Equivalence of NFAs and DFAs

DFAs and NFAs are equivalent; that is, they recognize the same set of languages, regular languages. To prove their equivalence, it is necessary and sufficient to show that there exists an NFA that recognizes the language of a DFA, and vice versa. The proof of the former statement:

Theorem (DFA to NFA)**.**
For every DFA, there exists an NFA that recognizes the language it recognizes.

is

Proof. Let the DFA $M = (Q, \Sigma, \delta, q_0, F)$ recognize the language L. Then the NFA $N = (Q', \Sigma', \delta', q_0', F')$ where

1. $Q' = Q$,
2. $\Sigma' = \Sigma$,
3. $\delta' : Q' \times \Sigma'_\varepsilon \longrightarrow \wp(Q')$
$$\delta'(q, a) = \begin{cases} \emptyset & \text{if } a = \varepsilon \\ \{\delta(q, a)\} & \text{if } a \neq \varepsilon \end{cases}$$
4. $q_0' = q_0$
5. $F' = F$

recognizes L as required. □

Meanwhile, the proof of the latter statement:

Theorem (NFA to DFA)**.**
For every NFA, there exists a DFA that recognizes the language it recognizes.

Proof. Let the NFA $N = (Q, \Sigma, \delta, q_0, F)$ recognize the language L, and

$$E(R) = \bigcup_{r \in R} e(r) e(q) \qquad = \begin{cases} \{q\} & \text{if } \delta(q, \varepsilon) = \emptyset \\ \{q\} \cup E(\delta(q, \varepsilon)) & \text{if } \delta(q, \varepsilon) \neq \emptyset \end{cases}$$

where $q \in Q$ and $R \subseteq Q$, then the DFA $M = (Q', \Sigma', \delta', q_0', F')$ where

1. $Q' = \wp(Q)$,
2. $\Sigma' = \Sigma$,
3. $\delta' : Q' \times \Sigma \longrightarrow Q'$
$$\delta(R, a) = \bigcup_{r \in R} E(\delta(r, a))$$
4. $q_0' = E(\{q_0\})$
5. $F' = \{R : R \in Q' \wedge R \cap F \neq \emptyset\}$

recognizes L as required. The DFA can remove any state in Q' of M that does not have a path to it from q_0'; these states are unreachable, and thus, unnecessary. □

The following algorithm will allow the construction of the equivalent DFA of an NFA that omits unreachable states of $\wp(Q)$:

```
DFABUILD($N = (Q, \Sigma, \delta, q_0, F)$)
01. $q'_0 \leftarrow E(q_0)$
02. $\Sigma' \leftarrow \Sigma$
03. $Q' \leftarrow \{q'_0\}$
04. $C \leftarrow Q'$
05. $N \leftarrow \emptyset$
06. foreach $q$ in $C$, do
  01. foreach $a$ in $\Sigma'$, do
    01.   if $\delta'(q, a) \notin Q'$, then
      01. $N \leftarrow N \cup \delta'(q, a)$
07. if $N$ != $\emptyset$, then
  01. $Q' \leftarrow Q' \cup N$
  02. $C \leftarrow N$
  03. goto 6
08. $F' \leftarrow \emptyset$
09. foreach $q$ in $Q'$, do
  01. foreach $c$ in $q$, do
    01. if $c \in F$, then
      01. $F' \leftarrow F' \cup q.$
      02. break
10. return $(Q', Sigma', \delta', q'_0, F_\prime)$
```

In English, the algorithm states

1. Make the start state of the DFA, q'_0, the invocation of `E()` of q_0 of the NFA, and add it to the set of states of the DFA, Q'.
2. Add the δ' transitions of each state in Q' with each symbol in the alphabet, Σ', to $Q_\prime$.
3. If any new states are added to Q', repeat step 2.
4. Add any state in Q' that contains an accept state from F of the NFA to the accept set of the DFA, F'.