

David Keil
63.460 Theory of computing
Framingham State College
Spring 2010

Research paper

Each student will write a documented research paper of at least 600 words on a topic chosen by the student and approved by the instructor. Sources should be peer-reviewed or from edited periodicals, conference proceedings, or books. Include references within the text of your paper and a bibliography at the end in standard bibliographic format, including author name and publication information.

Research should be in some area covered by the course, with meaningful references to the course's textbook, sources, or handouts. Reference to the course material will be an important aspect of the research paper.

For information on formatting and documenting research see, for example, http://owl.english.purdue.edu/handouts/research/r_apa.html or http://webster.commnet.edu/apa/apa_intro.htm.)

The work will be done in three stages, each time with instructor feedback.

Proposal

Include:

- title (which like anything else in your proposal, you may change later);
- initial abstract: a paragraph or so about the topic, including some factual assertion about the topic and optionally a statement of what you hope to learn);
- short initial list of 2-3 sources in bibliographic format (see syllabus).

For links to papers with sample abstracts and sample bibliographic format, see www.framingham.edu/faculty/dkeil.

Preliminary draft

Draft, to be submitted in April, should respond to instructor comments on proposal and should be at least 300 words.

Final draft

Final draft should respond to instructor comments about preliminary draft and should be worthy of posting at instructor's web site.

Introductory assignment

A. Background and self introduction

At the Discussion Board, Forum “Introduction,” please comment in some way on your background or expectations for the course. (Questions are welcome to help fulfill this assignment!)

B. (Discrete Math review)

Prove by structural induction one of the following, depending on your student number for the course, 0 to 9:

1. For any full m -ary tree T , $|T| \bmod m = 1$. A *full m -ary tree* is one in which every non-leaf node has exactly m children.
2. For any graph, the sum of the connectivity numbers of the vertices is even.
3. For any graph, the number of vertices with odd connectivity numbers is even.
4. For any tree $T = (V, E)$, the $|V| = |E| + 1$.
5. The result of removing an edge from a tree is a non-tree.
6. The result of adding an edge to a tree is a non-tree.
7. The height of a complete binary tree with n vertices is $\log_2 n$.

Topic 1 assignment (Countability)

Each student will solve one of the following problems, corresponding to student classroom ID.

A. Diagonal proof

0. Let us consider a mobile robot that at each step of its existence, must decide whether to turn left or right 5 degrees, or go forward, based on its percept and state at that instant. Define a robot's *output behavior* as a set of infinite sequences of outputs in the set $\{left, forward, right\}$.

Use Cantor's diagonal proof method to show that the set of all possible behaviors is uncountable.

1. What is diagonal about "There are no true statements"?
2. A predicate on natural numbers, $p : \mathbf{N} \rightarrow \{0,1\}$ is a function that may be represented as an infinite sequence seq_p of values in $\{0,1\}$, where the j th bit in the sequence is the value of $p(j)$. For example, if p is the predicate *odd*, then the j th bit of the sequence seq_p is 1 if j is odd, 0 if j is even.

A Java program *computes* a certain predicate if, on input of some number x , the program outputs a 1 if $p(x) = 1$, and outputs a 0 when $p(x) = 0$. Any program can be described as a finite sequence of symbols, e.g., bits.

Use Cantor's proof method to show that some predicates are undecidable by any program; that is, for some predicate p there is no program that computes the function p .

Hint: Show that no bijection exists between the set of Java programs and the set of predicates.

3. Show that the power set of the set of strings over a finite alphabet Σ is uncountable (see proof in handout that the set of sets of natural numbers is uncountable).
4. In your own words, give Cantor's proof that rational numbers are countable.
5. Comment on the following: "Cantor constructed an enumeration of all natural numbers. He showed that a particular value in this list is not a real number. Cantor's proof is by induction."

6. Here is a diagram. Explain what it is used to show.

	1	2	3	4	5	6
1	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$	$b_{1,4}$	$b_{1,5}$	$b_{1,6}$
2	$b_{2,1}$	$b_{2,2}$	$b_{2,3}$	$b_{2,4}$	$b_{2,5}$	$b_{2,6}$
3	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$	$b_{3,4}$	$b_{3,5}$	$b_{3,6}$
4	$b_{4,1}$	$b_{4,2}$	$b_{4,3}$	$b_{4,4}$	$b_{4,5}$	$b_{4,6}$

7. Comment on the following: "The cardinality of the rational numbers is greater than that of the natural numbers, because between any two rational numbers there is another rational number."
8. What is diagonal about "It is self-evident that any claim not based on direct observation is meaningless"?
9. Answer the question, "If the haircutter is the person in the group who cuts everyone's hair except his or her own, then who cut's the haircutter's hair"? What is diagonal about your answer?
10. In your own words, what does assertion G in the proof of Godel's theorem say?

B. Coinduction

Consider the set of infinite sequences of inputs and outputs, when inputs are *pairs* of strings of symbols in the set DIGITS, and outputs are strings of DIGITS.

0. Formally define the set of input pairs.
 1. Formally define the set of output strings.
 2. How many input pairs exist?
 3. How many output strings?
 4. Formally define the set of *infinite sequences of* input pairs.
 5. Formally define the set of *infinite sequences of* input pairs and output strings.
 6. How many infinite sequences of input pairs exist?
 7. How many infinite sequences of input pairs and output strings exist?
 8. Formally define the set of *infinite* bit streams.
 9. Formally define the set of *infinite sequences of* pairs of bits.
 10. Formally define the set of *infinite sequences of* symbols from alphabet Σ .

Topic 2 assignment (DFAs and RLs)

Each student will solve one problem in each set A to E. The problem matches the student's one-digit section ID. Post solutions as replies to problem post or on paper. Subject line should specify problem #s. You may use pencil or JFLAP to draw DFA diagrams.

A. DFA construction

Construct a DFA that recognizes:

0. Inputs with an odd number of bits (accepts 01100, rejects 011101)
1. Inputs with exactly 2 '0's (accepts 0110, rejects 011001)
2. Inputs with no two consecutive 0s or 1s
3. Inputs with at least two consecutive 0s
4. Binary numerals divisible by 4 (accepts 01100, rejects 01101)
5. Unary numerals divisible by 4 (accepts 1111, rejects 111)
6. Integers (accepts 237, rejects x237)
7. Real numbers (accepts 1.23, rejects 1.2.3)
8. Java relational operators (accepts >=, >, rejects =<)
9. Strings in which all the 0's follow all the 1's
10. Strings that start with a '1' and have an even number of bits

B. Regular expressions

Write a regular expression for the language assigned for part A above.

C. NFA problems

Give NFAs with the specified number of states, recognizing each of the following languages. For each, describe the process for deriving the corresponding DFA by subset construction.

0. $\{ w \mid w \text{ ends with } 00 \}$ with 3 states
1. $\{ w \mid w \text{ contains the substring } 0101 \}$, with 5 states
2. $\{ w \mid w \text{ contains an even number of 0's or exactly two 1's} \}$, with 6 states
3. $0^*1^*0^*0$, with 3 states
4. $\{ w \mid w \text{'s second-last symbol is } 0 \}$
5. Strings containing both 00 and 11, or neither 00 nor 11 }

6. $(a|b)^*ba^*$
7. $(a|b)^*ba^*$
8. $(a|ab)^*ba^*$
9. $(ab|ba)^*a^*$
10. $(ab|ba)^*a^*$

Sources: (#1-4) Sipser, 1997; Goddard, 2008.

D. Pumping Lemma problems

Prove by the pumping lemma that the language among the following that corresponds to your student classroom ID is not regular:

0. The complement of $\{ 0^n 1^n \mid n \geq 0 \}$
 1. $\{ a^x b^y \mid x \neq y \}$
 2. $\{ w \mid w \text{ in } \{0,1\} \text{ is not a palindrome} \}$
 3. $\{ a^x b^{2y} \mid y = x \}$
 4. $\{ a^x b^y \mid y = x^2 \}$
 5. $u1u^R$ where u^R is the string u reversed
 6. numerals that represent the squares of whole numbers
 7. $\{ 0^m 1^n \mid m \geq n \}$
 8. $x0x$ where x is a string
 9. $\{0,1\}^m 0^m$
10. Inputs that start and end with 0 (accepts 01100, rejects 01101)

Source: Sipser, 1997.

E. Proof

Depending on your student number, write a regular expression, design a DFA that accepts exactly the following, and prove its correctness by induction on length of input.

Strings that:

1. have an even number of zeros
2. have an odd number of zeros
3. have an even number of ones
4. have an odd number of ones
5. have an even number of zeros
6. have exactly one 1
7. start with 0
8. end with 1
9. have exactly one zero
10. have not more than one 1 and are of odd length

DFA coding problems (extra credit)

You are to solve the (a) or (b) version of the problem described below.

Implement a lexical analyzer (tokenizer, scanner) in a higher-level language of your choice, based on a DFA M such that $L(M) = L(E)$, where $E =$

- (a) $(((' | ') | '+' | '-' | '*' | '/') | '>' | '<' | '=' | '>=' | '<=' | \{ '0', '1', \dots, '9' \}^*)^*$
- (b) $((\{ ' | ' \} | \text{keyword} | \text{identifier} | '=' | '.' | ';' | '(' | ') | \text{comment})^* \text{ where}$
identifier = (underscore | letter) (underscore | letter | digit)*
keyword = class | int | while | if
comment = $(/*)(\text{non-}^*)(*/)$

Thus, the regular language that M accepts should be the set of sequences of tokens used in arithmetic and relational expressions, with numeric literals, in languages like Java.

Note that $L(M)$ includes strings that are not *valid* arithmetic expressions, e.g., $>+2*-45)($. M should ignore white space.

Also, your program should output a linked list of token objects; each object should have two members: the string, e.g., $>=$ or 32 or $($), etc., and the identifier of its lexical category. Use the following lexical categories, left-paren, right-paren, addition operator, multiplication operator, relational operator, numeric literal.

Suggested approach

- The DFA's state will be represented by an integer variable. The state transition function is represented by a *switch* statement that assigns a new state value to the state variable depending on what the most recent input character is.
- The easiest way to develop and test is to read text files and display a "yes" or "no" result.
- In one of the states, a token has just been recognized. Make sure that when your DFA is in this state, a subprogram is called that adds an object to y
- +our linked list of token objects.

Topic 3 assignment (PDAs and CFLs)

Solve one problem under each problem set, corresponding to student ID number. Submit solutions on paper or post as replies to the assignment posting at Discussion Board Topic 3. Subject line should specify problem #s. You may use JFLAP.

A. CFG-CFL

Write a derivation (citing rule at each step) to show that the grammar specified in set braces generates the string indicated. Show parse tree.

- $\{S \rightarrow Xa \mid aX \mid SX, X \rightarrow ab \mid ba \mid \lambda \mid XX\}$, 'abaab'
- $\{S \rightarrow XY \mid S \rightarrow \lambda, X \rightarrow 0Y, X \rightarrow S1\}$, '00111'
- $\{S \rightarrow 00X, Y \rightarrow 1X1, Y \rightarrow \lambda, X \rightarrow Y0\}$, '001101010'
- $\{S \rightarrow SS \mid 0S1 \mid 1S0 \mid \lambda\}$, '001011'
- $\{S \rightarrow S2, S \rightarrow X, X \rightarrow 0X1, X \rightarrow \lambda\}$, '00112'
- $\{S \rightarrow 0S \mid S \rightarrow X \mid X \rightarrow 1X2, X \rightarrow \lambda\}$, '01122'
- $\{S \rightarrow X1, S \rightarrow 001, X \rightarrow 0, X \rightarrow X0\}$, '00001'
- $\{S \rightarrow 0S1S, S \rightarrow 1S0S, S \rightarrow \lambda\}$, '011001'
- $\{S \rightarrow X \mid X \rightarrow 0X, X \rightarrow \lambda, X \rightarrow Y, Y \rightarrow X1\}$, '0001'
- $\{S \rightarrow 0S, S \rightarrow X, X \rightarrow 0, X \rightarrow 11\}$, '00011'
- $\{S \rightarrow 0X1, X \rightarrow 0X1, X \rightarrow \lambda\}$, '000111'

B. Linear CFG

Define a *linear* CF grammar for

- $(0 \mid 10)^*$
- $(11 \mid 10) 1^*$
- $1^* \mid (01)^*$
- $1(0 \mid 01)^*$
- $01^* 0$
- $10(00)^* 01$
- $(11 \mid 01 \mid 10)^*$
- $(010)^* (101)^*$
- $0^* 11(10)^*$
- $1(0 \mid 1) 0^*$
- $(00 \mid 11)^*$

C. Define CFG

Define a CF grammar for

- $\{0^m 1^n 2^p \mid m = n \text{ or } n = p\}$
- $\{0^m 1^n 2^p \mid m = n + p\}$
- balanced parentheses (not only $((()))$ but also $()((()))$ etc)
- the set of Reverse Polish Notation expressions
- $0^n 1^{2n}$
- $\{x \mid n_0(x) = 2n_1(x)\}$
- Formulas in propositional logic (identifiers, \neg , \wedge , \vee , \Rightarrow , $'(, ')$)
- Set expressions (identifiers, $\{', '\}$, $;', \cap, \cup, \supseteq, \subset, \subseteq, \in$)
- Regular expressions over $\{0, 1, 2\}$ with operators for selection ($'|'$) and iteration ($'^*'$)
- $\{x \mid \text{length}(x) = 3n_1(x)\}$
- $\{x \mid n_0(x) = n_1(x)\}$ (equal numbers of 0's and 1's)
- $\{x0x^R \mid x \in \{0,1\}^*\}$

D. (Challenge) Parser problem

Write a top-down parser that accepts the language generated by the specified CF grammar, where all-caps expressions denote literal keywords and lower-case names denote lexical categories or nonterminals.

- $S \rightarrow$ class-defn S
 class-defn \rightarrow "class" identifier { decl-list }
 decl-list \rightarrow type-spec identifier () { stmt-list }
 type-spec \rightarrow "int" | "double"
- expr \rightarrow expression op simple-expression
 simple-expression \rightarrow num | (expr)
 op \rightarrow * | / | %
- $S \rightarrow$ term op simple-expression
 simple-expression \rightarrow num | (simple-expression)
 op \rightarrow + | -

E. (Challenge) RPN problem

- Define a PDA that accepts Reverse Polish Notation expressions.
- Is RPN a regular language? If so, write a regular expression for it; otherwise show that it is not regular

Topic 4 assignment (TMs)

Solve one problem under each problem set, corresponding to student ID number. Submit solutions on paper or post as replies to the assignment posting at Discussion Board Topic 4. Subject line should specify problem #s. You may use JFLAP for A.

A. TM construction

Write (as a diagram or table) the definition of the following Turing machine corresponding to your group number. Document by describing the purpose of each loop. If you use JFLAP, include as test results JFLAP traces of a few test inputs.

0. add a binary numeral to a unary numeral (tally, i.e., $11=2$, $111=3$, etc.), yielding binary output. Use '+' to separate the two parts of the input. For input $11+100$ ($3+4$), output should be 111 (7).
1. binary inverter
2. unary to binary converter
3. unary to unary adder
4. unary to unary subtracter
5. unary equality comparison (on input x_1, x_2 , if $x_1 = x_2$ output 1, else 0)
6. unary greater-than (on input x_1, x_2 , if $x_1 > x_2$ output 1, else 0)
7. logical OR for two 1-bit inputs (outputs 1 on 01, 10, or 11; 0 on 00)
8. logical AND for two 1-bit inputs
9. logical OR for multiple-bit input
10. logical AND for multiple-bit input
11. count 0's in input, display as unary numeral
12. compress bit string to eliminate 0's
13. string equality comparison (on input x_1, x_2 , if $x_1 = x_2$ output 1, else 0)
14. binary greater-than (on input x_1, x_2 , if $x_1 > x_2$ output 1, else 0)
15. search (find location of first 1 in a bit string, output its location in unary)
16. binary to unary converter

B. Theorems, definitions, and proofs

For the numbered item corresponding to your classroom ID:

- (a) Formally define the class described and state the theorem formally.
 - (b) State the proof method you used.
 - (c) Present a proof in your own words. State as many parts of the proof as possible in the language of predicate logic.
 - (d) Give sources where appropriate.
0. The class of non-deterministic TMs is equivalent to the class of deterministic TMs
 1. The TMs accept a superset of the RLs.
 2. The set of strings that consist of the same string concatenated to itself is TM decidable.
 3. The set of strings that consist of a series of 0s, followed by the same number of 1s, followed by the same number of 2s, is TM decidable.
 4. The two-dimensional TM is equivalent to the standard TM.
 5. There exists a universal TM.
 6. The TM model is not equivalent to the linear-bounded automaton model.
 7. There exists a language that is undecidable but is recognized by a TM.
 8. The two-stack automaton is equivalent to the TM.
 9. The multi-head TM is equivalent to the TM.
 10. The two-tape TM model is equivalent to the standard TM model.

C. Decidability and undecidability

In the case of assertions, below state each assertion formally and give brief proof sketches or explanations. Formal proof is not necessary. Give sources where appropriate.

0. State the theorems by Cantor on the cardinality of reals, Gödel on incompleteness, and Turing on undecidability. What basic method do the proofs share?
1. Suppose it were proven that problem P_1 is undecidable. How might this fact be used to show that problem P_2 is undecidable?
2. The problem of whether a given TM halts on blank input is undecidable.
3. The problem of whether a given TM halts on all inputs is undecidable.
4. The problem of whether a given TM recognizes the language \emptyset is undecidable.
5. What TMs address semidecidable problems and how?
6. When both a language and its complement are recursively enumerable, the language is Turing decidable.
7. Reducibility can be used to show undecidability.
8. If the halting problem were decidable, then every recursively enumerable language would be decidable.
9. The problem of whether two given TMs accept the same language is undecidable.
10. All regular languages are TM decidable.

Topic 5 assignment (RAMs and μ recursion)

A. Programs in the \mathcal{S} language

0. Show by construction, with respect to the RAM model of computation, that if f and g are computable functions, and $(\forall x) (h(x) = g(f(x)))$, then h is computable

Write programs in the \mathcal{S} language (including macros defined in the source material) that do the following. Each group solves the problem corresponding to the group number. Post solutions as replies to this post. Note that output is considered to be the value of y at termination of program.

1. inputs x and outputs 1 if $x = 0$, otherwise outputs 0.
2. inputs x and outputs 1 if $x > 0$, otherwise outputs 0.
3. inputs (x_1, x_2) and outputs 1 if $x_1 \geq x_2$, otherwise outputs 0.
4. inputs (x_1, x_2) and outputs $\max\{0, (x_1 - x_2)\}$.
5. inputs (x_1, x_2) and outputs $(x_1 \cdot x_2)$ (multiplication).
6. inputs (x_1, x_2) and outputs $(x_1 \div x_2)$ (division).
7. inputs (x_1, x_2) and outputs $(x_1 \bmod x_2)$.

B. μ -recursive functions

1. Use the computability of addition to show that multiplication is μ -recursive.
2. Use the computability of decrementing to show that subtraction is μ -recursive.
3. Use the computability of multiplication to show that finding the smallest x such that $(x^3$ is odd) is μ -recursive.
4. Distinguish primitive recursion from μ -recursion.
5. What proof approach might show that TMs are equivalent to the \mathcal{S} language?
6. What proof approach might show that TMs are equivalent to the μ -recursive functions?
7. What statement in the \mathcal{S} language enables looping, and how? How is looping done in recursive function theory?
8. Explain the notion of *composition* in recursive function theory.

Topic 6 assignment (Interaction)

A. Models

For the terms below, (a) give a definition; (b) give examples of models; (c) distinguish from the other terms in the list; (d) describe the role of synchrony/asynchrony if it applies

0. concurrency
1. parallelism
2. distributed processing
3. sequential interaction
4. multi-stream interaction
5. algorithm execution
6. coordination

C. PTMs

Define a PTM that does the following and state why it is not simulated by any TM.

0. Odometer
1. Adder
2. Vending machine that accepts \$1 and \$2 amounts and outputs candy and drinks, respectively
3. Command line processor that performs file copy operations
4. Answering machine

Topic 7 assignment (Concurrency)

A. Parallel models

Define the following and relate to parallel or distributed computing:

0. Java threads
1. CRCW
2. SIMD
3. Cellular automata
4. Neural networks
5. Sensor networks
6. Mesh architecture
7. PRAM
8. Cache coherency