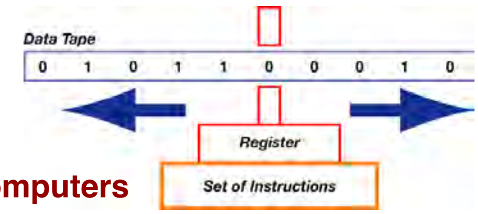
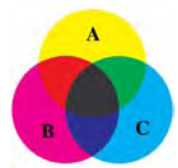
Computers, Computing, and Sets

Robert Stengel
Robotics and Intelligent Systems MAE 345,
Princeton University, 2017



Learning Objectives

- **Principles of Turing Machine**
- **Evolution of Calculators and Computers**
- **Programming**
- **Crisp Sets**
  - **Properties of collections of objects**
- **Fuzzy Sets and Control Systems**
  - **Effects of uncertainty or imprecision**



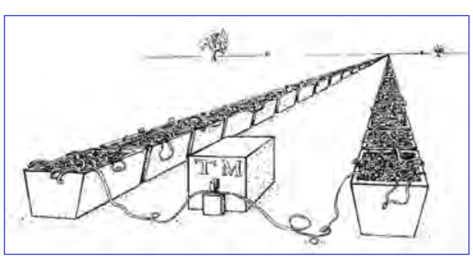
http://www.princeton.edu/~stengel/MAE345.html

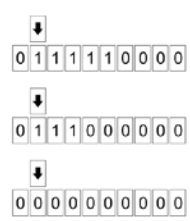
Turing Machine

(Logical Computing Machine)

Alan Turing
IAS, Princeton, *38

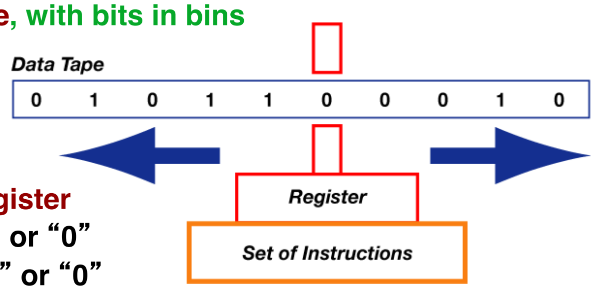
- **Abstract representation of programming for a computing device**
  - **Attempt to give mathematically precise definition to algorithm or mechanical (or effective) procedure**
  - **Hardware description as a machine is figurative**
- **Finite number of internal discrete states of the machine**
  - **States = steps or instructions of a program**
- **Unlimited amount of external input data on a tape**



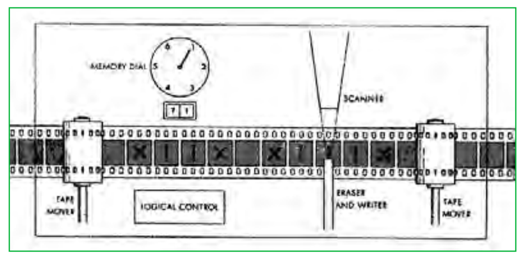


Elements of a Logical Computing Machine (LCM)

- **Data Tape, with bits in bins**

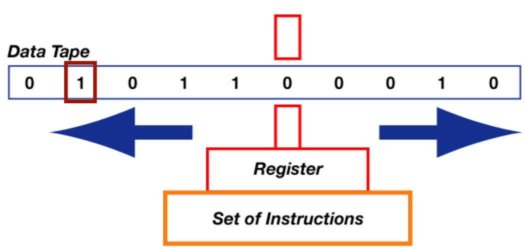


- **Read/Write Register**
  - **Senses "1" or "0"**
  - **Assigns "1" or "0"**
- **Set of Instructions (Program) for R/W Register**
  - **Define internal states, which are identified by state (i.e., program line) number**
- **Control Mechanism**
  - **Move read/write head over tape**
  - **Or move tape through read/write head**



Operation of a Turing Machine

Execution begins in field of "0"s to the left, with device at State #0



- **State #0 instruction moves device to the right until it encounters a "1" in bin (i.e., on the tape)**
- **Action of device depends on <u>current instruction</u> in a <u>Stored Program</u> (i.e., *Set of Instructions*)**
  - **Modifies the bin (or not)**
  - **Identifies next internal state**
  - **Moves one bin to right or left**

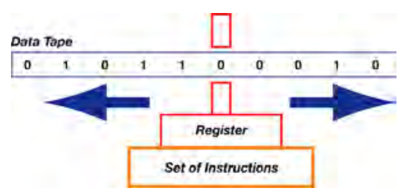
Example of a Turing Machine:
Denary (Base 10) Notation for State (Line Number)

- **Execution begins** in field of "0"s to the left, with device at **State #0**
- **State #0 instruction** moves device **to the right** until it encounters a "1"
- **Action of device depends on stored instructions**
  - **Modifies the bin** (or not)
  - **Identifies next internal state**
  - **Moves one bin** to right or left

[Program] Instruction State #	[Data] Register Contents	Next State	New Bin Contents	Direction of Move
0	0	0	0	R
0	1	13	1	L
1	0	65	1	R
1	1	1	0	R
2	0	0	1	R (Stop)
2	1	66	1	L
3	0	37	1	L
...	...	...	...	...
210	0	3	1	L
...	...	...	...	...
258	1	0	0	R (Stop)
259	0	97	1	R
259	1	0	0	R (Stop)

Penrose, 1989

Same Example:
Binary-coded Program Line Number and Move Direction



- Execution begins in field of "0"s to the left, with device at State #0
- State #0 instruction moves device to the right until it encounters a "1"
- Action of device depends on stored instructions
  - Modifies the bin (or not)
  - Identifies next internal state
  - Moves one bin to right or left

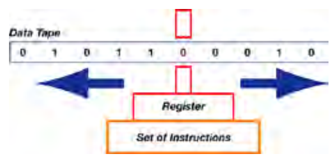
[Program] Instruction State #	[Data] Register Contents	Next State	New Bin Contents	Direction of Move
0	0	0	0	1
0	1	1101	1	0
1	0	1000001	1	1
1	1	1	0	1
10	0	0	1	1 (Stop)
10	1	1000010	1	0
11	0	100101	1	0
...	...	...	...	...
11010010	0	11	1	0

Unary, Binary, and Expanded Binary Coding of Data

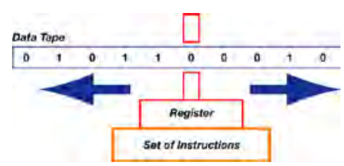
- Turing Machine to add "1" to a unary number (at right)
  - On data tape: 01111

[Program] Instruction State #	[Data] Register Contents	Next State	New Bin Contents	Direction of Move
0	0	0	0	R
0	1	1	1	R
1	0	0	1	R (Stop)
1	1	1	1	R

- Binary coding problems
  - Notation for terminating binary description of a number
  - Definition of space between numbers
  - Recognition of de-limiters (e.g., commas) and logical/ arithmetic operators
- Data could be stored in unary format
  - On data tape: 01110



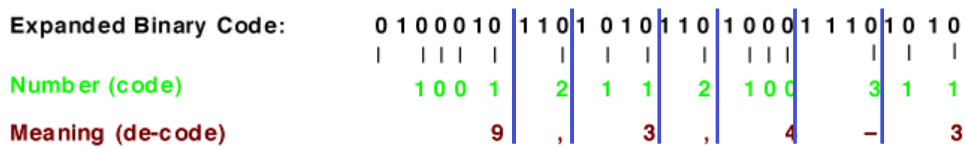
Unary and Expanded Binary Coding of Data



- Solution: Expanded Binary Coding
  - Contraction: Unary expression of small numbers, separated by "0"
  - Encode de-limiters and operators as numbers
  - Terminate numbers with commas (as delimiter)

Number	Meaning	Expanded Binary Notation
0	Binary "0"	0
1	Binary "1"	10
2	Comma	110
3	Minus Sign	1110
4	Plus Sign	11110

Expanded Binary Coding



Number	Meaning	Expanded Binary Notation
0	Binary "0"	0
1	Binary "1"	10
2	Comma	110
3	Minus Sign	1110
4	Plus Sign	11110

Expanded Binary Coding and Turing Machines

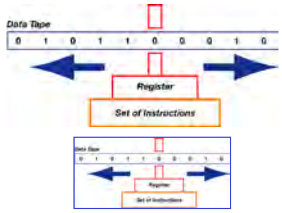
- Expression of arbitrary denary statement numbers
  - Convert from denary to binary
  - Convert from binary to expanded binary
- Define Turing Machine for operations on expanded binary code

Original:	0	1	13	1	L
Binary:	0	1	1101	1	0
De-Limited:	,0,1,1101,1,0,				
Expanded Binary	00011011010110101001011010110110				

Machine code is written in Expanded Binary (or similar) code

Logical Computing Machine

Computer Control Instructions



Further Evolution of the Turing Machine

- Universal Turing Machine *(Computer Program in "Machine Language")*
  – Turing Machine for control of a Turing Machine
  – Instructions on a separate tape or at beginning of data tape
  – 2nd TM with simple code set reads register contents
  – Instruction tape could be modified just like the data tape
    - Instruction branches can be conditioned on prior results
    - Self-modifying instruction set possible

- Church-Turing Thesis *("Hypothesis", "Conjecture")*
  – Turing Machine (*LCM*) defines what we mean by an algorithmic, mechanical, effective, or recursive procedure
  – *LCM* can do anything that could be described as a Rule of Thumb or "purely mechanical"

Calculation and Computing

Calculation

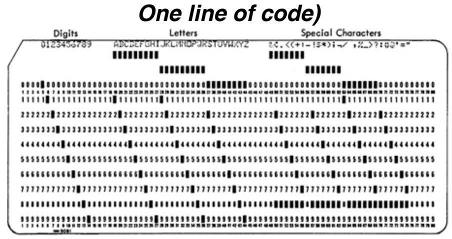
- Thinking, augmented by
  – Abacus
  – Slide rule
  – Math tables
  – Mechanical calculator

Computer

Early Computers - IBM

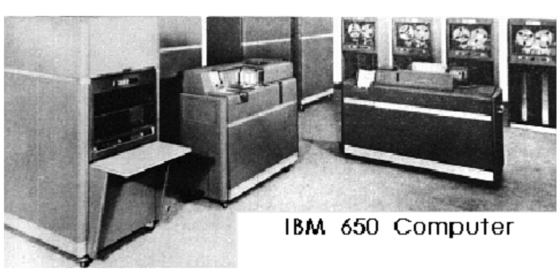
026 Key Punch

Punch Card
One line of code)



IBM 7094

http://en.wikipedia.org/wiki/IBM_7094

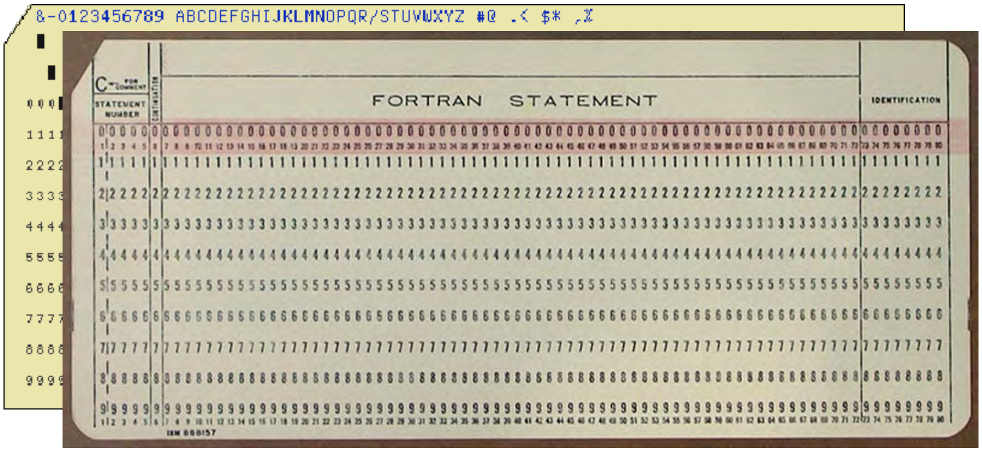


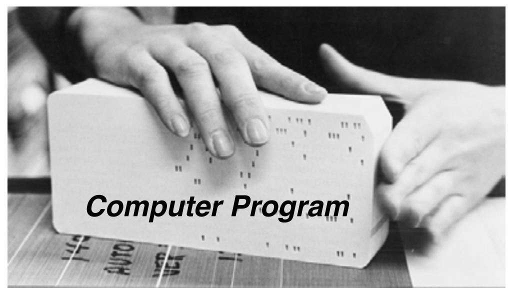
IBM 360/91

http://en.wikipedia.org/wiki/IBM_650

http://en.wikipedia.org/wiki/IBM_360

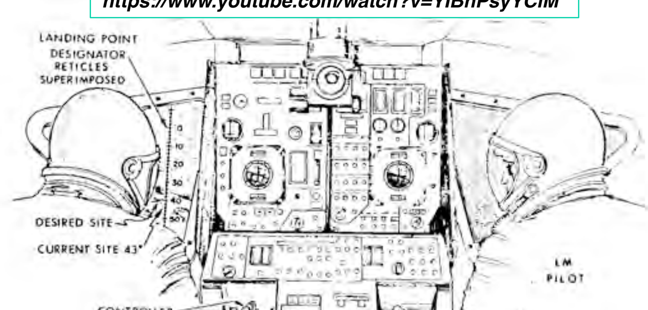
Early Computers – Punched Card





Apollo Guidance Computer

https://www.youtube.com/watch?v=YIBhPsyYCiM



http://klabs.org/history/build_agc/

iPhone 6 vs. Apollo Guidance Computer (1968)
https://www.youtube.com/watch?v=ULGi3UkgW30

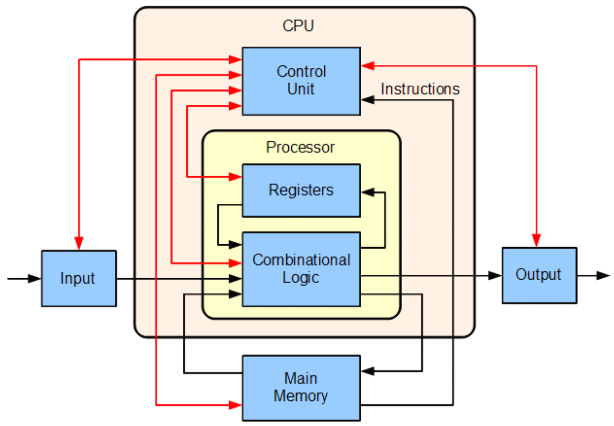
- **Parallel processor**
- **16-bit word length (hexadecimal)**
- **Memory**
  - **36,864 words (fixed)**
  - **2,048 words (variable)**
- **1st operational solid-state computer**
- **Identical computers in CSM and LM**
  - **Different software (with many identical subroutines)**

https://en.wikipedia.org/wiki/Apollo_Guidance_Computer

Hardware Architectures

Central Processing Unit (CPU)

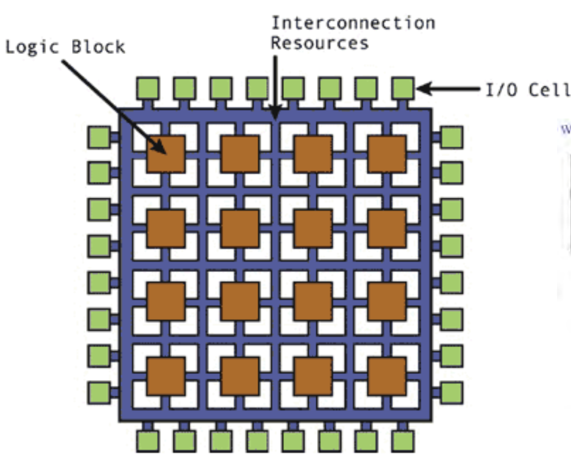
- Arithmetic Logic Unit (ALU)
- Processor registers (~ cache memory)
- Control unit

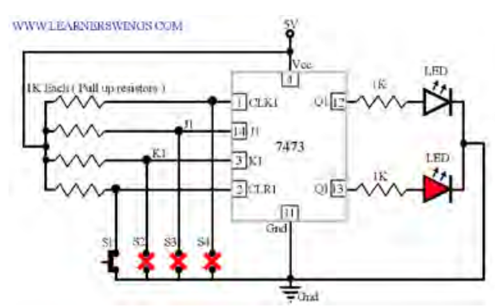


Hardware Architectures

Field-Programmable Gate Array (FPGA)

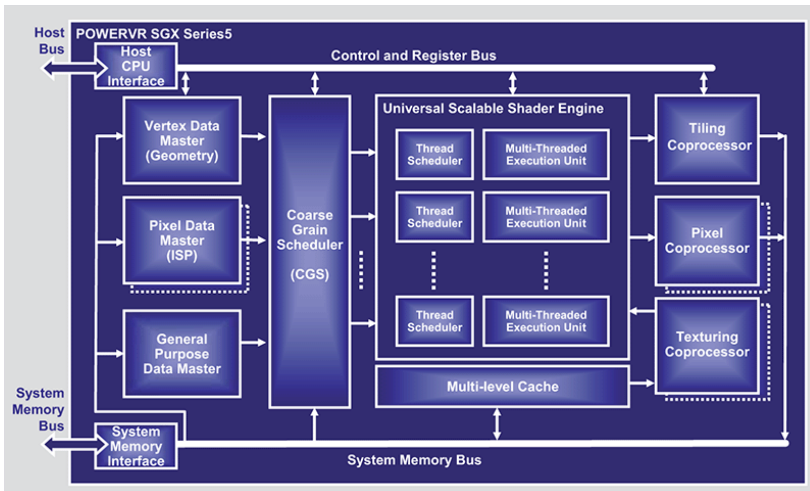
- Application-Specific Integrated Circuit (ASIC)
- Programmable logic blocks/“gates”
  - Look-up tables, flip-flops (bistable latches), and routing matrix
- Reconfigurable connections
- Data buses, timers, analog components





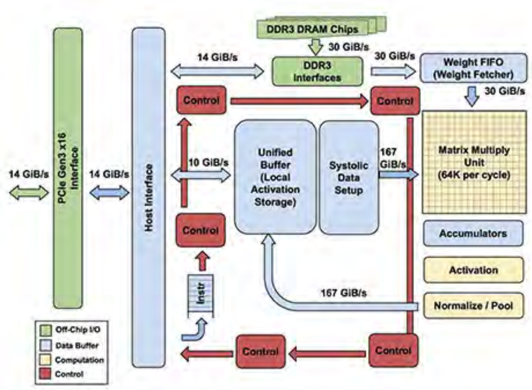
Hardware Architectures
Graphics Processing Unit (GPU)

- **Highly parallel structure for rendering images**
- **Transformation, clipping, texture mapping, shading, and lighting**
- **Specialization to vector-matrix operations**



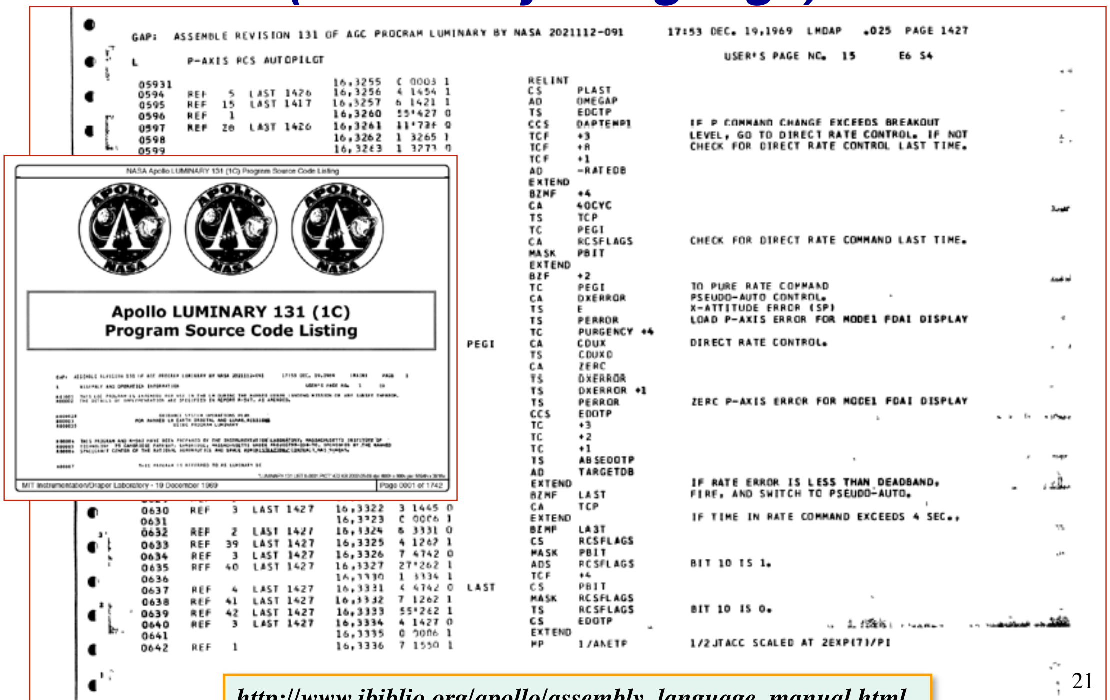
Hardware Architectures
Tensor Processing Unit (TPU)

- **Application-Specific Integrated Circuit (ASIC) for machine learning**
- ***Google TensorFlow* symbolic math CISC s/w library**
- **High-volume, reduced-precision logic (e.g., 256 x 256 8-bit matrix multiply, on-chip memory and accumulators)**
- **Arrays of arrays of TPU chips ~ 11.5 PFLOPS performance**



https://en.wikipedia.org/wiki/Tensor_processing_unit

A Little AGC Digital Autopilot Code
(Assembly Language)



```
GAP: ASSEMBLE REVISION 131 OF AGC PROGRAM LUMINARY BY NASA 2021112-091     17:53 DEC. 19,1969  LMDAP  .025  PAGE 1427
P-AXIS RCS AUTOPILOT                                                      USER'S PAGE NO. 15      E6 S4

        RELINT
        CS      PLAST
        AD      OMEGAP
        TS      EDOTP
        CCS     DAPTEMP1      IF P COMMAND CHANGE EXCEEDS BREAKOUT
        TCF     +3            LEVEL, GO TO DIRECT RATE CONTROL. IF NOT
        TCF     +8            CHECK FOR DIRECT RATE CONTROL LAST TIME.
        TCF     +1
        AD      -RATEDB
        EXTEND
        BZMF    +4
        CA      40CYC
        TS      TCP
        TC      PEGI
        CA      RCSFLAGS      CHECK FOR DIRECT RATE COMMAND LAST TIME.
        MASK    PBIT
        EXTEND
        BZF     +2
        TC      PEGI          TO PURE RATE COMMAND
        CA      DXERROR       PSEUDO-AUTO CONTROL.
        TS      E             X-ATTITUDE ERROR (SP)
        TS      PERROR        LOAD P-AXIS ERROR FOR MODE1 FDAI DISPLAY
        TC      PURGENCY +4
PEGI    CA      CDUX          DIRECT RATE CONTROL.
        TS      CDUXD
        CA      ZERO
        TS      DXERROR
        TS      DXERROR +1
        TS      PERROR        ZERO P-AXIS ERROR FOR MODE1 FDAI DISPLAY
        CCS     EDOTP
        TC      +3
        TC      +2
        TC      +1
        TS      ABSEDOTP
        AD      TARGETDB
        EXTEND                IF RATE ERROR IS LESS THAN DEADBAND,
        BZMF    LAST          FIRE, AND SWITCH TO PSEUDO-AUTO.
        CA      TCP
        EXTEND                IF TIME IN RATE COMMAND EXCEEDS 4 SEC.,
        BZMF    LAST
        CS      RCSFLAGS
        MASK    PBIT
        ADS     RCSFLAGS      BIT 10 IS 1.
        TCF     +4
LAST    CS      PBIT
        MASK    RCSFLAGS
        TS      RCSFLAGS      BIT 10 IS 0.
        CS      EDOTP
        EXTEND
        MP      1/ANETP       1/2JTACC SCALED AT 2EXP(T)/PI
```


http://www.ibiblio.org/apollo/assembly_language_manual.html

Evolution of Programming

- **History of programming languages**
  - https://en.wikipedia.org/wiki/History_of_programming_languages
- **The song, "99 Bottles of Beer on the Wall", programmed in 1,500 computer languages**
  - http://www.99-bottles-of-beer.net/
  - **In BASIC:**

```
10 REM BASIC Version of 99 Bottles of beer
20 FOR X=100 TO 1 STEP -1
30 PRINT X;"Bottle(s) of beer on the wall,";X;"bottle(s) of beer"
40 PRINT "Take one down and pass it around,"
50 PRINT X-1;"bottle(s) of beer on the wall"
60 NEXT
```

Programming Language Classes

- Expert systems can be programmed in almost any language
- Language is the interface between the programmer and the computer
- Higher-order
  - Lower-order
    - Interpreter
    - Compiler
      - Assembly language
        - » Machine code
- Critical differences
  - Instruction set
  - Execution speed
  - Memory use

- Procedural (e.g., FORTRAN, LISP, MATLAB, Python)
  - Imperative
  - Functional
- Non-Procedural, Query-based Languages (e.g., PROLOG)
  - Declarative
  - Non-Declarative

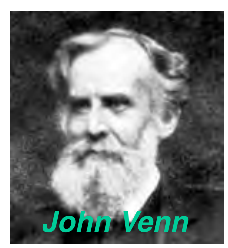
Ultimately, it is all machine code ("0"s and "1"s)

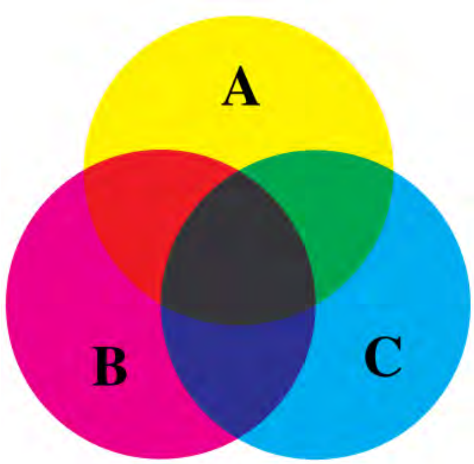
Crisp Sets

Naive (or Intuitive) Set Theory (1870s)

- **Deals with the properties of well-defined collections of objects**
- **Universal set = Universe of discourse = *U***
  - **Contains all elements of possible concern in a particular context**
- ***A* = a particular set in *U***
  - **defined in a list**
  - **by a rule, or**
  - **by a membership function describing elements (or members) of the set**

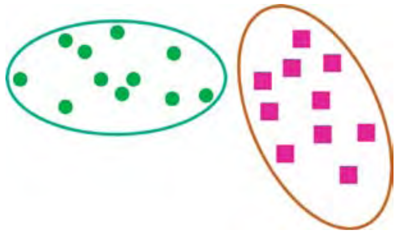
Venn Diagrams (1881)





- **A: All mammals**
- **B: All aquatic animals**
- **C: All gray, hairless objects**

- **A & B: Whales, dolphin, seals, …**
- **B & C: Fish, clams, whales, dolphins, …**
- **A, B, &C: Whales, dolphins, …**



Three Criteria for Membership in a Set

- **List** $A = \{x, y, z, ...\}$ Ordering of elements is not important

  $A = (x, y, z, ...)$ Ordering of elements is important

- **Rule** $A = \left\{ x \in U \middle| x \text{ meets some conditions} \right\}$ e.g., $A = \{x \in U \mid g(x) \le 0\}$

  $\mu_A(x)$

- **Membership function, e.g., categorical description**

$$\mu_A(x) = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{if } x \notin A \end{cases}$$

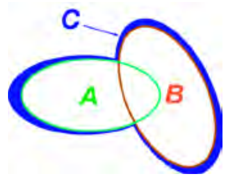
Membership in a Set

- ***A*** **= a particular set in *U***
  - defined in a list or rule, or
  - by a membership function describing elements (or members) of the set
- **Universal set = guests at a party**
- **Particular sets**
  - Current graduate students
  - Alumni
  - Spouses
  - Friends of students
  - Children
  - Same family
  - Visitors
  - Pilots
  - Teachers
  - Managers
  - Military officers
  - Women and men
  - US citizens or foreign nationals

Operations on Sets

- **Union of sets**

$$C = A \cup B$$



- **One-to-one correspondence**

$$A = B$$

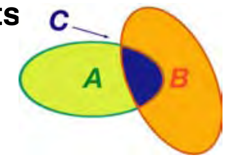
For example, as when

A = (x, y, z)

B = (4, 3, 9); then x = 4, y = 3, and z = 9

- **Intersection of sets**

$$C = A \cap B$$

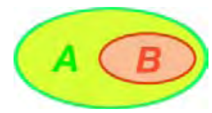


- **Proper Subset**

$$B \subset A$$

$$B \subseteq A, \text{ and}$$

$$B \neq A$$



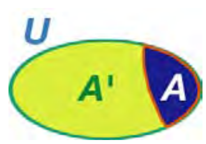
... in which case, A "is greater than" B

$$A \supset B$$

Properties of Sets

- **Complement**

$$A' = U - A$$



- **Reflexive property**
  - Relationships that bear same effect on own set as on other sets

$$e.g., \quad A = A, \quad A \geq A, \quad A \leq A$$

- **Transitive property**
  - Two sets bear same relationship to a third set

$$e.g., \text{if } A > B \text{ and } B > C, \text{ then } A > C$$

- **Empty (null) set**

$$\varnothing = A - A \text{ or } = U - U$$

- **Symmetry property**
  - Relationship of first to second set is the same as second to first set

$$e.g., A = B, B = A$$

- **Equivalence**
  - Reflexivity + Symmetry + Transitivity

$$A \sim B \text{ or } A \equiv B$$

What If Sets Have Uncertain or Contradictory Membership?

- **Example: *U* = All the cars in Berkeley**

$$A = \{x \in U \mid x \text{ has 4 cylinders}\}$$
$$B = \{x \in U \mid x \text{ has 6 cylinders}\}$$

Set	*Crisp Logic*
A	4-cylinder engine
B	6-cylinder engine
C	8-cylinder engine
D	Others

Set	*Fuzzy Logic*
E	US cars
F	Foreign cars

- **What makes a car "US" or "Foreign"?**
  - US cars may contain foreign parts
  - Foreign cars may contain US parts

List?
Rule?
Membership function?

How should we define Sets *E* and *F*?

Fuzzy Sets

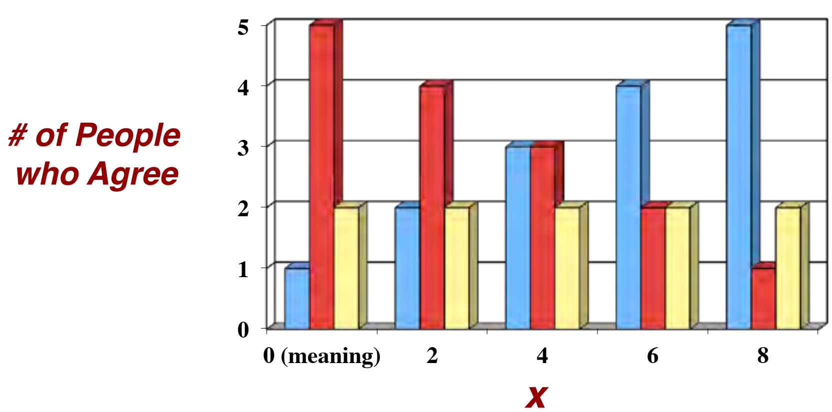
Hard and Soft Thinking

- **Problem-solving approaches**
  - Logical / Metaphorical
  - Reasonable / Dream-like
  - Serious / Humorous
  - Definite / Ambiguous
  - Consistent / Paradoxical
  - Laborious / Playful
  - Exact / Approximate
  - Real / Fantastic
  - Focused / Diffuse
  - Analytical / Illogical
  - Specific / General
  - Mature / Immature
- **Crisp / Fuzzy**

A Notional Fuzzy Experiment

- What does each term mean?

A few ≤ *x*
A lot ≥ *x*
Several ≈ *x*

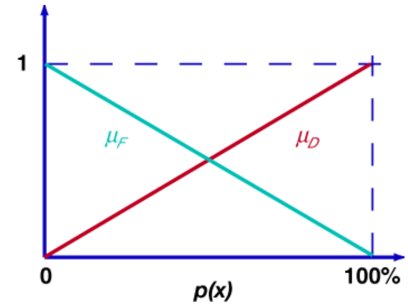


Normalize results so that the maximum is 1
Normalized plots are fuzzy membership functions

Fuzzy Sets

- **Fuzzy membership function,** $\mu_A(x) = 1$
  - **takes any value in [0, 1]**
- **Fuzzy set, *E***

$$E = \{x \in U | [x, \mu_E(x)]\}$$

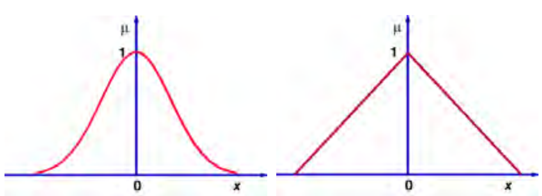


- ***U* = All the cars in Berkeley**
  - ***p(x)* = percentage of domestic parts =** $\mu_D(x) = 1$
  - **[1 – *p(x)*] = percentage of foreign parts =** $\mu_F(x) = 1$

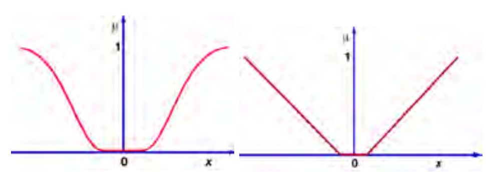
> - **Membership functions express a subjective utility that may be rigorous (e.g., based on probability) or not**
> - **"If 46% of a car's parts are domestic, it probably is a foreign car"**

Fuzzy Membership Functions

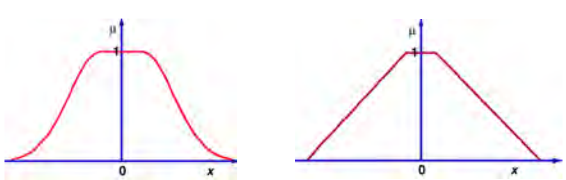
"Close to zero"



"Probably not *y*"

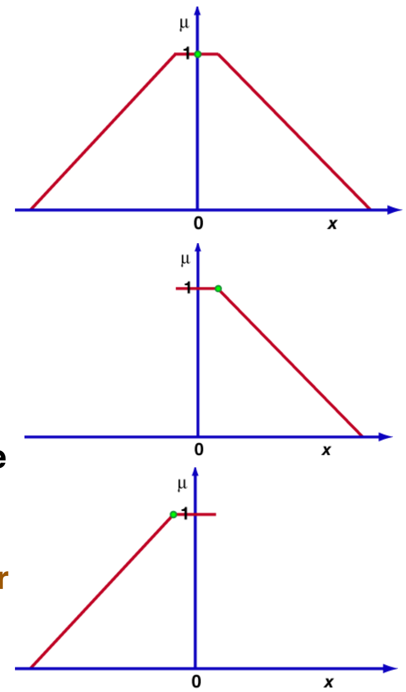


"About *y*"



"Young/old"

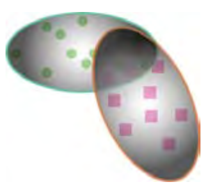
Fuzzy Set Definitions



- **Support** of a fuzzy set

$$\text{supp}(A) = \{x \in U \mid \mu_A(x) > 0\}$$

- **Center** of a fuzzy set
  - **Two-sided**: center = mean value
  - **Left-sided**: center = max (*x*) for which $\mu(x) = 1$
  - **Right-sided**: center = min (*x*) for which $\mu(x) = 1$



Fuzzy Logic Operations

Union of sets

$$A \cup B$$
$$\text{if and only if } \mu_{A \cup B}(x) = \max\left[\mu_A(x), \mu_B(x)\right] \; \forall \; x \in U$$

Intersection of sets

$$A \cap B$$
$$\text{if and only if } \mu_{A \cap B}(x) = \min\left[\mu_A(x), \mu_B(x)\right] \; \forall \; x \in U$$

A contains *B*

$$A \supset B$$
$$\text{if and only if } \mu_A(x) \geq \mu_B(x) \; \forall \; x \in U$$

More Fuzzy Logic Operations

Equivalence of sets

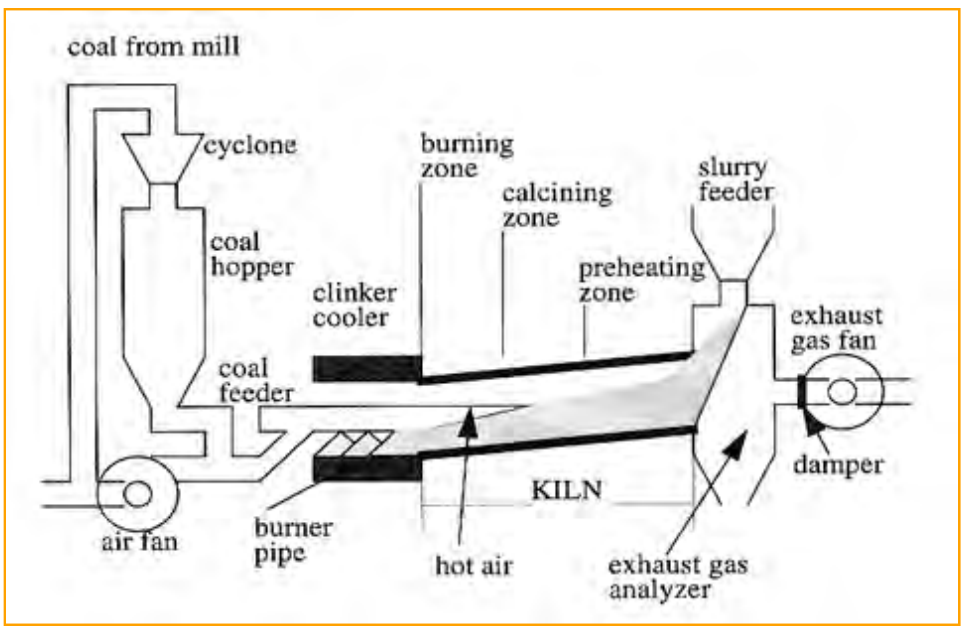
$$A \sim B$$
$$\text{if and only if } \mu_A(x) = \mu_B(x) \; \forall \; x \in U$$

Complement of sets

$$A' = U - A$$
$$\text{if and only if } \mu_{A'}(x) = 1 - \mu_A(x) \; \forall \; x \in U$$

Fuzzy logic is a generalization of crisp logic based on the definition of the membership function

Example: Cement Kiln Control

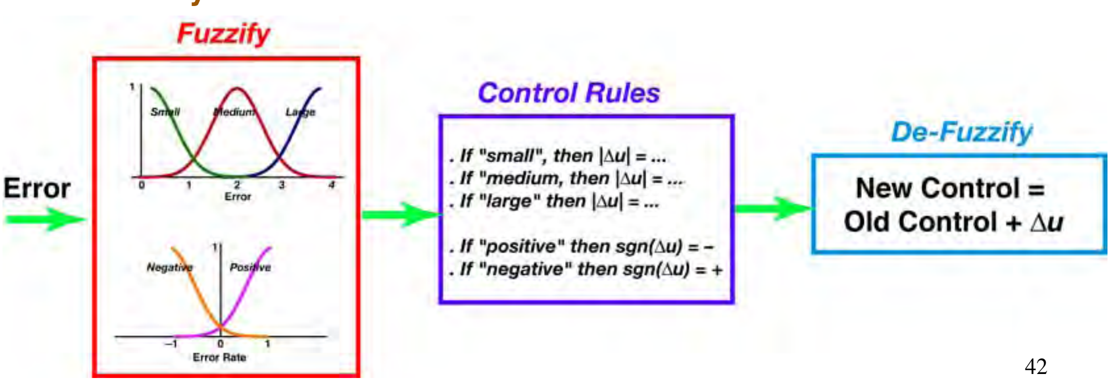


Cement Kiln Operator's Manual

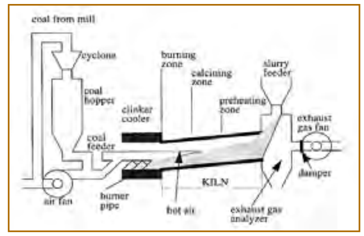
Case	Condition	Action to be taken	Reason
10	BZ ok OX low BE low	a. Increase air fan speed b. Increase fuel rate	To raise back-end temperature and increase oxygen for action 'b' To maintain burning zone temperature
11	BZ ok OX low BE ok	a. Decrease fuel rate speed slightly	To raise oxygen
12	BZ ok OX low BE high	a. Reduce fuel rate b. Reduce air fan speed	To increase oxygen for action 'b' To lower back-end temperature and maintain burning zone temperature
13	BZ ok OX ok BE ok	a. Increase air fan speed b. Increase fuel rate	To raise back-end temperature To maintain burning zone temperature
14	BZ ok OX ok BE ok	None. However, do not get overconfident and keep all conditions under observation	
15	BZ ok OX ok BE high	When oxygen is in upper part of range a. Reduce air fan speed When oxygen is in lower part of range b. Reduce fuel rate c. Reduce air fan speed	To reduce back-end temperature To raise oxygen for action 'c' To lower back-end temperature and maintain burning zone temperatur
16	BZ ok OX high BE low	a. Increase air fan speed b. Increase fuel rate	To raise back-end temperature To maintain burning zone temperature and reduce oxygen
17	BZ ok OX high BE ok	a. Reduce air fan speed slightly	To lower oxygen

Linguistic (Mamdani) Fuzzy Control Systems (after Schramm)

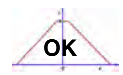
- ***Antecedent*** **and** ***consequent*** **are both fuzzy propositions**
  - e.g., "If *error* is *small* and *error rate* is *negative*, then *control command* is *small*"
  - What are "small", "medium", and "large"?
- **Must "fuzzify" physical error/rate, apply fuzzy rules, and "de-fuzzify" control command**

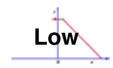


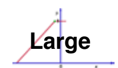
Mamdani Fuzzy Controller for Cement Kiln



- Linguistic Controller
  - Antecedents
    - BZ: Temperature in burning zone
    - OX: Oxygen in exhaust gas
    - BE: Temperature at end of kiln
  - Consequents
    - CR: Coal feed rate
    - DP: Exhaust damper position
  - 27 fuzzy rules, e.g.,
    - If **BZ** is **OK** and **OX** is **low** and **BE** is **low**, then set **CR** to **large**, and **DP** to **large**







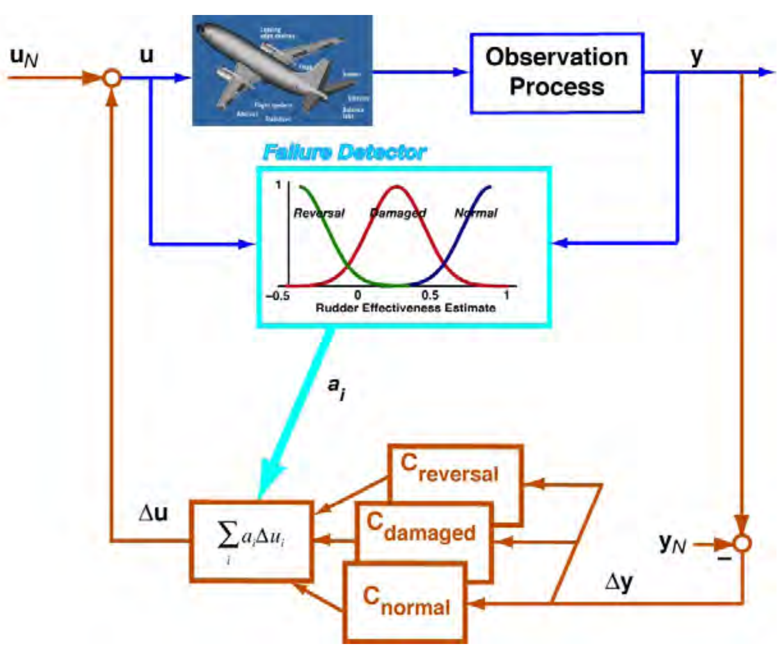
Controller is apparently symbolic, but symbols must have values for computation, i.e., <u>Fuzzy Membership Functions</u>

Probable Cause

- The *National Transportation Safety Board* determines that the probable cause of the *USAir flight 427* accident was
  - a loss of control of the airplane resulting from the movement of the rudder surface to its blowdown limit.
- The rudder surface most likely deflected in a direction opposite to that commanded by the pilots as a result of
  - a jam of the main rudder PCU servo valve secondary slide to the servo valve housing offset from its neutral position and
  - overtravel of the primary slide.

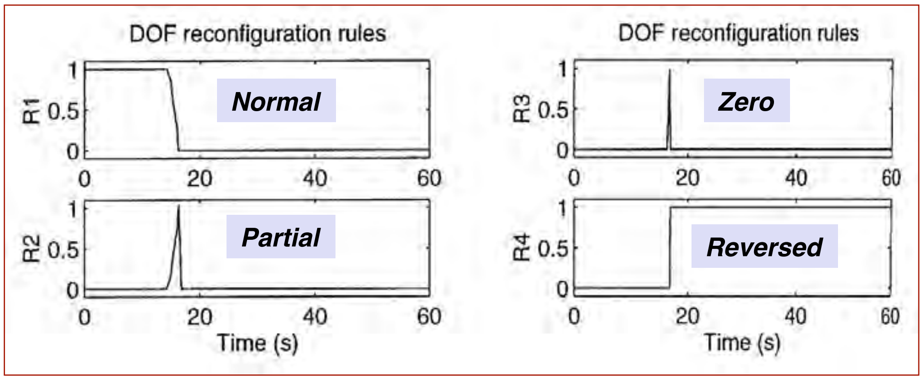
Gain-Scheduling (Takagi-Sugeno) Fuzzy Control Systems

(Schramm, Gopisetty, and Stengel, 1998)



Failure Detection for Simulated Rudder Failure

- Rudder reversal occurs at $t = 10$ s
- Heading angle change commanded at $t = 20$ s

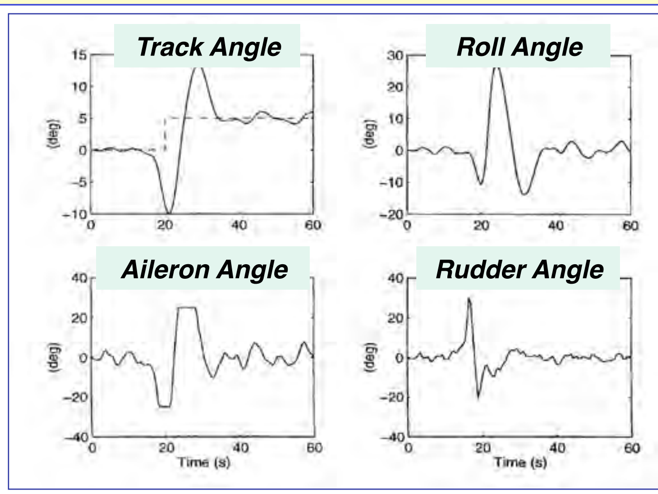


Schramm, 1998

Simulated Reconfiguration

- Failure detection logic detects nothing until rudder effect is expected
- Once detected, control signal is reversed

Schramm, 1998



Fuzzy Logic ≠ Fuzzy Thinking

- **Quantitative approach to reasoning under uncertainty**
- **"Possibility theory" vs. Probability theory (*Lotfi Zadeh, 1978*)**
- **Relationship to other uncertainty belief systems of artificial intelligence, e.g.,**
  - **Bayesian belief network**
  - **Dempster-Shafer theory**
  - **Transferable belief model**
  - **Certainty factors**
- **Propositions are true or false only within the context of a paradigm**

Next Time: Probability and Statistics

Supplemental Material

Turing Machine for Euclid's Algorithm

Penrose, R., *The Emperor's New Mind*, 1989, p. 41

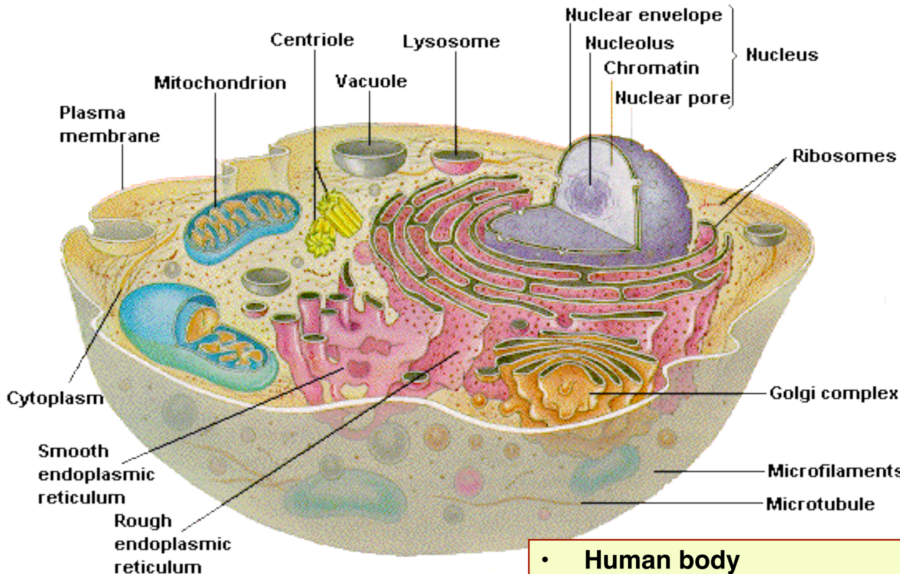
Since we wish to be able to include numerical data as part of our input, we shall want to have a way of describing ordinary numbers (by which I here mean the natural numbers 0, 1, 2, 3, 4, . . .) as part of the input. One way to do this might be simply to use a string of *n* 1s to represent the number *n* (although this could give us a difficulty with the natural number zero):

1 → 1, 2 → 11, 3 → 111, 4 → 1111, 5 → 11111, etc.

This primitive numbering system is referred to (rather illogically) as the *unary* system. Then the symbol '0' could be used as a space to separate different numbers from one another. It is important that we have such a means of separating numbers from one another since many algorithms act on *sets* of numbers rather than on just single numbers. For example, for Euclid's algorithm, our device would need to act on the *pair* of numbers A and B. Turing machines can be written down, without great difficulty, which effect this algorithm. As an exercise, some dedicated readers might perhaps care to verify that the following explicit description of a Turing machine (which I shall call **EUC**) does indeed effect Euclid's algorithm when applied to a pair of unary numbers separated by a 0:

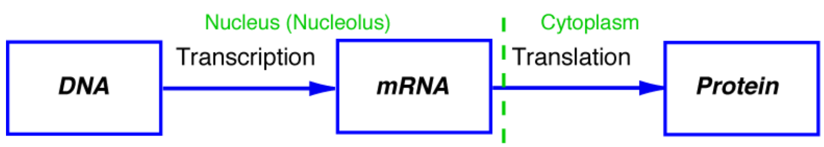
00→00R, 01→11L, 10→101R, 11→11L, 100→10100R,
101→110R, 110→1000R, 111→111R, 1000→1000R, 1001→1010R,
1010→1110L, 1011→1101L, 1100→1100L, 1101→11L, 1110→1110L,
1111→10001L, 10000→10010L, 10001→10001L, 10010→100R,
10011→11L, 10100→00STOP, 10101→10101R.

Turing Machines in Biological Cells



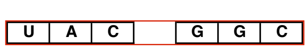
- **Human body**
  - **~100 trillion cells**
  - **typical diameter: ~10 μm**

The Central Dogma:
Core Process of Protein Production



- **4 Nucleotides (*A, T, C, G*)**
- **4 Nucleotides (*A, U, C, G*)**
- **20 Amino Acids (A, C, D, E, ...)**

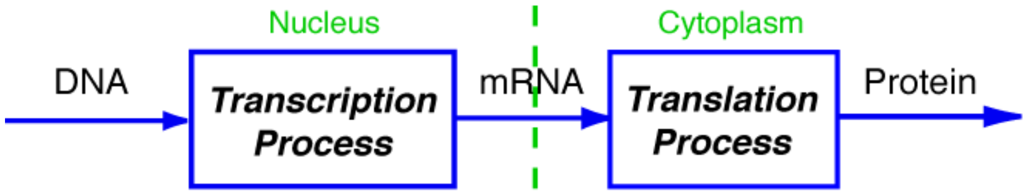
- **A gene is a sequence of nucleotides on a chromosomal strand of DNA**
- **Single-stranded messenger RNA (mRNA) <u>expresses</u> information from DNA to form a protein, a sequential molecule of amino acids**
- **Information coded in *nt* triplets (codons), with open reading frames defined by Start and Stop codons**



The Genetic Code

First Position	Second Position: *U*	*C*	*A*	*G*	Third Position
U	F	S	Y	C	*U*
	F	S	Y	C	*C*
	L	S	Stop	Stop	*A*
	L	S	Stop	W	*G*
C	L	P	H	R	*U*
	L	P	H	R	*C*
	L	P	Q	R	*A*
	L	P	Q	R	*G*
A	I	T	N	S	*U*
	I	T	N	S	*C*
	I	T	K	R	*A*
	M (start)	T	K	R	*G*
G	V	A	D	G	*U*
	V	A	D	G	*C*
	V	A	E	G	*A*
	V	A	E	G	*G*

System View of The Core Process



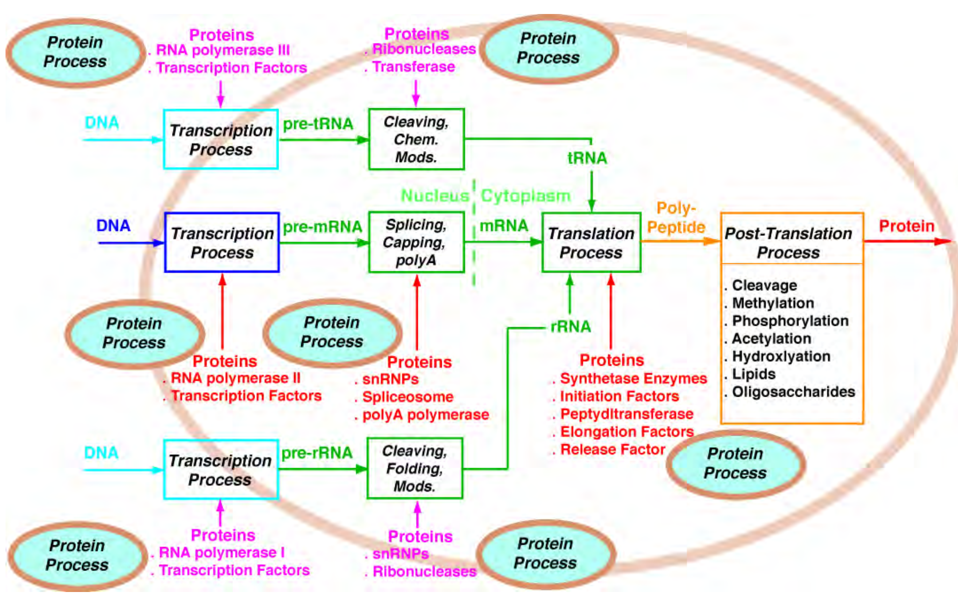
- **Polymerase unzips dual-stranded DNA**
- **Exons of DNA code mRNA sequence with help of transcription factors**
- **Introns and extragenic DNA are ignored (deleted) in RNA**

- **mRNA enters ribosome, which is composed of ribosomal RNA (rRNA) and proteins**
- **Transfer RNA (tRNA) transports amino acids to ribosome**
- **Amino acids assembled into protein according to RNA sequence (3 *nt* per codon)**

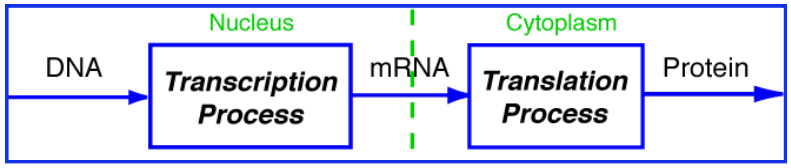
~20,000 human genes

~3 billion base pairs/human gamete

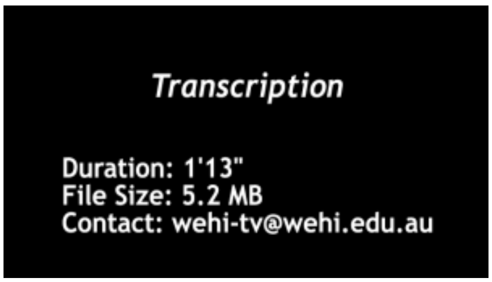
Block Diagram of the Protein Process



Protein Production is Dynamic

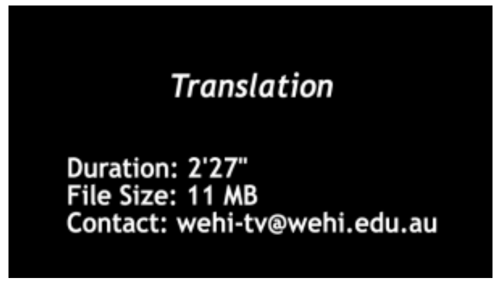


Transcription



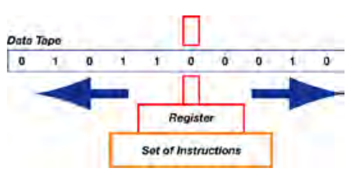
~50 bases/sec

Translation



~10 amino acids/sec

Polymerases and Ribosomes as Turing Machines

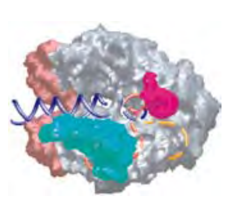


- Coding of amino acids from mRNA
- *AUCG* provides Base 4 coding

The Genetic Code

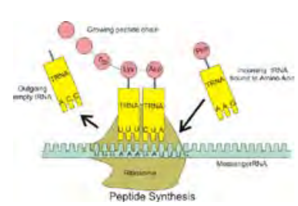
First Position	Second Position: U	C	A	G	Third Position
U	F	S	Y	C	U
	F	S	Y	C	C
	L	S	Stop	Stop	A
	L	S	Stop	W	G
C	L	P	H	R	U
	L	P	H	R	C
	L	P	Q	R	A
	L	P	Q	R	G
A	I	T	N	S	U
	I	T	N	S	C
	I	T	K	R	A
	M (start)	T	K	R	G
G	V	A	D	G	U
	V	A	D	G	C
	V	A	E	G	A
	V	A	E	G	G

RNA Polymerase II*



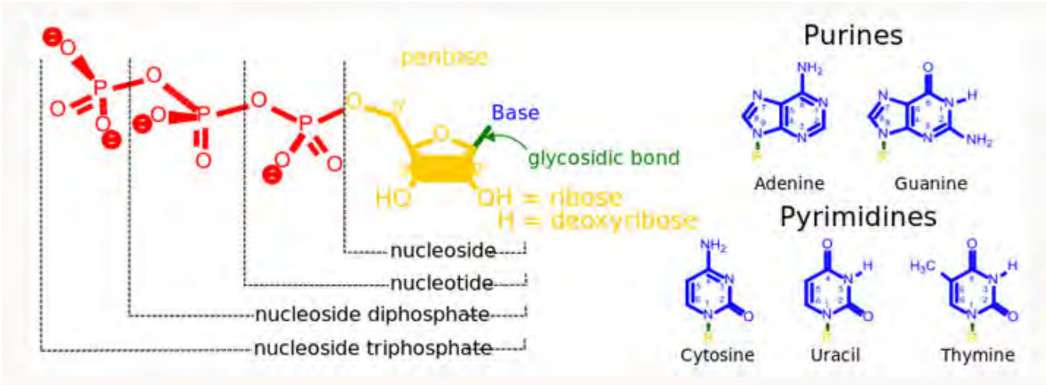
** = Enzyme = Catalytic protein*

Ribosome**

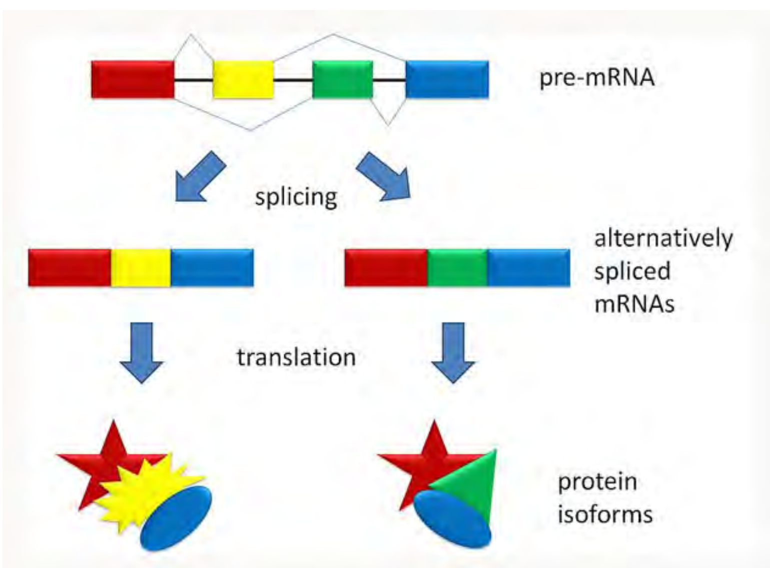


*** = Complex of rRNA and proteins*

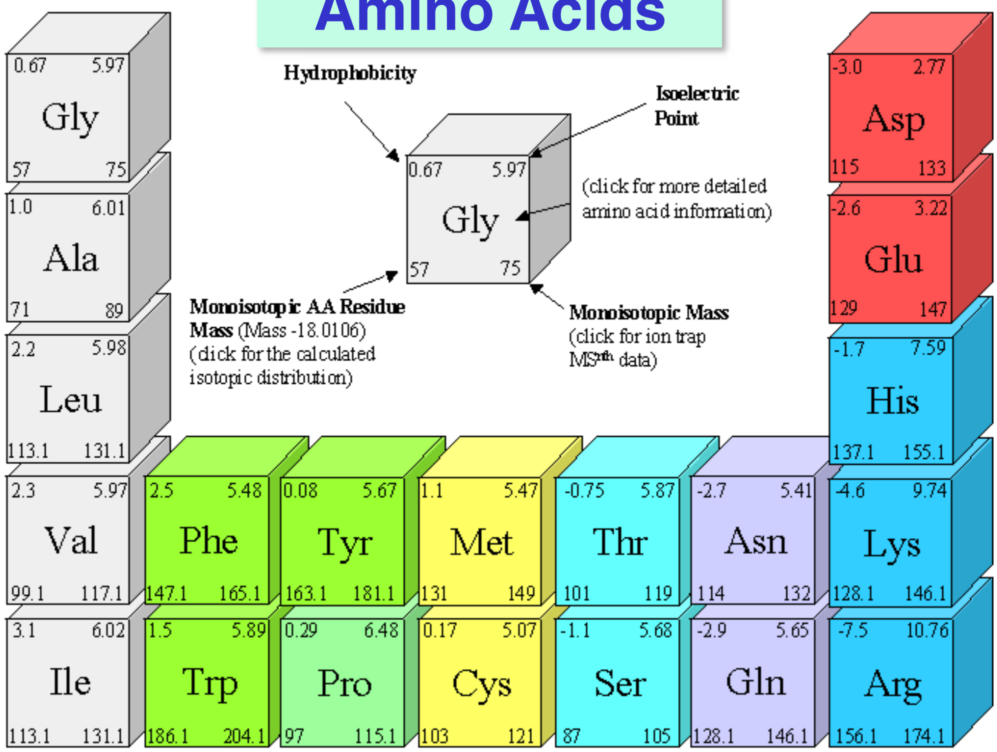
DNA/RNA Molecules



Alternative Splicing



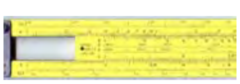
Amino Acids



How Do Slide Rules and Calculators Work?

- **Abacus**
  - **Unary number system**
  - *http://gwydir.demon.co.uk/jo/numbers/machine/abacus.htm*

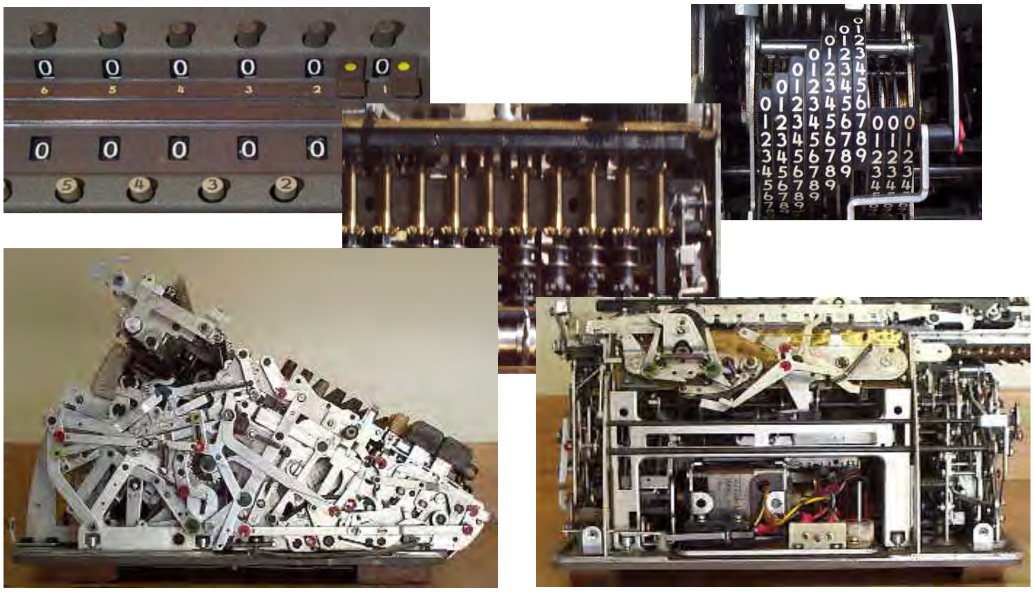
- **Mathematical Tables**
  - **Manual calculations**
  - *http://en.wikipedia.org/wiki/Mathematical_tables*



- **Slide rule**
  - **Logarithmic scales**
  - *https://www.youtube.com/watch?v=uUzSStVnAHk*

- **Mechanical Calculator**
  - **Add, subtract, and shift**
  - *http://en.wikipedia.org/wiki/Mechanical_calculator*
  - *https://www.youtube.com/watch?v=7S0BETniokI*

Friden STW-10 Calculator Mechanism



Bi-Quinary Control Panel Lights for the IBM 650

Computation with 10-digit words

Early Computers - DEC

PDP-1

SpaceWar

https://en.wikipedia.org/wiki/PDP-1

PDP-6
Time-Sharing

https://en.wikipedia.org/wiki/PDP-6

Computer Transistor Counts