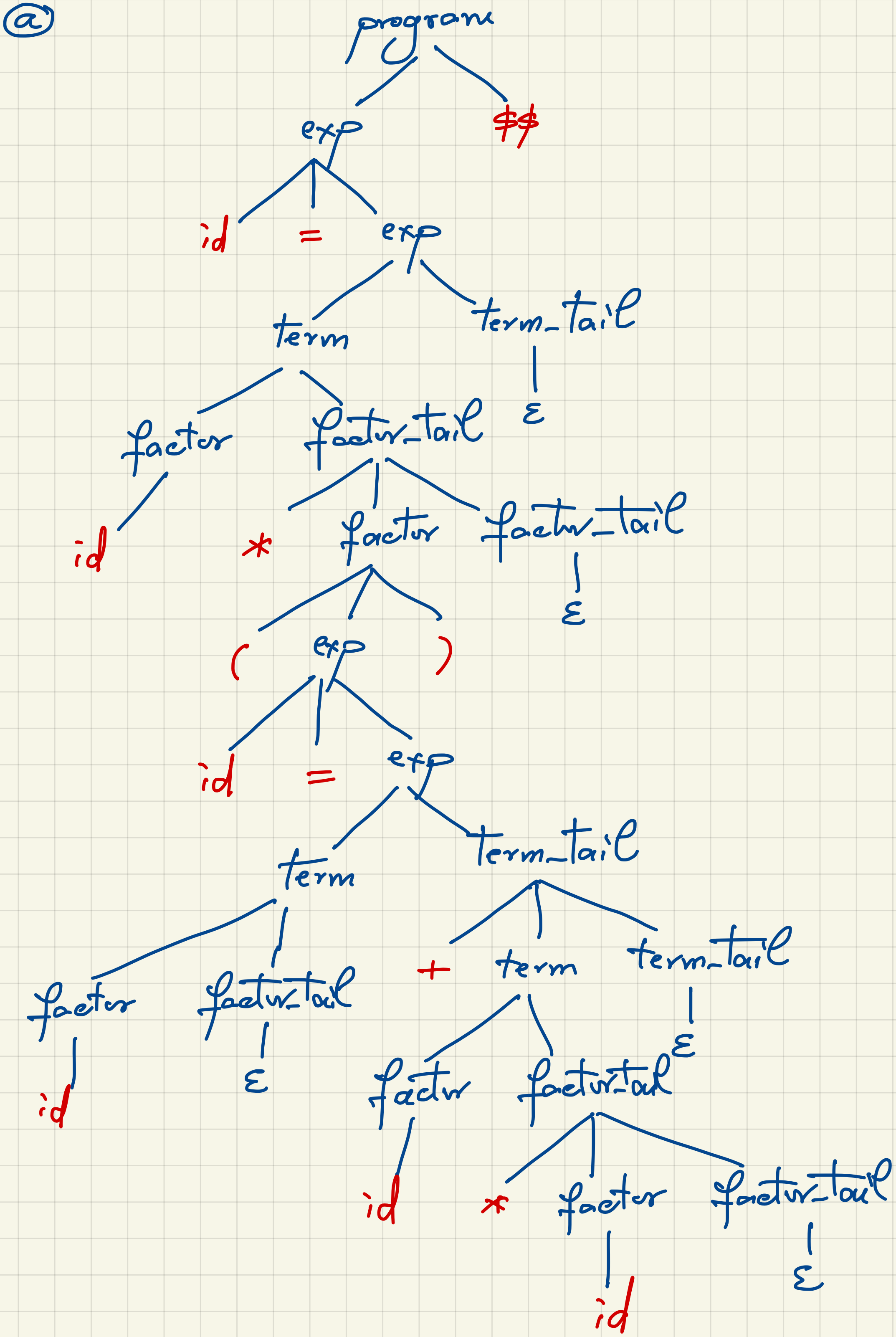
A2 -sol

(CS3342- win. 2024)

Q_1 a


program
exp
$$
id
=
exp
term
term_tail
factor
factor_tail
ε
id
*
factor
factor_tail
ε
(
exp
)
id
=
exp
term
term_tail
factor
factor_tail
+
term
term_tail
id
ε
ε
factor
factor_tail
id
*
factor
factor_tail
id
ε

(b)

α	FIRST(α)
exp $$	id:(1,1), (:(1,2)
id = exp	id:(0,–)
term term_tail	(,id:(1,5)
+ term term_tail	+:(0,–)
ε	∅
factor factor_tail	(:(1,8), id:(1,9)
* factor factor_tail	*:(0,–)
ε	∅
(exp)	(:(0,–)
id	id:(0,–)

A	FOLLOW(A)
program	∅
exp	$$:(2,0),):(2,8)
term	+:(2,2), $$,):(3,2)
term_tail	$$,):(3,2)
factor	*:(2,5), +, $$,):(3,5)
factor_tail	+, $$,):(3,5)

ⓒ

i	PREDICT(i)
0	id, (
1	id
2	(, id
3	+
4	$$,)
5	(, id
6	*
7	+, $$,)
8	(
9	id

ⓓ

conflict ⇒ not LL(1)

ⓔ There is only one conflict, between productions 1 and 2, on id.

2. exp → term term_tail

- can be replaced by:

2'. exp → factor factor_tail term_tail

- because there is only one production with LHS term.
- replace again by:

2''a : exp → (exp) factor_tail term_tail

2''b: exp → id factor_tail term_tail ← this is causing the conflict with prod. 1

- eliminate the common prefix id between:

1. exp → id = exp

2''b : exp → id factor_tail term_tail

gives:

2'''a. exp → id exp_tail

2'''b. exp_tail → = exp

2'''c. exp_tail → factor_tail term_tail

The new grammar is obtained by replacing productions 1 and 2 by 2''a, 2'''a-c, see next page.

1. program → exp $$
2. exp → (exp) factor_tail term_tail
3. exp → id exp_tail
4. exp_tail → = exp
5. exp_tail → factor_tail term_tail
6. term_tail → + term term_tail
7. term_tail → ε
8. term → factor factor_tail
9. factor_tail → * factor factor_tail
10. factor_tail → ε
11. factor → (exp)
12. factor → id

The only places where conflicts can apper are:

- productions 2, 3 – clearly no conflicts because RHS's start with different tokens

- productions 4, 5 : FIRST(factor_tail term_tail) = {*, +} does not include '=', therefore no conflict

⟹ new grammar is <u>LL(1)</u>.

Q2 (a)

1. $P \rightarrow E \ \$\$$
2. $E \rightarrow E \text{ or } T$
3. $E \rightarrow T$
4. $T \rightarrow T \text{ and } F$
5. $T \rightarrow F$
6. $F \rightarrow (E)$
7. $F \rightarrow id$

(b)

X	FIRST(X)	FOLLOW(X)
P	(, id	∅
E	(, id	\$\$, or,)
T	(, id	\$\$, or,), and
F	(, id	\$\$, or,), and

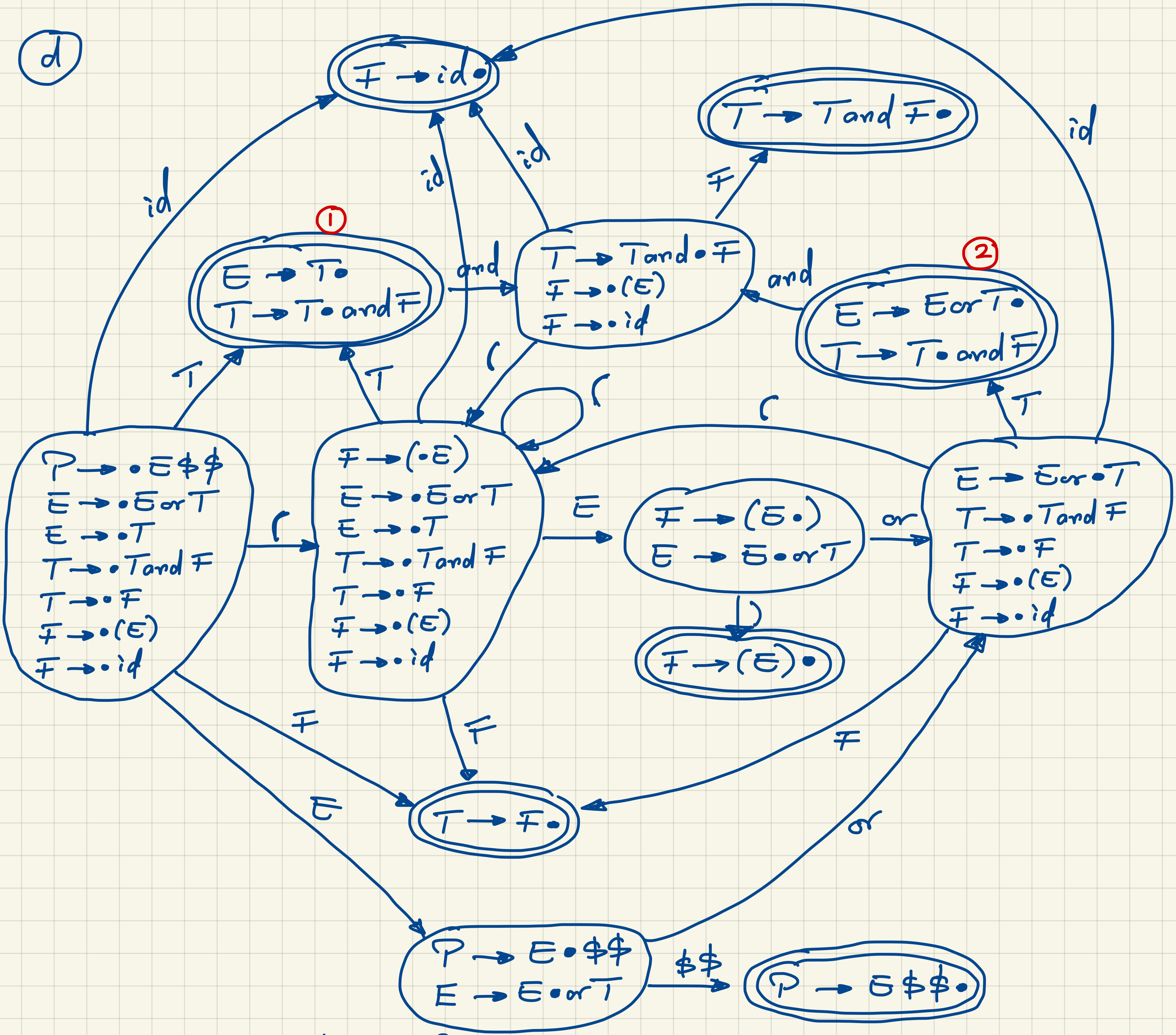
i	PREDICT(i)
1.	(, id
2.	(, id
3.	(, id
4.	(, id
5.	(, id
6.	(
7.	id

(c)

2., 3.: conflicts ⇒ not LL(1)

4., 5.: conflicts ⇒

d



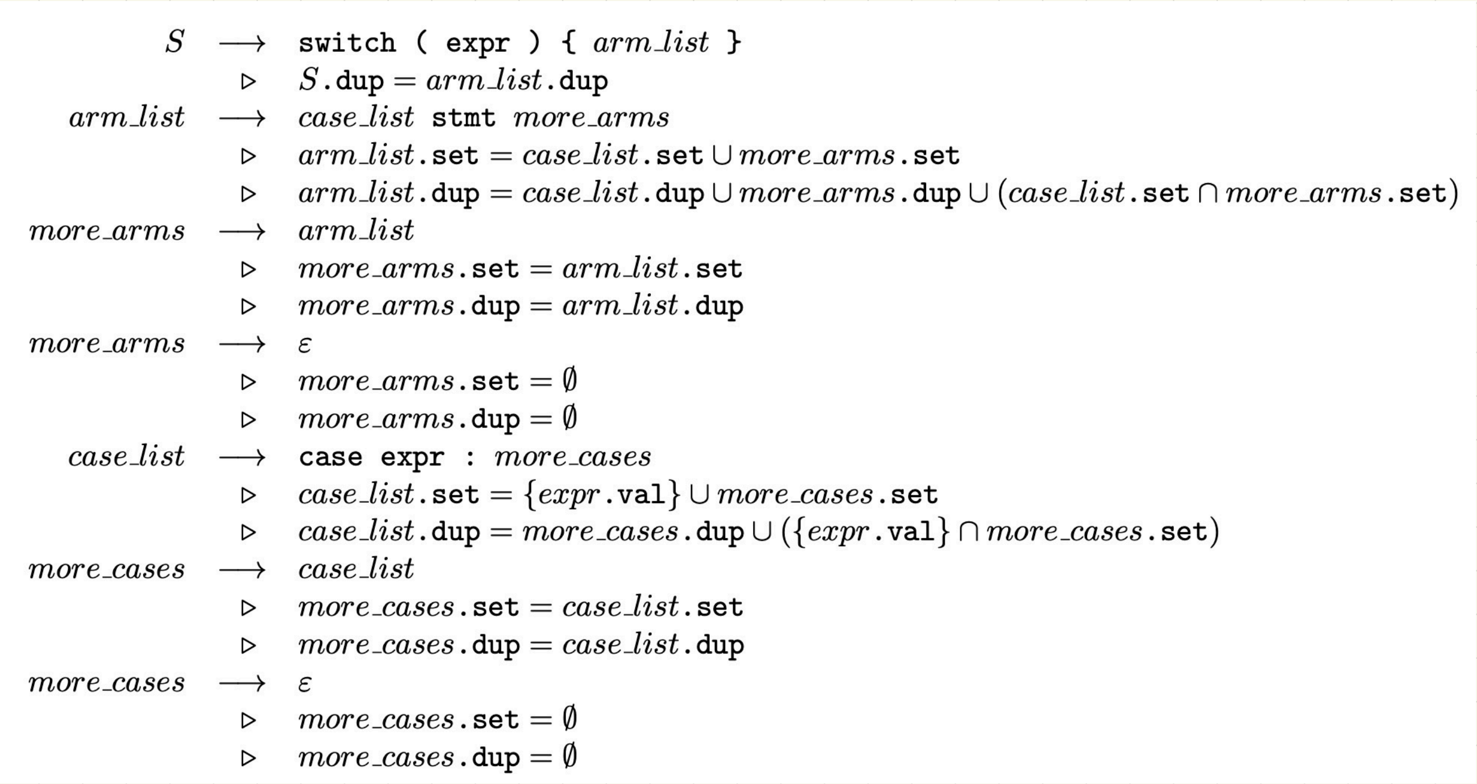
There are two states with potential conflicts : ① and ②.

① : and ∉ FOLLOW(E) => no conflict

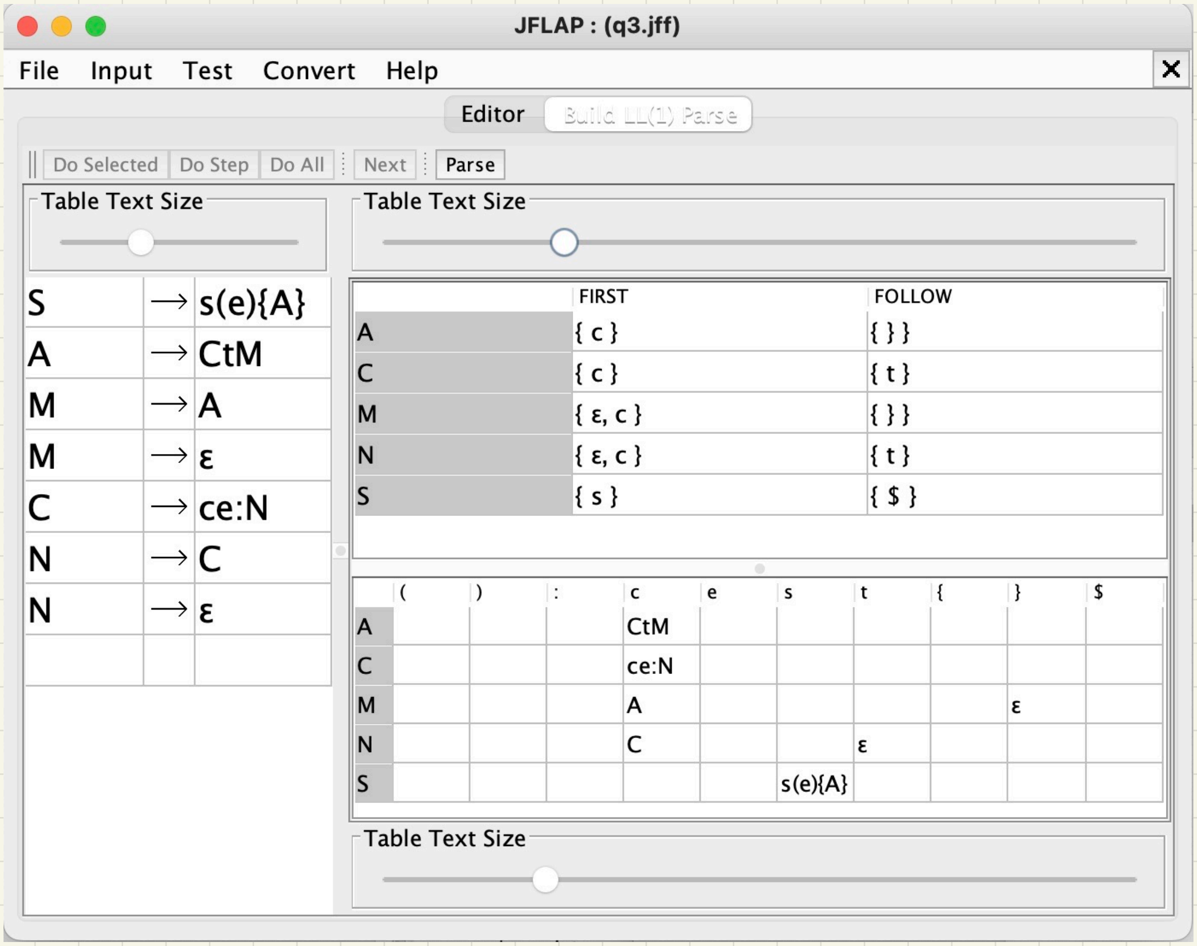
② same

=> <u>SLR(1)</u>

Q3 a) We separate the arms and cases, so that an arm can have several cases. Attribute <u>set</u> stores the values, <u>dup</u> stores duplicates.



$$\begin{array}{rcl}
S & \longrightarrow & \texttt{switch (expr) \{ } \mathit{arm\_list}\ \texttt{\}} \\
 & \triangleright & S.\texttt{dup} = \mathit{arm\_list}.\texttt{dup} \\
\mathit{arm\_list} & \longrightarrow & \mathit{case\_list}\ \texttt{stmt}\ \mathit{more\_arms} \\
 & \triangleright & \mathit{arm\_list}.\texttt{set} = \mathit{case\_list}.\texttt{set} \cup \mathit{more\_arms}.\texttt{set} \\
 & \triangleright & \mathit{arm\_list}.\texttt{dup} = \mathit{case\_list}.\texttt{dup} \cup \mathit{more\_arms}.\texttt{dup} \cup (\mathit{case\_list}.\texttt{set} \cap \mathit{more\_arms}.\texttt{set}) \\
\mathit{more\_arms} & \longrightarrow & \mathit{arm\_list} \\
 & \triangleright & \mathit{more\_arms}.\texttt{set} = \mathit{arm\_list}.\texttt{set} \\
 & \triangleright & \mathit{more\_arms}.\texttt{dup} = \mathit{arm\_list}.\texttt{dup} \\
\mathit{more\_arms} & \longrightarrow & \varepsilon \\
 & \triangleright & \mathit{more\_arms}.\texttt{set} = \emptyset \\
 & \triangleright & \mathit{more\_arms}.\texttt{dup} = \emptyset \\
\mathit{case\_list} & \longrightarrow & \texttt{case expr :}\ \mathit{more\_cases} \\
 & \triangleright & \mathit{case\_list}.\texttt{set} = \{\mathit{expr}.\texttt{val}\} \cup \mathit{more\_cases}.\texttt{set} \\
 & \triangleright & \mathit{case\_list}.\texttt{dup} = \mathit{more\_cases}.\texttt{dup} \cup (\{\mathit{expr}.\texttt{val}\} \cap \mathit{more\_cases}.\texttt{set}) \\
\mathit{more\_cases} & \longrightarrow & \mathit{case\_list} \\
 & \triangleright & \mathit{more\_cases}.\texttt{set} = \mathit{case\_list}.\texttt{set} \\
 & \triangleright & \mathit{more\_cases}.\texttt{dup} = \mathit{case\_list}.\texttt{dup} \\
\mathit{more\_cases} & \longrightarrow & \varepsilon \\
 & \triangleright & \mathit{more\_cases}.\texttt{set} = \emptyset \\
 & \triangleright & \mathit{more\_cases}.\texttt{dup} = \emptyset
\end{array}$$



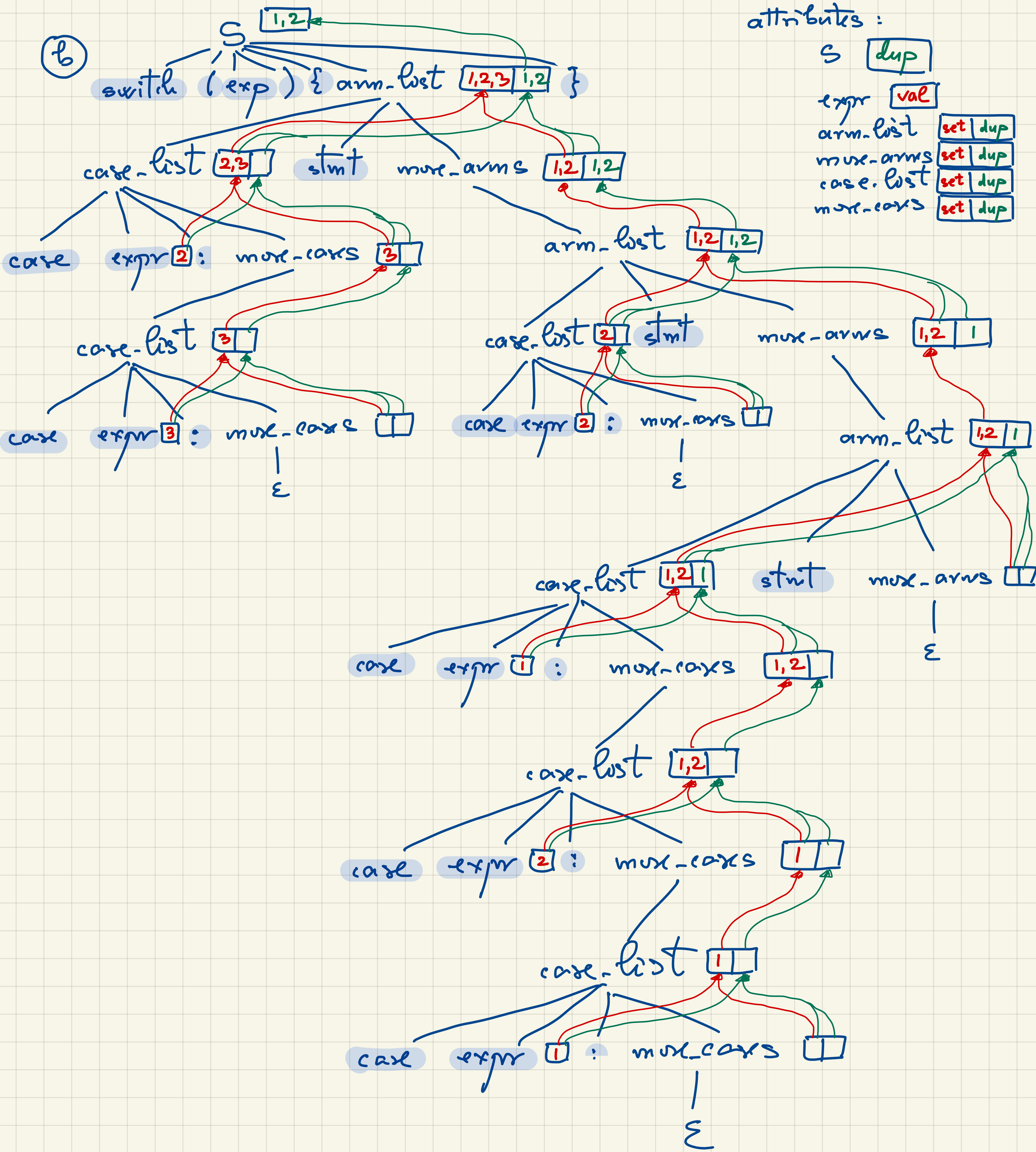
S	→	s(e){A}
A	→	CtM
M	→	A
M	→	ε
C	→	ce:N
N	→	C
N	→	ε

	FIRST	FOLLOW
A	{ c }	{ } }
C	{ c }	{ t }
M	{ ε, c }	{ } }
N	{ ε, c }	{ t }
S	{ s }	{ $ }

	(	)	:	c	e	s	t	{	}	$
A				CtM						
C				ce:N						
M				A					ε	
N				C			ε			
S						s(e){A}				

← no conflict
⇒ LL(1)

(b)


attributes:
S dup
expr val
arm_list set dup
mux_arms set dup
case_list set dup
mux_cases set dup
S 1,2
switch (exp) { arm_list 1,2,3 1,2 }
case_list 2,3
stmt
mux_arms 1,2 1,2
case expr 2 : mux_cases 3
arm_list 1,2 1,2
case_list 3
case_list 2 stmt
mux_arms 1,2 1
case expr 3 : mux_cases
ε
case expr 2 : mux_cars
ε
arm_list 1,2 1
case_list 1,2 1
stmt
mux_arms
ε
case expr 1 : mux_cases 1,2
case_list 1,2
case expr 2 : mux_cases 1
case_list 1
case expr 1 : mux_cars
ε