Big Questions In Computation

Samson Abramsky

Oxford University Department of Computer Science

Big Questions

The nature of 'Big Questions':
Tension between

- Big
- Well-defined, and able to be addressed

Big Questions

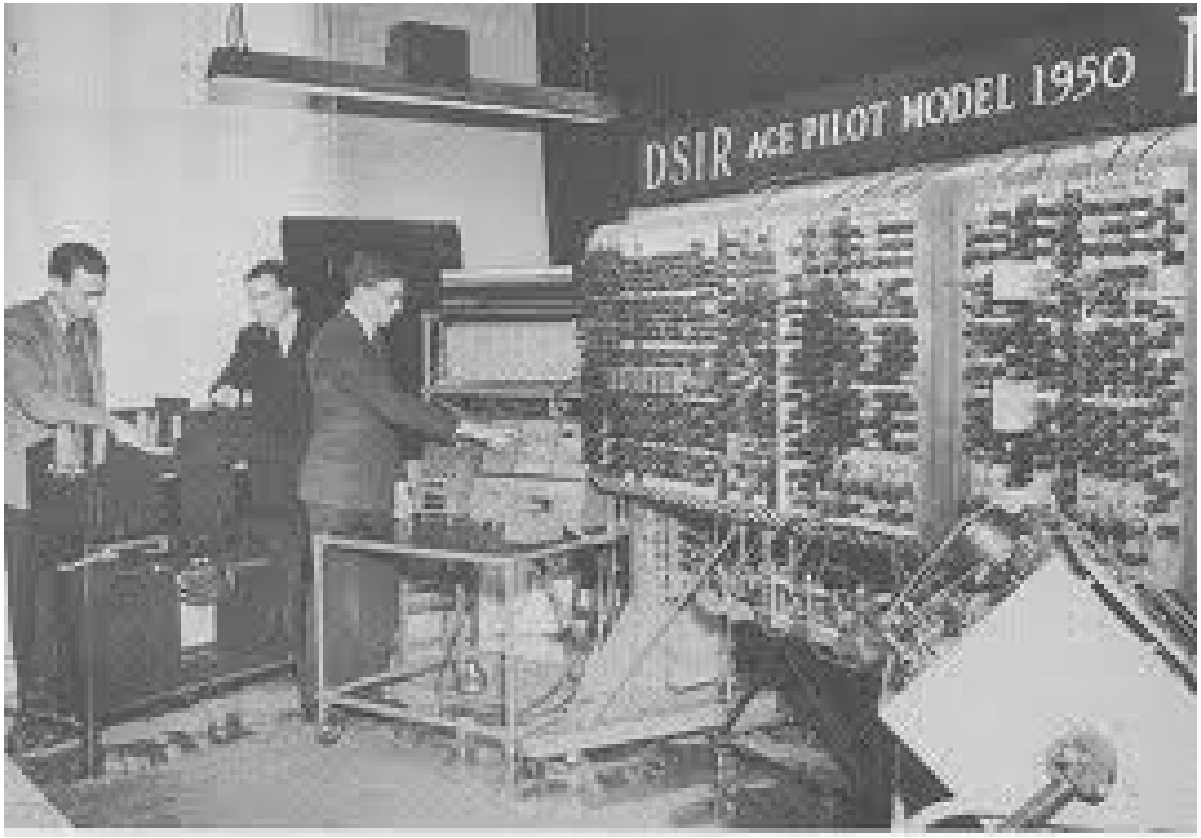
The nature of 'Big Questions':
Tension between

- Big
- Well-defined, and able to be addressed

I will suggest two, based on the following notions:

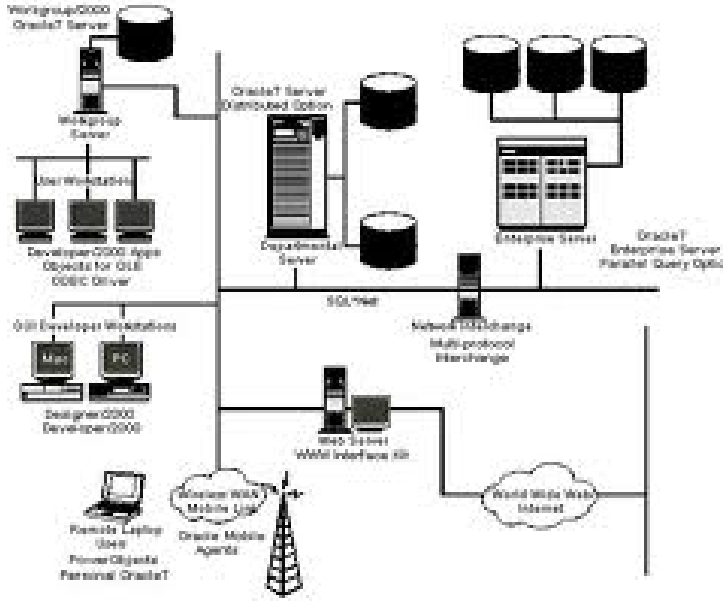
- Processes
- Contexts

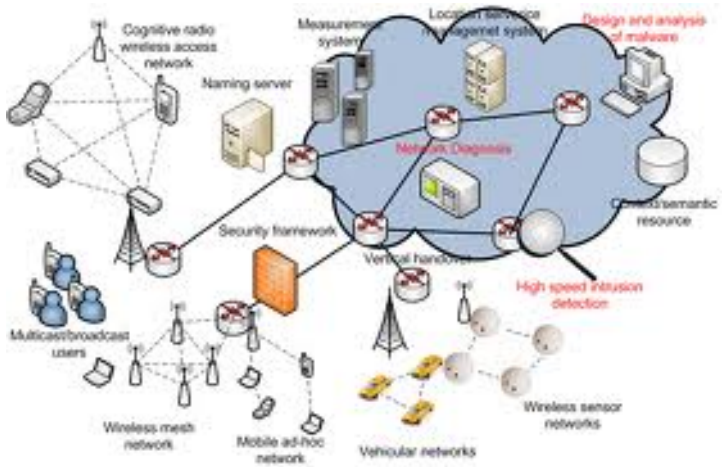
Computation: how it was



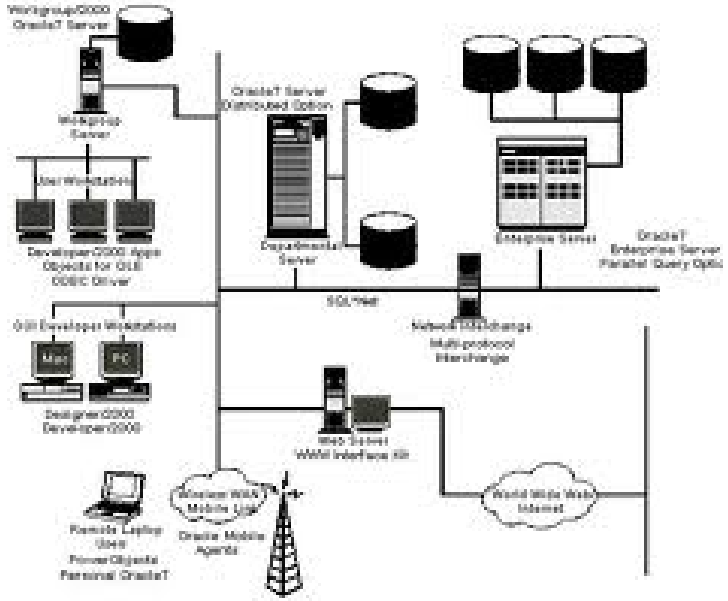
Computing in the isolation ward

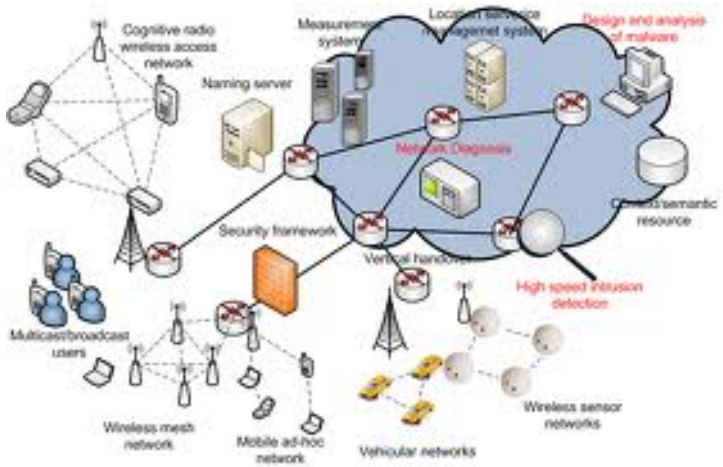
The changing face of computation

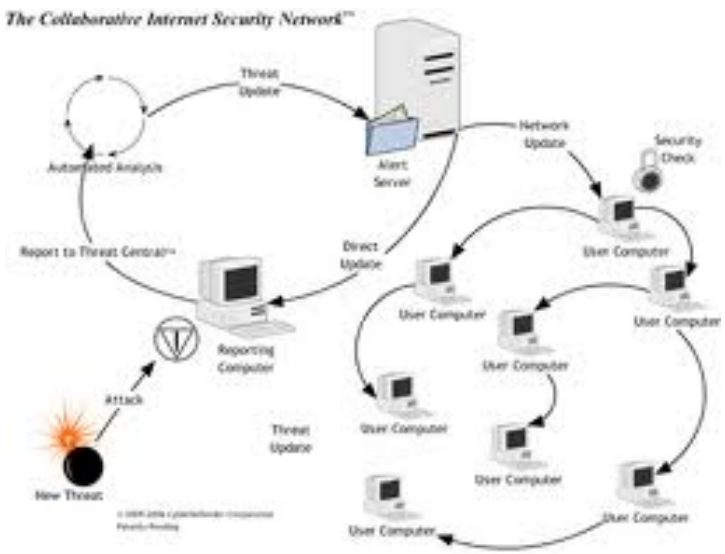


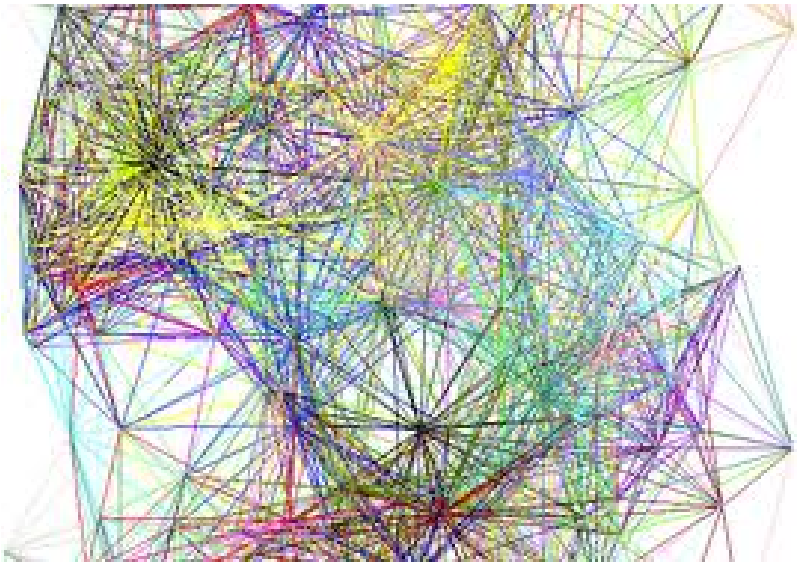


The changing face of computation









Processes, and why they matter in Computer Science

Processes, and why they matter in Computer Science

Computer Science asks a broader question:

What is a process?

Processes, and why they matter in Computer Science

Computer Science asks a broader question:

What is a process?

The purpose of much of the software we routinely run nowadays is not to compute a function, but to **exhibit some behaviour**.

Processes, and why they matter in Computer Science

Computer Science asks a broader question:

What is a process?

The purpose of much of the software we routinely run nowadays is not to compute a function, but to **exhibit some behaviour**.

What function does the Internet compute?

Processes, and why they matter in Computer Science

Computer Science asks a broader question:

What is a process?

The purpose of much of the software we routinely run nowadays is not to compute a function, but to **exhibit some behaviour**.

What function does the Internet compute?

Think of communication protocols, operating systems, browsers, iTunes, Facebook, Twitter, ...

Processes, and why they matter in Computer Science

Computer Science asks a broader question:

What is a process?

The purpose of much of the software we routinely run nowadays is not to compute a function, but to **exhibit some behaviour**.

What function does the Internet compute?

Think of communication protocols, operating systems, browsers, iTunes, Facebook, Twitter, ...

What is a process? When are two processes equivalent?

Processes, and why they matter in Computer Science

Computer Science asks a broader question:

What is a process?

The purpose of much of the software we routinely run nowadays is not to compute a function, but to **exhibit some behaviour**.

What function does the Internet compute?

Think of communication protocols, operating systems, browsers, iTunes, Facebook, Twitter, ...

What is a process? When are two processes equivalent?

The situation is very different to that which we find in computability theory.

Processes, and why they matter in Computer Science

Computer Science asks a broader question:

What is a process?

The purpose of much of the software we routinely run nowadays is not to compute a function, but to **exhibit some behaviour**.

What function does the Internet compute?

Think of communication protocols, operating systems, browsers, iTunes, Facebook, Twitter, ...

What is a process? When are two processes equivalent?

The situation is very different to that which we find in computability theory.

No established mathematical theory of what processes are.

The non-confluence of notions

The non-confluence of notions

In computability theory we have:

The non-confluence of notions

In computability theory we have:

- A confluence of notions

The non-confluence of notions

In computability theory we have:

- A confluence of notions
- A definitive calculus of functions: the λ-calculus.

The non-confluence of notions

In computability theory we have:

- A confluence of notions
- A definitive calculus of functions: the λ-calculus.

There has been active research on concurrency theory and processes for the past 5 decades in CS. Many important concepts and results have emerged. However, one cannot help noticing that:

The non-confluence of notions

In computability theory we have:

- A confluence of notions
- A definitive calculus of functions: the λ-calculus.

There has been active research on concurrency theory and processes for the past 5 decades in CS. Many important concepts and results have emerged. However, one cannot help noticing that:

- Hundreds of different process calculi, equivalences, logics have been proposed.

The non-confluence of notions

In computability theory we have:

- A confluence of notions
- A definitive calculus of functions: the λ-calculus.

There has been active research on concurrency theory and processes for the past 5 decades in CS. Many important concepts and results have emerged. However, one cannot help noticing that:

- Hundreds of different process calculi, equivalences, logics have been proposed.
- No λ-calculus for concurrency has emerged.

The non-confluence of notions

In computability theory we have:

- A confluence of notions
- A definitive calculus of functions: the λ-calculus.

There has been active research on concurrency theory and processes for the past 5 decades in CS. Many important concepts and results have emerged. However, one cannot help noticing that:

- Hundreds of different process calculi, equivalences, logics have been proposed.
- No λ-calculus for concurrency has emerged.
- There is no plausible Church-Turing thesis for processes.

The non-confluence of notions

In computability theory we have:

- A confluence of notions
- A definitive calculus of functions: the λ-calculus.

There has been active research on concurrency theory and processes for the past 5 decades in CS. Many important concepts and results have emerged. However, one cannot help noticing that:

- Hundreds of different process calculi, equivalences, logics have been proposed.
- No λ-calculus for concurrency has emerged.
- There is no plausible Church-Turing thesis for processes.

The 'next 700' syndrome from Peter Landin's classic 1966 paper 'The Next 700 Programming Languages' (!!).

The non-confluence of notions

In computability theory we have:

- A confluence of notions
- A definitive calculus of functions: the λ-calculus.

There has been active research on concurrency theory and processes for the past 5 decades in CS. Many important concepts and results have emerged. However, one cannot help noticing that:

- Hundreds of different process calculi, equivalences, logics have been proposed.
- No λ-calculus for concurrency has emerged.
- There is no plausible Church-Turing thesis for processes.

The 'next 700' syndrome from Peter Landin's classic 1966 paper 'The Next 700 Programming Languages' (!!).

It's a much harder problem! Will need a new Turing ...

Contextuality

Contextuality

Do physical variables have definite values, independently of how we choose to measure them?

Contextuality

Do physical variables have definite values, independently of how we choose to measure them?

Common sense says yes. QM says no!

Contextuality

Do physical variables have definite values, independently of how we choose to measure them?

Common sense says yes. QM says no!
How can we understand this contextuality? How can we use it!

Contextuality

Do physical variables have definite values, independently of how we choose to measure them?

Common sense says yes. QM says no!
How can we understand this contextuality? How can we use it!

	(R,R)	(R,G)	(G,R)	(G,G)
(X_1, Y_1)	1			
(X_1, Y_2)	0	1		
(X_2, Y_1)	0		1	
(X_2, Y_2)				0

Contextuality

Do physical variables have definite values, independently of how we choose to measure them?

Common sense says yes. QM says no!
How can we understand this contextuality? How can we use it!

	(R,R)	(R,G)	(G,R)	(G,G)
(X_1, Y_1)	1			
(X_1, Y_2)	0	1		
(X_2, Y_1)	0		1	
(X_2, Y_2)				0

If 'common sense' holds, there is a unique assignment that the outcome (R,R) for measurements (X_1, Y_1) could come from:

$$X_1 \mapsto R, \quad X_2 \mapsto G, \quad Y_1 \mapsto R, \quad Y_2 \mapsto G.$$

Contextuality

Do physical variables have definite values, independently of how we choose to measure them?

Common sense says yes. QM says no!
How can we understand this contextuality? How can we use it!

	(R,R)	(R,G)	(G,R)	(G,G)
(X_1, Y_1)	1			
(X_1, Y_2)	0	1		
(X_2, Y_1)	0		1	
(X_2, Y_2)				0

If 'common sense' holds, there is a unique assignment that the outcome (R, R) for measurements (X_1, Y_1) could come from:

$$X_1 \mapsto R, \quad X_2 \mapsto G, \quad Y_1 \mapsto R, \quad Y_2 \mapsto G.$$

However, this would require the possibility of outcomes (G, G) for measurements (X_2, Y_2), and this is precluded!

A world of processes and contexts

A world of processes and contexts

Traditionally, we view the world as made of objects and their properties. Moreover, realism implies that properties are objectively inherent in objects, independently of observers,

A world of processes and contexts

Traditionally, we view the world as made of objects and their properties. Moreover, realism implies that properties are objectively inherent in objects, independently of observers,

Taking processes as primary — 'relations without relata'.

A world of processes and contexts

Traditionally, we view the world as made of objects and their properties. Moreover, realism implies that properties are objectively inherent in objects, independently of observers,

Taking processes as primary — 'relations without relata'.

Taking contextuality as primary.

A world of processes and contexts

Traditionally, we view the world as made of objects and their properties. Moreover, realism implies that properties are objectively inherent in objects, independently of observers,

Taking processes as primary — 'relations without relata'.

Taking contextuality as primary.

Can we rethink our conception of the world in these terms?