1. **(20 points)** [Kozen HW2, Q2 ≡ Sipser problem 1.24]
Let $s \in \Sigma^*$ be a string. Define $\mathsf{rev}(s)$ to be the *reverse* of s:

$$\begin{array}{rcl} \mathsf{rev}(\epsilon) & = & \epsilon \\ \mathsf{rev}(s\mathsf{c}) & = & \mathsf{c} \cdot \mathsf{rev}(s), \quad \mathsf{c} \in \Sigma,\ s \in \Sigma^* \end{array}$$

Let $s^{\mathcal{R}} = \mathsf{rev}(s)$.

Let B be a language. Define $B^{\mathcal{R}} = \{s | s^{\mathcal{R}} \in B\}$. Prove that if B is regular, then $B^{\mathcal{R}}$ is regular as well.

Solution: Construct an NFA for $\Delta^{\mathcal{R}}$.

Because B is regular, we can represent it with an NFA. If we reverse the arcs between states and swap the start and accepting states, we get an NFA that recognizes $\Delta^{\mathcal{R}}$. In the stuff that follows, I'll formalize this description and then prove that it works as advertised.

B is an regular language. Let $M = (Q, \Sigma, \Delta, Q_0, F)$ be an NFA such that $L(M) = B$. Let $M^{\mathcal{R}} = (Q, \Sigma, \Delta^{\mathcal{R}}, Q_0^{\mathcal{R}}, F^{\mathcal{R}})$ with

$$\begin{array}{rcll} \Delta^{\mathcal{R}}(q,c) & = & \{p |\ q \in \Delta(p,c)\}, & \text{reverse the arcs} \\ Q_0^{\mathcal{R}} & = & F, & \text{use the accepting states of } M \text{ as start states} \\ F^{\mathcal{R}} & = & Q_0, & \text{use the start states of } M \text{ as accepting states} \end{array}$$

Now for the proof.

Define $\hat{\Delta} : 2^Q \times \Sigma^* \to 2^Q$ and $\hat{\Delta^{\mathcal{R}}} : 2^Q \times \Sigma^* \to 2^Q$ as in Kozen:

$$\begin{array}{rcl} \hat{\Delta}(A, \epsilon) & = & A \\ \hat{\Delta}(A, x\mathsf{c}) & = & \bigcup_{q \in \hat{\Delta}(A,x)} \Delta(q,c) \\ \hat{\Delta}^{\mathcal{R}}(A, \epsilon) & = & A \\ \hat{\Delta}^{\mathcal{R}}(A, x\mathsf{c}) & = & \bigcup_{q \in \hat{\Delta}^{\mathcal{R}}(A,x)} \Delta^{\mathcal{R}}(q,c) \end{array}$$

I'll make use of two lemmas from Kozen in what follows. For completeness, I'll state them here:

Lemma 6.1 (Kozen, p. 34)
For any $x, y \in \Sigma^*$, and for any $A \subseteq Q$,

$$\hat{\Delta}(A, xy) \quad = \quad \hat{\Delta}(\hat{\Delta}(A,x), y)$$

Lemma 6.2 (Kozen, p. 34)
For any $x \in \Sigma^*$, and for any $A_1, A_2, \ldots A_k \subseteq Q$,

$$\hat{\Delta}(\bigcup_i A_i, x) \quad = \quad \bigcup_i \hat{\Delta}(A_i, x)$$

An immediate consequence of this lemma is:

$$\hat{\Delta}(A, x) \quad = \quad \bigcup_{q \in A} \hat{\Delta}(\{q\}, x)$$

I make use of this second form in the proof below.

We want to prove $w^{\mathcal{R}} \in L(M^{\mathcal{R}}) \ \Leftrightarrow \ w \in B$. We start off by expanding the definition of $L(M^{\mathcal{R}})$ – it's pretty much the only thing we can do:

$$\begin{array}{rcll} & w^{\mathcal{R}} \in L(M^{\mathcal{R}}) & & \\ \Leftrightarrow & \hat{\Delta}^{\mathcal{R}}(Q_0^{\mathcal{R}}, w^{\mathcal{R}}) \cap F^{\mathcal{R}} \neq \emptyset, & \text{acceptance condition for an NFA} & \\ \Leftrightarrow & \hat{\Delta}^{\mathcal{R}}(F, w^{\mathcal{R}}) \cap Q_0 \neq \emptyset, & \text{def. } Q_0^{\mathcal{R}} \text{ and } F^{\mathcal{R}} & (1) \\ \Leftrightarrow & \cup_{q \in F} \hat{\Delta}^{\mathcal{R}}(q, w^{\mathcal{R}}) \cap Q_0 \neq \emptyset, & \text{Kozen, lemma 6.2} & \\ \Leftrightarrow & \exists q_f \in F.\ \exists q_s \in Q_0.\ q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, w^{\mathcal{R}}), & \text{set theory} & \end{array}$$

At this point, we recognize that can we perform the same steps starting with $w \in B = L(M)$ to obtain:

$$w \in B \quad \Leftrightarrow \quad \exists q_f \in F.\, \exists q_s \in Q_0.\, q_f \in \hat{\Delta}(\{q_s\}, w), \quad \text{equivalent to equation 1} \tag{2}$$

which means that to show $w^{\mathcal{R}} \in L(M) \Leftrightarrow w \in L(M^{\mathcal{R}})$ all we need to do is show

$$q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, w^{\mathcal{R}}) \quad \Leftrightarrow \quad q_f \in \hat{\Delta}(\{q_s\}, w)$$

In English, this says that if q_s is reachable by the backwards automaton, $M^{\mathcal{R}}$ starting at state q_f with input w^R, then q_f is reachable by the forwards automaton, M starting in state q_s with input w. In other words, if you can get from q_f to q_s by going backwards, then you can get from q_s to q_f by going forwards. We prove this by induction on w.

Induction Hypothesis, $q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, w^{\mathcal{R}}) \Leftrightarrow q_f \in \hat{\Delta}(\{q_s\}, w)$

Base case, $w = \epsilon$:

We just note that $\hat{\Delta}^{\mathcal{R}}(\{q_f\}, \epsilon) = \{q_f\}$ and likewise for $\hat{\Delta}(\{q_s\}, \epsilon)$ and the rest involves just a few simple set operations. For those who want to see all the details, here they are:

$$\begin{array}{lll}
 & q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, w^{\mathcal{R}}) & \\
\Leftrightarrow & q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, \epsilon^{\mathcal{R}}), & w = \epsilon \\
\Leftrightarrow & q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, \epsilon), & \epsilon = \epsilon^{\mathcal{R}} \\
\Leftrightarrow & q_s \in \{q_f\} & \text{def. } \hat{\Delta}^{\mathcal{R}} \\
\Leftrightarrow & q_s = q_f & \text{set theory} \\
\Leftrightarrow & q_f = \in \{q_s\} & \text{set theory} \\
\Leftrightarrow & q_f \in \hat{\Delta}(\{q_s\}, \epsilon), & \text{def. } \hat{\Delta}
\end{array}$$

□

Induction step, $w = x\mathtt{c}$: Noting that $(x\mathtt{c})^{\mathcal{R}} = \mathtt{c}x^{\mathcal{R}}$, we need to prove

$$q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, \mathtt{c}x^{\mathcal{R}}) \Leftrightarrow q_f \in \hat{\Delta}(\{q_s\}, x\mathtt{c})$$

Now, we step back and think about what this means: $q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, \mathtt{c}x^{\mathcal{R}})$ means that we can take a backwards step from q_f with input $\mathtt{c}$ and reach a state, let's call it u, such that we can go backwards from u to q_s with input $x^{\mathcal{R}}$. Conversely, $q_f \in \hat{\Delta}(\{q_s\}, x\mathtt{c})$ means that we can go forwards from q_s with input x to reach some state, v, and then continue with input $\mathtt{c}$ to reach state q_f. Clearly, the state that we want to go back to in the first backwards step, u, should be the same as the state that we reached going forward from q_s with input x, i.e. v. More succinctly, we want $u = v$.

This gives us a strategy. We'll expand the definitions of $\hat{\Delta}$ and $\hat{\Delta}^{\mathcal{R}}$ to separate the processing of x and $\mathtt{c}$. We'll find that there is this state in the middle in each case, i.e. the state that M reaches after processing x and the state that $M^{\mathcal{R}}$ reaches after processing $\mathtt{c}$. We will plan to show that can set these two equal to each other. Matching up forward and backward steps with input $\mathtt{c}$ should involve the relationship that we've defined between Δ and $\Delta^{\mathcal{R}}$. Matching up the forward and backward sequences of steps with input x should use the induction hypothesis. Here we go:

$$\begin{array}{lll}
 & q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, \mathtt{c}x^{\mathcal{R}}) & \\
\Leftrightarrow & q_s \in \hat{\Delta}^{\mathcal{R}}(\hat{\Delta}^{\mathcal{R}}(\{q_f\}, \mathtt{c}), x^{\mathcal{R}}), & \text{Kozen, Lemma 6.1} \\
\Leftrightarrow & q_s \in \hat{\Delta}^{\mathcal{R}}(\Delta^{\mathcal{R}}(q_f, \mathtt{c}), x^{\mathcal{R}}), & \text{def. } \hat{\Delta}^{\mathcal{R}} \\
\Leftrightarrow & q_s \in \hat{\Delta}^{\mathcal{R}}(\{u |\ q_f \in \Delta(u, \mathtt{c})\}, x^{\mathcal{R}}), & \text{def. } \Delta^{\mathcal{R}} \\
\Leftrightarrow & q_s \in \bigcup_{u:q_f \in \Delta(u,\mathtt{c})} \hat{\Delta}^{\mathcal{R}}(\{u\}, x^{\mathcal{R}}), & \text{Kozen, Lemma 6.2} \\
\Leftrightarrow & \exists u.\, (q_f \in \Delta(u, \mathtt{c})) \wedge (q_s \in \hat{\Delta}^{\mathcal{R}}(\{u\}, x^{\mathcal{R}})), & \text{set theory}
\end{array}$$

We now recognize that $q_s \in \hat{\Delta}^{\mathcal{R}}(\{u\}, x^{\mathcal{R}})$ matches the induction hypothesis and is therefore equivalent to $u \in \hat{\Delta}(\{q_s\}, x)$. This gives us

$$q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, \mathtt{c}x^{\mathcal{R}}) \quad \Leftrightarrow \quad \exists u.\, (q_f \in \Delta(u, \mathtt{c})) \wedge (u \in \hat{\Delta}(\{q_s\}, x))$$

Now, we can just reverse the steps that we took above to get to $q_f \in \hat{\Delta}(\{q_s\}, x\mathsf{c})$. For those who want to see all of the details, here they are:

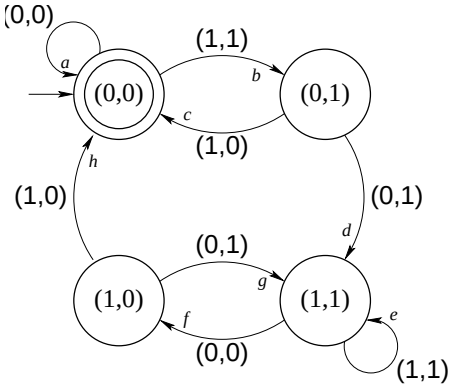
$$
\begin{array}{lll}
 & q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, \mathsf{c}x^{\mathcal{R}}) & \\
\Leftrightarrow & \exists u.\ (q_f \in \Delta(u, \mathsf{c})) \wedge (u \in \hat{\Delta}(\{q_s\}, x)), & \text{shown above} \\
\Leftrightarrow & q_f \in \bigcup_{u \in \hat{\Delta}(\{q_s\}, x)} \hat{\Delta}(\{u\}, \mathsf{c}), & \text{set theory} \\
\Leftrightarrow & q_f \in \hat{\Delta}(\hat{\Delta}(\{q_s\}, x), \mathsf{c}), & \text{def. } \hat{\Delta} \\
\Leftrightarrow & q_f \in \hat{\Delta}(\{q_s\}, \mathsf{c}), & \text{Kozen, Lemma 6.1}
\end{array}
$$

□

This completes the proof. I'll summarize it here to pull all of the pieces together:

$$
\begin{array}{rcll}
w^{\mathcal{R}} \in L(M^{\mathcal{R}}) & \Leftrightarrow & \exists q_s \in Q_0.\ \exists q_f \in F.\ q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, w^{\mathcal{R}}), & \text{equation 1} \\
w \in B & \Leftrightarrow & \exists q_s \in Q_0.\ \exists q_f \in F.\ q_f \in \hat{\Delta}(\{q_s\}, w), & \text{equation 2} \\
q_s \in \hat{\Delta}^{\mathcal{R}}(\{q_f\}, w^{\mathcal{R}}) & \Leftrightarrow & q_f \in \hat{\Delta}(\{q_s\}, w), & \text{shown by induction over } w \\
w^{\mathcal{R}} \in L(M^{\mathcal{R}}) & \Leftrightarrow & w \in \Delta^{\mathcal{R}}, & \text{the three steps above}
\end{array}
$$

□



y	0^0	0^0	1^0	0^1	0^1	1^1	0^1	1^1	1^1	1^0	$\overline{0}$	carry
$+\ 2y$	*0*	*1*	*0*	*0*	*1*	*0*	*1*	*1*	*1*	*0*	*0*	
$3y$	*0*	*1*	*1*	*1*	*0*	*0*	*0*	*1*	*0*	*1*	*0*	
x	0	1	1	1	0	0	0	1	0	1	0	
arc taken	*a*	*c*	*b*	*h*	*f*	*g*	*f*	*e*	*d*	*b*	*a*	

Figure 1: An NFA for recognizing $x = 3y$

2. **(20 points)** [Sipser problem 1.26]
Let $\Sigma = \{0,1\} \times \{0,1\}$. Define $\mathsf{val} : \Sigma^* \to \mathbb{Z} \times \mathbb{Z}$ as in the September 12 notes:

$$\begin{aligned} \mathsf{val}(\epsilon) &= (0,0) \\ \mathsf{val}(s \cdot (\mathsf{a},\mathsf{b})) &= (2 * \mathsf{first}(\mathsf{val}(s)) + \mathsf{a}, 2 * \mathsf{second}(\mathsf{val}(s)) + b) \end{aligned}$$

Let

$$B = \{s \in \Sigma^* | \mathsf{first}(\mathsf{val}(s)) = 3 * \mathsf{second}(\mathsf{val}(s))\}$$

Show that B is regular.

Solution:

First, think about how you would solve this problem manually. Given a string, e.g. $w = (1,0)(0,1)(0,0)(1,0)(1,1)$, you could extract the two binary numbers, let's call them x and y, with $x = 10011$, and $y = 01001$. We want to see if $x = 3 * y$. You might then convert them both to decimal to get $X = 19$ and $y = 9$ and conclude that this particular string is not in the language. Although constructing an automaton that converts binary to decimal and does decimal arithmetic is possible, it would be way too tedious. Let's do the arithmetic in binary instead.

To compute $3 * y$, we can shift y one place to the left to get $2y$ and then calculate the sum of y and $2y$ to get $3y$. This is basically the approach that we'll take. Here are a few observations:

(a) We don't actually output $3y$, we just compare our expectation with the value of x.

(b) Even though the problem specifies x and y as input with their most-significant bits first, it's easier (at least more "natural") to perform the addition starting with the least-significant bit and working up to the most significant. I'll describe a solution that works this way. In other words, my automaton will recognize $\Delta^{\mathcal{R}}$. This is OK. In problem 1, we showed that the regular languages are closed under reversal. Thus, showing that $\Delta^{\mathcal{R}}$ is regular establishes that B is regular.

(c) The automaton needs to keep track of two bits of information with each transition. One of these bits remembers the previous bit of y so we can shift y one place to the left. The other bit remembers the carry from adding $2y$ and y together.

Figure 1 shows the state transition diagram for my machine. It's an NFA because I didn't clutter the diagram with arcs from each state to a "garbage" state when the machine encounters a bit which violates the $x \neq 3y$ rule. Each state is labeled as a tuple $(\textit{carry}, \textit{prev-bit})$ where *carry* is the carry (0 or 1) from the addition to compute $3y$ for the previous step, and *prev-bit* is the bit for y from the previous step.

3. **(20 points)** Let Σ be an alphabet, and let

$$\begin{array}{rcll} \mathsf{half}(\epsilon) & = & \epsilon & \\ \mathsf{half}(\mathsf{c}) & = & \mathsf{c}, & \mathsf{c} \in \Sigma \\ \mathsf{half}(x\mathsf{c}_1\mathsf{c}_2) & = & \mathsf{half}(x)\mathsf{c}_1, & \mathsf{c}_1 \in \Sigma,\ x \in \Sigma^* \end{array}$$

Let $B \subseteq \Sigma^*$ be a language. Define

$$B^{\frac{1}{2}} \quad = \quad \{x \in \Sigma^* | \ \exists y \in \Sigma^*.\ (y \in B) \wedge (x = \mathsf{half}(y))\}$$

Prove that if B is regular, then $B^{\frac{1}{2}}$ is regular as well.

Oops:

The definition of half isn't quite what I intended. With the definition above, you can show:

$$\begin{array}{rcl} \mathsf{half}(\epsilon) & = & \epsilon \\ \mathsf{half}(\mathsf{a}) & = & \mathsf{a} \\ \mathsf{half}(\mathsf{ab}) & = & \mathsf{a} \\ \mathsf{half}(\mathsf{abc}) & = & \mathsf{ab} \\ \mathsf{half}(\mathsf{abcd}) & = & \mathsf{ac} \\ \mathsf{half}(\mathsf{abcde}) & = & \mathsf{abd} \\ \mathsf{half}(\mathsf{abcdef}) & = & \mathsf{acf} \end{array}$$

which is kind of awkward because it treats even and odd length strings differently. On the other hand, it only adds one extra step to the solution. Too bad no one caught it earlier to claim a bug bounty. Here it goes.

Solution to the original problem:

Let $M_B = (Q_B, \Sigma, \delta_B, q_0, F)$ be a DFA that recognizes language B.

The main idea in my solution is to construct an NFA that simulates two moves of M_B for every symbol that it reads. The first move corresponds to the symbol read, and the second uses non-determinism to guess a symbol that goes into a string y such that $\mathsf{half}(y) = x$ and $y \in B$. As noted above, there's a bit of a complication at first to handle the difference between odd and even length strings y. All we have to do is add a non-deterministic choice for whether or not a "guess" symbol is included after reading the first symbol. If the eventual length of y is even, then we guess a symbol after every symbol in x. If the eventual length of y is odd, we guess a symbol for y after every symbol of x except for the first one.

To model guessing a symbol, we define Δ_2 as shown below:

$$\Delta_2(q, \mathsf{c}) \quad = \quad \{q' | \exists \mathsf{d}.\ q' = \delta(\delta(q, \mathsf{c}), \mathsf{d})\}$$

As I'll show below in the proof, the non-determinism of the NFA handles the existential quantification of y in the definition of $B^{\frac{1}{2}}$.

Let $k = |Q_B|$, and assume that the states of M_B are $q_0, q_1, \ldots, q_{k-1}$. For the machine that recognizes $B^{\frac{1}{2}}$, I'll add a new state, q_k to take the place of the start state and handle the odd/even anomaly described above. Here's my NFA that recognizes $B^{\frac{1}{2}}$:

$$\begin{array}{rcll} M_{\frac{1}{2}} & = & (Q_{\frac{1}{2}}, \Sigma, \Delta_{\frac{1}{2}}, \{q_k\}, F_{\frac{1}{2}}) & \\ Q_{\frac{1}{2}} & = & \{q_i | 0 \leq i \leq k\} & \\ \Delta_{\frac{1}{2}}(q_i, \mathsf{c}) & = & \Delta_2(q_i, \mathsf{c}), & \text{if } 0 \leq i < k \\ & = & \{\delta(q_0, \mathsf{c})\} \cup \Delta_2(q_0, \mathsf{c}) & \\ F_{\frac{1}{2}} & = & F \cup \{q_k | \text{if } q_0 \in F\} & \end{array}$$

We define $\hat{\Delta}_{\frac{1}{2}}$ and $\hat{\delta}$ in the usual way. A property of $\Delta_{\frac{1}{2}}$ that we will use is that there are no transitions into state q_k. In other words, for all states $q \in Q_{\frac{1}{2}}$, and all symbols $\mathsf{c} \in \Sigma$, $\Delta_{\frac{1}{2}}(q, \mathsf{c}) \subseteq Q_B$. Likewise, for any string $w \in \Sigma^+$, $\hat{\Delta}_{\frac{1}{2}}(q, w) \subseteq Q_B$.

We will also make use of the following property of half:

$$\frac{|y|}{2} \quad \leq \quad |\mathsf{half}(y)| \quad \leq \quad \frac{|y+1|}{2}$$

The proof is by induction on the length of y.

case $|y| = 0$: This means $y = \epsilon$. Therefore, $\mathsf{half}(y) = \epsilon$, and $|\mathsf{half}(y)| = 0 = \frac{|y|}{2}$ which satisfies the claim.

case $|y| = 1$: This means $y = \mathsf{c}$, for some $\mathsf{c} \in \Sigma$. Therefore, $\mathsf{half}(y) = \mathsf{c}$, and $|\mathsf{half}(y)| = 1 = \frac{|y|+1}{2}$ which satisfies the claim.

case $|y| > 1$: Choose x, c, and d such that $y = x \cdot \mathsf{cd}$. From the definition of half, $\mathsf{half}(y) = \mathsf{half}(x) \cdot \mathsf{c}$. We have

1.	$\mathsf{half}(y)$	$=$	$\mathsf{half}(x \cdot \mathsf{cd})$,	$=$	$\mathsf{half}(x) \cdot \mathsf{c}$,	choices of x, c, and d; and def. half
2.	$\frac{\lvert x\rvert}{2}$	$\leq$	$\lvert\mathsf{half}(x)\rvert$	$\leq$	$\frac{\lvert x\rvert+1}{2}$,	induction hypothesis
3.	$\frac{\lvert x\rvert+2}{2}$	$\leq$	$\lvert\mathsf{half}(x)+1\rvert$	$\leq$	$\frac{\lvert x\rvert+3}{2}$,	add 1 to everything
4.	$\frac{\lvert x\cdot\mathsf{cd}\rvert}{2}$	$\leq$	$\lvert\mathsf{half}(x)\cdot\mathsf{c}\rvert$	$\leq$	$\frac{\lvert x\cdot\mathsf{cd}\rvert+1}{2}$,	def. $\lvert w\rvert$
5.	$\frac{\lvert y\rvert}{2}$	$\leq$	$\lvert\mathsf{half}(y)\rvert$	$\leq$	$\frac{\lvert y\rvert+1}{2}$,	$y = x \cdot \mathsf{cd}, \mathsf{half}(y) = \mathsf{half}(x) \cdot \mathsf{c}$

Because $M_{\frac{1}{2}}$ is an NFA, $L(M_{\frac{1}{2}})$ is regular, Now, I'll prove that $L(M_{\frac{1}{2}}) = B^{\frac{1}{2}}$ which will prove that $B^{\frac{1}{2}}$ is regular.

The proof, of course, is by induction on the length of x. We need an induction hypothesis. In particular, we need to keep track of where $M_{\frac{1}{2}}$ goes as it reads a string. Thus, our induction hypothesis describes a relationship between states of $M_{\frac{1}{2}}$ and states of M.

Induction Hypothesis:

$$(q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x)) \vee (x = \epsilon) \quad \Leftrightarrow \quad (\exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')) \vee (x = \epsilon)$$

In English, the induction hypothesis says that if $M_{\frac{1}{2}}$ can reach q by reading x, then there is some y with $x = \mathsf{half}(y)$ such that M reaches q' when reading y. The disjuncts for $(x = \epsilon)$ allow the first step to be special.

Case $x = \epsilon$:

This case is easily shown because $x = \epsilon$ ensures that both sides of the claimed equivalence are true.

Case $|x| = 1$:

Because $x \neq \epsilon$, we can simplify the claim to

$$q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x) \quad \Leftrightarrow \quad \exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$$

We will use start with this simplified version here and in the case for the induction step below. Because $|x| = 1$, x consists of a single symbol. We'll write c to denote this symbol. From the definitions of $\hat{\Delta}_{\frac{1}{2}}$ and $\Delta_{\frac{1}{2}}$, we obtain

$$q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x) \quad \Leftrightarrow \quad (q' = \delta(q_0, \mathsf{c})) \vee (q' \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c}))$$

We prove the $\Rightarrow$ and $\Leftarrow$ directions separately.

$\Rightarrow$

We need to prove:

$$((q' = \delta(q_0, \mathsf{c})) \vee (q' \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c}))) \quad \Rightarrow \quad (\exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q'))$$

We consider the $q' = \delta(q_0, \mathsf{c})$ and $q \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c})$ cases separately.

$q' = \delta(q_0, \mathsf{c})$:

Let $y = \mathsf{c}$. From the definition of half, $\mathsf{half}(y) = \mathsf{c} = x$. Furthermore, $\hat{\delta}(y) = \delta(q_0, \mathsf{c}) = q'$. Thus,

$$q' = \delta(q_0, \mathsf{c}) \quad \Rightarrow \quad \exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$$

$q' \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c})$:

Expanding the definition of $\Delta_{\frac{1}{2}}$, we obtain $\exists \mathsf{d}.\ q' = \delta(\delta(q_0, \mathsf{c}), \mathsf{d})$. Choose d such that $q' = \delta(\delta(q_0, \mathsf{c}), \mathsf{d})$, and let $y = \mathsf{cd}$. From the definition of half, $\mathsf{half}(y) = \mathsf{c} = x$. Furthermore, $\hat{\delta}(y) = \delta(\delta(q_0, \mathsf{c}), \mathsf{d}) = q'$.

$$q' \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c}) \quad \Rightarrow \quad \exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$$

Combining these, we conclude $((q' = \delta(q_0, \mathsf{c})) \vee (q' \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c}))) \ \Rightarrow \ (\exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q'))$ as required.

$\Leftarrow$

We need to prove:

$$((q' = \delta(q_0, \mathsf{c})) \vee (q' \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c}))) \quad \Leftarrow \quad (\exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q'))$$

Let y be a string such that $(x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$. By the case assumption, $x = \mathsf{c}$. Thus, $\mathsf{half}(y) = \mathsf{c}$. From the definition of half, we get that either $y = \mathsf{c}$ or there is some symbol d such that $y = \mathsf{cd}$. We consider these two cases separately.

$y = \mathsf{c}$:

From the definition of half, we conclude $x = \mathsf{c}$. We also have: $q' = \hat{\delta}(q_0, y) = \delta(q_0, \mathsf{c})$, which establishes the claim.

$y = \mathsf{cd}$:

From the definition of half, we conclude $x = \mathsf{c}$. We also have: $q' = \hat{\delta}(q_0, y) = \hat{\delta}(q_0, \mathsf{cd}) = \delta(\delta(q_0, \mathsf{c}), \mathsf{d})$, which establishes $\exists \mathsf{d}.\ q' = \delta(\delta(q_0, \mathsf{c}), \mathsf{d})$. Therefore $q' \in \Delta_2(q_0, \mathsf{c})$ and thus $q' \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c})$ as required.

We have now established, $((q' = \delta(q_0, \mathsf{c})) \vee (q' \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c}))) \ \Leftarrow \ (\exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q'))$ as required.

Having shown the $\Rightarrow$ and $\Leftarrow$ proves,

$$q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x) \quad \Leftrightarrow \quad (q' = \delta(q_0, \mathsf{c})) \vee (q' \in \Delta_{\frac{1}{2}}(q_0, \mathsf{c}))$$

and completed the $|x| = 1$ case.

Case $|x| > 1$**:**

Let $x = w\mathsf{c}$ where $w \in \Sigma^*$ and $\mathsf{c} \in \Sigma$. Because $x \neq \epsilon$, we again simplify the claim to:

$$q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x) \quad \Leftrightarrow \quad \exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$$

		$q' \in \hat{\Delta}_{\frac{1}{2}}(q_0, x)$	
1.	$\Leftrightarrow$	$q' \in \hat{\Delta}_{\frac{1}{2}}(q_0, w \cdot \mathsf{c})$,	choice of w and c
2.	$\Leftrightarrow$	$q' \in \cup q_j \in \hat{\Delta}_{\frac{1}{2}}(q_0, w)\Delta_{\frac{1}{2}}(q_j, \mathsf{c})$,	def. $\hat{\Delta}_{\frac{1}{2}}$
3.	$\Leftrightarrow$	$\exists q_j.\ (q_j \in \hat{\Delta}_{\frac{1}{2}}(q_0, w)) \wedge (q' \in \Delta_{\frac{1}{2}}(q_j, \mathsf{c}))$,	set theory
4.	$\Leftrightarrow$	$\exists q_j \in Q_B.\ (q_j \in \hat{\Delta}_{\frac{1}{2}}(q_0, w)) \wedge (q' \in \Delta_{\frac{1}{2}}(q_j, \mathsf{c}))$,	property of $\Delta_{\frac{1}{2}}$, $\hat{\Delta}_{\frac{1}{2}}$
5.	$\Leftrightarrow$	$\exists q_j \in Q_B.\ \exists v.\ (w = \mathsf{half}(v)) \wedge (q_j = \hat{\delta}(q_0, v)) \wedge (q' \in \Delta_{\frac{1}{2}}(q_j, \mathsf{c}))$,	induction hypothesis
6.	$\Leftrightarrow$	$\exists q_j \in Q_B.\ \exists v.\ (w = \mathsf{half}(v)) \wedge (q_j = \hat{\delta}(q_0, v)) \wedge \exists \mathsf{d}.\ q' = \delta(\delta(q_j, \mathsf{c}), \mathsf{d})$	induction hypothesis

We complete the proof by proving the $\Rightarrow$ and $\Leftarrow$ cases separately.

$\Rightarrow$ We need to show

$$q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x) \quad \Rightarrow \quad \exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$$

Assume $q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x)$. Choose q_j, v, and d such that $w = \mathsf{half}(v)$, $q_j = \hat{\delta}(q_0, v)$, and $q' = \delta(\delta(q_j, \mathsf{c}), \mathsf{d})$. As shown in step 3, such q_j, v, and d are guaranteed to exist. Let $y = v \cdot \mathsf{cd}$. We have:

		$\hat{\delta}(q_0, y)$	
7.	$=$	$\hat{\delta}(q_0, v \cdot \mathsf{cd})$,	choice of y
8.	$=$	$\delta(\delta(\hat{\delta}(q_0, v), \mathsf{c}), \mathsf{d})$,	def. $\hat{\delta}$
9.	$=$	$\delta(\delta(q_j, \mathsf{c}), \mathsf{d})$,	step 3: $\hat{\delta}(q_0, v) = q_j$
10.	$=$	q',	choice of q'

Furthermore,

		$\mathsf{half}(y)$	
11.	$=$	$\mathsf{half}(v \cdot \mathsf{cd})$,	choice of y
12.	$=$	$\mathsf{half}(v) \cdot \mathsf{c}$,	def. half
13.	$=$	$w \cdot \mathsf{c}$,	step 3: $\mathsf{half}(v) = w$
14.	$=$	x, choice of w and c: $x = w \cdot c$	

This establishes

$$q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x) \quad \Rightarrow \quad \exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$$

as required.

$\Leftarrow$ We need to show

$$q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x) \quad \Leftarrow \quad \exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$$

Assume $\exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$. Choose y such that $(x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$. Because $|x| > 1$, $|y| > 2$. Choose v, c, and d such that $y = v \cdot \mathsf{cd}$. Let $q_j = \hat{\delta}(q_0, v)$. We have:

15.	$\mathsf{half}(v \cdot \mathsf{cd}) = \mathsf{half}(v) \cdot \mathsf{c}$,	def. half
16.	$\mathsf{half}(y) = x$,	$y = v \cdot \mathsf{cd}, \mathsf{half}(v) = w, x = w \cdot \mathsf{c}$
17.	$q_j \in \hat{\Delta}_{\frac{1}{2}}(q_k, v)$,	induction hypothesis
18.	$q' = \delta(\delta(q_j, \mathsf{c}), \mathsf{d})$	def. $\hat{\delta}$, $q' = \hat{\delta}(q_0, v \cdot \mathsf{cd})$
19.	$q' \in \Delta_{\frac{1}{2}}(q_j, \mathsf{c}))$	def. $\Delta_{\frac{1}{2}}$
20.	$q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x))$	def. $\hat{\Delta}_{\frac{1}{2}}$, $x = \mathsf{half}(v \cdot \mathsf{cd})$

This establishes the $\Leftarrow$ case.

Having proven the $\Rightarrow$ and $\Leftarrow$ cases, we've completed to proof for $|x| > 1$. Having completed proofs for $|x| = 0$, $|x| = 1$, and $|x| > 1$, we've handled all possible strings x, which completes the proof of

$$(q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x)) \vee (x = \epsilon) \quad \Leftrightarrow \quad \exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q') \vee (x = \epsilon)$$

We now use this to show that $x \in L(M_{\frac{1}{2}}) \Leftrightarrow x \in B^{\frac{1}{2}}$.

case $x \in L(M_{\frac{1}{2}})$: This means that $\hat{\Delta}_{\frac{1}{2}}(q_k, x) \cap F_{\frac{1}{2}} \neq \emptyset$. If $x = \epsilon$, then q_k is accepting, which means that q_0 is accepting, and therefore $\epsilon \in B$ and $\epsilon \in B^{\frac{1}{2}}$. Otherwise, let q' be a state in $\hat{\Delta}_{\frac{1}{2}}(q_k, x) \cap F_{\frac{1}{2}}$. Because $x \in \Sigma^+$, $q' \in Q_B$ which means that $q' \in F$ (because $F_{\frac{1}{2}} \cap Q_B = F$). From our induction result, $\exists y.\ (x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$. Let y be a string such that $(x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) = q')$. Because $q' \in F$, $y \in B$. Therefore, $x \in B^{\frac{1}{2}}$ as required.

case $x \in B^{\frac{1}{2}}$: If $x = \epsilon$, then q_0 is an accepting state of M_B and q_k is an accepting state of $M_{\frac{1}{2}}$. This means that $x \in L(M_{\frac{1}{2}})$ as required. Otherwise, let y be a string such that $(x = \mathsf{half}(y)) \wedge (\hat{\delta}(q_0, y) \in F)$. Such a y exists because $x \in B^{\frac{1}{2}}$. Let $q' = \hat{\delta}(q_0, y)$. From our induction result, $q' \in \hat{\Delta}_{\frac{1}{2}}(q_k, x)$, and $q' \in F_{\frac{1}{2}}$ because $q' \in F$ and $F \subseteq F_{\frac{1}{2}}$. Thus, $x \in L(M_{\frac{1}{2}})$.

□

4. **(20 points)** The textbook for CpSc 121 (Rosen: *Discrete Mathematics and Its Applications,* 5^{th} *edition*, p. 12), suggests searching the web for universities in Mexico by looking for pages that contain the word "MEXICO" but not the word "NEW" (to exclude pages about universities in New Mexico). He writes his search as

(MEXICO *AND* UNIVERSITY) *AND NOT* NEW

Searches are assumed to be case-insensitive.

(a) (5 points) What is wrong with Rosen's proposed search criterion? Give an example of text that could appear on a web page for which this search would do something different than its informal description.

(b) (10 points) Now, write a regular expression that much better matches the informal specification. In particular, it shouldn't have the problem that you identified for Rosen's query in part (a).

(c) (5 points) Presumably, you haven't solved the natural language understanding problem, so your expression will also fail to meet the informal specification for some web pages. Describe a web page for which your regular expression does the "wrong" thing.

(d) (2 points, extra credit) Write a program in a language for which there is a compiler on the CS department undergraduate machines that solves the natural language understanding problem. This program should be able to take as input books, journal articles, and other English text and answer questions posed in natural English about their contents.

Solution:

(a) **What's wrong with Rosen's query?**
Rosen's criterion rejects any web page that includes the word "NEW". For example:

> "Jalisco was originally part of New Galicia before it became part of greater Mexico in 1821... . So important to the area is tequila, that the local university offers a course in tequila engineering."

From: http://www.ianchadwick.com/tequila/country.html

(b) **Write a regular expression to get a better query.**
My solution is the expression α defined below:

$$\begin{aligned}
\alpha &= @(\alpha_M @ \alpha_U + \alpha_U @ \alpha_M)@ \\
\alpha_M &= \alpha_{\sim NEW} \alpha_{WS}^* \mathsf{MEXICO} \\
\alpha_U &= \mathsf{UNIVERSITY} \\
\alpha_{\sim NEW} &= \alpha_{\sim W} + \alpha_{\sim E}\mathsf{W} + \alpha_{\sim N}\mathsf{EW} + (\#\cap \sim \alpha_{WS})\alpha_{\sim N}\mathsf{EW} \\
\alpha_{\sim W} &= \#\cap \sim (\mathsf{W} + \alpha_{WS}) \\
\alpha_{\sim E} &= \#\cap \sim \mathsf{E} \\
\alpha_{\sim N} &= \#\cap \sim \mathsf{N} \\
\alpha_{WS} &= \text{any white space characters}
\end{aligned}$$

The @s at the beginning and end of α allow the matches for MEXICO and UNIVERSITY to appear anywhere in the document. Furthermore, α appears the matches for MEXICO and UNIVERSITY to appear in either order and to be separated by any amount of text in between.
The expression for $\alpha_{\sim NEW}$ makes sure that the word MEXICO in the match is not immediately preceded by the word NEW. I took some care to handle the white space between the word that precedes MEXICO and the word that precedes NEW. Since Rosen didn't talk about white space, it's OK if a solution doesn't address this.

(c) **Describe a page for which this expression does the "wrong" thing.**
This sentence matches the regular expression that I gave, but it doesn't pertain to a university in Mexico. When I put the solutions on the course web, this will be an example of a page for which the regular expression will generate a false positive.

(d) **Extra credit: solve the natural language understanding problem.**
Not attempted.

5. **(20 points)** Let M be a two-input-tape finite automaton as described in the September 14 lecture notes. Define

$$L_\forall(M) \quad = \quad \left\{ s_1 \in \Sigma_1^* \;\middle|\; \forall s_2 \in \Sigma_2^{|s_1|}.\ \mathsf{weave}(s_1, s_2) \in L(M) \right\}$$

We call a machine with this acceptance condition a universally quantified, two-input-tape, finite automaton.

(a) (10 points) Prove that $L_\forall(M)$ is regular.

(b) (10 points) In the September 14 notes, we showed that we could simplify the presentation of a state transition diagram for an existentially quantified, two-input-tape finite automaton by dropping the Σ_2 component of each label and omiting arcs to terminally non-accepting states. The equivalent simplifications for universally quantified, two-input-tape, finite automaton are dropping the Σ_2 component of each label and omittng arcs to terminally accepting states. If a state has no outgoing arc for some input symbol, it is assumed that the machine transitions to a terminally accepting state.

The automaton corresponding to the simplified state-transition diagram is called a $\forall$-automaton (pronounced "for all automaton"). This is a generally accepted technical term (whereas I made up the names "existentially-" and "universally-quantified, two-input-tape, finite automata" just so we could talk about them).

Draw the state transition diagram for a $\forall$-automaton that accepts a string $s \in \{\mathsf{a}, \mathsf{b}\}^*$ iff:

every 'a' is followed immediately by a 'b';

and the number of 'a' symbols in the input is even;

and the number of 'b' symbols in the input is a multiple of three.

Solution:

(a) **Prove that $\mathbf{L_\forall(M)}$ is regular.**
I will first show that $\sim L_\forall(M)$ is regular.

$$\begin{aligned} \sim L_\forall(M) \quad &= \quad \left\{ s_1 \in \Sigma_1^* \;\middle|\; \sim \forall s_2 \in \Sigma_2^{|s_1|}.\ \mathsf{weave}(s_1, s_2) \in L(M) \right\} \\ &= \quad \left\{ s_1 \in \Sigma_1^* \;\middle|\; \exists s_2 \in \Sigma_2^{|s_1|}.\ \mathsf{weave}(s_1, s_2) \in \sim L(M) \right\} \end{aligned}$$

Assume $M = (Q, \Sigma_1, \Sigma_2, \delta, q_0, F)$ and define $M' = (Q, \Sigma_1, \Sigma_2, \delta, q_0, \sim F)$. We now have

$$\sim L_\forall(M) \quad = \quad L_\exists(M')$$

In the Sept. 14 notes, we showed that $L_\exists(M')$ is simply the language of an NFA; thus, it is regular. Therefore, $\sim L_\forall(M)$ is regular. Because the regular languages are closed under complement, $L_{\forall(M)}$ is regular as well.

(b) **Draw the state transition diagram…**
See figure 2. Note that the non-determinism in this machine is the "forall" form described above, not the "exists" form that we typically use in class. This machine has three start states. If a string w is *not* in the language, then at least one of the three component machines must reject it. The $\forall$ non-determinism selects a component machine that will reject w and the machine runs from there. On the other hand, if w is in the language, then it will be accepted no matter what component machine is chosen.
To keep the duality with existentially quantified machines, states that have no successors for some input symbol transition to a permanently accepting state (as described above). Thus, I had to include a "garbage" state for the machine that checks that every 'a' is followed immediately by a 'b'.

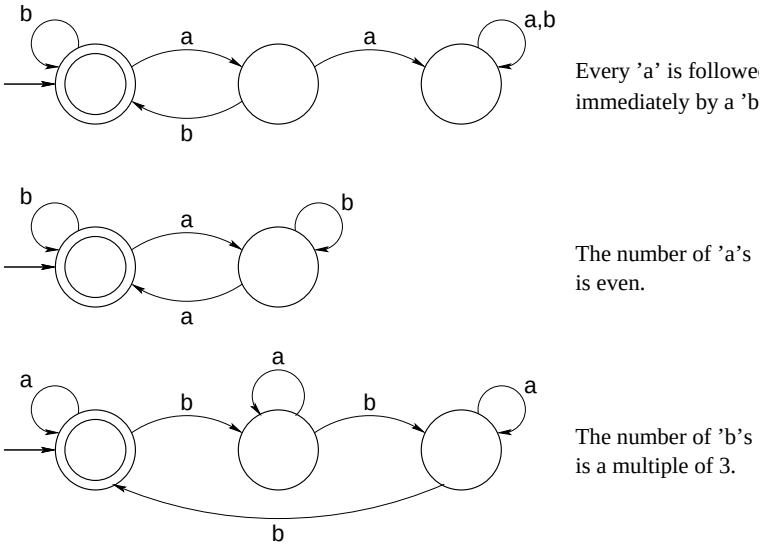


Figure 2: A $\forall$-automaton for problem 5b

Shorter Proofs

The proofs for problems 1 and 3 were rather long. That is in part because I included many explanatory comments and showed lots of details. I assume that this will help some students who want to study the proofs. On the other hand, this could be intimidating if you get the idea that you need to write multiple page proofs for your homework solutions. This isn't necessary. I will give a "short" proof for each of these two problems. In each case, I'll take a different approach to solving the problem as well, just to give you more examples of how to use the concepts that we've covered in the class so far.

1. Prove that regular languages are closed under reversal.

Let B be a regular language. Because it is regular, there is a regular expression, α that recognizes it. We define $\mathsf{rev}(\alpha)$ as shown below:

$$\begin{aligned}
\mathsf{rev}(\emptyset) &= \emptyset \\
\mathsf{rev}(\epsilon) &= \epsilon \\
\mathsf{rev}(\mathsf{c}) &= \mathsf{c} \\
\mathsf{rev}(\alpha_1 + \alpha_2) &= \mathsf{rev}(\alpha_1) + \mathsf{rev}(\alpha_2) \\
\mathsf{rev}(\alpha_1 \cdot \alpha_2) &= \mathsf{rev}(\alpha_2) \cdot \mathsf{rev}(\alpha_1) \\
\mathsf{rev}(\alpha^*) &= \mathsf{rev}(\alpha)^*
\end{aligned}$$

We now prove by induction on the structure of α that $L(\mathsf{rev}(\alpha)) = (L(\alpha))^{\mathcal{R}}$.

$\alpha = \emptyset$:
In this case, both $L(\alpha)$ and $L(\mathsf{rev}(\alpha))$ are the empty language and the claim holds.

$\alpha = \epsilon$:
In this case, both $L(\alpha)$ and $L(\mathsf{rev}(\alpha))$ contain only the empty string. The empty string is its own reverse; so, the claim holds.

$\alpha = \mathsf{c}$:
In this case, both $L(\alpha)$ and $L(\mathsf{rev}(\alpha))$ contain only the string consisting of a single symbol c. The string c is its own reverse; so, the claim holds.

$\alpha = \alpha_1 + \alpha_2$:
Consider $w \in L(\alpha)$. Then either $w \in L(\alpha_1)$ or $w \in L(\alpha_2)$. In the first case, then $w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1))$;

therefore, $w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha)$. An analogous argument shows that if $w \in L(\alpha_2)$ then $w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_2)) \subseteq L(\mathsf{rev}(\alpha))$.

Now consider $w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha))$. Then either $w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1))$ or $w \in L(\mathsf{rev}(\alpha_2))$, and arguments analogous to those above show that $w \in L(\alpha)$ as required.

$\alpha = \alpha_1 \cdot \alpha_2$:

My proof uses the fact that $y^{\mathcal{R}} \cdot x^{\mathcal{R}} = (xy)^{\mathcal{R}}$. I'll prove that by induction first (i.e. a lemma), and then get on with the main proof. The proof is by induction on y:

Induction hypothesis: $y^{\mathcal{R}} \cdot x^{\mathcal{R}} = (xy)^{\mathcal{R}}$.

Base case: $y = \epsilon$.

$$y^{\mathcal{R}} \cdot x^{\mathcal{R}} \quad = \quad \epsilon \cdot x^{\mathcal{R}} \quad = \quad x^{\mathcal{R}} \quad = \quad (x\epsilon)^{\mathcal{R}} \quad = \quad (xy)^{\mathcal{R}}$$

Induction step: $y = z\mathsf{c}$.

$$\begin{array}{lll} & (xy)^{\mathcal{R}} & \\ = & (xz\mathsf{c})^{\mathcal{R}}, & y = z\mathsf{c} \\ = & \mathsf{c}(xz)^{\mathcal{R}}, & \text{def. } \mathsf{rev} \\ = & \mathsf{c}z^{\mathcal{R}}x^{\mathcal{R}}, & \text{induction hypothesis} \\ = & (z\mathsf{c})^{\mathcal{R}}x^{\mathcal{R}}, & \text{def. } \mathsf{rev} \\ = & y^{\mathcal{R}} \cdot x^{\mathcal{R}}, & y = z\mathsf{c} \\ \Box & & \end{array}$$

Now, we do the main proof for this case. If $w \in L(\alpha)$ then there are strings x and y such that $w = xy$, $x \in L(\alpha_1)$ and $y \in L(\alpha_2)$. By the induction hypothesis, $x^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1))$ and $y^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_2))$. Thus, $y^{\mathcal{R}}x^{\mathcal{R}} \in L(\mathsf{rev}(\alpha))$ because $\mathsf{rev}(\alpha) = \mathsf{rev}(\alpha_2) \cdot \mathsf{rev}(\alpha_1)$.

The proof that if $w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha))$ then $w \in L(\alpha)$ is equivalent to the one above.

$\alpha = \alpha_1^*$:

If $w \in L(\alpha_1^*)$, then there exists k such that $w \in L(\alpha_1^k)$; likewise if $w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1)^*)$. This leads to a proof by induction on k:

Induction hypothesis: $w \in L(\alpha_1^k) \Leftrightarrow w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1)^*)$.

Base case: $k = 0$.

$$w \in L(\alpha_1^0) \quad \Leftrightarrow \quad w \in L(\epsilon) \quad \Leftrightarrow \quad w = \epsilon \quad \Leftrightarrow \quad w^{\mathcal{R}} = \epsilon \quad \Leftrightarrow \quad w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1)^0) \quad \Leftrightarrow \quad w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1)^*)$$

Induction step: $k > 0$.

$$\begin{array}{lll} & w \in L(\alpha_1^k) & \\ \Leftrightarrow & w \in L(\alpha_1^{k-1}\alpha_1), & \text{def. } \alpha_1^k,\ k > 0 \\ \Leftrightarrow & w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1)\mathsf{rev}(\alpha_1)^{k-1}), & \text{induction (on } \alpha_1\text{) hypothesis} \\ \Leftrightarrow & w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1)^k), & \text{def. } \alpha_1^k,\ k > 0 \\ \Leftrightarrow & w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha_1)^*), & \text{def. asteration} \\ \Leftrightarrow & w^{\mathcal{R}} \in L(\mathsf{rev}(\alpha)), & \mathsf{rev}(\alpha_1^*) = \mathsf{rev}(\alpha_1)^* \\ \Box & & \end{array}$$

3. Prove that if B is regular, then so is $\mathsf{half}(\mathrm{B})$.

First, I'll take this occasion to fix the half function. Let

$$\begin{array}{rcll} \mathsf{half}(\epsilon) & = & \epsilon & \\ \mathsf{half}(\mathsf{c}) & = & \mathsf{c}, & \mathsf{c} \in \Sigma \\ \mathsf{half}(x \cdot \mathsf{c}_1\mathsf{c}_2) & = & \mathsf{half}(x)\mathsf{c}_2, & \mathsf{c}_1, \mathsf{c}_2 \in \Sigma,\ x \in \Sigma^* \end{array}$$

and I'll assume that half is defined using this version of half. abcdefg

In the September 26 lecture, we saw that the regular languages are closed under homomorphisms and inverse homomorphisms (see Kozen lecture 10). Let B be a language over Σ, and let ρ be a symbol not in Σ. Let $\Sigma_2 = (\Sigma \cup \rho) \times \Sigma$. The purpose for ρ is to provide a padding for the first symbol of a string in Σ_2^* if we're trying to match it with an odd-length string in Σ. Let $H_1 : \Sigma_2 \to \Sigma^*$ be the homomorphism

$$\begin{array}{rcll} H_1((\mathsf{c}_1, \mathsf{c}_2)) & = & \mathsf{c}_1\mathsf{c}_2, & \text{if } \mathsf{c}_1 \neq \rho \\ & = & \mathsf{c}_2, & \text{if } \mathsf{c}_1 = \rho \end{array}$$

Because the regular languages are closed under inverse homomorphism, $H_1^{-1}(B)$ is a regular language. This isn't quite what I want because it can have ρ symbols anywhere. I want to only allow ρ symbols in the first symbol of the string. There is a regular language that only recognizes strings in Σ_2^* that either have no ρ components, or that only have a single ρ as the first component of the first symbol. Let B_ρ be this language. The regular languages are closed under intersection. Therefore, $B_1 = H_1^{-1}(B) \cap B_\rho$ is regular.

Now, I want to discard every other symbol in the original string. This corresponds to discarding the first component of each symbol in a string over Σ_2^*. Let

$$H_2((\mathsf{c}_1, \mathsf{c}_2)) \quad = \quad \mathsf{c}_2$$

Let $B_2 = H_2(B_1)$. Because the regular languages are closed under homomorphism, B_2 is regular.

I claim that $B_2 = B^{\frac{1}{2}}$. To prove this, I will show

$$\{w | \exists z \in B_\rho.\ (H_1(z) = y) \wedge (w = H_2(z)\} \quad \Leftrightarrow \quad \{\mathsf{half}(y)\}$$

The proof is by induction on the length of y.

$|y| = 0$
: This means $y = \epsilon$. The only $z \in B_\rho$ with $H_1(z) = y$ is ϵ. Thus, if $x \in \{w| \ldots\}$, then $x = \epsilon = \mathsf{half}(y)$.

$|y| = 1$
: This means that y consists of a single symbol from Σ. Let $y = \mathsf{c}$. The only $z \in B_\rho$ with $H_1(z) = y$ is (ρ, c). Thus, if $x \in \{w| \ldots\}$, then $x = \mathsf{c} = \mathsf{half}(y)$.

$|y| > 1$
: Let $y = v \cdot \mathsf{c}_1\mathsf{c}_2$. By the induction hypothesis, there is some $u \in B_\rho$ such that $H_1(u) = v$ and for every such u, $H_2(u) = \mathsf{half}(v)$. Let u be some such string in B_ρ. Let, $z = u \cdot (\mathsf{c}_1, \mathsf{c}_2)$. From the definition of H_1, $H_1(z) = y$, and we conclude: $H_2(z) = H_2(u) \cdot\ \mathsf{c}_2 = \mathsf{half}(v) \cdot \mathsf{c}_2 = \mathsf{half}(y)$.

: Now let z be any string in Σ_2^* such that $H_1(z) = y$. Because $y \neq \epsilon$, $z \neq \epsilon$. Let $z = u \cdot (\mathsf{d}_1, \mathsf{d}_2)$. Because $|y| > 1$, $\mathsf{d}_1 \neq \rho$ (a simple argument by contradiction: if $\mathsf{d}_1 = \rho$, then $u = \epsilon$ which means $y = \mathsf{d}_2$ and $|y| = 1$). By the choice of z, $y = H_1(z) = H_1(u) \cdot \mathsf{d}_1\mathsf{d}_2$. Thus $H_1(u) = v$, $\mathsf{d}_1 = \mathsf{c}_1$, and $\mathsf{d}_2 = \mathsf{c}_2$. By the induction hypothesis, $H_2(u) = \mathsf{half}(v)$. Thus,

$$H_2(z) = H_2(u \cdot \mathsf{c}_1\mathsf{c}_2) = H_2(u) \cdot \mathsf{c}_2 = \mathsf{half}(v) \cdot \mathsf{c}_2 = \mathsf{half}(v \cdot \mathsf{c}_1\mathsf{c}_2) = \mathsf{half}(y)$$

: □

We have shown that $B_2 = \{x | \exists y \in B.\ x = \mathsf{half}(y)\}$. In other words, $B_2 = B^{\frac{1}{2}}$. The language B_2 is regular by its construction. Thus, $B^{\frac{1}{2}}$ is regular.