Scientific Programming 10 (2002) 1–19 SPR139
IOS Press

Individual-based simulation of the clustering behaviour of epidermal growth factor receptors

Jacki P. Goldman[a,b], William J. Gullick[b] and Colin G. Johnson[a,*]
[a]*Computing Laboratory, University of Kent, Canterbury, Kent, CT2 7NF, UK*
[b]*Department of Biosciences, University of Kent, Canterbury, Kent, CT2 7NJ, UK*

Abstract. This paper describes ongoing work on a project to simulate the behaviour of *epidermal growth factor receptors*. These are structures which can be found on the surface of cells in the body, which receive and process chemical signals concerned with cell growth. The implementation of a program which simulates the stimulation and clustering behaviour of these structures is described, then the paper discusses how the simulation can be scaled up so that a whole cell can be simulated on a tractable timescale. Finally some some early results are given which show the effect of changing parameters in the system, and discuss ongoing work on calibrating the simulation against results from experiments.

1. Biological background

In order for the body to function, cells need to communicate with each other. One important mechanism for this is the diffusion of chemical signalling molecules (known as *ligands*) in tissues, which are in turn received by *receptors* on the surface of cells [20,28]. Once these signalling molecules have bound to the receptors, they trigger a set of events inside the cell (Fig. 1).

A particular example of a system of this type is the *epidermal growth factors* (EGFs) and their receptors [48]. This consists of four different receptors (one of which can exist in four different forms) which can be bound to by at least ten different ligand types. Once the receptors have bound ligand, they form clusters which stimulate within-cell signalling events concerned with cell growth. Thus the cell can communicate with the extracellular environment without molecules being passed through the cell membrane.

This system is of medical importance, because it can go wrong in a number of ways leading to the formation of cancers. Three mechanisms are known to cause such problems. Firstly a mutation can cause the cell to generate its own growth factor, thus generating unchecked cell growth. Secondly mutations in the genes specifying the receptor can cause it to be in a constantly activated state regardless of the presence or absence of ligand. A third problem occurs when too many receptors are created because of e.g. a problem with the expression of the receptor genes, meaning that the cell exhibits excessive growth for small amounts of signal.

We would like to understand the dynamics of this clustering process better and to understand what causes the clusters to form in a particular way and how they develop with time. The cluster formation can be observed by tagging the receptors with a fluorescent protein and filming the motion under an optical microscope [18]. At first the fluorescence is distributed evenly around the cell, but five minutes after ligand molecules have been introduced clusters can clearly be seen (Fig. 2(a)). Using image processing software clusters can be identified (Fig. 2(b)) and the number of clusters of various sizes counted (Fig. 2(c)). Nonetheless this data provides little direct insight into the processes which are causing this cluster formation. To understand this better we have been developing computer simulations which

[*] Corresponding author. Tel.: +44 1227 827562; Fax: +44 1227 762811; E-mail: C.G.Johnson@ukc.ac.uk.

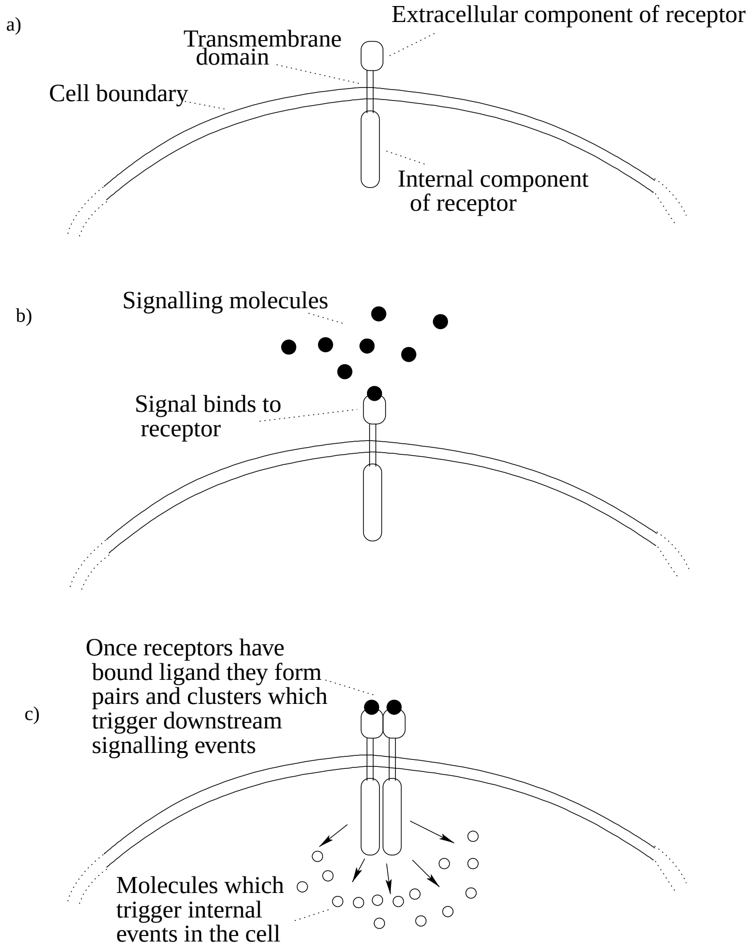
a)
Extracellular component of receptor
Transmembrane domain
Cell boundary
Internal component of receptor
b)
Signalling molecules
Signal binds to receptor
c)
Once receptors have bound ligand they form pairs and clusters which trigger downstream signalling events
Molecules which trigger internal events in the cell

Fig. 1. How cells receive signals using receptors.

will allow us to try out different hypotheses for the behaviour underlying this experimental data, for example by simulating different conjectured mechanisms which give rise to the clustering and adjusting the parameters within those mechanisms.

2. Related work

The most traditional use of computer simulation in cellular biology has been in modelling the stoichiometrics of reactions within the cell [15]. Such approaches have used both programs specifically written to carry out these calculations, such as *Gepasi* [30,31] and SCAMP/*Jarnac* [41,42], as well as general mathematical programs such as Matlab [38] and Maple [21] and numerical DE-solvers. Other work (e.g. [1–3]) has modelled the developmental process in the early life of the cell.

Such modelling techniques cannot be used to understand phenomena which depend on the spatial distribution of individual molecules, where the complete mixing assumption which underlies the above projects cannot be applied. Instead we need to take an approach to simulation which is based on creating interacting models of individual components in the simulation; such models have a long history of success in ecological research [13,29] but have seen fewer applications at the cellular level.

Nonetheless there have been some successful applications of this type of modelling to the analysis of systems such as cell-signalling networks [9,10], the G-protein cascade [26,25] and the microphysiology of synaptic transmission [6].

Other systems have been developed which attempt to simulate the "whole cell". Examples of this are *E-Cell* [47] and the *Virtual Cell* [43]. These are very exciting and ambitious projects; however to use such

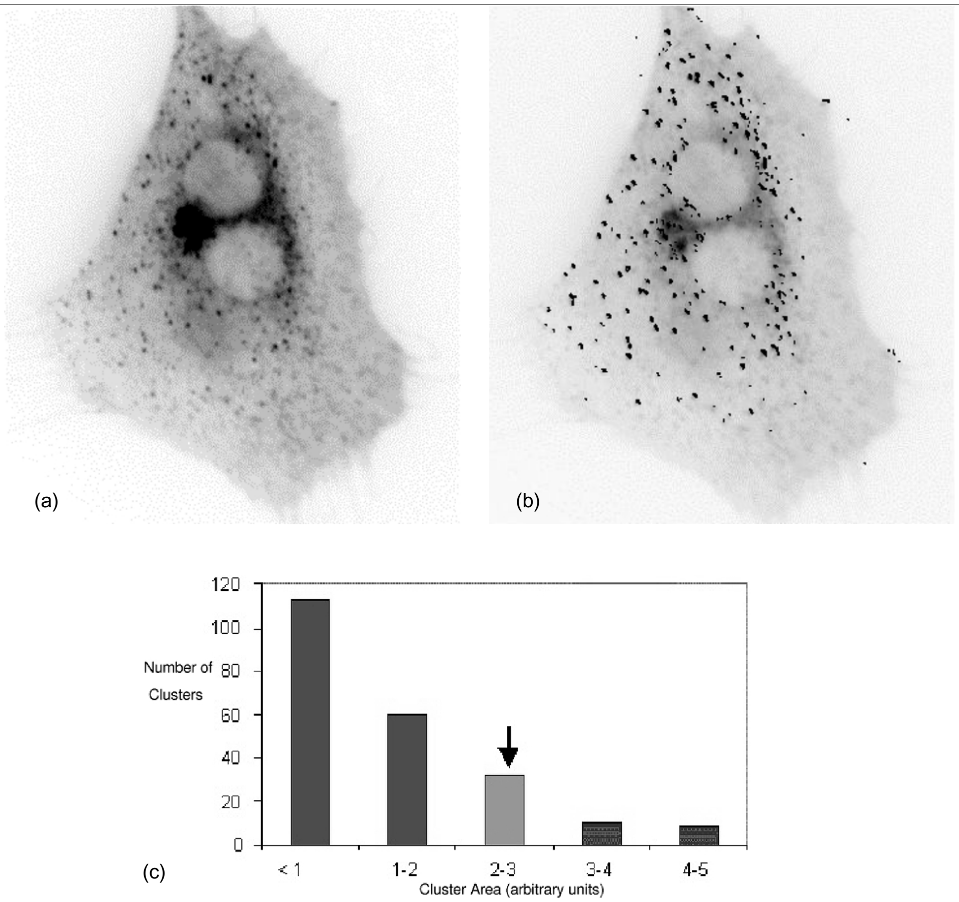

Fig. 2. Image analysis of EGFR clustering in a mouse fibroblast cell micro-injected with a GFP-tagged EGFR cDNA. (a) Original image obtained 5 minutes after addition of 500 nm EGF. (b) Image after analysis using SimplePCI software, with objects of interest highlighted. (c) Data on cluster area generated using the SimplePCI software.

a system requires much of knowledge about the particular type of cell in question, in particular detailed sequence- and structure- data about the proteins in the cell. In problems such as the one studied in this paper, there are limited amounts of data. Nonetheless the questions we would like to answer are correspondingly limited, and we have created a focused tool which allows us to exploit what data is available and actively apply the simulation system to discovering conjectures for parameters in the real system. This “middlebrow modelling” approach allows investigations of focused problems which fall between broad qualitative “models of ideas” [40] and models which aim to reproduce the entire cell *in silico*.

3. Simulation aims and design

We have created a simulation of this clustering behaviour, based on an individual-based object-oriented model of the cell surface. Object-oriented (OO) modelling is a powerful technique for modelling complex systems [8,22,32]. Designing a single algorithm which defines the outcome of a simulation of a system such as the above would be very difficult, if not impossible. Instead, objects are used to represent real-world entities which are components of the system. Objects encapsulate the state and behaviour of real-world entities. This behaviour can include interactions with other objects, which in turn may influence the state and behaviour of these objects. The complexity of the system emerges from these interactions.

This object-oriented approach provides an approach to modelling systems which can be carried through from the analysis phase of studying the system to be put on the computer, through the design of the computer program, through to the final implementation through the use of object-oriented programming languages. Another advantage is that OO approaches can be readily combined with parallel computing approaches – the two approaches are not mutually exclusive choices, and indeed the OO structure can suggest ways of naturally parallelizing the problem [36,37].

The primary classes of objects used in modelling growth factor receptor aggregation are the receptor molecules themselves. In order to create objects which represent molecules, it was necessary to decide which characteristics of molecules are important in the context of the simulation. In reality, cell surface molecules exist somewhere in the cell membrane and they exhibit Brownian motion (except when there are interactions between the receptors) [24]. Growth factor receptors (represented by a `Molecule` class in the system) have a certain affinity for other receptors of the same family to form dimers, which is greatly increased by the addition of ligand. These dimers then form larger clusters.

The `Molecule` class was written to contain information about individual receptor monomers in the simulation. The state of a `Molecule` object can be represented by its position in the cell membrane, the direction and speed with which it diffuses in the cell membrane, its size and whether or not it has ligand bound. Thus, the attributes representing the state of a `Molecule` object include x- and y- floating-point coordinates, to indicate location, and values for dx and dy to indicate changes to x and y caused by moving (i.e. to define the direction of its next move). A `Molecule`'s size is represented by a diameter which is proportional to the number of subunits it contains. A `Molecule`'s type contains information about the number of subunits and whether or not it has ligand bound. The behaviour of `Molecule` objects includes, for example, the ability to move, to bind ligand, and to form aggregates with other `Molecule`s.

Dimers and higher-order oligomers (clusters of receptors) have many of the same characteristics as receptor monomers and behave much the same way on the cell surface. They have a location, size, they move, collide and bind to each other. One powerful feature of object-oriented design is the use of inheritance, which allows a subclass to inherit the attributes and behaviour of another while allowing added functionality not included in the superclass. The `Multimer` class is a subclass of `Molecule` to represent aggregates of one or more receptor monomers. It inherits most the functionality of `Molecule`, but has an additional collection attribute to keep references to the individual `Molecule`s of which it is composed. In this way, a `Multimer` is both a `Molecule` and a collection of `Molecule`s (Fig. 3). It also has the ability to dissociate, a behaviour that is not allowed in `Molecule` objects.

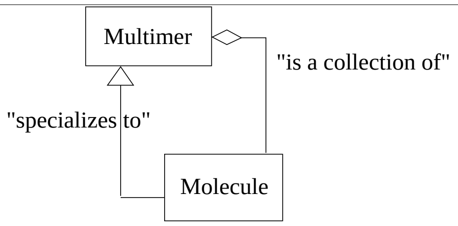

Fig. 3. Modelling the multimer class as simultaneously a collection of and special case of the molecule class.

This property of `Multimer`s, where they are both a type of `Molecule` in their own right and a collection of `Molecule`s demonstrates a *pattern* which might well be found in other scientific models. It would be worthwhile to study a number of models and attempt to see what similar features emerge. Are there *design patterns* which are science-specific which could help in the design of simulations, in the way in which general design patterns [17] have helped in programming more generally?

An object of the `CellSurface` class acts as the simulation engine. This object correlates to a rectangular portion of a cell membrane, in which the `Molecule`s move and interact. The area it covers is represented by an attribute which is a two dimensional Cartesian plane with the location of objects within that plane given by floating-point coordinates, so that the location of the `Molecule`s can be determined precisely. The cell surface is clearly not a strictly planar structure as it is highly dynamic forming structures such as filopodia and lammelapodia. However our model however only simulates a small, typical, patch of the cell surface and thus this can be considered as essentially flat. Although a cell is a three dimensional entity, cell surface molecules are embedded in or attached to the cell membrane, which is essentially a planar surface, particularly over small areas. The motion of these molecules is therefore constrained to two dimensions, and can be modelled in this way. `Molecule`s move on the `CellSurface` in a toroidal fashion. As a `Molecule` leaves the confines of the `CellSurface`, its position is reset such that it re-enters on the opposite side. This provides an easy way to represent a typical area of cell surface where the incoming flux of `Molecule`s into that area is the same as the outgoing.

Each `CellSurface` object is associated with an `AffinityTable` object. This provides a two dimensional array of values for association probabilities, the indices of which correspond to the `Molecule` types. Thus, when two `Molecule`s collide, their affinity can be looked up in this table based on their values for type

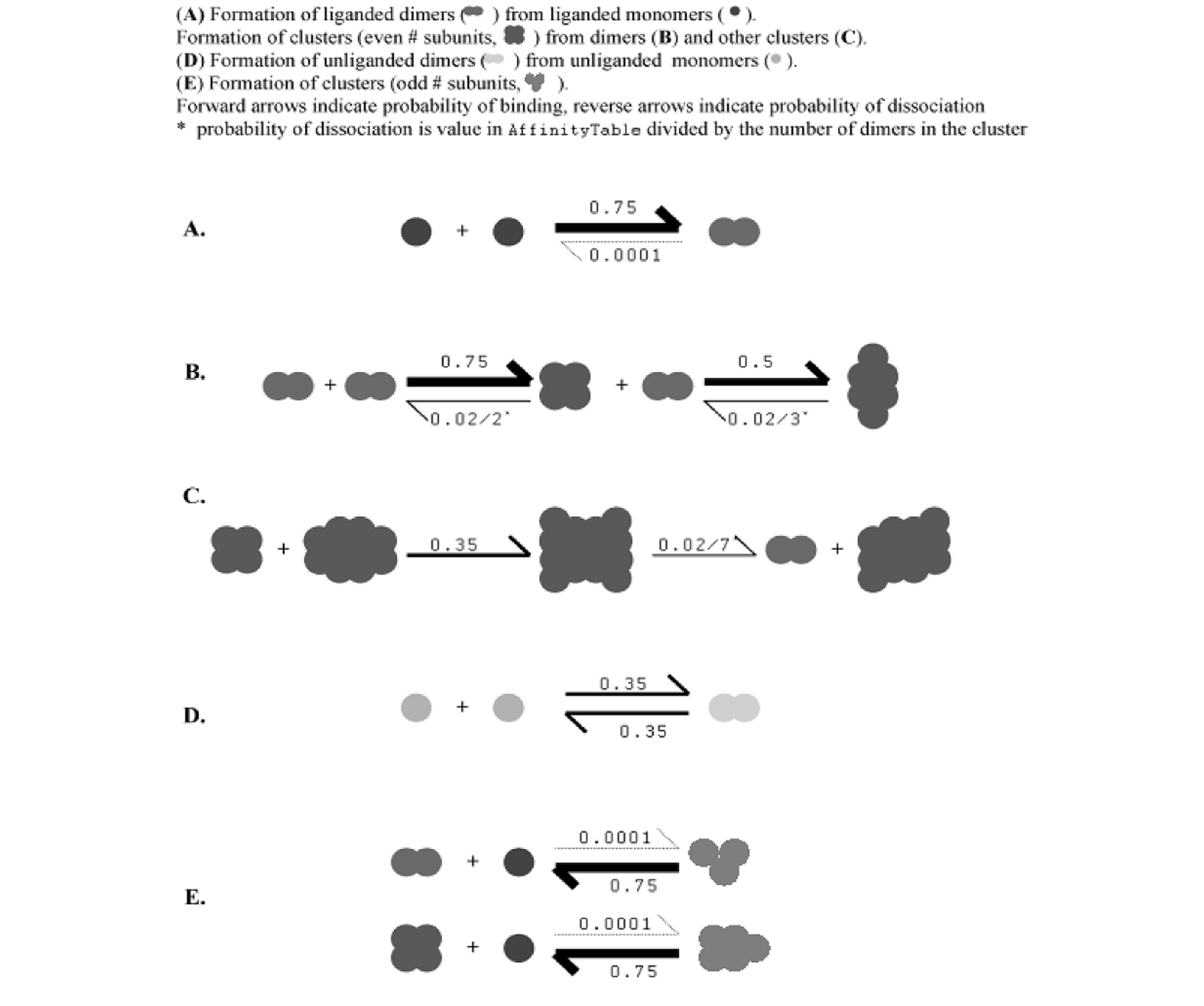
(A) Formation of liganded dimers ( ) from liganded monomers (●).
Formation of clusters (even # subunits,) from dimers (B) and other clusters (C).
(D) Formation of unliganded dimers ( ) from unliganded monomers (●).
(E) Formation of clusters (odd # subunits,).
Forward arrows indicate probability of binding, reverse arrows indicate probability of dissociation
* probability of dissociation is value in AffinityTable divided by the number of dimers in the cluster
A.
+
0.75
0.0001
B.
+
0.75
0.02/2*
+
0.5
0.02/3*
C.
+
0.35
0.02/7
+
D.
+
0.35
0.35
E.
+
0.0001
0.75
+
0.0001
0.75

Fig. 4. An example of transition probabilities between various states.

(an example is given in Fig. 4). The `AffinityTable` also has an array for dissociation probabilities in which the indices correspond to molecule type.

The design is summarized in a UML [16,46] diagram in Fig. 5.

The simulation begins with the initialisation of the cell surface. This involves creating a `CellSurface` object and populating it with unliganded monomers distributed randomly about the surface with respect to a uniform distribution. This is certainly true at the level of resolution of the light microscope (see for instance Fig. 3(a) at http://www.cs.ukc.ac.uk/people/staff/cgj/research/receptors.html). Some evidence has been reported of very small structures called lipid rafts [11] in which it has been suggested that some proteins (including growth factor receptors) may be concentrated. There are however contrary reports and as this is not well established as a first approximation we have chosen to give the monomers a random distribution at the initial state of the model.

The program then runs as a loop (representing the passage of time) in which the `Molecule`s are moved in a manner which simulates Brownian motion. This simulates the collision of the molecules of interest with the ambient molecules in the cell which have no probability of binding to them, i.e. the vast majority of collisions which happen during the lifetime of the receptor. At the beginning of any cycle through this loop the user can choose to simulate the addition of ligand to the `CellSurface`, which initiates the aggregation process. The user can vary the quantity of ligand added by determining the percentage of the existing unliganded receptor monomers to change their type to liganded.

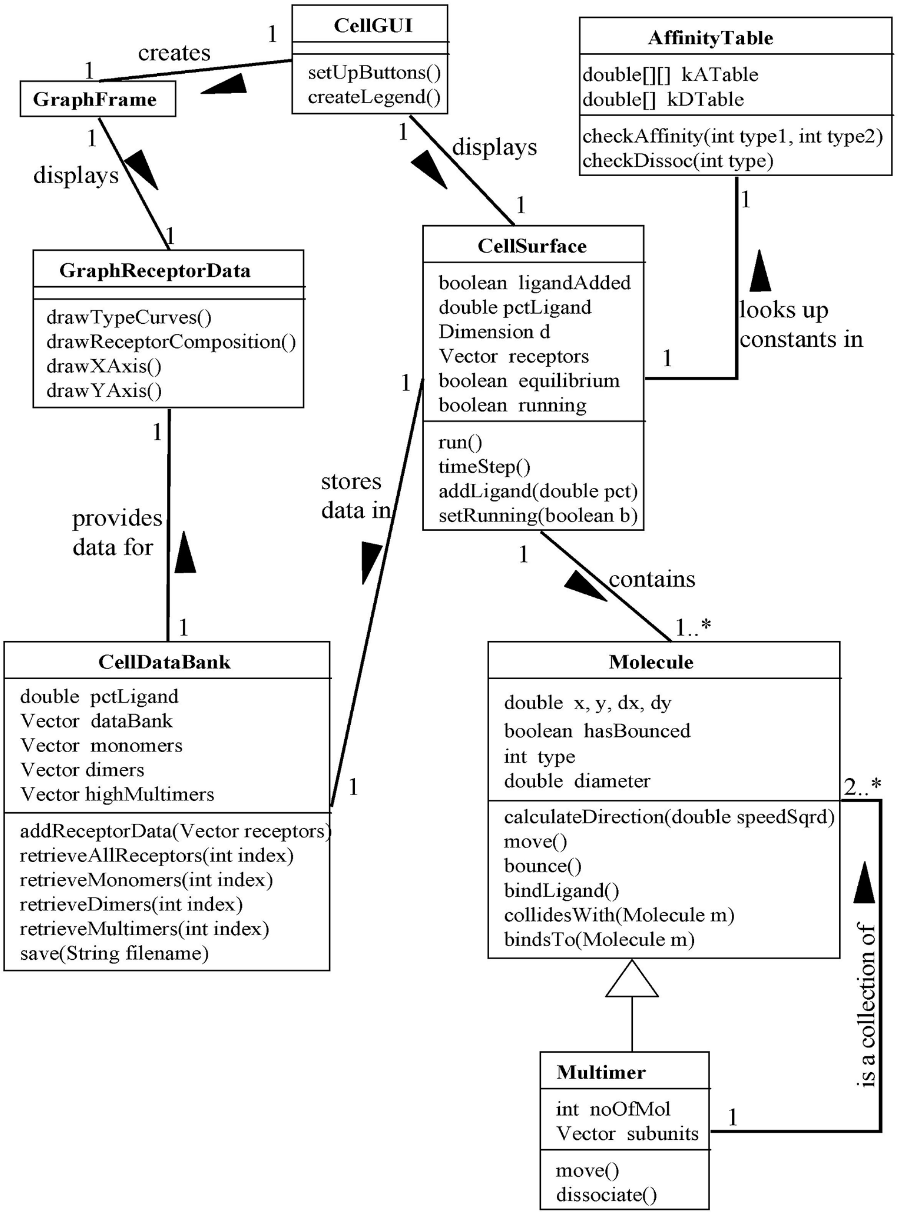

Fig. 5. A UML diagram to summarize the design of the simulation.

All these liganding events occur at the beginning of the current cycle through the time-loop.

When the positions of two `Molecule`s indicate that a collision between the two has occurred the `CellSurface` uses the `AffinityTable` to look up values for binding and dissociation constants. The sequence is as follows: a `Molecule` moves and its position is checked with regard to other `Molecule`s in the simulation. If there is any overlap between the area covered by that `Molecule` and another `Molecule`, a collision is deemed to have taken place. A random probability is chosen, and if this is below the affinity threshold from the `AffinityTable` a `Multimer` is formed whose subunits consist of the monomers from the colliding `Molecule`s. If, however, the probability is above the affinity threshold, the moving `Molecule` stops at the point in its trajectory just outside the area of the other `Molecule`. Thus, `Molecule`s cannot move through each other, and no two `Molecule`s can occupy the same space.

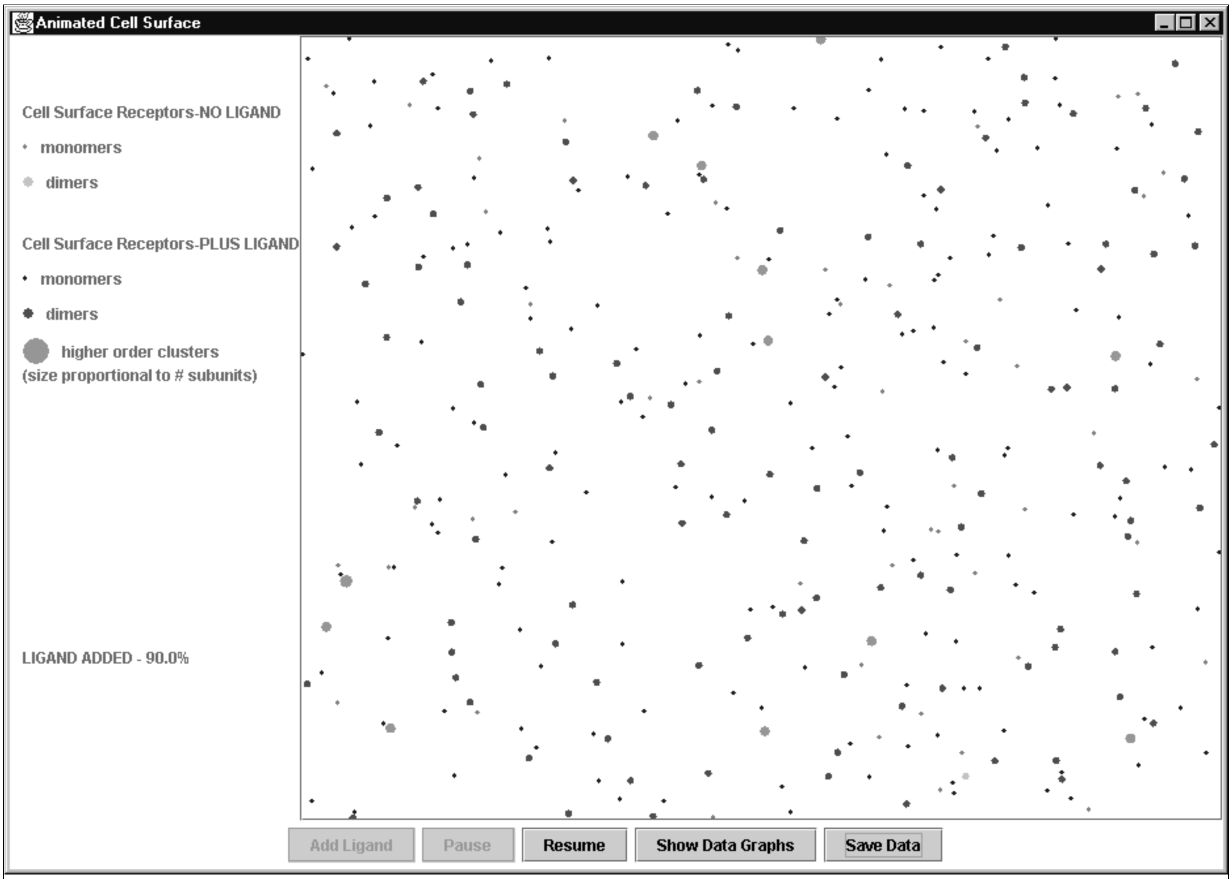

Fig. 6. A snapshot from the animation of a portion of the cell surface. The cell surface shortly after ligand addition.

The graphical display of the `CellSurface` draws each `Molecule` as a circle whose area is proportional to the number of subunits it contains. This can be used to generate a computergraphical animation of the changing state of the cell surface (Figs 6, 7 and 8). The program can also generate statistics about the changing state of the cell with time.

4. Speeding up the simulation

The cell surface is represented as a two dimensional rectangle on which the various components are represented via a pair of floating-point coordinates. This space represents a part of the surface of the cell, and toroidal coordinates are used (a component leaving the top of the rectangle returns at the base of the cell, one leaving the right hand side returns on the left, and *vice versa*) as a simple way of ensuring that the number of molecules in the area is conserved. This is a simple model which works well, as in a typical region on the surface the inward flux of material is equal to the outward flux.

This use of floating-point positions contrasts with the common use of lattice models in individual-based modelling (e.g. [27]). In lattice models each component is placed at a pair of integer coordinates on a lattice, which can lead to artifacts which do not represent the real world; an example would be that (given a simple move up-down-left-right model of motion) there is a bias toward motion in the grid directions which does not reflect the uniformity of the real world. It also contrasts with continuum models based upon density gradients of molecules (e.g. [35,12,5]). In summary the objects which are discrete in the real world (i.e. the individual molecules) are represented by discrete computational objects, whilst the parts of the system which are continuous (i.e. the ambient medium in which the molecules move) are represented by a continuum of values in the computer.

One reason why lattice models are typically used in individual based spatial modelling is to simplify the

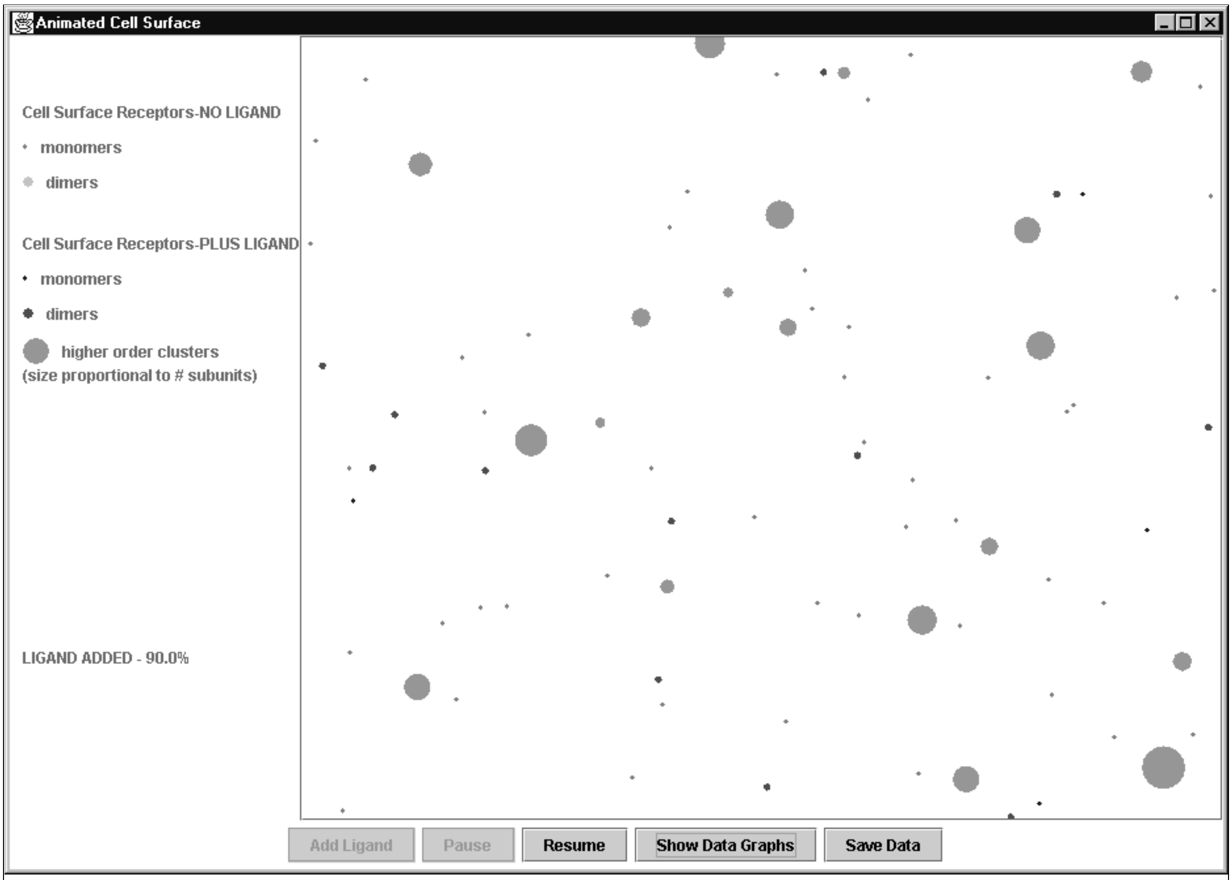

Fig. 7. A snapshot from the animation of a portion of the cell surface. Approximately half way through the simulation (∼ 12000 iterations).

problem of collision detection between particles in the system. However in our model we have introduced a technique which allows fast collision detection whilst retaining the idea of each object's position being represented by floating-point coordinates.

This system exploits the ability of OO systems to link two pieces of information together so that each of the two pieces is mutually aware of the other. We use this to combine the advantages of the lattice and continuum representations. The main representation places the receptors on the 2-dimensional floating-point continuum; all motion takes place in this space. However in addition to this space a grid spans the space.

Each point on this grid "owns" the rectangle of the surface for which it is the top left-hand corner. At each crossing point on the grid there is a list (a variable-length array type such as a Java `ArrayList`) of references (C-style pointers) to objects which are of the class of the items on the surface. This is easily extended to multiple types of object moving on the surface, by having the pointers point to a Java-style interface type, which is implemented by all of the points on the surface.

The simulation begins by iterating through the objects on the surface, calculating which region(s) they belong to, and adding a reference from the appropriate grid point to the object (Fig. 9). The program then makes a list of references (C-style pointers again, or just coordinates) *back* from the object to the grid points which contain them. This ability for two objects to both know data about each other is a powerful technique in object-oriented modelling, and is a powerful way to get away from an oversimple hierarchical view where each piece of data contains other data, and not *vice versa*.

Once this data structure has been established, checking for collisions is easy. Take the object which we are interested in, and iterate through its list of grid points. This gives us a list of regions which the object occupies. These grid points are then taken in turn and their list of objects iterated through. This gives a list of objects which occupy the region in question. An intersection

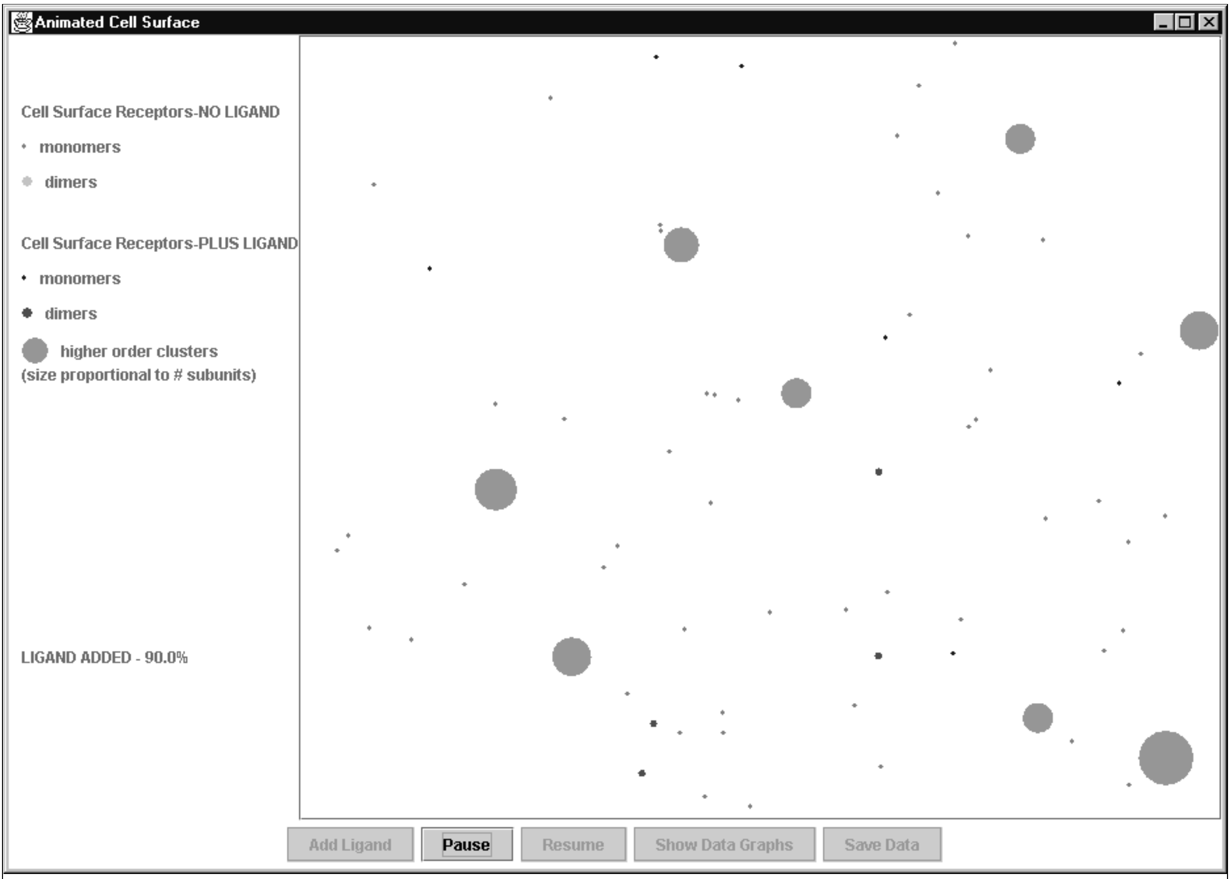

Fig. 8. A snapshot from the animation of a portion of the cell surface. Simulation complete (∼ 24000 iterations).

check can then be done on each of those objects with the original object.

If an object moves a single method can be called which updates its grid position. This method calculates which regions of the grid the object now interacts with. Using this list and the previous list, a message could be sent to each of the grid-points whose regions are not longer occupied asking that it be removed from that grid-point's list of objects. It then removes its own links to any regions which are no longer occupied, and sends messages to the grid-points belonging to any newly occupied region asking for it to be added to its list. This can be neatly contained in its own method.

There are similarities between this idea and various other approaches used in fluid mechanics, plasma simulation, et cetera [39,14,4,44,37]. However one of the strengths of the approach above is that the code which keeps track of the position of the particles relative to the grid points is self-contained; we do not have to incorporate code which takes this issue into account every time we write a piece of code which can move particles. A further discussion of this is given in [23].

Figures 10, 11 and 12 give results for three experiments with a different number of molecules in each experiment. The results are clear; using the grid speeds up the program immensely, reducing the increase in computational load as we increase the scale of the experiments to almost constant regardless of the size of the experiment. This is beneficial as we want to simulate a realistic cell with hundreds of thousands of relevant molecules. In figure 10 a reduction in the run time as the number of molecules is increased is observed; this appears to be an experimental fluctuation rather than a genuine phenomenon.

Further experiments have been aimed at finding the optimal size for the grid. If the size of the squares in the grid is too large, then the program needs to look through too many objects each time a collision check is performed. If the grid becomes too small, then there needs to be large numbers of links, because each object occupies multiple grid-regions. Intuitively the best size

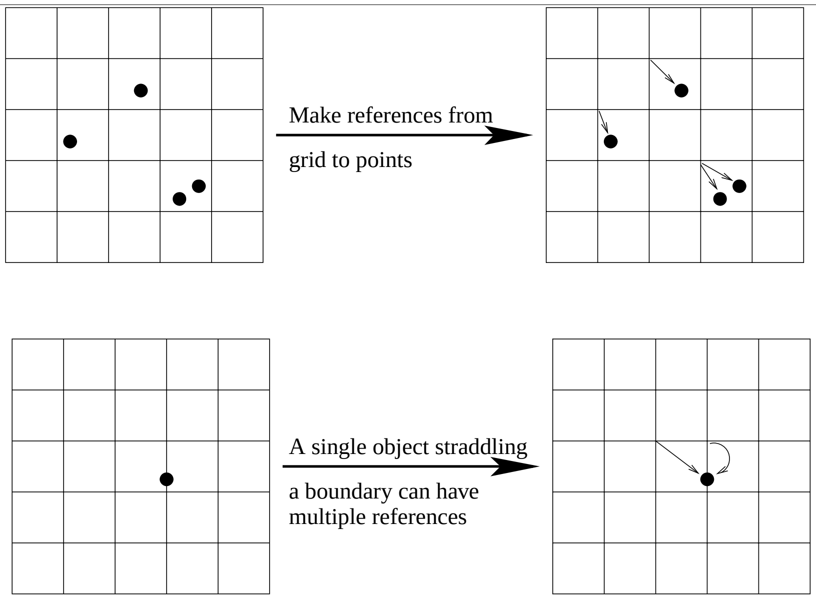

Fig. 9. Clamping the objects to the corners of the grids which represent them.

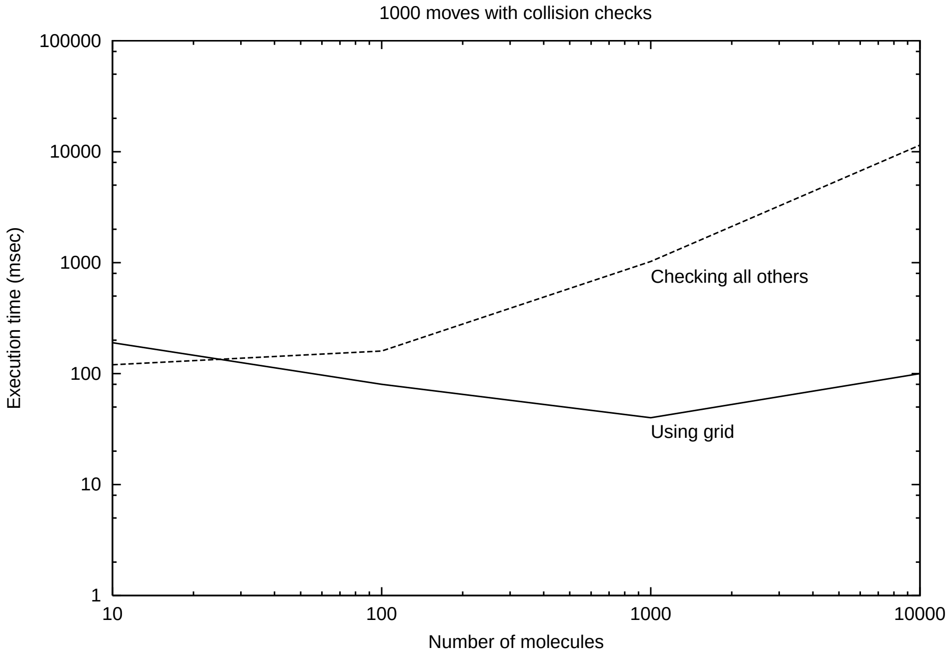

Fig. 10. Comparing the grid-based algorithm with the algorithm which checks all possibilities (1000 moves).

would seem to be slightly larger than the typical size of the object, as this would prevent both of these extremes – we might characterize this by saying that the typical tile should contain one object most of the time. Thus we expect a U-shaped curve. Results confirming this are given in Fig. 13.

5. Experiments

Development of the simulation is ongoing as is integrating the simulation work with current experimental work. In this section we present some early experiments with the system. The results were carried out

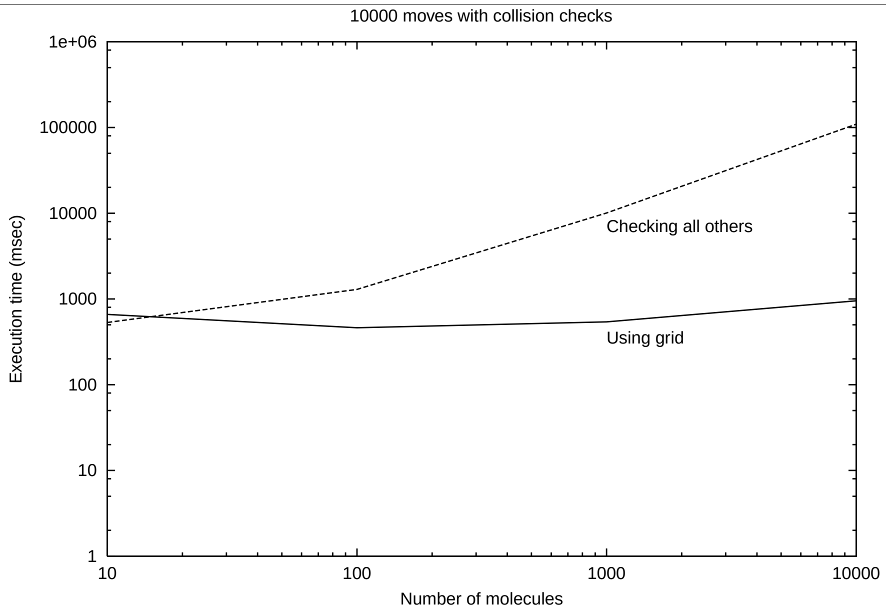

Fig. 11. Comparing the grid-based algorithm with the algorithm which checks all possibilities (10000 moves).

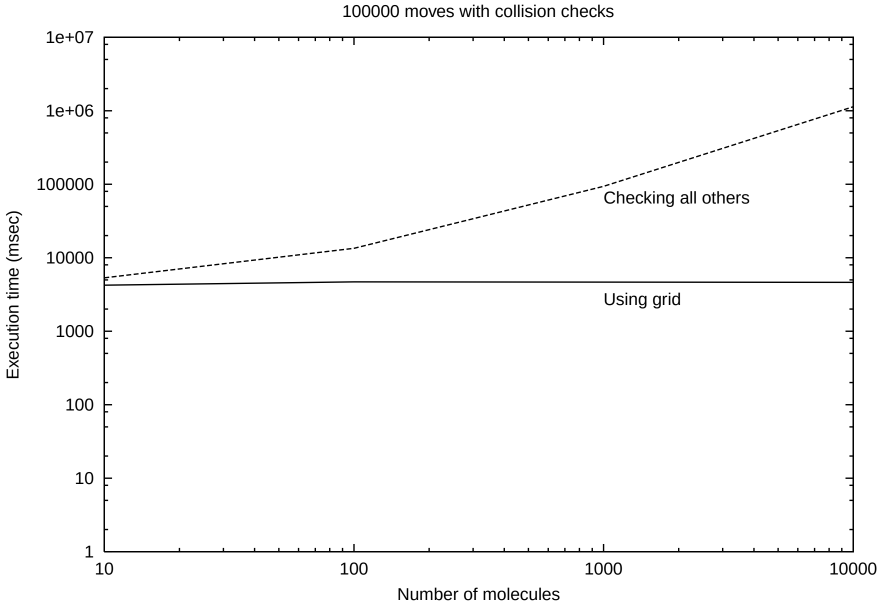

Fig. 12. Comparing the grid-based algorithm with the algorithm which checks all possibilities (100000 moves).

using Java1.3 on a standard desktop PC (a Pentium III with 256 MB of RAM).

5.1. *Confirming that Brownian motion is being accurately simulated*

Earlier on in the paper it was suggested that one of the important aspects of object-oriented programming is that individual components of the system can be programmed "in the small" and let the global properties of the system emerge. An important part of confirming that the system is working correctly is checking that certain standard behaviours emerge.

One example of this is confirming that the underlying motion rules give rise to Brownian motion. Berg [7]

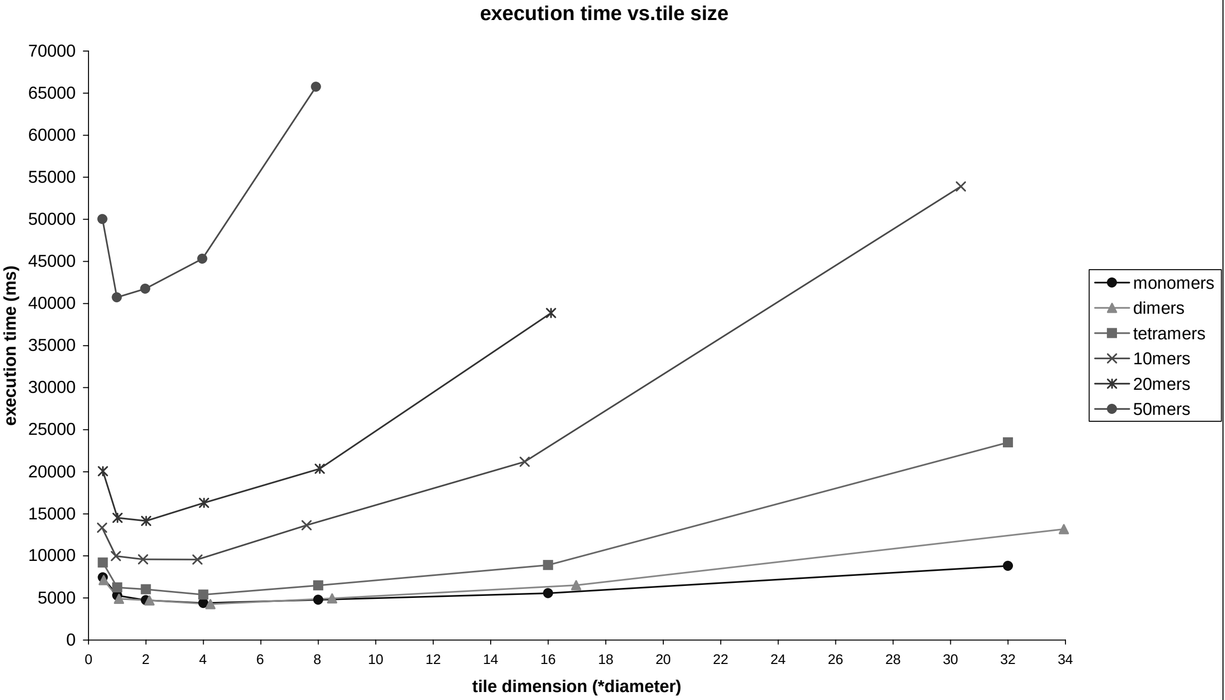

Fig. 13. Results: Best grid size.

shows that the relationship between the mean square distance of diffusing particles from the origin $\langle r^2 \rangle$, the diffusion coefficient D and time t is of the form $\langle r^2 \rangle = 4Dt$. Therefore given that the diffusion coefficient in our system is constant, we would expect $\langle r^2 \rangle$ to increase linearly with time in our system. A first experiment was carried out in which only the *implicit* collisions were carried out, i.e. those changes in molecule direction which occurred at the end of a timestep because of a collision with a molecule not explicitly represented in the system.

These experiments had the following characteristics. A value for D was chosen as $5 \times 10^-9 \mathrm{cm}^2/\mathrm{s}$, which is an estimate for the diffusion coefficient for a small protein in a lipid bilayer [7]. However the precise value of this coefficient is not significant with respect to determining whether the relationship is linear, however it is useful for estimating how much real time is represented by each timestep of the simulation. The simulations were carried out with 10000 molecules.

The results from this are shown in Figs 14(a) and (b). The relationship is clearly linear, and the gradient of the increase varies according to molecular weight as expected.

A second experiment was carried out which included the explicit collisions, i.e. the collisions between two molecules which are explicitly represented in the system. The probability of aggregation was set to zero; therefore all changes of direction represent either an explicit or implicit collision. The results of this experiment are given in Fig. 15. Again there is a clear match between predicted and experimental results.

5.2. *Parameter sensitivity*

One way in which this system can be used is to probe the parameter-space of the system to discover which of the parameters in the system are most significant in producing the behaviour of interest. This is important for a number of reasons. Firstly it provides a way for experimental scientists to decide which components of the system are worth investigating experimentally, by seeing which parts of the simulated system most affect that phenomenon of interest. These can then be prioritised for experimental investigation, whereas those components which seem to have little effect can be treated as a lower priority.

A second motivation is medical in origin. One of the aims of understanding systems such as this is the desire to change the behaviour of the system so as to attack the underlying causes of illnesses. In order to provide an effective treatment the treatment agent must affect some part of the system which causes the phenomenon of interest to change.

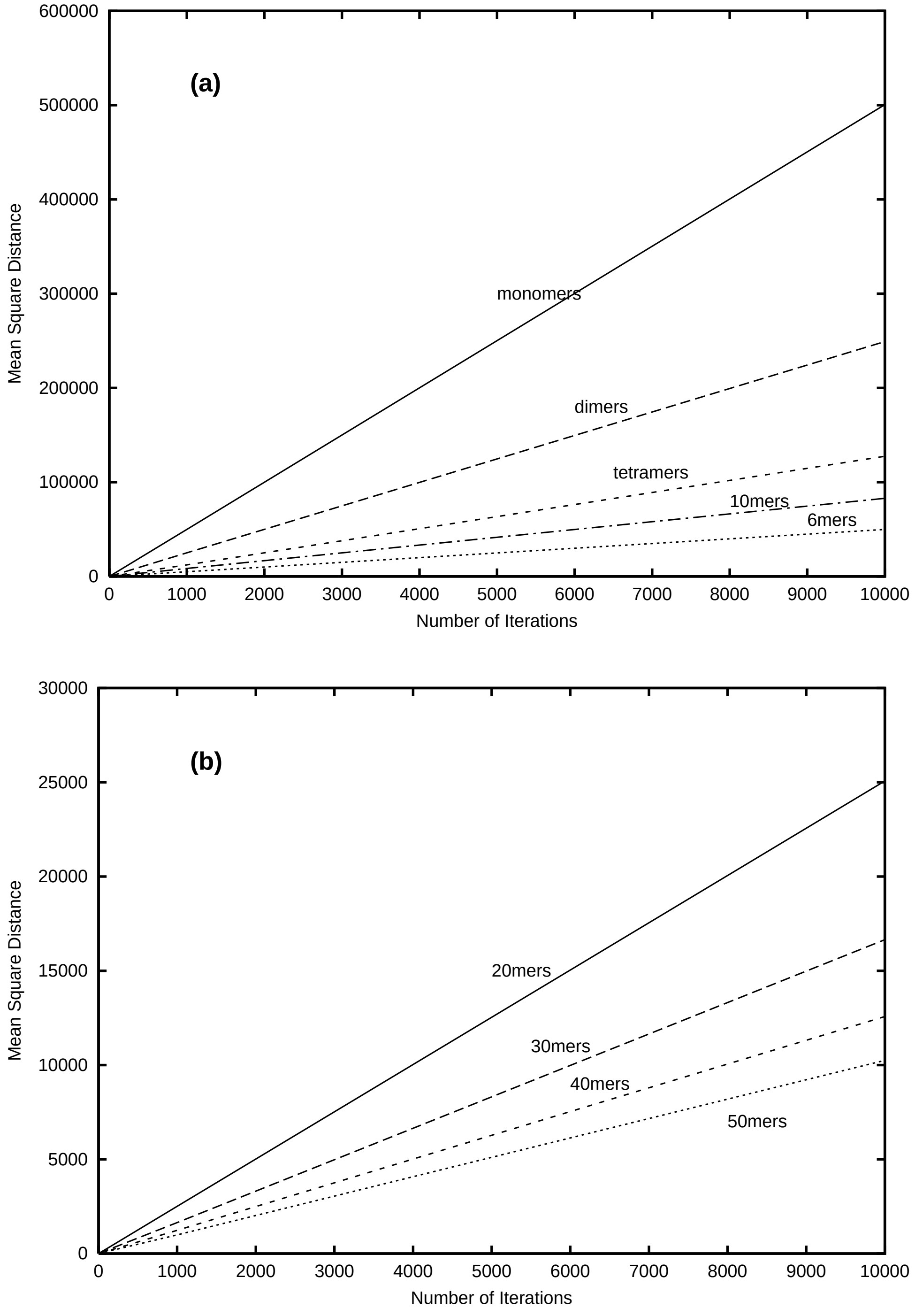

Fig. 14. Tests to confirm that brownian motion is emerging from the basic motions of the particles in the system. (a) gives results for 1–10 mers, (b) gives results for 20–50 mers.

Thirdly such studies are of interest because they shed light on how “generic” a particular phenomenon is. Is a particular phenomenon the typical result of a system of that kind, regardless of the details of the parameter settings? Is it an isolated phenomenon which can only be found in a small area within the parameter space? Are similar phenomena caused by nearby parameter settings, and if so does this provide a poten-

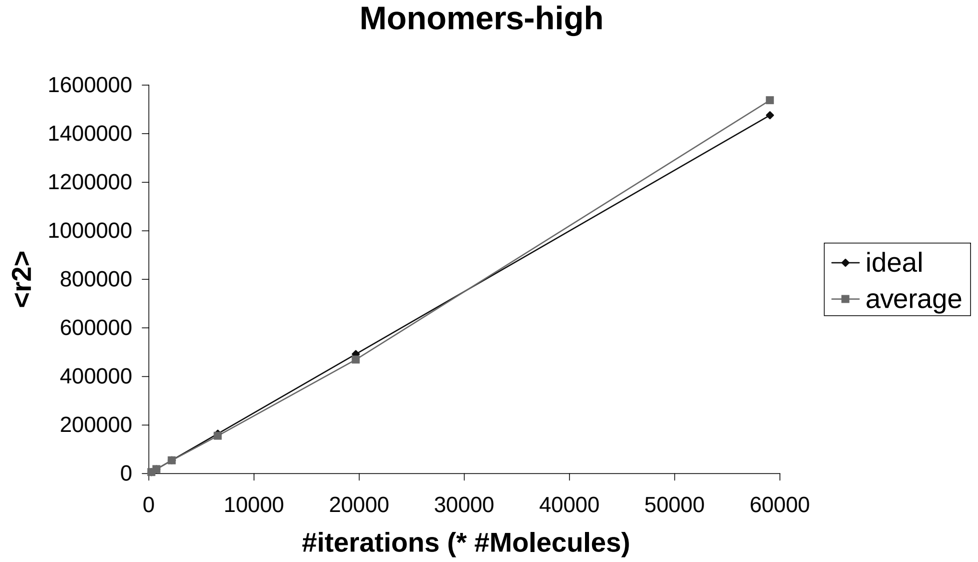

Fig. 15. Testing the brownian motion when both explicit and implicit collisions are included in the system.

tial route through which an evolutionary process could have reached the current situation?

All experiments were run three times. There are 500 receptors on the cell surface, and the cell surface represents approximately $2 \times 10^{-12}\mathrm{m}^2$, which is approximately $1/200$th of the area of the cell. The graphs give the average value for each experiment. In all experiments the following characteristics were measured: the mean cluster size, the number of monomers, the number of clusters, and the number of dimers on the cell surface at the end of the run when the system has reached equilibrium (i.e. where the amount of dissociation balances out against association and cluster formation).

In the first experiment the initial ligand concentration was varied from 10% to 100% and the aforementioned characteristics measured. The results are given in Fig. 16. The results demonstrate that the relationship between this and the various parameters is typically linear. Clearly the number of monomers declines linearly with increased ligand occupancy, because almost all liganded monomers are bound into dimers or clusters with time. The number of clusters and their size increases steadily with increased ligand concentration. This is unsurprising; the more liganded molecules present the more likely it is that clusters will form in particular regions in the cell, and once a cluster reaches a certain size it takes a long time to move close to another cluster, therefore that cluster will be likely to grow by accretion of dimers from its local area. The number of dimers at first appears to be very variable, however the overall numbers are very small compared to the overall number of molecules in the simulation; it would require more experiments to see whether the shape of the curve there is really more than just random fluctuation.

The second and third experiments varied the rate respectively of association (Fig. 17) and dissociation Fig. 18) in the cell. The difference is marked. Changing levels of association have a fairly minor effect on the final state of the cell, whilst there is a sharp change in behaviour within a very small area on the dissociation probability axis. One possible explanation for this is that in the long run the number of opportunities for association is large, so even if the probability of association is small the long term behaviour will be the same. The suddenness of the change in the behaviour when the dissociation is an interesting phenomenon, reminiscent of phase-transition behaviour; this would be an interesting phenomenon to investigate further.

6. Ongoing work

Having developed this basic simulation we are currently extending it in a number of different directions. At the most basic level we are looking at how the simulation can be extended to model the full range of receptors and ligands in the system. At present the system only deals with one kind of ligand binding to one receptor type. This limits the scope of the model; in particular it is impossible to study the structure of receptor clusters using the current model. In the real system there are seven receptor types which bind at least ten ligands with different binding strengths [19,48], and clusters containing a mixture of receptor types have been observed [34]. Another factor which might influence the dynamics of the system are *lipid rafts* which concentrate receptors in parts of the cell [11].

In addition to further development of the model we are looking to apply the model to a number of scientific questions. The most important of these is obtaining estimates of the values of various parameters in the system that are inaccessible to direct experimental observation, for example probabilities of association and dissociation. We are pursuing two approaches to this. The first is a visual approach whereby we will create computergraphical "films" of the simulation which can be looked at alongside films of the real system obtained by microscopy (these can be seen at http://wwwbio.ukc.ac.uk/gullick/icrf.htm). This will allow experts familiar with the system to adjust parameters until the simulation film has similar characteristics to the real film. The second approach involves carrying out image processing on the real films to obtain statistics such as time series of changes in cluster size. We will then apply optimization techniques such as genetic algorithms and tabu search to find parameter settings which match these statistics in the output from the simulation.

Another use to which the simulation will be put is in searching for components of the system which are most sensitive to change, as such regions might be suitable targets for therapeutic intervention. Characteristics of the system such as association and dissociation probabilities can be affected by binding other molecules onto the receptors or ligands. An example of the successful therapeutic application of interventions aimed at disrupting an overexpressed growth factor receptor system is the anti-cancer drug trastuzumab (Herceptin$^{\mathrm{TM}}$), which blocks a form of growth factor receptor [45]. To create such treatments in a rational way we need to search the space of possible intervention sites for those where small interventions have large effects, and computational search techniques such as genetic algorithms and active nonlinear tests [33] provide a potential method for doing this in a tractable way.

A more general ongoing project is examining how we can create more general object-oriented modelling techniques for interacting protein systems, and more generally still how we need to adapt object-oriented techniques for the type of programming needed for science rather than business applications.

7. Notes

This work has been funded by the UK Medical Research Council and the University of Kent. The current version of the model can be found on the web at http: //www.cs.ukc.ac.uk/people/staff/cgj/research/receptors.html.

The authors would like to thank Dennis Bray for his comments on the work.

References

[1] P. Agarwal, Simulation of aggregation in *Dictyostelium* using the cell programming language, *Computer Applications in the Biosciences* **10**(6) (1994), 647–655.

[2] P. Agarwal, The cell programming language, *Artificial Life* **2**(1) (1995), 37–77.

[3] P. Agarwal, Cellular segregation and engulfment simulations using the cell programming language, *Journal of Theoretical Biology* **176**(1) (1995), 79–89.

[4] M.P. Allen and D.J. Tildesley, *Computer Simulation of Liquids*, Clarendon Press, 1987.

[5] A.R.A. Anderson, M.A.J. Chaplain, E.L. Newman, R.J.C. Steele and A.M. Thompson, Mathematical modelling of tumour invasion and metastasis, *Journal of theoretical medicine* **2** (2000), 129–154.

[6] T.M. Bartol and J.R. Stiles, MCell: A general monte carlo simulator of cellular microphysiology. Website describing ongoing project, at http://www.mcell.cnl.salk.edu/ (visited December 2000), 1999.

[7] H.C. Berg, *Random Walks in Biology*, (expanded second edition), Princeton University Press, 1993.

[8] G. Booch, *Object-oriented design with applications*, (second edition), Benjamin/Cummings, 1994.

[9] D. Bray and S. Lay, Computer simulated evolution of a network of cell-signaling molecules, *Biophysical Journal* **66** (1994), 972–977.

[10] D. Bray, M.D. Levin and C.J. Morton-Firth, Receptor clustering as a cellular mechanism to control sensitivity, *Nature* **393** (1998), 85–88.

[11] G. Carpenter, The EGF receptor: a nexus for trafficking and signalling, *Bioessays* **22**(8) (2000), 697–707.

[12] P.D. Dale, L. Olsen, P.K. Maini and J.A. Sherratt, Travelling waves in wound healing, *FORMA* **10** (1995), 205–222.

[13] D. DeAngelis and L. Gross, eds, *Individual-based Models and Approaches in Ecology*, Chapman and Hall, 1992.

[14] J.J. Erpenbeck and W.W. Wood, Molecular dynamics calculations of shear viscosity time correlation functions for hard spheres, *Journal of Statistical Physics* **24** (1981), 455–468.

[15] D.A. Fell, *Understanding the Control of Metabolism*, Portland Press, London, 1996.

[16] M. Fowler, *UML Distilled*, Addison-Wesley, 1997.

[17] E. Gamma, R. Helm, R. Johnson and J. Vlissides, *Design Patterns*, Addison-Wesley, 1994.

[18] H. Gillham, M.C.H.M. Golding, R. Pepperkok and W.J. Gullick, Intracellular movement of green fluorescent protein-tagged phosphatidylinositol 3-kinase in response to growth factor receptor signalling, *Journal of Cell Biology* **146**(4) (1999), 869–880.

[19] W.J. Gullick, The type 1 growth factor receptors and their ligands considered as a complex system, *Endocrine-Related Cancer* **8** (2001), 75–82.

[20] J. Hancock, *Cell Signalling*, Longman, 1997.

[21] K.M. Heal, M.L. Hansen and K.M. Rickard, *Maple V: Learning Guide*, Springer, 1996.

[22] I. Jacobson, *Object-Oriented Software Engineering: A Use Case Driven Approach*, Addison-Wesley, 1992.

[23] C.G. Johnson and J.L. Whalley, Detecting collisions in sets of moving particles: a survey and some experiments, Technical Report 8-02, University of Kent, 2002.

[24] A. Kusumi, Y. Sako and M. Yamamoto, Confined lateral diffusion of membrane receptors as studied by single particle tracking (nanovid microscopy). effects of calcium-induced differentiation in cultured epithelial cells, *Biophysical Journal* **65**(5) (1993), 2021–2040.

[25] T. Lamb and L. Wischik, Walk: A stochastic simulation of the G-protein cascade of phototransduction. Available on the WWW at http://www.physiol.cam.ac.uk/staff/lamb/Walk/index.html (visited September 2000), 1996.

[26] T.D. Lamb, Stochastic simulation of activation in the G-protein cascade, *Biophysical Journal* **67** (1996), 1439–1454.

[27] C.G. Langton, Studying artificial life with cellular automata, *Physica D* **22** (1986), 120–149.

[28] D.A. Lauffenburger and J.J. Lindeman, *Receptors: Models for Binding, Trafficking and Signaling*, Oxford University Press, 1993.

[29] C.C. Maley and H. Caswell, Implementing i-state configuration models for population dynamics: An object-oriented programming approach, *Ecological Modelling* **68** (1993), 68–75.

[30] P. Mendes, Gepasi: A software package for modelling the dynamics, steady states and control of biochemical and other systems, *Computer Applications in Biosciences* **9** (1993), 563–571.

[31] P. Mendes, Biochemistry by numbers: simulation of biochemical pathways with Gepasi, *Trends in Biochemical Sciences* **22** (1997), 361–363.

[32] B. Meyer, *Object-Oriented Software Construction*, (second edition), Sams, 2000.

[33] J.H. Miller, Active nonlinear tests (ANTs) of complex simulation models, *Management Science* **44**(6) (1998).

[34] C.R.F. Monks, B.A. Freiberg, H. Kupfer, N. Sciaky and A. Kupfer, Three-dimensional segregation of supramolecular activation clusters in T-cells, *Nature* **395** (September 1998), 82–86.

[35] J.D. Murray, *Mathematical Biology*, Springer, 1993.

[36] C.D. Norton, V.K. Decyk and B.K. Szymanski, On parallel object oriented programming in Fortran 90, *ACM SIGAPP Applied Computing Review* **4**(1) (1996), 27–31.

[37] C.D. Norton, B.K. Szymanski and V.K. Decyk, Object-oriented parallel computation for plasma simulation, *Communications of the ACM* **38**(10) (1995), 88–100.

[38] E. Part-Enander and A. Sjoberg, *The MATLAB 5 Handbook*, Longman, 1999.

[39] D.C. Rapaport, The event scheduling problem in molecular dynamics, *Journal of Computational Physics* **34** (1980), 184–201.

[40] J. Roughgarden, A. Bergman, S. Shafir and C. Taylor, Adaptive computation in ecology and evolution: A guide for future research, in: *Adaptive Individuals in Evolving Populations*, R.K. Belew and M. Mitchell, eds, Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley, 1996, pp. 25–30.

[41] H.M. Sauro, SCAMP: a general-purpose simulator and metabolic control analysis program, *Computer Applications in Biosciences* **9**(4) (1993), 441–450.

[42] H.M. Sauro and D.A. Fell, SCAMP: A metabolic simulator and control analysis program, *Mathematical and Computational Modelling* **15**(12) (1991), 15–28.

[43] J. Schaff, C. Fink, B. Slepchenko, J. Carson and L. Loew, A general computational framework for modeling cellular structure and function, *Biophysical Journal* **73** (1997), 1135–1146.

[44] H. Sigurgeirsson, A.M. Stuart and W.-L. Wan, Collision detection for particles in a flow, *Journal of Computational Physics* **172** (2001), 766–807.

[45] M.X. Sliwkowski, Nonclinical studies addressing the mechanism of action of trastuzumab (herceptin), *Seminars in Oncology* **26** (1999), 60–70.

[46] P. Stevens, *Using UML*, (original edition 1999), Addison-Wesley, 2000.

[47] M. Tomita, K. Hashimoto, K. Takahashi, T. Shimizu, Y. Matsuzaki, F. Miyoshi, K. Saito, K. Yugi, J.C. Venter and C. Hutchinson, E-CELL: software environment for whole cell simulation, *Bioinformatics* **15**(1) (1999), 72–84.

[48] Y. Yarden and M.X. Sliwkowski, Untangling the ErbB signalling network, *Nature Reviews: Molecular Cell Biology* **2** (February 2001), 127–137.

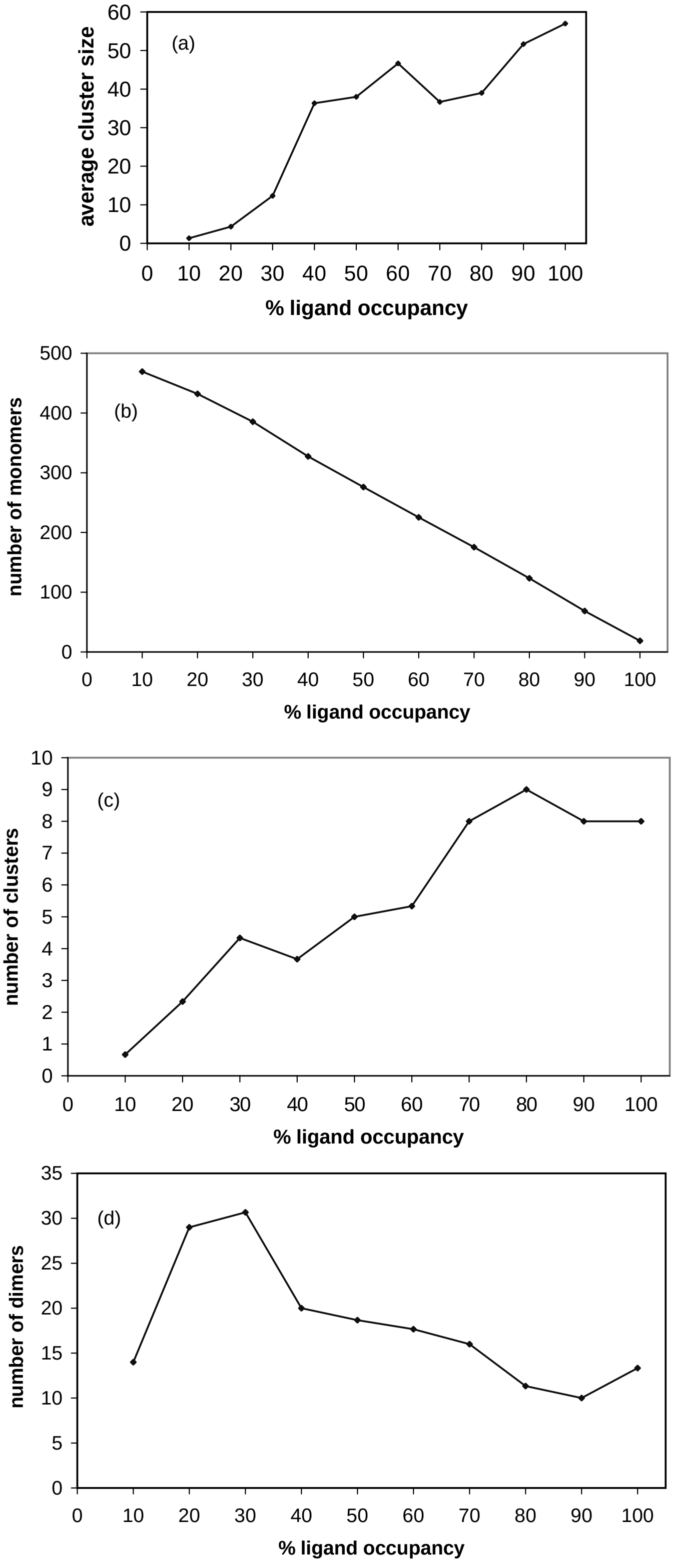

Fig. 16. Characteristics of the system at equilibrium with various initial ligand concentrations.

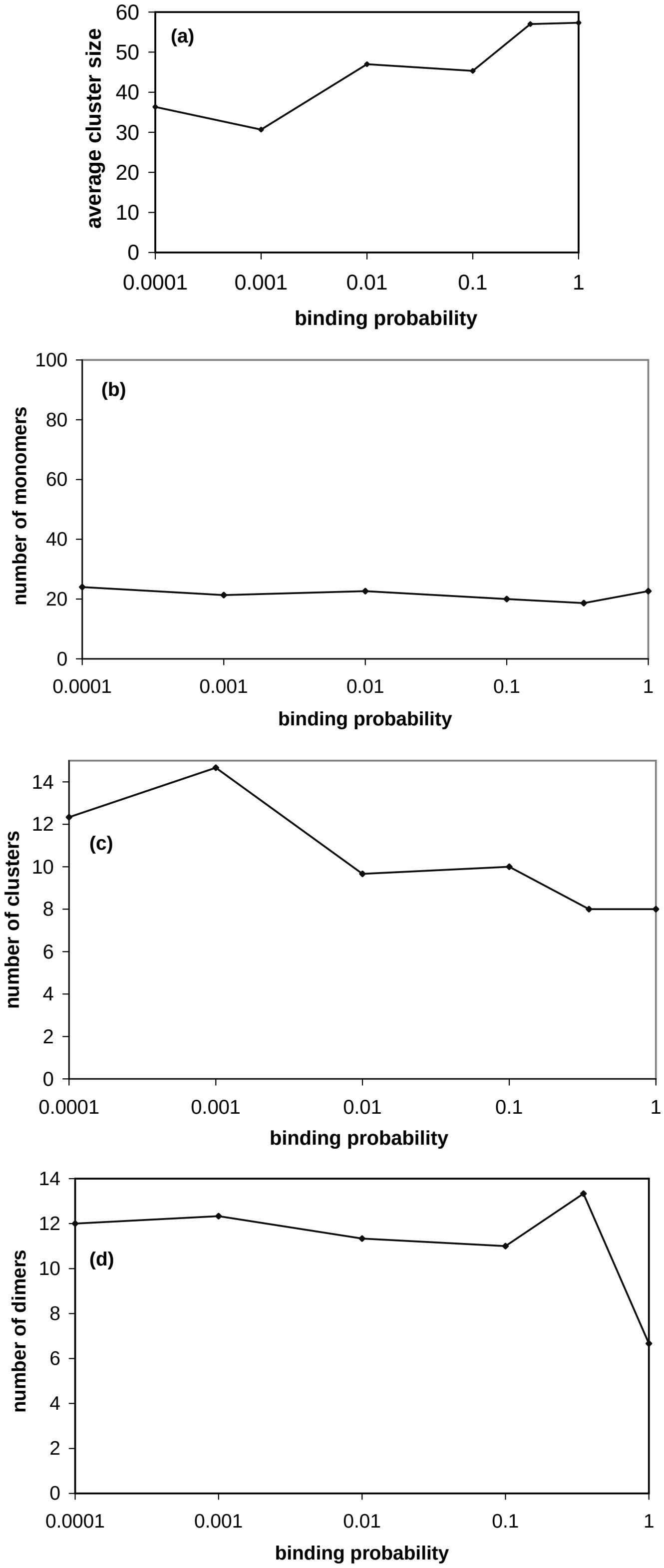

Fig. 17. Characteristics of the system at equilibrium with various association rates.

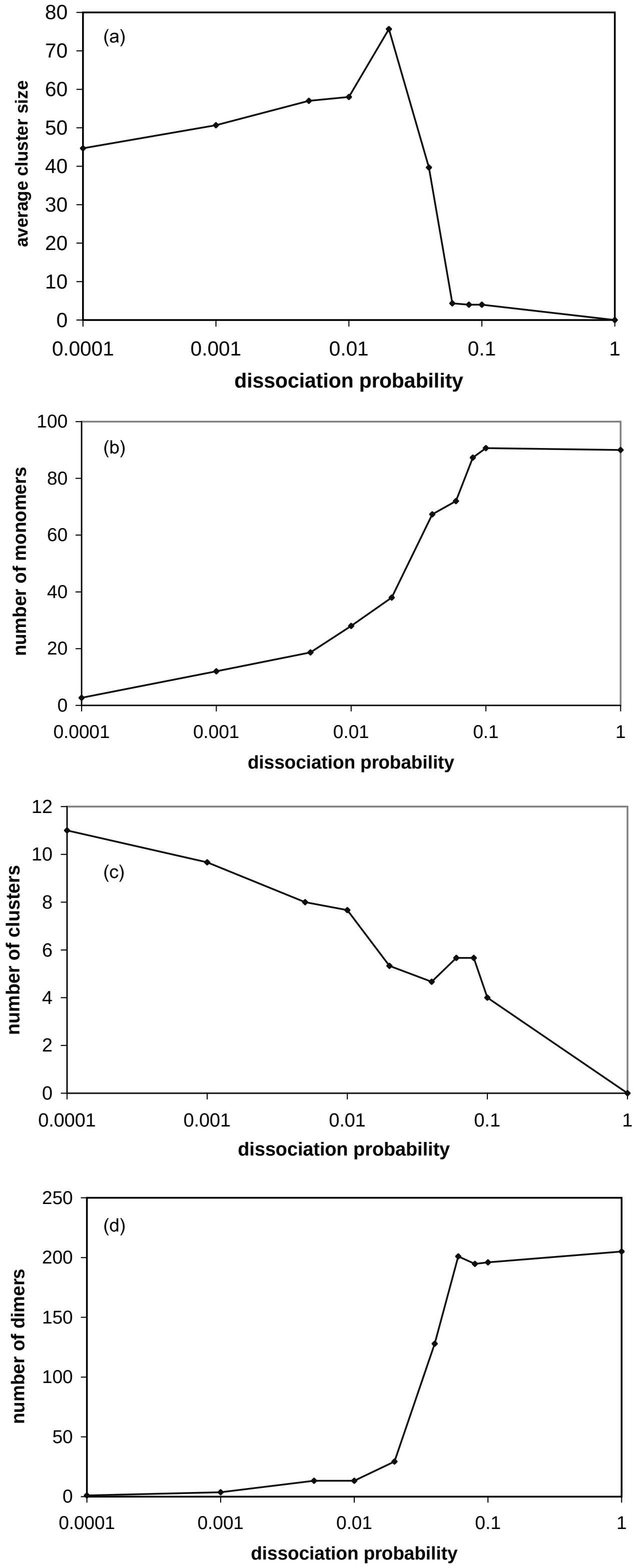

Fig. 18. Characteristics of the system at equilibrium with various dissociation rates.