Samples Solution Keys to Bonus Homework

7.26; 7.30; 7.31; 7.35; 7.38; 7.43.

Problem 7.26

a. If σ is an $\neq$-assignment and σ' is its negation, then for any clause c, $c\sigma$ is true because it contains a true literal under σ; $c\sigma'$ is also true because $c\sigma$ contains a false literal, which becomes true under σ'.
b. The reduction from 3SAT to $\neq$SAT takes polynomial time because each clause becomes two clauses through the reduction. Let ϕ be an instance of 3SAT and φ is the result of reduction from ϕ. If σ is a model of ϕ, i.e., $\phi\sigma$ = true, then $\sigma' = \sigma \cup \{ z_i = \neg(y_1\sigma \vee y_2\sigma) \mid 1 \leq i \leq m, c_i \text{ is } (y_1 \vee y_2 \vee y_3) \} \cup \{ b = \text{false} \}$ is an $\neq$-assignment for φ, where m is the number of clauses. On the other hand, if σ is an $\neq$-assignment for φ and $b\sigma$ = false (if not, we take the negation of σ), then σ must be a model of ϕ.
c. Since 3SAT is NP-complete and it's easy to see that $\neq$SAT is in NP, so $\neq$SAT is NP-complete because of b.

Problem 7.30.

To show that SET-SPLITTING is in NP, we guess a coloring of elements of S and check if every subset Ci contains both colors. To show SET-SPLITTING is NP-hard, we reduce $\neq$SAT to SET-SPLITTING: For any instance ϕ of $\neq$SAT which has n variables and m clauses, let S be the set of literals of and $C = \{ C_1, \ldots, C_m \mid C_i \text{ is the set of literals in clause } i \} \cup \{ \{ x_i, \neg x_i \} \mid 1 \leq i \leq n \}$. Then SET-SPLITTING has a solution iff ϕ has an $\neq$-assignment.

Problem 7.31.

It is easy to show the Exam Scheduling is in NP by guessing the triples (S_i, F_j, k), meaning student S_i takes exam F_j at time slot k, $1 \leq k \leq h$, and check that no time conflicts in poly-time. To show it's NP-hard, we reduce the graph coloring problem to the Exam Scheduling. Let b(G, k) e an instance of the graph coloring, where G=(V,E). We transform (G, k) into an instance (S, F, k) of the Exam Scheduling, where S = V, the set of exams, and $F = \{ \{x, y\} \mid (x, y) \in E \}$, that is, each edge becomes a student who will take two exams. Then G has a k-coloring of vertices iff all exams can be scheduled in k time slots without conflicts. The details of the correctness proof is omitted here.

Problem 7.35.

To show that is Dominating Set in NP, we guess a subset of vertices and check that it contains k vertices, and every vertex not in the subset is connected to a vertex in the subset. This takes linear time to check. To show Dominating Set is NP-hard, we reduce Vertex Cover to Dominating Set: For any instance (G, k) of Vertex Cover, where G=(V,E), let G' = (V', E'), where $V' = V \cup \{ z_{x,y} \mid (x, y) \in E \}$, $E' = E \cup \{ (x, z_{x,y}), (y, z_{x,y}) \mid (x, y) \in E \}$. Then G has a vertex cover of size k iff G' has a dominating set of size k. Its correctness proof goes as follows:

(a) If G has a vertex cover S of size k, then S is a dominating set of G', because for each edge (x, y) of E, either x or y is in S and that vertex dominates x, y, and $z_{x,y}$.

(b) Suppose G' has a dominating set T of size k. For each $z_{x,y}$ in T, we replace $z_{x,y}$. by either x, without destroying the property of dominating. After the above changes, T does not contain any $z_{x,y}$. Thus T is a vertex cover of G.

Problem 7.38.

If P=NP and ϕ is an instance of SAT, then we have a polynomial time algorithm A which returns true iff ϕ is satisfiable. We may use A to find a truth assignment σ such that $\sigma(\phi)$ = true. This is done by the following algorithm, which takes polynomial time:

Boolean findAssignment (ϕ)

// suppose ϕ contains n Boolean variables, $x_1, x_2, \ldots, x_n$.

0. $\sigma := \{ \}$
1. For i := 1 to n do {
2. $\quad \varphi := \phi[x_i \leftarrow \text{true}]$ // replace each occurrence of x_i by true in ϕ
3. $\quad$ If (A(ϕ) = true) { $\sigma := \sigma \cup \{ x_i \leftarrow \text{true} \}$; $\phi := \varphi$}
4. $\quad$ Else { $\sigma := \sigma \cup \{ x_i \leftarrow \text{false} \}$; $\phi := \phi[x_i \leftarrow \text{false}]$ }
5. }
6. Return σ;

Problem 7.43.

For any CNF ϕ which has m variables, $x_1, x_2, \ldots, x_m$, and c clauses, $C_1, C_2, \ldots, C_c$, we can construct an NFA N of cm+2 states. We have an initial state q_0, a single accepting state q_f, and for each clause C_j in ϕ, we dedicate m states to C_j: $q_{j,1}, q_{j,2}, \ldots, q_{j,m}$., with the transition that $\delta(q_0, \varepsilon) = \{ q_{j,1} \mid 1 \le j \le c \}$, and

- if x_i appears in C_j, $\delta(q_{j,i}, 0) = \{ q_{j,i+1} \}$, $\delta(q_{j,i}, 1) = \{ \}$;
- if $\neg x_i$ appears in C_j, $\delta(q_{j,i}, 0) = \{ \}$, $\delta(q_{j,i}, 1) = \{ q_{j,i+1} \}$;
- if neither x_i nor $\neg x_i$ appears in C_j, $\delta(q_{j,i}, 0) = \delta(q_{j,i}, 1) = \{ q_{j,i+1} \}$,

where $q_{j,m+1}$ represents q_f. Then any unsatisfying assignment σ of ϕ can be represented by a binary string $b_1b_2\ldots bm$, where $\sigma(x_i) = b_i$, which will be accepted by N and N accepts only such binary strings.

If NFAs can be minimized in polynomial time, we may solve SAT in polynomial time as follows: Let CNF ϕ have m variables. (1) Construct NFA N as described above; (2) Minimize N to N'. (3) If N' is an NFA of m+1 states which accept every binary string of length m, then claim "ϕ is unsatisfiable", else "ϕ is satisfiable". This is because if ϕ is unsatisfiable, then every assignment is an unsatisfying assignment and every binary string of length m will be accepted by N'. A minimal such NFA will only need m+1 states.