AN INVESTIGATION INTO THE PERFORMANCE OF INTEGRATION METHODOLOGIES FOR REAL TIME MASS-SPRING CLOTH SIMULATIONS

by

CHRISTOPHER PHILLIPS
ID: 15022229

A dissertation submitted in partial fulfilment of the requirements for the award of

MASTER OF SCIENCE IN COMPUTER GAMES PROGRAMMING

September 2016

Department of Computing
Staffordshire University
Stafford ST18 0AD

Supervised by: Christopher McCreadie

I declare that this dissertation is my own work and that the work of others is acknowledged and indicated by explicit references.

Christopher Phillips
September 2016

Abstract

There has been much research into teaching computers to play games effectively, resulting in programs which are capable of beating the best human players at chess and other board games. These programs typically make use of strongly defined rules to provide their intelligence. This project suggests a different technique; applying stochastic optimisation techniques to effectively learn the strategic rules for playing Connect 4.

Three popular algorithms, a genetic algorithm, evolution strategies and particle swarm optimisation, were implemented to play Connect 4 in The Arena, a Java based framework providing the implementation of the game. Following a training period to learn the rules of the game, the playing performance of these algorithms was tested. The test results showed limited success, with only one of the algorithms able to play relatively successfully. The time constraints for the development of this project may be the reason for the poor performance of the algorithms, therefore more research and testing should be considered.

Acknowledgements

With thanks to Dr James Heather, for his help during the development of this project, and Dr Johann A. Briffa, for no doubt saving the author countless arguments with word processing software whilst trying to format this dissertation.

Contents

List of Figures

List of Tables

Chapter 1

Literature Review

1.1 Cloth Simulation

The simulation of cloth is a relatively old and well studied field with applications in many different areas, including, but not limited to:

- Virtual Garment Design
- Virtual Fitting Rooms
- Films
- Video Games

Different use cases require different things from the cloth simulation. For example, in virtual garment design, the physical accuracy of the simulation is paramount whereas cloth simulation for video games prioritises real-time simulation, sacrificing accuracy. Hence, many different models for cloth simulation have been proposed.

Since this project is concerned with the real-time simulation of cloth, only those models appropriate for real-time simulations have been studied in detail. However, a brief overview of other techniques will be provided. For a more detailed overview of cloth simulation techniques, see Ng and Grimsdale (1996).

1.1.1 Cloth Properties

Cloth has several properties that should be considered for modelling.

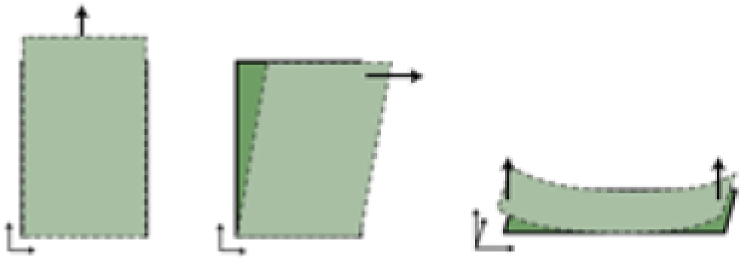


Figure 1.1: Mechanical properties of cloth (*Techniques for Animating Cloth*, p. 1)

1.1.1.1 Mechanical Properties

Cloth has three mechanical properties that control its behaviour; stretching, shearing and bending, fig. 1.1 shows how each property affects the cloth.

Stretching is the displacement of the cloth in either the horizontal or vertical direction. Most cloth has a high resistance to stretching and can typically only be stretched by 10% (*Techniques for Animating Cloth*, p. 1; Provot 1995, p. 4).

Shearing is the displacement of the cloth in a diagonal direction. Again, most cloths have high shearing resistance and this, coupled with high stretch resistance, makes cloth incompressible.

Finally, bending is the overall curvature of the cloth surface. Typically, cloth has low bending resistance and so is easily folded.

1.1.1.2 Visual Properties

The mechanical properties of cloth, discussed above, cause cloth to exhibit two visual properties. These properties arise from the fact that cloth is typically non-elastic, due to stretch and shear resistances, but highly flexible, due to low bend resistance.

Firstly, cloth will drape over objects and secondly the cloth will form many folds and wrinkles. Fig 1.2 demonstrates the visual properties of cloth.

1.1.2 Cloth Models

Techniques for modelling cloth are usually classified as either Geometric or Physically-based, and the choice of which modelling method to use depends on the use-case for the simulation.

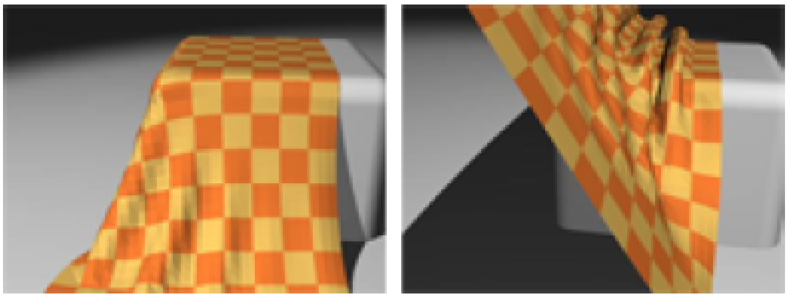


Figure 1.2: Visual properties of cloth (*Techniques for Animating Cloth*, p. 1)

1.1.2.1 Geometric Models

This family of techniques were the first models used to simulate cloth. They model the cloth using geometric equations and are especially good at modelling folds and wrinkles.

Weil was the first to propose a geometric model in 1986 and uses catenary curves to model the drape and folds of a hanging cloth. Following on from Weil, a number of other geometric models were proposed (see Ng and Grimsdale (1996) for more information).

All geometric techniques focus on simulating the appearance of cloth, rather than the physical properties. As such, geometric models are typically more computationally efficient than physically-based models, as there is no need to solve a series of complex equations. However, geometric techniques are unable to accurately simulate the motion of cloth, (Mongus et al. 2012, p. 1; Zhang and Yuen 2001, p. 2; Xinrong et al. 2009, pp. 1-2), and so are mostly useful for static cloth simulations.

As such, geometric models have not been considered for this project.

1.1.2.2 Physically-based Models

By contrast, physically-based models are concerned with the accurate modelling of the physical properties of the cloth and can therefore be used to produce realistic animations.

These models typically use a system of partial differential equations (PDE), or other differential equations, to model the cloth. These equations cannot be solved analytically and therefore the system requires discretisation to solve the equations at specific points in space and time. Following discretisation, a physically-based model typically requires the solving of an ordinary differential equation (ODE) of the form (Baraff and Witkin 1998, p. 1):

$$\ddot{x} = M^{-1}\left(-\frac{\delta E}{\delta x} + F \right)$$

where:

x is a vector representing the geometric state of the system

M is a diagonal matrix representing the mass distribution of the system (1.1)

E is a function of x which yields the internal cloth energy

F is a function of x and $\dot{x}$ which describes other forces

Physically-based models can be classified as either Continuum or Discrete.

1.1.2.2.1 Continuum Models

Continuum models were the first physical models to be proposed. Techniques in this family model cloth as a continuous surface and utilise continuum mechanics to calculate its behaviour; the Lagrange equations are most commonly used.

To discretise the continuous model, a numerical technique, such as a finite element method, is used. This is one of the advantages of continuum methods; they allow the use of a low resolution discretisation without sacrificing the accuracy of the simulation (Wacker, Thomaszewski, and Keckeisen 2005, pp. 4-5).

Another advantage of continuum models are that they are accurate; "they provide accurate models of the material derived directly from mechanical laws and models of material properties" (Magnenat-Thalmann and Thalmann 2004, p. 200).

This accuracy comes at the cost of computational performance, the main disadvantage of these techniques. The accuracy also renders these models inappropriate for use in dynamic simulations; "the formal and analytical description they require for the mechanical behavior of the material cannot easily be altered to represent transitory and non-linear events. Hence, phenomena such as frequent collisions or other highly variable geometrical constraints cannot be conveniently taken into account" (ibid., p. 200).

Hence, continuum models are typically only considered appropriate for static simulations, or simulations where accuracy is paramount. As a result, continuum models have not been considered.

1.1.2.2.2 Discrete Models

According to Choi and Ko (2002, p. 2), "Cloth is not a homogeneous continuum. Therefore modeling fabrics as a continuum and employing FEM or FDM has several potential drawbacks". As a result several discrete, or particle, models have been proposed.

With these techniques the discretisation in space is carried out by modelling the cloth as a discrete mesh, either regular or triangular, of point masses, called particles. This sacrifices some of the accuracy of continuum models, as the accuracy will depend on the number of particles used. However, this loss in accuracy is traded off against better computational performance; physical models are the only models that can be used for dynamic, real-time simulations. The discretisation of the cloth has a direct affect on the performance of the simulation; Volino and Magnenat-Thalmann (2001, p. 5) have shown that simulation time varies cubically with mesh size. Hence, there is a trade off to be made between accuracy and performance when using physically-based models, depending on the use-case.

By far the most popular technique is the mass-spring model. This model is popular as it is simple, easy to implement and offers a good balance between accuracy and efficiency. This model is the most common model used for dynamic, real-time simulations and as such, it has been chosen for this project and will now be described in more detail.

1.1.3 Mass-Spring Models

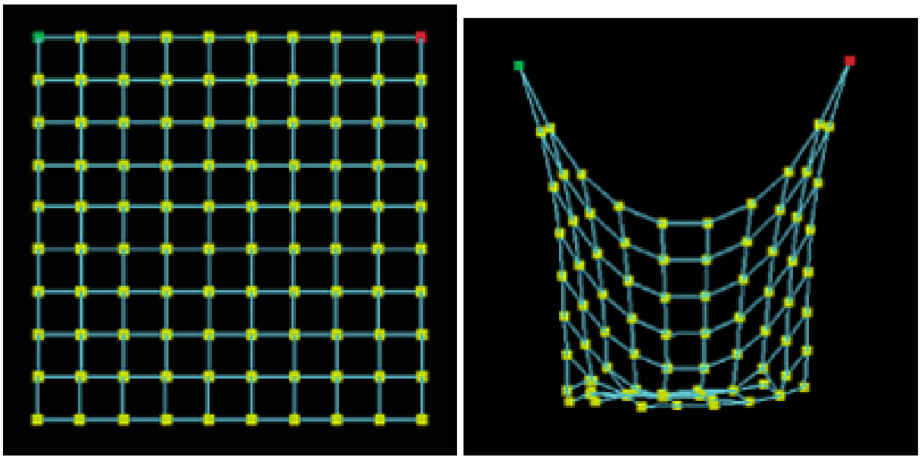
Mass-spring models were first proposed for use in cloth simulation in Provot (1995). Using these models, the cloth is discretised as a 2-dimensional mesh of point masses, either regular or triangular, connected by linear springs.

1.1.3.1 Provot Model

Using the mass-spring model proposed in Provot (ibid.) the cloth is modelled as a regular mesh of point masses. The points are connected together using three different types of springs:

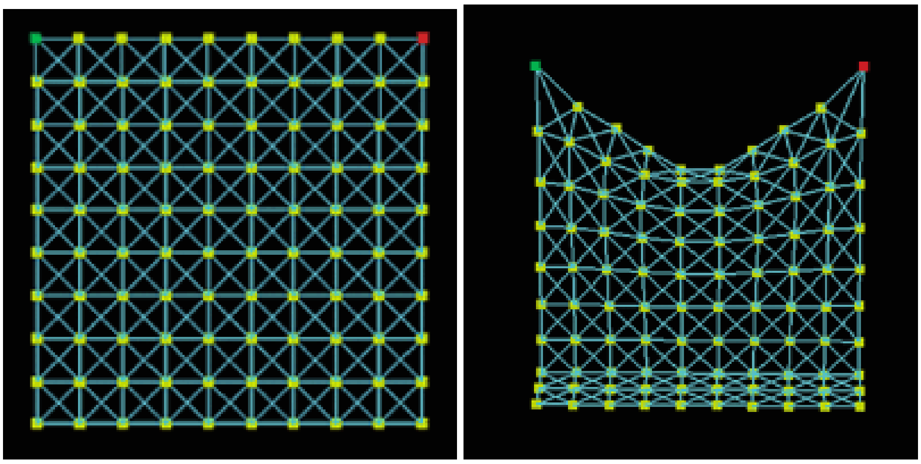
- Structural springs, connecting particle [i, j] to particles [i + 1, j] and [i, j + 1]. These springs resist structural deformations of the cloth, and provide the overall cloth structure. Structural springs are not enough to provide a realistic cloth model. Fig 1.3 shows the results of running a cloth simulation with structural springs only. As can be seen, this does not produce a realistic image

- Shear springs, connecting particle [i, j] to particle [i + 1, j + 1] and particle [i + 1, j] to particle [i, j + 1]. These springs provide shearing resistance for the cloth. By adding shear springs, the realism of the model is improved; Fig 1.4 shows the improved fidelity afforded by shear springs



(a) Initial configuration (b) Result of running the simulation

Figure 1.3: Cloth with structural springs only (Lander 2000b, p. 2)



(a) Initial configuration (b) Result of running the simulation

Figure 1.4: Cloth with structural and shear springs (Lander 2000b, p. 2)

- Bend springs, connecting particle [i, j] to particles [i + 2, j] and [i, j + 2]. These springs model bend resistance

Fig 1.5 shows the arrangement of these springs for a small cloth model.

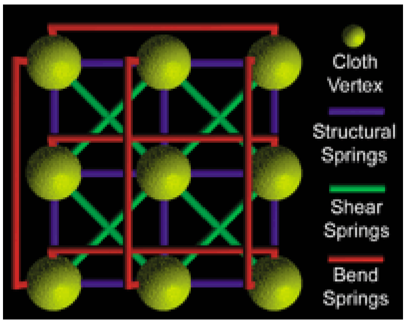


Figure 1.5: Provot cloth model (Lander 2000b, p. 2)

To animate the mesh, forces are applied to the particles and are calculated using Newton's second law:

$$F_{ij} = m_{ij} a_{ij}$$

where:

F_{ij} is the total sum of forces acting on particle ij (1.2)

m_{ij} is the mass of particle ij

a_{ij} is the acceleration of particle ij

The total force acting on a particle is defined as:

$$F_{total} = \Sigma F_{external} + \Sigma F_{internal} \tag{1.3}$$

$F_{external}$ are external forces acting on the mesh, such as gravity and wind.

Gravity is calculated by:

$$F_g = m_{ij} G$$

where: (1.4)

G is the gravitational constant

Wind is calculated by:

$$F_{wind} = w(n_{ij} \bullet \overrightarrow{W})$$

where:

n_{ij} is the surface normal of particle ij (1.5)

$\overrightarrow{W}$ is the wind direction vector

w is the wind constant

$F_{internal}$ are the resultant forces of the springs connecting the mesh.

The spring force is calculated using the Hooke equation (Parent 2012, p. 201):

$$F_{spring} = -k_s(L_c - L_r)\frac{p_2 - p_1}{\parallel p_2 - p_1 \parallel} \tag{1.6}$$

where:

k_s is the spring stiffness coefficient

L_c is the current length of the spring

L_r is the initial, or rest, length of the spring

p_1 & p_2 are the positions of the two connected particles

Using this equation, the cloth will be modelled with pure elastic springs and will oscillate indefinitely. However, as mentioned in 1.1.1.2 and (Provot 1995, p. 1), cloth is a non-elastic medium and therefore the model needs to account for the energy lost due to internal friction in the cloth.

This is typically modelled as an extra internal damping force, calculated using (Parent 2012, p. 201):

$$F_{damping} = -k_d(\dot{p}_2 - \dot{p}_1) \bullet \left(\frac{p_2 - p_1}{\parallel p_2 - p_1 \parallel}\right)\left(\frac{p_2 - p_1}{\parallel p_2 - p_1 \parallel}\right) \tag{1.7}$$

where:

k_d is the spring damping coefficient

$\dot{p}_1$ & $\dot{p}_2$ are the velocities of the two connected particles

1.1.3.2 Choi Ko Model

A different mass-spring model was proposed in Choi and Ko (2002) and aims to improve the buckling behaviour of the cloth, resulting in more realistic draping and wrinkling behaviour.

The cloth is modelled in a similar way to the Provot method, but additional bend springs are added, connecting particle [i, j] to particle [i + 2, j + 2] and particle [i + 2, j] to particle [i, j + 2]; fig 1.6 shows the arrangement of springs.

This model uses an energy-based approach, of the general formula 1.8(Bartels 2014, p. 3), to calculate the forces acting on individual particles.

$$F = \left(\frac{\delta E(S)}{\delta x}, \frac{\delta E(S)}{\delta y}, \frac{\delta E(S)}{\delta z}\right) \tag{1.8}$$

where:

E(S) is an energy function of S, a representation of the cloth's state

Two types of interactions are defined, type 1 and type 2. Type 1 interactions model stretch and shear resistances, the red lines in 1.6, and are represented by a linear spring model. Type 2 interactions model

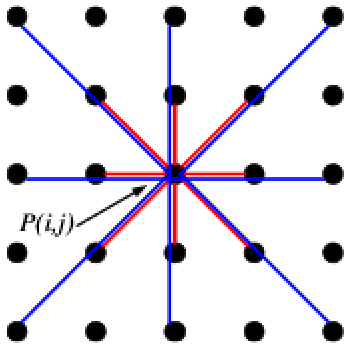


Figure 1.6: Choi Ko cloth model (Choi and Ko 2002, p. 2)

bend forces, the blue lines in 1.6, and helps prevent the so called post-buckling instability problem. The interested reader should see Choi and Ko (2002) for more information on the energy functions for each interaction type.

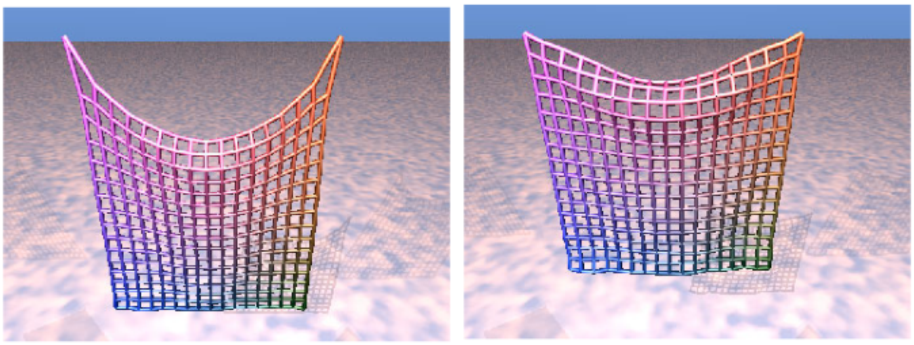
1.1.3.3 Justification of Choice

Mass-spring models are suitable for use in this project as they are efficient and simple to implement; "Mass-spring models are the most efficient as well as the simplest of the cloth models. This method is one of the most popular techniques for simulating cloth, especially when interactive frame rates are required" (Zink and Hardy 2007, p. 2). As this project is concerned with the real time animation of cloth, mass-spring models are therefore the obvious choice.

Mass-spring models are also the most common method of modelling cloth in video games, used in games such as Alan Wake, (Enqvist 2010, p. 2), and Hitman: Codename 47, (Jakobsen 2005, p. 1), as well as commercial physics engines for games, such as Havok® and PhysX®. Again, since this project is concerned with the simulation of cloth for use in a video game, mass-spring models are the logical choice.

In particular, the Provot model will be used for this project as the force-based approach requires the solving of a much simpler series of equations than the energy-based approach of the Choi Ko model, and is therefore likely to be more computationally efficient.

One disadvantage of the mass-spring model is that it does not achieve realistic animation of cloth as "mass-spring systems do not model any specific material and are not related to measured properties of real clothes" (Wacker, Thomaszewski, and Keckeisen 2005, p. 3). However, by careful tuning of the spring stiffness coefficients pleasing results can be achieved; Mongus et al. (2012) have shown that,



(a) Super-elasticity effect (b) Result of correcting super-elasticity

Figure 1.7: Super-elasticity problems (Provot 1995, pp. 4,6)

with tuning, mass-spring models can reproduce the drape of a cloth with an accuracy of 97%. Another disadvantage of mass-spring models is what Provot calls 'super-elasticity'; springs are allowed to deform too much, leading to unrealistic looking cloth. This is caused because "the springs are “ideal” and they have unlimited linear deformation rate" (Vassilev, Spanlang, and Chrysanthou 2001, p. 3). To counter this effect, Provot suggests enforcing length constraints on structural and shear springs. First the position of the particles are updated, then the deformation of the springs are calculated. If this deformation value is greater than some threshold, τ_c, then the position of the connected particles are adjusted so the deformation rate equals τ_c. Fig 1.7 shows the super-elasticity problem and the results of employing Provot's corrective method. As can be seen, correcting the super-elasticity results in a much more realistic model.

1.1.4 Numerical Integration for Mass-Spring Models

As mentioned above, physically-based models for cloth simulation require solving a series of differential equations, discretised in space and time. According to Wacker, Thomaszewski, and Keckeisen (2005, p. 5), "since particle systems already represent a discretization in space, only a system of ordinary differential equations has to be solved". For the Mass-Spring model using Newtonian mechanics a series of second order ODEs, of the form 1.9, must be solved (Zink and Hardy 2007, p. 5).

$$\frac{\delta^2 x}{\delta t^2} = M^{-1} F(x,v) \tag{1.9}$$

This can be converted into a coupled series of first order ODEs with the addition of an extra variable (ibid., p. 5):

$$\frac{\delta}{\delta t}\begin{pmatrix} x \\ v \end{pmatrix} = \begin{pmatrix} v \\ M^{-1}F(x,v) \end{pmatrix} \tag{1.10}$$

Equations 1.9 and 1.10 cannot be solved analytically, and therefore it is necessary to use a numerical method, or integrator, to approximate them at discrete time intervals. There are many integration methods that could be chosen, typically classified as either explicit or implicit, and the most popular choices will be described now.

1.1.4.1 Explicit Integrators

Most of the early work on physically-based cloth simulation used explicit integrators as they are simple and easy to implement; they only require information about the state of the system at the previous interval to calculate the current state.

The most commonly used explicit integrators are the Runge-Kutta family of integrators.

1.1.4.1.1 Euler

The first order Runge-Kutta integrator, or explicit Euler, was used in Provot (1995) to approximate a Mass-Spring system.

Equation 1.10 is approximated by (Wang, Hu, and Zhuang 2009, p. 3):

$$
\begin{aligned}
& v_{i+\Delta t} = v_i + \Delta t F(t_i, v_i) \\
& \text{where:} \\
& \quad v_i \text{ and } x_i \text{ are the velocity and position of a particle at time interval i} \\
& \quad \Delta t \text{ is the time step}
\end{aligned}
\tag{1.11}
$$

When applied to the Provot Mass-Spring model, this gives (Provot 1995, p. 3):

$$
\begin{aligned}
& a_{i,j}(t+\Delta t) = \frac{1}{m_{i,j}} F_{i,j}(t) \\
& v_{i,j}(t+\Delta t) = v_{i,j}(t) + \Delta t a_{i,j}(t+\Delta t) \\
& x_{i,j}(t+\Delta t) = x_{i,j}(t) + \Delta t v_{i,j}(t+\Delta t) \\
& \text{where:} \\
& \quad a_{i,j},\ m_{i,j},\ v_{i,j} \text{ and } x_{i,j} \text{ are the acceleration, mass, velocity and position of particle i, j respectively}
\end{aligned}
\tag{1.12}
$$

The explicit Euler method is computationally cheap but can result in numerical instability if too large a time step is used. Mathematically, the explicit Euler method is stable only if the time step is less than the natural period of the system, approximated as $\pi\sqrt{\frac{m}{K}}$, where K is the maximum stiffness in the system. Vassilev, Spanlang, and Chrysanthou (2001, p. 2) found that in fact explicit Euler is only stable for Δt

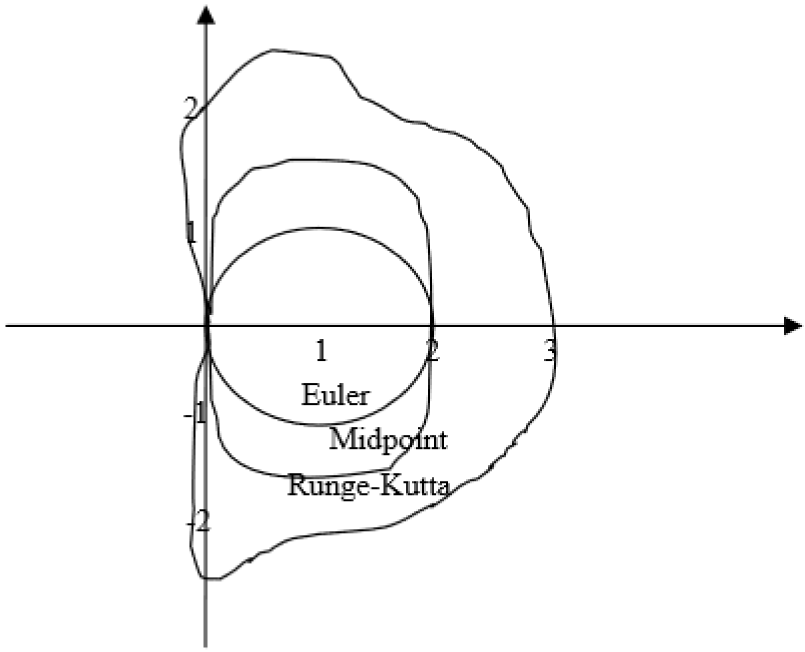


Figure 1.8: Explicit integrator stability regions (Wang, Hu, and Zhuang 2009, p. 4)

values less than $0.4\pi\sqrt{\frac{m}{K}}$.

As cloth generally does not stretch easily, this results in high stiffness in the structural and shear springs, which necessitates the use of a small time step if the explicit Euler integrator is chosen. This can impact the overall performance of the simulation, as while this integrator is cheap, the frequency of calculations is high as a result of the time step limitations.

1.1.4.1.2 Midpoint

The explicit Midpoint integrator is a second order Runge-Kutta method and modifies the Euler integrator to give greater stability.

Equation 1.11 is modified to give the following (Wang, Hu, and Zhuang 2009, p. 3):

$$v_{i+\Delta t} = v_i + \Delta t F\left(t_i + \frac{\Delta t}{2}, v_i + \frac{\Delta t}{2} F(t_i, v_i)\right) \tag{1.13}$$

Since this method requires two derivatives, the computational cost is greater than Euler. However, because the midpoint method affords greater numerical stability (see fig 1.8), a larger time step can be used which increases the overall simulation performance; Wang, Hu, and Zhuang (ibid.) have shown that the midpoint integrator offers close to twice the simulation performance over explicit Euler.

1.1.4.1.3 Fourth order Runge-Kutta

The Fourth order Runge-Kutta integrator offers greater stability using larger time steps over the midpoint method and is formulated as (Wang, Hu, and Zhuang 2009, p. 3):

$$
\begin{aligned}
v_{i+\Delta t} &= v_i + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\
k_1 &= F(t_i + v_i) \\
k_2 &= F\left(t_i + \frac{\Delta t}{2}, v_i + \frac{\Delta t}{2}k_1\right) \\
k_3 &= F\left(t_i + \frac{\Delta t}{2}, v_i + \frac{\Delta t}{2}k_2\right) \\
k_4 &= F\left(t_i + \Delta t, v_i + \frac{\Delta t}{2}k_3\right)
\end{aligned}
\tag{1.14}
$$

This integrator has a significantly higher computational cost than the other methods discussed, however Volino and Magnenat-Thalmann (2001, p. 4) have shown that the Runge-Kutta integrator supports time steps almost six times larger than the midpoint method. Therefore, given that Runge-Kutta is three times as computationally expensive as midpoint, this suggests that this integrator can lead to twice the overall simulation performance. Wang, Hu, and Zhuang (2009, p. 4) have also shown that the fourth order Runge-Kutta integrator offers simulation performance over midpoint and explicit Euler.

1.1.4.1.4 Verlet

Verlet integration is an alternative to the Runge-Kutta family of integrators. It avoids velocity calculations by approximating the velocity of a particle using its previous positions.
A particle's new position is approximated by (Mongus et al. 2012, p. 2):

$$x_{i+\Delta t} = 2x_i - x_{i-\Delta t} + a_{i+\Delta t}\Delta t^2 \tag{1.15}$$

This method is computationally fast and reasonably stable, as "velocity is implicitly given and consequently it is harder for velocity and position to come out of sync" (Jakobsen 2005, p. 1). However it does still suffer from time step issues; figures 11 and 13 in Wacker, Thomaszewski, and Keckeisen (2005, pp. 14-15) show that below a certain threshold Verlet integration is less stable than explicit Euler, but more stable over that threshold.

1.1.4.2 Implicit Integrators

According to Baraff and Witkin (1998, p. 1), "Explicit methods are ill-suited to solving stiff equations because they require many small steps to stably advance the simulation forward in time". Therefore they propose an implicit integrator for use in cloth simulation since implicit integrators are unconditionally stable regardless of step size.

However implicit integrators are computationally more expensive than their explicit equivalents, as "they involve the resolution of a large and sparse linear equation system for each iteration" (Volino, Cordier, and Magnenat-Thalmann 2005, p. 4). Since they are unconditionally stable however, this can be countered by simply using a larger time step. The guaranteed stability of these methods also reduces the accuracy of the simulation, as they introduce inherent numerical damping, which increases as the time step increases (see Volino and Magnenat-Thalmann (2001, p. 4)). As such, there is a balance to be found between performance and accuracy of the simulation with implicit integrators.

The most common implicit integrator is the implicit, or backward, Euler method which will be described now. It should be noted that there are many other implicit integrators available.

1.1.4.2.1 Euler

First proposed for use in cloth simulation in Baraff and Witkin (1998), the implicit, or backward, Euler method is an adaptation of the explicit Euler method.

Equation 1.12 is modified to give (Kang, Choi, and Cho 2000, p. 3):

$$v_i^{t+\Delta t} = v_i^t + F_i^{t+\Delta t} \frac{\Delta t}{m_i} \tag{1.16}$$

$F_i^{t+\Delta t}$ cannot be calculated at the current time step, and so must be approximated as (ibid., p. 3):

$$F^{t+\Delta t} = F^t + \frac{\delta F}{\delta x} \Delta x^{t+\Delta t} \tag{1.17}$$

$\Delta x^{t+\Delta t}$ can be written as $\Delta t(v^t + \Delta v^{t+\Delta t})$ and so eq. 1.17 can be rewritten, giving the series of linear equations (ibid., p. 3):

$$\left(I - \frac{\Delta t^2}{m} \frac{\delta F}{\delta x}\right) \Delta v^{t+\Delta t} = F^t \frac{\Delta t}{m} \tag{1.18}$$

where:

I is the identity matrix

Thus, implicit Euler involves calculating $\Delta v^{t+\Delta t}$ every iteration using:

$$\Delta v^{t+\Delta t} = \left(I - \frac{\Delta t^2}{m} \frac{\delta F}{\delta x}\right)^{-1} F^t \frac{\Delta t}{m} \tag{1.19}$$

$\frac{\delta F}{\delta x}$ is the negated Hessian matrix, denoted as H, and can be approximated as (Kang, Choi, and Cho 2000, p. 3):

$$H_{ij} = \begin{cases} k_{ij} & \text{if } i \neq j \\ -\sum_{i \neq j} k_{ij} & \text{if } i = j \end{cases} \tag{1.20}$$

where:

k_{ij} is the spring stiffness

As a result, the implicit Euler method requires the computation of an n x n matrix each iteration, and thus increasing the size of the mesh greatly impacts the performance of this method.
However, if the stiffness of the springs is constant throughout the simulation then $\left(I - \frac{\Delta t^2}{m} H\right)^{-1}$ can be precomputed, giving performance gains.

Appendix A

Class Diagrams

A.1 Optimisation library Class Diagram

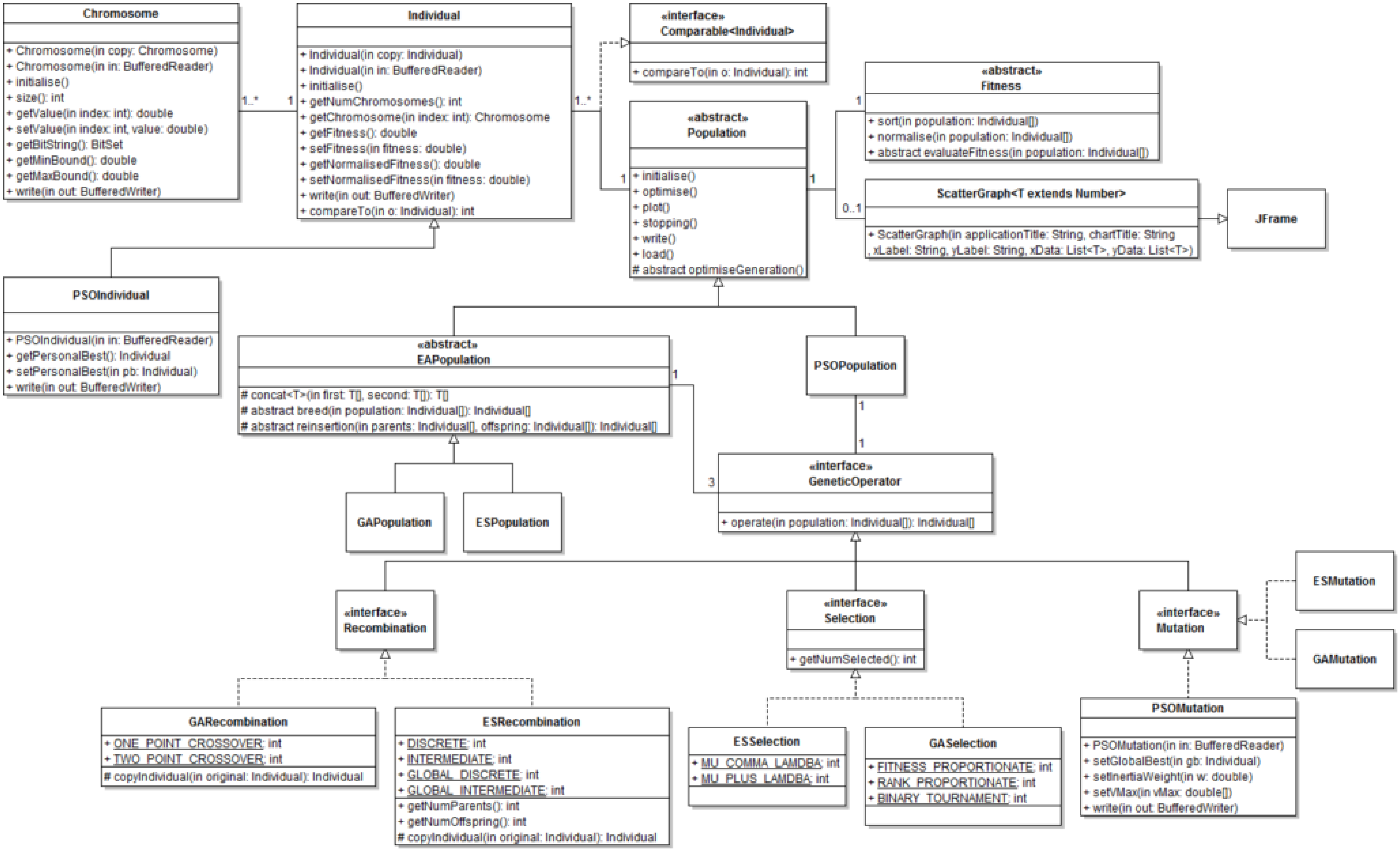


Figure A.1: Class diagram for the optimisation library

A.2 The Arena Class Diagram

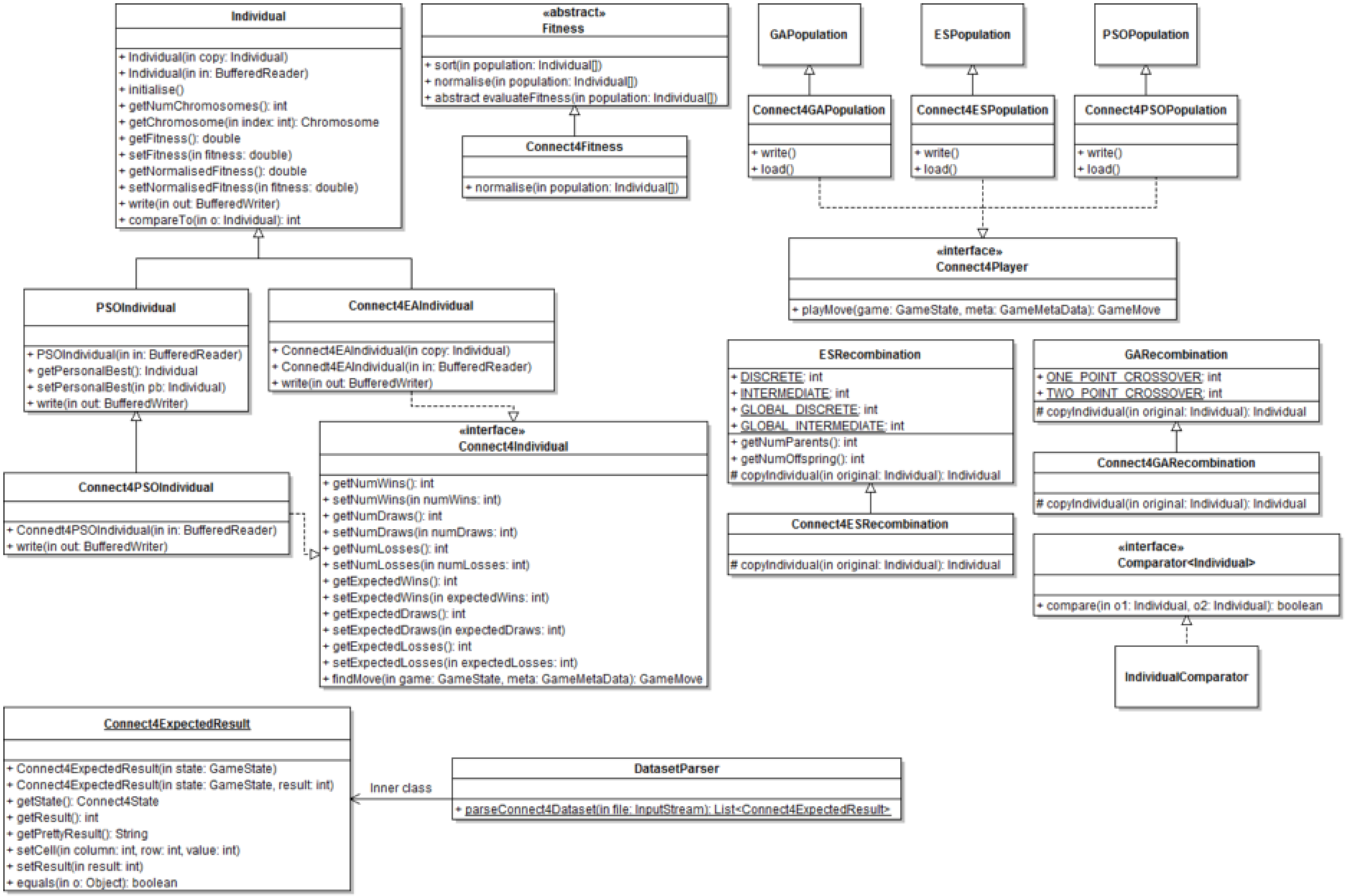


Figure A.2: Class diagram showing the additional Arena classes

Appendix B

Phase One Test Results

B.1 Genetic Algorithm

B.1.1 Binary Tournament Selection With Two-point Crossover

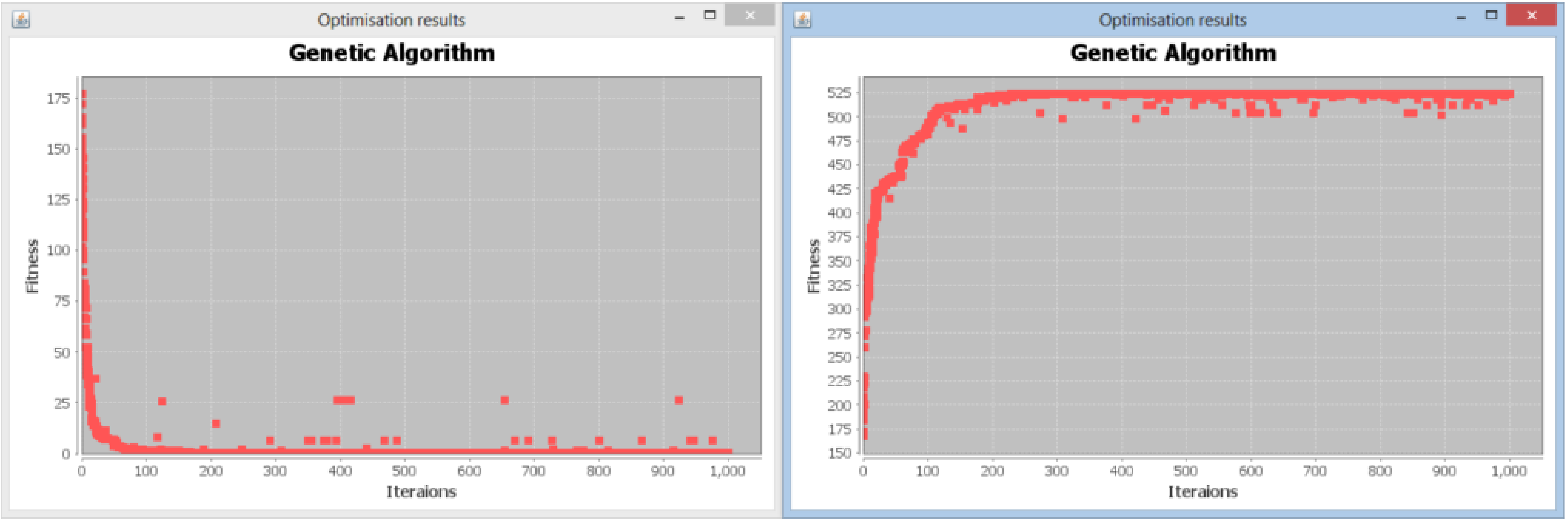


Figure B.1: Optimisation results of a genetic algorithm using binary tournament selection and two-point crossover

B.1.2 Binary Tournament Selection With One-point Crossover

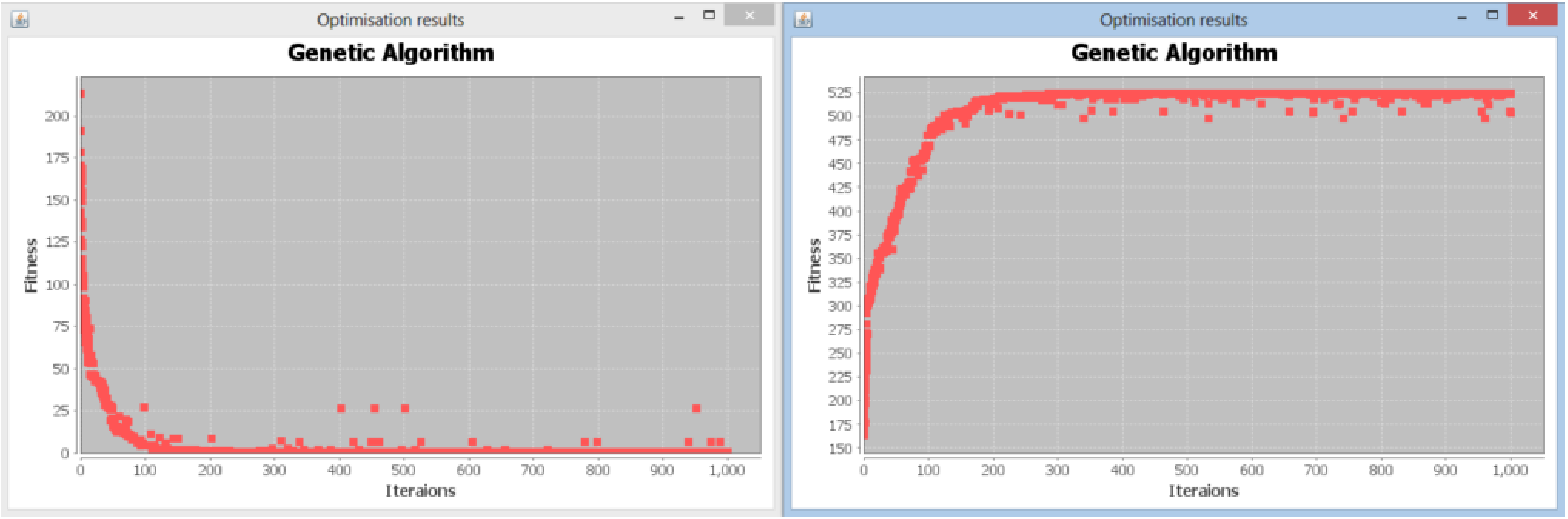


Figure B.2: Optimisation results of a genetic algorithm using binary tournament selection and one-point crossover

B.1.3 Rank Selection With Two-point Crossover

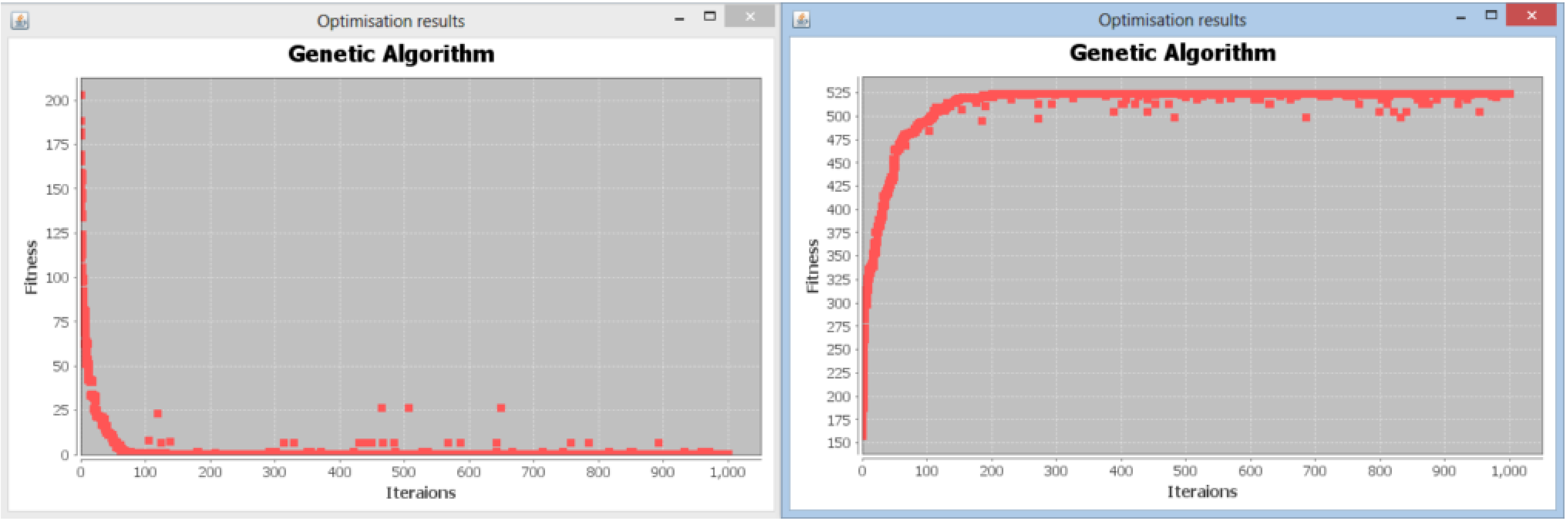


Figure B.3: Optimisation results of a genetic algorithm using rank selection and two-point crossover

B.1.4 Rank Selection With One-point Crossover

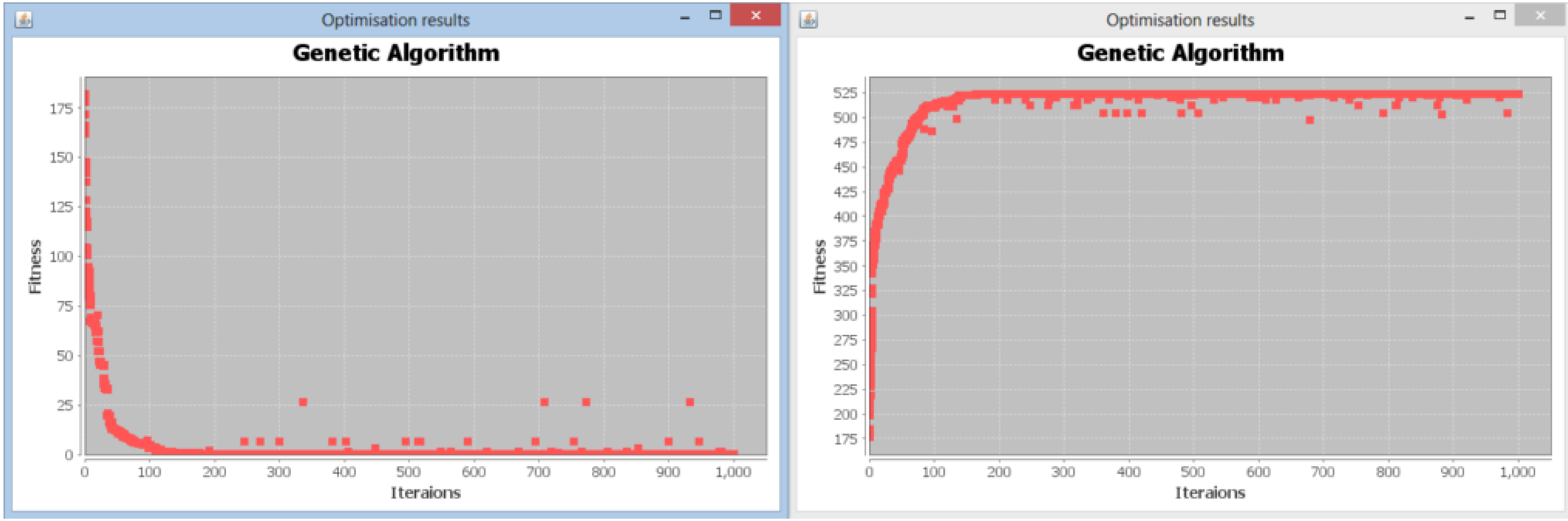


Figure B.4: Optimisation results of a genetic algorithm using rank selection and one-point crossover

B.1.5 Proportional Selection With Two-point Crossover

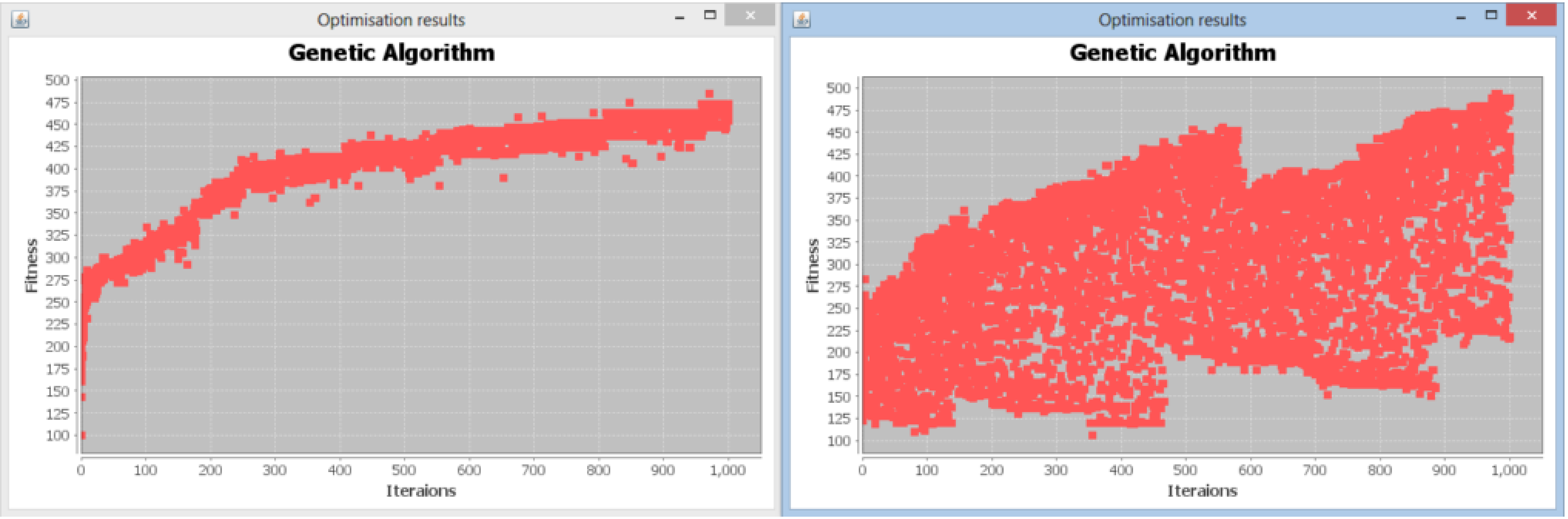


Figure B.5: Optimisation results of a genetic algorithm using proportional selection and two-point crossover

B.1.6 Proportional Selection With One-point Crossover

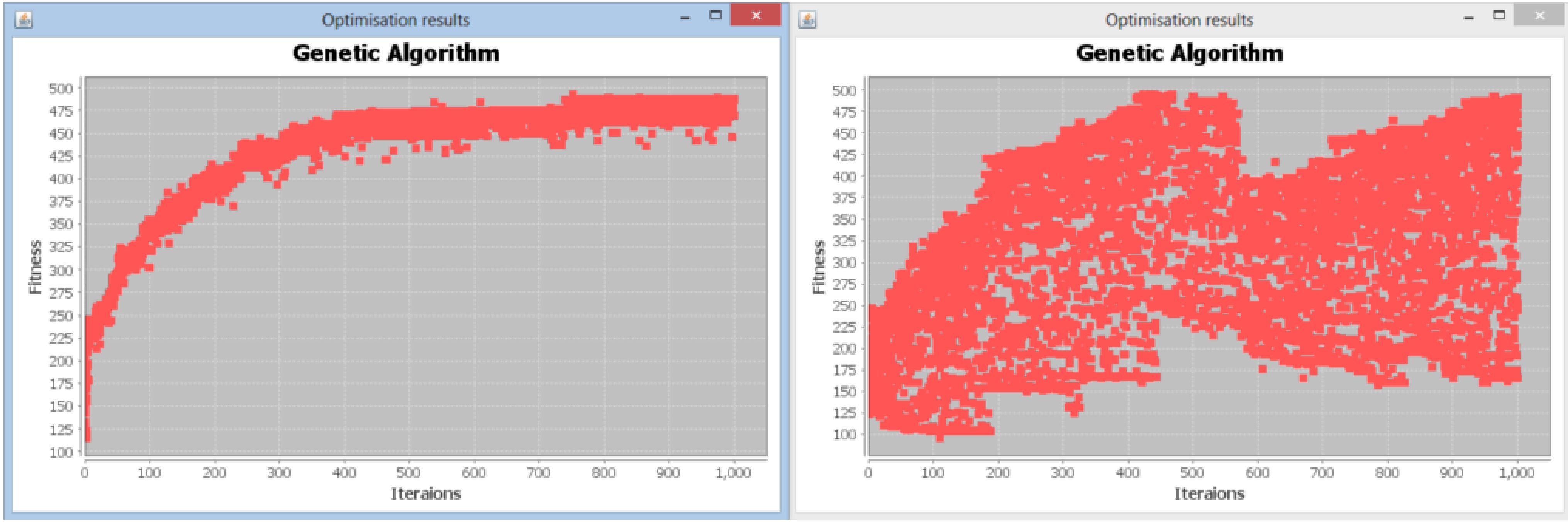


Figure B.6: Optimisation results of a genetic algorithm using proportional selection and one-point crossover

B.1.7 Proportional Selection With Adjusted Numbers of Parents

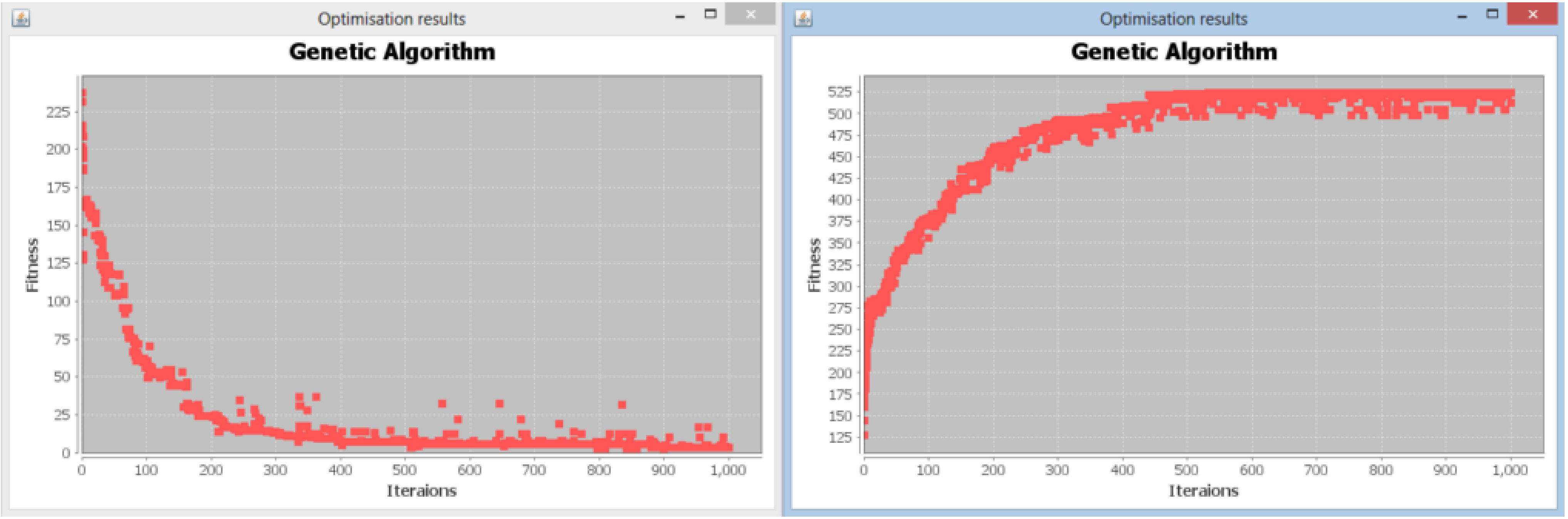


Figure B.7: Optimisation results of a genetic algorithm using proportional selection and two-point crossover. The number of parents has been adjusted to give better results.

B.1.8 Binary Tournament Selection With Two-point Crossover in Shark

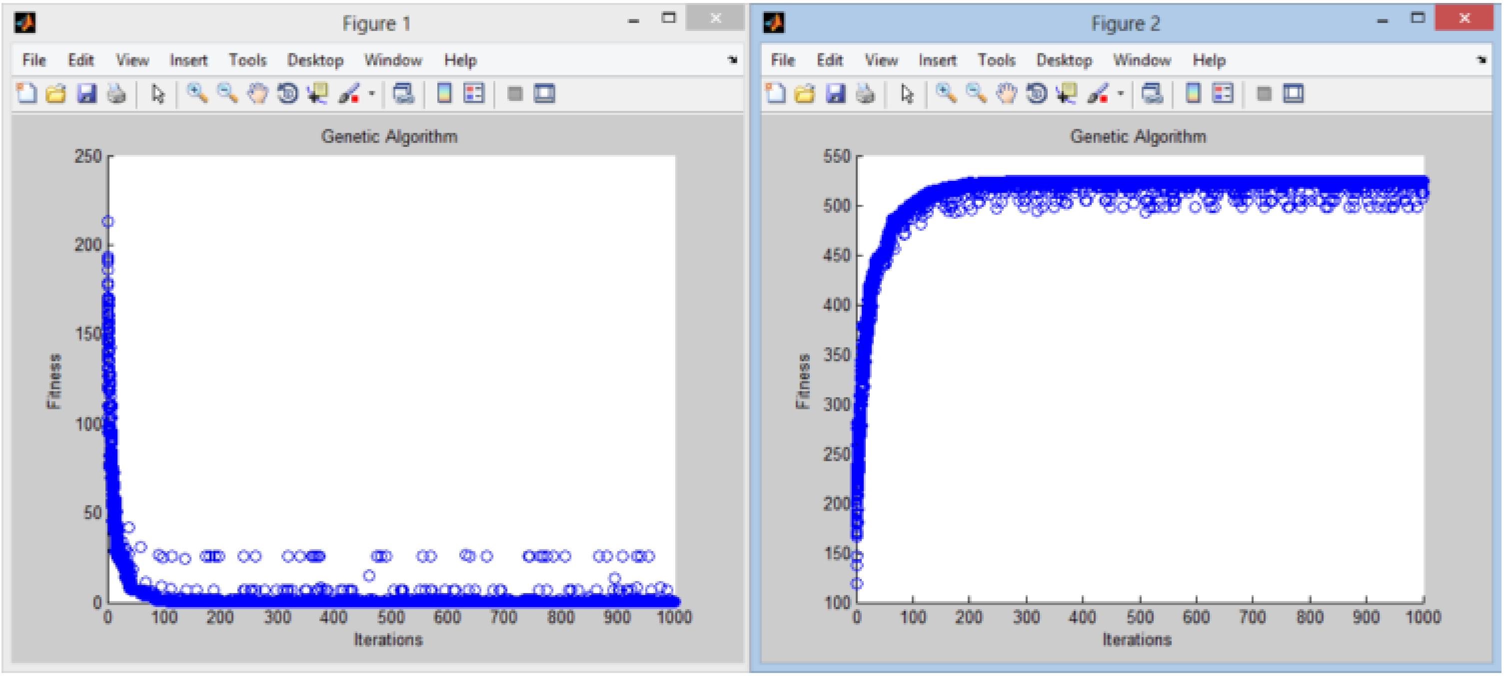


Figure B.8: Optimisation results of a genetic algorithm using binary tournament selection and two-point crossover implemented in Shark

B.2 Evolution Strategies

B.2.1 (μ,λ) Selection With Discrete Recombination

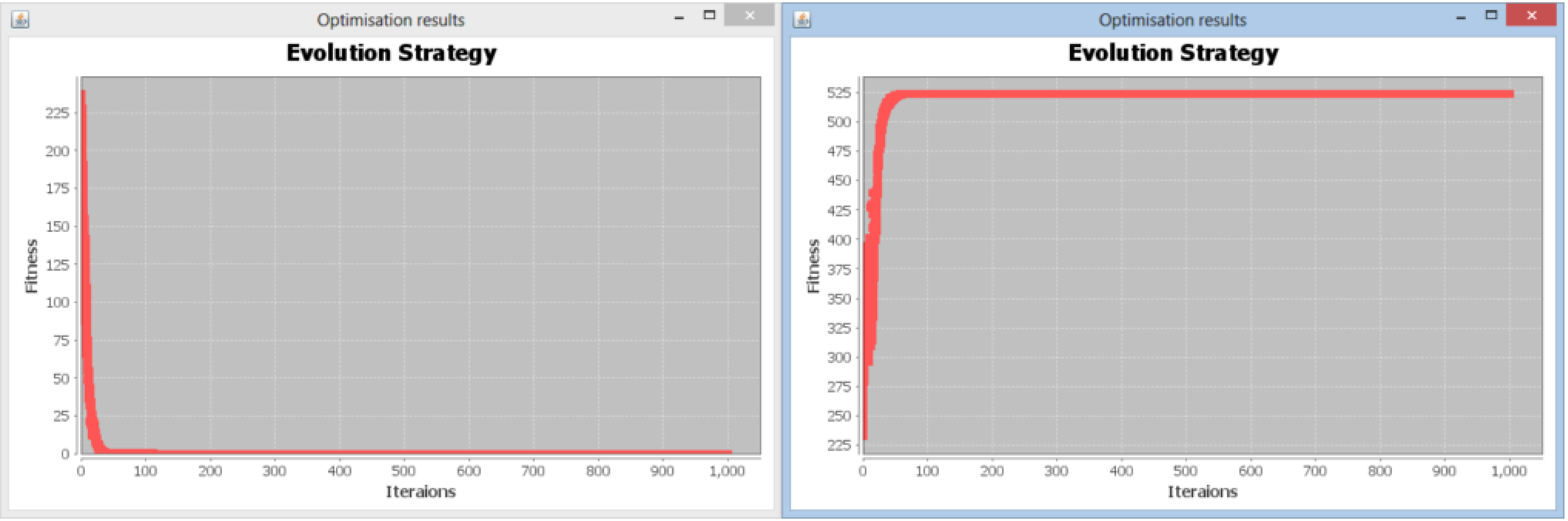


Figure B.9: Optimisation results of evolution strategies using (μ,λ) selection and discrete recombination

B.2.2 (μ, λ) Selection With Intermediate Recombination

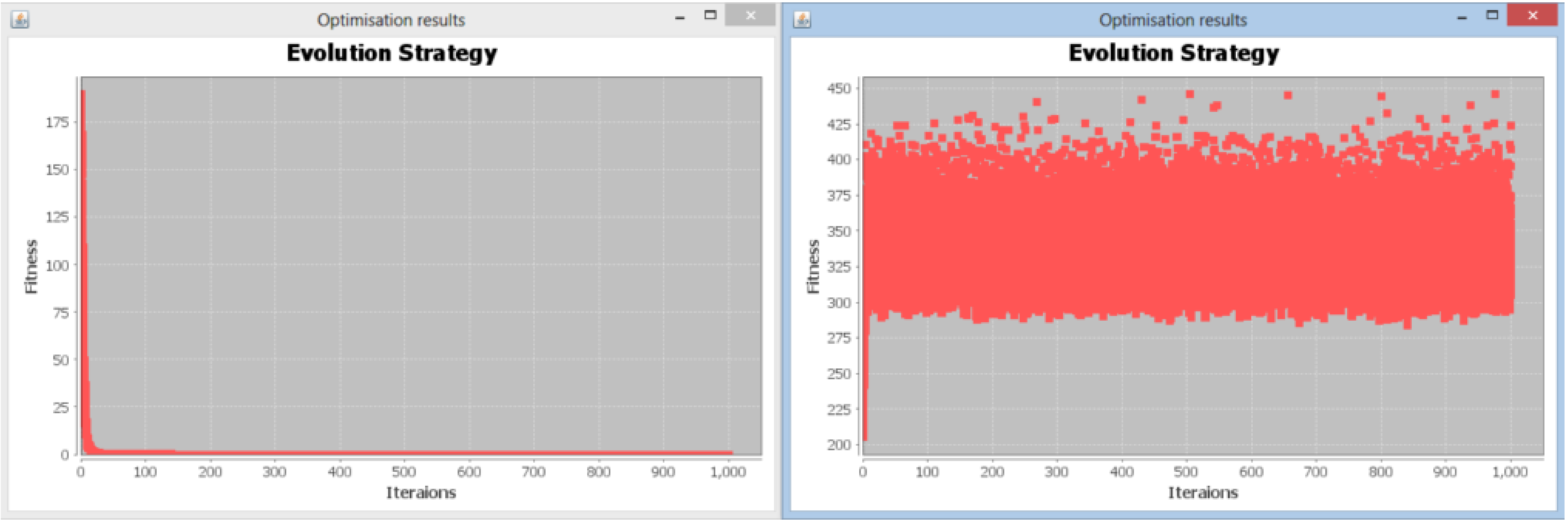


Figure B.10: Optimisation results of evolution strategies using (μ, λ) selection and intermediate recombination

B.2.3 (μ, λ) Selection With Global Discrete Recombination

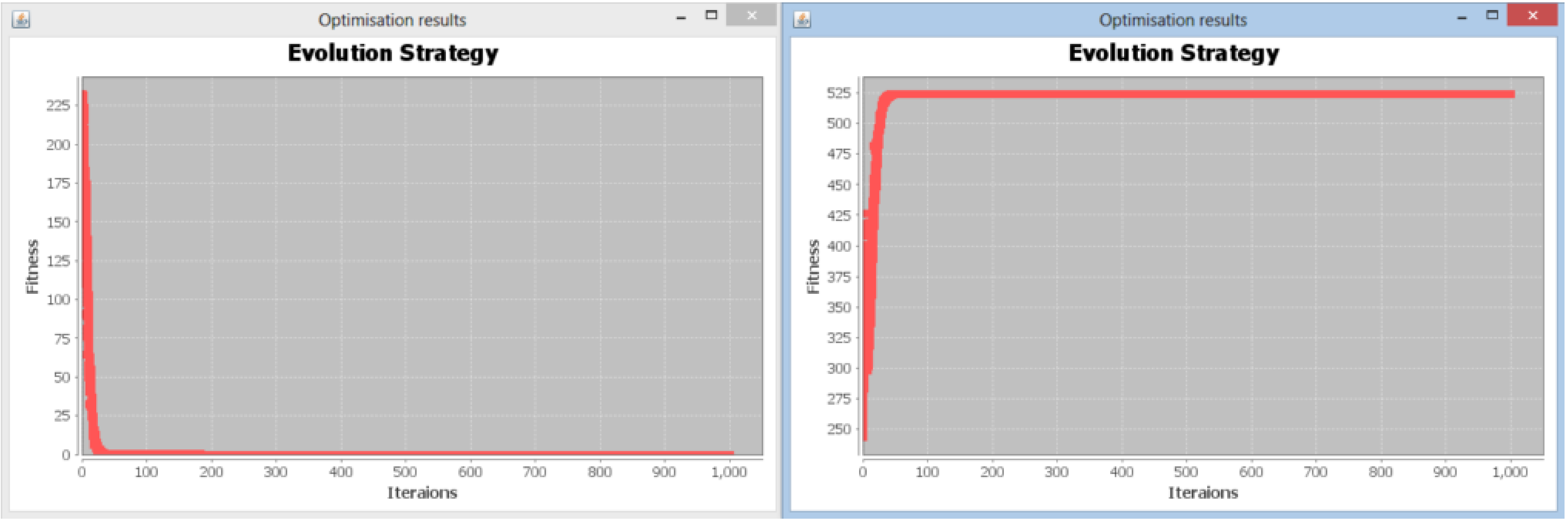


Figure B.11: Optimisation results of evolution strategies using (μ, λ) selection and global discrete recombination

B.2.4 (μ,λ) Selection With Global Intermediate Recombination

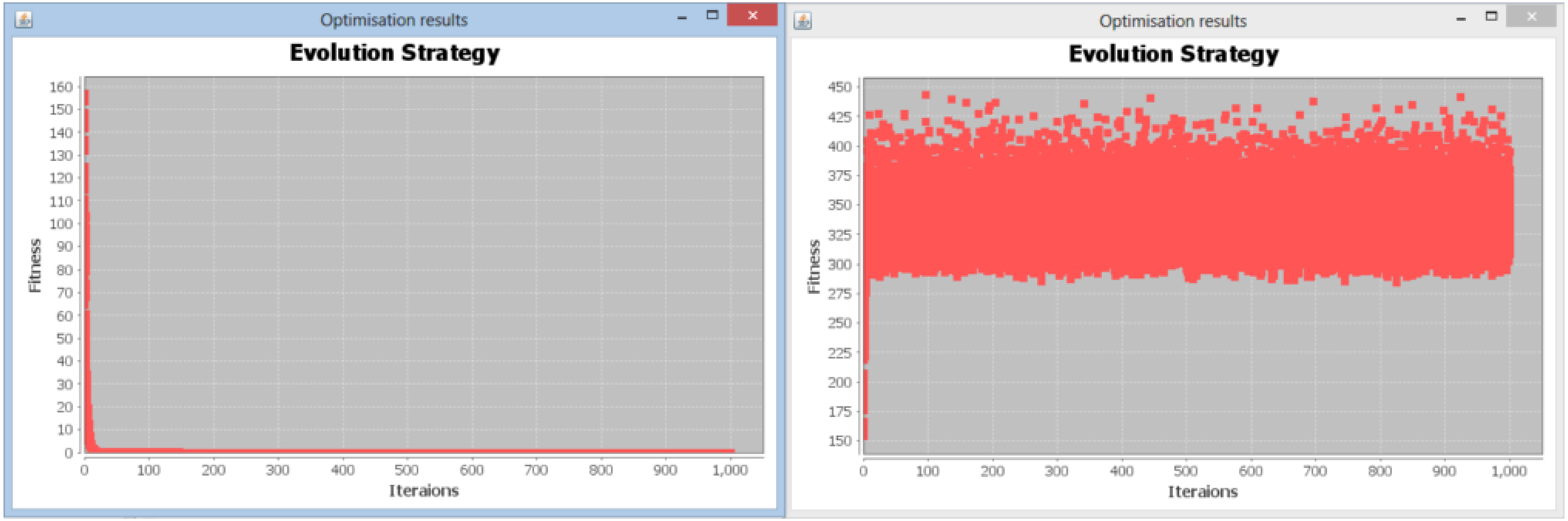


Figure B.12: Optimisation results of evolution strategies using (μ,λ) selection and global intermediate recombination

B.2.5 $(\mu + \lambda)$ Selection With Discrete Recombination

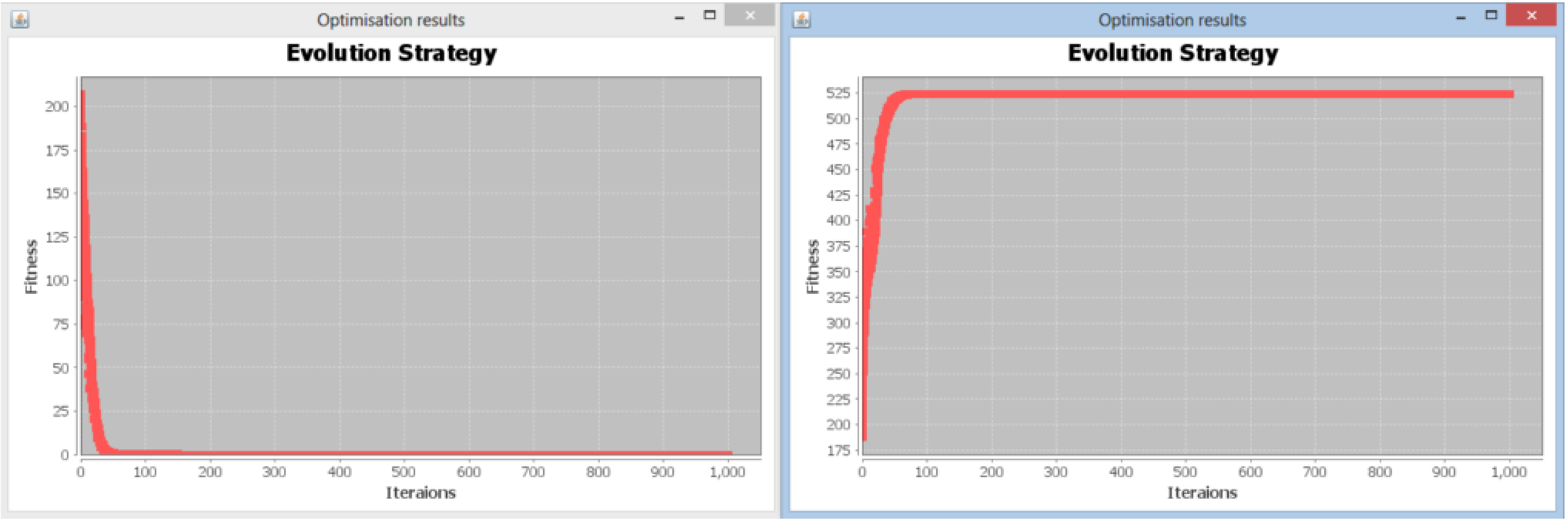


Figure B.13: Optimisation results of evolution strategies using $(\mu + \lambda)$ selection and discrete recombination

B.2.6 $(\mu+\lambda)$ Selection With Intermediate Recombination

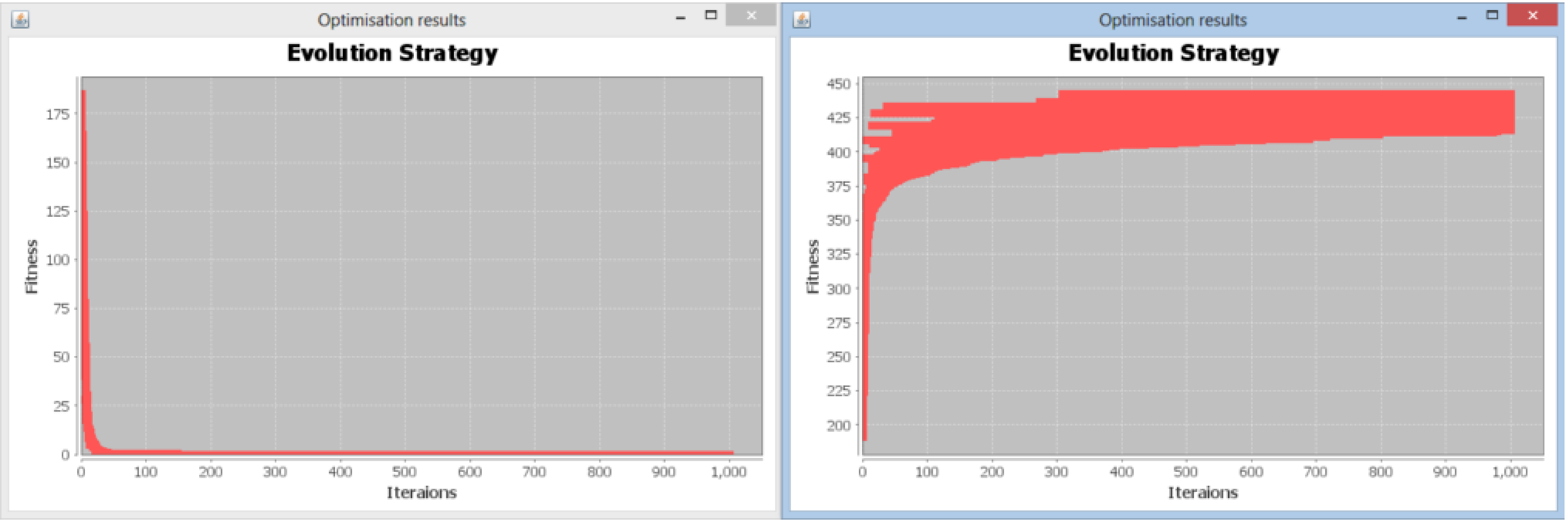


Figure B.14: Optimisation results of evolution strategies using $(\mu+\lambda)$ selection and intermediate recombination

B.2.7 $(\mu + \lambda)$ Selection With Global Discrete Recombination

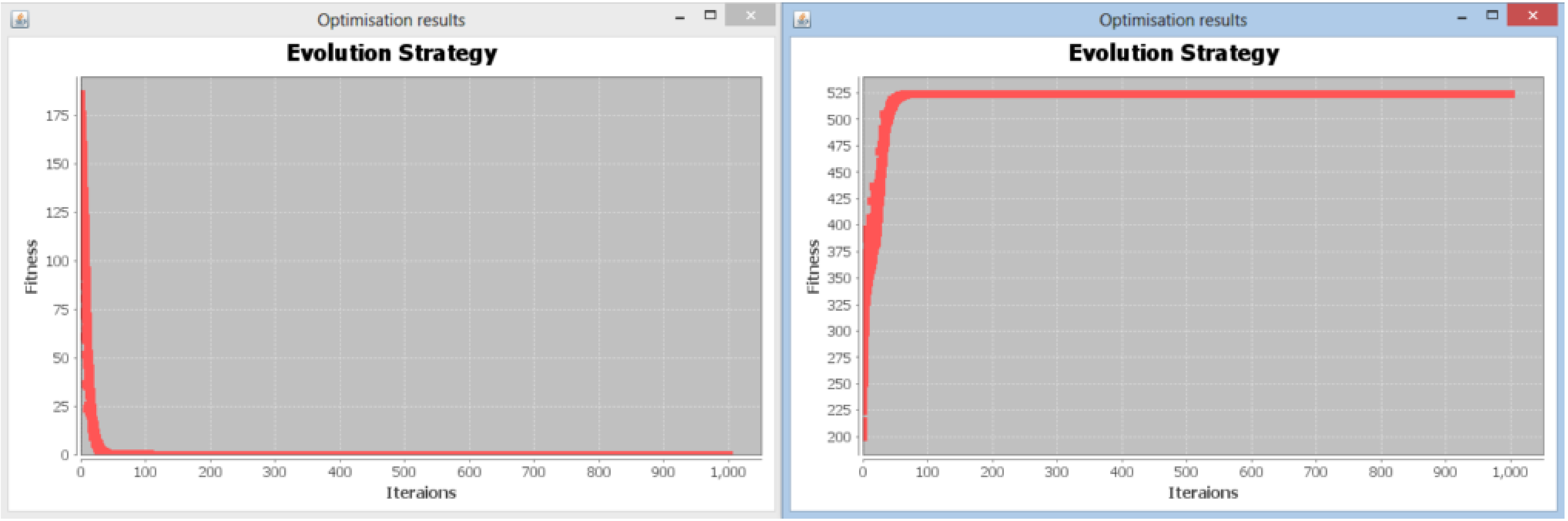


Figure B.15: Optimisation results of evolution strategies using $(\mu + \lambda)$ selection and global discrete recombination

B.2.8 $(\mu + \lambda)$ Selection With Global Intermediate Recombination

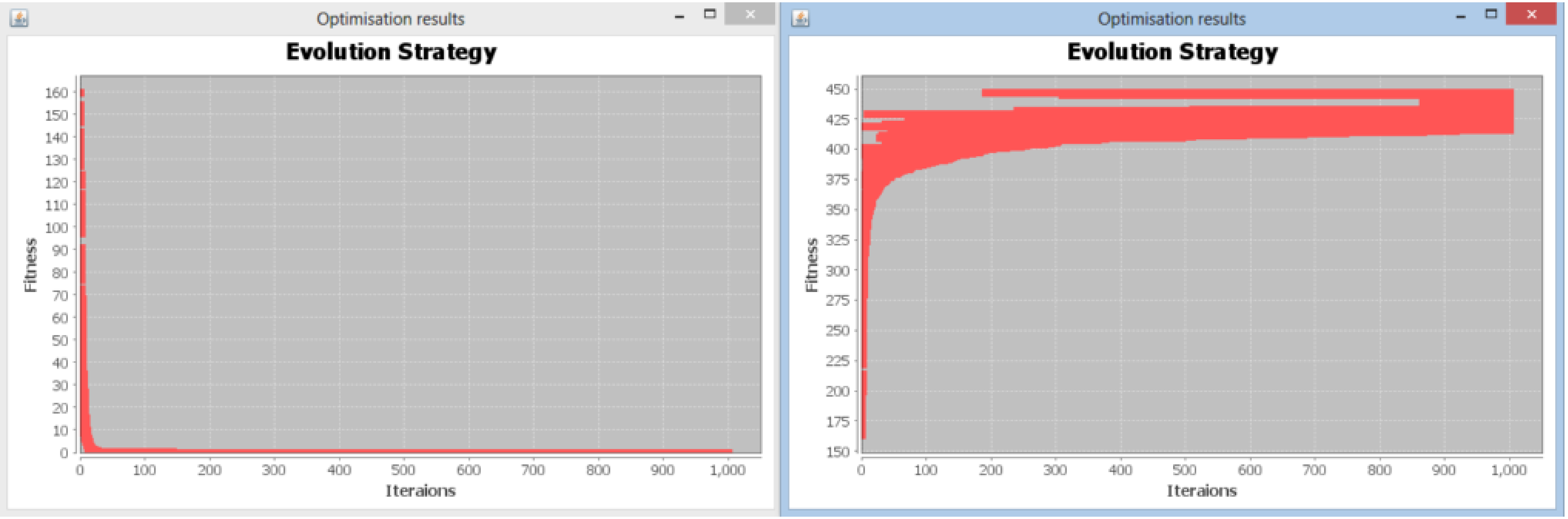


Figure B.16: Optimisation results of evolution strategies using $(\mu + \lambda)$ selection and global intermediate recombination

B.2.9 $(\mu + \lambda)$ Selection With Discrete Recombination in Shark

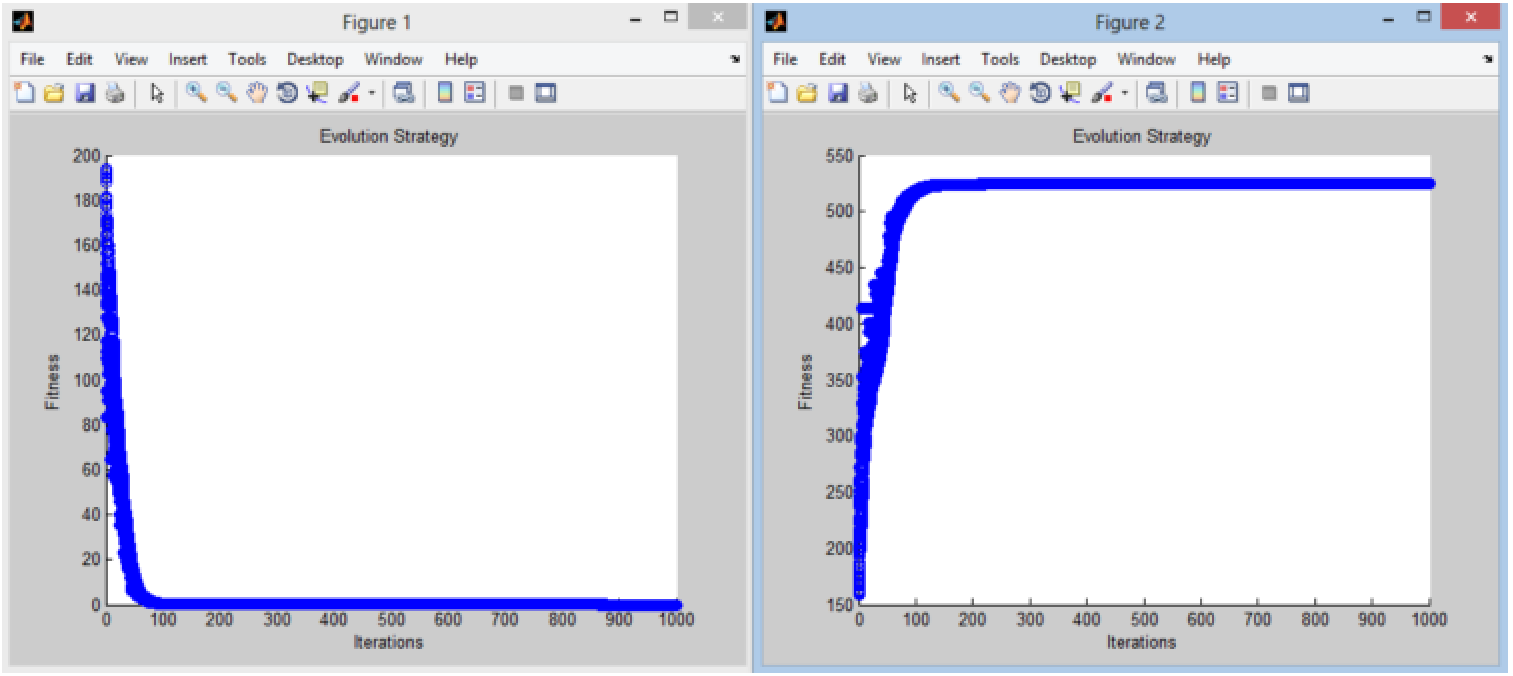


Figure B.17: Optimisation results of evolution strategies using $(\mu + \lambda)$ selection and discrete recombination implemented in Shark

B.3 Particle Swarm Optimisation

B.3.1 Optimisation Results

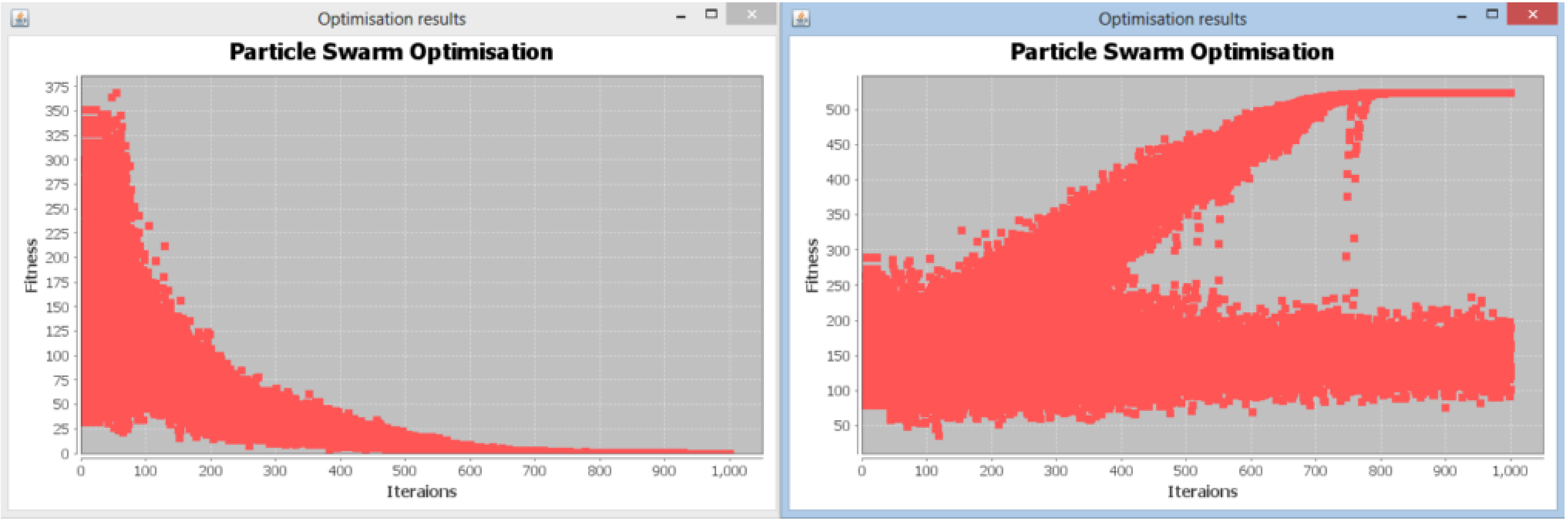


Figure B.18: Optimisation results of particle swarm optimisation

B.3.2 Optimisation Results in Shark

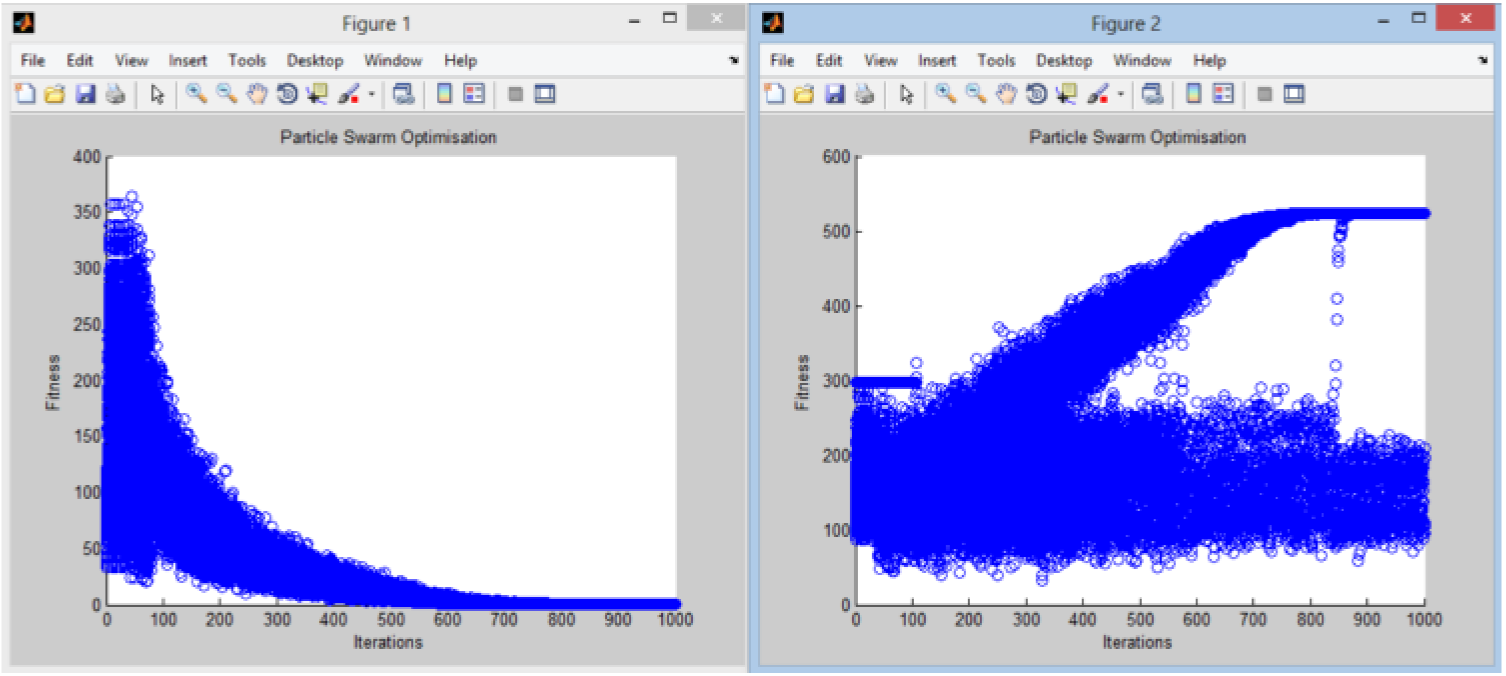


Figure B.19: Optimisation results of particle swarm optimisation implemented in Shark

References

Baraff, David and Andrew Witkin (1998). "Large Steps in Cloth Simulation." In: *SIGGRAPH*, pp. 43–54.

Bartels, Pieterjan (2014). *Simulation of Tearable Cloth*. URL: `https://nccastaff.bournemouth.ac.uk/jmacey/CGITech/reports2015/PBartels{\_}PieterjanBartels{\_}CGITECH{\_}final.pdf` (visited on 03/10/2016).

Choi, Kwang-Jin and Hyeong-Seok Ko (2002). "Stable but Responsive Cloth." In: *SIGGRAPH*, pp. 604–611.

Enqvist, Henrik (2010). *The Secrets Of Cloth Simulation In Alan Wake*. URL: `http://www.gamasutra.com/view/feature/132771/the{\_}secrets{\_}of{\_}cloth{\_}simulation{\_}in{\_}.php?page=1` (visited on 03/03/2016).

Jakobsen, Thomas (2005). *Advanced Character Physics*. URL: `http://www.gotoandplay.it/{\_}articles/2005/08/advCharPhysics.php` (visited on 03/03/2016).

Kang, Young-Min, Jeong-Hyeon Choi, and Hwan-Gue Cho (2000). "Fast and Stable Animation of Cloth with an Approximated Implicit Method." In: *Computer Graphics International*. Geneva.

Lander, Jeff (2000b). *Devil in the Blue Faceted Dress: Real Time Cloth Animation*. URL: `http://www.gamasutra.com/view/feature/131851/devil{\_}in{\_}the{\_}blue{\_}faceted{\_}dress{\_}.php` (visited on 03/03/2016).

Magnenat-Thalmann, Nadia and D. Thalmann (2004). *Handbook of Virtual Humans*. 1st. Chichester: Wiley.

Mongus, D. et al. (2012). "A hybrid evolutionary algorithm for tuning a cloth-simulation model." In: *Applied Soft Computing Journal*.

Ng, Hing N. and Richard L. Grimsdale (1996). "Computer Graphics Techniques for Modeling Cloth." In: *IEEE Computer Graphics & Applications* 16, pp. 28–42.

Parent, Rick (2012). *Computer Animation Algorithms and Techniques*. Third. Waltham: Elsevier. URL: `https://www.dawsonera.com/readonline/9780124159730/startPage/20`.

Provot, Xavier (1995). "Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behaviour." In: *Graphics Interface'95*, pp. 141–155.

Vassilev, T, B Spanlang, and Y Chrysanthou (2001). "Fast Cloth Animation on Walking Avatars." In: *Eurographics*.

Volino, Pascal, Frederic Cordier, and Nadia Magnenat-Thalmann (2005). "From early virtual garment simulation to interactive fashion design." In: *Computer-Aided Design* 37.6, pp. 593–608.

Volino, Pascal and Nadia Magnenat-Thalmann (2001). "Comparing Efficiency of Integration Methods." In: *Conference on Computer Graphics International*.

Wacker, Markus, Bernhard Thomaszewski, and Michael Keckeisen (2005). "Physical Models, Numerical Solvers for Cloth Animation and Virtual Cloth Design." In: *Eurographics*.

Wang, Jianchun, Xinrong Hu, and Yi Zhuang (2009). "The dynamic cloth simulation performance analysis based on the improved spring-mass model." In: *International Conference on Wireless Networks and Information Systems, WNIS 2009*, pp. 282 –285.

Xinrong, Hu et al. (2009). "Review of cloth modeling." In: *2009 Second ISECS International Colloquium on Computing, Communication, Control, and Management, CCCM 2009*.

Yalçın, M Adil and Cansın Yıldız. *Techniques for Animating Cloth*. URL: `http://www.cs.bilkent.edu.tr/{~}cansin/projects/cs567-animation/cloth/cloth-paper.pdf` (visited on 05/25/2016).

Zhang, Dongliang and Matthew M F Yuen (2001). "Cloth simulation using multilevel meshes." In: *Computers and Graphics (Pergamon)*.

Zink, Nico and Alexandre Hardy (2007). "Cloth Simulation and Collision Detection using Geometry Images." In: *AFRIGRAPH*. New York: ACM, pp. 187–195.

Bibliography

Akiya, Naoko, Scott Bury, and John M. Wassick (2011). "A General Model for Soft Body Simulation in Motion." In: *Winter Simulation Conference*. 2010, pp. 2194–2205.

Baraff, David and Andrew Witkin (1998). "Large Steps in Cloth Simulation." In: *SIGGRAPH*, pp. 43–54.

Bartels, Pieterjan (2014). *Simulation of Tearable Cloth*. URL: `https://nccastaff.bournemouth.ac.uk/jmacey/CGITech/reports2015/PBartels{\_}PieterjanBartels{\_}CGITECH{\_}final.pdf` (visited on 03/10/2016).

Bender, Jan, Daniel Weber, and Raphael Diziol (2013). "Fast and stable cloth simulation based on multi-resolution shape matching." In: *Computers and Graphics (Pergamon)*.

Bourg, David M. and Bryan Bywalec (2013). *Physics for Game Developers*. Second. Sebastopol: O'Reilly. ISBN: ISBN 9781449361051. URL: `https://www.dawsonera.com/abstract/9781449361051`.

Boxerman, Eddy (2003). "Speeding Up Cloth Simulation." Master's Thesis. The University of British Columbia.

Bridson, Robert, Ronald Fedkiw, and John Anderson (2002). "Robust Treatment of Collisions, Contact and Friction for Cloth Animation." In: *SIGGRAPH*, pp. 594–603.

Choi, Kwang-Jin and Hyeong-Seok Ko (2002). "Stable but Responsive Cloth." In: *SIGGRAPH*, pp. 604–611.

— (2005). "Research problems in clothing simulation." In: *Computer Aided Design* 37, pp. 585–592.

Conger, David (2004). *Physics Modeling for Game Programmers*. First. Boston: Course Technology / Cengage Learning. URL: `http://site.ebrary.com/lib/staffordshire/reader.action?docID=10065752`.

De Farias, Thiago S M C et al. (2008). "A high performance massively parallel approach for real time deformable body physics simulation." In: *Proceedings - Symposium on Computer Architecture and High Performance Computing*.

Desbrun, Mathieu, Peter Schröder, and Alan Barr (1999). “Interactive Animation of Structured Deformable Objects.” In: *Graphics Interface*. San Francisco: Morgan Kaufmann Publishers Inc., pp. 1–8.

Enqvist, Henrik (2010). *The Secrets Of Cloth Simulation In Alan Wake*. URL: `http://www.gamasutra.com/view/feature/132771/the{\_}secrets{\_}of{\_}cloth{\_}simulation{\_}in{\_}.php?page=1` (visited on 03/03/2016).

Ertekin, Burak (2014). *Cloth Simulation*. URL: `https://nccastaff.bournemouth.ac.uk/jmacey/CGITech/reports2015/BErtekin{\_}burakertekin-cgitech.pdf` (visited on 03/10/2016).

Garre, Carlos and Alvaro Pérez (2008). *A simple Mass-Spring system for Character Animation*. URL: `http://www.gmrv.es/{~}cgarre/APO{\_}MassSpring{\_}Jun08.pdf` (visited on 05/25/2016).

Gibson, Sarah F. F. and Brian Mirtich (1997). *A Survey of Deformable Modeling in Computer Graphics*. Tech. rep. Cambridge: MERL.

Harada, Takahiro (2006). “Real-time Cloth Simulation Interacting with Deforming High-Resolution Models.” In: *SIGGRAPH*.

— (2008). “Real-Time Rigid Body Simulation on GPUs.” In: *GPU Gems 3*. Pearson Education Inc. Chap. 29. URL: `http://http.developer.nvidia.com/GPUGems3/gpugems3{\_}ch29.html`.

Jakobsen, Thomas (2005). *Advanced Character Physics*. URL: `http://www.gotoandplay.it/{\_}articles/2005/08/advCharPhysics.php` (visited on 03/03/2016).

Kang, Young-Min, Jeong-Hyeon Choi, and Hwan-Gue Cho (2000). “Fast and Stable Animation of Cloth with an Approximated Implicit Method.” In: *Computer Graphics International*. Geneva.

Kim, Tae-Yong (2011). *Character Clothing in PhysX-3*.

Lander, Jeff (2000a). *Collision Response: Bouncy, Trouncy, Fun*. URL: `http://www.gamasutra.com/view/feature/3427/collision{\_}response{\_}bouncy{\_}.php` (visited on 03/24/2016).

— (2000b). *Devil in the Blue Faceted Dress: Real Time Cloth Animation*. URL: `http://www.gamasutra.com/view/feature/131851/devil{\_}in{\_}the{\_}blue{\_}faceted{\_}dress{\_}.php` (visited on 03/03/2016).

Magnenat-Thalmann, Nadia and D. Thalmann (2004). *Handbook of Virtual Humans*. 1st. Chichester: Wiley.

Mesit, Jaruwan, Ratan Guha, and Shafaq Chaudhry (2007). “3D soft body simulation using mass-spring system with internal pressure force and simplified implicit integration.” In: *Journal of Computers*.

Mesit, Jaruwan and Ratan K. Guha (2008). “Soft Body Simulation with Leaking Effect.” In: *2008 Second Asia International Conference on Modelling & Simulation (AMS)*. IEEE, pp. 390–395.

Miguel, E et al. (2012). “Data-Driven Estimation of Cloth Simulation Models.” In: *Computer Graphics Forum* 31.2, pp. 519–528.

Mongus, D. et al. (2012). “A hybrid evolutionary algorithm for tuning a cloth-simulation model.” In: *Applied Soft Computing Journal*.

Müller, Matthias et al. (2006). “Position Based Dynamics.” In: *VRIPHYS*, pp. 71–80.

Ng, Hing N. and Richard L. Grimsdale (1996). “Computer Graphics Techniques for Modeling Cloth.” In: *IEEE Computer Graphics & Applications* 16, pp. 28–42.

NVidia (2007). *Cloth Simulation*. URL: `http://developer.download.nvidia.com/SDK/10/direct3d/Source/Cloth/doc/Cloth.pdf` (visited on 05/25/2016).

O ’connor, Corey and Keith Stevens (2003). *Modeling Cloth Using Mass Spring Systems*. (Visited on 03/10/2016).

Ozgen, Oktar et al. (2010). “Underwater Cloth Simulation with Fractional Derivatives.” In: *ACM Trans. Graph* 29.23.

Parent, Rick (2012). *Computer Animation Algorithms and Techniques*. Third. Waltham: Elsevier. URL: `https://www.dawsonera.com/readonline/9780124159730/startPage/20`.

Plath, Jan (2000). “Realistic modelling of textiles using interacting particle systems.” In: *Computers & Graphics* 24, pp. 897–905.

Provot, Xavier (1995). “Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behaviour.” In: *Graphics Interface’95*, pp. 141–155.

Santos Souza, Marco, Aldo Von Wangenheim, and Eros Comunello (2014). “Fast Simulation of Cloth Tearing.” In: *SBC Journal on Interactive Systems* 5.1, pp. 44–48.

Schmitt, N et al. (2013). “Multilevel Cloth Simulation using GPU Surface Sampling.” In: *Workshop on Virtual Reality Interaction and Physical Simulation VRIPHYS*.

Selle, Andrew et al. (2009). “Robust High-Resolution Cloth Using Parallelism, History-Based Collisions and Accurate Friction.” In: *IEEE Transactions on Visualization and Computer Graphics* 15, pp. 339–350.

Tang, Min et al. (2013). “A GPU-based streaming algorithm for high-resolution cloth simulation.” In: *Computer Graphics Forum*.

Vassilev, T, B Spanlang, and Y Chrysanthou (2001). “Fast Cloth Animation on Walking Avatars.” In: *Eurographics*.

Vassilev, Tzvetomir and Bernhard Spanlang (2002). “A Mass-Spring Model For Real Time Deformable Solids.” In: *East-West Vision*.

Vassilev, Tzvetomir Ivanov (2010). “Comparison of Several Parallel API for Cloth Modelling On Modern GPUs.” In: *CompSysTech*, pp. 131–136.

— (2011). “Comparison of Parallel Algorithms for Modelling Mass-springs Systems with Several APIs on Modern GPUs.” In: *CompSysTech*, pp. 204–209.

Volino, Pascal, Frederic Cordier, and Nadia Magnenat-Thalmann (2005). “From early virtual garment simulation to interactive fashion design.” In: *Computer-Aided Design* 37.6, pp. 593–608.

Volino, Pascal, Martin Courchesne, and Nadia Magnenat-Thalmann (1995). “Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects.” In: *SIGGRAPH*, pp. 137–144.

Volino, Pascal and Nadia Magnenat-Thalmann (1997a). “Developing Simulation Techniques for an Interactive Clothing System.” In: *International Conference on Virtual Systems and MultiMedia*. IEEE, pp. 109–118.

— (1997b). “Interactive Cloth Simulation: Problems and Solutions.” In: *JWS97-B*. Geneva.

— (2001). “Comparing Efficiency of Integration Methods.” In: *Conference on Computer Graphics International*.

Wacker, Markus, Bernhard Thomaszewski, and Michael Keckeisen (2005). “Physical Models, Numerical Solvers for Cloth Animation and Virtual Cloth Design.” In: *Eurographics*.

Wang, Jianchun, Xinrong Hu, and Yi Zhuang (2009). “The dynamic cloth simulation performance analysis based on the improved spring-mass model.” In: *International Conference on Wireless Networks and Information Systems, WNIS 2009*, pp. 282 –285.

Wang, Xiuzhong and Venkat Devarajan (2004). “2D Structured Mass-spring System Parameter Optimization based on Axisymmetric Bending for Rigid Cloth Simulation.” In: *SIGGRAPH*, pp. 317–323.

Xinrong, Hu et al. (2009). “Review of cloth modeling.” In: *2009 Second ISECS International Colloquium on Computing, Communication, Control, and Management, CCCM 2009*.

Yalçın, M Adil and Cansın Yıldız. *Techniques for Animating Cloth*. URL: `http://www.cs.bilkent.edu.tr/{~}cansin/projects/cs567-animation/cloth/cloth-paper.pdf` (visited on 05/25/2016).

Zeller, Cyril (2005). “Cloth Simulation On The GPU.” In: *SIGGRAPH*.

Zhang, Dongliang and Matthew M F Yuen (2001). “Cloth simulation using multilevel meshes.” In: *Computers and Graphics (Pergamon)*.

Zhenfang, Cao and He Bing (2012). “Research of Fast Cloth Simulation Based on Mass- Spring Model.” In: *National Conference on Information Technology and Computer Science*, pp. 323–327.

Zink, Nico and Alexandre Hardy (2007). "Cloth Simulation and Collision Detection using Geometry Images." In: *AFRIGRAPH*. New York: ACM, pp. 187–195.