Steffen Staab
University of Karlsruhe
sst@aifb.uni-karlsruhe.de

Neurons, Viscose Fluids, Freshwater Polyp Hydra—and Self-Organizing Information Systems

I first encountered the principle of self-organization in my last year of high school, when our teacher had us give a series of talks on a high-level view of the works of Ilya Prigogine, Humberto Maturana, Hermann Haken, and others. My talk happened to be on self-organization in the neocortex, where an orchestration of neurons seemed to find their places and functions without any conductor (also see Nigel Shadbolt's article "Beyond Brittleness, *IEEE Intelligent Systems*, Nov./Dec. 2002,). The principal lesson we learned then—that complex, functioning systems might develop bottom-up rather than top-down—was very intriguing. At the end of the 1980s, however, these principles—developed since the beginning of the 1960s—appeared to be dormant in engineering. With some exciting exceptions (for example, genetic and ant algorithms or artificial life), they had not really moved into mainstream computer science.

Now, however, the idea of self-organization is more vivid than ever in computer science, becoming apparent in diverse fields relevant for intelligent systems, such as fields that are concerned with ad hoc assembly when

- The sum of the individual, rather than a centralized authority, determines the system shape
- It involves resources that no single authority can provide
- Authority can no longer remain centralized in applications such as knowledge management
- Centralized orchestration becomes too unwieldy to enforce—such as in applications with larger numbers of autonomous agents
- Mobile environments enforce ad hoc coordination

Correspondingly, our discussion here contributes to these different observations and needs. Gary Flake, David Pennock, and Daniel Fain reflect on how the Web is shaped from the bottom up. David De Roure discusses how he views e-science, requiring a grid of intelligent components interacting toward a greater understanding of the science at heart. Karl Aberer applies self-organization to peer-to-peer systems that lend themselves to information systems that do not work with centralized authority, such as knowledge management for skilled knowledge workers. Wei-Min Shen elaborates on how self-organization might be exploited in software or hardware agents. Eventually, current mobile applications often defy centralized control, as Olivier Dousse and Patrick Thiran explain. Before we start, Francis Heylighen and Carlos Gershenson briefly introduces the principles of self-organization that the other contributors exploit.

Now, are you still wondering about viscose fluids and hydra? These were some topics of our classes then. A surprising fact about the former (and in contrast to less viscose fluids such as water under "normal" circumstances) is that if you cautiously warm a flat dish of viscose fluid underneath, you will observe that the heat does not propagate chaotically. Rather, inside the fluid, macroscopical structures (rolling cylinders) show up that remain stable (unless you overheat) and transport warmth to the top. You will find out about the latter in the remainder of this department. And you will see by these examples that the transfer of self-organization from a descriptive approach in the sciences to an engineering approach in intelligent systems now appears imminent.

The Meaning of Self-Organization in Computing

Francis Heylighen and Carlos Gershenson,
Free University of Brussels

The first user-friendly PCs in the 1980s and the Web in the 1990s unleashed a wave of innovation that seems to have drowned in complexity and confusion. Software developers are scrambling to keep their systems up to date with all the new standards, plug-ins, and extensions. Although we constantly hear announcements of spectacular innovations, few seem to reach maturity. The problem is, developers tend to underestimate the task environment's complexity: Today's information systems depend on so many modules, data sources, network connections, and input and output devices that predicting or controlling their interactions has become impossible. The result is software full of bugs, corrupted data, security holes, viruses, and other potentially catastrophic side effects. Moreover, systems have become so complex that the human mind can no longer learn or remember all procedures needed to use them. Mix in the constant change in hardware, software, protocols, data, and user expectations, and you have a recipe for chaos.

This complexity bottleneck not only creates stress and confusion, it also severely limits the speed of further progress. The number of possible interactions grows exponentially with the number of components. Because components and their capacity increase exponentially, overall complexity increases superexponentially. Developers' cognitive capacity obviously increases much more slowly, so it lags further and further. This means that large projects, such as the Semantic Web, will either get endlessly delayed or end up with unworkable products.

We need a radically different approach to overcome this bottleneck. One step forward is IBM's autonomic-computing initiative (www.research.ibm.com/autonomic). IBM researchers envision systems that function largely independently from their human supervisors, adapting, correcting, and repairing themselves whenever a problem occurs. IBM uses the metaphor of the autonomic nervous system, which runs our body for us without conscious intervention. How they hope to achieve this is less clear. Their suggestions seem to center around models of feedback, adaptation, and control first proposed in the 1950s. Although we applaud this "cybernetic" approach to computing, we believe an even more radical vision is needed: self-organization.

Self-organizing systems

A self-organizing system not only regulates or adapts its behavior, it also creates its own organization. In that respect it differs fundamentally from our present systems, which designers create. We define organization as *structure with function.* Structure means that a system's components are arranged in a particular order. It requires both connections that integrate the parts into a whole and separations that differentiate subsystems to avoid interference. Function means that this structure fulfills a purpose.

Designers obviously create systems for a particular purpose. A watch's function is to tell time, a database's is to store data, and a spreadsheet's is to calculate. But natural systems have functions, too. Roots exist to extract nutrients from the soil, stomachs to digest, and eyes to see. In the 18th century, the sophisticated structures and functions of living organisms led William Paley to argue that they must have an intelligent designer—God. We now know that no designer is necessary to produce such intelligent organization; natural systems appear to have emerged and evolved without outside intervention or programming. Yet, they are incredibly robust, flexible, and adaptive, tackling problems far more complex than any computer system.[1,2]

Self-organization then means that a functional structure appears and maintains itself spontaneously. The control needed to achieve this must be distributed over all participating components. If it were centralized in a subsystem or module, then you could in principle remove this module, and the system would lose its organization. Remove the processor chip from a computer and it becomes useless. Take any small piece of tissue from a living brain (as commonly happens during brain surgery), and the brain will continue to function more or less as it did before.

Self-organizing systems are intrinsically robust—they can withstand various errors, perturbations, or even partial destruction. They will repair or correct most damage themselves, returning to their initial state. When the damage becomes too great, their function will start to deteriorate, but "gracefully," without sudden breakdown. They will adapt their organization to environmental changes, learning new tricks to cope with unforeseen problems. Out of chaos, they will generate order. Seemingly random perturbations will help—rather than hinder—them in achieving an ever better organization.

This description might sound too good to be true. Yet, plenty of systems exist that exhibit these qualities. We find them to varying degrees in organisms, brains, ecosystems, societies, markets, swarms, and dissipative chemical systems. Computing also offers a few examples. Scientists designed the TCP/IP protocol that underlies the Internet to be robust enough to maintain communication during a nuclear war. It achieves this by cutting up messages into packets that are sent through different routes and reassembling them at the destination—resending those that get lost if necessary. Neural networks that have learned to recognize patterns, such as handwriting, still produce pretty good results when part of their nodes and links are deleted. Genetic algorithms solve complex problems by evolving subsequent generations of candidate solutions.[3] They mutate, recombine, and reproduce only the best ones, until they find a good-enough solution.

Mechanisms of self-organization

These computing examples show the power of self-organization but only in limited contexts, where both the components and their desired functions are well defined. How can we apply self-organization to an environment as complex and diverse as the Web, a corporate intranet, or even a desktop computer with its ever-changing software and data configuration? To achieve self-organization on that scale, we need a much deeper insight into how it works.[2] Let us return to the systems we find in nature and analyze their mechanisms in terms general enough to apply to complex information systems.

A self-organizing system consists of many interacting components, such as molecules, neurons, insects, or people. The system is dynamic; the components are constantly changing state relative to each other. But owing to mutual dependency, changes are not arbitrary. Some relative states are "preferable," in that they will be reinforced or stabilized, while others are inhibited or eliminated. For example, two molecules that approach each other in the right geometrical configuration might react—forming a chemical bond and thus a larger molecule—or simply drift past each other. Two people discussing might either find common ground and establish a working relation, or leave in disagreement.

Changes are initially local. Components only interact with their immediate neighbors. They are virtually independent of components farther away. But self-organization is often defined as global order emerging from local interactions. We can picture this process as follows. Two interacting components pass through various configurations until they find one that is mutually satisfactory—that is, stable. We might say that they have adapted to each other and now fit together. To achieve global order, this fit must propagate to the other components. For example, two molecules that have bonded might be joined by a third one, and a fourth one, and so on, eventually forming a macroscopic crystal.

Two people who discover a common interest might start talking about it to others and end up founding a club, political movement, or company. If the components are interchangeable, such as molecules of the same chemical substance, the resulting structure will be regular, like a crystal. If each components has its own, individual characteristics, such as a species in an ecosystem, the structure will be more complex. Each component must fit in its own niche within the environment formed by others.

This propagation of fit is typically self-reinforcing—additional components join ever more quickly. This occurs because a larger assembly exerts a stronger attraction on the remaining independent components, offering more niches in which they can fit. This positive feedback produces an explosive growth or reproduction of the assembly. Growth only stops when the resources are exhausted—that is, when all components that could be fit into the assembly have been fit. This might happen because the remaining components are too different to fit in this type of configuration or because they were assimilated into a rival assembly. For example, a chess club will stop growing when the remaining people in town are either not interested in chess or already belong to a different chess club.

Once the assembly stabilizes, feedback becomes mostly negative. This means that it will counteract any loss of organization. Self-maintenance has become its implicit purpose, and each component will perform its function toward this goal. For example, the rules and individual relationships in an established club make it difficult for members to switch to a rival club. The assembly as a whole, however, can still interact with other assemblies but at a different level. For example, two chess clubs might engage in an interclub tournament. So, assemblies formed from individual components start acting like higher-level components. These can in turn self-organize into even higher-level components, the way chess clubs can assemble into a federation. This process continues recursively, for as long as there are components to interact with, generating ever higher levels of complexity.

The self-organized system is stable or robust, but this does not mean static or rigid. When the environment changes, the components that directly interact with it must adapt their states until they are fit again. This fit will propagate inward, until the whole assembly is adapted to the new situation. Thus, the system constantly reorganizes, mutually balancing the different internal and external pressures for change, while trying to maintain its essential organization. The more perturbations it encounters, the larger the variety of different configurations it will explore, and therefore the "better" the eventual solution it settles in. The cyberneticist Heinz von Foerster called this principle "order from noise." The thermodynamicist Ilya Prigogine[4] called it "order through fluctuations."

The future of computing?

How can we apply this general vision to information systems? Imagine various components—hardware and software modules, files, Web sites, interfaces, and users. They all interact by exchanging information. Assume that neighboring components can mutually adapt. By sending messages back and forth, they negotiate until they achieve a common "understanding" in which they both can settle. But coordination does not stop there; it propagates back and forth between all components, creating a globally stable order. Any change, such as a newly introduced component or local breakdown, will restart the negotiation process with its immediate neighbors. Its effects will ripple further through the neighborhood until this perturbation is also absorbed and the system is back to equilibrium. Of course, we will need to overcome important hurdles before we can achieve this vision. Most obviously, we must create a universal protocol for interaction that supports unrestricted self-organization.

This all sounds very general and abstract. Can we think of more concrete applications? A well-known example of self-organization is the way ants lay trails of pheromones between various sources of food. Initially, ants explore and leave pheromones randomly, but good trails leading to rich sources through quick routes are reinforced through positive feedback, while poor trails eventually evaporate. Thus, food sources get organized into a dense, efficient network of foraging paths. Marco Dorigo, a pioneer in the domain of ant algorithms, has shown how a similar mechanism can tackle various computing problems, including the notorious traveling salesman problem.[5] An important application is the routing of messages along the nodes and links of a communication network. Efficient routes are reinforced, less efficient ones abandoned. The resulting organization adapts in real time. If routes become congested, their priority is immediately downgraded and the algorithm explores new routes.

One of us has applied a similar idea to the Web's hyperlink organization.[6,7] Our learning Web algorithms reinforce paths of links that users travel frequently, eventually replacing them with a single link. While users decide locally which link to explore next, the effects of their choice propagate throughout the Web. This should eventually lead to a global order—Web pages that fit together, in the sense that one page is very relevant for users of the other, are linked directly. Thus, documents that cover the same subject get clustered. A higher-level document can now represent this cluster or assembly, presenting an index or summary for the subject. Higher-level components themselves become linked and clustered in categories and further into supercategories. Eventually, the Web as a whole might self-organize into an efficient, hierarchically structured network of associations that adapts continuously to newly introduced documents, changes in user demand, and so on.

With some minor variations, we could apply this scheme to the construction of shared ontologies, clustering similar concepts into categories and linking the categories that are most strongly associated. Object-oriented programming also could profit from this approach, with objects mutually negotiating message-passing protocols and spontaneously assembling into higher-level objects. For example, the Swarm programming environment (www.swarm.org) supports the latter. The same goes for ubiquitous computing or intelligent environments. Various devices—such as refrigerators, thermostats, or phones—connected to a network can learn to mutually coordinate their activities, thus minimizing the user's burden.

The possibilities seem endless. Is this the future of computing? Only time can tell.

References

1. K. Kelly, *Out of Control*: *The New Biology of Machines*, Addison-Wesley, 1994.

2. F. Heylighen, "The Science of Self-Organization and Adaptivity," *The Encyclopedia of Life Support Systems*, Eolss Publishers, 2003.

3. J.H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, 1992.

4. I. Prigogine and I. Stengers, *Order Out of Chaos*, Bantam Books, 1984.

5. E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence*, Oxford Univ. Press, 1998.

6. J. Bollen and F. Heylighen, "Algorithms for the Self-Organization of Distributed, Multi-User Networks," *Proc. Cybernetics and Systems 96*, Austrian Soc. for Cybernetics, 1996, pp. 911–916.

7. F. Heylighen and F.J. Bollen, "Hebbian Algorithms for a Digital Library Recommendation System," *Proc. 2002 Int'l. Conf. Parallel Processing Workshops*, IEEE CS Press, 2002.

The Self-Organized Web: The Yin to the Semantic Web's Yang

Gary William Flake, David M. Pennock, and Daniel C. Fain, *Overture Services*

Assigning superlatives to the Web is easy: it's massive, it's dynamic, it's decentralized—it's unlike anything else in the world. But one of the Web's most amazing attributes is that it is arguably the largest self-organized artifact in existence. Every day millions of Web publishers add, delete, move, and change their pages and links, yet what results is far from random or haphazard. Rather, from these millions of uncoordinated decisions emerges a startling number of regularities and patterns. The Web programmer's task—whether working on search, collaborative filtering, data mining, e-commerce, or scientific analysis—is to improve these structures to make the Web more digestible to users. This goal complements the Semantic Web's goal: to have humans help make the Web more digestible for computers. Exploiting the self-organized Web will improve tomorrow's algorithms; manually adding computer-friendly annotations to the Semantic Web will help today's less-sophisticated algorithms cope.

Self-organization and the Web

Although Web authors' choices worldwide are largely uncoordinated, they are anything but uncorrelated. Unlike any (nondegenerate) random graph, the Web graph's large-scale structure has a "bow tie" organization that contains four distinct regions: a strongly connected core, an origination bow, a termination bow, and disconnected islands.[1] In the Web's core, hyperlinks are relatively sparse, yet they collectively possess small-world properties, forming many redundant and relatively short paths between most pages.[2] When sampled across the Web, inbound and outbound hyperlink distributions follow a clear power law distribution that resembles distributions in biology.[3] Viewed on a small scale, simple and small bipartite subgraphs are a signature of topical Web communities' formation.[4]

We also see clear structural self-organization at intermediate levels. For example, when aggregating only over a specific type of pages (for example, movie, newspaper, photography, or university homepages), hyperlink distributions shift from a strict power law to a unimodal form, not unlike the change in the biomass distribution when aggregated over a single species instead of all species. A generative Web growth model with only one free parameter explains this unusual hyperlink distribution property with remarkable simplicity and accuracy.[5] Moreover, a simple definition—that a Web community is a collection in which each member is predominately hyperlinked to other community members—has yielded an efficient procedure for identifying self-organized Web communities. Empirically, these communities are topically and textually focused, even though the identification procedure uses only hyperlinks.[6] Figure 1 depicts a few of the Web's self-organizing properties.

Web data mining

Given the Web's size and decentralization, it seems almost a paradox that pure hyperlink data mining algorithms work. Nearly all current hyperlink methods strongly overlap with methods pioneered in graph clustering and where the problems are typically NP-hard.[7] The Web's enormous size suggests that it would be a more difficult domain, yet the reality is somewhat different.

Two popular Web data mining algorithms, HITS[8] (hypertext induced topic search) and PageRank,[9] have shown considerable success when coupled with text retrieval. Both methods attempt to capture Web pages' importance by recursively analyzing how pages hyperlink to each other. The PageRank algorithm can be roughly interpreted as simulating how a random walker would traverse the Web graph over forward links if also allowed to teleport to other pages with some small probability. After completion, PageRank assigns to a Web page a score that is about equal to the probability that the random walker will visit that page. Intuitively, the PageRank score is similar to the recursive definition that a Web page is important if other important pages link to it.

PageRank clearly improves text retrieval, as evidenced by the popularity of Google (the inventors' search engine[9]), which largely attributes its competitive edge to PageRank. What is truly interesting, however, is that PageRank is an existence proof that:

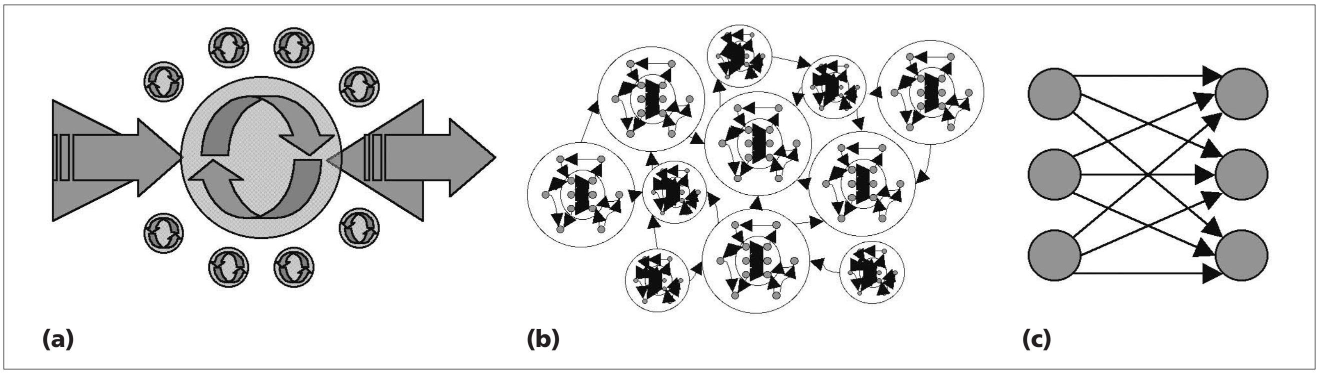


Figure 1. Self-organized Web structures of various scales: (a) the large-scale bow tie; (b) intermediate-scale Web communities; (c) small-scale bipartite community cores.

1. The Web is well behaved in that worst-case complexity results are far too pessimistic.
2. The Web's self-organization can be used to improve Web search and data mining.

Consider the first point. PageRank is technically equivalent to a power method estimation of the maximal eigenvector of a simple transformation of the Web graph's adjacency matrix. Linear algebra shows that the power method converges at a rate related to the ratio of the first two eigenvalues of the matrix being used. Simply put, the more similar the first two eigenvalues, the slower the procedure's convergence rate.

Many matrices are ill suited for power method procedures because of the convergence properties. However, in the Web's case, the power law distribution on inbound hyperlinks nearly guarantees that the Web will never possess such pathologies because Web pages with many other pages linking to them are exceedingly rare and in some sense very competitive with each other. In other words, the Web doesn't seem to like having two maximal eigenvalues of nearly equal value.[10]

As for the second point, think of PageRank as something of a collective voting scheme, where pages not only vote for each other but also vote to determine how many votes each should have. The votes are labeled, in a sense, with anchor text; a string match in a referring page's anchor text is weighed more heavily than a match in the target page itself. The links and labels from the Open Directory Project (ODP)—a voluntary effort to categorize Web sites—are the most important. In such a collective, distributed framework, it isn't at all obvious that the aggregate vote would add any value to the retrieval task. But, when coupled with text retrieval, PageRank is remarkably adept at producing the "correct" answer when correctness and popularity are highly correlated.

Top-down, bottom-up, and lessons learned

Recently, many have championed the Semantic Web[11] as a means to improve information retrieval on the Web. Proponents argue that the Web is ill suited in its current form for automated processing because the information is unstructured to the point that semantics are nearly impossible for machines to infer. In the Semantic Web, authors will use a markup language to annotate data with semantic labels so machines can identify content meaning and use rules for manipulating semantic information appropriately. In a best-case scenario, the markup language will be nearly complete and agreeable, and used consistently by Web authors. Authoring tools might generate the markups implicitly, but such markups will need to make sense alongside those the authors add manually. Implicit markup is easy to envision for product catalogs, but semantically marking up long passages of text—magazine articles, for example—could be daunting.

Realistically, Semantic Web advocates understand that some holes and inconsistencies in the markup language are inevitable and that author adoption rates and proficiencies will be heterogeneous. Witness the failure of Xanadu,[12] a hypertext framework arguably superior to HTML/HTTP. It failed partly because its rules and guarantees imposed too great a burden on authors. A simple example of Web authors' natural laziness is the prevalence of rasterized text unmarked by ALT text tags. Some of Xanadu's ideas were revived in less-demanding forms. For example, instead of keeping an up-to-date set of backlinks at the target page, Web backlinks are retained in search engines. But relaxing enforcement in the Semantic Web will lead to another form of self-organized entity, even if more structured than today's Web. To add to the confusion, as long as there is search spam, a contingent supplying false information through metadata will exist. This underscores the value of objective third-party annotation, even when minimal, such as ODP.

Users will benefit most if work on creating the Semantic Web coevolves with work on tools for data-driven analysis of the self-organized Web.

A complementary best-case scenario envisions Web algorithms intelligent enough to infer semantics from the current, nonannotated, self-organized Web, without the aid of semantic markups. Information extraction tools are progressing, making inroads on problems such as parsing resumes and finding contact information, but today's best algorithms still fall woefully short of people's capability to extract meaning from the Web. We have only scratched the surface of the self-organized Web's potential. Future data mining methods will use efficient algorithms optimized for the expected case of Web data (rather than the worst case or random case), use predictable Web properties to reduce problem sizes and input dimensionality, and have performance bounds consistent with generative Web growth models. With these algorithms, search engines will be able to cluster and classify the entire Web along multiple dimensions (topic, type, and genre, for example). The most advanced search engines and autonomous Web agents will be able to use richer forms of metadata if and when a new markup structure is adopted.

But for the foreseeable future, efforts to leverage the self-organized Web will complement efforts to build the Semantic Web. Both open up opportunities for innovative new algorithms—data mining on one hand, symbolic inference on the other. Where these efforts meet, tools will arise for vastly improved search, filtering, personalization, economic efficiency, and scientific understanding of the social forces and trends reflected in the Web. We believe that the Web's properties—structure, content, and explicit or inferred metadata—will continue to evolve in a decentralized and self-organized way. Users will benefit most if work on creating the Semantic Web coevolves with work on tools for data-driven analysis of the self-organized Web.

References

1. A. Broder et al., "Graph Structure in the Web: Experiments and Models," *Proc. 9th World Wide Web Conf.* (WWW9), Elsevier, 2000.
2. D.J. Watts and S.H. Strogatz, "Collective Dynamics of 'Small World' Networks," *Nature*, vol. 393, no. 6684, June 1998, pp. 440–442.
3. A.L. Barabási and R. Albert, "Emergence of Scaling in Random Networks," *Science*, vol. 286, no. 5439, Oct. 1999, pp. 509–512.

4. R. Kumar et al., "Trawling the Web for Emerging Cyber-communities," *Proc. 8th World Wide Web Conf.* (WWW8), Elsevier, 1999.

5. D.M. Pennock et al., "Winners Don't Take All: Characterizing the Competition for Links on the Web," *Proc. Nat'l Academy of Sciences*, vol. 99, no. 8, Apr. 2002, pp. 5207–5211.

6. G.W. Flake et al., "Self-organization of the Web and Identification of Communities," *Computer*, vol. 35, no. 3, Mar. 2002, pp. 66–71.

7. M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, 1979.

8. J. Kleinberg, "Authoritative Sources in a Hyperlinked Environment," *J. ACM*, vol. 46, no. 5, Sept. 1999, pp. 604–632.

9. S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," *Proc. 7th World Wide Web Conf.* (WWW7), Elsevier, 1998.

10. F. Chung and L. Lu, "The Average Distances in Random Graphs with Given Expected Degrees," *Proc. Nat'l Academy of Sciences*, vol. 99, no. 25, Dec. 2002, pp. 15,879–15,882.

11. T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, no. 5, May 2001, pp. 34–43.

12. T.H. Nelson, "Xanalogical Structure, Needed Now More Than Ever: Parallel Documents, Deep Links to Content, Deep Versioning, and Deep Re-use," *ACM Computing Surveys*, vol. 31, no. 4, Dec. 1999, pp. 194–225.

On Self-Organization and the Semantic Grid

David De Roure, *University of Southampton*

The term *grid computing* has previously suggested a world of networked supercomputers, Beowulf clusters, fat pipes, and petabyte storage. But now the "grid problem" is defined as "resource sharing and coordinated problem solving in dynamic, multi-institutional virtual organizations."[1] High-performance computing might once have focused on accelerating scientific computation, but contemporary grid computing is also about accelerating the scientific process. It is the infrastructure of e-science, and indeed e-engineering, and potentially e-many-other-things too.

The notion of "the Grid" is an analogy to the electricity power grid—you can plug into the common interface to tap its computational power. The middleware that implements it, de facto the Globus Toolkit, hides the heterogeneity of the diverse computational resources. In principle, the middleware makes it easier to cross not only operating system and version boundaries but also organizational boundaries. This is significant because the computational power can come from different power generators.

Grid infrastructure and applications are typically service oriented, and in the last year or so, the middleware effort has fallen in line with the prevailing movement to Web services to enjoy a degree of industry-standard interoperability. This takes the form of the Open Grid Services Architecture,[2] an enhanced Web services model created to meet the grid community's specialist requirements.

A huge number of grid projects exist, fueled by national funding programs such as the UK's US$500 million e-Science program, which reaches across a spectrum of research disciplines. Every time we embark on a new project, we would like to reuse and repurpose the data, services, software components, resources, and indeed knowledge from our own scientific communities and others' previous projects. So, where we once needed the Grid to hide computational resources' heterogeneity, the new Grid problem is hiding the heterogeneity of the bits and pieces needed to quickly and easily assemble new projects, or even new grids. The vision is a generically usable e-science infrastructure, comprising easily deployed components whose utility transcends their immediate application. Our goal is the automated construction of the desired services, adapted to the current requirements and circumstances. This is the self-organization we seek—the assembly and adaptation of Grid services.

The Semantic Grid

Underlying any form of automated composition we need an infrastructure where all resources, including services and workflows, are adequately described in a machine processable form; that is, knowledge is explicit—Semantic Web technologies provide the infrastructure. This has led us to the Semantic Grid,[3] where we apply those technologies in grid-computing developments, from grid infrastructure machinery (such as the Globus Toolkit's grid services) to grid applications. "Semantics" permeates the full vertical extent of the Grid and is not just a semantic layer on top; it is semantics in, on, and for the Grid. Some of these Semantic Web technologies are ready for immediate deployment (the Resource Description Framework tools, for example), while others are on the research agenda. Realizing the Semantic Grid vision involves bridging two rather disjoint communities, each with its own standards process. At one end is the World Wide Web Consortium and the Semantic Web research community; at the other is the Global Grid Forum and Grid research community. GGF now has a Semantic Grid Research Group, chartered to help the Grid developers apply established metadata technologies while tracking, for example, the evolution of OWL (the Web Ontology Language) and associated tools.

The Semantic Grid is a complex and large-scale piece of machinery. Some aspects of the system are beyond centralized control. Looking inside will let us see the opportunities for self-organization.

The Semantic Grid's computational fabric comprises interconnected processors—perhaps with some dedicated networking locally—that are globally interconnected through the Internet. The Internet is undeniably a large-scale decentralized system. In fact, the Internet is an adaptive system, comprising simple components with local knowledge that together provide a global network service that copes with network component failure. Above this, we might consider the Web's machinery to be a large-scale decentralized system, but I dispute the scale of distributed processing—a typical Web transaction involves just a server, a browser, and a proxy or two. The evolution of the Web infrastructure's deployment, however, is an example of a self-organizing system. The large-scale decentralized system in the Web is actually its content, the global linking infrastructure that has generated much analysis in recent years. The Semantic Web gives us a much richer way to describe the associations within Web content, transcending the limited expressiveness the simple navigational linking model provides. As the decentralized metadata grows, it is interlinked by the resources it describes. This is a critical basis for Semantic Web content's self-organization.

The Semantic Web assumes a service-oriented model, such as Web services or

Grid services, or agents, which are producers, consumers, and brokers of services. We are beginning to see the application of Semantic Web technologies within that infrastructure, in multiagent systems, and through Semantic Web services. This emerging, semantically rich service-oriented infrastructure is an important large-scale, decentralized system in our Semantic Grid. It's the last to emerge, but it fits in the middle—it is the semantic middleware.

Opportunities for self-organization

These three ingredients—communication or computation, semantic middleware, and semantic content—provide opportunities for self-organization that support the vision of the self-organizing Semantic Grid.

The first is at the level of job submission and control over the computational fabric of large clusters or supercomputers, and the aggregations of these into grids. A single parallel computation can be predictable, with a known shape and extent, or dynamic and evolving. When you run an entire user community on that fabric, with myriad, diverse computations accessing distributed data, you have what is essentially a huge optimization problem. With job scheduling currently managed by individual schedulers, we should enjoy self-organization's benefits as we scale up. We would surely benefit from the fault tolerance that self-organization could provide because component failure is inevitable in systems of this scale.

The second opportunity is at the service level because this level has some homogeneity and scale and is where the many grids are beginning to meet to form the Grid. The organization task here is discovering and binding the services together to meet the requirements. This might be highly engineered, with prescribed workflows, for example. It might also be highly dynamic, with opportunistic use of services that have been instantiated and local data. The picture bears some relationship with peer-to-peer. Again, we have a massive distributed optimization problem.

The third opportunity relates to the information and knowledge that is the "stuff" of e-science—the computations and experiments' inputs and outcomes, and the descriptions of the processes that created them. Here we have scale, and we need to join the information with the processes that consume and generate it. Science is a collaborative process, and we might also view matching information to individuals as a process of organization.

Reality intrudes

But is the Grid really a large-scale distributed system? Anticipation of distributed processing on a massive scale has motivated its development. But currently many disjoint grids exist, just as many networks once existed that eventually combined into the internetwork known as the Internet. We can foresee the current grid islands similarly aggregating into an "intergrid," known as the Grid. Surely this kind of aggregation was the reason for the Web's clichéd "exponential growth." Once the Grid aggregates, we will have a large-scale distributed system with highly engineered, complex infrastructure machinery. Internet or Web nodes forward packets and documents; the Grid's nodes will do that and some computation. We do not yet know how much intergrid coupling will occur at the lower level (Web couples using the HTTP protocol) versus at a higher level (as with the integration of scientific services).

The Grid community is a coherent, organized, enthusiastic community with real application drivers and could genuinely benefit from semantic interoperability.

Of course, the notion of systems that can self-diagnose and self-repair is compelling, and the Grid community has recently shown some enthusiasm for *autonomic computing*,[4] drawing inspiration from the autonomic nervous system. We are already achieving some degree of fault tolerance by engineering an infrastructure that monitors and repairs faults. This touches a key point: Do we need a self-organizing system, or can we engineer the necessary behavior? To put it another way, who needs slime mold when shell scripts will do? While our computing infrastructure is organized as an aggregation of managed resources, we can achieve a degree of centralization in control and coordination of the function of those resources. We are not obliged to be decentralized. Self-organization is attractive when it offers some optimization without risk—as Web caching does—but can we deploy it in a world where code is handcrafted for performance?

Of course, this vision assumes we have a Semantic Grid and shares some of the usual Semantic Web obstacles. It requires a metadata-enabled world, but where will the metadata come from? Except where classifications and domain-specific ontologies already exist, the Grid community must agree on the schema and ontologies. What will motivate this effort? I would claim that with the Semantic Grid, we have a fighting chance. The Grid community is a coherent, organized, enthusiastic community with real application drivers and could genuinely benefit from semantic interoperability. It will also stress Semantic Web technologies because it demands an extraordinary degree of automated interoperability, such as ontology mapping, and large-scale performance.

The vision also assumes masses of computational power, but we can be somewhat confident in this trend. Apart from more powerful processors, we will simply have more processors, emphasizing the issues of scale. Consider the provocative amorphous computing example of embedding processors in materials (for example, "concrete by the megaflop"), which, incidentally, insists on self-organization (see www.swiss.ai.mit.edu/projects/amorphous). Notice also the broad similarity between the service-oriented adaptive Semantic Grid vision and the service-oriented adaptive ubiquitous-computing vision. Several services must come together locally to meet our needs on a distributed architecture where devices come and go.

Back to the future

Let's push that autonomous processing a stage further and combine self-organizing services with self-organizing information and knowledge. Our self-organizing Semantic Grid is now a constantly evolving organism, with ongoing, autonomous processing rather than on-demand processing. This evolving, organic Grid can generate new processes and new knowledge. Consider, for example, a genetic approach to creating new workflows to solve a scientific problem. Imagine watching this autonomous, self-organizing Semantic Grid in action, visualizing the information flows in the system. It might be very similar to watching a bunch of e-scientists at work. But surely individual scientists bring problem-solving insight to the process? Maybe, but on the macro scale is there much difference? Remember, this is Grid computing, and we have the facility for large-scale processing. Additionally, people are slow. The DNA microarray has caused a revolution by permitting masses of parallel experimentation, as has combinatorial chemistry. Experimentation in silico is another such shift. Some Grid applications can and will absorb as much computational power as is available to produce better or faster results, but we will also be able to do more parallel experiments more quickly, monitoring and steering them dynamically.

I've shown that we have some opportunities for self-organization, but the current infrastructure might not be ready for it. And I've shown a vision of a self-organizing Semantic Grid that serves the scientist by assembling services on the fly and could even embark autonomously on scientific discovery. The challenge is to determine which parts of our self-organization are highly engineered, perhaps through the proactive behavior of agents, and when to let go.

References

1. I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations." *Int'l J. Supercomputer Applications*, vol. 15, no. 3, 2001.

2. I. Foster et al., "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration," Open Grid Service Infrastructure WG, Global Grid Forum, 22 June 2002; www.globus.org.

3. D. De Roure, N. Jennings, and N. Shadbolt, "The Semantic Grid: A Future e-Science Infrastructure," *Grid Computing: Making The Global Infrastructure a Reality*, F. Berman, A.J.G. Hey, and G. Fox eds., John Wiley & Sons, 2003, pp. 437–470; www.semanticgrid.org.

4. IBM, *An Architectural Blueprint for Autonomic Computing*, Apr. 2003, www.ibm.com/autonomic.

Self-Organization and P2P Systems

Karl Aberer, *Swiss Federal Institute of Technology, Lausanne*

The Internet has enabled the provision of global-scale distributed applications. Such applications' providers include some of the best-known Internet companies, such as eBay, Yahoo, and Google. These applications' centralized client-server architecture leads to heavy resource consumption at the server sites; for example, Google is operating a workstation cluster of about 15,000 Linux servers. From this observation, you might conclude that providing a global-scale application necessarily implies a major development, infrastructure, and administration investment. However, a new class of applications—initially developed to help users share information, such as music files, recipes, and so on—shows that this conclusion is inaccurate. These systems are commonly called *peer-to-peer file-sharing systems*. The main service they provide is *resource location*, which lets users search for resources based on a *resource identifier*. Napster was the first and most famous proponent of this new class of systems, which essentially exploit the principle of *resource sharing*. The Internet makes plenty of resources available at its "edges"—that is, at end-user computers. By integrating these computers into a larger system, user communities can build applications at a global scale without experiencing the investment bottleneck mentioned earlier.

In Napster's case, the coupling of resources was facilitated through a central directory server, where users registered their music files and could search for other users' files. After locating the files, users could download them directly from other peers—therefore, we consider Napster a P2P system. Through this scheme, Napster exploited several resources the community of cooperating peers made available. The most notable of these was storage and bandwidth for handling large music files; less obviously, users provided knowledge when annotating files during registration. Finally, music file ownership was a key factor in Napster's fast adoption, but it also caused the music industry to react strongly to potential copyright infringement. Napster was eventually shut down, which was possible because the system depended on a central directory server.

Another music file-sharing system, Gnutella, then entered the stage. Although Gnutella provides essentially the same functionality as Napster, it uses no central directory server. Search requests are simply flooded over the network. Thus Gnutella avoids any distinguished component in its architecture: It is a fully decentralized system. Because Gnutella avoids any single point of failure, attacks (legal, economic, or malicious) are very difficult.

Self-organization in P2P systems

Gnutella's lack of central coordination makes you wonder how any useful behavior occurs. Unsurprisingly, this is where self-organization comes into play. But what exactly does it mean, in Gnutella's case, to be self-organizing? To shed light on this question, we must more closely understand how Gnutella operates. In fact, the Gnutella protocol has two components. The first is a *network maintenance protocol*, which constructs and maintains a network of Gnutella peers called the *overlay network*. The other is a *search protocol*, which helps users locate resources in the overlay network. By using the network maintenance protocol, a peer discovers new peers in the network by flooding "ping" messages. Other peers respond with "pong" messages announcing their network participation. From the responses, the first peer selects certain peers and establishes direct network links with them. In Gnutella, each peer maintains a fixed number of active links. So, using the network maintenance protocol, the network continuously changes its structure.

This process is clearly self-organizing in the sense of Francis Heylighen's characterization: "The basic mechanism underlying self-organization is the noise-driven variation, which explores different regions in a system's state space until it enters an attractor."[1] The state space consists of all directed overlay network graphs with constant out-degree, and the noise results from the random time of node joins, communication latency, and autonomous local decisions on connectivity. Many recent experimental investigations have attempted to identify the attractor—that is, the resulting overlay network's graph structure. Researchers have found two main properties.[2]

First, the network has a small diameter, which ensures that a message-flooding approach for searching a resource in a network works with a relatively low time to life (approximately seven network hops).

Second, the overlay network's node degrees follow a power law distribution. So, a few peers have many incoming links, whereas most peers have only a few such links. Researchers have discovered power law distributions of node degrees for many types of networks, such as the World Wide Web, citation networks, or genetic networks. The property is credited to the network construction mechanism—the networks continuously grow, and new nodes preferentially attach to already well-connected nodes.[3]

Both of these properties affect a Gnutella network's performance. This is apparent for the first property. For the second property, we can observe that the highly connected nodes are typically those that provide high bandwidth and have high availability. These nodes naturally form a backbone that (presumably) optimizes resource consumption, although we can't say whether such effects were expected during the Gnutella protocol's design.

Despite the similarity of the network maintenance and search protocols in Gnutella, the two protocols serve fundamentally different purposes and are independent. The network maintenance protocol implements a self-organization process that changes the system state—that is, the overlay network's structure. The search protocol implements a distributed algorithm for searching the overlay network. Other Gnutella-style networks have more efficient search protocols that reduce the high message load for search, which limits the total system throughput. Examples are the random-walker model[4] and the percolation search model.[5] These approaches assume a Gnutella-style network; that is, they support a combination of the Gnutella ~~group mem-~~
n.m. ~~bership~~ protocol and an improved search protocol. These observations illustrate that a fully decentralized P2P resource location system comprises two orthogonal components that must complement each other:

- *Network construction* is a self-organizing process that exhibits adaptive behavior with respect to resource consumption.
- *Search* is a randomized, distributed algorithm that exploits the emergent network structure.

However, this observation applies so far only to unstructured P2P systems, such as Gnutella, which you can characterize by the fact that peers are unaware of the kinds of resources neighboring peers maintain. So, peers are fairly independent and can act autonomously, which facilitates self-organization. As a drawback, searches are forwarded "blindly." This results in the typical high message loads because to locate a resource, all peers in the network must eventually be contacted, setting aside possible optimizations using replication.

Unstructured to structured P2P systems

To reduce the message bandwidth during searches in P2P systems, researchers have developed several decentralized data access schemes, commonly called *structured P2P systems*. You can characterize them by

Using a decentralized self-organizing process, we constructed a prefix-routing infrastructure that adapts to a given distribution of resource identifiers.

peers' specialization to specific kinds of resources and the use of *routing tables* to selectively forward search requests to peers that are more likely to provide a requested resource. Frequently, such systems are based on variations of *prefix routing*. Prefix routing is based on an underlying logical trie structure from which each peer randomly selects a trie node. This selection defines the resources the peer holds and serves as the peer's identifier. Additionally, peers maintain *routing tables* with links to other peers along the path from the trie root to their own trie nodes, so the system can forward search requests not pertaining to one peer's trie node to other peers associated with other trie branches. This organization enables answering search requests with a logarithmic number of messages in the number of peers participating in the network. This improvement in search performance from linear to logarithmic cost comes at a price: peers are no longer independent. When peers update their data or when the network changes its structure—as when nodes join the network, for example—multiple peers are affected. Specifically, updates to the routing tables become necessary. So, structured P2P networks require distributed algorithms to maintain the peer network's consistency during updates.

What about self-organization in structured P2P systems? In many original designs, it is not present. The network construction is based on distributed algorithms for node join that are (mostly) deterministic once peers have chosen their identities. Typically, peers perform this choice randomly to achieve a uniform coverage in the resource identifier space. When resource identifiers carry application meaning (for example, being verbose filenames), they are generally nonuniformly distributed in the identifier space. Consequently, choosing peer identifiers uniformly randomly from the identifier space results in nonuniform workload for peers. So, adapting the peer-identifier distribution with respect to the actual distribution of resource identifiers is necessary. Because achieving this adaptation through central control is undesirable, a self-organizing process, as for Gnutella, becomes the natural solution.

Can we really adapt structured P2P networks in this way, while maintaining the dependencies among the peers implied by the routing infrastructure? My colleagues and I answered this question positively when developing P-Grid.[6] Using a decentralized self-organizing process, we constructed a prefix-routing infrastructure that adapts to a given distribution of resource identifiers. The process is based on pairwise interactions of peers in which they locally decide (if enough data is present) whether to refine the routing infrastructure in a given resource identifier subspace. Consequently, the shape of the (virtual) trie underlying the routing tables' organization will adapt to the resource identifier distribution. This leads to an interesting problem with respect to search. In the worst case, for degenerate resource identifier distributions, the trie shape no longer provides an upper bound for search cost because it might be up to linear depth in network size. However, we can show that for a (sufficiently) randomized selection of links to other peers in the routing tables, probabilistically the search cost in terms of messages remains logarithmic; it is independent of the length of the paths occurring in the virtual trie.[7] This result illustrates that the principal observation we derived for unstructured P2P systems also holds in structured P2P systems. A fully decentralized and adaptive P2P resource location system builds on two principles: a self-organization process to construct the network and suitable randomized, distributed algorithms that can exploit the specific emergent network structure.

Hierarchies of self-organization

Resource location is not the final purpose of P2P systems but rather the most basic infrastructure for enabling higher-level services. Nothing prevents us from assuming that these services in turn build on self-organization principles. At ~~the Swiss Federal Institute of Technology~~ EPFL, we have started investigating some examples of such self-organizing services for peer-identity management in the presence of changing physical addresses, for reputation and trust management, and for discovering heterogeneous schema correspondences (see www.p-grid.org for details). An interesting problem for future research in such multilayer self-organizing systems is the higher-level self-organization processes' dependencies on lower-level ones. We still don't know to what extent we can construct such complex, multilayer systems on the basis of analytically understanding their components and at what point we will have to rely more and more on adaptive, self-organizing processes to construct such systems.

References

1. F. Heylighen, "The Science of Self-Organization and Adaptivity," *Principia Cybernetica Web*, http://pespmc1.vub.ac.be/SELFORG.html.
2. M. Ripeanu and I. Foster, "Mapping the Gnutella Network: Macroscopic Properties of Large-Scale Peer-to-Peer Systems," *1st Int'l Workshop Peer-to-Peer Systems* (IPTPS 2002), LNCS 2429, Springer-Verlag, 2002.
3. A. Barabasi and R. Albert, "Emergence of Scaling in Random Networks," *Science*, vol. 286, 1999, pp. 509–512.
4. Q. Lv et al., "Search and Replication in Unstructured Peer-to-Peer Networks," *Proc. 16th Ann. ACM Int'l Conf. Supercomputing*, ACM Press, 2002.
5. N. Sarshar, V. Roychowdury, and P. Oscar Boykin, "Percolation-Based Search on Unstructured Peer-to-Peer Networks, *Proc. 1st Int'l Workshop Peer-to-Peer Systems* (IPTPS 2002), LNCS 2429, Springer-Verlag, 2002.
6. K. Aberer et al., "Improving Data Access in P2P Systems," *IEEE Internet Computing*, vol. 6, no. 1, Jan./Feb. 2002, pp. 58–67.
7. K. Aberer, "Scalable Data Access in P2P Systems Using Unbalanced Search Trees," to be published in *Proc. Workshop Distributed Data and Structures* (WDAS 2002), Carleton Scientific, 2003.

Self-Organization through Digital Hormones

Wei-Min Shen, *University of Southern California*

Self-organization is ubiquitous in nature, and it appears in almost all branches of science, including physics, chemistry, materials sciences, biology, life sciences, social organization, and astronomy. The classic explanation for such common phenomena is that self-organization is a spontaneous and leaderless behavior emerging from interactions among massive autonomous entities. Alan Turing's diffusion-reaction model is perhaps the earliest example of this explanation.[1]

Self-reconfiguration is a special type of self-organization wherein global patterns and activities involve not only changes inside component entities but also modifications in configuration and structure among the entities. Technically speaking, the requirements for self-reconfigurable systems include the following steps when a task is given from an external source:

1. Execute the current behavior with the current configuration in the current environment.
2. Detect that the current configuration cannot accomplish the task in the current environment.
3. Select a better (or the best) configuration for the current task and environment.
4. Morph into the selected configuration.
5. Select an appropriate behavior in the new configuration for the task.
6. If the task is not accomplished, return to Step 1 or accept a new task.

It is now time to launch a full-scale investigation in self-reconfiguration. Recent progress in biology, information systems, distributed computing, biomimetic robotics, metamorphic robots, material science, and hardware manufacture have provided a solid foundation for developing a complete self-reconfiguration solution. (See the sidebar for possible real-world self-reconfiguration applications.) Biology, in particular, has accumulated considerable evidence and knowledge for self-reconfiguration in nature. Software engineers have built distributed real-time controllers for very large commercial systems, and many robotics researchers have built prototype self-reconfigurable robots and demonstrated great potential for further investigation.[2–4] Some elements of the self-reconfiguration theory have already been developed and demonstrated in selected cases.

The Digital Hormone Model[5] (DHM) is one such attempt. Biological experiments and mathematical modeling suggest that we can model and control self-reconfiguration through a framework similar to the hormone receptor mechanism in biological cells. Colleagues and I have developed the DHM to control self-reconfigurable robots and swarms of robots, but its implications go beyond this.

Self-reconfiguration in biology

Morphallaxis is a typical example of self-reconfiguration wherein a biological organism can regenerate a part or all of itself from a fragment by self-reorganizing existing cells without cell proliferation. You can observe this tissue reorganization process in many lower animals following severe injury, such as bisection. The process involves reforming cells, moving organs, and redifferentiating tissues. The result is usually a smaller but complete individual, derived entirely from the tissues of part of the original animal.

Scientists believe that this reorganization process is the most efficient way for simple organisms to self-heal and self-regenerate. A remarkable example of morphallaxis is an invertebrate freshwater animal called a hydra.[6] If you cut a hydra in half, the head end reconstitutes a new foot, while the basal portion regenerates a new hydranth with a mouth and tentacles. Even if you mince a hydra and scramble the pieces, the fragments grow together and reorganize themselves into a complete whole. The

Self-Reconfiguration and Real-Life Applications

In engineering, self-reconfiguration provides a new and critical capability for many real-world applications. Self-reconfiguration will let smart materials self-assemble into different structures for best performance. In homeland security, self-reconfiguration will let unmanned ground or air vehicles restructure and repair their organization in unexpected situations. In information systems, software agents could adjust their relationships to gather and deliver critical information in a timely fashion. In search-and-rescue or inspect-repair applications, self-reconfigurable robots could maneuver in tight spaces that are hard to reach for humans or conventional robots.

A self-reconfigurable robot could become a ball to roll down a slope, slither between stones as a "snake" to locate a person or artifact, morph smoothly into a "crab" and climb over rubble, and then transform a leg into a gripper to grasp and carry objects. Underwater, a self-reconfigurable robot might become an "eel" to swim in open water, change into an "octopus" to grasp objects, and then spread itself into many small but agile units to monitor large areas.

Self-reconfiguration could help teach engineering students new principles for designing novel fault-tolerant, distributed, and multifunctional systems. In science, principles of self-reconfiguration might help us deepen our understanding of how self-organizing natural systems evolve and adapt to their environments.

hydra might be this indestructible because even the intact animal is constantly regenerating itself. Just below its mouth is a growth zone from which cells migrate to the tentacles and to the foot, where they eventually die. So, the hydra is in a ceaseless state of turnover, with cell loss at the foot and the tentacle tips balanced by the production of new cells in the growth zone. X-raying a hydra, however, inhibits the proliferation of new cells, and it gradually shrinks and eventually dies owing to the inexorable demise of cells and the inability to replace them.[6] Although how organisms initiate and control such a process is still a mystery, a recent discovery in developmental biology[7] suggests a hypothesis that cells secrete hormonal chemicals that regulate cell pattern formation. This discovery was one factor that inspired the DHM.

The DHM

Three other factors influenced the creation of the DHM:

- Existing self-organization models, such as Turing's reaction-diffusion model
- Stochastic cellular automata
- Distributed control systems for self-reconfigurable robots

In biological systems, different cells respond to different hormones because they have different receptors designed to bind with particular hormones. The different types of hormones and target cells in vertebrates are so great that virtually every cell either processes or responds to one hormone or another. Hormones provide the common mechanism that lets cells communicate without identifiers and addresses, and they support a broad spectrum of seemingly diverse biological effects.

The DHM's basic idea is that a self-reconfigurable system is a network of autonomous entities or agents that can dynamically change their physical or logical links. Through the network links, entities use hormone-like messages to communicate, collaborate, and accomplish global behaviors. The hormone-like messages are similar but not identical to content-based messages. They do not have addresses but propagate through the links in the organization. All organizational entities run the same decision-making protocol, but they will react to hormones according to their local topology (where they are in the current configuration) and state information. So, a single hormone might cause different robots or robotic modules in the network to perform different actions. Hormone propagation is different from message broadcasting. There is no guarantee that all entities in the network will receive the same copy of the original message, because a hormone might be modified during its propagation. Hormones also differ from *pheromones* because they do not leave residues in the external environment.

Mathematically speaking, the DHM has three components: a dynamic network specification, a probabilistic function for individual robot behavior, and a set of equations for hormone reaction, diffusion, and dissipation. We specified the first component as a network of autonomous nodes. A unique property of such networks is that each node has a set of reconfigurable *connectors* that can dynamically change the connections with other nodes. The connectors can be physical, as in a reconfigurable robot, or logical, as in an agent-human organization. The interaction between the system, the task, and the current environment triggers the self-reconfiguration process, which produces a new configuration along with a better solution for the task. This is a dynamic and distributed learning process that must continuously propose and execute new configurations and solutions for new tasks and new environments. Unlike classical models, nodes do not have unique IDs. The number of nodes and links in the network is unknown, and there is no global broadcast. A node can communicate only with its current neighbors through its current links. Through local communication, nodes can either generate or propagate hormones. By default, a node will propagate a generated hormone to all its current neighbors. When receiving a hormone, a node will propagate the hormone to all its neighbors except the one that sent the hormone The second component is a specification of individual node behavior, which is similar to *receptors* in biological cells. A network node can select its actions on the basis of a probability function conditioned on four local factors: the local topological information (how it connects to the neighbors), the local sensor information, the local state variables, and the received hormones. This function is local and homogenous for all nodes but can greatly influence the network's global behaviors and predict and analyze the global network performance. This function differentiates the DHM from standard diffusion-reaction models for self-organization, and it can determine whether the system can form any global patterns. You can program these functions at the outset, or the nodes can dynamically change them through a learning technique just as hormones might influence a cell's behaviors by activating or inhibiting different receptors.

The third component is the specification for hormone reaction, diffusion, and dissipation, similar to but different from those specified by Turing[1] and, later, Andrew Witkin and Michael Kass.[8] Each hormone's concentration is a function of position and time that we specify as a set of differential equations. These equations' parameters control the diffusion rate and the manner of reaction among hormones. The differences between the DHM and early diffusion-reaction models are that hormone propagation can be realized through wireless communications, thus achieving "remote" influences that are not limited by geographic distances.

The DHM's execution is simple. All nodes in a self-reconfigurable system asynchronously execute the basic control loop as follows:

1. Select actions based on behavior functions.
2. Execute the selected actions.
3. Perform hormone generations and propagations.
4. Simulate hormone diffusion, reaction, and dissipation.
5. Return to Step 1.

We have successfully applied the DHM to several types of self-reconfiguration systems, including the simulation of embryo skin cells in forming feather buds,[9] the distributed control of robot swarms,[10] and the adaptive communication and control of self-reconfigurable robots.[5]

Under the control of the DHM, the CONRO robot[5] has demonstrated several unique features for self-reconfigurable systems, including online bifurcation, unification, and behavior shifting. No fixed "brain" module exists in CONRO, and every module behaves properly according to its relative position in the current configuration. For example, we can bifurcate a moving CONRO snake robot into pieces, yet each individual piece will "elect" a new head and continue to behave as an independent snake. Multiple snakes can be concatenated (for unification) while running and become a single, coherent snake. For online behavior shifting, we can disconnect a snake's tail-spine module and reconnect it to the side of the body while the system is running, and its behavior will automatically change to that of a leg (the reverse process is also true). For fault tolerance, if a multilegged robot loses some legs, it can still walk on its remaining legs without changing the control program.

In addition, CONRO also demonstrated self-reconfiguration from a snake to a two-legged creature that has a locomotion gait similar to the two-arm butterfly stroke in swimming. This is probably the first time any robot has performed such a metamorphic action in a physical system using a totally distributed control method. In this reconfiguration sequence, the tail module first docks to a body module's left connector to form a loop. Then the middle two modules in the loop disconnect so that the entire configuration becomes a two-legged creature with a body. After that, the new configuration automatically switches on the butterfly behavior for locomotion. You can find video that demonstrates what self-reconfigurable systems can accomplish at www.isi.edu/robots.

Future research in self-reconfigurable system must solve the six challenging problems listed at the beginning of the article in a coherent and integrated way. The solutions might call for new approaches for distributed control, sensor fusion, agent coordination, and understanding the interaction between configurations, tasks, and environments.

Acknowledgments

I thank Cheng-Ming Chuong and Peter Will for their valuable input and discussions during the DHM's development. This work is partially supported by research grants from the Airforce Office of Scientific Research.

References

1. A.M. Turing, "The Chemical Basis of Morphogenesis," *Philosophical Trans. of the Royal Society of London*, Series B, Biological Sciences, vol. 237, no. 641, 14 Aug. 1952, pp. 37–72.

2. M. Yim, Y. Zhang, and D. Duff, "Modular Robots," *IEEE Spectrum*, vol. 39, no. 2, Feb. 2002, pp.30–34.

3. D. Rus et al., "Self-Reconfiguring Robots," *Comm. ACM*., vol. 45, no. 3, Mar. 2002, pp. 39–45.

4. W.-M. Shen and M. Yim, eds., Special Issue on self-reconfigurable modular robots, *IEEE/ASME Trans. Mechatronics*, vol. 7, no. 4, Dec. 2002.

5. W.-M. Shen, B. Salemi, and P. Will, "Hormone-Inspired Adaptive Communication and Distributed Control for CONRO Self-Reconfigurable Robots," *IEEE Trans. Robotics and Automation*, vol. 18, no. 5, Oct. 2002, pp. 700–712.

6. "Regeneration," *Encyclopædia Britannica Online*, http://search.eb.com/eb/article?eu=119855.

7. M. Yu et al., "The Morphogenesis of Feathers," *Nature*, vol. 420, no. 21, Nov. 2002, pp. 308–312.

8. A. Witkin and M. Kass, "Reaction-Diffusion Textures," *Computer Graphics*, vol. 25, no. 3, 1991.

9. W.-M. Shen, C.-M. Chuong, and P. Will, "Digital Hormone Models for Self-Organization," *Proc. 8th Int'l Conf. Simulation and Synthesis of Living Systems* (ALIFE VIII), MIT Press, 2002.

10. W.-M. Shen, P. Will, and C.-M. Chuong, "Hormone-Inspired Self-Organization and Distributed Control for Massive Robot Swarms," submitted to *Autonomous Robots*, 2002; www.isi.edu/robots.

Physical Connectivity of Self-Organized Ad Hoc Wireless Networks

Olivier Dousse and Patrick Thiran, *Swiss Federal Institute of Technology, Lausanne*

In wireless cellular networks (such as Group Spéciale Mobile or Universal Mobile Telecommunications System), nodes can connect to each other through a dense network of antennas (base stations) linked by a wired network. In wireless ad hoc networks (such as Bluetooth), nodes connect to each other without using fixed base stations. If nodes lie in a limited geographical area, destinations can be reachable within a single hop from the source. The problem becomes much more complex when the ad hoc network comprises several nodes scattered over a wide area, because communications then require multihop connections.

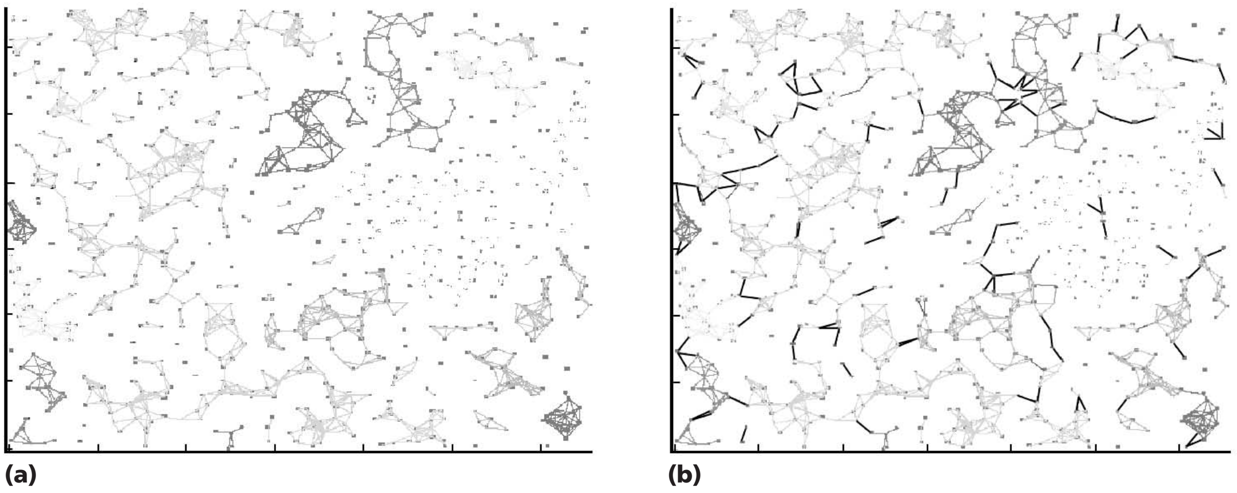


Figure 2. The Poisson Boolean model: (a) the subcritical phase (*R* is slightly less than R_c); (b) the supercritical phase (*R* is slightly larger than R_c).

A cellular network's infrastructure is owned by an operator who can allocate resources and control network performance. In contrast, self-organizing ad hoc networks have no infrastructure and rely entirely on distributed, local decisions for every operational aspect: self-configured deployment, topology discovery, routing, scheduling, medium access control, and cooperation mechanisms for relaying other users' connections. Even the most basic network property, its connectivity, becomes a nontrivial issue in multihop ad hoc networks, as you will see.

Network connectivity

Suppose that all nodes randomly placed on a plane transmit at maximum power. (Forget, temporarily, interferences between simultaneous communications.) In this case, a simple criterion for deciding whether two nodes can connect in a single hop is that their distance be less than some prescribed maximal range *R*. What is the resulting network's connectivity?

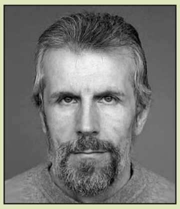
One approach is to assume that the number of nodes *N* is very large (theoretically infinite), but that they all lie within a finite geographical area; in other words, the geographical density of nodes per surface area λ tends toward infinity. Piyush Gupta and Panganamala Kumar have proven that, if *R* decreases at a rate slower than $\sqrt{(\log N / N)}$, the network is then almost fully connected.[1]

A second approach is to allow a finite geographical density of nodes per surface area λ but assume that nodes are dispersed in the entire plane. Assume, moreover, that this process is Poisson. Because the number of nodes is unbounded, some of them will be disconnected. The question is then, are long-distance communications still possible, or are they all limited to some local range? This question is related to the *continuum percolation theory* for the Poisson Boolean Model.[2] The percolation probability is the probability that an arbitrary node will belong to a cluster of infinite size (and thus that an arbitrary node can be connected to another node arbitrarily far away by a multihop path).

The percolation theory's main result is that a finite, positive value λ_c of λ (or equivalently, a finite, positive value R_c of *R*) exists under which the percolation probability is zero (the subcritical phase) and above which it is nonzero (the supercritical phase). In the subcritical phase, all clusters are almost surely finite, and their size is a random variable whose distribution has an exponentially decreasing tail. In the supercritical phase, the percolation probability usually increases rapidly with λ. Figure 2 illustrates this phase transition phenomenon. By comparing the two graphs, you will notice that a slight increase of the radius *R* has created only a few edges (black lines in Figure 2b), but that these few edges connect most of the colored clusters into one large cluster (theoretically infinite). This probability's exact value is still an open question. Ronald Meester and Rahul Roy, and Edgar Gilbert obtained some bounds on the critical intensity λ_c, below which the percolation probability is zero.[2,3]

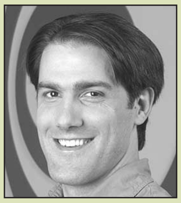
Why is knowing that the network is a supercritical phase so interesting? After all, this property "only" means that a node cluster of infinite size with positive probability exists, which is not a satisfactory guarantee for users of large-scale ad hoc networks. If the percolation probability were a smoothly increasing function of λ and *R*, this property would probably indeed not have much impact. Its importance is the sudden phase transition phenomenon that occurs in the Boolean model and also occurs in other models of random graphs.[4]

To illustrate this claim, consider a hybrid (or multihop cellular) network, which is an intermediate solution between fully cellular networks and pure ad hoc networks. In such a network, base stations are linked together by a wired network, but only a few exist, so that most nodes need many hops to connect to a base station. What is the benefit, in terms of connectivity, of having such a sparse network of base stations regularly placed in the network? If nodes are scattered in two dimensions, and if the node density (or radius) is above the percolation threshold, there is virtually no benefit. Indeed, there is an infinite cluster, which will almost surely connect to at least one base station. So, the only nodes that will connect to the infinite cluster exclusively owing to the presence of base stations are the few isolated nodes that lie within a distance *R* from a base station.



Francis Heylighen is a research professor and codirector of the interdisciplinary center Leo Apostel at the Free University of Brussels (VUB). His research focuses on the self-organization and evolution of complex systems, which he addresses from a cybernetic perspective. He received his PhD in mathematical physics from VUB. He is editor of the Principia Cybernetica Project, chairman of the Global Brain Group, and member of the editorial boards of the *Journal of Memetics*, *Journal of Collective Intelligence*, *Informatica*, and *Entropy*. Contact him at the Center Leo Apostel, Vrije Universiteit Brussel, Krijgskundestraat 33, B-1160 Brussels, Belgium; fheyligh@vub.ac.be.

Carlos Gershenson is a PhD student at the Center Leo Apostel at the Free University of Brussels and a contributing editor for the weekly electronic distribution *Complexity Digest*. His interests include self-organizing and complex systems, distributed cognition, theories of evolution, and philosophy of science. He received his MSc in evolutionary and adaptive systems at the University of Sussex. Contact him at the Center Leo Apostel, Vrije Universiteit Brussel, Krijgskundestraat 33, B-1160 Brussels, Belgium; cgershen@vub.ac.be.



Gary William Flake is the chief science officer of Overture Services. His research interests include machine learning, Web data mining, efficient algorithms, and complex systems. He received his PhD in computer science from the University of Maryland. He is a member of the ACM, the IEEE, and DIMACS. Contact him at gary.flake@overture.com.

David M. Pennock is a senior research scientist at Overture Services. His research interests include electronic commerce, Web modeling, and artificial intelligence. He received his PhD in computer science from the University of Michigan. He is a member of the ACM, the IEEE, the AAAI, DIMACS, INFORMS, and AAAS. Contact him at david.pennock@overture.com.

Daniel C. Fain is a the director of R&D Algorithms at Overture Services. His research interests include information extraction and retrieval, in particular, topic identification, relevance evaluation, and applications of machine learning. He received his PhD in computation and neural systems from the California Institute of Technology. Contact him at dan.fain@overture.com.

David De Roure is a professor of computer science at the University of Southampton, where he is head of Grid and Pervasive Computing in the Department of Electronics and Computer Science and codirector of the Southampton e-Science Center. His interests include large-scale distributed systems and the relationship between semantic, pervasive, and grid computing. He received his PhD in distributed systems from the University of Southampton. Contact him at the Dept. Electronics and Computer Science, Univ. of Southampton, Southampton, SO17 1BJ, UK; dder@ecs.soton.ac.uk.

Karl Aberer is a professor for distributed information systems at the Swiss Federal Institute of Technology, Lausanne. His research interests focus on self-organization principles in information management, addressing issues of P2P data management, semantic interoperability, trust management, and Web services. Contact him at LSIR-Distributed Information Systems Laboratory, EPFL-I&C, 1015 Lausanne, Switzerland; karl.aberer@epfl.ch; lsirwww.epfl.ch.

Wei-Min Shen is the director of the Polymorphic Robotics Laboratory at the Information Sciences Institute and a research assistant professor in computer science at the University of Southern California. His research interests include artificial intelligence, robotics, and life science. He received his PhD from Carnegie Mellon University under Herbert A. Simon on the subject of autonomous learning from the environment based on actions and percepts. He is a member of the IEEE, the ACM, and AAAI. Contact him at the Information Sciences Institute, Univ. Southern California, 4676 Admiralty Way, Marina del Rey, CA 90292; shen@isi.edu; www.isi.edu/~shen.

Olivier Dousse is a PhD student from the Institute for Communication Systems at the Swiss Federal Institute of Technology, Lausanne (EPFL). His research interests include connectivity and capacity of multihop ad hoc and cellular networks. He received his MS in physics from EPFL. He is a member of the research group IP1 of the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS). Contact him at LCA-ISC-I&C, Ecole Polytechnique Federale de Lausanne, CH-1015 Lausanne, Switzerland; Olivier.dousse@epfl.ch.

Patrick Thiran is a professor of communication systems at the Swiss Federal Institute of Technology, Lausanne (EPFL). His research interests include communication networks (in particular ad hoc networks), performance analysis, dynamical systems, and stochastic models. He received his PhD in electrical engineering from EPFL. Contact him at LCA-ISC-I&C, Ecole Polytechnique Federale de Lausanne, CH-1015 Lausanne, Switzerland; patrick.thiran@epfl.ch.

Because few such nodes and few base stations exist, the benefit remains marginal.

On the other hand, if the node density is less than the percolation threshold but still large enough, the presence of a few base stations does help, although many nodes still remain unconnected. The benefit is much more dramatic when the nodes' spatial distribution is one-dimensional. In this case, the network is in the subcritical phase for all finite values of λ and is thus made of finite clusters whose size distribution has an exponential tail. If λR is large, then base stations can be far away from each other yet make long-distance communications possible. These features hold for node distributions besides Poisson.[5] However, these results apply only to connectivity properties and not to capacity, which can increase owing to the presence of wired base stations.

Percolation might, however, impact capacity too. Figure 2b shows that—with a barely supercritical network—the black links connect many colored clusters of nodes but still form bottlenecks. This affects capacity and routing in the network. Only for very large spatial densities can these network "hot spots" disappear.

Accounting for interferences

Wireless channels' physical features will require more sophisticated models than the Boolean model just described.

One feature is the anisotropy of radiation in real networks, which replaces the Boolean model's circle of radius R with an irregular, rotationally nonsymmetric contact surface. Although many links will disappear in comparison to the Boolean model, the resulting graph will have a few long edges essential for its long-range connectivity, such as in small-world graphs.[6] The irregularity is therefore a positive factor, letting the percolation occur at a lower density than in a Boolean model.[7]

A second feature is the interferences between simultaneous channels. One model assumes that a link exists between two nodes if and only if the signal-to-noise ratio at the receiver (noise being the sum of the background noise and the interference contributions from neighboring nodes) is larger than some prescribed value.[8] The problem is more complex because now direct connections between two particular nodes depend on other nodes' positions. The node degree is bounded, contrary to the Boolean model. Moreover, two parameters (the orthogonality coefficient of the CDMA [code division multiple access] system coping with interferences and emitting power) now exist, instead of one (power). The conditions for a supercritical phase are more stringent than in the Boolean model, but percolation still occurs despite the interferences for a nonzero value of the code's coefficient of orthogonality, provided it is small enough. As such, a CDMA code might be difficult to use in self-organized ad hoc networks. An alternative is to adopt a simple TDMA (time division multiple access) system, where each node randomly picks one time slot every n time slots to transmit, while all nodes listen at all times. We found that such a system led to similar connectivity as the original CDMA scheme with an orthogonality factor n times smaller. The qualitative explanation is that the increased irregularity in the resulting graphs obtained in the TDMA case, with many short edges and a few long ones, compensates for a large orthogonality factor.[8]

The idea of applying percolation theory to ad hoc networks dates back to Gilbert's pioneering work[3] and transcends the basic property of potential long-distance communications in a purely ad hoc network. As we outline in this article, percolation theory is a key to understanding many different performance aspects of large-scale ad hoc and sensor networks. Interest is growing in percolation analysis based on extensions of the Boolean model that capture additional features of ad hoc networks' physical layers. Some results can be initially surprising (for example, irregularity is good for connectivity). Percolation will also be instrumental in analyzing the higher layers of self-organized networks. For instance, you might view setting a time-out value in a flooding algorithm for P2P networks as a percolation problem because the search algorithm must reach nodes arbitrarily far away from the request source. Percolation has been applied to devise probabilistic broadcast algorithms in mobile ad hoc networks.[9] ■

Acknowledgments

This work was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems, which the Swiss National Science Foundation supports under grant number 5005-67322.

References

1. P. Gupta and P.R. Kumar, "Critical Power for Asymptotic Connectivity in Wireless Networks," *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming*, W.M. McEneany, G. Yin, and Q. Zhang, eds., Birkhäuser, 1998, pp. 547–566.
2. R. Meester and R. Roy, *Continuum Percolation*, Cambridge Univ. Press, 1996.
3. E.N. Gilbert, "Random Plane Networks," *SIAM J.*, vol. 9, Dec. 1961, pp. 533–543.
4. B. Bollobas, *Random Graphs*, Academic Press, 1985.
5. O. Dousse, P. Thiran, and M. Hasler, "Connectivity in Ad Hoc and Hybrid Networks," *Proc. INFOCOM 2002*, vol. 2, IEEE Press, 2002, pp. 1079–1088.
6. D. Watts and S. Strogatz, "Collective Dynamics of Small World Networks," *Nature*, vol. 363, June 1998, pp. 202–204.
7. L. Booth et al., "Ad Hoc Networks with Noisy Links," to be published in *Proc. IEEE Information Theory Symposium* (ISIT 03), 2003.
8. O. Dousse, F. Baccelli, and P. Thiran, "Impact of Interferences on Connectivity in Ad Hoc Networks," *Proc. INFOCOM 2003,* IEEE Press, 2003.
9. Y. Sasson, D. Cavin, and A. Schiper, "Probabilistic Broadcast for Flooding in Wireless Mobile Ad Hoc Networks," *Proc. 2003 IEEE Wireless Comm. and Networking Conf.* (WCNC), IEEE Press, 2003.