Regular expressions from Finite Automata

The key idea for the construction that creates a regular expression from a finite automaton is to allow edge labels that are regular expressions. Sipser calls this a Generalized Finite Automaton but the formal description is more constrained than I think is convenient. (The main new thing I would like is to allow parallel edges between states). The intuition is that in following an edge labelled by regular expression \( r \), some prefix of the input remaining to be read is in \( L(r) \), the language represented by \( r \) and following the edge means reading such a prefix. A string \( x \) will be accepted if and only if there is some path from the start state to a final state whose labels concatenated together form a regular expression whose associated language contains \( x \). Notice that our standard NFA's and DFA's are special cases of this where all our regular expressions turn out be some \( a \in \Sigma \) in the DFA case and either \( a \in \Sigma \) or \( \varepsilon \) in the NFA case.

For the construction we first add a new start state and a new final state connected to (resp. from) the old ones via \( \varepsilon \)-moves. (This is so that no start or final state is on a cycle.) There are only two rules which we apply until the graph is reduced to a single labelled edge which will have the regular expression on it.

**Rule 1.** Combination of Parallel Edges: If \( q_1 \) and \( q_2 \) are any two states (possibly \( q_1 = q_2 \)) then replace

\[
q_1 \xrightarrow{r} q_2 \quad \text{by} \quad q_1 \xrightarrow{r \cup s} q_2.
\]

**Rule 2.** Removal of States: If \( q_3 \) is not either the new start state or the new final state then for every pair of states \( q_1 \) and \( q_2 \) (again possibly \( q_1 = q_2 \)) replace

\[
q_1 \xrightarrow{r} q_3 \xrightarrow{s} q_2 \quad \text{by} \quad q_1 \xrightarrow{r \cdot s^* \cdot t} q_2.
\]

Turn page over for an example (\( \varepsilon \) stands for \( \varepsilon \) in the example).
Rule 2 to $q_f$

Rule 1 to edges

Rule 2 to $q_2$

Rule 1 to edges

Rule 2 to $q_0$