Probabilistic Pharmaceutical Modelling: A Comparison Between Synchronous and Asynchronous Cellular Automata

Marija Bezbradica(✉), Heather J. Ruskin, and Martin Crane

School of Computing, Centre for Scientific Research and Complex Systems Modelling (Sci-Sym), Dublin City University, Dublin, Ireland
{mbezbradica,hruskin,mcrane}@computing.dcu.ie

Abstract. The field of pharmaceutical modelling has, in recent years, benefited from using probabilistic methods based on cellular automata, which seek to overcome some of the limitations of differential equation based models. By modelling discrete structural element interactions instead, these are able to provide data quality adequate for the early design phases in drug modelling. In relevant literature, both synchronous (CA) and asynchronous (ACA) types of automata have been used, without analysing their comparative impact on the model outputs. In this paper, we compare several variations of probabilistic CA and ACA update algorithms for building models of complex systems used in controlled drug delivery, analysing the advantages and disadvantages related to different modelling scenarios. Choosing the appropriate update mechanism, besides having an impact on the perceived realism of the simulation, also has practical benefits on the applicability of different model parallelisation algorithms and their performance when used in large-scale simulation contexts.

Keywords: Discrete systems · Controlled drug delivery systems · Complex modelling · Parallel algorithms

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 novel, more complex, drug delivery systems (DDS) [1].

Unlike traditional, differential equation based models, [2,3], CA attempt to recreate system-level behaviour by *in silico* simulations of individual interactions within the modelled device. This fits naturally with the early stages of the design process, in which global physico-chemical behaviours of DDS are investigated. By providing a low-cost alternative to lengthy, and potentially expensive, *in vitro* experiments, probabilistic computational modelling becomes an integral part of the drug design process. Nevertheless, uncertainties are inherent in this approach to modelling physical phenomena and parameters can multiply rapidly, due to

R. Wyrzykowski et al. (Eds.): PPAM 2013, Part II, LNCS 8385, pp. 699–710, 2014.
DOI: 10.1007/978-3-642-55195-6_66,

the many different physical interactions within the model, so good understanding of model design choices is crucial.

In research papers covering the application of CA to the field, both synchronous and asynchronous update methods have been used [3–6], without going into deeper analysis of pros and cons of each. Choosing the algorithm for iterating through the cellular matrix and the order of application of the local rules 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 grows, the need for efficient parallelisation of model space restricts the application of the asynchronous methods due to performance reasons [7]. Therefore, it is of importance to understand the effects of synchronicity and asynchronicity to the model outputs, in order to be able to make optimal choices during model development. The transition of CA to ACA in general has been investigated in literature in a number of modelling contexts [7–9].

In this paper we compare behavioural characteristics, model outputs and performance for different synchronous and asynchronous CA 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 resulting models for a specific case of *coated drug bead formulations* [10].

In what follows, Sect. 2 presents the design methodology used for developing the CA rule sets, along 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 Sect. 4, followed by the final discussion (Sect. 5).

2 CA and ACA Modelling

2.1 Design Methodology

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:

- The shape and geometry of the system (slab, cylindrical or spherical) - captured in the shape and size of the model cellular matrix;
- Polymer composition of the device defined by states of the matrix cells;
- Polymer physico-chemical interaction mechanisms (laws) - described by characteristic behaviours that occur inside the DDS;
- Drug loading and initial dispersion within the device;
- Influence of the dissolution environment on polymer behaviour.

The models thus obtained, with the above characteristics, are classified as *kinetic* CA or ACA models [11]. Based on the way we choose to represent the physical phenomena modelled, we can adopt rules, either deterministic or probabilistic, (or a combination of both) affecting individual cell behaviour and the surrounding neighbourhood.

Although it is common for various families of CA update mechanisms to operate under periodic boundary conditions [12,13], pharmaceutical models benefit from using the fixed equivalent, as drug movement across the matrix boundaries is used as a direct method for calculating release rates. To satisfy the condition that the models need to mimic the non-homogeneousness of the physical device, with exact distributions of polymer and drug properties not available from experimental data, the model establishes the initial cell states using stochastic distributions within the known device geometry. Therefore, various direct Monte Carlo algorithms provide a natural solution to the initial condition problem, by establishing a random starting state for each simulation run.

2.2 Update Methods

Crucially, a good description of the model dynamics, i.e. the closest qualitative match to the behaviour of the pharmaceutical system modelled, relies on choosing the appropriate update method of the cellular automaton itself. This is addressed here, with particular emphasis on the correctness of the update rules as we consider several standard synchronous and asynchronous CA update methods [11,14,15].

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

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

where G denotes a set of cell coordinates (the model matrix in our meta model). In the case of a 3D system:

$$G = Z^3 = \{(i, j, k) \mid 1 \leq i \leq N_x, 1 \leq j \leq N_y, 1 \leq k \leq N_z\} \tag{2}$$

A is the model alphabet - a finite set of possible cell states, (aggregate polymer states in the conceptual model), and U denotes the cell neighbourhood (including cell itself). Then *A(U(x), t)* denotes the state of a neighbourhood of cells U around a given cell x at a moment in time t. The behaviour of the system is described by a set of elementary transition rules for the conceptual model, F, where these are applied to the states of a neighbourhood of cells U. For the sequential case of synchronous updates, the general form of the rules (F_s) can be written as following:

$$F_s = \{f(x) : A(U(x), t) \rightarrow A(U(x), t + 1) \mid x \in G\} \tag{3}$$

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. [16]) 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. The first ACA update method investigated is the **random order algorithm** which 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. The **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. In the **random independent** method one cell is chosen 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 significantly more often. To achieve uniform selection, the algorithm thus depends heavily on the size of the sequence of the random number generator implementation. The *Mersenne Twister* algorithm has been used in this case, to reduce bias [17]. Finally, the **fixed cyclic sequential algorithm** was used in two forms: in the first one, cells of G are visited in sequential order of their coordinates (first width, then depth, then height in 3D). In the second form, cells are sorted based on their state (A), so that polymers of certain type are given simulation priority over polymers of other types (outer coating, then the inner coating, then the core). The order of cell simulation within the same polymer type is sorted by its coordinates.

2.3 Equivalence of Sequential and Parallel Implementations

In a concrete, model execution context, the mechanisms presented 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 Eq. 3:

$$F_p = \{f(x_1, \ldots, x_n) : \bigcup_{i=1}^{n} A(U(x_i), t) \rightarrow \bigcup_{i=1}^{n} A(U(x_i), t+1) \,|\, x_1, \ldots, x_n \in G\} \quad (4)$$

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 n cells at a time, where n represents the degree of parallelisation. The exact selection of cells $x_1, \ldots, x_n$ 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 [18].

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.

Therefore, in the case of random order and random cyclic updates, we expect the parallelisation to always result in slightly different model output. The same holds for different variations of *fixed cyclic rules*, where the idea of the underlying algorithm essentially breaks down. The only exception to this rule is the *random independent* order of updates, which might produce equivalent results, but only if, in each individual iteration, cells x_i are chosen in each parallel domain so that their neighbourhoods $U(x_i)$ are non-overlapping.

From an implementation point of view, when implementing parallelisation of pharmaceutical models using some of the industry standard parallel APIs, such as Message Passing Interface (MPI), synchronous updates are preferable from the execution speed point of view, as simulations have a practical wall-time limit of 24 h, 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 to send and receive primitives. Asynchronous parallelisation schemes have to utilise 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 [19].

Finally, it is important to note here that according to [20], 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

Table 1 outlines the main CA rules used in the resulting model for each of the simulated processes. Following the notation from Sect. 2.2, each of the transition functions is applied to an alphabet of CA states:

$$A = \{P_{COAT}, P_{CORE}, PW_{COAT}, PW_{CORE}, B, D\} \tag{5}$$

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 cells and D drug molecules. The rules affecting each cell type can be described using a formal notation:

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.

- **Erosion:** Polymer lifetime of a given cell x ($l(x)$) decreases linearly with time according to the following function: $f_e(x) : \{l(x) \rightarrow l(x) - t \mid \forall x \in \{P_{COAT}, PW_{COAT}, P_{CORE}, PW_{CORE}\}\}$
- **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}: $f_{diff}(x_a, x_b) : \{d(x_a, x_b) \xrightarrow{p_{diff}} d(x_a) - \Delta d, d(x_b) + \Delta d \mid \forall x_a \in \{D\}, x_b \in U(x_a)\}$
- **Swelling:** The amount of polymer present in cell is distributed in a similar fashion, using probability p_s: $f_s(x_a, x_b) : \{l(x_a, x_b) \xrightarrow{p_s} l(x_a) - \Delta l, l(x_b) + \Delta l \mid \forall x_a \in \{PW_{CORE}\}, x_b \in U(x_a), x_b \in \{P_{COAT}, P_{CORE}, B\}\}$
- **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: $f_{diss}(x) : \{d(x) \xrightarrow{p_d} d(x) - n \mid \forall x \in \{B\}, n \leq d(x)\}$

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 Eq. 5.

4 Experimental Results

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

- The shape of the release curve during a 24 h period (a characteristic of GI tract transition time for the drug);

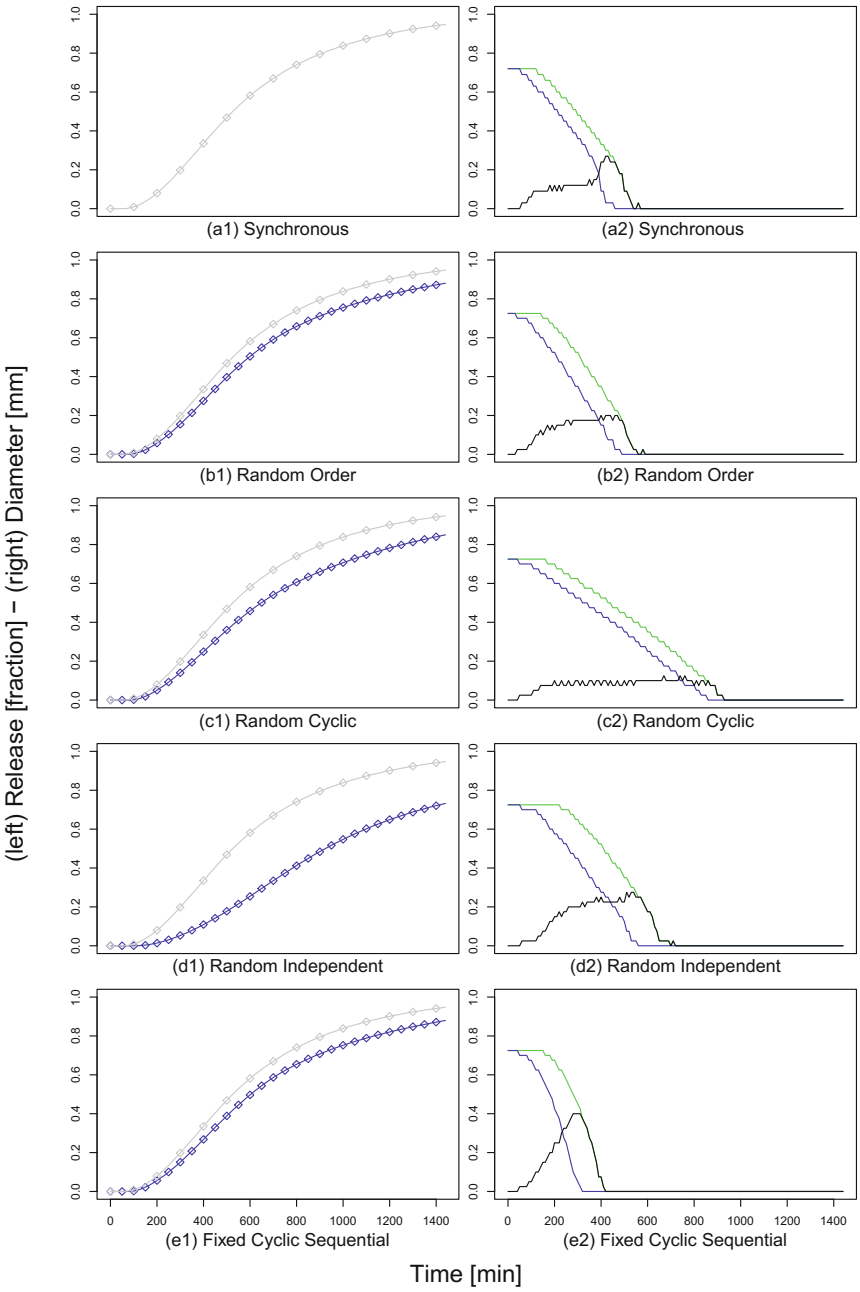

Fig. 1. 24 h simulation period for different update methods. **Left** - Drug release curves (blue - release fraction for appropriate update mechanism, gray - synchronous reference); **Right** - dissolution fronts changes over time (green swelling front, blue - erosion front, black gel layer thickness) (Color figure online)

- The radii of two main reaction fronts: the swelling front, (where the polymer moves from being in a static to dynamic state), and the erosion front (where polymer starts dissolving). These are the best indicators of the spatial scope of the underlying phenomena, which cannot be observed directly from release data alone;
- The device composite structure changes during characteristic stages of the dissolution, through visualisation of details.

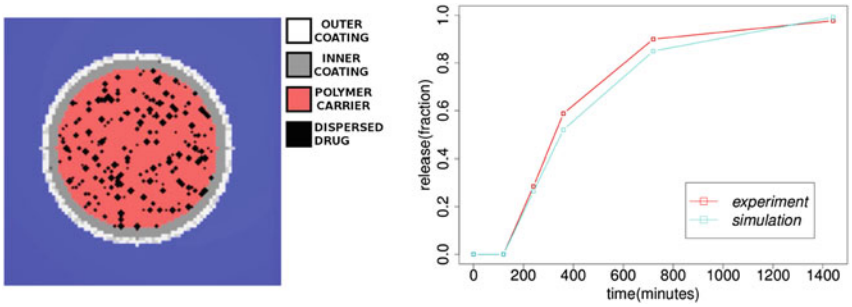

Fig. 2. Left- modelled device schematics; **Right** - An example experimental vs. simulated results for the case of synchronous updates (experimental data provided by Sigmoid Pharma Ltd) (Color figure online)

In Fig. 1 we show the results for *synchronous* updates (used as a basis for relative comparison with all subsequent ACA methods). The presented data is considered stable, as variations between different runs of the same parameter set were negligible. Comparing with *random order updates*, (Fig. 1b), we find a good match, with negligible release curve difference (indicating that the methods are effectively interchangeable e.g. where synchronous update is deemed more appropriate (for specified structure for example [21])). By comparing the curves analytically using the f_2 similarity factor [22], we obtain the 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 %) fit so looking at the release curves alone is not enough to establish a clear advantage of one over another.

Figure 1 shows possible alternatives to the asynchronous *random order method*. It can be seen that *random independent* selection (Fig. 1d) produces release curves, which are significantly shifted with respect to those expected, 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 the model visualisations, where large *drug clusters* (black diamonds) occur as a consequence of some cells being updated more often than others (Fig. 3d). *Random cyclic updating*, on the other hand, produces release curves which are qualitatively similar to those expected, (Figs. 1(c1), (c2) and

3(c)), although the radii decrease dynamics are much slower. Finally, Figs. 1(e1), (e2) and 3(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. As expected, since the stochastic model is both slowly changing, with usual probability values used are much smaller than 1, and highly symmetrical, due to the spherical device geometry, the results obtained do not show highly anomalous results as would be expected in general CA to ACA transition [23], in line with theorems presented in [20] relating to equivalences between slow changing synchronous and asynchronous update processes in CA models.

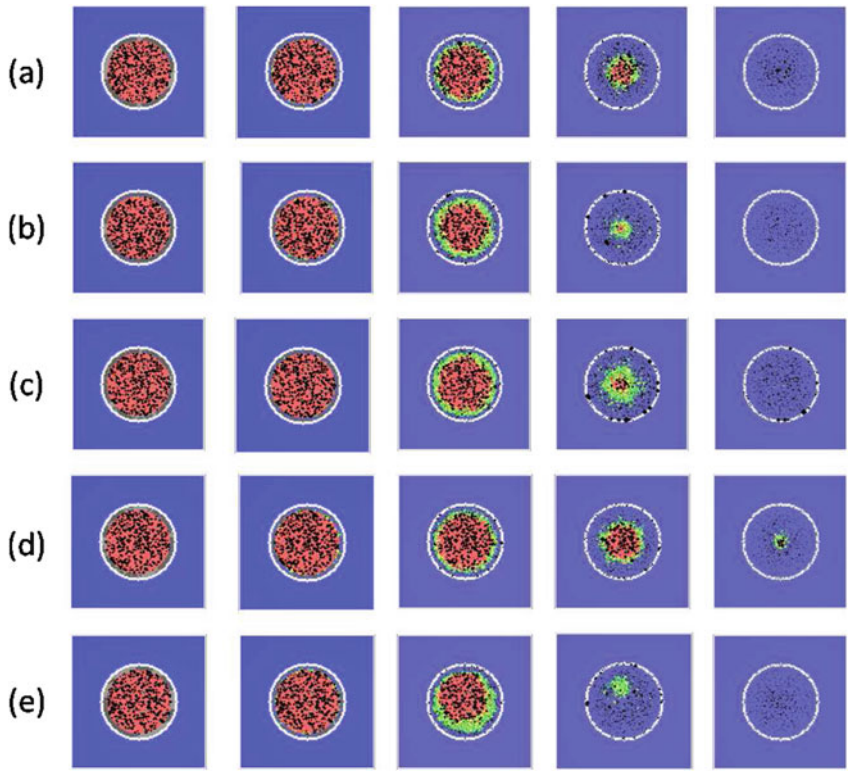

Fig. 3. Model visualisation during 10, 30, 150, 400 and 700 min interval, respectively: (a) synchronous; (b) random order; (c) random cyclic; (d) random independent; (e) fixed cyclic sequential. (Color figure online)

The validity of the synchronous updates when compared against experimental data is shown in Fig. 2, with obtained similarity factor showing a match within the standard variability range (<6 %).

At the end, we examine the overall performance in terms of simulation length using different ACA mechanisms in thread-level parallelisation context.

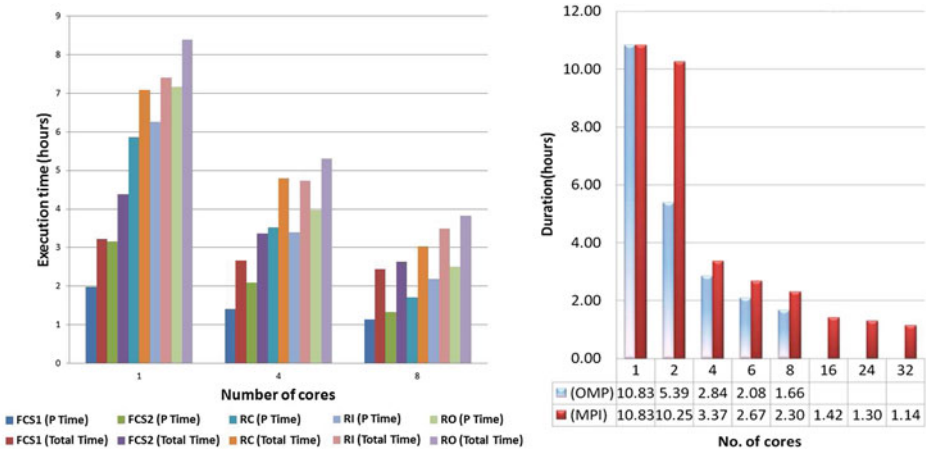

Fig. 4. Comparison of parallel and total simulation times for different synchronous and asynchronous update mechanisms. **Left** - comparison of different ACA update methods using thread-level parallelism. **Right** - comparison of synchronous update mechanism for thread-level (blue) vs. process-level (red) (Color figure online)

As expected, sequential algorithms were the fastest, as these could utilise the CPU memory cache better. The performance is closely followed by random cyclic variants, which might make an optimal choice for the scenario where the best simulation performance is needed, as opposed to state update realism, considering the accuracy of model outputs presented earlier. The worst performing are the 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 and precision when compared to the synchronous variant, so the pros and cons of each should be weighed when making the decision. Figure 4 shows the performance profile for synchronous updates when switching from thread-level to process-level parallelisation model [21]. Although synchronous updates do not perform on the same level as asynchrounous ones, they do not have a parallelisation limit, and thus, ultimately, can be scaled to any number of nodes allowed by the model size.

5 Conclusions and Future Work

We have presented an overview of methodological considerations, important to modelling drug delivery systems using CA and ACA, and have analysed advantages and disadvantages of each update method. While some flexibility is possible in choosing between asynchronous and synchronous methods for approximate solutions in this context, this is governed by structural requirements. Our findings show that one of the most adequate solutions is *random order asynchronous*. In this regard, model visualisation provides valuable additional insight on structural behaviour and dissolution mechanisms, which is not readily apparent from

working with standard release curve data alone, or which are intractable to supplementary experiment. The findings are useful for future modelling scenarios where it may be necessary to switch from one update mechanism to the other, both in terms of large-scale optimisation, but also in response to the need for describing component interactions in tailored solutions for individualised treatment. The CA pharmaceutical models presented here are a step in that direction. Conclusions drawn in this paper can also be applied in general to any slow chaining CA system, such as those used in social behaviour modelling for example (especially the performance part).

Acknowledgments. Financial support from the ERA-Net Complexity Project, P07217, is warmly acknowledged. The parallel simulations discussed in this paper were performed on computational resources provided by Sci-Sym, DCU, and by the Irish Centre for High-End Computing.

References

1. Haddish-Berhane, N., Jeong, S.H., Haghighi, K., Park, K.: Modeling film-coat nonuniformity in polymer coated pellets: a stochastic approach. Int. J. Pharm. **323**(1–2), 64–71 (2006)
2. Kaunisto, E., Marucci, M., Borgquist, P., Axelsson, A.: Mechanistic modelling of drug release from polymer-coated and swelling and dissolving polymer matrix systems. Int. J. Pharm. **418**(1), 54–77 (2011)
3. Siepmann, J., Siepmann, F.: Mathematical modeling of drug delivery. Int. J. Pharm. **364**(2), 328–343 (2008)
4. Barat, A., Crane, M., Ruskin, H.J.: Quantitative multi-agent models for simulating protein release from PLGA bioerodible nano- and microspheres. J. Pharm. Biomed. Anal. **48**(2), 361–368 (2008)
5. Göpferich, A.: Bioerodible implants with programmable drug release. J. Controlled Release **44**(2–3), 271–281 (1997)
6. Laaksonen, H., Hirvonen, J., Laaksonen, T.: Cellular automata model for swelling-controlled drug release. Int. J. Pharm. **380**(1–2), 25–32 (2009)
7. Bandman, O.: Synchronous versus asynchronous cellular automata for simulating nano-systems kinetics. Bull. Novosib. Comput. Cent. Ser. Comput. Sci. **25**, 1–12 (2006)
8. Alba, E., Giacobini, M., Tomassini, M., Romero, S.: Comparing synchronous and asynchronous cellular genetic algorithms. In: Merelo Guervós, J.J., Adamidis, P.A., Beyer, H.-G., Schwefel, H.-P., Fernández-Villacañas, J.L. (eds.) PPSN VII. LNCS, vol. 2439, pp. 601–610. Springer, Heidelberg (2002)
9. Kalgin, K.V.: Parallel simulation of asynchronous cellular automata evolution. Bull. Novosib. Comput. Cent. Ser. Comput. Sci. **27**, 55–62 (2008)
10. Bezbradica, M., Ruskin, H.J., Crane, M.: Modelling drug coatings: a parallel cellular automata model of ethylcellulose-coated microspheres. In: Proceedings of the International Conference on Bioscience, Biochemistry and Bioinformatics (ICBBB 2011), vol. 5, pp. 419–424 (2011)
11. Bandini, S., Bonomi, A., Vizzari, G.: What do we mean by asynchronous CA? A reflection on types and effects of asynchronicity. In: Bandini, S., Manzoni, S., Umeo, H., Vizzari, G. (eds.) ACRI 2010. LNCS, vol. 6350, pp. 385–394. Springer, Heidelberg (2010)

12. Burstedde, C., Klauck, K., Schadschneider, S., Zittartz, J.: Simulation of pedestrian dynamics using a two-dimensional cellular automaton. Physica A **295**(3–4), 507–525 (2001)
13. Xiao, X., Shao, S., Ding, Y., Huang, Z., Chen, X., Chou, K.C.: Using cellular automata to generate image representation for biological sequences. Amino Acids **28**, 29–35 (2005)
14. Baetens, J.M., der Weeën, P.V., Baets, B.D.: Effect of asynchronous updating on the stability of cellular automata. Chaos, Soliton. Fract. **45**(4), 383–394 (2012)
15. Schoenfisch, B., de Roos, A.: Synchronous and asynchronous updating in cellular automata. Biosystems **51**(3), 123–143 (1999)
16. Cornforth, D., Green, D.G., Newth, D.: Ordered asynchronous processes in multi-agent systems. Physica D **204**(1–2), 70–82 (2005)
17. Matsumoto, M., Nishimura, T.: Mersenne Twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. **8**, 3–30 (1998)
18. Bandman, O.: Parallel simulation of asynchronous cellular automata evolution. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) ACRI 2006. LNCS, vol. 4173, pp. 41–47. Springer, Heidelberg (2006)
19. Thakur, R., Gropp, W.D., Toonen, B.: Minimizing synchronization overhead in the implementation of MPI one-sided communication. In: Kranzlmüller, D., Kacsuk, P., Dongarra, J. (eds.) EuroPVM/MPI 2004. LNCS, vol. 3241, pp. 57–67. Springer, Heidelberg (2004)
20. Toffoli, T., Margolus, N.: Cellular Automata Machines: A New Environment for Modeling. MIT Press, Cambridge (1987)
21. Bezbradica, M., Crane, M., Ruskin, H.J.: Parallelisation strategies for large scale cellular automata frameworks in pharmaceutical modelling. In: 2012 International Conference on High Performance Computing and Simulation (HPCS), pp. 223–230 (2012)
22. Moore, J.W., Flanner, H.H.: Mathematical comparison of curves with an emphasis on in-vitro dissolution profiles. Pharm. Technol. **20**(6), 64–74 (1996)
23. Fatès, N., Thierry, E., Morvan, M., Schabanel, N.: Fully asynchronous behavior of double-quiescent elementary cellular automata. Theoret. Comput. Sci. **362**(1–3), 1–16 (2006)