Review of PhD thesis

*Pattern matching in tree structures*

Author: Ing. Tomáš Flouri

The thesis is focused on algorithms on tree structures expressed in a linear notation. These algorithms use pushdown automata (PDA). The scientific contribution of the author is a development of methods for performing tree pattern matching, string indexing, and computation of repetitions using finite automata.

In a broader context, the author’s work contributes to so-called arbology, i.e. an algorithmic discipline studying problems on trees similar to problems as the stringology does on strings. The difference in both disciplines is that stringology uses rather finite automata that are not usable in linear representations of trees since these form context-free languages. Based on the fact that the deterministic PDA is an appropriate model for computation for a proper superset of the regular tree languages, it was possible to develop most results of the thesis.

The trees studied in the thesis are either ranked (each node knows the number of its children) or unranked. The former use in their linear notation either prefix of postfix one. For unranked trees so called bar notation is used (prefix or postfix).

As the core of thesis I consider Chapters 4-7. The explanation is conducted by formulation of 7 problems introduced and solved in particular chapters. In Chapter 4 tree pattern matching is studied. The author distinguishes between subtree and tree template matching and proposes a number of algorithms solving the following problems: (1) subtree matching for trees in prefix notation, (2) multiple subtree matching for trees in prefix notation, and (3) tree template matching in ranked trees using postfix notation. The chapter is completed by studying (4) tree template matching in unranked ordered trees in postfix bar notation.

In Chapter 5 new types of PDA are introduced. They serve for indexing ranked tree structures and suppose the prefix notation of the input tree. The indexing is meant as the problem (6) – to query effectively whether a given subtree exists in a given tree (similarly for a template), even all its occurrences. Computing and classifying all repeating subtrees within a tree is the topic of problem (5). The PDAs developed are analogous to suffix automata used in stringology.

Chapter 6 is focused on the last problem studied in the thesis - (7) computing all repeating subtrees in a given tree. Here also some experiments are mentioned. Yes, but with regard to non-existence any other experiments in the Thesis, the section 6.3. looks a bit strangely.

All algorithms are described in a usual approach (reasonable pseudo-code). Their correctness and complexities are proved. In interesting question is whether the algorithms have been also implemented and examined on real datasets. The results of the thesis have been published (or accepted for publication) in significant journals or proceedings of related conferences.

The theory and associated algorithms can find application in a number more technological disciplines, as e.g. querying XML data (see, e.g., twig pattern queries, etc.) based on tree template matching in unranked ordered trees.

*Some small flaws:*

The thesis uses a good formal apparatus introduced in Chapters 2-3. Interpretation of some definition could be discussed, e.g.

- in the dynamic set on p. 34, probably, $a_i$ are not attributes, but values. An attribute is intension, it has values and a name. Is $n$ fixed for all set elements? What is meant by the “key value”? It seems
that each dynamic set is created only by operations insert and delete starting from empty set. In opposite case the stack can not work. Moreover, a stack needs to “remember” the sequence of update operations, etc. Its implementation by an array of the length $n$ is natural, but the array is something more than a dynamic set. (By the way, the $n$ here is not the same $n$ as in the dynamic set). In linked lists, next and prev are attributes (they have values), but a?

- Procedure NEW-STACK is not specified on p. 47. Possibly, the basic operation empty() could be used.

Conclusion: The thesis demonstrates that the candidate achieved valuable scientific results. In accordance with §47, letter (4) of the Law Nr. 11/1998 (The Higher Education Act), I do recommend the thesis for the presentation and defense with the aim of receiving PhD degree.

Prague, March 18, 2013

Prof. RNDr. Jaroslav Pokorný, CSc.
Referee’s report on the PhD Thesis by Tomáš Flouri

Referee: Prof. RNDr. Alexander Meduna, CSc
PhD Thesis: Pattern matching in tree structures
Author: Tomáš Flouri
Supervisor: Doc. ing. Jan Janousek, PhD

Summary

The scope of the work is to establish the theoretical foundations of a new algorithmic discipline called Arbology, and provide a set of algorithms solving a number of problems that deal with tree structures. Arbology is considered to be a generalization of stringology, and transforms well-known principles and methods for string processing into the tree domain by linearizing a tree structure to a string and performing tree pattern matching in similar fashion as in the string domain. Due to the fact that trees are generated by context-free grammars, the pushdown automaton (PDA) seems to be an appropriate model of computation. This work presents some basic arbology results and principles. The first part of this work describes the methods for transforming a tree structure to the string domain in the form of four linear notations, which can be used to represent ordered, unordered, ranked and unranked trees as strings. Then, existing algorithms from stringology which are based on finite automata are redesigned and adapted to work with these linear notations. Notably, the search PDA is presented, which is analogically equivalent to the KMP-based finite automaton used in stringology. The search PDA can be used to search for a given tree pattern in a subject tree by preprocessing the pattern and then using this PDA to scan the subject tree and locate all occurrences of the pattern in time linear to the size of the subject tree. The result then is adapted to work with a set of given tree patterns in the same fashion as the Aho-Corasick automaton works for string patterns. Moreover, two algorithms which construct PDA for template matching in a) ranked and b) unranked tree structures are presented. The next part of the work deals with tree indexing in general. New data structures such as the subtree and tree pattern PDA are presented which serve as an index for a given tree in order to efficiently query it. Finally, the problem of computing all subtree repeats in ranked, ordered trees is considered. The solution presented is a bottom-up algorithm which locates and groups all subtree repeats in equivalence classes and which runs in time linear to the size of the subject tree.

The subject of this work and its significance

Trees are fundamental data structures in a number of both theoretical and applied fields such as computer science, mathematics, and evolutionary biology. This work introduces a theoretical overview and presents asymptotically optimal algorithms for extracting structural similarities such as repeats in tree structures. With the recent advances in computing and hardware in general, huge amounts of data are generated and are available for analysis and processing in emerging and promising fields such as evolutionary biology and computational molecular evolution. These vast amounts of data need faster algorithms for not only analyzing them, but also storing them in a compact and compressed form. Since many natural sciences use tree structures for representing data there will soon be a need for methods that efficiently store them, and therefore specific compression techniques will be in demand, which by definition, require algorithms for extracting repetitive substructures and regularities. Computing repetitive structures is also in great demand in sciences such as computational
Methods

The computational model used for this work is mainly the pushdown automaton. The PDA seems to be an obvious and natural model of computation for algorithms on trees because of the following facts. First of all, most tree processing algorithms use recursive procedures which rely on a pushdown store for their implementation. For example, finite tree automata are implemented by recursive procedures. Another point is that the presented linear notations of trees are actually context-free languages and pushdown automata are the corresponding models of computation for context-free languages. One of the main arbology results proves that the class of tree languages whose linear notation can be accepted by deterministic pushdown automata is a proper superclass of regular tree languages, which are recognized by finite tree automata. The last point is that the theory of finite automata has been applied with success in stringology --- an algorithmic discipline dealing with string processing. Therefore, since arbology deals with analogous problems in the tree domain, it seems natural that the pushdown automaton can serve as a generalization for the finite automata used in stringology. Moreover, it is possible to adapt existing finite automata based string algorithms on tree structures with very small modifications as described in the chapter where the search PDA is introduced.

Evaluation of the Achieved Results

The objectives of this dissertation thesis are achieved to the extent described in the abstract. This thesis presents tree pattern matching and indexing algorithms with the use of pushdown automata as the computational model as a generalization of string matching and indexing using finite automata. In addition, it presents the first linear time algorithm for computing subtree repeats in a given tree structure. While this result may be handy in theoretical areas such as mechanical theorem proving and term-rewriting systems, with the advent of next-generation sequencing technologies it is particularly useful in areas such as computational molecular evolution and phylogenetics, i.e. in the computation of tree likelihood in maximum-likelihood and bayesian methods. Moreover, some results, such as the computation of subtree repeats may be effectively used for designing specific tree-based compression algorithms.

Open problems

The last chapter of this work presents directions for future research and lists a number of open problems. A natural continuation of this field of study would be the adaptation of the presented algorithms to unordered trees which present a more generalized and natural version of tree structures. Another interesting, open-problem is the construction of a subtree PDA (the equivalent of DAWG) in time linear to the size of the input tree. Finally, another open-problem would be the inspection of various regularities in tree structures, such as the computation of covers. The analysis of massive data produced due to the rapid advances in applied fields such as evolutionary and molecular biology indicates that further elaboration on tree structures and the design of specific algorithms for tree data compression is another point for future research.
molecular evolution and phylogenetics where the extraction of repetitive substructures may save a lot of computational time in operations such as likelihood computation and model optimization.

Structure of the work

The central topic of this work is algorithms for performing pattern matching on tree structures. Before the presentation of the main topics, the work includes a preliminaries section where basic definitions from graph theory, formal languages and elementary data structures which are used throughout the text are presented and explained. The main results of the work are then presented. Chapter 3 presents four linear notations, i.e. transformations of trees to strings, and their properties are introduced and proved. Two of them --- prefix and postfix notation --- are used to represent rooted, ordered, ranked trees as a string and correspond to the pre-order and post-order traversal of the tree, respectively. The last two notations --- prefix and postfix bar notation --- are used to represent unranked, rooted, ordered trees. These linear notations are then used for representing trees throughout the rest of the text. For example, in chapter 4 they are used for constructing the so-called search PDA, a PDA version of the KMP-based finite automaton, which given a subject tree t and some tree pattern p, can find all occurrences of p in t, in time linear to the size of t. This automaton is then extended to work with a set of tree patterns, i.e. to find all occurrences of a set of tree patterns in the subject tree t which is analogous to the Aho-Corasick automaton used in string matching. Chapter 4 also considers template matching in ranked and unranked ordered trees and two algorithms are presented for constructing the respective tree template matchers using PDA. The time for template matching is linear with respect to the size of the input tree, and it is proved that there might exist cases (and those cases are listed) which require exponential amount of memory with respect to the size of the tree template given. Chapter 5 gives a brief overview of standard string indexing methods and presents two novel indexing structures that can be used for creating an index over a tree structure in order to efficiently perform queries in time linear to the size of the query. Finally, the last chapter presents a novel algorithm that can compute the subtree repeats in a given tree in time (and space) linear to its size.

Objectives

The main objective of this work is to present a way of generalizing finite automata based algorithms from the field of stringology in a way such that they may be applied on tree structures. Specifically, to develop methods based on the finite automata approaches in stringology for tree pattern matching which is analogous to string matching, and tree template matching which is analogous to string matching with wild-cards. Another objective is to devise a suffix automaton (or DAWG) based indexing data structure for trees that may allow one to efficiently perform queries on the indexed tree in optimal time. The last objective is to extract repetitive substructures in trees. The subtree and tree template matching are crucial since they represent the first linear-time algorithms on the size of the searched tree for this kind of matching, when the patterns to be searched are small. The other important result is the subtree repeats algorithm which may be particularly useful in area such as data compression and evolutionary biology.
Questions

Consider the open problem areas sketched above. Are there of some interest? Have you studied any of them? If not, do you plan to study them?

Conclusion

The PhD thesis has achieved the defined objectives, used appropriate methods and brought new knowledge, which is both significant in theory and applicable in practice. Therefore, this thesis satisfies all the Ph.D. requirements, and it is ready for its final defence.

February 8, 2013

Prof. RNDr. Alexander Meduna, CSc.
Department of Information Systems
Faculty of Information Technology
Brno University of Technology
2 April 2013

Prof. Dr. Jan Janousek
Faculty of IT
Czech Technical University in Prague
Czech Republic

Subject: Dissertation thesis (Ph.D) of Tomas Flouri

Dear Professor Janousek,

This letter constitutes my report on the Ph.D dissertation thesis (entitled “Pattern Matching in Trees”) by Mr. Tomas Flouri.

This thesis presents the new area of Arbology, along with some specific (and new) algorithms for classical problems in the area of tree pattern matching. The area of tree manipulation, pattern matching, etc. is a relatively mature one and has been studied since the early seventies, first in the context of instruction selection in compilers, and more recently in much larger trees representing XML (and other structured) documents. Flouri and his supervisors have been exceptionally active in these areas (at first mainly Prof. Melichar), especially more recently in the creation/definition of the field of arbology.

The last two decades have seen a heavy focus on tree automata (an extension of string automata) as the main computational formalism; this has been rather successful, with several books and dissertations (including one under my supervision). The Prague group has taken a different approach based on a very interesting (re)insight: trees can be linearized (structurally simplified/flattened into strings in several ways), seemingly simplifying the problem; simply applying normal string automata is far too weak and leaves us in a worse situation than just using plain tree automata; the solution indeed proves to be a kind of string automaton, in particular pushdown automata. The presence of the pushdown stack deals with the intrinsic structure of the trees, and indeed this also proves to be more powerful than plain tree automata.

This direction is not entirely new – it was initially explored (with some success in compiler code generators) by Graham and Glanville in the 1970’s, though their application orientation (and restriction to using slightly modified parser generators) also hindered progress in a more general theory. Enter arbology! The candidate, with Profs. Melichar and Janousek, have the background in automata over strings, as well as parsing in general, to bring those existing results to the land of trees. This work presents the following:

1. The key notions of arbology are introduced and formalized. This essentially includes an introduction to what most people should’ve covered in some formal language theory course.

---

1 Throughout this report, I omit the Czech accents, for which I apologize.
2. **Tree pattern matching.** Here, Flouri covers the classical pattern matching problem in a few different flavours.

3. **Tree indexing,** the extension to trees of algorithms known for strings.

4. **Computing repeats,** in which the classical problem/algorithms from strings are also extended. These generally find place in aspects of compression, or simply the combinatorial study of strings.

I have the following specific comments on this work, ranging from typographical to more global:

1. I note that Flouri starts indexing at 1 (e.g. in strings). Why not at zero?

2. “Pref” in mathmode appears to not be protected as \textit{Pref} or something similar.

3. Definition 10 gives “occurrence”, though why not just combine this with “factor”, since they’re the same?

4. Throughout the dissertation, I noticed many places where the typography would be a tiny bit nicer with using “math display mode” (using \[ and \] instead of $\$ $).

5. Similarly, I am a fan of fleqn.

6. Why is Remark 1 (page 29) important?

7. Definition 36 uses the xa form of \$\delta^*$ definition instead of the ax form. Is there a stylistic (or other) reason?

8. Flouri went to a lot of trouble with proofs, which are very much appreciated. A few notes about them:
   - a. Their typography could be better, in helping to make them stand out, much as the theorems, etc. do.
   - b. Some of the proofs could have been done in a more *calculational* style (cf. the book of Gries & Schneider). This is really a stylistic comment, though I think it could improve the readability.

9. Section 2.4, “We use the term...” seems to be missing some words.

10. Many of the stack manipulation algorithms (i.e. almost all of them, given this dissertation is heavily about PDA’s) involve direct stack manipulation. I would like to know from the candidate how (if at all?) these could have been simplified (i.e. not using lots of index manipulation, for TOS, etc.) in a recursive or functional programming context?

11. Definition 52 is somewhat unfortunately named, given the previous use of Pref and Suff.

13. Occasionally, I have seen an “En” dash where an “Em” dash would have been more appropriate.

14. Text before Lemma 9 on page 74: these space comments should be connected back to the classical string situation.

15. The “to appear” citations in 2012: have they now appeared?

16. Another very difficult stylistic comment: looking at Algorithm 32, I get the sense that there is something to be gained from notation design. These remain tougher to read than they need to be. I don’t intend this as a criticism of Flouri, but a more general call-for-change.

17. Section 7.0.1: this is a \subsection, where it should be a \section.

18. I thoroughly enjoyed the Conclusions.

19. An Index would have been incredibly useful.

20. My own previous work has often been very implementation oriented (in the form of toolkits, etc.). With such an elegant formalism (PDAs for trees), I find it unfortunate that this work did not include implementation aspects, such as toolkit construction or performance measurements.

21. The bibliography of this dissertation is essentially complete and well-structured. Similarly, the typesetting, internal structure and language use is outstanding – especially for someone without English-as-first-language.

On balance, I find this an excellent piece of work. Flouri has clearly mastered this field. His ongoing involvement in the Prague Stringology Group (if at a distance) is to be encouraged.

The author of the thesis proved to have an ability to perform research and to achieve scientific results. I do recommend the thesis for presentation with the aim of receiving the Degree of Ph.D.

Sincerely,

Prof. Dr. Dr. Bruce W. Watson
Full Professor of Information Science

---

Inligtingwetenskap
Sentrum vir Kennisdynamika en Besluitneming
Kamer 437, LSW
Stellenbosch Universiteit
Privaatsak X1
7602 Matieland, Suid-Afrika

T: 021-808-2423
F: 021-808-2117
E: nodysa@sun.ac.za
www.informatics.sun.ac.za

Information Science
Centre for Knowledge Dynamics and Decision-making
Room 437, LSW
Stellenbosch University
Private bag X1
7602 Matieland, South Africa