Learners as modellers of complex systems: what can be automated?

Manuel Marques and Helen Pain

University of Edinburgh
Division of Informatics, ICCS
80 South Bridge, Edinburgh, EH1 1HN, UK
{ manuelp, helen }@dai.ed.ac.uk

Abstract

When building simulations of complex decentralised systems, programming is not the same as modelling. This study is the beginning of a process in which we are searching for answers to questions related to how could computers support the process of modelling. Our characterisation of modelling is influenced by ideas from Developmental Psychology, and Artificial Intelligence. This early work starts by looking at the problem from a Representational point of view.

Context

The polarisation of the reductionist thinking paradigm has been transformed into what has been called the centralised and decentralised mindsets (Resnick 1994). The world seen from a centralised view will have centres of control and coordination (amongst others) concentrated in certain parts of the whole system. On the other hand, a world conceived with a decentralised mindset will see local and simple interactions occurring between agents of the perceived world where the characteristics of coordination, order, control and many others emerge from these interactions. Systems characterised in this way are also referred to as Complex Systems. Understanding the decentralised mindset is now becoming part of the educational curriculum of many schools, a challenge for many scientists in a diversity of fields and also an issue to think about in our everyday relationship with an increasingly complex environment. Our society, to varying extents, is moving towards a stable balance of these two paradigms for understanding the world. At this stage, however, the focus is tending more and more to be on the decentralised aspect. This is because it is novel and useful to understand how several natural systems really work and also because it is in itself a completely different way of thinking which is demonstrating its importance in developmental and educational aspects.

One of the key parts of this process is, without any doubt, the computer. This is also an evolving entity, which is now able to play the role of the canvas and brushes for people to paint the pictures produced by these new ways of thinking. The sort of pictures we are interested in come in the form of computer models. In an evolutionary sense we are interested in understanding what Joel de Rosnay has called *Cybionts* (Rosnay 2000) which consist of the unified systems that humans and technological entities are starting to become (including aspects such as individual characteristics of each system and mutual adaptation for a goal driven "symbiosis").

Continuing with the analogy of the picture creation, when building computer models of decentralised phenomena, there are, in a sense, several "painting styles". As the surrealistic mind find its expression in a set of specific techniques, colours and themes, the decentralised mind, having a modelling goal, may find a way of expression in specific computer modelling tools. The foundational ideas of these tools can be very different and that will influence the sort of phenomena that can be modelled as well as the way in which they can be modelled (Schmucker 1999).

Simulation building and the "Square Question"

StarLogo (Resnick 1994, Resnick 1996) and Swarm (Minar et al. 1996), are in our view two of the most interesting tools available for building models of self-organising systems (especially from the educational point of view). These tools are not only for building models, they are also simple to use and allow the modeller to learn from the model-building experience by reflecting on their own way of thinking.

The StarLogo tutorial (available online) is about modelling the behaviour of a community of termites, specifically their natural drive to pile wood chips together. This tutorial shows a moment in the modelling process when the model design has been already done and is being "translated" into StarLogo programming code. In this example, there are many agents ruled by the same principles (termites) and they all interact with an environment which initially has a certain number of randomly distributed wood chips. The rules (in an abstract level) that govern the behaviour of each termite (as designed for this tutorial) are the following:

a) Termites move randomly in the environment until they find a wood chip. (a yellow square or patch in their environment).
b) Once a termite has found a wood chip they will pick it up and find a place where another wood chip is.

c) When carrying a wood chip and having found a place where another one is, find an empty place beside it and put the wood chip they are carrying down.

The implementation of this in StarLogo code is extremely simple. When the simulation is run, as many termites as the modeller has created will be interacting with a predetermined density of wood chips in the virtual environment. Little piles[1] start to emerge after various cycles (in each cycle all termites run their rules in a simulated parallel way) and in the "end" the geometric arrangement of the wood chips become morphologically stable in only one pile that tends to be circular. When looking at this result, a question very related to the model design sprung to our minds: how could the model be modified so that in the end we could get a triangle, or a square? This question is central to our discussion, thus we will refer back to this question from now on as the *square question*.

The answer to the *square question* is not as simple as some would expect it to be. An expert would, perhaps, start analysing the initial conditions of the simulation, the way turtles move, the design of the environment and even the functions that generate the random numbers used in the simulation, among other elements. One reason for this is that the rules for the termites' behaviour used in the tutorial model do not explicitly prescribe the formation of a circle-like shape. Although this way of looking at the problem opens up many interesting branches in which to find valuable clues for a solution, there are still other ways of looking at this and certainly some of them will be more suitable and natural for non-experts and learners (also for some experts as well). In order to find out more about these other ways, we must consider the following dimensions:

a) cognitive aspects of the modeller's mind;
b) the design of simulation building tools and the computational environment in which they work;
c) the domain of knowledge related to the phenomena being modelled;
d) the interaction between the modeller and the computer;
e) the model's design seen as an artefact (which should, in general, reflect its designer "thinking style")

Our focus is on how and to what extent computers reflect the cognitive activity of the modellers in terms of how the former could be able to support the human's model building *process*, at least to more than only allowing high level programming constructs to be introduced and executed, taking into account individual thinking styles and a diversity of knowledge resources (this shall be developed in more detail later). StarLogo and Swarm (as well as some other tools) have started to generate solutions in this direction, offering gadgets to analyse and viewing different aspects of simulations such as dials (to control numbers of agents, for example), monitors (to know the value of variables) and the ability to produce graphics. An enormous potential is still to be exploited in this area of simulation building tools design.

In the type of simulation building tools we are considering, the cognitive reflection introduced above occurs in an unidirectional way (clearly identifiable and analysable from modeller to computer) when they write StarLogo or Swarm source code (and describing the model in these terms) and in the opposite direction when they observe the model in action. These two types of basic interaction (or non-interaction to be more precise) seem to correspond to two peak stages or cycles in knowledge acquisition processes occurring in modeller's mind. Developmental psychologist Annette Karmiloff-Smith has proposed a cognitive account of these processes in what she has called the *Representational Redescription Model* (RR Model, Karmiloff-Smith 1992). However, before getting into details about the RR Model, we shall introduce a small set of other concepts and issues which will be useful to set the appropriate context in which the RR Model relates to the main issue of this paper: the *square question*.

Conceptual Spaces, Creativity and The Cognitive Pyramid

The first concept to be discussed in this subsection is what Margaret Boden defines as *Conceptual Space* (Boden 1990). The second reifies the idea that creativity is a key issue in human modelling activities (Langley et al. 1987, Boden 1990, Naur 1994, Klahr et al. 2000). The third and final foundational concept is what we will define as the *Cognitive Pyramid* that encompasses the previously mentioned elements under a dynamicist approach.

A C-space (Conceptual Space) is a way of defining a knowledge domain using notational basic building blocks (Generative Elements) governed by basic principles (Generative Principles) which constrain their possible organisational structure. The spaces defined by these building blocks and principles provide the potential characteristics and applications of a C-space.

One very basic example of a C-Space is the board game Tic-Tac-Toe. In this game the *Generative Elements* are the noughts and crosses used by the players and the 3x3 matrix board and the *Generative Principles* are the game's rules: each player will take turns, one of them starts and during each turn players will place a nought or a cross in an available place (empty) on the board. The game ends either when one of the players has three spaces of the board occupied with their chosen symbol (nought or cross) in a line or diagonal or when the board is full and no more turns can take place. For an excellent exploration into the nature of this sort of characterisation see John Holland's work on Arthur Samuel's Checkers Player (Holland 1998).

The general goal of the game described is for each player to try and get three of their symbols in a consecutive row,

[1] In the StarLogo termites model, piles are two dimensional only. This means that no wood chip is placed on the top of another one.

column or diagonal. The aim of achieving the general goal results in a set of *heuristics* or ways of playing the game. in C-space terms, its particular organisation will provide these heuristics as the possible ways of "navigating" it. This is important because it is possible to conceive knowledge domains as something that can represented using different C-spaces. Specific types of C-spaces can be generated according to the kinds of heuristics that will solve the particular goals that a cognitive system may have when creating them.

The enterprise of exploring the nature of Conceptual Spaces, although seemingly a daunting prospect, offers the possibility of finding many of the answers needed to improve the way humans think and conceptualise their world. This is valid either when learning new things for solving concrete problems as well as when determining how computers could offer the possibilities of novel kinds of interactions with a view to the same goals. One of the main aids within our reach is the potential diversity of a) *External Notations* used by people to put their ideas across. These notations determine how b) *learning* takes place, basically by interacting with and manipulating them, by c) *reasoning* about a domain in order to solve problems. If we imagine these three aspects of the human consciousness forming a triangle on a plane, it is possible to see an interlocked complex system of interdependencies.

These complex interdependencies are substantial issue on their own. Variations in representation might result in less or more effective overall performance of the cognitive system within its environment, (where performance could be defined in several different ways). The availability and use of potential "candidate" heuristics will influence the way learning occurs. In a similar way, errors in learning trials might trigger representational redescriptions, and so on. That is, the dimensions of Representation, Learning and Reasoning are important not only independently but also in the combined system that they constitue. We will call this whole system *C-triangle* (Cognitive Triangle) from now on.

One of our arguments is that many unexplored aspects of human creativity (and consequently of the human modelling activity will emerge from the C-triangle (if seen as a complex whole), making it possible to give a more complete and computationally plausible account of the creative ability of natural cognitive systems. The main advantage of this kind of account is that it can be computationally modelled, explored and set into evolution in artificial systems. Following a similar naming convention, we will refer to this expanded C-triangle as C-*pyramid* for Cognitive Pyramid. The internal states of the system just defined – the C-pyramid – vary according to a series of recurrent phases. These phases are what constitute the foundation of the RR model.

The Representational Redescription Model

Now we can resume the description of the RR model in our current context. The RR model consists of three *recurrent phases* (Karmiloff-Smith emphasises that they are not stages or steps, Karmiloff-Smith 1992). The first phase consists of the cognitive system focusing on the environment and receiving as input "raw data". Data are raw in the sense that their content does not relate to or alter previously existing knowledge in the cognitive system. This phase lasts until the system has achieved what Karmiloff-Smith calls "behavioural mastery", meaning that enough interactions with the novel context have occurred, and hence, a range of answers are available for a variety of inputs or situations.

During the second phase the cognitive system stops focusing on the environment, and the internal dynamics of the system start to be the centre of activity. Internal representations are used instead of environmental data. This fact can lead, as Karmiloff-Smith suggests, to inflexibility and error, being that part of the raw material used for learning. External notations do not occur until the very end of this phase when very basic conscious access to the "outputs" of these internal dynamics are manifested in the form of drawings, diagrams or vague verbal explanations. A common phrase we will hear people saying when going through this phase is: "I know how to do it, but I cannot tell you how or why!"

The third phase takes place when the internal dynamics of the cognitive system become "stable" and they can be increasingly accessed consciously and related to acquired knowledge within the current and other external domains. As very little is known about the nature of internal representations, even less is known about the dynamics that produce states in the cognitive system that produce the shifts between levels. The RR model suggests interesting clues for solving this puzzle (Karmiloff-Smith 1992).

With regard to the representational formats Karmiloff-Smith differentiates four different format levels. She calls them *Implicit* (I), *Explicit level 1* (E1), *Explicit level 2* (E2) and *Explicit level 3* (E3).

The format level *I* consists of black-boxes (content not accessible) "compiled" procedures for analysing and responding to changes in the environment. At this level procedures are independent and no relations exist between them and any other piece of knowledge, either within the current domain or across other domains. Format level *E1* results from a process of *redescription* in which a constructive process happens, linking (in an evolutionary way) successful procedures with other pre-existing knowledge, these new procedures being redescribed in terms of what already resides in the cognitive system. This format level is more abstract that the *I* level but still no conscious access or explicit description is possible. This starts to become available when redescription occurs again leading to the *E2* format level, in which basic notations can be used to redescribe the understanding that is internally taking place – usually in the form of drawings, diagrams or simple verbal

accounts. The highest representational format in the RR model is *E3,* in which fully conscious access to the contents of knowledge "stored" in the cognitive system is possible as well as the creation of inter-domain connections and manipulation of more sophisticated notational systems.

Our Story so far

Putting the fragments of our story together, we can start by conceiving tools such as StarLogo or Swarm as C-Spaces themselves. In this way, simulations built using them are subspaces just as a particular song is a subspace of the C-Space of all possible songs. This type of conceptualisation will prove to be of great importance when trying to conceive simulations as artefacts. Computer models defined in this way resemble natural landscapes under the view of a cartographer and this means that some analogous functions can be explored. For example, the map construction activity (in which different types of map satisfy different cartographic goals) will have strong conceptual connections with idea of Ontologies and consequently to associated heuristics.

With regard to the *square question*, a first step could be taken by seeing the cognitive activity of model building from the perspective of the RR model. Building correspondence mappings between the phases of the RR model and what can be observed during the interactions between modellers and simulation building tools, it can be clearly noticed that the second phase of the RR model (between the unavailability and availability of accounts for what has been learnt) has no counterpart in the set of possible interactions with simulation building tool. In terms representational format levels this missing aspect maps to the *E1* and *E2* levels, when and where, from the developmental point of view, knowledge is being actively *constructed* in a Piagetian sense (new contents are built from pre-existing modules).

More specifically we can characterise the situation as follows:

a) The interaction between the learner/modeller and the simulation building tool starts after a process of which there are no inputs but the actual programming code. This high level descriptions (in RR format) belong to the late verbal *E2* and *E3* levels; their contents come fundamentally from the modeller's design process which is carried out without any supporting interaction with the computer.
b) The simulation ends or starts a process of change when the learner/modeller observes in the virtual environment how the model is performing. This corresponds to the *I* representational level of the RR model. When this is happening, the only way back to the model (from the computer's perspective) occurs by reflecting late verbal *E2* and *E3* representational level content (a).

One of our main claims is that a combined framework consisting of the C-pyramid and the RR model, is valuable :

a) for making clearer the difference between modelling and programming from the computer's perspective in relation to the modeller's mind;
b) for finding many of the missing pieces for a more symbiotic Cybiont (in the context of educational simulation building).

The cognitive aspects of modelling, and the missing elements for a more symbiotic Cybiont, can also be an input for a complementary perspective more related to the specific goal of building or creating the design for a simulation. This perspective concerns the creative process itself. Graham Wallas (Wallas 1926) identified four stages in the process of creating a novel item -- Preparation, Incubation, Illumination and Verification. Our combined framework, consisting of the RR model and the C-pyramid, will be used as the cognitive foundation stone to endow computer systems with the ability to support (in better ways) the stages of Preparation and Incubation, with a view to supporting the last two stages as well. We shall now move to discussing the issues regarding the computational component of the Cybiont.

Automation and the "Square Question"

The problem of automation in the present context is the core issue to be discussed in this section. What can be automated? There is not a straightforward answer to this, however, if we put this question in the framework of the C-pyramid (described in the previous section) we can get a interesting sense of direction.

The present authors have firstly chosen to focus on the representational vertex of the C-pyramid and from the results (to be obtained) set out to explore the connections that the development of this aspect will have on the learning and reasoning vertexes. This will subsequently lead towards conclusions about the emergent aspects of creative thinking at a later stage. Our ongoing work currently focuses on the exploration of two different, but complementary, approaches to solving the *square problem*. Both try to offer alternatives for improving the model construction process by embodying a diversity of "knowledge resources" in simulation building tools such as StarLogo or Swarm. The first of these is conceptualised in a predominantly top-down manner whereas the second goes in the opposite way (bottom-up) focusing, as stated, on the representational perspective.

Languages that learn and adapt

The first approach we are investigating consists of the incremental development of a visual language which would work on top of the StarLogo/Swarm current programming languages. This extension will remain faithful to commonly used design principles such as object orientation, grammar-like organisation of valid code constructions, iconicity and so on.

The novel aspect of this consists of designing more complex building blocks to become part of the programming language. Several sets of language items can be endowed with a *partial* digital chromosome as defined by the theories of Genetic Algorithms and Genetic Programming (Mitchell 1995, Michalewicks 1995, Banzhaf et al. 1998). Each complete valid programming construction must have a *complete* digital chromosome. This kind of structure would be able to encode possible "meanings" of a particular construction in a specific context according to its "experience" or explicit training. The aim is for the language to be capable of "knowing something" about ontologies (contexts in which its constructions are being used) and related goals that become plausible and meaningful in each of them. This would mean a diversification (although still somehow primitive) of the possible meanings of a visual programming construction.

The main problem of this approach is the fact that the definition of chromosome's alleles (which are the building blocks of the chromosome that encode a single trait or aspect of the whole construction) have to be developed and we do not yet have a framework or reference to decide what the traits for different blocks could be or what abstract variables could offer relevant information when building models based on the decentralised paradigm.

The idea of a visual programming language that learns and adapts to different modelling contexts and stages is not discarded, but certainly it is a mandatory requirement that the alleles from which the digital chromosomes are to be built have a stronger foundation still unavailable. Our second line of research starts to address this issue.

Bringing development to model building

The puzzle described in the previous subsection has several facets related to the different aspects of the C-pyramid. Keeping the focus on the representational vertex there is still a lot of complexity linked to the nature of "concepts" which basically can be categorised in two main dimensions: The *Encodigist* and the *Interactivist*. (Bickhard and Terveen 1995)

The encodigist dimension refers to the traditional approach to concepts in which notations have, a correspondent "meaning" counterpart in the cognitive system. Mark Bickhard presents arguments against this (Bickhard and Campbell 1996) and favours an interactivist approach to concepts in which they change from being entities of their own to become a dynamic source of potential interactions. To make this difference between Encodigism and Interactivism clearer we can think about the concept of *door*. For the encodigist a door will have a series of structural characteristics like shape, colour, position, its mobility and so on. (this allows for the classical structure mapping commonly used for modelling analogical reasoning in Cognitive Science where analogies consist basically of structural matching processes) For the interactivist, a door represents the possibility of pushing it (for example) in order to accomplish the goal of entering/leaving a space. Our claim is that these two approaches are not mutually exclusive. Moreover, we are assuming that they mutually "depend on" each other. Nevertheless, a more detailed discussion of this is out of the scope of this paper.

Sticking to the RR model ideas, and the correspondence we have discussed between this and the process of model construction in StarLogo or Swarm, we have set the focus of our work in "bridging" between the virtual environment in which models are executed (where information is represented in a computational Implicit *I* analogous format) and the programming code used to create the models. Our current ongoing work is on finding pieces of knowledge to account for the *E1* and *E2* analogous in the simulation building environment.

For this, we are initially approaching the context and virtual environment in which the *square problem* was found with a connectionist view. More concretely, the StarLogo running environment becomes a two-dimensional grid of sensors where dynamic patterns of termites' behaviours and arrangements of wood chips take place. These patterns can be analysed using feature mapping (connectionist) algorithms (Hertz et al. 1991, Kohonen 1997).

This feature mapping work corresponds to one thread of what the first RR computational phase should be. The concept of this thread is derived fundamentally from the research programme proposed by Peter Gärdenfors (Gärdenfors 2000) in which he supports the idea of a *Conceptual Level* connecting the subsymbolic (connectionist) and symbolic approaches to representation. Gärdenfors also suggests that topographic and geometric characteristics of dynamical systems could be the missing link between the two most explored and still disconnected views on Knowledge Representation.

Our work, then, consists basically of two main cycles. In the first we are looking for taxonomies of patterns (termites' behaviours and chips arrangements, for example) categorised in geometric/topographic property-mappings. Once this taxonomised information is collected and processed, it can be used (in a second cycle) to support the construction of a *E1-E2* visual language (based on the principles of C-Spaces).

This visual language based on the collected data from the first cycle will help us understand the implications of building an analogous computational RR first phase (first cycle) when trying to build analogues further up in the RR computational model (the actual usable visual language). These data will also be used to start building a framework to provide information that could be valuable in solving the design problem of the digital language items' chromosomes/alleles discussed previously.

"Resourcefulness" and improvement

A good follow-up question to what has been just presented relates to the extent to which the final products, which will make complete or partially complete a computational (and symbiotic) analogue of the RR model, are usable by learners in modelling contexts. This is a question that, in our view, is closely linked to the field of Intelligent/Adaptive Human Interfaces.

In previous sections we have discussed the idea of C-Spaces as defined by Margaret Boden and mentioned then how the nature of the generative principles of a C-Space will shape to a large extent, the heuristics applicable for navigating these spaces according to goals that can be accomplished depending on the characteristics of a particular space (domain ontology). Consequently, the issue of usability becomes a problem of adaptability of a new kind of system which interacts with humans having more resources at hand.

References

Banzhaf, W. Nordin, 1998. P. and Francone, F. *Genetic Programming, An Introduction.* Morgan Kauffman.

Bickhard, M. Terveen, L. 1995. *Foundational Issues in Artificial Intelligence and Cognitive Science: Impasse and Solution*. Elsevier Scientific.

Bickhard, M. Campbell, R. 1996. Topologies of Learning and Development. *New ideas in Psychology*, 14(2), 111-156.

Boden, M. 1990. *The Creative Mind, Myths and Mechanisms*. Weidenfield and Nicholson.

Gärdenfors, P. 2000. *Conceptual Spaces: The Geometry of Thought*. MIT Press.

Hertz, J. Palmer, R. Krogh, A. 1991. *Introduction to the Theory of Neural Computation*. Perseus Press

Holland, J. 1998. *Emergence*: From Caos to Order. Oxford University Press.

Karmiloff-Smith, A. 1992. *Beyond Modularity: A Developmental Perspective to Cognitive Science*. MIT Press

Klahr, D. Dunbar, K. Simon, H. 2000. *Exploring Science*. MIT Press

Kohonen, T. 1997. *Self Organizing Maps*. Springer Verlag.

Langley, P. Simon, H. Bradshow, G. and Zytkow, J. 1987. *Scientific Discovery:Computational Explorations of the Creative Process*. MIT Press

Michalewicks, Z. 1995. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag

Minar, N. Buckhart, R. Langton, C. and Askenazi, M. 1996. The Swarm Simulation System: A Toolkit for Building Multi-Agent Systems. www.santafe.edu/projects/swarm

Mitchell, M. 1995. *An Introduction to Genetic Algorithms*. MIT Press

Naur,P. 1994. *Knowing and the Mystique of Logic and Rules*. Kluwer Academic Press

Resnick, M. 1994. *Turtles*, *Termites and Traffic Jams: Explorations in Massively Parallel Microworlds*. MIT Press

Resnick, M. 1996. *New Paradigms for Computing, New Paradigms for Teaching*. In *Constructionism in Practice: Designing, Thinking and Learning in a Digital World*. Lawrence Erlbaum Associates.

Rosnay, J. 2000. *The Symbiotic Man*. McGraw-Hill.

Schmucker, K. 1999. *A Taxonomy of Simulation Software (A work in progress)* In *Learning Technology Review*. www.apple.com/education/LTReview/spring99/simulation

Wallas, G. 1926. *The Art of Thought*. Butler and Tanner