Master's thesis

Master's Programme in Computer Science

Implementation and benchmarking of Ukkonen 1990 -algorithm

Arttu Kilpinen

February 7, 2022

Faculty of Science

University of Helsinki

Supervisor(s)

Assoc Prof. Simon Puglisi

Examiner(s)

Prof. Dunno yet

Contact information

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki,Finland

Email address: info@cs.helsinki.fi
URL: http://www.cs.helsinki.fi/

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Study programme
Faculty of Science	Master's Programme in Computer Science

Tekijä — Författare — Author

Arttu Kilpinen

Työn nimi — Arbetets titel — Title

Implementation and benchmarking of Ukkonen 1990 -algorithm

Ohjaajat — Handledare — Supervisors

Assoc Prof. Simon Puglisi

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's thesis	February 7, 2022	11 pages

Tiivistelmä — Referat — Abstract

Abstract here. Last thing to write

ACM Computing Classification System (CCS)
Theory of Computation → Design and Analysis of Algorithms → Data Structures Design and Analysis → Data Compression
Theory of Computation → Design and Analysis of Algorithms → Data Structures Design and Analysis → Pattern Matching
Theory of Computation → Design and Analysis of Algorithms → Data Structures Design and Analysis → Sorting and Searching

Avainsanat — Nyckelord — Keywords

Implementation, Shortest Common Superstring

Säilytyspaikka — Förvaringsställe — Where deposited

Helsinki University Library

Muita tietoja — övriga uppgifter — Additional information

Algorithms study track

Contents

1 Introduction **2**

1.1 Motivation 2

1.2 Related Work 2

1.3 Structure of the Thesis 2

2 Shortest Common Superstring **3**

3 Ukkonen's Algorithm **4**

3.1 AC-Machine 5

3.2 Pseudocode 5

3.3 correctness 5

4 Relative Lempel-Ziv **8**

5 experiments **9**

5.1 Implementation 9

5.2 Benchmark Data 9

5.3 Results 9

5.4 Discussion 9

6 Conclusions **10**

Bibliography **11**

Puglisin kommentteja: - Experiments section saattaa paisua aika paljon. Voisi mahdollisesti jakaa useaan kappaleeseen. Esim implementation kohtaa voisi laittaa myös ukkonen kappaleeseen.

Intro ja Conclusions luonnollisella kielellä. Muualla teknistä kamaa.

Introduction

vitusti viitteitä ja pelkkää LUONNOLLISTA kieltä. Ei esim määritelmiä!

Motivation

introssa motivaatiossa rlz mainittu mutta myöhemmin teknisesti

alanko or someone mentionet thet it would be interesting to see an implementation

Related Work

related workista oma kappale jos muita vertailuja kuin alanko ja norri. muuten ehkä introon.

Structure of the Thesis

Shortest Common Superstring

Tätä voi ehkä jakaa sectioneihin?

Definition 2.1 (Shortest Common Superstring)**.**

Definition of SCS here.

preliminaries including syntax

what is an approximation algorithm

Ukkonen's Algorithm

List of Algorithms

1 Aho and Corasic Algorithm 2, Construction of the goto function 6
2 Aho and Corasix Algorithm 3, Construction of the failure function 7

AC-Machine

hjallis

Pseudocode

Tänne bugikorjaukset sähköpostista joka lähetetty 30.4

Ja sitten failuren laskeminen

correctness

approx factor (ei välttämättä epsilon mutta jotkut boundit on olemassa ainakin kompressiolle)

Algorithm 1 Aho and Corasic Algorithm 2, Construction of the goto function

Input: Set of keywords $K = \{y_1, y_2..., y_k\}$.

Output: Goto function g

Method: We assume *output(s)* is empty when state s is first created, and $g(s, a) = fail$ if a is undefined or if $g(s, a)$ has not yet been defined. The procedure *enter(y)* inserts into the goto graph a path that spells out y.

```
1: function CALCULATEGOTOFUNCTION(K = {y_1, y_2..., y_k})
2:     newstate ← 0
3:     for i ← 1 until k do
4:         enter(y_i)
5:     end for
6:     for all a s.t. g(0, a) = fail do
7:         g(0, a) ← 0
8:     end for
9: end function
10: function ENTER(y = (a_1, a_2, ..., a_m))
11:     state ← 0
12:     j ← 1
13:     while g(state, a_j)! = fail do
14:         state ← g(state, a_j)
15:         j ← j + 1
16:     end while
17:     for p ← j until m do
18:         newstate ← newstate + 1
19:         g(state, a_p) ← newstate
20:         state ← newstate
21:     end for
22: end function
```

Algorithm 2 Aho and Corasix Algorithm 3, Construction of the failure function

Input: Goto function g from algorithm

Output: Failure function f

```
1: function CALCULATEFAILUREFUNCTION(g : ℕ → ℕ)
2:     queue ← empty
3:     for each a s.t. g(a, 0) = s ≠ 0 do
4:         queue ← queue ∪ {s}
5:         f(s) ← 0
6:     end for
7:     while queue ≠ empty do
8:     let r be the next state in queue
9:         queue ← queue\{r}
10:        for each a s.t. g(r, a) = s ≠ fail do
11:            queue ← queue ∪ {s}
12:            state ← g(r)
13:            while g(state, a) = fail do
14:                f(s) ← g(state, a)
15:            end while
16:        end for
17:    end while
18: end function
```

Relative Lempel-Ziv

Lempel-Ziv dictionary construction.

subsections?

experiments

Implementation

Benchmark Data

HW + instances

Results

Discussion

Conclusions

1. (Aho and Corasick, 1975) describes the Aho-Corasic machine for the first time. It gives the pseudocode to creation and search.

2. alanko dissertation, no bibtex yet. Discusses some things related to this topic.

3. (Alanko and Norri, 2017) describes approx scs algorithm for compact space.

4. statistics.pdf desccribes the dataset pizzachili.

5. (Ukkonen, 1990) is the most important reference in this thesis. Describes the main scs algorithm.

6. (Tarhio and Ukkonen, 1988) Describes the same algorithm as ukkonen 90 but not in linear time.

Bibliography

Aho, A. V. and Corasick, M. J. (1975). "Efficient String Matching: An Aid to Bibliographic Search". In: *Commun. ACM* 18.6, pp. 333–340. ISSN: 0001-0782. DOI: 10.1145/360825.360855. URL: https://doi.org/10.1145/360825.360855.

Alanko, J. and Norri, T. (2017). "Greedy Shortest Common Superstring Approximation in Compact Space". In: *String Processing and Information Retrieval - 24th International Symposium, SPIRE 2017, Palermo, Italy, September 26-29, 2017, Proceedings*. Ed. by G. Fici, M. Sciortino, and R. Venturini. Vol. 10508. Lecture Notes in Computer Science. Springer, pp. 1–13. DOI: 10.1007/978-3-319-67428-5_1. URL: https://doi.org/10.1007/978-3-319-67428-5%5C_1.

Tarhio, J. and Ukkonen, E. (1988). "A Greedy Approximation Algorithm for Constructing Shortest Common Superstrings". In: *Theor. Comput. Sci.* 57, pp. 131–145. DOI: 10.1016/0304-3975(88)90167-3. URL: https://doi.org/10.1016/0304-3975(88)90167-3.

Ukkonen, E. (1990). "A Linear-Time Algorithm for Finding Approximate Shortest Common Superstrings". In: *Algorithmica* 5.3, pp. 313–323. DOI: 10.1007/BF01840391. URL: https://doi.org/10.1007/BF01840391.