Discrete II: Theory of Computation
Fall 2014

Charles E. Hughes
School of Electrical Engineering and Computer Science
University of Central Florida

email: charles.e.hughes@knights.ucf.edu

Structure: TR 1330-1745 (1:30PM - 2:45PM), MSB-359; 28 class periods, each 75 minutes long.
Go To Week 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

Instructor: Charles Hughes; Harris Engineering Center 247C; 823-2762
Office Hours: TR 1515-1630 (3:15PM - 1:30PM)
GTA: Melanie Kaprocki; HEC-234 (Computer Vision Lab II); OH: M 1600-1730 (4:00PM - 5:30PM); F 1330-1500 (1:30PM - 3:00PM)
Text: Sipser, *Introduction to the Theory of Computation 2nd Ed.*, Cengage Learning, 2013**+**Notes
Secondary References: Hopcroft, Motwani and Ullman, *Introduction to Automata Theory, Languages and Computation 3rd Ed.*, Prentice Hall, 2006.
Prerequisites: COT 3100; COP3503

Web Pages:
Base URL: http://www.cs.ucf.edu/courses/cot4210/Fall2014
Notes URL: http://www.cs.ucf.edu/courses/cot4210/Fall2014/Notes/COT4210Notes.pdf

Assignments: 8 or so.

Exams: 2 midterms and a final.

Material: I will draw heavily from Chapters 0-7 of Sipser. Some material will also come from Hopcroft et al.. You are responsible for material discussed in notes and in in-class discussions. Not all of this is addressed in either of these texts. I highly recommend attending class, interacting with me and listening very carefully when I say a topic is important to me; hint, hint about exam questions ;-)

Important Dates: Exam#1 -- September 26; Withdraw Deadline -- October 27; Exam#2 -- October 31; Final -- Tuesday, December 9, 1300-1550 (1:00PM-3:50PM)

Evaluation (Tentative):
Mid Terms -- 100 points each
Final Exam -- 150 points

Assignments -- 100 points
Bonus -- 50 points added to your best exam
Total Available: 500
Grading will be A >= 90%, B+ >= 87%, B >= 80%, C+ >= 77%, C >= 70%, D >= 50%, F < 50%

Weeks#1: (8/19, 8/21) -- Notes pp. 2-27 (Chapter 0 of Sipser); Syllabus

1. **Introduction to computability: an historical perspective**
2. **Basic notions of automata theory, formal languages, computability and complexity**
3. **Chomsky Hierarchy (sets context for much of course)**
   Classes of languages
   Phrase Structured Languages = (Recursively) Enumerable = Turing Recognizable = Semi-Decidable; Membership in an re language is undecidable
   Decidable = Recursive: This class cannot be generated by any definable class of grammars
   Context Sensitive Languages (CSL) are generated by Context Sensitive Grammars (CSG or Type1) and regognized by Linear Bounded Automata (LBA); Membership in a CSL is decidable
   Context Free Languages (CFL) are generated by Context Free Grammars (CFG or Type2) and recognized by Non-Deterministic Pushdown Automata (NPDA)
   Deterministic (unambiguous) Context Free Grammars are an undecidable subset of CFG, but all deterministic CFLs are generated by LR(1) grammars (one token lookahead)
   Regular Languages are generated by Regular (Right Linear) Grammars (Type3) and recognized by Finite State Automata (FSA, DFA or NFA)
4. **Some history of the genesis of the theory of computation (Hilbert, Godel, Turing, Post, Kleene)**
5. **Languages (countable sets of strings); why there are problems that no program can address**
6. **Sets, sequences, relations and functions (review)**
7. **Computability: basic goals and definitions (solved, solvable, semi-decidable and complements)**
8. **How computability concepts relate to each other**
9. **Complexity theory: basic goals; big question is P=NP?**

Assignment #1

> **See page 47 of Notes**
> **Sample problems (of same genre) and solutions to make your life a bit easier**

Due: 8/28 by 1:30PM (Key)

Week#2: (8/26, 8/28) -- Notes pp. 28-70 (Chapters 0 and 1 of Sipser); Dr. Workman's Notes pp. 1-8

1. **Discussions about non-determinism versus determinism; universal versus existential quantification; discrete versus continuous solution spaces**
2. **Formal definition of a Determinitic Finite (State) Automaton (DFA)**
   A = (Finite State Set, Finite Input Alphabet, Transition Function, Start State, Final States), where transtion function maps (Current State, Current Input Symbol) to Next State
   Can represent Transition Function by State Transition Table or State Transtion Diagram

(preferred)

3. **Determinitic Finite (State) Automaton (DFA): what and why?**
   Sequential circuits; pattern matchers; lexical analyzers; simple game/simulation behaviors
   Formal definition and examples
   State transition diagrams versus state transition tables
   Simple closure (union, intersection, difference, exclusive or, negation)
4. **Non-determinism (NFA)**
   Closure under concatenation, Kleene *
   Formal definition and examples

Assignment #2

See page 59 of Notes
Sample problems (of same genre) and solutions to make your life a bit easier

Due: 9/4 by 1:30PM (Key)

Week#3: (9/2, 9/4) -- Notes pp. 71-79 (Chapter 1 of Sipser); Regular Equations; Samples; Dr. Workman's Notes pp. 8-57

1. **The epsilon (lambda) closure of a state of set of states**
2. **Equivalence of DFAs and NFAs (subset of all states construction)**
3. **Reachable states from some given state**
4. **Reaching states to some given state**
5. **Minimizing the states of a DFA (indistinguishable vs distinguishable states)**
6. **Minimization example using notion of distinguishable states**
7. **Regular Expressions (closure of primitive sets under union, concatenation and star)**
8. **Every language defined by a Regular Expression is accepted by an NFA**
9. **Generalized NFA (GNFA) -- Definition**
10. **Every language accepted by a DFA is defined by a Regular Expression (ripping states out)**
11. **Alternate approach through Rij(k) construction**
12. **Languages defined by Regular Equations (not in text) and NFAs without lambda transitions**
13. **Closure of Regular Languages under Reversal**

Assignment #3

See page 71 of Notes
Sample problems (of same genre) and solutions to make your life a bit easier

Due: 9/11 by 1:30PM (Key)

Week#4: (9/9, 9/11) -- Notes pp. 80-97 (Chapter 1 and 2 of Sipser); Samples; Dr. Workman's Notes pp. 17-57

1. **Reachable states from some given state**
2. **Reaching states to some given state**
3. **Closure of Regular Languages under Prefix, Postfix, Substring, Min and Max**
4. **Classic non-regular languages $\{0^n1^n \mid n \text{ is greater than or equal to } 0\}$**
5. **Pumping Lemma for Regular Languages**
6. **Proofs that certain languages are not regular using Pumping Lemma**
7. **Basic notion of grammars and languages generated by grammars**
8. **Chomsky hierarchy (Type 3 through Type 0 grammars and languages)**
9. **Regular (right linear, Type 3) grammars**

Assignment #4

See page 90 of Notes
Sample problems (of same genre) and solutions to make your life a bit easier

Due: 9/18 by 1:30PM (Key)

Week#5: (9/16, 9/18) -- Notes pp. 98-100 (Chapters 1 and 2 of Sipser); Samples; Dr. Workman's Notes pp. 41-57

1. **Every language generated by a regular grammar is regular**
2. **Every regular language is generated by a type 3 grammar**
3. **The right and left linear grammars generate equivalent classes of languages**
4. **Can extend regular grammars to include strings rather than single characters**
5. **Right-invariant equivalence relationships and Myhill-Nerode Theorem**
6. **Proofs that certain languages are not regular using Myhill-Nerode Theorem**
7. **Right-invariant equivalence relationships and Myhill-Nerode Theorem**
8. **Existence of minimal state machine for any Regular Language**
9. **More proofs that certain languages are not regular using Pumping Lemma and Myhill-Nerode**
10. **Topics and Promises for MidTerm # 1**
11. **Sample Exam -- Complete this for discussion on 9/23 (key)**

Week#6: (9/22 (Help Session), 9/23 (review), 9/25 (Midterm1)

1. **Help Session in ENG1-224 on 9/22 from 4:00PM to 5:30PM**
2. **Review and go over sample questions**
3. **MidTerm 1 (Chapters 0, 1; Notes pp. 1-97; Assignments 1-4; Regular Equations; Pumping Lemma; Dr. Workman's Notes, pp. 1-57 that were covered in class)**

Week#7: (9/30, 10/2) -- Notes pp. 101-118; Chapter 2 of Sipser; Dr. Workman's Notes pp. 34-63

1. **Grammars and closure properties (union, concatenation, Kleene *)**

2. **Transducers (automata with output); Mealy and Moore Models**
3. **Closure under homomorphism and substitution**
4. **Constructions using NFAs to show closure under homomorphism and substitution**
5. **Closure under quotient, prefix, substring and suffix, using substitution and intersection**
6. **Constructions using regular expressions and NFAs**
7. **Difficulty with direct proof and right linear grammars**
8. **Review Chomsky hierarchy**
9. **Notion of instantaneous description for automata and grammars**
10. **Notions of derivation and the language generated by a grammar**
11. **Derivations (leftmost/rightmost**
12. **Parsing (parse trees; top down/bottom up)**
13. **Notion of ambiguity (inherent versus due to a specific grammar)**
14. **Context free grammars (CFGs) and languages (CFLs)**
15. **Sample grammars: {a^n b^n}; {w w(reversed)}; {w | #a(w) = #b(w)}**

Assignment #5

TBD

Due: 10/9 by 1:30PM (Key) Comments

Week#8: (10/7, 10/9) -- Notes pp. 110-119; Chapter 2 of Sipser; Dr. Workman's Notes pp. 57-68

1. **Pumping Lemma for CFLs (no proof)**
2. **Closure properties for CFLs (union, concatenation, Kleene *)**
3. **Non-CFLs {a^n b^n c^n }, {ww | w is a string over Sigma* }**
4. **Non-closure (intersection and complement)**
    a. **Intersection of {a^n b^n c^m} and {a^m b^n c^n}**
    b. **Complement of {ww | w is a string over Sigma* } is a CFL**
5. **Parse trees, leftmost and rightmost derivations, and ambiguity**
6. **Chomsky Normal Form (CNF) and the algorithm to convert a CFG to a CNF**
    a. **Eliminate lambda rules and accommodate for nullable non-terminals**
    b. **Eliminate unit rules (chains of non-terminals)**
    c. **Eliminate useless non-terminals (no terminal string can be generated from them)**
    d. **Eliminate unreachable non-terminals (cannot get to them from S)**
    e. **On rhs's of length >1, replace each terminal with symbol that derives it directly**
    f. **Change rhs of length k, k>2, to two rules, one with rhs of length 2 and the other of length k-1**
7. **More on parse trees and relation of height to longest length of string produced**
8. **Midterm Exam#1 Key**

Week#9: (10/14, 10/16) -- Notes pp. 118-138; Chapter 2 of Sipser; Dr. Workman's Notes pp. 68-77, 86, 87

1. **Every CFL is frecognized by a PDA**
2. **Top-down vs bottom-up acceptance**

3. **PDA equivalence to CFG**
4. **Cocke-Kasami-Younger polynomial time CFG parser (details on pp. 86,87 of Dr. Workman's Notes)**
5. **Shorthand notation for PDA**
6. **Proof of Pumping Lemma for CFLs**
7. **Pushdown automata (PDA) non-determinstic vs deterministic**
8. **Equivalence of a variety of PDA formalizations**
    a. **Bottom of stack marker versus none at start**
    b. **Ability to push none or one, versus many characters on stack**
    c. **Limitation of pop or push as only stack moves**
    d. **Recognition by accepting state, by empty stack and by both**
9. **Complete PDA equivalence to CFG by showing every language accepted by PDA is a CFL**
10. **Closure properties for CFLs (intersection with regular, substitution)**
11. **Using substitution and intersection with regular to get many more, e.g., prefix, suffix and quotient with regular**

Assignment #6 (also a prep for Exam#2)

TBD

Due: 10/23 by 1:30PM (Key)

Week#10: (10/21, 10/23) -- Notes pp. 139-164; Chapter 2, 3 of Sipser); Dr. Workman's Notes pp. 78-85

1. **Non closure of CFLs under complement and intersection, min and max**
2. **Start PDA equivalence to CFG by showing every language accepted by PDA is a CFL**
3. **Complete PDA equivalence to CFG by showing every language accepted by PDA is a CFL**
4. **Greibach Normal Form**
5. **Sample Exam2 is Assignment#6 plus these supplementary questions**
6. **Supplementary questions (key)**
7. **Topics and Promises for MidTerm # 2**
8. **Turing Machines**
9. **Variants of Turing Machines**
10. **Turing Machines as enumerators/recognizers**
11. **Factor Replacement Systems (FRS) as simple models of computation**
12. **Petri Nets as a model of synchronization; order their transitions and they are Turing-enabled**
13. **The necessity of order with FRSs**
14. **Note: 10/27 is the Withdraw Deadline**

Week#11: Notes pp. 141-144; (10/28 (Review), 10/29 (Help Session), 10/30 (Midterm2))

1. **Review and go over topics and sample questions**
2. **Help Session in TBD on TBD from TBD to TBD**
3. **MidTerm 2**

Week#12: (11/4, 11/6) -- Notes pp. 185-216; Chapters 4, 5 of Sipser

1. **The Halting Problem -- classic undecidable but recognizable (semi-decidable, enumerable) problem**
2. **Diagonalization proof for Halting Problem**
3. **Reducibility as a technique to show undecidability**
4. **The set of Algorithms (TOTAL)**
5. **Total is not even re shown by diagonalization**
6. **Reducibility as a technique to show non-re-ness**
7. **Using reducibility to show properties of Zero = { f | for all x f(x) =0 }**
8. **Many-one reducibility and equivalence**
9. **Notion of (many-one) re-complete sets (HALT is one such set)**
10. **The sets K and K0 are equivalent and re-complete**
11. **STP predicate (STP(f,x,t) iff f(x) converges in t steps or fewer)**
12. **VALUE(f,x,t) = f(x) if STP(f,x,t), else 0**
13. **Finally, proof that re and semi-decidable are same**
14. **Proof that re and co-re iff recursive (decidable, solvable)**

Assignment #7

TBD

Due: 11/13 by 1:30PM (Key)

Week#13: (Veterans Day, 11/13) -- Notes pp. 217-241 Chapters 5 of Sipser

1. **Rice's Theorem (weak and strong forms)**
2. **Applying Rice's**
3. **Quantification as a tool to identify upper bound complexity**
4. **Undecidable problems in formal language theory**
5. **Post Correspondence Problem (PCP)**
6. **Applications of PCP (Undecidability of Ambiguity of CFLs and Non-emptiness of Intersection of CFLs)**
7. **Application of PCP to Undecidability of Non-emptiness of CSLs**
8. **Traces (computational histories)**
9. **Is L = Sigma*? Is L = L^2? Non-closure of L1/2 (both CFLs)**
10. **Midterm Exam#2 Key**

Assignment #8

TBD

Due: 11/25 by 1:30PM (Key)

Week#14: (11/18, 11/20) -- Notes pp. 257-271; Chapters, 5 and 7 of Sipser

1. **Introduction to Complexity Theory**
2. **Verifiers versus solvers**
3. **NP as verifiable in deterministic polynomial time**
4. **P as solvable in deterministic polynomial time**
5. **NP as solvable in non-deterministic polynomial time**
6. **Million dollar question: P = NP ?**
7. **Some NP problems that do not appear to be in P: SubsetSum, Partition**
8. **Concepts of NP-Complete and NP-Hard**
9. **Canonical NP-Complete problem: SAT (Satisfiability)**
10. **Construction that maps every problem solvable in non-deterministic polynomial time on TM to SAT**
11. **3SAT as a second NP-Complete problem**
12. **3SAT to SubsetSum reduction**
13. **SubsetSum equivalence (within a polynomial factor) to Partition**

Week#15: (11/25, Thanksgiving) -- Notes pp. 272-285; Chapter 7 of Sipser

1. **Scheduling on multiprocessor systems**
2. **NP-Hard**
3. **QSAT as an example of NP-Hard, possibly not NP, problem**
4. **co-NP**
5. **Final Exam Topics (Notes pp. 280-286)**
6. **Sample Final (Key)**

Review Sessions: TBD

Final Exam; Tuesday December 9; 1300 - 1550 (1:00PM - 3:50PM)