CECS 329, Exam 2 Spring 2025, Dr. Ebert

IMPORTANT: READ THE FOLLOWING DIRECTIONS. Directions,

- For each of LO's 4-8 and the Additional problem, write your solution using a **SINGLE SHEET OF PAPER ONLY (BOTH FRONT AND BACK)**. Write **NAME** and **PROBLEM NUMBER** on each sheet.
- For Unit-1 LO's it's OK to use the same sheet for two or more problems if there is sufficient space.

Unit-2 LO's (25 Points Each)

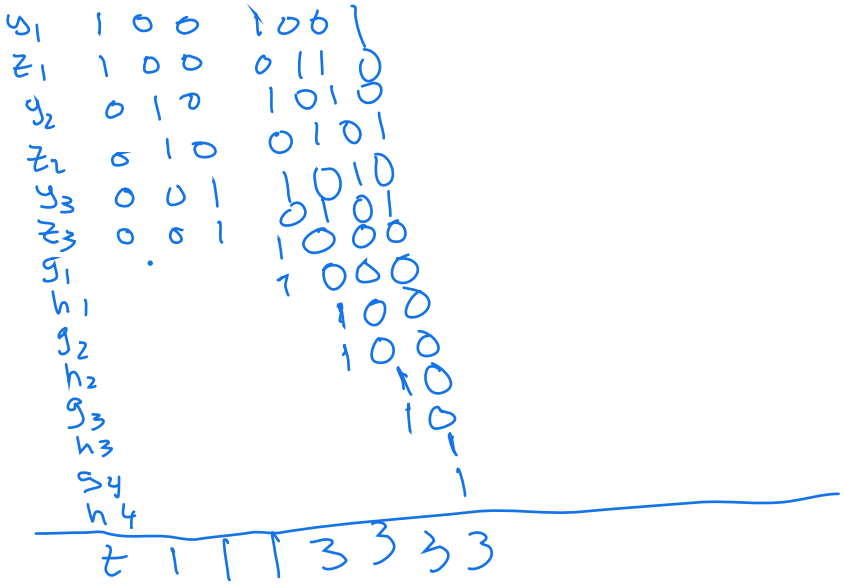
LO4. Do the following.

a. Consider the `3SAT` instance

$$\mathcal{C} = \{c_1 = (x_1, x_2, x_3), c_2 = (\overline{x}_1, \overline{x}_2, \overline{x}_3), c_3 = (\overline{x}_1, x_2, x_3), c_4 = (x_1, \overline{x}_2, \overline{x}_3)\}.$$

Consider the mapping reduction f : `3SAT` $\rightarrow$ `Subset Sum` from `3SAT` to `Subset Sum` that was presented in lecture.

i. Draw the complete table associated with $f(\mathcal{C}) = (S, t)$. (8 pts)
Solution.



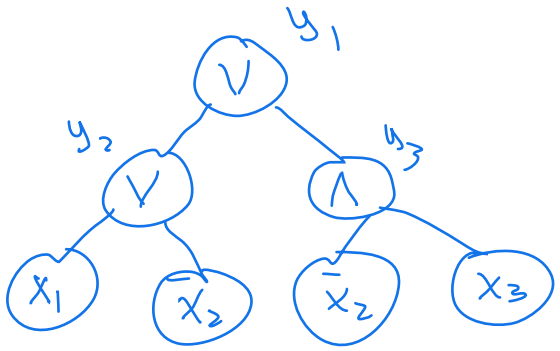
ii. Verify that $\mathcal{C}$ is satisfiable and use the satisfying assignment α to construct a corresponding subset A of S that sums to t. Please list the members of A using their alphabetical names (rather than their numerical values). (10 pts)
Solution. $\alpha = (x_1 = 1, x_2 = 0, x_1 = 1)$ satisfies $\mathcal{C}$. Verify that the members of A

$$A = \{y_1, z_2, y_3, g_1, g2, h_2, g_3, h_3, g_4\}$$

sum to $t = 1,113,333$.

b. Draw the parse tree for $F(x_1, x_2, x_3) = (x_1 \vee \overline{x}_2) \vee (\overline{x}_2 \wedge x_3)$, label the internal nodes with appropriate variables, and use them to provide the initial formula that begins the Tseytin transformation from F to an instance of `3SAT`. Do *not* simplify the formula. (7 pts)

Solution.

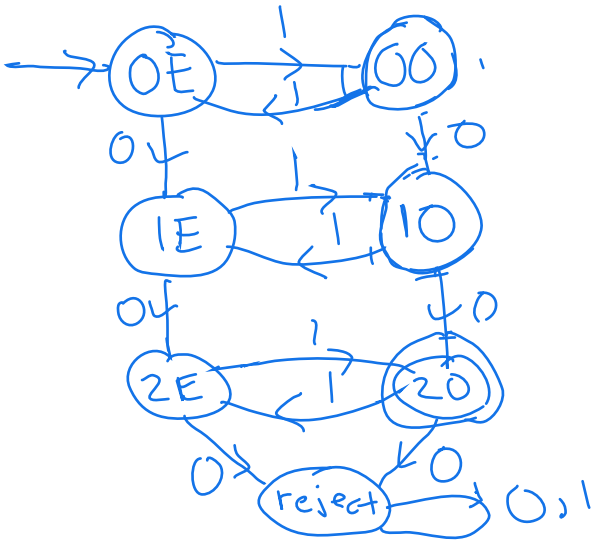


$$y_1 \wedge (y_1 \leftrightarrow y_2 \vee y_3) \wedge (y_2 \leftrightarrow (x_1 \vee \overline{x}_2)) \wedge (y_3 \leftrightarrow (\overline{x}_2 \wedge x_3))$$

LO5. Do the following.

a. Let L be the language of binary words that contain an odd number of 1's and exactly two 0's. Provide the state diagram for a DFA M that accepts L. (12 pts)

Solution.

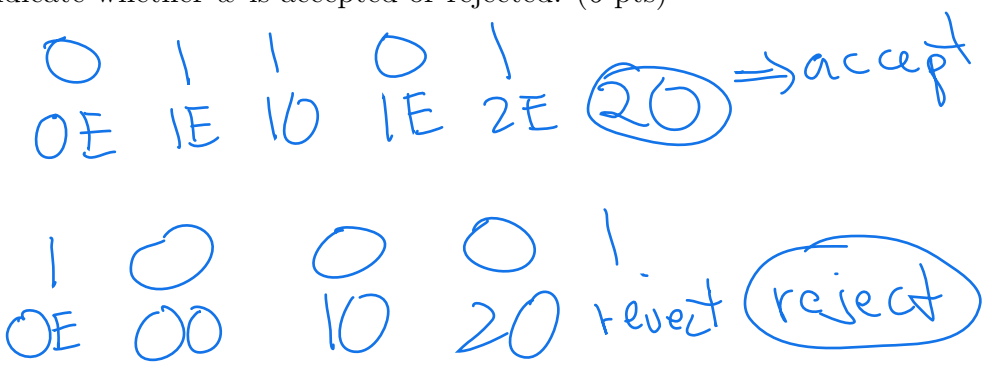


b. For each state s that appears in your DFA from part a, provide a sentence that describes what must be true about *any* word that, when read by M causes M to finish its computation in state s. (7 pts)

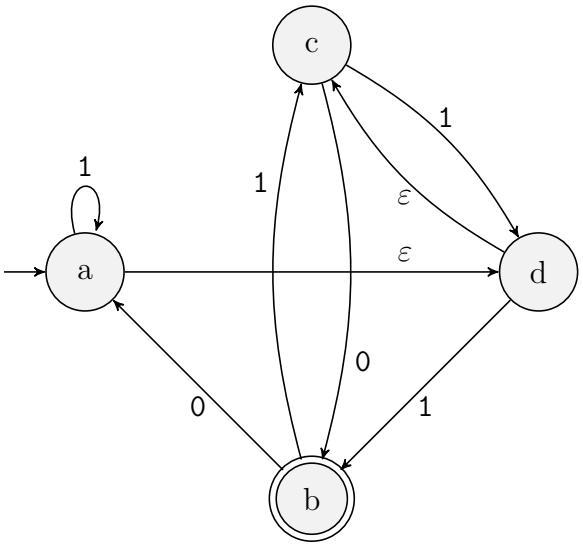
Solution. Let u denote the word that is associated with the sequence of input symbols that have thus far been read. When entering state `reject`, then u has three or more 0's. When entering ZE, $Z = 0, 1, 2$, then u has Z 0's and an even number of 1's. ZO is defined similarly, but now O stands for "odd".

c. Demonstrate the computation of M on inputs i) $w_1 = 01101$ and ii) $w_2 = 10001$. For each computation, indicate whether w is accepted or rejected. (6 pts)

Solution.



LO6. For the NFA N whose state diagram is shown below, provide a table that represents N's δ transition function.



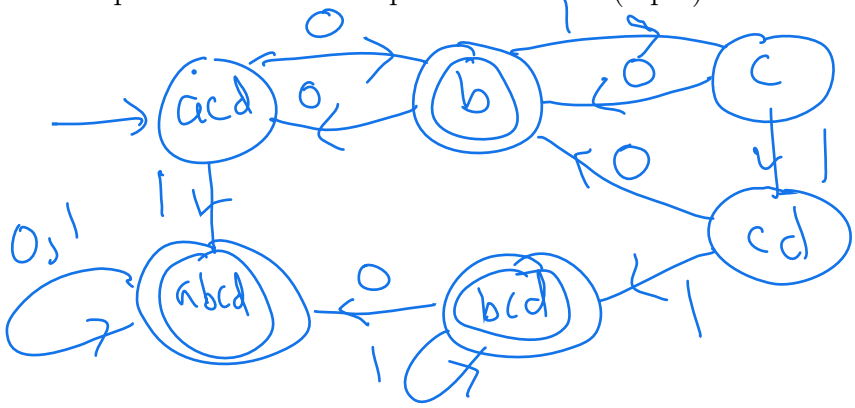
	0	1
a	∅	acd
b	acd	c
c	b	cd
d	∅	b

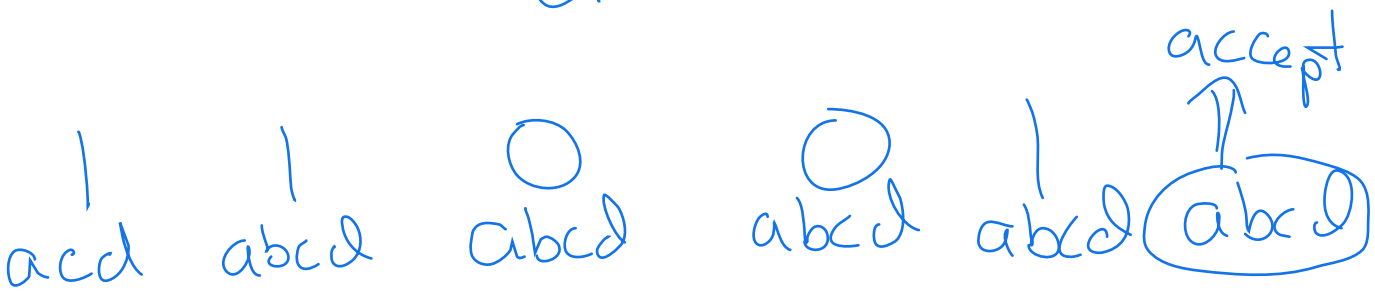
a. Provide a table that represents N's δ transition function. (12 pts)

b. Use the table from part a to convert N to an equivalent DFA M using the method of subset states. Draw M's state diagram. (8 pts)

c. Show the computation of M on input $w = 11001$. (5 pts)

Solution.





LO7. Do the following.

a. Let L_1 denote the language of binary words that contain exactly two 0's and an odd number of 1's and, L_2 denote the language consisting of all binary words w that have at least two 0's and at most two 1's. Use set notation to write the members of $L_1 \cap L_2$. (6 pts)

Solution. $L_1 \cap L_2 = \{100, 010, 001\}$.

b. Is 1000101010010 a member of L_1^*? Explain. (6 pts)

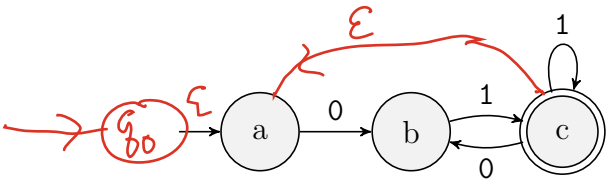
Solution. No, since, when parsing 1000101010010 as a concatenation of words in L_1, the first two words are forced to be 100 and 010 which makes it impossible to obtain a third word since the constraint of "exactly two 0's" forces the third word to be 1010 which has an even number of 0's.

c. Provide a regular expression that represents the language consisting of all binary words w for which the first bit does not equal the last bit, and there are exactly two 0's.

Solution. $1^+01^*0 \cup 01^*01^+$.

(13 pts)

LO8. Consider the NFA N shown below.



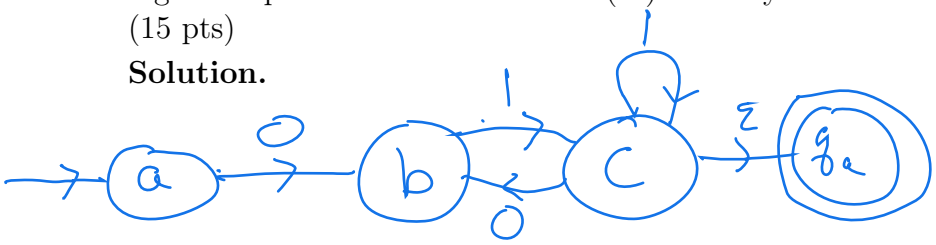
and let $L(N)$ denote the language that it recognizes.

a. Use N to construct the NFA N' that accepts $(L(N))^*$ and uses the algorithm that converts an NFA that accepts L, to one that accepts L^*. (10 pts)

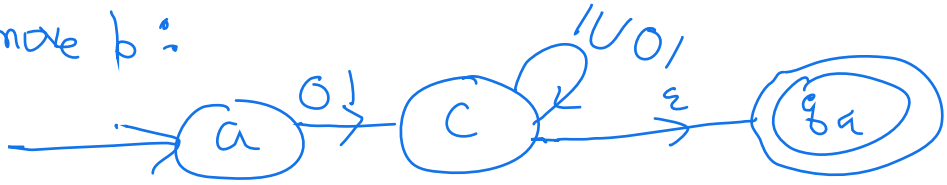
Solution. NFA N' is N with the additional new start state and edges (in red) shown above.

b. Demonstrate each step of the GNFA-to-Regular-Expression algorithm that computes a regular expression that describes $L(N)$. Hint: your initial GNFA should have four states. (15 pts)

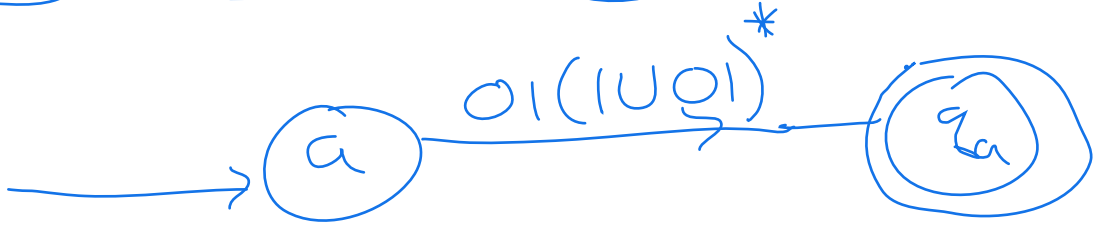
Solution.



Remove b:



Remove C:



N accepts the language described by 01(1U01)*

Additional Problem

Answer the following.

1. What does it mean for a decision problem to be NP-Complete? Provide the official definition. (10 pts)

   Solution. See lecture notes.

2. Provide the chain of mapping reductions that establishes the NP-completeness of the Traveling Salesperson decision problem. Hint: SAT $\leq_m^p$ 3SAT is the first link of the chain. (8 pts)

   Solution. See lecture notes.

3. Professor Jones has just discovered a polynomial-step algorithm for deciding any instance of the Independent Set decision problem. Explain why this proves that P = NP (12 pts)

   Solution. Let $L \in$ NP be given. Since IS is NP-complete (why?), we have $L \leq_m^p$ IS via some mapping reduction f. Let b be the size parameter for L and $p(b)$ the number of steps required to compute $f(x)$ for some instance x of L having size b. Then the size of $f(x)$ is at most $p(b)$ (here we assume that the output instance $f(x)$ of IS can have size at most $p(b)$ since at most one bit of memory can be created per algorithm step). Now let $q(m, n)$ be a polynomial that bounds the number of steps of Professor Jones' IS-algorithm, where m and n are the usual size parameters for IS. Then, the solution to instance x of L can be found by first computing $f(x)$ and then running Jones's IS-algorithm on instance $f(x)$. Finally, the bound on the number of steps required by this two-step process is $q(p(b), p(b))$ which is a polynomial in b since the composition of two polynomials is a polynomial. Therefore, L is in P and, since L was an arbitrary NP problem, it follows that P = NP.

Unit-1 LO's (0 Points Each)

LO1. Consider the 2SAT instance

$$\mathcal{C} = \{(\overline{x}_1, x_4), (x_1, \overline{x}_5), (\overline{x}_2, \overline{x}_3), (\overline{x}_2, x_4), (x_2, x_6), (x_3, x_4), (\overline{x}_3, x_6), (\overline{x}_4, \overline{x}_5), (\overline{x}_4, x_5)\}.$$

   (a) Draw the implication graph $G_{\mathcal{C}}$.

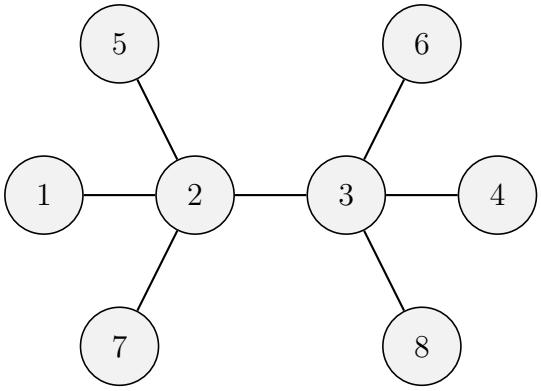
   (b) Perform the Improved 2SAT algorithm by computing the necessary reachability sets. Use numerical order (in terms of the variable index) and positive literal before negative literal when choosing the reachability set to compute next. Draw the resulting reduced 2SAT instance whenever a consistent reachability set is computed. Either provide a final satisfying assignment for $\mathcal{C}$ or indicate why $\mathcal{C}$ is unsatisfiable.

   (c) If an instance $\mathcal{C}$ has 336 variables and 2027 clauses, then what is the least number of queries that must be made to the Reachability oracle in order to confirm that $\mathcal{C}$ is satisfiable? Explain.

LO2. Answer the following.

(a) Provide the definition of what it means to be a mapping reduction from decision problem A to decision problem B.

(b) For the mapping reduction f : `Vertex Cover` $\rightarrow$ `Half Vertex Cover`, draw $f(G, k)$ for the `Vertex Cover` instance whose graph is shown below, and for which $k = 2$.



(c) Verify that both (G, k) and $f(G, k)$ are either both positive instances, or are both negative ones. Explain and show work.

LO3. An instance of `Feedback Arc Set (FAS)` is a directed graph $G = (V, E)$ and a natural number $k \geq 0$. The problem is to decide if there is a set S of k vertices of G for which, when removing the vertices of S from G (and all edges incident with them) the resulting graph is acyclic. To see that `FAS` is an `NP` problem we define a certificate for instance (G, k) to be a set $S \subseteq V$ of k vertices. The following pseudocode is used by the verifier to determine if S is in fact a set of vertices whose removal from G creates an acyclic graph. Note: a minimum of 18 points is needed to pass this LO.

$G' = (V', E')$, where $V' = V - S$ and $E' = \{(u, v) \in E \mid u, v \in V'\}$.

For each $u \in V'$,

  For each $v \in V'$,

    If $(u, v) \in E'$ and Reachable(G', v, u), then return 0.

    //v can reach u and so there is a cycle that includes v

Return 1.

(a) Provide size parameters for the `FAS` problem and describe what each represents in relation to an `FAS` problem instance. Hint: there are two of them. (6 pts)

(b) Use the size parameters to provide the big-O number of steps that is required by the verifier to check the validity of a certificate. Hint: a single call to function Reachable requires a linear number of steps with respect to the size parameters of G. Justify your answer. (7 pts)

(c) Classify each of the following problems as being in `P`, `NP`, or co-`NP` (3 points each).

i. An instance of `Fallible` is a Boolean formula F. The problem is to decide if there an assignment that can be made to the variables of F so that F evaluates to 0.

ii. An instance of `Mine Sweep` is a simple graph $G = (V, E, f)$ where $f : V \rightarrow \{-1, 0, 1, \ldots\}$ is a function from the set of vertices to the set of natural numbers, including -1. If $f(v) \geq 0$, then it means that a total of $f(v)$ neighbors of v (i.e., vertices that are

adjacent to v) must have a mine placed on them. On the other hand, if $f(v) = -1$, then there is no constraint on how many neighbors of v must have a mine. The problem is to decide if there is a function $g : V \to \{0, 1\}$, such that i) $g(v)$ indicates whether or not a mine is placed on vertex v, and ii) for all $v \in V$, if $f(v) \geq 0$, then

$$f(v) = \sum_{u \in N(v)} g(u),$$

where $N(v)$ is the set of all neighobrs of v. In other words, function g meets all the mine constraints that are indicated by f.

iii. An instance of `Palindrome` is an array a of integer, and the problem is to decide if a reads the same forwards as backwards, meaning that, for all $i \in \{0, 1, \ldots, n-1\}$, $a[i] = a[n-1-i]$.

iv. An instance of `Large Vertex Covers` is a graph $G = (V, E)$, and the problem is to decide if every vertex cover of G has size at least $n/2$, where $n = |V|$.