Exercise 11.1
Construct the intersection of the two following Büchi automata:

\[ \begin{array}{c}
\text{A:} & b & c & a & b \\
\dot{p} & a & q & v & r \\
\dot{s} & a, c & t \\
\end{array} \]

Exercise 11.2
Consider the following Büchi automaton \( B \) over \( \Sigma = \{a, b\} \):

\[ \begin{array}{c}
\dot{q}_0 & \overset{a, b}{\longrightarrow} & \dot{q}_1 \\
\dot{q}_1 & \overset{b}{\longrightarrow} & \dot{q}_0 \\
\end{array} \]

(a) Sketch \( \text{dag}(abab^\omega) \) and \( \text{dag}((ab)^\omega) \).
(b) Let \( r_w \) be the ranking of \( \text{dag}(w) \) defined by

\[ r_w(q, i) = \begin{cases} 
1 & \text{if } q = q_0 \text{ and } (q_0, i) \text{ appears in } \text{dag}(w), \\
0 & \text{if } q = q_1 \text{ and } (q_1, i) \text{ appears in } \text{dag}(w), \\
\perp & \text{otherwise.} 
\end{cases} \]

Are \( r_{abab^\omega} \) and \( r_{(ab)^\omega} \) odd rankings?
(c) Show that \( r_w \) is an odd ranking if and only if \( w \not\in L_\omega(B) \).
(d) Construct a Büchi automaton accepting \( \overline{L_\omega(B)} \) using the construction seen in class. Hint: by (c), it is sufficient to use \{0, 1\} as ranks.
Exercise 11.3
Show that for every DBA $A$ with $n$ states there is an NBA $B$ with $2n$ states such that $B = \overline{A}$. Explain why your construction does not work for NBAs.

Exercise 11.4
Give Büchi automata for the following ω-languages:

- $L_1 = \{ w \in \{a,b\}^\omega : w$ contains infinitely many $a$’s $\}$,
- $L_2 = \{ w \in \{a,b\}^\omega : w$ contains finitely many $b$’s $\}$,
- $L_3 = \{ w \in \{a,b\}^\omega :$ each occurrence of $a$ in $w$ is followed by a $b$ $\}$,

and intersect these automata. Decide if this automaton is the smallest Büchi automaton for that language.
Solution 11.1

Solution 11.2

(a) $\text{dag}(abab^\omega)$:
dag((ab)\omega):

- r is not an odd rank for dag(abab\omega) since
  \langle q_0, 0 \rangle \xrightarrow{a} \langle q_0, 1 \rangle \xrightarrow{b} \langle q_0, 2 \rangle \xrightarrow{a} \langle q_0, 3 \rangle \xrightarrow{b} \langle q_1, 4 \rangle \xrightarrow{b} \langle q_1, 5 \rangle \xrightarrow{b} \cdots
  is an infinite path of dag(abab\omega) not visiting odd nodes infinitely often.

- r is an odd rank for dag((ab)\omega) since it has a single infinite path:
  \langle q_0, 0 \rangle \xrightarrow{a} \langle q_0, 1 \rangle \xrightarrow{b} \langle q_0, 2 \rangle \xrightarrow{a} \langle q_0, 3 \rangle \xrightarrow{b} \langle q_0, 4 \rangle \xrightarrow{b} \langle q_1, 5 \rangle \xrightarrow{b} \cdots
  which only visits odd nodes.

(c) \Rightarrow Let w \in L_\omega(B). We have w = ub\omega for some u \in \{a, b\}^*.
This implies that
  \langle q_0, 0 \rangle \xrightarrow{u} \langle q_0, |u| \rangle \xrightarrow{b} \langle q_1, |u| + 1 \rangle \xrightarrow{b} \langle q_1, |u| + 2 \rangle \xrightarrow{b} \cdots
  is an infinite path of dag(w). Since this path does not visit odd nodes infinitely often, r is not odd for dag(w).

\Leftarrow Let w \not\in L_\omega(B). Suppose there exists an infinite path of dag(w) that does not visit odd nodes infinitely often. At some point, this path must only visit nodes of the form \langle q_1, i \rangle. Therefore, there exists u \in \{a, b\}^* such that
  \langle q_0, 0 \rangle \xrightarrow{u} \langle q_1, |u| \rangle \xrightarrow{b} \langle q_1, |u| + 1 \rangle \xrightarrow{b} \langle q_1, |u| + 2 \rangle \xrightarrow{b} \cdots
  This implies that w = ub\omega \in L_\omega(B) which is contradiction.

(d) Recall that we construct an NBA with an infinite number of states whose runs on an \omega-word w are the rankings of dag(w). The automaton accepts a ranking R if and only if every infinite path of R visits nodes of odd rank i.o. By (c), for every w \in \{a, b\}^\omega, if dag(w) has an odd ranking, then it has one ranging over 0 and 1. Therefore, it suffices to execute CompNBA with rankings ranging over 0 and 1 (and our NBA is now finite). We obtain the following Büchi automaton, for which some intuition is given below:
Any ranking \( r \) of \( \text{dag}(w) \) can be decomposed into a sequence \( lr_1, lr_2, \ldots \) such that \( lr_i(q) = r(q, i) \), the level \( i \) of rank \( r \). Recall that in this automaton, the transitions \( \delta(lr(q_0)), lr(q_1) \rightarrow \delta(lr'(q_0)), lr'(q_1) \) represent the possible next level for ranks \( r \) such that \( lr(q) = r(q, i) \) and \( lr'(q) = r(q, i + 1) \) for \( q = q_0, q_1 \).

The additional set of states in the automaton represents the set of states that “owe” a visit to a state of odd rank. Formally, the transitions are the triples \( \delta(lr, O) \rightarrow \delta(lr', O') \) such that \( lr \rightarrow lr' \) and \( O' = \{ q' \in Q | lr'(q') \text{ is even} \} \) if \( O \neq \emptyset \), and \( O' = \{ q' \in Q | lr'(q') \text{ is even} \} \) if \( O = \emptyset \).

Finally the accepting states of the automaton are those with no “owing” states, which represent the breakpoints, i.e. a moment where we are sure that all runs on \( w \) have seen an odd rank since the last breakpoint.

★ It is enough to only consider the blue states, as any other state cannot reach a level in which there is an odd rank; descendants of \( \text{dag} \) states with rank 0 can never be assigned an odd rank.

**Solution 11.3**

Observe that \( A \) rejects a word \( w \) iff its single run on \( w \) stops visiting accepting states at some point. Hence, we construct an NBA \( B \) that reads a prefix as in \( A \) and non deterministically decides to stop visiting accepting states by moving to a copy of \( A \) without its accepting states.

More precisely, we assume that each letter can be read from each state of \( A \), i.e. that \( A \) is complete. If this is not the case, it suffices to add a rejecting sink state to \( A \). The NBA \( B \) consists of two copies of \( A \). The first copy is exactly as \( A \). The second copy is as \( A \) but restricted to its non accepting states. We add transitions from the first copy to the second one as follows. For each transition \( (p, a, q) \) of \( A \), we add a transition that reads letter \( a \) from state \( p \) of the first copy to state \( q \) of the second copy. All states of the first copy are made non accepting and all states of the second copy are made accepting. Note that \( B \) contains at most \( 2n \) states as desired.

Here is an example of the construction:

This construction does not work on NBAs. Indeed, we have \( A = B = \{ a^\omega \} \) below:
Solution 11.4
The following Büchi automata respectively accept $L_1$, $L_2$ and $L_3$:

\[
\begin{align*}
\text{p}_0 &\xrightarrow{b} \text{p}_1 & \text{q}_0 &\xrightarrow{a} \text{q}_1 & \text{r}_0 &\xrightarrow{a} \text{r}_1 \\
\text{p}_1 &\xrightarrow{a} \text{p}_0 & \text{q}_1 &\xrightarrow{a} \text{q}_0 & \text{r}_1 &\xrightarrow{b} \text{r}_0 \\
\end{align*}
\]

Taking the intersection of these automata leads to the following Büchi automaton:

\[
\begin{align*}
\text{p}_0, \text{q}_0, \text{r}_0 &\xrightarrow{a} \text{p}_1, \text{q}_0, \text{r}_1 & \text{p}_1, \text{q}_0, \text{r}_1 &\xrightarrow{a} \text{p}_1, \text{q}_0, \text{r}_1 \\
\text{p}_1, \text{q}_0, \text{r}_0 &\xrightarrow{b} \text{p}_1, \text{q}_0, \text{r}_1 & \text{p}_1, \text{q}_0, \text{r}_1 &\xrightarrow{b} \text{p}_1, \text{q}_0, \text{r}_1 \\
\end{align*}
\]

★ Note that the language of this automaton is the empty language.