University of Victoria Computer Science

Csc586A Theoretical Models
Final Exam

Braden Simpson
braden@uvic.ca
V00685500

August 6, 2013

1 Question One

1.1 Part A

In this course, the theoretical model I chose to study most was for the project that Jordan and I did, and that model is the Abstract Syntax Tree (AST), more specifically, algorithms to find edit distances between them. The AST is a tree represents the syntactic structure of source code, where each node is a construct (while, return, if etc.), leaves are variables, and branches are blocks of code. See Appendix A for an example.

By using these trees and the algorithms for edit distance, outlined by the fluri et al. [1], we were able to implement an algorithm for finding the changes between two ASTs. I learned from the literature how to perform algorithms on these ASTs to find matching leaves, using levehnstein, n-grams, and other measures.

I also studied procedural generation and markov chains, but they weren't as in depth as the the trees. I found the procedural generation algorithms such as L-Systems and Noise generation to be particularly interesting, especially because of their real-world use in video games and simulations. Because the framework for L-Systems is so easy to learn, anybody can envision how they might be used in nature, which is sometimes hard to do when talking about theoretical models.

1.2 Part B

I used multiple different sources for my work, most of all the paper which helped us learn the algorithms required to perform tree edit distance calculations from fluri et

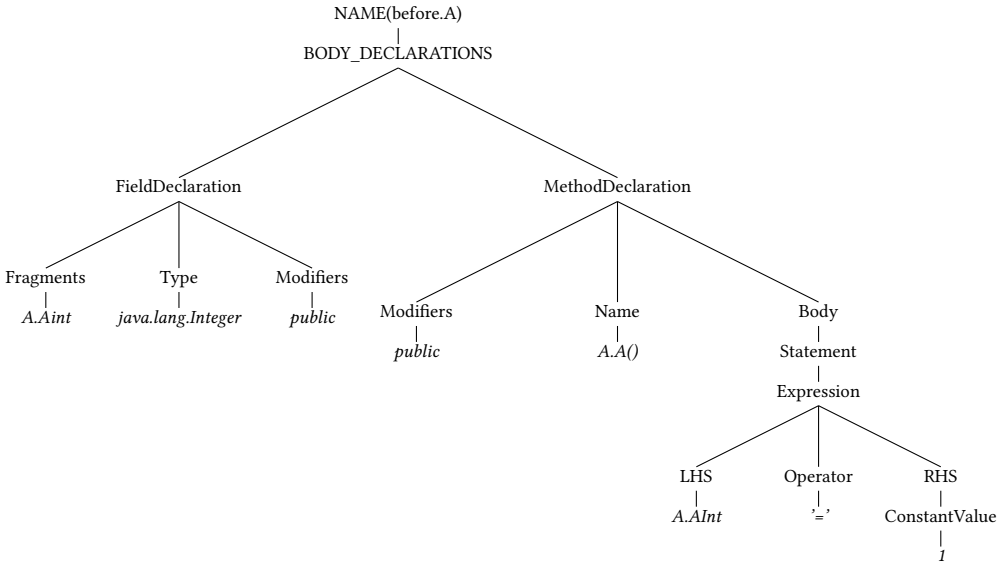
al. [1]. As well I used different research on MSR conferences to learn how to analyze and interpret the data correctly, even as I was there this year, seeing the different ways people inetepret datasets has given me the insight required to critically assess what the data means, what inferences we can get, and more importantly, what we cannot get.

1.3 Part C

For this section I will do an example run of our algorithm for finding the change types in two ASTs for a file (before and after). This will show how the algorithms are used to

```
1  package before;
2
3  public class A {
4      public Integer AInt = null;
5      public A() {
6          AInt = 1;
7      }
8  }
9
```

The code for class A before a change.



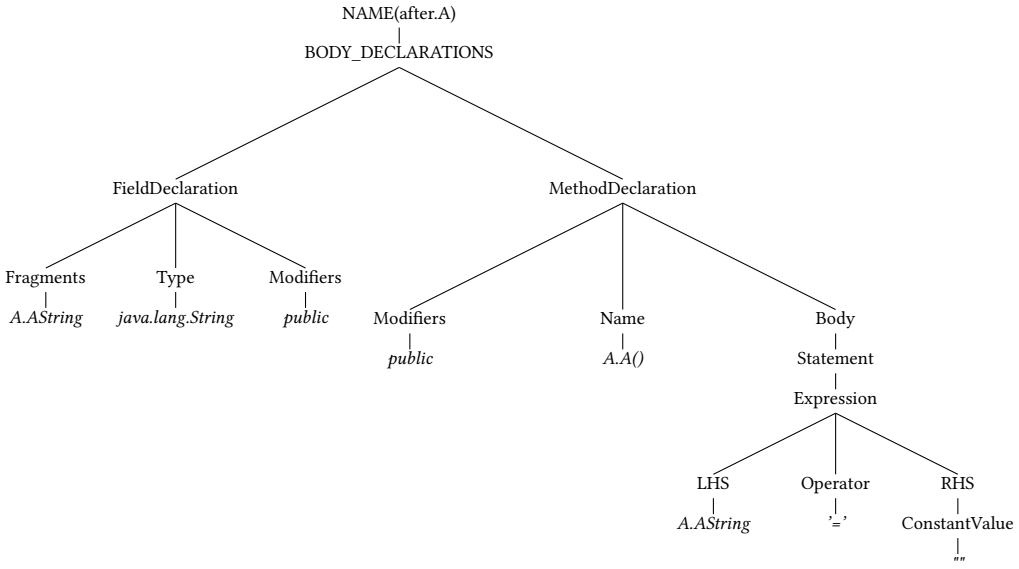
The AST that corresponds to Figure 1.3

```java
package after;

public class A {
    public String AString = null;
    public A() {
        AString = "";
    }
}

```

The code for class A after a change.



The AST that corresponds to Figure 1.3

Once we have the two ASTs, we use methods in [1] to match the nodes, using Levehnstein and n-gram similarity, to match the nodes. Then we apply our rules, to find an edit script, namely what we need to do to get $Tree_1$ to $Tree_2$. Using these rules:

INSERT $INST((l, v), y, k)$; Insert a new leaf node with label l and value v as the kth child of node y.

DELETE $DEL(x)$; Delete node x from it's parent.

ALIGNMENT $MOV(x, p(x), k)$; Node x becomes the *kth* child of *p(x).*

MOVE $MOV(x, y, k), p(x) \neq y$; Node x becomes the *kth* child of y, and is deleted from *p(x).*

UPDATE $UPD(x, val)$; Update *v(x)* with *val*, that is $val = v_{new}(x)$ and $v_{old}(x) \neq v_{new}(x)$.

We then result in the following changes.

1. DELETE - This is a delete into the method block of method *A.a()*. We then map this to our metric *PUBLIC_CHANGED_INTERNAL_METHODS*, since it's a change to the internals of a method.

2. INSERT - The first is an insert into the method block of method *A.a()*. We then map this to our metric *PUBLIC_CHANGED_INTERNAL_METHODS*, since it's a change to the internals of a method. The node was first deleted, and now it's inserted with a new value.

3. DELETE - Removed the object state *A.AInt*, causing a class field change, mapped to our metric *PUBLIC_CHANGED_CLASSES*.

4. INSERT - Added the object state *A.AString*, causing a class field change, mapped to our metric *PUBLIC_CHANGED_CLASSES*.

We then do this for every file, for each commit, to sum up how many times each metric is changed. We are then able to do our analysis on the data, interpreting it as a graph, in which we can make inferences about the trends of changes. All of this work can be seen at our website[1]

2 Question Two

I am choosing the following papers for this question: *A Density-Based Algorithm for Discovering Clusters in Large Spatial Dataases with Noise (DBSCAN)* [2], and *Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures* [3].

2.1 DBSCAN

This paper was a really good read and offered impressive results. The authors first outline the scope of the research, which is clustering algorithms for spacial data. The authors first talk about what the clustering algorithms are, and come up with a list of requirements for a good clustering algorithm.
In their abstract, the authors talk about the current best algorithm, and relate their proposed algorithm, DBSCAN against it, offering some instant validity. The requirements for their clusting algorithm are outline, and they immediately give motivation for DBSCAN by saying that no other algorithms satisfy these requirements. Furthermore, they say that DBSCAN is more efficient, robust, and functional, offering itself to a larger range of spatial databases.
Section 2 of their paper discusses in detail all the leading algorithms according to their requirements, offering more evaluation of related work. Then in Section 4, the authors introduce DBSCAN and finally in section 5 they use experimental data and benchmarks to validate their algorithm against the competitors in a simulation.

[1] http://beast.segal.uvic.ca:3000

[1] Note that for Section 1.3 I omitted a few nodes in the ASTs for simplicity, there are many that are generated by the JVM.

This paper compared the runtime of each algorithm, and did many experiments to give a more general case to show an approximation of the running times of DBSCAN. One of my main issues with the paper is that the authors compared DBSCAN against a different type of clustering algorithm (CLARANS), the two algorithms don't share any quantitative output, meaning that they have to be compared visually, which presents quite a large threat to the validity of the comparisons they do. One thing I would have liked to see from the authors woulb be a comparison with an algorithm of the same type as DBSCAN, to avoid this problem.
Finally, the authors setup future work by considering objects other than point objects (polygons?) and they want to do more work with high dimensional feature spaces. One thing they don't mention is comparison to any other related work or extra runtime evaluations.

2.2 Semantic distance in WordNet

This paper, by Budanitsky et al. presents definitions of *semantic similarity* and more generally, *relatedness* between words. The authors then do an thorough evaluation of five different techniques for determining similarity.
Firstly the authors introduce the concepts and describe their evaluation methods, which were as follows:

1. First a theoretical examination of a measure for desired properties (Wei 1993, Lin 1998) a good first coarse filter.

2. Human judgements on the similarity, very hard to do because of time constraints, and can be hard to get subject-independant results.

3. Thirdly, the authors use a Natural language parsing application to evaluate their measures, mixed with human evaluation.

The authors used previous work by Rubenstein and Goodenough(1965), as well as Miller and Charles(1991), which used human subjects to rank the similarity between a set of 65 pairs of words. The authors implemented the five measures, on the 65 pairs of words to compare how well their implementations were to the human judgement. The authors are critical of their method, saying that in the perfect world, they would have a really large human judged test set, but that is a very difficult task to accomplish. They then talk about methodological problems with the prior human studies.

> “It was implicit in the Rubenstein–Goodenough and Miller–Charles experiments that subjects were to use the dominant sense of the target words. But what we are really interested in is the relationship between the concepts for which the words are merely surrogates; the human judgments that we need are of the relatedness of word-senses, not words. So the experimental situation would need to set up contexts that bias the sense selection for each target word and yet don't bias the subject's judgment of their a priori relationship, an almost self-contradictory situation.
>
> *Budanitsky et al. [3]* ”

Next the authors evaluated the measures using the property of *malapropism* which is, spelling errors in open class words. The way the authors performed the tests was to do more simulations on 500 articles from the *Wall Street Journal*, with 107,233 unique words, 1408 of which were malapropisms.
The authors then did an retrospective analysis of their results in the conclusions which showed their critique of their own algorithms, and insights as to which algorithm is best, and why. They also talked about their limitations, and problems of the study.

3 Question 3

For question three, the two papers I am choosing as *optimization* papers are *Partial Parsing via Finite-State Cascades* [4], and

3.1 Partial Parsing

This paper was the first thing that came to my mind in terms of optimization. To begin, the paper is performing a task that is not novel, so the paper's contribution must be something which improves, sheds new light on, or modifies an existing method.
This paper introduces a new method of parsing through unstructured text such as human language. It does so by using a method of cascading finite state transducers which transform text based on some grammars. The authors evaluate these methods with an extensive set of language, with both german and english.
The reason why this paper *optimizes*, is because they have quantitative measures of how fast the traditional parsers can run (words/second), and then use that as their baseline for their new method. This gives them a goal to beat. The properties being optimized in this case are speed and accuracy, and the method these authors provide is measured for both. The speed is optimized and shown to be around 3000:1 increase in words/second. And the table below (taken from the paper [4]) shows the optimization in accuracy.

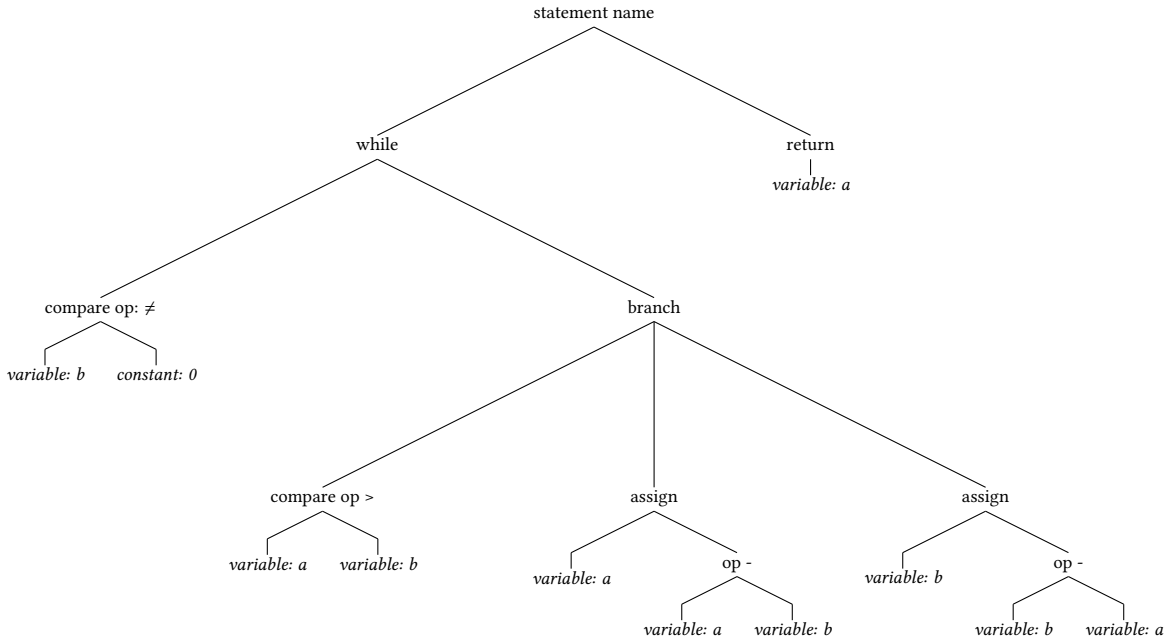
Table 2. *Evaluation of Parser Accuracy*

		cass2	marc
sample size	N	1000	
answers[a] in common	X	921	934
chunks in tst	t	390	381
chunks in std	s	394	
chunks in common	x	343	348
per-word accuracy	X/N	$92.1 \pm 1.7\%$[b]	$93.4 \pm 1.5\%$
precision	x/t	$87.9 \pm 3.2\%$	$91.3 \pm 2.8\%$
recall	x/s	$87.1 \pm 3.3\%$	$88.3 \pm 3.2\%$

The importance of both speed and accuracy in parsing of unstructured text is paramount, without quick, accurate parsers such as the ones proposed in this paper, and further studies, we would not have the technology like google, siri, many data-mining techniques, and much more.

3.2

A Abstract Syntax Trees



Data: An abstract syntax tree with matching psuedocode for Euclidean Algorithm. Taken from the wikipedia entry [a]

```
while b ≠ 0 do
    if a > b then
        a = a - b
    else
        b = b - a
    end
end
return a
```

[a] http://en.wikipedia.org/wiki/Abstract_syntax_tree

References

[1] B. fluri, M. Wuersch, M. PInzger, and H. Gall, "Change distilling: Tree differencing for fine-grained source code change extraction," *IEEE Trans. Softw. Eng.*, vol. 33, no. 11, pp. 725–743, Nov. 2007. [Online]. Available: http://dx.doi.org/10.1109/TSE.2007.70731

[2] M. Ester, H. peter Kriegel, J. S, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." AAAI Press, 1996, pp. 226–231.

[3] A. Budanitsky and G. Hirst, "Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures," in *IN WORKSHOP ON WORDNET AND OTHER LEXICAL RESOURCES, SECOND MEETING OF THE NORTH AMERICAN CHAPTER OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, 2001.

[4] S. Abney, "Partial parsing via finite-state cascades," *Nat. Lang. Eng.*, vol. 2, no. 4, pp. 337–344, Dec. 1996. [Online]. Available: http://dx.doi.org/10.1017/S1351324997001599