Regular Expressions: E_Σ

Some properties : Occasions for remarks and exercises

- Are * and . such that: $(e_1.e_2)^* = e_1^*.e_2^*$? - Give a proof of your claim.
- Is | *left distributive* over . ? - Prove your claim.
- Prove that * is *idempotent*.
- Why the knowledge of the operator properties is a relevant task? - Show a concrete situation.

Exercises - 1

- Are * and . such that: $(e_1.e_2)^* = e_1^*.e_2^*$? - Give a proof of your claim.
  - No
  - Proof. (by counterexample) Let e1=a, e2=b. Then we show that for $s=a^2b$: $s \in e1^*e2^*$ but $s \notin (e1e2)^*$.

    $e1^*e2^* = (\cup_{i\in\aleph} e1^i)X.(\cup_{i\in\aleph} e2^i) = (\cup_{i\in\aleph}\{a\}^i)X.(\cup_{i\in\aleph}\{b\}^i)$

    $= \cup_{i,j\in\aleph} (\{a\}^i X.\{b\}^j) = \cup_{i,j\in\aleph}\{a^i b^j\}$

    hence $s \in \cup_{i,j\in\aleph} \{a^i b^j\}$, for i=2, j=1

    $(e1e2)^* = \cup_{i\in\aleph}(\{a\}X.\{b\})^i = \cup_{i\in}\{ab\}^i$

    hence $s \notin \cup_{i\in\aleph}\{ab\}^i$, because each *a* must be always, followed by one *b*.

- Is | *left distributive* over . ? - Prove your claim.
  - No. We prove that ∃ e1,e2,e3 such that: $(e1.e2)|e3 \neq (e1|e3).(e2|e3)$
  - Proof. ….

2

Exercises - 2

- Prove that * is *idempoten*

  proof: For all e, let Sem(e)=[e]. Then:

  $(e^*)^* = \cup_{i\in\aleph}(Sem(e^*))^i = \cup_{i\in\aleph}(\cup_{j\in\aleph} Sem(e)^j)^i$

  $= \cup_{i\in\aleph}(\cup_{j\in\aleph} [e]^j)^i = \cup_{i,j\in\aleph} [e]^{i*j}$

  since ($\forall$ i,j$\in\aleph$, i*j $\in\aleph$) and, ($\forall$n $\in\aleph$, $\exists$i,j $\in\aleph$ i*j=n), i*j computes only and all naturals when i,j are ranging on $\aleph$. Then,

  $\cup_{i,j\in\aleph} [e]^{i*j} = \cup_{i\in\aleph} [e]^i = e^*$

- Why the knowledge of the operator properties is a relevant task? - Show a concrete situation.
  - it helps in studying and simplifying problems and solutions that are using the operators
  - As a concrete situation consider: (e1*|e2*|e1*)* can be replaced by (e1|e2)

Exercise: Application of the Iteration Theorem

Problem. Prove that the language $L = \{a^n b^n \mid n \in \aleph, a,b \in \Sigma\}$, for given Σ, is not a regular language

Proof. (by contradiction, using pumping lemma)

- Assume L be regular.
- Then, the pumping lemma applies to L. Let m be the characteristic constant #S, of L, mentioned in the lemma.
- Then, let $x = a^m b^m$:
  - $x \in L$, since $m \in \aleph$
  - $|x| = 2m > m$, since $m>0$
  - hence, $\exists$ u,w, v:
    - $x = uwv$
    - $|uw| \leq m$ & $|w| \neq 0$:
    - hence, $\exists$ m1,m2,m3:
      - $m2 \neq 0$ & $m=m1+m2+m3$ & $w=a^{m2}$ & $x= a^{m1} a^{m2} a^{m3} b^m$
      - hence:
        - $a^{m1} a^{m3} b^m \in L$ according Lemma since $k*m2=0$ for $k=0$
        - $a^{m1} a^{m3} b^m \notin L$ by definition of L since $m1+m3 \neq m1+m2+m3$ when $m2 \neq 0$

Exercises – Part2

Consider the lexics L of the numerals for integer and fixed-point numbers in decimal notation and arbitrary number of digits.

1. Give separate regular expressions for: integers, S, fixed point integers, F, the union of the two.
2. Give a grammar for lexic L.
3. Give an automaton for L.
4. Give a deterministic automaton for L.
5. Give deterministic recognizer Y for L.
6. Modify Y for recognizing word sequences of L, separated by any character not in {0,..,9,' .' }. The new recognizer generates a sequence of <pi,li> where pi=position, li=length of the i-th recognized word. (words not in L are ignored).
7. Modify Y so that it recognizes the first, longer (7.1- shorter) word of the lexics that occurs in a string on an alphabet containing {0,..,9,' .' }.
8. Give the same of exercises 1.-4. for a lexics of signed and unsigned, integers and fixed-point integers, in decimal notation.

Exercise - 1

1. Give separate regular expressions for: integers, S, fixed point integers, F, the union of the two.

S = digit digit*
F = digit*.digit digit*
Digit = 0|1|…|9

Exercise - 2

2. Give a grammar for lexic L.

S = digit digit*
F = digit*.digit digit*
Digit = 0|1|…|9

It is a Regular Grammar since: digit < S,F

or

F = digit*.S
S = digit digit*
Digit = 0|1|…|9

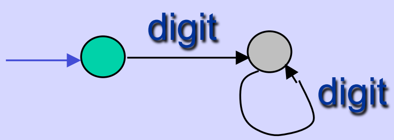
It is a Regular Grammar since: digit < S < F

Exercise – 8.2/8.3

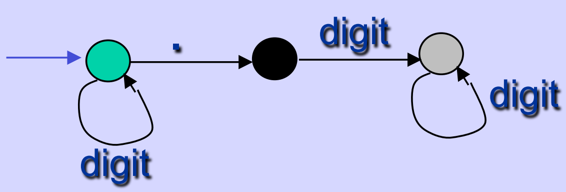
3. Give an automaton for L.

Remark. The use of digit as a single char simplifies the construction but it is formally incorrect ane may be source of mistakes in an implementation of the automaton.

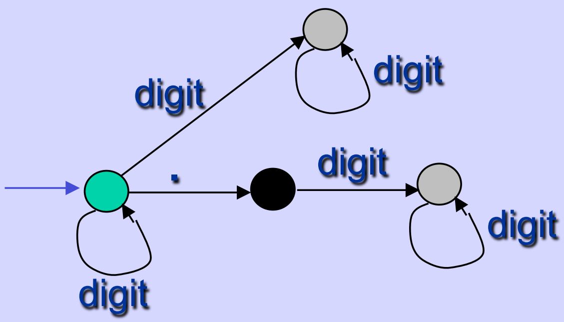
S = digit digit*



F = digit*. digit digit*



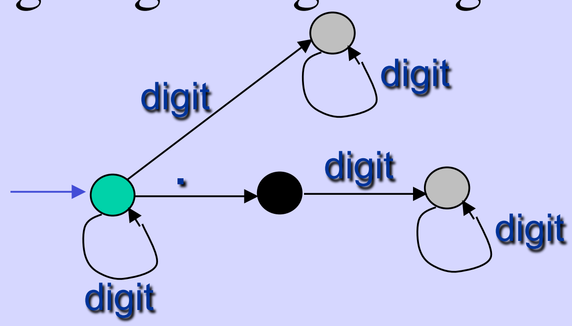
digit digit*| digit*. digit digit*



Exercise - 4

4. Give a deterministic automaton for L.

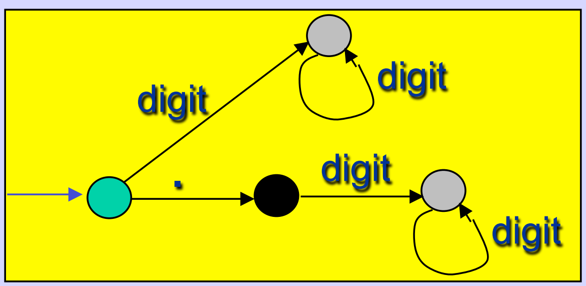
digit digit*| digit*. digit digit*



	digit	.
0	{0,1}	{2}
1	{1}	
2	{3}	
3	{3}	

Movestar

	digit	.
{0}	{0,1}	{2}
{0,1}	{0,1}	{2}
{2}	{3}	
{3}	{3}	



9

Exercise - 5

5. Give deterministic recognizer Y for L.

```
Answer StarDriver()
{s= s0;
 nextchar(c);
 while(c≠eof) and (s≠{}) {
          s=Star[s]c;
          nextchar(c);}
 if ((s∉F) or (c≠eof)) answer='noaccept'
 else answer='accept';
 return (answer);}
```

Tabella di analisi: **Star**

	digit	.
{0}	{0,1}	{2}
{0,1}	{0,1}	{2}
{2}	{3}	
{3}	{3}	

Exercise - 6

6. Modify Y for recognizing word sequences of L, separated by any character not in {0,..,9,' .' }. The new recognizer generates a sequence of <pi,li> where pi=position, li=length of the i-th recognized word. (words not in L are ignored).

```
Answer StarDriver()
{input=0;
  while(nextchar(c) != eof){
     length=0; cur=input;
     s= s0;
     while(c ∈Σ && s≠{}) {
              s=Star[s]c;
              nextchar(c); //incr. anche input
              length++;}
     if (s∈F) answer= answer  ++ <cur,length>;}
return (answer);}
```

Tabella di analisi: **Star**

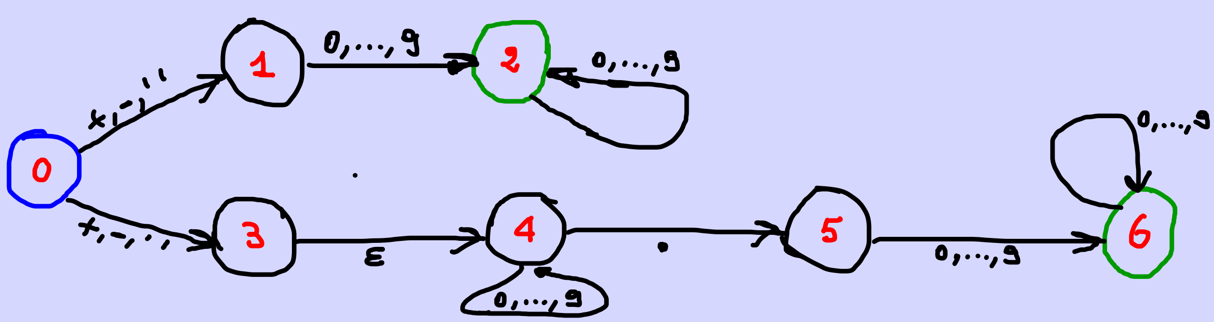
	digit	.
{0}	{0,1}	{2}
{0,1}	{0,1}	{2}
{2}	{3}	
{3}	{3}	

Exercise – 8.2/8.3

8.2. Give a grammar for signed and unsigned, integers and fixed-point integers, in decimal notation.

```
S ::= K D D*
F ::= K D*. D D*
D ::= 0|...|9
K ::= +|-|' '
```

8.3. Give an automaton for signed and unsigned, integers and fixed-point integers, in decimal notation.



Exercises – Part3

1. Let E=ab*b be a regular expression. Give the dotted automaton of E.
2. Let G be the grammar whose production set is:

   S::=AB
   A::=a
   B::=b*b

   where A is an auxiliary nonterminal. Give the dotted automaton of G.
3. Let A be the automaton whose transition function move is:

   move= {<<0,a>1>,<<1,b>2>,<<2,b>2>}

   where the start state is 0 and the final state set is {2}. Compute the minimal automaton of A.
4. Let A be the automaton whose transition function move is:

   move= {<<0,a>0>,<<0,b>1>,<<0,c>3>, <<1,a>0>,<<1,b>0>,<<1,c>2>,<<2,c>2>,<<3,c>2>}

   where the start state is 0 and the final state set is {2,3}. Give the minimal automaton.
5. Let A be the nondeterministic automaton whose transition function move is:

   move= {<<0,a>0>,<<0,a>1>,<<0,b>3>, <<1,a>0>,<<1,b>0>,<<1,c>2>,<<2,c>2>,<<3,c>2>}

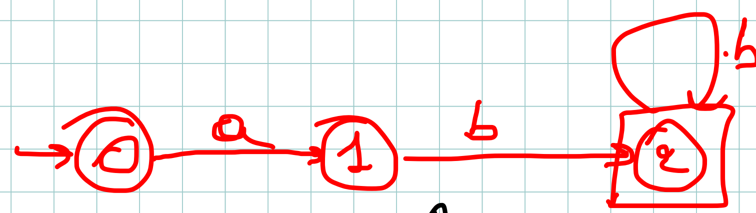
   where the start state is 0 and the final state set is {2,3}. Give the minimal automaton.
6. Operator "$^+$" is used, in regular expressions, with the meaning of "at least one".
   a. Give the semantics of operator "$^+$" for regular expressions, i.e. $[e^+]$ = ...
   b. Modify the algorithm of dotted autom,a for expressions, in order to include regular expressions with operator $^+$
7. Let G be the grammar whose production set is:

   S::=digit digit*
   F::=digit*.digit digit*

   Give the dotted automaton of G.

$E = a b^* b$

$I_0 = C_S(\{\triangle a b^* b\}) = C(\triangle a b^* b)$
$= \{\triangle a b^* b\}$



A

$M(0,a) = C_S(\{a \triangle b^* b\}) = C(\triangle b^* b)$
$= C(\triangle b b^* b) \cup C(\triangle b)$
$= \{\triangle b b^* b, \triangle b\} = I_1$

$M(1,b) = C_S(\{\triangle b^* b, \triangle\}) = C(\triangle b^* b) \cup \{\triangle\} = C(\triangle b b^* b) \cup C(\triangle b) \cup \{\triangle\}$
$= \{\triangle b b^* b, \triangle b, \triangle\} = I_2$

$M(2,b) = C_S(\{\triangle b^* b, \triangle\}) = I_2$

Is A a minimal automaton?

- $\Pi = \{\{0,1\},\{2\}\}$
- $\Pi_r = \{\{0\},\{1\},\{2\}\}$ because $M(0,a) = 1 \wedge M(1,a) = \bot$
- Π_r cannot be further refined
- $\Pi = \Pi_m$ and A is minimal

$S ::= AB$

$A ::= a$

$B ::= b^* b$

$I_0 = C_S(\{S ::=_\Delta AB,\ B ::=_\Delta b^* b\}) = C(S ::=_\Delta AB) \cup C(B ::=_\Delta b^* b)$

$= C(S ::=_\Delta aB) \cup \{B ::=_\Delta b b^* b,\ B ::=_\Delta b\}$

$= \{S ::=_\Delta aB,\ B ::=_\Delta b b^* b,\ B ::=_\Delta b\}$

$M(0,a) = C_S(\{S ::=_\Delta B\}) = C(S ::=_\Delta b^* b) = \{S ::=_\Delta b b^* b,\ S ::=_\Delta b\} = I_1$

$M(0,b) = C_S(\{B ::=_\Delta b^* b,\ B ::=_\Delta\}) = \{B ::=_\Delta b b^* b,\ B ::=_\Delta b,\ B ::=_\Delta\} \equiv I_2$

$M(1,b) = C_S(\{S ::=_\Delta b^* b,\ S ::=_\Delta\}) = \{S ::=_\Delta b b^* b,\ S ::=_\Delta b,\ S ::=_\Delta\} \equiv I_3$

$M(2,b) = C_S(\{B ::=_\Delta b^* b,\ B ::=_\Delta\}) \equiv I_2$

$M(3,b) = C_S(\{S ::=_\Delta b^* b,\ S ::=_\Delta\}) \equiv I_3$

Exercise - 7

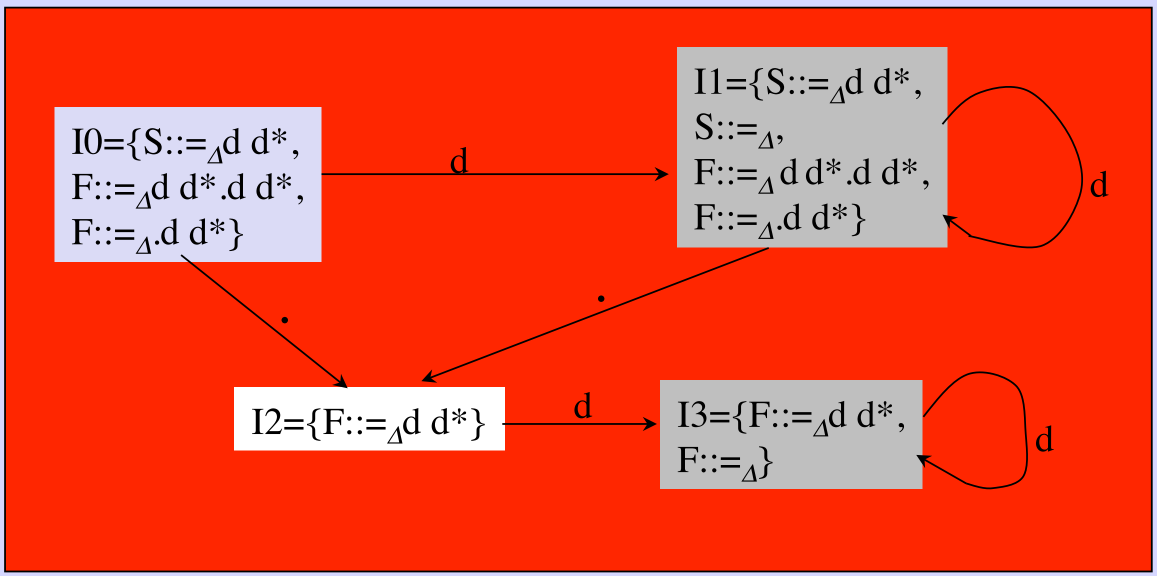
7. Let G be the grammar whose production set is:

S::=digit digit*

F::=digit*.digit digit*

Give the dotted automaton of G.

S::= digit digit*
F::= digit*.digit digit*



	digit	.
0	1	2
1	1	2
2	3	
3	3	