Comparative Analysis of Asynchronous Cellular Automata in Stochastic Pharmaceutical Modelling

Marija Bezbradica, Heather J. Ruskin, Martin Crane

Centre for Scientific Research and Complex Systems Modelling (Sci-Sym), School of Computing, Dublin City University, Dublin, Ireland

Abstract

In pharmaceutical modelling, cellular automata have been used as an established tool to represent molecular changes through discrete structural interactions. The data quality provided by such modelling is found suitable for the early drug design phase where flexibility is paramount. While both synchronous (CA) and asynchronous (ACA) types of automata have been used, analysis of their nature and comparative influence on model outputs is lacking. In this paper, we outline a representative probabilistic CA for modelling complex controlled drug formulations and investigate its transition from synchronous to asynchronous update algorithms. The key investigation points include quantification of model dynamics through three distinct scenarios, parallelisation performance and the ability to describe different release phenomena, namely erosion, diffusion and swelling. The choice of the appropriate update mechanism impacts the perceived realism of the simulation as well as the applicability of large-scale simulations.

Keywords: complex systems, controlled drug delivery, high-performance computing, swellable devices, probabilistic models

1. Introduction

Probabilistic models based on Monte Carlo and CA frameworks have emerged in recent years as a viable response to the modelling needs imposed by design requirements of next generation drug delivery systems (DDS) [10]. In research papers covering the application of CA to the field, both synchronous and asynchronous update methods have been used [15, 11],

without going into deeper analysis of advantages and disadvantages of each. Choosing the algorithm for iterating through the cellular matrix affects how temporal realism of the physical process is represented. As DDS is biological in nature, chaotic or random updates might represent the system dynamics better than synchronous, "all-at-once", changes. On the other hand, as size and complexity of the models grow, the need for efficient parallelisation of model space restricts the application of asynchronous methods due to performance reasons [5]. The transition of CA to ACA in general has been investigated in literature in a number of modelling contexts [5, 1].

In this paper we compare behavioural characteristics, model outputs and performance for different CA and ACA update mechanisms in the context of probabilistic models used in controlled drug delivery and their parallel implementation, where differential equations are not applicable due to inherent unknowns in the parameter space. Finally, we analyse the results obtained by running the models for a specific case of *coated drug bead formulations* and test it for three distinct scenarios.

We provide here an extension of the work, initially presented in [7]. In what follows, Section 2 presents the design methodology used for developing the CA rule sets, together with comparison of different CA and ACA update mechanisms when used in the context of the model and gives a theoretical analysis of their properties and variations in parallel and sequential implementations. Section 3 describes the developed model, with analysis of obtained results in Section 4, followed by the final discussion, (Section 5).

2. Methodology

2.1. Update Algorithms

As for any model build, the first stage involves transfer of domain knowledge of structural and behavioural characteristics of the DDS to the CA model. There are several distinct DDS characteristic categories to be considered, including shape and geometry, polymer composition and interactions, drug loading and influence of the dissolution environment. The models obtained are classified as *kinetic* CA or ACA models [3]. Based on the way we choose to represent the physical phenomena modelled, we adopt rules, either deterministic or probabilistic, (or a combination of both) affecting individual cell behaviour and the surrounding neighbourhood.

To satisfy the condition that the models need to mimic the non-homogeneity of the physical device, with exact distributions of polymer properties not

usually available from experimental data, the model initialises cell states using stochastic distributions within the device geometry. Therefore, various direct Monte Carlo algorithms provide a natural solution to the initial condition problem. Here we address the choice of the appropriate cellular automaton update method, with particular emphasis on the CA rule correctness as we consider several standard synchronous and asynchronous algorithms [3].

Mathematically, the principal features of the 3-dimensional DDS models can be represented as a cellular automaton by a tuple representation:

$$\{G, A, U, \Theta, F\} \tag{1}$$

where $G \subset Z^3$ denotes a set of cell coordinates (the model matrix) and A is the model alphabet - a finite set of possible cell states, (aggregate polymer states), and U denotes the cell neighbourhood (including the cell itself). Then *A(U(x), t)* denotes the state of a neighbourhood of cells U around a given cell $x \in G$ at a moment in time t. The behaviour of the system is described by a set of elementary transition rules, F, where these are applied to the states of a neighbourhood of cells U. For the sequential (i.e. non-parallel) case of both synchronous and asynchronous updates, the general form of the rules (F_s) can be written as following:

$$F_s = \{f(x,t) : A(U(x),t) \rightarrow A(U(x),t+1) \mid x \in G\} \tag{2}$$

with f indicating the specific rule. Finally, Θ denotes the CA/ACA update order function, applied to G and A in order to advance the global model state. As a basis for Θ we investigated the application of several random and ordered asynchronous update methods, (see e.g. [8]) and compared these to the well-used synchronous method.

We implement the different update forms as modifications of the basic **synchronous CA two-phase update algorithm** of the main matrix G:

- **Random order algorithm** involves updating cells of G in a random order which is changed every time a full cell sweep is finished. All cells are updated in each time step of the simulation.

- **Random cyclic algorithm** is a variation of the random order algorithm, with the difference that a single random permutation of G is always used. Random permutation of G is chosen at the beginning of the simulation.

- **Random independent** method chooses one cell at random for updating at each time step. In the overall simulation, each cell should thus be updated approximately the same amount of times. However, over shorter time periods a given cell may be updated much more frequently. To achieve uniform selection, the algorithm thus depends heavily on the size of the sequence of the random number generator. The *Mersenne Twister* algorithm has been used in this case, to reduce bias [12].

- **Fixed cyclic sequential algorithm** is used in its first form where cells of G are visited in sequential order of their coordinates, (first width, then depth, then height in 3D).

2.2. Equivalence of Sequential and Parallel Implementations

In the representative model the mechanisms described above only apply as long as the simulation is *sequential*. Once the algorithm has to scale up to be applicable to large data sets, the inclusion of parallelisation will have fundamental impact on the update logic. It can be shown that in our case synchronous matrix updates are more suitable to parallelisation, as the effect of parallel updates on the resulting state should be equivalent. Consider a parallel version (F_p) of the fundamental rule set given in equation 2:

$$F_p = \{f(x_1, ..., x_k, t) : \bigcup_{i=1}^{k} A(U(x_i), t) \rightarrow \bigcup_{i=1}^{k} A(U(x_i), t{+}1) | x_1, ...x_n \in G\} \quad (3)$$

Essentially, parallelising the update mechanism by splitting the CA space into disjoint domains, each having a set of boundary cells, introduces a simultaneous update of k cells at a time, where k represents the degree of parallelisation. The exact selection of cells x_1, ... , x_k depends on the particular parallel algorithm being used. In the synchronous case, the state of a neighbourhood of cells $U(x)$ at moment t only depends on the same state for the previous moment $t-1$, and not on any currently updated state of any of the other neighbourhoods. Therefore, for synchronous updates, it holds that $F_S \Leftrightarrow F_P$, which is in line with [4].

For asynchronous updates, the equivalence of sequential and parallel implementation breaks down. As the parallel version of the rule set presents a composition of functions applied simultaneously, the order of their application can result in a different overall state of the matrix. This is always true if any of the chosen neighbourhoods $U(x_i)$ overlap.

When implementing parallelisation of pharmaceutical models using some of the industry standard APIs, such as Message Passing Interface (MPI), for course grained parallelisation, or OpenMP, for a more fine grained one, synchronous updates are preferable from the execution speed point of view, as simulations have a practical wall-time limit of 24hrs, the amount of time it would take to run a single *in vitro* experiment. Synchronous updates are extremely efficient in terms of execution speed especially as they can utilise two-sided communication using MPI "send" and "receive" primitives. Asynchronous parallelisation schemes have to use one-sided communication primitives such as MPI "put" and "get", utilising the remote memory access mechanism, which, although slower, allows for the cell state to be asked for or provided on demand, without the need to wait on some eventual update [16].

Finally, it is important to note that according to [17], for relatively slow changing stochastic CA models, the expected variance in outputs between synchronous and asynchronous update methods would be small. This results from the fact that large-scale, low-probability models do not have too many cell state updates in each iteration, which in turn limits the number of cases where overlapping neighbourhoods are updated.

3. CA Model for Coated Drug Formulations

Following the notation from Section 2.1, each of the transition functions is applied to an alphabet of CA states:

$$A = \{P_{COAT}, P_{CORE}, PW_{COAT}, PW_{CORE}, B, D\} \tag{4}$$

where P_{COAT}, PW_{COAT}, P_{CORE} and PW_{CORE} denote the coating layers and core polymers, and their wetted state, respectively. B represents buffer (the solvent) cells and D drug molecules (the solute). Table 1 outlines the behavioural characteristics of each state. The rules affecting each cell type can be described using a formal notation:

- **Erosion:** Polymer lifetime of a given cell x ($l(x)$) decreases linearly with time according to the following function: $f_e(x) : \{l(x) \rightarrow l(x) - t \mid \forall x, A(x) \in \{P_{COAT}, PW_{COAT}, P_{CORE}, PW_{CORE}\}, l(x) \in R^+\}$
- **Diffusion:** The amount of drug present in cell x_a ($d(x_a)$) partially transitions to a neighbouring cell x_b with probability p_{diff}: $f_{diff}(x_a, x_b) : \{d(x_a, x_b) \xrightarrow{p_{diff}} d(x_a) - \Delta d, d(x_b) + \Delta d \mid \forall x_a, A(x_a) \in \{D\}, \forall x_b, A(x_b) \in U(x_a), d(x) \in N^+, \Delta d \in N^+\}$

Table 1: Cell types and rules of behaviour for the examined model

Cell type	Behaviour description
Buffer (B)	Acts as a perfect sink for drug dissolution; Rules: diffusion; dissolution.
Coating polymer (P_{COAT})	Protective coating layer. Upon water penetration erodes into (PW_{COAT}); Rules: erosion; Initial state: assigned random lifetime using Erlang distribution.
Core polymer (P_{CORE})	Binds drug in the solid phase. Upon water penetration erodes into (PW_{CORE}); Rules: erosion.
Wet coating polymer (PW_{COAT})	Coating layer with some water penetration through the polymeric chains, allowing drug to diffuse. Applicable rules: diffusion.
Wet core polymer (PW_{CORE})	Result of core erosion allowing drug diffusion through relaxed chains; Rules: erosion; diffusion; swelling.
Drug packet (D)	Agent, initially dispersed in core polymer cells. Each cell can hold a maximum (saturation) amount of drug "packets". Initial distribution of packets throughout the sphere is determined using MC methods.

- **Swelling:** The amount of polymer present in cell is distributed in a similar fashion, using probability p_s: $f_s(x_a, x_b) : \{l(x_a, x_b) \xrightarrow{p_s} l(x_a) - \Delta l, l(x_b) + \Delta l \mid \forall x_a, A(x_a) \in \{PW_{CORE}\}, \forall x_b \in U(x_a), A(x_b) \in \{P_{COAT}, P_{CORE}, B\}, l(x) \in R^+, \Delta l \in R^+\}$
- **Dissolution:** Finally, the process of partial or total drug dissolution is described as the reduction in drug molecule count of a given cell once it transitions to solvent state: $f_{diss}(x) : \{d(x) \xrightarrow{p_d} d(x) - n \mid \forall x, A(x) \in \{B\}, n \leq d(x), d(x) \in N^+, n \in N^+\}$

Multiple rule combinations can be superimposed (e.g. $f(x) = f_e(f_{diff}(f_s(x)))$) to fully define a cell behaviour during a single iteration if the given cell state satisfies all the alphabet preconditions of the rules given in Equation 4.

3.1. Analytical measures of release and model dynamics

The dynamics of the resulting structural changes are primarily observed using model visualisation, which will highlight differences in key moments.

However, in addition to this it is of interest to examine a more formalised analytical measure of the impact that different update algorithms have on model dynamics over time. When it comes to measuring these in CA systems, several key indicators have been used, such as Hamming distance or Lyapunov exponents [2]. In a system of this complexity, where the cell state transitions are not directly caused by isolated changes in the neighbourhood, but instead follow probabilistic dependencies on the neighbourhood as a whole, as well as having internal state degradation, the former metric was considered to be more readily applicable.

We calculate the *Hamming distance* as the total number of symbols from A different across the entire model space between current and initial time ($\langle H \rangle(t) = \sum a(x), x \in G$, where $a(x) = 1$ if $A(x,t) \neq A(x,0)$, 0 otherwise). We disregard the internal changes of the cell not directly caused by the change in neighbourhood, such as state degradation over time as these proceed at the same rate irrespective of the update algorithm used, and cause a corresponding symbol change in roughly the same time interval (depending on whether the cell was visited in every iteration). Thus, the number of symbol changes over time provide a plausible overview of the model dynamics.

For describing relationship between dominant release kinetics within the modelled device, we make use of the *Peppas power-law* [14]. The Peppas n-factor is used to describe the drug release mechanism as predominately erosion or swelling controlled. It has become the standard method of analysis of any pharmaceutical device, used to estimate the linearity of release for different drug fractions. Even though it cannot fully explain all swellable systems of interest, it has proved very useful in the investigation of complex formulations in the absence of adequate experimental data [11]. Information about the appropriate value of n is needed for adequate classification of a given system.

Values of n are obtained from resulting simulated release curves, using linear regression methods, (the slope of the log-log line, of cumulative release against elapsed time), for the period during which the majority of the drug dissolution occurs.

4. Experimental Results

For each of the described update mechanisms, simulations investigated the following:

- The shape of the release curve during a 24hr period

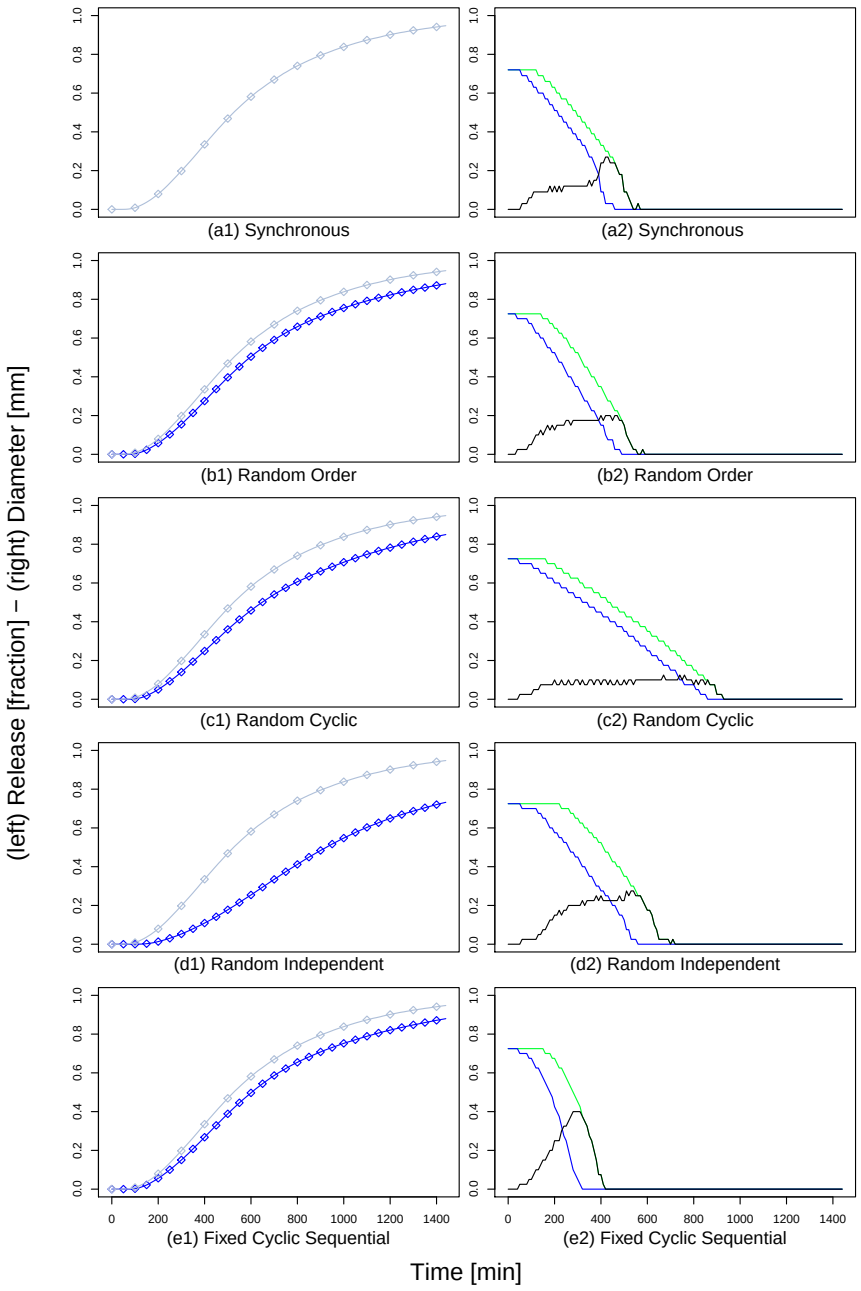

Figure 1: Simulated 24hr period for different update methods applied. **Left** - Drug release curves (blue - release fraction for appropriate update mechanism, gray - synchronous reference); **Right** - dissolution front changes over time (green - swelling front, blue - erosion front, black - gel layer thickness)

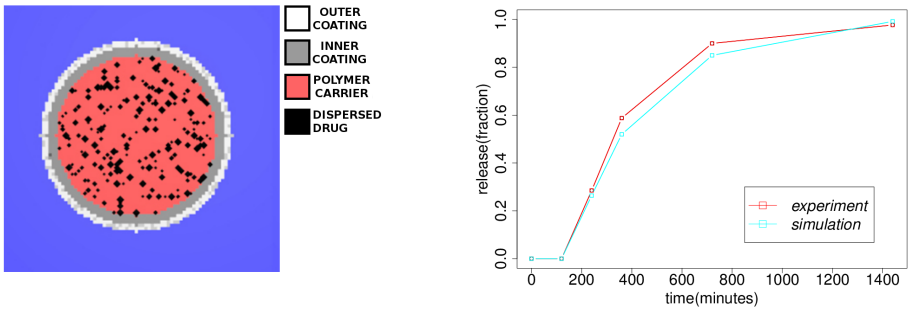

Figure 2: **Left** - modelled device schematics; **Right** - An example experimental vs. simulated results for the case of synchronous updates (experimental data provided by Sigmoid Pharma Ltd.)

- The radii of two main reaction fronts: the swelling and the erosion front
- Visualisation of the device composite structure changes, along with its numerical quantification using Hamming distances
- Key release indicators in three characteristic release scenarios.

In Figure 1 we show the results for *synchronous* updates, (used as a basis for relative comparison with all subsequent ACA methods). The presented data are considered stable, as variations between different runs of the same parameter set were negligible. Comparing with *random order updates*, (Figure 1b), we find a good match, with negligible release curve difference, (indicating that the methods are effectively interchangeable e.g. for the case where synchronous update is deemed more appropriate, (for specified structure [6]). By comparing the curves analytically using the f_2 factor, defined in [13] and commonly used to establish the similarity of two drug dissolution profiles (with $f_2 \in (50, 100)$ representing difference of $\leq 10\%$), we obtain results ranging from 50.93 (10% fit) for random independent to 77.83 (3% fit) for random order type of updates. However, random order, random cyclic and fixed cyclic independent have very similar f_2 values (3%-4%) so considering release curves alone is not enough to establish a clear advantage of one over the other.

Figure 1 shows possible alternatives to the asynchronous *random order method*. *Random independent* selection (Figure 1d) produces shifted release

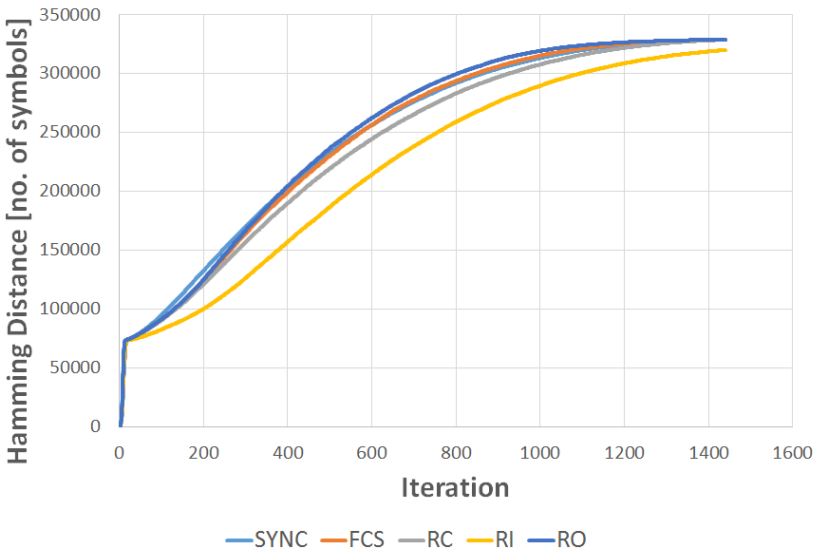

Figure 3: Speed of symbol changes over time (Hamming distance).

curves, although the radii behaviour is similar, in the sense that polymer transitions occur at the same rate. The features which give rise to this discrepancy can be observed in (Figure 4d), where large *drug clusters* (black diamonds) occur as a consequence of some cells being updated more often than others. *Random cyclic updating*, on the other hand, produces release curves which are qualitatively similar to those expected, (Figures 1(c1), 1(c2) and Figure 4(c)), although radii decrease dynamics are much slower. Finally, Figures 1(e1), 1(e2) and Figure 4(e), show results obtained using sequential matrix sweeps. This approach is not recommended due to the significant *bias*, which can be observed in the visualisation, leading to highly unrealistic radii dynamics.

Figure 3 shows the *Hamming distance* for different asynchronous methods and their comparison with synchronous updates. The first section of the curve represents the speed of decay of the outer coating, which as noted in Section 3.1, generally occurs at the same speed irrespective of the method. The second, larger, portion of the curve shows dynamics primarily caused by the decay of the core. In line with the release results presented in Figure 1, the asynchronous mechanisms are displaying slower release dynamics in greater or lesser extent. While random order is again the most favourable, fixed cyclic and random cyclic algorithms show close matches as well. The

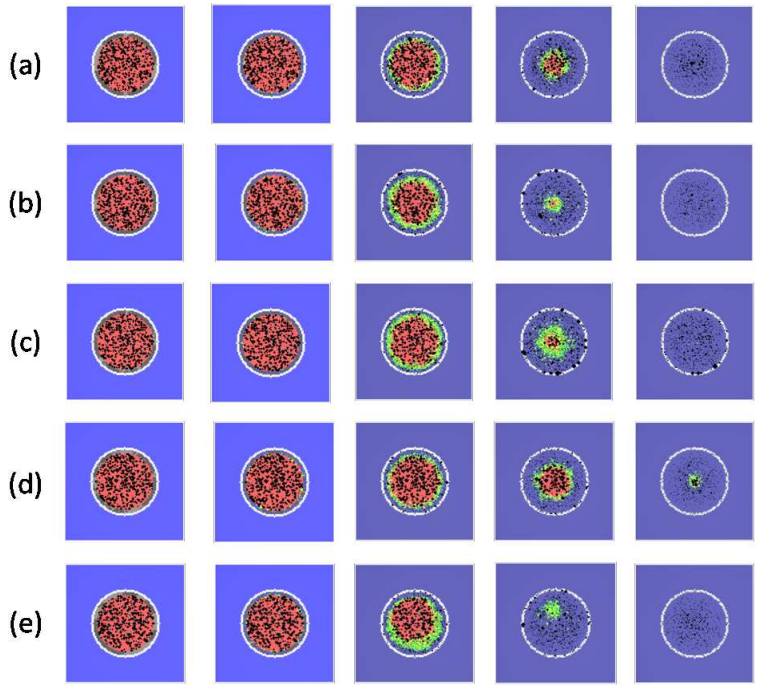

Figure 4: Model visualisation during 10, 30, 150, 400 and 700 minute interval, respectively: (a) synchronous; (b) random order; (c) random cyclic; (d) random independent; (e) fixed cyclic sequential.

outlier is the random independent method, discussed previously, which may not traverse all cells depending on the random sequence generated, and thus tends to model a slower rate of change. Interesting to note is that all update mechanisms show qualitatively the same profile of structural changes, with only relative speed being affected. Not unexpectedly, since the stochastic model itself changes slowly, with usual probability values used much smaller than 1, and changes being highly symmetrical, due to the spherical device geometry. Results obtained are less anomalous than would be expected in general CA to ACA transition [9], (in agreement with theory, [17]). The validity of the synchronous updates when compared against experimental data is shown in Figure 2, with the similarity factor falling within the standard variability range ($< 6\%$).

Next, we examine the overall simulation length using different ACA mechanisms in a thread-level parallelisation context. The best results are obtained for random cyclic variants, an optimal choice when best simulation perfor-

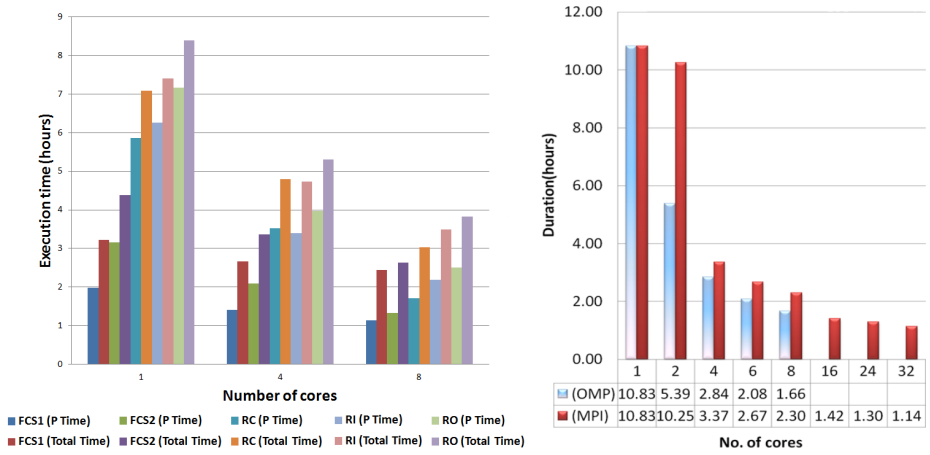

Figure 5: Comparison of parallel and total simulation times for different synchronous and asynchronous update mechanisms. **Left** - comparison of different ACA update methods using thread-level parallelism. **Right** - comparison of synchronous update mechanism for thread-level (blue) vs. process-level (red).

mance is desired, as opposed to preservation of state update realism, (see earlier comments). The poorest performance is that of random order algorithms which are not able to leverage the processor cache due to constantly changing order of memory access. However, these offer the best simulation realism with respect to process represented, so the advantages and disadvantages should be weighed in light of simulation objectives when making a decision. Figure 5 shows an example performance profile for synchronous updates when switching from a thread-level to a process-level parallelisation model [6]. Although synchronous updates do not perform at the same level as asynchronous ones, the former do not have a parallelisation limit, and thus, ultimately, can be scaled to any number of nodes allowed by the model size.

Finally, as the initial dataset presented an image of the model performance for an isolated scenario, we also tested the adequacy of different asynchronous methods in describing three distinct release scenarios: (i) erosion dominated release; (ii) swelling dominated release and (iii) equilibrium between the two - which should lead to a constant gel layer determining the shape of the release profile. We compare n and f_2 factors across all methods (Figure 6). Figure 6 (left panel) shows side-by-side comparison of n for synchronous, fixed and random cyclic, random independent and random order updates. We observe that asynchronous variants exhibit varied dynamics across a more erosion-driven range ($n > 0.85$). For devices with constant gel layer, random

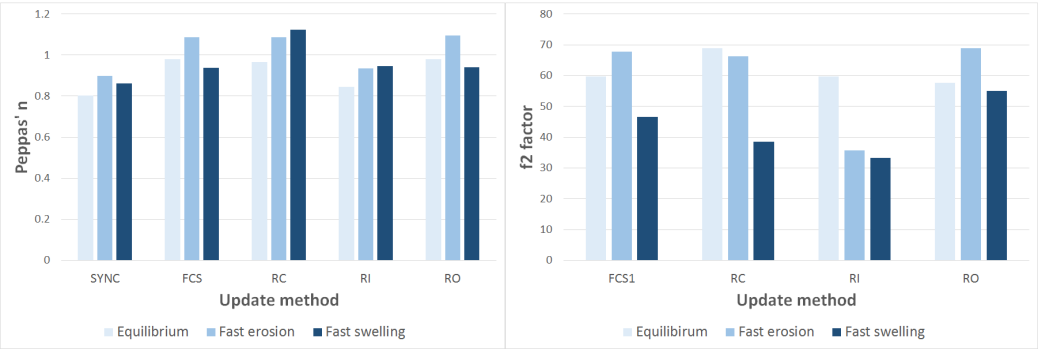

Figure 6: Comparison of release dynamics for different update methods (Peppas' n factor), (**left**), and for different dissolution dynamics (f_2 factor - higher is better), (**right**)

independent and random order provide the closest match to synchronous updates, in line with visual observations from Figure 2. Results are similar for highly swellable devices, while performance of random order updates is somewhat poorer for predominantly erodible ones. This highlights the fact that erosion is expected to be faster as the single-iteration random updates across the matrix tend to exacerbate the effect of rules which deal with decay in polymer state, as transitions to wet states are immediate.

The f_2 factor was again employed to assess the quantitative similarity between the release curves obtained for each scenario. Looking at Figure 6 (right panel), we can observe that, while scenarios with constant gel layer thickness are adequately described by all methods, scenarios with more extreme behaviour give rise to much wider variability in the resulting release curves.

5. Conclusions

Advantages and disadvantages of CA and ACA update methods, important for modelling of DDS were investigated using different metrics. While all methods have merit, findings show that performance can be affected by update rules, with one of the most robust solution found for *random order asynchronous*. Model visualisation, augmented by analytical measures of the change dynamics, provided valuable insight on structural behaviour and dissolution mechanisms, which is not readily apparent from working with standard release curve data alone. We tested the applicability of ACA for three extreme cases. The findings are useful for future modelling scenarios

both where it may be necessary to switch from one update mechanism to another for models describing a wide variety of drug-polymer interactions, as well as to accommodate large-scale optimisation. The CA pharmaceutical models presented here mark some progress toward these goals.

Acknowledgments.

Financial support from the ERA-Net Complexity Project, P07217, is warmly acknowledged.

References

[1] E. Alba, M. Giacobini, M. Tomassini, S. Romero: Comparing synchronous and asynchronous cellular genetic algorithms. In: J.J.M. Guervós et al. (Eds.) PPSN VII. LNCS, vol. 2439, Springer, Heidelberg pp. 601–610 (2002)

[2] J.M. Baetens, B. Baets: A Lyapunov View on the Stability of Two-State Cellular Automata. In: H. Zenil (Ed.) Irreducibility and Computational Equivalence, vol. 2, Springer, Heidelberg pp. 25–33 (2013)

[3] S. Bandini, A. Bonomi, G. Vizzari: What do we mean by asynchronous CA? A reflection on types and effects of asynchronicity. In: S. Bandini et al. (Eds.) ACRI 2010. Springer, Heidelberg LNCS, vol. 6350, pp. 385–394 (2010)

[4] O. Bandman: Parallel simulation of asynchronous cellular automata evolution. In: S. El Yacoubi et al. (Eds.) ACRI 2006. LNCS, vol. 4173, Springer, Heidelberg pp. 41–47 (2006)

[5] O. Bandman: Synchronous versus asynchronous cellular automata for simulating nano-systems kinetics. Bulletin of the Novosibirsk Computer Center, Computer Science vol. 25 pp. 1–12 (2006)

[6] M. Bezbradica, M. Crane, H.J. Ruskin: Parallelisation strategies for large scale cellular automata frameworks in pharmaceutical modelling. In: High Performance Computing and Simulation (HPCS), 2012 International Conference on. pp. 223–230 (2012)

[7] M. Bezbradica, H.J. Ruskin, M. Crane: Probabilistic Pharmaceutical Modelling: A Comparison Between Synchronous and Asynchronous

Cellular Automata, To appear in LNCS, 10th International Conference on Parallel Processing and Applied Mathematics (PPAM) (2013)

[8] D. Cornforth, D.G. Green, D. Newth: Ordered asynchronous processes in multi-agent systems. Physica D 204(1–2) pp. 70–82 (2005)

[9] N. Fatès, E. Thierry, M. Morvan, N. Schabanel: Fully asynchronous behavior of double-quiescent elementary cellular automata. Theoretical Computer Science 362(1–3) pp. 1–16 (2006)

[10] N. Haddish-Berhane, S.H. Jeong, K. Haghighi, K. Park: Modeling film-coat non-uniformity in polymer coated pellets: A stochastic approach. Int. J. Pharm. 323(1-2) pp. 64–71 (2006)

[11] H. Laaksonen, J. Hirvonen, T. Laaksonen: Cellular automata model for swelling-controlled drug release. Int. J. Pharm. 380(1–2) pp. 25–32 (2009)

[12] M. Matsumoto, T. Nishimura: Mersenne Twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. 8 pp. 3–30 (1998)

[13] J.W. Moore, H.H. Flanner: Mathematical comparison of curves with an emphasis on *in-vitro* dissolution profiles. Pharm. Tech. 20(6) pp. 64–74 (1996)

[14] P.L. Ritger, N.A. Peppas: A simple equation for description of solute release I. Fickian and non-Fickian release from non-swellable devices in the form of slabs, spheres, cylinders or discs. J. Control. Rel. 5(1) pp. 23–36 (1987)

[15] J. Siepmann, F. Siepmann: Mathematical modeling of drug delivery. Int. J. Pharm. 364(2) pp. 328–343 (2008)

[16] R. Thakur, W. Gropp, B. Toonen: Minimizing synchronization overhead in the implementation of MPI one-sided communication. In D. Kranzlmüller et al. (Eds.): EuroPVM/MPI. LNCS, vol. 3241, Springer, Heidelberg pp. 57–67 (2004)

[17] T. Toffoli, N. Margolus: Cellular Automata Machines: A New Environment for Modeling. MIT Press, Cambridge MA (1987)