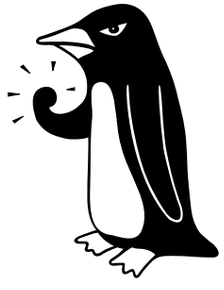
Introduction and Analogies



Warwick RSE

17/03/2020

Why analogies?

- Parallel computation feels like something that is very outside of your normal experience

- Actually a lot of the general things that you have to understand make intuitive sense

- Most of the problems, and solutions, to running things on multiple processors are the same as trying to do multiple things in real life

Limits to analogies

- Be very careful with analogies!

- They can be extremely valuable and quite often a feature that you can identify in the analogy is also a feature in the real problem

- But there are definitely cases where the feature that you identify is an **analogy breakdown** and the feature is not present in the real problem

- If you find these analogies useful then you might want to try to extend them yourselves, but always check to see if the extension apply to the real problem

Where to start?

- The simplest thing that you can think of when it comes to doing things at the same time is eating in a buffet restaurant
- Everyone does what they want independently
- People only interfere with each other if they want to use the same resource at the same time

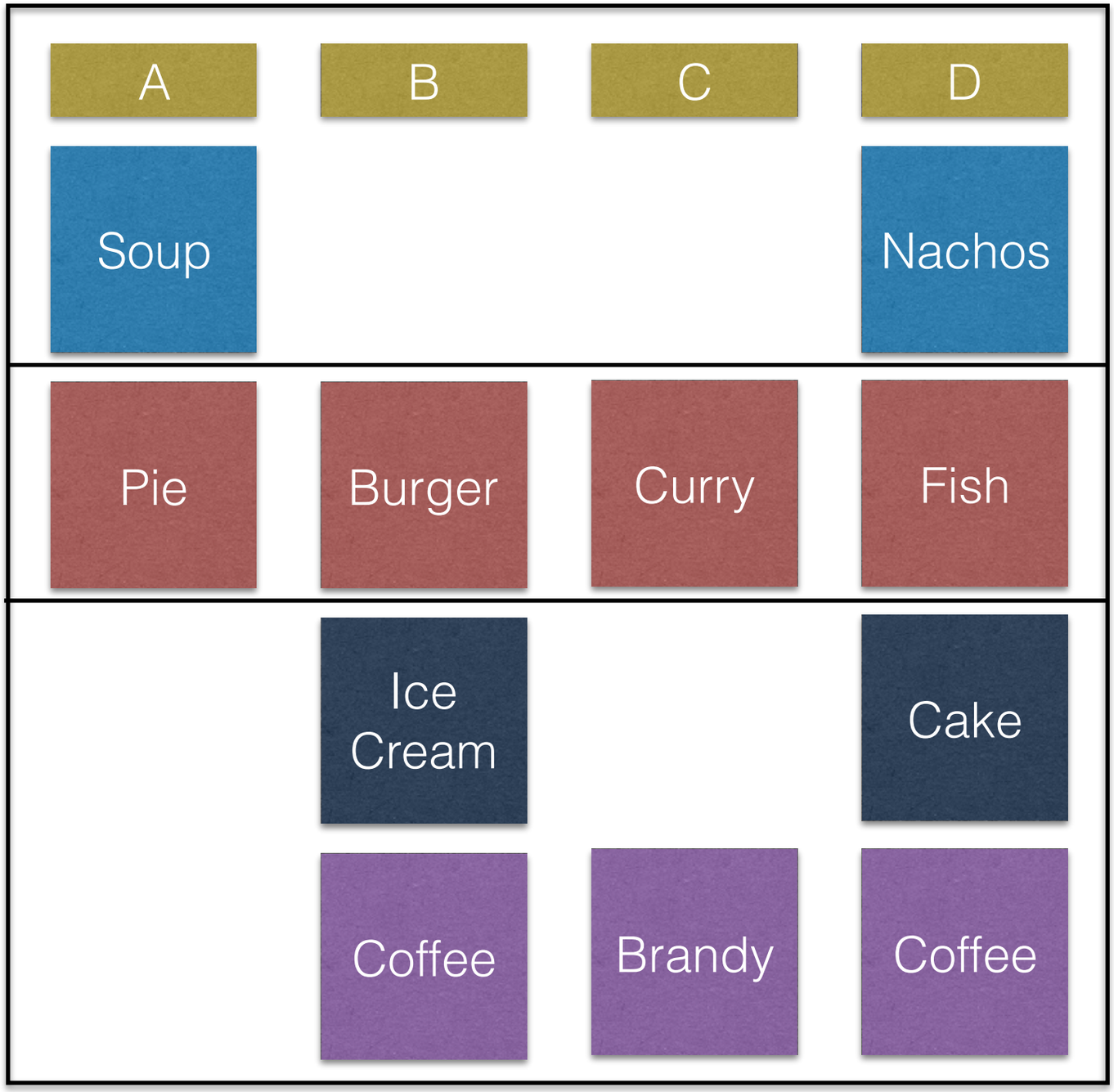
Buffet

A	B	C	D
Duck		Chilli	Pudding
Curry	Sushi	Pudding	Pudding
Pause for Drinks	Duck		Pudding
Pudding	Liqueurs		Curry
			Curry
			Pudding

Computer Analogy

- Normal operations of the computer
  - Programs are split onto different processors so that they are all used
  - Running two programs at once will normally put one on each processor
- “Embarrassing Parallelism”
  - Solve many problems by running several programs or many copies of one program
  - Don’t want more programs than you have processors or some have to wait their turn, just like at a buffet

A La Carte


A
B
C
D
Soup
Nachos
Pie
Burger
Curry
Fish
Ice Cream
Cake
Coffee
Brandy
Coffee

Computer Analogy

- Weakly coupled parallel computer programs
    - Mostly things do their own thing with little reference to other things that are doing their work
- You occasionally have to synchronise information between running things to make things happen at the same time
    - Or to make things have a specific sequence

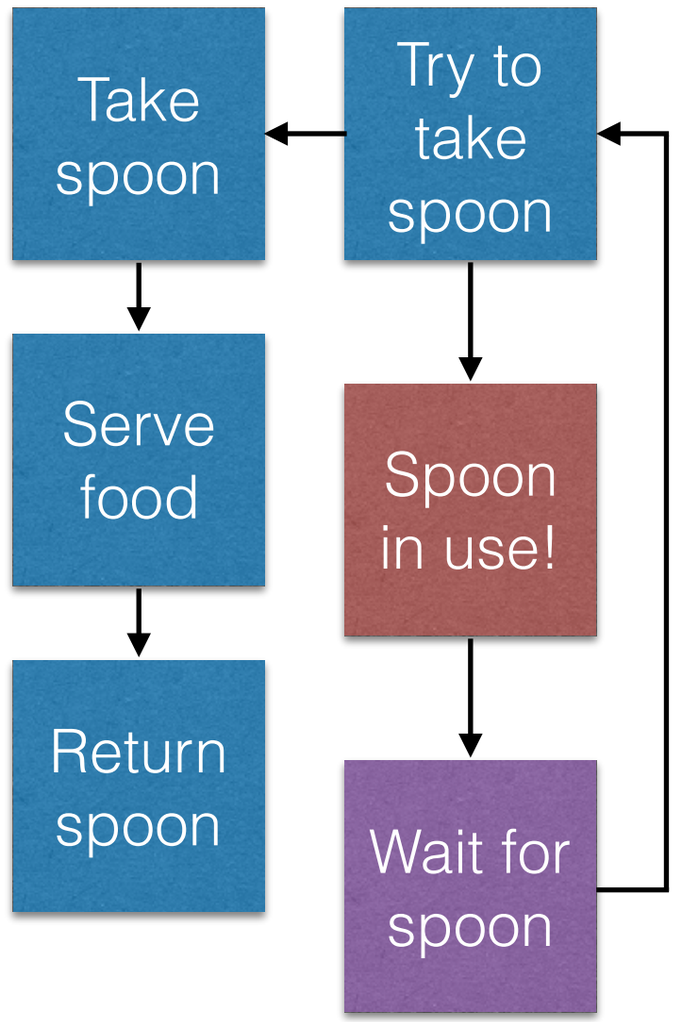
Prix Fixe

A	B	C	D
Starter	Starter	Starter	Starter
Main Course	Main Course	Main Course	Main Course
Dessert	Dessert	Dessert	Dessert
Coffee	Coffee	Coffee	Coffee

Computer Analogy

- Tightly coupled parallel processing
- Many sequencing points where things don't advance to the next level until everyone has reached the same point
- Things may be doing different tasks but you have to treat them as if they are all the same because you can't move on until all of them are finished

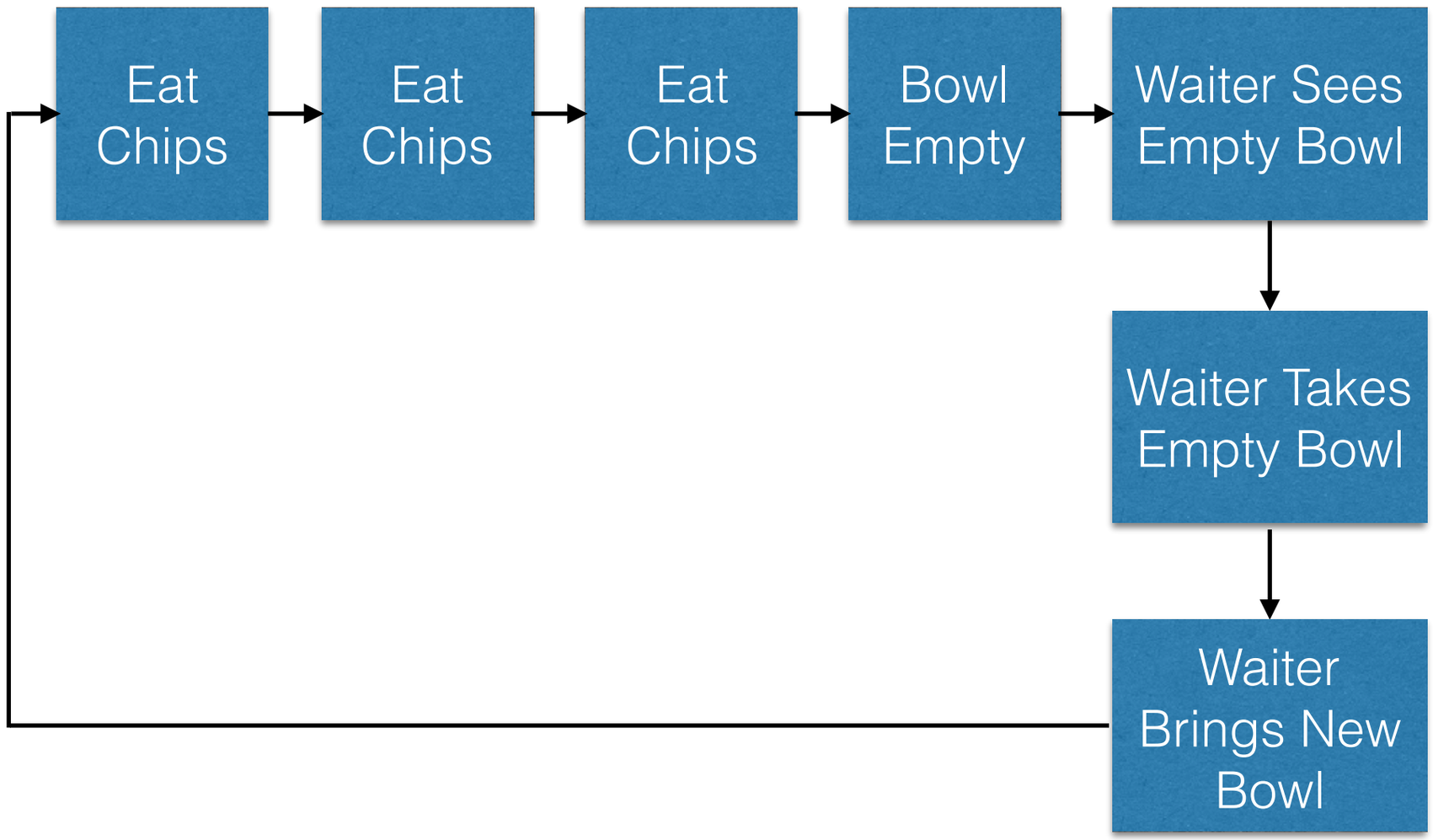
The Buffet Spoon



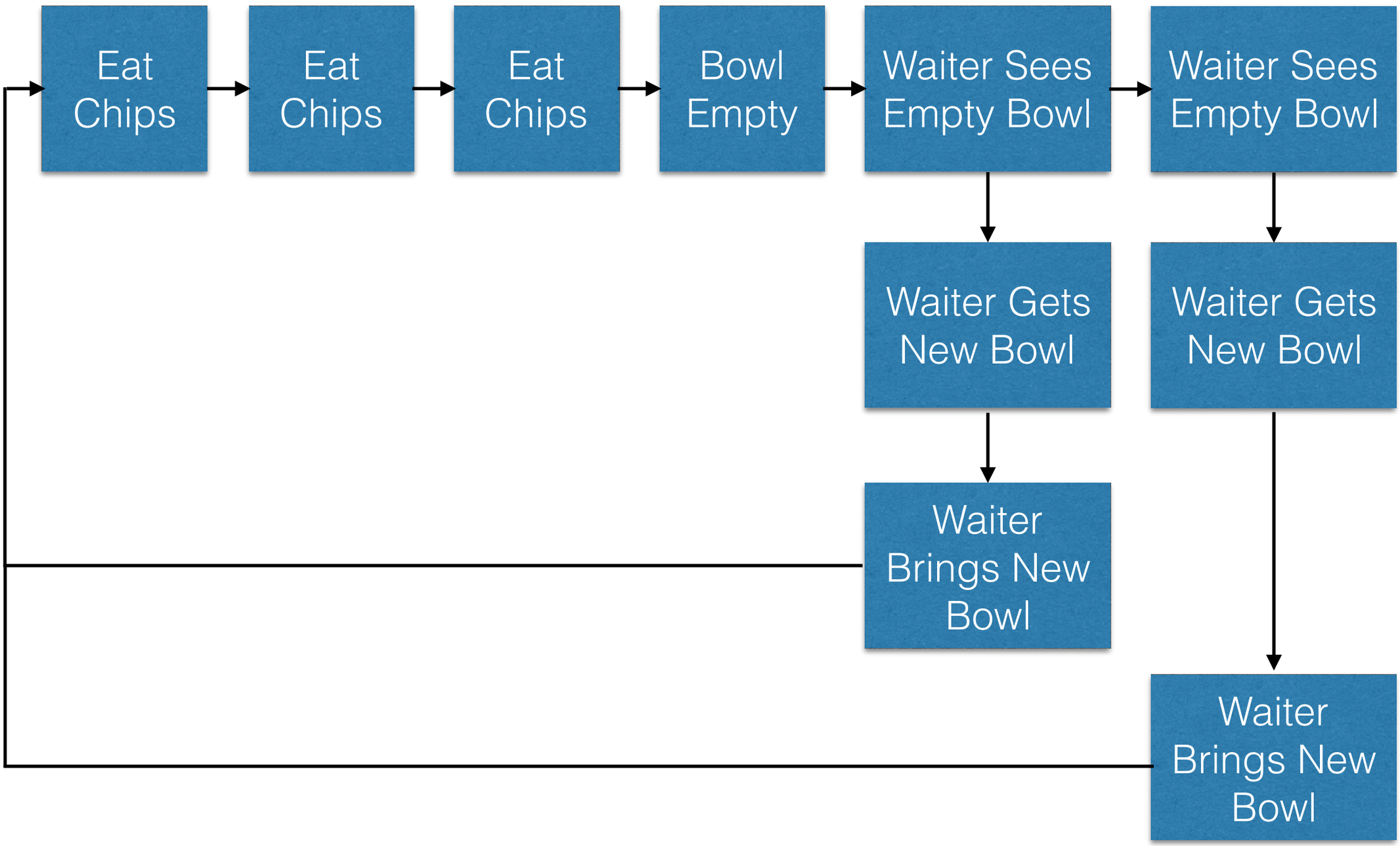
Computer Analogy

- Lots of things!
  - This is how almost all resources that can only be used by one thing at a time work in computers
  - In particular you can deliberately make it happen by using **Mutual Exclusion** objects. Things that only one processor can have at a time. More on this later
  - There's actually quite a lot of work devoted to making sure that everything gets its turn with the spoon (actual queues are unusual in computers)

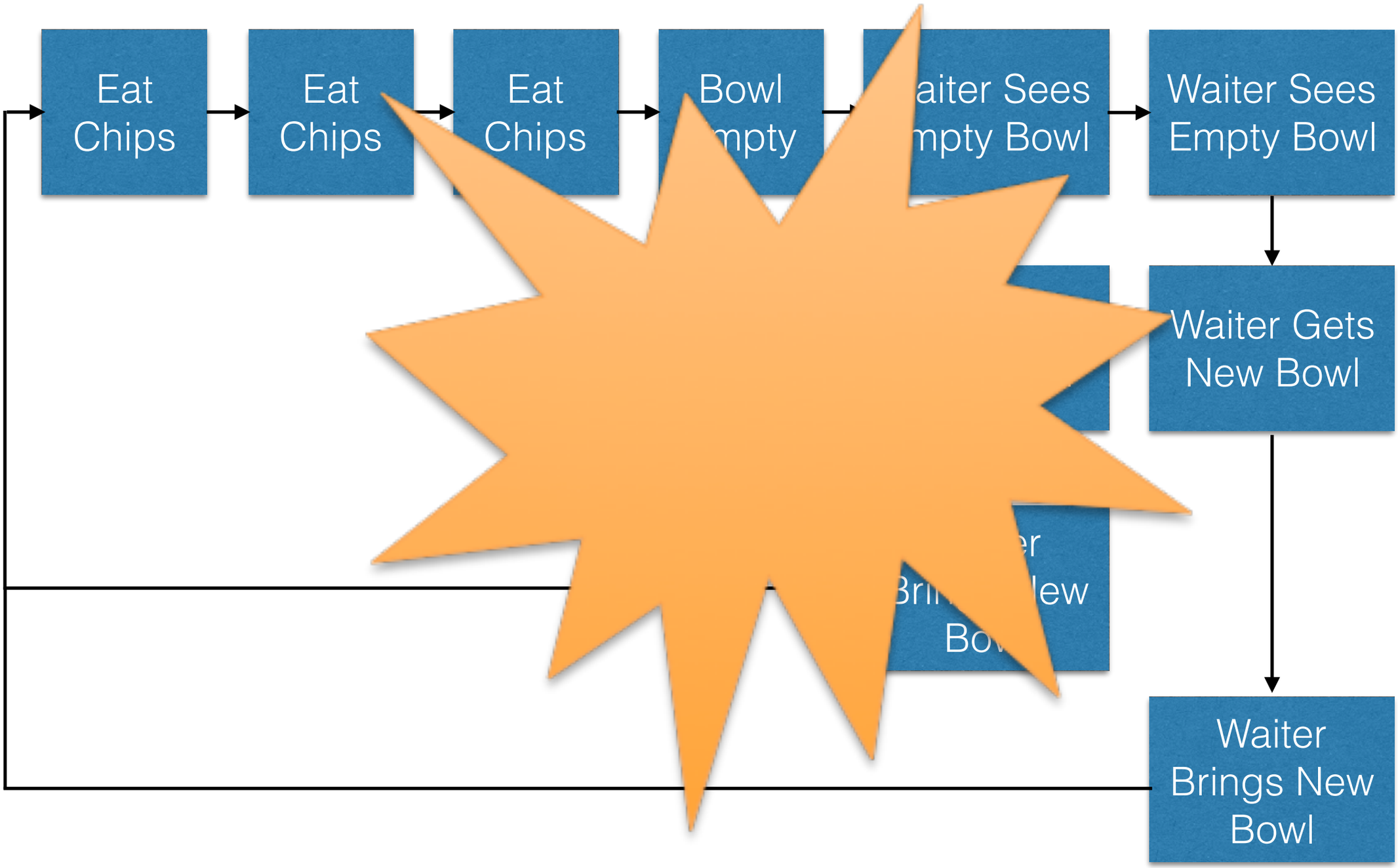
The Bowl ‘o Chips



The Bowl ‘o Chips



Chipslosion!



Analogy

- This is an example of something called a race condition
- If a waiter sees you have an empty bowl he will go and get you a new one
- If two waiters both see the empty bowl then you wind up with two bowls of chips
- If the first waiter gets back before the second waiter sees the empty bowl then you only get one bowl
    - Hence race condition. The result depends on how fast your waiter is
- If the first waiter to see you have an empty bowl takes it away then everything will work (no bowl isn't an empty bowl)

Computer Analogy

- Exactly the same happens with computers
- If two threads are waiting for a signal then things can happen twice if the first one doesn't signal that it has handled it before the second one sees the signal
- How you actually do the signalling involves things called atomic (Greek - Indivisible) operations
  - We'll see these briefly later
- There are other race conditions e.g. two threads setting a variable to a given value - the value will depend on which processor happens to get there last
  - They are always a bad thing

Computers

Computer Terms

- Probably a lot of this is familiar but it is important that you understand some important terms before we start

- Core - the smallest unit capable of independent general purpose computation. You can get 64 cores in a single chip nowadays

- CPU (or chip) - Central Processing Unit. Physical computer chip that will (nowadays) usually contain multiple cores

Computer Terms

- Socket - Used to refer to the number of physical chips in a computer (“a 2 socket workstation”)
- Node - A single complete computer but usually one that is part of a **cluster**
- Cluster - A collection of computers connected together for parallel processing

Computer Terms

- RAM - Random Access Memory. The part of the computer that stores the data that you are working with

- Hard Drive/Disk - Long term stable storage medium that persists data after the computer is switched off.

Computer Terms

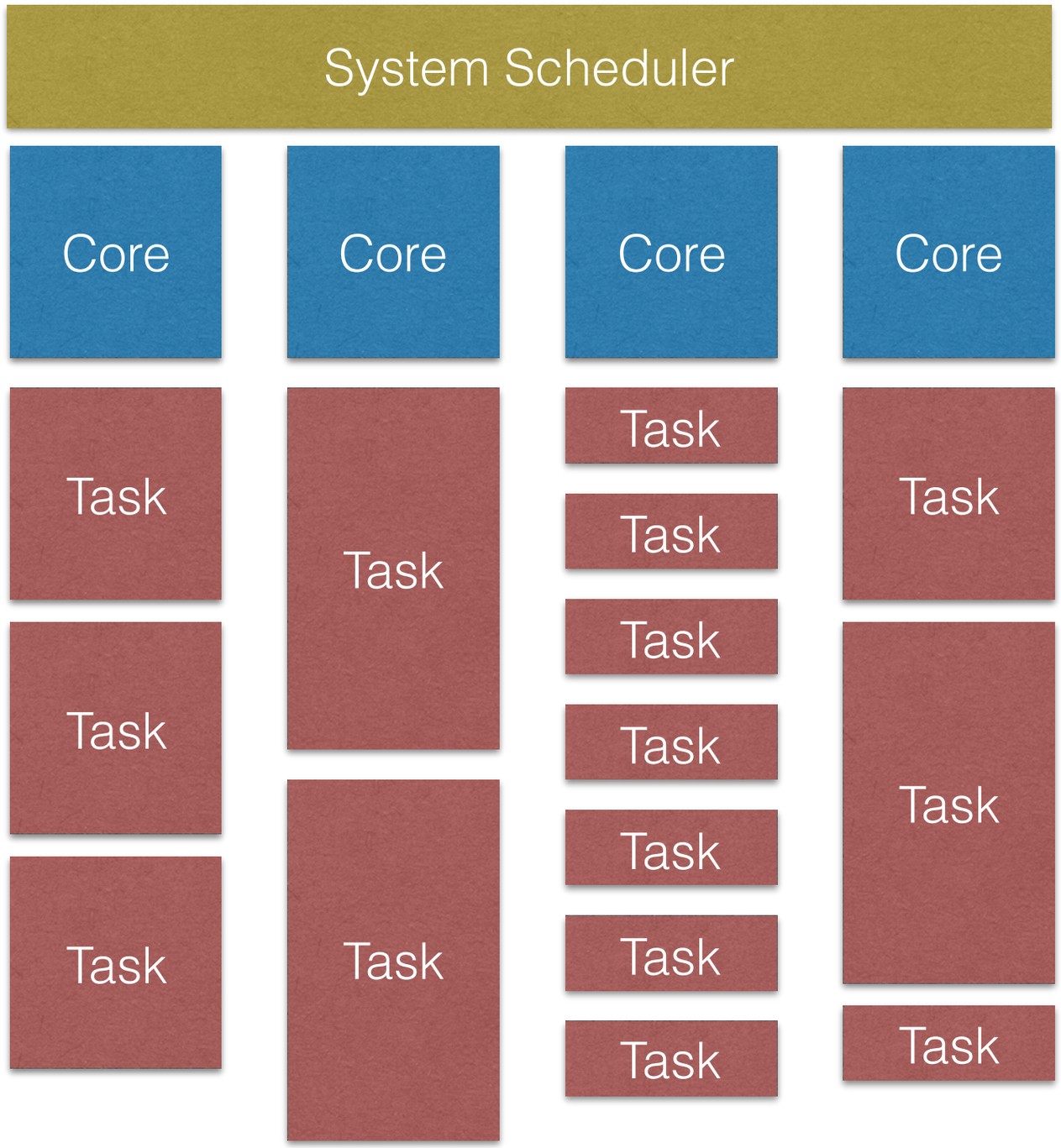
- Process - Technically the name for a running program. Processes do not interact directly with each other (normally, there are mechanisms to do it) and will generally run on separate cores to each other if possible

- Thread - A part of a process created to run independently. Threads are connected to a single process and a given process can have many threads.

CPUs

- Mostly we're going to be talking about cores here because they are what actually does the parallel processing
  - The idea is to move from using one core to many cores
  - Sometimes very many
- CPU behaviour in a computer doing many things is very like that buffet example

4 Core CPU

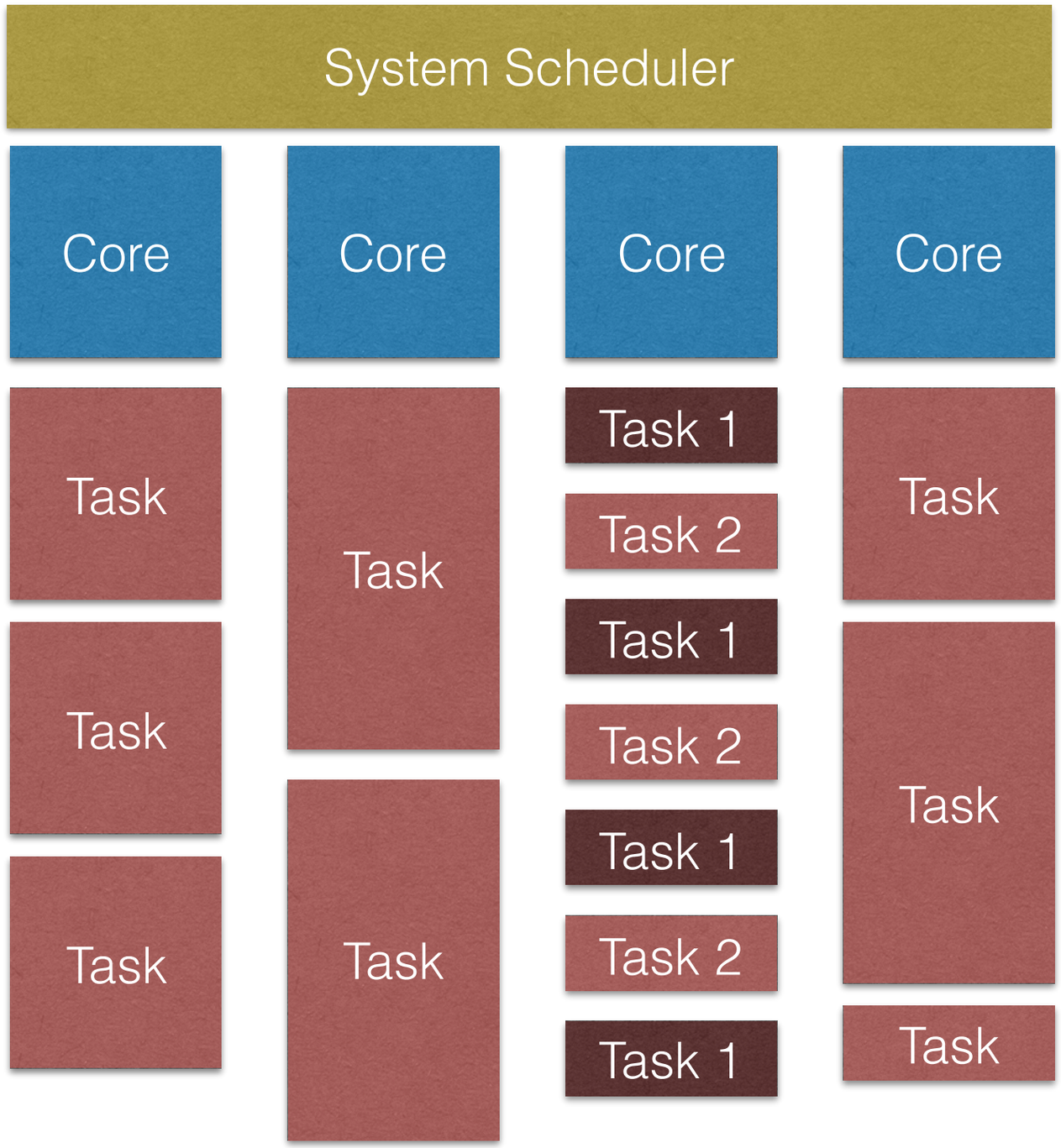


A possible, heavily simplified schedule

CPUs

- We mentioned that in the buffet people have to wait if they both want to get the same food
- Computers don’t quite work like this (analogy failure)
  - If they did then as soon as you had one program running per processor in your computer you would have to wait for one of them to finish before you could do anything else
- Timeslicing - suspend one program to allow another to run

CPU Multitasking



A possible, heavily simplified schedule with one more task than available CPUs

CPUs

- Time slicing is brilliant for problems like web servers.
    - The thread that is serving a website to any one person is mostly not doing much (waiting for the person to load new data)
        - So having multiple threads being interleaved works quite well even when you have more threads than you do cores
- This isn't generally true for academic codes where all threads are working flat out at all times

Hyperthreading

- Technically “simultaneous multithreading” (SMT)
  - Hyperthreading is Intel’s proprietary name
- Remember that a core is “the smallest unit capable of independent general purpose computation”
- Since cores can do different types of processing this means that some bits will always be idle since you aren’t doing that type **right now**
- SMT means that if another thread wants those unused bits it can use them as well