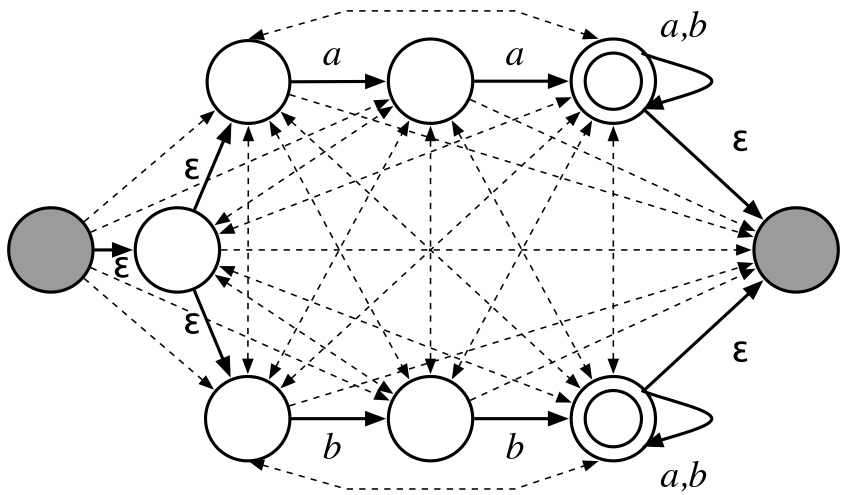
Regular languages can be expressed as regular expressions.

A **general nondeterministic finite automaton (GNFA)** is a kind of NFA such that:

- There is a unique start state and is a unique accept state.
- Every pair of nodes are connected by an arrow in each direction, each labeled with a regular expression. Exceptions are:
  - The start state has no incoming edges.
  - The accept state has no outgoing edges.

The language accepted by the GNFA is the union of all $L(R)$ such that R is the regular expression constructed by concatenating all the regular expressions appearing on the path from the start state to the accept state in the order they appear.

An Example of GNFA



Here the dashed arrows represent arrows with $\emptyset$ as the label.

Proof Plan

1. Show that any NFA can be converted to an equivalent GNFA.
2. Show that any GNFA can be converted to a regular expression.

From an NFA to a GNFA

Given an NFA N, construct a GNFA G as follows:

1. Add a special start state q_{start} and connect it to the initial state of N with ϵ as the label. Connect q_{start} to each remaining state with $\emptyset$ as the label.

From an NFA to a GNFA

Given an NFA N, construct a GNFA G as follows:

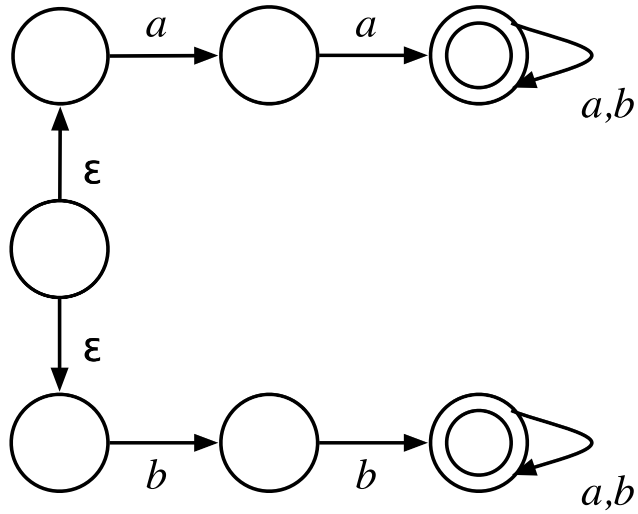
1. Add a special start state q_{start} and connect it to the initial state of N with ϵ as the label. Connect q_{start} to each remaining state with $\emptyset$ as the label.
2. Add a special accept state q_{accept} and connect to it from each final state of N with ϵ as the label. Connect from each of the remaining state of N to q_{accept} with $\emptyset$ as the label.

From an NFA to a GNFA

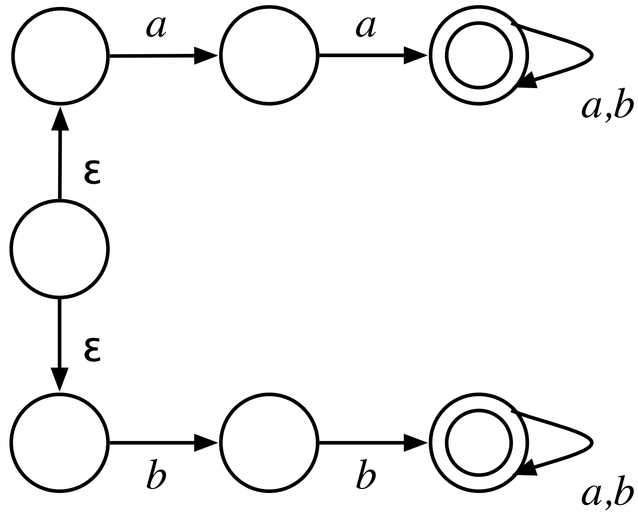
Given an NFA N, construct a GNFA G as follows:

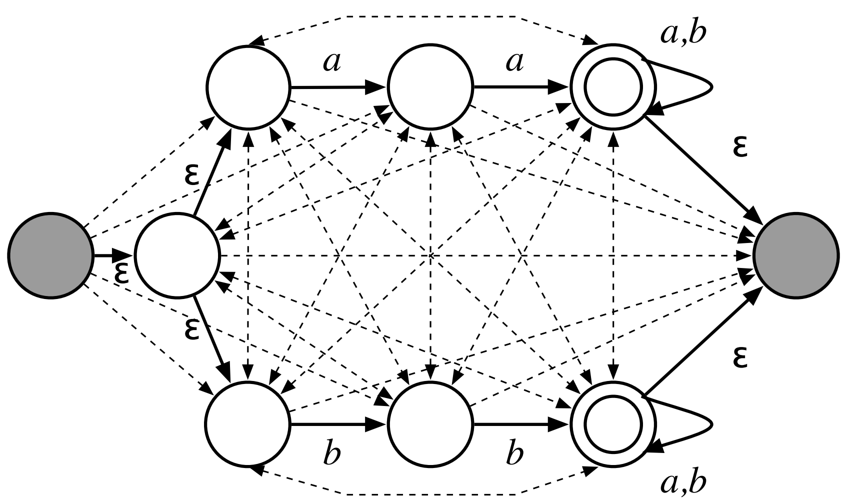
1. Add a special start state q_{start} and connect it to the initial state of N with ϵ as the label. Connect q_{start} to each remaining state with $\emptyset$ as the label.
2. Add a special accept state q_{accept} and connect to it from each final state of N with ϵ as the label. Connect from each of the remaining state of N to q_{accept} with $\emptyset$ as the label.
3. Add an arrow with $\emptyset$ as the label wherever necessary.

Before and After



Before and After





What Have We Done?

We have constructed our initial GNFA G so that:

(I) For each word w in $L(N)$, there is a path $[u_0, \ldots, u_m]$ in G from q_{start} to q_{accept} such that

  (a) w is decomposed as $w_1 w_2 \cdots w_m$;

  (b) for all i, $1 \leq i \leq m$, w_i is a member of the language represented by the regular expression of the arrow (u_{i-1}, u_i).

What Have We Done?

We have constructed our initial GNA G so that:

(I) For each word w in $L(N)$, there is a path $[u_0, \ldots, u_m]$ in G from q_{start} to q_{accept} such that

(a) w is decomposed as $w_1 w_2 \cdots w_m$;

(b) for all i, $1 \le i \le m$, w_i is a member of the language represented by the regular expression of the arrow (u_{i-1}, u_i).

(II) For each word w that can be decomposed as in the above, w is a member of $L(N)$.

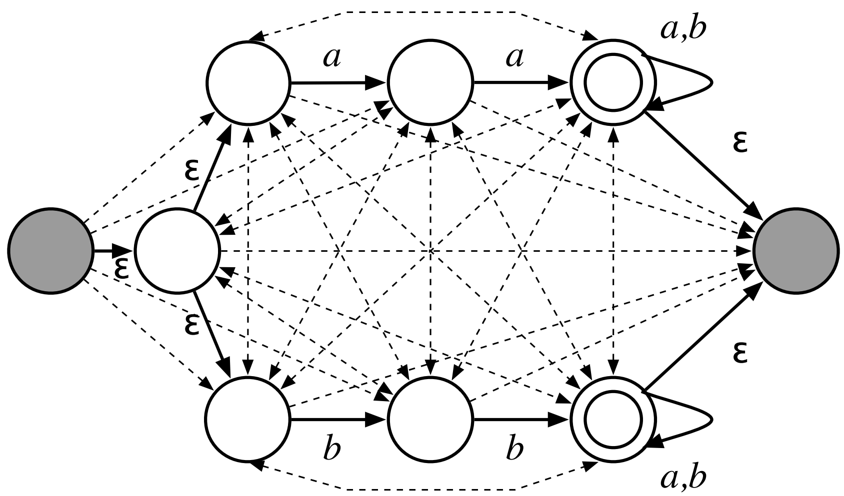
Convert a GNFA to a Regular Expression

We will remove intermediate nodes one after the other while preserving the two properties.

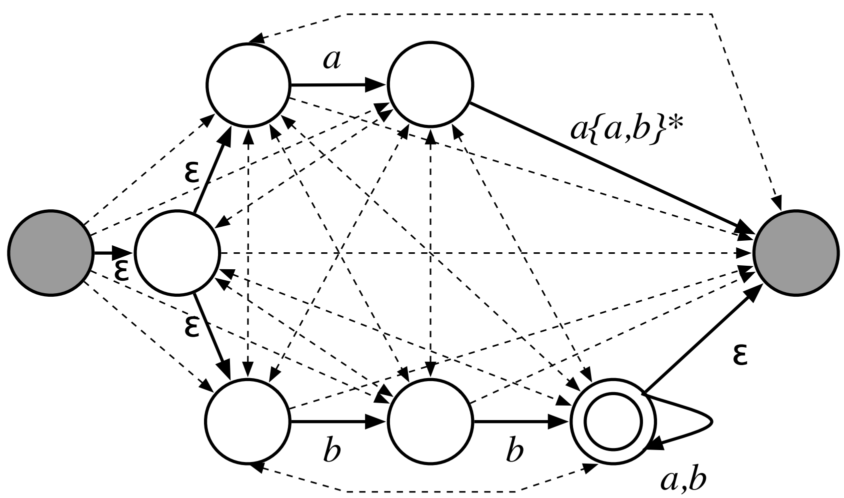
Repeat the following until there is no state left other than the start state and the accept state.

- Select an arbitrary state s.
- Produce from the current GNFA an equivalent GNFA by:
  - For each arrow (p, q) such that $p, q \neq s$, replace its label with the label corresponding to the paths $[p, q], [p, s, q], [p, s, s, q], [p, s, s, s, q], \ldots$.
  - Remove s and all edges attached to it.

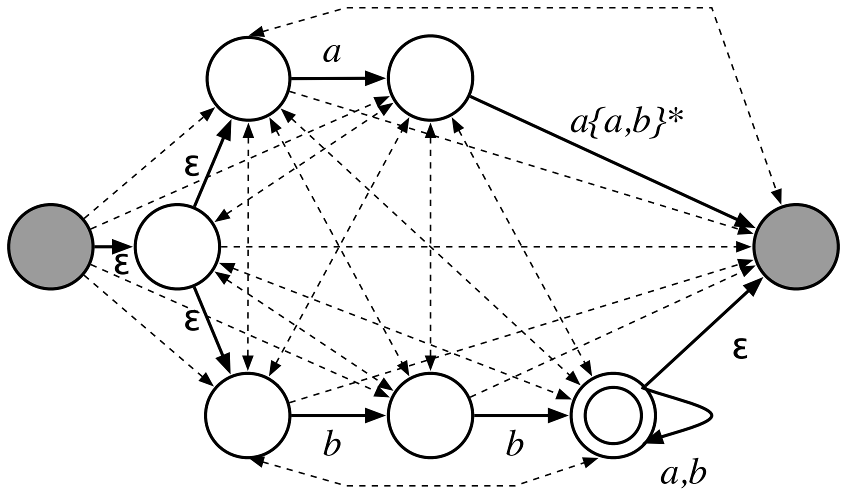
State Elimination



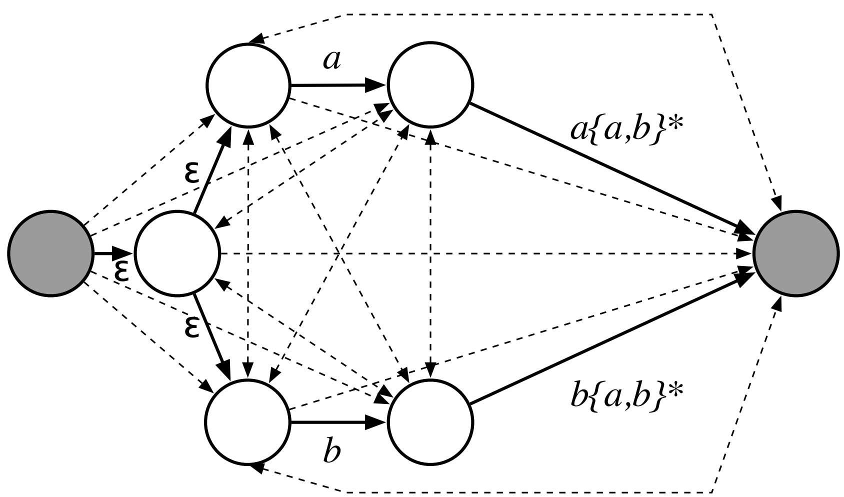
becomes



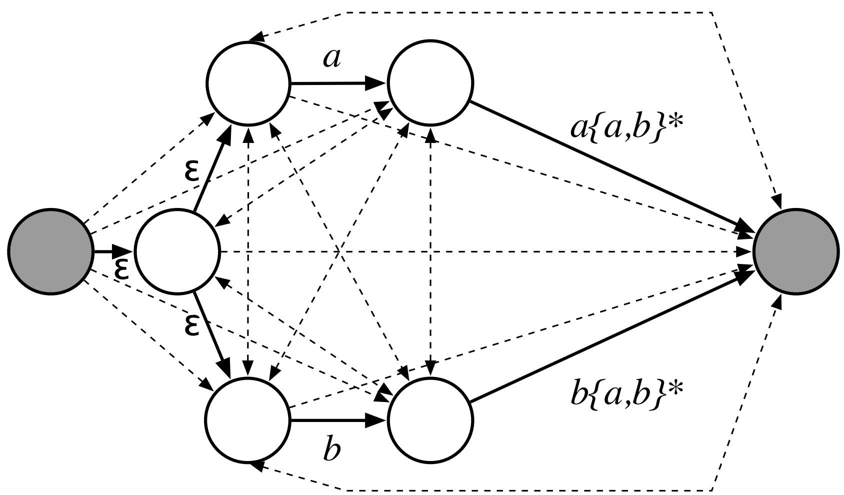
State Elimination



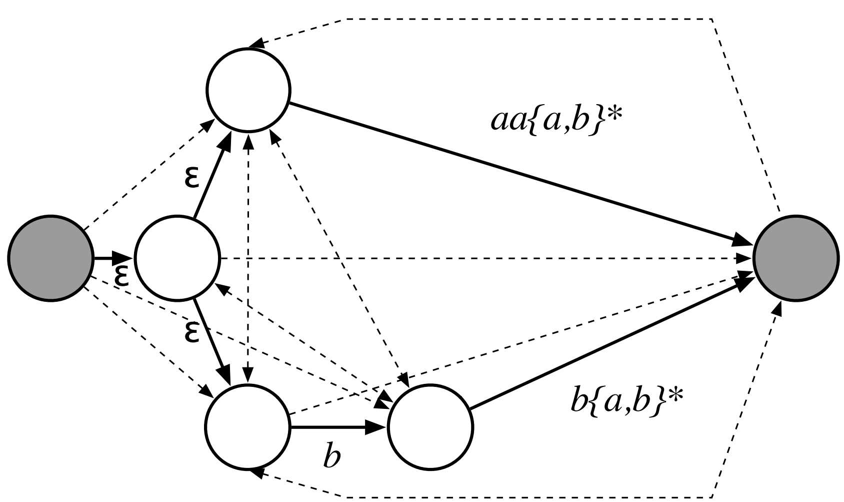
becomes



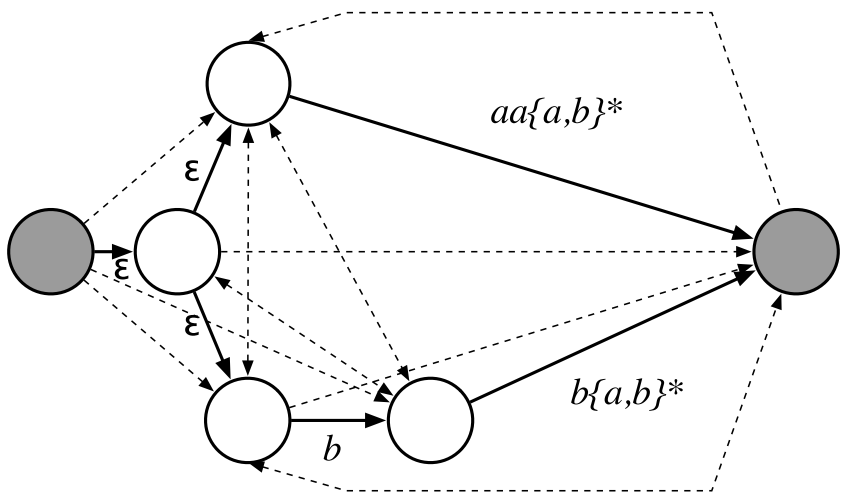
State Elimination



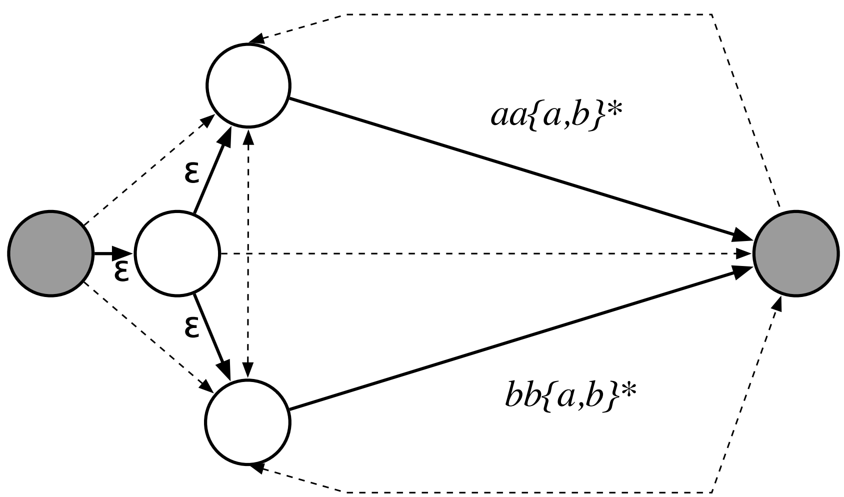
becomes



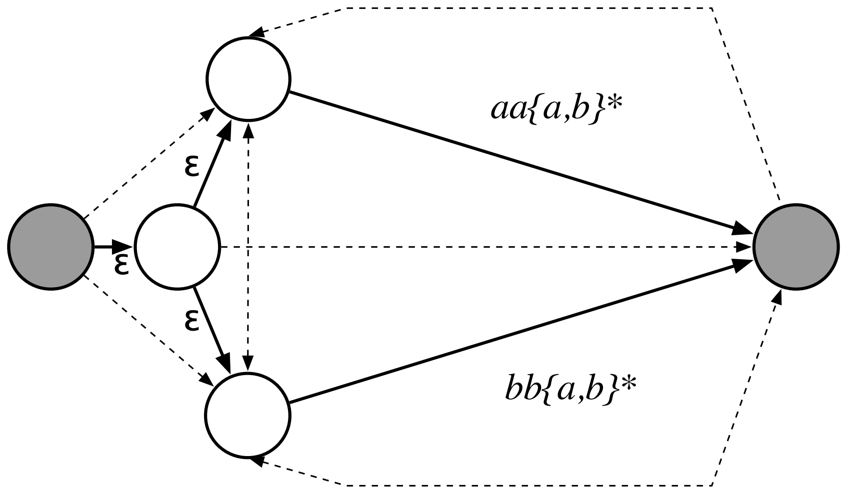
State Elimination



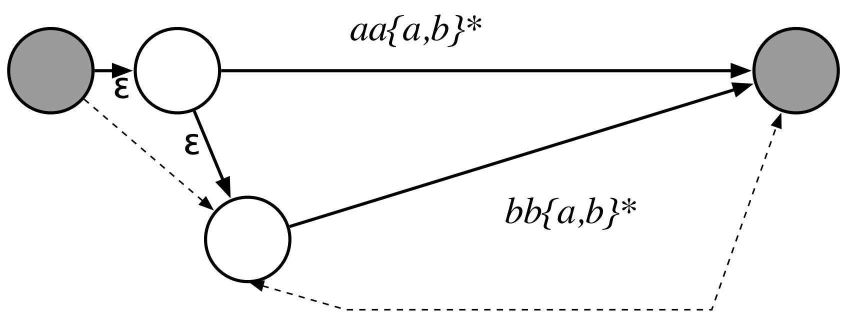
becomes



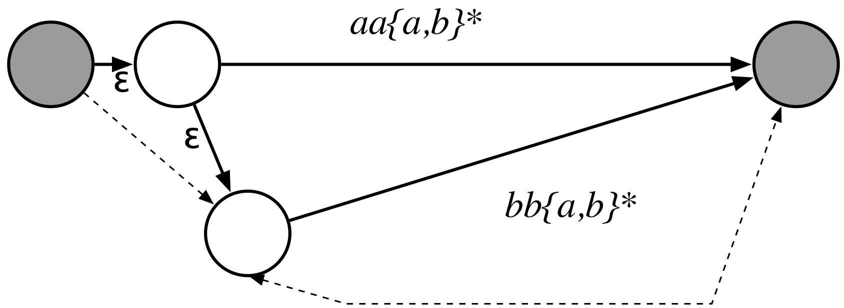
State Elimination



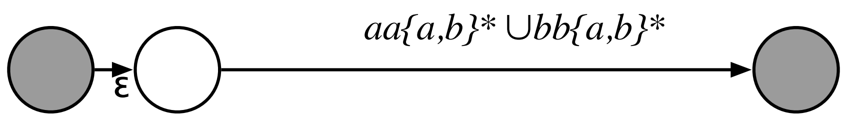
becomes



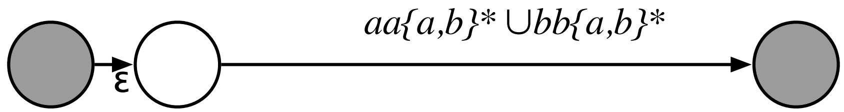
State Elimination



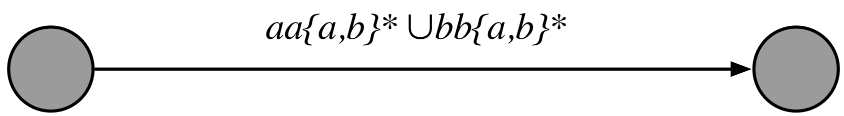
becomes



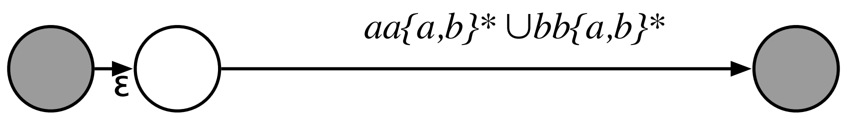
State Elimination



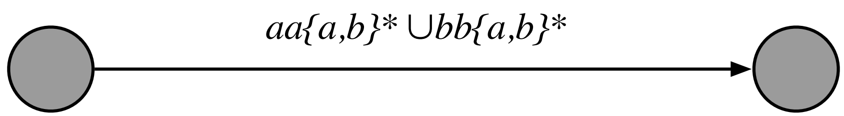
becomes



State Elimination



becomes



The conversion has been completed.