# Virtual Machines and the Metaphysics of Science

Aaron Sloman

http://www.cs.bham.ac.uk/~axs
School of Computer Science

The University of Birmingham

I am not a computer scientist.

These slides are available online at
http://www.cs.bham.ac.uk/research/cogaff/talks/#mos09

See also: the Cognition and Affect Web Site
http://www.cs.bham.ac.uk/research/cogaff/

(Everything I do is work in progress. Comments and criticisms welcome.)

I have a related set of slides debunking "The 'hard' problem of consciousness"
http://www.cs.bham.ac.uk/research/projects/cogaff/talks/#cons09

Titles used for previous talks on this general topic:

What Cognitive Scientists Need to Know
About Virtual Machines

Virtual Machines in Philosophy,
Engineering & Biology

Why virtual machines really matter –
for several disciplines

What Are Virtual Machines?
Are They Real?

## Abstract for talk at MOS'09 Nottingham 12 Sept 2009

Philosophers regularly use complex (running) virtual machines (not virtual realities) composed of enduring interacting non-physical subsystems (e.g. operating systems, word-processors, email systems, web browsers, and many more).

These VMs can be subdivided into different kinds with different types of function, e.g. "specific-function VMs" and "platform VMs" (including language VMs, and operating system VMs) that provide support for a variety of different "higher level" VMs, with different functions.

Yet, almost all philosophers ignore (or misdescribe) these VMs when discussing functionalism, supervenience, multiple realisation, reductionism, emergence, and causation.

Such VMs depend on many hardware and software designs that interact in very complex ways to maintain a network of causal relationships between physical and virtual entities and processes.

I'll try to explain this, and show how VMs are important for philosophy, in part because evolution long ago developed far more sophisticated systems of virtual machinery (e.g. running on brains and their surroundings) than human engineers so far. Most are still not understood.

This partly accounts for the complexity, and in some cases apparent intractability, of several philosophical problems, including problems about self-awareness and the contents of consciousness.

E.g. running VM subsystems can be disconnected from input-output interactions for extended periods, and some can have more complexity than the available input/output bandwidth can reveal.

Moreover, despite the advantages of VMs for self-monitoring and self control, they can also lead to self-deception.

A longer abstract is here
`http://www.cs.bham.ac.uk/research/projects/cogaff/09.html#vms`

# From The Computer Revolution in Philosophy, Ch.1 (1978)

**Extract from §1.3. Themes from the Computer Revolution [pp 9–10]**

“6. One of the major new insights is that computational processes may be markedly decoupled from the physical processes of the underlying computer. Computers with quite different basic components and architecture may be equivalent in an important sense: a program which runs on one of them can be made to run on any other either by means of a second program which simulates the first computer on the second, or by means of a suitable compiler or interpreter program which *translates* the first program into a formalism which the second computer can execute. So a program may run on a *virtual* machine.

Differences in size can be got round by attaching peripheral storage devices such as magnetic discs or tapes, leaving only differences in speed.

So all modern digital computers are theoretically equivalent, and the detailed physical structure and properties of a computer need not constrain or determine the symbol-manipulating and problem-solving processes which can run on it: any constraints, except for speed, can be overcome by providing more storage and feeding in new programs. Similarly, the programs do not determine the computers on which they can run.

7. Thus reductionism is refuted. For instance, if biological processes are computational processes running on a physico-chemical computer, then essentially the same processes could, with suitable re-programming, run on a different sort of computer. Equally, the same computer could permit quite different computations: so the nature of the physical world need not determine biological processes. Just as the electronic engineers who build and maintain a computer may be quite unable to describe or understand some of the programs which run on it, so may physicists and chemists lack the resources to describe, explain or predict biological processes. Similarly psychology need not be reducible to physiology, nor social processes to psychological ones. To say that wholes may be more than the sum of their parts, and that qualitatively new processes may ‘emerge’ from old ones, now becomes an acceptable part of the science of computation, rather than old-fashioned mysticism. Many anti-reductionists have had this thought prior to the development of computing, but have been unable to give it a clear and indisputable foundation.

8. There need not be only *two* layers: programs and physical machine. A suitably programmed computer (e.g. a computer with a compiler program in it), is itself a new computer a new ‘virtual machine’ which in turn may be programmed so as to support new kinds of processes. Thus a single process may involve many layers of computations, each using the next lower layer as its underlying machine. But that is not all. The relations may sometimes not even be hierarchically organised, for instance if process A forms part of the underlying machine for process B and process B forms part of the underlying machine for process A. Social and psychological, psychological and physiological processes, seem to be related in this mutually supportive way. Chapters 6 and 9 present some examples. The development of good tools for thinking about a system composed of multiple interlocking processes is only just beginning.” `http://www.cs.bham.ac.uk/research/projects/cogaff/crp/chap1.html`

These slides both elaborate on and to some extend diverge from that position, since some virtual machines may need special-purpose physical devices: **not all physical computing machines can support all virtual machines.**

## Themes (not necessarily presented in this order)

- Explain what a machine is.
- Explain "physical machine (PM)"
- Explain non-physical, i.e. virtual (non-physically-describable), machine. (VM, NPDM)
- Why Running VMs (RVMs) are useful in engineering – e.g. self-monitoring and self-control.
- Summarise some of the complex history of how VMs were developed.
- Conjecture that biological evolution found problems requiring VMs and solutions that use VMs.
- Illustrate the variety of types of VMs (including "specific-function VMs" and "platform VMs" that support a variety of different VMs, with different functions).
- Identify some of the open questions about VMs that are worthy of further research, including the importance of understanding more deeply the tradeoffs between different types
- Explain some of the scientific problems about VMs and some of the consequences of using them for self-control.
- List some of the open research questions, especially about biological VMs, how they evolved, what they can do, how they do it, how they develop in individuals, how genomes specify them, and what PMs are needed to support them.
- Introduce some of the philosophical problems about VMs and philosophical problems for which a study of VMs may suggest new answers.

  e.g. problems about supervenience, causation, mind-body relations, self-awareness.

I'll start by testing your intuitions by asking you to vote on a few questions.

## Let’s vote

Does **your** ontology include virtual machines?

Who agrees with the following?

- Ignorance can cause poverty?
- Over-zealous selling of mortgages can cause hardship for people in many countries?
- Voting can influence decisions?

**If you AGREE with any of these, i.e. if you think such an effect CAN occur, then it appears that you (wittingly or unwittingly) have social and/or socio-economic virtual machines in your ontology.**

What that means is another (hard) question, partially answered below.

In order to explain what a virtual machine is we need to:

- Explain what a machine is
- Describe “physical machine” in terms of concepts that suffice to describe the structure and operation of the machines.
- Define “virtual machine” as a machine that is not (fully) physically describable.

  (The phrase “virtual machine” is unfortunate, but it’s too wide-spread to change.)

## Some tempting over-generalisations

Two that are very common:

- **Every computer presentation is a powerpoint presentation.**
- **Every computer is a Turing machine.**

Are philosophers willing learn more about these matters?

I hope philosophers here don't share the view that philosophy can ignore all empirical sciences because philosophy is only concerned with conceptual truths.

Logical topographies underlie logical geographies:
`http://www.cs.bham.ac.uk/research/projects/cogaff/misc/logical-geography.html`

**I am really asking for help:**

I think many common assumptions made by philosophers about information-processing machines are mistaken (implicit mistakes) because of what philosophers are generally NOT taught.

... so, many common assumptions about some central philosophical problems are wrong.

But finding correct alternatives (based on good theories about logical topographies) is a non-trivial task.

SHOW DEMO

## What is a machine (natural or artificial)?

**A machine** (as I use the word) **is a complex enduring entity** with parts

(possibly a changing set of parts)

that **interact causally** with other parts, and other "external" things, as they change their properties and relationships.

The internal and external interactions may be

- **discrete** or **continuous,**
- **concurrent** (most machines), or **sequential** (e.g. row of dominoes, a fuse(?))

Contrast Turing machines: Everything is

- Internal
- Discrete
- Sequential

Concurrent and synchronized machines are equivalent to sequential machines.

But some concurrent machines are not synchronised

## What is a machine (natural or artificial)?

**A machine** (as I use the word) **is a complex enduring entity** with parts

(possibly a changing set of parts)

that **interact causally** with other parts, and other “external” things, as they change their properties and relationships.

The internal and external interactions may be

- **discrete** or **continuous,**
- **concurrent** (most machines), or **sequential** (e.g. row of dominoes, a fuse(?))

**NOTEs**

1. Machines, in this general sense, do not have to be artificial, or man-made, or deliberately designed to do what they do.
2. The perception of machines and how they work is one of the important functions of human visual perception, and haptic/tactile perception, (possibly also in some other species).

That includes the perception of structures, processes and causal relationships (proto-affordances).

This is generally ignored by vision researchers.

Perception of affordances is a special case of this. E.g. See

Architectural and Representational Requirements for Seeing Processes, Proto-affordances and Affordances,
`http://drops.dagstuhl.de/opus/volltexte/2008/1656`

# Typical features of machines (natural and artificial):

Machines

- can have various degrees and kinds of complexity

  (often hierarchical – machines composed of machines)

- allow changes/processes to occur within them

  usually concurrent changes (e.g. gear wheels turning, ends of lever moving in opposite directions)

- include processes that involve causation
  - within the machine

    E.g. parts moving other parts, forces transmitted, information stored, retrieved, derived or transmitted, parts controlling or activating, other parts.
  - partly within the environment

    E.g. if there are sensors, motors, and communication channels
  - involving matter, motion, forces, energy, **information**, ... and more

- are usually embedded in a complex environment with which they interact. Often the boundary between machine and environment is different for different sub-systems of the machine.

  E.g. for reasoning, you may use pen and paper as an extension of yourself.
  Sloman IJCAI 1971:
  `http://www.cs.bham.ac.uk/research/cogaff/04.html#200407`

- may include some internal processes whose effects are not externally detectable

  (More on this later)

## What is a **physical machine** (PM)?

If a machine and its operations (processes, and causal relationships) are fully describable using concepts of the physical sciences (plus mathematics), it is a physical machine (PM).

That’s a first draft specification.
(There is probably a variant definition that does not mention concepts, but I am not sure.)

**Of course, this notion is relative to the current contents of physics, which change over time.**

**Examples of physical machines include:** levers, assemblages of gears, mechanical clocks, audio amplifiers, electronic devices, clouds, tornadoes, plate tectonic systems, atoms, bacteria, brains, and myriad molecular machines in living organisms.

There is much we don’t know about what sorts of machine can be built out of chemical components. E.g. read recent issues of *Scientific American*.

Our notion of a virtual machine (VM) is defined as a machine that is not a physical machine, in the sense of “physical machine” defined above:

i.e. a VM can have interacting parts, but its operations are not fully describable physically – its full description requires concepts not definable in terms of concepts of the physical sciences. E.g. spelling checker, chess program, proof checker.

This notion will now be elaborated.

## Non-physically-definable (NPD) concepts

Certain states, processes and interactions of some machines

cannot be described using **only** concepts
that are definable in terms of concepts
of the physical sciences
(E.g. the concepts of physics, chemistry, plus mathematics.)

Information-processing machines are examples.

"Information" is not used here in Shannon's sense,
but in the sense that includes "reference", "meaning",
with properties and relations like
**truth, consistency, contradiction,**
This concept is not **definable** in terms of concepts of the physical sciences.

Though every information-using machine must be implemented (realised) in a physical machine.
See `http://www.cs.bham.ac.uk/research/projects/cogaff/misc/whats-information.html`

Non-physical machines include: socio-economic machines, ecosystems, many biological control systems, and many of the things that run in computers, including games, spelling checkers, operating systems and networking systems.

## Compare pictures, statues, etc.

A configuration of coloured marks on white paper can have its physical constitution (e.g. the various types of molecule involved, and their reflective properties) described using concepts of the physical sciences.

Suppose the configuration is

- The national flag of some particular country
- A picture of someone falling over
- A picture of happy face
- The hand-written word "aspidistra"
- A musical score (e.g. the opening bars of a Schubert quintet)
- A morse code transcription of a Shakespeare Sonnet
- A polygon with a prime number of sides
- A pattern that reminds someone of his mother's face...
- A picture that machine M has been programmed to react to by exploding.
  That picture has specific causal powers.

Are all these descriptions merely subjective interpretations of the marks?

All the examples need careful analysis.

For now I merely wish to draw attention to the fact that non-static physical configurations (i.e. physical processes) can also have non-physical descriptions of various sorts.

## More on non-physically describable machines

**Non-Physically-Describable Machines (NPDMs)** are the subject matter of common sense, gossip, novels, plays, legends, history, the social sciences and economics, psychology, and various aspects of biology.

Examples of such non-physically-definable concepts:

“information”, “inference”, “contradiction”, “strategy”, “desire”, “belief”, “mood”, “promise”, “contract”, “checking spelling”, “file access violation”, “sending email”, “playing chess”, “winning”, “threat”, “defence”, “plan”, “poverty”, “crime”, “economic recession”, “election”, “war”, ...

**For now I’ll take that indefinability as obvious:**
It would take too long to explain and defend.

This is connected with the falsity of “concept empiricism” and “symbol grounding theory”. See
`http://www.cs.bham.ac.uk/research/projects/cogaff/talks/#models`
`http://www.cs.bham.ac.uk/research/projects/cogaff/misc/whats-information.html`

**In computer science and software engineering NPDMs are often called “virtual machines”** (terminology possibly derived from some of the earliest examples: virtual memory systems).

This terminology is unfortunate – since it can suggest that such machines don’t really exist – like the entities represented in virtual reality systems.

Nevertheless we are stuck with it.

(Like “Artificial Intelligence”, which includes the study and modelling of natural intelligence.)

## Computer Scientists refer to two sorts of VM

We contrast the notion of a PHYSICAL machine with two other notions:

1. a VM which is **an abstract mathematical object** (e.g. the Prolog VM, the Java VM)
2. a VM that is **a running instance of such a mathematical object**, controlling events in a physical machine, e.g. a **running** Prolog or Java VM.
   Running VMs (RVMs) are what this presentation is about.

| Physical processes: | Mathematical models: | Running virtual machines: |
|---|---|---|
| currents | numbers | calculations |
| voltages | sets | games |
| state-changes | grammars | formatting |
| transducer events | proofs | proving |
| cpu events | Turing machines | parsing |
| memory events | TM executions | planning |

**VMs as mathematical objects are much studied in meta-mathematics and theoretical computer science.**

**They can have complex structures, but are no more causally efficacious than numbers.**

The main theorems of computer science, e.g. about computability, complexity, etc. are primarily about mathematical entities

They are applicable to non-mathematical entities with the same structure – but no non-mathematical entity can be proved mathematically to have any particular mathematical properties.

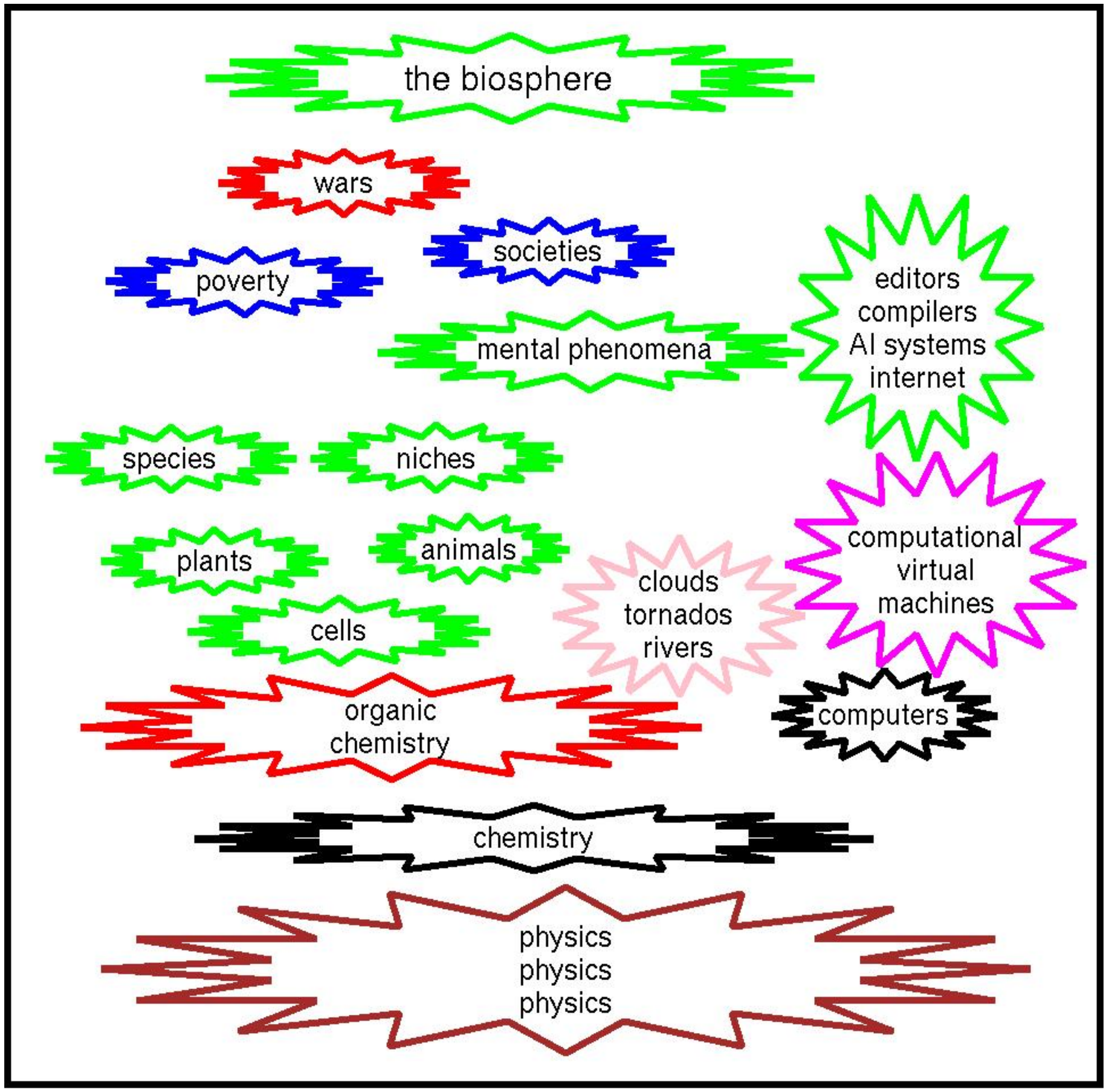
There's more on varieties of running virtual machines (RVMs) in later slides.

## Running virtual machines are everywhere

At all levels there are objects, properties, relations, structures, mechanisms, states, events, processes and also many CAUSAL INTERACTIONS.

E.g. poverty can cause crime.

- All levels are ultimately realised (implemented) in physical systems.
- Different disciplines use different approaches (not always good ones).
- Nobody knows how many levels of virtual machines physicists will eventually discover. (Uncover?)
- The study of virtual machines in computers is just a special case of more general attempts to describe and explain virtual machines in our world.



NB: Universal Turing Machines are universal only relative to a restricted class of machines.

**Some VMs may require specialised PMs.**

## Some unusual features of this characterisation

Often the idea of a virtual machine is introduced in connection with one or more of the following ideas:

- Computer program (software)
- Turing machines
- Multiple realisability

I did not emphasise any of those in defining VMs as machines whose operations cannot be fully described using the language of the physical sciences.

However

- The best known examples involve running computer programs
  (often a collection of different programs, interacting in various ways, including an operating system various device drivers, and different application programs, all sharing one machine).

  But there is no implication that all RVMs are based on computers. There may be VMs implemented in specific biological machinery that cannot do anything but run a particular type of virtual machine.
- Turing machines do not have to be involved in the VMs I have been talking about:

  there is no implication that a VM has to be running on some sort of extremely general purpose computing engine, though the VM may need to be able to switch between different special cases of the tasks that it can perform.
- No multiple realisability requirement has been mentioned:

  the underlying hardware may need to be usable for different sub-tasks, processing different information contents, at different times, or more generally running different VMs at different times, e.g. switching between playing chess and playing draughts (checkers).

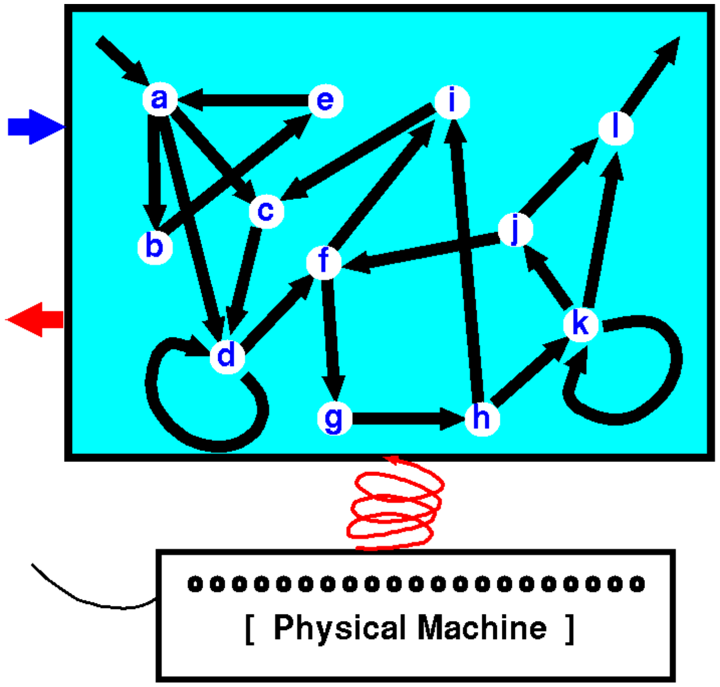
# So, what exactly are Running VMs?

## Some misconceptions: Atomic State VMs

It is often thought that everything of interest about a computer reduces to its being a finite state machine, following a trajectory through its state space, where each state corresponds to a collection of switches being on or off.

The letters label states and the arrows label transitions that can be triggered by inputs (blue arrow). Some transitions can also produce outputs (red arrow).

States in a Turing machine are essentially like this, with the tape replacing the world, and with extremely restricted inputs and outputs.



The trajectory of states in such a VM is usually very close to the trajectory of the underlying PM.

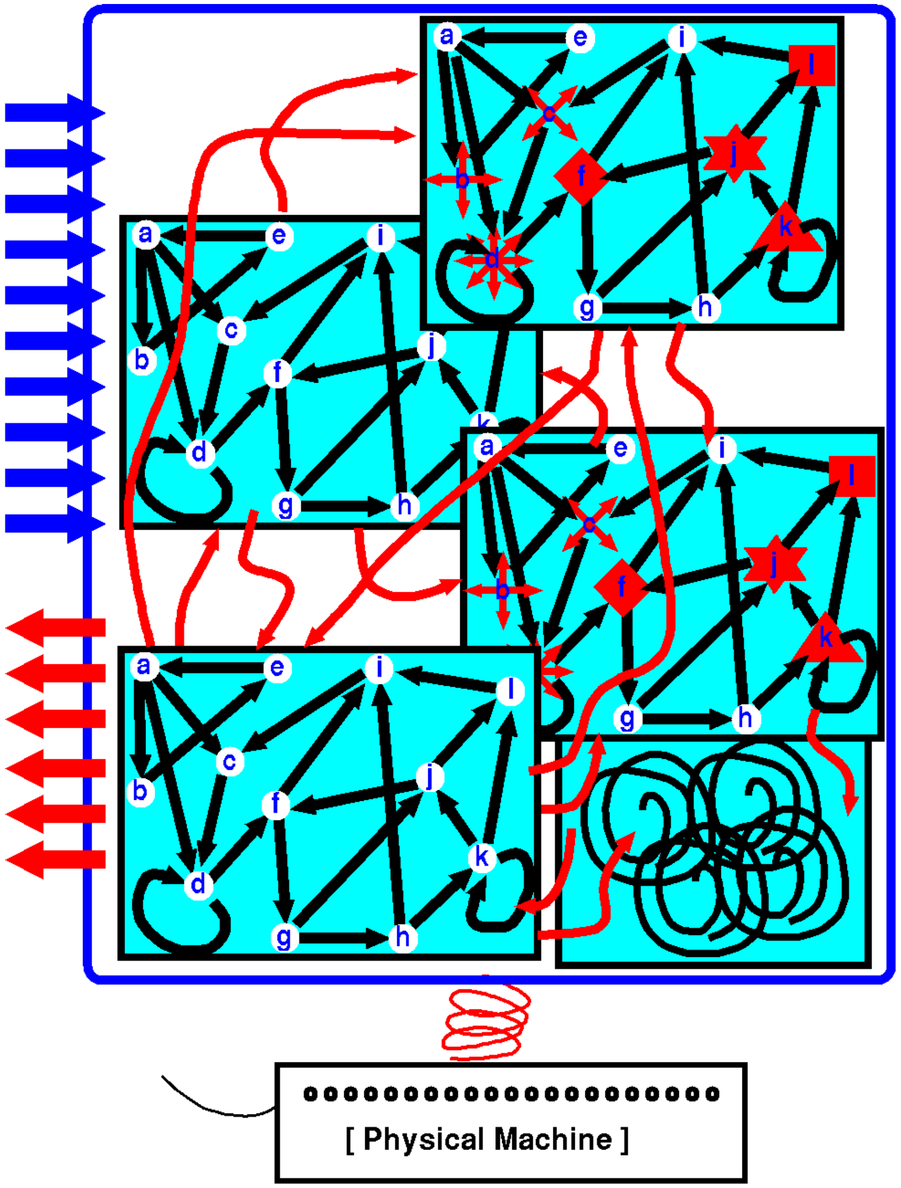
That’s unlike modern computing systems, in which physical state transitions are very different from virtual machine transitions.

## A richer model of a VM

**Instead of a single sequential process, a VM can have parts that run in parallel (even asynchronously), interacting with one another and the environment.**

**Instead of having a fixed set of sub-processes, many computing systems allow new VMs to be constructed dynamically,**

- of varying complexity
- some of them running for a while then terminating,
- others going on indefinitely.
- some spawning new sub-processes...
- some discrete, some continuous
- with some transitions probabilistic
- with multiple internal connections (e.g. communication channels)
- a subset connected to external interfaces (possibly sharing input and output devices).



See "The mind as a control system"

`http://www.cs.bham.ac.uk/research/projects/cogaff/81-95.html#18`

# Could such virtual machines run on brains?

It can be very hard to monitor and control all the low level physical processes going on in a complex machine: so it is often useful to introduce virtual machines that are much simpler and easier to control, at the level of their intended functionality.

**Perhaps evolution “discovered” the importance of using virtual machines to enable very complex systems to control themselves, long before we did?**

So, VMs running on brains could provide a high level control interface including self-monitoring, and self-modulation.

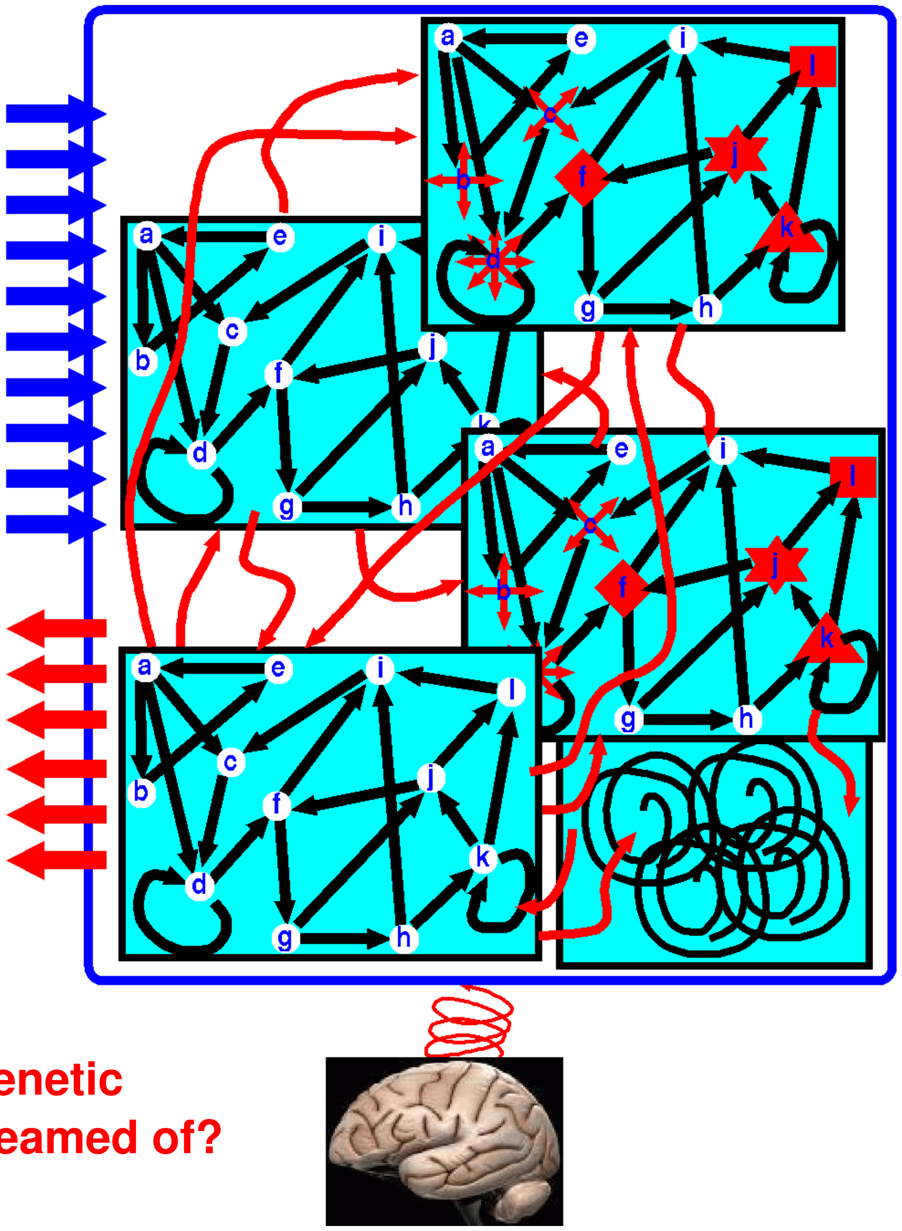
**Questions:**

**How would the genome specify construction of virtual machines?**

**What role could the environment play?**

**Could there be things in DNA, or in epigenetic control systems, that we have not yet dreamed of?**

Chappell & Sloman, IJUC, 2007



## Not all VMs can run on Turing machines

**I am not defining "virtual machine" in terms of runnability on a Turing Machine or any other particular kind of machine.**

Theorems about the universality of Turing machines can be misleading: they are universal only given to certain assumptions, which may not be satisfied by the physical universe.

- TMs (and von Neumann machines) can approximate machines with continuous changes, but cannot implement them exactly, and if a continuous machine has non-linear feedback loops it may be chaotic and impossible to approximate discretely, even over short time scales.
- If a machine is composed of asynchronous concurrently running parts, and their relative speeds vary randomly then that machine cannot be implemented on a Turing machine.

  If two TMs, T1, outputting an unending sequence of 0s, and T2 outputting an unending sequence of 1s, have relative speeds that vary randomly (e.g. controlled by a geiger counter), they can jointly produce a non-computable binary sequence.

  (The vast majority of infinite binary sequences are not computable, because there are far more of them than finite turing machines.)
- Machines connected to the environment may have unpredictable behaviours.

So, even familiar computers connected to the environment may have behaviours that cannot be modelled on a stand-alone Turing Machine.

## "Sense–Think–Act" models are too restrictive

Concurrent interacting VMs remove the need for "Sense–Think–Act" loops.

Many computational modellers and AI researchers assume that an intelligent machine must repeatedly:

- Sense/perceive what is in the environment
- Think about implications and what to do
- Act according to the decisions taken.
- Repeat the process

  (Some researchers add more stages, e.g. "reflective" stages, while keeping them sequential.)

This leads to (a) impoverished architectural designs, and (b) a substantial literature on how to optimise the allocation of processor time to the three (or more) stages.

**In contrast, in biological brains there is considerable separation of function so that different tasks can be done concurrently (using their own specialised hardware).**

Examples include seeing, walking, thinking, and talking concurrently, and integrating use of multiple sensor and effector channels in dynamical systems, e.g. while running, jumping, etc.

Also various kinds of internal self-monitoring, e.g. during speech production and other processes.
`http://www.cs.bham.ac.uk/research/projects/cosy/papers/#tr0803`,
Varieties of Meta-cognition in Natural and Artificial Systems

Of course, there can be subsystems with sense-think-act loops.

## Concurrent VMs and Cognitive Science

**IF** there is no universal requirement for sequential (uni-process) operation in RVMs

(e.g. because a VM can be composed of dynamically constructed concurrently interacting VMs)

**THEN** a task for **theoretical** cognitive science is to understand

- the implications of such concurrency + asynchrony + some analog computations;
- the trade-offs between different ways of allocating dedicated resources to different sorts of functionality

  e.g. sensory mechanisms, motor mechanisms, different short-term and long term memories, learning mechanisms, motive-generation mechanisms, self-monitoring mechanisms, perceptual interpretation layers, different control layers for actions, (e.g. subsumption mechanisms) etc.
- the tradeoffs compared with the more common AI designs sharing all processing power sequentially between different functions.

A task for **empirical** cognitive science is to find out what sorts of concurrent VMs are implemented in different animal brains (including humans of various ages), and why.

This may provide new inspiration for designs for robots, and other artificial systems.

In chapter 6 of *The Computer Revolution in Philosophy*(1978) I argued for considerable concurrency in a human-like architecture and in chapter 9 demonstrated some of the ways this could work in visual subsystems processing different ontological levels in parallel. `http://www.cs.bham.ac.uk/research/projects/cogaff/crp/`

Compare: “The mind as a control system”:
`http://www.cs.bham.ac.uk/research/projects/cogaff/81-95.html#18`

# Dennett on Virtual Machines (1)

Dennett writes more about virtual machines than most people. E.g. there are several index entries for the phrase in *Consciousness Explained*. Much of the time I think I agree with Dennett, and then he surprises me...

In "Heterophenomenology reconsidered" he writes about: *"standard practice in computer circles, where virtual machine talk is allowed to pass for the truth with no raised eyebrows, no need for the reminder that it is, officially, just metaphorical."*

No: when software engineers talk about what went wrong in a program, and what needs to be fixed to make it work properly, they are not just talking in metaphors.

**It could be the literal truth that a decision failed to take account of some information, or that some out of date information was mistakenly treated by the machine as true because it failed to perform some test.**

This is not misleading, metaphorical shorthand, for some much more complex statement about what the transistors, or electrons did or did not do.

There's a big difference between

1. a disaster caused by some physical component going wrong, and
2. a disaster caused because some information was misinterpreted, or a condition wasn't checked, or a faulty rule was used.

## Dennett on Virtual Machines (2)

In *Consciousness Explained* he writes, in connection with virtual machines:

> "In a von Neumann machine, you just 'load' the program off a disk into the main memory and the computer thereby gets an instant set of new habits...." (p. 218)

That's true if all you are doing is adding a new application VM (defined later) to an already fully installed computing system, e.g. installing a simple game program.

But the process can be very different (and far from "instant") if you acquire a computer without an operating system and want to start installing software.

Even with an operating system, not all software will be runnable on it: e.g. you cannot simply load and run MSOffice on Linux.

(OpenOffice is a good replacement for many users, but the virtual machine it installs is different.)

Moreover, if you want the machine to be able to see and manipulate things in the environment you may have a choice of cameras and arms, but the software needed to drive them will be different.

- The existence and usefulness of running VMs does not presuppose any restriction to use of Universal Turing Machines (UTMs).
- The idea of a UTM which can emulate any other TM is very important in theoretical computer science, but engineers build boxes containing different specialised computing devices providing different sorts of functionality, going beyond what a TM can do, because a TM is not interfaced with a physical environment.
- Biological evolution has also produced boxes containing lots of different information processing subsystems supporting different kinds of functionality.

## Dennett on Virtual Machines (3)

These quotes from Dennett are better:

> "A virtual machine is What you get when you impose a particular pattern of rules (more literally: dispositions or transition regularities) on all that plasticity."
> *Consciousness Explained* p.211

The existence of a particular set of dispositions, or transition regularities, described at a virtual machine level, is important, and will be discussed further later.

But the pattern of rules need not be imposed: the rules may be grown "organically" by the information processing system interacting with its environment – like a human learning a language.

**The following is not quite correct, in general:**

> "A virtual machine is a temporary set of highly structured regularities imposed on the underlying hardware by a program: a structured recipe..."
> (Densmore and Dennett, 1999)

The regularities do not need to be temporary: there is no reason why a particular virtual machine should not be directly supported by dedicated hardware.

Common examples are things like a floating-point unit.

However, it is often the case that the hardware used could have been included in a computing unit supporting different functionality (with different firmware, or software).

# Dennett on Virtual Machines (4)

All of the above leaves out the important question:
Can events and processes in VMs be causes, producing effects?

Densmore and Dennett clearly seem to say yes:

*"If you want to explain why Churchland gets the answers to the long division problems right so often, why certain problems take him longer than others, and why his pencil-pushing behavior produces the patterns of marks on the paper that it does, then the level to which you must ascend to explain is the level at which he is hand simulating the long-division machine. If instead what you want to explain are some other regularities in his behavior, such as his humming or whistling while he works, or his periodic lapses into dreamy-eyed grinning, or his muttered sub-vocalizations, then, according to Dennett, you had best descend to a somewhat lower level, but not – if you actually want to explain these patterns – all the way down to the level of the recurrentPDP networks, which are at least one level too low into the trees to permit us to see the woods we are interested in."*

Here they are talking about the reality, and the causal powers, of VM components as a software engineer would.

These components are not merely hypothesised by someone adopting the "intentional stance" on the assumption of rationality.

The existence of a particular VM running in some complex machine is a matter of fact, not a mere locutionary convenience.

How that fact can be established is another matter: it can be very difficult if you are not the designer of the machine.

## Dennett on Virtual Machines (5)

Suppose you are shown X, then later you are sure you saw Y, Dennett contrasts two explanations of what happened, then asserts that the distinction is bogus:

- Orwellian **O**: You saw X and experienced X, very briefly, and stored a memory of the experience; but immediately your stored memory of that experience was changed so that you remember seeing Y.
- Stalinesque **S**: Your visual subsystem got information about X, but some malfunction caused the information to be altered so it referred to Y. That was then stored, and you remember seeing Y.

Discussed in

Steven J. Todd A Difference That Makes a Difference: Passing through Dennett's Stalinesque/Orwellian Impasse, *Br J Philos Sci* 2009 60: 497-520; `http://bjps.oxfordjournals.org/cgi/reprint/60/3/497`

Dennett and Kinsbourne: there is no such thing as a theatre of consciousness (ToC), only "multiple drafts" of received information (MD), so there cannot be any empirically detectable difference between **O** and **S**, and so the idea of a difference is incoherent.

From our point of view the information that you experience and are capable of reporting must reach a VM subsystem from which it can be accessed by the "meta-management" subsystem which has meta-semantic competences and also communicative competences: these are portions of the virtual machine that is your mind.

So it does make sense to distinguish two sources of error about what you experienced:

- (O) the information reached the meta-management subsystem, where it was explicitly noted, and then subsequently all records were altered, or
- (S) the information was somehow mangled in earlier stages of processing and was already incorrect by the time it reached the meta-management subsystem.

**We can build working models showing this distinction.**

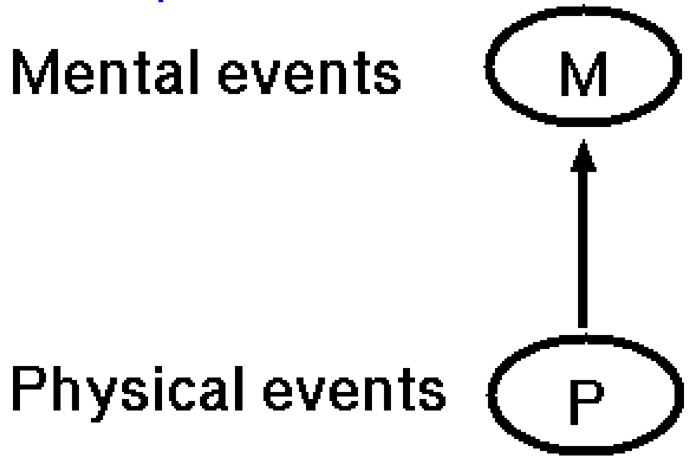
## How VM events and processes can be causes

We need an explanation of how VM causes and PM causes can co-exist and both be causes of other VM and PM events and processes.

Here is an impoverished way of thinking about this (often associated with "property supervenience": a non-physical state or property supervenes on a physical one).

The diagram (crudely) shows how many have thought about the causal relation between mental and physical states and processes: physical/chemical processes somehow produce something else, which supervenes on them.

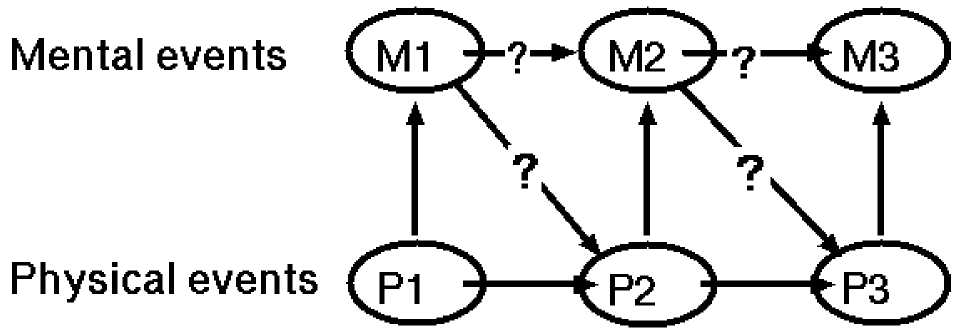
But the "somethings else" (e.g. M) have no causal powers of their own.



This leads to puzzles about how mental events can cause either other mental events or physical events.

**If P1 causes M1 and P2, and P2 causes M2 and P3, and P3 causes M3, then no M can cause anything.**

**A sequence of physical and mental events**



This seems to prove that the M's make no difference: they are epiphenomenal.

But it assumes causation goes only along and upwards ...

## People mostly agree that there are non-physical causes

- Ignorance can cause poverty.
- Poverty can cause crime (including objects being moved, broken, burnt, etc.)
- Over-zealous selling of mortgages can cause hardship for people in many countries.
- Voting can influence decisions.
- Percepts can cause emotions.
- Emotions can affect facial expressions, posture, words uttered...

### How is that possible?

Let’s look more closely at virtual machines running on computers: events in RVMs (e.g. word processors) can cause or modulate not only other VM events and processes but can also cause or modulate physical events and processes (in screens and magnetic drives).

### How is that possible?

We know very little about how minds and brains work – as a result of millions of years of evolution.

But we know a lot about how RVMs and computers work, – as a result of 60-70 years of hardware and software engineering design.

**So why have philosophers investigating supervenience and causation generally completely ignored what’s happening under their noses?**

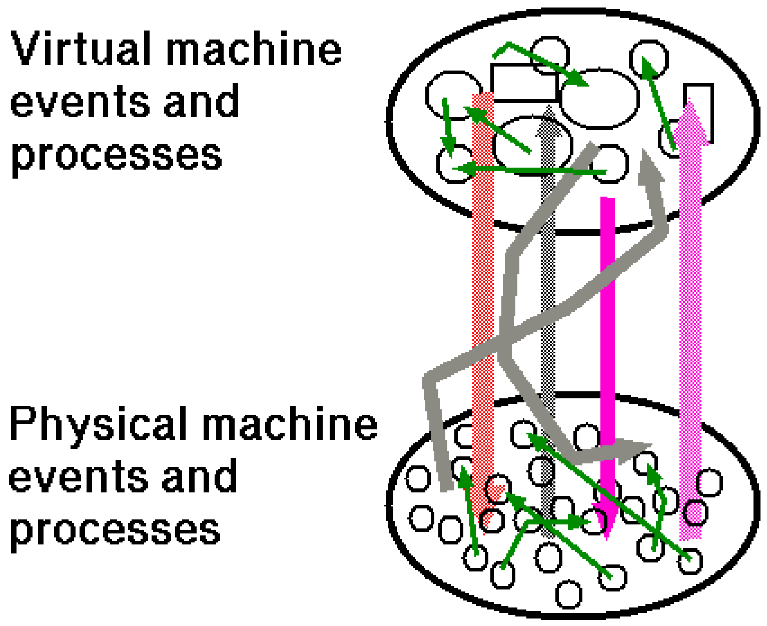
## The arrow diagrams were wrong

Our previous diagrams implicitly supported a prejudice by showing a single upward pointing arrow from each physical state to the mental, or virtual machine state, above it.

This implied a simple one-way dependency relationship, with VM changes proceeding in step with PM changes.

**We need a richer model:**

- In modern computing systems VM changes have a different granularity from PM changes and they do not occur in step.
- Use of PM components can be shared between VM components.
- Mappings between PM components and VM components can change rapidly
- PM events and processes can be linked by networks of causation between VM events and processes.
- There are true counterfactual conditional statements about VM phenomena that cannot be translated into statements using only the language of electronic engineering:
  **they involve dispositions of more abstract types.**


Virtual machine events and processes
Physical machine events and processes

The previous arrow diagrams present a seriously misleading picture:

They oversimplify even what goes on in computers, let alone social systems and minds.

(Modern computing systems have little resemblance to Turing machines.)

## How is this possible? (Forget Turing Machines!)

Look at the history of computer systems over many decades.

- The move from low level machine-level instructions to higher level languages (using compilers and especially interpreters)
- Memory management systems made physical memory reference context dependent.
- Virtual memory (paging and cacheing) and garbage collection switched virtual memory contents between faster and slower core memories and backing store, and between different parts of core memory: constantly changing PM/VM mappings.
- Networked file systems changed apparent physical locations of files.
- Device interfaces ‘translated’ physical signals into VM signals and vice versa.
- Devices could themselves run virtual machines with buffers, memories, learning capabilities...
- Device drivers (software) handled mappings higher level and lower level VMs – and allowed devices to be shared between VMs (e.g. network interfaces).
- Interrupt handlers help to distribute causal powers over more functions.
- Non-active processes persist in memory and can have effects on running processes.
- Multi-cpu systems allow mappings between instruction executions associated with VMs and PMs to change.
- Multiplicity of concurrent functions grew – especially on networked machines.
- More and more control functions were concerned with VM states and processes.

## Physical $\Longleftrightarrow$ virtual interfaces at different levels

Starting with simple physical devices implementing interacting discrete patterns, we have built layers of interacting patterns of ever increasing spatial and temporal complexity, with more and more varied functionality.

- Physical devices can combine continuously varying states constrained to allow only a small number of discrete stable states (e.g. only two)

  (e.g. using mechanical ratchets, electronic valves (tubes), aligned magnetic molecules, transistors etc.)
- Networks of such devices can constrain relationships between discrete patterns.

  E.g. the ABCD/XY example: a constraint can ensure that if devices A and B are in states X and Y respectively then devices C and D will be in states Y and X (with or without other constraints).
  So, a device network can rule out some physically possible combinations of states of components, and a new pattern in part of the network will cause pattern-changes elsewhere via the constraints.
  Compare: one end of a rigid lever moving down or up causes the other end to be moving up or down.
- Such networks can form dynamical systems with limited possible trajectories, constraining both the possible patterns and the possible sequences of patterns.
- A network of internal devices can link external interfaces (input and output devices)
  thereby limiting the relationships that can exist between patterns of inputs and patterns of outputs, and also limiting possible sequences of input-output patterns.
- Patterns in one part of the system can have meaning for another part, e.g.
  - constraining behaviour (e.g. where the pattern expresses a program or ruleset) or
  - describing something (e.g. where the pattern represents a testable condition)
- Such patterns and uses of such patterns in interacting computing systems may result from design (e.g. programming) or from self-organising (learning, evolving) systems.
- Some useful patterns need not be describable in the language of physics.

## Try playing with interacting abstract patterns

There are movies showing some videos of some interacting virtual machines implemented in the (freely available) Poplog SimAgent Toolkit here:

`http://www.cs.bham.ac.uk/research/projects/poplog/packages/simagent.html`

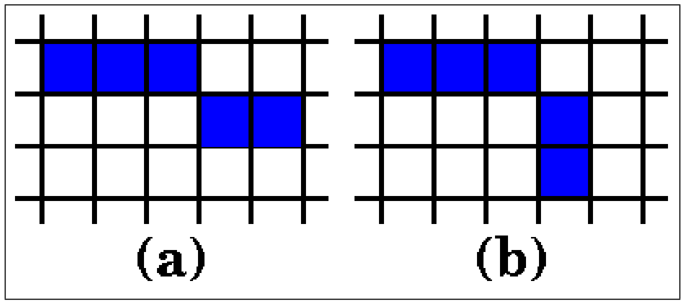
John Conway's "Game of Life" provides many examples of interacting abstract patterns.

An excellent online version can be run in a (java-enabled) web browser here:

`http://www.bitstorm.org/gameoflife/` (It allows mouse interaction with a running process.)

Anyone can play with interacting patterns by clicking on squares to make an initial pattern or set of patterns (or selecting a pre-built pattern) from the menu labelled 'Clear', and then stepping through the interactions by repeatedly pressing 'Next', or launching the process by pressing 'Start'.

For example, a starting pattern of five squares like the pattern (a) very quickly fizzles out, whereas moving one of the blue squares to form the starting pattern (b) on the right produces a totally different result: once started the display continues changing hundreds of times without repeating itself until it eventually (after more than 1100 changes – if the board is big enough) gets stuck with a collection of static or repeating small patterns.



Such demonstrations illustrate ways in which abstract patterns can be made to interact causally (parts of patterns cause other parts to change), by harnessing physical machinery to operate according to a set of constraints (Conway's four rules).

The constraints can be implemented via mechanical linkages, or via electronic circuits, or via a computer program that forces the computer hardware to behave as if wired to obey the rules: as a result a group of squares changing state can cause a new pattern to come into existence.

## Which patterns interact?

When Conway's machine runs, do patterns on the screen interact causally? NO!

The screen merely displays what is going on in a virtual machine running in the computer.

In the running VM the abstract data-structures in the 2-D grid interact.

Those changing VM structures are represented by changing squares on the screen merely to inform us what is happening inside.

- In a different physical design the screen could be part of the implementation of the VM.
- If the grid information used in applying the rules were stored in the pixels of the screen display, then we could say that the cells in the screen interact – but not their visual effects.
- Most Conway implementations would go on running even if the screen were disconnected: they would merely cease to display anything. That is how some people understand contents of consciousness – i.e. as causally/functionally ineffective (e.g. P-C as defined by Block).
  (Perhaps they think zombies are possible because they imagine turning off an internal screen.)
- However, if you are inspired to interact by clicking in the grid, because of what you see, then the visible screen patterns are causally effective in changing how the process develops, and your brain becomes part of the implementation of a very sophisticated distributed virtual machine!

The causation observed in a Conway machine is normally only in the (invisible) virtual machine that causes what is on the screen to change: the screen display is (relatively) epiphenomenal, like changing expressions on people's faces, when they are alone!

For more examples and references see `http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life`

## More complex interacting patterns

If two separated patterns are initialised in the machine then after a while they can interact and how they interact will depend on the precise “spatial” relations between the patterns and their structures.

Compare what happens when two virtual machines running on a computer compete in a chess game, sharing a virtual board, and interacting through moves on the board.

- In some cases, like the game of life, the interacting processes are purely reactive: on every cycle every square immediately reacts to the previous pattern formed by its neighbours – possibly by doing nothing.
- If two instances of a chess program (or instances of different chess programs) interact by playing chess in the same computer, their behaviour is typically no longer purely reactive. If they are any good they will often have to search among possible sequences of future moves to find a good next move.
- Each may do some searching in advance while waiting for the opponent’s next move.
- Then each instance is a VM with its own internal states and processes interacting richly, and a less rich interaction with the other VM is mediated by changes in the shared board state (represented by an abstract data-structure).

  For more on varieties of deliberation see:
  `http://www.cs.bham.ac.uk/research/projects/cosy/papers/#dp0604`

## Intentionality in a virtual machine

A running chess program (a VM) takes in information about the state of the board after the opponent moves, and then builds internal structures that it uses to represent the board and the chess pieces on it, and their relationships, including threats, opportunities, possible traps, etc.

- In particular it uses those representations in attempting to achieve its goals.

  So, unlike the interacting Conway patterns mentioned earlier, some of the patterns in the chess virtual machine are treated by the machine as representations, that refer to something.

- During deliberation, some created patterns will be treated as referring to non-existent but possible future board states, and as options for moves in those states.

  They are treated that way insofar as they are used in considering and evaluating possible future move sequences in order to choose a move which will either avoid defeat (if there is a threat) or which has a chance of leading to victory (check-mate against the opponent).

- In this case the chess VM, unlike the simplest interacting Conway patterns, exhibits intentionality: the ability to refer. (The programmer need not know what's going on.)

  Since the Conway mechanism is capable of implementing arbitrary Turing machines, it could in principle implement two interacting chess virtual machines, so there could be intentionality in virtual machines running on a Conway machine.

- The intentionality of chess VMs is relatively simple because they have relatively few types of goal, relatively few preferences, and their options for perceiving and acting are limited by being constrained to play chess:

  For a human-like robot the possibilities would be much richer, and a far more complex architecture would be required. See `http://www.cs.bham.ac.uk/research/projects/cogaff/03.html#200307`

## Emerging varieties of functionality

Computer scientists and engineers and AI/Robotics researchers have been learning to add more and more kinds of control, kinds of pattern, and ways of interpreting patterns of varying levels of abstraction.

- A simple machine may repeatedly take in some pattern and output a derived pattern,
  e.g. computing the solution to an arithmetical problem.
- More complex machines can take in a pattern and a derivation-specification (program) and output a derived pattern that depends on both.
- Other machines can continually receive inputs (e.g. from digitised sensors) and continually generate outputs (e.g. to digitally controlled motors).
- More sophisticated machines can
  – solve new problems by searching for new ways of relating inputs to outputs instead of using only fixed collections of instructions.
  – interpret some patterns as referring to the contents of the machine (using a somatic ontology) and others to independently existing external entities, events, processes (using an exosomatic ontology)
  – extend their ontologies and theories about the nature and interactions of external entities
  – perform tasks in parallel, coordinating them,
  – monitor and control some of their own operations – even interrupting, modulating, aborting, etc.
  – develop meta-semantic ontologies for representing and reasoning about thinking, planning, learning, communicating, motives, preferences, ...
  – acquire their own goals and preferences, extending self-modulation, autonomy, unpredictability, ...
  – develop new architectures which combine multiple concurrently active subsystems.
  – form societies, coalitions, partnerships ... etc.
- Biological evolution did all this and more, long before we started learning how to do it.

## Causal networks linking layered patterns

How can events in virtual machines be causes as well as effects, even causing physical changes?

The answer is

**through use of mechanisms that allow distinct patterns of states and sequences of patterns to be linked via strong constraints to other patterns of states and sequences of patterns (as in the ABCD/XY example, and the Conway machines, mentioned above).** (Some VMs may use probabilistic/stochastic constraints.)

What many people find hard to believe is that this can work for a virtual machine whose internal architecture allows for divisions of functionality corresponding to a host of functional divisions familiar in human minds, including

- interpreting physical structures or abstract patterns as referring to something (intentionality)
- generation of motives,
- selection of motives,
- adoption of plans or actions,
- perceiving things in the environment,
- introspecting perceptual structures and their changes,
- extending ontologies,
- forming generalisations,
- developing explanatory theories,
- making inferences,
- formulating questions,
- and many more.

## Supervenience of RVMs: a complex relation

- The two machines (the PM and a VM running on the PM) need not be isomorphic: they can have very different structures.
- There need not be any part of the PM that is isomorphic with the VM.
- Not only static parts and relations but also processes and causal relations, and even whole machines and their histories, can supervene on physical phenomena.
- The structure of a VM can change significantly (parts added and removed, and links between parts being added and removed) without structural changes occurring at the physical level –

  though the physical states of millions of components (switches, or neurones, or ...) may change as the (much simpler, more abstract) VM changes and causal interactions between its parts occur.
- **The mappings between PM components and VM components may be complex, subtle in kind, and constantly changing – even on a special-purpose PM supporting a particular kind of VM**
- In computers this is a result of decades of development

  of memory management mechanisms, paging systems, garbage collectors, device drivers, compilers, interpreters, interrupt handlers, network interfaces, shareable file-systems, distributed file-systems, analog to digital converters, interfacing standards, .... and many more.
- In animals this is a result of millions of years of development, of mechanisms not yet identified or understood.

## Types of VM content

RVMs are complex, but they don’t all have the same kind of complexity.

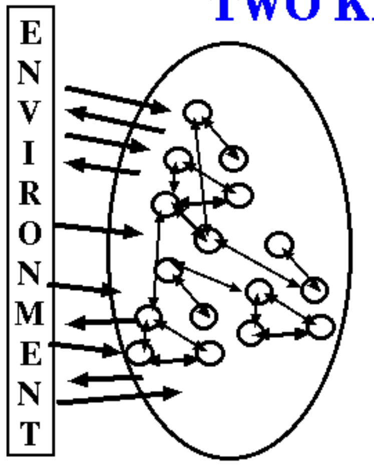
- Some are composed entirely of constantly changing multiply-interconnected entities (which may or may not map closely onto physical processing entities)

  Various forms of artificial neural nets take this form, where the patterns of change can differ, e.g. between continuous feedback loops, or regular “sweeps” of activation in different directions, or more complex activation patterns.

- Some include a mixture of rapidly, possibly continuously changing, components closely coupled with sensors and motors and other components that either change very slowly or persist in their original form after they are created, e.g.
  - internal information about changing input and output signals providing transient information about exactly what is happening at a particular time, or producing particular actions at particular times
  - More enduring information about what has happened, or is likely to happen or is intended to happen, or may be happening out of sight, etc.
  - Enduring generalisations about regularities in sensory-motor patterns (using a “somatic” ontology) or regularities in behaviours of things outside the individual’s body (using an “exosomatic” ontology).

# Varieties of dynamical system

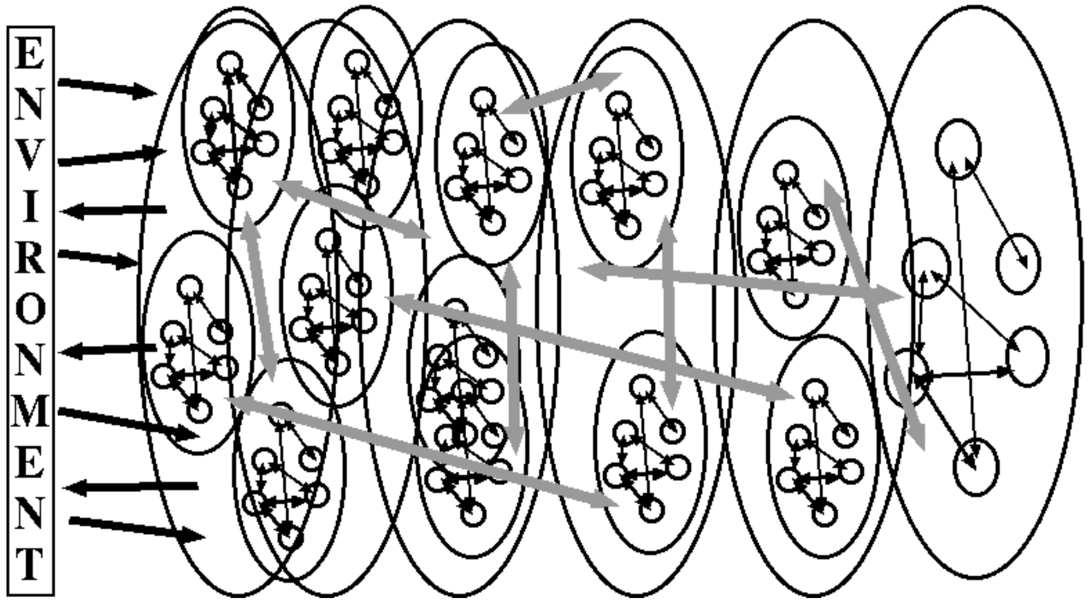
## TWO KINDS OF DYNAMICAL SYSTEM



One kind, on the left, is closely coupled with the environment through sensors and effectors.

The other kind, below, has many levels of abstraction and decoupling from the environment, including theorising, reasoning remembering, planning, predicting subsystems.

Evolution produced both kinds and many intermediate kinds.



**Semantic contents of sub-systems remote from the sensorymotor interface (on the right) could refer deep into the environment (on the left): e.g. theories of sub-atomic particles or other minds.**

## Varieties of supervenience

Several varieties of supervenience can be distinguished:

- **property supervenience**: (e.g. having a certain temperature supervenes on having molecules with a certain average kinetic energy);
- **pattern supervenience**: (e.g., supervenience of various horizontal, vertical and diagonal rows of dots on a rectangular array of dots, or the supervenience of a rotating square on changes in the pixel matrix of a computer screen);
- **mereological, or agglomeration, supervenience**: possession of some feature by a whole, arising from a summation of features of parts (e.g. supervenience of the centre of mass of a rock on the masses and locations of its parts, each with its own mass);
- **mathematical supervenience**: e.g. Euclidean geometry can be modelled in arithmetic, using Cartesian coordinates, and in that sense geometry supervenes on arithmetic.
- **mechanism supervenience**: supervenience of one machine on another: a set of interacting objects, states, events and processes supervenes on a lower level reality (e.g., supervenience of a running operating system on the computer hardware).

My topic is mechanism supervenience, relating RVMs to PMs – not the simple case of one property, or entity, relating to others, but a complex *ontology* (collection of diverse entities, events, processes, states, with many properties, relationships and causal interactions) relating to another ontology. Donald Davidson (“Mental Events”, 1970) described supervenience as a relation between properties or “respects”, whereas mechanism supervenience involves a relation between interacting parts and relations of complex ontology-instances, not just properties.

## The more general notion of supervenience

Supervenience is often described as a relation between **properties:** e.g. a person’s mental properties supervene on his physical properties (or “respects”).

‘[...] supervenience might be taken to mean that there cannot be two events alike in all physical respects but differing in some mental respects, or that an object cannot alter in some mental respect without altering in some physical respect.’

D. Davidson (1970), ‘Mental Events’, reprinted in: *Essays on Action and Events* (OUP, 1980).

**It’s better described as a relation between *ontologies* or complex, interacting, parts of ontology-instances (RVMs) not just properties.**

The cases we discuss involve not just one object with some (complex) property, but a collection of VM components enduring over time, changing their properties and relations, and interacting with one another: e.g. data-structures in a VM, or several interacting VMs, or thoughts, desires, intentions, emotions, or social and political processes, all interacting causally – the whole **system** supervenes.

A single object with a property that supervenes on some other property is just a very simple special case. We can generalise Davidson’s idea:

A functioning/working ontology supervenes on another if there cannot be a change in the first without a change in the second.

NOTE: the idea of “supervenience” goes back to G.E.Moore’s work on ethics. A useful introduction to some of the philosophical ideas is: Jaegwon Kim, *Supervenience and Mind: Selected philosophical essays*, 1993, CUP.

## Why were artificial VMs developed?

- **Many complexities** in the design and construction of VMs arise from **diverse engineering requirements** confronted by designers of complex information-processing systems.

  (The requirements emerged gradually over the last 60 years – and there are more to come!)

- Different researchers and engineers (hardware and software) work on different subsets of the requirements and designs.

  (Partly explained later.)

- The total picture is rich and deep and, as far as I know, has never been completely documented (my own work is still only partial).

- Some of the problems are concerned with

  (a) supporting new low-level operations (e.g. string matching, floating point arithmetic, lists);

  (b) enabling efficient use to be made of limited resources (e.g. limited memory, limited computing power) by multiple, changing, processes running concurrently;

  (c) allowing file systems and other resources to be shared between computers;

  (d) enabling several physical devices to work together including interface devices, such as keyboards, screens, microphones, network interfaces, ...

- Some of the problems are concerned with allowing software running in the computer to monitor and control things happening in the computer,

  E.g. a game program monitoring its performance in order to improve itself.

## MAIN CONJECTURE

- Biological evolution "discovered" many of the problems long before human engineers,
- and produced solutions involving complex VMs whose rationale and operation have not yet been understood,
- using PMs of staggering complexity – supporting extremely complex VMs (and some simpler VMs)
- Some of the drivers of this process may have been
  - The need for concurrent processing of multiple sensory inputs and motor outputs;
  - the need for motive generators and "alarm" systems to be able to operate in parallel with actions based on current motives;
  - the need for animal vision systems to perceive complex changing/interacting 3-D structures in the environment;
  - development of deliberative mechanisms for exploring multiple multi-step possible futures, in order to find a plan or make a prediction;
  - development of self-monitoring, self-modulating control systems.

Often the use of a virtual machine makes kinds of self monitoring and self-modulation possible that would otherwise be impossible, because of the complexity of detail at the physical level.

The separation of levels also facilitates exploration of different designs, by decomposing the problems into simpler problems.

This often involves the use of "platform virtual machines".

## Steve Burbeck's web site

Note added 9 Oct 2009

There is extraordinary overlap between the ideas presented here and the ideas on Steve Burbeck's web site, though we approach the ideas from (nearly) opposite directions.

**He writes:**

"Nature offers many familiar examples of emergence, and the Internet is creating more.

The following examples of emergent systems in nature illustrate the kinds of feedback between individual elements of natural systems that give rise to surprising ordered behavior. They also illustrate the trade off between the number of elements involved in the emergent system and the complexity of their individual interactions. The more complex the interactions between elements, the fewer elements are needed for a higher-level phenomenon to emerge. Hurricanes and sand dunes form from vast numbers of very simple elements whereas even small groups of birds can exhibit flocking behavior.

What is the lesson for multicellular computing? It is that networks of computers support many sorts of emergent meta-level behavior because computers interact in far more complex ways than air and water molecules or particles of sand and are far more numerous than even the largest flocks of birds. Some of this emergent behavior is desirable and/or intentional, and some (bugs, computer viruses, dangerous botnets, and cyber-warfare) are not."
`http://evolutionofcomputing.org/Multicellular/Emergence.html`

"All living organisms, from single cells in pond water to humans, survive by constantly processing information about threats and opportunities in the world around them. For example, single-cell E-coli bacteria have a sophisticated chemical sensor patch on one end that processes several different aspects of its environment and biases its movement toward attractant and away from repellent chemicals. At a cellular level, the information processing machinery of life is a complex network of thousands of genes and gene-expression control pathways that dynamically adapt the cell's function to its environment."
`http://evolutionofcomputing.org/Multicellular/BiologicalInformationProcessing.html`

# The issue of objectivity

(Slide hastily added – with thanks to Elanor Taylor - 13 Sep 2009)

A talk by Elanor Taylor (University of North Carolina) on “Real Patterns” at the Metaphysics of Science Conference, raised a question about descriptions of complex systems using concepts/formalisms that make those complex systems intelligible, or manageable (e.g. for prediction and explanation) reflect “objective” facts about those systems, or merely indicate something about what the user of the description finds convenient.

It occurred to me after hearing the talk that even if the claim

(a) that a description of type D is useful for thinker T to understand object O

is in part a claim regarding a “subjective” fact about T, the claim

(b) that using a description of type D (e.g. referring to virtual machine phenomena) is a requirement for the system O to be able to understand itself in ways that are needed for various kinds of learning, development and self control

could be an objective fact about things of type O.

I believe that this kind of objective fact about intelligent systems is part of what I am referring to in this presentation. (To be revised and clarified.)

Note added: 16 Sep 2009. See also “The Future of Psychology: Connecting Mind to Brain”, by Lisa Feldman Barrett, *Perspectives on Psychological Science* Volume 4, Issue 4. APA, Sept 2009.

## The unnoticed relevance to Cognitive Science

Many cognitive scientists have heard about virtual machines (VMs) contrasted with physical machines (PMs)

Also philosophers, psychologists, neuroscientists, AI researchers, computer scientists, software engineers.

But there are differing views about how to define “virtual machine” and about the relationships between the two sorts of machine, e.g.

– Is every VM **identical** with some PM that implements or realises it?
– Can events in VMs be **causes,** or only **effect**s (only “epiphenomenal”)?
– Do VMs exist or are they just **a convenient fiction**, a manner of speaking?

(Dennett: a “centre of narrative gravity”.)

If it is agreed that virtual machines and their contents can be causes (as implied by common answers to the votes at the beginning of this presentation), then saying that they don’t exist, or that they are just convenient fictions seems pointless.

Is poverty just a convenient fiction???

Even outstanding thinkers who know about VMs sometimes forget them.

E.g. Newell and Simon’s “Physical symbol system” hypothesis is wrong: they are really talking about physically implemented symbol systems in virtual machines.

(The symbols manipulated in a Lisp program are not physical objects.)

## The invisibility of VMs

All academics and researchers (well, almost all) now **use** VMs every day.

But for some reason they don’t notice their existence – they have not learnt to think about different sorts of machines.

However, a small subset of users have also designed, built, debugged, sold and maintained VMs.

Their understanding of some of the varieties of VM, how they work, what they are useful for, etc., is much deeper than the understanding of most users, but it is not usually made explicit.

Most electronic and software engineers and computer scientists lack the philosophical expertise required to make such knowledge explicit and to characterise the concepts used.

The “big picture” probably is not yet understood in all its depth and generality by anyone.

(Not by me!)

## The problems include:

Implications for philosophy, neuroscience, developmental psychology, developmental biology, genetics, information engineering in general, and AI/Robotics – viewed as both science and engineering.

Examples

- Running VMs can be disconnected from input-output interfaces, permanently or temporarily, making conventional means of scientific observation impossible.

  E.g. some problem-solving subsystem may mostly run without connections to sensor or motors and only occasionally produce some result that is transferred to other subsystems. These could justify the cost of the disconnected subsystem.

- Output mechanisms may be incapable of reporting VM processes (e.g. not enough bandwidth).
- Systems using a VM interface for self-monitoring and self control will have timing delays and other discrepancies between processes at the different levels.
- This can lead to misperceptions, during both normal and abnormal operation:

  e.g. Libet effects, phantom limbs, hallucinations, proof-reading errors, psychotic phenomena, mis-descriptions of motivations, etc.

- Much more may be taken in and processed than a self-reporting VM can detect, unless redirected.
- Self-reports in laboratory experiments may report only what the VM architecture makes accessible to the reporting mechanism. (Here be qualia?)
- We need much deeper analyses of varieties of VMs, and different forms of representation, and their possible uses in different subsystems within minds (of animals or machines).

## Major problem:

**Are there types of virtual machine**

**that we have not yet thought about**

**that are required for explaining/replicating**

**human/animal competences that are currently**

**beyond the state of the art in AI?**

To answer that we have to understand the requirements.

Understanding the requirements requires us to look very closely and analytically at many of the competences shown by humans (of all ages) and other animals both when interacting with the physical environment and when doing other things (e.g. proving theorems in their heads).

## Human and animal vision requires virtual machines

Much research on natural and artificial vision assumes that vision is concerned with recognition of objects.

That is a deep error: recognition of objects as normally construed is not required for seeing those objects.

More importantly, human vision is often perception of processes.

Consider what happens when you turn a corner in a busy part of a large town.

There are vast and rapid changes in what is perceived – this cannot involve comparable rapid physical reorganisation of neuronal structures: e.g. rearranging neurons to form a representation of a bus.

**But it could involve rapid reorganisation of the contents of an information-processing virtual machine.**

For examples see the pictures here

`http://www.cs.bham.ac.uk/research/projects/cogaff/misc/multipic-challenge.pdf`

## Two major kinds of running VM

There are various types of VM with different capabilities and different roles in larger systems.

Two important classes are:

### Specific function/application VMs

E.g. a chess-playing VM, a word-processor, an email front-end.

### Platform VMs

Capable of supporting a variety of other VMs

E.g. Operating systems (e.g. Linux, Unix, OSX, Windows, VMS, ...)

Language VMs (e.g. Lisp VM. Prolog VM, Java VM, Pop-11 VM)

NB: we are talking about running instances, not abstract specifications.

It seems that there are some biological platform VMs, which get extended in various ways.

An important research problem is to investigate the various types, their functions, which species use them, how they evolved, how they develop in individuals, etc.

One type of extension of biological VM capability involves learning new languages and notations.

E.g. learning a new programming language.

## Natural and Artificial Platform VMs

- Platform VMs designed by human engineers provide a basis for constructing new VMs that are implemented in terms of the facilities provided by the platform VM:
  But most such extensions do not arise spontaneously.

  The operating system on your PC can be left running for many years and over time you and others may design and install different software packages, or attach new hardware devices along with device drivers for them.

  If left to itself, a normal operating system will just go on forever waiting for instructions, without initiating any major extensions, though some of them are designed to detect the availability of new versions of old subsystems and download and install them.

- Biological platform VMs, however, are not extended by external designers:
  They have to build and extend themselves

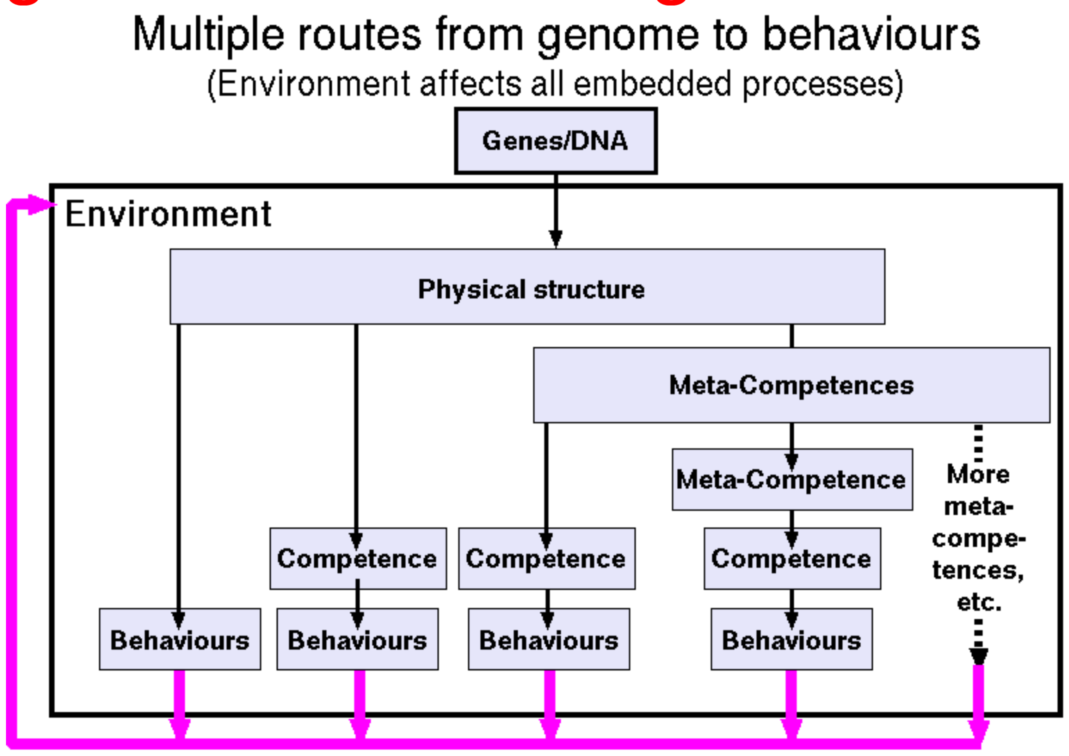
  (partly on the basis of external influences from both the physical environment and conspecifics).

  The requirements to support this have never, as far as I know, been identified.

  The problem is not addressed by research in developmental psychology on which concepts, knowledge or competences are innate.
  Some of the requirements for a “well-designed child” were discussed by McCarthy in this paper `http://www-formal.stanford.edu/jmc/child.html` (written 1996 and published in 2008).

- In humans, biological platform VMs seem to grow throughout infancy and childhood, and for some people (e.g. academics) continue being extended until late in life.
  The extensions support new competences of many kinds, including manipulative and perceptual competences, linguistic, musical and artistic competences, mathematical competences, extended ontologies, new planning and reasoning capabilities, new forms of motivation, new control regimes,

# Biological VMs have to grow themselves



Humans and some other species require layered construction of competences and meta-competences in a layered architecture.

**Not core competences as normally construed: core architecture-building competences, metacompetences, ...**

Work done with Jackie Chappell (IJUC, 2007) – Chris Miall helped with diagram.
"Natural and artificial meta-configured altricial information-processing systems"
`http://www.cs.bham.ac.uk/research/projects/cosy/papers/#tr0609,`

**Corollary: we can expect major variations in the individual VMs of humans (and other altricial species). Compare differences of language, culture, level of education, results of abuse, etc. ....**

## Warning from biology:

Don’t expect a **sharp** divide between systems using only physical machines and those also using virtual machines: biology provides intermediate cases for most distinctions,

e.g. is a homeostatic control loop a VM?

Neither biology nor engineering needs to respect philosophers’ desires for simple classification schemes:

there tend to be many small discontinuities rather than just a few big ones.

But differences across multiple steps can be huge.

# RVMs with temporarily or partly ‘decoupled’ components

## A challenge for philosophy of science and psychological methodology.

- “Decoupled” subsystems may exist and process information, even though they have no connection with sensors or motors.

  They may have biological value because they occasionally connect with other subsystems.

- Theories referring to them cannot be decisively proved or refuted using behavioural experiments.

  Compare Lakatos on methodology of scientific research programmes

- For instance, a machine playing games of chess with itself, or investigating mathematical theorems, e.g. in number theory.
- Some complex systems “express” some of what is going on in their VM states and processes through externally visible behaviours.

  However, it is also possible for internal VM processes to have a richness that cannot be expressed externally using the available bandwidth for effectors.

  Likewise sensor data may merely introduce minor perturbations in what is a rich and complex ongoing internal process.

## This transforms the requirements for rational discussion of some old philosophical problems about the relationship between mind and body:

E.g. some mental processes need have no behavioural manifestations, though they might, in principle, be detected using ‘decompiling’ techniques with non-invasive internal physical monitoring.
(This may be impossible in practice, or at best only a matter of partly testable conjecture.
Compare theoretical physics.)

## Problem: Supervenience and Causation

Mental states and processes are said to supervene on physical ones.

But there are many problems about that relationship:

Can mental process cause physical processes?

(Sometimes called “downward causation”.

How could something happening in a mind produce a change in a physical brain?

**If previous physical states and processes suffice to explain physical states and processes that exist at any time, how can mental ones have any effect?**

How could your decision to come here make you come here – don’t physical causes (in your brain and in your environment) suffice to make you come?

If the physical processes suffice, e.g. because the physical world is (mostly) deterministic and causally closed, how could anything else play a role?

I think we use wrong models of causation, when we think about these things.

THESE SLIDES ARE STILL IN A MESS AND NEED MUCH MORE WORK

(especially sorting out the issues about causation.)

## Explaining what's going on in VMs requires a new analysis of the notion of **causation**

The relationship between objects, states, events and processes in virtual machines and in underlying implementation machines is a tangled network of causal interactions.

Software engineers have an intuitive understanding of it, but are not good at philosophical analysis.

Philosophers mostly ignore the variety of complex mappings between RVMs and PMs when discussing causation and when discussing supervenience,

Even though most of them now use multi-process VMs daily for their work.

Explaining how virtual machines and physical machines are related requires a deep analysis of causation that shows how the same thing can be caused in two very different ways, by causes operating at different levels of abstraction.

Explaining what 'cause' means is one of the hardest problems in philosophy.

For a summary explanation of two kinds of causation (Humean and Kantian) and the relevance of both kinds to understanding cognition in humans and other animals see:
`http://www.cs.bham.ac.uk/research/projects/cogaff/talks/#wonac`

## We often assume multiple varieties of causation

A person drives home one night after drinking with friends in a pub.

As he goes round a bend in the road he skids sideways into an oncoming car and and the driver in the other car dies.

In court, the following facts emerge.

- The driver had exceeded the recommended alcohol limit for driving, but had often had the extra glass and then driven home on that route without anything going wrong.
- There had earlier been some rain followed by a sharp drop in temperature, as a result of which the road was unusually icy.
- The car was due for its MOT test, and he had been given two dates for the test, one before the accident and one after. He chose the later date. Had he taken the earlier date worn tires would have been detected and replaced with tires that could have gripped ice better.
- There had been complaints that the camber on the road was not steep enough for a curve so sharp, though in normal weather it was acceptable.
- The driver was going slightly faster than normal because he had been called home to help a neighbour who had had a bad fall.
- A few minutes after the accident the temperature rose in a warm breeze and the ice on the road melted.

What caused the death of the other driver?

## Biological Virtual Machines

Biological VMs may be required in plants and animals

– at cellular levels (e.g. control of metabolism)

– in various physiological control mechanisms (e.g. control of balance, running)

– in information-processing architectures of varying complexity
(e.g. various kinds of perception, learning, motivation, decision-making, ...)

Some biological VMs are pre-specified in the genome, (“precocial VMs”) while others (“altricial VMs”) are constructed during individual development – in some cases partly under the control of the environment (epigenesis of virtual machines).

[Chappell & Sloman
“Natural and artificial meta-configured altricial information-processing systems”, IJUC 2007]

**NOTE**

The role of the environment in “controlling” both evolution and individual development implies that the nature of the environment needs to be much more an object of study in developmental psychology and AI than it normally is. (Cf. Ulric Neisser, 1976)

## The inside-outside distinction blurs

Often the boundary between machine and environment is different for different sub-systems of the machine.

(Explained later)

The physical implementations of some VMs can cross superficial PM boundaries – e.g. VMs that refer to remote, or past, or future entities or events may use external intermediaries to help “tether” (not “ground”) the semantic content.

(P.F.Strawson *Individuals*, 1959)

Different parts of the machine, e.g. different sensors and effectors, may interact with different parts of the environment concurrently.

The machine may treat parts of itself as parts of the environment (during self-monitoring), and parts of the environment as parts of itself (e.g. tools, external memory aids).

See Sloman 1978, chapter 6

`http://www.cs.bham.ac.uk/research/projects/cogaff/crp/chap6.html`

## Physics also deals with different levels of reality

- The "observable" level with which common sense, engineering, and much of physics has been concerned for thousands of years:
  - levers, balls, pulleys, gears, fluids, and many mechanical and hydraulic devices using forces produced by visible objects.
- Unobservable extensions
  - sub-atomic particles and invisible forces and force fields, e.g. gravity, electrical and magnetic forces.
- Quantum mechanical extensions
  - many things which appear to be inconsistent with the previous ontology of physics

Between the first two levels we find the ontology of chemistry, which includes many varieties of chemical compounds, chemical events, processes, transformations, causal interactions.

The chemical entities, states, processes, causal interactions are normally assumed to be "fully implemented" (fully grounded) in physics.

We don't know how many more levels future physicists will discover.

IS THERE A 'BOTTOM' LEVEL?

# More on virtual machines (VMs/NPDMs)

Recapitulation:

A virtual machine (VM) is a NPDM, a machine containing causally interacting components with changing properties and relationships, whose best description uses concepts that cannot be defined in terms of concepts of the physical sciences.

This implies (non-obviously) that there are states and processes in VMs that cannot be measured or detected using the techniques of the physical sciences (e.g. physics, chemistry), though in order to exist and work, a VM needs to be **implemented** in a physical machine.

An example is a collection of running computer programs doing things like checking spelling, playing chess, sorting email, computing statistics, etc.

**"Incorrect spelling" cannot be defined in terms of concepts of physics, and instances of correct and incorrect spelling cannot be distinguished by physical measuring devices.**

However, a second virtual machine that is closely coupled with the first, might be able to detect that the first is doing those non-physical things.

A socio-economic system is a more abstract and complex form of virtual machine: "economic inflation" and "recession" cannot be defined in terms of concepts of physics

Mental states and processes in humans and other animals can be regarded as states and processes in virtual machines, implemented in brains.

## A possible objection: only one CPU?

Some will object that when we think multiple processes run in parallel on a single-CPU computer, interacting with one another while they run, we are mistaken because only one process can run on the CPU at a time, so there is always only one process running.

This ignores the important role of memory mechanisms in computers.

- Different software processes have different regions of memory allocated to them, which endure in parallel. So the processes implemented in them endure in parallel, and a passive process can affect an active one that reads some of its memory.

Moreover

- It is possible to implement an operating system on a multi-cpu machine, so that instead of its processes sharing only one CPU they share two or more.
- In the limiting case there could be as many CPUs as processes that are running.
- The differences between these different implementations imply that
  how many CPUs share the burden of running the processes is a contingent feature of the implementation of the collection of processes and does not alter the fact that there can be multiple processes running in a single-cpu machine.

A technical point: software interrupt handlers connected to constantly on physical devices, e.g. keyboard and mouse interfaces, video cameras, etc., can depend on some processes constantly “watching” the environment even when they don’t have control of the CPU,

In virtual memory systems, and systems using “garbage collection” things are more complex than suggested here: the mappings between VM memory and PM memory keep changing.

## Multiple layers of virtual machinery

There can be more than two layers of machinery.

It should be clear by now that we don't simply have

- Physical machinery
- Virtual machinery

However, just as some physical machines (e.g. modern computers) have a kind of generality that enables them to support many different virtual machines

(e.g. the same computer may be able to run different operating systems
– Windows, or Linux, or MacOS, or ....)

so do some platform VMS have a kind of generality that enables them to support many different "higher level" VMs running on them

(e.g. the same operating system VM may be able to run many different applications, that do very different things, – window managers, word processors, mail systems, spelling correctors, spreadsheets, compilers, games, internet browsers, CAD packages, virtual worlds, chat software, etc. ....)

**It is possible for one multi-purpose VM to support another multi-purpose VM, which supports additional VMs.**

**So VMs may be layered:**

**VM1 supports VM2 supports VM3 supports VM4, etc.**

The layers can branch, and also be circular, e.g. if VM1 includes a component that invokes a component in a higher level VM$_k$, which is implemented in VM1.

## Not to be confused with control hierarchies

Layered virtual machines are not the same as

- Hierarchical control systems
- Brooks' "subsumption architecture"

Here the different layers implement different functions for the whole system, and can be turned on and off independently (mostly).

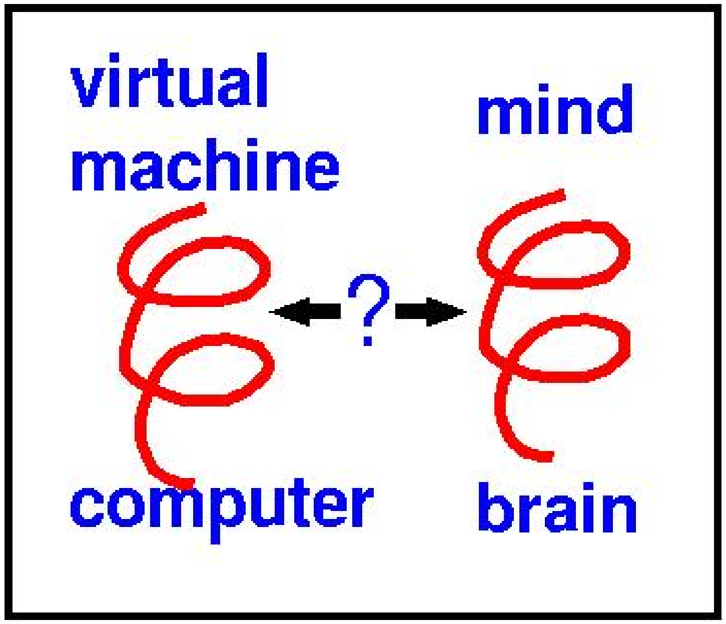
In contrast, a higher level VM provides functionality that is implemented in lower levels: the lower levels don't provide different competences that could be added or removed, e.g. damping.

Removing a lower level VM layer makes the whole thing collapse, unless it replaced by an equivalent lower level VM (e.g. a different operating system with similar functionality).

No time to explain the difference.

## ‘Emergence’ need not be a bad word

People who have noticed the need for pluralist ontologies often talk about ‘emergent’ phenomena.
But the word has a bad reputation, associated with mysticism, vitalist theories, sloppy thinking, wishful thinking, etc.



If we look closely at the kinds of ‘emergence’ found in virtual machines in computers, where we know a lot about how they work (because we designed them and can debug them, etc), then we’ll be better able to go on to try to understand the more complex and obscure cases, e.g. mind/brain relations.

Virtual machine emergence adds to our ontology: the new entities are not definable simply as patterns or agglomerations in physical objects (they are not like ocean waves).

My claim is that engineers discussing implementation of VMs in computers and philosophers discussing supervenience of minds on brains are talking about the same ‘emergence’ relationship – involving VMs implemented (ultimately) in physical machines.

**NB. It is not just a metaphor: both are examples of the same type.**

## Why virtual machines are important in engineering

ONE reason is: They provide “vertical” separation of concerns

Contrast “horizontal” separation: different kinds of functionality that can be added or removed independently, e.g email, web browsing, various games, spreadsheets – or the parts of such subsystems.

- Both horizontal decomposition and vertical decomposition involve modularity that allows different designers to work on different tasks.
- But vertical decomposition involves layers of necessary support.
- VMs reduce combinatorial complexity for system designers
- They can also reduce the complexity of the task of self-monitoring and self control in an intelligent system.
- Evolution seems to have got there first
- That includes development of meta-semantic competences for self-monitoring, self-debugging, etc.
- It can also lead to both incomplete self knowledge and to errors in self analysis, etc.

See also The Well-Designed Young Mathematician, *AI Journal*, December 2008
`http://www.cs.bham.ac.uk/research/projects/cosy/papers/#tr0807`

## Concurrent, interacting virtual machines sharing a substrate

In a multi-processing computer the complexity would be totally unmanageable if software designers had to think about all the possible sequences of machine instructions.

Instead we use a VM substrate for handling multiple processes, with mechanisms for

- memory management
- context switching
- scheduling
- handling privileges and access rights, etc.
- filestore management
- various device drivers
- networking
- and in some cases use of multiple CPUs

**Some VMs implement a particular kind of functionality (e.g. a chess playing VM) whereas others provide a platform of resources that can be combined in different ways to support multiple kinds of functionality (e.g. operating systems, and various kinds of software development toolkits).**

**How much of that did evolution discover?**

## Self-monitoring and virtual machines

Systems dealing with complex changing circumstances and needs may need to monitor themselves, and use the results of such monitoring in taking high level control decisions.

E.g. which high priority task to select for action.

Using a high level virtual machine as the control interface may make a very complex system much more controllable: only relatively few high level factors are involved in running the system, compared with monitoring and driving every little sub-process, e.g. at the transistor level.

The history of computer science and software engineering since around 1950 shows how human engineers introduced more and more abstract and powerful virtual machines to help them design, implement, test debug, and run very complex systems.

When this happens the human designers of high level systems need to know less and less about the details of what happens when their programs run.

Making sure that high level designs produce appropriate low level processes is a separate task, e.g. for people writing compilers, device drivers, etc. Perhaps evolution produced a similar “division of labour”?

Similarly, biological virtual machines monitoring themselves would be aware of only a tiny subset of what is really going on and would have over-simplified information.

**THAT CAN LEAD TO DISASTERS, BUT MOSTLY DOES NOT.**

## Robot philosophers

The simplifications in self-monitoring VMs could lead robot-philosophers to produce confused philosophical theories about the mind-body relationship – e.g. theories about "qualia".

Intelligent robots will start thinking about these issues.

As science fiction writers have already pointed out, they may become as muddled as human philosophers.

So to protect our future robots from muddled thinking, we may have to teach them philosophy!

BUT WE HAD BETTER DEVELOP GOOD PHILOSOPHICAL THEORIES FIRST!

---

The proposal that a virtual machine is used as part of the control system goes further than the suggestion that a robot builds a high level model of itself, e.g. as proposed by Owen Holland in

`http://cswww.essex.ac.uk/staff/owen/adventure.ppt`

For more on robots becoming philosophers of different sorts see

Why Some Machines May Need Qualia and How They Can Have Them:
Including a Demanding New Turing Test for Robot Philosophers

`http://www.cs.bham.ac.uk/research/projects/cosy/papers/#tr0705`
Paper for AAAI Fall Symposium, Washington, 2007

## Biological evolution probably “discovered” all this (and more) first

Even though biological evolution does not need an intelligent designer to be involved, there are strategies that could be useful for evolution for the same reason as they are useful for designers.

That includes the use of virtual machines, for example.

- More precisely – it could turn out that a modification of a design for an organism that gives it a kind of self-understanding its competitors lack, could make it more successful.
- E.g. it may monitor its own reasoning, planning, and learning processes (at a certain level of abstraction) and find ways to improve them.
- If those improved procedures can also be taught, the benefits need not be rediscovered by chance.

## Why the same considerations are relevant to biology

Conjecture: biological evolution “discovered” long ago that separating a virtual machine level from the physical level made it possible to use the VM as a platform on which variants could be explored and good ones chosen,

e.g. different behaviours, or different control mechanisms, different mechanisms for choosing goals or planning actions, or different mechanisms for learning things.

- Long before that, the usefulness of “horizontal” modularity had already been discovered, with different neural or other control subsystems coexisting and controlling different body parts, or producing different behaviours, e.g. eating, walking, breathing, circulating blood, repairing damaged tissue.
- But developing new parts with specific functions is different from developing new behaviours for the whole organism.
- If each new behaviour has to be implemented in terms of low level states of muscles and sensors that could be very restrictive, making things hard to change.
- But if a VM layer is available on which different control regimes could be implemented, the different regimes will have much simpler specifications.
- This allows one genome to support multiple possible development trajectories, depending on environment (as in altricial species).

  Conjecture: this allows common functionality to exist following different trajectories (in different individuals with that genome) e.g. doing mathematics or physics in English or Chinese?

# A first draft ontology for architectural components

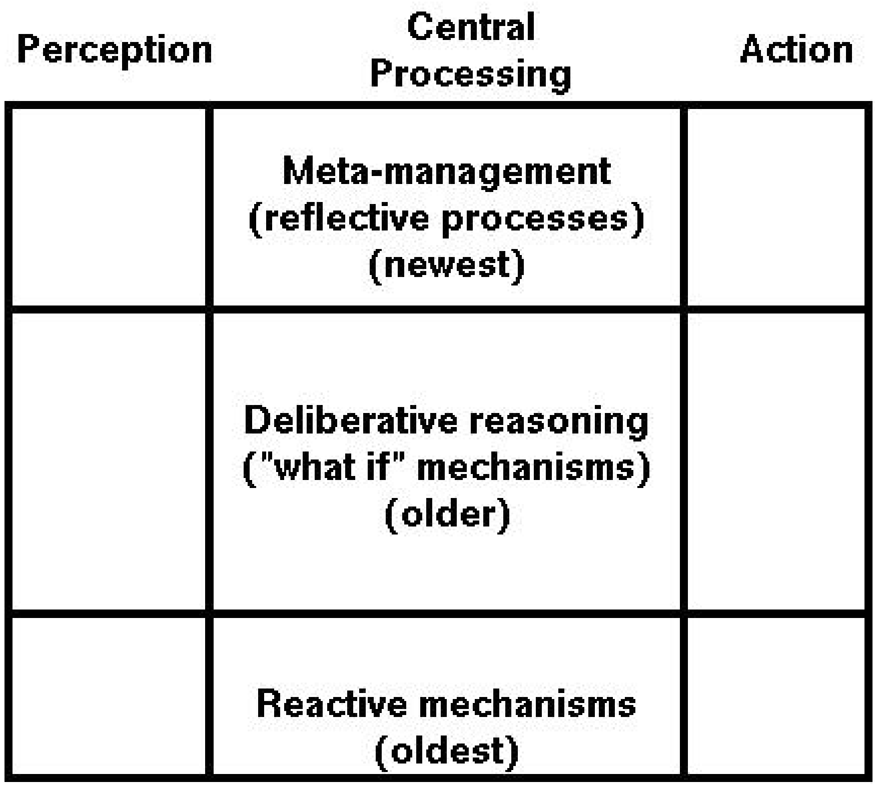
## THE COGAFF ARCHITECTURE SCHEMA

For now let’s pretend we understand the labels in the diagram.

On that assumption the diagram defines a space of possible information-processing architectures for integrated agents, depending on what is in the various boxes and how the components are connected, and what their functions are.

So if we can agree on what the types of layers are, and on what the divisions between perceptual, central and motor systems are, we have a language for specifying functional subdivisions of a large collection of possible architectures, ....

even if all the divisions are partly blurred or the categories overlap.


Perception
Central Processing
Action
Meta-management (reflective processes) (newest)
Deliberative reasoning ("what if" mechanisms) (older)
Reactive mechanisms (oldest)

Note: Marvin Minsky’s book *The emotion machine* uses finer-grained horizontal division (six layers). There’s largely because he divides some of these cogaff categories into sub-categories, e.g. different sorts of reactive mechanisms, different sorts of reflective mechanisms.

## Logical topographies generated by architectures

Why should philosophers care about information processing VM architectures?

One reason: because each architecture generates a space of relevant concepts for describing states and processes.

Compare: A theory of the architecture of matter, allows us to generate families of chemical elements, as in the Periodic Table of the elements.

Similarly: for each architecture, by considering the states and transitions it allows (possibly in combination with various environment environmental sttes and processes) we can generate a "logical topography" for possible concepts describing minds – of that sort.

> Then various cultures, and various individuals, can impose different "Logical geographies" (Ryle) on that objective logical topography
> `http://www.cs.bham.ac.uk/research/projects/cosy/papers/#dp0703`

In principle, the physical architecture can also be used to generate such ontologies.

But the number and variety of possible variations at the neuronal or atomic level will be astronomically large and probably useless for most ordinary practical purposes.

Architecture-based concepts at a VM level are more likely to be useful, not only for scientists and others trying to understand and communicate with such systems, but even for the ability of such individuals to understand themselves.

But the evaluation of competing theories of the underlying architecture can take a long time (a long research programme). (Lakatos)

## Is this philosophy or science?

Actually I don’t care.

More importantly: How can these ideas be evaluated?

The answer is complex, but basically it amounts to the idea of Lakatos:

It takes time (maybe decades) to decide whether a research programme is

- Progressive

  or
- Degenerative

## MORE TO BE SAID BUT NO MORE TIME

**Summary:**

The idea of a virtual machine (or NPDM) is deep, full of subtleties and of great philosophical significance, challenging philosophical theories of mind, of causation, and of what exists.

The use of virtual machines has been of profound importance in engineering in the last half century, even though most of the people most closely involved have not noticed the wider significance of what they were doing –

> especially the benefits of vertical separation of concerns, and the complexity of what has to be done to make all this work.

Biological evolution appears to have "discovered" both the problems and this type of solution long before we did, even long before humans existed.

Despite the benefits, the use of virtual machines can bring problems and some of those problems may afflict future intelligent machines that are able to think about themselves.

See also `http://www.cs.bham.ac.uk/research/projects/cogaff/talks/#virt`

---

There are lots more slides and more on the web

> Give to google: "Aaron Sloman" talks
>
> My talks page has several related presentations.

## Further reading: Background to these slides

For many years, like many other scientists, engineers and philosophers, I have been writing and talking about "information-processing" systems, mechanisms, architectures, models and explanations, e.g.:

My 1978 book *The Computer Revolution in Philosophy*, now online here:
`http://www.cs.bham.ac.uk/research/cogaff/crp/` (especially chapters 6 and 10)

A. Sloman, (1993) 'The mind as a control system,' in *Philosophy and the Cognitive Sciences,* Cambridge University Press, Eds. C. Hookway & D. Peterson, pp. 69–110.
Online here: `http://www.cs.bham.ac.uk/research/cogaff/`

Since the word "information" and the phrase "information-processing" are both widely used in the sense in which I was using them, I presumed that I did not need to explain what I meant. Alas I was naively mistaken:

- Not everyone agrees with many things now often taken as obvious, for instance that all organisms process information.
- Some people think that "information-processing" refers to the manipulation of bit patterns in computers.
- Not everyone believes information can cause things to happen.
- Some people think that talk of "information-processing" involves unfounded assumptions about the use of representations.
- There is much confusion about what "computation" means, what its relation to information is, and whether organisms in general or brains in particular do it or need to do it.
- Some of the confusion is caused by conceptual unclarity about virtual machines, and blindness to their ubiquity.

## Further Reading

A very stimulating and thought provoking book overlapping with a lot of this presentation is

George B. Dyson *Darwin among the machines: The Evolution Of Global Intelligence* 1997, Addison-Wesley.

A debate about VMs. (The paper by Densmore and Dennett is referred to several times above).

Shannon Densmore & Daniel Dennett, "The Virtues of Virtual Machines", *Philosophy and Phenomenological Research*, 59, 3, Sep, 1999, pp. 747–761, `http://www.jstor.org/stable/i345616`,

P. M. Churchland, "Densmore and Dennett on Virtual Machines and Consciousness", *Philosophy and Phenomenological Research*, 59, 3, September, 1999, pp. 763–767, `http://www.jstor.org/stable/2653794`,

Papers and presentations on the Cognition and Affect & CoSy web sites expand on these issues, e.g.

- A. Sloman & R.L. Chrisley, (2003),
  Virtual machines and consciousness, in *Journal of Consciousness Studies*, 10, 4-5, pp. 113–172,
  `http://www.cs.bham.ac.uk/research/cogaff/03.html#200302`
- A. Sloman, R.L. Chrisley & M. Scheutz,
  The Architectural Basis of Affective States and Processes, in *Who Needs Emotions?: The Brain Meets the Robot,* Eds. M. Arbib & J-M. Fellous, Oxford University Press, Oxford, New York, 2005.
  `http://www.cs.bham.ac.uk/research/cogaff/03.html#200305`
- A. Sloman and R. L. Chrisley,
  More things than are dreamt of in your biology: Information-processing in biologically-inspired robots, *Cognitive Systems Research*, 6, 2, pp 145–174, 2005,
  `http://www.cs.bham.ac.uk/research/cogaff/04.html#cogsys`
- A. Sloman
  The well designed young mathematician. In *Artificial Intelligence* (2008 In Press.)
  `http://www.cs.bham.ac.uk/research/projects/cosy/papers/#tr0807`
- "What's information?"
  `http://www.cs.bham.ac.uk/research/projects/cogaff/misc/whats-information.html`
- Presentations `http://www.cs.bham.ac.uk/research/projects/cogaff/talks/`

M. A. Boden, 2006, *Mind As Machine: A history of Cognitive Science* (2 Vols), Oxford University Press

There are many other books in philosophy of mind and cognitive science.

## For more on all this

Some computer scientists and AI researchers have appreciated the importance of these ideas, and are investigating ways of giving machines more self awareness, in order to make them more intelligent.

John McCarthy, “Making robots conscious of their mental states”.
`http://www-formal.stanford.edu/jmc/consciousness.html`

John L. Pollock (a computationally informed philosopher)
“What Am I? Virtual machines and the mind/body problem”, *Philosophy and Phenomenological Research*, 2008, 76, 2, pp. 237–309,
`http://philsci-archive.pitt.edu/archive/00003341`
Also work by Dave Clark at MIT on ‘The knowledge layer’ in intelligent self-monitoring networks.
Arnold Trehub’s 1991 book: *The Cognitive Brain*:
`http://www.people.umass.edu/trehub/`

Cognition and Affect Project and CoSy Project papers and talks:
`http://www.cs.bham.ac.uk/research/cogaff/`
`http://www.cs.bham.ac.uk/research/cogaff/talks/`
`http://www.cs.bham.ac.uk/research/projects/cosy/papers/`

The Tutorial presentation by Matthias Scheutz and myself on Philosophy of AI at IJCAI’01.
`http://www.cs.bham.ac.uk/research/cogaff/talks/#talk5`

Collaborative work with Jackie Chappell on cognitive epigenesis
Jackie Chappell and Aaron Sloman, ‘Natural and artificial meta-configured altricial information-processing systems,’ in *International Journal of Unconventional Computing*, 3, 3, pp. 211–239,
`http://www.cs.bham.ac.uk/research/projects/cosy/papers/#tr0609`

# Previous related Presentations

- 10-12 November 2008: Workshop on Philosophy and Engineering (WPE'08)
  Royal Academy of Engineering, London

  Extended abstract: Virtual Machines in Philosophy, Engineering & Biology
  `http://www.cs.bham.ac.uk/research/projects/cogaff/08.html#803`
  Slides here `http://www.cs.bham.ac.uk/research/projects/cogaff/talks#wpe08`
- 16 Oct 2008: School of Computer Science Seminar, Birmingham

  Why virtual machines really matter – for several disciplines
  `http://www.cs.bham.ac.uk/events/seminars/seminar_details.html?seminar_id=560`
- 21 Oct 2008: The Great Debate, Newcastle

  What can biologists, roboticists and philosophers learn from one another? (Unnoticed connections)
  `http://thegreatdebate.org.uk/UnnoticedConnections.html`
- 1-2 Nov 2008: Weekend course Mind as Machine, Oxford

  Why philosophers need to be robot designers
  `http://oxfordphilsoc.org/`

I have been writing about the importance of virtual machines intermittently since 1978 (*The Computer Revolution in Philosophy*)

Earlier talks used the slides in this (PDF) presentation
`http://www.cs.bham.ac.uk/research/projects/cogaff/talks/#virt`