CS250 Fall 2020 Final *Solutions*

Q1 (35p). (a, 15p) Translate the statements into predicate logic. State what your predicates mean.
1. An exam is easier than another iff everyone who passed the second exam also passed the first, and there is someone who passed the first but not the second.
2. A exam that is not easier than some other exam is hard.
3. Someone is unsurpassed if noone passes an exam that they did not pass.
C. Someone who passes all hard exams is unsurpassed.

$P(x, y)$: x passes exam y. $E(x, y)$: exam x is easier than y. $H(x)$: exam x is hard. $U(x)$: x is unsurpassed.

1. $\forall e : \forall e_2 : E(e, e_2) \leftrightarrow (\forall x : P(x, e_2) \to P(x, e)) \land \exists y : P(y, e) \land \neg P(y, e_2))$
2. $\forall e : (\neg \exists e_2 : e_2 \neq e \land E(e, e_2)) \to H(e)$
3. $\forall x : (\neg \exists y : \exists e : (P(y, e) \land \neg P(x, e)) \to U(x)$ or $\forall x : (\forall y : \forall e : P(y, e) \to P(x, e)) \to U(x)$

C. $\forall x : (\forall e : H(e) \to P(x, e)) \to U(x)$

(b, 8p) Assume a finite number of people pass each exam. Write the following statement in logic, and prove it, assuming statements (1)-(3): "Any exam that is not hard is easier than some hard exam." You may reason in words, but be precise.

Let $S(e)$ be the set of people passing exam e. By contrapositive of (2), $\neg H(e) \to \exists e_2 : E(e, e_2)$, and $E(e, e_2) \to S(e_2) \subset S(e)$ (1). The implication $P(x, e_2) \to P(x, e)$ ensures $\subseteq$ and the second conjunct implies the inclusion is strict. Since $S(e)$ is finite, any chain of strict subsets will be finite (by the least number principle). Therefore, for any exam e, any chain of exams "harder" than e must stop at some e_0. By (2), e_0 is a hard exam. We write $\forall e : \neg H(e) \to \exists e_0 : E(e, e_0) \land H(e_0)$.

(c, 12p) If a finite number of people pass each exam, prove formally, showing deduction rules used, that statements (1)-(3) imply C, or show a counterexample. You may use the statement at (b).

Solution: We use proof by cases, for hard exams (premise of C), and those that aren't (b).

(4) $\forall e : H(e) \to P(x, e)$	Assumption (x arbitrary)
(5) $H(e) \to P(x, e)$	Specification (4, e arbitrary)
(6) $\forall e : \neg H(e) \to \exists e_0 : E(e, e_0) \land H(e_0)$	Statement (b)
(7) $\neg H(e) \to \exists e_0 : E(e, e_0) \land H(e_0)$	Specification (6)
(8) $\neg H(e)$	Assumption
(9) $\exists e_0 : E(e, e_0) \land H(e_0)$	Modus Ponens (7, 8)
(10) $E(e, f) \land H(f)$	Instantiation (9, f constant)
(11) $H(f)$	Separation (10)
(12) $H(f) \to P(x, f)$	Specification (4)
(13) $P(x, f)$	Modus Ponens (11, 12)
(14) $E(e, f) \leftrightarrow (\forall x : P(x, f) \to P(x, e)) \land \exists y : P(y, e) \land \neg P(y, f)$	Specification (1)
(15) $E(e, f) \to (\forall x : P(x, f) \to P(x, e)) \land \exists y : P(y, e) \land \neg P(y, f)$	Equivalence and Implication (14)
(16) $E(e, f)$	Separation (10)
(17) $(\forall x : P(x, f) \to P(x, e)) \land \exists y : P(y, e) \land \neg P(y, f)$	Modus Ponens (15, 16)
(18) $\forall x : P(x, f) \to P(x, e)$	Separation (17)
(19) $P(x, f) \to P(x, e)$	Specification (18)
(20) $P(x, e)$	Modus Ponens (13, 19)
(21) $\neg H(e) \to P(x, e)$	Deduction/Fantasy Rule (8, 20)
(22) $P(x, e)$	Proof by Cases (5, 21)
(23) $P(y, e) \to P(x, e)$	Trivial Proof (22, y arbitrary)
(24) $\forall y : \forall e : P(y, e) \to P(x, e)$	$2\times$ Generalization (23)
(25) $(\neg \exists y : \exists e : (P(y, e) \land \neg P(x, e)) \to U(x)$	Specification (3, x arbirtrary)
(26) $U(x)$	Modus Ponens (24, 25)
(27) $(\forall e : H(e) \to P(x, e)) \to U(x)$	Deduction/Fantasy Rule (4, 26)
(28) $\forall x : (\forall e : H(e) \to P(x, e)) \to U(x)$	Generalization (27)

Q2 (30p). Let L be the regular language $((ab^*)^*c)^*$.

(a, 10p) Find a recurrence for the number S_n of strings of length n in L, using the sum of S_k for $k<n$. Hint: Consider what a string in L can start with and write it as $w = uv$, where u is the shortest nonempty prefix such that $v \in L$.

Solution: A string in L starts with a or c. If it starts with c, it has the form cv, with $v \in L$, $|v| = n-1$. We have S_{n-1} such strings. If it starts with a, it has the form ab^*v, with $v \in L$, $|v| \geq 1$. We get $\sum_{k=1}^{n-1} S_k$ such strings. Thus, $S_n = S_{n-1} + \sum_{k=1}^{n-1} S_k$, for $n > 0$. We have $S_0 = 1$, since $\lambda \in L$.

(b, 5p) Find a recurrence for S_{n+1} in terms of S_n and S_{n-1}. Hint: consider the difference $S_{n+1} - S_n$. For partial credit, find S_2 and S_3 by counting strings, then find the recurrence coefficients. (5p)

Solution: From the recurrence $S_n = S_{n-1} + \sum_{k=1}^{n-1} S_k$, we get $S_{n+1} - S_n = S_n - S_{n-1} + \sum_{k=1}^{n} S_k - \sum_{k=1}^{n-1} S_k = S_n - S_{n-1} + S_n$, so $S_{n+1} = 3S_n - S_{n-1}$, for $n > 0$, with $S_0 = 1$, and $S_1 = 1$ ($c \in L$).

Partial credit: We have $S_2 = 2$ (a, c) and $S_3 = 5$ (aac, abc, acc, cac, ccc). We write $S_{n+1} = cS_n + dS_{n-1}$, with $2 = c + d$, $5 = 2c + d$. We get $c = 3, d = -1$, and $S_{n+1} = 3S_n - S_{n-1}$ for $n > 0$.

(c, 5p) Prove by induction that $S_n = \frac{1}{2}(1 - \frac{1}{\sqrt{5}})(\frac{3+\sqrt{5}}{2})^n + \frac{1}{2}(1 + \frac{1}{\sqrt{5}})(\frac{3-\sqrt{5}}{2})^n$.

Solution: The initial exam draft had "Find and prove a closed form for S_n.", asking to also find the formula above. The characteristic equation is $x^2 - 3x + 1 = 0$, with roots $r_{1,2} = \frac{3\pm\sqrt{5}}{2}$. The general form is $S_n = A\frac{3+\sqrt{5}}{2} + B\frac{3-\sqrt{5}}{2}$. We get $A + B = 1$ and $(A+B)\frac{3}{2} + (A-B)\frac{\sqrt{5}}{2} = 1$, so $B - A = \frac{1}{\sqrt{5}}$. Thus, $A = \frac{1}{2}(1 - \frac{1}{\sqrt{5}})$ and $B = \frac{1}{2}(1 + \frac{1}{\sqrt{5}})$. We have $S_n = \frac{1}{2}(1 - \frac{1}{\sqrt{5}})(\frac{3+\sqrt{5}}{2})^n + \frac{1}{2}(1 + \frac{1}{\sqrt{5}})(\frac{3-\sqrt{5}}{2})^n$.

Having the formula, we can prove it by strong induction. Let $P(n)$ mean the formula is true for n.
Base cases: $n = 0$: $\frac{1}{2}(1 - \frac{1}{\sqrt{5}}) + \frac{1}{2}(1 + \frac{1}{\sqrt{5}}) = 1 = S_0$.
$n = 1$: $\frac{1}{2}(1 - \frac{1}{\sqrt{5}})(\frac{3+\sqrt{5}}{2}) + \frac{1}{2}(1 + \frac{1}{\sqrt{5}})(\frac{3-\sqrt{5}}{2}) = \frac{1}{2} \cdot 2 \cdot (1 \cdot \frac{3}{2} - \frac{1}{\sqrt{5}} \cdot \frac{\sqrt{5}}{2}) = \frac{3}{2} - \frac{1}{2} = 1 = S_1$.
Inductive step: Assume the formula is true for all $k \leq n$, and prove it for $n+1$. With the notations above, since r_1 and r_2 are the roots of $x^2 = 3x-1$, $Ar_1^{n+1} + Br_2^{n+1} = A(3r_1^n - r_1^{n-1}) + B(3r_2^n - r_2^{n-1}) = 3(Ar_1^n + Br_2^n) - (Ar_1^{n-1} + Br_2^{n-1}) = 3S_n - S_{n-1} = S_{n+1}$, which completes the inductive goal.

(d, 10p) Prove that $S_{2k+2} \leq 2 \cdot 7^k$ and $S_{2k+3} \leq 5 \cdot 7^k$ for $k \geq 0$.
First, prove that $S_{n+1} = 7S_{n-1} - S_{n-3}$ for $n \geq 3$.

We expand $S_{n+1} = 3S_n - S_{n-1} = 3(3S_{n-1} - S_{n-2}) - S_{n-1} = 7S_{n-1} + S_{n-1} - 3S_{n-2} = 7S_{n-1} - S_{n-3}$.

We do a proof by induction over k. Since $S_2 = 2$ and $S_3 = 5$, the statements hold for $k = 0$. Assume the statements hold for arbitrary $k \geq 0$. We have $S_{2k+4} = 7S_{2k+2} - S_{2k} < 7S_{2k+2} \leq 7 \cdot 2 \cdot 7^k = 2 \cdot 7^{k+1}$, and $S_{2k+5} = 7S_{2k+3} - S_{2k+1} < 7S_{2k+3} \leq 7 \cdot 5 \cdot 7^k = 5 \cdot 7^{k+1}$, which proves both inequalities.

Q3 (10p). Show that $(n/2)^n > n!$ for all $n > 6$.
Hint: The binomial expansion says $(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k, where \binom{n}{k} = \frac{n!}{(n-k)!k!}$.

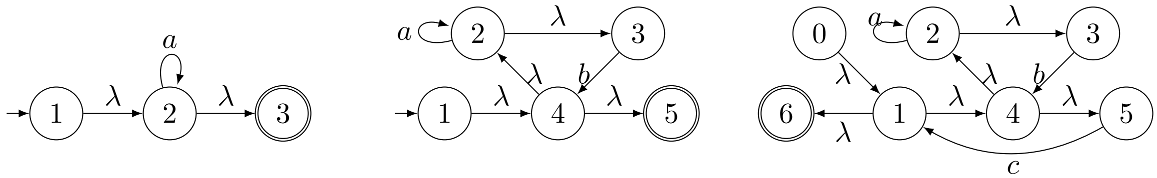
Base case: We show $(7/2)^7 > 7!$, or equivalently, $7^7 > 2^7 \cdot 7!$, or $7^6 > 2^7 \cdot 6!$. We have $7^6 = 49^3 > 48^3 = 2^6 \cdot 12^3$, and $2^7 \cdot 6! = 2^6 \cdot 12 \cdot 5!$. We have $12^2 = 144 > 120 = 5!$, so the inequality holds.

Assume $(n/2)^n > n!$ for arbitrary $n \geq 7$. To prove $(\frac{n+1}{2})^{n+1} > (n+1)!$ it suffices to take the ratio of both sides and prove that $(n+1)^{n+1}/n^n \cdot 2^n/2^{n+1} \geq (n+1)!/n!$, or $(n+1)^{n+1}/n^n/2 \geq (n+1)$, or $(n+1)^n \geq 2n^n$. The latter is true, since $(n+1)^n \geq n^n + \binom{n}{1} n^{n-1}$ for $n \geq 1$, so the proof holds.

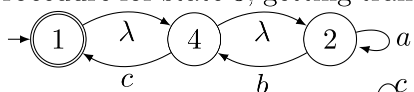
Q!4 (35p). Kleene constructions.

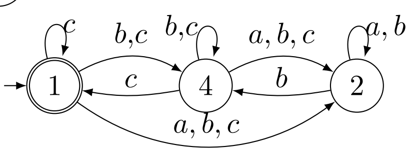
(a, 8p) Construct a λ-NFA for the regular expression $((a^*b)^*c)^*$. Use one of the constructions from the book or class. Show intermediate stages for each Kleene star.

We use the simplified Kleene star construction:



(b, 8p) Build an NFA from the λ-NFA obtained previously, using the construction from the book. You may simplify the λ-NFA first, stating the rules you use.

We can eliminate states 0 and 6 added for the normal form, and make 1 both initial and final state. We can also apply the state elimination procedure for state 3, getting transition $2 \xrightarrow{b} 4$, and state 5, getting $4 \xrightarrow{c} 1$. The simplified λ-NFA is 

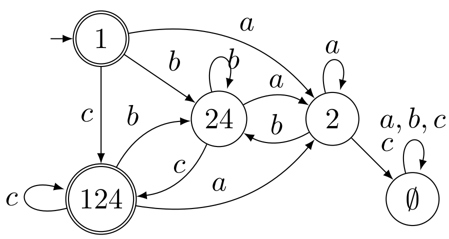


The transitive closure adds move $1 \xrightarrow{\lambda} 2$. We get the letter moves: 14 $2 \xrightarrow{a} 2$, 14 $2 \xrightarrow{b} 4$ 2, and 1 $4 \xrightarrow{c} 1$ 24, totaling 15 transitions. The initial state is already final. The NFA is shown right.

(c, 7p) Using the subset construction, build a DFA from the NFA obtained at the previous point.

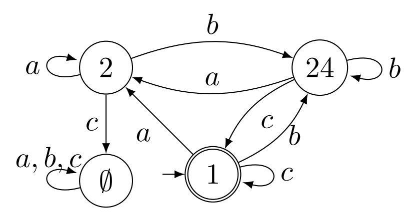
We build the transition table and draw the DFA.
States 1 and 124 are final.

	a	b	c
1	2	24	124
2	2	24	$\emptyset$
24	2	24	124
124	2	24	124



(d, 5p) Minimize the DFA obtained previously obtained. If not using the minimization construction, show that your result is indeed minimal.

States 1 and 124 transition to the same states, they are equivalent.
After merging them, the automaton is minimal.
1 is the only final state, λ distinguishes it from all others.
24 and 2 are distinguished by c (final/non-final).
2 and 24 are distinguished from $\emptyset$ by bc (final/non-final).



(e, 7p) Using the state elimination construction, find a regular expression from the produced minimal DFA, and argue it is equivalent to the initial regular expression.

We add a new initial and final state and eliminate the dead state $\emptyset$. We then eliminate state 2, with transitions from 1 and 24, and to 24. We simplify $b + aa^*b = (\lambda + aa^*)b = a^*b$. We then eliminate state 24, with transitions from and to 1. With $R = a^*b$, we simplify in the same way $c + RR^*c = (\lambda + RR^*)c = R^*c = (a^*b)^*c$. Finally, eliminating 1 we get back to $L = ((a^*b)^*c)^*$.