

COMP1200

Perspectives on Computing

Lecture notes by
Brendan McKay

Review: $O()$ notation – (1)

Suppose we have an algorithm for solving some problem, and we want to investigate how efficient it is.

Instances of the problem have a **size** that we will call n .

Examples:

- Sort n integers into ascending order.
- Find the inverse of a matrix of size $n \times n$.
- Find the shortest route that visits a particular set of n cities.

Review: $O()$ notation – (2)

Observation 1: The same algorithm will run faster on a faster computer, or if it is programmed in a more efficient language.

Solution: Ignore constants, and only care about how fast the cost increases as n increases.

For example, we will regard $3n^2$, $6n^2$ and $100n^2$ as essentially the same.

Review: $O()$ notation – (3)

First approximation:

$O(n^2)$

means (approximately)

“ at most a constant times n^2 ”

For example:

$3n^2$, $6n^2$ and $100n^2$ are all $O(n^2)$.

Review: $O()$ notation – (4)

But note:

“ **at most** a constant times n^2 ”

It doesn't say “at least” .

For example:

$3n$ and 100 are also $O(n^2)$, since (for positive n):

$$3n \leq 3n^2$$

$$100 \leq 100n^2$$

Review: $O()$ notation – (5)

In general, a function $f(n)$ is $O(n^2)$ if

$$f(n) \leq Cn^2$$

for some constant C .

$$6n^2 = O(n^2) \quad \text{obviously}$$

$$2n^2 + 5n = O(n^2) \quad \text{since } 2n^2 + 5n \leq 7n^2$$

$$2n^2 - 5n = O(n^2) \quad \text{since } 2n^2 - 5n \leq 2n^2$$

$$6n^3 \neq O(n^2) \quad \text{since } 6n^3 > Cn^2 \text{ if } n \text{ is large enough}$$

Review: $O()$ notation – (6)

More generally, a function $f(n)$ is $O(g(n))$ if

$$f(n) \leq Cg(n)$$

for some constant C . (not exactly, see later)

$$6n^5 = O(n^5) \text{ obviously}$$

$$2n^4 + 5n^2 = O(n^4) \text{ since } 2n^4 + 5n^2 \leq 7n^4$$

$$2n^3 - 5n^2 + 2 = O(n^3) \text{ since } 2n^3 - 5n^2 + 2 \leq 4n^3$$

$$6n^3 - 8n^2 \neq O(n^2) \text{ since } 6n^3 - 8n^2 > Cn^2 \text{ for large } n$$

Review: $O()$ notation – (7)

In writing a function in terms of $O()$ only the fastest growing term matters.

$$36n^7 + 12n^5 - n^{3/2} + 24n + 6 = O(n^7).$$

n^4 grows faster than n^3 , which grows faster than n^2 , which grows faster than n , which grows faster than $n^{1/2}$. And so on.

Higher powers of n grow faster.

Review: $O()$ notation – (8)

$\log(n)$ grows slower than any positive power of n .

A^n (if $A > 1$) and e^{Bn} (if $B > 0$) grow faster than any power of n .

$$2^n + n^5 = O(2^n)$$

$$2n^4 + 5n^3 \log n = O(n^4)$$

$$24n3^n + 253^n + n^9 = O(n3^n)$$

In all cases, select the term that grows fastest and forget about its constant.

Review: $O()$ notation – (9)

In analysing algorithms, **logarithmic factors** usually arise in one of two equivalent ways.

- If we start with a constant (often 1) and keep multiplying it by a constant (often 2) until it reaches n , the number of steps required is $O(\log n)$.
- If we start with n , and keep dividing it by a constant (often 2) until it becomes small (often 1), the number of steps required is $O(\log n)$.

Review: $O()$ notation – (10)

Logarithmic factors illustrate a **small quibble** in the definition of $O()$.

We would like to write

$$n \log n + 3n = O(n \log n)$$

but there is **no** constant C such that

$$n \log n + 3n \leq Cn \log n$$

when $n = 1$, because $\log 1 = 0$.

Review: $O()$ notation – (11)

To avoid this quibble, we agree to **ignore small n** .

The **real meaning** of saying a function $f(n)$ is $O(g(n))$ is that

$$f(n) \leq Cg(n)$$

for some constant C , **when n is large enough**.

We can make “when n is large enough” formal by saying “for $n \geq n'$ for some n' ”.

Review: $O()$ notation – (12)

Another quibble is that an algorithm might take different amounts of time for different problem instances of the same size.

Example: sorting n numbers may take less time if the numbers happen to be already sorted.

The most common convention is to take the **worst case** time. For each n , use the longest time that any problem instance of size n takes.

Some people also consider **average case** time, but we won't.

Review: $O()$ notation – (13)

An algorithm is **polynomial time** if the running time is $O(n^k)$ for some k . A problem is **polynomial time** if there is some polynomial time algorithm for solving it.

- The problem “sort a list of n numbers” is polynomial time because there are algorithms like selection sort ($O(n^2)$) and merge sort ($O(n \log n)$) for solving it.
- The problem “find the shortest route visiting n specified cities” is *not known* to be polynomial time: every known algorithm takes at least A^n time for some $A > 1$, in the worst case.

Review: $O()$ notation – (14)

Recall that $f(n)$ is $O(g(n))$ if there is some constant C such that $f(n) \leq Cg(n)$ for large enough n . That is, $O()$ gives an **upper bound**.

There are also:

- $f(n)$ is $\Omega(g(n))$ if there is some constant $B > 0$ such that $f(n) \geq Bg(n)$ for large enough n . That is, $\Omega()$ gives an **lower bound**.
- $f(n)$ is $\Theta(g(n))$ if $f(n)$ is both $O(g(n))$ and $\Omega(g(n))$. That is, $\Theta()$ indicates both upper and lower bounds.

Review: $O()$ notation – (15)

$$13n^3 + 5n^2 = O(n^3)$$

because $13n^3 + 5n^2 \leq 18n^3$, and

$$13n^3 + 5n^2 = \Omega(n^3)$$

because $13n^3 + 5n^2 \geq 13n^3$, so in total

$$13n^3 + 5n^2 = \Theta(n^3).$$

Review: Number representations – (1)

Unsigned (non-negative) integers are represented in binary in the obvious fashion.

00101001 represents $2^5 + 2^3 + 2^0 = 32 + 8 + 1 = 41$.

Conversions: For small numbers, just use the definition.

97 = 64 + 32 + 1 = $2^6 + 2^5 + 2^0$ in binary is **1100001**.

Add leading **0s** as desired.

Better methods exist for larger numbers.

Review: Number representations – (2)

To represent integers that might be negative, some convention is required. The most common convention is **“two’s complement”**.

| binary | unsigned | signed | binary | unsigned | signed |
|--------|----------|--------|--------|----------|--------|
| 0000 | 0 | 0 | 1000 | 8 | -8 |
| 0001 | 1 | 1 | 1001 | 9 | -7 |
| 0010 | 2 | 2 | 1010 | 10 | -6 |
| 0011 | 3 | 3 | 1011 | 11 | -5 |
| 0100 | 4 | 4 | 1100 | 12 | -4 |
| 0101 | 5 | 5 | 1101 | 13 | -3 |
| 0110 | 6 | 6 | 1110 | 14 | -2 |
| 0111 | 7 | 7 | 1111 | 15 | -1 |

Review: Number representations – (3)

Definition of two's complement representation.

- If the first bit is **0**, the value is positive.
- If the first bit is **1**, the value is negative.
- The representation of $-n$ is the two's complement of the representation of n .

We need to know the total number of bits being used so that we know which bit is the “first bit”. Typical sizes in practice are 8, 16, 32, 64. We often hear “32-bit two's complement” meaning “two's complement representation in a word of 32 bits”.

Review: Number representations – (4)

Two's complement operation – method 1:

Flip every bit then add 1.

| | |
|----------|----------------|
| 01011100 | represents +92 |
| 10100011 | |
| +1 | |
| 10100100 | represents –92 |

Two's complement operation – method 2:

Flip every bit to the left of the last 1.

| | |
|------------------|----------------|
| 01011 100 | represents +92 |
| 10100 100 | represents –92 |

Review: Number representations – (5)

Question: Convert -34 to 8-bit two's complement.

Solution:

1. Convert $+34$ to binary using 8 bits: $34 = 00100010$
2. Take the two's complement: 11011110 .

Question: Convert the 8-bit two's complement value 11011110 to decimal.

Solution:

1. First bit is 1 , so take two's complement: 00100010
2. $00100010 = 34$, so answer is -34 .

Review: Number representations – (6)

One reason two's complement is popular:

The standard addition algorithm works for both positive and negative numbers.

$$\begin{array}{r} 10100100 \\ + 00101000 \\ \hline 11001100 \end{array} \quad \begin{array}{l} \text{represents } -92 \\ \text{represents } +40 \\ \text{represents } -52 \end{array}$$

Any carry off the left end is discarded.

Network Protocols

- Protocols are the rules by which network activities are conducted
 - e.g. transmitting messages around a network
- Token Ring Protocol
 - transmit messages in one direction around the ring
 - when the message reaches its destination, a copy is taken and the message is forwarded on
 - when the message reaches its starting point, the message is removed from the ring

Token Ring Protocol

- The token ring protocol relies on inter-machine cooperation
- Machines are not permitted to send any message they want
- A unique bit pattern (token) is passed around the ring
- Possession of the token allows a machine to transmit its own messages
- Without the token only forwarding is allowed
- When the message completes its cycle the machine forwards the token to the next machine

Carrier Sense, Multiple Access with Collision Detection (CSMA/CD)

- Used in bus topology networks
- Each message is broadcast to all machines on the bus
- Machines monitor all messages but keep only their own
- To transmit a message, a machine waits for no activity
- If two machines transmit at once, they both wait for a random period of time before trying again

Ethernet

- Ethernet is a set of standards for implementing a LAN with a bus or star topology
- Ethernet has become one of the most popular standards for transmitting data between computers on a LAN
- Ethernet will transmit data at either 10 or 100Mbps

Combining Networks

- Sometimes it is necessary to combine networks into larger ones
- This is achieved by means of different devices
- Repeater
 - connects two buses
 - passes messages back and forth without considering their meaning

Combining Networks

- Bridge
 - connects two buses
 - passes messages across the connection only if that message is destined for a machine on the other side
- Switch
 - connects more than two buses
 - when connected they look like spokes on a wheel
 - passes messages only to the spoke that contains the machine that a message is destined for

Combining Networks

- Sometimes, networks that have incompatible characteristics must be connected
 - A ring network using the token ring protocol
 - A bus network using CSMA/CD
- Build a network of networks called an internet
- The individual networks maintain their individuality
- Connection is handled by a machine called a router
- It's task is more significant than a bridge or repeater

Internet Protocols

- A layered approach to Internet software
- Principal task of networking software is to provide an infrastructure for transferring messages
- Internet software has 4 layers of routines
 - Application layer
 - Transport layer
 - Network layer
 - Link layer

Internet Protocols

- Messages originate in the application layer
- They are passed through the transport layer and the network layer
- They are transmitted by the link layer
- The target machine receives the message via the link layer
- The message is then passed up through the network layer and the transport layer to the application layer

Internet Protocols

- Application layer
 - Software units that use Internet communication to carry out tasks
 - Responsible for providing an address for the message that is compatible with the transport layer

Internet Protocols

- Transport layer
 - Ensures messages are properly formatted for transport
 - Divides message into small segments
 - Adds sequence numbers to each segment (so that it can be re-assembled later)
 - Attaches addresses to each segment (packet)
 - Passes the packets to the network layer

Internet Protocols

- Network layer
 - Responsible for forwarding the packets it receives from one network to another
 - If the destination address is within the current network, it sends the packets to that address
 - Otherwise it sends them to a router in the network
 - Once it knows where to send the package, it appends this intermediate address to the packets and passes them to the link layer

Internet Protocols

- Link layer
 - sends the packets to the intermediate address
- The message is received by the link layer and passed to the network layer
- If the destination address is not local it is passed to the link layer with a new intermediate address
- If the packets have reached their final destination, they are passed to the transport layer
- The transport layer re-assembles the packets and passes the message to the application layer

TCP/IP

- The TCP/IP protocol suite is a collection of protocols used to implement the 4 level hierarchy
- TCP = Transmission Control Protocol
- IP = Internet Protocol
- There are many more
- TCP defines a version of the transport layer
- The User Datagram Protocol (UDP) also defines a version of the transport layer

TCP/IP

- Two basic differences
 - TCP sends its own message before sending packets
 - TCP transport layers at origin and destination communicate to ensure all packets have been sent correctly
- IP is the Internet's standard for the network layer
- A hop-count is appended to each packet
- This limits the number of times a packet can be forwarded
- 32 is usually enough