Midterm Topics

Formal Languages & Automata Theory 1

- Regular Languages: DFA, NFA as acceptors; Regular (right or left linear) Grammars as generators; regular expressions and regular equations as descriptors; For language L, Pumping Lemma for Regular Languages and Right Invariant Equivalence Relation R_L where
  $x\ R_L\ y \Leftrightarrow \forall z\ [xz \in L \Leftrightarrow yz \in L]$ as evidence L is NOT Regular. Uniqueness of minimal state DFA based on Myhill-Nerode.
- Context Free Languages: Deterministic versus Non-Deterministic PDAs; Bottom-Up versus Top-Down Parsing; Context Free Grammars including reduced grammars and Chomsky Normal Form; CKY Dynamic Programming $O(N^3)$ parsing
- Context Sensitive Languages: Linear Bounded Automata; Context Sensitive Grammars
- Phrase Structure (re) Languages: Turing Machines; Phrase Structured Grammars

Formal Languages & Automata Theory 2

- Closures and Decision Problems for Each Language Class
- Use of closure under substitution, homomorphism and intersection with Regular Template
- h(f(L) $\cap$ Regular Language) with homomorphism g often used in Regular Language
- I will not ask you to create any automata or grammars (except maybe for a PCP application) but I do expect you to be able to apply the Pumping Lemmas, Myhill-Nerode, CKY and to understand the closure and decision problem properties for Regular, CFL, CSL and PSL (re), and to be able to apply the method involving closure under substitution, homomorphism and intersection with Regular..

Computability 1

- Basic Notions of solved, solvable, unsolved, unsolvable, re, non-re
- Relations between rec, re, co-re, re-complete, non-re/non-co-re
- Proofs about relations, e.g., re & co-re => decidable;
  union of re and rec is re but can be rec; what about intersection, exclusive union, +, *, - ?
- Use of quantified decidable predicates to categorize complexity (be able to do these)
- Reduction (many-one)
- Rice's Theorem (including its proof)
- Applications of Rice's Theorem (be able to do these)
- Proof of re-completeness (re and known re-complete reduces to problem)

Computability 2

- Basic decidability results in formal grammars (know them)
- Post Correspondence Problem (details) – Note PCP over 0ne-letter is decidable
- Semi-Thue word problem to PCP (no details)
- PCP and context free grammars
- From any PCP instance, P, can specify CFGs, G1 and G2, such that $L(G1) \cap L(G2) \neq \emptyset$ iff P has a solution – note: $L(G1) \cap L(G2)$ can be a CSL
- Merging these together to new grammar G with start symbol S and rule $S \rightarrow S1 \mid S2$ where S1 is start symbol of G1 and S2 is start symbol og G2 we have that G is ambiguous iff P has a solution
- PCP and context sensitive grammars
  From any PCP instance, P, can specify CSG, G, such that
  $L(G) \neq \emptyset$ iff P has a solution; it is also the case that L(G) is infinite if so
  Note that this is second proof of undecidability of emptiness for CSG

Computability 3

- Trace languages (CSL) and complement of trace languages (CFL)

  $L = \Sigma^*$ for CFL, $L \neq \emptyset$ for CSL

  For CFL L, $L = L^2$?

- Partial Trace Languages (CFL) get every other pair right

  Given TM, M, can specify CFGs, G1 and G2, such that
  $L(G1) / L(G2) = L(M)$

  Given TM, M, can specify CFG, G, such that
  $\exists n>0\ L(G)^n = L(G)^{n+1}$ iff $M \in CT = \{M \mid M \text{ halts in constant time } \}$