COT6410 Topics for Final Exams

**Computability Theory**

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
Use of quantified decidable predicates to categorize complexity
Reduction (many-one); degrees of unsolvability (many-one)
Rice’s Theorem (including its proof)
Applications of Rice’s Theorem
Proof of re-completeness (re and known re-complete reduces to problem)
Basic decidability results in formal grammars
Trace languages (CSL) and complement of trace languages (CFL)
\[ L = \Sigma^* \text{ for CFL, } L \neq \emptyset \text{ for CSL} \]
For CFL \( L = L^2 \)?
For CFL \( L, \exists n L^n = L^{n+1} \)?
Post Correspondence Problem
Semi-Thue word problem to PCP
PCP and context free grammars
From any PCP instance, \( P \), can specify CFGs, \( G_1 \) and \( G_2 \), such that
\[ L(G_1) \cap L(G_2) \neq \emptyset \text{ iff } P \text{ has a solution} \]
Merging these together to new grammar \( G \) with start symbol \( S \) and rule
\( S \rightarrow S_1 | S_2 \) where \( S_1 \) is start symbol of \( G_1 \) and \( S_2 \) is start symbol of \( G_2 \) 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
\[ L(G) \neq \emptyset \text{ iff } P \text{ has a solution} \]; it is also the case that \( L(G) \) is infinite if so
Note that this is second proof of undecidability of emptiness for CSG
PSG
Given TM, \( M \), can specify PSG, \( G \), such that \( L(G) = L(M) \)
Every PSL is homomorphic image of a CSL
Closure of CSL’s under \( \lambda \)-free homomorphisms
Quotient
Given TM, \( M \), can specify CFGs, \( G_1 \) and \( G_2 \), such that \( L(G_1) / L(G_2) = L(M) \)
Complexity Theory

P, NP (verification vs non-det. Solution), co-NP, NP-Complete
Polynomial many-one versus polynomial Turing reductions
Problems I will focus on
  Polynomial-time bounded NDTM to SAT (basic idea)
  Polynomial step bounded Semi-Thue to Bounded Tiling
  SAT to 3-SAT; 3SAT to Independent Set problem (IS) for undirected graph
  3SAT to SubsetSum; SubsetSum to Partition
  Knapsack is limited to one bin and asks for best fit (usually with values & weights)
    SubsetSum optimization problem for \( \leq G \) when weight and value are same
  BinPacking allows multiple bins and optimizes number of bins of some fixed size
  Scheduling with fixed number \((p)\) of processors and no deadlines
    Goal is to finish all tasks as soon as possible
    This is an optimization version of a p-partition problem
Deadline scheduling
  BinPacking uses all items in list so list could be times of tasks leading to an Optimization problem to minimize the number of processors while obeying a deadline
Scheduling heuristics and anomalies
  Unit execution scheduling of tree/forest and of anti-tree/anti-forest
Hamiltonian circuit (cycle)
  Travelling Salesman adds distances (weights) and seeks circuit of distance \( \leq K \)
    Reduce HC to TSP by setting distances to 1 where links and to K+1 otherwise
    Optimization version looks for minimum distance circuit
Integer Linear Programming Feasibility
  Is there an assignment that satisfies the constraints?
    SAT3 and 0-1 case.
k-vertex cover, k-coloring (3-coloring),
  Optimization versions: min vertex cover; min coloring
Tiling the plane
Polynomial step bounded Semi-Thue to Bounded Tiling
Parallels and non-parallel to Recursive, RE, RE-Complete,
  Co-RE, Co-RE-Complete, RE-Hard (Turing versus many-one reductions)
NP-Easy, NP-Hard, NP-Equivalent
  NP-Equivalent Optimization Problems associate with K-Coloring (min coloring) and SubsetSum
    (max SubsetSum less than a Goal value) – reduction and proof
What about Deadline Scheduling that optimizes number of processors with a fixed deadline?
What about min vertices in Vertex Cover?
P \( \subseteq \) NP \( \subseteq \) PSPACE = NPSPACE \( \subseteq \) EXPTIME \( \subseteq \) NEXPTIME \( \subseteq \) EXPSPACE
  \( \not\subseteq \) 2-EXPTIME \( \not\subseteq \) 3-EXPTIME \( \not\subseteq \ldots \not\subseteq \) ELEMENTARY \( \not\subseteq \) PRF \( \not\subseteq \) REC
P \( \neq \) EXPTIME; At least one of these is true
  P \( \not\subseteq \) NP
  NP \( \not\subseteq \) PSPACE
  PSPACE \( \not\subseteq \) EXPTIME
NP \( \neq \) NEXPTIME
  Note that EXPTIME = NEXPTIME iff P=NP
  Note that k-EXPTIME \( \not\subseteq \) (k+1)-EXPTIME, k>0
PSPACE \( \not\neq \) EXPSPACE; At least one of these is true
  PSPACE \( \not\subseteq \) EXPTIME
  EXPTIME \( \not\subseteq \) EXPSPACE
ATM (Alternating Turing Machine)
  AP = PSPACE, where AP is solvable in polynomial time on an ATM
  QSAT is solvable by an alternating TM in polynomial time and polynomial space
  QSAT is PSPACE-Complete
  Petri net reachability is EXPSPACE-hard and requires 2-EXPTIME
  Presburger arithmetic is at least in 2-EXPTIME, at most in 3-EXPTIME, and can be solved by an ATM with n alternating quantifiers in doubly exponential time