Universiteit Leiden
The Netherlands

Opleiding Informatica

Using Heterogeneous Primordial Particle Systems

to Analyse Life-like Behaviour.

Vincent Prins

Supervisors:
Mike Preuss and Walter Kosters

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

20/07/2021

Abstract

This thesis presents Ursoup, a Primordial Particle System (PPS) simulator based on work by Schmickl et al. At a specific parameter set, Schmickl et al. found a PPS that exhibits life-like behaviour in the form of simple cellular life, capable of reproduction. However, the particles in their PPS are homogeneous in nature. Ursoup is capable of running heterogeneous PPS simulations. We have investigated the effects of the β, speed, and radius parameters of a PPS, using heterogeneous two-species simulations, where we change one of these parameters in one of the two species. Doing so, we attempt to gain a better understanding of the functions of specific particle types participating in the life-like behaviour. In cell nuclei, for instance, we find particles with high β and speed values, whereas in cell walls, we find the particles to have low β and speed values. Additionally, we found cells that display a strict division of roles between the two species, where species A homogeneously populates the cell nucleus, and species B homogeneously forms the cell wall. Yet these two species come together to form one heterogeneous cell.

Contents

1 Introduction

The universe can be rather chaotic at times, and yet it gave rise to the complexity of life. Complexity arising from chaos is called emergence. This emergence is fuelled by a set of rules. Gravity, for example, is a rule that gathers matter in large dense clumps, forming stars. Thus, rule-based emergence is key to complexity.

In the computer sciences, emergence is often seen in the field of artificial life (AL). AL is characterised by a chaotic initial condition, and a (often small and simple) set of rules. For instance, Boids shows emergent behaviour in the form of flocks of birds, guided by a small rule-set [Rey87].

Another example of AL is a so-called Primordial Particle System (PPS), presented by Schmickl et al. [SS15, SSC16]. This system demonstrates a rich variety of emergence. At a specific parameter setting, a self-reproducing cell-like structure has been found that displays a complex growth cycle. Particles gather in small clumps, accumulate more particles and eventually expand into cells. These cells may live on for quite a while, replicate, or die out. Overall, the cell population does grow. It is unclear, though, as to why this behaviour emerges at precisely that specific parameter setting.

In the "classic" Primordial Particle System, all particles follow the same rules. In that sense, it is a homogeneous system. Emergence is however not limited to homogeneous systems. After all, life itself on earth is very much heterogeneous, not homogeneous. An example of heterogeneous emergence is *Clusters*, a Magnetic Particle System, where multiple species of particles attract and repel each other [Ven17]. This, too, facilitates the emergence of complex structures.

The research on heterogeneous PPS is severely limited. Thus, the aim of this research is to use heterogeneity as a tool to gain a better understanding of the life-like structures in PPS.

1.1 Contributions

As Schmickl et al. did not provide any code, we decided to create our own PPS simulator. Therefore, this thesis presents Ursoup (*ur-* meaning primordial). Ursoup is a PPS simulator program with a graphical window (Figure 1), logging and screenshot functionality, as well as a headless mode. It supports both heterogeneous and homogeneous simulations.

Ursoup is available on GitHub at `https://github.com/ursoup/ursoup`.

1.2 Thesis overview

This chapter contains the introduction; Section 2 discusses related work; Section 3 describes the workings of Ursoup; Section 4 describes the experiments and their outcome; Section 5 presents the conclusions. Lastly, also we discuss possible further research in that section.

This bachelor thesis is the result of research by Vincent Prins under the supervision of Mike Preuss and Walter Kosters at the Leiden Institute of Advanced Computer Science (LIACS).

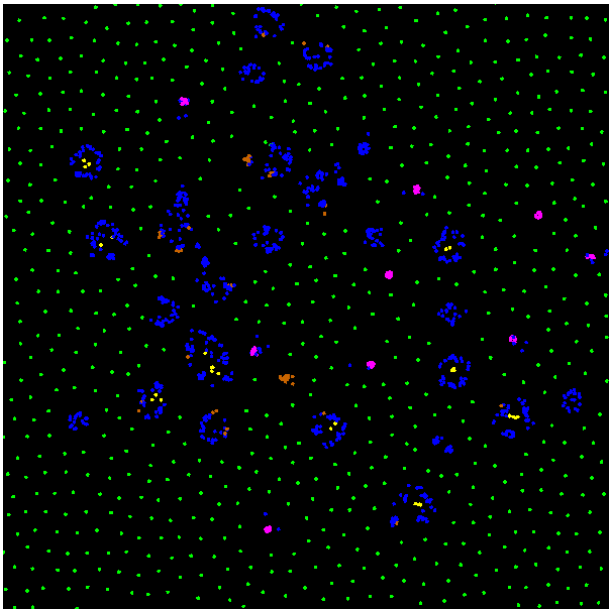

Figure 1: The graphical window of Ursoup.

2 Background

In this section we will discuss two types of *Particle Systems* (PS): a *Primordial PS* and a *Magnetic PS*. We argue that these systems can be considered simple programs. These PS also demonstrate the difference between homogeneity and heterogeneity. Lastly, we will discuss the use of heterogeneity in the analysis of life-like behaviour.

2.1 Simple programs

In *A New Kind of Science*, Wolfram discusses how many simple programs are counter-intuitively capable of producing complex behaviour [Wol02]. The workings of any *simple program* can be described with a couple of rules, a few lines of code or a simple graphic. The argument goes that these programs cannot program for all the resulting complexity, thus emergence must be at play.

An example of a simple program is Conway's Game of Life, a cellular automaton [Gar70]. Game of Life is set in a 2D world consisting of discrete cell spaces, that can be alive or dead. A simulation starts out with an initial configuration of alive and dead cells. The fate of each cell in the next time step depends on its neighbours in the current time step. Dead cells come alive when they are surrounded by three alive cells, including the diagonals. Living cells stay alive only when they have two or three neighbours, if not, they die. The result is that some initial configurations die out, some form stable patterns whether stationary or cyclic, and others continue indefinitely. An example of the latter is the so-called *glider* pattern that indefinitely "glides" through the world.

The two particle systems that will be discussed hereafter are both moderately simplistic in nature. Admittedly, they are more complex than Game of Life, but the interactions between particles are local, the number of parameters is limited and the rules set is still quite small. Unlike cellular automata, however, they operate in continuous space, as opposed to discrete space.

2.2 Primordial Particle Systems

The basis of any Primordial Particle System (PPS) is its particles. Each particle has a position and a rotation (also called ϕ). At each time step of a PPS simulation, a set angle called α is added to the rotation. After the rotation has been applied, the particle moves forward by a set distance v, also called its *speed*. Each particle has a perception *radius* in which it detects neighbours. At each time step it will count up the number of neighbours, denoted as N. This value is multiplied by a factor called β and is also added to the rotation. Thus, if a particle has no neighbours, β has no effect. Likewise, a particle with two neighbours wil turn more than a particle with just one neighbour.

Additionally, each particle keeps track of the number of neighbours to the left and the right of its current heading. It will always turn towards the side with more particles. If both sides have an equal number of particles, it will turn to the right. All of this is summed up in Figure 2.

```
#Pseudo-code of PPS
loop foreach timestep {
  loop foreach particle {
    L = (count other particles in left semicircle with radius r)
    R = (count other particles in right semicircle with radius r)
    N = L + R
    delta_phi = alpha + beta * N * sign(R - L)
    rotate delta_phi #positive values: rotate to the right side
    move-forward v
  }
}
```

Figure 2: The pseudocode describing the implementation of a PPS as seen in [SSC16].

Each PPS simulation is set in a 2D torus world, where the edges of space wrap around to the other side. The PPS system has however also been shown to work in three dimensions [SS19]. A simulation is run at a certain *particle density*, and the number of particles stays the same during the entire simulation. The position and rotation of all particles are initialised randomly.

At the specific parameter set of $[radius = 5, \alpha = 180, \beta = 17, speed = 0.67]$, Schmickl et al. found life-like cell structures that exhibit a complex cycle of life. Note that at $\alpha = 180$ neighbourless particles are effectively motionless, as each time step they undo the movement of the previous time step. The cycle of life is depicted in Figure 3. The colours of the particles are based on the number of neighbours in the perception radius of each particle (Table 1). Green particles are called *nutrients* and have 13 or fewer neighbours. When these nutrients clump up they form *premature spores*. At 14 or 15 neighbours, these premature spores turn brown. Once they grow further by attracting more nutrients, they develop into *mature spores*. The particles in mature spores are magenta and have more than 15 neighbours, measured in a radius of $r = 1.3$, unlike the other colours that are measured at the full perception radius (in this case $r = 5$). Once the spores have accumulated enough nutrients, they expand into ring-shaped structures. These rings are like the *cell walls* of a cell. Particles in the cell walls are blue. However, these cells can still grow, and if they grow enough, a *cell nucleus* will start to appear. Once the cell nucleus has emerged, we call the structure a *cell*, as it has the characteristic cell wall and cell nucleus of a simple single-cell organism (albeit simplified). The particles in the cell nucleus have over 35 neighbours, and turn yellow. During its growth the cell also takes on different shapes. Cells eventually tend to die or

reproduce. However, the overall "population" of cells does increase. Note how this is different from an increase in particles. The number of particles always stays constant, it is the number of particles that exist in cellular structures that increases. Lastly, it is unclear as to why this behaviour emerges at this specific parameter set.

Colour	Function	Neighbours
yellow	cell nucleus	$N_{t,r=5} > 35$
blue	cell wall	$15 < N_{t,r=5} \leq 35$
brown	premature spores	$13 < N_{t,r=5} \leq 15$
magenta	mature spores	$N_{t,r=1.3} > 15$
green	nutrients	$N_{t,r=5} \leq 13$

Table 1: The function and number of neighbours of each particle type. Here, $N_{t,r}$ represents the number of neighbours N, at time step t in radius r. Note how magenta uses a different radius.

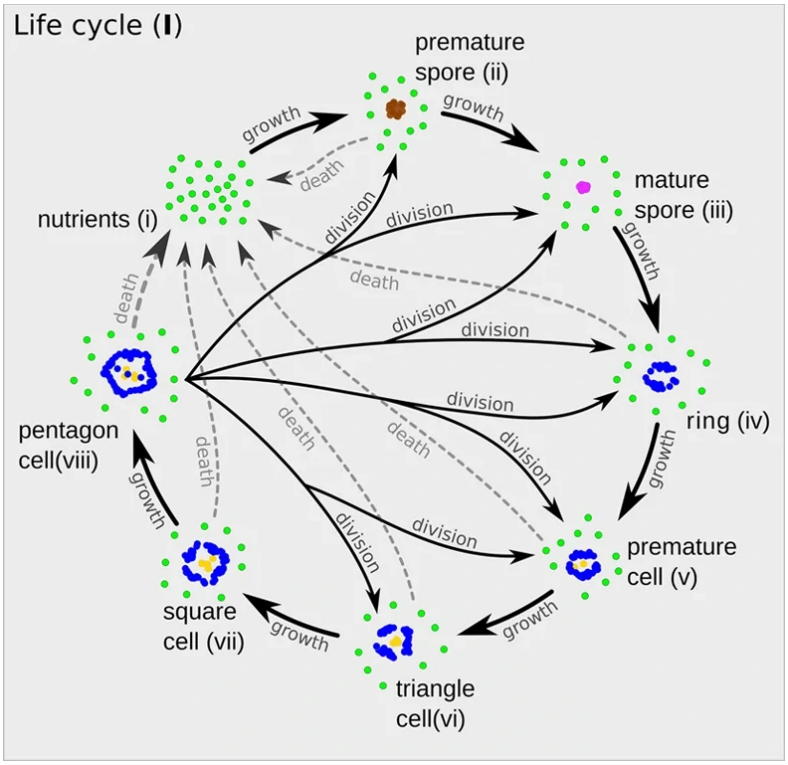

Figure 3: The cycle of life as seen in [SSC16].

2.3 Homogeneity and heterogeneity

The PPS, as presented by Schmickl et al., is a homogeneous system. Something is homogeneous when all of its components belong to the same kind. Thus, if something has a homogeneous make-up, swapping two of its components should have no effect. So, if all particles in a PPS are of the same species, have the same attributes and follow the same rules, we can say that the PPS is homogeneous. Heterogeneity, on the other hand, is simply the act of not being homogeneous. A

water molecule, for example, is heterogeneous. Its consists of two component types, oxygen and hydrogen, not just one.

2.4 Magnetic Particle Systems

One example of a heterogeneous system, is Ventrella's CLUSTERS (Figure 4) [Ven17]. Clusters is best characterised as a heterogeneous Magnetic Particle System (MPS). All particles belong to one of many species, signified by their colour. Each species has a separate attraction value towards every other species. If that value is negative, it turns into repulsion. The particles only interact with each other within a set radius. These interactions result in the emergence of complex, often moving structures. Clusters also allows small groups of particles to be moved around by the mouse cursor.

Unfortunately, the code is not available. However, the YouTuber CodeParade, known on GitHub as HackerPoet, tried to recreate this simulation and made the code available [Hac18]. For this recreation, called PARTICLE LIFE, HackerPoet also mentions that particles that chase each other (due to A being attracted to B, but B being repulsed by A) could infinitely build up speed. Thus, friction was introduced to counter this. Although Ventrella does not mention friction, seeing as Clusters does not display infinite speed build-up, some speed limitation is presumable at play too. Upon HackerPoet's basis, Peterson created a version that runs in a web browser (Figure 5) [Pet21].

Unlike PPS and Particle Life however, Clusters does not operate in toroidal space. The result is that most structures end up brushing against the walls, or even crashing apart.

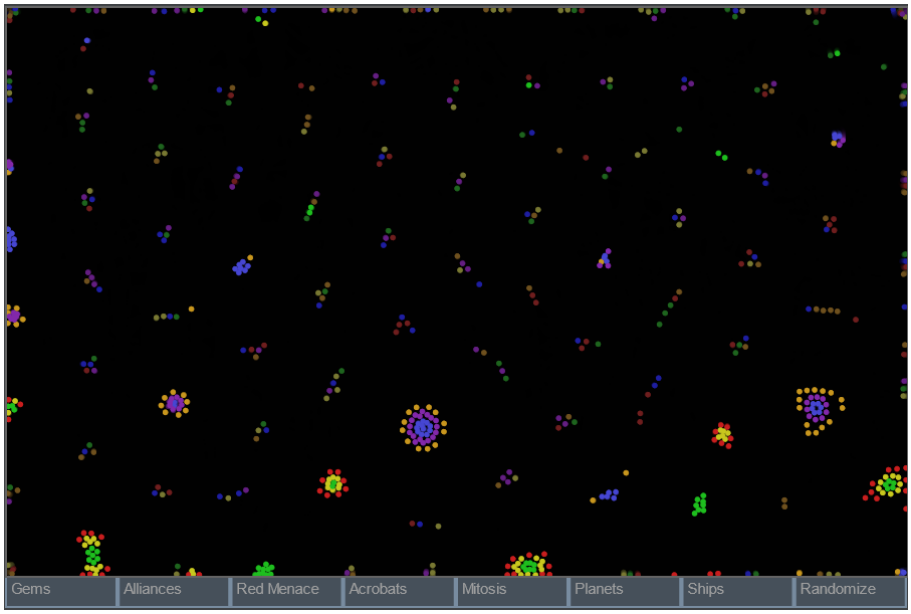

Figure 4: Interface of J. Ventrella's CLUSTERS. Here, the program is displaying the *Gems* setting. The larger structures pictured are predominantly stationary. Available at http://www.ventrella.com/Clusters/.

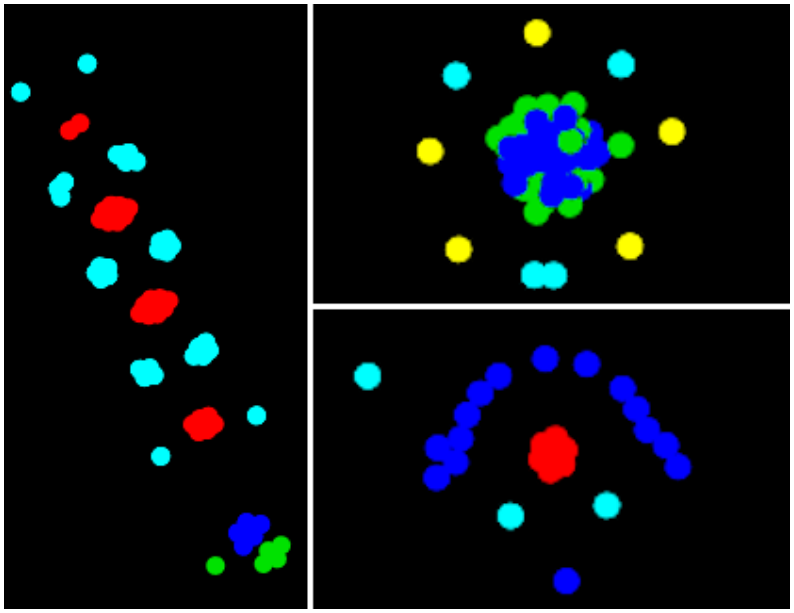

Figure 5: Examples of structures that emerge in PARTICLE LIFE. From Peterson's web browser version available at https://fnky.github.io/particle-life/.

2.5 Heterogeneity in Primordial Particle Systems

Heterogeneity in PPS seems to be mostly unexplored. We could only find one example. A YouTuber by the name of TheRainHarvester showcases a name-less heterogeneous PPS simulator in multiple videos. Unfortunately, no code was made available. In one of these videos, TheRainHarvester explores separating particles into species denoted by different colours, and changing the radius of one species such that it differs from the rest [The20]. In Figure 6, the red species has a radius of 10, whereas the other particles have a radius of 20. This being the case, all non-red particles actually belong to the same species, despite their different colours. The red particles form a hexagonal pattern, surrounding the other species that are pushed towards the centre of each hexagon. This shows that complex behaviour can emerge from heterogeneous PPS.

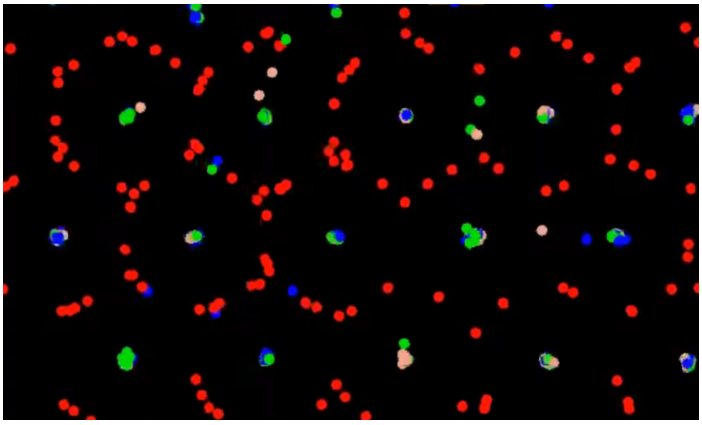

Figure 6: A heterogeneous PPS simulation made by TheRainHarvester, in which the red particles have a smaller radius than the other colours. The name of the simulator program used is unknown.

Additionally, in another video TheRainHarvester explores the idea of "visibility" [The19]. This means that certain species can or cannot see other species, including being able or unable to see themselves. An example of behaviour that emerges from this method can be seen in Figure 7. The behaviour shown resembles a heart and blood vessel system. Large groups of particles form heart-like structures surrounded by a clear wall acting as a boundary. The surrounding vessels are filled with individual "blood" particles, similar to the green nutrients in the classic PPS.

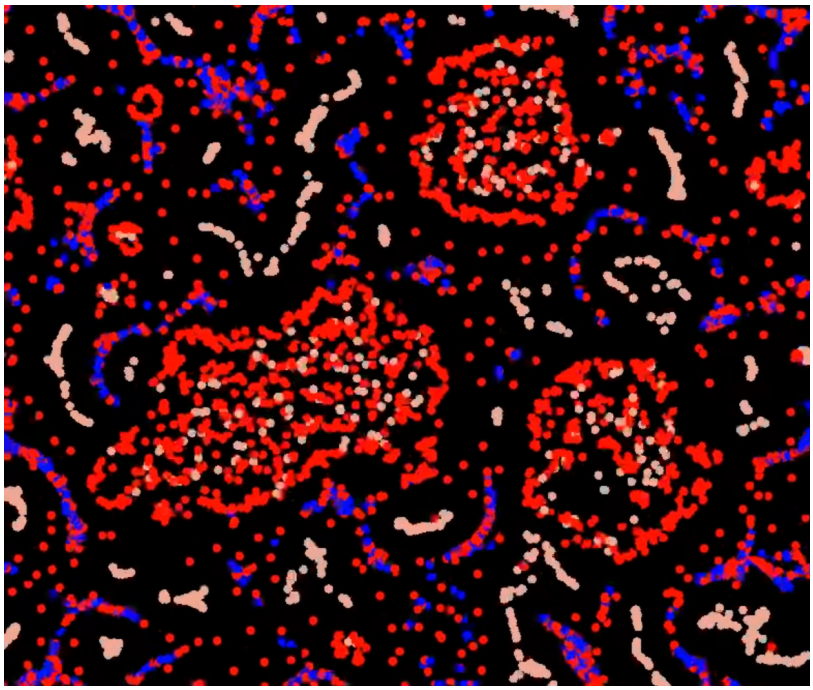

Figure 7: A heterogeneous PPS simulation with "visibility' made by TheRainHarvester, where some species cannot see other species or even themselves. The name of the simulator program used is unknown.

2.6 Heterogeneity as a tool

It is unclear as to why the parameter set discovered by Schmickl et al. produces such intricate life forms. In this thesis we want to further analyse the role that the values of β, speed and radius have on each particle type (i.e., cell walls, nuclei, spores, etc.). We will not be changing α, as $\alpha = 180$ is what creates the unique, stationary behaviour of nutrients that are vital to the system. The main idea is to change one of these value for half of the particle population. Doing so, we create a heterogeneous two-species system. The species at $[radius = 5, \alpha = 180, \beta = 17, speed = 0.67]$ shall be referred to as the *base species* as it forms the basis of our investigations. All other species will be called *modified species*. The usefulness of heterogeneity is that by changing one value at a time, we can see what role the modified species takes on with regard to the base species (or vice versa). We can measure this by looking at the number of particles of each type/colour. As the colours of the particles represent the density of the particle's perception radius we shall call this measurement the particle density distribution (PDD).

For example, if increasing the β of the modified species results in more yellow particles of the

modified species, we can conclude that cell nuclei (yellow) have a preference for containing particles with a higher β. In this way, we hope to gain a better understanding of each particle type.

3 Ursoup

This thesis presents Ursoup, a Primordial Particle System simulator. In this section we shall discuss the implementation details of our program. Additionally, we will go over the features that Ursoup offers, and their usage.

Ursoup is has been made available on GitHub at https://github.com/ursoup/ursoup.

3.1 Implementation

The basis of our implementation is a Boids simulator written by R. Strauss that we have adapted heavily [Str21]. The program is written in C++ and makes use of SFML (*Simple and Fast Multimedia Library*). The `Cxxopts` library is used as its command line option parser [Bec21]. Additionally, Strauss has implemented a *k-d tree* to speed up searching for neighbours. We have adapted the k-d tree search function to work with toroidal space. Furthermore, we have implemented the ability to run homogeneous and heterogeneous PPS simulations, as well as logging information and screenshots.

Now, let us take a closer look at Ursoup in action. The program starts by parsing the command line options and reading the config file. From the config file it constructs a list of species. It then starts spawning particles at random positions, such that the average density of the simulation approximates the set particle density. All species are made to occur equally frequent. The particles store three core pieces of information: position, rotation, and species type. All species-dependant properties can be deduced from the species type, so these properties are not stored in the particle, but globally. Species-dependant properties are α, β, radius and species colour (as opposed to density colour, see Section 3.2.3). Additionally though, particles keep track of the number of "regular" neighbours (within their full perception radius) and close neighbours (always within $r = 1.3$, regardless of their perception radius). This facilitates the neighbourhood colouring. For the colouring we attempted to use the colouring conditions described by Schmickl et al. However, the magenta condition was never triggered. As it turns out, the magenta condition should be the first condition to be checked, not the last one, otherwise no magenta particles seem to appear at all. Hence, a revised list of colour conditions is:

```
if (N_{t,r=1.3} > 15)
    return magenta;
else if (15 < N_{t,r=perception} ≤ 35)
    return blue;
else if (N_{t,r=perception} > 35)
    return yellow;
else if (13 < N_{t,r=perception} ≤ 15)
    return brown;
else
    return green;
```

Once all particles have been spawned, the update loop starts running. Firstly, a k-d tree is constructed. Secondly, the rotation of the particles is updated. This is where the k-d tree is used, as particles need to know how many neighbours they have to correctly update their rotation. Thereafter, the particles move forward by their speed value. At this point, it is time to start drawing on the screen, for which we need colours. The correct density colouring is selected for each particle based on their neighbour count. These values also form the particle density distribution (PDD). If the logging interval has been reached, the program now logs the PDD to the output file. Lastly, everything is drawn to the screen. However, the graphical window skips drawing every other frame. This is to avoid excessive flashing at $\alpha = 180$. Doing so shows neighbourless particles as stationary. The update loop is then repeated. The loop exits once the `--exit_after` value has been reached and the program shuts down.

3.2 Usage

In this section, we will explain in depth the function of each command line option and keyboard option that Ursoup features. For quick reference, these are summarised in Table 2 and Table 3.

Option	Default	Function
`--simulation_width`, or `height`	150 for both	defines the size of the simulation world
`--window_width`, or `height`	600 for both	defines the size of the graphical window
`--draw_size`	2	defines the size at which particles are drawn
`--exit_after`	$100,000$	exits program after the set number of time steps
`--headless`	off	disables the graphical window
`--particle_density`	0.08	approximate particle density
`--config_file`	`config.ini`	defines what file to use as species configuration
`--log_interval`	500	number of time steps between PDD logging
`--log_file`	`log.csv`	defines what file to log to
`--log_screenshot`	off	whether or not to log screenshots

Table 2: The function of each command line option in Ursoup.

Option	Function
space	pause simulation
F	move forward by one frame when paused
S	save screenshots of current time step
C	change colour mode

Table 3: The function of each keyboard option in Ursoup.

3.2.1 Running simulations

By default, simulations are run in a 150×150 world. This can be adjusted with `--simulation_width` and `--simulation_height`. The size of the graphical window is separate from the simulation size. The relevant options here are `--window_width` and `--window_height` (default: 600×600). Due to the scaling factor, it might be desirable to increase the size of the particles. This is what the

option `--draw_size` provides. All simulations exit after n time steps. The exit point can be set with `--exit_after` (default: $100,000$ time steps). Lastly, the graphical window can be turned off with the option `--headless`.

The particle density of a simulation is measured in the number of particles in a 1×1 area. The density is set using `--particle_density` (default: 0.08).

When the graphical window is used, the simulation can be paused by pressing space. Pressing F while paused will progress the simulation by one frame (equal to two time steps). Screenshots can also be taken at any time by pressing S. This produces two screenshots: one for each colour mode (Section 3.2.3).

In the graphical window, the left mouse button can be used to add more particles to the simulation at the location of the mouse cursor.

3.2.2 Configuration files

The configuration file determines what species will be run in the simulation. Each line corresponds to one species and values are separated by spaces. Thus, a single line file constitutes a homogeneous simulation. The format is as follows:

`[speed] [perception radius]` $[\alpha]$ $[\beta]$ `[R] [G] [B] [A]`

By default, Ursoup will try to open `config.ini`. Custom config files can be loaded in with the option `--config_file my_file`. An example two-species config file would look like this:

```
0.67 5 180 17.00 255 0   0   255
0.67 5 180 14.00 255 255 255 255
```

3.2.3 Colour modes

Ursoup's graphical interface supports two colour modes: species mode and density mode (Figure 8). Pressing C switches the view. In density mode, all particles are coloured based on their neighbourhood density. This colouring matches that of Schmickl et al, as long as the perception radius is equal to 5. For instance, a perception radius of 50 will (most likely) cause all particles to display as yellow. In species mode, the particles take on the colour designated to their species as specified in the configuration file.

3.2.4 Logging

Ursoup logs the particle density distribution (PDD) at set intervals, measured in time steps. This interval can be set using `--log_interval` (default: 500). The PDD is the average number of green, yellow, etc., particles over the last n time steps. The PDD of each species is tracked separately. These values are logged in a csv file that can be set using the option `--log_file` (default: *log.csv*).

The output for one species may look like this:

```
Time,Yellow,Blue,Brown,Magenta,Green,
500,0,0,0.066,0,849.934,
1000,0,0.006,0.072,0,849.922,
```

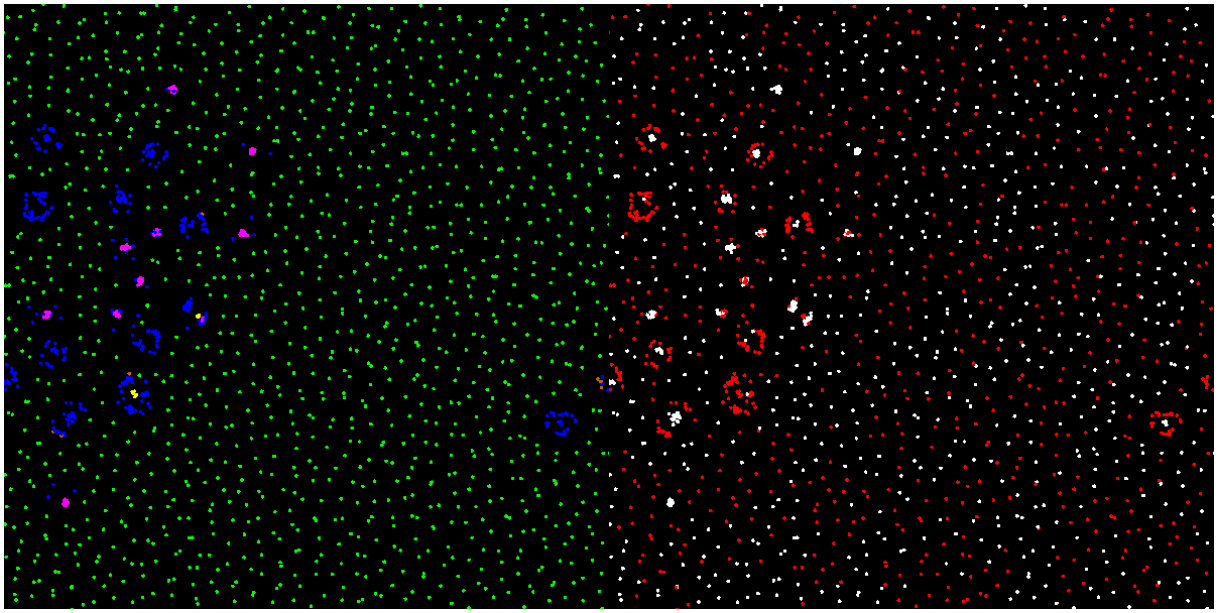

Figure 8: A heterogeneous simulation with two species, red and white;
left: density colour mode; right: species colour mode.

Additionally, Ursoup can output screenshots of the simulation (even in headless mode) at each log interval. The option `--log_screenshot` enables this behaviour.

4 Experiments

In this section, we shall be investigating what effect a change in β, speed and radius has on the density of each species in a two-species PPS. For each parameter, we want to analyse the effect of values lower and higher that the base value. Thus, the following experiments will go through a range of values, where the base value forms the centre value of each range (although for speed, 0.67 is rounded up to 0.675). Each range is kept somewhat small, such that the simulations could be performed within a reasonable amount of time.

We shall be analysing the effects of each change in two aspects. The first aspect is the particle density distribution (PDD) of each species. From the PDD of each species we should be able to identify which species, base or modified, occurs more frequently for each particle type. With that, we can establish a "preference" for either higher or lower values of β, speed and radius with regard to the base value.

The second aspect is a visual inspection of the spore and cell-structures that form. This way we can take a closer look at what kind of structure form and in what nature (homogeneous or heterogeneous), as that cannot be deduced from the PDD. We hypothesise that any effects that may occur due to changes in β, speed or radius will be most pronounced in the extremity values of each value range. Thus, for each experiment we will only inspect the two extremities.

At the standard particle density of 0.08 used by Schmickl et al., the simulations would not always form cell structures. Therefore, each simulation shall be run at a particle density of 0.1. This assures

that cells will form within the first couple of time steps. The simulation then runs until time step $50,000$. However, if we were to include data from the early stages of the simulation, we would be mostly looking at an undeveloped population, and nutrients would be overly represented. Thus, we shall be looking at a *post-initial* stage of the simulation. We have determined that after 5000 steps, the simulation has stabilised enough, so the results discard everything before that. The PDD is logged every 500 time steps. Lastly, we always use a simulation world size of 100×100.

Unfortunately, Ursoup occasionally stopped updating all particles, yet kept logging the PDD for the rest of the simulation until it reached the *exit after* value. When this bug occurred, it was obvious from the repeated entries in the log file. During our experiments we had this happen occasionally. Luckily, this could easily be solved by rerunning the simulation.

Lastly, In all screenshots of Ursoup simulations in species colour mode, we have used the colour red to denote the base species and white to denote the modified species.

4.1 Experiments with β

In this experiment we gave the modified species the values $\beta = 14.00, 14.25, \ldots, 19.75, 20.00$. For the base species, β is 17.

4.1.1 Particle density distribution

In the following figures, we have split up the PDD into its separate colours. By looking at which species occurs more frequently at which β, we can determine whether each particle type has a consistent preference for lower or higher β–values.

In Figure 10 and Figure 12 we see that blue and magenta particles are more likely to belong to the species with the lower β. In Figure 9, Figure 11 and Figure 13 on the other hand, we see that yellow, brown and green particles are more likely to belong to the species with the higher β. In conclusion, blue and magenta particles prefer low betas, whereas yellow, brown and green prefer higher betas (see Table 4).

4.1.2 Visual inspection

We shall now continue with the visual inspection. In Figure 14 and Figure 15 a screenshot of a simulation using the two extremes $\beta = 14.00$ and $\beta = 20.00$ is shown. Of particular note in these simulations are the heterogeneous *proto-cell-like* structures highlighted in Figure 16. These cells are not quite as large as the regular cells (hence the denomination *proto*), and do not contain any yellow particles. Additionally, we see a strict division of roles between the two species, where one homogeneously populates the cell nucleus and the other species homogeneously populates the cell wall. Taking Figure 14 ($\beta = 14$) as an example, the modified species homogeneously forms the cell nuclei of the proto-cells. Extrapolating this behaviour, one would expect the modified species to also dominate the cell nuclei of the regular-sized cells, and by extension the yellow particles. Counter-intuitively however, the opposite is true as seen in Figure 9. Lastly, the spores in both simulations occur homogeneously as well as heterogeneously.

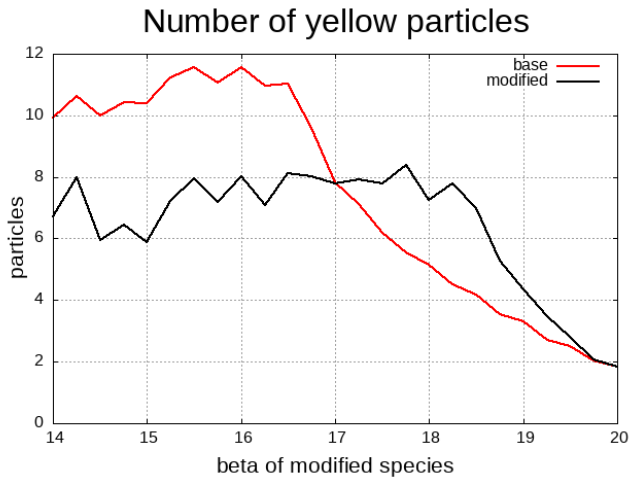

Figure 9: Average number of yellow ("cell nucleus") base and modified particles per time step.

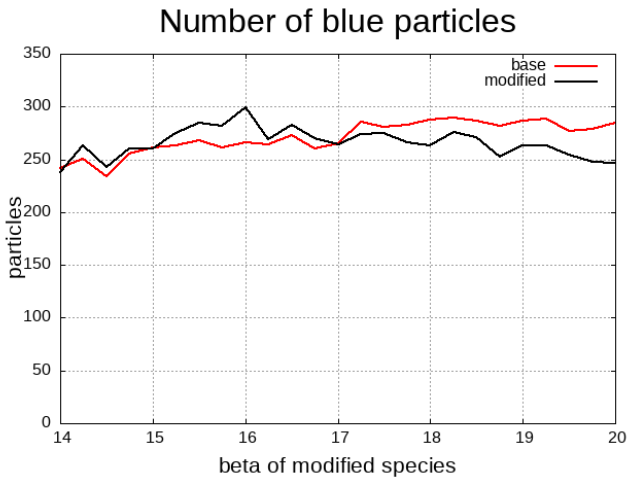

Figure 10: Average number of blue ("cell wall") base and modified particles per time step.

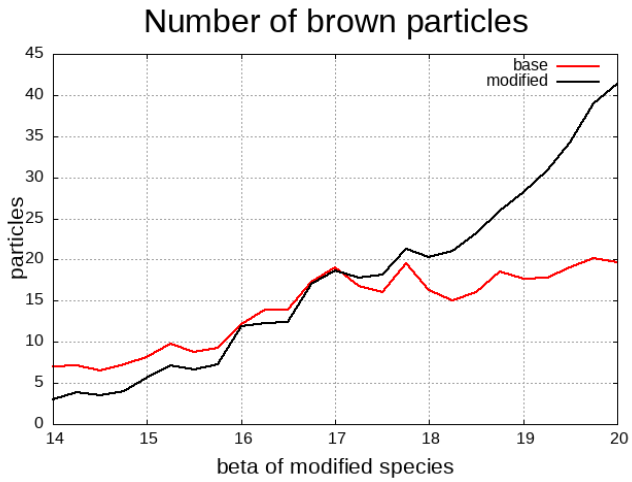

Figure 11: Average number of brown ("premature spore") base and modified particles per time step.

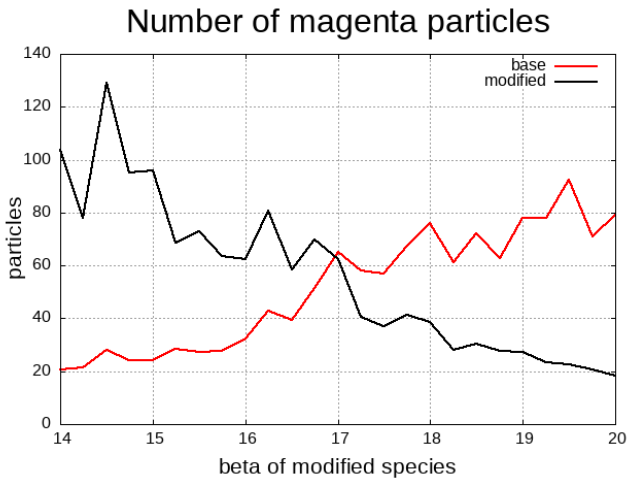

Figure 12: Average number of magenta ("mature spore") base and modified particles per time step.

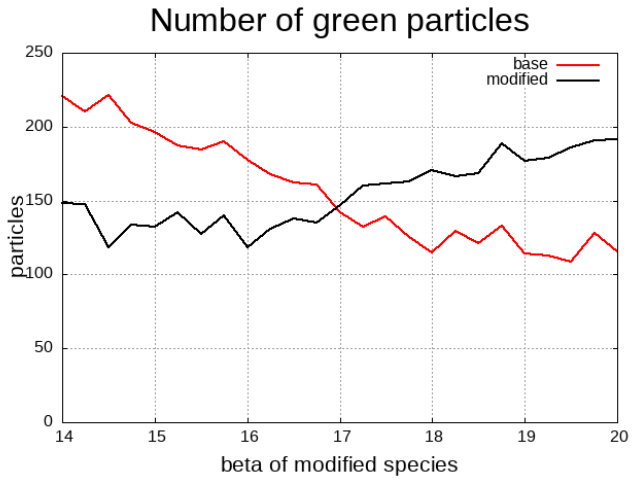

Figure 13: Average number of green ("nutrient") base and modified particles per time step.

Colour	Function	β
yellow	cell nucleus	high
blue	cell wall	low
brown	premature spores	high
magenta	mature spores	low
green	nutrients	high

Table 4: Preferred β for each particle type.

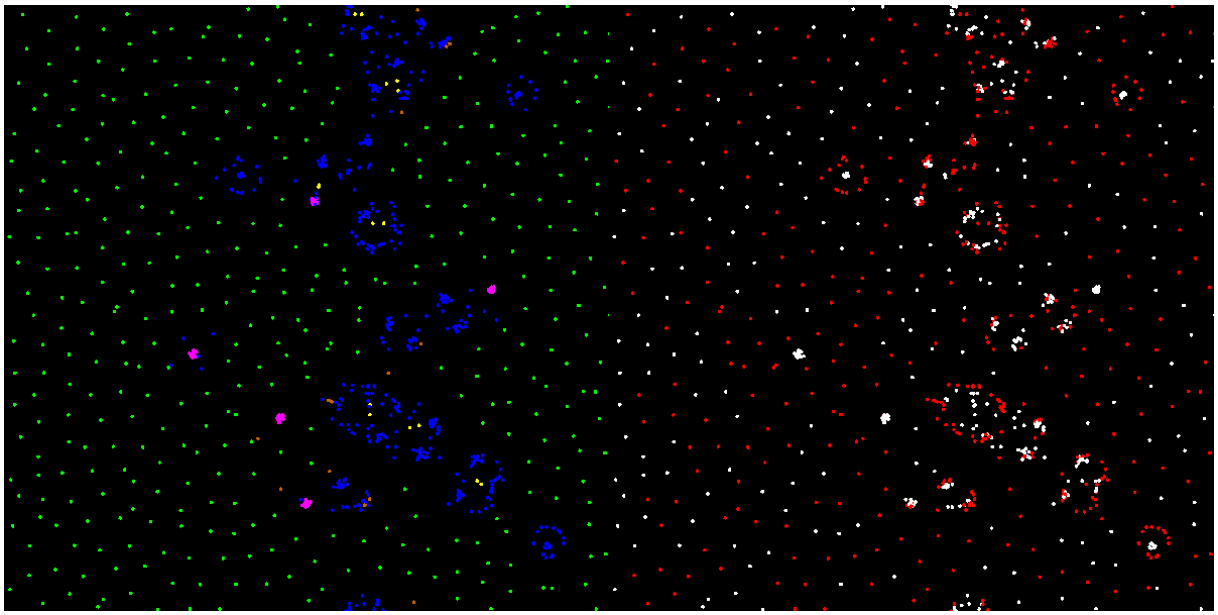

Figure 14: Screenshot of a simulation at $t = 8059$ with the modified species at $\beta = 14$.

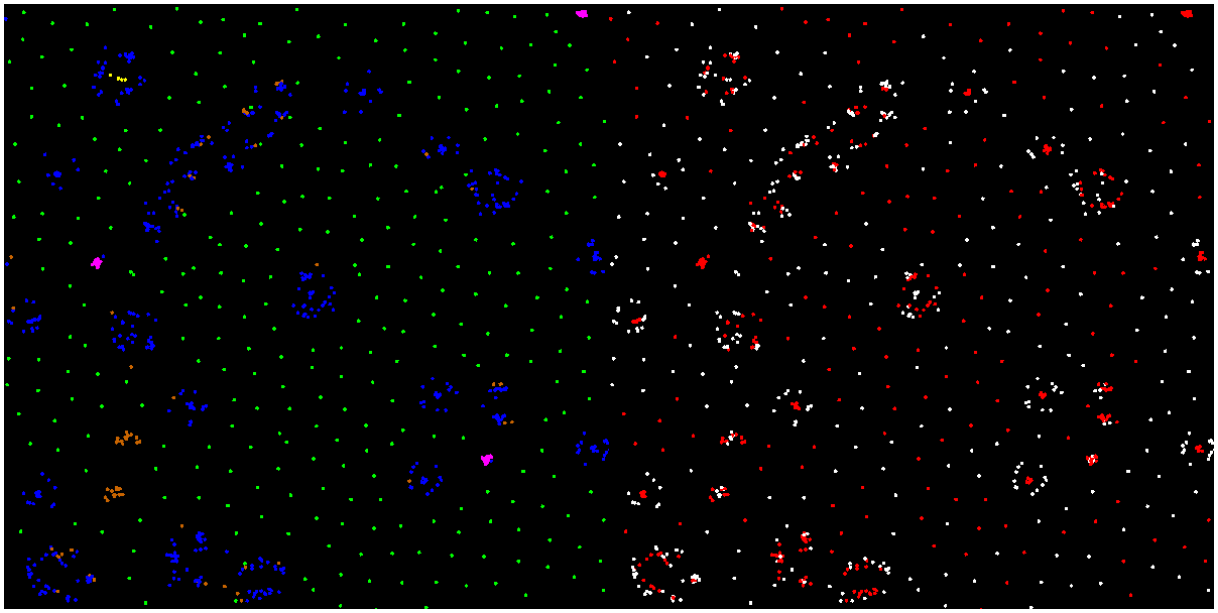

Figure 15: Screenshot of a simulation at $t = 8610$ with the modified species at $\beta = 20$.

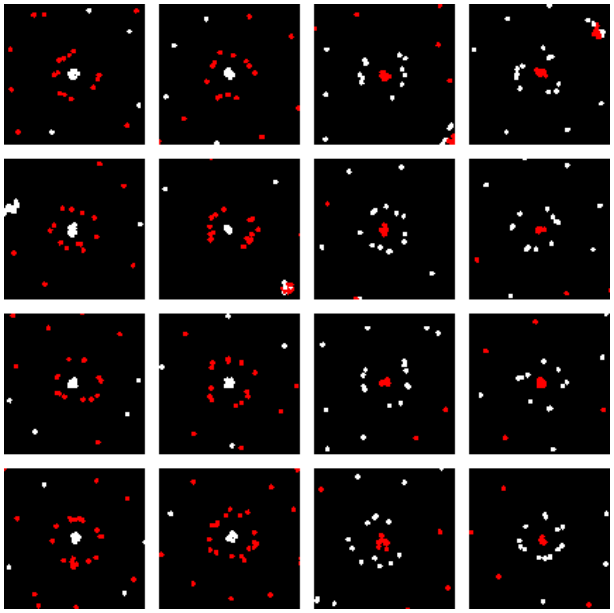

Figure 16: Close-up of the heterogeneous proto-cells with a clear division of roles; left: modified species at $\beta = 14$; right: modified species at $\beta = 20$.

4.2 Experiments with speed

In this experiment we gave the modified species the values $speed = 0.400, 0.425, \ldots, 0.975, 1.000$. For the base species, speed is 0.67.

4.2.1 Particle density distribution

In the following figures, we have split up the PDD into its separate colours. By looking at which species occurs more frequently at which speed, we can determine whether each particle type has a consistent preference for lower or higher speed values.

In Figure 18 and Figure 20 we see that blue and magenta particles are more likely to belong to the species with the lower speed. By contrast, in Figure 17, Figure 19 and Figure 21 we see that yellow, brown and green particles are more likely to belong to the species with the higher speed. The difference is rather slim though for the yellow and brown particles. So overall, blue and magenta particles prefer low speeds, whereas yellow, brown and green prefer higher speeds (see Table 5).

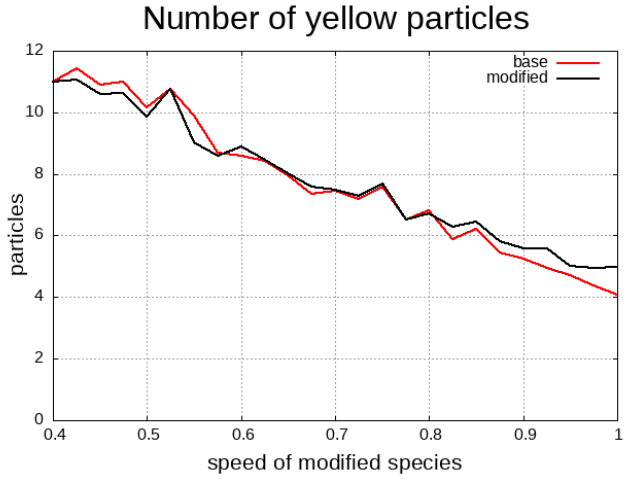

Figure 17: Average number of yellow ("cell nucleus") base and modified particles per time step.

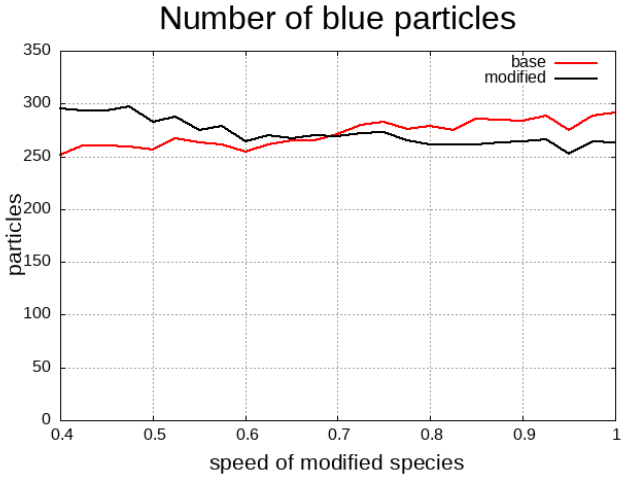

Figure 18: Average number of blue ("cell wall") base and modified particles per time step.

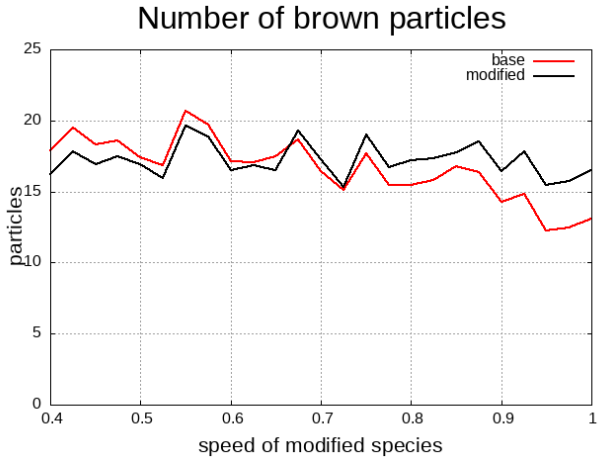

Figure 19: Average number of brown ("premature spore") base and modified particles per time step.

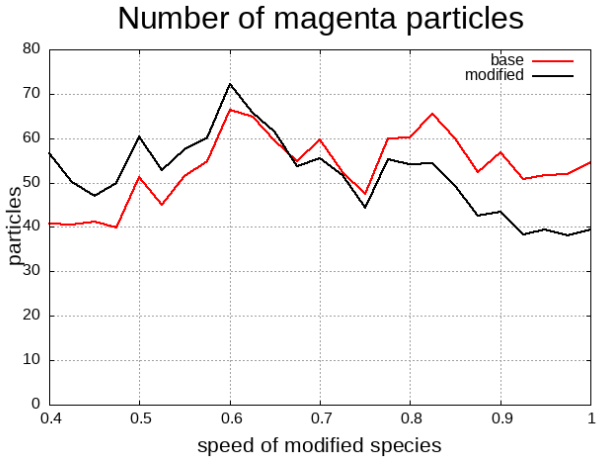

Figure 20: Average number of magenta ("mature spore") base and modified particles per time step.

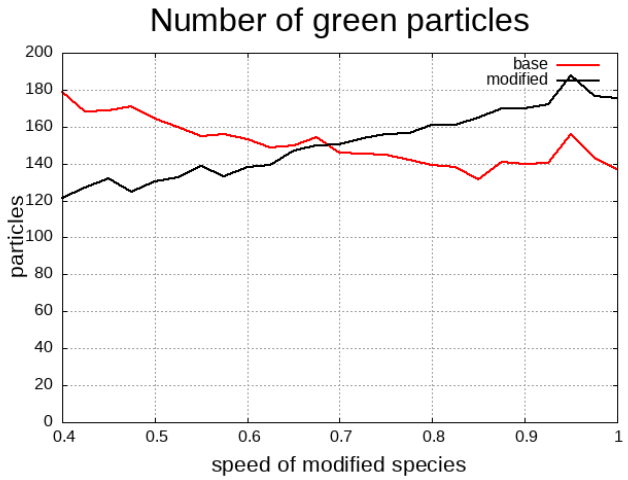

Figure 21: Average number of green ("nutrient") base and modified particles per time step.

Colour	Function	Speed
yellow	cell nucleus	high
blue	cell wall	low
brown	premature spores	high
magenta	mature spores	low
green	nutrients	high

Table 5: Preferred speed value for each particle type.

4.2.2 Visual inspection

We shall now continue with the visual inspection. In Figure 22 and Figure 23 a screenshot of a simulation using the two extremes $speed = 0.400$ and $speed = 1.000$ is shown. In both figures, cell walls, cell nuclei and spores only seem to occur heterogeneously. There is however no clear division of roles between the two species.

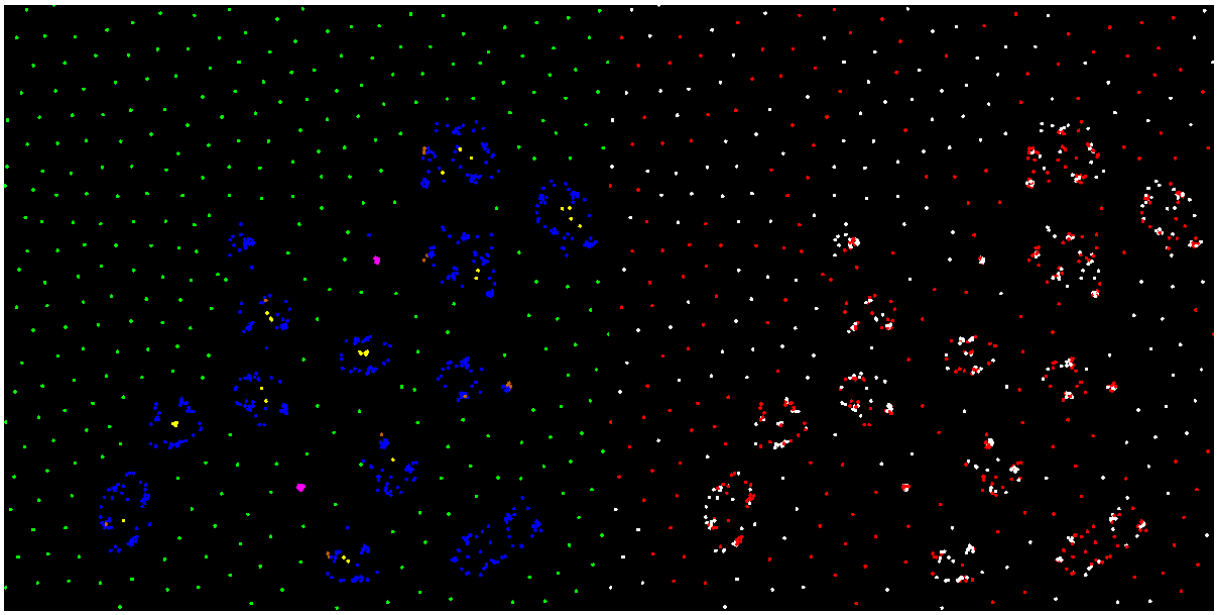

Figure 22: Screenshot of a simulation at $t = 6285$ with the modified species at $speed = 0.4$.

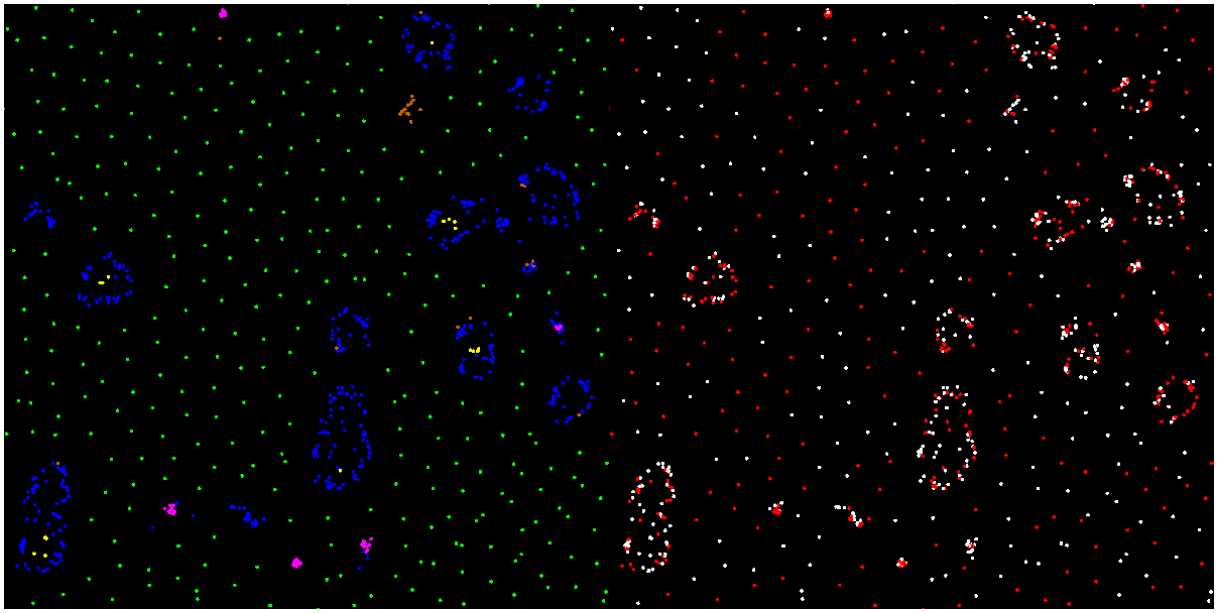

Figure 23: Screenshot of a simulation at $t = 6903$ with the modified species at $speed = 1$.

4.3 Experiments with radius

In this experiment we gave the modified species the values $radius = 3.00, 3.25, \ldots, 6.75, 7.00$. For the base species, radius is 5. Of note is that the code that determines the colour of particles uses the perception radius of that particle. A particle with a larger perception radius has an 'unfair' advantage in that it will turn yellow/blue/etc. much more easily. Therefore, any increase in the number of dense modified particles could be attributed to a change in the density calculation itself. The radius of the base species does however stay the same, so we can still use those results. We will still include the modified species in the results for completeness sake, though.

4.3.1 Particle density distribution

In the following figures, we have split up the PDD into its separate colours. By looking at which species occurs more frequently at which radius, we can determine whether each particle type has a consistent preference for lower or higher radius values.

In Figure 24 and Figure 25 we see that the number of yellow and blue particles of the base species greatly diminishes once it no longer has the highest radius in the simulation. On the other hand, in Figure 26 and Figure 27 we see a v-shaped pattern for the brown and magenta base species. From this we cannot conclude any preference, as the brown and magenta neither consistently prefer the lower nor the higher radius. Lastly, in Figure 28 we see that the number of green particles of the base species increases once it has the lower radius of the two in the simulation. Overall, we can say that yellow and blue particles prefer high radii, green particles prefer lower radii, and brown and magenta particles are inconclusive (see Table 6).

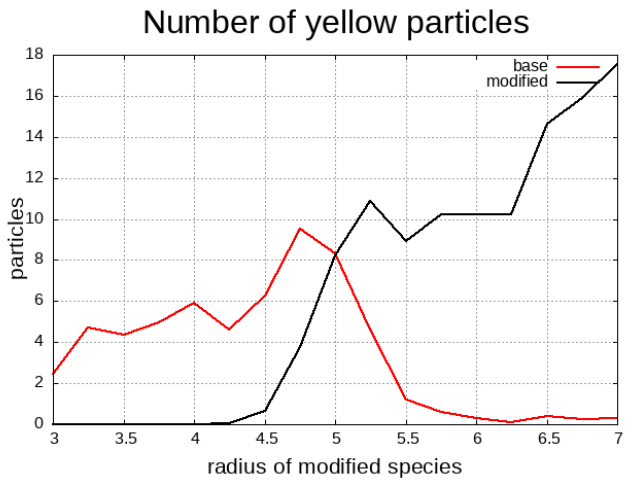

Figure 24: Average number of yellow ("cell nucleus") base and modified particles per time step.

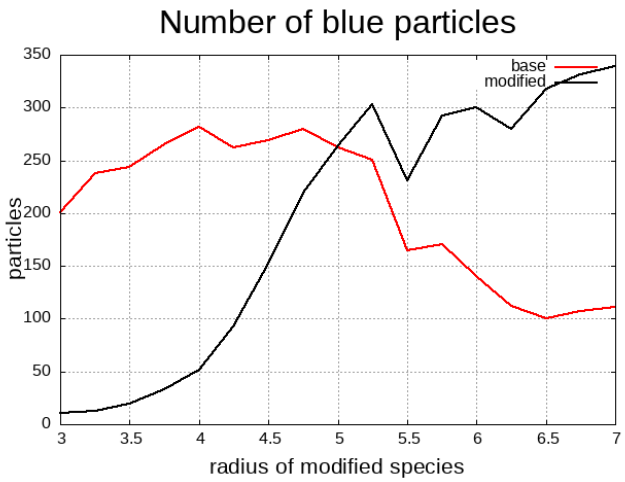

Figure 25: Average number of blue ("cell wall") base and modified particles per time step.

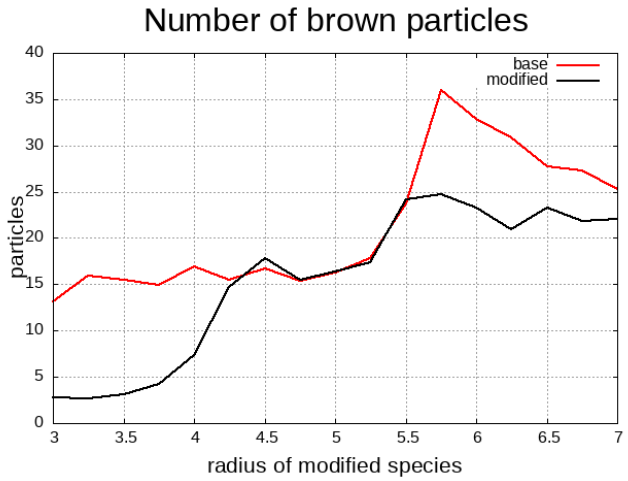

Figure 26: Average number of brown ("premature spore") base and modified particles per time step.

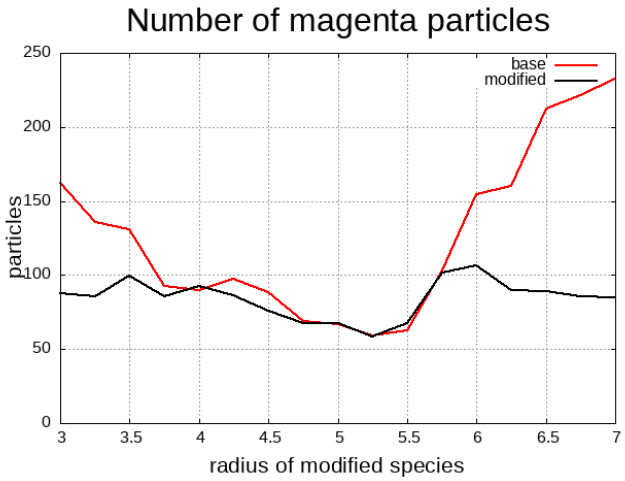

Figure 27: Average number of magenta ("mature spore") base and modified particles per time step.

4.3.2 Visual inspection

We shall now continue with the visual inspection. In Figure 29 and Figure 30 a screenshot of a simulation using the two extremes $r = 3.00$ and $r = 7.00$ is shown. Figure 29 ($r = 3.00$) shows both homogeneous and heterogeneous spores. Cells, however, only occur homogeneously. Figure 30 ($r = 7.00$) on the other hand, shows four different cell types. Firstly, there are homogeneous cells of both species. The homogeneous cells of the modified species are much larger than those of the base species, which may be attributed to their larger perception radius. Additionally, there are two types of heterogeneous cells (Figure 31), similar in nature to those in the β–experiments (Section 4.1). In these cells, the cell walls homogeneously consists of species A, whereas the cell

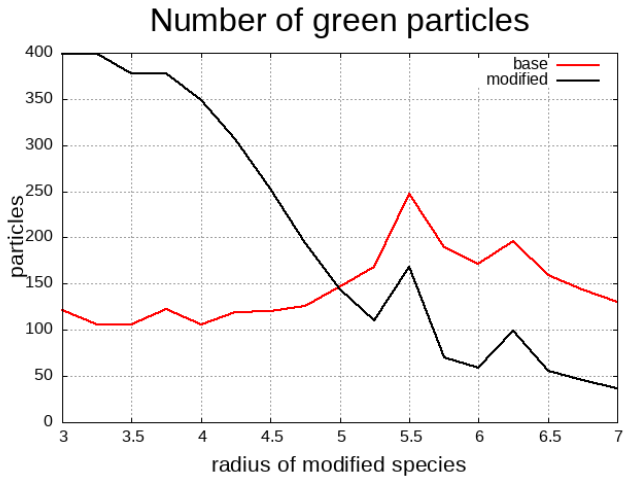

Figure 28: Average number of green ("nutrient") base and modified particles per time step.

Colour	Function	Radius
yellow	cell nucleus	high
blue	cell wall	high
brown	premature spores	—
magenta	mature spores	—
green	nutrients	low

Table 6: Preferred radius value for each particle type.

nucleus is homogeneously made up of species B (and vice versa). Having said that, unlike the proto-cells in the β–experiments, we do see yellow particles in (only) one of the types. The modified particles in the cell nucleus can turn yellow, which is most likely due to their unfair colouring conditions. It is however rather difficult to capture all four cell types at once, as they rarely occur all at the same time. Lastly, the spores occur homogeneously as well as heterogeneously.

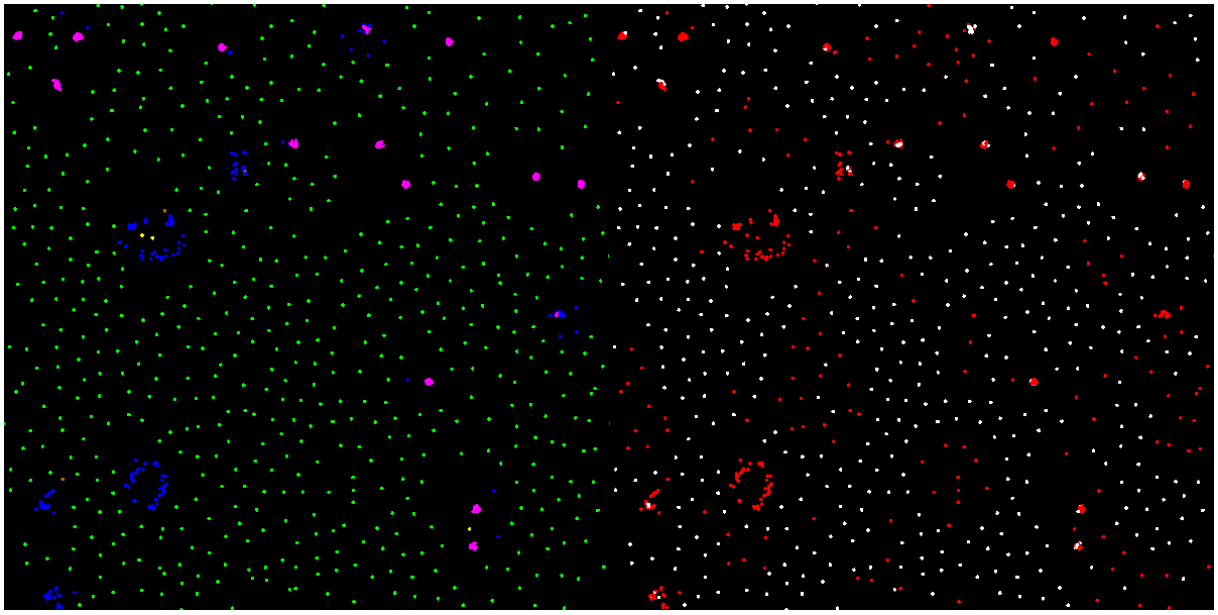

Figure 29: Screenshot of a simulation at $t = 7885$ with the modified species at $r = 3$.

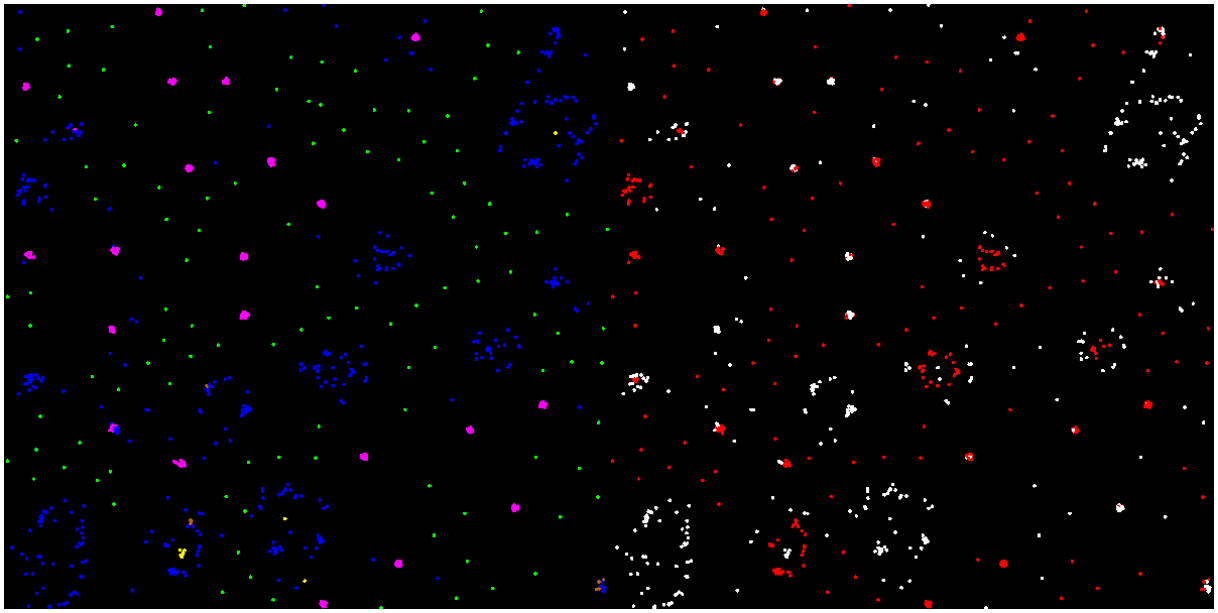

Figure 30: Screenshot of a simulation at $t = 32,571$ with the modified species at $r = 7$.

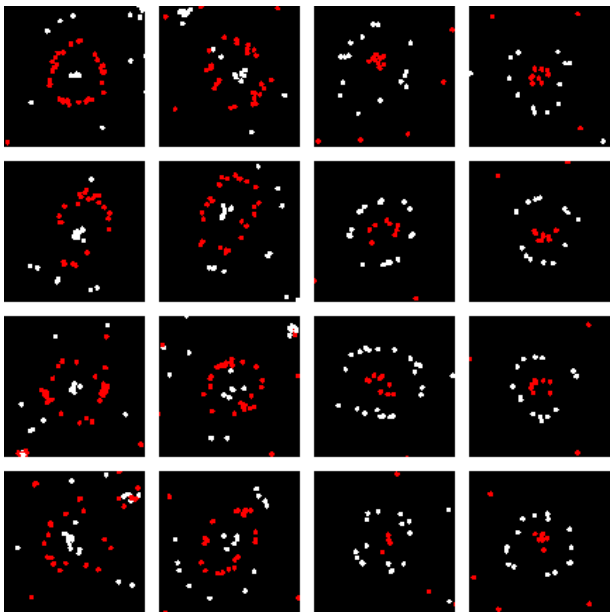

Figure 31: Close-up of the heterogeneous cells with a clear division of roles, found in a simulation with the modified species at $r = 7$.

5 Conclusion and further research

In this thesis, we have used heterogeneity as a tool to gain a better understanding of the cell-like structures in Primordial Particle Systems that emerge at the parameter set $[radius = 5, \alpha = 180, \beta = 17, speed = 0.67]$. The particles in these cell structures take of various roles, namely that of cell walls, cell nuclei, spores and nutrients. In heterogeneous two-species PPS simulations, we have separately changed the β, speed and radius parameters of one of the species, called the *modified*

species as opposed to the none-modified *base species*. In these three experiments we have observed the resulting changes in the particle density distribution (PDD) of each species. From the PDD of each species we were able to identify which species, base or modified, occurred more frequently for each particle type. With that, we could establish a "preference" for either higher or lower values of β, speed and radius with regard to the base value.

The results of the experiments have been combined in Table 7. Here we have summarised the conditions that favour each particle type. In a heterogeneous PPS close to the parameter set $[radius = 5, \alpha = 180, \beta = 17, speed = 0.67]$, particles that are found in the cell nuclei are more likely to belong to a species with a higher speed, β, or radius (or a combination of those) than any other species. Cell wall particles are likely to have low speed or β; or a high radius value. Furthermore, the particles that make up nutrients are likely to have a high speed and β, whereas their radius value tends to be low. A low perception radius in particular makes sense for nutrients, as having a low radius means having fewer neighbours and hence more chance of turning green. Now for spores, it is hard to say what influence the perception radius has. As for speed and β though, premature spores will most likely contain particles with high speed and β–values. Mature spores on the other hand, most likely consist of particles with low speed and β. Overall, we can also see that there seems to be a relation between β and speed.

Colour	Function	Speed	β	Radius
yellow	cell nucleus	high	high	high
blue	cell wall	low	low	high
brown	premature spores	high	high	—
magenta	mature spores	low	low	—
green	nutrients	high	high	low

Table 7: Characteristics of particle types.

Furthermore, in our experiments with β and radius we have found cells that show a distinct division of roles between the species. These cells contain homogeneous cell walls of species A, and homogeneous cell nuclei of species B. Together, however, they form one heterogeneous cell. In our experiments with β, these cells where considerably smaller then any other cells. Thus we coin the term *proto-cells* for these structures.

5.1 Further research

In this thesis we have shown the existence of proto-cells in a heterogeneous two-species PPS near $[radius = 5, \alpha = 180, \beta = 17, speed = 0.67]$, whose species differ slightly in their β–value. That being said, we have not documented if these participate in a similar cycle of life. The same can be said for the other heterogeneous cells that exhibit a strict division of roles seen in the experiments with radius.

Additionally, we have limited ourselves to two-species PPS, but looking at more species could be of interest too. Likewise, we have only looked at changing β, speed and radius. One property that we have not looked at is visibility. The YouTuber *TheRainHarvester* has already shown that this can achieve complex emerging behaviour (Figure 7), thus a further inspection could be worthwhile, especially when making use of the PDD to analyse what changes occur.

References

[Bec21] J. Beck. Cxxopts. https://github.com/jarro2783/cxxopts, 2021. Last accessed on 2021-07-12.

[Gar70] M. Gardner. Mathematical games: The fantastic combinations of John Conway's new solitaire game "life". *Scientific American*, 223(4):120–123, 1970.

[Hac18] HackerPoet. Particle life. https://github.com/HackerPoet/Particle-Life, 2018. Last accessed on 2021-07-12.

[Pet21] C. Petersen. Particle life. https://github.com/fnky/particle-life, 2021. Last accessed on 2021-07-12.

[Rey87] C.W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, pages 25–34, 1987.

[SS15] T. Schmickl and M. Stefanec. Population dynamics of self-replicating cell-like structures emerging from chaos. *arXiv preprint arXiv:1512.04478*, 2015.

[SS19] T. Schmickl and M. Stefanec. A primordial particle system in three dimensions. *arXiv preprint arXiv:1901.09293*, 2019.

[SSC16] T. Schmickl, M. Stefanec, and K. Crailsheim. How a life-like system emerges from a simplistic particle motion law. *Scientific Reports*, 6(1):1–15, 2016.

[Str21] R.R. Strauss. Boids. https://github.com/rystrauss/boids, 2021. Last accessed on 2021-07-12.

[The19] TheRainHarvester. "Heartbeats & blood flow": Primordial particle system. https://www.youtube.com/watch?v=gaFKqOBTj9w, 2019. Last accessed on 2021-07-12.

[The20] TheRainHarvester. Primordial particle system. experiment 45, "A larger radius for red". https://www.youtube.com/watch?v=hhK8J8i6CTg, 2020. Last accessed on 2021-07-12.

[Ven17] J. Ventrella. Clusters. http://www.ventrella.com/Clusters/, 2017. Last accessed on 2021-07-12.

[Wol02] S. Wolfram. *A New Kind of Science*. Wolfram Media, 2002.