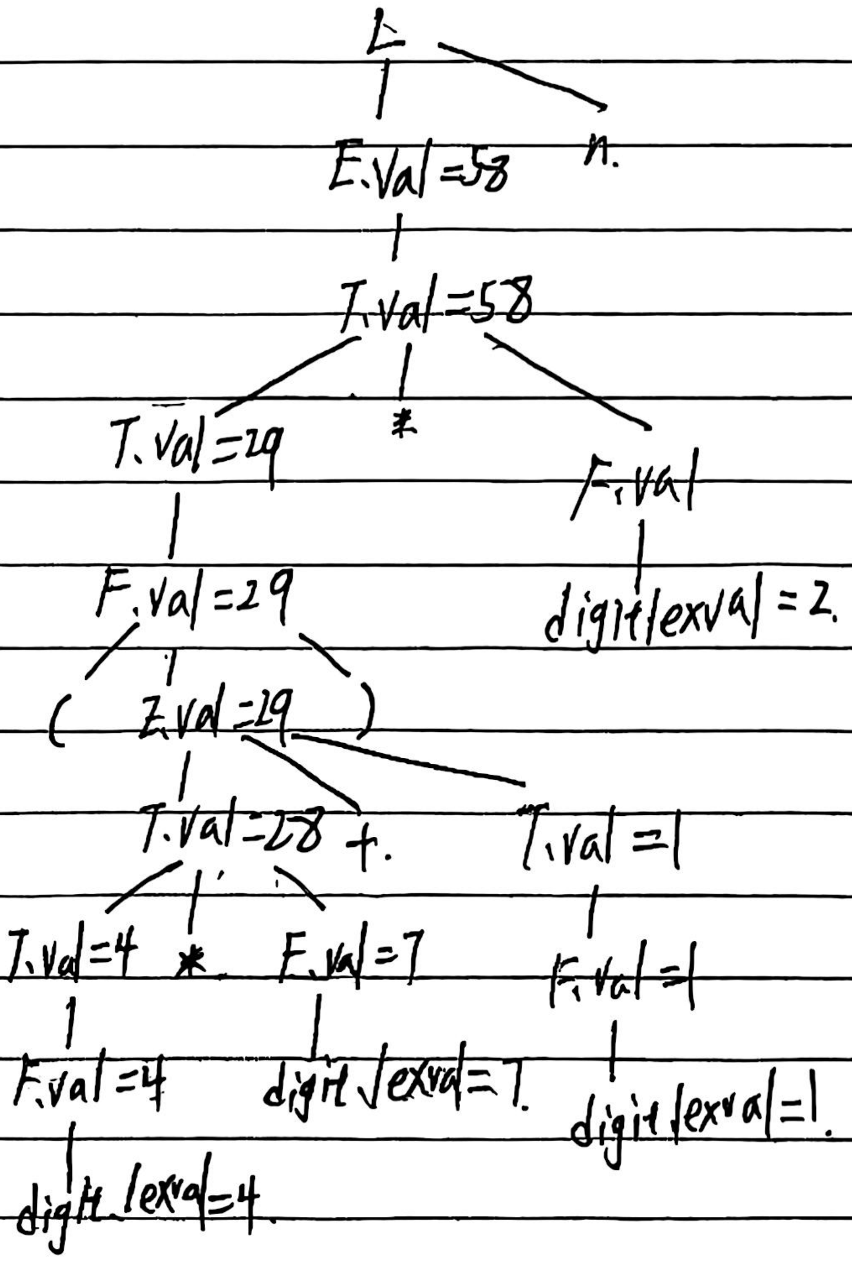
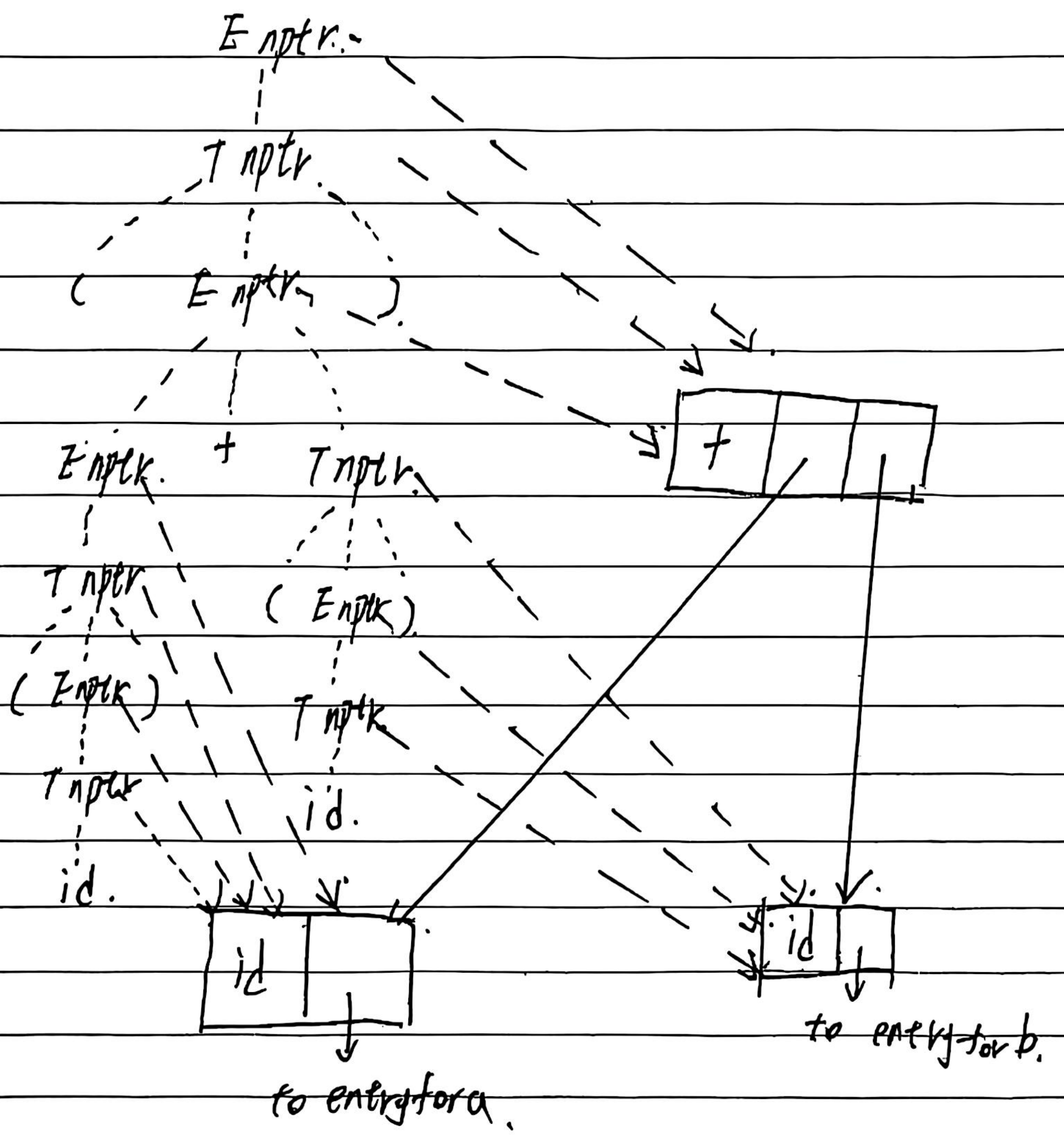
存在 移进-归约矛盾，所以不是 LR(1) 文法。

P104.

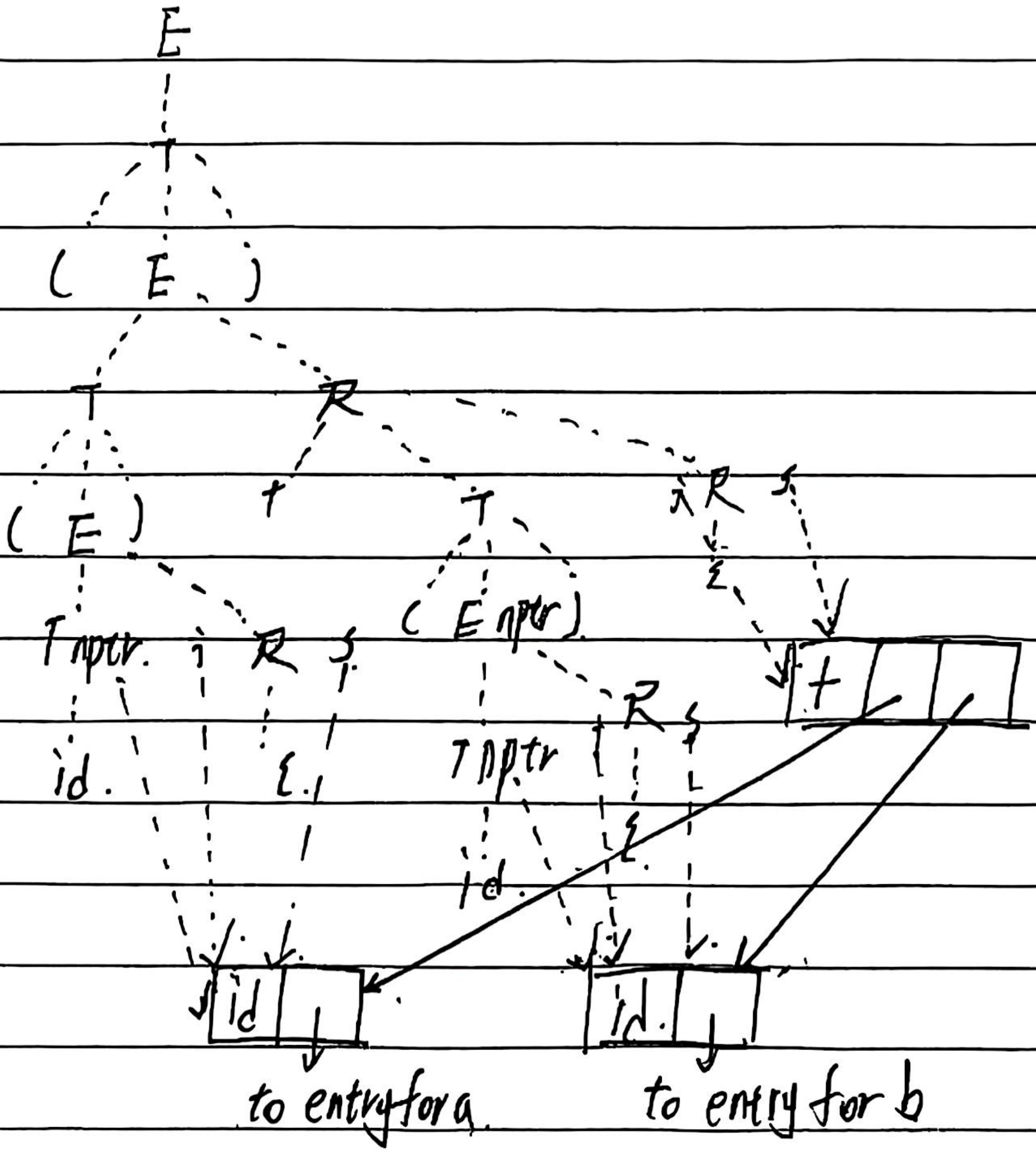
1.



2.


E nptr
T nptr
(
E nptr
)
E nptr
+
T nptr
T nptr
(E nptr)
T nptr
id
(E nptr)
T nptr
id
+
id
to entry for a
id
to entry for b

(2)



5、

(1). E→E1+T { if (E1.type = int) and (T.type = int)
then E.type := int
else E.type := real. }

E→T { E.type := T.type }

T→num.num { T.type := real }

T→num { T.type := int }.

(2).

```
E→E1+T     if E1.type=real and T.type=int then
               begin
                 E.type:=real;
                 print(T.lexeme);
                 print('inttoreal')
               end
           else if E1.type=int and T.type=real then
               Begin
                 E.type:=real;
                 print('inttoreal');
                 print(T.lexeme)
               end.
           else begin.
                   E.type:=E1.type;
                   print(T.lexeme)
                   end;
               print('+');
E→T        E.type:=T.type; print(T.lexeme)
T→num1.num2. T.type:=real; T.lexeme:=num1.lexeme||'.'||num2.lexeme.
T→num.     T.type:=int; T.lexeme:=num.lexeme.
```

7、 $S' \to S$	[print(S.val)]
$S \to L_1 . L_2$	[$S.val := L_1.val + L_2.val / 2^{L_2.lengh}$]
$S \to L$	[$S.val := L.val$]
$L \to L_1 B$.	[$L.val := L_1.val * 2 + B.val$]
	[$L.length := L_1.length + 1$].
$L \to B$	[$L.val := B.val$; $L.length := 1$]
$B \to 1$	[$B.val := 1$]
$B \to 0$	[$B.val := 0$].

八、

(1).

$D \to id\ L$	[addtype(id.entry, L.type)].
$L \to , id\ L_1$	[addtype(id.entry, L_1.type); L.type:=L_1.type;]
$L \to : T$	[L.type:=T.type]
$T \to$ integer.	[T.type:=integer]
$T \to$ real.	[T.type:=real].

(2).

```
procedure D;
  var L_type: Ttype.
    begin
      if sym="id" then.
```

```
        begin
          advance;
          1_type:=L;
          addtype(id.entry,1_type)
        end
    else error.
  end;
  procedure L;
      var 1_type: Ttype;
  begin
    if saym=",",then
      begin
      advance;
      if sym ="id" then
         begin
          advance;
          1_type:=L;
          addtype(id.entry,1_type)
        end
     else error;
```

```
      end
    else if sym=":" then
      begin
        advance;
        l_type:=T;
      end
      else error;
      return(l-type);
      end
    else error;
    return(l-type);
    end;
procedure T;
   var t-type:Ttype
begin
    if sym="integer" then
      begin
        advance;
        t_type:=integer;
      end.
```

```
else if sym = "real" then
     begin
       advance;
       t_type := real;
     end
else error;
 return(t_type);
 end;
```