Reducing graphic conflict in scale reduced maps using a genetic algorithm

Mark Ware, Ian Wilson and Andrew Ware

School of Computing
University of Glamorgan

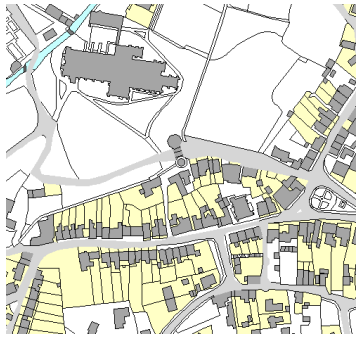
Problem Definition

- Automated production of reduced scale products from large scale digital map source.

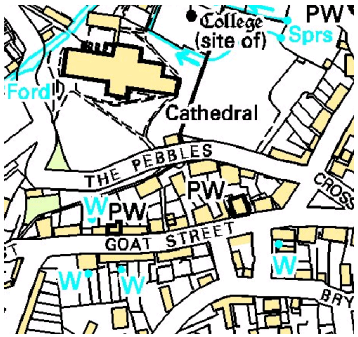
- Deriving new products from OS Mastermap.

In particular, OS 1:1250 to 10000

Reduce scale

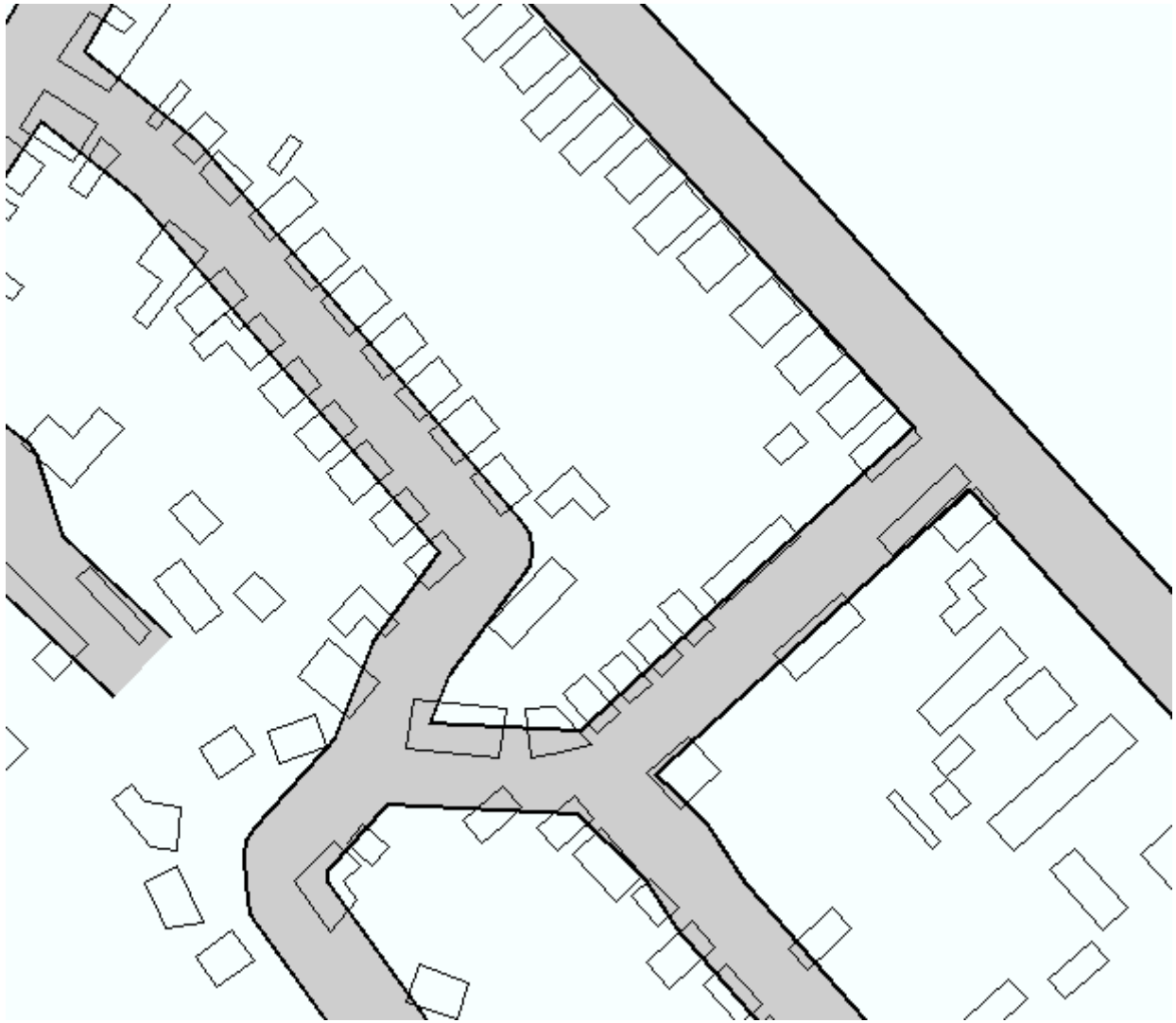


Generalize



Specific Problem

- Graphic conflict caused by road symbolization
- Road symbol thickened
- Roads interfere with buildings $\longrightarrow$ graphic conflict
- Resolve by displacing buildings



Graphic Conflict Resolution by Building Feature Displacement

- Road features symbolized
- Assume position of road features remains fixed
- Assume building features can be displaced from origin up to some maximum distance

Other Research

- Agent-based solutions
- Least squares
- Finite elements
- Snakes
- Elastic beam
- Direct solutions

Metaheuristic Search

- Simulated Annealing
- Tabu Search
- **Genetic Algorithms**

Genetic Algorithms

- Based on the theory of Darwinian evolution
- Solutions (rather than organisms) evolve
- Initial population of possible “solutions”
- Reproduction $\rightarrow$ Selection/Crossover
- Mutation
- Natural selection

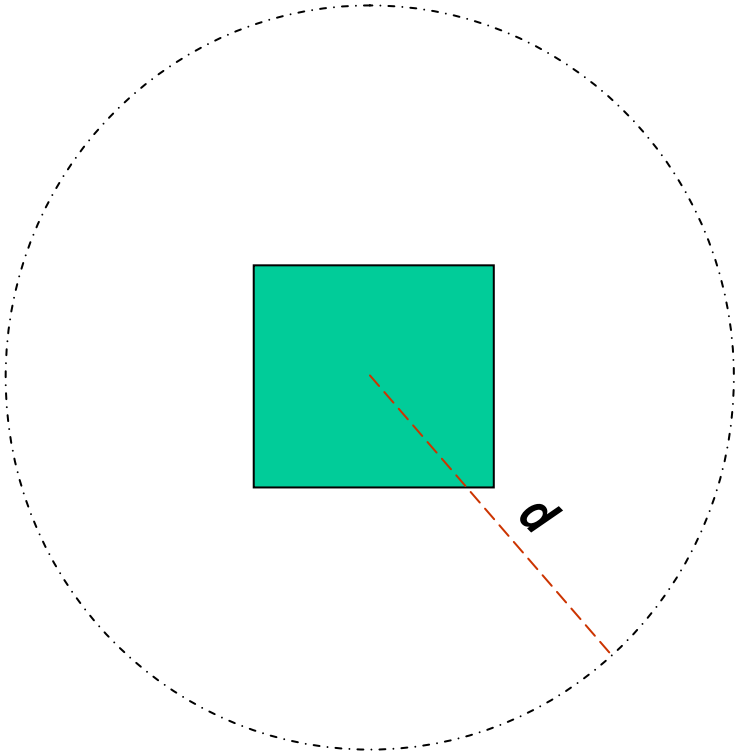
Standard Genetic Algorithm

- initialize population (p_m randomly generated realizations)
- evaluate each population member
- while stopping conditions not met
  - select individuals for breeding
  - breed offspring
  - mutate offspring
  - evaluate offspring
  - update population
  - check stopping conditions
- endwhile

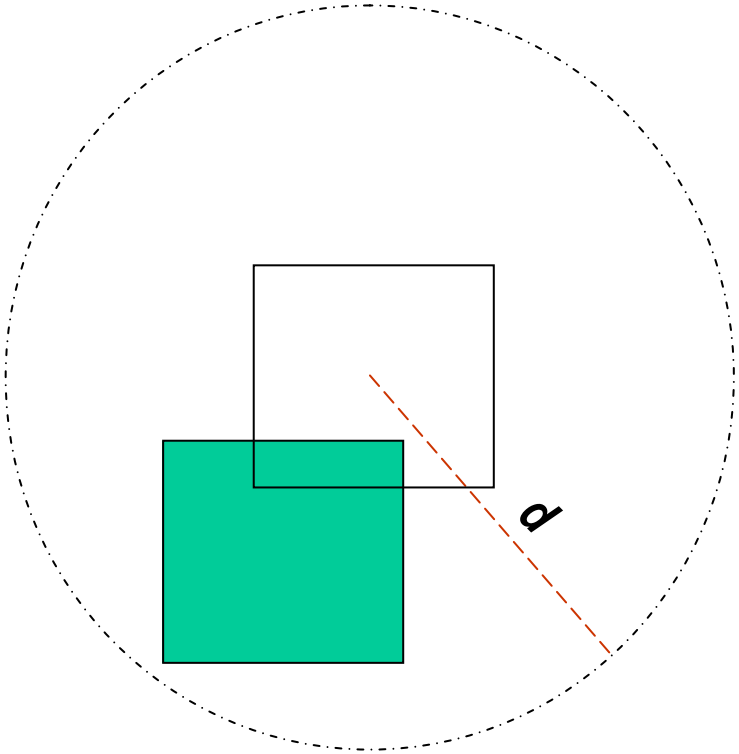
Chromosome

- A particular “solution” is represented by a chromosome.
- Chromosome can itself be represented in a variety of ways.
- We make use of a Binary String.

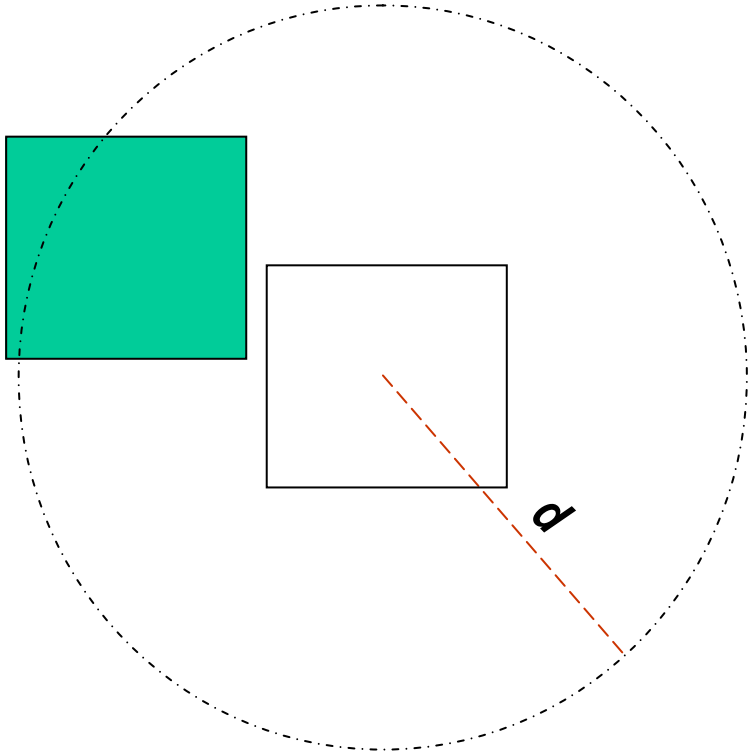
In any attempt to resolve conflict, a building is allowed to move up to a maximum distance *d* from its start location.



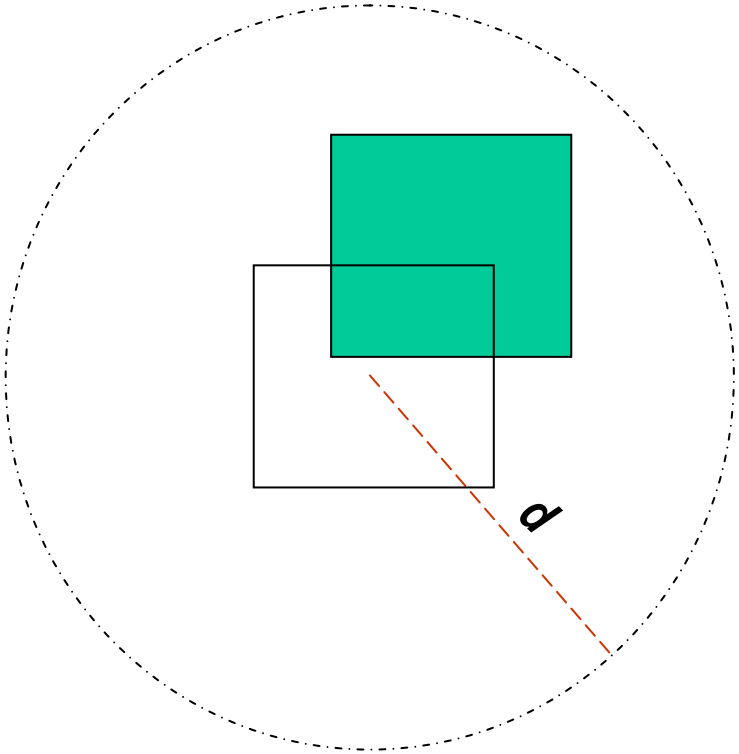
In any attempt to resolve conflict, a building is allowed to move up to a maximum distance *d* from its start location.



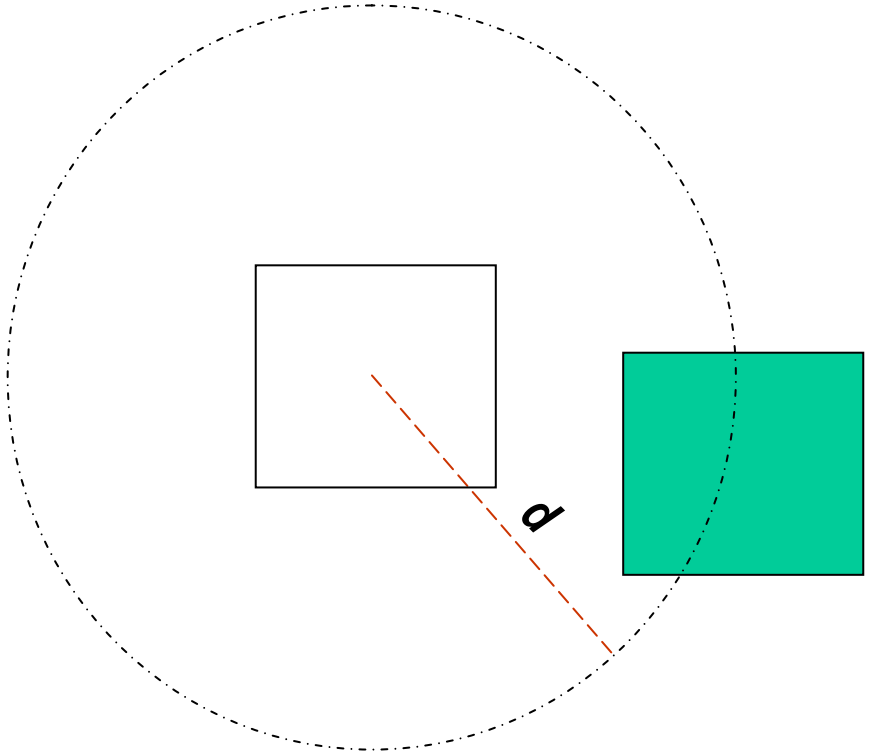
In any attempt to resolve conflict, a building is allowed to move up to a maximum distance *d* from its start location.



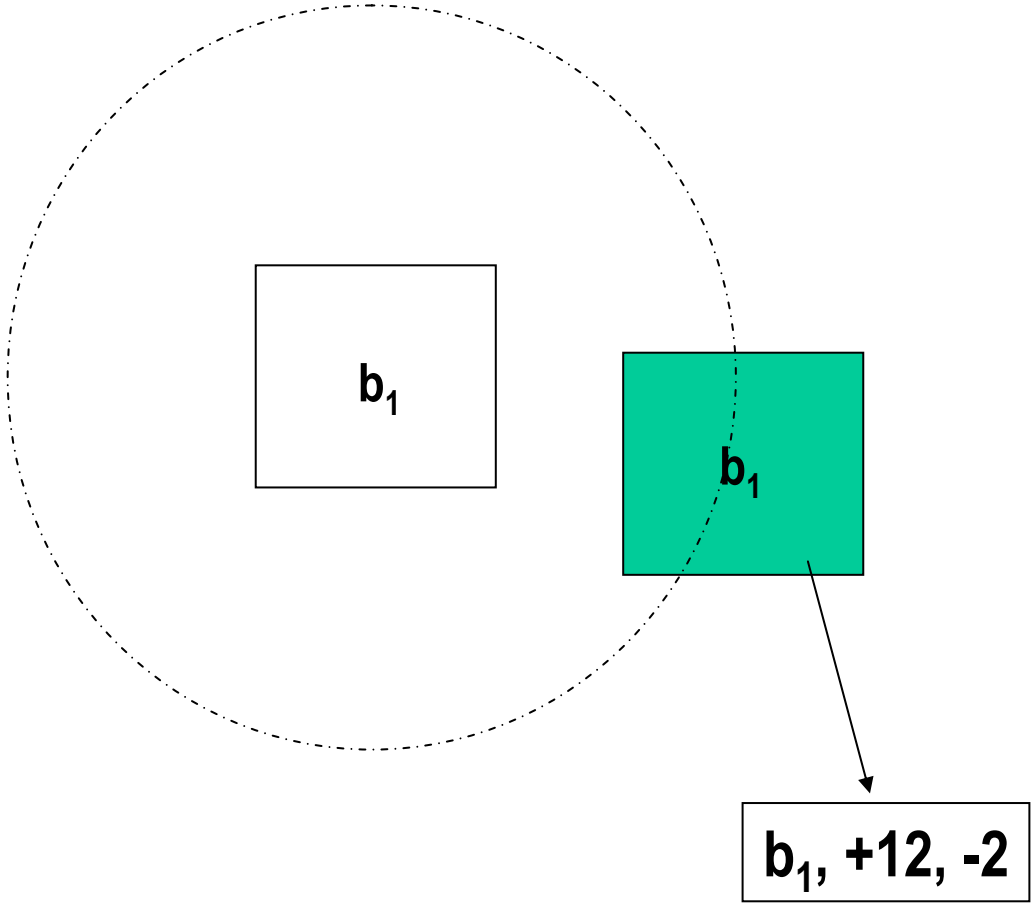
In any attempt to resolve conflict, a building is allowed to move up to a maximum distance *d* from its start location.



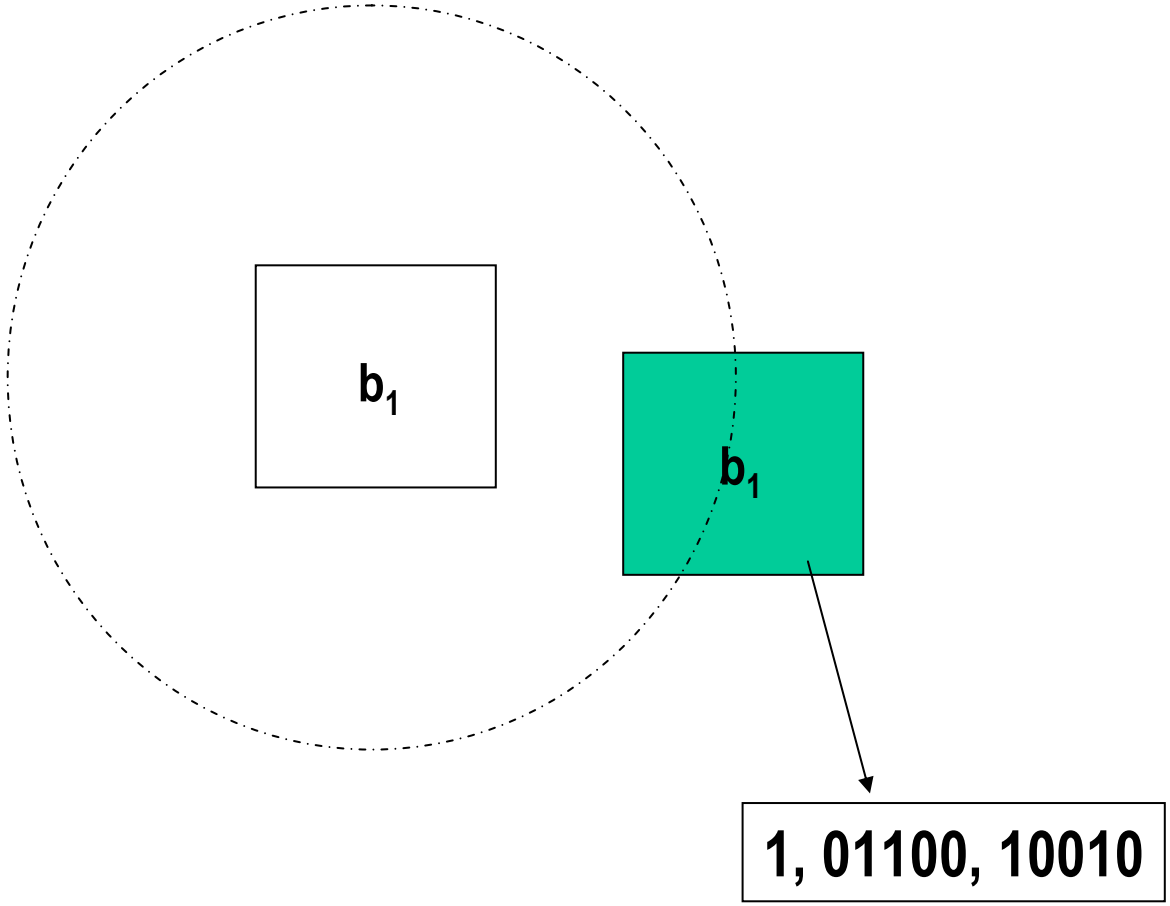
In any attempt to resolve conflict, a building is allowed to move up to a maximum distance *d* from its start location.



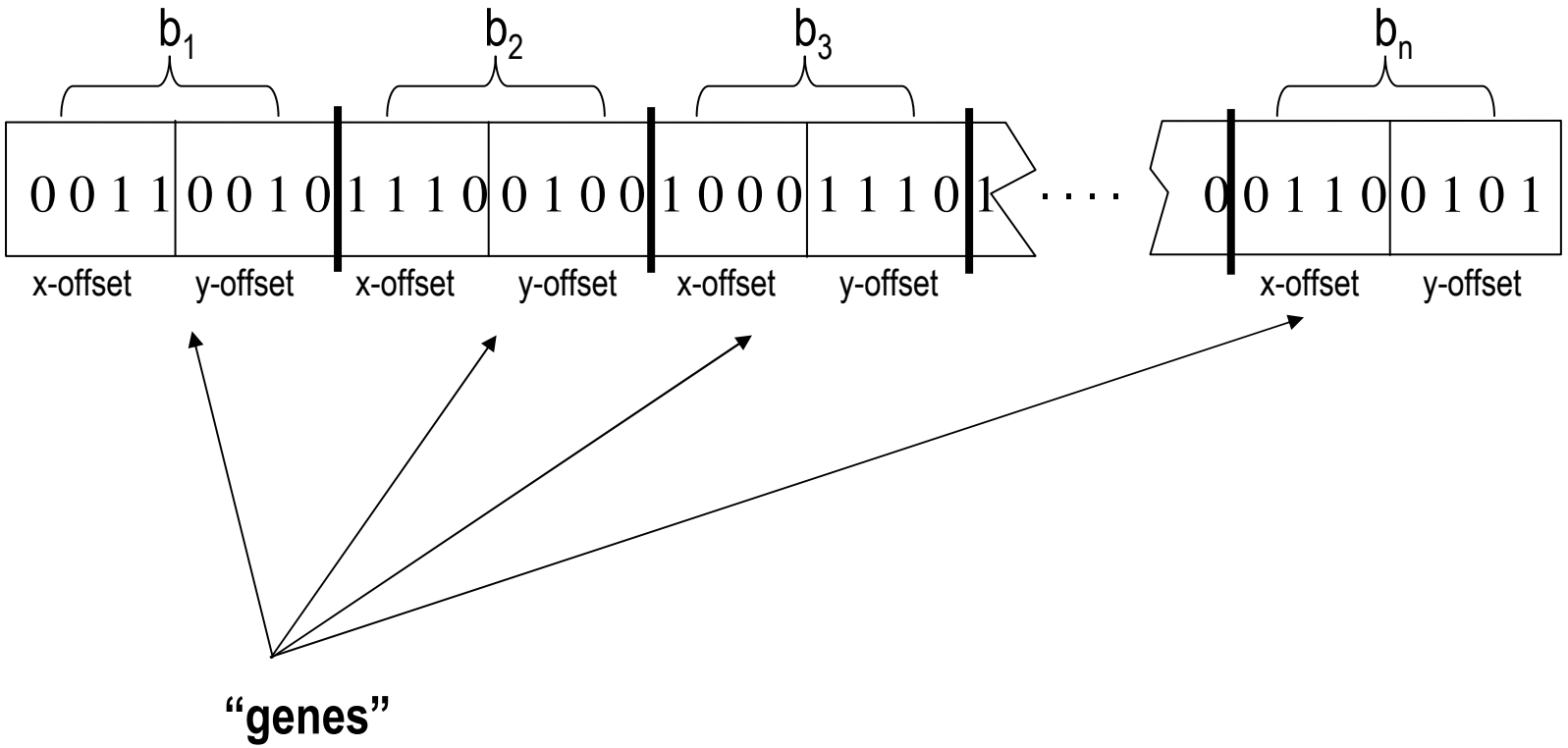
For a particular “solution”, the position of an object can be represented by a reference to original building location plus an x-offset and a y-offset



Offsets can be represented as binary numbers.



A “solution” represented by binary offsets for each object, giving us a “Chromosome”

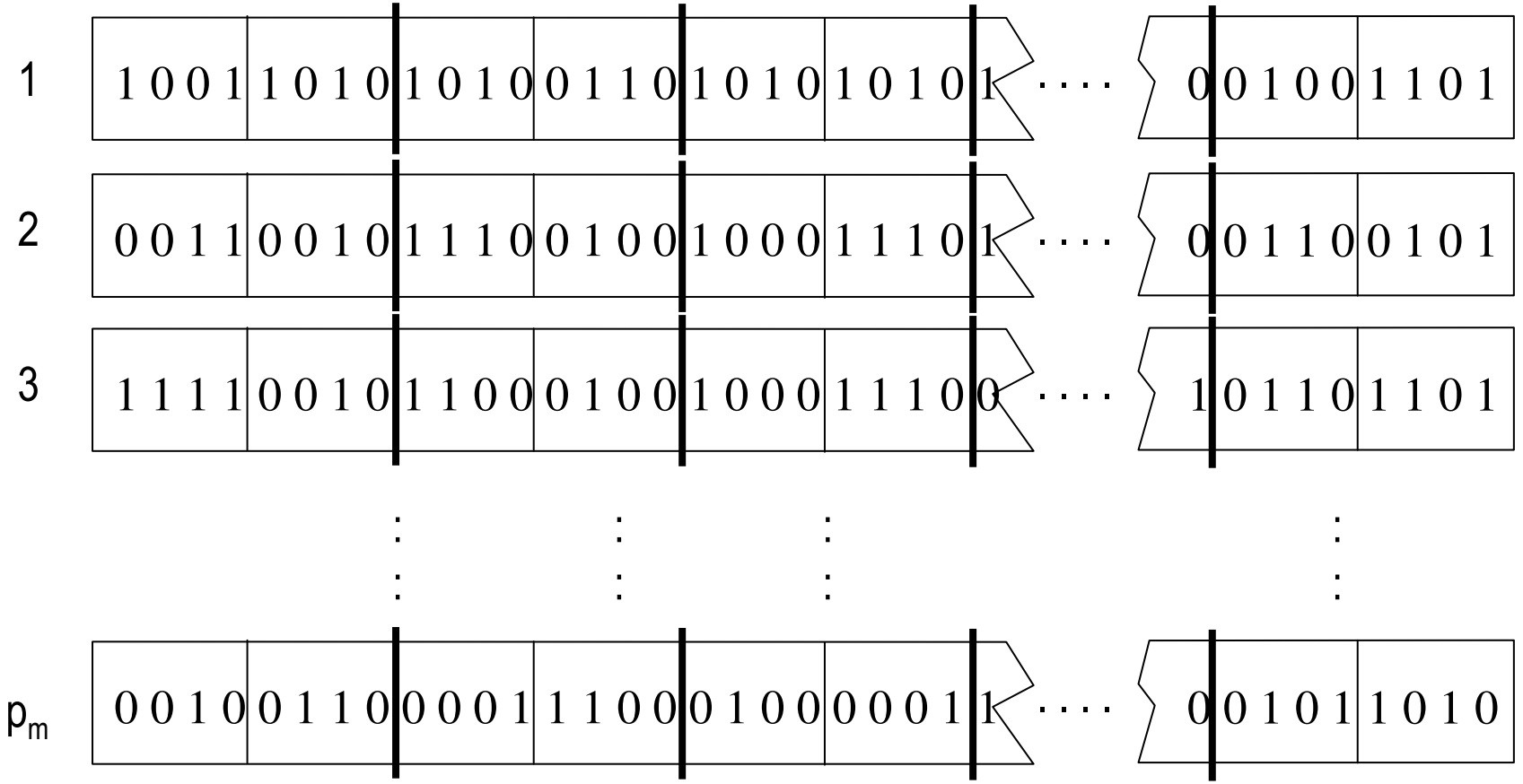


Genetic Algorithm (steady state)

- initialize population (p_m randomly generated realizations)
- evaluate each population member
- while stopping conditions not met
  - select individuals for breeding
  - breed offspring
  - mutate offspring
  - evaluate offspring
  - update population
  - check stopping conditions
- endwhile

Initialise

p_m (randomly) generated realizations



Genetic Algorithm (steady state)

- initialize population (p_m randomly generated realizations)
- evaluate each population member
- while stopping conditions not met
  - select individuals for breeding
  - breed offspring
  - mutate offspring
  - evaluate offspring
  - update population
  - check stopping conditions
- endwhile

Each solution has a cost (which has to be calculated)

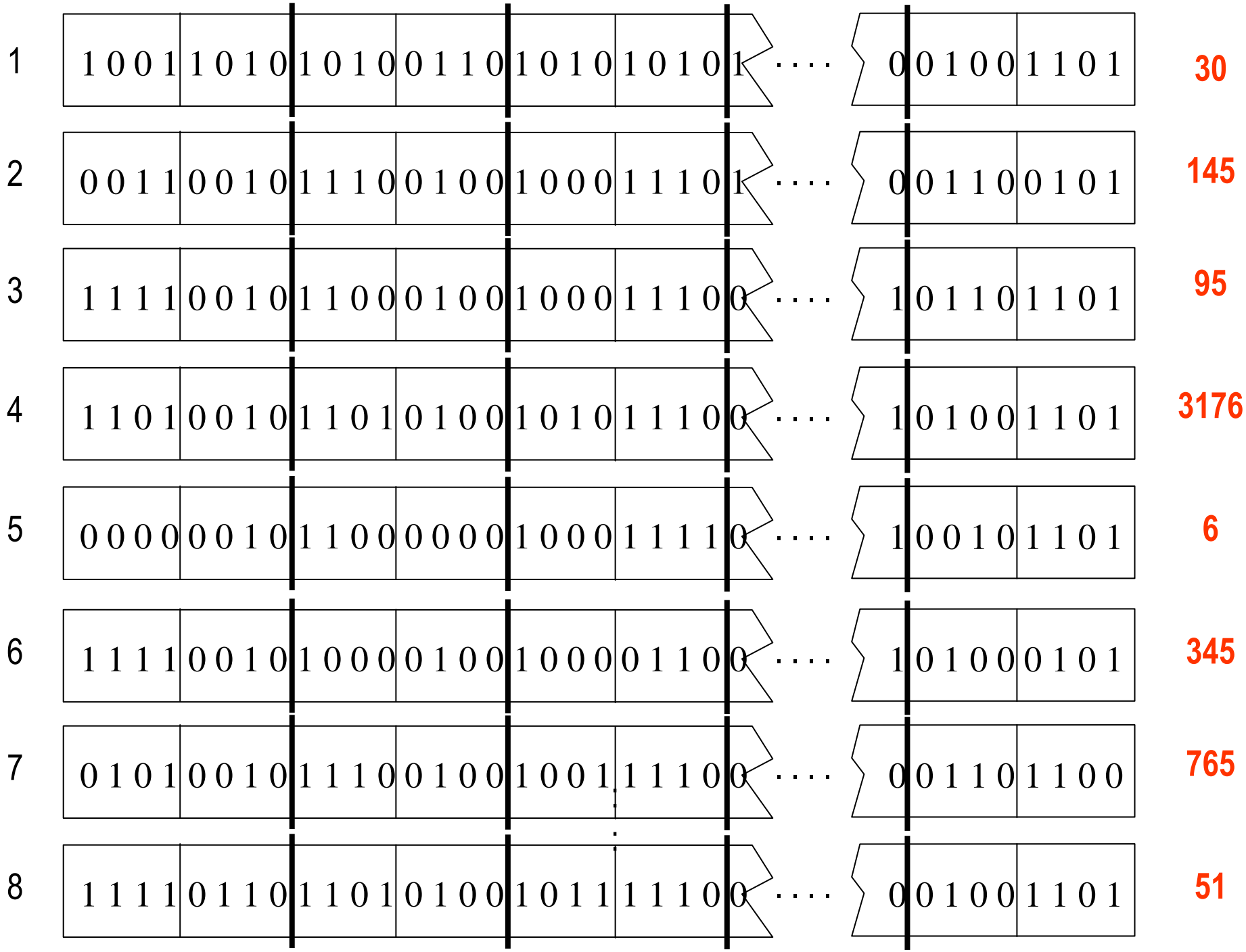
C1 x Number of building/road violations

+

C2 x Number of building/building violations

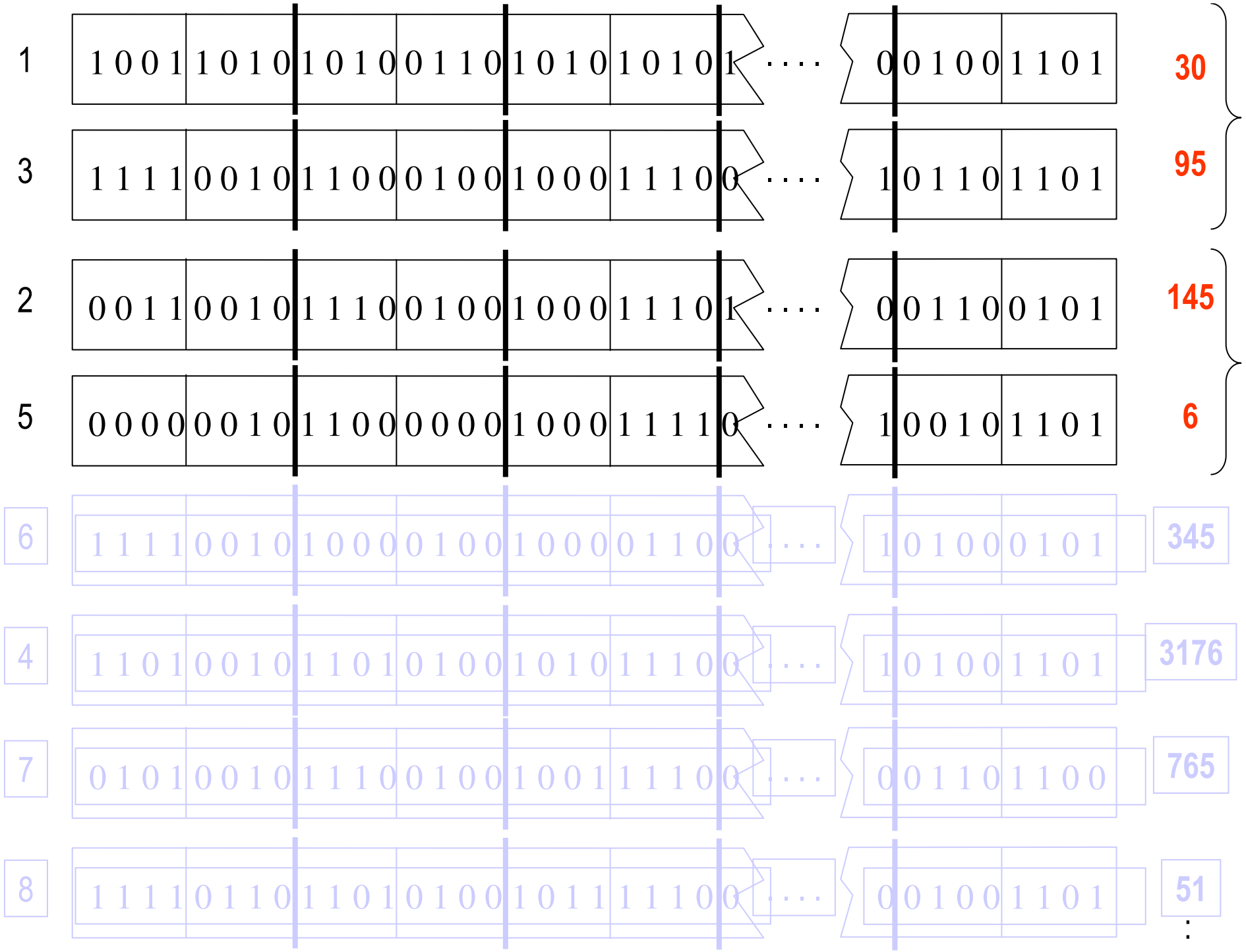
+

C3 x Total displacement


1
1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 1 0 1 . . . . 0 0 1 0 0 1 1 0 1
30
2
0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 0 1 . . . . 0 0 1 1 0 0 1 0 1
145
3
1 1 1 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 1 0 0 . . . . 1 0 1 1 0 1 1 0 1
95
4
1 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 1 1 0 0 . . . . 1 0 1 0 0 1 1 0 1
3176
5
0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 . . . . 1 0 0 1 0 1 1 0 1
6
6
1 1 1 1 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0 0 . . . . 1 0 1 0 0 0 1 0 1
345
7
0 1 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 1 1 1 1 0 0 . . . . 0 0 1 1 0 1 1 0 0
765
8
1 1 1 1 0 1 1 0 1 1 0 1 0 1 0 0 1 0 1 1 1 1 1 0 0 . . . . 0 0 1 0 0 1 1 0 1
51

Genetic Algorithm (steady state)

- initialize population (p_m randomly generated realizations)
- evaluate each population member
- while stopping conditions not met
  - select individuals for breeding
  - breed offspring
  - mutate offspring
  - evaluate offspring
  - update population
  - check stopping conditions
- endwhile

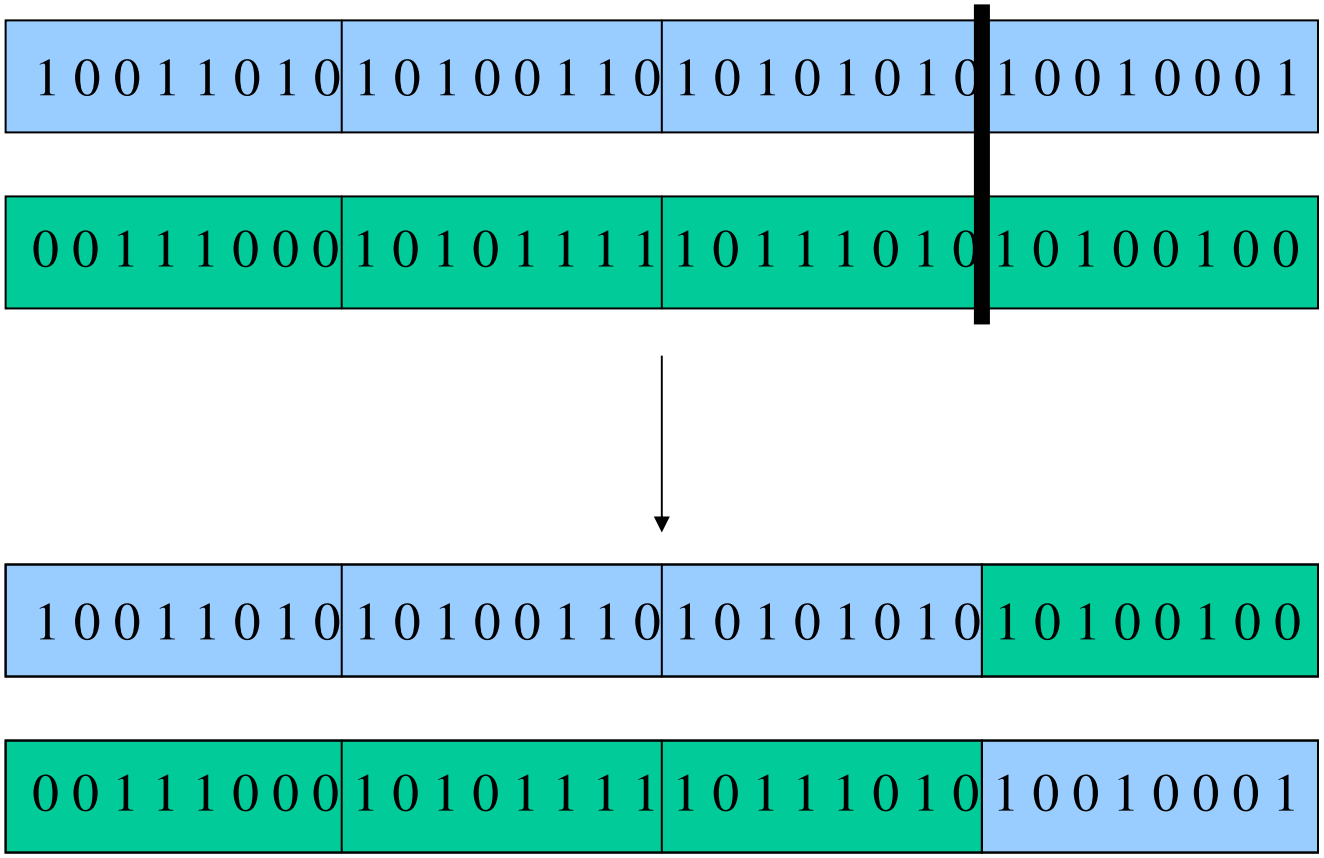

1
1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 1 0 1 · · · · 0 0 1 0 0 1 1 0 1
30
3
1 1 1 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 1 0 0 · · · · 1 0 1 1 0 1 1 0 1
95
2
0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 0 1 · · · · 0 0 1 1 0 0 1 0 1
145
5
0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 · · · · 1 0 0 1 0 1 1 0 1
6
6
1 1 1 1 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0 0 · · · 1 0 1 0 0 0 1 0 1
345
4
1 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 1 1 0 0 · · · 1 0 1 0 0 1 1 0 1
3176
7
0 1 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 1 1 1 1 0 0 · · · 0 0 1 1 0 1 1 0 0
765
8
1 1 1 1 0 1 1 0 1 1 0 1 0 1 0 0 1 0 1 1 1 1 1 0 0 · · · 0 0 1 0 0 1 1 0 1
51

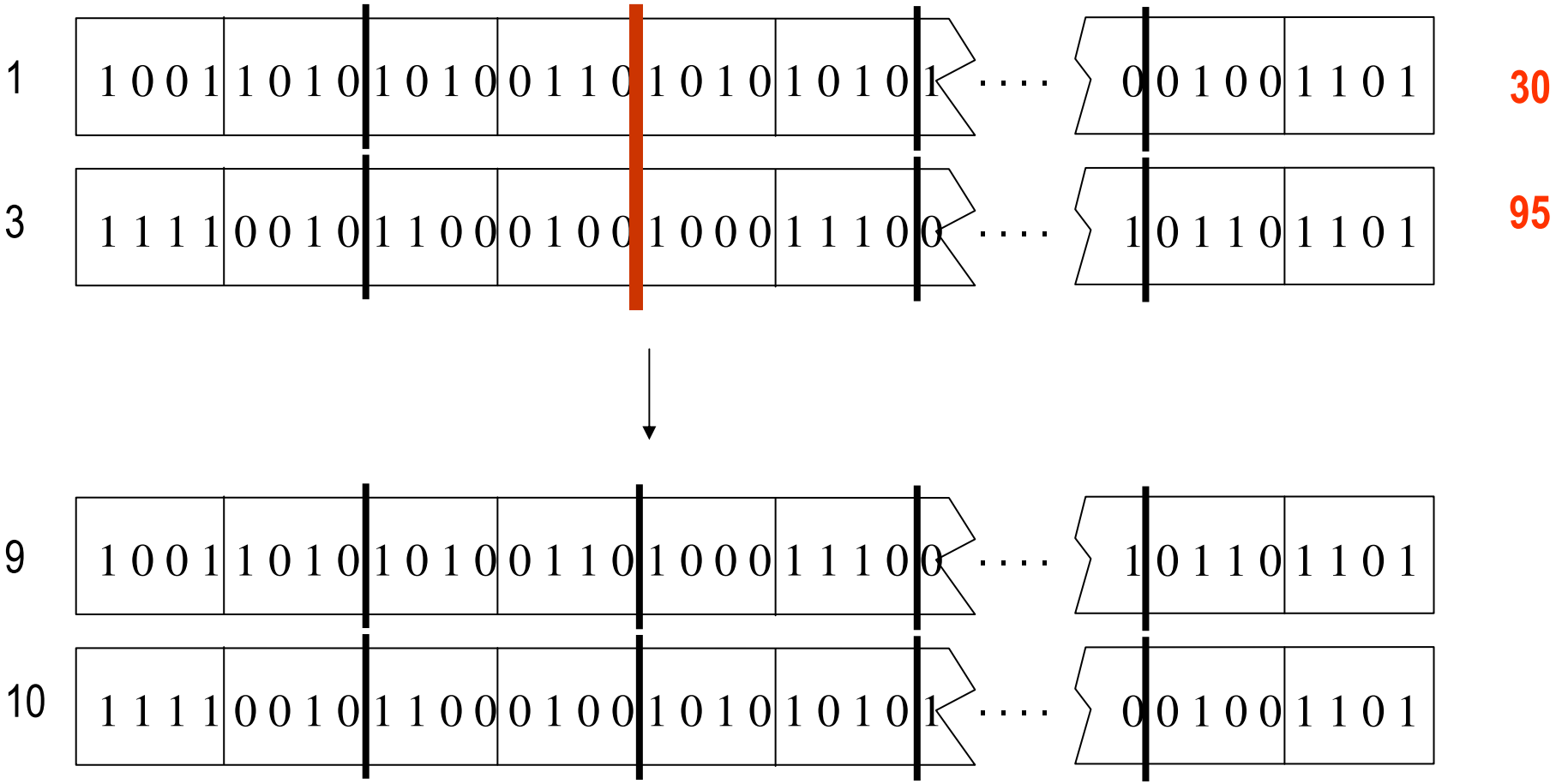
Genetic Algorithm (steady state)

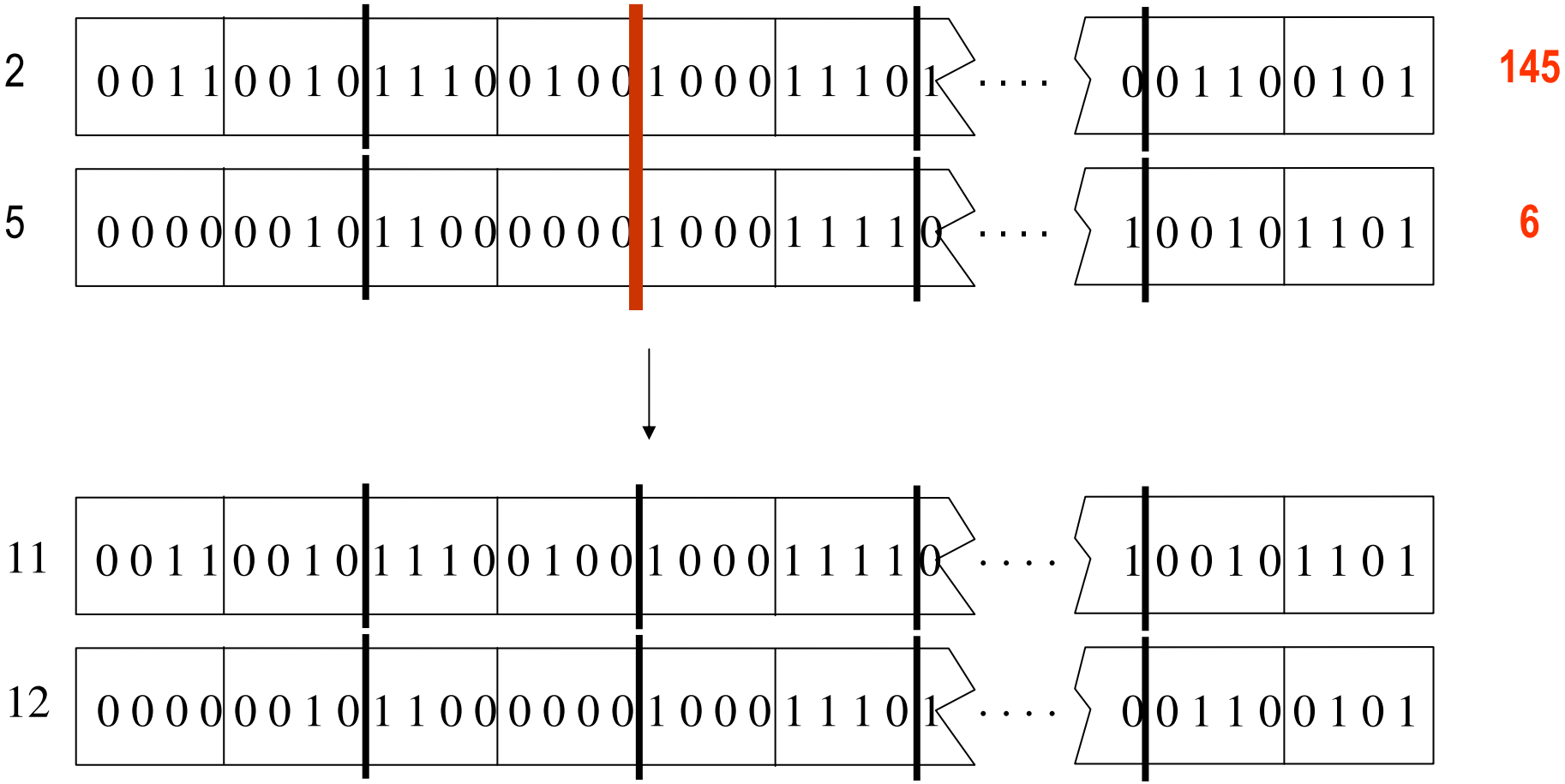
- initialize population (p_m randomly generated realizations)
- evaluate each population member
- while stopping conditions not met
  - select individuals for breeding
  - breed offspring
  - mutate offspring
  - evaluate offspring
  - update population
  - check stopping conditions
- endwhile

Crossover

single (random) point




1
1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 1 0 1 · · · · 0 0 1 0 0 1 1 0 1
30
3
1 1 1 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 1 0 0 · · · · 1 0 1 1 0 1 1 0 1
95
9
1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 0 0 1 1 1 0 0 · · · · 1 0 1 1 0 1 1 0 1
10
1 1 1 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0 1 0 1 0 1 0 1 · · · · 0 0 1 0 0 1 1 0 1

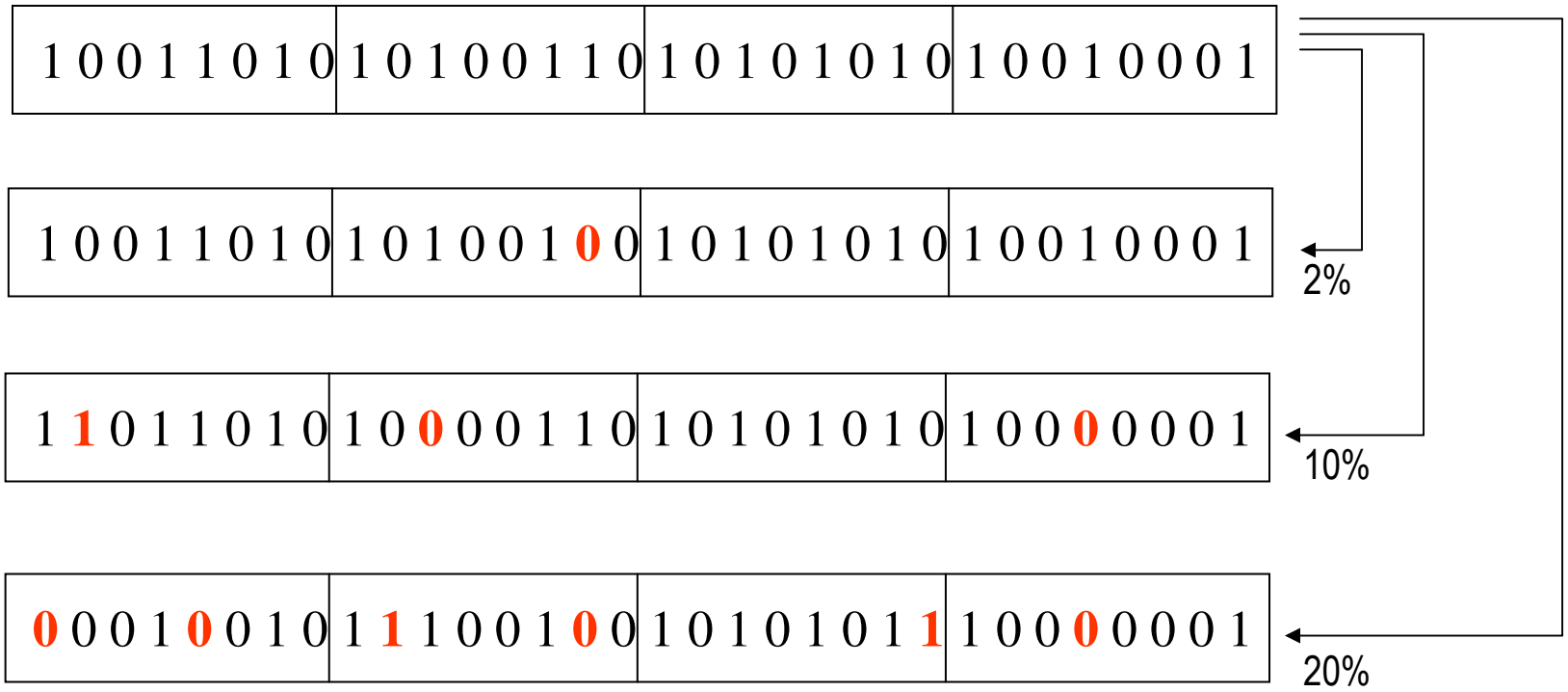

2
0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 0 1 . . . . 0 0 1 1 0 0 1 0 1
145
5
0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 . . . . 1 0 0 1 0 1 1 0 1
6
11
0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 1 0 . . . . 1 0 0 1 0 1 1 0 1
12
0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 1 1 0 1 . . . . 0 0 1 1 0 0 1 0 1

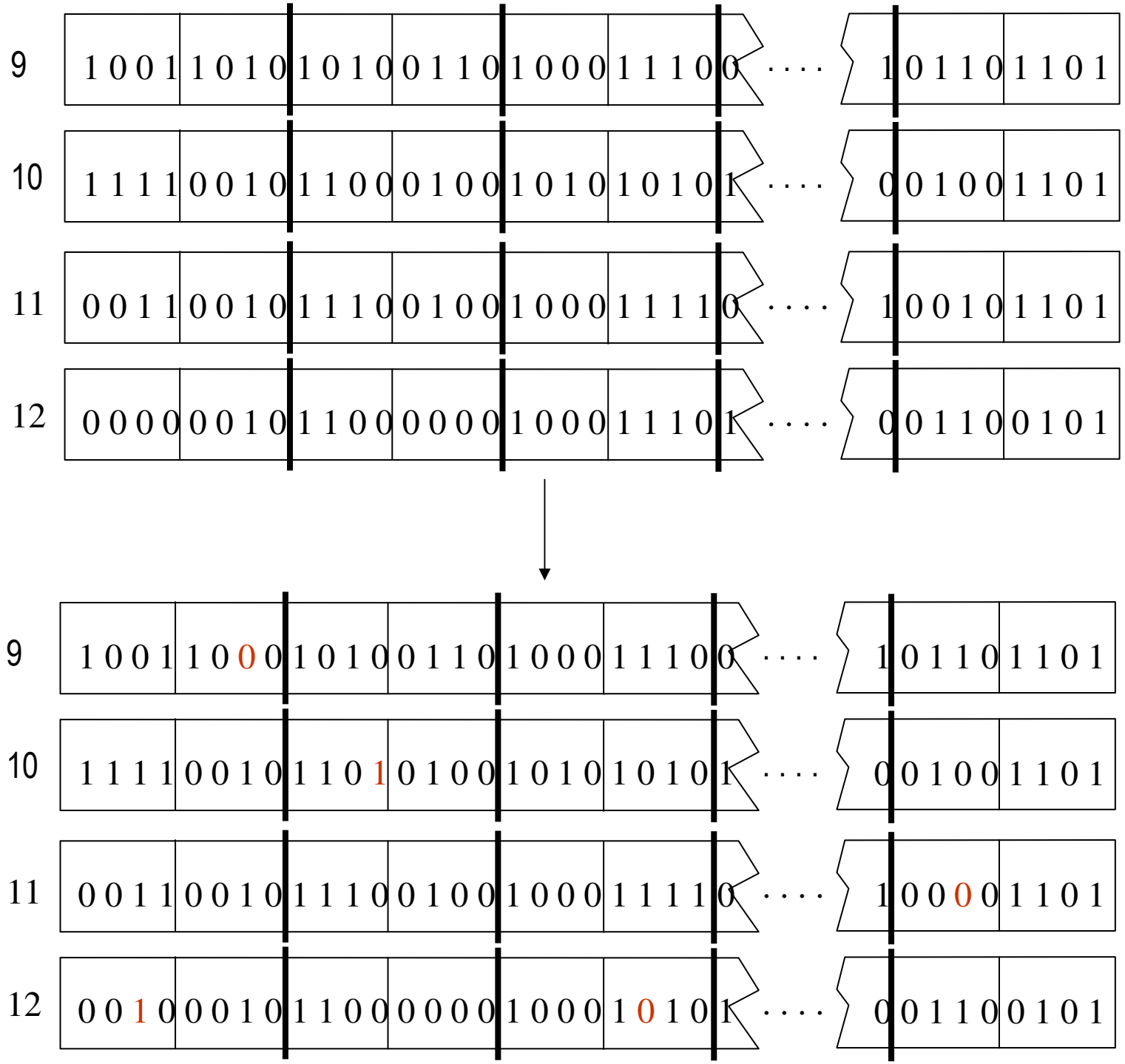
Genetic Algorithm (steady state)

- initialize population (p_m randomly generated realizations)
- evaluate each population member
- while stopping conditions not met
  - select individuals for breeding
  - breed offspring
  - mutate offspring
  - evaluate offspring
  - update population
  - check stopping conditions
- endwhile

Mutation

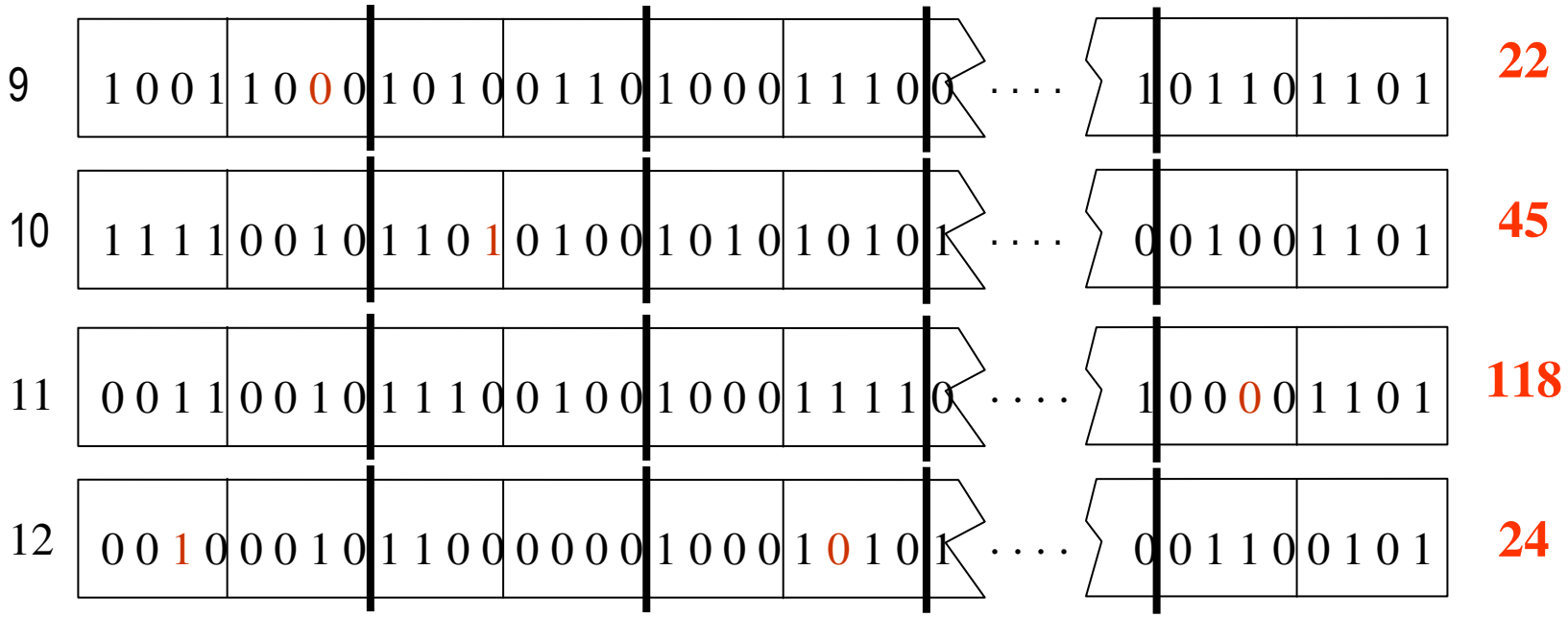
(random), based on mutation probability




9
1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 0 0 1 1 1 0 0 · · · · 1 0 1 1 0 1 1 0 1
10
1 1 1 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0 1 0 1 0 1 0 1 · · · · 0 0 1 0 0 1 1 0 1
11
0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 1 0 · · · · 1 0 0 1 0 1 1 0 1
12
0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 1 1 0 1 · · · · 0 0 1 1 0 0 1 0 1
9
1 0 0 1 1 0 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 1 1 0 0 · · · · 1 0 1 1 0 1 1 0 1
10
1 1 1 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 0 1 0 1 · · · · 0 0 1 0 0 1 1 0 1
11
0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 1 0 · · · · 1 0 0 0 0 1 1 0 1
12
0 0 1 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 0 1 0 1 · · · · 0 0 1 1 0 0 1 0 1

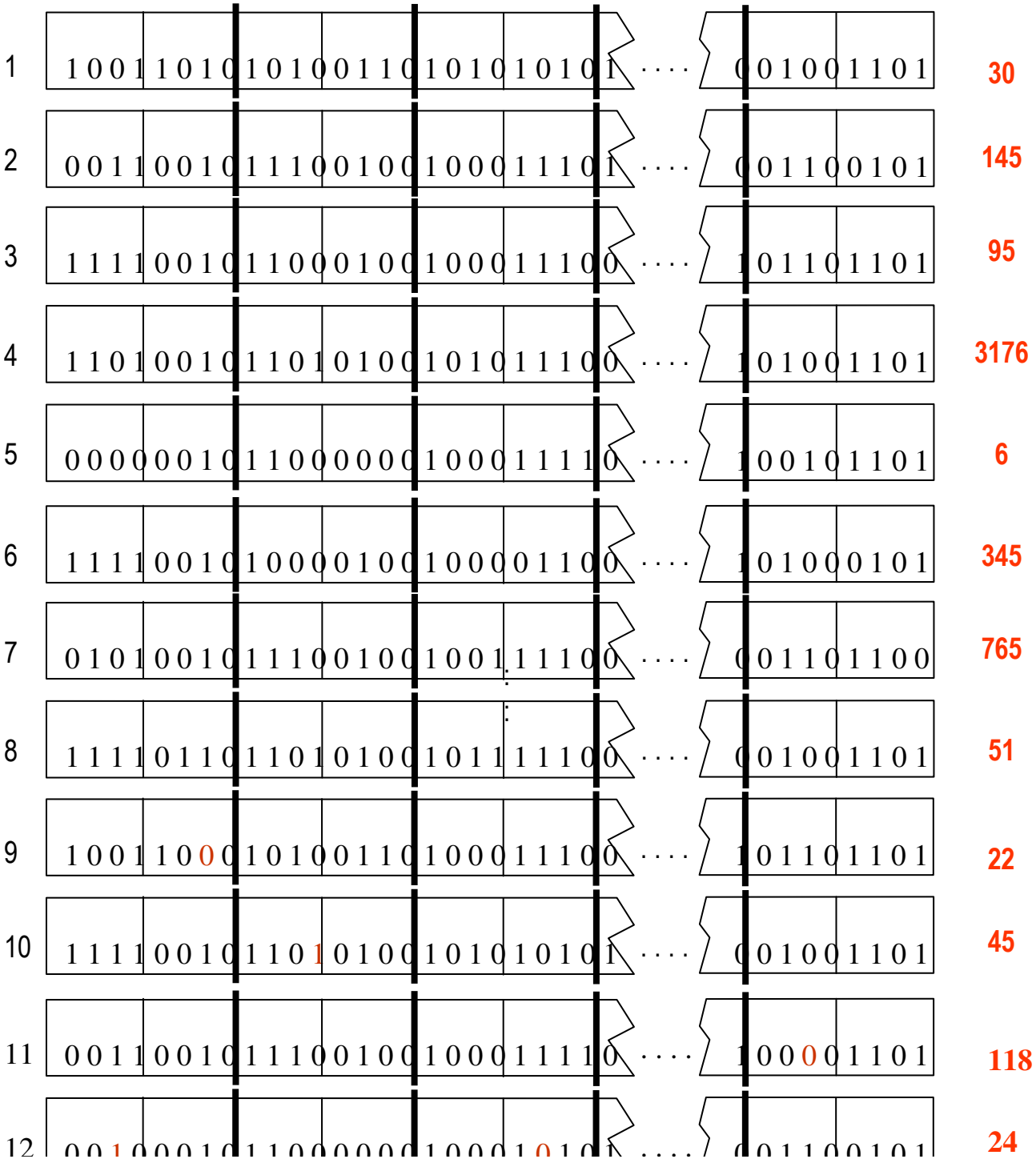
Genetic Algorithm (steady state)

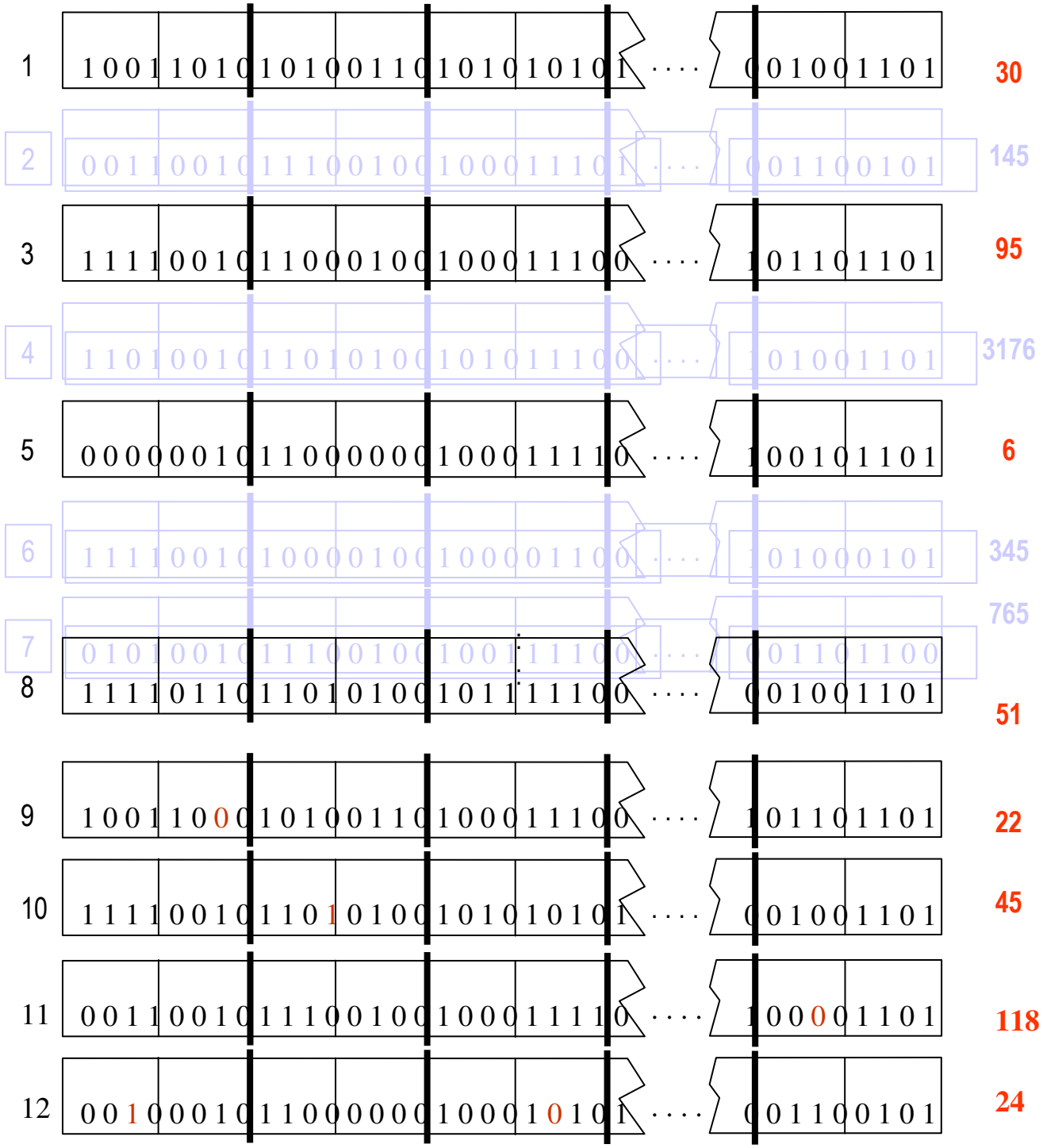
- initialize population (p_m randomly generated realizations)
- evaluate each population member
- while stopping conditions not met
  - select individuals for breeding
  - breed offspring
  - mutate offspring
  - evaluate offspring
  - update population
  - check stopping conditions
- endwhile


9
1 0 0 1 1 0 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 1 1 0 0 · · · · 1 0 1 1 0 1 1 0 1
22
10
1 1 1 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 0 1 0 1 · · · · 0 0 1 0 0 1 1 0 1
45
11
0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 1 0 · · · · 1 0 0 0 0 1 1 0 1
118
12
0 0 1 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 0 1 0 1 · · · · 0 0 1 1 0 0 1 0 1
24

Genetic Algorithm (steady state)

- initialize population (p_m randomly generated realizations)
- evaluate each population member
- while stopping conditions not met
  - select individuals for breeding
  - breed offspring
  - mutate offspring
  - evaluate offspring
  - update population
  - check stopping conditions
- endwhile


1 1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 1 . . . . 0 0 1 0 0 1 1 0 1
30
2 0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 0 1 . . . . 0 0 1 1 0 0 1 0 1
145
3 1 1 1 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 1 0 0 . . . . 1 0 1 1 0 1 1 0 1
95
4 1 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 1 1 0 0 . . . . 1 0 1 0 0 1 1 0 1
3176
5 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 . . . . 1 0 0 1 0 1 1 0 1
6
6 1 1 1 1 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0 0 . . . . 1 0 1 0 0 0 1 0 1
345
7 0 1 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 1 1 1 1 0 0 . . . . 0 0 1 1 0 1 1 0 0
765
8 1 1 1 1 0 1 1 0 1 1 0 1 0 1 0 0 1 0 1 1 1 1 1 0 0 . . . . 0 0 1 0 0 1 1 0 1
51
9 1 0 0 1 1 0 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 1 1 0 0 . . . . 1 0 1 1 0 1 1 0 1
22
10 1 1 1 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 0 1 0 1 . . . . 0 0 1 0 0 1 1 0 1
45
11 0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 1 0 . . . . 1 0 0 0 0 1 1 0 1
118
12 0 0 1 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 0 1 0 1 . . . . 0 0 1 1 0 0 1 0 1
24


1 1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 1 · · · · 0 0 1 0 0 1 1 0 1
30
2 0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 0 1 · · · · 0 0 1 1 0 0 1 0 1
145
3 1 1 1 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 1 0 0 · · · · 1 0 1 1 0 1 1 0 1
95
4 1 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 1 1 0 0 · · · · 1 0 1 0 0 1 1 0 1
3176
5 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 1 1 1 0 · · · · 1 0 0 1 0 1 1 0 1
6
6 1 1 1 1 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0 0 · · · · 1 0 1 0 0 0 1 0 1
345
7 0 1 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 1 1 1 0 0 · · · · 0 0 1 1 0 1 1 0 0
765
8 1 1 1 1 0 1 1 0 1 1 0 1 0 1 0 0 1 0 1 1 1 1 0 0 · · · · 0 0 1 0 0 1 1 0 1
51
9 1 0 0 1 1 0 0 1 0 1 0 0 1 1 0 1 0 0 0 1 1 1 0 0 · · · · 1 0 1 1 0 1 1 0 1
22
10 1 1 1 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 0 1 0 · · · · 0 0 1 0 0 1 1 0 1
45
11 0 0 1 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 0 1 1 1 1 0 · · · · 1 0 0 0 0 1 1 0 1
118
12 0 0 1 0 0 0 1 0 1 1 0 0 0 0 0 0 1 0 0 0 1 0 1 0 1 · · · · 0 0 1 1 0 0 1 0 1
24

Genetic Algorithm (steady state)

- initialize population (p_m randomly generated realizations)
- evaluate each population member
- while stopping conditions not met
  - select individuals for breeding
  - breed offspring
  - mutate offspring
  - evaluate offspring
  - update population
  - check stopping conditions
- endwhile

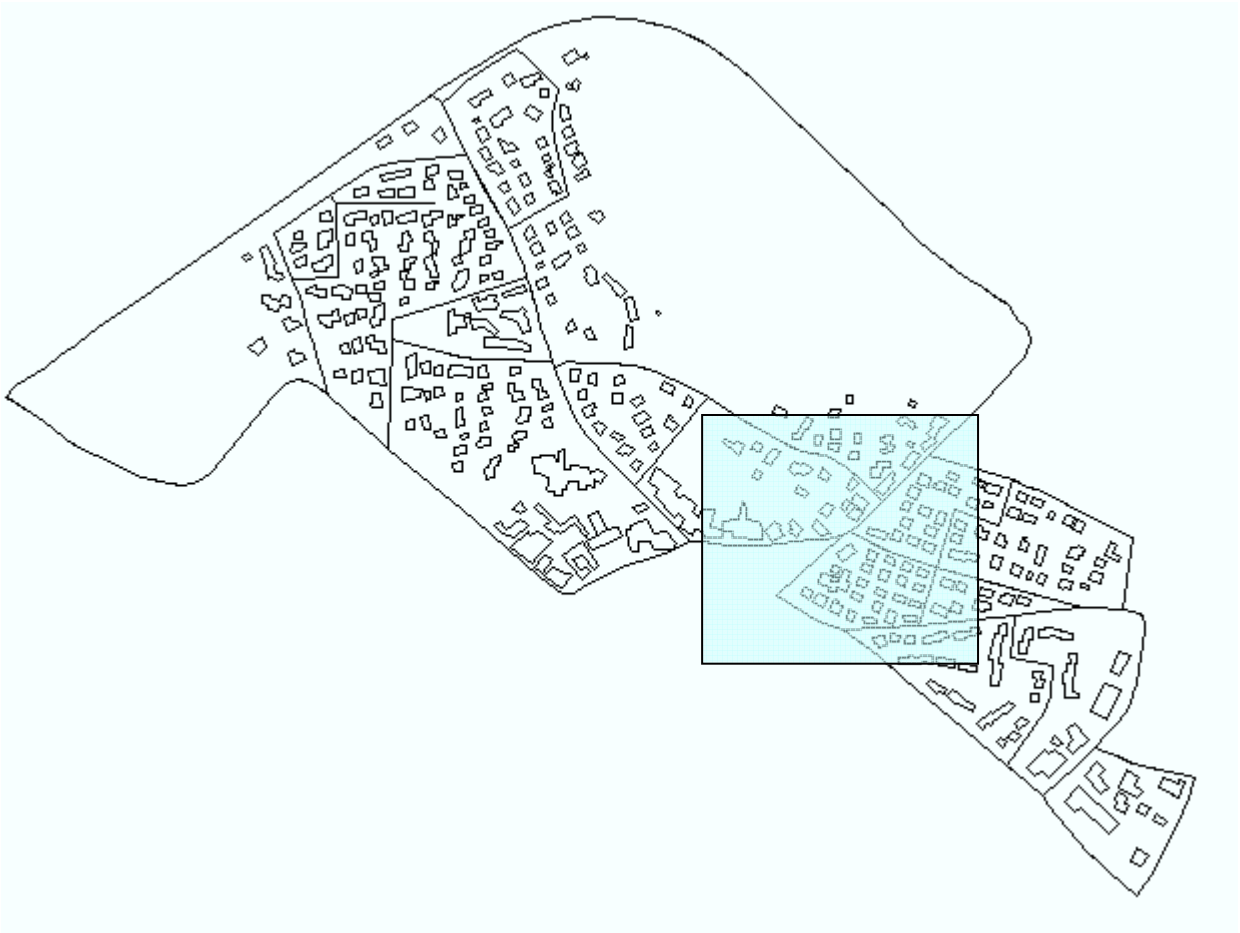
Implementation

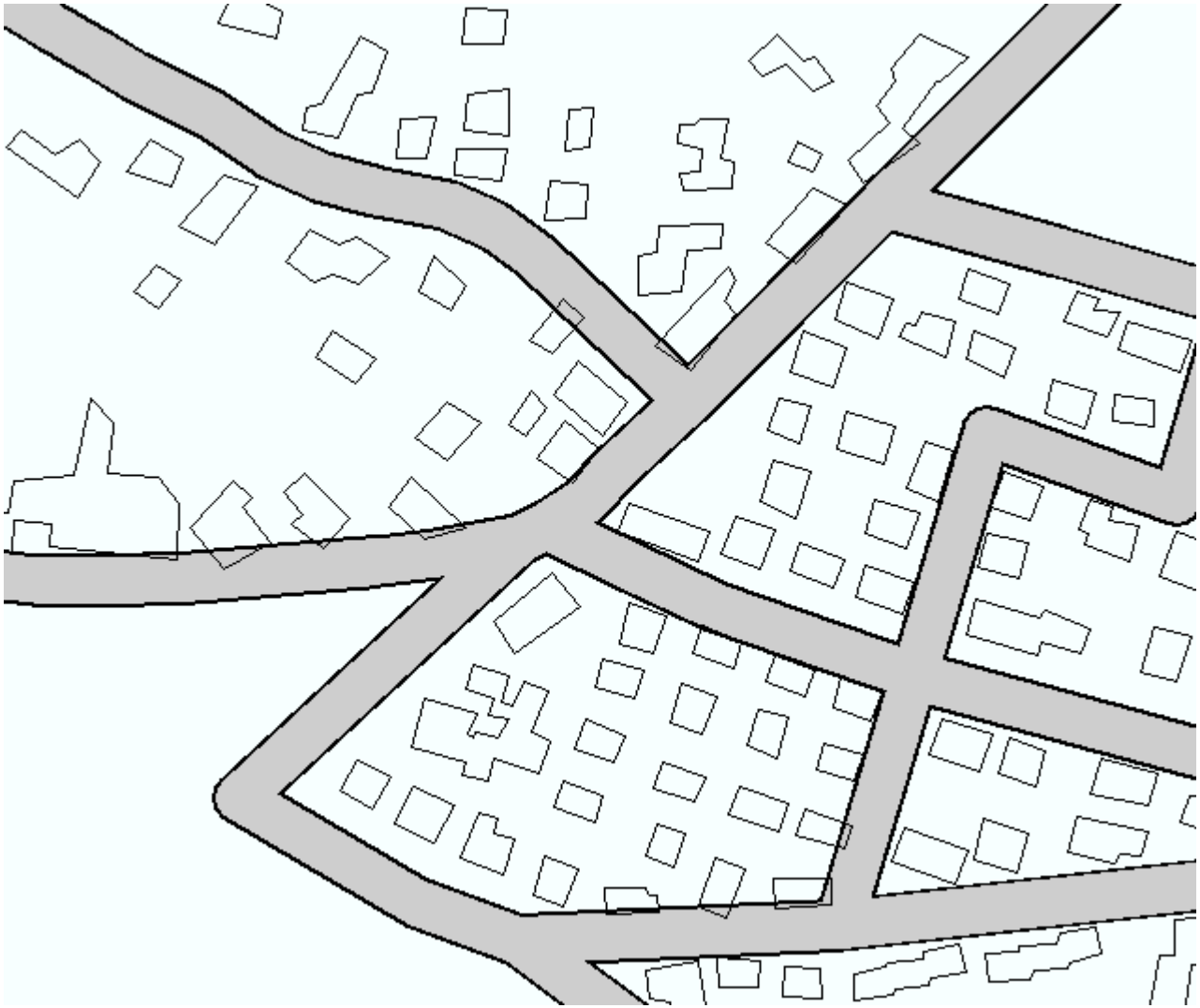
- C++
- GAlib

Experiments

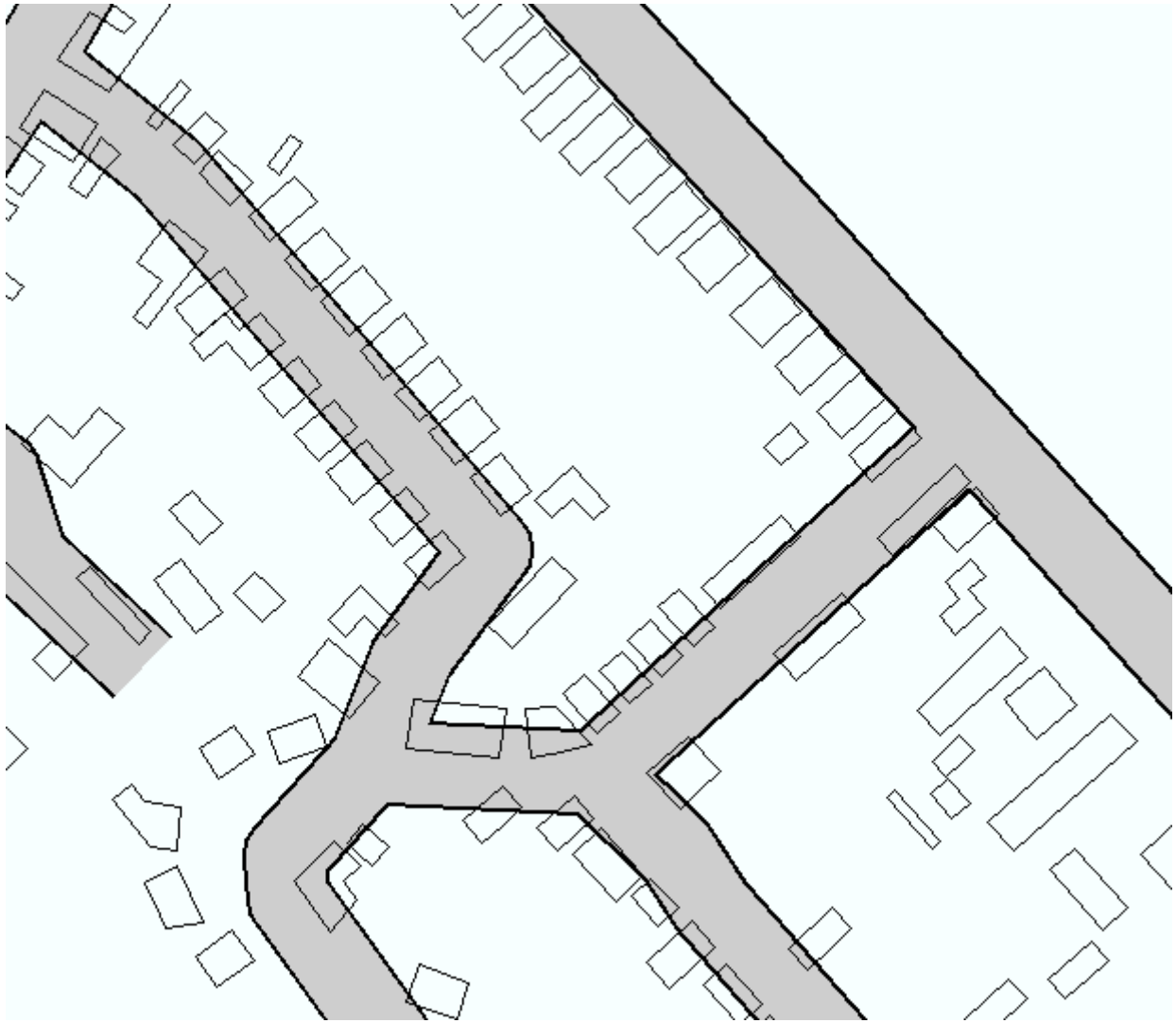
- IGN BDTopo Data
- OS MasterMap

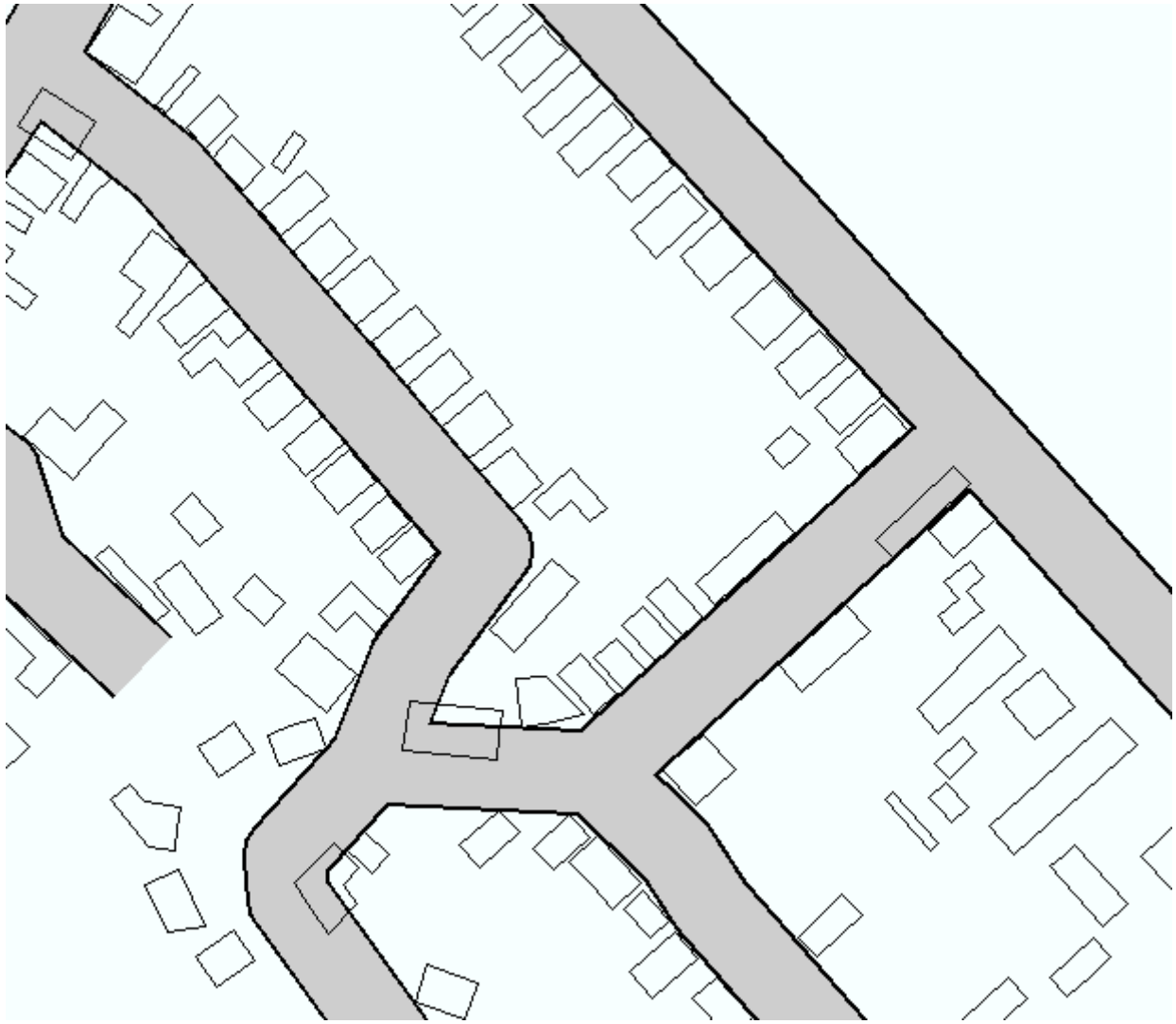
IGN BDTopo Data





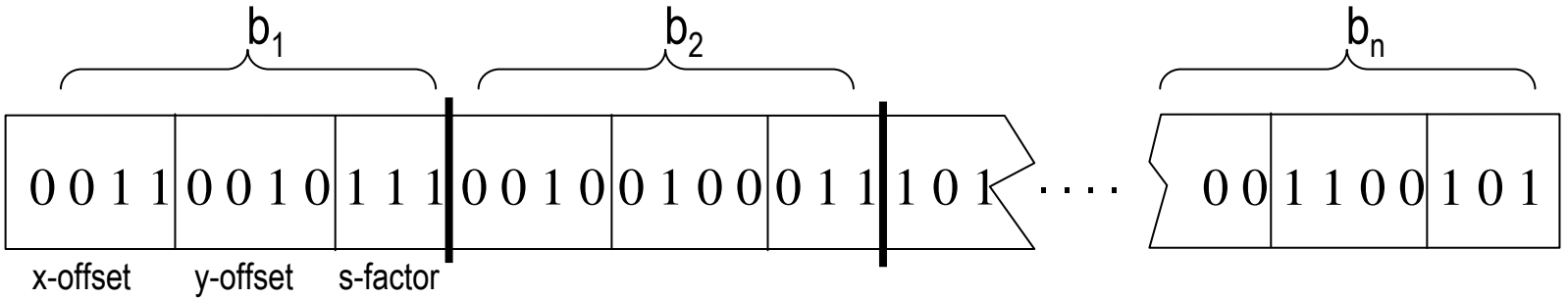
OS Mastermap (assume scale change from 1:1250 to 1:10000)

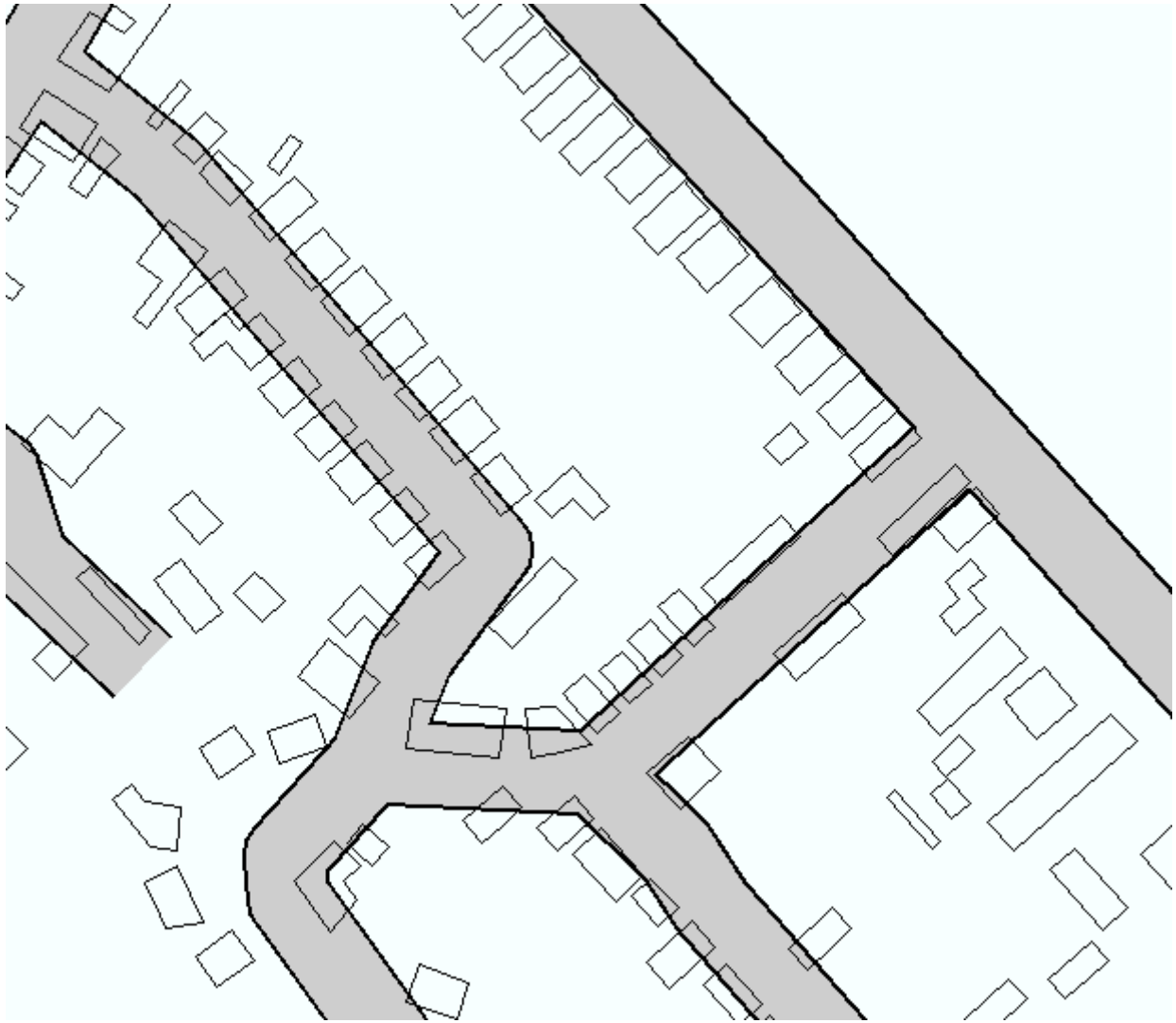


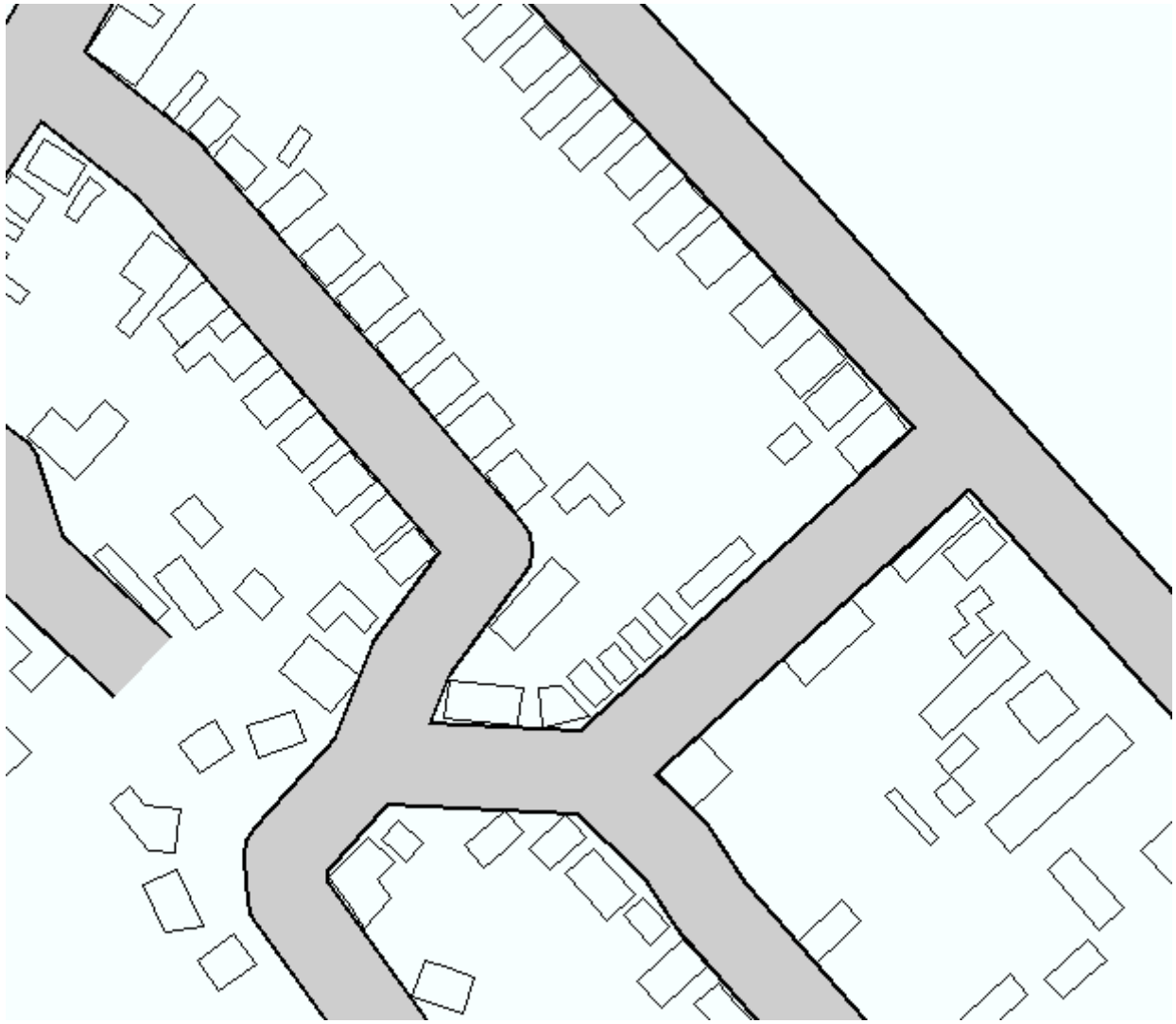


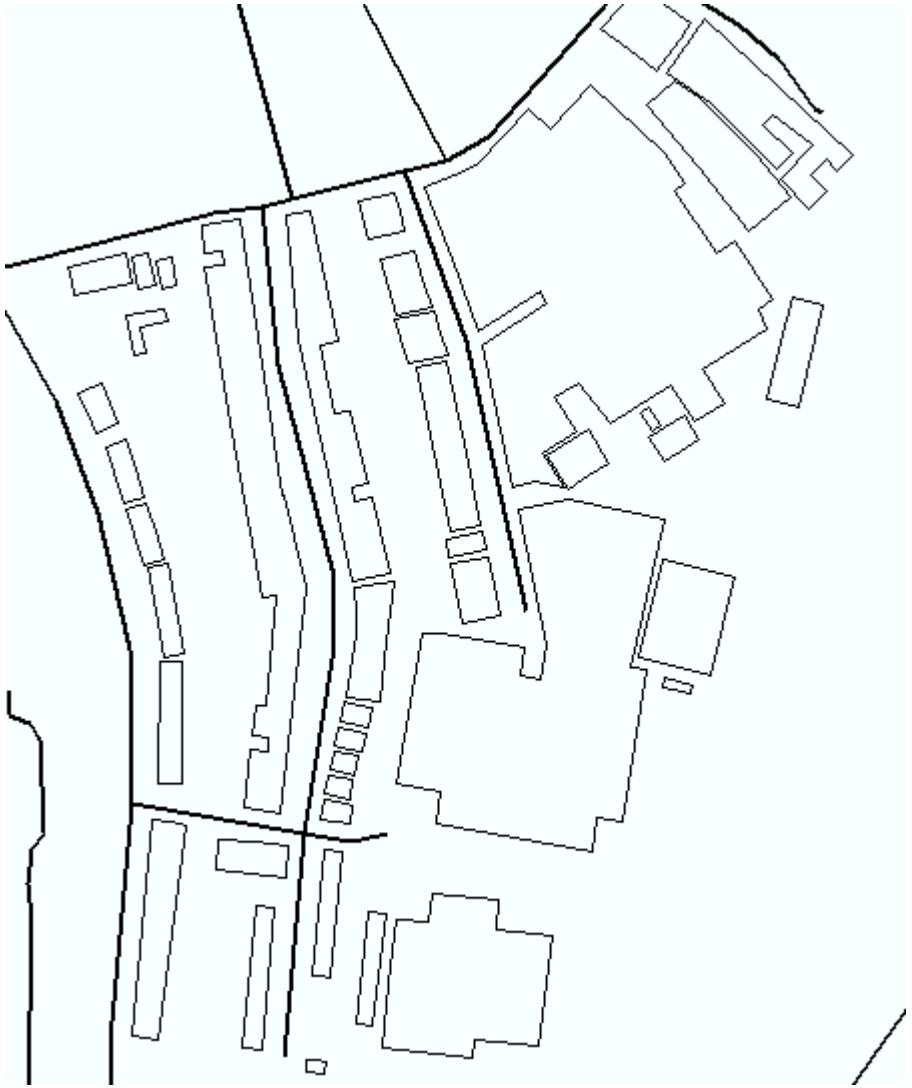
Additional Operator

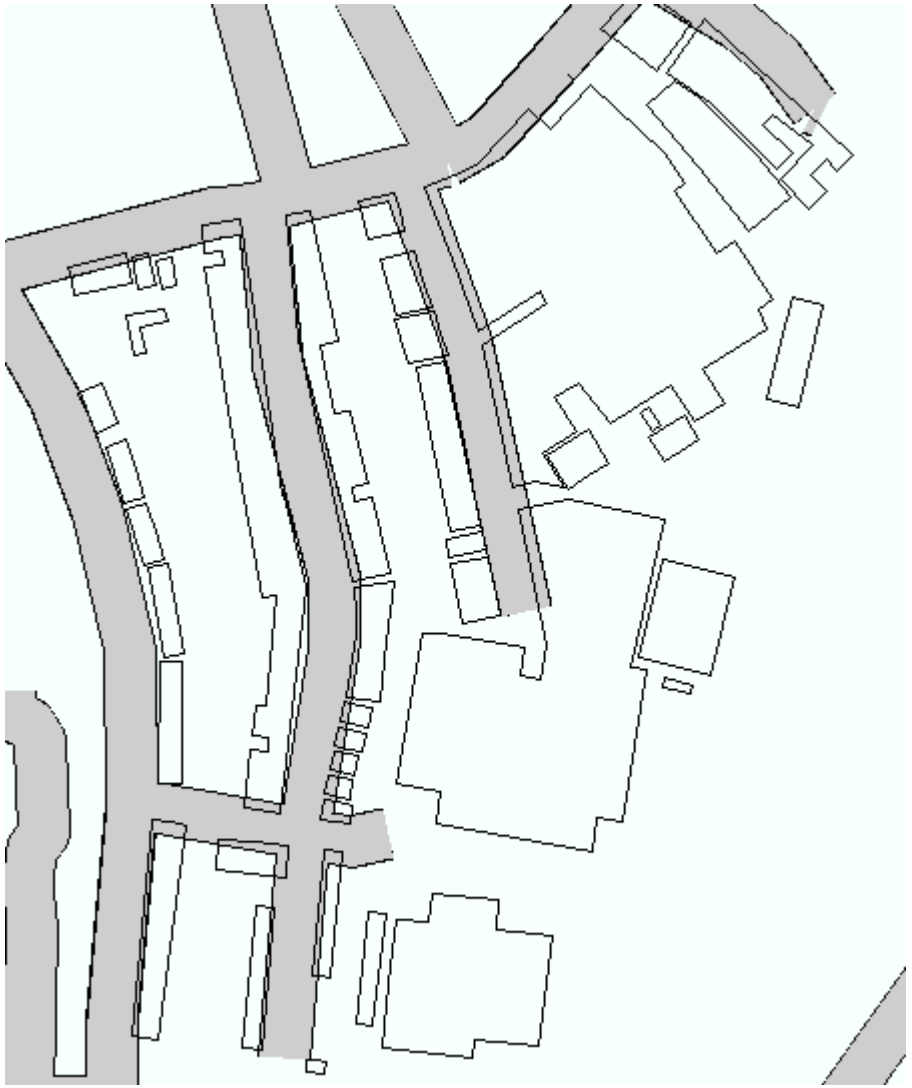
- Size reduce

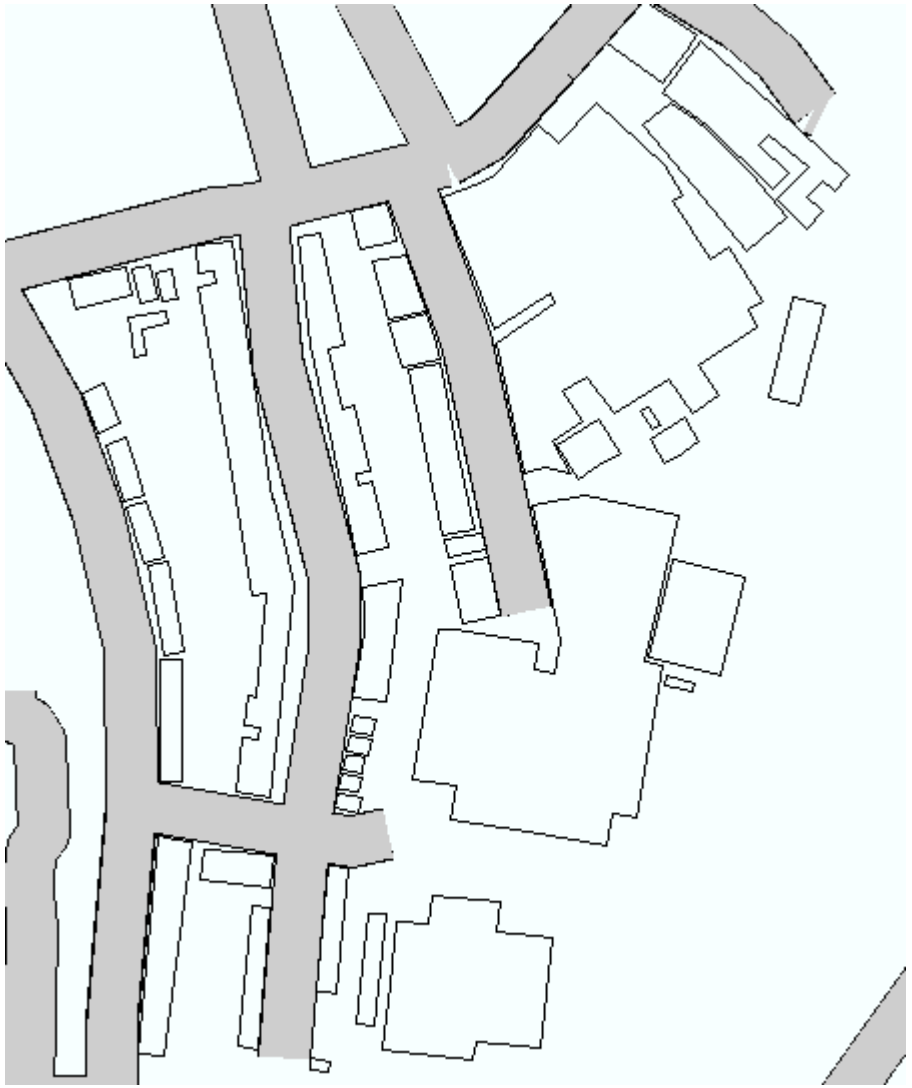






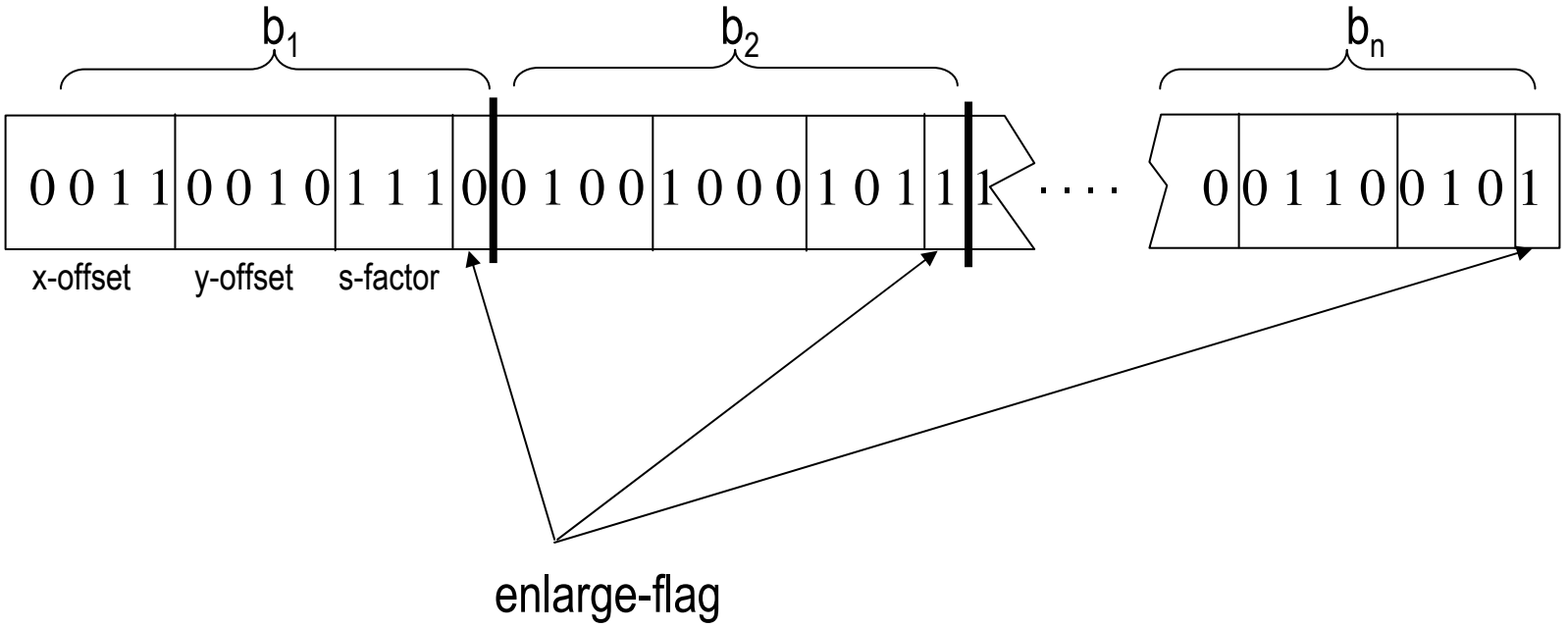


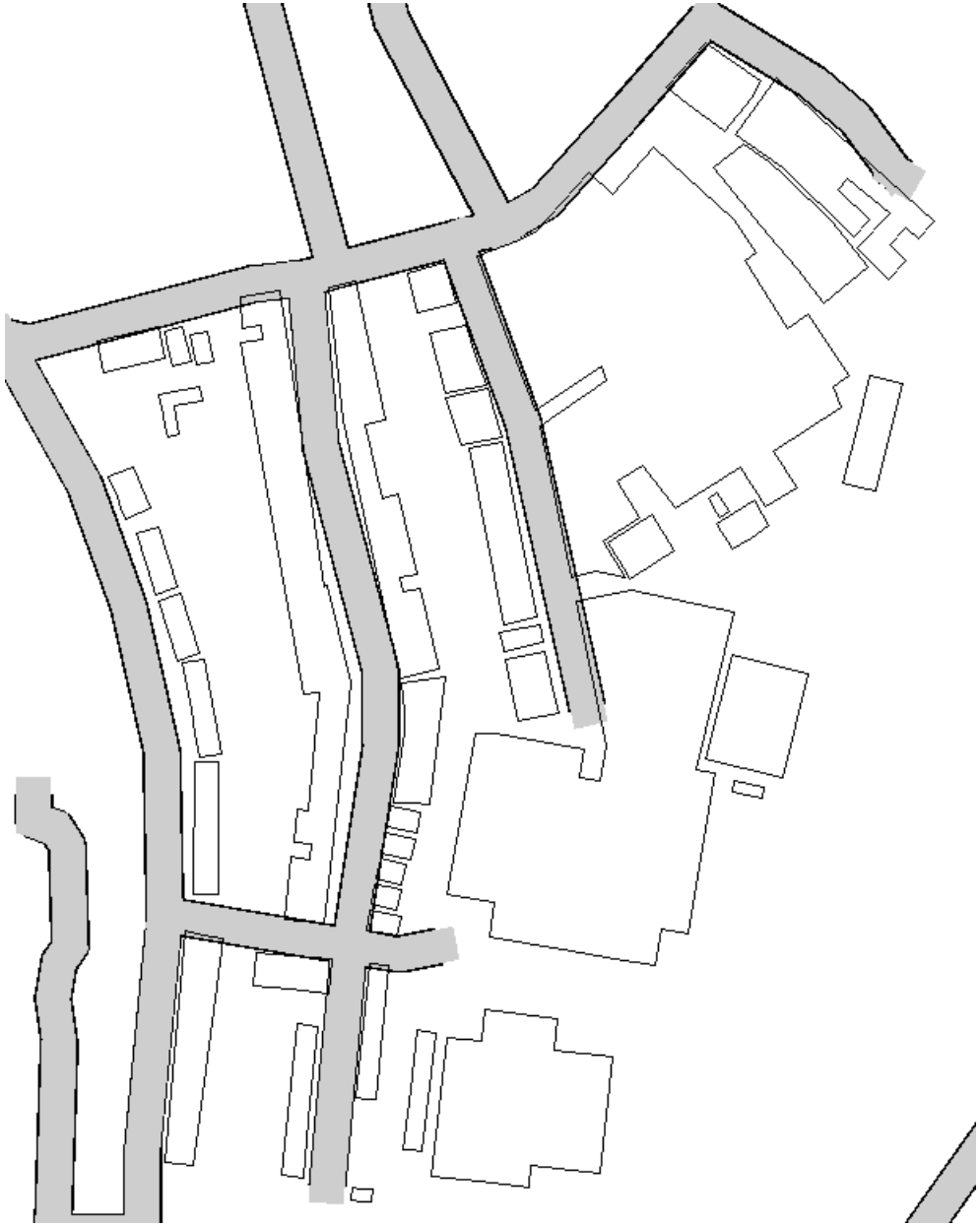


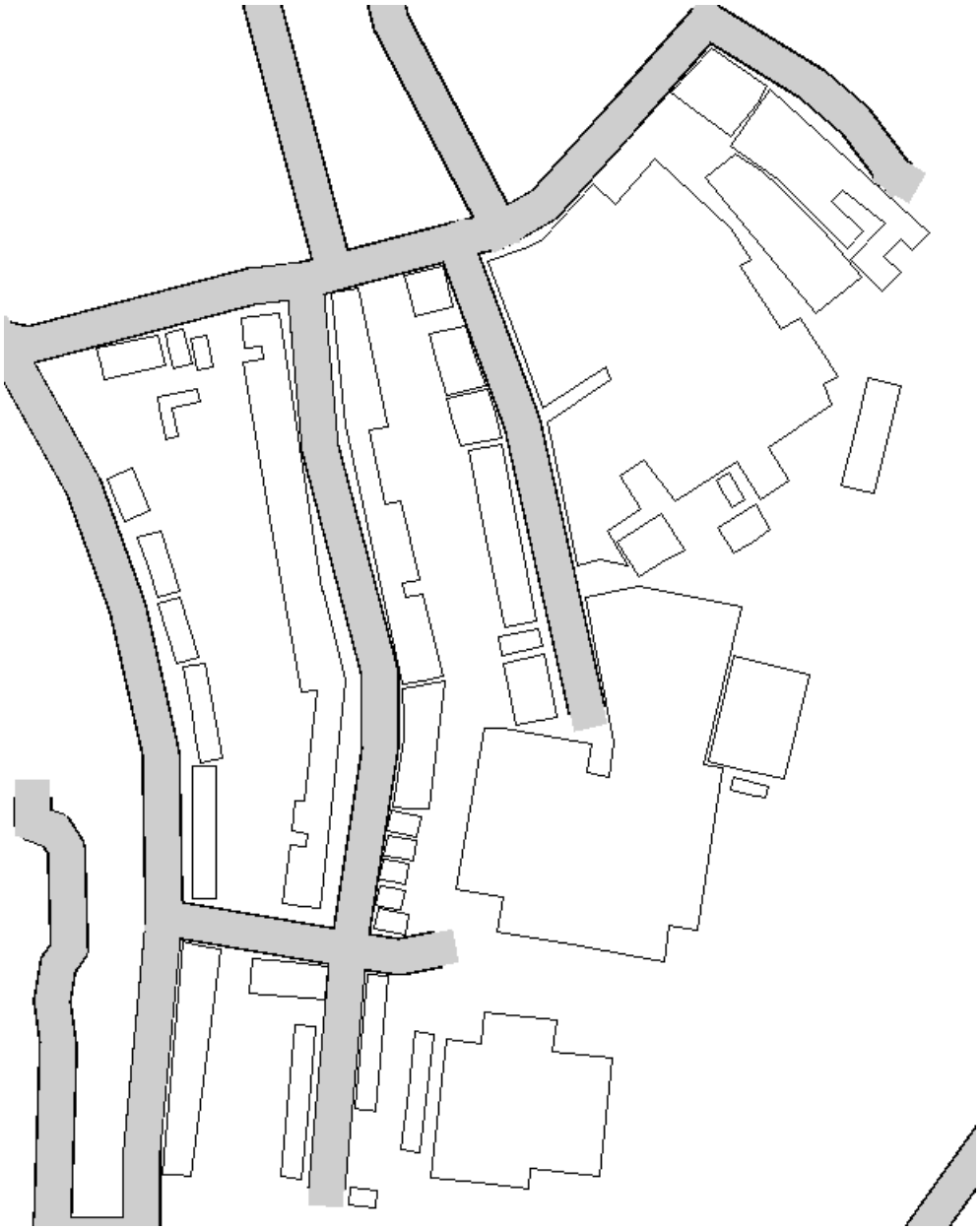


Additional Operator

- Size enlarge







Conclusions

- GA + trial positions is successful in resolving conflict (in reasonable time)
- Introducing size reduce and size enlarge straightforward
- Not a complete map generalization solution, rather
  - a menu bar tool for semi automated system
  - a method available to objects within an AGENT system
  - one step in a more elaborate process, defined maybe something like a Global Master Plan

Future Work

- Much more experimentation with GA
  - see work of van Dijk, Thierens and de Berg (Geoinformatica 6:4)
- Compare with SA and TS
- If appropriate
  - More operators
  - More feature classes
  - Incorporate into more complete system (in which GA solution becomes just one of a number of functions available)