Homework 1 - 2

Menu

HW#1 Problem 1

(Exercise 0.7; 30 points) For each part, give a binary relation that satisfies the condition. *Please illustrate the relation using a directed graph.*

(a) Reflexive and symmetric but not transitive

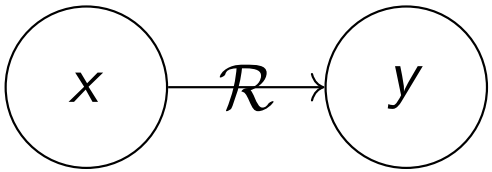
(b) Reflexive and transitive but not symmetric

(c) Symmetric and transitive but not reflexive

HW#1 Problem 1

Directed graph of a binary relation $\mathcal{R}$:

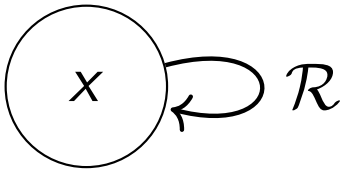
$x \mathcal{R} y$:



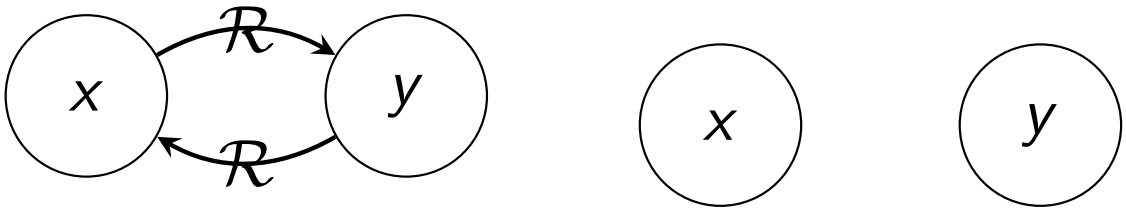
HW#1 Problem 1

Let $\mathcal{R}$ be a binary relation on a set S:

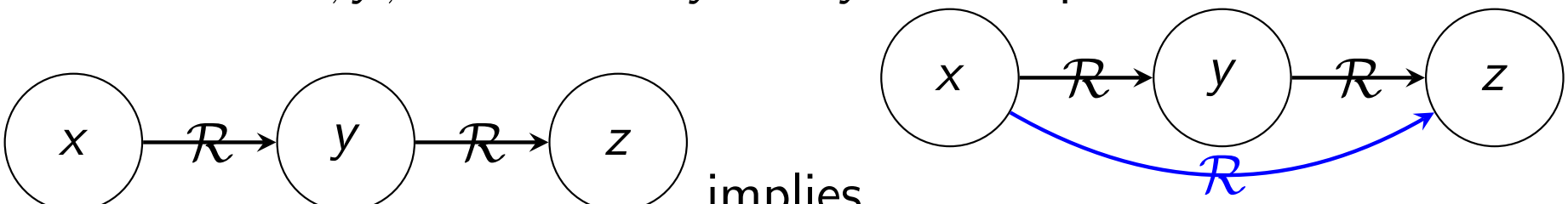
Reflexive: $\forall x \in S,\ x\ \mathcal{R}\ x$.



Symmetric: $\forall x, y \in S,\ x\ \mathcal{R}\ y$ iff $y\ \mathcal{R}\ x$.

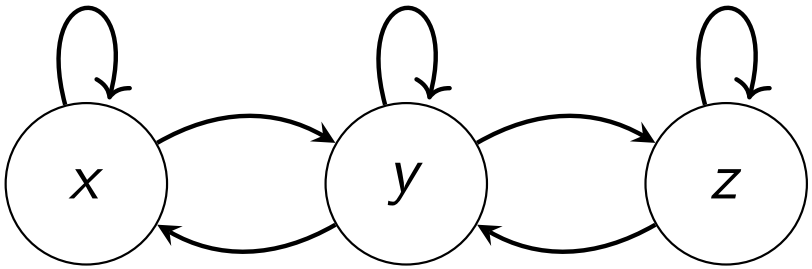


Transitive: $\forall x, y, z \in S,\ x\ \mathcal{R}\ y$ and $y\ \mathcal{R}\ z$ implies $x\ \mathcal{R}\ z$.



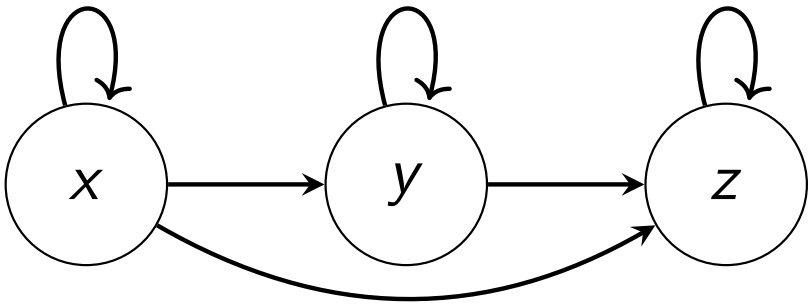
HW#1 Problem 1

(a) Reflexive and symmetric but not transitive



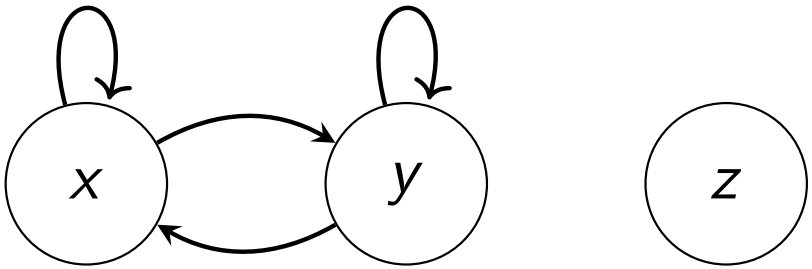
HW#1 Problem 1

(b) Reflexive and transitive but not symmetric



HW#1 Problem 1

(c) Symmetric and transitive but not reflexive



HW#1 Problem 2

2. (20 points) For each part, determine whether the binary relation on the set of integers or reals is an equivalence relation. If it is, please provide a proof; otherwise, please give a counterexample.

   (a) For a fixed non-zero divisor, the two numbers have the same remainder. (Note: for instance, suppose 2 is the divisor. Numbers 4 and 6 have the same remainder, while 4 and 5 do not.)

   (b) The two real numbers are approximately equal. Note: it is up to you to define the notion of "approximately equal" more precisely, but it must not be the same as exactly equal.

HW#1 Problem 2

A binary relation $\mathcal{R}$ on a set S is an equivalence relation if

- $\mathcal{R}$ is reflexive: $\forall x \in S$, $x \mathcal{R} x$,
- $\mathcal{R}$ is symmetric: $\forall x, y \in S$, $x \mathcal{R} y$ iff $y \mathcal{R} x$, and
- $\mathcal{R}$ is transitive: $\forall x, y, z \in S$, $x \mathcal{R} y$ and $y \mathcal{R} z$ implies $x \mathcal{R} z$.

HW#1 Problem 2

(a) $\mathcal{R}$: For a fixed non-zero divisor d, the two numbers have the same remainder r.

- Reflexive: satisfied, $\forall x \in \mathbb{N}$, for a fixed non-zero divisor, x has same remainder with itself.
- Symmetric: satisfied, $\forall x, y \in \mathbb{N}$, x, y have same remainder implies y, x have same remainder.
- Transitive: satisfied,$\forall x, y, z \in \mathbb{N}$. If x, y have the same remainder r, and y, z have the same remainder, the remainder of z is also r. Therefore x, z have same remainder.

So, $\mathcal{R}$ is an equivalence relation.

HW#1 Problem 2

(b) $\mathcal{R}$: The two real numbers are approximately equal.
Suppose we define that two real numbers x and y are approximately equal if $|x - y| \leq 0.1$

- Reflexive: satisfied, $\forall x \in \mathbb{R}$, $|x - x| = 0 \leq 0.1$
- Symmetric: satisfied, $\forall x, y \in \mathbb{R}$, if $|x - y| \leq 0.1$, then $|y - x| = |x - y| \leq 0.1$
- Transitive: violated, counterexample: $|0.1 - 0.2| \leq 0.1$, $|0.2 - 0.3| \leq 0.1$, but $|0.1 - 0.3| = 0.2 > 0.1$

So, $\mathcal{R}$ is not an equivalence relation.

HW#1 Problem 3

(20 points) In class, following Sipser's book, we first studied the formal definition of a function and then treated relations as special cases of functions. Please give instead a direct definition of relations and then define functions as special cases of relations. Your definitions should cover the arity of a relation or function and also the meaning of the notation $f(a) = b$.

HW#1 Problem 3

- A relation $\mathcal{R}$ is a subset of the Cartesian product of several sets.
- A relation $\mathcal{R} \subseteq A_1 \times A_2 \times \cdots \times A_k$ is called a *k-ary* relation.
- A 2-*ary* relation is usually called a binary relation.

HW#1 Problem 3

- A function is a binary relation that follows the form $f \subseteq (A_1 \times \cdots \times A_k) \times B$, namely the element of a function is a pair, and the first element of the pair is also a k-tuple.
- For all $a \in (A_1 \times \cdots \times A_k)$, exists $b \in B$ such that $(a, b) \in f$, where a is the domain of the function and b is the range.
- For all $a_i, a_j \in (A_1 \times \cdots \times A_k)$, $b_i, b_j \in B$.
  If $(a_i, b_i) \in f$, $(a_j, b_j) \in f$ and $a_i = a_j$, then $b_i = b_j$.
- A function with a *k-ary* relation as its first component of the pair is called a *k-ary* function.
- We write $f(a) = b$ if $(a, b) \in f$. Similarly, we write $f(a_1, a_2, \cdots, a_k) = b$ if $((a_1, a_2, \cdots, a_k), b) \in f$

HW#1 Problem 4

(Problem 0.10; 20 points) Show that every graph having two or more nodes contains two nodes with the same degree. (Note: we assume that every graph is simple and finite, unless explicitly stated otherwise.)

HW#1 Problem 4

Note: Self loop is not allowed in a simple graph.

Proved by contradiction.

Supposed that there is a graph with n nodes having no nodes with the same degree.

No nodes with the same degree means that : each node has distinct degree from 0 to $n-1$.
the node with $n-1$ degree must be connected by every other nodes in this graph because no self loop in this gragh.
but in our assumption, there must be a node with 0 degree.
Contradiction happens.

HW#1 Problem 5

(10 points) Consider a round-robin tournament among n players. In the tournament, each player plays once against all other $n-1$ players. There are no draws, i.e., for a match between p and p', the result is either p beat p' or p' beat p. Prove *by induction* that, after a round-robin tournament, it is always possible to arrange the n players in an order $p_1, p_2, \ldots, p_n$ such that p_1 beat p_2, p_2 beat p_3, $\cdots$, and p_{n-1} beat p_n. (Note: the "beat" relation, unlike "$\geq$", is not transitive.)

HW#1 Problem 5

The proof is by induction on the number n of players.
Base case ($n = 2$): There are exactly two players, say A and B. Either A beat B, in which case we order them as A, B, or B beat A, in which case we order them as B, A.

HW#1 Problem 5

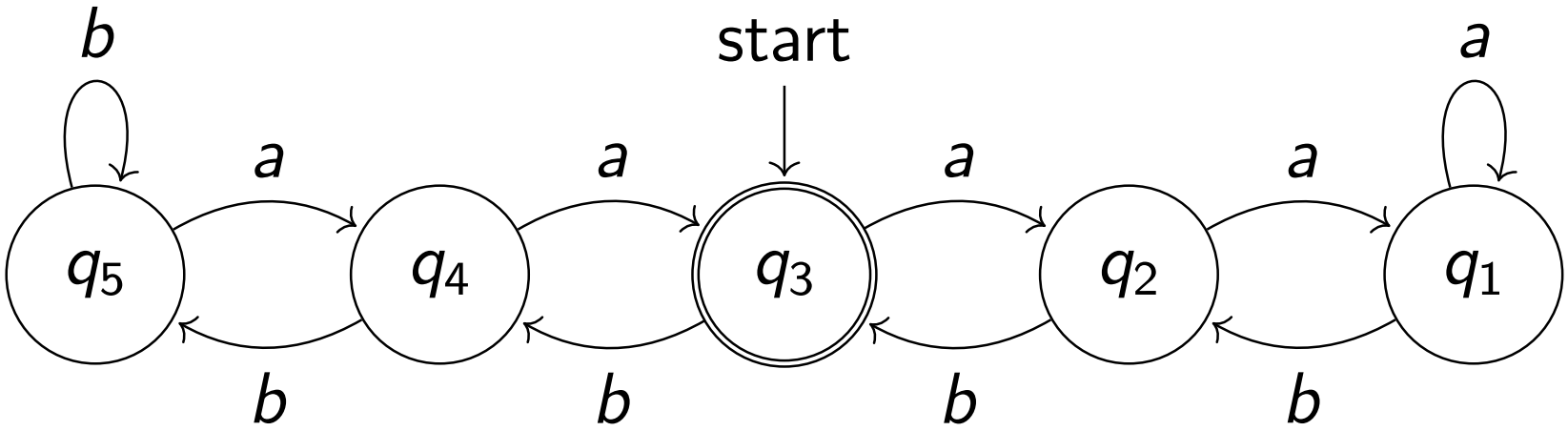
Induction step ($n > 2$): Pick any of the n players, say a. From the induction hypothesis, the other $n-1$ players can be ordered as $p_1, p_2, ..., p_{n-1}$ such that p_1 beat p_2, p_2 beat p_3, ..., and p_{n-2} beat p_{n-1}. We now exam the result of the match played between a and p_1. If a beat p_1, then we get a satisfying order a, $p_1, p_2, ..., p_{n-1}$. Otherwise (p_1 beat a), we continue to exam the result of the match played between a and p_2. If a beat p_2, then we get a satisfying order p_1, a, $p_2, ..., p_{n-1}$. Otherwise (p_2 beat a), we continue as before. We end up either with $p_1, p_2, ..., p_{i-1}$, a,$p_i, ..., p_{n-1}$ for some $i \leq n-1$ or eventually with $p_1, p_2, ..., p_{n-1}$, a if a is beaten by every other player, in particular p_{n-1}.

HW#2 Problem 1

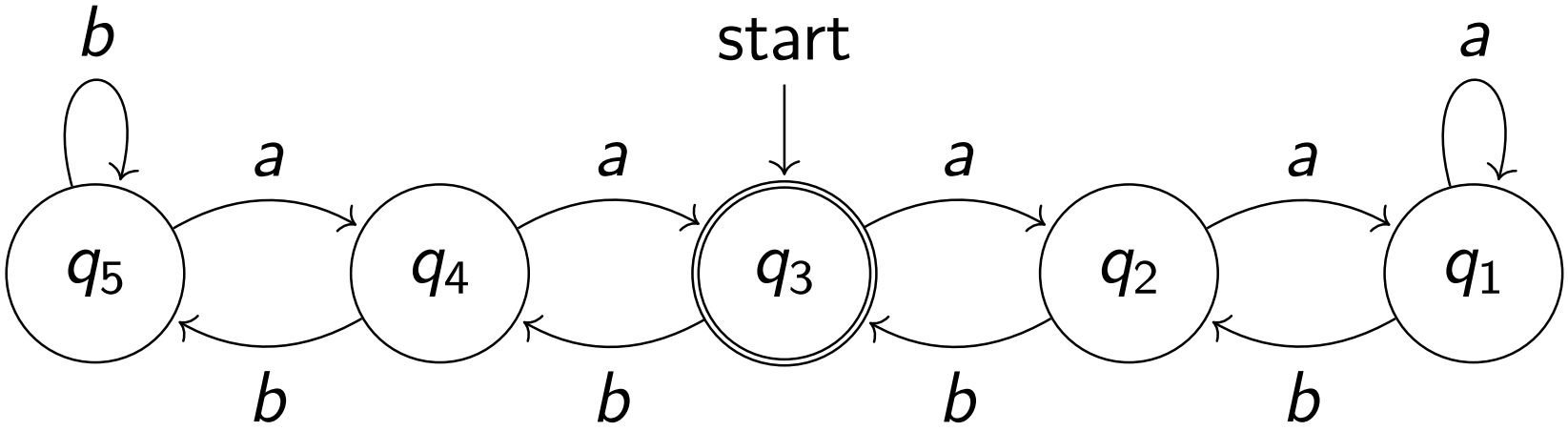
(Exercise 1.3 adapted; 10 points) The formal definition of a DFA M is $(\{q_1, q_2, q_3, q_4, q_5\}, \{\texttt{a}, \texttt{b}\}, \delta, q_3, \{q_3\})$ where δ is given by the following table. Draw the state diagram of M and give an intuitive characterization of the strings that M accepts.

	a	b
q_1	q_1	q_2
q_2	q_1	q_3
q_3	q_2	q_4
q_4	q_3	q_5
q_5	q_4	q_5

HW#2 Problem 1



HW#2 Problem 1



Intuitive characterization of the strings that M accepts:

Let $x := 0$.
$x := x + 1$ when M reads a, $x := x - 1$ when M reads b. The value of x should be in $[-2, 2]$ (which means $-2 - 1 = -2$ and $2 + 1 = 2$).
M accepts if the final value of $x = 0$.

HW#2 Problem 2

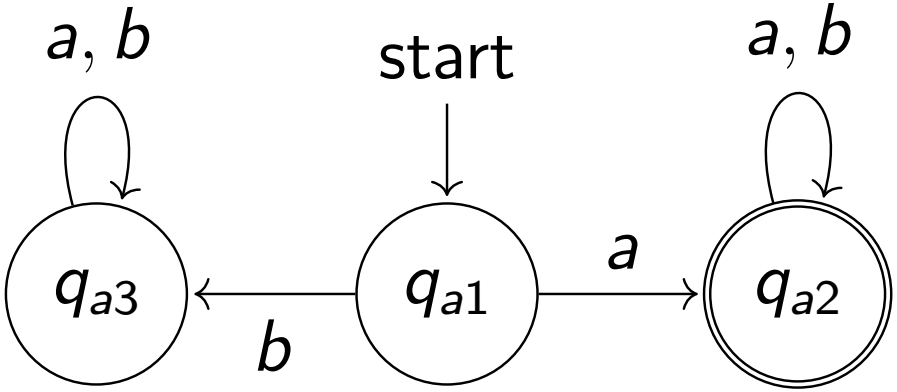
(Exercise 1.4; 20 points) Each of the following languages is the intersection of two simpler regular languages. In each part, construct DFAs for the simpler languages, then combine them using the construction discussed in class (see also Footnote 3 in Page 46 of [Sipser 2006, 2013]) to give the state diagram of a DFA for the language given. In all parts, the alphabet is $\{\mathtt{a}, \mathtt{b}\}$.

(a) $\{w \mid w$ starts with an $\mathtt{a}$ and has at most one $\mathtt{b}\}$.

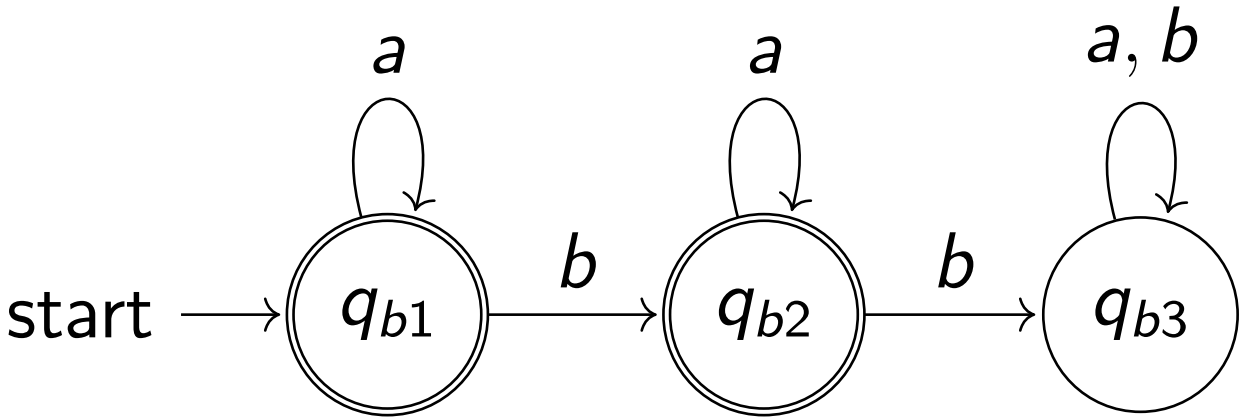
(b) $\{w \mid w$ has an odd number of $\mathtt{a}$'s and ends with a $\mathtt{b}\}$.

HW#2 Problem 2 (a)

Simpler language: $\{w \mid w \text{ starts with an } a\}$.

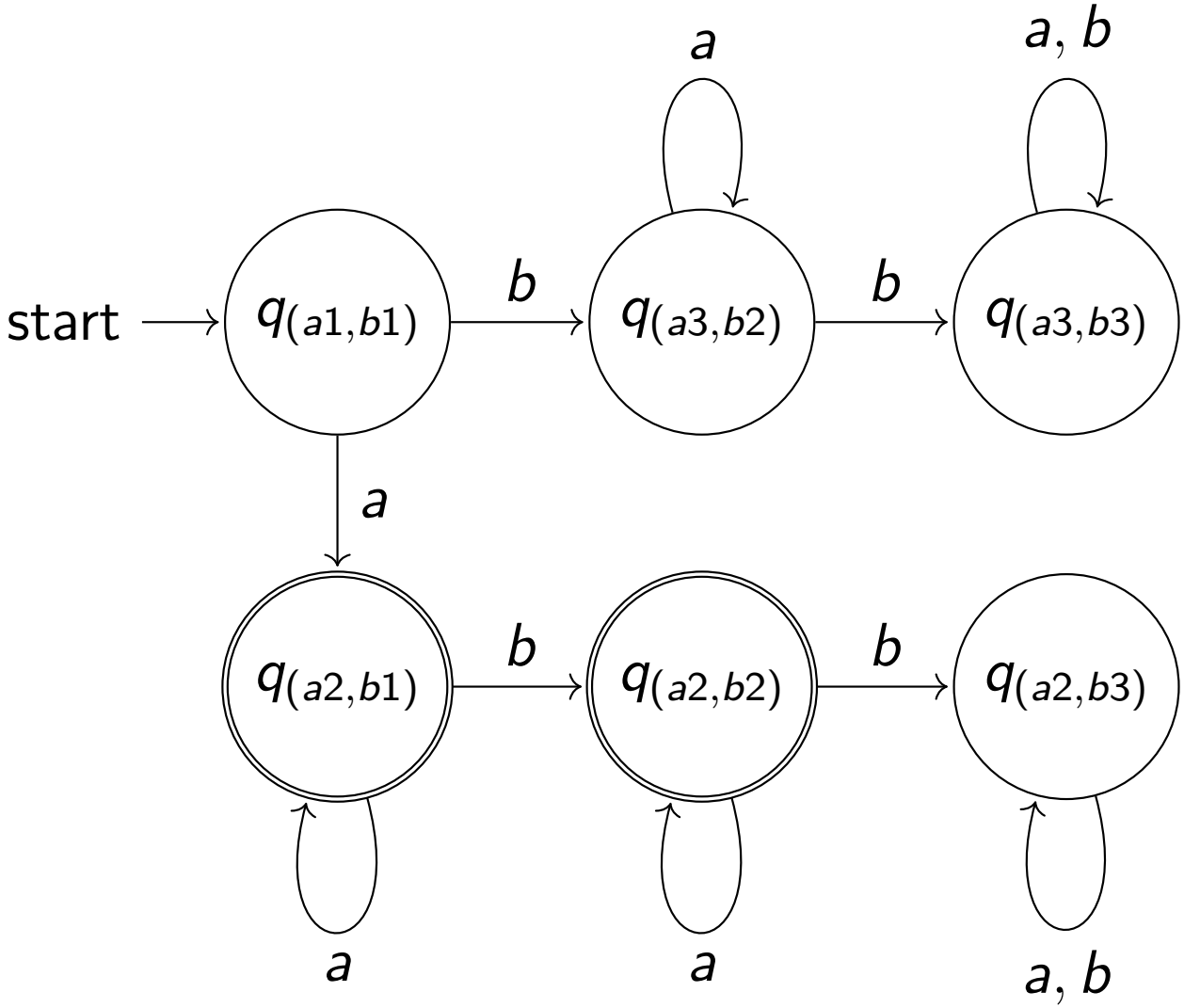


Simpler language: $\{w \mid w \text{ has at most one } b\}$.



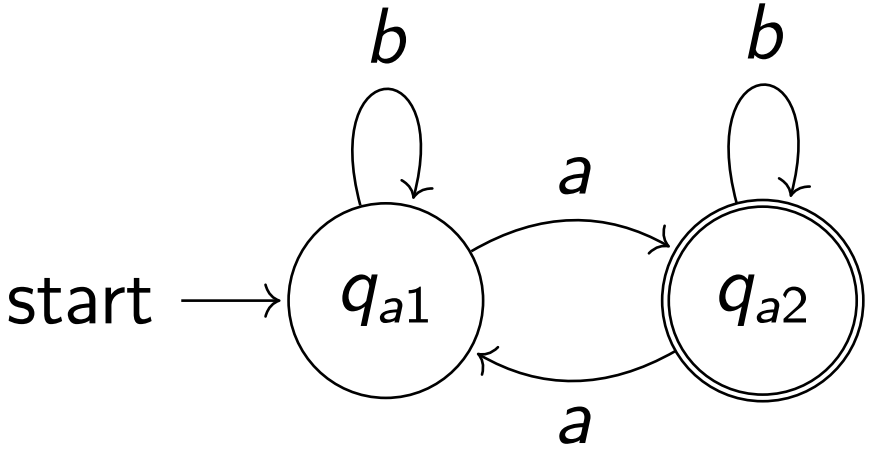
HW#2 Problem 2 (a)

Language: $\{w \mid w \text{ starts with an } a \text{ and has at most one } b\}$.

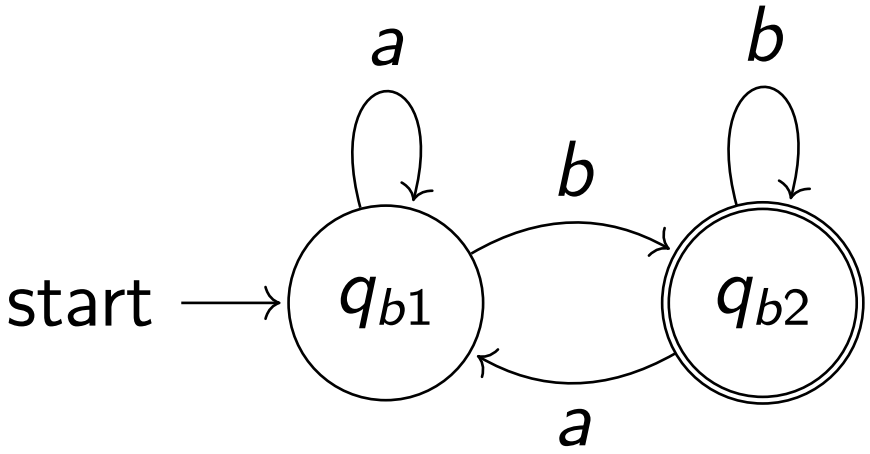


HW#2 Problem 2 (b)

Simpler language: $\{w \mid w \text{ has an odd number of } a\text{'s}\}$.

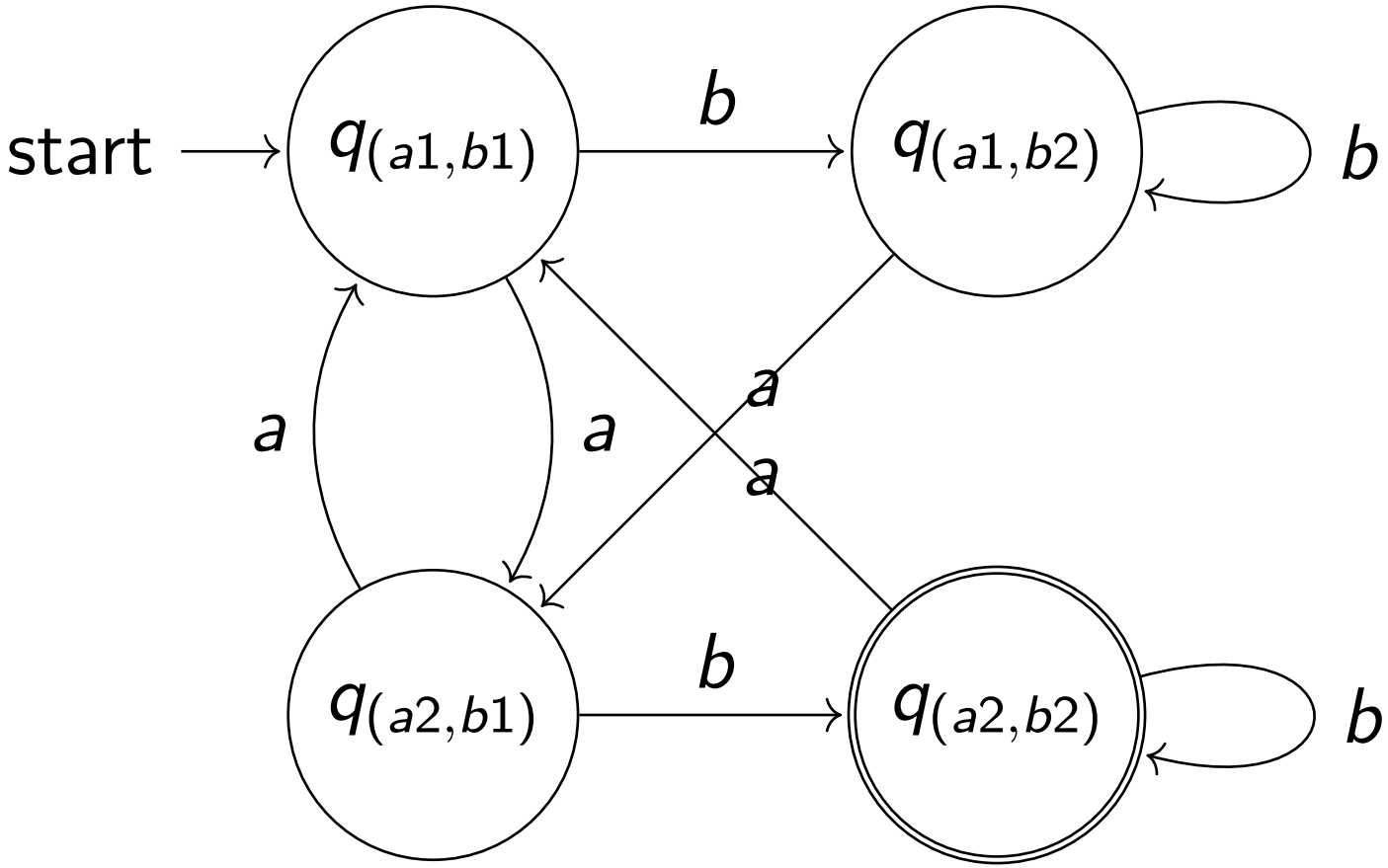


Simpler language: $\{w \mid w \text{ ends with a } b\}$.



HW#2 Problem 2 (b)

Language: $\{w \mid w$ has an odd number of a's and ends with a $b\}$.



HW#2 Problem 3

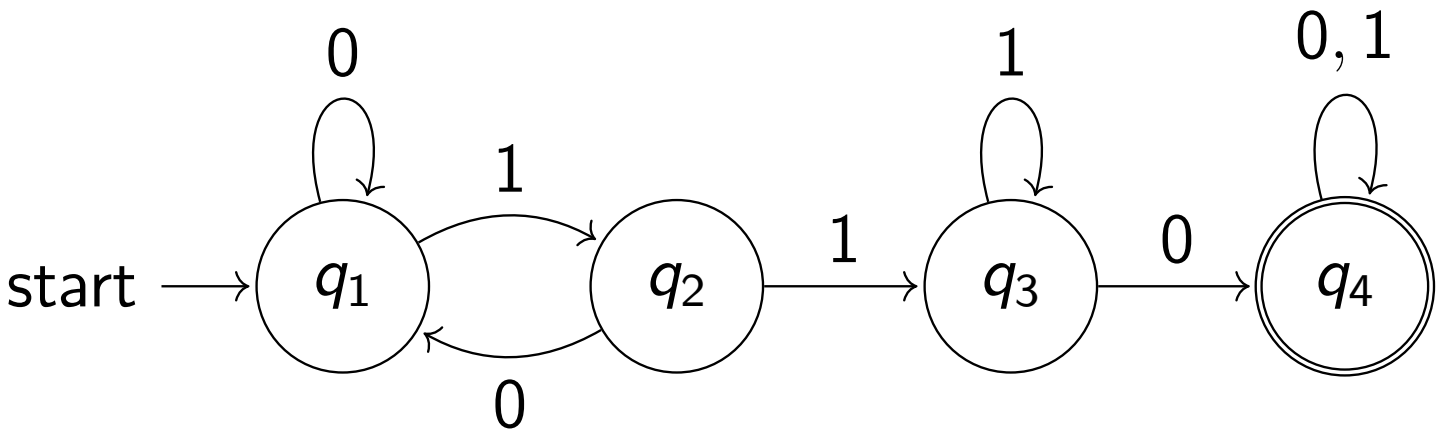
(Exercise 1.6; 20 points) Give state diagrams of DFAs recognizing the following languages. In all parts, the alphabet is $\{0, 1\}$.

(a) $\{w \mid w$ doesn't contain the substring $110\}$.

(b) $\{w \mid$ every odd position of w is a $1\}$ (Note: see w as $w_1 w_2 \cdots w_n$, where $w_i \in \{0, 1\}$).

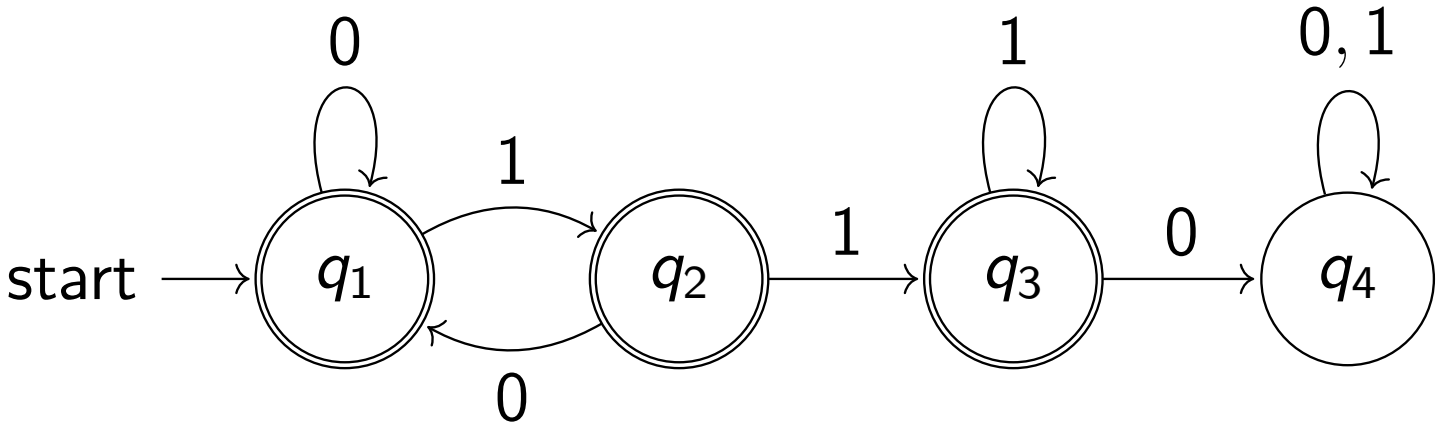
HW#2 Problem 3 (a)

Simpler language: $\{w \mid w \text{ contains the substring } 110\}$.



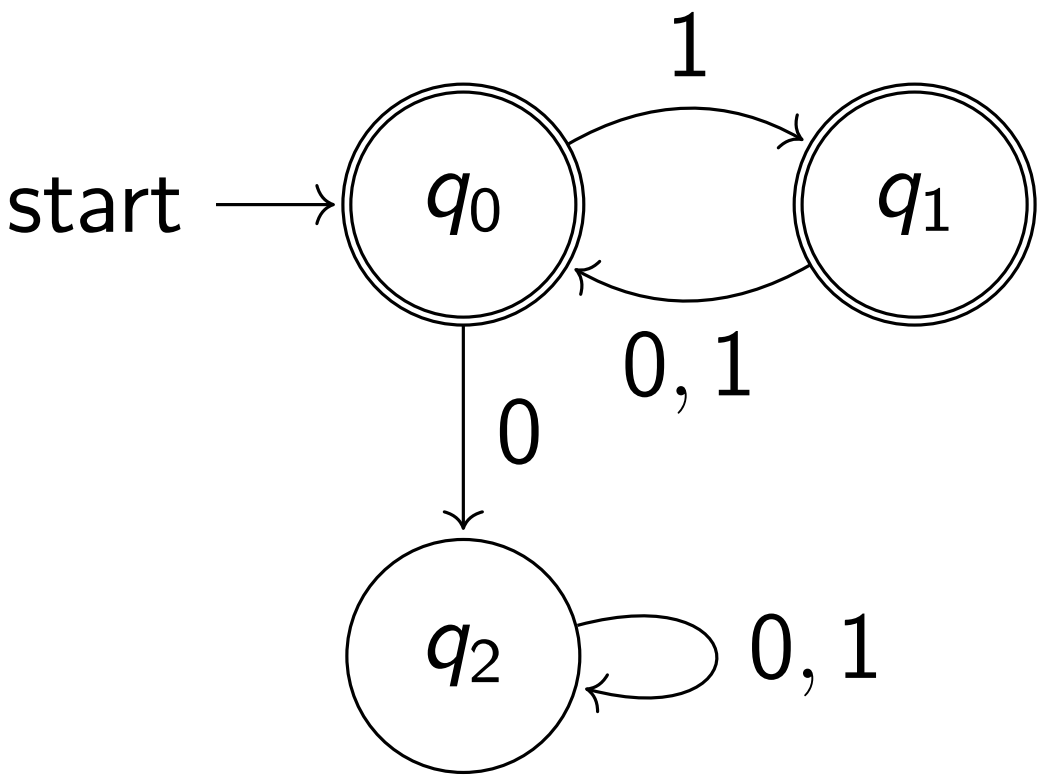
HW#2 Problem 3 (a)

Language: $\{w \mid w \text{ doesn't contain the substring } 110\}$.



HW#2 Problem 3 (b)

Language: $\{w \mid \text{every odd position of } w \text{ is a } 1\}$.



HW#2 Problem 4

(Problem 1.36; 10 points) For any string $w = w_1 w_2 \cdots w_n$, the *reverse* of w, written w^R, is the string w in reverse order, $w_n \cdots w_2 w_1$. For any language A, let $A^R = \{w^R \mid w \in A\}$. Show that if A is regular, so is A^R.

HW#2 Problem 4

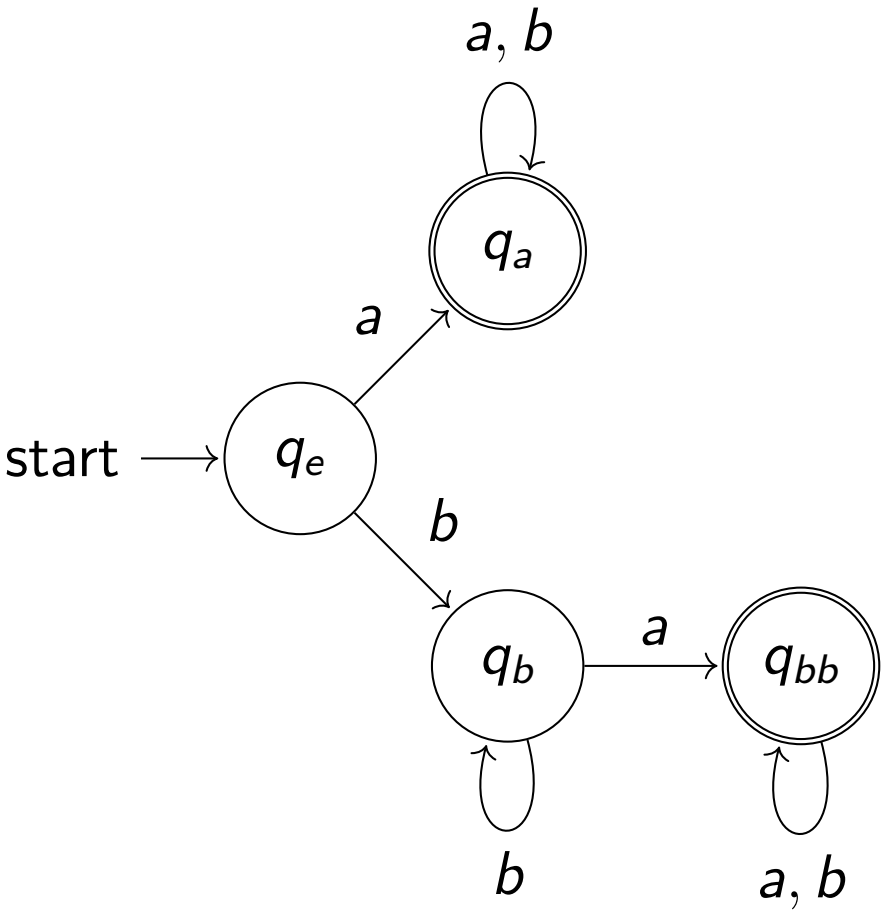
Let DFA M recognizes the language A, and we can construct a NFA M^R which recognizes A^R according to the following:

- M^R conserves all states from M and M^R's alphabet is as same as M.
- Reverse all the transitions of M as the transitions of M^R. e.q. $\delta(q_1, a) = q_2 \rightarrow \delta^R(q_2, a) = q_1$.
- Add an additional initial state q_0 to M^R. Construct the translations from q_0 to all the accepting states of M with the label ϵ.
- The accepting state of M^R is M's initial state.

For any string w, M accept w iff M^R recognize w^R.
Because M^R recognizes A^R, A^R is regular.

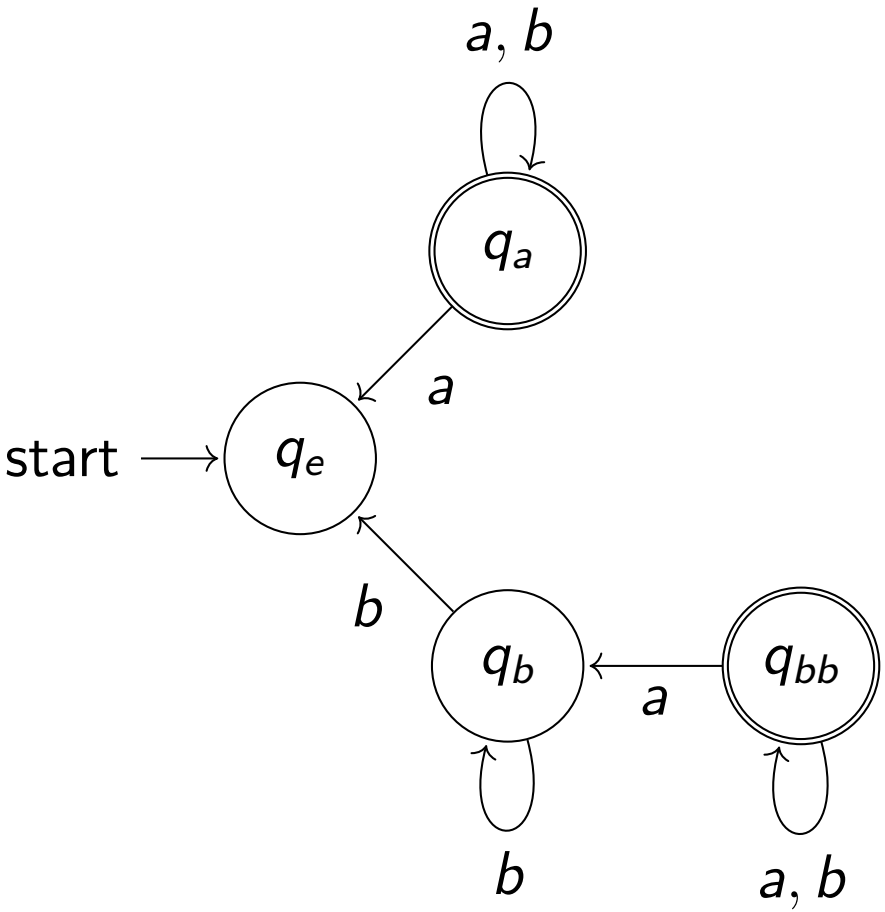
HW#2 Problem 4

e.q. : M



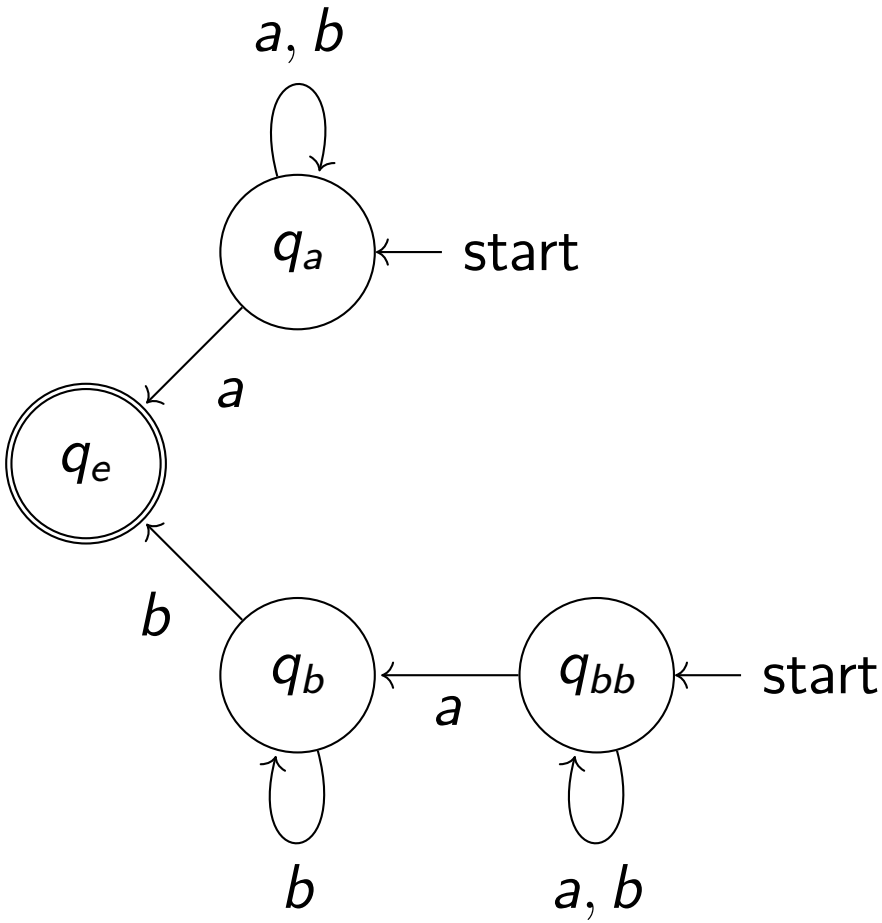
HW#2 Problem 4

Reverse all the transitions:



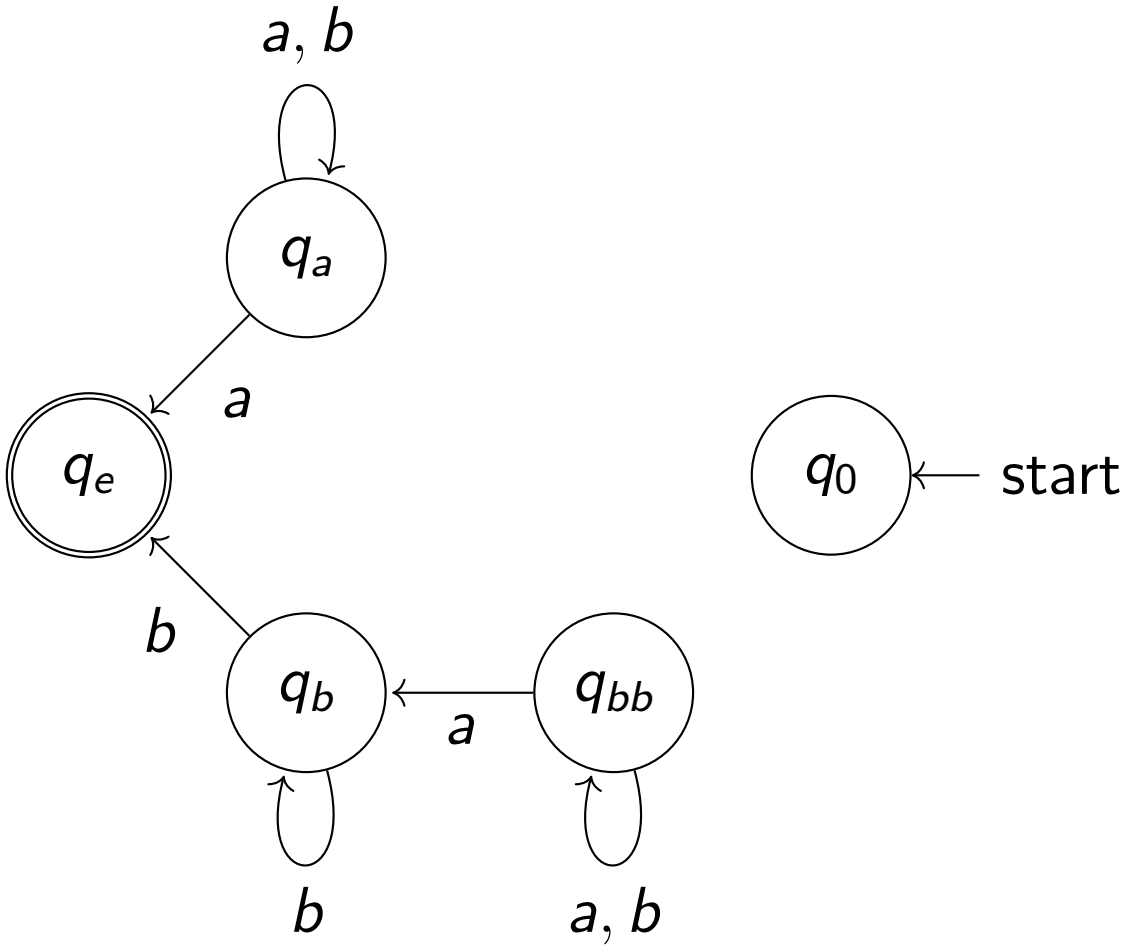
HW#2 Problem 4

Change the initial state into accepting state:



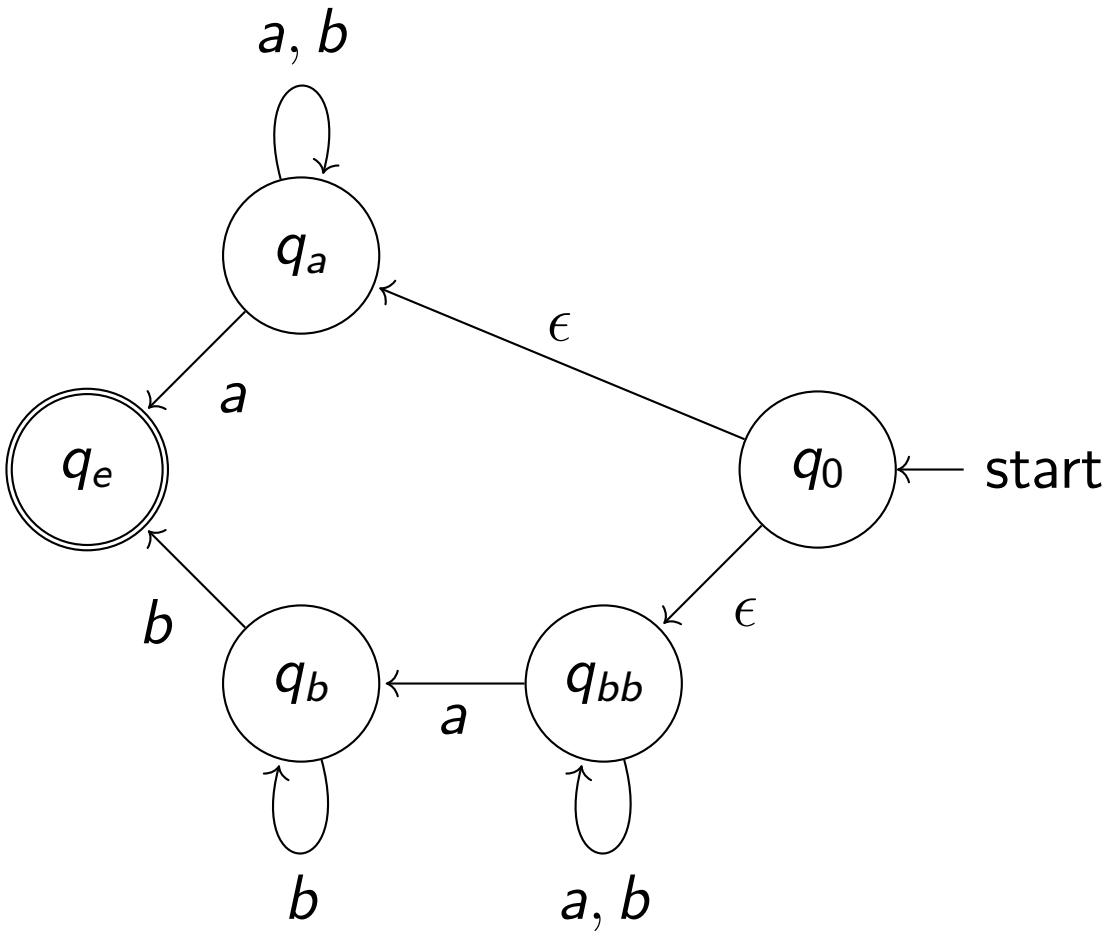
HW#2 Problem 4

Add an additional initial state q_0:



HW#2 Problem 4

Construct the translations from q_0 to all the accepting states of M with the label ϵ, then we can get the NFA M^R:



HW#2 Problem 5

(Problem 1.37; 20 points) Let

$$\Sigma_3 = \left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \cdots, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\}.$$

Σ_3 contains all size 3 columns of 0s and 1s. A string of symbols in Σ_3 gives three rows of 0s and 1s. Consider each row to be a binary number and let

$$B = \{w \in \Sigma_3^* \mid \text{the bottom row of } w \text{ is the sum of the top two rows}\}.$$

For example,

$$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \in B, \text{but } \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \notin B.$$

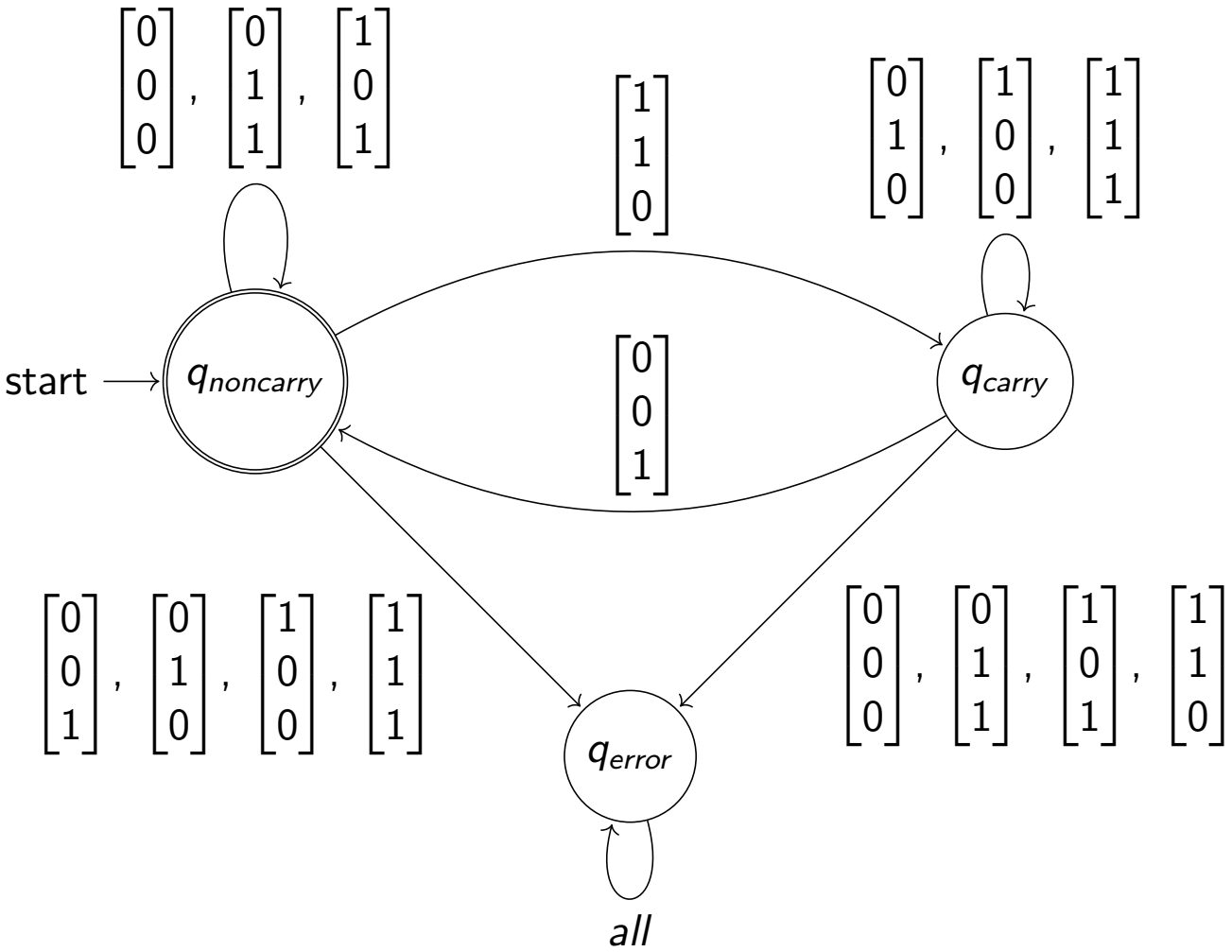
Show that B is regular. (Hint: working with B^R is easier. You may assume the result claimed in the previous problem (Problem 1.36).)

HW#2 Problem 5

Consider the situation of carry, starting from the tail of B is easier than starting from the head. So we first show that B^R is regular. We can construct a DFA that recognizes B^R when considering the carry and the correctness of calculation.

HW#2 Problem 5

The DFA that recognizes B^R:



HW#2 Problem 5

Because there is a DFA that recognizes B^R, B^R is regular. According to the result claimed in Problem 4 (if A is regular, so is A^R), we can say that $(B^R)^R = B$ is regular.

HW#2 Problem 6

(20 points) Generalize the proof of Theorem 1.25 of [Sipser 2006, 2013] (Pages 45–47) to handle A_1 and A_2 with different alphabets.

HW#2 Problem 6

Suppose $M_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$ recognizes A_1 and $M_2 = (Q_2, \Sigma_2, \delta_2, q_2, F_2)$ recognizes A_2.
Construct $M = (Q, \Sigma, \delta, q_0, F)$ to recognize $A_1 \cup A_2$:

- $Q = (Q_1 \cup \{q_f\}) \times (Q_2 \cup \{q_f\})$.
- $\Sigma = \Sigma_1 \cup \Sigma_2$.
- $\delta((r_1, r_2), a) =$

$$\begin{cases} (\delta_1(r_1, a), \delta_2(r_2, a)) & \text{if } (r_1, r_2 \neq q_f) \text{ and } (a \in (\Sigma_1 \cap \Sigma_2)) \\ (\delta_1(r_1, a), q_f) & \text{if } (r_1 \neq q_f \wedge a \in \Sigma_1) \text{ and } (r_2 = q_f \vee a \notin \Sigma_2) \\ (q_f, \delta_2(r_2, a)) & \text{if } (r_2 \neq q_f \wedge a \in \Sigma_2) \text{ and } (r_1 = q_f \vee a \notin \Sigma_1) \\ (q_f, q_f) & \text{if } (r_1 = q_f \vee a \notin \Sigma_1) \text{ and } (r_2 = q_f \vee a \notin \Sigma_2) \end{cases}$$

- $q_0 = (q_1, q_2)$.
- $F = \{(r_1, r_2) \mid r_1 \in F_1 \text{ or } r_2 \in F_2\}$.

HW#2 Problem 6

Why we need q_f?

Because when we read a character a that in Σ_1 but not in Σ_2, A_2 cannot recognize a so M_2 must fail and never accept. If there's no q_f, M cannot find out this situation.