1. (30 points) Use the procedure described in class to construct the machine \( M_3 \) that corresponds to the “product” of machines \( M_1 \) and \( M_2 \). In other words, \( L(M_3) = L(M_1) \cap L(M_2) \).

\( M_1 \) accepts the strings that do not contain \( 'aa' \).

\( M_2 \) accepts the strings that end with \( 'ab' \).

Test all three machines with the four strings \( aa, ab, aba, aab \) and state whether they are accepted.

2. (30 points) Use Theorem 5.5.3 and Example 6.1.1 to convert the following regular expression into an NFA-\( \lambda \). Apply the full steps for converting a regular expression to an NFA-\( \lambda \). Do not simplify the machine by removing \( \lambda \) transitions or making other changes. Do not construct the machine “directly”. For your convenience, it is acceptable to label machines corresponding to segments of the regular expression and use them in subsequent drawings (see class examples).

\[(a \cup b)^* b b a^* b^*\]

3. (15+25 points) Let \( M_1 \) be the following NFA:

(a) Give the transition function \( t \) for \( M_1 \) in tabular form.

(b) Use algorithm 5.6.3 to construct a state diagram of a DFA that is equivalent to \( M_1 \). Give the transition function and draw the state diagram of the equivalent DFA.