IB Computer Science Notes

May 2021

Contents

Note

These notes were compiled using content from the textbook "Core Computer Science for the IB Diploma Program" by Kostas Dimitriou & Markos Hatzitaskos. I've tried to follow the IB CS Guide as closely as possible, and sections and subsections are divided according to the guide, as closely as possible (subsubsections, however, do not follow the guide).

For topic 4, despite being the largest SL topic, the section has very little actual content worth putting into notes. It's mostly just practicing algorithms, pseudocode and so on, which is why I'd recommend putting more time into practicing, either with past papers or other resources, rather than using notes. Do remember to take a look at the IB pseudocode guides as well. There are two documents, one regarding pseudocode notation conventions, will be available to candidates during the examination, and another regarding standard data structures that may appear on external examinations. This document will not be available during external examinations.

1 System Fundamentals

1.1 Systems in organization

1.1.1 Planning and system installation

Stages through which the development of a new system passes through.

It might generally look like:
Analysis → Design → Implementation → Operation → Maintenance

Context for a new system
A new system can be created to replace one that is inefficient, redundant or outdated. The first stage is planning, which will often require a feasibility study for the new system:

- Technical feasibility: Is the technology sufficient to implement the system?
- Economic feasibility: Is the system cost effective
- Legal feasibility: Does the system adhere to relevant laws and regulations
- Operational feasiblity: Are existing practices and procedures sufficient to support the system?
- Schedule feasibility: How long will the new system take?

Change management
Change management is the process of shifting individuals, teams, departments, and organizations from the present to a desired state.

Compatibility issues
Legacy system: Old technology which is now out-dated, but possibly still in use by a system due to various reasons such as being difficult to replace due to complexity or using data which is incompatible with new data formats etc.

Business merger: the combining of two or more business entities. Has benefits such as reduced costs but presents challenges such as ensuring both systems are compatible.

SaaS
Software-as-a-Service (SaaS): A software delivery method where the software is hosted remotely, and users pay to access it on a subscription basis.

System installation processes
Implementation methods for moving from old to new system:

- Parallel: both systems work in parallel for a short time, until the transition to the new one is finished.
- Direct: The old one is halted, and the new one is immediately put into place.
- Pilot: the new one is implemented in a smaller part of the organization, and slowly extended over time.
- Phased: parts of the old system are replaced by parts of the new system one by one gradually.

Data migration
Data migration: transfer of data between formats, storage types or computer systems.

Issues with data migration include compatibility issues and data loss.

Testing
Functional testing: the specific functions of a program are tested.

Data testing: normal, abnormal and extreme data is provided to the system.

Alpha testing: testing before the software is made public.

Beta testing: tested by initial users and feedback is obtained.

Dry-run testing: conducted using pen-and-paper by the programmers.

Integration testing: Entire system is tested

User acceptance testing: system is tested to see if it satisfied the customers' needs.

Debugging: process of finding and correcting bugs in programs.

Validation: evaluating whether data input follows appropriate specifications.

Verification: ensuring data input is the same as the original source data.

1.1.2 User focus

Documentation
Internal documentation: documentations embedded within the source code itself (comments, types, etc.)

External documentation: usually written as a separate document from the program.

Providing documentation

- Manuals
- Email support
- Embedded assistance or integrated user assistance
- Frequently Asked Questions
- Live chat sessions
- Online web portals
- Remote desktop connections

User training

- Self-instruction
- Formal classes
- Online training

1.1.3 System backup

Data loss
Causes:

- Accidental deletion
- Administrative errors
- Poor data storage system
- Building fires
- Natural disasters
- Viruses/malware
- Data corruption
- Firmware corruption

Prevention:

- Regular backups
- Firewall/Anti-virus installation
- Offsite storage
- Removed hard copies
- Failover system: A system that, upon error or failure, is able to switch over to a spare/backup system and continue functioning.

1.1.4 Software deployment

Software releases and updates

- Patches: updates to fix bugs and issues
- Upgrades: Novel functionalities or features as well as bug fixes.
- Updates: Minor upgrade
- Release: Finalised working version of software.

1.2 System design basics

1.2.1 Components of a computer system

- Hardware: physical elements.
- Software: instructions that can be understood and executed by the CPU.
- Peripheral device: Auxiliary device that can communicate with a computer.
- Computer network: Interconnected computer systems.
- Human resources: people.

Roles of a computer

- Client: recieves data from a server
- Server: sends data to a client
- Dumb terminal: consists of keyboard, monitor, network card.

- Thin client: low performance terminal.
- Email server: manages flow of email in and out of network.
- Router: accepts incoming data packets, reads their destination address, and distributes them across the network according to a routing policy.
- Domain Name System (DNS) server: attributes names to network addresses and resolves names by assigning them to the appropriate network entity.
- Firewall: hardware or software network infrastructure that controls and limits data flow access among entities. Used to offer protection.
- Client-server: architecture where client and server are connected.

Social and ethical issues

- Reliability
- Integrity
- Inconsistency
- Security
- Privacy
- Anonymity
- Intellectual property
- Digital divide and equality of access
- Surveillance
- Global and cultural diversity

IT policies: enforceable procedures to ensure appropriate and secure use of a computer system.

Standards and protocols: technical rules hardware and software should follow.

People and machines: Internet addiction is a social impact. Use of AI in military or law-enforcement situations.

Digital citizenship: appropriate behavior in a digital world.

1.2.2 System design and analysis

Stakeholders: individuals, teams or organizations with an interest in the project/system/etc.

Obtaining requirements from stakeholders

- Structured/Unstructured interviews
- Closed/Open questionnaires
- Direct observation.

Gathering information to arrive at a workable solution: examining current systems, examining competing products.

Requirements Specification Document: specifies requirements.

Illustrating system requirements

- Flowcharts
- Data-flow diagrams
- Structure charts

Prototyping: working or non-working preliminary version of product.

Iteration: repeating the process multiple times until arriving at the desired point.

1.2.3 Human interaction with the system

- Accessiblity: potential of a product to meet the needs of as many individuals as possible.
- Usability: potential of a product to accomplish user goals.
- Ergonomics: design of safe and comfortable products.

Components of usability

- Simplicity
- Effectiveness

- Efficiency
- Error
- Learnability
- Memorability
- Readability
- Satisfaction

2 Computer Organization

2.1 Computer organization

2.1.1 Computer architecture

- Control Unit (CU): retrieves instructions from primary memory and handles the sequence of their execution.
- Arithmetic Logic Unit (ALU): performs all basic arithmetic, logic, input/output operations.
- Memory Address Register (MAR): holds memory address of data to be used by the ALU
- Memory Data Register (MDR): holds data to be used by the ALU.

Primary memory

- Cache: a small amount of fast temporary memory placed between CPU and main RAM, used as cache. Separated into L1 and L2 caches.
- Random Access Memory (RAM): temporary data
- Read-only memory (ROM): permanent data

Instruction cycle
Known as the fetch-decode-execute cycle:

1. Fetch instructions from primary memory to CU
2. Decode instruction in CU
3. Execute instruction
4. Store result of execution and check for next instruction

2.1.2 Secondary memory

Secondary memory is a slower type of memory for data storage that can be both read to and written from. It is non-volatile. Examples are USBs, hard drives, etc.

2.1.3 Operating and application systems

Functions of an OS

- Peripheral communication
- Memory management
- Resource monitoring and multi-tasking
- Networking
- Disk access and data management
- Security

Software applications
Software applications: word processors, spreadsheets, database systems, etc.

GUI: graphical user interface

2.1.4 Binary representation

bit: basic unit of information, value of either 0 or 1.

$$1 \text{ byte} = 8 \text{ bits}$$

Number systems: decimal - base 10, binary - base 2, hexadecimal - base 16.

Convert from decimal to binary: repeatedly divide by 2 and keep track of the remainders (which will be 0 or 1). The reverse order of the remainders obtained is the number in binary.

Covnert from binary to decimal: multiply each digit with its respective multiplier and sum the results.

2.2 Negatives in binary

Most significant bit (MSB): the bit having the largest value in a binary number.

Least significant bit (LSB): the bit having the smallest value in a binary number.

Negative numbers (Two's complement): used to represent signed binary numbers. To find a number's two's complement representation, first write the absolute value in binary, then invert the bits, then add 1. The largest number possible in this system is 127, while the smallest is -128.

Negative numbers (Sign and magnitude representaiton): the left-most bit represents the sign, while the rest carries the magnitude.

Binary fractions (fixed-point representation): the number is divided at a chosen fixed point. the part to the left of it is the integer part while the right is the fractional part. To the left of the fixed point are increasing powers of two as per usual, and to the right are decreasing powers, just like in the normal decimal system.

Hexadecimal: base-16 system.

ASCII: character-encoding scheme for English alphabet.

Strings: sequence of characters.

Colours: can be represented using hexadecimal RGB colour values - a six-digit hexadecimal number where the first two digits represent red, next two green, last two blue.

2.2.1 Simple logic gates

[INCOMPLETE]

3 Networks

Definitions:

- bus topology: computer network in which a 'bus' connects all devices together through a common cable.
- cable: wire or fibre used to connect computers in a network.

- check digit: extra digit added to numerical data that is used to check for data integrity after data has been transmitted, stored, input or processed.
- data integrity: accuracy of data
- check sum: error-detecting procedure that generates a sum from the digits of a number.
- data packet: portion of a message that is transmitted through a network. contains data such as check digits and destination address.
- gateway: link residing between computer networks and is responsible for converting data passing through into the appropriate format.
- handshaking: exchange of predetermined signals to signify that a connection has been established.
- hub: network connection point for devices. data arriving at a hub is copied and sent to all connected devices.
- switch: like a hub, but can identify which device is connected to which port, and then transmit data to specific port/device as needed.
- router: connects multiple networks together to form larger networks
- ISDN: integrated services digital network - international communication standard allowing for transmission of audio/video/etc. over digital telephone lines.
- local area network (LAN): network where connected computers are within a limited geographical area.
- microwave transmission: electronic communication without cables.
- modem (modulator/demodulator): converts digital signals into audio signals and back. allows distant communication.
- network: interconnected computer systems, which can share data.
- packet: group of bits.
- packet switching: network communication method that create and transmits small units of data through a network, independetly of the overall message.
- parity bit: error-detecting procedure that appends binary digits to a group of binary digits. the sum of the digits, including the appended digit, establishes the accuracy of the data.
- protocol: international rules that ensure the transfer of data between systems.
- TCP/IP: protocol used to connect hosts to the internet.
- wide area network (WAN): larger geographical area than LAN.

3.1 Networks

3.1.1 Network fundamentals

Types of networks

- local area network: connects devices within a limited geographical area, such as an office building or school. one advantage is it allows sharing of peripheral devices (like printers) between devices.
- wireless local area network: like LANs, but wireless. Being wireless is convenient, but one disadvantage is that it is also much less secure compared to LANs.
- virtual local area network: this is used to partition LANs into *logical separate networks*. For example, different departments within the same office might have their own VLAN, allowing the network to be more organized without much costs.
- wide area network: connects devices across a large geographical area. The internet is a WAN.
- storage area network: a network created for accessing large storage devices from servers more conveniently.
- intranet: broad term for a collection of private networks utilizing protocols like TCP/IP. they can be thought of as a more private internet.
- internet: connects a large number of networks together. largest WAN in the world. it is also a decentralized network. no one entity controls the internet.
- extranet: utilizes the internet to allow **controlled access** by specific users to a specific LAN or WAN.
- internet of things: network of 'things' that are able to connect to the internet, e.g. refrigerators, cars, etc.
- virtual private network: allows clients from remote locations to connect to the network and appear to be inside the LAN/WAN as if they were physically present. it is much more secure and can bypass geographical restrictions.
- personal area network: much shorter range; connects devices centered around one individual, e.g. your mobile and drawing tablet and printer and camera and so on.

- peer-to-peer: a network which does not utilize the client-server model. Instead, it is a distributed network where all systems, called nodes or peers, act as both the client and the server at the same time, consuming and supplying resources from and to other devices at the same time.

Importance of standards
Standards enable compatibility through a common 'language' and predefined rules that all parties agree upon.

Communication over networks
Open Systems Interconnection Model, established by the International Standards Organization (ISO). Contains seven layers aimed at facilitating communication across systems:

- Physical: transmits binary data over media between devices. e.g RS232-C.
- Data link: error handling of physical transmission, flow control and handling transmission rates. e.g Ethernet
- Network: handles routing of packets across a network. e.g IP
- – (Above: Physical communication) – (Below: Virtual communication)
- Transport: end to end connection (between hosts). this is where data segments are defined, data transfer occurs, and then the data is reassembled at its destination. e.g TCP
- Session: managing sessions between two users. e.g windows system core protocol.
- Presentation: data format information, compression information, encryption information, etc. e.g Portable Network Graphics (PNG)
- Application: performs services consumed by end users. e.g HTTP

TCP/IP Model: Similar to the OSI model, describes functions taking place at each layer of protocols within the TCP/IP suite:

- Network Access: layer 1 and 2 of OSI
- Internet: layer 3 of OSI
- Transport: layer 4
- Application: layers 5, 6 and 7

Virtual Private Networks (VPNs)
All traffic on VPNs is encrypted, authenticated and then sent along virtual tunnels. Secure VPN technologies include the internet protocol security protocol (IPSec protocol), the Secure Sockets Layer, or Transport Layer Security with encryption.

Common VPN types:

- Site-to-site VPN: Connects entire networks together through gateways, for secure data interchange.
- Remote-access VPN: Connects individual hosts to private networks. Every host has a VPN client software installed.

Use of a VPN
Benefits:

- Easier communication
- Secure connections
- Remote access

3.1.2 Data transmission

Necessity of Protocols
Protocols are important because they define rules about message/data formats, error checking methods, data compression and ecnryption/decryption and so on.

Protocols also guarantee:

- Source integrity: identity of the sender has been validated
- Flow control: Controlling the rate of data transmission to prevent overload of resources while also maintaining efficiency
- Congestion management: Handling congestions which occur when requests to the network exceed the offered capacity
- Deadlock prevention: preventing situations where two or more networks are waiting for the other to finish the process, thus neither is able to finish.
- Error checking and correction: determining if an error has occurred, and recovering data lost is possible.

Factors that affect speed of data transmission

- Bandwidth of network
- Data transfer rate of storage devices
- Interferences
- Malicious software
- Packet loss and retransmission

Data compression

- Lossy: some loss of information is acceptable, and there is no way to retrive the original data.
- Lossless: reduces the number of bits by identifying redundant data and encoding it in a more compact format to save space without losing the original data.

Packet switching
A data packet is a unit of information in a form suitable for transferring between computers.

Packet switching is a method where data is grouped into packets. The original file is divided into packets, each of which may follow a different path to the destination.

3.1.3 Wireless Networking

Advantages:

- Less hassle and possibly costs
- Easy to access

Disadvantages:

- Relatively low speed
- Less secure and higher error rates

Hardware components of a wireless network

INCOMPLETE

Software components of a wireless network

INCOMPLETE

Methods of network security

- Passwords
- Antivirus program
- Firewall (software/hardware)

Encryption
Encryption: Uses mathematical algorithms and encryption keys to protect data. In symmetric-key encryption, the same key is used for encryption and decryption. Public-key encryption uses a public key which is available to everyone for encryption, and a private key for decryption. Both are mathematically linked. This is used on the internet, such as TLS and SSL protocols.

4 Computational Thinking

4.1 General principles

4.1.1 Algorithms

Algorithms are a series of instructions designed to solve a problem. Algorithms can be expressed in different ways, for example with natural language, flow charts or diagrams, pseudocode or programming languages.

Algorithms are (often) procedural, that is, they follow a set of given instructions one by one. They may also include sub-procedures, which break the problem into smaller sub-problems and solve them one by one, in order to more easily solve the original problem at hand.

Algorithms can consist of conditionals for decision-making, iteration constructs for repeating tasks, inputs and outputs for manipulating or consuming data, pre-conditions (data and information needed before a procedure is called) and post-conditions (after a procedure finishes), exceptions (events that might disrupt the flow of the algorithm's execution) and so on.

Concurrency means something happening at the same time as something else. In terms of computation, it refers to instructions being processed at the same time. Here, procedures are broken into sub-procedures that can be handled concurrently by processors. This helps efficiency but requires more planning and coordination.

Abstraction refers to when unnecessary low-level implementation details of something are hidden, and only the relevant and fundamental parts of it are considered.

Object-oriented programming uses abstraction by representing entities as programming objects. These objects describe the data/properties and behavior/methods of the entity it represents.

Variables are identifiers that can be used to reference values stored in the computer, during program execution.

4.2 Connecting computational thinking and program design

4.2.1 Linear arrays

Linear arrays are one-dimensional arrays that can store multiple values, unlike variables.

Sequential search: goes through each item one by one until the search is successful or there are no more items.

Binary search: works on sorted lists by dividing them into half at each iteration and recursively searching through the sublist containing larger or smaller elements depending on the search value

Bubble sort: sorting algorithm; repeatedly steps through the array, compares adjacent elements and puts them in order. makes multiple passes until no swaps are necessary and the array is sorted. slow and impractical algorithm.

Selection sort: sorting algorithm; divides the input array into two sub-arrays, one containing the sorted elements, the other unsorted elements. At first, the size of the first sub-array is 0. The algorithm goes through the unsorted array looking for the max/min element and places it at the leftmost position, next to the sorted array, and then extends the borders of the sorted array. it then repeats until the size of the unsorted array is 0.

4.2.2 Collection operations

- addItem(): add an item to the collection
- getNext(): returns items from the collection, starting from the first and then the next items upon repeated calls of this method.
- resetNext(): resets the getNext() function to go back to the beginning.
- hasNext(): returns true if the collection has items left that have not yet been returned by getNext(), otherwise false.
- isEmpty(): whether the collection has items or not

4.2.3 Fundamental operations of a computer

- LOAD: load a value from RAM into the CPU's control unit
- STORE: store a value from the control unit into the RAM
- ADD: adds two numbers together
- COMPARE: compare two numbers together

Compound operations are made up of many fundamental operations, to do more complex tasks.

4.3 Introduction to programming

4.3.1 Features of a programming language

A programming language is a combination of its semantics (the meaning of every construction possible in the language) and syntax (the structure of the language). A grammar is a meta language used to define the syntax of the language. Languages also have a specific vocabulary, i.e keywords that have special meanings. Commands in computer languages are always unambiguous.

4.3.2 Higher-level languages

Higher-level languages are necessary because they make program development simpler, faster and more understandable. However, these languages cannot be executed by a CPU which only understands simple fundamental instructions. Thus we use compilers and interpreters:

- *Compiler:* translates from high to low level only once, and produces an output file that can be directly executed by the CPU.
- *Interpreter:* reads the high-level instructions, analyzes, transaltes and executes them on the fly. This happens every time the program needs to be run.

Some languages are compiled to bytecode, which is a more simpler and compact set of instructions compared to high-level languages but still not as low-level as direct CPU instructions. This bytecode is executed by a virtual machine program designed for that bytecode.

4.3.3 Common features of a language

- Variable: stores a data element
- Constant: stores data that does not change
- Operator: performs operations on data types
- Object: made up of data and methods that describe a certain entity.

4.3.4 Advantages of using sub-programs

- Code reuse
- Breaking down problem into easier sub-problems
- Maintainability and traceability
- Abstraction

5 Abstract Data Structures

5.1 Abstract Data Structures

5.1.1 Thinking recursively

Recursion is when a method calls itself, until some terminating condition is met. It does not use looping constructs (like for or while loops).

It follows the basic problem solving principle of breaking a problem down into smaller sub-problems and solving them.

A recursive algorithm can be written using looping constructs, and vice versa.

Examples of recursive algorithms: Tower of Hanoi problem, Koch snowflakes.

5.1.2 Two dimensional arrays

A one-dimensional array can be considered a single line of elements. A two-dimensional array would then be a *table* of elements.

2d-arrays are indexed by two subscripts: array[y][x]. If we think of the 2d-array as a plane, then the y subscript indicates the position of the element along the vertical axis (starting from 0 at the top), while the x subscript indicates the position along the horizontal axis (starting from 0 at the left). The top-left element would thus be array[0][0], while the bottom right would be array[n-1][n-1] fora 2d array of size n by n.

5.1.3 Stacks

Stacks are a last-in, first-out (LIFO) data structure, meaning that the last element inserted into the stack is the first element retrieved from it.

Stacks have three important methods:

- push(): adds an item onto the stack
- pop(): removes the last item added to the stack
- isEmpty(): checks if the stack has any elements or not

Applications:

- Back button of a web browser: the last page visited is the first one the back button points to. as we keep pressing it, the browser keeps going back to the previous pages opened. thus it follows a LIFO structure

5.1.4 Queues

Queues are a first-in, first-out (FIFO) structure.

The three queue methods are:

- enqueue(): adds an item to the end of the queue

- dequeue(): removes an item from the start of the queue
- isEmpty(): checks if queue is empty or not

Applications:

- Used to model physical queues

5.1.5 Linked lists

Each node in linked lists contains two items: one is the data and the other is a pointer to the next element. A header node is used at the beginning which only contains one item, the pointer to the next node. The last node's pointer points to a null value to indicate the end of the list.

To remove a node, simply change the pointer of the previous node to the next node instead.

To insert a node in some position, change the pointer of the previous node to the new node, and set the pointer of the new node to the next node.

5.1.6 Double linked lists

In double linked lists, each node has two pointers - one points to the previous element, and one points to the next element.

5.1.7 Circular linked lists

In circular linked lists, the last node does not point to a null value, but rather back to the header node (or the first node if there is no header node).

5.1.8 Trees

Definitions:

- Children: the nodes below a given node
- Key: the data field of a node
- Leaf: a node with no children

- Level: distance of a particular node from the root node
- Height: number of edges from the top node to the deepest leaf
- Parent: the node above a given node
- Path: sequence of nodes travelled to get from one to the other
- Root: the node at the top of the tree, with no parents
- Subtree:
- Traversing: to visit all the nodes of the tree in some order
- Visiting: arrive at a node and perform some operation on it. if the algorithm just passes over the node then it is not considered a visit

Inorder traversal: The algorithm calls itself on the left subtree, then visits the current node, then calls itself on the right subtree. (LCR)

Preorder traversal: The algorithm visits the current node, then calls itself on the left subtree, then calls itself on the right subtree. (CLR)

Postorder traversal: The algorithm calls itself on the left subtree, then calls itself on the right subtree, then visits the current node.

An easy way to remember is that the namer of the traversal tells you when the current node is visited, otherwise the algorithm first calls itself on the left subtree, and then the right one. For inorder, the current is *in* between the other positions, L-C-R. For pre, it is *before* the others, so C-L-R. For post, it is *after* the others, so L-R-C.

Operations on binary trees

Binary trees are sorted, meaning that values to the left are smaller and values to the right are larger. This is useful for determining the result of adding or removing nodes from a tree: add/remove the node in its correct position.

5.1.9 Static and dynamic data structures

In static structures, the size of the structure is constant and pre-determined. Memory allocation is fixed and the space reserved will always be available. Since the size is predetermined, there is also less overhead and hence faster speed.

In dynamic structures, the size changes depending on how many items are being stored. They are generally slower due to overhead of constantly checking memory allocation and size to make sure there is enough space. However, they are more space efficient since only the required amount of memory is allcoated.

6 Resource Management

6.1 Resource Management

6.1.1 Critical resources

Primary memory: This is connected directly to the CPU which uses it to store instructions. It includes RAM of which there are two types: static and dynamic. SRAM is more expensive, but retains data as long as there is power. Data in DRAM gradually leaks and needs to be refreshed preiodically. Read-only memory (ROM) is also primary memory which is slower than RAM. It holds critical information needed to boot the PC. It usually holds the Basic IO System (BIOS) and sometimes the entire OS as well.

Secondary storage: these are low cost, have larger capacity and permenent storage. There are two broad types: direct access (USBs, DVDs), where data can be accessed, retrieved, and stored *directly* without having to read through all the previous data. The other type is sequential acccess (magnetic tapes), where previous data has to be read sequentially until the requested data can be located.

Processor speed: MIPS (million instructions per second) is used to measure processor speed. Clock rate refers to the frequency at which the processor runs, and can be a rough indicator of processor performance. It is usually measured in gigahertz.

Bandwidth: Memory bandwidth is the rate at which data can travel from SRAM to DRAM to the processor and vice versa. It is expressed in MB/s. Peak theoretical bandwidth is the theoretical limit, while sustained memory bandwidth is usually less and affected by various factors.

Screen resolution: number of pixels, height times width. The more pixels, the better the resolution.

Disk storage: e.g. hard disk drive, solid state drives, etc.

Sound processor: Sound cards are used to process audio.

Graphics processor: these are massively parallel processors which are very efficient at manipulating and processing graphics and images.

Network connectivity: A Network Interface Card is used by devices for network connections.

6.1.2 Availability of resources

Mainframes: used by large organizations to handle large-bandwidth communication, bulk data processing, etc.

Supercomputers: very fast and expensive, focused on mathematical calculations. Performance measured in floating-point operations per second (FLOPS).

Other devices: servers, PCs, tablets, digital cameras, etc.

6.1.3 Limitation of resources

[INCOMPLETE]

6.1.4 Problems due to insufficient resources

[INCOMPLETE]

6.1.5 Role of the Operating System

Managing memoery and processes: [INCOMPLETE]

Managing peripherals: [INCOMPLETE]

Scheduling: [INCOMPLETE]

Policies and account management: [INCOMPLETE]

Interrupts: [INCOMPLETE]

Polling: [INCOMPLETE]

Dedicated OSs: [INCOMPLETE]

OS and complexity hiding: [INCOMPLETE]

7 Control

7.1 Control

7.1.1 Centralized control systems

Control system: one or more devices that guide other devices or systems, allowing for complete automation of tasks. They may contain sensors to gain feedback from the environment and motors to control actuators in an appropriate manner. An example is automatic doors in supermarkets, which have sensors to detect when someone is entering or leaving and send signals to a microprocessor. This allows the operating system to operate the actuators and open/close doors.

Control systems are mostly input, process and output systems. An input is provided to some algorithm, which performs some action and provides an output. This output can then be measured again by the control system and provided back as input.

Examples:

- Heating systems: an initial temperature is taken as input, which indicates the ideal value of the output and the 'goal' of the system. Feedback is used heavily. Sensors measure the temperature, the processor decides what action to perform to reach the goal, the output transducers change the temperature accordingly. The sensor again takes the new temperature as input, and the loop continues.
- Taxi meters:
- Elevators: takes the desired destination as input and processor decides how to operate machinery to reach the destination. other inputs may be taken such as current location, location of other elevators, load on the elevator etc.
- Washing machines: sensors determine and control the load size in the washing machine, water level, temperature, and the user interface (buttons or a touch screen interface). The processor takes this information and decides how to operate the machinery. actuators are used to turn motors and operate the washing machine.
- Process control: used to maintain the output of a specific process within a desired range. open and closed loop controllers are used. For example, a tank with liquid in it and a coil for heating the liquid. An open feedback loop would need to know the relationship between the heat dissipated and

the temperature increase of the liquid to determine the liquid's temperature. In closed loop, the temperature of the liquid would be detected using temperature sensors, and the coil temperature adjusted accordingly to achieve the desired liquid temperature.

- Device drivers: computer programs for controlling computer devices, such as keyobards and printers and graphics cards. these let the OS access the device through a layer of abstraction.
- Domestic robots: e.g. automatic vacuum cleaners. a number of sensors are used within these to gather information about their surroundings and perform their task.
- GPS systems: a large complex computer controlled system. a GPS receiver includes a sensor that locates multiple GPS satellites in space, figures out the distance from each, calculates the time difference and uses a process called 'trilateration' to deduce its location. also includes inputs from users to pinpoint desired locations.
- Traffic lights: as discussed before, traffic lights can be fixed time, which are open loop control systems in that they dont receive feedback from the environment. the are pre-configured to change at intervals. dynamic control traffic lights use a close loop control system, that is they gather feedback from the environment, to dynamically control traffic, based on some appropriate algorithm. they may use mounted sensors, located on the traffic lights, or embedded sensors in the surface of the road.

Use of microprocessors and sensor inputs

- General purpose: capable of running a wide range of programs, usually integrated into a larger system, e.g. CPUs in desktop computers
- Embedded (micro)controller: standalong chips designed to perform some specific task, and do not need an entire computer system. found in many control systems described above. generally need less power and smaller in size as well.
- GPU: special kind of microprocessor for graphics and images.

The main advantage of microprocessors in control systems is that process data much faster than humans could and can react to changes quickly. They are also automated and hence less error-prone, and can operate with little off-time.

The main disadvantage is that they are pre-programmed and cannot handle unexpected situations, which may be common in some systems. They also cannot function if there is a power shortage.

Input sensors convert physical quantities such as speed, temperature to discrete digital quantities using an analog-to-digital converter. These can then be read by microprocessors. Sensors share the following properties that determine their quality:

- Accuracy of measurement for the physical quantity
- Range of measurement for the physical quantity
- Resolution (the smallest increment detectable) for the physical quantity.

Different input devices for data collection
Sensors are used as input devices. There are many types of sensors:

- sound
- motion
- vibration
- image/optical: active pixel sensors (for cameras), infrared (alarm systems)
- pressure
- temperature
- proximity

Transducers are devices that convert one form of energy into another. In computer systems, they are responsible for converting physical quantities into electrical signals and vice versa. This process is called transduction. Sensors can also be categotized as transducers since they detect physical quantities and convert them into digital quantities.

Actuators are another type of transducers, used when returning the output of the processor. They are responsible for moving some kind of mechanism. They take some form of energy, usually electric current, and convert it into motion.

Thus a control system may look like:
Input → Transducer (sensor) → Processore → Transducer (actuator) → Output

Role of feedback
Feedback is the process wher information from the result of an output is used as new input to the control system, creating a sort of loop.

It is important in control systems that react to their environment and its changes. It helps keep the system stable.

For example, a dynamic traffic light which uses feedback: first, the motion sensor detects that there are no cars (input), the processor decides to turn the light red and allow pedestrians to pass (output). After a while, a line of cars forms since the light is red. The system again detects (2nd input) this and now turns the light green to allow cars to pass (2nd output). This is an example of how feedback from one output is used to determine the next input and action, allowing the system to remain in a stable state.

Social impacts and ethical considerations

- Electronic tagging: [INCOMPLETE]
- Surveillance: [INCOMPLETE]
- Safety systems: [INCOMPLETE]

7.1.2 Distributed systems

Centralized system: all resources reside in a single system. clients need to be able to connect to this system. Main benefit is lower operational costs, greater security, less administrative overhead and backup complexity.

Distributed system: resources are distributed across various systems and each system is self-sustaining. a central system may coordinate distributed ones, but each can still function on its own, and failure of one distributed system will not be fatal to the overall system. these systems are more scalable and fault-tolerant.

Agents: these can by anything that perceives its environment, through sensors and acts upon it through effectors.

Autonomous agents: these are entities acting on behalf of an owner, with a degree of autonomy and little to no interference from the owner. Their environment often plays a significant role in their function:

- Accessible/Inaccessible: the agent may or may not be able to obtain compelte and accurate information about the environment.
- Deterministic/Non-deterministic: whether the agent's actions will have a pre-defined, guaranteed effect. The next state of the environment depends entirely on its initial state.
- Episodic/Non-episodic: whether an agent's actions are divided into 'episodes' or not. Episodes are independent, i.e. actions in previous episodes do not affect actions in subsequent episodes.
- static/dynamic: the environment may change while the agent is deliberating an action

- discrete/continuous: the environment may have discrete, clearly defined states or continuous states.