



EEE4120F



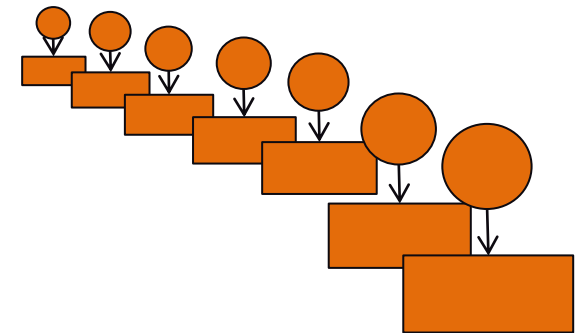
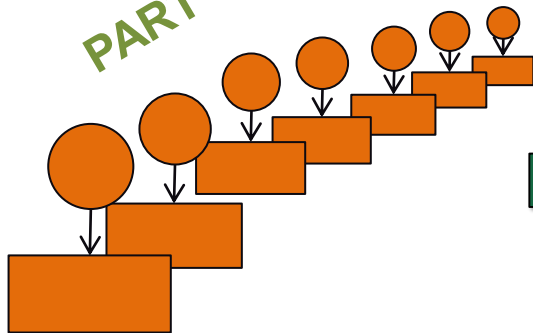
High Performance Embedded Systems

Lecture 11: Design of Parallel Systems

Presented by

Simon Winberg

PART 3/3



Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)

Steps in Designing Parallel Programs

**Continuing the path
to HPES Applications**



Hardcore
competent
HPES
programmers
(leading the
way to greater
feats)

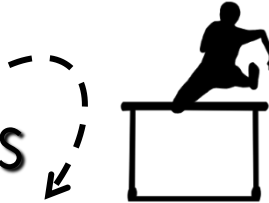


← Sequential programmers in
their comfort zone.

The steps in designing parallel programs

The hardware may be done first... or later.

The main steps:

1. Understand the problem
2. Partitioning (separation into main tasks)
3. Granularity
4. Communications  see lecture 10
5. Identify data dependencies
6. Synchronization
7. Load balancing
8. Performance analysis and tuning



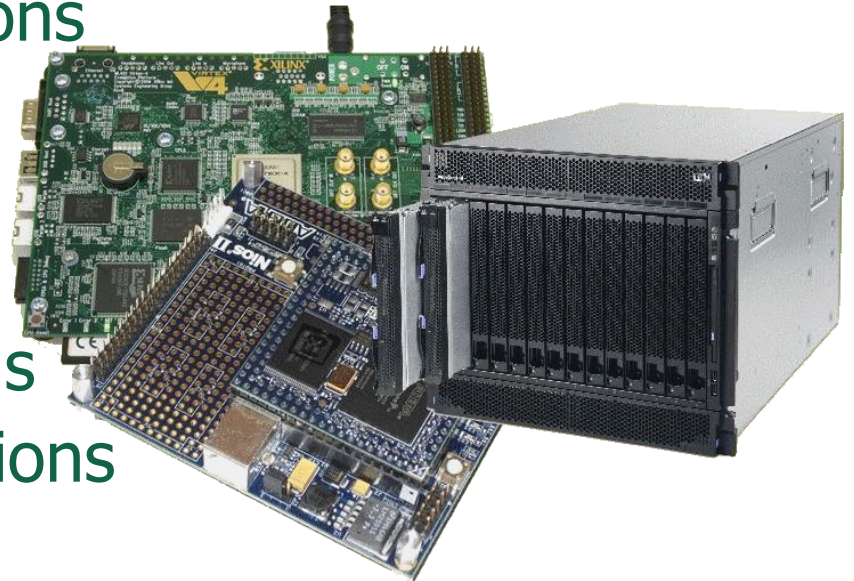
this lecture
PART 3/3

prev. lecture
PART 2/3

Lecture Overview

- Step 4: communication
 - Factors related to Communication
 - Cost of communications
 - Latency vs. Bandwidth
 - Baud rate vs. Bandwidth
 - Effective bandwidth
 - Visibility of Communications
 - Synchronous vs. asynchronous
 - Scope of comms
 - Collective communications
 - Efficiency of communications

A lot going on with communications!



Not examined

Brief mention of systems thinking ...

EEE4120F

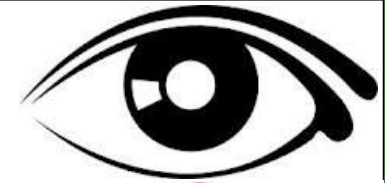
Reasoning for why this is mentioned:

Systems thinking gives a useful perspective and thinking approach for thinking about the interconnectedness of complex computer systems, not just how their constituent pieces communicate with one another, but also how the system being built is part of a larger system, workplace, environment or ecology. Thus it encourages not just a focus on the system being designed but the awareness of relevant scientific knowledge and potential impact or dependence that the system has on related or broader systems it fits within. (These slides should have been discussed prior to the GA assignment).

Different perspectives:

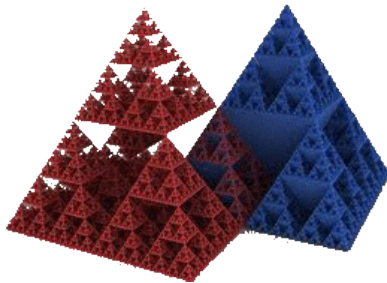
Gaining a better understand of the problem

Not examined



General Systems Thinking (GST)

- How does the problem of focus relate to more general problems?
- What other systems are being depended on or are affected?
- Has a systems approach been considered in addition to the usual divide and simplify approach?



Critical Analysis (& Critical Thinking)



*Applying rational and logical thinking while deconstructing texts or subject matter studied.**

- Let's consider what happens, in a logical scientific / deductive manner, if we discard or replace some of the traditions or 'conditioned choices'.
- Reflecting on choices and descriptions
- Considering 'what if...' scenarios (e.g. what if a chair had only one leg instead of four)



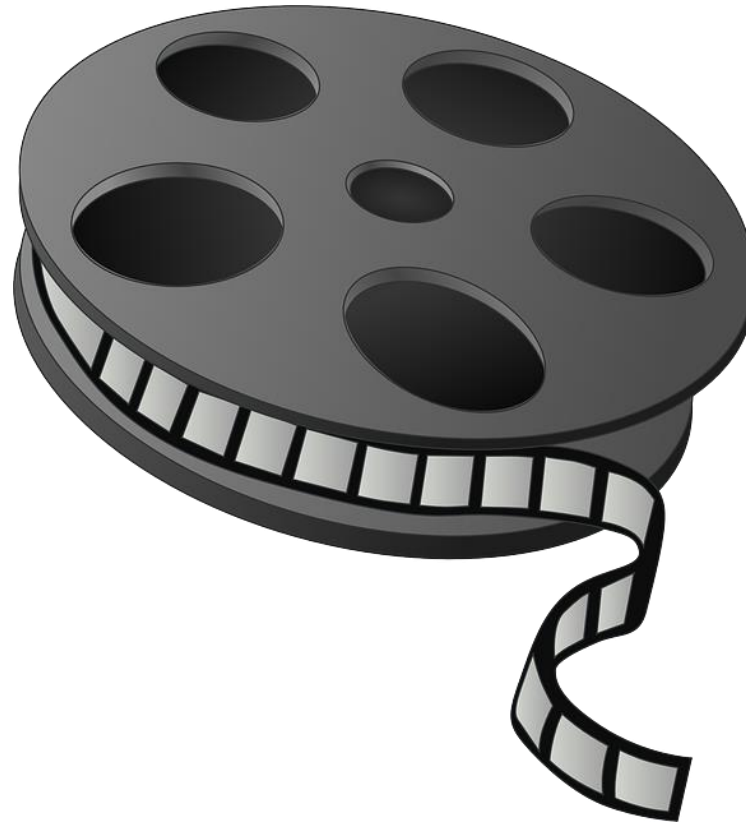
See last page for suggested sites to learn more about critical analysis

* Adapted from: Browne, M & Keeley, S 2001, *Asking the right questions: a guide to critical thinking*, 6th edn, Prentice-Hall, Upper Saddle River, N.J.

Not examined

Video Clip: Systems Thinking

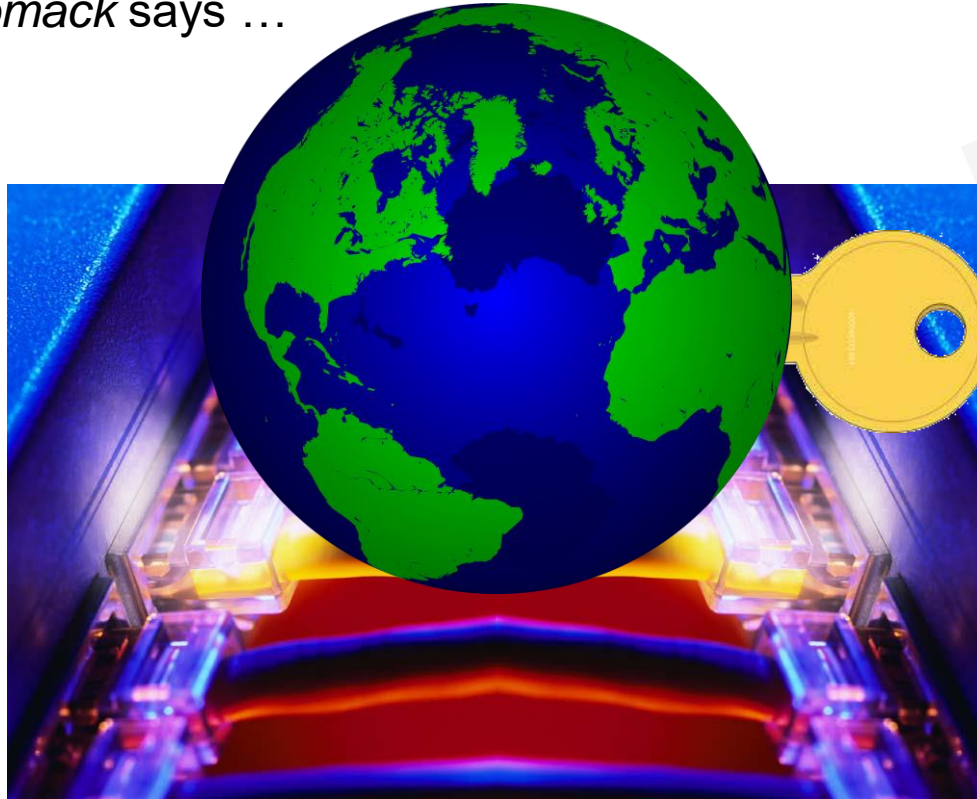
This is a suggested video clip to view for gaining an insight into what Systems Thinking is about.



YouTube clip: Systems thinking introduction

<https://youtu.be/Fo3ndxVOZEo>

Like *Bobby Womack* says ...



Step 4: Communication

EEE4120F



* Media clip source: Quality sounds of 'Communication' by Bobby Womack <https://youtu.be/jk-gP8oBihA>

Lets try to...

Get the

Communications Right

to get our

Systems Right

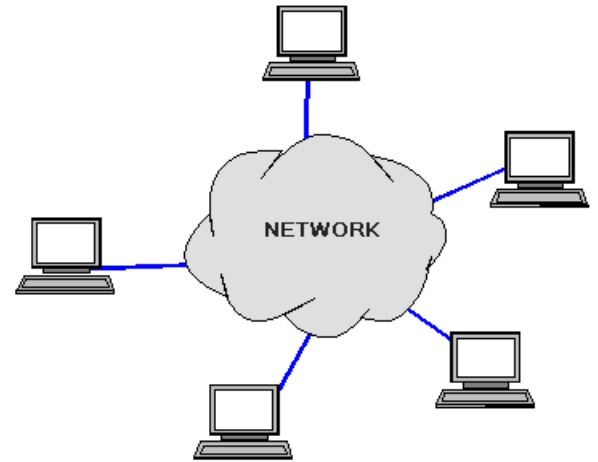
(PS: that's not necessarily a blast from the past!! You might still be making those noises but higher frequency ☺)

Step 4: Communications

- The **communications needs** between tasks **depends on your solution**:
- Communications **not** needed for
 - Minimal or no shared data or results
 - E.g., an image processing routine where each pixel is dimmed (e.g., 50% dim). Here, the image can easily be separated between many tasks that act entirely independently of one other.
 - Usually the case for embarrassingly parallel solutions⁴

Communications

- The communications needs between tasks depends on your solution:
- Communications *is* needed for...
 - Parallel applications that need to share results or boundary information. E.g.,
 - E.g., modeling 2D heat diffusion over time – this could divide into multiple parts, but boundary results need to be shared. Changes to an elements in the middle of the partition only has an effect on the boundary after some time.



Factors related to Communication

- Cost of communications
- Latency vs. Bandwidth
- Visibility of communications
- Synchronous vs. asynchronous communications
- Scope of communications
- Efficiency of communications
- Overhead and Complexity

Cost of communications

- Communication between tasks has some kind of overheads, such as:
 - CPU cycles, memory and other resources that could be used for computation are instead used to package and transmit data.
 - Also needs synchronization between tasks, which may result in tasks spending time waiting instead of working.
 - Competing communication traffic could also saturate the network bandwidth, causing performance loss.

Latency vs. Bandwidth

- Communication Latency =
 - Time it takes to send a minimal length (e.g., 0 byte) message from one task to another. Usually expressed as microseconds.
- Bandwidth =
 - The amount of data that can be sent per unit of time. Usually expressed as bits per second (bps) or megabits/sec, and sometimes for convenience as megabytes/sec or gigabytes/sec.



Latency vs. Bandwidth

- Many **small messages** can result in latency dominating communication overheads.
- If many small messages are needed:
 - It can be more efficient to package small messages into a larger ones, to increasing the **effective bandwidth*** of communications.

* Explained soon!

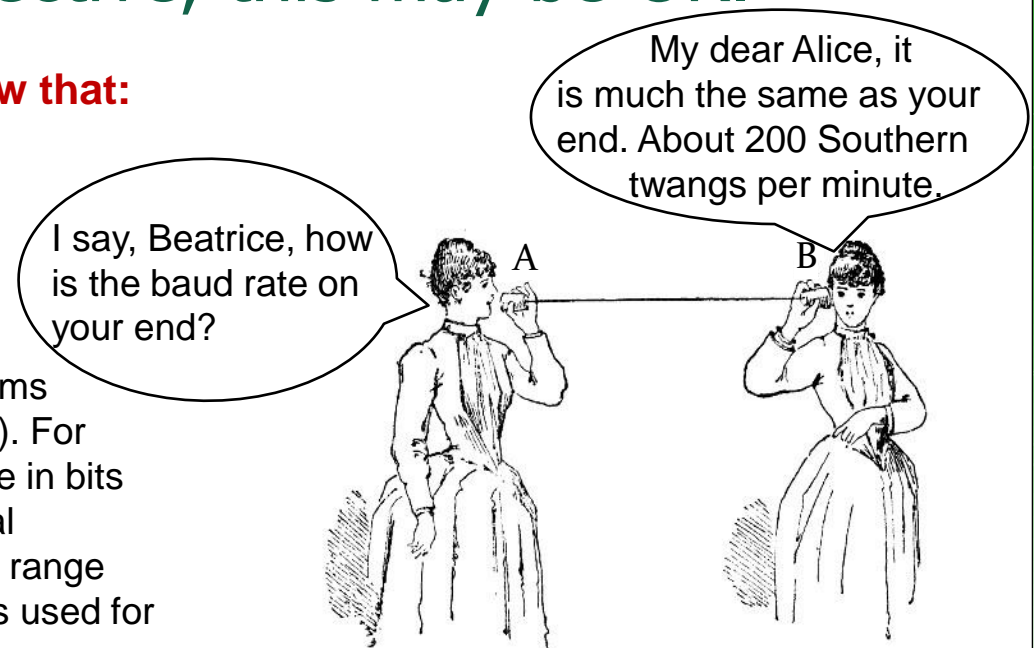
Baud rate vs. Bandwidth

- Avoid misuse of these terms by using them interchangeably
- The general public seem to think these refer to the same thing, and in some cases, from a user perspective, this may be OK.

Nevertheless an engineer should know that:

baud specifies the symbol rate (or signal changes) per unit of time, and this could be an entirely analogue measure.

Bandwidth is different depending whether you are considering computer networks/telecoms or signals more generally (e.g. in a RF system). For computing Bandwidth is a measure of data rate in bits per second. But more generally in the electrical engineering Bandwidth refers to the frequency range within a particular band (typically a band that is used for transmitting some sort of signal).



Baudrate – an issue of standards

(from a computing/telecoms perspective)

- Baudrate is not always interpreted the same way for all applications when it comes to deciding the amount of bytes that the link provides.
- This is due to other aspects of the protocol used, e.g. if each byte is surrounded by error detection bits.

Baudrate – an issue of standards

(from a computing/telecoms perspective)

- Remember:
 - A byte is always understood as 8 bits – but the byte is just the 'useful data' that is being transferred.
- If the connection is specified as a speed in bits per second, then it is 8 bits per byte.
 - When you specify the speed in baud rate, you're specifying symbol rate (see previous slide) in which case you might be using 10 symbols to a byte when using start-data-stop or 8b/10b encoding schemes.
 - So, in other words you may quite likely find yourself in a situation where you have a bandwidth of say 1Mbps but a baud rate of 1.25 mega symbols per second.

Baudrate – an issue of standards

(from a computing/telecoms perspective)

- There is even an official SI unit for “baud”
- It is written as “Bd” units for specifying symbol rate
- This should not be confused with the more commonly known bps (bits per second) or Bps (bytes per second).
- *So you should now know:*
x MBd is not always = x Mbps
- Example:
 - If you have an 8b/10b encoded line that has a bandwidth of 10 MBd, you can automatically say that it has a bandwidth of 8 Mb/s, or 1 MB/s (excluding other overhead such as packet headers, of course).



Suggested further reading on baud rate and bandwidth



Supplementary reading

If you're one of those super keen students, you might like to look at these nuggets of knowledge and web pages for further reading on this topic:

- PCIe 1.0 and 2.0 (including Ethernet and USB) uses **8b/10b encoding**. PCIe 3.0 and 4.0 uses **128b/130b encoding**. Read more at: http://en.wikipedia.org/wiki/PCI_Express#History_and_revisions
- Historically, **Hamming code** refers to the Hamming (7, 4) code, which uses 7 symbols for every 4 data bits. There are many other variations on this which you can explore at http://en.wikipedia.org/wiki/Hamming_code
- DVD encoding uses **Reed-Solomon error correction** in two stages (note, in case you were wondering, this encoding has nothing to do with grass or mines) read more at http://en.wikipedia.org/wiki/Reed%E2%80%93Solomon_error_correction#Data_storage.
- There are even more interesting cases where **run-length limited codes** are used with error correction and (for argument sake) 16 QAM signal encoding. 1 MBd 16 QAM has a bandwidth of 4 Mb/s (rather interesting case because the baud rate here is a quarter of the bit-rate! You also get 32, 64, 128 and 256 QAM systems....)

(these readings are not examined)



Effective

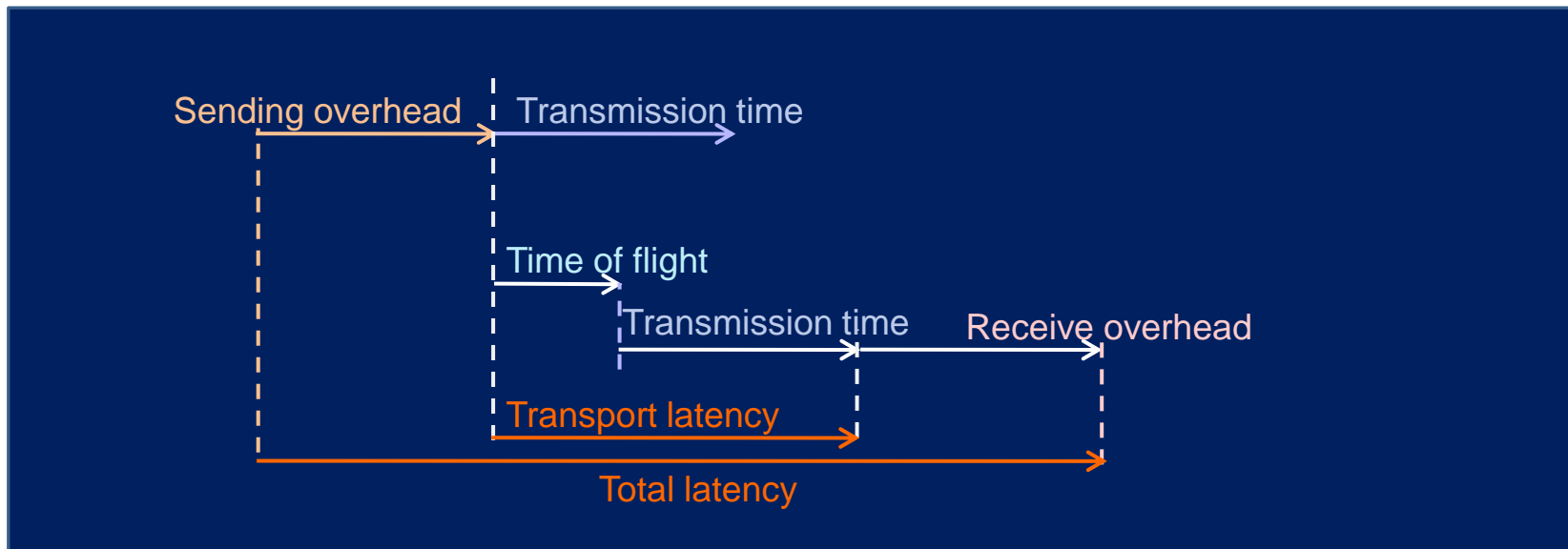
Bandwidth

Effective bandwidth

PS: a favourite thing to ask in tests and exams

Total latency = Sending overhead + Transmission time + time of flight + Receiver overhead

Effective bandwidth = Message size / total latency



time of flight is also referred to as 'propagation delay' – it may depend on how many channels are used. E.g. a two-channel path will give an effective lower propagation. With switching circuitry, the propagation delay can increase significantly.

Effective bandwidth calc.

CLASS
ACTIVITY 1!

Sample Question...

Example:

You are wanting to connect two computers together using a copper wire.

The distance between them is: 100m

Raw bandwidth (limited by comms devices) of the channel is: 10Mbit/s

Message to send is: 10,000 **bytes**

Sending device overhead: 200us

Receiving device overhead: 300us

TODO: Calculate the Effective Bandwidth of this connection.

Get a copy of the handout. Can work in teams.

Effective bandwidth calc.

CLASS
ACTIVITY 1
SOLUTION!

Example:

Distance 100m

Raw bandwidth 10Mbit/s

Message 10,000 bytes

Sending overhead 200us

Receiving overhead 300us

TODO: Calculate the Effective Bandwidth of this connection.

Solution: ...

$$\text{Transmission time} = \frac{80,000 \text{ bits}}{10 \text{ Mbits/s}} = \frac{80,000 \text{ bits}}{10 \text{ bits}/\mu\text{s}} = 8,000 \mu\text{s}$$

$$\text{Time of flight} = \frac{100 \text{ m}}{3 \times 10^8 \text{ m/s}} = \frac{1 \text{ m}}{3 \times 10^6 \text{ m/s}} = 0.33 \mu\text{s}$$

Total latency = Sending overhead + Transmission time +
time of flight + Receiver overhead

$$\text{Total latency} = 200\mu\text{s} + 8,000\mu\text{s} + 0.33\mu\text{s} + 300\mu\text{s} = 8,500.33\mu\text{s}$$

Effective bandwidth = Message size / total latency

$$\text{Effective bandwidth} = 80,000 \text{ bits} / 8,500\mu\text{s} = 9.41 \text{ Mbits/s} \quad 94\% \text{ efficient}$$

Visibility of Communications



- Communications is usually both explicit and highly visible when using the **message passing** (MP) programming model.
- Communications may have poor visibility when using the data parallel programming model (e.g. shared memory).
- For data parallel design on a distributed system, communications may be entirely invisible, in that the programmer may have no understanding (and no easily obtainable means) to accurately determine what inter-task communications is happening.

Synchronous vs. asynchronous

- Synchronous communications

- Require some kind of handshaking between tasks that share data / results.



- May be explicitly structured in the code, under control of the programmer – or it may happen at a lower level, not under control of the programmer.

- Synchronous communications are also referred to as blocking communications because other work must wait until the communications has finished.

Synchronous vs. asynchronous

○ Asynchronous communications



- Allow tasks to transfer data between one another independently. E.g.: task A sends a message to task B, and task A immediately begins continues with other work. The point when task B actually receives, and starts working on, the sent data doesn't matter.
- Asynchronous communications are often referred to as non-blocking communications.
- Allows for interleaving of computation and communication, potentially providing less overhead compared to the synchronous case

Scope of communications

- Scope of communications:
 - Knowing which tasks must communicate with each other
- Can be crucial to an effective design of a parallel program.
- Two general types of scope:
 - Point-to-point (P2P)
 - Collective / broadcasting

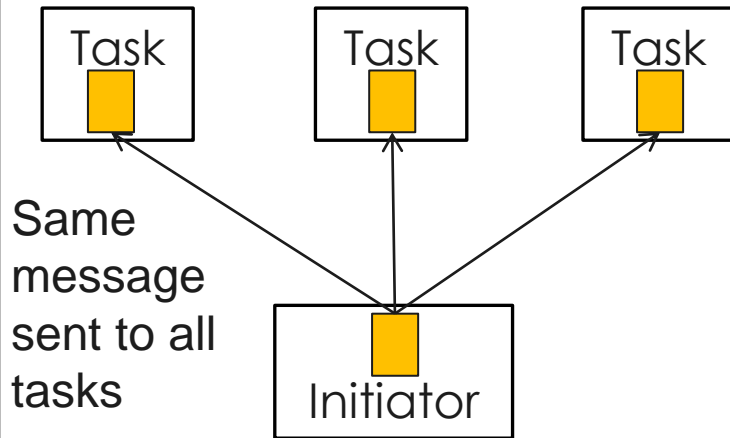
Scope of communications

- Point-to-point (P2P)
 - Involves only two tasks, one task is the sender/producer of data, and the other acting as the receiver/consumer.
- Collective
 - Data sharing between more than two tasks (sometimes specified as a common group or collective).
 - Both P2P and collective communications can be synchronous or asynchronous.

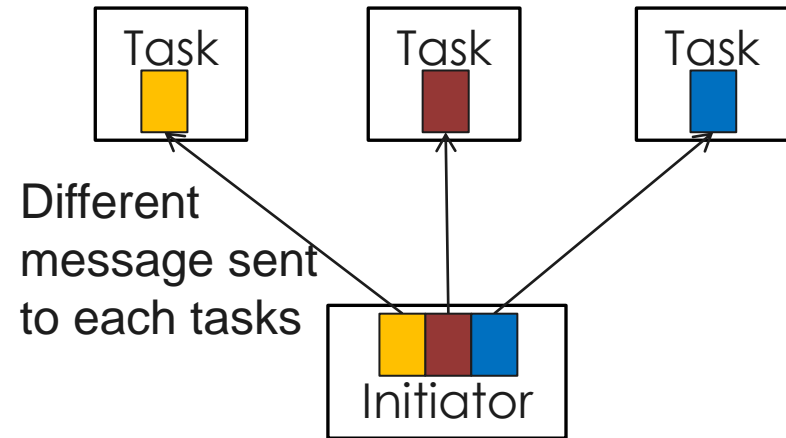
Collective communications

Typical techniques used for collective communications:

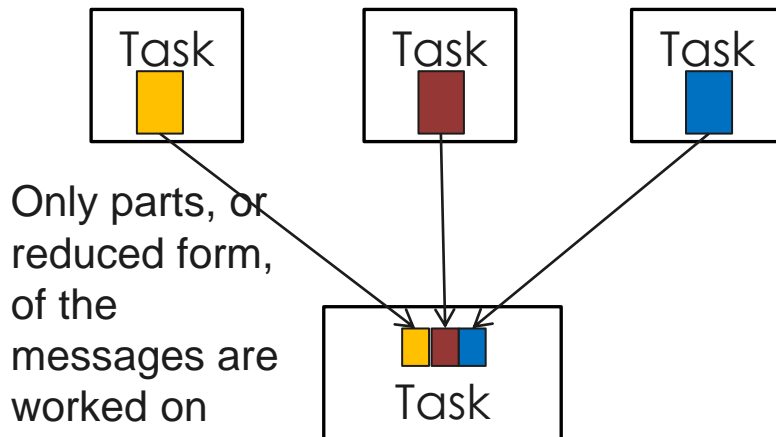
BROADCAST



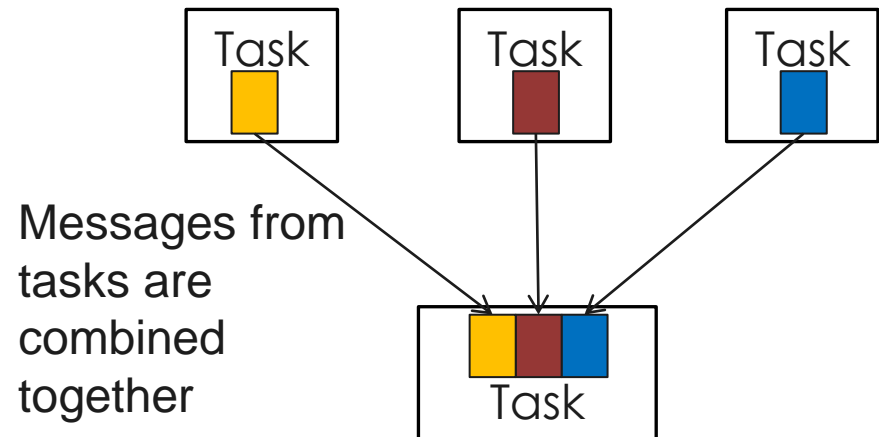
SCATTER



REDUCING



GATHER



Efficiency of communications



- There may be a choice of different communication techniques
 - In terms of hardware (e.g., fiberoptics, wireless, bus system), and
 - In terms of software / protocol used
- Programmer may need to use a combination of techniques and technology to establish the most efficient choice (in terms of speed, power, size, etc).

Intermission

Onwards to distributed and shared memory architecture models ...

Disclaimers and copyright/licensing details

I have tried to follow the correct practices concerning copyright and licensing of material, particularly image sources that have been used in this presentation. I have put much effort into trying to make this material open access so that it can be of benefit to others in their teaching and learning practice. Any mistakes or omissions with regards to these issues I will correct when notified. To the best of my understanding the material in these slides can be shared according to the Creative Commons “Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)” license, and that is why I selected that license to apply to this presentation (it’s not because I particulate want my slides referenced but more to acknowledge the sources and generosity of others who have provided free material such as the images I have used).

Image sources:

Clipart sources – public domain CC0 (<http://pixabay.com/>)

PxFuel – CC0 (<https://www.pxfuel.com/>)

Pixabay

commons.wikimedia.org

Images from flickr

