Set symbols

Set	{}	a collection of elements	A = {3,7,9,14}, B = {9,14,28}
Such that	|	such that	A = {x | x∊\mathbb{R}, x<0}
Intersection	A∩B	belongs to both A AND B at	A ∩ B = {9,14}
Union	A∪B	belongs to either A OR B	A ∪ B = {3,7,9,14,28}
Subset	A⊆B	A is a subset of B. Set A is included in set B.	{9,14,28} ⊆ {9,14,28}
Proper subset / strict subset	A⊂B	A is a subset of B, but A is not equal to B	{9,14} ⊂ {9,14,28}
Not a subset	A⊄B	Set A is not a subset of set B	{9,66} ⊄ {9,14,28}
Superset	A⊇B	A is a superset of B. Set A includes set B	{9,14,28} ⊇ {9,14,28}
Proper superset / strict superset	A⊃B	A is a superset of B, but B is not equal to A	{9,14,28} ⊃ {9,14}
Not superset	A⊅B	Set A is not a superset of set B	{9,14,28} ⊅ {9,66}
Power set	2^A	All subsets of A	
Power set	P(A)	All subsets of A	

Set symbols (cont)

Equality	A=B	Both sets have the same members	A={3,9,14}, B={3,9,14}, A=B
Complement	A^c	All the objects that do not belong to set A	
Relative complement	A\B or A-B	Objects that belong to A and not to B	A = {3,9,14}, B = {1,2,3}, A \ B = {9,14}
Symmetric difference	A∆B or A⊖B	Objects that belong to A or B but not to their intersection	A = {3,9,14}, B = {1,2,3}, A ∆ B = {1,2,9,14}
Element of	a∊A	Set membership	A={3,9,14}, 3 ∊ A
Not element of	x∉A	No set membership	A={3,9,14}, 1 ∉ A
Ordered pair	(a,b)	Collection of 2 elements	
Cartesian product	A×B	Set of all ordered pairs from A and B	
cardinality	|A| or #A	The number of elements of set A	A={3,9,14}, |A|=3

Set operations

Associativity	(A∪B) ∪ C = A∪(B∪C) (A∩B)∩C = A∩(B∩C)
Commutativity	A∪B = B∪A A∩B = B∩A
Complementation	A∪Not(A) = U A∩Not(A) = ∅
Idempotence	A∪A = A et A∩A
Identité	A∪∅ = A A∩U = A

Set operations (cont)

Extrémité	A∪U = U A∩∅ = ∅
Involution	Not(A) = A
Morgan's law	(Not(A∪B)) = Not(A)∩Not(B) (Not(A∩B)) = Not(A)∪Not(B)
Distribuvité	A∪(B∩C) = (A∪B) ∩ (A∪C) A∩(B∪C) = (A∩B) ∪ (A∩C)

Languages - Definitions

Formal language	vocabulary + grammar rules
Monoid	Set having an internal binary operation (+,-,*/, U,...) and having a neutral element ∊. Ex: <E,+,∊> means all strings of E will be concaneted with operator +
Grammar	(Vn, Vt, P, S) when: - Vn is a finite non-empty set whose elements are variables - VT is a finite non-empy set of terminal states - Vn∩ε = ∅ -P is a finite set whose elements are α -> β, known as production rules when α, β, ∊ (Vn∪ε)* but α should contain at least 1 symbol from Vn - S is a start symbol, where S ∊ Vn
Symbol	Element of a set. Ex for set A = {1, 2, 3}, 2 is an element
Alphabet	A finite and non empty set of symbols

Languages - Definitions (cont)

Length of a string	Number of symbols in a string. Ex: for w = aabbc, |w| = 5
Empty language	∅ contains no string
Empty string	ε where |ε| = 0
A^n	Word of length **n** from the alpabet **A**
A^*	Words of finite length from the alphabet A or null (can be empty: ε)
A^+	Words of finite length from the alphabet A and NOT NULL (no ε)
Regular expression	Rercursive language, the rules of the expression must be indepent.Ex: L1 = $\{a^n n^{2n} \| n>=0, m>= 0\}$ and L2 = $\{a^n b^m \mid n = 2m\}$ *Here **L1 is regular and L2 is not**. Because in L2 there is a dependency between n and m*

Maths set reminders

N	{0,1,2,3,...}
Z	{...,-3,-2,-1,0,1,2,3,...}
D	{d | d is a nb. having a finite number of decimals }
Q	{ r | r is a rational nb. that can be written as the quotien a/b of a real integer number a by a whole integer (not null) b}}
IR	{...,-3,-2,-1,0,Ɣ, 1, √2, 2, e, 3, π,...}
Successive inclusions	N ⊂ Z ⊂ D ⊂ Q ⊂ IR

Automata definitions

DFA	Deterministic Finite Automata. A DFA has a is deterministic because from each state we are able to determine the next state.
NDFA	Non Deterministic Finite Automata. A NDFA is non deterministic because we can't always determiner determine which will be the next state from the present state.

Operations on Languages

Union	L∪M = {x | x ϵ L or x ϵ M}
Intersection	L∩M = {x | x ϵ L and x ϵ M}
Difference(exclusion)	L\M = {x | x ϵ L and x ∉ M}
Complement on A^*	Comp(L) = A|*L = {x|x ϵ A* and x ∉ L}

L∪Ø = L
L∩Ø = L

Identity of Regular Expressions

Ø + R = R
Ø.R = R.Ø = Ø
Λ.R = R.Λ = R
Λ* = Λ and Ø* = Λ
R+R = R
R*R*=R*
R.R* = R.*R
(R*)=R*
Λ+R.R* = R* = Λ+R*.R
(P.Q)*P=P(Q.P)*
(P+Q)*=(P*.Q*)*=(P*+Q*)*
(P+Q).R=P.R+Q.R and R(P+Q)=R.P+R.Q

Graph types

Reflexif	All states have a loop, meaning δ(state1, input) = state1
Symetric	All states have a direct way back , meaning: δ(A, x) = B and δ(B,y) = A

Graph types (cont)

Anti symetric	There never is a direct way back to the same state meaning, if : δ(A,x) = B then δ(B, y) ≠ A
Transitive	There a shortcuts to a state, meaning if:

Grammar construction

Problem − *Suppose :*
L(Gr) = $\{a^n b^m c^k \mid n >= 0, k > 1, m = n+k\}$
We have to find out the grammar Gr which produces L(Gr).
Solution:
Possible word **w = $a^4b^9c^5$**
We need to decompose b^9: **w = $a^4b^4b^5c^5$**
Now we can define :
- **S1** having the **same number of a** followed by the **same number of b**.
- **S2** having the **same number of b** followed by the **same number of a**.

So:
S = S1 S2
S1 = a S1 b | Λ
S2 = b S2 a | bbcc
*Why **bbcc** as alternative to S2? Because in the case of a minimum word **w = b^2c^2 = bbcc** because k > 1 so in this case we do have m = n+k = 0+2*