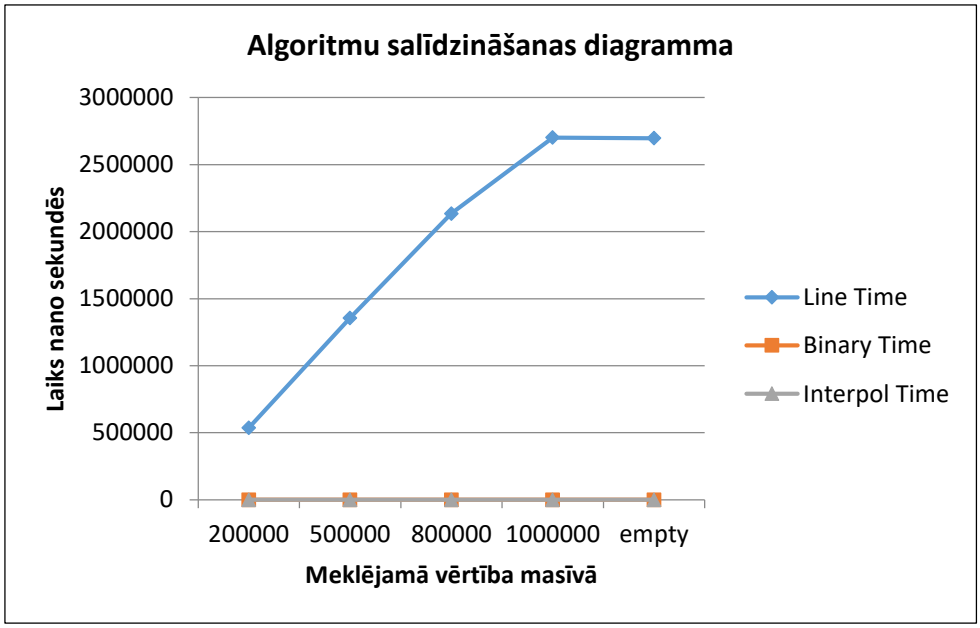
Eksperimentā tika izmantots sakārtots veselu skaitļu masīvs ar 1 000 000 elementiem un tikai veikti 10 eksperimenta mēģinājumi uz katru meklēšanas algoritmu, lai noteiktu vidējo laiku vajadzīgās vērtības atrašanai, kas ir parādīts 1. tabulā. Meklējamās vērtības tika izvēlētas patvaļīgi visa masīva garumā, lai noteiktu kā algoritmi strādā meklējot dažādas vērtības viena garuma masīvā, kā arī tika veikta pārbaude uz meklējamo vērtību, kura neatrodas masīvā.

1. tabula

Algoritmu darbības laiks (nano sekundēs)

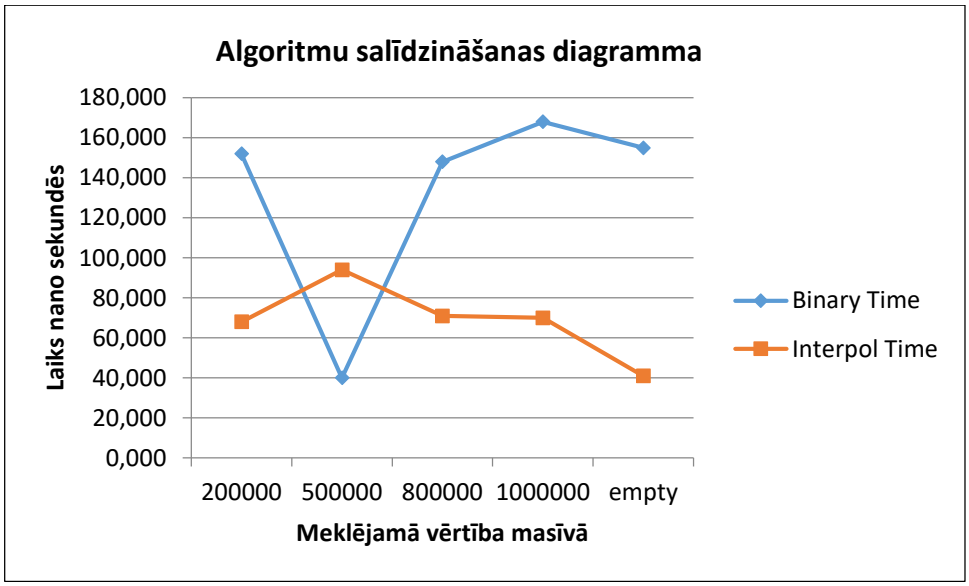
Nr.	Meklējamā vērtība	Lineārā meklēšana	Binārā meklēšana	Interpolācijas meklēšana
1.	200 000	535331	152,009	68,0039
2.	500 000	1356480	40,0023	94,0054
3.	800 000	2134420	148,008	71,0041
4.	1 000 000	2701450	168,01	70,004
5.	Nav vērtības masīvā	2697150	155,009	41,0024

Veicot visu trīs algoritmu salīdzināšanu tika iegūta sekojošs rezultāts, kur ir attēlots diagrammā (skat. 1. attēls).



1. attēls **Trīs meklēšanas algoritmu salīdzināšanas diagramma**

Tāda diagramma tika iegūta sakarā ar to, ka lineārās meklēšanas algoritma darbības laiks salīdzinot ar binārās un interpolācijas meklēšanas laiku ir daudz lielāks, tāpēc binārās un interpolācijas meklēšanas asis ir uz x ass. Lai iegūtu priekštatu par interpolācijas darbības laiku meklējot vajadzīgo elementu, tas tika salīdzināts ar binārās meklēšanas algoritmu un tika iegūts rezultāts, kas tiek attēlots 2. attēlā(skat. 2. attēls).



2. attēls **Binārās un interpolācijas salīdzināšanas diagramma**

Kā redzams diagrammā interpolācijas meklēšanas darbības laiks ir daudz mazāks nekā binārajā meklēšanā. Veicot eksperimentu binārā meklēšanas darbības laiks bija ātrākas nekā interpolācijas meklēšanas darbības laiku, kad tika meklēta vidējā vērtība meklēšanas apgabalā, tas ir redzams 2. attēlā(skat. 2. attēls). To var izskaidrot sekojošo, binārā meklēšana meklēšanas apgabalu sadala uz pusēm un sāk meklēt no viduspunkta turpinot sadalīt meklēšanas apgabalu uz pusēm, kā tas tiek attēlots attēlā 3 (skat. 3. attēls).

3. attēls **Binārās meklēšanas darbības princips**

Savukārt interpolācijas meklēšana tiek veikta meklēšanas apgabala sākumā un beigās, kā tas tiek attēlots attēlā 4., un pēc tam tiek "uzminēta" meklējamā elementa atrašanās vieta. Tas tiek atkārtots kamēr meklējamais elements nav atrasts.

4. attēls **Interpolācijas meklēšanas darbības princips**

Secinājumi

1. Lineārais meklēšanas algoritms neizmanto meklēšanai lielu apmēru datiem;
2. Binārā meklēšanas darbības laiks ir pietiekami ātrs, lai nevajadzētu izmantot interpolācijas meklēšanu;
3. Interpolācijas meklēšanas priekšrocības parādās, ja nepieciešams meklēt starp miljardiem ierakstu;
4. Teorētiski binārās meklēšanas ātrdarbība ir tāda pati kā interpolācijas, bet praktiski interpolācijas meklēšana strādā ātrāk.

Summary

Interpolation search is an algorithm for searching for a given key value in an indexed array that has been ordered by the values of the key. It parallels how humans search through a telephone book for a particular name, the key value by which the book's entries are ordered. In each search step it calculates where in the remaining search space the sought item might be, based on the key values at the bounds of the search space and the value of the sought key, usually via a linear interpolation. The key value actually found at this estimated position is then compared to the key value being sought. If it is not equal, then depending on the comparison, the remaining search space is reduced to the part before or after the estimated position. This method will only work if calculations on the size of differences between key values are sensible.

By comparison was used three different algorithm, the binary search always chooses the middle of the remaining search space, discarding one half or the other, again depending on the comparison between the key value found at the estimated position and the key value sought. The remaining search space is reduced to the part before or after the estimated position. The linear search uses equality only as it compares elements one-by-one from the start, ignoring any sorting. Comparing was made in C++ programm where was using integer array with size 1 million and was searched different value. As was said before linear search compare all value one-by-one, binary search choose middle of searching space and interpolation search "predict" where looking value can be in array.

By doing experiment was made some conclusion. That linear searching algorithm better use for small array size; binary search theoretical performance is same as interpolation, but in practice interpolation search work faster. Interpolation search better use for search in array with milliard records in it.

Literatūra

1. Andersson, Arne, and Christer Mattsson. ‘Dynamic Interpolation Search in o(log log n) Time’. In Automata, Languages and Programming, edited by Andrzej Lingas, Rolf Karlsson, and Svante Carlsson, 700:15–27. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 1993 (angļu valodā, skatīts 2016.03.30)
2. Algoritmi: Interpolācijas meklēšana - http://www.stoimen.com/blog/2012/01/02/computer-algorithms-interpolation-search/ (angļu valodā, skatīts 2016.04.01)
3. Armenakis, A. C., Garey, L. E., Gupta, R. D., An adaptation of a root finding method to searching ordered disk files, BIT Numerical Mathematics, Volume 25, Number 4 / December, 1985. (angļu valodā, skatīts 2016.03.30)
4. Interpolācijas meklēšana datu struktūrā - https://www.quora.com/What-is-interpolation-search-in-data-structures (angļu valodā, skatīts 2016.04.02)
5. Interpolācijas meklēšanas algoritms - http://www.tutorialspoint.com/data_structures_algorithms/interpolation_search_algorithm.htm (angļu valodā, skatīts 2016.04.04)