THE PROBLEM WITH ASYNCHRONOUS UPDATING

Joseph Kehoe
Institute of Technology, Carlow
email: `joseph.kehoe@ITCarlow.ie`

KEYWORDS
Asynchronous Updating, Synchronous Updating, Agent Based Models

ABSTRACT

Asynchronous updating is the default mode of updating used in Agent Based Models and Agent Based Social Systems. It has been long known that the same simulation can have different outcomes when different updating strategies are used. This means that results may be "artefacts" of the specific updating strategy employed. It is shown here that asynchronous updating breaks the assumption of bounded rationality and leads to inconsistent transmission of information across simulation spaces. An analysis demonstrates that simulations with high population densities (such as Cellular Automata) will make accentuate this effect. These results help explain findings from previous work on comparing different updating strategies. It also calls into question the preference for using asynchronous updating instead of synchronous updating.

INTRODUCTION

A simulation is the imitation of a system over time. Development of a simulation is dependent on having a model of the system being simulated where this model represents the key characteristics or behaviours of the system. An Agent Based Model (ABM) is one where individuals within the system and their interactions are explicitly represented. Within ABM updating is either Synchronous (SU) or Asynchronous (AU). (Huberman and Glance 1993) distinguishes between these approaches.

The SU approach is defined as having a global clock that synchronises the updating of all agent states so that all updates occur in unison. AU, on the other hand, have no global clock and updates of agent state occur in some defined sequential order. It has been demonstrated that SU and AU implementations of the same ABMs can result in widely differing behaviours. It is assumed that real world systems are asynchronous and the AU implementations correlate more closely with the reality they are attempting to simulate[1].

Although we are concerned here with ABM the issues concerning SU and AU implementations are also applicable to Discrete Event Simulations (DES). (Brailsford 2014) states that DES is a subset of ABM while (Onggo 2010) claims that any ABM can be translated into an equivalent DES. In any case the two approaches have much in common (Siebers et al. 2010) and the issue of SU or AU implementation of events is relevant to both approaches.

The assumptions made in (Huberman and Glance 1993) regarding real world simulations are examined and it is shown that although it is the case that AU and SU approaches can give different results, it has not been conclusively demonstrated that the asynchronous approaches are the more accurate. We show that the approach to implementing asynchrony advocated in (Huberman and Glance 1993) contains some flaws.

In the next section the differences between the synchronous and asynchronous approaches are outlined. Following this the problems associated with AU are examined. Finally the conclusion lists a summary of the arguments in favour of a synchronous approach.

SYNCHRONOUS AND ASYNCHRONOUS UPDATING

In the paper *Evolving games and computer simulations* (1993) a case was made for an asynchronous approach to real world simulations. Synchronous simulations are defined as those in which all the agents in that simulation are updated simultaneously and instantaneously at each time step. Each step is a discrete quantum of time and the simulation progresses as a sequence of discrete states, one per time step. The state at step n is dependent solely on the state at step $n-1$. The best known simulation of this type is Conway's Game of Life (Gardner 1970).

It is argued (Huberman and Glance 1993, Caron-Lormier et al. 2008, Cornforth et al. 2005) that natural social systems belong to the class of asynchronous systems where there is no "global clock" that causes all agents to update their state in unison. These papers cite, in support of this, the fact that agents in a social setting act at different and uncorrelated times on the basis of information that may be imperfect and delayed.

[1] We define accuracy in terms of how closely a simulation replicates the real-world system being modelled. For a simulation to be accurate anything that is not allowed within the system we are modelling must also be excluded from the simulation.

Using SU, Nowak developed a simulation of Spatial Iterated Prisoners' Dilemma (Nowak and May 1992) and showed that the simulation generates chaotically changing spatial patterns, in which *cooperators* and *defectors* both persist indefinitely. (Huberman and Glance 1993) then used the AU approach on the same simulation to instead show that the simulation always evolves, within 100 generations into a steady state where all agents become *defectors*. Thus we have two clearly contradictory results deriving from the application of the same rule differing only in the updating technique.
The conclusion drawn by (1993) is that to mimic continuous real world systems we need a procedure that ensures the updating of interacting agents is continuous and asynchronous. AU is implemented by:

> choosing an interval of time small enough so that at each step at most one individual agent is chosen at random to interact with its neighbours. During this update, the state of the rest of the system is held constant. The procedure is then repeated throughout the array for one player at a time, in contrast to a synchronous simulation in which all the agents are updated at once. (Huberman and Glance 1993)

This finding has been backed up since then by numerous studies comparing synchronous and asynchronous Cellular Automata (Caron-Lormier et al. 2008, Schönfisch and de Roos 1999, Cornforth et al. 2005, Ruxton and Saravia 1998). It has become standard practice to use AU for Agent Based Models (ABM) and in particular Agent Based Social Simulations (ABSS). It has been adopted by the major agent toolkits such as NetLogo, Repast, Mason and Swarm(North et al. 2013, Luke et al. 2005, Berryman 2008). Standard ABSSs, such as the canonical Sugarscape (Epstein and Axtell 1996), assume an AU implementation. The ability to execute a sequence of agent actions in a random order, an essential part of asynchronous simulations, forms part of *StupidModel* (Railsback et al. 2005), a suite of models designed to test the suitability of any toolkit for ABM development.
For AU to work each action would have to be instantaneous in time (Fatès 2013), that is each action has no duration, otherwise it makes no sense to say that the time interval is fixed in size yet any number of sequential actions can take place within this time interval.
Some researchers assume that real world systems are asynchronous in nature (Caron-Lormier et al. 2008, Cornforth et al. 2005). Rather interestingly, there does appear to be consensus that the order of the updates is important yet debate remains as to which is best (Ruxton and Saravia 1998).
The preference for AU has been accentuated by a lack of synchronous algorithms that can handle the complex interactions that occur in ABM. This contrasts with Cellular Automata (CA) based simulation where, because the interactions are simpler SU algorithms exist. For example, CA-based Simulations of traffic flow, a real world system, employ SU (Burstedde et al. 2001) as standard. It is not uncommon to see CA based simulations employ both updating methods and comparisons made from the results as in (Bezbradica et al. 2014, Bach* et al. 2003). Even in CA, Grilo and Correia (2011) states that the effects of the two different strategies are not well understood.
It is our contention that the case for choosing asynchronous over synchronous in ABM has not been convincingly made and given that it is accepted that the outcomes of a simulation can depend on the approach taken we need to reexamine the preference for the AU and exclusion of SU. In this paper it is demonstrated that asynchronous updating violates *Bounded Rationality*, an important property of Agent Based Social Simulations.

FLAWS IN ASYNCHRONOUS UPDATING

Bounded rationality is the idea that in decision-making rationality of individuals is limited by the information they have available to them, the cognitive limitations of their minds, and the finite amount of time they have to make a decision. Many consider this an essential property within ABSS Epstein (1999). Within ABM this principle is generally taken to mean that agents do not have access to global information. This principle of bounded rationality is enforced in ABM by ensuring that *an agent can only be aware of the state of other agents within its neighbourhood (locality)*. This places limits on the information agents have available at any one time. They can see their locality as it is now and gain information about the world outside their locality through time delayed information transfer (e.g. gossip). Locality is, of course, defined in a simulation specific way. In some cases locality resembles physical proximity in the real world but in others (for example, simulating Facebook or Twitter connections to study how memes traverse social networks) it does not. Here locality is defined information theoretically. Anyone who I receive information from in real time (instantaneously) is in my "locality" or "neighbourhood". If I am undertaking a video conference call with colleagues then the people physically present in the room with me as well as the people on screen (even though they are physically removed from me) are in my locality but everyone else outside of the room is not.
A local neighbourhood may change from step to step but within each time step it is fixed. The results of actions taken during the current step can change the neighbourhood of an agent for the following step. This principle guarantees that an agent cannot be omniscient, that is, an agent cannot be aware of the global state as it is *now* or have complete knowledge of the universe. If

an agent is not in my immediate neighbourhood then I cannot know what it is doing now. I may find out at a later time what it was doing now through time delayed information diffusion (e.g. gossip) but that is time delayed information. SU guarantees bounded rationality by imposing a consistent speed for the transmission of information across a simulation space. If this property is important in a particular ABM then more consideration should be given to using SU.
Take as a *gedankenexperiment* or thought experiment the case of a simulation where agents perform some action as soon as they become aware that some particular agent A is dead. An agent B can become aware of this fact under two conditions:

1. If B is a neighbour of A and witnesses A's death;

2. If B is not a neighbour of A but one of B's neighbours is aware of A's death and informs B.

Now once A dies the amount of time steps it takes for any other agent to find out should vary in direct proportion to the distance that agent is from A. Agents in the immediate locality of A should be first to know (as they witness A's death) and this information should then percolate through the system from local neighbourhood to local neighbourhood over one or more subsequent steps. Under SU this is exactly what happens. Now consider what happens under AU if we are an immediate neighbour of A. Once A dies in step i when do we find out? It depends on the order in which the actions occur *during* the step. If A's behaviour is scheduled under AU to occur before ours in this step then we will be aware of A's death within this step (instantaneously). However, if A's behaviour is scheduled after ours then we will not find out until the following step. On average half of A's neighbours will be aware of its demise immediately and half will not!
This random sequencing of actions means that agents **not** within A's neighbourhood can also be aware of A's change of state instantly (within the same time step). We can construct a specific sequence of an update ordering $[a_1, a_2, ..., a_n]$ where each agent in the sequence a_i runs before a_{i+1} within this step and every a_i is a neighbour of a_{i+1}. If a_1 dies then every agent in the chain will pass this information on to their neighbour within the same step. This gives a_n instant access to information from outside of its neighbourhood.
There are two related issues here:

1. Agents who are not within the locality of A can be immediately aware of the change of state of A thus violating bounded rationality;

2. Agents further away from A can be aware of A's change of state before agents closer to A are (inconsistent speed of information transfer through the system).

It is clear that instantaneous information transfer accross local boundaries (we term such events *leakages*) can be caused by AU but to determine how much this can affect simulations we need to know how often such leakages occur.

PROBABILITY OF INFORMATION LEAKAGE ACROSS LOCAL BOUNDARIES

For a leakage to occur a chain of three (or possibly more) agents must be updated in an order that allows information to move across neighbourhood boundaries in a single timestep. The amount of leakage will be directly proportional to the number of such chains occuring during each timestep.
To help calculate the liklihood of leakages occuring we make some basic assumptions:

- The lattice in which the ABM runs is a torus structure. That is, it wraps around the edges. This is true of Sugarscape and not uncommon in general. This asumption makes the following calculations simpler but the use of a non torus structure does not have a major effect on the outcome of the calculations;

- The locality of an agent is determined by a range R. That is, all locations within R steps of the current location are within its locality. We also assume that $R < N/2$ where N is the dimension of the simulation lattice. This is true in all but the most trivial ABM;

- Agents use a *von Neumann* neighbourhood when calculating locality. In a one dimensional lattice the neighbourhood extends to the left and the right. In a two dimensional lattice the neighbourhood extends in four directions (left, right, up and down);

- Agents are distributed evenly throughout the lattice;

We will return to reexamine the implications of the last two assumptions after we make our calculations.

One Dimensional Lattices
To begin with we will limit ourselves to one dimensional lattices. Each location has neighbours only to its left and right. Leakage in this case can only occur between a minimuim of three locations where these locations have the following properties:

- The first location, A, is a neighbour of the second location B;

- The second location B is a neighbour of the third location C;

- The third location C is **not** a neighbour of A.

We will label any three locations that satisfies these properties a *Leakable Triple*. Such a chain of locations will be updated under AU in one of the following orders

1. A B C;
2. A C B;
3. B A C;
4. B C A;
5. C A B;
6. C B A.

Two of these six possibilities result in leakage (sequences 1 and 6 only) while the other four of the six possibilites (sequences from 2 to 5) do not. In a one dimensional lattice of N locations we can determine the number of such chains as follows:
We first need to enumerate the total number of possible *leakable triples* in the lattice. Then we need to calculate how many of these possible leakable triples have all three locations occupied by agents. Finally, we need to calculate how likely each occupied leakable triple is to be updated in an order that causes leakage. This, as we have seen above, is $\frac{2}{6}$ or 33%.
Every leakable triple has a rightmost location defined by the lattice dimensionality. Therefore if we can calculate the number of leakable triples that have a location L_i as a rightmost location then we can use this to calculate the total number of leakable triples on the lattice (by multiplying the number of leakable triples with a fixed rightmost location by the total number of locations in the lattice).
For any starting location A there will be R locations within its locality (or neighbourhood) to the left.

- The first such location (call it B) is one step away and will only have one other location (call it C) within its range but outside of A's neighbourood;
- The second location to the left of A is two steps away and this location will have two other locations within its range and outside of A's neighbourhood;
- The third location to the left of A is three steps away and this location will have three other locations within its range and outside of A's neighbourhood;
- The i^{th} location ($i < R$) is i steps away and this location will have i other locations within its range and outside of A's neighbourhood;
- The R^{th} location is R steps away and this location has R other locations within its range and outside of A's neighbourhood.

In total we can see that this gives a total of $1 + 2 + ... + i + ... + R$ (or $\Sigma_{i=1}^{i=R} i$) chains for this one rightmost location. In general terms we can see that this gives a total of $\frac{R(R+1)}{2}$ leakable triples with the same rightmost location. Since there are N locations in the lattice the total number of chains must be N times this number:

$$N \times \frac{R(R+1)}{2} \tag{1}$$

Each chain is relevent only if all three locations contain agents. If there are A agents on a lattice of size N then the probability of a location being occupied is $\frac{A}{N}$. We let P represent this probability of a location containing an agent, $P = \frac{A}{N}$. For CA, it is always the case that $P = 1$ as each location is also an agent ($A = N$) but for an ABM this figure will be lower as the number of agents will be less than the number of locations. The probability of all three locations within a chain containing agents is P^3. Therefore the number of occupied chains is this times the total number of possible leakable triples:

$$P^3 \times N \times \frac{R(R+1)}{2} \tag{2}$$

We know that the possibility of AU updating such a chain in a way that preserves locality is $\frac{2}{3}$ so the probability of locality being preserved throughout the entire lattice is:

$$(\frac{2}{3})^{P^3 \times N \times \frac{R(R+1)}{2}} \tag{3}$$

It immediately follows that the possibility of leakage is then given by equation 4:

$$1 - (\frac{2}{3})^{P^3 \times N \times \frac{R(R+1)}{2}} \tag{4}$$

As R and P increase in value to the probability of leakage reaches 1 *during each step*.

Two Dimensional Lattices
Extending this to a two dimensional $N \times N$ lattice we can see that this lattice consists of N horizontal one dimensional lattices of size N as well as N vertical one dimensional lattices of size N. Combining these we get a total number of possible chains equal to:

$$2 \times N \times N \times \frac{R(R+1)}{2} \tag{5}$$

Therefore the probability of no leakage occuring in an $N \times N$ lattice is:

$$(\frac{2}{3})^{P^3 \times N^2 \times R(R+1)} \tag{6}$$

From this we can again deduce that the probability of leakage during each step is (equation 7):

$$1 - (\frac{2}{3})^{P^3 \times N^2 \times R(R+1)} \tag{7}$$

In any CA it is always the case that $P = 1$ and generally the case that $R = 1$ making the probability of no leakage $(\frac{2}{3})^{2N^2}$. For any reasonable value of N therefore the probability of leakage occuring approaches 1. In an ABM on the other hand $P < 1$. Remembering that $P = \frac{A}{N}$ we can see that in general by inserting $\frac{A}{N}$ for P and simplifying we get:

$$(\frac{2}{3})^{\frac{A^3 R(R+1)}{N}} \tag{8}$$

Based on this we can see that leakage is much more likely to occur in a CA than an ABM. Of course this is based on the assumptions listed at the start of this section. These assumptions are reasonable but there are some final points that we should note:

- The assumption that agents are distributed evenly throughout the lattice is open to question. In an ABM this is not necessarily the case. Often the simulation will have clustering of agents into groups resulting in a higher likelyhood of leakage within these groups and less leakage between groups;
- ABMs require inter agent communication for overall behaviours to emerge. The liklihood that we will design an ABM that has little communications between agents is low. Most simulations will have a large amount of interagent communication thus leading to increased probability of leakage;
- If a *Moore* neighbourhood is used instead of a *von Neumann* neighbourhood then the probability of leakage will increase.

This analysis leads to the conclusion that leakage is a bigger issue in CA than ABM. It also explains the finding in Caron-Lormier et al. (2008) that high population densities enhance the differences in outcomes between asynchronous and synchronous versions of the same ABM.

Any simulation where bounded rationality and/or consistency in the speed at which information spreads through the simulation space are necessary properties should therefore consider using SU to ensure that any results are not due to artefacts of the AU strategy. Of note here are two findings from other researchers.

1. May (1973) found that delay is a potential contributor to periodicity in systems. AU can interfere with, or even remove, the delay that we would expect in any system with bounded rationality (remember bounded rationality is enforced by the fact that agents can only view their immediate neighbours). Any natural systems where this delay is present would need to be cautious about employing AU or employ SU alongside AU for comparison. Our analysis explains why SU is the preferred updating strategy in traffic flow and pedestrian dynamics simulations as it picks up the important periodic changes in overall state (e.g. traffic jams caused by cars slowing as they pass an accident) by properly modelling the flow of information through the system.

2. High population densities enhance the differences in outcomes between asynchronous and synchronous versions of the same ABM Caron-Lormier et al. (2008). The fact that in high density populations violations of bounded rationality are more likely to occur gives rise to larger differences in outcomes. This helps explain the stark differences produced by the two versions of the Spatial Iterated Prisoner's Dilemma as the simulation had a maximal population density of one agent per location.

The synchronous approach, in contrast, guarantees bounded rationality and imposes a consistent speed on the transmission of information across a simulation space.

CONCLUSIONS

It is commonly assumed that ABM, and particularly ABSS, are more accurate when modelled using AU rather than SU. While asynchronous and synchronous simulations can have different outcomes there is no evidence that the asynchronous outcomes are more accurate than asynchronous ones. AU posits that by sequentially updating the state of each individual agent while holding all other agents' state constant we obtain the behaviour of an asynchronous system. For this argument to hold each action would have to be instantaneous in time, something that is seldom true in the real world systems.

It has been shown here that the use of asynchronous simulation breaks the bounded rationality principle. It allows agents to be immediately aware of events not within their neighbourhood and to be aware of these events even before other agents who are within the neighbourhood of the events are aware of them. In fact, delayed information is only properly (and consistently) produced by the synchronous method of updating.

Our analysis suggests that:

1. Asynchronous updating of a system, by interfering with the speed of transmission of information across the simulation space, will produce less periodicity in the system than synchronous updating;

2. High density populations within systems are more likely to enhance the differences in outcomes between synchronous and asynchronous implementations of the system.

Both of these are borne out in the literature (Caron-Lormier et al. 2008, May 1973).

FURTHER WORK

Recently SU algorithms that can handle the full range of behaviours in ABMs have been developed (**?**). This allows SU to be used for complex ABM/ABSS and side by side comparisons to be made between the two approaches to updating (Kehoe 2016). More work needs to be done to see how the choice of updating method effects simulation outcomes and how large (or otherwise) these differences are.

REFERENCES

L. A. Bach*, D. J. Sumpter, J. Alsner, and V. Loeschcke. Spatial evolutionary games of interaction among generic cancer cells. *Journal of Theoretical Medicine*, 5(1):47–58, 2003.

M. Berryman. Review of software platforms for agent based models. Technical report, DTIC Document, 2008.

M. Bezbradica, H. J. Ruskin, and M. Crane. Comparative analysis of asynchronous cellular automata in stochastic pharmaceutical modelling. *Journal of Computational Science*, 5(5):834–840, 2014.

S. Brailsford. DES is alive and kicking. *J Simulation*, 8(1):1–8, Feb 2014. ISSN 1747-7778. URL `http://dx.doi.org/10.1057/jos.2013.13`.

C. Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications*, 295(3):507–525, 2001. URL `http://EconPapers.repec.org/RePEc:eee:phsmap:v:295:y:2001:i:3:p:507-525`.

G. Caron-Lormier, R. W. Humphry, D. A. Bohan, C. Hawes, and P. Thorbek. Asynchronous and synchronous updating in individual-based models. *Ecological Modelling*, 212(3 4): 522 – 527, 2008. ISSN 0304-3800. doi: http://dx.doi.org/10.1016/j.ecolmodel.2007.10.049. URL `http://www.sciencedirect.com/science/article/pii/S0304380007006011`.

D. Cornforth, D. G. Green, and D. Newth. Ordered asynchronous processes in multi-agent systems. *Physica D: Nonlinear Phenomena*, 204(1 2):70 – 82, 2005. ISSN 0167-2789. doi: http://dx.doi.org/10.1016/j.physd.2005.04.005. URL `http://www.sciencedirect.com/science/article/pii/S0167278905001338`.

J. M. Epstein. Agent-based computational models and generative social science. *Complex.*, 4 (5):41–60, May 1999. ISSN 1076-2787. doi: 10.1002/(SICI)1099-0526(199905/06)4:5<41::AID-CPLX9>3.0.CO;2-F. URL `http://dx.doi.org/10.1002/(SICI)1099-0526(199905/06)4:5<41::AID-CPLX9>3.0.CO;2-F`.

J. M. Epstein and R. Axtell. *Growing Artificial Societies: Social Science from the Bottom Up*. The Brookings Institution, Washington, DC, USA, 1996. ISBN 0-262-55025-3.

N. Fatès. A guided tour of asynchronous cellular automata. In J. Kari, M. Kutrib, and A. Malcher, editors, *Automata*, volume 8155 of *Lecture Notes in Computer Science*, pages 15–30. Springer, 2013. ISBN 978-3-642-40866-3.

M. Gardner. Mathematical games: The fantastic combinations of John Conway's new solitaire game "life". *Scientific American*, pages 120–123, Oct. 1970.

C. Grilo and L. Correia. Effects of asynchronism on evolutionary games. *Journal of Theoretical Biology*, 269(1):109 – 122, 2011. ISSN 0022-5193. doi: http://dx.doi.org/10.1016/j.jtbi.2010.10.022. URL `http://www.sciencedirect.com/science/article/pii/S0022519310005564`.

B. A. Huberman and N. S. Glance. Evolutionary games and computer simulations. *Proceedings of the National Academy of Sciences*, 90(16):7716–7718, 1993.

J. Kehoe. Robust reproducibility of agent based models. In *The European Simulation and Modelling Conference*. Inderscience, Oct 2016.

S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan, and G. Balan. Mason: A multiagent simulation environment. *Simulation*, 81(7):517–527, jul 2005. ISSN 0037-5497. doi: 10.1177/0037549705058073.

R. M. R. M. May. *Stability and complexity in model ecosystems*. Monographs in population biology. Princeton, N.J. Princeton University Press, 1973. ISBN 0-691-08125-5.

M. J. North, N. T. Collier, J. Ozik, E. R. Tatara, C. M. Macal, M. Bragen, and P. Sydelko. Complex adaptive systems modeling with repast simphony. *Complex Adaptive Systems Modeling*, 1(1):1–26, 2013.

M. A. Nowak and R. M. May. Evolutionary games and spatial chaos. *Nature*, 359(6398):826–829, Oct. 1992. doi: 10.1038/359826a0.

B. S. Onggo. Running agent-based models on a discrete-event simulator. In *Proceedings of the 24th European Simulation and Modelling Conference*, pages 25–27. Citeseer, 2010.

S. Railsback, S. Lytinen, and V. Grimm. Stupidmodel and extensions: A template and teaching tool for agent-based modeling platforms, 2005.

G. D. Ruxton and L. A. Saravia. The need for biological realism in the updating of cellular automata models. *Ecological Modelling*, 107(2 3):105 – 112, 1998. ISSN 0304-3800. doi: http://dx.doi.org/10.1016/S0304-3800(97)00179-8.

B. Schönfisch and A. de Roos. Synchronous and asynchronous updating in cellular automata. *Biosystems*, 51(3):123 – 143, 1999. ISSN 0303-2647. doi: http://dx.doi.org/10.1016/S0303-2647(99)00025-8. URL `http://www.sciencedirect.com/science/article/pii/S0303264799000258`.

P.-O. Siebers, C. M. Macal, J. Garnett, D. Buxton, and M. Pidd. Discrete-event simulation is dead, long live agent-based simulation! *J. Simulation*, 4(3):204–210, 2010.