Complexity Theory Final Exam Topics

Charles E. Hughes

COT6410 – Spring 2023 Notes

Formal Languages and Automata Theory

Recognition Hierarchy

- Recognition models
  - Finite State
    - DFA = NFA
      - Exponential explosion when go from DFA to NFA
    - Reg Expressions, Regular Equations
  - PDA (can be stateless)
    - DPDA < NPDA
  - Linear Bounded
    - DLBA(n^2) ⊇ NLBA(n) and may be better (Savitch)
      - arXiv battle has arisen on this one
    - LBAs closed under complement and so CSLs are
  - Turing Machine
    - DTM = NTM

Grammar Hierarchy

- Regular / Right Linear / Left Linear
  - Arden's Theorem: R = Q + RP, P does not contain lambda, R = QP*
  - MyHill-Nerode (Consequences: unique min state and alternative to PL)
- CFGs
  - Ambiguity (inherent versus just incidental to a grammar)
  - Reduced Grammars and CNF (implications not constructions)
  - $O(N^3)$ CFL parser based on CNF grammar – Dynamic Programming
  - Incorrect traces (related result on complement of ww)
- CSGs
  - Trace languages
- PSGs

Closure

- Closure and non-closure:
  - Pumping Lemmas (What they are; not their proofs or applications)
  - Meta approach with intersection with regular and subst.
  - Why does it fail on CSLs?
- Various operations on CSLs, CFLs and Regular Languages
  - Examples: Union, Intersection, Quotient, Complement, Prefix, Suffix, Substring
- Interactions
  - What happens when sets interact:
    - Can we get Regular, CFL non-Reg, CSL non-CFL, RE non-CSL?
- Decidable Problems and why they are decidable
  - Examples: Membership, Emptiness, S*

Computability

Model Properties

- Models of computation and required elements (divergence, ability to branch on absence/presence)
- Determinism vs non-determinism; why non-det is not always better
- Relationships between rec, re, co-re, re-complete, non-re/non-co-re
  - Proofs about relations
    - re & co-re iff rec; re iff semi-dec.;
    - inf. rec iff range of mono increasing total function
- Various operations on non-re/non-co-re, re and recursive sets (Examples Sum, Product)

Complexity of Undecidables

- Use of quantified decidable predicates to get upper bound on complexity
- Reduction (many-one); m-1 degrees of unsolvability
- Rice's Theorem (including its proof)
- Applications of Rice's Theorem; when does it fail?
- Proof of re-completeness (re and known re-complete reduces to problem)

Post Correspondence

- Semi-Thue word problem to PCP (No details, quick pathway)
  - Other rewrite models: Post Canonical, Thue, Post Normal, Tag
- 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
  - 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 of 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

Trace Languages

- Trace languages (CSL) and complement of trace languages (CFL)
- $L = \Sigma^*$ for CFL, $L \neq \emptyset$ for CSL
- Quotients
  - Given TM, M, specify CFGs, G1 and G2, such that L(G1) / L(G2) = L(M)
  - Consider terminal traces (even/odd; odd/even correctness)

Phrase Structured

- 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 λ–free homomorphisms

Constant Execution Time

- Notion of arbitrary starting point
  - Why is this re and not worse?
  - What is notion of an infinite rather than unbounded tape?
  - What is mortality and how does constant time TM relate to mortal TM?
- Finite Power of CFLs
- Reducing is $L = \Sigma^*$ to is $L = L^2$
  Remember start point is to check if $\Sigma \cup \{\lambda\}$
- Reducing traces that have a fixed maximum length to $\exists n\ L^n = L^{n+1}$
  Remember trick of a language with three parts (bad traces, pairs of configs, $\{\lambda\}$)

Factor Repl. with Residue

- Factor Replacement Systems with Residue
  - Use residue to check for non-divisibility, thereby avoiding need for determinism
    - $2x + 1 \rightarrow 6x + 4$
    - $2x \rightarrow x$
  - Collatz Conjecture is that starting at any positive integer this eventually reaches 1 and cycles there on $1 \rightarrow 4 \rightarrow 2 \rightarrow 1$

Complexity Theory

Basics

- P, NP (verification vs non-det. solution)
- Co-NP
- NP-Hard, co-NP-Hard
- NP-Complete, co-NP-Complete
- Polynomial many-one versus polynomial Turing reductions

Base Plus Easy NP-C

- Polynomial-time bounded NDTM to SAT (basic idea)
- SAT to 3-SAT
- 3SAT to SubsetSum; SubsetSum to Partition
- Weighted MaxCut and Partition
- Integer Linear Programming Feasibility
  - Is there an assignment that satisfies the constraints?
  - SAT (not necessarily 3SAT) and 0-1 case.
- 3SAT to Independent Set problem (IS) for undirected graph (clause gadgets and added links)

Graph Problems

- 3SAT to Independent Set problem (IS) for undirected graph (clause gadgets and added links)
- k-vertex cover, k-coloring (3-coloring),
  - Optimization versions: min vertex cover; min coloring
  - Gadgets for each

Hamiltonian Path/Circuit

- Hamiltonian circuit (cycle)
  - Gadget to show NP-Hard
- Traveling Salesman adds distances (weights); seeks circuit of distance $\leq$ K
- Reduce HC to TSP set K to |V| and distances to 1 where there are links and to K+1 otherwise
- Optimization version looks for minimum distance circuit
  - Best known classical algorithm is 1.999^n
  - Best known quantum algorithm is 1.728^n
- Finding Triangle Strips is NP-Complete
  - NP-Hard by reducing Hamiltonian Path to Triangle Strips

Scheduling

- 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
- Scheduling with partial ordering (dag)
- Unit execution scheduling of tree/forest and of anti-tree/anti-forest

Knapsack

- Knapsack is limited to one bin and asks for best fit (values & weights)
  - SubsetSum optimization problem for $\leq$ G when weight and value are same
- Knapsack 0-1 Problem
  Dynamic Programming (differing dimensions –
  n,W: $O(n*W)$ vs n: $O(2^n)$)
- BinPacking allows multiple bins and optimizes number of bins of some fixed size; this problem is strongly NP-Complete

Some Analogies

- Parallels and non-parallels 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 associated with SubsetSum (max SubsetSum less than a Goal value) –
    - reduction using power of two values
  - K-Coloring (min coloring) – binary search

2-SAT

- 2-SAT
  - Use of Implication Graph and SCC (Strongly Connected Components)
  - Positive Min-Ones 2-SAT and relation to VC (Vertex Cover)
  - NP-Equivalence based on VC

Big Picture

- Weakly versus Strongly NP-Hard/Complete
- $P \subseteq NP \subseteq PSPACE = NPSPACE \subseteq EXPTIME \subseteq NEXPTIME \subseteq EXPSPACE \not\subseteq 2\text{-}EXPTIME \not\subseteq 3\text{-}EXPTIME \not\subseteq \ldots \not\subseteq ELEMENTARY \not\subseteq PRF \not\subseteq REC$
- $P \neq EXPTIME$; At least one of these is true
  - $P \not\subseteq NP$; $NP \not\subseteq PSPACE$; $PSPACE \not\subseteq EXPTIME$
  - $NP \neq NEXPTIME$; at least one of thsee is true
  - $NP \not\subseteq PSPACE$
  - $PSPACE \not\subseteq EXPTIME$
  - $EXPTIME \not\subseteq NEXPTIME$
  - Note that EXPTIME = NEXPTIME iff P=NP
  - Note that $k\text{-}EXPTIME \not\subseteq (k+1)\text{-}EXPTIME$, $k>0$
- $PSPACE \neq EXPSPACE$; At least one of these is true
  - $PSPACE \not\subseteq EXPTIME$
  - $EXPTIME \not\subseteq EXPSPACE$

ATMs

- ATM (Alternating Turing Machine) – This is just concept stuff with no details except all paths operate in paralell
- 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 (Why?)
- 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

PSPACE

- Savitch's Theorem: NPSPACE(f(n)) ⊆ DPSPACE($f(n)^2$)
  - Uses extreme time-space tradeoff – we don't care about time, only space
  - Limit depth of recursion in search for path from start to ending configuration
  - Do this by a recursive binary search using all possible intermediaries
  - Bad for time but good for max level of recursion
  - Assume space is lg N (valid for retaining node number or SAT assignments)
  - Time for DFS (non-det or det) is O(N).
  - Space for non-det is lg N; for det is N lg N (why?)
  - With ignoring time can get $(\lg N)^2$ space. Shows poly growth.
  - Time is $O(N^{\lg N})$

Functional Problems

- FP, FNP, TFNP
- Constraints
  - Promise Problems
  - Example is 4-coloring (planar is Promise Set; rest are maybes)
  - Promise set of VALUE(f,x,t) when STP(f,x,t) is true
  - CLP(R)

Khot’s Conjecture

- Graph Coloring with pairwise constraints is NP-Hard even when we know there is a coloring that satisfies almost all constraints, and we just need a coloring that satisfies a small percentage
- if Khot’s conjecture is true and P $\neq$ NP, then NP-Hard problems not only require exponential time but also getting good, generally applicable, polynomial-time approximations is hard