Distributed Computer Systems

David Hales, University of Bologna, Italy (hales@cs.unibo.it)

Abstract

Ideas derived from social simulation models can directly inform the design of distributed computer systems. This is particularly the case when systems are "open", in the sense of having no centralised control, where traditional design approaches struggle. In this chapter we indicate the key features of social simulation work that are valuable for distributed systems design. We also discuss the differences between social and biological models in this respect. We give examples of socially inspired systems from the currently active area of peer-to-peer systems and finally we discuss open areas for future research in the area.

Introduction

Massive and open distributed computer systems provide a major application area for ideas and techniques developed within social simulation and complex systems modelling. In the early years of the 21st century there has been an explosion in global networking infrastructure in the form of wired and wireless broadband connections to the internet encompassing both traditional general purpose computer systems, mobile devices and specialist appliances and services. The challenge is to utilise such diverse infrastructure to provide novel services that satisfy user needs reliably. Traditional methods of software design and testing are not always applicable to this challenge. Why is this? And what can social simulation and complexity perspectives bring to addressing the challenge? This chapter answers these questions by providing a general overview of some of the major benefits of approaching design from socially inspired perspective in addition to examples of applications in the area of peer-to-peer (P2P) systems and protocols. Finally we will speculate on possible future directions in the area.

This chapter is not an exhaustive survey of the area, for example, we have not discussed the application of social network analysis techniques to web graphs and social networks that are constructed within, or facilitated by, distributed software systems (Staab et al 2005). Both these are active areas. Further, we have not discussed the active research area based on randomised "Gossiping" approaches, where information is diffused over networks through randomised copying of information between adjacent nodes (Wang et al 2007).

What's wrong with traditional design approaches?

Traditional design approaches to systems and software often assume that systems are essentially "closed" – meaning they are under the control of some administrative authority that can control access, authenticate users and manage system resources such as issuing software components and updates. Consider the simplest situation in which we have a single computer system that is not connected to a network that is required to solve some task. Here, design follows a traditional process of analysis of requirements, specification of requirements then design and iterative refinement, until the system meets the specified requirements. User requirements are assumed to be discoverable and translatable into specifications at a level of detail that can inform a design. The designer has, generally, freedom to dictate the how the system should achieve the required tasks via the coordination of various software components. This coordination generally follows an elaborate sequencing of events where the output from one component becomes the input to others. The design task is to get that sequence right.

In “open systems” it is assumed that there are multiple authorities. This means that components that comprise the system cannot be assumed to be under central control. An extreme example of this might be an open peer-to-peer (P2P) system in which each node in the system executes on a user machine under the control of a different user. Popular file-sharing systems operate in this way, allowing each user to run any variant of client software they choose. These kinds systems function because the client software implements publicly available peer communication protocols allowing the nodes to interconnect and provide functionality. However, due to the open nature of such systems, it is not possible for the designer, *a prior*, to control the sequence of processing in each node. Hence the designer needs to consider what kinds of protocols will produce *acceptable* system level behaviours under *plausible* assumptions of node behaviour. This requires a radically different design perspective in which systems need to be designed as self-repairing and self-organising systems in which behaviour emerges bottom-up rather than being centrally controlled.

One term for this approach, based on self-organisation and emergence, is so-called Self-Star (or Self-*) systems (Babaoglu et al 2005). The term is a broad expression that aims to capture all kinds self-organising computer systems that continue to function acceptably under realistic conditions.

But what kinds of design approach can be employed? Currently there is no accepted general theory of self-organisation or emergence – rather there are some interesting models at different levels of abstraction that capture certain phenomena. Many such models have been produced within the biological sciences to explain complex self-organising biological phenomena. Biological systems, particularly co-evolving systems, appear to evidence many of the desirable properties required by self-* computer systems. Hence, several proposed self-* techniques have drawn on biological inspiration (Babaoglu et al 2006).

Socially versus Biological Inspiration

It is useful to ask in what way social organisation differs from the biological level. In this section we briefly consider this question with regard to desirable properties for information systems. An important aspect of human social systems (HSS) is their ability (like biological systems) to both preserve structures - with organisations and institutions persisting over time - and adapt to changing environments and needs. The evolution of HSS is not based on DNA, but rather on a complex interplay between behaviour, learning, and individual goals. Here we present some distinguishing aspects of HSS.

Rapid Change

A feature of HSS is the speed at which re-organisations can occur. Revolutions in social organisation can take place within the lifetime of a single individual. Hence although HSS often show stable patterns over long periods, rapid change is possible. The ability to respond rapidly would appear to be a desirable property in rapidly changing information system environments; however, for engineering purposes, one must ensure that such fast changes (unlike revolutions!) can be both predicted and controlled.

Non-Darwinian Evolution

HSS do not evolve in a Darwinian fashion. Cultural and social evolution is not mediated by random mutations and selection of some base replicator over vast time periods, but rather follows a kind of collective learning process. That is, the information storage media supporting the change by learning—and hence (as noted above), both the mechanisms for change and their time scale—are very different from those of Darwinian evolution. Individuals within HSS can learn both directly from their own parents (vertical transmission), from other members of the older generation (diagonal transmission), or from their peers (horizontal transmission). Hence, new cultural traits (behaviours, beliefs, skills) can be propagated quickly through a HSS. This can be contrasted with simple Darwinian transmission in which, typically, only vertical transmission of genetic information is possible. Although it is possible to characterise certain processes of cultural evolution based on the fitness of cultural replicators (Boyd and Richerson 1985) or memes (Dawkins 1976) it is important to realise such replicators are not physical - like DNA - but part of a socio-cognitive process - passing through human minds - and may follow many kinds of selective process (Lumsden and Wilson 1981).

Stable Under Internal Conflict

HSS exist because individuals need others to achieve their aims and goals. Production in all HSS is collective, involving some specialisation of roles. In large modern and post-modern HSS roles are highly specialised, requiring large and complex co-ordination and redistribution methods. However, although HSS may sometime appear well integrated, they also embody inherent conflicts and tensions between individual and group goals. What may be in the interests of one individual or group may be in direct opposition to another. Hence, HSS embody and mediate conflict on many levels.

This aspect is highly relevant to distributed and open information systems. A major shift from the closed monolithic design approach is the need to deal with and tolerate inevitable conflicts between sub components of a system. For example, different users may have different goals that directly conflict. Some components may want to destroy the entire system. In open systems this behaviour cannot be eliminated and hence need in some way to be tolerated.

Only Partial Views and Controversy

Although HSS are composed of goal directed intelligent agents, there is little evidence that individuals or groups within them have a full view or understanding of the HSS. Each individual tends to have a partial view often resulting from specialisation within, and complexity of, the system. Such partial views, often dictated by immediate material goals, may have a normative (how things “should” be) character rather than a more scientific descriptive one (how things “are”). Consequently, the ideas that circulate within HSS concerning the HSS itself tend to take on an “ideological” form. Given this, social theories are rarely as consensual as those from biological sciences. Thus, social theories include a high degree of controversy, and they lack the generally accepted foundational structure found in our understanding of biology. However, from an information systems perspective, such controversy is not problematic: we do not care if a given social theory is true for HSS or not; we only care if the ideas and mechanisms in the theory can be usefully applied in information systems. This last point, of course, also holds for controversial theories from biology as well (e.g. Lamarkain evolution).

Trust and Socially Beneficial Norms

In trying to understand the stability of socially functional behaviour, much work within the social sciences has focused on the formation and fixation of “norms” of behaviour. Many researchers working with Multi-Agent Systems (MAS) have attempted to create artificial versions of norms to regulate MAS behaviours - although much of these have not been based on theories from HSS (although see Conte and Paolucci 2002). Certainly the establishment and stability of beneficial norms (such as not cheating ones’ neighbour) is a desirable property visible in all stable HSS (Hales 2002). This point (the existence and power of norms) is of course closely related to the previous point, which notes that norms can influence understanding and perception.

It is widely agreed that, in HSS, many observed behaviours do not follow the same pattern as would be expected from simple Darwinian evolution or individual “rational” behaviour - in the sense of maximising the chance of achieving individual goals. Behaviour is often more socially beneficial and co-operative or altruistic, generally directed toward the good of the group or organisation within which the individual is embedded. (We note the widespread appearance of altruistic behavior among many species of social mammals—such that, once again, we speak here of a difference in degree between HSS and other social animals.) Many theories and mechanisms have been proposed by social scientists for this kind of behaviour (Axelrod 1984), with many of these formalised as computer algorithms; furthermore, several of these have already been translated for use in information systems (Cohen 2003; Hales and Edmonds 2005].

Generalised Exchange and Economics

Almost all HSS evidence some kind of generalised exchange mechanisms (GEM) - i.e. some kind of money. The emergence of GEM allows for co-ordination through trade and markets. That, is, collective co-ordination can occur where individual entities (individuals or firms) behave to achieve their own goals. It is an open (and perhaps overly simplified) question whether certain norms are required to support GEM or, rather, most norms are created via economic behaviour within GEM (Edmonds and Hales 2005). Certainly, the formation and maintenance of GEM would be an essential feature of any self-organised economic behaviour within information systems - currently many information systems work by assuming an existing GEM *a priori* - i.e. they are parasitic on HSS supplying the trust and norms required. Such systems require trusted and centralised nodes before they can operate because they do not emerge such nodes in on-going interaction. However, given that GEM exist, a huge amount of economic theory, including new evolutionary economics and game theory, can be applied to information systems.

What Can Social Simulation Offer the Designer?

Social simulation work has the potential to offer a number of insights that can be applied to aid design of distributed computer systems. Social simulators have had no choice but to start from the assumption of open systems composed of autonomous agents – since most social systems embody these aspects. In addition much social simulation work is concerned with key aspects of self-* systems such as:

- Emergence and Self-organisation: understanding the micro-to-macro and the macro-to-micro link. Phenomena of interest often result from bottom-up processes that create emergent structures that then constrain or guide (top-down) the dynamics of the system.

- Cooperation and Trust: getting disparate components to "hang-together" even with bad guys around. In order to for socially integrated cooperation to emerge it is generally necessary to employ distributed mechanisms to control selfish and free-riding behaviour. One mechanism for this is to use markets (see Chapter **) but there are other methods.

- Evolving robust network structure: Constructing and maintaining functional topologies robustly. Distributed systems often form dynamic networks in which the maintenance of certain topological structures improves system level performance.

- Constructing adaptive / evolutionary heuristics rather than rational action models. Models of both software and user behaviour in distributed systems are based on implicit or explicit models. Traditional approaches in game theory and economics have assumed rational action but these are rarely applicable in distributed systems.

These key aspects have import into two broad areas of system design. Firstly, simulation models that produce desirable properties can be adapted into distributed system protocols that attempt to reproduce those properties. Secondly, models of agent behaviour, other than rational action approaches, can be borrowed as models of user behaviour in order to test existing and new protocols.

Currently, however, it is an open question as to how results obtained from social simulation models can be productively applied to the design of distributed information systems. There is currently no general method whereby desirable results from a social simulation model can be imported into a distributed system. It is certainly not currently the case that techniques and models can be simply "slotted into" distributed systems. Extensive experimentation and modification is required. Hence, in this chapter we give specific examples from P2P systems where such an approach been applied.

What about agent orientated design approaches?

Multi-Agent Systems (MAS) design approaches have been previously been proposed (Wooldridge and Jennings 1995) which attempt to address some the design issues raised by open systems. Those approaches start with a "blank-sheet" design approach rather than looking for biological or social inspiration. The focus therefore has tended to be on logical foundations, proof, agent languages and communication protocols. For example, the BDI agent framework starts from the assumption that agents within a system follow a particular logical architecture based on "folk psychological" cognitive objects – such as beliefs or intentions (Rao and Georgeff 1991). However, such approaches have difficulty scaling to large societies with complex interactions particularly where the effects of emergence and self-organisation are important. A more recent approach within MAS work has been to look towards self-organising approaches using simulation to capture processes of emergence (Brueckner et al 2006). In this work heavy use has been made of biological and socially inspired approaches.

Examples of socially inspired P2P systems

Here we give very brief outlines of some P2P protocols that have been directly inspired by social simulation models. We should state that it is not only P2P systems that have benefited from social inspiration but we have focused on this particular technology because it is currently, at the time of writing, a very active research area and increasingly widely deployed on the internet.

Reciprocity based BitTorrent P2P System

BitTorrent (Cohen 2003) is an open P2P file sharing system that draws directly from the social simulation work of Robert Axelod (1984) on the evolution of cooperation. The protocol is based on a form of the famous Tit-For-Tat (TFT) strategy popularised by Axelrods computer simulation tournaments. Strategies were compared by having agents play the canonical Prisoner's Dilemma (PD) game.

The PD game captures a social dilemma in the form of a minimal game in which two players each select a move from two alternatives (either to Cooperate or Defect) then each player receives a score (or pay-off). If both players Cooperate then both get a Reward payoff (R). If both Defect they are Punished, both obtaining payoff P. If one player selects defect and the other selects cooperate then the defector gets T (the Temptation), the other getting S (the Sucker). When these pay-offs, which are numbers representing some kind of desirable utility (for example, money), obey the following constraints: $T>R>P>S$ and $2R>T+S$ then we say the game represents a Prisoner's Dilemma. When both players cooperate this maximizes the collective good but when one player defects and another cooperates this represents a form of free riding with the defector gaining a higher score (T) at the expense of the co-operator (S).

Axelrod asked researchers to submit computer programs to a “tournament” where they repeatedly played the PD against each other accumulating payoffs. The result of the tournaments was that a simple strategy, TFT, did remarkably well against the majority of other submitted programs - although other strategies can also survive within the complex ecology that occurs when there is a population of competing strategies.

TFT operates in environments where the PD is played repeatedly with the same partners for a number of rounds. The basic strategy is simple: an player starts by cooperating then in subsequent rounds copies the move made in the previous round by its opponent. This means defectors are punished in the future: the strategy relies on future reciprocity. To put it another way, the "shadow" of future interactions motivates cooperative behaviour in the present. In many situations this simple strategy can outperform pure defection.

In the context of BitTorrent the basic mechanism is simple - files are split into small chunks (about 1mb each) and downloaded by peers, initially, from a single hosting source. Peers then effectively "trade" chunks with each other using a TFT-like strategy - i.e. if two peers offer each other a required chunk then this equates to mutual cooperation. However if either does not reciprocate then this is analogous to a defect and the suckered peer will retaliate in future interactions.

The process is actually a little more subtle because each peer is constantly looking at the upload rate / download rate from each connected peer in time - so it does not work just by file chunk but by time unit within each file chunk. While a file is being downloaded between peers, each peer maintains a rolling average of the download rate from each of the peers it is connected to. It then tries to match it's uploading rate accordingly. If a peer determines that another is not downloading fast enough then it may "choke" (stop uploading) to that other. Figure 1 shows a schematic diagram of the way the Bittorrent protocol structures population interactions.

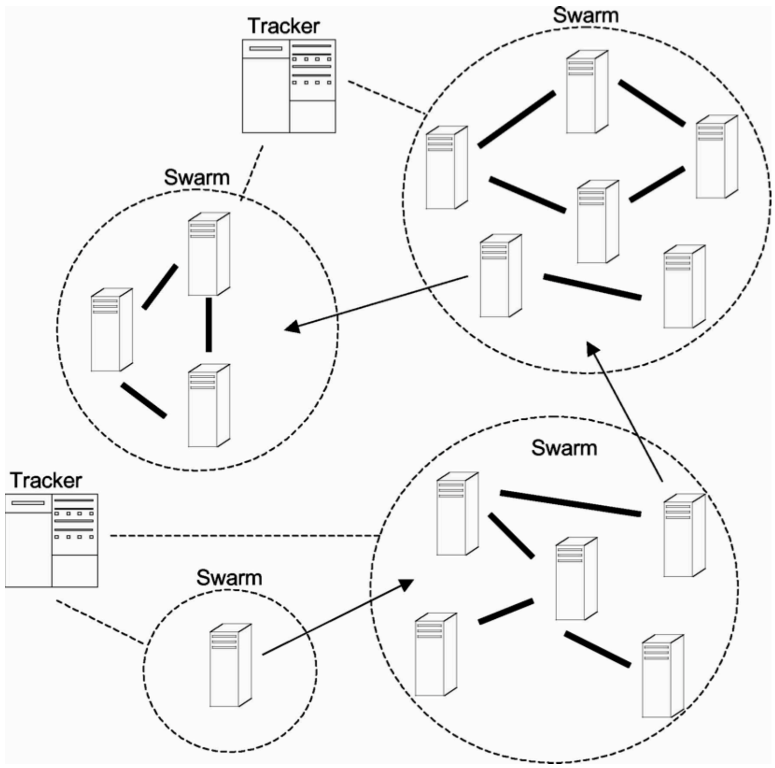


Figure 1: A schematic of a portion of a BitTorrent system. The trackers support swarms of peers each downloading the same file from each other. Thick lines indicate file data. They are constantly in flux due to the application of the TFT-like "choking" protocol. Trackers store references to each peer in each supported swarms. It is not unknown for trackers to support thousands of swarms and for swarms to contain hundreds of peers. Arrows show how peers might, through user intervention, move between swarms. Generally at least one peer in each swarm would be a "seeder" that holds a full copy of the file being shared

Additionally, peers periodically try connecting to new peers randomly by uploading to them – testing for better rates. This means that if a peer does not upload data to other peers (a kind of defecting strategy) then it is punished by other peers in the future (by not sharing file chunks) - hence a TFT-like strategy based on punishment in future interactions is used.

Axelrod used the TFT result to justify sociological hypotheses – such as understanding how fraternisation broke out between enemies across the trenches of WW1. Cohen has applied a modified form of TFT to produce a file sharing system resistant to free riding. However, TFT has certain limitations, it requires future interactions with the same individuals and each has to keep records of the last move made by each opponent. Without fixed identities it is possible for hacked clients to cheat BitTorrent. Although it appears that widespread cheating has not actually spread in the population of clients. It is an open question as to why this might be (but see Hales and Patarin (2006) for a hypothesis).

Group Selection Based P2P Systems

Recent work, drawing on agent-based simulations of cooperative group formation based on "tags" (social labels or cues) and dynamic social networks suggests a mechanism that does not require reciprocal arrangements but can produce cooperation and specialisation between nodes in a P2P (Riolo 2001; Hales and Edmonds 2005). It is based on the idea of cultural group selection and the well-known social psychological phenomena that people tend to favour those believed to be similar to themselves – even when this is based on seemingly arbitrary criteria (e.g. supporting the same football team). Despite the rather complex lineage, like TFT, the mechanism is refreshingly simple. Individuals interact in cliques (subsets of the population). Periodically, if they find another individual who is getting higher utility than themselves they copy them – changing to their clique and adopting their strategy. Also, periodically, individuals form new cliques by joining with a randomly selected other.

Defectors can do well initially, suckering the co-operators in their clique – but ultimately all the co-operators leave the clique for pastures new – leaving the defectors all alone with nobody to free-ride on. Those copying a defector (who does well initially) will also copy their strategy, further reducing the free-riding potential in the clique. So a clique containing any free riders quickly dissolves but those containing only co-operators grow.

Given an open system of autonomous agents all cliques will eventually be invaded by a free rider who will exploit and dissolving the clique. However, so long as other new cooperative cliques are being created then cooperation will persists in the overall population. In the context of social labels or "tags" cliques are defined as those individuals sharing particular labels (e.g. supporting the same football team). In the context of P2P systems the clique is defined as all the other peers each peer is connected to (it's neighbourhood) and movement between cliques follows a process of network "re-wiring".

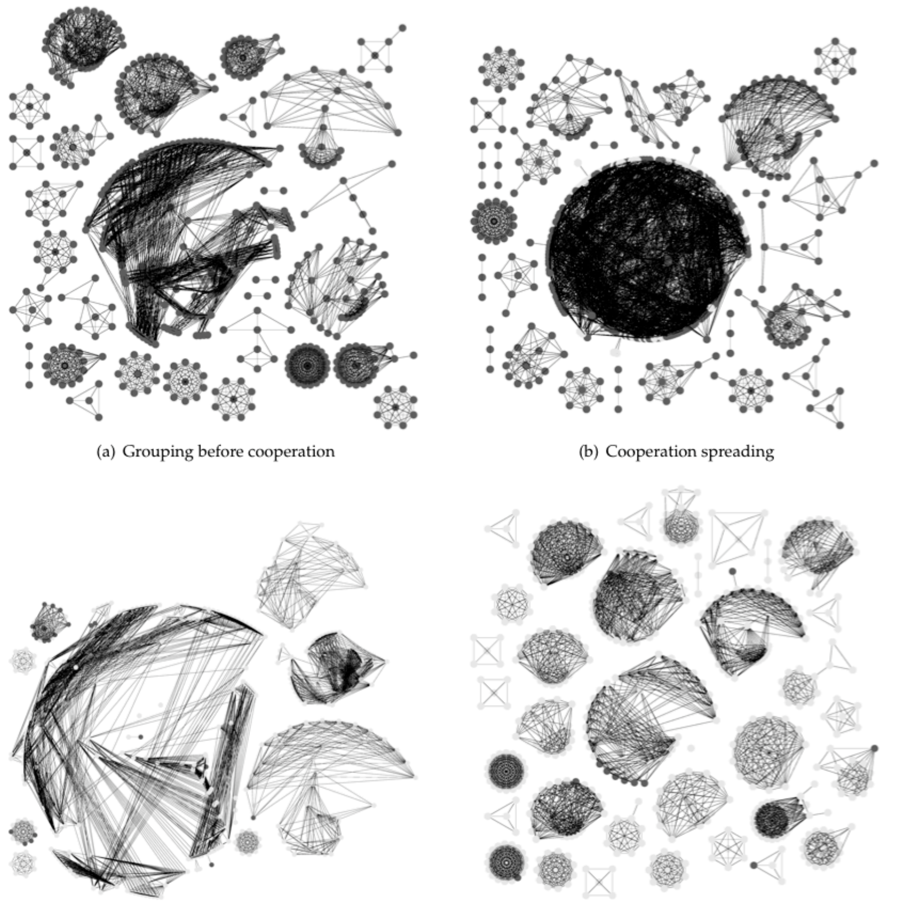


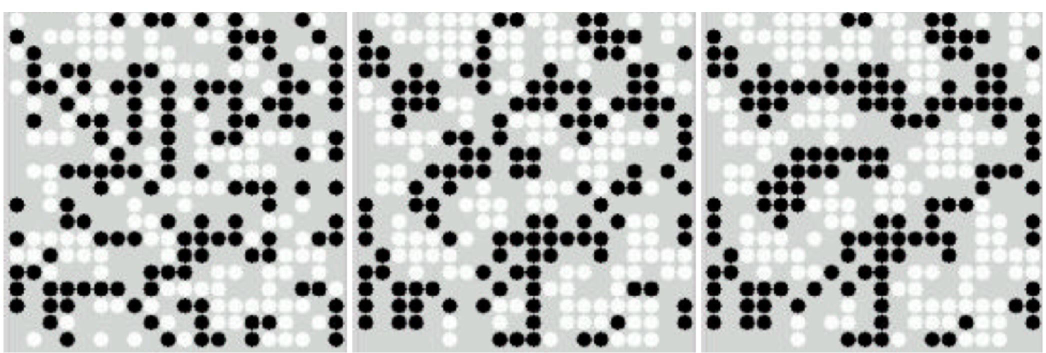
Figure 2: Evolution of a SLAC network with nodes playing the Prisoner 's Dilemma. From an initially random topology composed of all nodes playing the defect strategy (dark shaded nodes), components quickly evolve, still containing all defect nodes (a). Then a large cooperative component emerges in which all nodes cooperate (b). Subsequently the large component begins to break apart as defect nodes invade the large cooperative component and make it less desirable for cooperative nodes (c). Finally an ecology of cooperative components dynamically persists as new components form and old components die (d). Note: the cooperative status of a node is indicated by a light shade.

Through agent-based simulation, the formation and maintenance of high levels of cooperation in the single round PD and in a P2P file-sharing scenario has been demonstrated (Hales and Edmonds 2005). The mechanism appears to be highly scalable with zero scaling cost – i.e. it does not take longer to establish cooperation in bigger populations. Figure 2 shows the evolution of cooperative clusters within a simulated network of peer nodes. A similar approach was presented by Hales and Areteconi (2006) that produced small-world connected cooperative networks rather an disconnected components.

In addition to maintaining cooperation between nodes in P2P, the same group selection approach has been applied to other areas such as the coordination of robots in a simulated warehouse scenario and to support specialisation between nodes in a P2P job sharing system (Hales and Edmonds 2003; Hales 2006).

Segregation Based P2P Systems

The model of segregation proposed by Thomas Schelling is well known within social simulation (Schelling 1969, 1971). The model demonstrates how a macro-structure of segregated clusters or regions robustly emerges from simple local behaviour rules. Schelling's original model consists of agents on a two dimensional grid. Each grid location can hold a single agent or may be empty. Each agent maintains a periodically updated satisfaction function. Agents take one of two colours that are fixed. An agent is said to be satisfied if at least some proportion of adjacent agents on the grid have the same colour, otherwise the agent is said to be not satisfied. Unsatisfied agents move randomly in the grid to a free location. The main finding of the model is that even if the satisfaction proportion is very low, this still leads to high levels of segregation by colour – i.e. large clusters of agents emerge with the same colour. Figure 4 shows an example run of the model in which clusters of similar colours emerge over time.



(a) (b) (c)

Figure 4: Example run of the Schelling segregation model. From an initially random distribution of agents (a) clusters emerge over successive iterations (b) and (c). In this example the satisfaction proportion is 0.5, meaning that agents are unsatisfied if less than 50% of neighbours in the grid are a different colour (taken from Edmonds and Hales 2005).

The results of the segregation model are robust even when nodes randomly leave and enter the system – the clusters are maintained. Also agents in the segregation model only require local information in order to decide on their actions. These properties are highly desirable for producing distributed information systems and therefore it is not surprising that designs based on the model have been proposed.

Sing and Haahr (2006) propose a general framework for applying a modified form Schelling's model to topology adoption in a P2P network. They show how a simple protocol can be derived that maintains a "hub-based" backbone topology within unstructured networks. Hubs are nodes in a network that maintain many links to other nodes. By maintaining certain proportions of these within networks it is possible to improve system performance for certain tasks. For many tasks linking the hubs to form a backbone within the network can further increase performance. For example, the Gnutella[1] file-sharing network maintains high-bandwidth hubs called "super-peers" that speed file queries and data transfer between nodes. Figure 5 shows an example of small network maintaining a hub backbone.

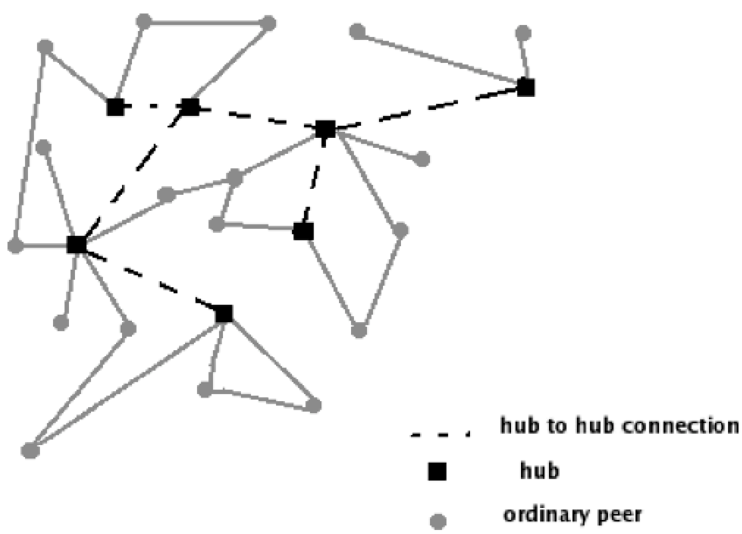


Figure 5: An example of a hub based peer-to-peer topology. Hubs are linked and perform specialist functions that improve system performance (taken from Singh and Haahr 2006).

In the P2P model nodes represent agents and neighbours are represented by explicit lists of neighbour links (a so-called View in P2P terminology). Nodes adapt their links based on a satisfaction function. Hub nodes (in the minority) are only satisfied if they have at least some number of other hubs in their view. Normal nodes are only satisfied if they have at least one hub node in their view. Hence different node types use different satisfaction functions and exist in a network rather than lattice. It is demonstrated via simulation that the network maintains a stable and connected topology supporting a hub backbone under a number of conditions, include dynamic networks in which nodes enter and leave the network over time.

The approach presented by Sing and Haahr is given as general approach (a template design pattern) that may be specialised to other P2P application areas rather than just self-organising hub topologies. For example, they apply the same pattern to decrease

[1] www.gnutella.com

bandwidth bottlenecks and increase system performance of a P2P by clustering similar nodes based on bandwidth capacity (Singh and Haahr 2004).

Possible Future Research

In the following sections we give a brief outline of some promising possible areas related to socially inspired distributed systems research.

Design Patterns

Social simulators and distributed systems researchers currently constitute very different communities with different backgrounds and goals. A major problem for moving knowledge between these disciplines is the different language, assumptions and outlets used by them. One promising approach for communicating techniques from social simulation to distributed systems designers is to develop so-called "design patterns" which provide general templates of application for given techniques. This approach has been influential within object-orientated programming and recently biologically inspired approaches have been cast as design patterns (Gamma et al 1995; Babaoglu et al. 2006). Design patterns are not formal and need not be tied to a specific computer language but rather provide a consistent framework and nomenclature in which to describe techniques that solve recurrent problems the designer may encounter. At the time of writing few, if any, detailed attempts have been made to present techniques from social simulation within a consistent framework of design patterns.

The human in the loop: Techo-social systems

Most distributed and open systems function via human user behaviour being embedded within them. In order to understand and design such systems some model of user behaviour is required. This is particularly important when certain kinds of user intervention are required for the system to operate effectively. For example, for current file sharing systems to operation (e.g. BitTorrent) users are required to perform certain kinds of altruistic actions such as initially uploading new files and maintaining sharing of files after they have been downloaded (so called "seeding"). Web2.0 systems often require users to create content, upload and maintain it (e.g. Wikipeadia). It seems that classical notions of rational action are not appropriate models of user behaviour in these contexts. Hence, explicitly or implicitly such distributed systems require models of user behaviour which capture, at some level, realistic behaviour. Such systems can be viewed as Techo-social systems – social systems that are highly technologically mediated.

One promising method for understanding and modelling such systems is to make use of the Participator Modelling approach discussed in Chapter **. In such a system user behaviour is monitored within simulations of the technical infrastructure that mediates their interactions. Such an approach can generate empirically informed and experimentally derived behaviour models derived from situated social interactions. This is currently, at the time of writing, an underdeveloped research area.

Interestingly, from the perspective of distributed systems, if it is possible to model user behaviour at a sufficient level of detail based on experimental result then certain aspects of that behaviour could be incorporated into the technological infrastructure itself as protocols.

Power, Leadership and Hierarchy

A major area of interest to social scientists is the concept of power --- what kinds of process can lead to some individuals and groups becoming more powerful than others? Most explanations are tightly related to theories of inequality and economic relationships; hence this is a vast and complex area.

Here we give just a brief very speculative sketch of recent computational work, motivated by sociological questions, that could have significant import into understanding and engineering certain kinds of properties (e.g. in peer-to-peer systems), in which differential power relationships emerge and may, perhaps, be utilised in a functional way.

Interactions in human society are increasing seen as being situated within formal and informal networks (Kirman 2001). These interactions are often modelled using the abstraction of a game capturing interaction possibilities between linked agents (Zimmerman et al 2001). When agents have the ability to change their networks based on past experience and some goals or predisposition, then, over time, networks evolve and change.

Interestingly, even if agents start with more-or-less equal endowments and freedom to act, and follow the same rules, vastly unequal outcomes can be produced. This can lead to a situation in which some nodes become objectively more powerful than other nodes through topological location (within the evolved network) and exploitative game interactions over time.

Zimmerman et al (2001) found this in their simulations of agents playing a version of the Prisoner's Dilemma on an evolving network. Their motivation and interpretation is socio-economic: agents accumulate `wealth' from the payoffs of playing games with neighbours and make or break connections to neighbours based on a simple satisfaction heuristic similar to a rule discussed in Kirman (1993).

Figure 6 shows an example of an emergent stable hierarchical network structure. Interestingly, it was found that, over time, some nodes accumulate large amounts of `wealth' (through exploitative game behaviour) and other nodes become `leaders' by being at the top of a hierarchy. These unequal topological and wealth distributions emerge from simple self-interested behaviour within the network. Essentially, leaders, through their own actions, can re-arrange significantly the topology of the network - those on the bottom of the hierarchy have little `topological power'.

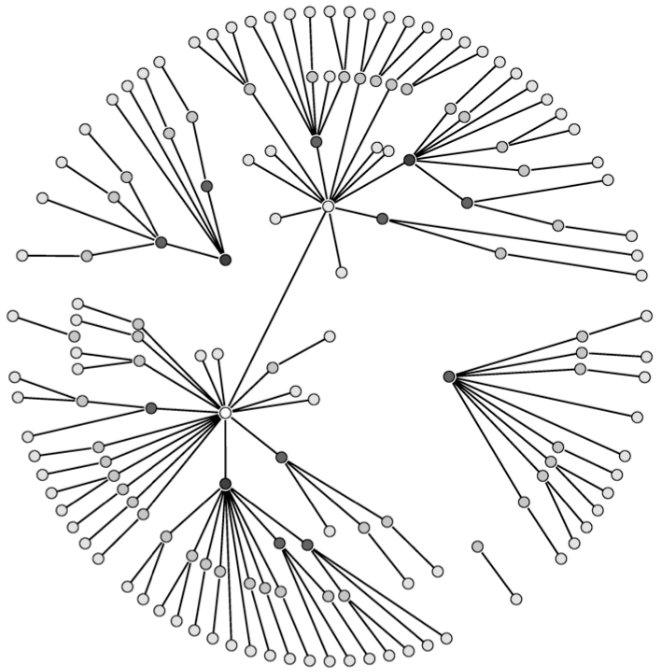


Figure 6: Forms of `hierarchy', `leadership' and unequal wealth distribution have been observed to emerge in simulated interaction networks (from Zimmerman et al 2001). Nodes play PD-like games with neighbours and break connections based on a simple satisfaction rule. Hierarchies are produced in which some nodes are more connected and hence can affect the network dramatically by their individual actions --- a form of `topological power'.

The idea of explicitly recognising the possibility of differential power between sub-units in self-* systems and harnessing this is an idea rarely discussed in engineering contexts but could offer new ways to solve difficult co-ordination problems.

Considering P2P applications, one can envisage certain kinds of task in which differential power would be required for efficient operation - e.g. consider two nodes negotiating an exchange on behalf of their `group' or `follower' nodes. This might be more efficient than individual nodes having to negotiate with each other every time they wished to interact. Or consider a node reducing intra-group conflict by imposing a central plan of action.

We mention the notion of engineering emergent power structures, briefly and speculatively here, because we consider power to be an under-explored phenomenon within evolving information systems. Agents, units or nodes are often assumed to have equal power. It is rare for human societies to possess such egalitarian properties and perhaps many self-* like properties are facilitated by the application of unequal power relationships. We consider this a fascinating area for future work.

Summary

We have introduced the idea of social inspiration for distributed systems design and given some specific examples from P2P systems. We have argued that social simulation work can directly inform protocol designs. We have identified some of the current open issues and problem areas within this research space and pointed out promising areas for future research.

Increasingly, distributed systems designers are looking to self-organisation as a way to address their difficult design problems. Also, there has been an explosive growth in the use of such systems over the Internet, particularly with the high take-up of peer-to-peer systems. Also the idea of "social software" and the so-called "web2.0" approach indicate that social processes are becoming increasingly central to system design. We believe that the wealth of models and methods produced with social simulation should have a major impact in this area over the coming decade.

References

Axelrod, R. (1984) The evolution of cooperation. N.Y.: Basic Books

Babaoglu, O., Canright, G., Deutsch, A., Di Caro, G., Ducatelle, F., Gambardella, L., Ganguly, N., Jelasity, M., Montemanni, R., Montresor, A., and Urnes, T. (2006) Design Patterns from Biology for Distributed Computing. In ACM Transactions on Autonomous and Adaptive Systems, vol. 1, no. 1, 26–66.

Babaoglu, O., Jelasity, M., Montresor, A., et al., (2005). Self-Star Properties in Complex Information Systems: Conceptual and Practical Foundations, LNCS 3460, Springer.

Boyd, R. and Richerson, P. (1985). Culture and the Evolutionary Process. University of Chicago Press, Chicago.

Brueckner, S., Di Marzo Serugendo, G., Hales, D., Zambonelli, F. (eds.) (2006) Engineering Self-Organising Systems. Proceedings of the 3rd Workshop on Engineering Self-Organising Applications (EOSA'05). Lecture Notes in Artificial Intelligence, 3910. Springer.

Cohen, Bram (2003), Incentives Build Robustness in BitTorrent. Presented at the 1st Workshop on the Economics of Peer-2-Peer Systems, June 5-6, 2003, Berkley, CA.

Conte, R. and Paolucci, M. (2002) Reputation in Artificial Societies. Social Beliefs for Social Order, Kluwer.

Dawkins, R. (1976). The Selfish Gene. Oxford University Press, Oxford.

Edmonds, B. and Hales, D. (2005) Computational Simulation as Theoretical Experiment, Journal of Mathematical Sociology 29(3):209-232.

Gamma, E., Helm, R., Johnson, R., Vlissides, J. (1995) Design Patterns: Elements of Reusable Object-Oriented Software, Reading, Mass.: Addison-Wesley.

Hales, D. & Edmonds, B. (2003) Evolving Social Rationality for MAS using "Tags", In.Rosenchein, J.S., et al (ed.) Proceedings of the 2nd International Conference on Autonomous Agents and Multi-agent Systems, Melbourne, July 2003 (AAMAS 2003), ACM Press, 497-503.

Hales, D. & Patarin, S. (2006) How to cheat BitTorrent and Why Nobody Does. Proceedings of the European Conference on Complex Systems (ECCS2006), Oxford.

Hales, D. (2002) Group Reputation Supports Beneficent Norms. The Journal of Artificial Societies and Social Simulation (JASSS) vol. 5, no. 4.

Hales, D. (2006) Article: Emergent Group-Level Selection in a Peer-to-Peer Network. Complexus 2006;3:108-118.

Hales, D. and Arteconi, S. (2006) SLACER: A Self-Organizing Protocol for Coordination in P2P Networks. IEEE Intelligent Systems 21(2):29-35.

Hales, D. and Edmonds, B. (2005) Applying a socially-inspired technique (tags) to improve cooperation in P2P Networks. IEEE Transactions in Systems, Man and Cybernetics - Part A: Systems and Humans, 35(3):385-395.

Kirman, A. (1993) Ants, Rationality and Recruitment. Quarterly Journal of Economics, 108, 137-156.

Kirman, A.P., and Vriend, N.J. (2001). Evolving Market Structure: An ACE Model of Price Dispersion and Loyalty. Journal of Economic Dynamics and Control, 25, Nos. 3/4, 459-502.

Lumsden, C. and Wilson, E. (1981). Genes, Mind and Culture. Harvard University Press, London.

M. Wooldridge and N. R. Jennings. (1995) Intelligent agents: Theory and practice. The Knowledge Engineering Review, vol. 10(2) pp. 115–152.

Rao, A. S., and Georgeff, M. P., (1991) Modeling rational agents within a BDI-architecture. In R. Fikes and E. Sandewall, editors, Proceedings of Knowledge Representation and Reasoning (KR&R-91). Morgan Kaufmann Publishers: San Mateo, CA, April 1991, pp. 473–484.

Riolo, R. Cohen, M. and Axelrod, R. (2001) Evolution of cooperation without reciprocity, Nature 414, pp. 441-443.

Schelling, T. (1969) Models of Segregation. American Economic Review 59:488-493.

Schelling, T. (1971) Dynamic models of segregation, Journal of Mathematical Sociology, 1(1):143-186.

Singh, A. and Haahr, M. (2004) Topology adaptation in P2P networks using Schelling's Model. In Proceedings of the Workshop on Games and Emergent Behaviours in Distributed Computing Environments, Birmingham, U.K., Sept. 2004.

Singh, A. and Haahr, M. (2006) Creating an adaptive network of hubs using Schelling's model. Commun. ACM 49, 3 (Mar. 2006), 69-73.

Staab, Steffen., Pedro Domingos, Peter Mika, Jennifer Golbeck, Li Ding, Tim Finin, Anupam Joshi, Andrzej Nowak, Robin R. Vallacher, "Social Networks Applied," IEEE Intelligent Systems, vol. 20, no. 1, pp. 80-93, Jan/Feb, 2005

Sun, Q. & Garcia-Molina, H. (2004) SLIC: A Selfish Link-based Incentive Mechanism for Unstructured Peer-to-Peer Networks. In Proceedings of the 24th IEEE International Conference on Distributed Systems. IEEE computer Society.

von Neumann, J. and Morgenstern, O. (1944) Theory of Games and Economic Behavior. Princeton.

Wang, Fei-Yue., Carley, Kathleen M., Zeng, Daniel and Mao, Wenji. (2007) Social Computing: From Social Informatics to Social Intelligence. IEEE Intelligent Systems, vol. 22, no. 2, 2007, pp. 79-83.

Zimmermann, M.G., Egufluz, V.M. and San Miguel (2001) Cooperation, adaptation and the emergence of leadership. In A. Kirman and J.B. Zimmermann (eds.) Economics with Heterogeneous Interacting Agents, pp. 73-86. Berlin: Springer.