Algorithmics (6EAP)
MTAT.03.238

Organisation of course

Jaak Vilo

2022 Fall
Lecturer

• 1986-1991 U Tartu (diploma)
• 1991-1999 U Helsinki, PhD (pattern discovery)
• 1999-2002 EMBL-EBI, UK (bioinformatics)
• 2002- EGeen, Quretec (Biobank and Data Mgmnt)
• Uni Tartu CS, professor (Bioinformatics) 2007
  – STACC – Software Technologies and Applications Competence Center (Tarkvara TAK)
  – research projects, BIIT research group biit.cs.ut.ee
  – Head of #UniTartuCS (from 2011)
Co-lecturer

Kallol Roy, PhD

Kallol Roy is currently working as an Assistant Professor at Institute of Computer Science, University of Tartu, Estonia. His research interests are Deep Learning, Computer Vision, AI accelerator, and electromagnetics.

He worked as a postdoctoral researcher at Packaging Research Center, Georgia Institute of Technology, Atlanta, USA and Statistical Artificial Intelligence Lab, Ulisan National Institute of Technology, South Korea and at Department of Mathematics, Indian Institute of Science Bangalore. He did his Bachelors in Electrical Engineering from Indian Institute of Technology (IIT K), Kanpur and Ph.D. in Electrical Communication Engineering from Indian Institute of Science (IISc) Bangalore.

He is a recipient of APS-IUSSTF Physics Student Visitation Award, 2012 Microsoft Travel Award, Sterlite Best Paper Award at Photonics 2010, IIT Guwahati, MHRD Scholarship, Government of India 2007, Jawaharlal Nehru Scholarship, Steel Authority of India Limited, 2000.

The current project running in the lab is: Machine learning to predict future crimes.

https://kallolroy.me/
Teaching Assistants

please upload better pictures here :)

Ayushmat Soni  Eduardo Brito  Muhammad Uzair
• Lectures – recorded and available on Panopto

• Meetings – for interactive sessions.
• Be ready to present, to ask questions, discuss

• Zoom, potentially recorded (online or local)
  – Practice sessions are not recorded (for student privacy)
Goals

• To learn the main concepts and techniques of the algorithm design and analysis – the practical skills and (some) theoretical basis

• To be able to choose, implement, design, analyze and compare algorithms and data structures

• Learn to learn, use knowledge, solve tasks, program efficiently, read, write, and present

• Equalizing backgrounds for students from different degree programmes etc. (Leveling course)
Algorithms, data structures

• Organising data (in memory, on disk, files, …)
• Scaling data management
• Search
• Optimization

• Bread and butter of efficiency
Algorithmics

This course aims to be a general overview of algorithms, data structures, practical programming, heuristic search, etc. As such it is aimed as "equalizer" course for students coming from various backgrounds.

This course is an obligatory course in the MSc of Computer Science (usually taken during the 1st semester). It is also highly advisable for other curricula like Data Science, Cybersecurity and Software Engineering or specializations in interdisciplinary subjects like bioinformatics, statistics or mathematics.

Lectures

Lecture/consultation: Tuesday 10.15 - 12.00 Delta - 1021 (Jaak Vilo, Kallol Roy)

(let's agree on this in the first lecture)

Main content of lectures has been pre-recorded and available online on Panopto.ut.ee

Practice session

(every week starting from week 2; Registration via SIS/ÖIS ois.ut.ee)

- group 1. Tuesday 12.15 - 14.00 Delta - 1019 (Ayushmat Soni)
- group 2. Thursday 14.15 - 16.00 Delta - 2034 (Eduardo Brito)
- group 3. Friday 14.15 - 16.00 (Muhammad Uzair, Online only)

Contacts

- Responsible Lecturer: Prof. Jaak Vilo vilo@ut.ee (Delta:3119)
- Teaching assistants:
  - Kallol Roy kallol.roy@ut.ee
  - Eduardo Brito eduardo.ribas.brito@ut.ee
  - Ayushmat Soni ayushmats@gmail.com
Contact hours

• Lectures: Jaak Vilo on Panopto (from 2020)
  – Tue 10-12 (Delta 1021; mostly Online; consulting)
  – Jaak Vilo and Kallol Roy

• Weekly practical sessions (homework, discuss):
  G1. Tuesday 12.15 - 14.00 Delta - 1019 (Ayushmat Soni)
  G2. Thursday 14.15 - 16.00 Delta - 2034 (Eduardo Brito)
  G3. Friday 14.15 - 16.00 (Muhammad Uzair, Online only)
Slack

Slack:
https://join.slack.com/t/unitartucretsalg-feb1051/shared_invite/zt-1erjhkk02-ExFOOGAw3LW4v0oLKrgGXQ
Practical work and grading

- Homework
- Essay → Initial submission → Peer review → Final submission
- Project → Implementation → Presentation
- Exam
Course Grade

- Lectures
- Homework 30 + bonus points 0-30
- Project 20
- Essay 10
- Exam 40

All components obligatory

- Total 100p
Homework (obligatory)

Most essential part of the course

- **12 weeks of homeworks** ($12w \times 5 = 60$ tasks)
- Points: nr of tasks – 20  (e.g. 40hw -> 20 p)
- Presentations orally during the practicals
- Submissions over the web,
- **deadline** – Every **Monday 23:59**
50% minimum threshold

• Obligatory to get **a minimum of 50% done**
  – 30 tasks - 20 = **10 points (out of 30 max)**

• Obligatory presence at the practice sessions
  – 70% (8 out of 12 weeks)
  – Missed out of 8 – subtract **3** points for absence
  • E.g. **6 times present** – deduct **6** points from already earned practice session points
“Essay” (obligatory)

- Will be based on some article – 2 page summary

- To be decided during the course

- Reading and writing skills, peer review

- A format of the scientific article (abstract, citations, etc)
Project (obligatory)

• A practical algorithm implementation plus analysis and comparisons of efficiency

• Presentation in the form of a project report in scientific style (poster, report, ...)

Memory Complexity Improvement for the Prime-Box Parallel Search Algorithm

Gregor Eesmaa, Mathias Plans, Roberts Oskars Komarovskis
University of Tartu, Institute of Computer Science, Algorithmics (MTAT.03.238)
https://github.com/mathiasplans/primebox

Abstract

We focus on reducing space complexity in "Prime Box Parallel Search" by Pandey et al, describing the obtained solution and providing comparison of the methods.

Introduction

The Prime-box Parallel Search algorithm by Pandey et al has a serial time complexity of O(n) for dictionary search, where n is the length of the searched word. However, there were some apparent space limitations - as an example, up to four-letter words could be stored, possibly not all.

Indices of words in a lookup are computed as products of different prime numbers (Table 1). For example, index of "TIB" would be 71 * 193 * 251 * 103563. As such, the memory is left unbalanced by a factor space population of the array. We aim to improve this, while following an otherwise similar approach - retaining the time complexity and the parallelization scheme.

<table>
<thead>
<tr>
<th>ABC</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>71</td>
</tr>
<tr>
<td>2</td>
<td>193</td>
</tr>
<tr>
<td>3</td>
<td>251</td>
</tr>
<tr>
<td>4</td>
<td>103563</td>
</tr>
</tbody>
</table>

Table 1. A Prime Box by Pandey et al.

Implementation

Our approach is to change the "Prime Box" indexing scheme proposed by Pandey et al. Our solution, a Tightly Spaced Indexing Scheme (TSI) is equivalent to a multi-dimensional lookup table, where every letter supplies an additional dimension. This lets go of multiplication of prime numbers and using additions as the top-level operation, achieving a continuous value space for indices.

TSI, similarly to the "Prime Box", can be represented as a table (Table 2), with columns corresponding to letters in the alphabet and rows corresponding to word indices. The first value of a level is the last value of the previous level, initially 1. Every other value is an increment by the first value at this level. For example, index of "TIB" would be 20 + 26 + 2 * 26 = 1398.

<table>
<thead>
<tr>
<th>ABC</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>26</td>
</tr>
<tr>
<td>3</td>
<td>52</td>
</tr>
<tr>
<td>4</td>
<td>1398</td>
</tr>
</tbody>
</table>

Table 2. TSI indexed as a table

TSI can further be derived to an equation that gives the value of a letter at a specific level (Equation 1). Doing so, a predefined table needs to exist, which is also an improvement over the "Prime Box Parallel Search". To acquire the index of a word, the individual values for each letter have to be summed together.

\[ \text{value} = \text{letter value} \times \text{26}^{(\text{level} - 1)} \]  \hspace{1cm} (Eq. 1)

Results

The total memory consumption depends on the longest word required and the alphabet used. The memory consumption can be calculated with Equation 2, where m is the length of the longest word and w is the length of the alphabet. Figures 1 and 2 visualize the memory consumption differences between Prime Box and TSI algorithms when different alphabet or word sizes are used.

\[ \sum_{i=1}^{m} w^i \]  \hspace{1cm} (Eq. 2)

Figure 1. Array size comparison of TSI and Prime box algorithms based on different alphabet sizes with fixed word length of 5.

Figure 2. Array size comparison of TSI and Prime box algorithms based on different word lengths with fixed alphabet size of 26 letters.

TSI has space complexity of \(O(p^w)\) where p is the size of the alphabet and w is the maximum word length, while Prime Box has \(O(p^w)\), where p is the maximum prime. The huge difference in apparent. Using a 70-bit index (approx 256MB) with a 10x array, TSI can index all words of length up to 10, compared to 5 with Prime Box.

From a tight index space, it is not possible to improve the space complexity further without compromising in time complexity. However, simulating "hitting" using modulo could be used to trade actual space usage with performance in equivalent complexity bounds.

Conclusions

Proposed TSI assigns a memory location for every possible letter combination, which still seems impractical, but better than "Prime Box Parallel Search", while maintaining the same time complexity. The steps of the proposed algorithm also can be parallelized equivalently.

Further work could find better uses for a TSI-based search. Perhaps by not limiting the alphabet to letters, but places, data density could further be increased. There might also be ways where every index has a differently-sized alphabet. Further, TSI could be used to achieve better than AES256 encoding.
Exam (obligatory, minimum 50%)

• Will be based on questions similar to the homework assignments
  – Last two years we performed home exam, 1+ week to solve problems like in HW

• Knowledge of the basic principles of algorithms; broad understanding of the course materials.
<table>
<thead>
<tr>
<th>Activity</th>
<th>Lectures</th>
<th>Practice sessions</th>
<th>Homeworks</th>
<th>Essay</th>
<th>Project</th>
<th>Exam preparation</th>
<th>Exam</th>
<th>Total</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>22</td>
<td>12</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>1.5</td>
<td>0.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>36</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>40</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>6</strong></td>
<td><strong>26</strong></td>
<td><strong>155</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>156</strong></td>
</tr>
</tbody>
</table>

**EAP**
• All deadlines – strict

• Plagiarism – not tolerated (will lead to exmatriculation quickly)
  – Any material used should be referenced & cited properly
  – Develop your solutions, your opinions, etc.
  – Study group work should be finalised privately
Questionnaire (tbd)

• To assess the basic starting point and expectations before the course start

• Please fill in the form to the best of your ability *as is* during the next 15-20 minutes.

• Lectures on “Courses” Algorithmics page