EECS 510 INTRO TO THE THEORY OF COMPUTATION

JULY 3RD 2022

0.1 AUTOMATA, COMPUTABILITY, & COMPLEXITY

WE START BY ADDRESSING THE QUESTION, WHAT ARE THE <u>FUNDAMENTAL CAPABILITIES & LIMITATIONS OF COMPUTERS</u>? THIS QUESTION WAS ORIGINALLY ADDRESSED BY MATHMATICAL LOGICIANS BY EXPLORING THE MEANING OF COMPUTATION.

<u>COMPLEXITY THEORY</u>

COMPUTER PROBLEMS COMES IN DIFFERING VARITIES.

SORTING PROBLEMS — ARRANGE LIST OF $\mathbb{R}$ IN ASCENDING ORDER, 1MILL $\mathbb{R}$ IS EASY.

SCHEDULING PROBLEMS — FINDING A SCHEDULE OF CLASSES FOR AN ENTIRE UNIVERSITY TO SATISFY SOME REASONABLE CONSTRAINT, SUCH THAT NO TWO CLASSES TAKE PLACE IN THE SAME ROOM AT THE SAME TIME. THE SCHEDULING PROBLEM SEEMS TO BE MUCH HARDER THAN THE SORTING PROBLEM. IF YOU HAVE 1000 CLASSES, FINDING THE "BEST" SCHEDULE MAY REQUIRE CENTURIES, EVEN WITH A SUPERCOMPUTER.

THE CENTRAL QUESTION OF COMPLEXITY THEORY IS, WHAT MAKES PROBLEMS COMPUTATIONALLY HARD & OTHERS EASY? THERE IS ONE SCHEME FOR CLASSIFYING PROBLEMS ACCORDING TO THEIR COMPUTATIONAL DIFFICULTY. THIS SCHEME IS ANALOGOUS TO THE PERIODIC TABLE FOR CLASSIFYING ELEMENTS ACCORDING TO THEIR CHEMICAL PROPERTIES. WITH THIS SCHEME WE CAN DEMONSTRATE A METHOD FOR GIVING EVIDENCE THAT CERTAIN PROBLEMS ARE COMPUTATIONALLY HARD, EVEN IF WE ARE UNABLE TO PROVE THAT THEY ARE.

SCHEME :=

A CLASSIFICATION SCHEME IN INFORMATION SCIENCE & ONTOLOGY, A CLASSIFICATION SCHEME IS THE PRODUCT OF ARRANGING THINGS INTO KINDS OF THINGS (CLASSES).

ONTOLOGY :=

ENCOMPASSES A REPRESENTATION FORMAL NAMING, AND DEFINITION OF CATEGORIES, PROPERTIES, AND RELATIONS BETWEEN CONCEPTS, DATA, AND ENTITIES THAT SUBSTANTIATE ONE, MANY, OR ALL DOMAINS OF DISCORSE).

CONFRONTING A PROBLEM THAT IS COMPUTATIONALLY HARD:

1. UNDERSTAND WHAT ASPECT OF THE PROBLEM IS AT THE ROOT OF THE DIFFICULTY. ALTER THE PROBLEM SO ITS MORE EASILY SOLVABLE.
2. <u>SETTLE FOR A LESS THAN PERFECT SOLUTION TO THE PROBLEM</u> IN CERTAIN CASES, FINDING SOLUTIONS THAT ONLY APPROXIMATE THE PERFECT ONE IS RELATIVELY EASY.
3. SOME PROBLEMS ARE DIFFICULT IN THE WORST CASE SITUATION, BUT EASY MOST OF THE TIME.

PROBLEM $\longrightarrow$ PROCEDURE $\longrightarrow$ SOLUTION

CRYPTOGRAPHY IS AN APPLIED AREA THAT HAS BEEN DIRECTLY AFFECTED BY COMPLEXITY THEORY.

EECS 510 INTRO TO THE THEORY OF COMPUTATION

JULY 3RD 2022

0.1 AUTOMATA, COMPUTABILITY, & COMPLEXITY

COMPLEXITY THEORY

THE OBJECTIVE OF COMPLEXITY THEORY, THE OBJECTIVE IS TO CLASSIFY PROBLEMS AS EASY ONES AND HARD ONES. COMPUTABILITY THEORY IS THE CLASSIFICATION OF PROBLEMS BY THOSE THAT ARE SOLVABLE & THOSE THAT ARE NOT.

{NO COMPUTER ALGORITHM CAN DETERMINE WHETHER A MATHMATICAL STATEMENT IS TRUE OR FALSE.}

THE CONSEQUENCE OF THAT PHENOMENON LED TO THE PROFOUND RESULT THAT DEVELOPED THE IDEAS CONCERNING THEORITICAL MODELS OF COMPUTERS THAT LED TO THE CONSTRUCTION OF ACTUAL COMPUTERS.

AUTOMATA THEORY

AUTOMATA THEORY IS CONCERNED WITH PROPERTIES & DEFINITIONS OF MATHMATICAL MODELS OF COMPUTATION.

MATHMATICAL MODELS:

(1) FINITE AUTOMATON IS USED IN TEXT PROCESSING, COMPILERS, AND HARDWARE DESIGN.

(2) CONTEXT — FREE GRAMMER IS USED IN PROGRAMMING LANGUAGES AND ARTIFICAL INTELLIGENCE.

THE THEORIES OF COMPLEXITY & COMPUTABILITY REQUIRE A PRECISE DEFINITION OF A COMPUTER. AUTOMATA THEORY ALLOWS PRACTICE WITH FORMAL DEFINITIONS OF COMPUTATION.

0.2 MATHMATICAL NOTATIONS & TERMINOLOGY

INTRO TO OBJECTS, TOOLS, AND NOTATION WE EXPECT TO USE.

(1) SETS := GROUP OF OBJECTS REPRESENTED AS A UNIT.

(2) OBJECTS := ELEMENTS OR MEMBERS OF A SET $\in$

SET NOTATION

MEMBER / OBJECT / ELEMENT

$$\boxed{S} = \{\boxed{7}, 21, 57\}$$

SET

$$7 \in S \quad || \quad 7 \in \{7, 21, 57\}$$
$$8 \notin S \quad || \quad 8 \notin \{7, 21, 57\}$$

SUBSET NOTATION

FOR TWO SETS A & B IF $\forall$ MEMBER OF A IS ALSO A MEMBER OF B WE SAY,

$A \subseteq B$:= A SUBSET OF B.

IF A IS A SUBSET OF B AND IS NOT $\neq$ TO B.

$A \subsetneq B$:= A IS A PROPER SUBSET OF B

ORDERS OF SETS DO NOT MATTER NOR DOES IT'S REPETING NUMBERS.

$$\{7, 21, 57\} = \{57, 7, 7, 7, 21\}$$

MULTISET CARES ABOUT DUPLICATIVE OCCURANCES

$$\{7\} \neq \{7, 7, 7\}$$

INFINITE SET

$\{\ldots\}$ WHY IN TEXTIN ooo OCCURS BECAUSE AN INFINITE AMOUNT OF STD::STRING/STR ARE POSSIBLE.

$\{\ldots\}$:= CONTINUE THE SEQUENCE FOREVER.

$\mathbb{N} := \{1, 2, 3, \ldots\}$

$\mathbb{Z} := \{\ldots, -2, -1, 0, 1, 2, \ldots\}$

EMPTY SET / NULL := $\{\emptyset\}$, E.G. $A \cap \emptyset = \emptyset$, BUT $A \cup \emptyset = A$

SET WITH ONE MEMBER IS A SINGLETON SET.

SET WITH TWO MEMBERS IS AN UNORDERED PAIR.

$\{N \mid$ RULE ABOUT $N\}$

$\{N \mid N = M^2$ FOR SOME $M \in \mathbb{N}\}$:= THE SET OF PERFECT SQUARES

UNION

$A \cup B$, $\forall$ ELEMENT IN A & B IS COMBINED INTO ONE SET

IS THE SET WE GET WHEN WE GET WHEN COMBINING ALL OF THE ELEMENTS IN A & B INTO A SINGLE SET. THUS,

$$A \cup B := \{x \in S \mid x \in A \text{ or } x \in B\}$$

E.G. $A := \{1, 2, 3\}$, $x_1 = 1$, $x_2 = 2$, $x_3 = 3$

$B := \{4, 5, 6\}$, $x_4 = 4$, $x_5 = 5$, $x_6 = 6$

$A, B \subset S$ $\quad S := \{\{1, 2, 3\}, \{4, 5, 6\}\}$

INTERSECTION

$A \cap B$ SET OF ELEMENTS THAT ARE IN BOTH A & B.

E.G. $A \cap B := \{x \in A \mid x \in B\}$

$A := \{-2, -1, 0, 1, 2, 3\}$

$B := \{1, 2, 3, 4, 5, 6\}$

$A, B \subset S$, $\quad S := \{1, 2, 3\}$

COMPLEMENT

$\overline{A}$, IS THE SET OF ALL ELEMENTS UNDER CONSIDERATION THAT ARE NOT IN A.

VENN DIAGRAM

A PICTURE OFTEN HELPS CLARIFY A CONCEPT

A VENN DIAGRAM REPRESENTS SETS AS REGIONS ENCLOSED BY CIRCULAR LINES.

EECS 510 INTRO TO THE THEORY OF COMPUTATION

JULY 3RD 2022

0.1 AUTOMATA, COMPUTABILITY, & COMPLEXITY

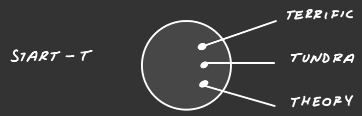
VENN DIAGRAM cont.

LET THE SET START-T BE THE SET OF ALL ENGLISH WORDS THAT START WITH THE LETTER "T".

CIRCLE REPRESENTS START-T

POINTS REPRESENTS AS POINTS INSIDE THE CIRCLE.

(0.1) START-T - VENN DIAGRAM FOR THE SET OF ENGLISH WORDS STARTING WITH "T"



(0.2) END-Z - VENN DIAGRAM FOR THE SET OF ENGLISH WORDS ENDING WITH "Z"

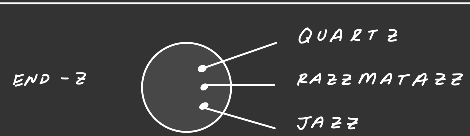


FIG (0.3) OVERLAPPING CIRCLES INDICATE COMMON ELEMENTS

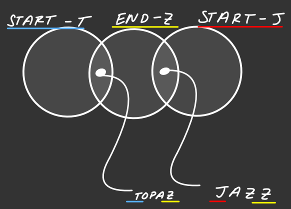
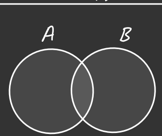
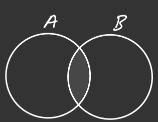


FIG (0.4) DIAGRAMS FOR $A \cup B$ & $A \cap B$



$$A \cup B := \{x \in S \mid x \in A \text{ or } x \in B\}$$



$$A \cap B := \{x \in A \mid x \in B\}$$

OR

SEQUENCES & TUPLES

A SEQUENCE OF OBJECTS IS A LIST OF THOSE OBJECTS IN SOME ORDER. SEQUENCES ARE DESIGNATED BY WRITING THE LIST WITH PARENTHESIS.

$$(7, 21, 57) \neq (57, 7, 21)$$

REPETITION DOES MATTER IN A SEQUENCE

REPETITION DOES NOT MATTER IN A SET.

$$(7, 7, 21, 57) \neq (7, 21, 57) \neq (57, 7, 21)$$

$$\{7, 7, 21, 57\} = \{7, 21, 57\}$$

EECS 510 INTRO TO THE THEORY OF COMPUTATION

JULY 3RD 2022

0.1 AUTOMATA, COMPUTABILITY, & COMPLEXITY

SEQUENCES MAYBE FINITE OR INFINITE

FINITE SEQUENCES ARE CALLED <u>TUPLES</u>

A SEQUENCE WITH K-ELEMENTS IS A <u>K-TUPLE</u>

THUS, TUPLE EXAMPLES ARE,

$(7, 21, 57)$ IS A 3-TUPLE

$(7, 21)$ IS ORDERED PAIR

SET & SEQUENCES MAY APPEAR AS ELEMENTS OF OTHER SETS & SEQUENCES. FOR EXAMPLE THE POWER SET OF A IS THE SET OF ALL SUBSETS OF A.

POWERSETS

IF $A := \{0, 1\}$

THE $P(A) := \{\emptyset, \{0\}, \{1\}, \{0, 1\}\}$

THE SET OF ALL ORDERED PAIRS WHOSE ELEMENTS ARE 0s & 1s IS:

$\{(0,0), (0,1), (1,0), (1,1)\}$

THE CARTESIAN PRODUCT / CROSS PRODUCT OF A & B

$A \times B$ IS THE SET OF ALL ORDERED PAIRS WHEREIN THE FIRST ELEMENT IS A MEMBER OF A AND THE SECOND IS A MEMBER OF B.

EXAMPLE (0.5)

IF, $A = \{1, 2\}$ & $B = \{x, y, z\}$

THEN, $A \times B = \{(1,x), (1,y), (1,z), (2,x), (2,y), (2,z)\}$

CARTESIAN PRODUCT OF K SETS

$A_1, A_2, A_3, \ldots, A_K$ WRITTEN AS $A_1 \times A_2 \times A_3 \times \ldots \times A_K$

IS THE SET CONSISTING OF ALL K-TUPLES $(a_1, a_2, \ldots, a_K)$ WHERE $a_i \in A_i$

EXAMPLE (0.6)

IF, $A = \{1, 2\}$ & $B = \{x, y, z\}$

THEN, $A \times B \times A := \{(1,x,1), (1,x,2), (1,y,1), (1,y,2), (1,z,1), (1,z,2),$
$(2,x,1), (2,x,2), (2,y,1), (2,y,2), (2,z,1), (2,z,2)\}$

IF WE HAVE THE CARTESIAN PRODUCT OF A SET WITHIN ITSELF, WE USE THE SHORTHAND

$$\underbrace{A \times A \times A \times \cdots \times A}_{K} = A^K$$

EXAMPLE (0.7)

THE SET $N^2 = N \times N$. IT CONSISTS OF ALL ORDERED PAIRS OF NATURAL NUMBERS.
WE ALSO MAY WRITE IT AS $\{(i,j) \mid i,j \geq 1\}$

FUNCTIONS AND RELATIONS

FUNCTIONS ARE CENTRAL TO MATHMATICS.
A FUNCTION IS AN OBJECT THAT SETS UP AN INPUT-OUTPUT RELATION.
A FUNCTION TAKES AN INPUT AND PRODUCES AN OUTPUT.
IN EVERY FUNCTION, THE SAME INPUT ALWAYS PRODUCES THE SAME OUTPUT.

IF f IS A FUNCTION WHOSE OUTPUT VALUE IS B WHEN THE INPUT VALUE OF a, WE WRITES

$$f(A) = B$$

A FUNCTION IS ALSO CALLED A MAPPING

IF $f(A) = B$ THEN "f MAPS A TO B"

EXAMPLE

```
def abs(a):
    …
    return(b)
```

THE ABSOLUTE FUNCTION TAKES a AS AN INPUT AND RETURNS b IF a IS POSITIVE AND $-b$ IF a IS NEGATIVE. THUS $abs(2) = abs(-2) = 2$. SAME WITH ADDITION.

THE SET OF POSSIBLE INPUTS TO THE FUNCTION IS CALLED ITS DOMAIN.
THE OUTPUTS OF A FUNCTION COME FROM A SET CALLED ITS RANGE
THE NOTATION FOR SAYING THAT f IS A FUNCTION WITH DOMAIN D AND RANGE R IS,

$$f : \underbrace{D}_{\text{INPUT}} \longrightarrow \underbrace{R}_{\text{OUTPUT}}$$

MAPPING THE ABSOLUTE VALUE FUNCTION

$$f : D \longrightarrow R$$

$ABS : \mathbb{Z} \longrightarrow \mathbb{Z}$ (ABS TAKES ONE PARAMETER)

IN THE CASE OF THE ABSOLUTE VALUE FUNCTION WE ARE WORKING WITH ALL INTEGERS $\mathbb{Z}$

$ADD : \mathbb{Z} \times \mathbb{Z} \longrightarrow \mathbb{Z}$ INT ADD (INT X1, INT X2) $\mathbb{Z} \times \mathbb{Z}$

IN THE CASE OF THE ADDITION FUNCTION FOR INTEGERS, THE DOMAIN IS THE SET OF PAIRS OF INTEGERS $\mathbb{Z} \times \mathbb{Z}$.

THE FUNCTION ABS : $\mathbb{Z} \longrightarrow \mathbb{Z}$ MAY NOT NECESSARILY USE ALL THE ELEMENTS OF THE SPECIFIED RANGE. FOR EXAMPLE EVEN THOUGH $-1 \in \mathbb{Z}$ THE RANGE WILL NEVER TAKE ON THAT VALUE -1.
THEREFORE A FUNCTION THAT DOES USE ALL THE ELEMENTS OF THE RANGE IS SAID TO BE $\boxed{\text{ONTO THE RANGE.}}$

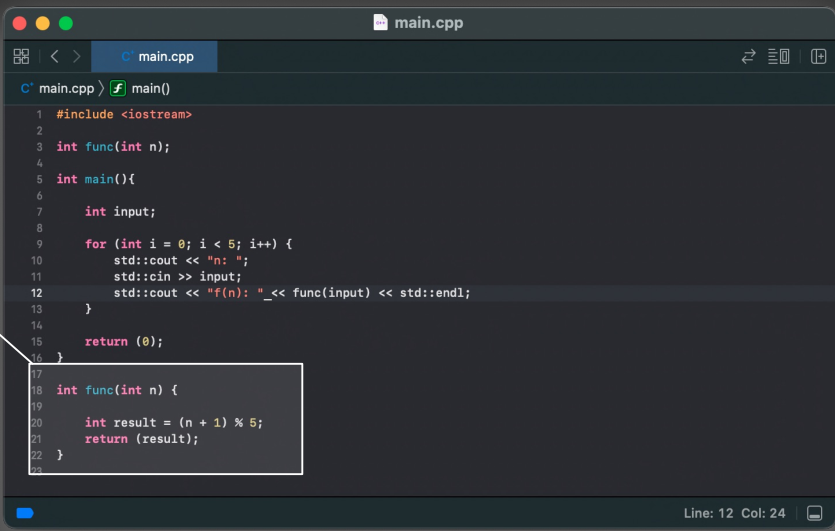
ANOTHER FORM OF MAPPING NOTATION IS A PROCEDURE FOR COMPUTING AN OUTPUT FROM A DEFINED INPUT, BY PROVIDING A TABLE.

EXAMPLE (0.8)

CONSIDER THE FUNCTION, $f : \{0, 1, 2, 3, 4\} \longrightarrow \{0, 1, 2, 3, 4\}$

N	F(N)
0	1
1	2
2	3
3	4
4	0

$$\boxed{f : \mathbb{Z}_5 \longrightarrow \mathbb{Z}_5}$$




```cpp
#include <iostream>

int func(int n);

int main(){

    int input;

    for (int i = 0; i < 5; i++) {
        std::cout << "n: ";
        std::cin >> input;
        std::cout << "f(n): " << func(input) << std::endl;
    }

    return (0);
}

int func(int n) {

    int result = (n + 1) % 5;
    return (result);
}
```

$F(N) = (N+1) \bmod 5$

WHEN WE DO MODULAR ARITHMETIC WE DEFINE $\mathbb{Z}_m = \{0, 1, 2, \dots, m-1\}$
WITH THIS NOTATION, THE AFOREMENTIONED FUNCTION f HAS THE FORM $f : \mathbb{Z}_5 \longrightarrow \mathbb{Z}_5$

EECS 510 INTRO TO THE THEORY OF COMPUTATION

JULY 3RD 2022

0.1 AUTOMATA, COMPUTABILITY, & COMPLEXITY

EXAMPLE (0.9) FUNCTIONS & RELATIONS

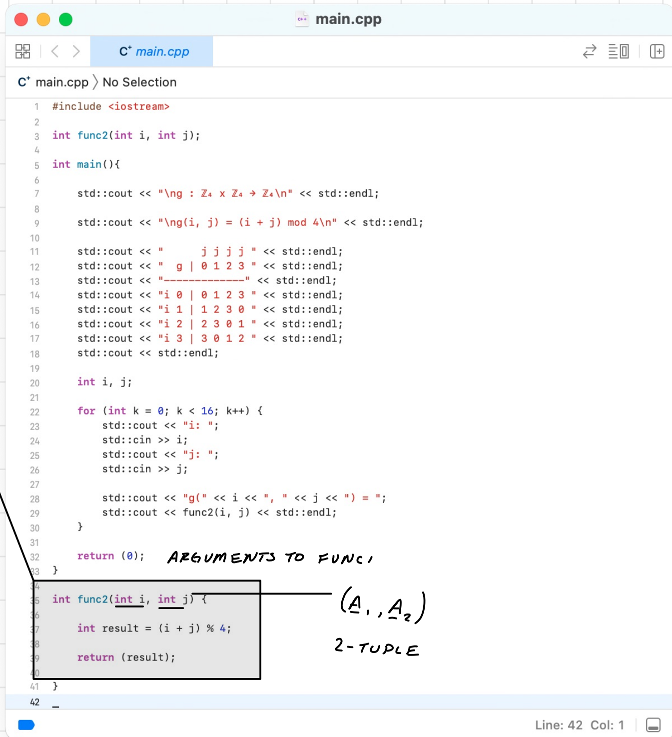
A 2D TABLE IS MADE IF THE FUNCTION TAKES IN TWO INPUTS, IF THE FUNCTION TAKES TWO INPUTS THEN THE DOMAIN OF THAT FUNCTION IS THE CARTESIAN PRODUCT OF TWO SETS.

$$g : \mathbb{Z}_4 \times \mathbb{Z}_4 \longrightarrow \mathbb{Z}_4$$

$$g(i, j) = (i + j) \bmod 4$$

THE ENTRY AT THE ROW LABELED i AND THE COLUMN LABLED j IN THE TABLE IS THE VALUE OF $g(i,j)$

		j_0	j_1	j_2	j_3
	g	0	1	2	3
i_0	0	0	1	2	3
i_1	1	1	2	3	0
i_2	2	2	3	0	1
i_3	3	3	0	1	2



```cpp
#include <iostream>

int func2(int i, int j);

int main(){

    std::cout << "\ng : ℤ₄ x ℤ₄ → ℤ₄\n" << std::endl;

    std::cout << "\ng(i, j) = (i + j) mod 4\n" << std::endl;

    std::cout << "      j j j j " << std::endl;
    std::cout << "  g | 0 1 2 3 " << std::endl;
    std::cout << "-------------" << std::endl;
    std::cout << "i 0 | 0 1 2 3 " << std::endl;
    std::cout << "i 1 | 1 2 3 0 " << std::endl;
    std::cout << "i 2 | 2 3 0 1 " << std::endl;
    std::cout << "i 3 | 3 0 1 2 " << std::endl;
    std::cout << std::endl;

    int i, j;

    for (int k = 0; k < 16; k++) {
        std::cout << "i: ";
        std::cin >> i;
        std::cout << "j: ";
        std::cin >> j;

        std::cout << "g(" << i << ", " << j << ") = ";
        std::cout << func2(i, j) << std::endl;
    }

    return (0);
}

int func2(int i, int j) {

    int result = (i + j) % 4;

    return (result);

}
```


ARGUMENTS TO FUNC1

$(\underline{A}_1, \underline{A}_2)$

2-TUPLE

WHEN THE DOMAIN OF THE FUNCTION, f IS $A_1 \times A_2 \times \dots \times A_k$ FOR SOME SETS $A_1, \dots, A_k$ THE INPUT TO f IS A K-TUPLE $(\underline{A}_1, \underline{A}_2, \dots, \underline{A}_k)$ AND WE CALL THE $\underline{A}_i$ THE ARGUMENTS TO f.

A FUNCTION WITH K-ARGUMENTS IS A K-ARY FUNCTION

K IS THE ARITY OF THE FUNCTION.

K-ARY FUNCTION — K NUMBER OF ARGUMENTS

UNARY FUNCTION — 1 ARGUMENT

BINARY FUNCTION — 2 ARGUMENTS

INFIX NOTATION FOR BINARY FUNCTIONS HAVE A SYMBOL BETWEEN THE TWO ARGUMENTS

PREFIX NOTATION SYMBOL PRECEDING IT.

$\underline{A + B}$ — INFIX NO.

$\underline{ADD(A, B)}$ — PREFIX NO.

EECS 510 INTRO TO THE THEORY OF COMPUTATION

JULY 3RD 2022

0.2 MATHMATICAL NOTATION & TERMINOLOGY

PREDICATE / PROPERTY

IS A FUNCTION WHOSE RANGE IS {TRUE, FALSE}

~~BINARY OR BOOLEAN FUNCTION~~ (E.G. BOOL EVEN(4) = TRUE // BOOL EVEN(5) = FALSE)

PROPERTY WHOSE DOMAIN IS THE SET OF K-TUPLES $A \times \ldots \times A$ IS A RELATION, K-ARY RELATION, OR K-ARY RELATION ON A.

BINARY RELATION

$<$ "LESS THAN" INFIX OPERATION SYMBOL FOR RELATIONS

$=$ "EQUALITY" INFIX OPERATION SYMBOL FOR RELATIONS

IF R IS A BINARY RELATION THEN aRb MEANS THAT,

$aRb = \text{TRUE}$

IF R IS A K-ARY RELATION, THEN

$R(a_1, a_2, \ldots, a_k)$ MEANS,

$R(a_1, a_2, \ldots, a_k) = \text{TRUE}$

EECS 510 INTRO TO THE THEORY OF COMPUTATION

JULY 3RD 2022

0.2 MATHMATICAL NOTATION & TERMINOLOGY

DESCRIBING A PREDICATE WITH SETS INSTEAD OF FUNCTIONS CAN BE MORE CONVIENT

A PREDICATE IS A FUNCTION WHOSE RANGE/RETURN VALUE

PREDICATE

$$P : D \to \{ \text{TRUE}, \text{FALSE} \}$$

$$\Updownarrow$$

$$S = \{ a \in D \mid P(a) = \text{TRUE} \}$$

THUS THE RELATION BEATS MAYBE WRITTEN AS,

{ (SCISSORS, PAPER), (PAPER, STONE), (STONE, SCISSORS) }

EQUIVALENCE RELATION

CAPTURES THE NOTION OF TWO OBJECTS BEING EQUAL IN SOME FEATURE.

A BINARY RELATION R IS AN EQUIVALENCE IFF:

1. R IS REFLEXIVE IF FOR EVERY x, xRx;
2. R IS SYMMETRIC IF FOR EVERY x AND y, xRy IMPLIES yRx; AND
3. R IS TRANSITIVE IF FOR EVERY x, y, AND z, xRy AND yRz IMPLIES xRz

EXAMPLE (0.11)

DEFINE AN EQUIVALENCE RELATION ON THE NATURAL NUMBERS WRITTEN,

$$\equiv_7$$

$i \equiv j \bmod 7$

$i \% 7 = j \% 7$

$i \bmod 7 = j \bmod 7$

FOR, $i, j \in \mathcal{N}$

SAY THAT, $i \equiv_7 j$

IF, $i - j$ IS A MULTIPLE OF 7

THIS IS AN EQUIVALENCE RELATION BECAUSE IT SATISFIES THREE CONDITIONS:

1. REFLEXIVE, AS $i - i = 0$, WHICH IS A MULTIPLE OF $7 \to 0 = 7x$
2. SYMMETRIC, AS $i - j$ IS A MULTIPLE OF $7 \longrightarrow (i-j) = 7x$

   IF $j - i$ IS A MULTIPLE OF $7 \longrightarrow (j-i) = 7x$
3. TRANSITIVE

AS WHENEVER $i - j$ IS A MULTIPLE OF 7

AND $j - k$ IS A MULTIPLE OF 7

THEN $(i-k) = (i-j) + (j-k)$ IS THE SUM OF TWO MULTIPLES OF 7

AND HENCE A MULTIPLE OF 7 TOO

EECS 510 INTRO TO THE THEORY OF COMPUTATION

JULY 8TH 2022

0.2 MATHMATICAL NOTATION & TERMINOLOGY

GRAPHS

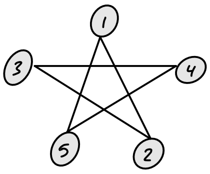
GRAPHS

AN UNDIRECTED GRAPH OR SIMPLY A GRAPH IS A SET OF POINTS WITH LINES CONNECTING SOME OF THE POINTS.

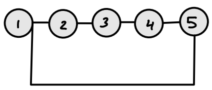
THE POINTS ARE CALLED NODES OR VERTICES.

THE EDGES OF A GRAPH ARE THE LINES WHICH CONNECT THESE NODES.

EXAMPLE (0.12)



ALL NODES ARE OF DEGREE 2



HEAD → HEAD.NEXT

TAIL → HEAD

THE NUMBER OF EDGES AT A PARTICLAR NODE IS THE DEGREE OF THAT NODE.

NO MORE THAN ONE EDGE IS ALLOWED BETWEEN ANY TWO NODES

WE MAY ALLOW AN EDGE FROM A NODE TO ITSELF, CALLED A SELF-LOOP

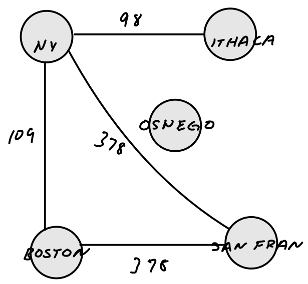
IN A GRAPH G THAT CONTAINS NODES (i, j), THE PAIR (i, j) REPRESENTS THE EDGE THAT CONNECTS i & j.

TUPLE ORDER IS IRRELEVANT IN AN UNDIRECTED GRAPH.

GRAPHS FREQUENTLY ARE USED TO REPRESENT DATA.

NODES CAN REPRESENT ANYTHING (PEOPLE, CITIES, ETC.) FOR THIS WE HAVE LABELED GRAPHS.

IN THE FOLLOWING NODES ARE CITIES WHOSE EDGES ARE LABELED WITH THE DOLLAR COST OF THE CHEAPEST NON STOP AIRFARE.



SUBGRAPHS ARE GRAPHS WHERE THE NODES OF THE SUBGRAPH ARE A SUBSET OF THE NODES OF THE ORIGINAL GRAPH & THE EDGES OF THE SUBGRAPH