triple**C** cognition
communication
co-operation

The Great Chains of Computing:
Informatics at Multiple Scales

Kevin Kirby[1], James Walden[1], Rudy Garns[2], Maureen Doyle[1]

[1]Department of Computer Science and [2]Department of Philosophy
Northern Kentucky University, Highland Heights, KY 41099 USA
Corresponding author email: kirby@nku.edu

Abstract: *The perspective from which information processing is pervasive in the universe has proven to be an increasingly productive one. Phenomena from the quantum level to social networks have commonalities that can be usefully explicated using principles of informatics. We argue that the notion of* scale *is particularly salient here. An appreciation of what is invariant and what is emergent across scales, and of the variety of different types of scales, establishes a useful foundation for the transdiscipline of informatics. We survey the notion of scale and use it to sort out characteristic features of information statics (data), kinematics (communication), and dynamics (processing). We then explore the analogy to the principles of plenitude and continuity that feature in Western thought, under the name of the "great chain of being", from Plato through Leibniz and beyond, and show that the pancomputational turn is a modern counterpart of this ruling idea. We conclude by arguing that this broader perspective can enhance informatics pedagogy.*

1. The Transdiscipline of Informatics

The perspective from which information is an organizing principle of the universe has become increasingly compelling. Science and technology have both led us to this same place, where the notion of pervasive information, and of pervasive information processing, seems useful.

It is typical to cite cellular biology and quantum physics as sciences yielding technologies that have come to depend on informational concepts, but even "information technology" itself has broken through its own envelope and led to emergent informational phenomena. This kind of emergence is not merely a function of new devices, nor of the synergy between new devices and users. Rather, it is due to the synergy between the new devices and *very large numbers* of users. This is one of several places in which the notion of scale must enter our vocabulary.

The setting of our exploration of scale is within the field of "general informatics" or "theoretical informatics", to use the terminology of Hofkirchner et al. (2007). For short, we will just use the abbreviation "informatics" here, with the caveat that we mean something more expansive than the common European sense of the term which is used to denote computer science and information technology. In some sense, it is the discipline generated by the intersection of all the disciplines of *x*-informatics, as *x* ranges over {*bio*, *social*, *geo*, ...}. It is pointedly *not* the union of these fields. This is consistent with a recently emerging use of the term "informatics" to denote academic units in the United States. (The authors of this paper are from a "College of Informatics".)

For our uses in this paper, we take informatics to be, at least, a *weak transdiscipline*, in that it engages with, transforms, and is transformed by the various disciplines named by *x* in the previous

paragraph. In this weak sense, it is akin to mathematics, which transforms and is transformed by other scientific fields. We reserve the term *strong transdiscipline* for the extension of informatics that has the additional property of having at its core a normative component, for example a social and democratic drive, following the Salzburg model (Hofkirchner, et al 2007). Our view is consistent with strong transdisciplinarity, but, except for some discussion in the last section of this paper, is largely based on weak transdisciplinarity.

Even in its weakly transdisciplinary form, however, informatics does generate tension and conflict as disciplines interact; see for example D'Inverno and Prophet (2004) for a computation / art / biology example. But just as there is trade in "dangerous ideas," there is also the exchange of "gloss", as, for example, computer science gains in its appeal (to students, to funding agencies, to researchers) through its links with the arts and humanities, and vice versa.

More philosophically, the point of view of this paper is one of "metaphorical pancomputationalism", in the terminology of Dodig-Crnkovic and Müller (2010). That is, we assume that all processes can be described as if they were computational processes, where computation is interpreted in the broad sense which includes natural computation (Kirby, 1998). This assumption is weaker than it may appear at first glance, as it does not rule out the ability to get "better" descriptions of processes by using non-computational language. Indeed, it can be very interesting to see exactly where and when this may happen.

This paper is an invitation to consider scale, in particular the notion of multiple scales, as fundamental in informatics, a notion that sits well alongside pancomputationalism. We want to understand informational phenomena that are *invariant* across scales, and to understand informational phenomena that *emerge* along scales. As such, it may be viewed as a generalization of the approach of Marijuan (1998) in which information processing is used to expose links between two widely separated points on a scale: cellular processes and economies. He called this an example of "vertical information science." Indeed our approach could be alternately phrased as a theory of "plural verticalities."

In the next section we examine the notion of scale in itself; it is necessary to clear up some ambiguities, as the metaphorical range of the term is vast. In section 3 we look on informatics as if it were a kind of mechanics, using the classification of statics, kinematics, and dynamics, as areas which exhibit scale phenomena. Section 4 explores some analogies to a pervasive idea in Western thought through the eighteenth century, the so-called "great chain of being", in which a scale-conscious form of pancomputationalism can be thought of as a "great chain of computing." We conclude with some implications for transdisciplinary pedagogy.

2. Scale and Scales

Our story of scale starts with a storybook. In 1957 the Dutch schoolteacher Hans Boeke wrote a picture book for children entitled *Cosmic View: The Universe in 40 Jumps*. It began with an aerial view of a child in a chair holding a cat (scale on the order of meters). Step by step it zoomed out by powers of 10 to encompass the breadth of the known universe, then zoomed in to the scale of an atomic nucleus. Video would capture this progression even more directly than a storybook, and popular videos were soon made. This is the universe at multiple scales. Across ranges of the scale we see a lot of homogeneity and invariance. At other points we see interesting phenomena emerge. Most dramatically, at the scale of micrometers upward things get quite interesting for a while, as we encounter life. And most peculiarly, at the so-called Planck scale (which the book did not depict but subsequent videos did), the whole notion of spacetime --and perhaps the notion of scale itself -- breaks down. It is this notion of scale that we wish to generalize, and apply to informatics.

Formal notions of scale require only graduated or progressive ordering; most familiar are the scales we associate with space (size) and time (duration, past-future). The powers of 10 idea draws attention to exponential changes in perspective that are

required to understand the world, taking us out of a comfort zone to perspectives that are extremely small or extremely large. We confront the vast range when we explore the post-Newtonian world of nanoscale and astrophysical phenomena. The concepts, theories and methodologies we employ at one level are often stretched beyond recognition for use at another and we search for something to unify our understanding at all levels of the scales.

Not only do we look *along* scales (the big and small of size and the short and long of time) but we need to consider the application of informational concepts to different scales; we need to look across scale. The approach is thus multi-dimensional. Space and time alone may prove relatively uninteresting next to scales of complexity and organization. It is not the measureable size or weight of the Internet that creates new and interesting phenomena and poses new and interesting problems, it's the different levels of complexity and organization. In other words, e need to contend with the emergence of new types of phenomena that arise with increased complexity and new forms of organization. Clearly pronounced in terms of social organization and networking, we see it also in scientific reduction, levels of explanation and disciplinary difference (the different ontologies and epistemologies / methodologies of different sciences).

The drivers are philosophical, scientific, social and economic. New interest in the importance of information and computation plays fundamental roles in a wide range of explanatory and problem-solving contexts. Because the range is wide, we need to think about information and computation from a variety of different disciplinary perspectives. Also because the range is wide, we need to view information and computation along and across different scales. Notice, too, that 'communication' gets used across a wide variety of contexts: neurons communicate, people communicate, networks communicate.

Transdisciplinarity and the scales perspective are interdependent ideas, Moving across different disciplines (from the natural sciences to the social sciences) naturally moves one along a scale of ontological and methodological commitment, from simple phenomena in physics to complex phenomena in social and behavioral science. Richard Dawkins (1986) nicely describes this in the opening chapter of his popular book *The Blind Watchmaker*. From causal or mechanical explanations in natural science to intentional and teleological explanations in social science, from quantum or deterministic causal explanations in physics to statistical and probabilistic explanations in social science, as information is used in different disciplines one must navigate the scales, moving from the concept of a bit of information transmitted through a channel through complicated networks set in complex social and economic frameworks.

The term "multiple scales" is ambiguous. On the one hand, when we say a phenomenon X *occurs at multiple scales* we might mean that it occurs, say, at the microscopic spatial scale and also at the macroscopic spatial scale. This is probably the most common usage. (Similarly, in the study of fractals one speaks of patterns recurring at "all scales.") However, in a different sense of the term, that usage refers to just a *single* scale: the spatial scale. Other scales are of course possible: temporal scales, numerical scales, scales of increasing abstraction, scales of increasing complexity, scales of increasing scope (e.g. across source code in a program or across a neighborhoods of interconnectivity in a graph).

The usage in the title of this paper includes both senses: we wish to speak of invariant and emergent phenomena at a variety of points along a *single* scale, and we wish to speak of a variety of *different types* of scales.

The notion of scale itself contains within it two dual notions: one of *extent* and one of *resolution*. As one moves up a given scale, there is the idea that one is in some sense becoming more encompassing, more comprehensive. This could mean literally spatial extent (e.g. from city to a region to a planet), or extent in some less literal sense (e.g. increasingly more general types in a type hierarchy). It could simply mean an increasing value of a natural number n, which, viewed as a cardinal number, does capture a kind of extent.

Dual to the notion of extent is the notion of *resolution*. As we move up in scale, distinctions that we made at lower levels may possibly be collapsed, or at least no finer distinctions will be made. One thinks of zooming out in a satellite map, or the transition from cellular biology to ecology (moving from cells to individuals to species). There is a loose closure to discourse at a given point on a scale. We rarely need to talk about DNA when we put forth sociological explanation (and when we do it is remarkable enough just because it is a transgression of normal scales).

We can pin down what parts of the initially vague scale metaphor we will employ by being a bit more formal. We capture the extents of different points on a scale by an ordered family of sets $\{V_i\}$ where the index i ranges over a linear order, perhaps but not necessarily a finite set of integers $i=1,2,...n$. These sets form a sequence of inclusions: $i < j$ implies $V_i \subseteq V_j$. We can capture the idea of resolution by introducing an equivalence relation E_i on each set V_i. This is a relation that is reflexive, symmetric, and transitive, and thus partitions each V_i into different sets of elements that are viewed as equivalent (indistinguishable) at that point in the scale. That is, if $x,y \in V_i$ then $(x,y) \in E_i$ means that x and y are equivalent. The quotient set V_i / E_i is the set of equivalence classes, and is the set that represents a point on a scale. For this to match the metaphor, as we go "higher" up the scale resolution can only get "lower". This appealing contravariance is captured by requiring that if i precedes j, then $(x,y) \in E_i$ must imply $(x,y) \in E_j$. In sum:

A *scale* is a sequence of set pairs $\{(V_i, , E_i) \mid i \in I$ with $E_i \in V_i \times V_i \}$ where (*a*) I is a linearly ordered index set; (*b*) each E_i is an equivalence relation; and (*c*) if $i < j$ then $V_i \subseteq V_j$ and $E_i \subseteq E_j$.

To be clear, this is a formalism, not a formal model. It serves merely to add a precise backdrop to our discussion of scale. It would exclude, for example, the notion of a musical scale. Perhaps its most straightforward instantiation occurs with spatial scale. One could take each V_i, as a region of space, where we zoom in and out by changing i. The elements of V_i / E_i are regions of space treated as one unit, and could be formed as a grid, i.e. as Cartesian products of real intervals. The formalism also captures extent in the scale of numerical size, when one might speak of increasing orders of magnitude: a network with tens of users is different from an internet with millions of users.

Our formalism is consistent with the formalism of Floridi (2008) for gradients of abstraction, in particular with his notion of a *nested gradient of abstraction*. Floridi identifies a level of abstraction with a set of typed observables, and defines behaviors at a level as predicates over those variables. He is interested in reduction along a gradient of abstraction, so he defines relations between observables in adjacent levels to capture this. He is interested in the truth of statements about behaviors posed at different levels of abstraction, and so his formalism is more specialized than ours. In particular, his discussion of emergence is more specialized, and is ontic where ours is epistemic. Our view of scale is relaxed enough to give meaning to the sense of epistemic "surprise" as in the discussion of emergence by d'Inverno and Prophet (2004), and accepts as informative scientific discourse (such as Marr 1982) in which one speaks of computational, algorithmic and implementational levels, even though this does not fit well with Floridi's formalism.

3. Information mechanics surveyed

To organize our review of scales in informatics, we break up informatics into three interrelated areas along the same lines as mechanics in physical science: statics, kinematics, dynamics. These deal respectively with information in storage, information in communication, and information in processing.

3.1. Statics

Information statics is the study of information in storage. At its center we find the most basic numerical scale: memory size.

This scale is vast, pragmatically logarithmic, and ranges from bits to yottabytes (2^{80} bytes). The area of statics itself organizes itself along the following conceptual scale:

- Information architecture
- Data structures
- Data encoding
- Physical encoding

The so-called memory hierarchy also plays a crucial role:

- Off-line storage (tape, optical)
- On-line storage (hard drives, flash drives, network area storage)
- Main memory (RAM)
- Cache (three levels are typical today)
- Directly accessible (processor registers)

Points on the memory hierarchy scale near the top support larger amounts of data but require longer access times. At the other end, where access has virtually no delay, there is room for only hundreds of bytes. While programs using small amounts of data can typically ignore the memory hierarchy, large scale systems like the Google search database that use petabytes of storage, require intense focus on memory hierarchy scales to return results quickly to users.

3.2. Kinematics

Information kinematics is the study of information in transit from one location to another. The Internet at all levels, from social networks to the transmission of electromagnetic waves through wires or the air, is the domain of kinematics. Information kinematics covers communication phenomena over a wide variety of conceptual scales, including:

- Social
- Content
- Application
- Presentation
- Session
- Transport
- Network
- Data Link
- Physical

These conceptual scales are often called layers in field of data communication. The third through ninth scales are the layers of the O.S.I. (Open System Interconnection) model, which is the most widely used conceptual model for discussing data communications. Each layer depends on the layer directly below it and supports the layer above it.

Content is transmitted through application layer protocols, such as Hypertext Transfer Protocol (HTTP), which is used by web browsers and servers to communicate, or Simple Mail Transfer Protocol (SMTP), which is used for e-mail. The presentation scale deals with the representation of content transmitted through the application scale, including issues such as byte order and character sets for different languages.

The session scale manages user sessions with applications, including keeping consistent user identity through connections going to a farm of servers and synchronizing audio and video streams used for network video transmission.

Transport scale protocols create long term dialogues between devices, breaking up application data into units of data called segment for transmission across the network. These protocols ensure that data segments are reliably transmitted across the network. Transmission Control Protocol (TCP) is the best known transport scale protocol and is one of two core protocols of the Internet.

Network scale protocols address end-to-end delivery of data packets, including routing through intermediate network nodes, while data link protocols address simple node to adjacent node communication. The Internet Protocol (IP) is the best known network scale protocol and the second of the two core protocols of the Internet. This protocol is also responsible for assigning network addresses (IP addresses) to each network node.

Data link protocols deal with the encoding of data packets into different electronic media protocols, including Ethernet and IEEE 802.11 wireless.

The physical scale addresses the encoding of bits on a wire or radio spectrum and signaling techniques. Individual Ethernet protocols such as 100BASE-TX or 1000BASE-T (gigabit) are physical scale protocols.

The layer model becomes even more profound when it is transgressed. In *tunneling*, data that is delivered can belong to a layer lower down (e.g. the data link layer) but is wrapped so that it is delivered as if it were at a higher level (e.g. the transport layer). It is through tunneling, for example, that firewalls that attempt to block certain services can be subverted. This freedom to "wrap" data to transgress the layers is the information kinematic analog to universality and software emulation in information dynamics, which we turn to next.

3.3. Dynamics

Information dynamics is the study of the transformation of information through the process of computation. Information processing happens along a very rich scale:

- Social
- Desktop
- Applications
- Frameworks
- Libraries
- System Calls
- Kernel
- Drivers
- Assembly
- Macro-ops
- Micro-ops
- Gates
- Elementary particles and fields

There are a dozen conceptual levels of abstraction between the user at the keyboard and the physical process of computation proceeding inside a modern computer. This set of scales has originated through a gradual process of invention from the earliest electronic computers which were programmed by manually manipulating switches and wires and used by a single person at a time to modern personal computers which do not require users to write programs and which interact with millions of people through a global network.

This expansion has been enabled by the exponential growth in computational capacity described by Moore's Law because the addition of each scale of abstraction has a cost in terms of performance. Our computations would proceed more rapidly if we represented them as an optimized set of micro-operations that the microprocessor could directly process without any intermediaries. However, it would take us a tremendous amount more time and effort to represent our computations in such a manner. Instead we perform our computations through applications with usable human interfaces that are built on lower scales of software that can be re-used for a wide variety of tasks.

The social scale is concerned with the design and specification of such user interfaces, the field of human computer interaction, while the desktop scale addresses the combination of applications, such as the window manager, file browser, and a variety of hardware management tools, that interact to form the user interface. Applications, such as web browsers, mail clients, and instant messagers, run within the desktop environment at the application scale, dependent on the frameworks below them.

The framework scale contains both desktop frameworks such as Gnome and web frameworks such as Ruby on Rails. Software frameworks promote a standard structure for applications and control the flow of control between parts of an application. Frameworks are often built on top of a wide variety of libraries at the library scales. Libraries are collections of subroutines that perform a common task, such as data compression or image manipulation, required by many applications.

Libraries request service from the operating system kernel through system calls. System calls provide the interface between the application and the operating system. They are needed for any task requiring access to the hardware or communication with other processes or machines. The kernel scale is

responsible for managing system resources, including the processor, memory, and input/output resources such as disk, graphics, and network, and providing a layer of abstraction between applications and the hardware that actually performs computations.

Kernels are written in assembly language or a combination of assembly language another low level programming language like C. Assembly languages provide a symbolic representation of the machine code that is used by a particular microprocessor.

Microprocessors are typically programmed in a machine code that is backwards compatible with many older generations of microprocessors. To enable speed improvements while retaining backwards compatibility, the microprocessor performs arithmetic and logical operations on micro-ops, a lower level machine code that is generated by translation hardware on the microprocessor from macro-ops, the machine code that assembly language is translated into. This translation uses programmable memory on the processor, allowing the processor's machine language to be modified by the kernel to fix bugs or even add features.

The computation described by a micro-op, such as addition of two numbers, is performed by a set of logic gates, combining sets of binary numbers and producing a binary number as output. At the bottom scale of electrons, ones are represented by a high voltage level and zeros are represented by a low voltage level. At this level, computers are analog devices, having to deal with noise and timing issues in determining whether a particular voltage is a low or high level representing a one or zero respectively.

The survey of information dynamics above has led us through the zone of discrete artificial computation. As such, its foundation rests on the notion of a universal computer, arguably the simplest model of which is the Turing machine. Universality appears at many scales in information dynamics. The language of the microprocessor is every bit as Turing universal as, many layers up, a scripting language for a 3D animation program would be. Indeed both have access to similar scale-invariant features that have evolved to suit the needs of humans: decision statements, loop statements, behavior encapsulation in terms of procedures, and so on, precisely the features that make it universal.

Still, as we go up the scale we make use of more powerful abstractions. Abstractions in computer science are more dynamic and *ad hoc* in computer science than in mathematics (Colburn and Shute 2007), and are more subtle than the simple collapsing of distinctions into broader equivalence classes. Indeed each lower layer presents a public picture of itself to the higher layers, often through an Application Program Interface (API). These can be very elegant object-oriented interfaces, or very "leaky", allowing layer-violating access to lower levels.

There is heterogeneity of design here, and human programmer communities struggle to build complex systems and come up with a variety of solution schemes. What software engineers are trying to manage here is a scale of complexity, gauged for example by the integral numerical scale of *lines of code* (LOC). From scripts a few tens of lines long, to operating systems a few tens of millions of lines long, this is a great height one must scale. (It is interesting that English can use the verb *to scale* to denote the act of *traversing* a scale, as well as to establish a scale.) The deep idea here is that to handle the vast scale of LOC, software engineers have turned to another type of scale, a scale of increasing abstraction. This subverts the LOC gauge by rescaling it: a ten-line script in Python for a game encapsulates hundreds of thousands of lines of code that execute when it runs. Programmers have often seen the humor in this (Munroe 2009).

The field of artificial computation is of course pluralistic. Beyond the traditions sketched above, there is also the tradition coming out of the lambda calculus, and the tradition coming out of nondeterministic and stochastic programming. The Church programming language (Goodman et al. 2008) is a new example of work that combines these traditions.

Underlying artificial computation is the idea of a text that is both dead (data) and alive (executable). The power and limits of Turing computation arise from the consequences of treating code as text and conversely. The

performative nature of code (Austin 1975) gives it its power as it acts on the world. (The simple proof of the unsolvability of the halting problem is only a few lines long, a meditation on the notion of executable text.)

It is the notion of *text* that gives us the LOC scale; it is the notion of *execution* that gives us the time scales used in asymptotic complexity analysis. These are complementary. Across program text we have the notion of *scope*, as in the range of text over which a variable has a given meaning. This is a special case of locality of reference (Denning 2005) along a scale. There are many scales of abstract time complexity, the coarsest of which is (P, NP, non-NP).

Natural computation also has its own set of scales, the most obvious being the spatiotemporal scale. Quantum computing and biological computing are two areas of interest here, and insights there have the promise of shaking up both computer science and information theory (Kirby 1998, 2002). For example, the so-called quantum-to-classical transition, and the very qbit/bit distinction, is a scale issue. The emergence of self-replicating macromolecules, and then the emergence of information processing out of uninterpreted dynamics occurs across a complexity scale.

The pervasiveness of scale in computing, taken together with the pervasiveness of computing itself, actually pulls us into a world picture that is quite different from the mechanical world view (Dodig-Crnkovic and Müller 2010) It is this we will explore in the next section.

4. The Great Chains of Computing

In his William James Lectures delivered at Harvard University in 1933, Arthur Lovejoy called his listeners' attention to an idea that had persisted in Western thought from the time of Plato through the eighteenth century. This idea, called the "great chain of being", was not some tacit idea or *episteme* that covertly shaped European thinking. Rather it was an explicit, named, and fully articulated idea that influenced not only philosophy and theology, but also the nascent fields of astronomy and biology. The idea was that modes of being in the universe are arranged on a scale. These modes of being ranged from inanimate objects like rocks, through plants, then to lower and higher animals, then humans, and then (in its common theological form) though various celestial beings, including angels, terminating at God. The "models" of God used were variously theist and deist, and in the seventeenth century there was the introduction of supposed life on other planets, and in the eighteenth the introduction of microorganisms.

This scale was characterized by gradation, plenitude, and continuity. Gradation gave it a linear scale. Plenitude was the idea that every possible mode of being necessarily had to exist. Continuity was the idea that between any two points on the scale there was another point. The continuity idea was introduced by Aristotle, though it ran counter to the main Aristotelian emphasis on discrete classification. Indeed, the thesis of Lovejoy's lectures and subsequent book (Lovejoy 1936) was that the ideas were unstable and (to employ a more recent term) only paraconsistent. They nevertheless served as a generator of ideas for philosophers, scientists, and even the Romantic poets.

In the thought of Leibniz, whom we bring in here as arguably the first computer scientist for his work on the universal characteristic, the idea of the great chain of being reaches a very developed form. For him, the modes of being are not life forms per se, but the mind-like entities he called monads. His "panpsychism" located these monads along a graded, full and continuous scale. Yet despite this panpsychism he was not an idealist in the purest sense: physics problems were certainly genuine to him.

Lovejoy claimed that the persistence of this not-quite-consistent, not-quite-supported idea of a great chain of being was in large part due to its "metaphysical pathos", its appeal to both casual and deep thinkers, as describing a universe that they felt at home in, one that resonated with their everyday life world.

In the everyday life world of those of us who live in societies dominated by information technologies, who have universal computers on our desks, in our pockets and in our cars, who manage our human relationships online,

what idea could have more metaphysical pathos than pancomputationalism? We move toward a world which consists not of billions of monads but billions of computers. We scale these great chains of computing (note the plural) as we move from entangled electrons, to logic gates, to microprocessors, to operating systems, to APIs, and on up to planet-scale social networks. There are indeed multiple scales here: we can also move from particles to macromolecules to cells to cell networks to organisms and to ecosystems, which we have nowadays come to view computationally. Whether quantum effects percolate up the scale to affect life is a deep scale-based question. Indeed Michael Conrad (1994) called his non-standard model of quantum gravity, in which there were such percolating influences, "the great chain of being model."

Just as Leibniz used his principle of sufficient reason to get causality and teleology into the great chain of being, through programming at all levels we have the introduction of artificial causality (flow of control) and teleology (through users) in our chains of computing. And just as Leibniz's panpsychism did not lead him to deny the importance of physics, our pan-computationalism (still of the metaphorical variety) does not mean there is no role for matter in computation. Indeed the physical substrate in natural computing is crucial (Kirby and Conrad, 1986), and there is a complementarity between Turing's tape-based state machine and his morphogenesis model. Exploration of the latter continues (see e.g. Furusawa and Kaneko, 1998).

Still, at the root of many of the great chains of computing we can find fundamental principles of *physical* theory that are informational in nature. One example of this is the information causality principle (Pawlowski et al 2009), an information-based constraint that may provide an informational characterization that singles out "real" quantum mechanics from other possible quantum-like theories.

Like the great chain of being, the great chains of computing idea serves as a generator of ideas and inspiration, a tentative and paraconsistent world view. The world may be seen not merely as a vast set of natural and artificial computers; it is a vast set of computational scales.

5. Implications for Pedagogy

Computing is not just for computer scientists. As educators as well as researchers, we believe strongly that the (weak) transdisciplinarity of informatics should allow it to engage with people of all different interests and skills, and that the (strong) transdisciplinarity of informatics should motivate them to use this as a tool for shaping a sustainable world. After all, the metaphysical pathos of the great chains idea is at bottom all about *appeal*, and this is what we would like to leverage.

The authors of this paper are currently working on a project funded by the U.S. National Science Foundation as part of a program to "revitalize" education in computing. One component of this project is to build transdisciplinary teams of faculty to develop a course on the principles of informatics (cf. Wing 2008) that can not only serve beginning students in computer science, but also students in fields ranging from information systems to journalism to electronic media and broadcasting. (These programs are brought together in a College of Informatics.)

Although the first class will not be offered until Fall 2010, the transdisciplinary team, involving to date computer scientists, a philosopher, a musician and composer, and a social communications theorist, has already experienced in their planning sessions some of the creative conflict that will no doubt be replicated in the classroom. Even with a team motivated from the start to engage in transdisciplinary interactions, there are clearly misunderstanding that we have found need to be sorted out. Some are intellectual: for example, what counts as a theory? Some are cultural: must one be a "geek" to love to write code? We also wish to have our students sort through this. Most of these students will be in the age range 18-22 and thus are likely to have information-based applications play an even more pervasive role in their life compared to the lives of their teachers.

Accordingly, we expect this generation of students to be more at home within these great chains of computing. As teachers, we hypothesize that an approach based on mutual respect across disciplines and on principles, especially the principle of scale, will inspire them to think and work deeply. And perhaps, punning on the other sense of the word "chains", they will find a way to cut them and be liberated.

References

Austin, J.L. (1975). *How To Do Things With Words,* Second Ed. Cambridge, Massachusetts: Harvard University Press.

Boeke, K. (1957). *Cosmic View: The Universe in 40 Jumps.* New York: John Day.

Colburn, T. and Shute, G. (2007). Abstraction in Computer Science. *Minds and Machines 17,* 169-184.

Conrad, M. (1994). From brain structure to vacuum and back again: the great chain of being model. *Nanobiology, 3,* 99-121.

Dawkins, R. (1986). *The Blind Watchmaker.* New York: Norton.

Denning, P.J. (2005). The locality principle. *Communications of the ACM, 48(7),* 19-24.

D'Inverno, M. and Prophet, J. (2004). Creative conflict in Interdisciplinary Collaboration: interpretation, scale, emergence. In E. Edmonds and R. Gibson (Eds.), *Interaction: Systems, Theory and Practice* (pp. 251–270). ACM Press.

Dodig-Crnkovic, G. and Müller, V. (2010) A dialogue concerning two world systems: Info-computational vs. mechanistic. In Dodig-Crnkovic, G. and Burgin, M. (Eds.), *Information and Computation.* Singapore: World Scientific.

Floridi, L. (2008). The method of levels of abstraction. *Minds and Machines*, *18 (3)*, 303-329.

Furusawa, C. and Kaneko, K. (1998). Emergence of rules in cell society: differentiation, hierarchy and stability. *Bulletin of Mathematical Biology 60*, 659-687.

Goodman, N., Mansingkha, V., Roy, D., Bonawitz, K. and Tenenbaum, J. (2009). Church: a language for generative models. Proc. 24th Conference on Uncertainty in Artificial Intelligence, 220-229.

Hofkirchner, W., Fuchs, C., Raffl, C., Schafranek, M., Sandoval, M., and Bicher, R. (2007) ICTs and Society: The Salzburg Approach. *ICT&S Research Paper, 3,* Center for ICTs and Society, University of Salzburg.

Kirby, K.G. (1996). Exaptation and torsion: Toward a theory of natural information processing. *BioSystems 46,* 81-88.

___ (2002). Biological adaptabilities and quantum entropies. *BioSystems 64*, 33-41.

Kirby, K.G. and Conrad, M. (1986). Intraneuronal dynamics as a substrate for evolutionary learning. *Physica D 22*, 150-175.

Lovejoy, A. (1936). *The Great Chain of Being: A Study of the History of an Idea.* New York: Harper.

Marijuan, P. (1998). Information and the unfolding of social life: molecular-biological resonances reaching up to the economy. *BioSystems 46,* 141-151.

Marr, D. (1982) *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* San Francisco: W.H. Freeman.

Munroe, R. (2009). Abstraction. *Xkcd.* http://xkcd.com/676/.

Pawlowksi, M., Paterek, T.,Kaszlikowski, D., Scarani, V., Winter, A. and Zukowski, M. 2009. Information causality as a physical principle. *Nature 461,* 1101-1104.

Wing, J. (2008). Five deep questions in computing. *Communications of the ACM 51(1),* 58-60.

About the Authors

Kevin Kirby
Evan and Lindsay Stein Professor of Biocomputing and Chair of the Department of Computer Science at Northern Kentucky University, Kirby earned his PhD in Computer Science under Michael Conrad at Wayne State University in 1988. He co-edited the proceedings of the FIS 1996 conference and presented at the FIS 2005 conference in Paris.

James Walden
Assistant Professor of Computer Science at Northern Kentucky University, Walden earned his PhD in particle physics at Carnegie Mellon University, and worked for Intel Corporation. His current research centers on secure software engineering.

Rudy Garns
Associate Professor of Philosophy at Northern Kentucky University, Garns earned his PhD in philosophy under William Alston at Syracuse University in 1989. His current research interests include the philosophy of mind and information

Maureen Doyle
Assistant Professor of Computer Science at Northern Kentucky University, Doyle earned her PhD in computational mathematics at Stanford University. Her current research centers on software engineering and pedagogy.