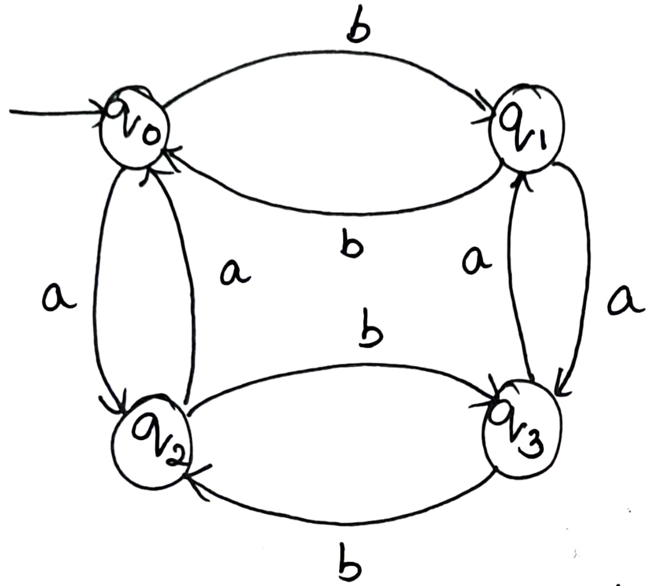
DFA

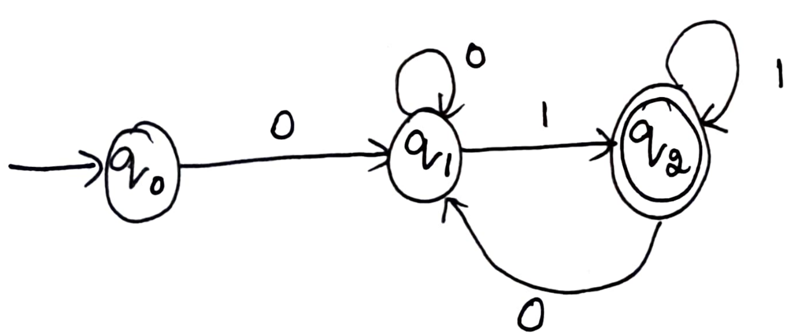
a) $L = \{ w \mid w \in \{a,b\}^* : w$ contains an even number of a's and odd number of b's $\}$.



{ b, aab, bbb, bbbbb, aaaab, aabbb, ababab a, bababbaab . . . }

b) Let $\Sigma = \{0, 1\}$. Design DFA to accept all strings starting with 0 and ending with

1.



{ 01, 001, 0111, 00011, 010101, 00110101 . . . }

2) What is the meaning of these following symbols in automata theory?
$\phi, \varepsilon, \Sigma, \Sigma^*$.

ϕ: empty language, is a language over any alphabet.

ε: empty string with length = 1
For any string $\varepsilon w = w\varepsilon = w$
ε is the identity for concatenation.

Σ: An Alphabet is a finite, nonempty set of Characters / Symbols.
eg: $\Sigma = \{a, b\}$
$\Sigma = \{0, 1\}$

Σ^*: defined as the set of all possible strings of all possible lengths of alphabet Σ, including 'ε' called as "Kleene Closure".

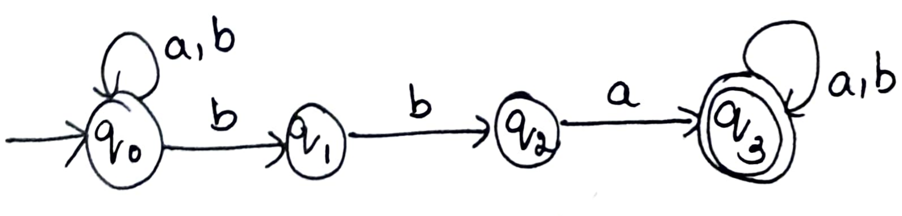
b) Show why $\varepsilon \neq \phi$

$\{\varepsilon\}$ the language consisting of only the empty string.
ϕ has no string.
Length of $\phi = 0$ $\therefore \varepsilon \neq \phi$
Length of $\varepsilon = 1$

Q 1B) Differentiate between NDFA & DFA.

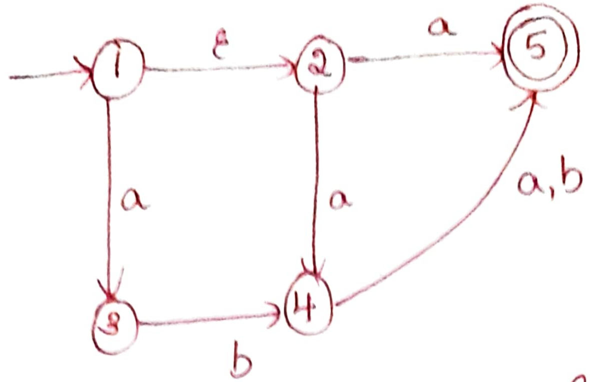
NDFA	DFA
1) The transition from a state can be to multiple next states for each input symbol. Hence it is called Non-deterministic.	1) The transition from a state is to a single particular next state for each input symbol. Hence it is called deterministic.
2) Permits (ε) empty string transitions.	2) empty string tr
3) Requires less space	3) Requires more space
4) A string is accepted by a NDFA, if at atleast one of all possible transitions ends in a final state	4) A string is accepted by a DFA, if it transits to a final state.

Q 2 a) Design an NDFA for $L = \{w \mid w \in \{a,b\}^*$: w contains the substring bba}



{ bba, abba, babba, ababba, ababba ..... }

3. Convert the following ε-NDFSM to DFSM



a) Give value for Epsilon Closure for each state.

ε-closure $(1) = \{1, 2\}$

ε-closure $(2) = \{2\}$

ε-closure $(3) = \{3\}$

ε-closure $(4) = \{4\}$

ε-closure $(5) = \{5\}$

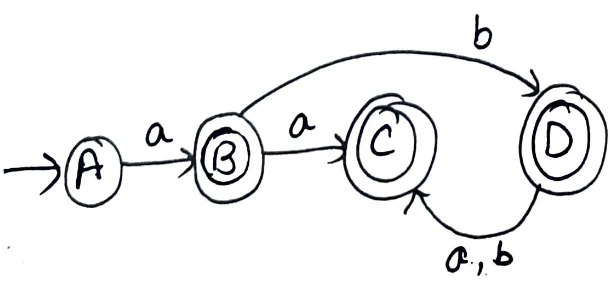
b) Give the Steps for Conversion.

ε-closure $(\{1\}) = \{1, 2\} = A$

ε-closure $(\{3, 4, 5\}) = \{3, 4, 5\} = B$

ε-closure $(\{5\}) = \{5\} = C$

ε-closure $(\{4, 5\}) = \{3, 4\} = D$

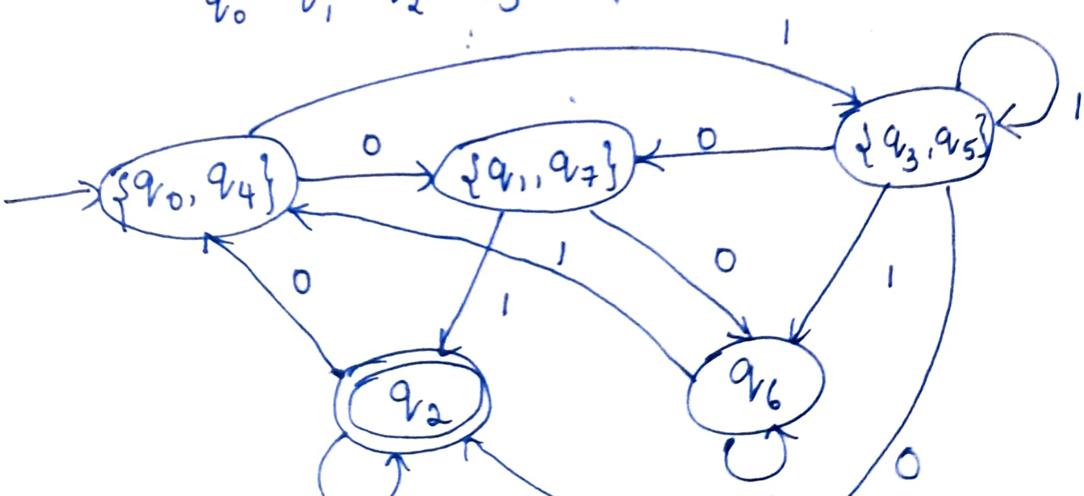


	a	b
→A = {1,2}	{3,4,5} B	ϕ
* B {3,4,5}	{5} C	{4,5} D
* C {5}	ϕ	ϕ
* D {4,5}	C {5}	C {5}

4) Minimize the following DFSM.

States	0	1
$\rightarrow q_0$	q_1	q_5
q_1	q_6	q_2
$*q_2$	q_0	q_2
q_3	q_2	q_6
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

q_1	X						
q_2	X	X					
q_3	X	X	X				
q_4		X	X	X			
q_5	X	X	X		X		
q_6	X	X	X	X	X	X	
q_7	X		X	X	X	X	X
	q_0	q_1	q_2	q_3	q_4	q_5	q_6

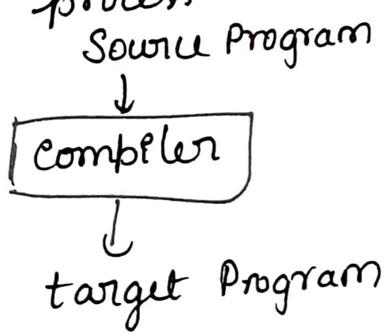


2. Write the difference between compiler & interpreter.

Compiler

1) A compiler is a program that can read a program in one language - the source language - and translate it into an equivalent program in another language the target language.

2) Role of compiler is to report any errors in the source program that it detects during the translation process.



Interpreter

1) An interpreter directly execute the operations specified in the source program on inputs supplied by the user.

2) Interpreter gives better error diagnostics than a compiler, because it executes the source program statement by statement.

5) i) convert the following NFA to DFA

δ	0	1
$\rightarrow p$	$\{p, q\}$	$\{p\}$
q	ϕ	$\{r\}$
$* r$	$\{p, r\}$	$\{q\}$

Theorem 2.11: If $D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ is the DFA constructed from NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ by the subset construction, then $L(D) = L(N)$.

PROOF: What we actually prove first, by induction on $|w|$, is that

$$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$$

Notice that each of the $\hat{\delta}$ functions returns a set of states from Q_N, but $\hat{\delta}_D$ interprets this set as one of the states of Q_D (which is the power set of Q_N), while $\hat{\delta}_N$ interprets this set as a subset of Q_N.

BASIS: Let $|w| = 0$; that is, $w = \epsilon$. By the basis definitions of $\hat{\delta}$ for DFA's and NFA's, both $\hat{\delta}_D(\{q_0\}, \epsilon)$ and $\hat{\delta}_N(q_0, \epsilon)$ are $\{q_0\}$.

INDUCTION: Let w be of length $n + 1$, and assume the statement for length n. Break w up as $w = xa$, where a is the final symbol of w. By the inductive hypothesis, $\hat{\delta}_D(\{q_0\}, x) = \hat{\delta}_N(q_0, x)$. Let both these sets of N's states be $\{p_1, p_2, \ldots, p_k\}$.

The inductive part of the definition of $\hat{\delta}$ for NFA's tells us

$$\hat{\delta}_N(q_0, w) = \bigcup_{i=1}^{k} \delta_N(p_i, a) \tag{2.2}$$

The subset construction, on the other hand, tells us that

$$\delta_D(\{p_1, p_2, \ldots, p_k\}, a) = \bigcup_{i=1}^{k} \delta_N(p_i, a) \tag{2.3}$$

Now, let us use (2.3) and the fact that $\hat{\delta}_D(\{q_0\}, x) = \{p_1, p_2, \ldots, p_k\}$ in the inductive part of the definition of $\hat{\delta}$ for DFA's:

$$\hat{\delta}_D(\{q_0\}, w) = \delta_D\big(\hat{\delta}_D(\{q_0\}, x), a\big) = \delta_D(\{p_1, p_2, \ldots, p_k\}, a) = \bigcup_{i=1}^{k} \delta_N(p_i, a) \tag{2.4}$$

Thus, Equations (2.2) and (2.4) demonstrate that $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$. When we observe that D and N both accept w if and only if $\hat{\delta}_D(\{q_0\}, w)$ or $\hat{\delta}_N(q_0, w)$, respectively, contain a state in F_N, we have a complete proof that $L(D) = L(N)$. $\square$

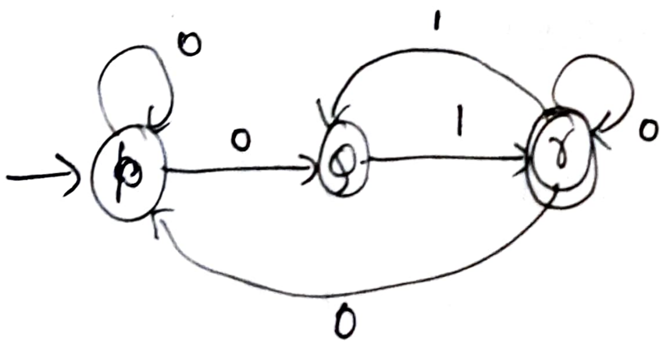
2)

δ	0	1
→ p	{p,q}	{p}
q	φ	{r}
* r	{p,r}	{q}

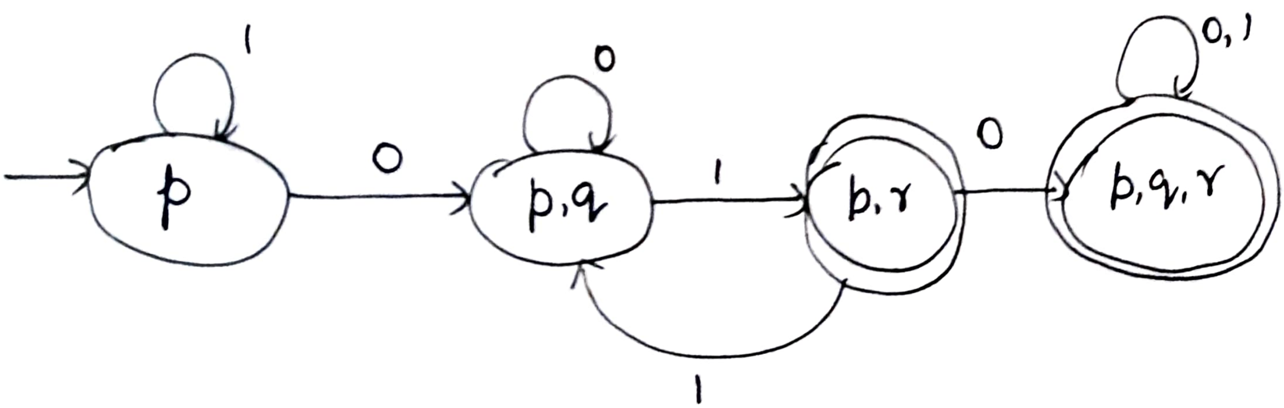
NFA.

DFA

δ	0	1
φ	φ	φ
{p}	{p,q}	{p}
{q}	φ	{r}
* {r}	{p,r}	{q}
{p,q}	{p,q}	{p,r}
* {p,r}	{p,q,r}	{p,q}
* {q,r}	{p,r}	{r,q}
* {p,q,r}	{p,q,r}	{p,q,r}



{ 001 , 001001, 00100 , ... }



length more than 3

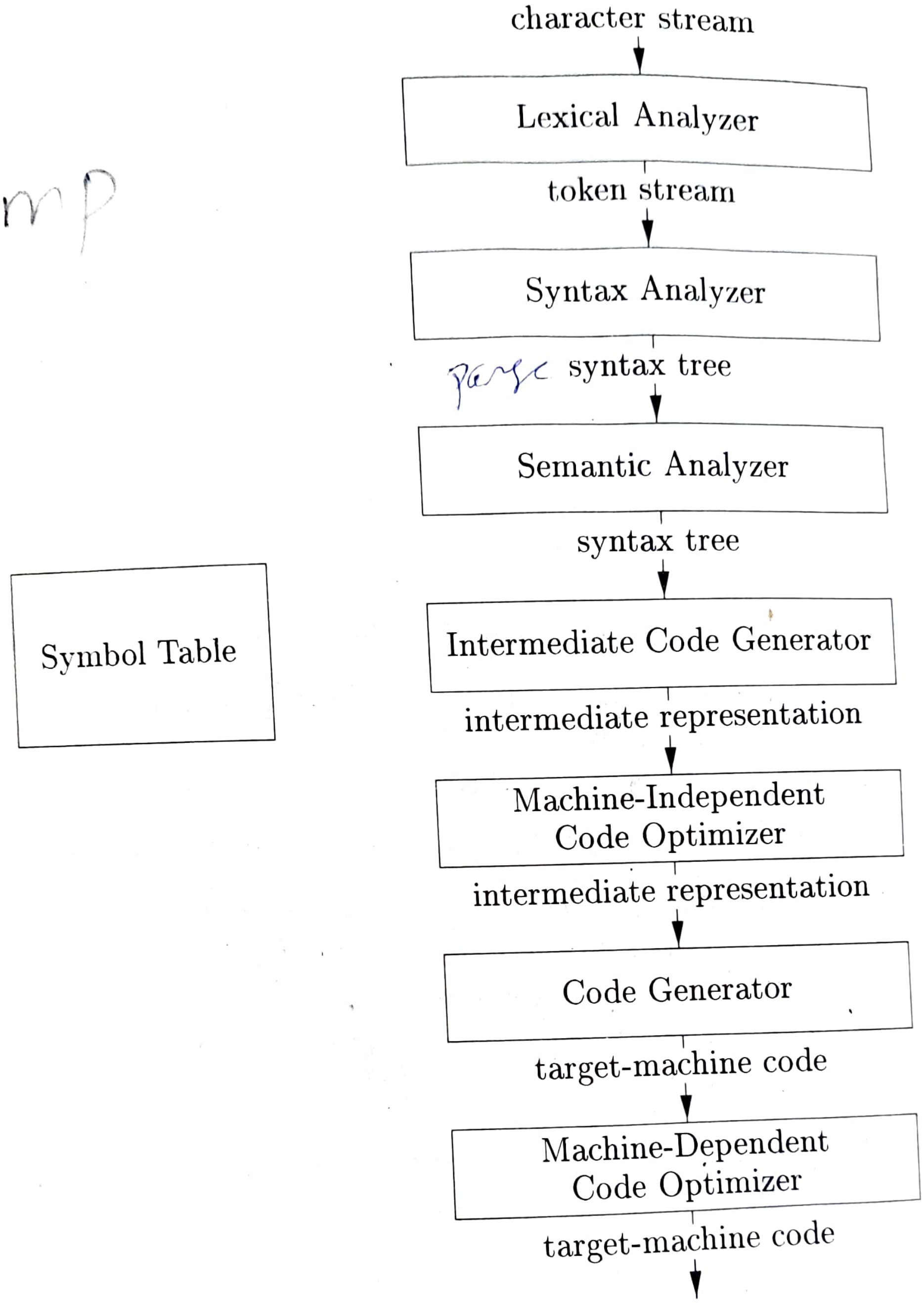


Figure 1.6: Phases of a compiler

entire source program, is used by all phases of the compiler.

Some compilers have a machine-independent optimization phase between the front end and the back end. The purpose of this optimization phase is to perform transformations on the intermediate representation, so that the back end can produce a better target program than it would have otherwise produced from an unoptimized intermediate representation. Since optimization is optional, one or the other of the two optimization phases shown in Fig. 1.6 may be missing.

1.2.1 Lexical Analysis

scanning. The lexa, b, c, d, e, f → floats

A = b/c * d − (e + f)

10m

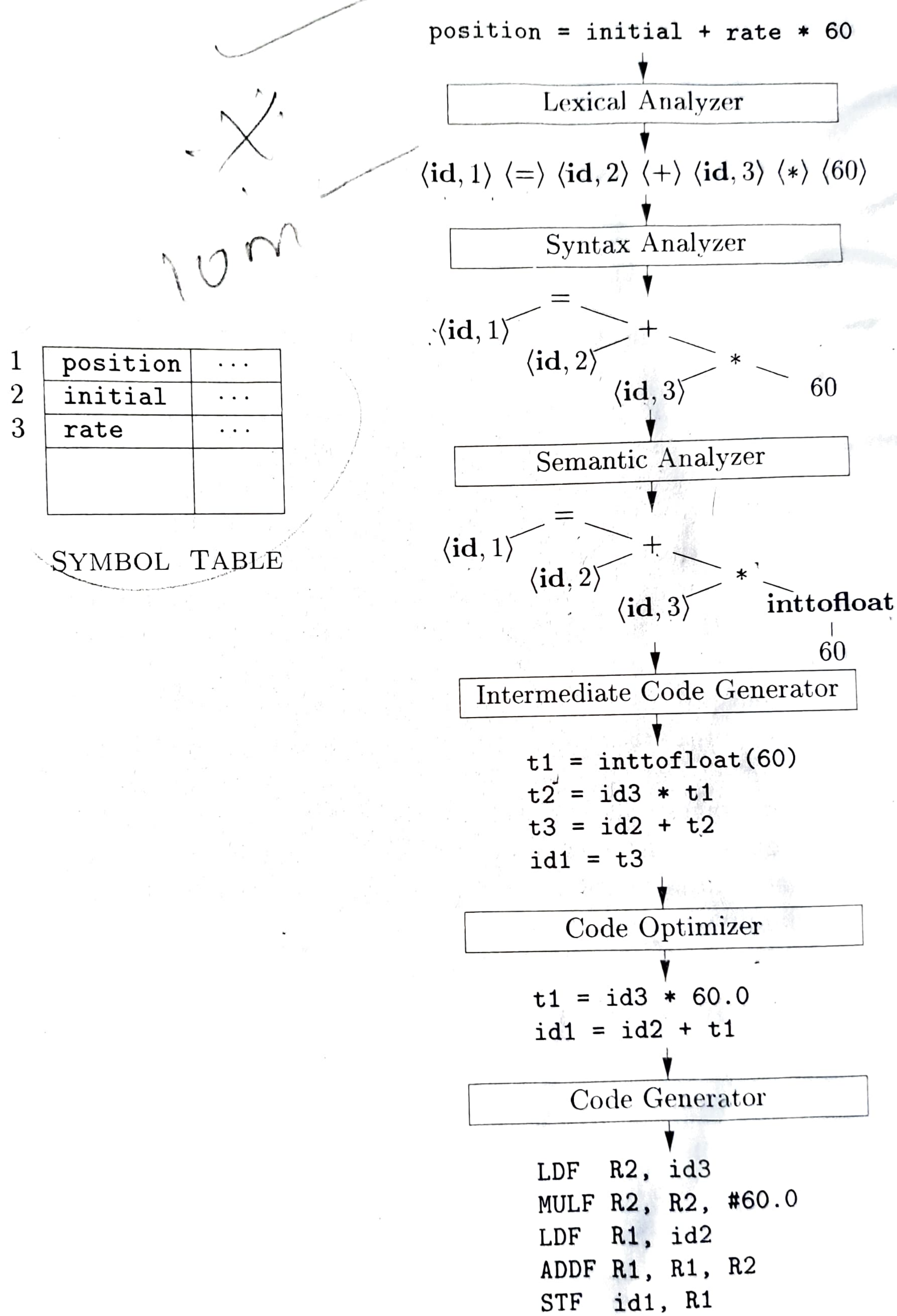
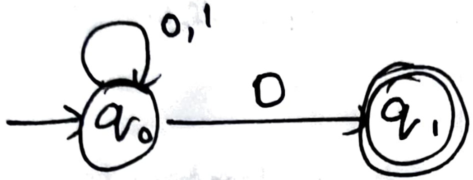


Figure 1.7: Translation of an assignment statement

6 2) Design a NFA for L = {set of all strings that end with 0}



{ , 010, 110, 10, 1110010, 1101010, 0110 . . . . .}