Given a language $L$ over an arbitrary alphabet $\Sigma$, we can define the language

$$\text{Cycle}(L) := \{yx \mid xy \in L\}$$

We can then ask the question, if $L$ is regular, is $\text{Cycle}(L)$ also regular? The answer is yes. Since $L$ is regular, there exists some DFA $M = (Q, \delta, s, A)$ such that $L = L(M)$. We will construct a new NFA $N$ from $M$ such that $N$ accepts $\text{Cycle}(L)$.

Given a string $w$, we need $N$ to accept $w$ if and only if $w \in \text{Cycle}(L)$. This means that there exists some way to split $w$ into two substrings $y$ and $x$ such that $w = yx$ and $xy \in L$. We are immediately confronted with two problems:

(i) There are $|w| + 1$ ways to split $w$ into $y$ and $x$, and we do not know which of these ways result in $xy$ being a string in $L$.

(ii) Even if there is a way to split $w$ into $w = yx$ such that $xy \in L$, we still need to contend with the fact that we are not reading $xy$, but rather $yx$, and furthermore since we are reading the input string one character at a time, we know nothing about how many characters belong to the $y$ part (or how many characters belong to the $x$ part) ahead of time.

We will get around both issues using non-determinism.

Here’s the idea. Suppose that $w \in \text{Cycle}(L)$. We would like to guess a decomposition $w = yx$ so that $xy \in L$, and then feed $xy$ through $M$ to verify that $xy \in L$. But we can’t, since we don’t know anything about $w$ yet. But we can try to simulate and guess some other things.

- Suppose we could guess the state $q_x := \delta^*(s, x)$ (i.e, the state we land on in $M$ if, starting from $s$, we read in $x$). If we are correct, then starting at $q_x$ and reading the first $|y|$ characters of $w$ (i.e., $y$) gets us to an accepting state, since $\delta^*(q_x, y) = \delta^*(\delta^*(s, x), y) = \delta^*(s, xy)$ which as an accepting state by the assumption $xy \in L$.

- Since we don’t actually know $y$, we can do the following instead. Suppose that, starting from $q_x$, we read some number of characters from $w$ and end at an accepting state. At this point, we could guess that we actually correctly read $y$.

- At this point, we have made two guesses: the state $q_x$, and if we are done reading $y$. But how do we know that our two guesses were actually correct? Well, suppose we go back to the starting state $s$ and read in the rest of $w$. If our two guesses were correct, then the rest of the input is $x$, and reading the rest of the input should get us to $\delta^*(s, x) = q_x$.

Well, how do we actually do any of that? Well, we will build an NFA that simulates doing the above. This NFA needs to keep track of at least the following pieces of information: the state we guessed to be $q_x$, whether or not we have guessed that we are done reading $y$, and what state we’re at in the simulation of the DFA $M$. Before doing anything, this NFA needs to simulate starting at the guessed state $q_x$, and then after guessing that we are done reading $y$, the NFA needs to simulate starting at $s$ again. The NFA will accept if and only if the state we end up at is in fact $q_x$. 

1
Thus, given a DFA $M = (Q, \delta, s, A)$ for $L$, we construct an NFA $N = (Q', \delta', s', A')$ for $\text{Cycle}(L)$ as follows.

- The **states** of $N$ are

$$Q' := (Q \times Q \times \{\text{Before}, \text{After}\}) \cup \{s'\}$$

where $s'$ is a **new starting state**.

The meaning of a state $(q, q, b) \in Q \times Q \times \{\text{Before}, \text{After}\}$ is as follows: $q$ is the state we guessed to be $q_x$, $r$ is the current state in the simulation, and $b$ is **Before** if we haven’t guessed that $y$ is done, and **After** if we have.

- From the new starting state $s'$, we have $\varepsilon$-transitions to each state of the form $(q, q, \text{Before})$. Each of these $\varepsilon$-transitions represents guessing a state $q$ to be $q_x$ and starting the simulation of $M$ at $q$. We need these to be $\varepsilon$-transitions so that we can non-deterministically choose one of them before the simulation.

For each state $(q, r, \text{Before})$ where $r \in A$ ($r$ is an accepting state in $M$), we have an $\varepsilon$-transition to the state $(q, s, \text{After})$ ($s$ is the starting state of $M$). This represents the possibility of guessing that we are done reading $y$, and are thus ready to read in $x$ and verify that the initial guess was correct. Even if $M$ only had one accepting state $r$, we would still use an $\varepsilon$-transition instead of combining $(q, r, \text{Before})$ and $(q, s, \text{After})$ to avoid the possibility of being able to move back to Before states from an After state.

Otherwise, we are just reading in characters and simulating various parts of $M$ on the input. This can be done by advancing $r$ along $\delta$.

In summary, we can write down the following **transition function** $\delta'$:

\[
\delta'(s', \varepsilon) := \{(q, q, \text{Before}) \mid q \in Q\}
\]

For $r \in A$, \[\delta'(q, r, \text{Before}), \varepsilon) := \{(q, s, \text{After})\}\]

\[\delta'(q, r, b), a) := \{(q, \delta(r, a), b)\}\]

**All unspecified transitions do not exist** (i.e., have value $\emptyset$).

- Finally, all that remains to be specified is the set of **accepting states** $A'$. Recall that our guesses were correct if, after reading in what we guessed to be the $x$ part of $w$ from $s$, we end at what we guessed to be $q_x$. In other words:

$$A' := \{(q, q, \text{After}) \mid q \in Q\}$$

This completes the construction.
As a concrete example, consider the following DFA for the language $L = L(00^*11^*)$:

![DFA Diagram]

The NFA we construct for $\text{CYCLE}(L)$ is shown below. **Bolded** transitions are new $\epsilon$-transitions; all other transitions are the same as their corresponding transitions in the DFA above.
Finally, let’s verify that the general construction is correct, i.e., \( L(N) = \text{Cycle}(L) \). The following “proof” is missing a few details, in particular, a proof of the interaction between \((\delta')^*\) and \(\delta^*\). Feel free to follow along on the concrete example on the previous page.

- First, suppose that \( w \in L(N) \). Then there is some path that we can follow while reading in \( w \) such that, starting from \( s' \), we reach an accepting state, i.e., a state of the form \((q, q, \text{AFTER})\).

By construction, this path must start by taking an \( \epsilon \)-transition to a state of the form \((q, q, \text{BEFORE})\), followed by some walk inside of the states \( \{q\} \times Q \times \{\text{BEFORE}\} \). At some point, an \( \epsilon \)-transition leaving \( \{q\} \times Q \times \{\text{BEFORE}\} \) must be taken; call the input read up to this point \( y \). The \( \epsilon \)-transition leaving \( \{q\} \times Q \times \{\text{BEFORE}\} \) must come from a state of the form \((q, r, \text{BEFORE})\) where \( r \in A \), and goes to \((q, s, \text{AFTER})\); at this point reading the rest of the input must lead to \((q, q, \text{AFTER})\). Call the rest of the input \( x \).

At this point we know that \( \delta^*(s, x) = q \), and furthermore \( \delta^*(q, y) \in A \). In other words, \( x y \in L \). We therefore conclude that \( w = y x \in \text{Cycle}(L) \).

This shows that \( L(N) \subseteq \text{Cycle}(L) \).

- Next, suppose \( w \in \text{Cycle}(L) \). Then \( w = y x \) where \( x y \in L \). We need to show that there is a path from \( s' \) to a an accepting state in \( N \), i.e., a state of the form \((q, q, \text{AFTER})\).

Set \( q = \delta^*(s, x) \). Before reading any of \( w \) we take an \( \epsilon \)-transition to \((q, q, \text{BEFORE})\). After reading \( y \), we must be in the state \((q, \delta^*(q, y), \text{BEFORE})\) where \( \delta^*(q, y) = \delta^*(\delta^*(s, x), y) = \delta^*(s, x y) \in A \). By construction there exists an \( \epsilon \)-transition from \((q, \delta^*(q, y), \text{BEFORE})\) to \((q, s, \text{AFTER})\). After taking this transition, we read in the rest of \( w \) (i.e., \( x \)) and reach \((q, \delta^*(s, x), \text{AFTER}) = (q, q, \text{AFTER})\), which is an accepting state of \( N \).

We therefore conclude that \( w \in L(N) \), and thus \( \text{Cycle}(L) \subseteq L(N) \).