

Degree Project

Presentation #1

Erik Regla

Universidad de Talca

April 28, 2020

- ▶ Advisor: Rodrigo Paredes (rapa)
- ▶ Thema: Worst-Case optimal incremental sorting for discrete classes
- ▶ Current Status: A paper. Initial tests and code ready. Pending documentation.
- ▶ Motivation: Everyone loves to talk big about algorithms, but no one actually implements them.

IQS (Set A , Index idx , Stack S)

// Precondition: $idx \leq S.\text{top}()$

1. **If** $idx = S.\text{top}()$ **Then** $S.\text{pop}()$, **Return** $A[idx]$
2. $pid x \leftarrow \text{random}[idx, S.\text{top}()-1]$
3. $pid x' \leftarrow \text{partition}(A, A[pid x], idx, S.\text{top}()-1)$
// Invariant: $A[0] \leq \dots \leq A[idx-1] \leq A[idx, pid x' - 1] \leq A[pid x']$
// $\leq A[pid x' + 1, S.\text{top}()-1] \leq A[S.\text{top}(), m-1]$
4. $S.\text{push}(pid x')$
5. **Return** **IQS**(A, idx, S)

Figure: IQS algorithm as published in 10.1007/s00453-010-9400-6

```

1: procedure IIQS( $A, S, k$ )
2:   while  $k < S.top()$  do
3:      $pid_x \leftarrow \text{random}(k, S.top() - 1)$ 
4:      $pid_x \leftarrow \text{partition}(A_{k, S.top()-1}, pid_x)$ 
5:      $m \leftarrow S.top() - k$ 
6:      $\alpha \leftarrow 0.3$ 
7:      $r \leftarrow -1$ 
8:     if  $pid_x < k + \alpha m$  then
9:        $r \leftarrow pid_x$ 
10:       $pid_x \leftarrow \text{pick}(A_{r+1, S.top()-1})$ 
11:       $pid_x \leftarrow \text{partition}(A_{r+1, S.top()-1},$ 
12:                            $pid_x)$ 
13:     else if  $pid_x > S.top() - \alpha m$  then
14:        $r \leftarrow pid_x$ 
15:        $pid_x \leftarrow \text{pick}(A_{k, pid_x})$ 
16:        $pid_x \leftarrow \text{partition}(A_{k, r}, pid_x)$ 
17:        $r \leftarrow -1$ 
18:     end if
19:      $S.push(pid_x)$ 
20:     if  $r > -1$  then
21:        $S.push(r)$ 
22:     end if
23:   end while
24:    $S.pop()$ 
25:   return  $A_k$ 
26: end procedure

```

Figure: IQS algorithm as published in 10.1109/SCCC.2015.7416566

https://ieeexplore.ieee.org/document/7416566

IEEE.org | IEEE Xplore Digital Library | IEEE-SA | IEEE Spectrum | More Sites

Cart | Create Account | Personal Sign In

IEEE Xplore® Digital Library

Institutional Sign In

IEEE

Browse My Settings Get Help Subscribe

All Enter keywords or phrases (Note: Searches metadata only by default. A search for "smart grid" = "smart AND grid")

Advanced Search | Other Search Options

Conferences > 2015 34th International Conference on Distributed Computing and Security

Worst-case optimal incremental sorting

Publisher: IEEE

Cite This PDF

2 Author(s) Erik Røglø, Rodrigo Paredes View All Authors

41 Full Text Views

Abstract

Document Sections

1. Introduction
2. Incremental Quicksort
3. Sketch of Our Proposal
4. Known Solutions and Issues
5. Introduction on IQS

Authors

Figures

Abstract:

We present an online algorithm to, given a fixed array A , retrieve its k smallest elements in optimum time for the worst case, that presents fast response in practice. For this, we devise an introspective version of the Incremental Quicksort (IQS) algorithm, which controls the size of the auxiliary stack of the IQS algorithm via an introspective criteria.

Published in: 2015 34th International Conference of the Chilean Computer Science Society (SCCC)

Date of Conference: 9-13 Nov. 2015

Date Added to IEEE Xplore: 25 February 2016

ISBN Information:

INSPEC Accession Number: 15000127

DOI: 10.1109/SCCC.2015.7416566

Publisher: IEEE

Conference Location: Santiago, Chile

More Like This

GPU sample sort
2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS)
Published: 2010

Exploring the Energy Consumption of Data Sorting Algorithms in Embedded and Mobile Environments
2006 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware
Published: 2009

Top Organizations with Patents on Technologies Mentioned in This Article

ORGANIZATION 1

ORGANIZATION 2

ORGANIZATION 3

ORGANIZATION 4

Figure: IIQS implementation changes the partition method in order to guarantee a partition of linear time and at the same time guarantee a reduction on the search space. (10.1109/SCCC.2015.7416566).

Current scope is limited as an experimental algorithm design¹ to extend (I)IQS usage for haplotype plot² generation, which is an instance of the worst case for IQS but on a discrete space when $C \ll n$.

Tested variants of the original implementation are as follows:

- ▶ Test 1: Add incremental version of BFPRT algorithm
- ▶ Test 2: Change rules for introspective step
- ▶ Test 3: Bias the three-way-median returned index
- ▶ Test 4: Store the three-way-median result on the stack
- ▶ Test 5: Alter rules for storing pivots

¹[doi:10.1017/CBO9780511843747](https://doi.org/10.1017/CBO9780511843747)

²[doi:10.1111/2041-210X.12747](https://doi.org/10.1111/2041-210X.12747)

The beforementioned changes can induce new problems and behaviours, known but not limited to stack size, cache trashing, non-existent DRAM bursting, memory corruption, etc.

- ▶ Test 1: No changes to running speed, improved median selection with less memory usage
- ▶ Test 2: Depends on the distribution (not applicable if it is not known).
- ▶ Test 3: Stabilization over running time and stack usage. Increased DRAM burst toll.
- ▶ Test 4: Cache and DRAM burst trashing on low-power devices. Trade-off between use cases as it doesn't perform on the general case.
- ▶ Test 5: Depends on the input distribution, useless if not combined with one of the former tests.

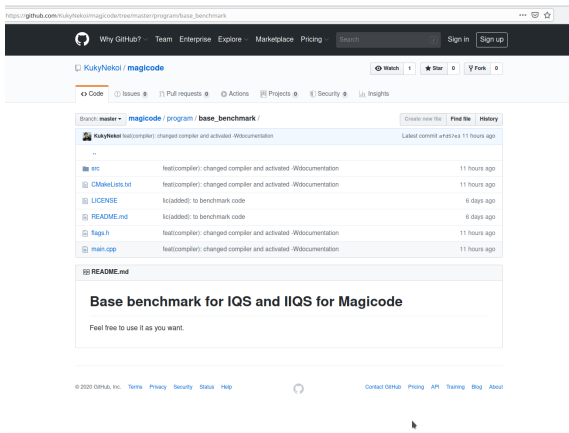


Figure: Implementation for the tests on C++ (STL-container-friendly implementations and without STL) available under GNU GPL license at GitHub

- ▶ Scope: Experimental design, setup and experimentation
- ▶ Part of magicode, a personal research on FPGA implementation of hardware accelerators for similarity search (the original thema).
- ▶ Got someone interested on using this algorithm for solving haplotype plots.

FIN