Semester Project

Structural Design

LB490A Section 3, Spring Semester 2014
Methods of Computational Science
Max Smith

Problem Statement

To construct the optimal structure for supporting a surface reaching the furthest possible distance with only connection to one horizontal surface using Lego® bricks.

Figure 1: Australian modern home with long single supported balconies.[1]

Benefits of a Solution

- Support Structure Design
  - Docks
  - Balconies
  - Surfaces (tables, chairs, etc.)
  - Bridges
- Cost
- Excuse to play with Lego® bricks

Solution Design

Since this problem has exact no formula or ideology to construct the solution, the second best method for designing a solution is chosen: the genetic algorithm.

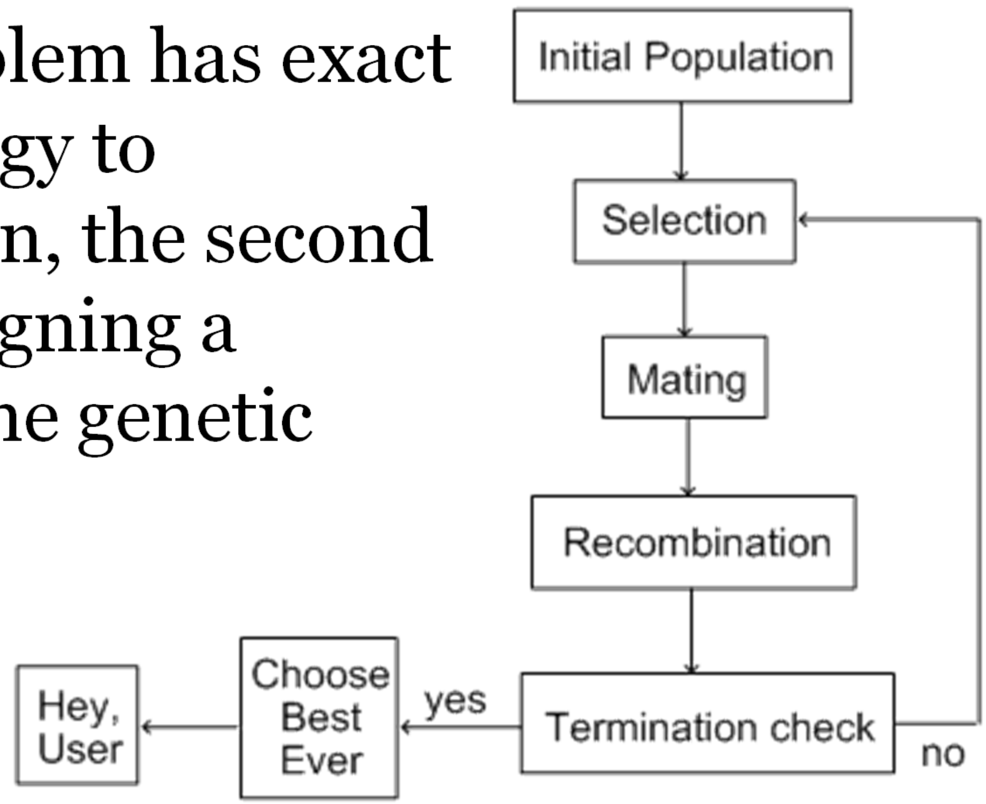


Figure 2: Simplistic flow chart of the canonical genetic algorithm[2]

Part 1 – Initial Population

The "initial population" consists of randomly generated structures Lego® structures with *N* bricks attached to a ground piece of length *s*, each structure being a "chromosome."

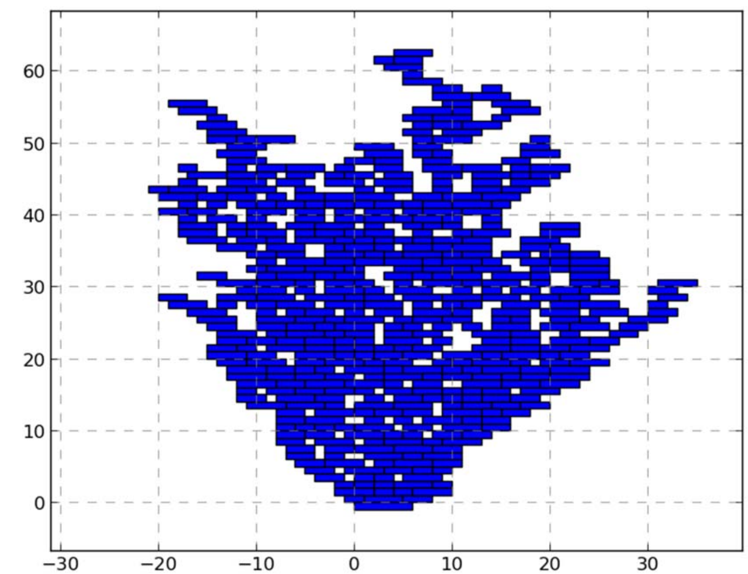


Figure 3: A single "chromosome" with *N*=500, and *s*=6.

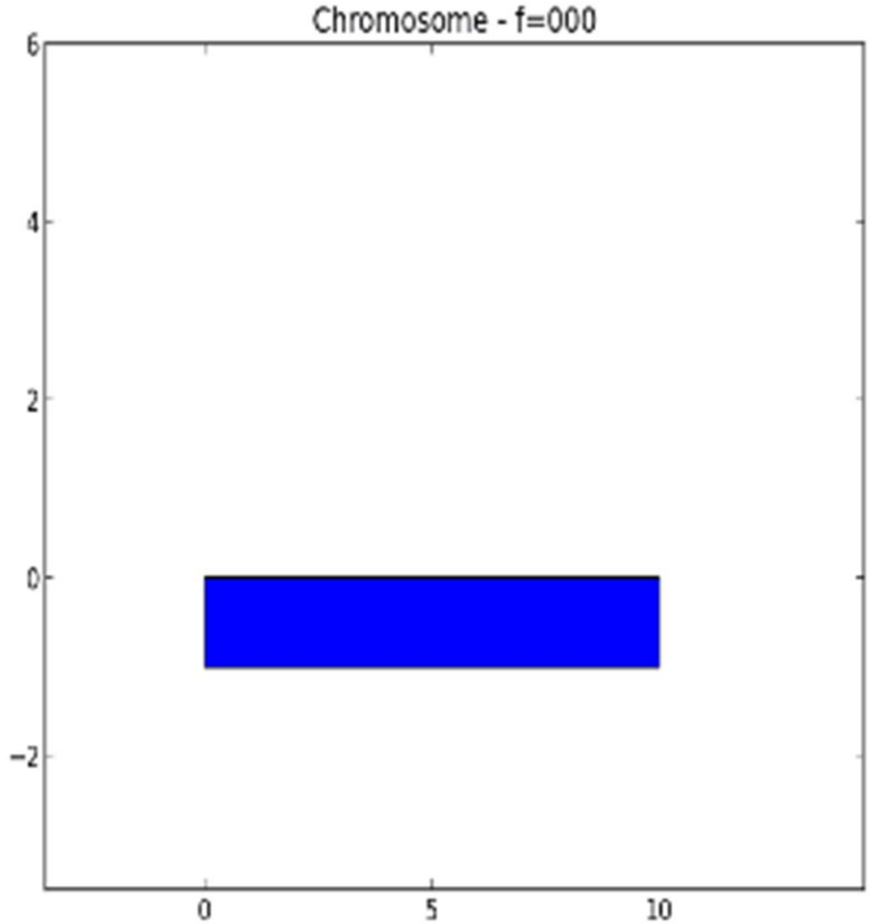


Figure 4: Initialization of a chromosome with N=100, and s=10

Part 2 - Selection

An intermediate population is selected from the current population based on fitness of the chromosomes.

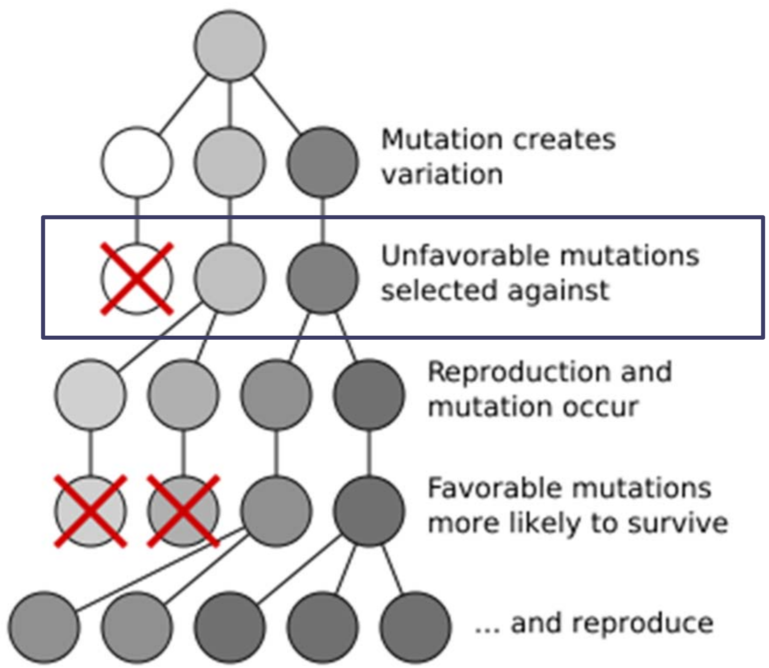


Figure 5: Figure depicting evolution, with the selection section outlined[3]

Fitness

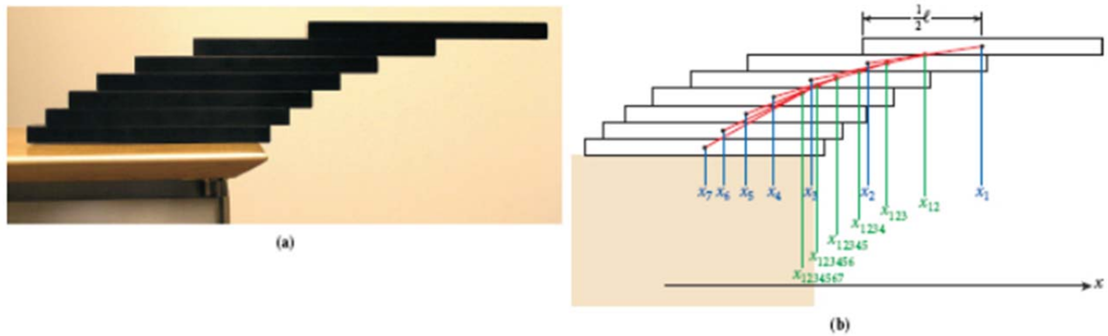


Figure 6: Stack of identical blocks showing center of mass calculation[7]

For this experiment fitness is calculated by finding the center of mass and scaling that value by how far the construct reaches to the right.

$$f = x_{max} \frac{\sum m_i x_i}{m_{total}}$$

$x \equiv x\ position$

$m \equiv mass$

Does not account for connection between bricks

```
def selection(pop=[]):
        '''
        Applies selection to the population
        '''
        pop_size = len(pop)
        list = []

        pop = np.unique(pop)
        #Remove empty chromosomes
        pop = [chrom for chrom in pop if len(chrom.plane.grid) > 1]

        #Remove unstable chromosomes
        pop = [chrom for chrom in pop if chrom.eval_func() > 0]

        #Format for weighted selection
        list = [[chrom.eval_func(), chrom] for chrom in pop]

        return WeightedSelectionWithReplacement(list, pop_size)
```

Figure 7: Code segment showing the selection function

Weighted Selection With Replacement

```
In [2]: x = [[10, 'a'], [2, 'b']]

In [3]: WeightedSelectionWithReplacement(x, 10)
Out[3]: ['b', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a', 'a']
```

Figure 8: Code segment showing weighted selection with replacement on an example array

Part 3 - Recombination

After selecting the more favorable chromosomes they must "mate" by crossing data between two chromosomes.

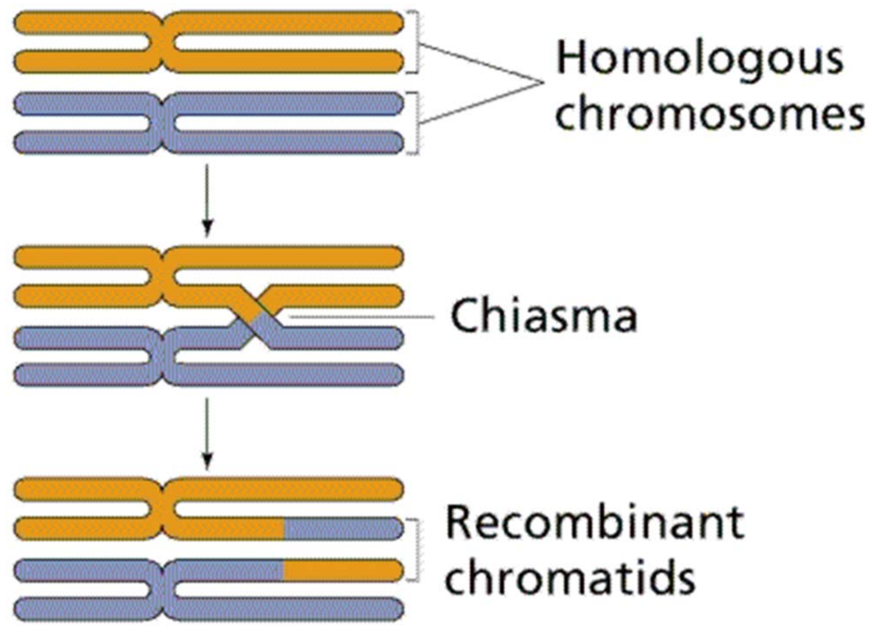


Figure 9: Flow chart showing chromosome crossover during cell division[4]

Splicing

The information for the Lego structures are the Lego pieces, so crossing over will be removing a chunk of bricks "splicing" and swapping it with a splice from another structure.

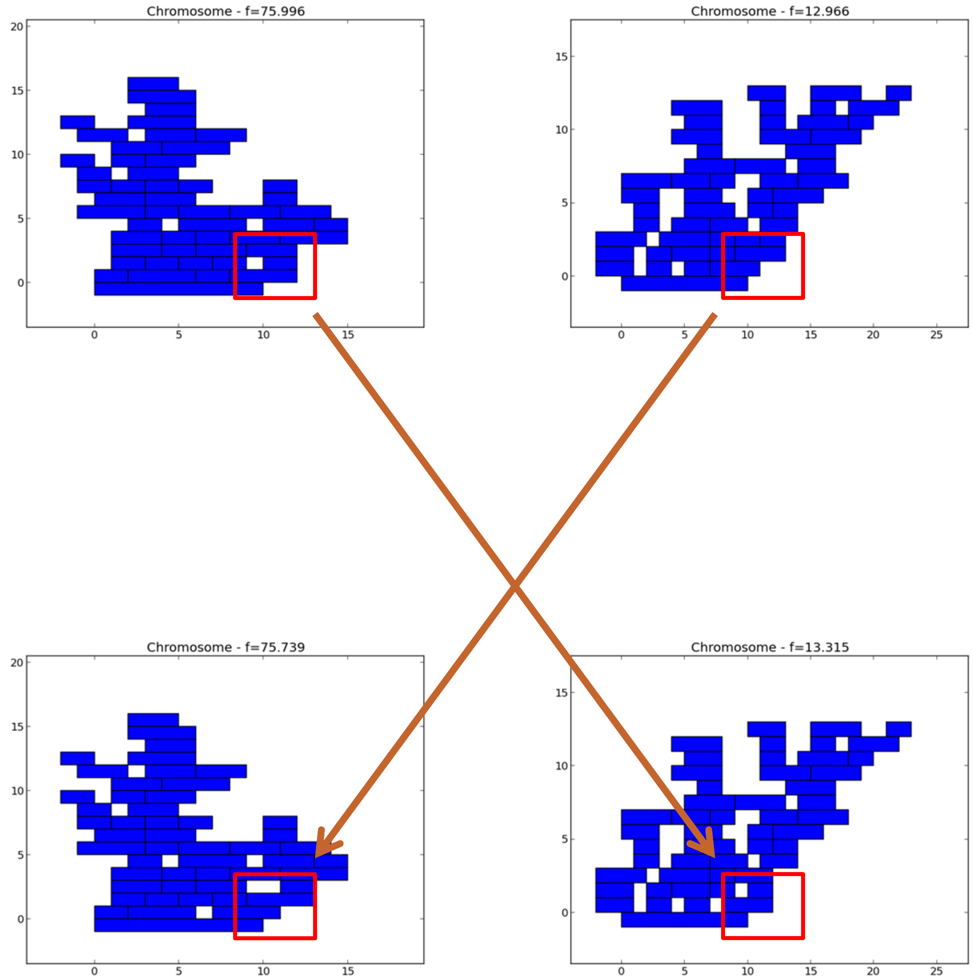
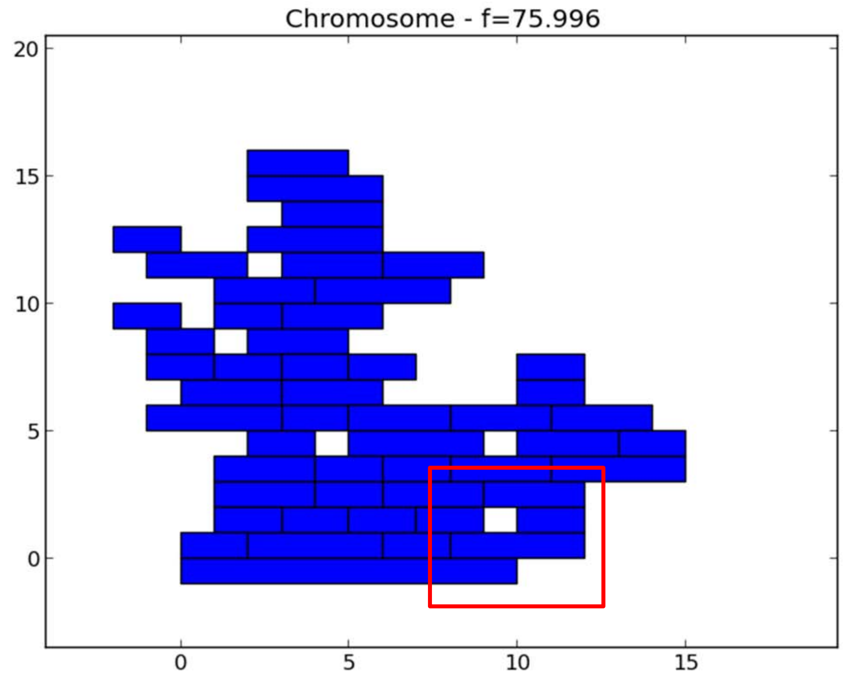


Figure 10: Splicing and crossover of two chromosomes, spliced portion highlighted in red.

Splicing Chromosome 1



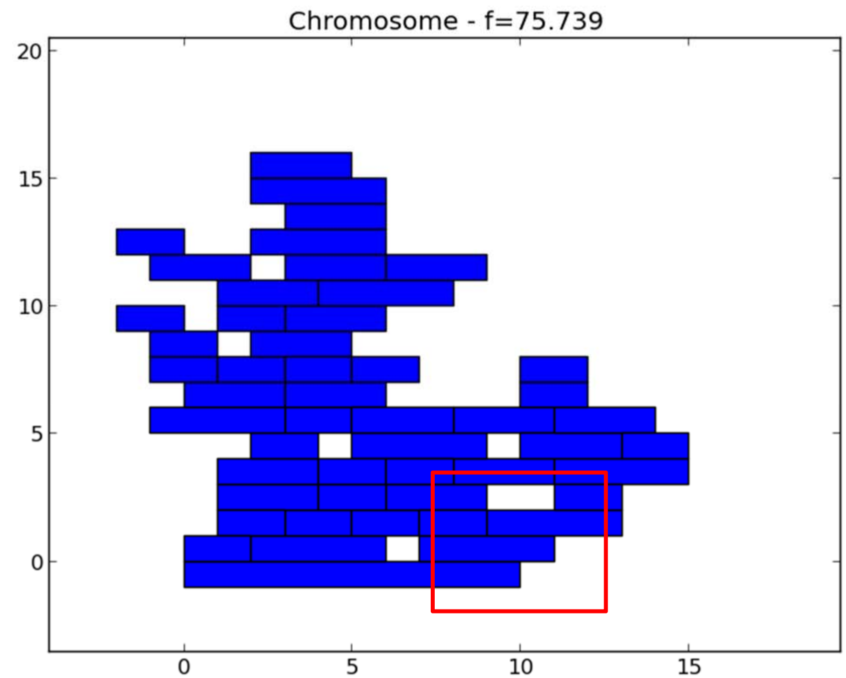


Figure 11.1 & 11.2: Enlarged image of the chromosomes with the crossover potion highlighted

Splicing Chromosome 2

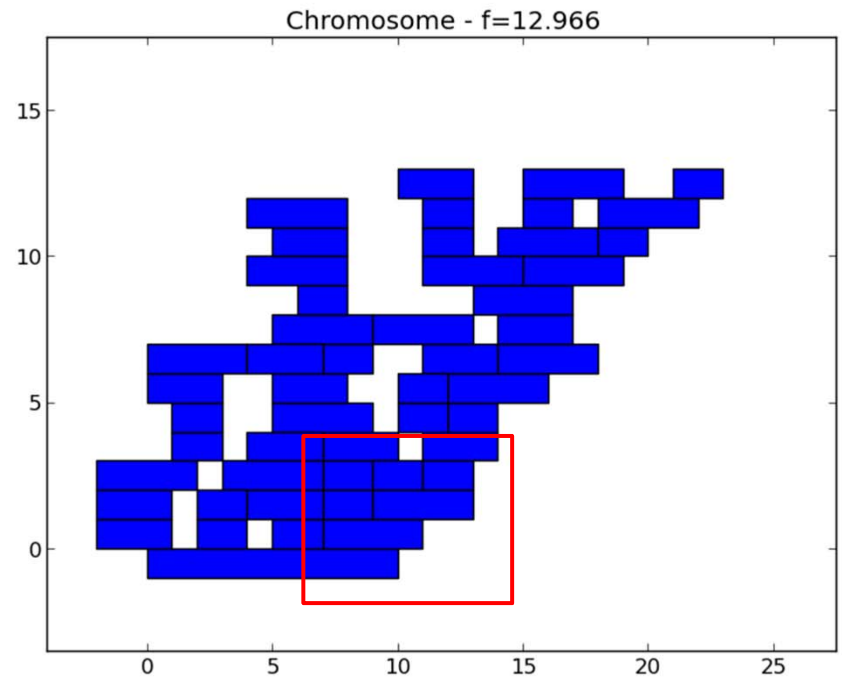
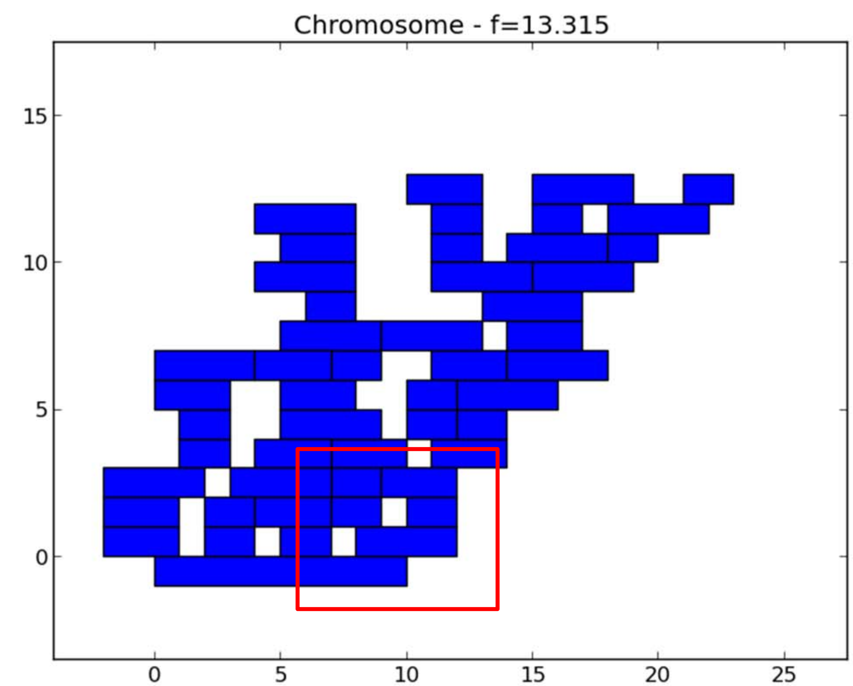


Figure 12.1 & 12.2: Enlarged image of the chromosomes with the crossover potion highlighted



Part 3 - Generations

The population was left to cycle through this process until it had completed G generations (where one generation is defined as one traversal through selection, recombination, and mutation).

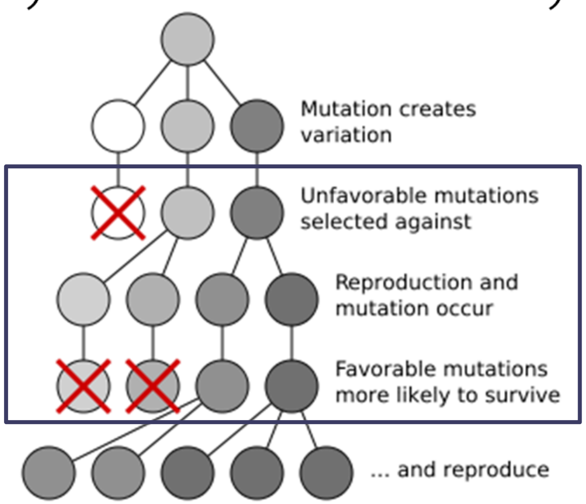


Figure 13: Flow chart of evolution with generations highlighted[3]

Part 4 - Conclusion

The system began to take advantage to the assumptions and set-up that I had created, exploiting it.

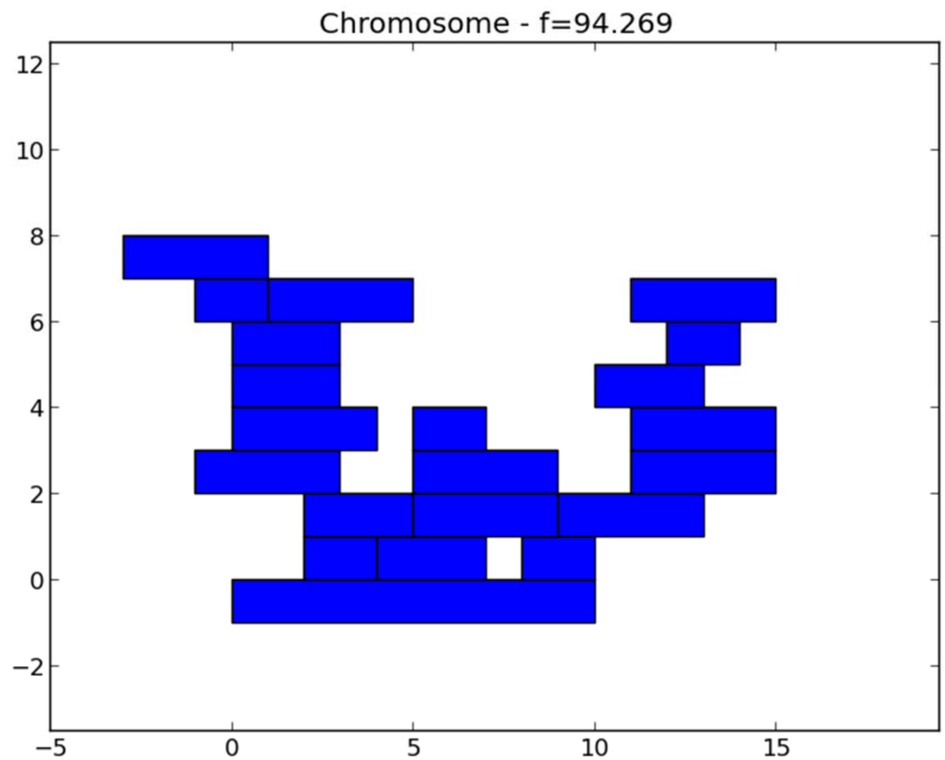


Figure 14: Enlarged image of the chromosomes with the crossover potion highlighted

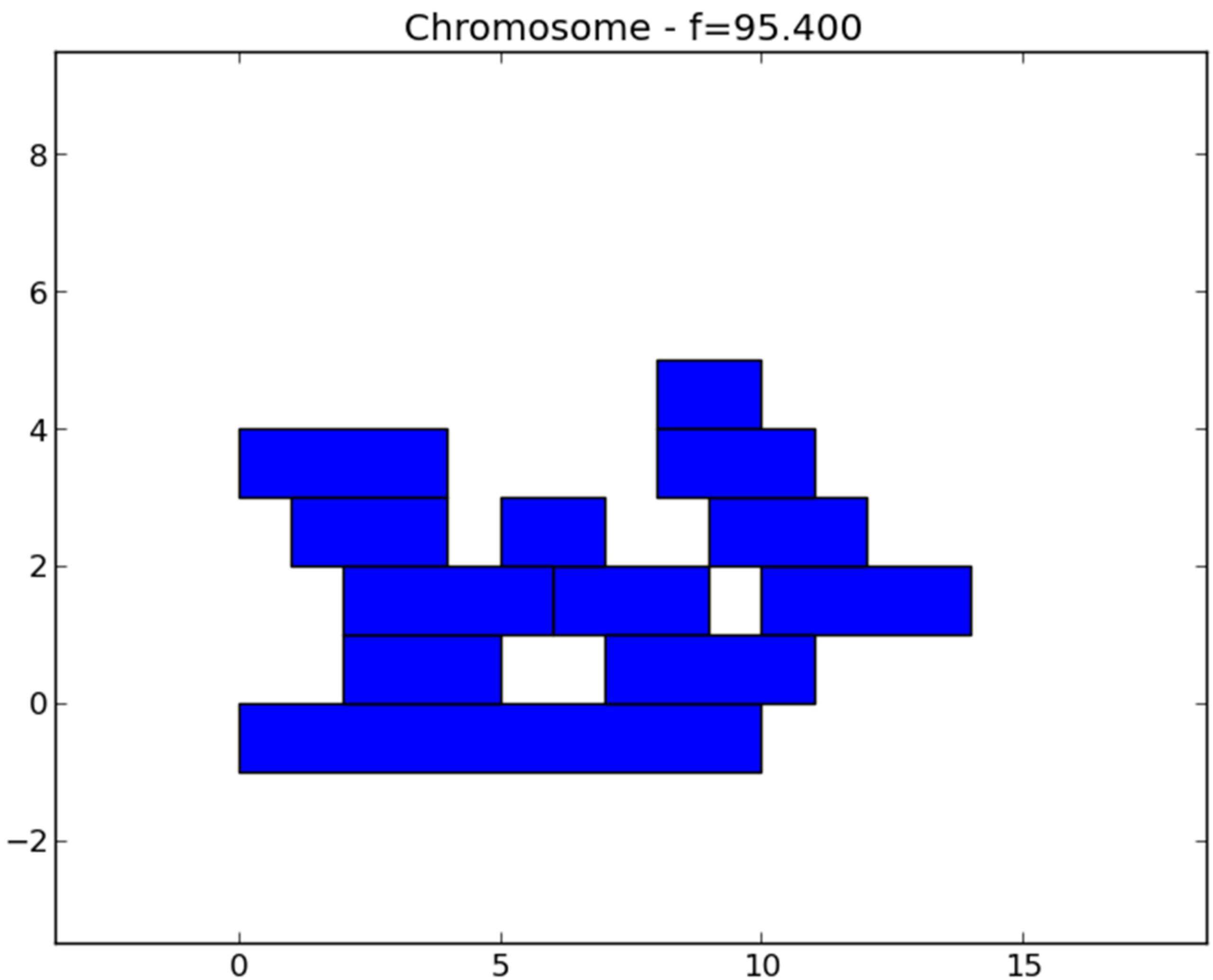


Figure 16: One of the resulting higher fitness chromosomes

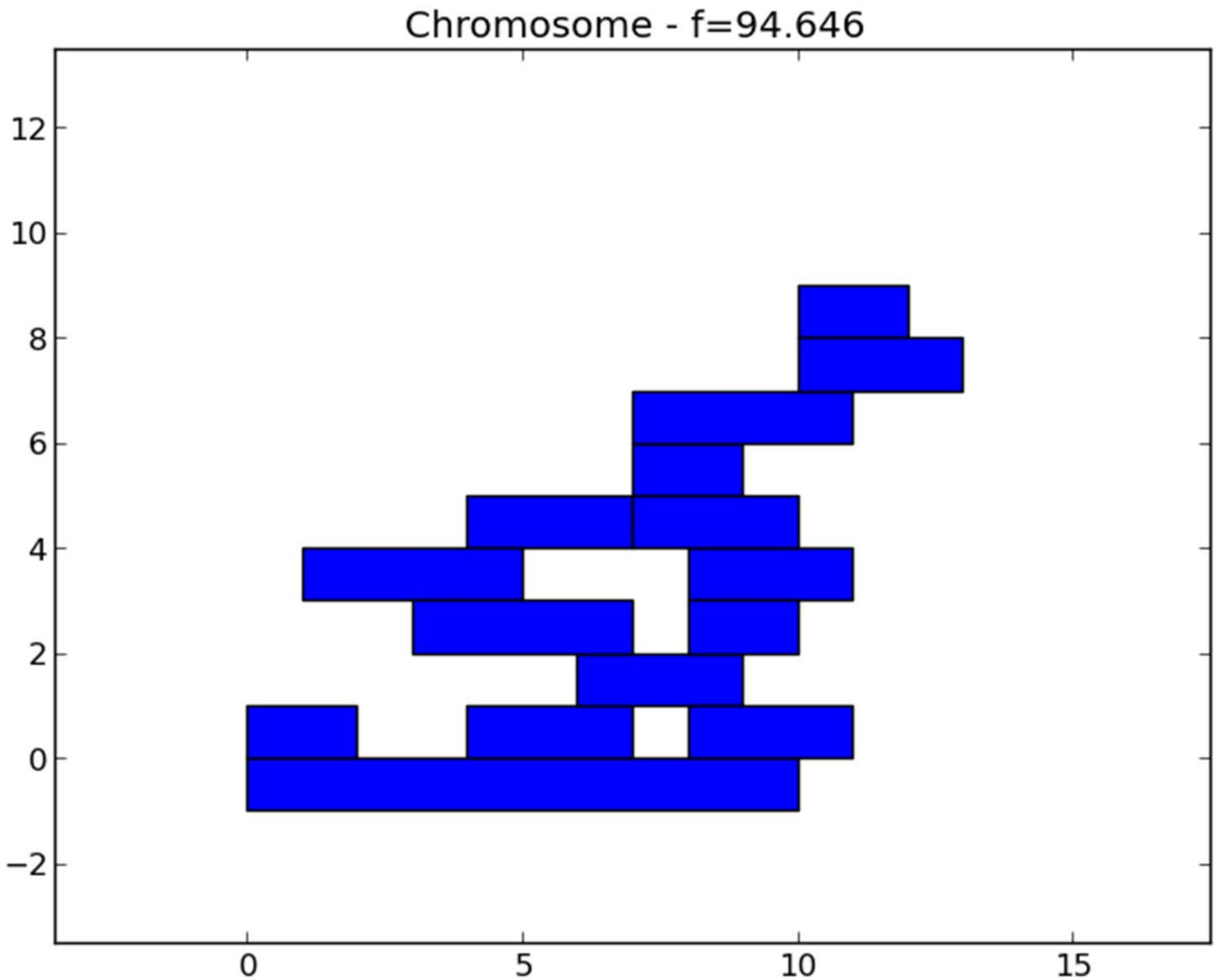


Figure 15: One of the resulting higher fitness chromosomes

Highest Fitness Chromosome

The most fit chromosome of this population

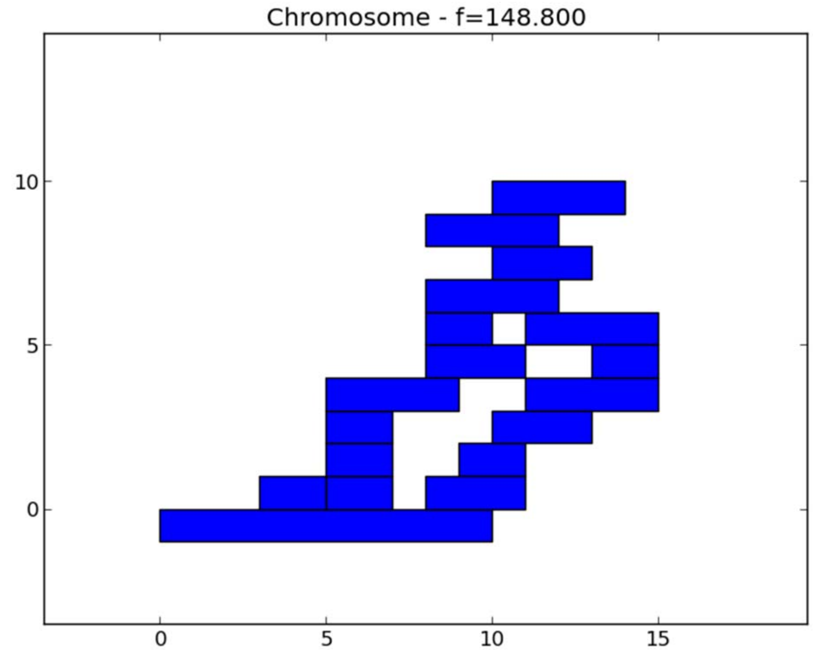


Figure 17: One of the resulting higher fitness chromosomes

Challenges

- Large variety of Lego shapes

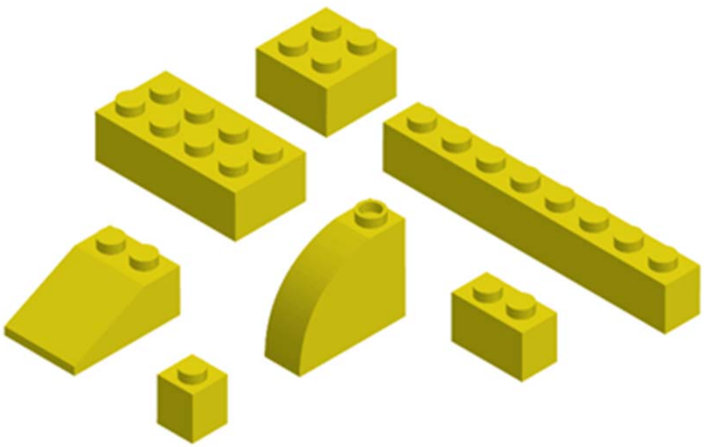


Figure 18: Various Lego® bricks[5]

- Complexity of computation in 3D & 2D

Challenges

- Framework
- Splicing: Removes bricks that occupy same space
- Fragments: Floating bricks

Figure 19: A floating Lego® piece[6]

Fragments

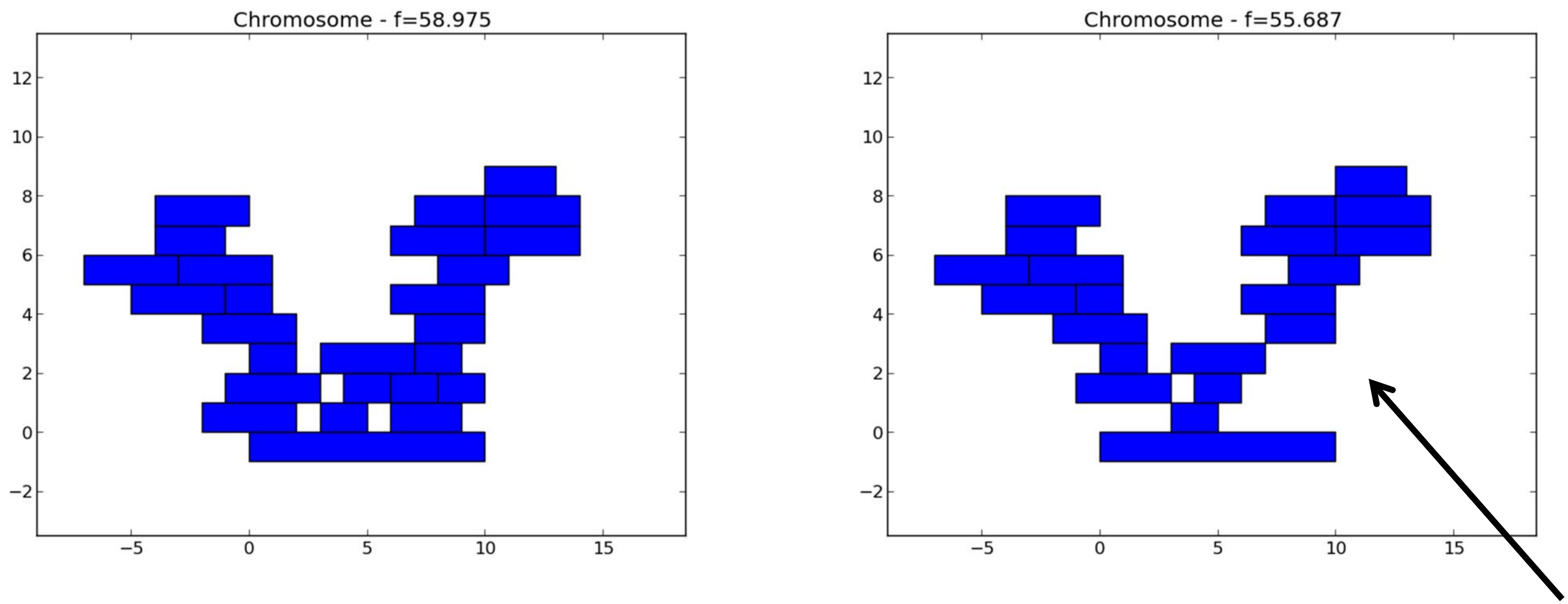


Figure 20.1 &20.2: A chromosome with several bricks removed to show a fragment error

Walltime Exceeded

PBS JOB 17598329.mgr-04.i Trash x

root <hpc@msu.edu> Apr 19 (1 day ago)

to smit2053

PBS Job Id: 17598329.mgr-04.i
Job Name: Lego_Structure
Exec host: css-006/5+css-006/6+css-006/7+css-006/8+css-006/9+css-006/10+css-006/11+css-006/12+css-006/13+css-006/14+css-002/4+css-002/5+css-002/6+css-002/7+css-002/8+css-002/9+css-002/10+css-002/11+css-002/12+css-002/13
Aborted by PBS Server
Job exceeded its walltime limit. Job was aborted
See Administrator for help

Figure 21: A walltime exceeded error

Acknowledgements

- Marcos Caballero, Ph.D.
- Brian O'Shea, Ph.D.
- William Punch, Ph.D.
- Devin Silvia, Ph.D.

Resources

[1]http://www.mabuno.com/simple-yet-sophisticated-modern-home-design-in-austria/shining-house-by-the-lake-with-glass-windows-and-long-balcony/

2http://www.doc.ic.ac.uk/~sgc/teaching/pre2012/v231/lecture16.html

[3]http://en.wikipedia.org/wiki/File:Mutation_and_selection_diagram.svg

[4]http://www2.estrellamountain.edu/faculty/farabee/biobk/biobookmeiosis.html

[5]http://3greatacts.com/the-nature-of-the-beast-ii-emergence/

[6]http://thebrickblogger.com/2012/07/lego-super-heroes-batman-2-video-game/

[7]http://textflow.mcgraw-hill.com//parser.php?secload=11.2&fake&print

[8]http://www.cs.colostate.edu/~genitor/MiscPubs/tutorial.pdf

[9]http://www.otlet-institute.org/wikics/2D_Genetic_Algorithms.html#toc-Section-6