What Deans of Informatics Should Tell Their University Presidents[1]

Robert L. Constable
Dean of the Faculty of Computing and Information Science
Cornell University

1. Introduction

In North America, a group of department chairs and deans would immediately infer from the title of this article that it is about promoting informatics. Indeed, this article is about how my senior colleagues and I create support for computing and information science (informatics) among university presidents, provosts (rectors), industry leaders, funding agency heads, legislators, potential donors, and other influential people who will listen. It is about the importance of computing and information science to universities and to society as a whole.

Appeals for support can be based on many approaches: the excitement of opportunities, fear of failure, envy of other institutions, responsibility for mission, duty to make the world a better place, pride of accomplishment, legacy, or on economic grounds. No matter what the approach, there are three basic facts that sustain the case for informatics. Here they are briefly summarized:

> **(1)** The ideas, methods, discoveries, and technologies of computing and information science continue to change the way we work, learn, discover, communicate, heal, express ourselves, manage the planet's resources, and play. *Computers and digital information change whatever they touch and have created a new knowledge paradigm.*
>
> **(2)** The impact of fact (1) on the academy has been large and increasing because computing and information science is transforming how we *create*, *preserve*, and *disseminate knowledge* – these are the core functions of universities. The transformation affects all disciplines.
>
> **(3)** Advanced economies are *knowledge-based*; they depend on information, computing and communication technologies and the people who create and deploy them – the ICT sector. Universities must offer a foundational education for the knowledge workers, researchers, and teachers. That education must provide for life-long learning in a fast moving sector of the economy.

These are powerful facts, inescapable facts. They have created new industries and vast wealth. Their impact on the core business of the academy attracts fewer headlines, but it is worthy of careful examination. New academic structures are being created (departments and colleges). Computer science education is important to more disciplines. New multidisciplinary and interdisciplinary programs connect computing and information science to other disciplines. Professional societies such as the Computing Research Association (CRA) in North America are expanding.

In due course, these factors will be examined by scholars, but our job now is to create opportunities for these facts to play out in full, to lead this transformation of the academy. Using

[1] This article is based on an invited lecture to the Eurotics meeting of CS department chairs and deans, held in Zurich on October 16-17, 2006.

these facts to inform policy and create opportunities requires that we understand them well and present them clearly. To this end, I will illustrate them from three areas: *computational biology*, *astronomy*, and *mathematics*.

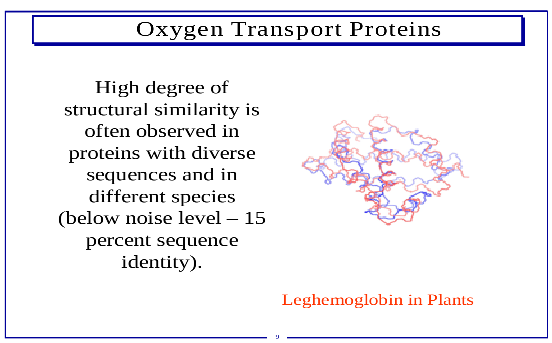
It is possible to find compelling examples in almost all areas – from the physical sciences, life sciences, medicine, law, social sciences, arts, and humanities. The ones I have selected are accessible because the press has educated people about them.

After these examples, I compare new college-level structures in the US that started with computer science at their core, including Cornell's Faculty of Computing and Information Science (CIS) of which I am dean. This comparison serves readers who want to see natural paths by which computer science has expanded and what its natural shape might become. This is an explicit topic discussed by CRA and Eurotics that I was asked to address. I conclude with an historical perspective and an observation about forces that will influence the course of events over the next decade.

2. Computational Biology

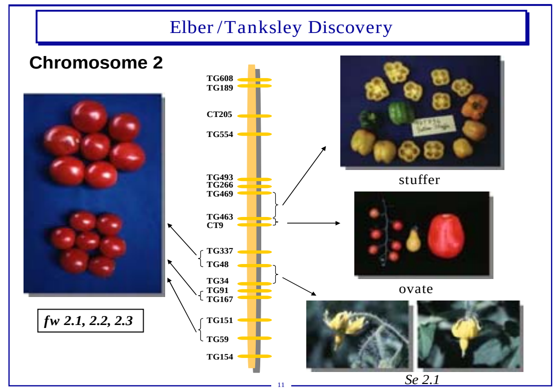
Biology is computational on every scale, from proteins to brains. Many scientists believe that *informatics will be to biology what mathematics is to physics*. I will look at the level of protein function. The results come from the plant biology lab of Steve Tanksley and the bioinformatics lab of Ron Elber, and were reported in *Science, 2000*.

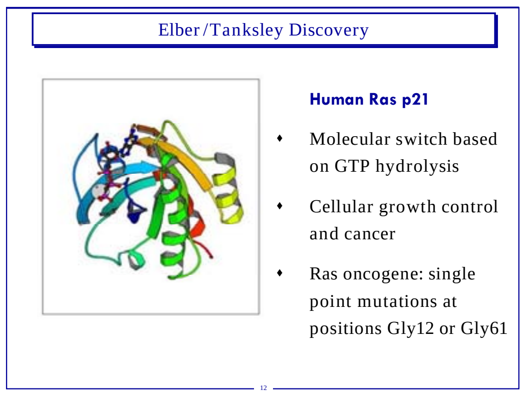
One way to discover the function of a gene is to determine the *sequence* of amino acids that chain together to make it, and then locate in the Protein Data Base (PDB) similar sequences whose function is known. This does not always work. For one thing, proteins that are not alike at the sequence level might nevertheless have similar function, which is determined by the *geometry of the 3d shape*. So it is important to recognize similar shapes. We can compare Hemoglobin and Leghemoglobin in this diagram "by eye."



We see that they are similar, but their sequences have less than 15% in common.

Steven Tankseley, the 2004 Wolf Prize Winner (along with Yuan Longping), has been investigating tomatoes to find the genes responsible for the size and shape of the tomato fruit and account for the transition from small, round wild tomatoes to the larger variably shaped ones of modern agriculture, see the diagram. His laboratory was unable to relate any of the genes they suspected to be growth genes to those in the PDB's 35 K entries using sequence comparisons. So he turned to computer scientist Ron Elber and his laboratory who were able to determine the shape of the proteins sufficiently well using their Loopp Software to predict the shape and then employ shape comparison software to find a match to the human Ras p21 gene based on geometric similarity. Ras p21 is an *oncogene* regulating cell growth and linked to human cancer. These results appeared in *Science in 2000,* 289, 85-88.



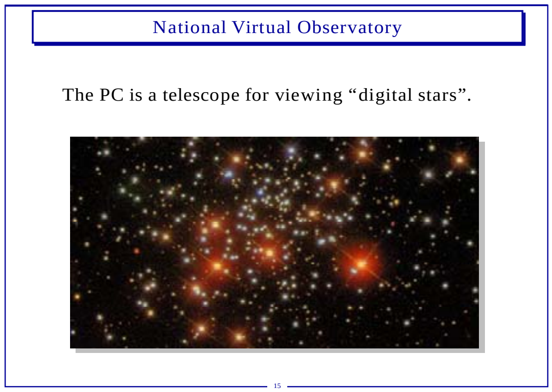


Answering questions of gene function depends on the deep understanding of genome assembly which was a 50/50 cooperation between biologists and computer scientists. Discoveries in string matching algorithms led to the assembly methods and to software tools such as Blast, CE, Dali, Loopp, and many others.   Structure comparison relies on results from *computational geometry*, one of the sub-areas of computing theory that has led to advances in computer vision, pattern recognition, and an understanding of the visual systems in animals. *The geometry of life is computational.*

Our ability to understand gene function is important to drug design and genetic medicine. It will help in our efforts to control bacterial infections and viral disease. Moreover, understanding protein to protein interactions is the basis for understanding cellular processes [5].

3. Astronomy

Astronomers in the United States are very organized about their research agenda, and they decide on priorities for five and ten year periods.   At the top of their current agenda is development of the National Virtual Observatory (NVO). This is a massive database project that is an excellent example of emerging *data-centric science*. Observations from the major telescopes are being digitized, stored, and organized so that anyone with a PC can examine the latest data, the *digital stars*. *Thus the PC is becoming a telescope for viewing the digital sky*. The digital stars are more than images of visible light, they include "observations" in the entire electromagnetic spectrum, from radio waves to x-rays. They include also the spectral analysis, and could index into everything known about a source.



The Cornell Arecibo Observatory, home of one of the world's largest radio telescopes, is involved in the effort to assemble a complete survey of *pulsars* – rotating neutron stars. The search methodology is no longer to assign the task to graduate students able to read the data; instead *data mining algorithms* are programmed to recognize pulsar candidates among the bursts of terabytes of data that arrive at the Cornell Theory Center. As a footnote on the process, the highest data rate is to ship USB-2 disks by air from the observatory to the university – that's 14 Terabytes every two weeks!

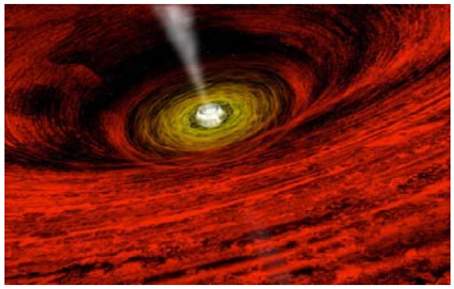
Data mining programs will search NVO data for more than pulsars. They will seek new exotic sources, and try to identify rare objects such as brown dwarfs, high-z quasars, ultra-luminous IR galaxies, and so forth. In the first few months of exploration, 21 new pulsars were found.

The astronomer Alexander Szalay and his team collaborate with computer scientists like Turing Award Winner Jim Gray at Microsoft Research (MSR). Szalay has said that the work of Gray has "changed astronomy as we know it" because he has shown how to use robust commercial products such as the SQL database system to integrate exceptionally rich data services and computational services for astronomy.

When Szalay was asked by fellow astronomers whether they will need to become computer scientists as well, he points out that they use sophisticated mathematics without being mathematicians. They will need to know basic principles of algorithmics and data structures. He notes that unless scientists are aware of the properties of algorithms and data structures, they can make poor choices and be unable to solve problems that are tractable for those educated in basic computer science.

The potential of the NVO and the related Sloan Digital Sky Survey (SDSS) are enormous for both discovery and education. People are universally fascinated by the heavens. Long before the Big Bang became the accepted cosmology, people suspected that the heavens held the clues to our origins. Contributing to the pure beauty of this field, no industry has a claim on the data; it is a free public resource, and the NVO is opening it to the world. This openness illustrates the democratizing and "leveling" character of the Information Revolution.

And we can even extrapolate to more complex exotic systems



17

4. Digital Age Mathematics

In the last thirty years, four long standing, high profile, open mathematical problems have been solved: the *Four Color Problem* in 1976 by Appel and Haken, *Fermat's Theorem* by Wiles in 1995, the *Poincaré Conjecture* by Perelman in 2002, and the *Kepler Conjecture* by Hales in 2005. Three of these involved computing and information science in essential ways – namely:

Consider these famous problems…

- The Poincare Conjecture
- The Four Color Theorem
- The Kepler Conjecture

Computers and the Web have fundamentally changed how they were solved.

18

Perelman's proof of the Poincaré conjecture was not published in a conventional journal, and it was not refereed in the normal way. He published in the on-line "journal" called the arXiv. It was essentially reviewed in that form and somewhat interactively in that Perelman published his result in three "installments," the later ones responding to community feedback on those earlier.

Digital Age Mathematics – The Poincaré Conjecture

On November 11, 2002 Perelman posted a proof of the Poincaré Conjecture on the Cornell arXiv, Paul Ginsparg's digital library of "e-prints." This posting stimulated the math community to "fill in the details."

(Paul Ginsparg is an Information Science professor in CIS, and Perelman's proof builds on the work of William Thurston, a Field's Medalist who has a joint CIS appointment with Math.)

20

The Four Color Theorem and the Kepler Conjecture used computers in an essential way, and raised important questions about the central concept of mathematics, proof; the very idea of a *proof* has changed. We will look closely at the role of computers in proof because *they have fundamentally changed a two thousand year old tradition.*

4.1 The Poincaré Conjecture

In 1904 the illustrious French Mathematician Henri Poincaré conjectured that "all closed simply connected 3d manifolds are spheres (have finite extent)." This is a theorem about the structure of the space we inhabit. A 3d manifold is simply connected if any loop (string) enclosing a region can be contracted to a point – by "pulling the string tight". We intuitively imagine our 3d space to be like that.

As one of the most famous open problems in mathematics, the Poincaré Conjecture is one of the seven Millennium problems for which the Clay Institute has offered a one million dollar prize. (The P=NP problem from computer science is also on this list along with the Riemann hypothesis.)

Grigory Perelman lives in Russia, and until recently, worked at the Steklov Institute in St. Petersburg. His proof was presented in the Russian style which leaves out many details. This combined with the unorthodox publication left room for others to take some credit for this extraordinary result [4].

With most proofs, eventually more detail is provided as they are presented to mathematicians who are not experts in the specific topic of the theorem, and in due course these proofs are taught to graduate students and perhaps eventually to undergraduates. The process of writing them over and over often leads to simplification as well as more detail.

The question of how much detail to provide depends on the audience and the purpose of the proof. Logicians have found that there is a limit to how much detail is needed, only enough to create a completely formal proof in a logical system. Computer scientists have given another characterization, which is mathematically equivalent. Namely, a completely formal proof is one that can be checked in every detail by a computer program. *This is the highest standard of correctness known in all of science*. Computer scientists have built several programs, called *provers*, that do this work. In the area of mathematics, three have been widely used, Coq, HOL, and Nuprl. (These systems are the products of sustained collaboration between Europe and North America over the past 30 years.) We will see that provers play a key role in the next two problems.

4.2 The Four Color Theorem

In 1976 Kenneth Appel and Wolfgang Haken announced a proof of *The Four Color Theorem*. Mathematicians had been seeking such a proof since 1852 when the problem was posed by F. Guthrie, a student of Augustus De Morgan.

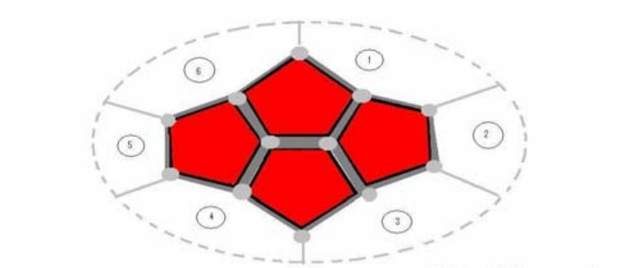
The theorem states that any planar graph can be colored with at most four colors such that any two adjacent regions have different colors. Regions are considered adjacent when they have a border that is more than a single point.

It is clear from maps of Europe or the United States that four colors are needed, consider Belgium, France, Germany and Luxembourg. Four colors will be needed even when the map does not contain four mutually neighboring states, as with these: Nevada, Oregon, Idaho, Utah, Arizona, and California.

Intuitively it seems that as maps become increasingly complex, they will need more colors. However, it was not long before it was proved by Percy Heawood in 1889 *that five colors suffice*. He built on a false proof of the four color result by A. Kempe in 1879. That false proof provided many of the ideas eventually used to solve the problem.

The Appel/Haken strategy for proving the Four Color Theorem was based on mainstream ideas studied since Kempe, and brought into modern form by Heinrich Heesch in 1935 [1]. This approach laid the basis for an algorithmic method that we discuss later. The idea is based on the concept of a *reducible configuration (graph)*. If a map has a reducible configuration *R*, then if we can color the graph with *R* removed, the coloring can be extended to the entire graph with possible recoloring. This suggests an approach to coloring *G*: look for a reducible configuration R_1, remove it, and try to color the remaining graph, $G - R_1$. To do this, look for another reducible graph, R_2, and try to color $(G - R_1) - R_2$. Eventually we are left with a small graph, and we know by inspection that they can all be colored. Indeed in 1967 Oystein Ore proved that all maps with up to forty countries could be colored. So, the question is whether we can always find a reducible configuration. An *unavoidable set of reducible configurations* is one such that any graph *G* must have at least one reducible configuration from this set.

Appel and Haken were looking for an unavoidable set, and a method of recoloring for each R in the set. Heesch believed there were many such sets, and his intuitions were rightly respected. He developed an uncanny knack for recognizing reducible configurations intuitively with 80% accuracy. In 1913, George Birkhoff showed that various graphs in the shape of rings surrounding a core were reducible, see the Birkhoff diamond as an example in the diagram.



Birkoff Diamond

In 1974 Appel and Haken proved that an unavoidable set with certain properties, called *geographically good* existed, but they did not know that every geographically good configuration was reducible, and they were not sure how large the set was – perhaps as large as 8,900 configurations that Heesch was contemplating. Finally, they found an unavoidable set with 1,936 configurations to check which they reduced to 1,482 and eventually to 1,405. These had to be checked for reducibility by a computer – an IBM 370-168 mainframe, running for 1,200 hours. In the end, they knew how to produce hundreds of unavoidable sets of reducible configurations. Finding one was enough to solve this open problem.

The Appel/Haken proof method provided an effective method of actually coloring a graph. At this point another major computational idea entered the picture. How efficient was the graph coloring algorithm implicit in their work? Was it feasible to run their method as a graph coloring program? Was there a graph coloring program from the Five Color Theorem, and would it run faster?

4.3 The Four Color Theorem – A Formal Proof

The Appel/Haken proof was published in the December 1977 issue of the *Illinois Journal of Mathematics*. There was criticism and a lukewarm reaction from many mathematicians who were uneasy about a proof that depended on programs and hundreds of hours of digital computation. The well-known expositor and mathematician Ian Stewart complained that the proof was unsatisfactory because no human could grasp all the details. Some logically minded mathematicians such as George Spencer-Brown claimed there is “no proof to be found in what they published.”

Various efforts were made to address the criticisms. The programs were rewritten and run on other computers. Computer scientists who studied program correctness, like David Gries, rewrote them in a higher level programming language and gave an informal correctness proof.

Interestingly, there is a realization that younger mathematics students who have grown up with computers feel comfortable with the proof and argue that they trust a machine to get 1,405 cases right, more than they would trust a person.

In 1994 Neil Robertson, Daniel Sanders, Paul Seymour, and Robin Thomas gave a new proof that required only 633 cases of reducibility to be checked and used only 32 so-called “discharging rules,” compared to 487 in Appel/Haken. Their work produced a coloring algorithm that runs in n^2 time in the number of nodes.

Another impressive reply to the criticism came from the computer scientists who build and use systems called *interactive theorem provers*. These systems help build completely formal proofs in which every single step of reasoning is recorded and checked. Moreover, some of these systems satisfy the *de Bruijn criterion* which requires that there is a very simple algorithm for checking the formal proof independently of the prover that created it.

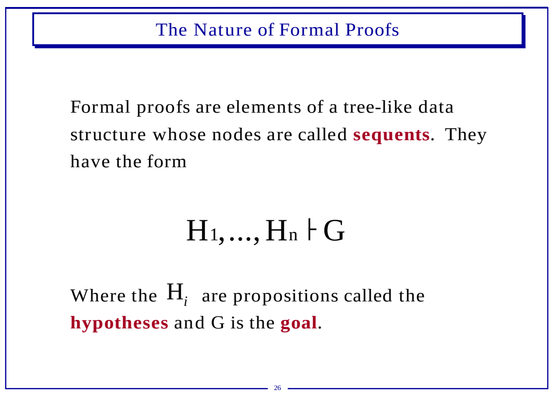
For several years George Gonthier of Microsoft Research (MSR) and Benjamin Werner of INRIA used INRIA’s Coq prover to construct a completely formal proof of The Four Color Theorem. Gonthier wrote an article in Computer-checked Proof of The Four Color Theorem and released the full Coq proof [7].

In this proof, the programs that check for reducibility are proved to be correct, and the computational results are integrated into the overall argument. This is a *formal tour de force* and is a more definitive proof than the original. In addition, the proof produces and algorithm for coloring planar graphs that runs in time n^2 in the number of nodes.

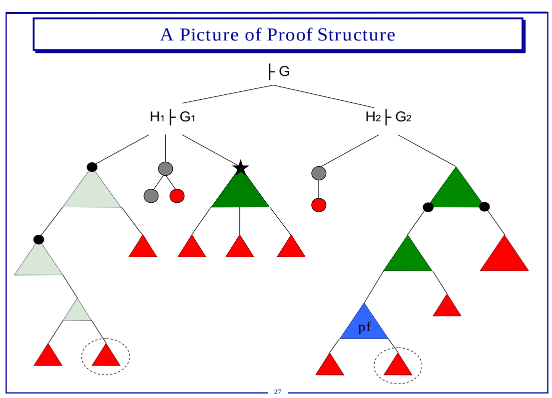
4.4 A Major Discovery about Formal Proofs

What is a *formal proof* ? It is a logical argument that can be checked by a computer. The process of going from this idea to a computer system that does the checking was arduous. First, the idea of an *assertion* had to be made precise, and this was part of the business of implementing programming languages. Then the notion of a step in an argument was made precise.

The proof step in provers like Coq, HOL and Nuprl is called an *inference* and it is represented by a list of assertions (logical formulas), say $H_1, \ldots, H_n$ and a conclusion assertion, say *G*. These are combined as a pair separated by the *inference sign* from logic, $\vdash$, called a turnstyle.

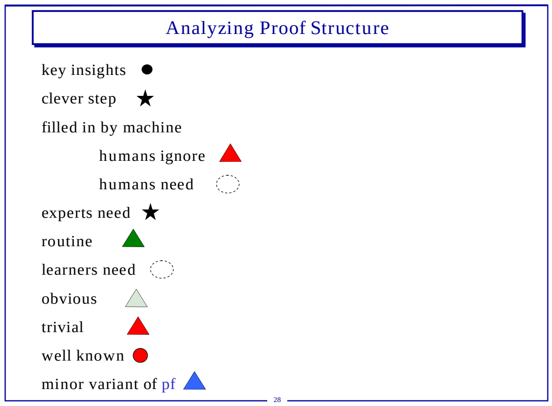


A proof is a *tree* whose nodes are inference steps.



An interactive prover works with a person to produce a proof tree. A prover could be "lazy" and require that the human propose all the inference steps which it simply checks for correctness. These are called *checkers*. However, the great value of provers is that they build large parts of the proof tree on their own. Some proofs are simple enough that the prover can build the whole thing. What is unexpected is how helpful provers are.

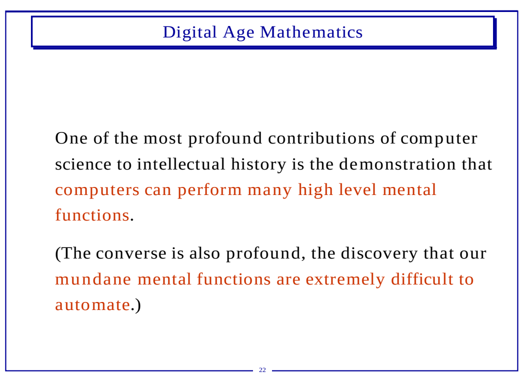
The next diagram illustrates how a proof tree can be analyzed in terms of the parts that computers can build and the parts that humans contribute or need the computer to make explicit to aid human understanding.



From the computer's viewpoint, it is a miracle that humans can figure out what is true without writing out a full argument…of course humans also seem to make horribly obvious mistakes from the computer's perspective, mistakes that take years to discover.

What is astonishing from a human's viewpoint is that the computer can do mathematics at all!

One of the major discoveries of computer science is that computers can automate many high level intellectual processes, such as playing chess, solving algebraic equations, discovering proofs, and helping create them.

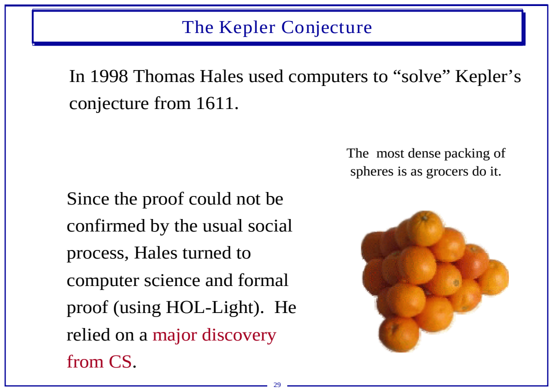


The converse discovery, that implementing the mundane mental functions is very hard, is well illustrated by the protein comparison shown earlier – the human eye sees the similarity immediately and computers can't…yet.

These discoveries of computer science help us understand the mysteries of the brain. The fact that Heesch could intuitively recognize reducible graphs after years of exploring them is a mystery that challenges computer science; one response is the exciting the field of machine learning.

4.5 The Kepler Conjecture

The final example from mathematics is part of an on-going saga [2]. In 1998 Thomas Hales used computers to solve Kepler's conjecture about packing balls as densely as possible in 3d space. Essentially in 1611 Johannes Kepler conjectured that the most dense packing is the way grocers stack fruit. Hilbert included this as problem 18 among 23 that he posed in 1900.



Hales' proof uses computers to check cases, just as Appel and Haken. But there are many more cases. As Ian Stewart remarked, "While Wile's proof of Fermat's theorem resembles *War and Peace*, Tom's proof of Kepler's conjecture resembles a telephone directory." The size of this proof and the role of machines made it difficult for Hales to get his proof refereed in his journal of choice, the *Annals of Mathematics*. Only in 2005 did a 20 page summary appear in that journal.

Hales frustration over the refereeing process led him to propose in 2003 a project (called FlySpeck) to create a completely computer checked proof (using HOL-Light). I see this as a remarkable step, signaling publicly in the mathematics community something that computer scientists have known since the late 80's, namely a new subject is being born – *Formal Mathematics*.

5. Computational Social Science and on to the Arts and Humanities.

5.1 Networks

It is already clear that computing and information science and social science are influencing each other. First there is a change of scale in measurement. Using the Web, social scientists have access to vast amounts of raw data and to computational methods to evaluate it.

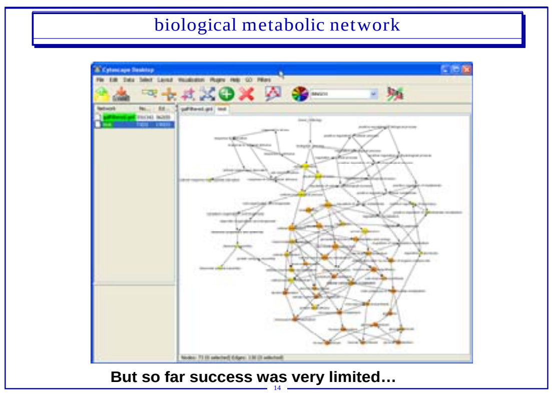
The statistical methods used to understand data sets at this new scale are computational, and conversely, computer scientists using large databases have found statistical methods to be indispensable. The field of *machine learning* has emerged as the systematic study of algorithms to extract information from large data sets. Some machine learning ideas have counterparts in statistics; they are examples of how computer science and statistics have moved closer as disciplines fundamentally concerned with extracting information from data.

Social scientists have long been interested in social networks. In 1967 Stanley Milgram at Harvard studied the "distance" between people in social networks reported in his paper "*The*

Small World Problem." He discovered empirically that people are separated by about six connections, an idea made popular in John Guare's play "*Six Degrees of Separation*" in which the character Ousa tells her daughter, "Everyone on this planet is separated by only six other people. Six degrees of separation."

The notion of distance in a network is a precise concept in mathematics. It was used famously by Cornell applied mathematicians Steve Strogatz and Duncan Watts in 1998 [9] to shed light on the Milgram result, and offer a precise definition of *small worlds*. Networks with small average distances between nodes and high degrees of clustering are called *small-world networks* [8].
In 2000, Jon Kleinberg [10] used the idea of a small world to facilitate algorithmic navigation in the Web – a very large network artifact.

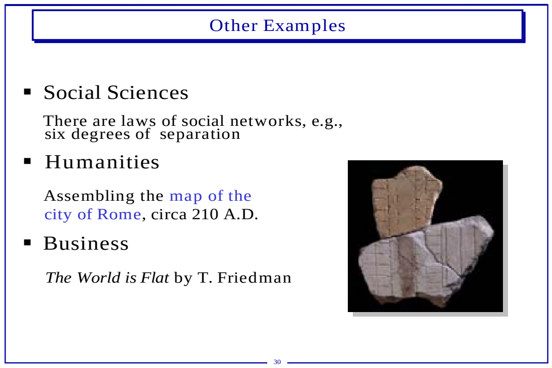
In addition to the Web and social networks, the graph of protein to protein interactions in cells is another *small world network*. The similarity between cells and social networks led the physicist Albert Libchaber to say in his 2004 Bethe Lecture how remarkable it is that, "we live the way we are built."



The study of *networks* is natural in computing and information science because the discrete mathematics used to analyze them inspires and guides algorithm discovery, and the algorithms are exceptionally practical. For example, the e-mail routing protocols have revolutionized communications. The linking structure among on-line documents is a basis for semantic analysis as well as for search. Complex systems such as cells and ecological systems are modeled by networks, and they can be analyzed by simulation and computational logic.

5.2 Arts and Humanities

We also see emerging computational methods in the humanities. Historians, for example, have clear needs to authenticate their primary sources, and computational techniques will be developed to achieve very high standards for the authenticity of digital information. We see another illustrative example in Marc Levoy's project on the *Forma Urbis Romae*, where computers have helped create new primary data from remaining shards of the great stone map of Rome circa 210 AD [6] by assembling small pieces into section of the original map. Every restored precinct gives historians new information. The key to doing this was to represent the shards so that they could be treated as geometric puzzle pieces that computers could attempt to assemble. The computers were then able to find enough new possible assemblies of pieces to keep historians active for years, saving months of puzzle solving in the basement of a museum.



Digital Arts are a thriving and respected area of modern art, spanning music, the visual arts, dance and film, from special effects and animation to interactive art. The computer, with its digital inputs and outputs, has become an expressive device of unmatched flexibility. The economic model for this art is also novel because elements of an international museum and gallery experience are freely available on the Web, as in the Guggenheim Museum's Virtual Projects.

What is different about the relationship between the humanities and the sciences in the age of Digital Information is that there is a strong tie between the highly technological information sciences and the humanities. That does not exist with the other technological sciences, such as the physical and life sciences. The intellectual ties are more like those between the social sciences and the humanities where disciplines such as linguistics and history straddle the boundary.

This means that we can imagine an "information-intensive" branch of history that becomes technical. Already *computational linguistics* has made important contributions. For example, it is very technical, requiring computer labs that CIS funds.

There are other threads that pull history, philosophy, music and literature into informatics the way they are pulled into psychology. Psychology is framed by the mind/body distinction and is informed by the emotional force of poetry and music. Likewise informatics is framed by the mind/software versus body/hardware distinction, and it strives to create affects in its animated characters. As another example, informatics must understand books the way a humanist does as well as the way a librarian does – say appreciating the creative, emotive, and practical dimensions of personal libraries. Because the modes of thought and the knowledge paradigm of informatics are so novel, informatics needs an historical perspective to chart development paths that promote wisdom and human values as it moves along, seemingly driven by a mysterious force demanding extensions and alternatives to human intelligence. Eventually informatics must face the collective id that manifests itself in the Frankenstein genre. This cannot be done without the humanists engaging individual scientists and the discipline as a whole, as Lewis Mumford does in his profound critique of "technics" in *The Myth of the Machine* [11].

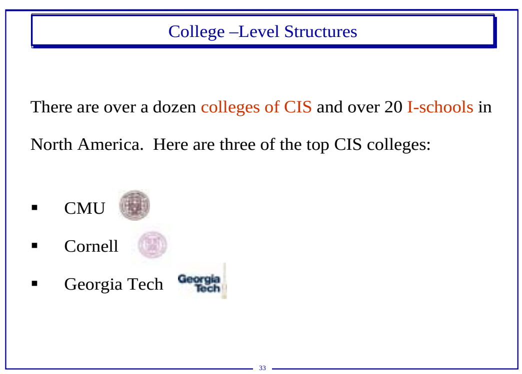
6. College-level Informatics Academic Units

We have seen several examples of how the ideas from computer science have had a major impact on science and mathematics. For instance, algorithms for comparing 3d shapes are now embedded in the fabric of genomics. The idea of a theorem prover has opened a new branch of mathematics. The idea that algorithms have a measurable computational complexity is important in all branches of science. Results about networks are a basis for understanding global scale phenomena previously inaccessible to analysis. The list goes on, showing that computing and information science is an essential intellectual partner in all sciences.

The deeper lesson from these examples is that the ideas, methods, discoveries, and technologies of informatics *have changed how we know and how we discover* [3]. This constellation of attributes, from ideas to technologies, is a new *knowledge paradigm*. There is a paradigm for the physical sciences and engineering, defined by ideas, methods, discoveries and technologies. There is one for the life sciences and social sciences as well. *The informatics knowledge paradigm* is different from all of the others, and universities must accommodate it.

One way universities have responded is by creating computer science departments and majors – now almost universally available. In response to the effectiveness of the of computer science knowledge paradigm, universities are now creating larger structures – colleges, faculties, schools, divisions, etc. There are over a dozen colleges of computing and information science in the United States. In addition, many former "information schools" or "I-schools" that arose earlier, sometimes related to libraries and communication, are broadening to include computing. At Indiana for example, their *School of Informatics* now includes the computer science department, and it is similar to the colleges that started with computer science, like Carnegie Mellon's *School of Computer Science*.

I will compare three of the most prominent college-level structures that started with computer science (CS) as the initial unit. These give an idea of what "CS in the large" will be like intellectually.



Inside these units we see similar subunits common to all, the departmental substructure of a traditional college. Essentially, all have Computer Science (CS), Information Science (IS), and some form of computational science (CSE). Two of them have a form of statistics.

Comparing Structures

Cornell	CMU	Georgia
- Computer Science	- Computer Science	- Computing Science & Systems
- Statistics	- Machine Learning	
- Information Science	- HCI Institute Language Technologies	- Interactive & Intelligent Computing
- Computational Biology	- Robotics	
- Computational Science & Engineering (CSE)	- Software Research	- CSE
- (Digital Arts)	- Entertainment Technologies	

The CS departments in these structures are traditional, covering systems, theory, and programming languages. Artificial intelligence (AI) is present in other subunits as well, e.g. in Machine Learning at CMU. The Information Science (IS) departments or programs include HCI, language technologies, digital libraries, information retrieval, and information networks. In CS, the personal computer is the iconic artifact, while in IS it is the Web.

Computational science includes high performance computing and data centric computing. At Cornell it is connected to e-science in a unit called the Theory Center. This is an interdisciplinary area connected to astronomy, physics, chemistry, and engineering.

At CMU, Irvine, and Cornell, statistics is a department in the new college level structure. There were both intellectual and organizational concerns driving this change.

7. Historical Perspective and Conclusion

Historians see the rise of computing and information science as a "new industrial revolution" or an Information Revolution. The first industrial revolution was about expanding muscle power and *automating mechanical processes*. The first revolution helped deepen the physical sciences, unifying the concept of energy, forming laws of thermodynamics, applying ideas of mass, force, and work. It also created modern engineering, and it is now driven by discovery in the physical sciences.

The Information Revolution is about expanding brain power and *automating intellectual processes*. It is creating the information sciences, developing ideas of information, computational complexity, information process, networks, and so forth. It is also creating colleges of computing and information science.

Historical perspective

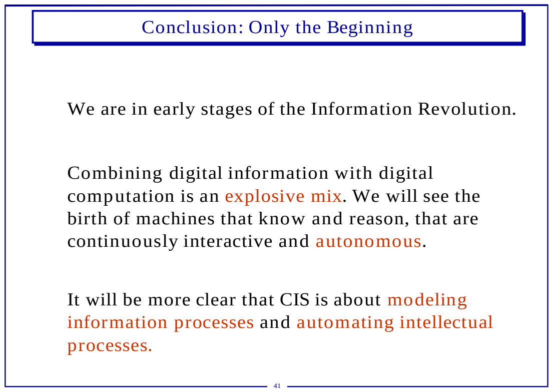
The Industrial Revolution (IR1) is about: extending muscle power (mass, energy, force, power, space, and time), automating mechanical process

The Information Revolution (IR2) is about: extending brains (information, intelligent processes, computation, complexity, and networks), automating intellectual processes

IR1 created colleges of engineering, shaping the physical sciences.

IR2 is creating colleges of computing, shaping the information sciences.

The remarkable fact is that the Information Revolution has already had such a large impact on society and the academy, remarkable because in a precise sense, it has only just begun. It is driven by an exponentially improving technology typified by Moore's Law. Unlike the first industrial revolution, it is fueled by a resource relationship of "energy to IQ" that is much less constrained than the energy to work relationship. Major improvements are not measured in small percentage increases in power, in computing and information technology, the improvements are by multiples of 100% every year! This is unprecedented.



References

1. Robin Wilson, *Four Colors Suffice: How the Map Problem Was Solved*, Princeton University Press, 2004

2. George Szpiro, *The Kepler Conjecture*, Wiley & Sons, 2003

3. Robert L. Constable, *Transforming the Academy: Knowledge Formation in the Age of Digital Information*, 2005, http://www.cis.cornell.edu/Dean/Presentations/Articles/TransformingtheAcademy.pdf

4. Sylvia Nasar and David Gruber, *The New Yorker* "Manifold Destiny, A legendary problem and the battle over who solved it." Issue of 2006-08-28

5. *2020 Vision: How computers will change the face of science*, Nature March 23, 2006, 440,14-21 http://research.microsoft.com/towards2020science/

6. Marc Levoy, *Digital Forma Urbis Romae Project*, http://graphics.stanford.edu/projects/forma-urbis/

7. Georges Gonthier, A Computer Checked Proof of The Four Color Theorem, http://research.microsoft.com/~gonthier/4colproof.pdf

8. A-L Barabasi, *Linked: The New Science of Networks*, Perseus Publishing, Cambridge, MA, 2002

9. Duncan Watts and Steve Strogatz, *Collective Dynamics of 'Small World' Networks*, *Nature* 393, 1998.

10. Jon Kleinberg, *Navigation in a Small World*, *Nature* 406, 2000.

11. Lewis Mumford, *The Myth of the Machine: Technics and Human Development,* Harcourt, Brace & World, NY, 1967.