COMPACTS TYPO-MORPHOLOGIES BY USE OF LOCAL SEARCH METHODS

Isabelle De Smet, PhD
Faculty of Architecture & Urban Planning UMONS

Quentin Meurisse, PhD Student
Faculty of Sciences (Computer Sciences) UMONS

CO-MOD PROJECT

UMONS
University of Mons

Faculté d'Architecture et d'Urbanisme

Faculté des Sciences

Elaboration et expérimentation d'un outil d'évaluation et d'aide à la conception compacts à dominante d'habitat suivant une densité de population cible

Development and experimentation of a tool aiming to assess and to assist the design of compact housing blocks with a target population density

Isabelle De Smet, PhD

(2018) Faculty of Architecture & Urban Planning UMONS

Elaboration et expérimentation d'un outil d'évaluation et d'aide à la conception compacts à dominante d'habitat suivant une densité de population cible

Development and experimentation of a tool aiming to assess and to assist the design of compact housing blocks with a target population density

Isabelle De Smet, PhD

(2018) Faculty of Architecture & Urban Planning UMONS

How can we urbanise to satisfy the demand for new housing
while reducing **peri-urbanization** and over-consumption of **land**, **energy** and **materials**?

Elaboration et expérimentation d'un outil d'évaluation et d'aide à la conception compacts à dominante d'habitat suivant une densité de population cible

Development and experimentation of a tool aiming to assess and to assist the design of compact housing blocks with a target population density

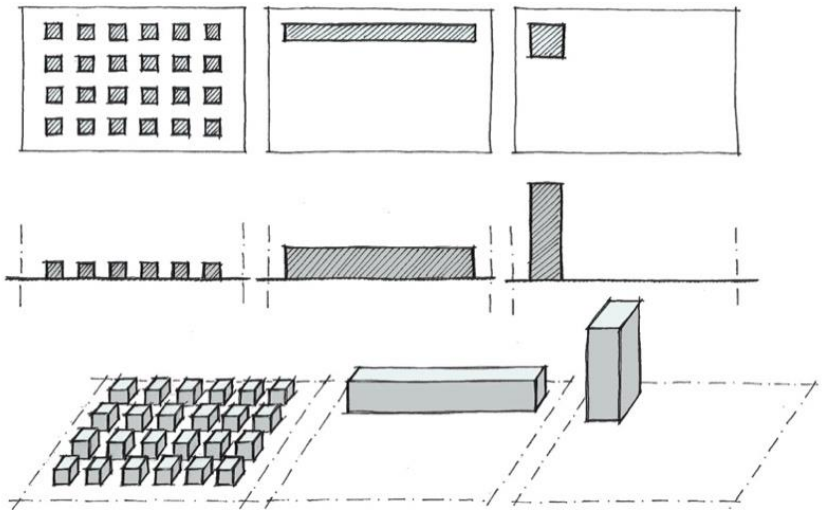
Isabelle De Smet, PhD

(2018) Faculty of Architecture & Urban Planning UMONS

How can we urbanise to satisfy the demand for new housing
while reducing **peri-urbanization** and over-consumption of **land**, **energy** and **materials**?

Acting on the **shapes**, both **volumes and surfaces**
Reducing the **built volume** and the **area of the outdoor spaces**, *in judicious way*

Building density

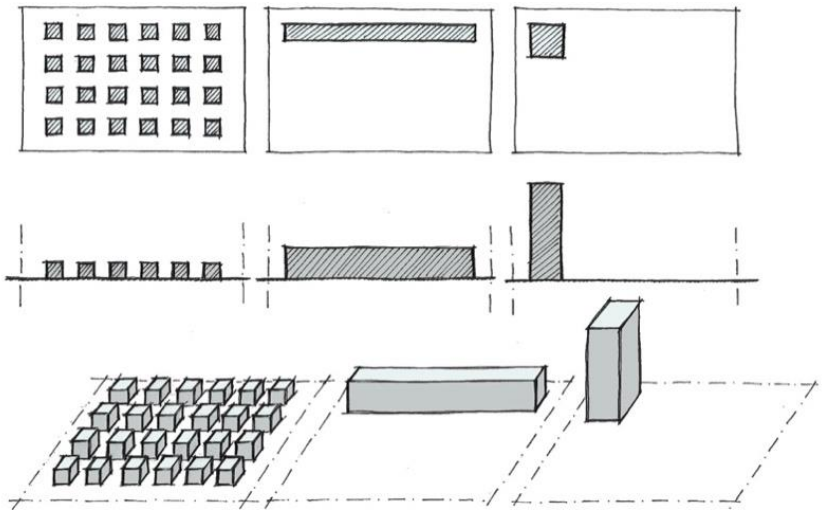


(De Smet, 2017) d'après (Fouchier 1997)

a value of density cannot be correlated with a typo-morphology

The **lack of bijective link** between **typo-morphology** and **density** can prevent efficient assessment of the impact of typo-morphologies **in terms of density** on the **built** and the **non-built environment**.

Building density



(De Smet, 2017) d'après (Fouchier 1997)

a value of density cannot be correlated with a typo-morphology

The **lack of bijective link** between **typo-morphology** and **density** can prevent efficient assessment of the impact of typo-morphologies **in terms of density** on the **built** and the **non-built environment**.

COMPACTNESS

COMPACTNESS

develop a **tool** to assist the design of **housing blocks** following a targeted **population density** and considering the concept of **compactness**

COMPACTNESS

develop a **tool** to assist the design of **housing blocks** following a targeted **population density** and considering the concept of **compactness**

Surface compactness (macro scale) + building compactness (micro scale)
Used in the tool on the **infra-local scale**

The plot **The buildings** **The non-built land**

COMPACTNESS

develop a **tool** to assist the design of **housing blocks** following a targeted **population density** and considering the concept of **compactness**

Surface compactness (macro scale) + building compactness (micro scale)
Used in the tool on the **infra-local scale**

The plot **The buildings** **The non-built land**

COMPACTNESS < > THE SHAPES

↘ **The land use**

↘ **The envelope surface of building**

↗ **The spatial qualities of compact in shape spaces**

COMPACTNESS

develop a **tool** to assist the design of **housing blocks** following a targeted **population density** and considering the concept of **compactness**

Surface compactness (macro scale) + building compactness (micro scale)
Used in the tool on the **infra-local scale**

The plot **The buildings** **The non-built land**

COMPACTNESS < > THE SHAPES

The land use

The envelope surface of building

The spatial qualities of compact in shape spaces

COMPACTNESS INDICATORS

=

One of the objectives of the tool

COMPACTNESS

SPATIAL & MORPHOLOGICAL CONSTRAINTS

COMPACTNESS

SPATIAL & MORPHOLOGICAL CONSTRAINTS

a minimum volume and area of accommodation by type of housing

+

COMPACTNESS

SPATIAL & MORPHOLOGICAL CONSTRAINTS

a minimum volume and area of accommodation by type of housing

+

a rate of green spaces area per block, 50% of which consist of a maximum of two spaces of a compact shape

+

COMPACTNESS

SPATIAL & MORPHOLOGICAL CONSTRAINTS

a minimum volume and area of accommodation by type of housing

+

a rate of green spaces area per block, 50% of which consist of a maximum of two spaces of a compact shape

+

sufficient light on facades, outdoor and indoor spaces

COMPACTNESS

SPATIAL & MORPHOLOGICAL CONSTRAINTS

a minimum volume and area of accommodation by type of housing

+

a rate of green spaces area per block, 50% of which consist of a maximum of two spaces of a compact shape

+

sufficient light on facades, outdoor and indoor spaces

+

a minimum distance between facades in terms of promiscuity and accessibility for emergency services

+

COMPACTNESS

SPATIAL & MORPHOLOGICAL CONSTRAINTS

a minimum volume and area of accommodation by type of housing

+

a rate of green spaces area per block, 50% of which consist of a maximum of two spaces of a compact shape

+

sufficient light on facades, outdoor and indoor spaces

+

a minimum distance between facades in terms of promiscuity and accessibility for emergency services

+

a feeling of closure of the whole and its components, enabling some porosity between the inside and the outside of the urban block

COMPACTNESS

SPATIAL & MORPHOLOGICAL CONSTRAINTS

a minimum volume and area of accommodation by type of housing

+

a rate of green spaces area per block, 50% of which consist of a maximum of two spaces of a compact shape

+

enough light on facades, outdoor and indoor spaces

+

a minimum distance between facades in term of promiscuity and accessibility of emergency services

+

a feeling of closure of the whole and its component, enabling a certain porosity between the inside and the outside of the urban block

DIFFICULT CONCILIATION BETWEEN VARIOUS QUANTITATIVE AND QUALITATIVE PARAMETERS

DIFFICULT CONCILIATION BETWEEN VARIOUS QUANTITATIVE AND QUALITATIVE PARAMETERS

STATIC TOOL

DIFFICULT CONCILIATION BETWEEN VARIOUS QUANTITATIVE AND QUALITATIVE PARAMETERS

***STATIC* TOOL**

***DYNAMIC* TOOL**

DIFFICULT CONCILIATION BETWEEN VARIOUS QUANTITATIVE AND QUALITATIVE PARAMETERS

***STATIC* TOOL**

***DYNAMIC* TOOL**

A PROCESS TO ORDER THE INDICATORS AND THE CONSTRAINS

DIFFICULT CONCILIATION BETWEEN VARIOUS QUANTITATIVE AND QUALITATIVE PARAMETERS

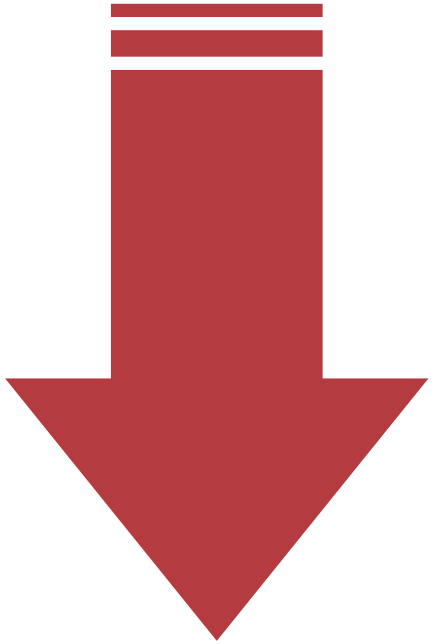
***STATIC* TOOL**

***DYNAMIC* TOOL**

A PROCESS TO HANDLE THE INDICATORS AND THE CONSTRAINS

A TRUE OPTIMIZATION PROCESS

DYNAMIC TOOL



Use of *local search* methods

- Local search is a method to solve *optimization problem.*

An *optimization problem* can be formulated in this way: given a set of possible *solutions* for a problem, we want to find the solution that minimize or maximize some criterion called *objective function*.

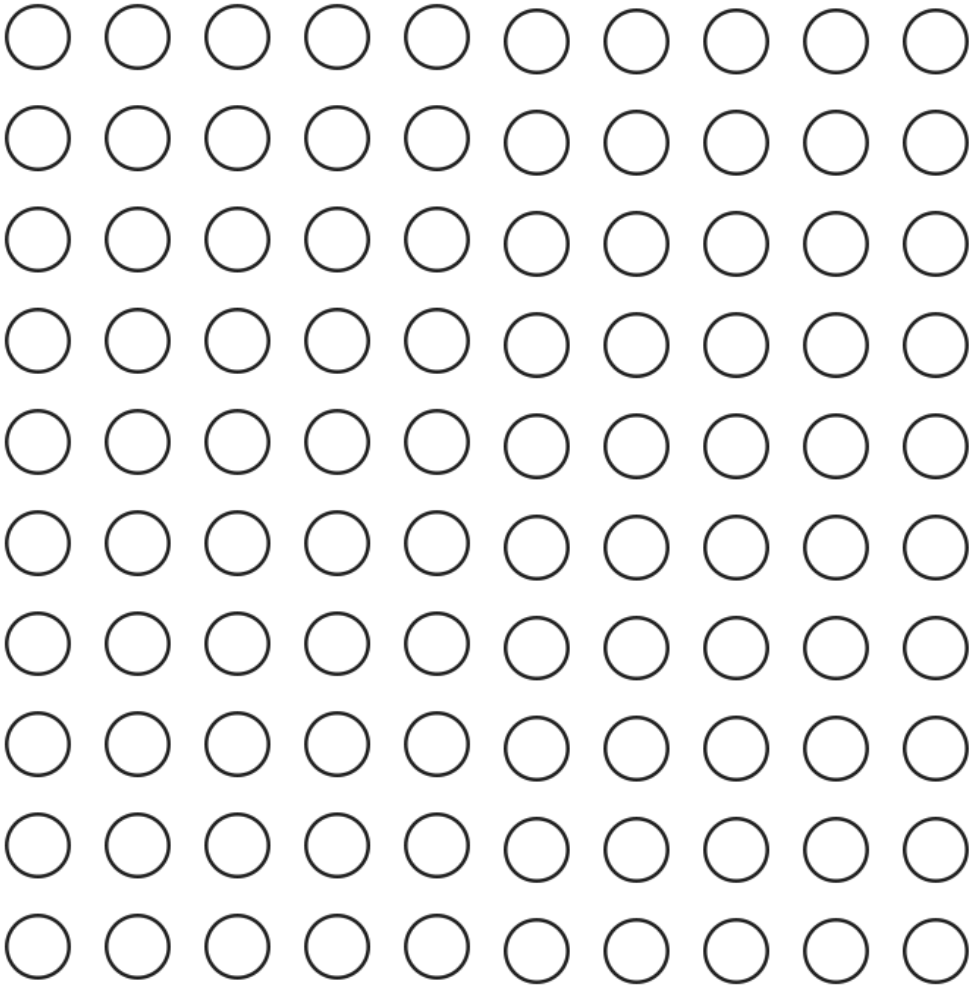
- Local search is a method to solve *optimization problem.*

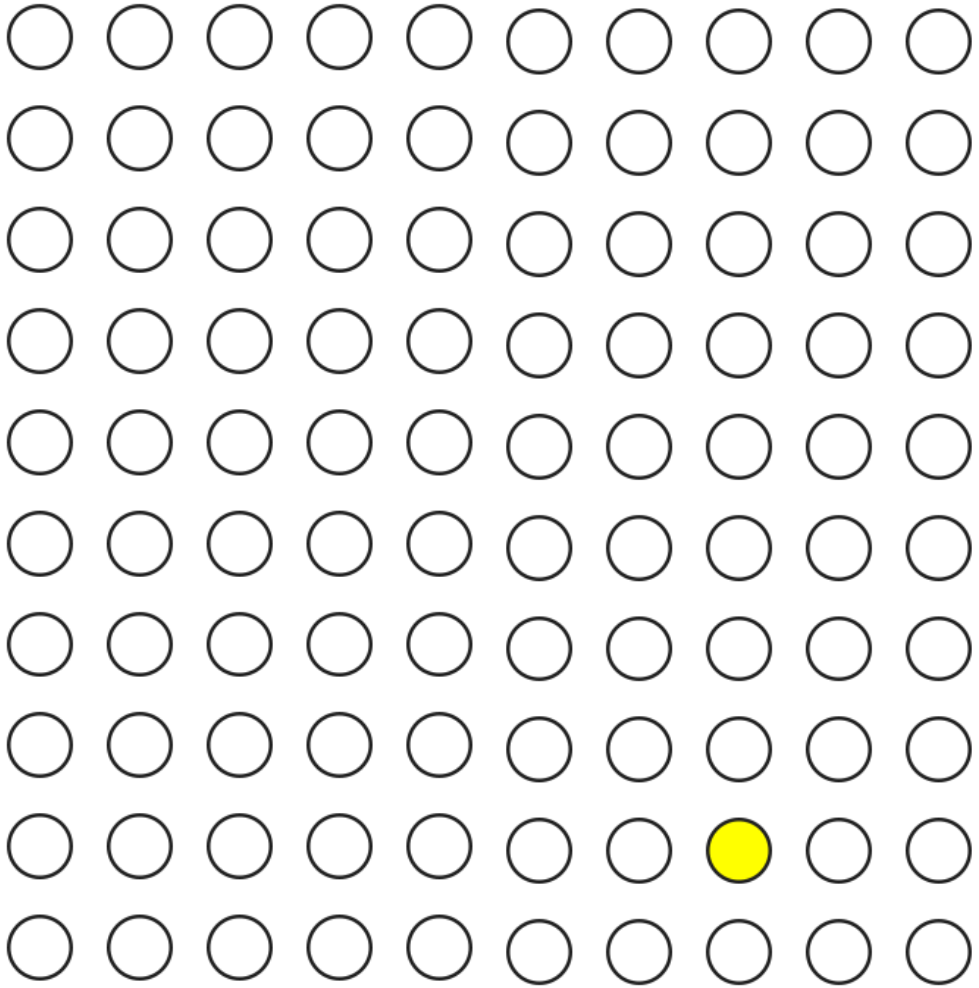
An *optimization problem* can be formulated in this way: given a set of possible *solutions* for a problem, we want to find the solution that minimize or maximize some criterion called *objective function*.

- Local search algorithms use the objective function to evaluate the quality of a solution.

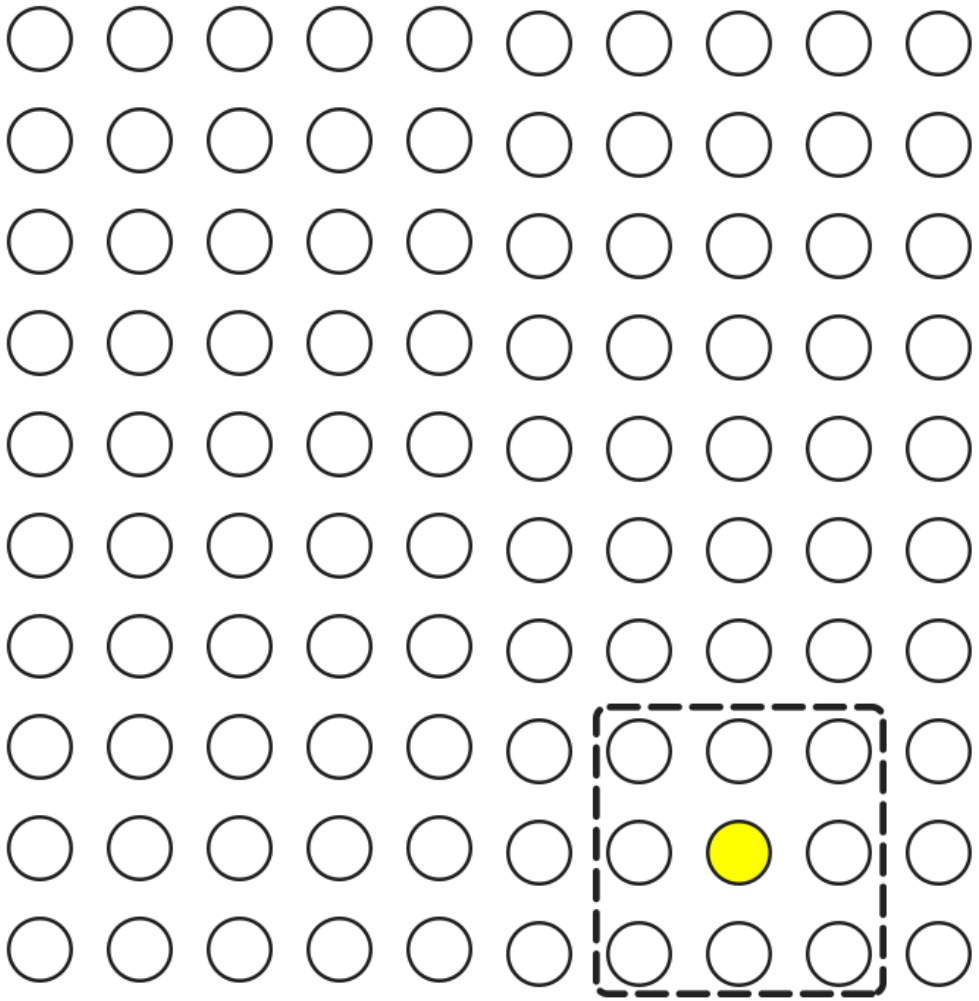
- To select solutions, local search use a neighbourhood operations. A neighbourhood of a solution s is a set of solutions that can be reach by performing a small transformation on s.

- Let a set of solutions.

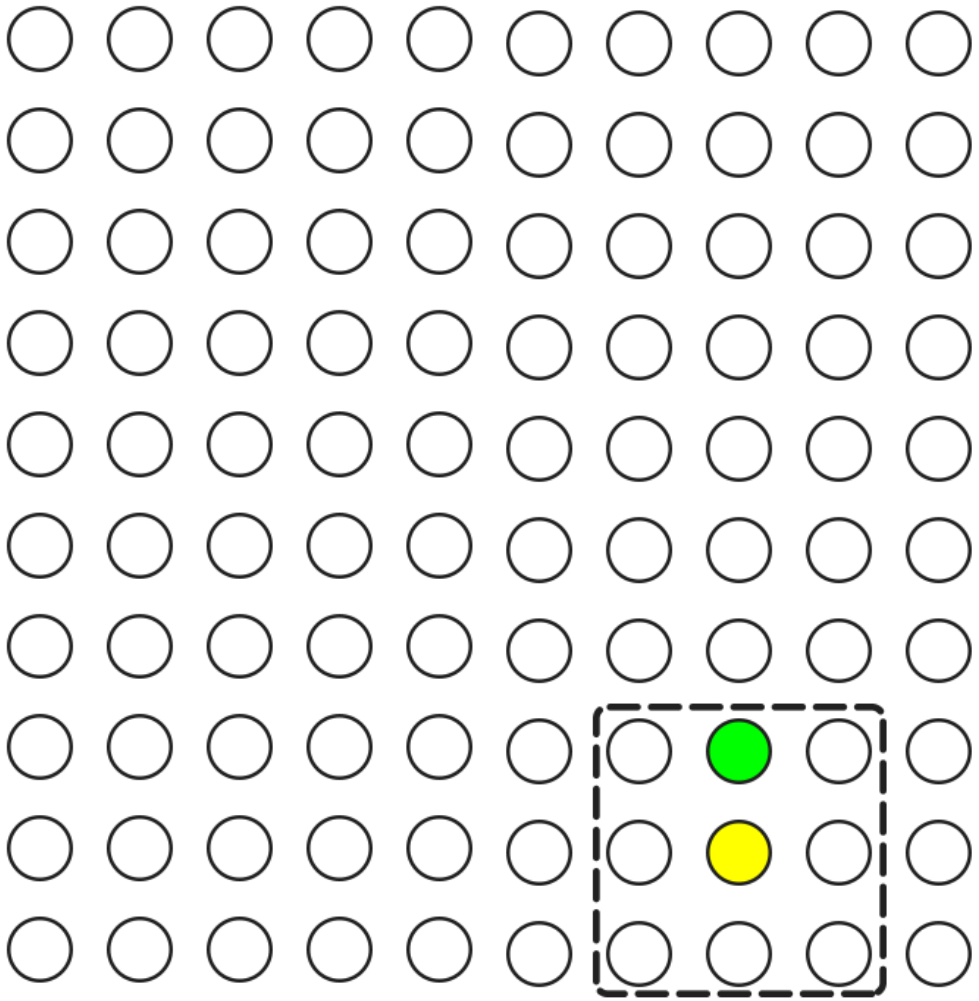




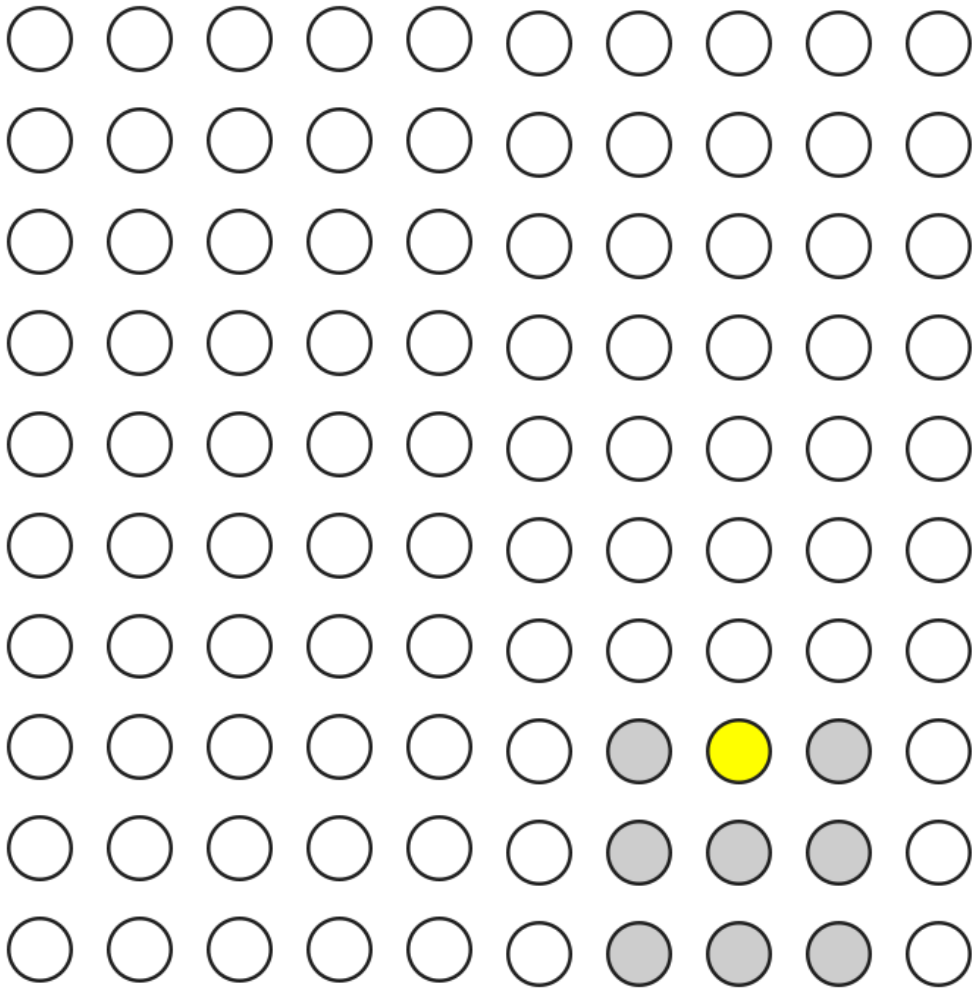
- Let a set of solutions.
- We start from a random solution (the yellow one)



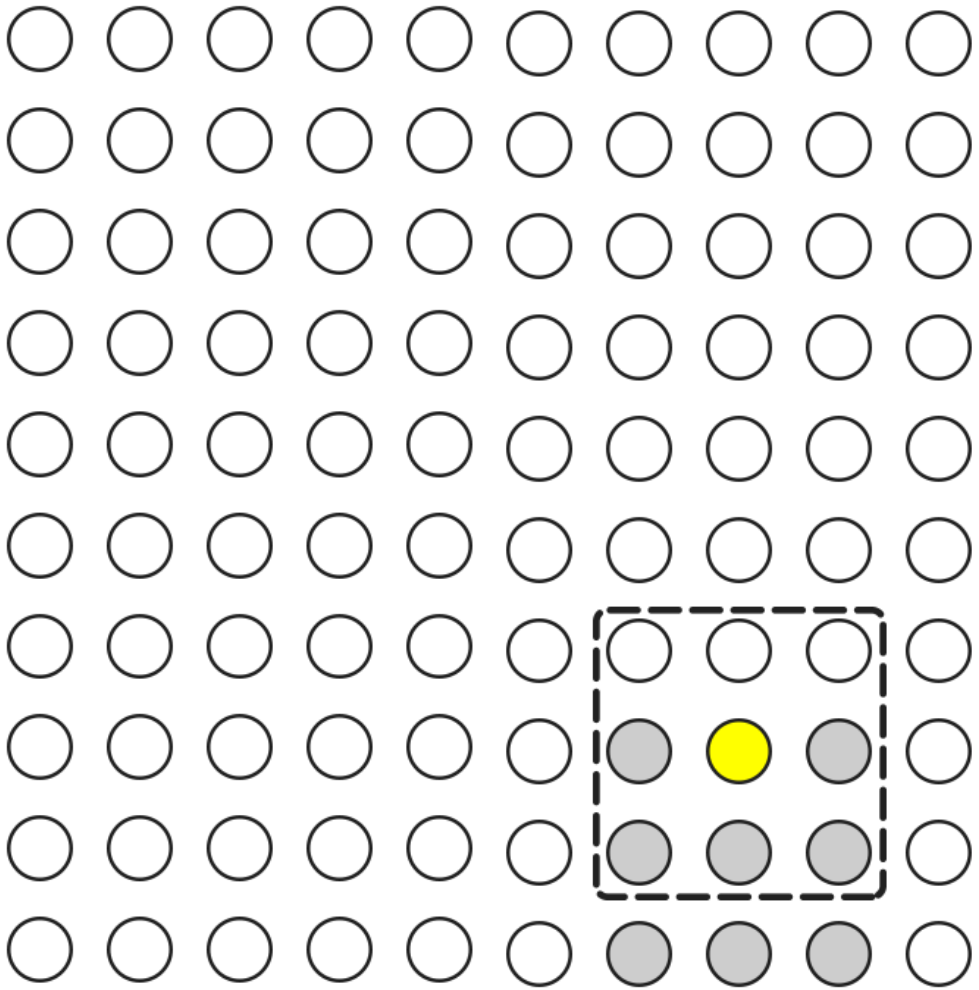
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution



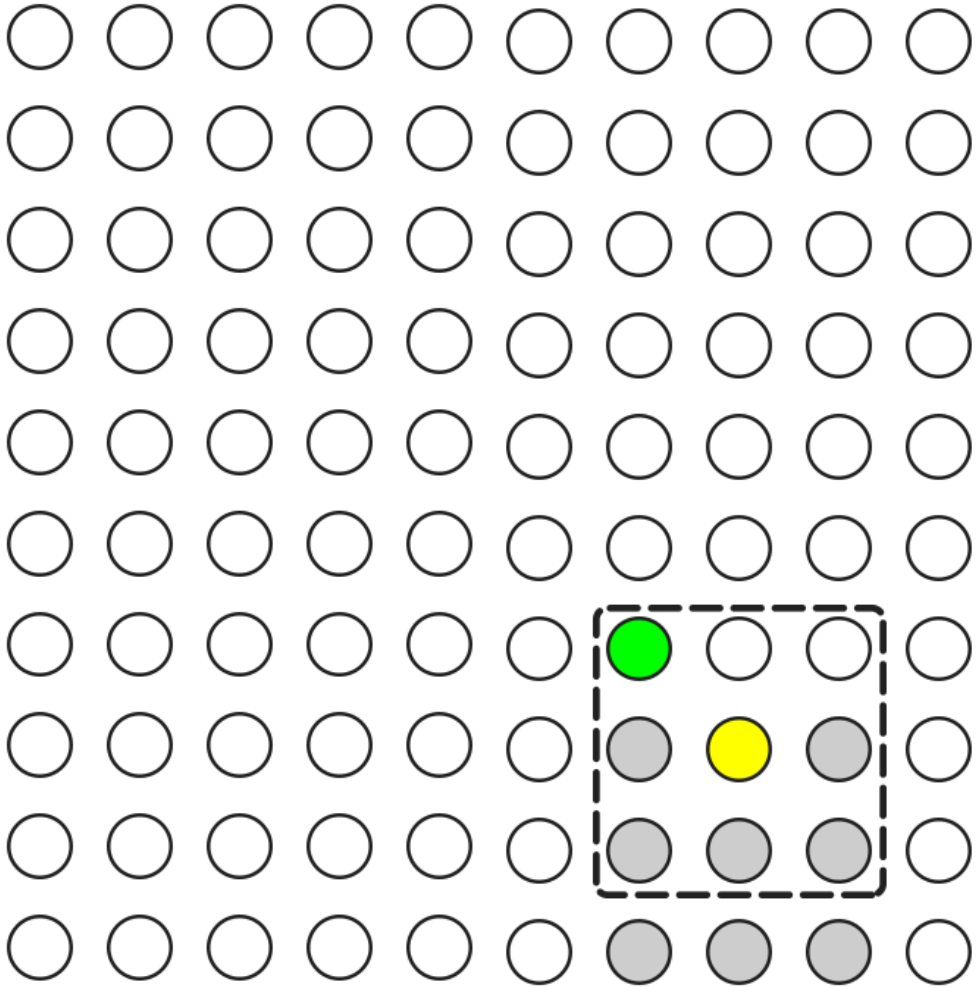
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)



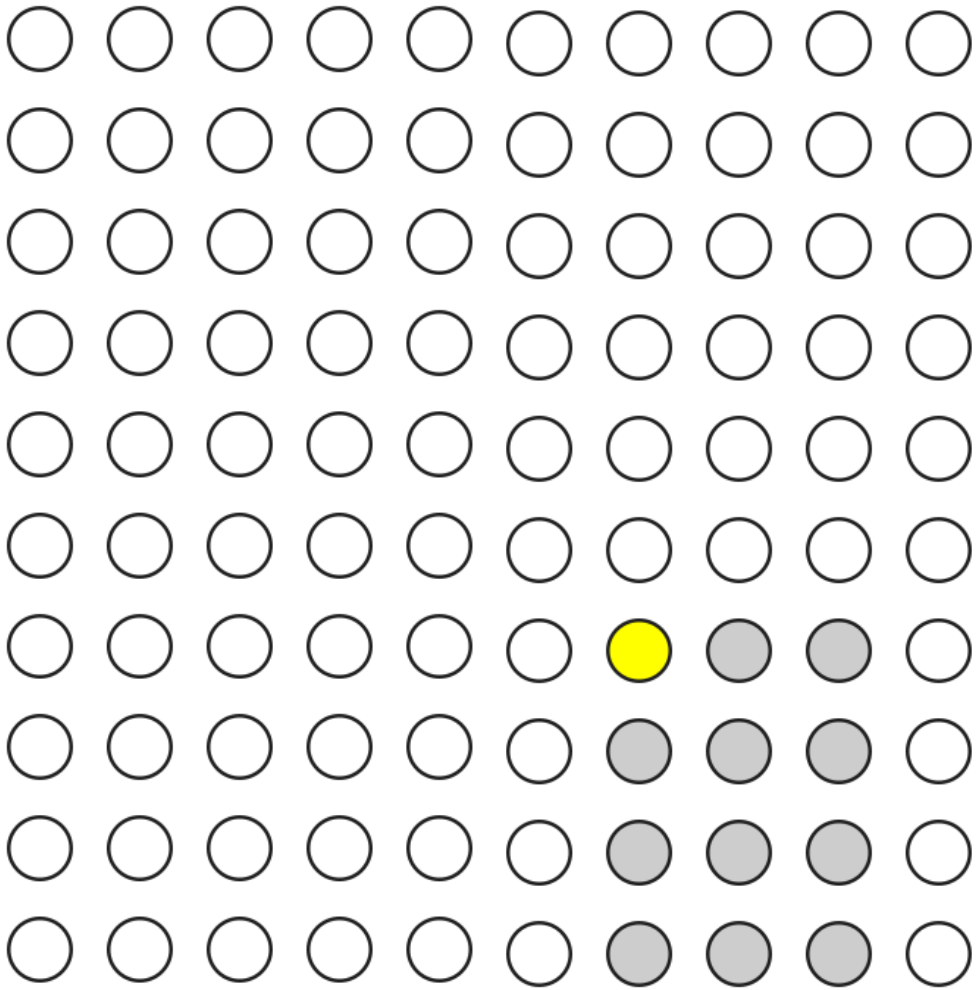
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution



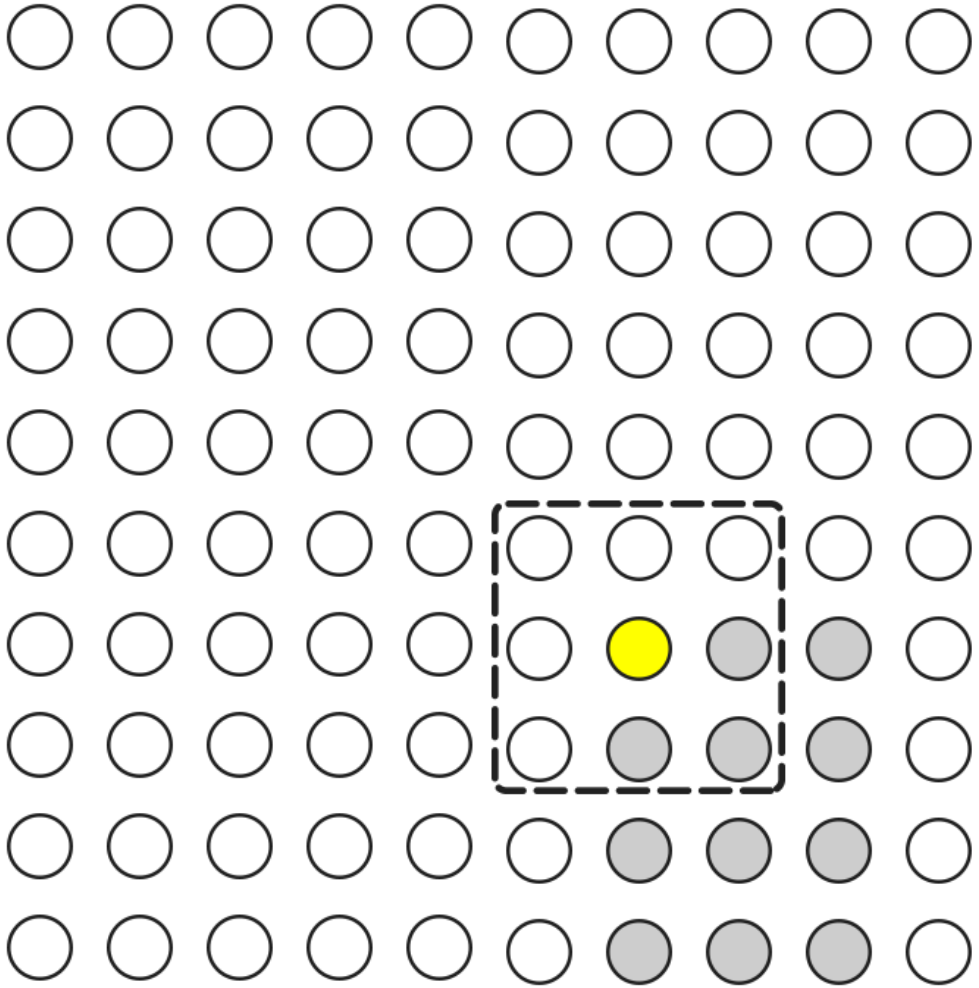
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



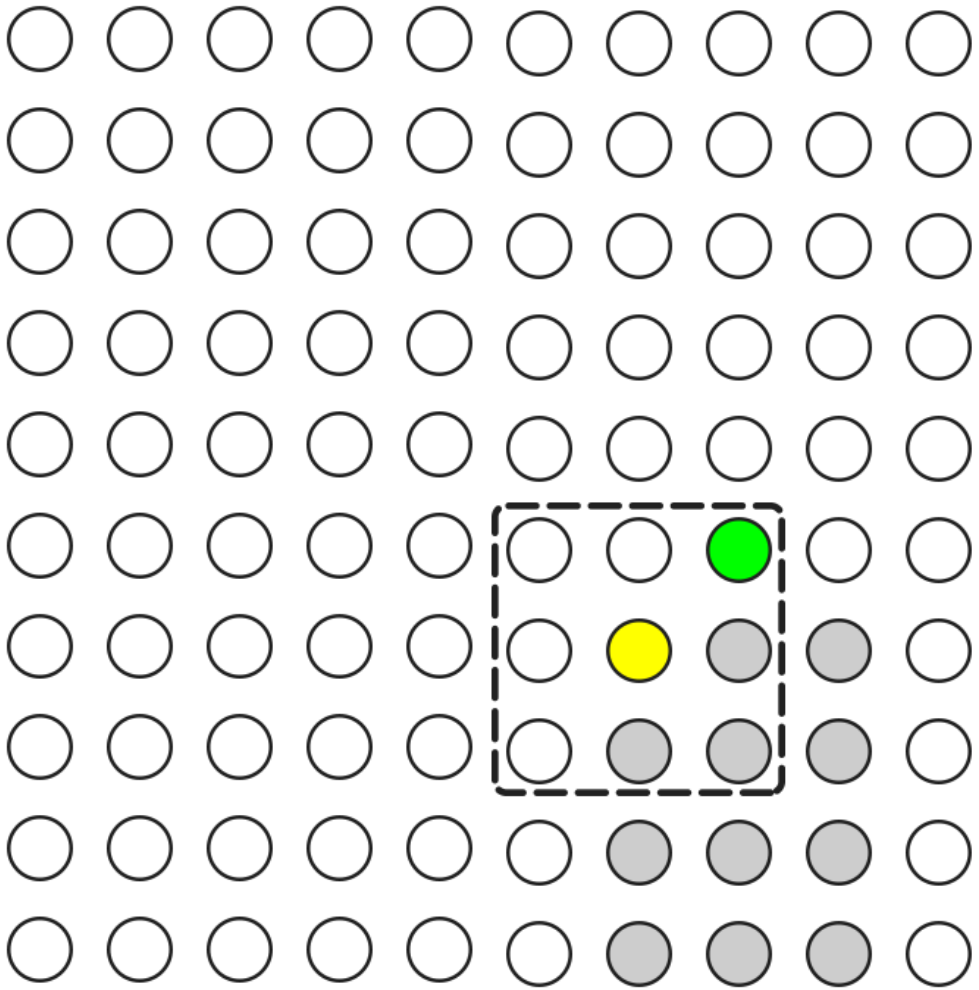
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



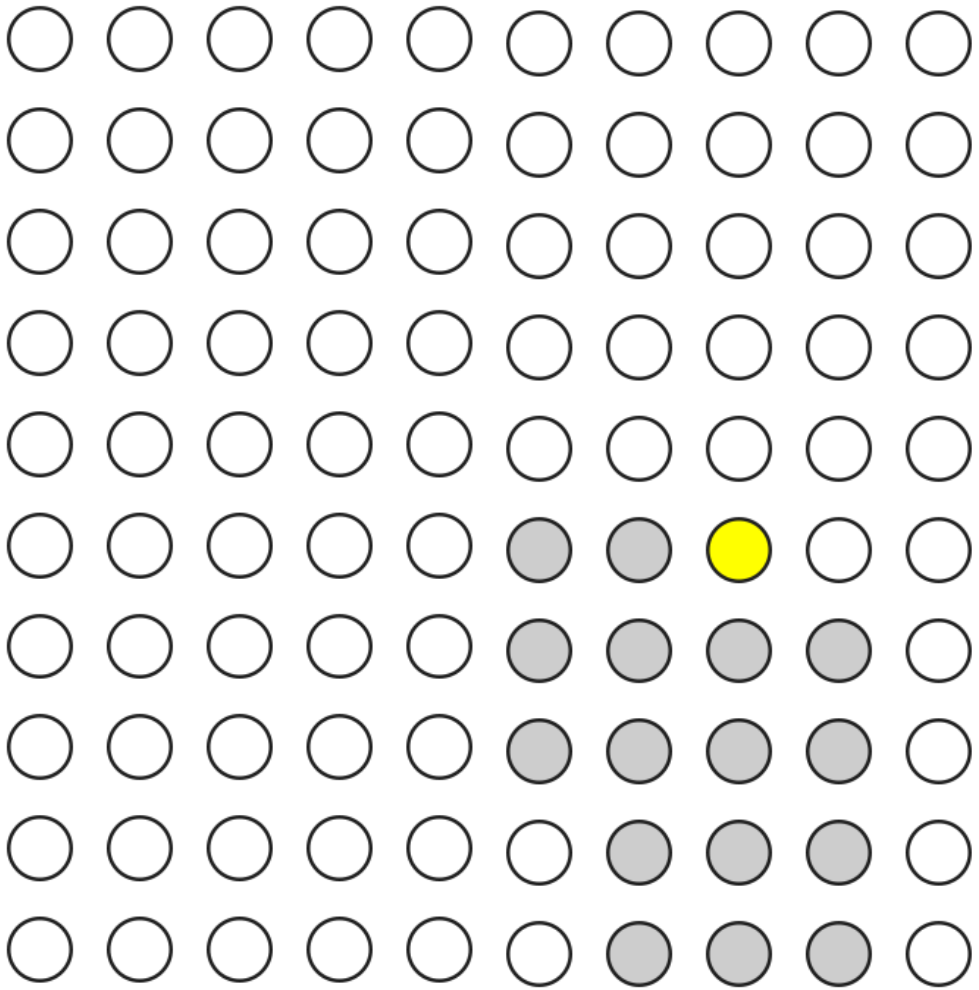
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



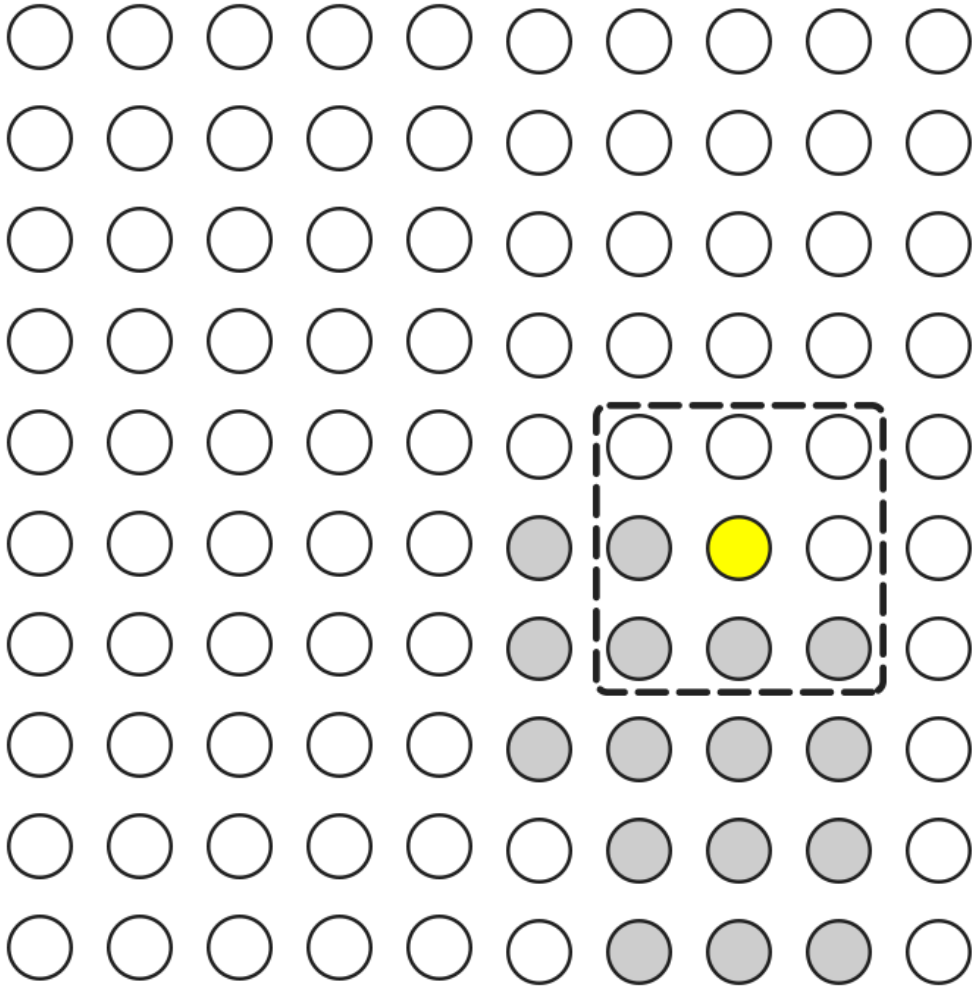
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



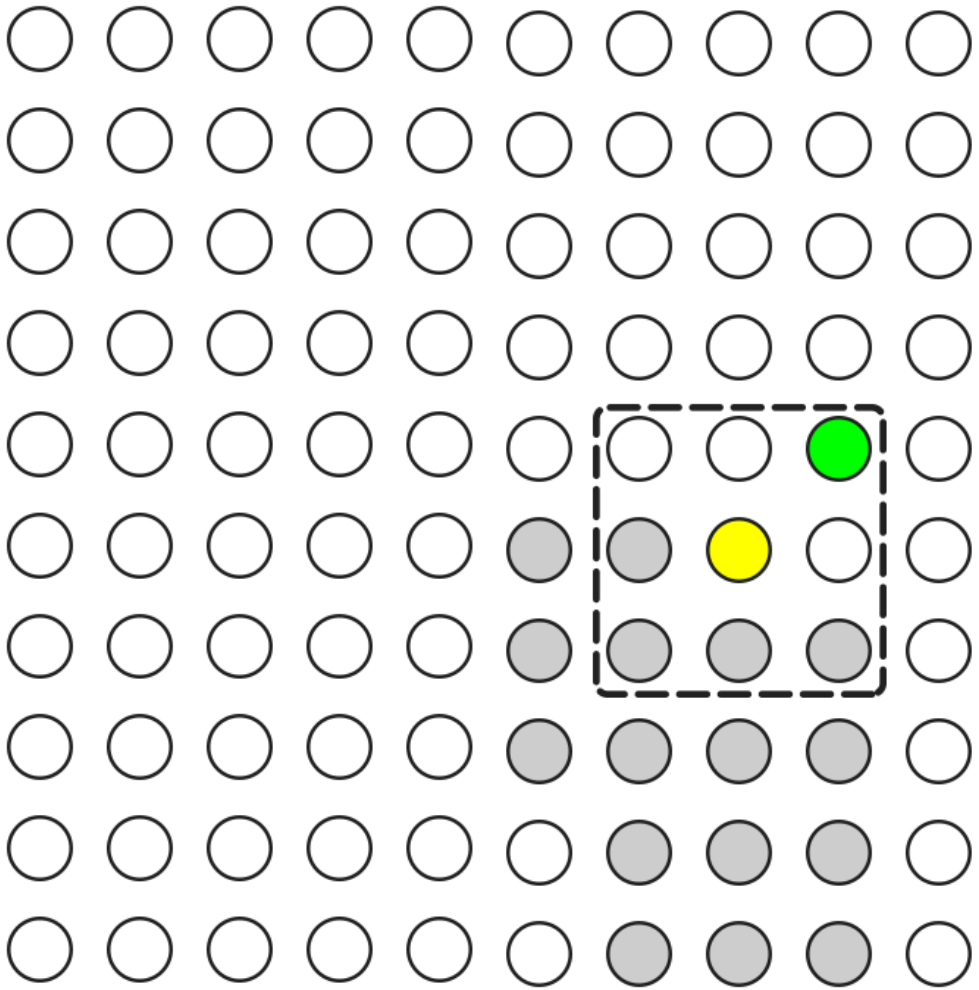
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



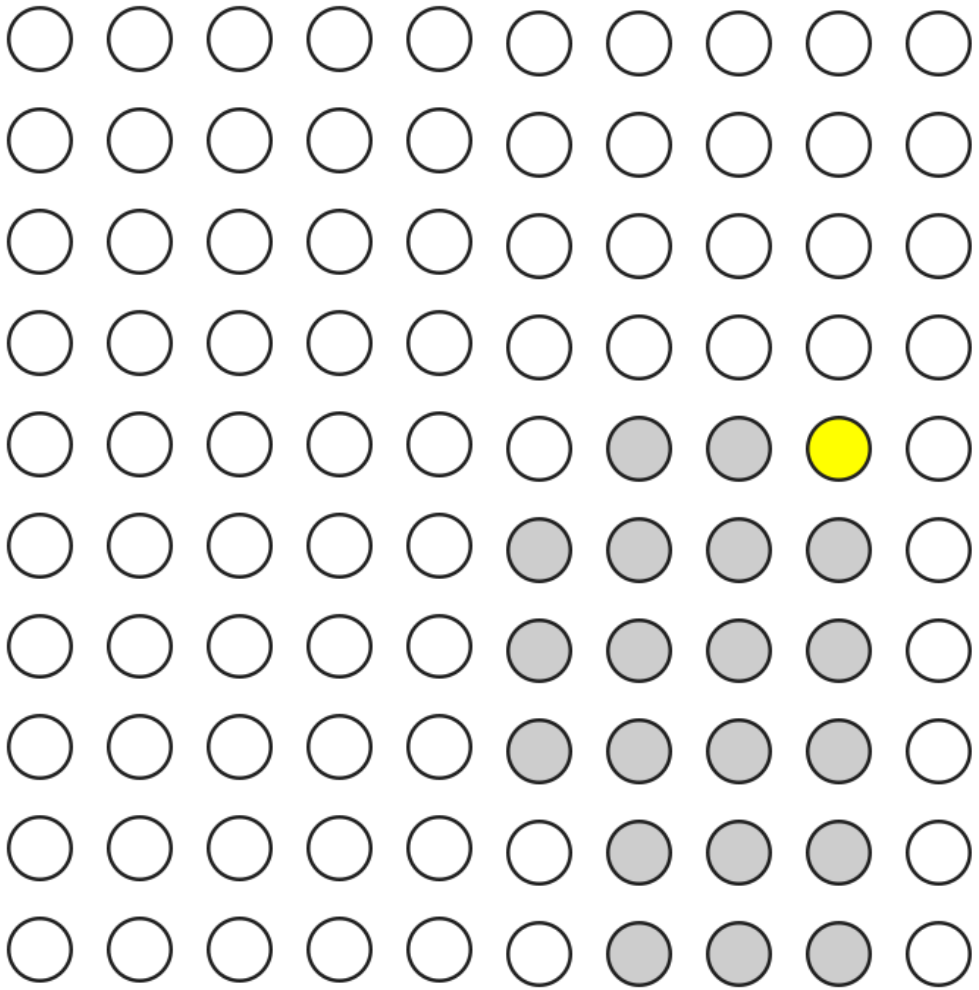
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



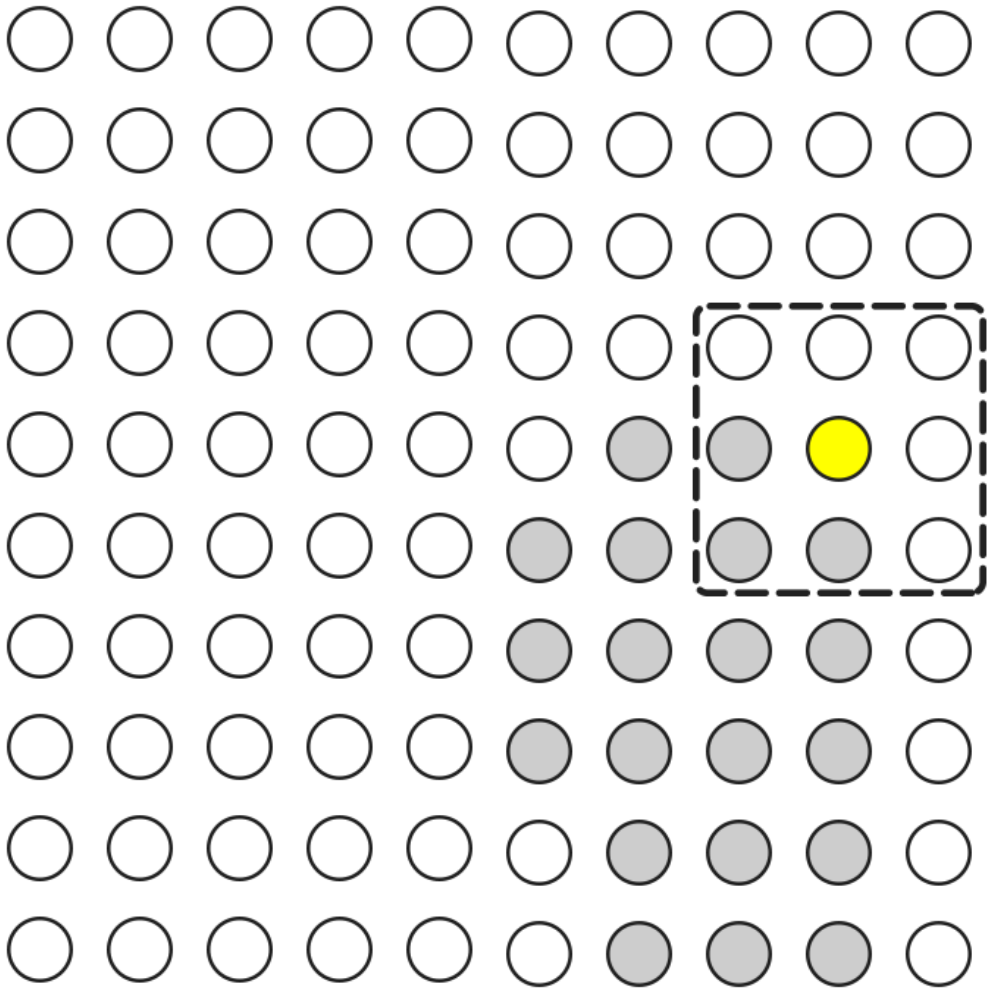
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



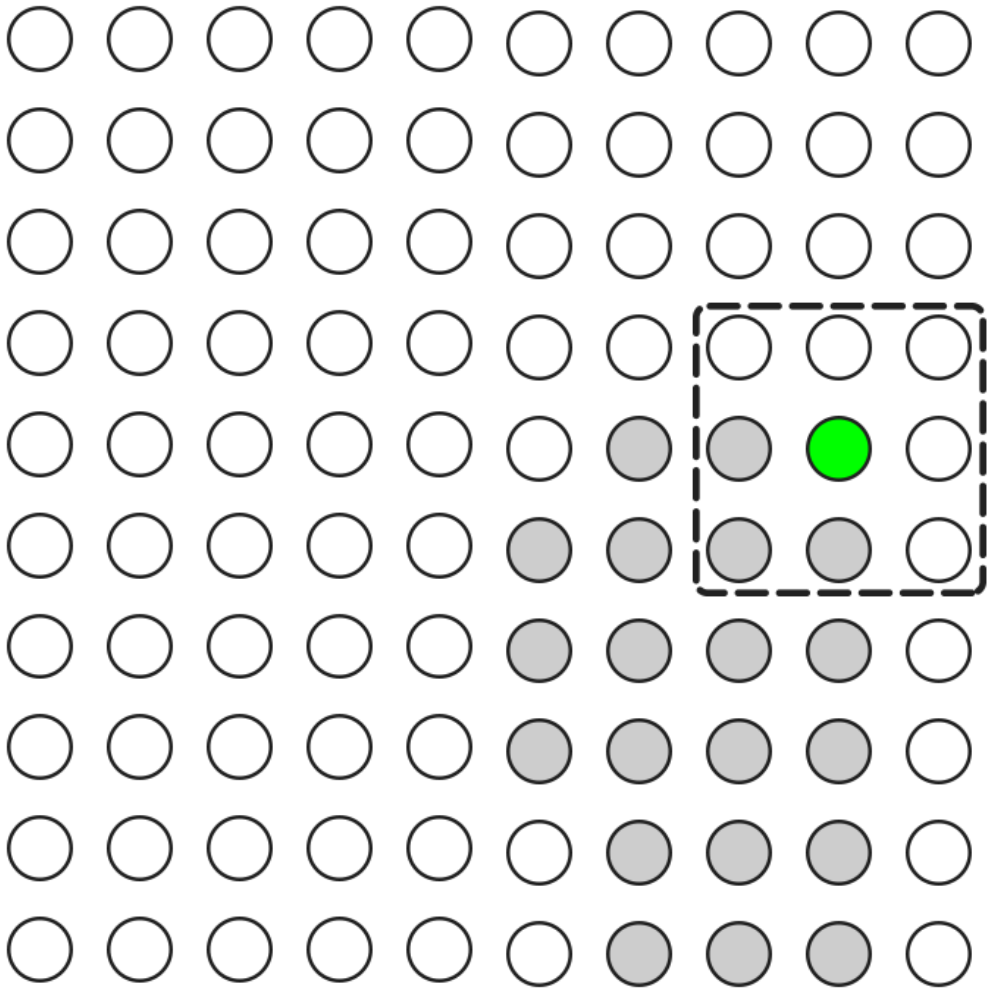
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



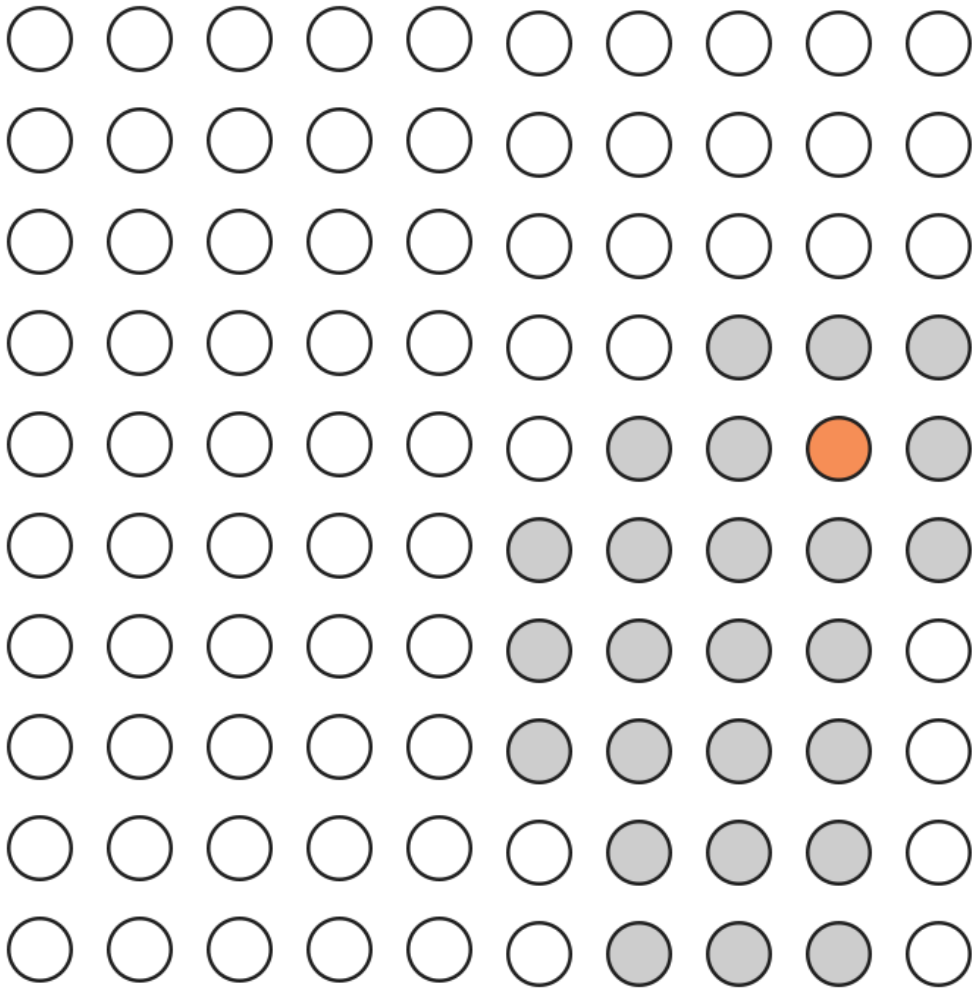
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution



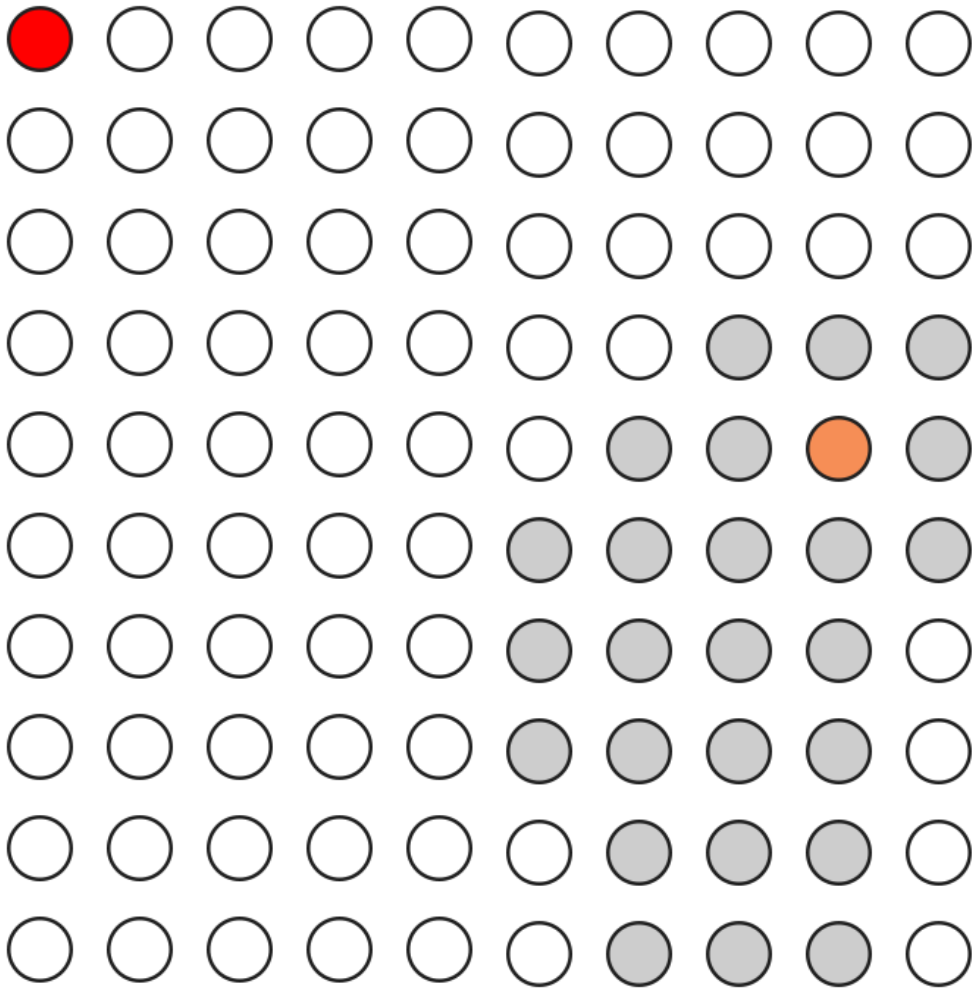
- Let a set of solutions.
- We start from a random solution (the yellow one)
- We evaluate the neighbours of this solution
- We find the best solution in our neighbourhood (the green one)
- The green solution becomes our current solution
- We iterate the procedure from our new yellow solution
- We stop when we find a solution that is better than all its neighbours.

- Local search is not perfect.
- The basic local search doesn't guarantees that we finally get the best solution.

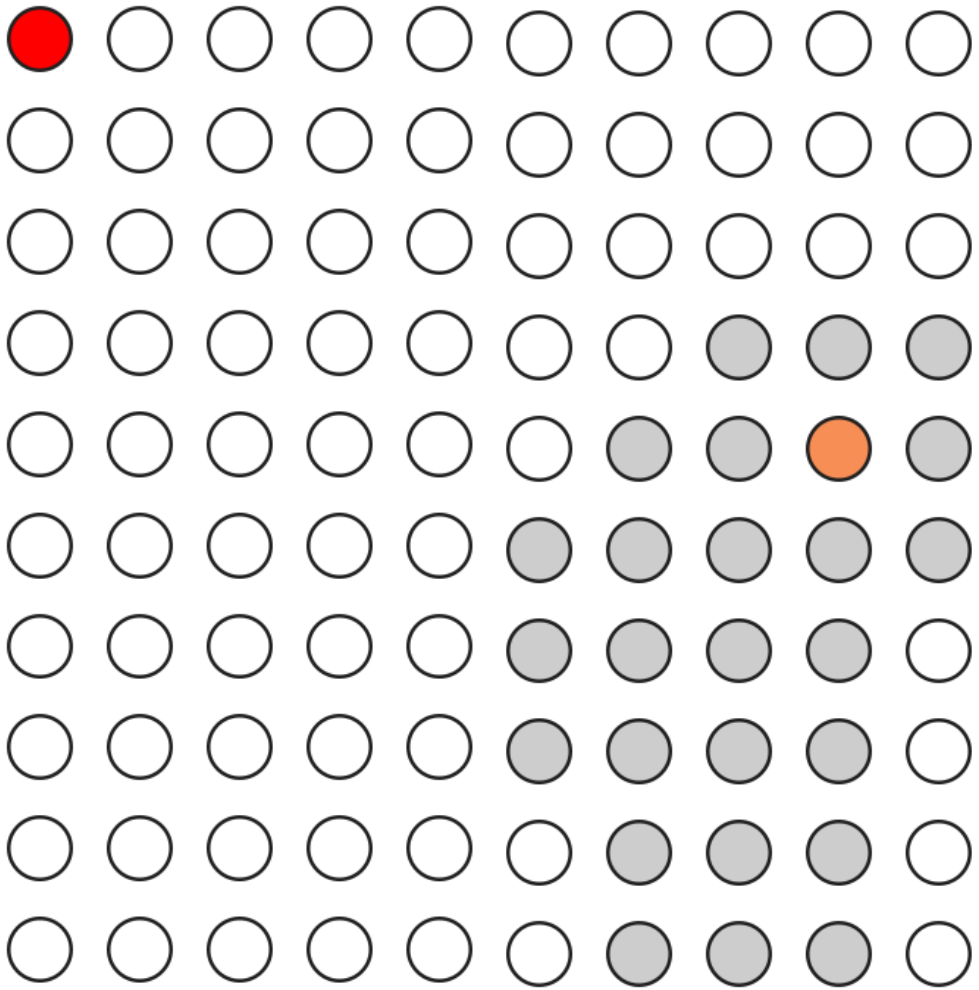
- Local search is not perfect.
- The basic local search doesn't guarantees that we finally get the best solution.
- The returned solution is the best *locally* and not *globally.*



- Local search is not perfect.
- The basic local search doesn't guarantees that we finally get the best solution.
- The returned solution is the best *locally* and not *globally.*
- In our example, we only visited the grey solutions and returned the orange one.



- Local search is not perfect.
- The basic local search doesn't guarantees that we finally get the best solution.
- The returned solution is the best *locally* and not *globally*
- In our example, we only visited the grey solutions and returned the orange one.
- However, the best solution is the red one.



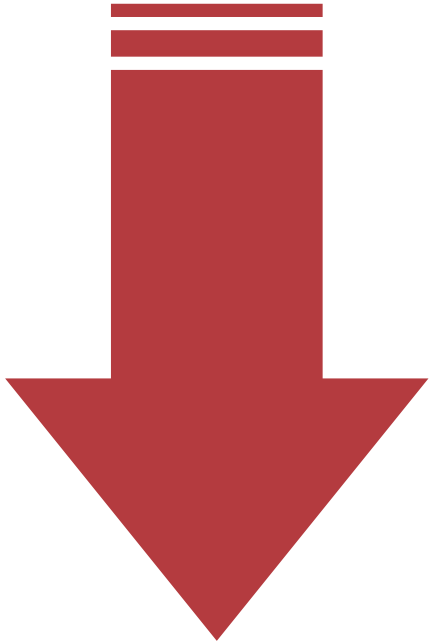
- Local search is not perfect.
- The basic local search doesn't guarantees that we finally get the best solution.
- The returned solution is the best *locally* and not *globally*
- In our example, we only visited the grey solutions and returned the orange one.
- However, the best solution is the red one.

We need other search methods called *metaheuristics.*

Besides to optimize the objective function, local search can try to satisfy some constraints.

- *Mandatory constraints*: local search just ignore solutions that don't verify some criterion.
- *Penalising constraints*: give a penalty to the evaluation of solutions that don't verify some criterion.

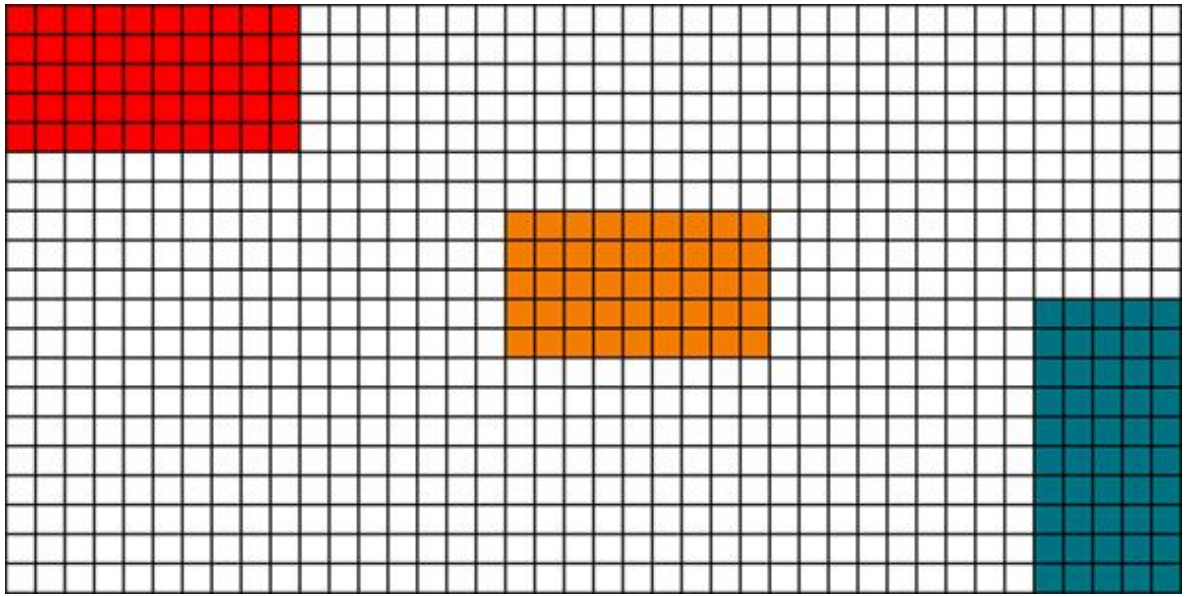
LOCAL SEARCH



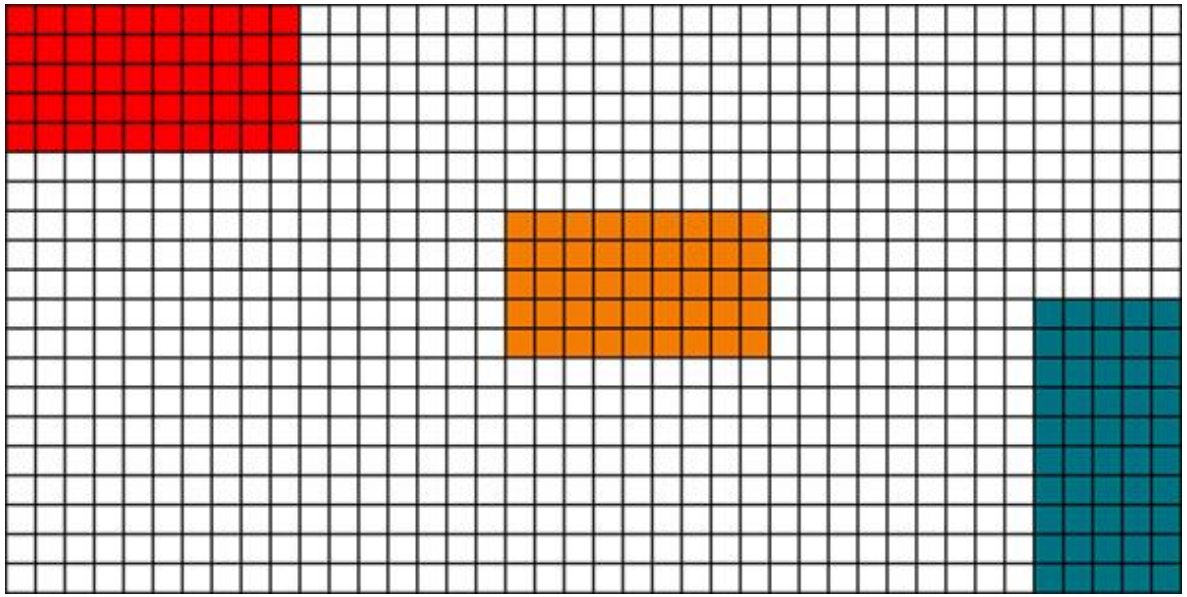
Our model

We are currently developing of prototype of software that generates urban blocks using local search algorithms. The generated blocks optimize compactness criteria with respect to a set of comfort, privacy and regulation constraints.

- A *solution* (an urban block) is represented as a chequered square or rectangle.
- Each cell of the quartering can have different colours: white for no-built environment and the other for buildings.
- A block of cells with the same colour is a building.

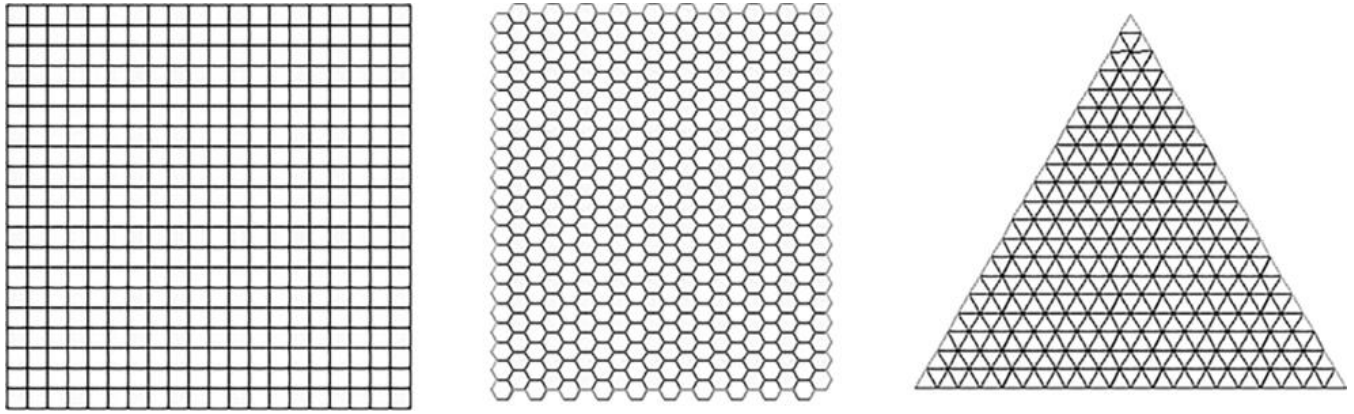


- A *solution* (an urban block) is represented as a chequered square or rectangle.
- Each cell of the quartering can have different colours: white for no-built environment and the other for buildings.
- A block of cells with the same colour is a building.



We currently don't worry about the height of the buildings.

- The tiling of an urban block is not limited to a square tiling. The model can work with any kind of tiling with regular polygons.
- The user can specify the size of the urban block and the size of the cells.
- The user can lock cells. This feature allows us to model the presence of pre-existing buildings or protected green space in our urban block.



Objective function: we first neglect the compactness of the urban block to focus on the porosity of the urban block (ratio between the built area and the total area of the urban block). We want to minimize the porosity.

Objective function: we first neglect the compactness of the urban block to focus on the porosity of the urban block (ratio between the built area and the total area of the urban block). We want to minimize the porosity.

Neighbourhood operation: just changing the colour of a cell. This action can have as effect to:

- create a new building,
- expand an existing building,
- simply change the colour of a building. Two sufficiently close buildings with the same colour can merge to form a bigger one.

Objective function: we first neglect the compactness of the urban block to focus on the porosity of the urban block (ratio between the built area and the total area of the urban block). We want to minimize the porosity.

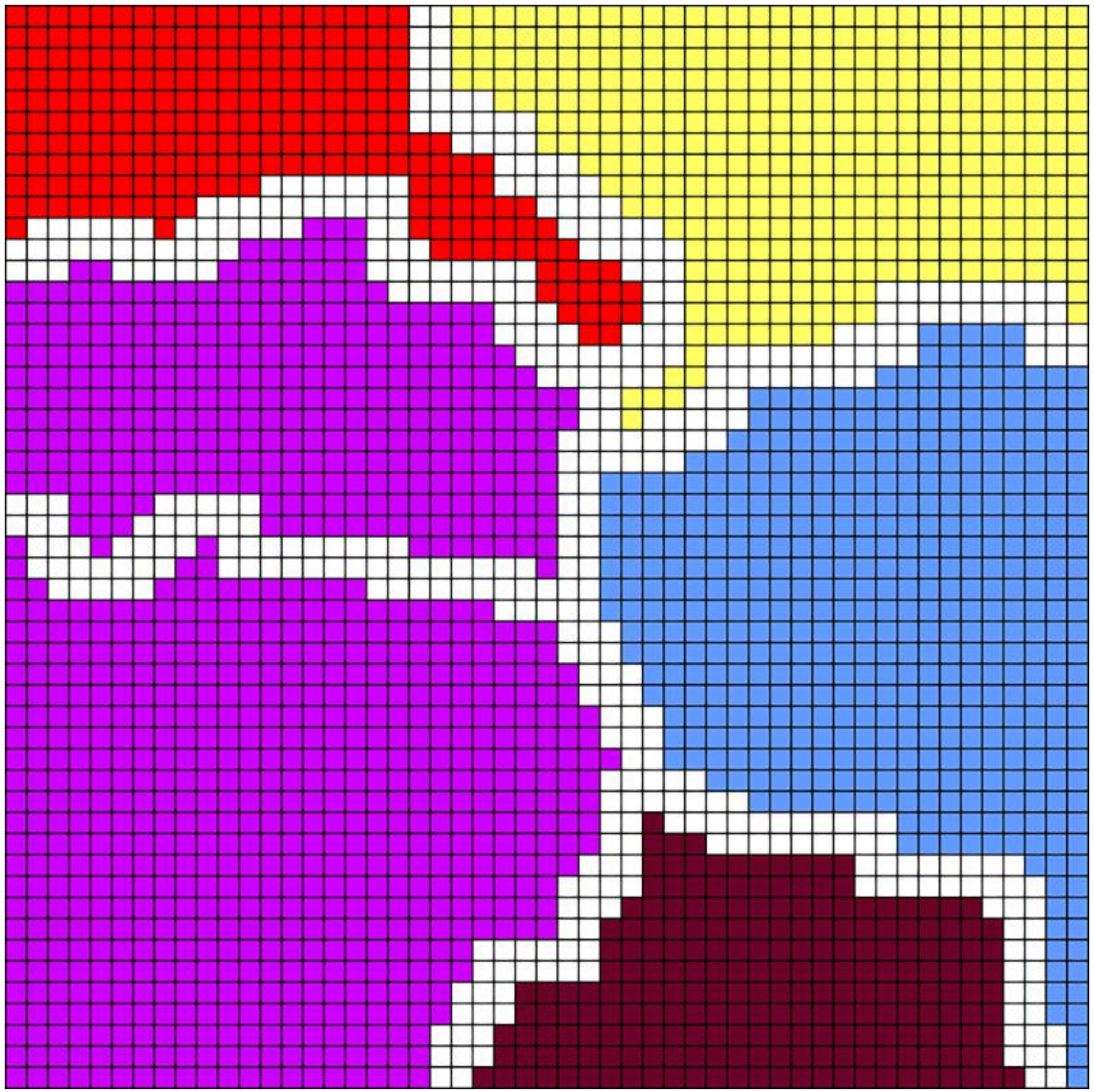
Neighbourhood operation: just changing the colour of a cell. This action can have as effect to:

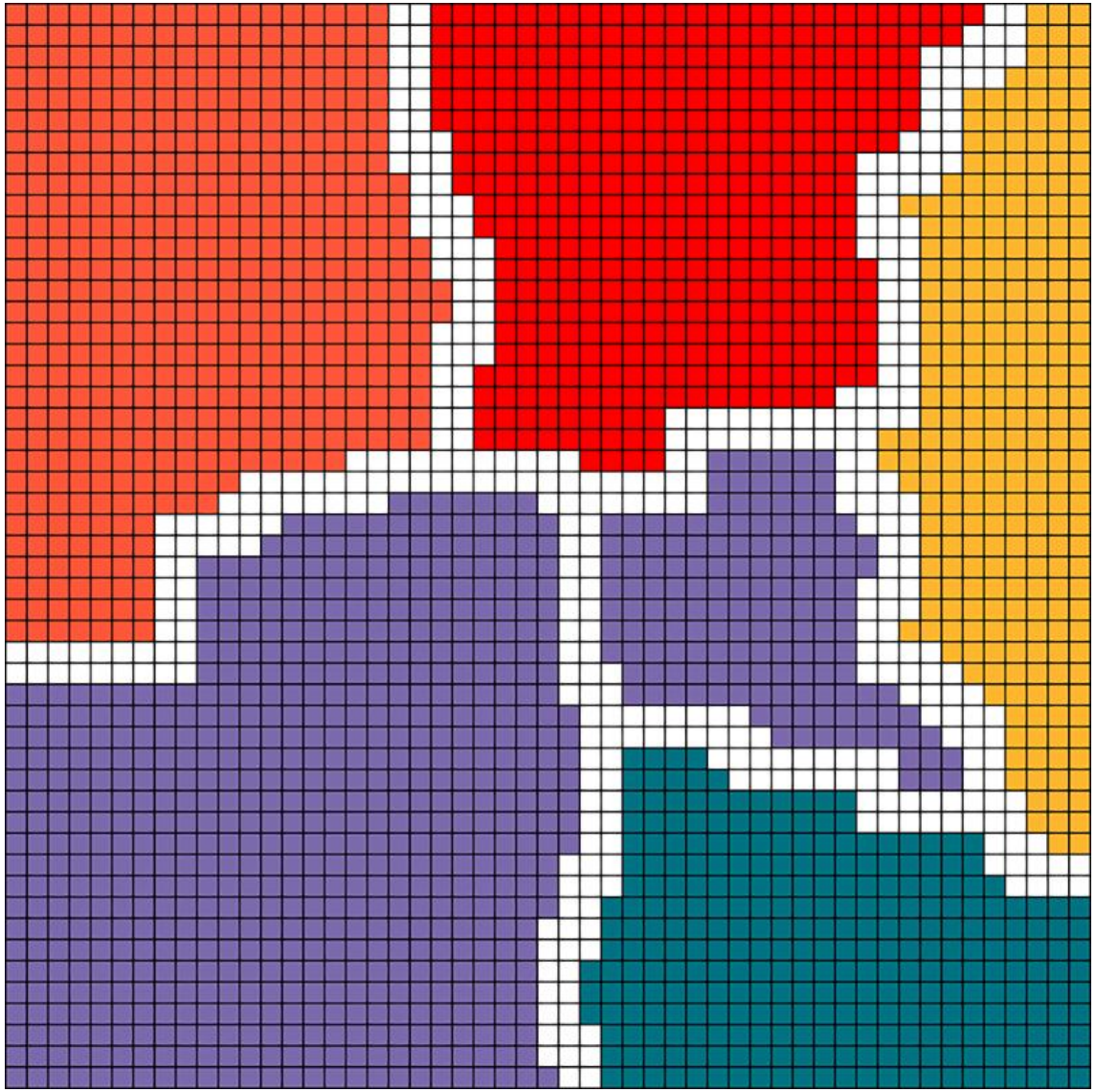
- create a new building,
- expand an existing building,
- simply change the colour of a building. Two sufficiently close buildings with the same colour can merge to form a bigger one.

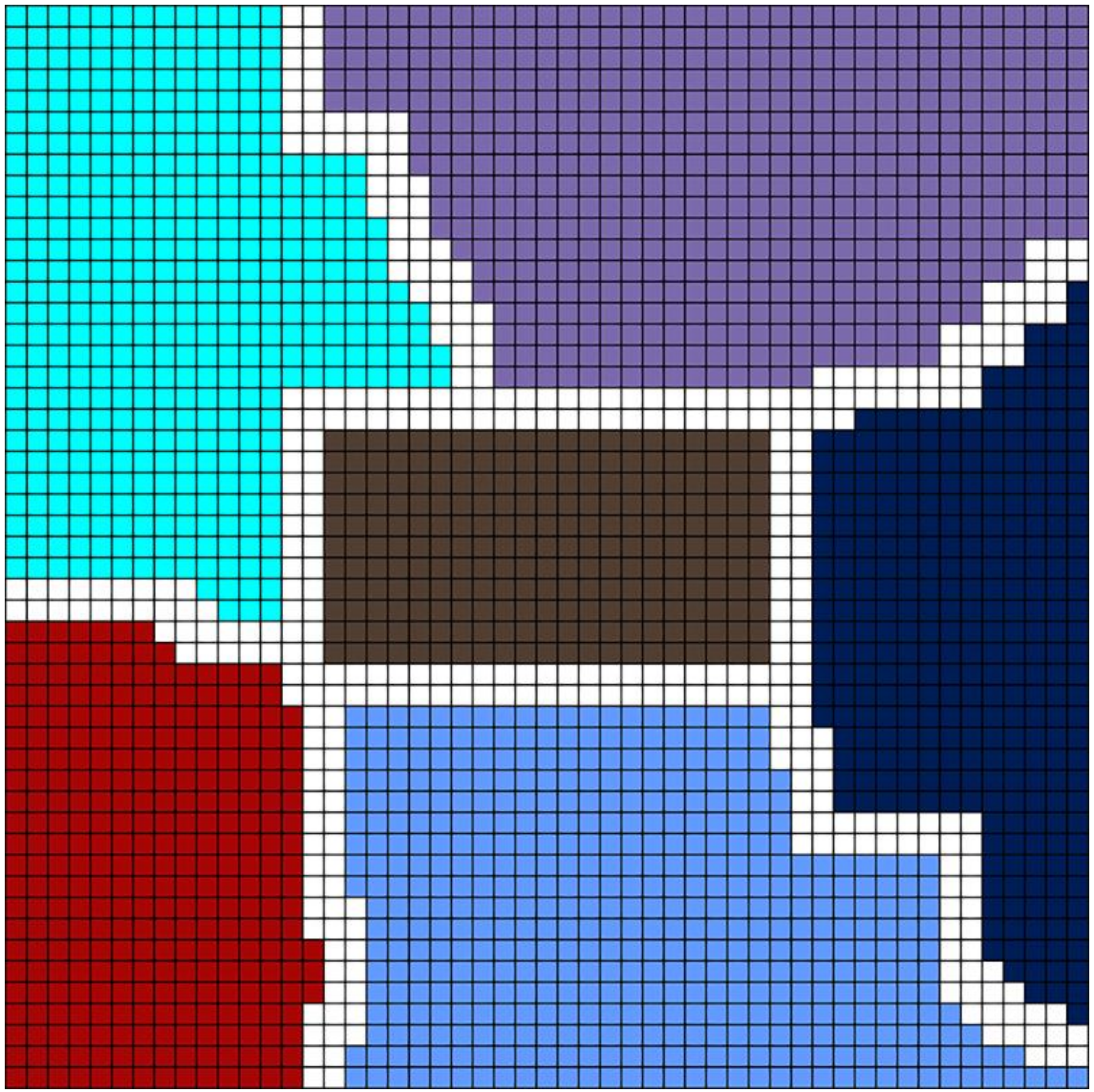
We have also implemented some *constraints* to guide the tool to more realistic solutions.

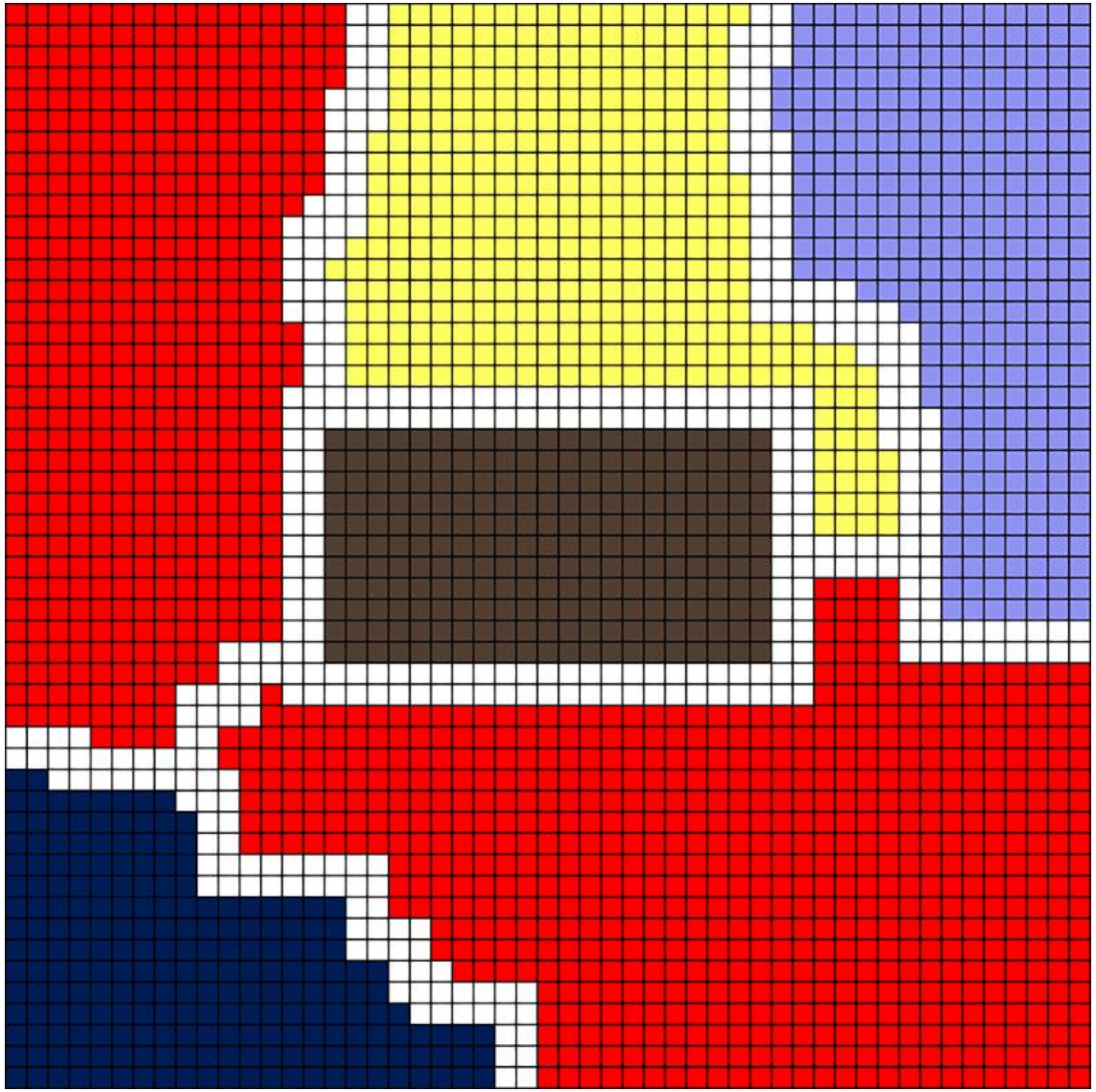
- A mandatory constraint that ensure a minimal distance between the facades of different buildings.
- Constraints that penalise solutions if the number of buildings, the total perimeter, the perimeter of each building or the area of each building doesn't belong to a certain range.

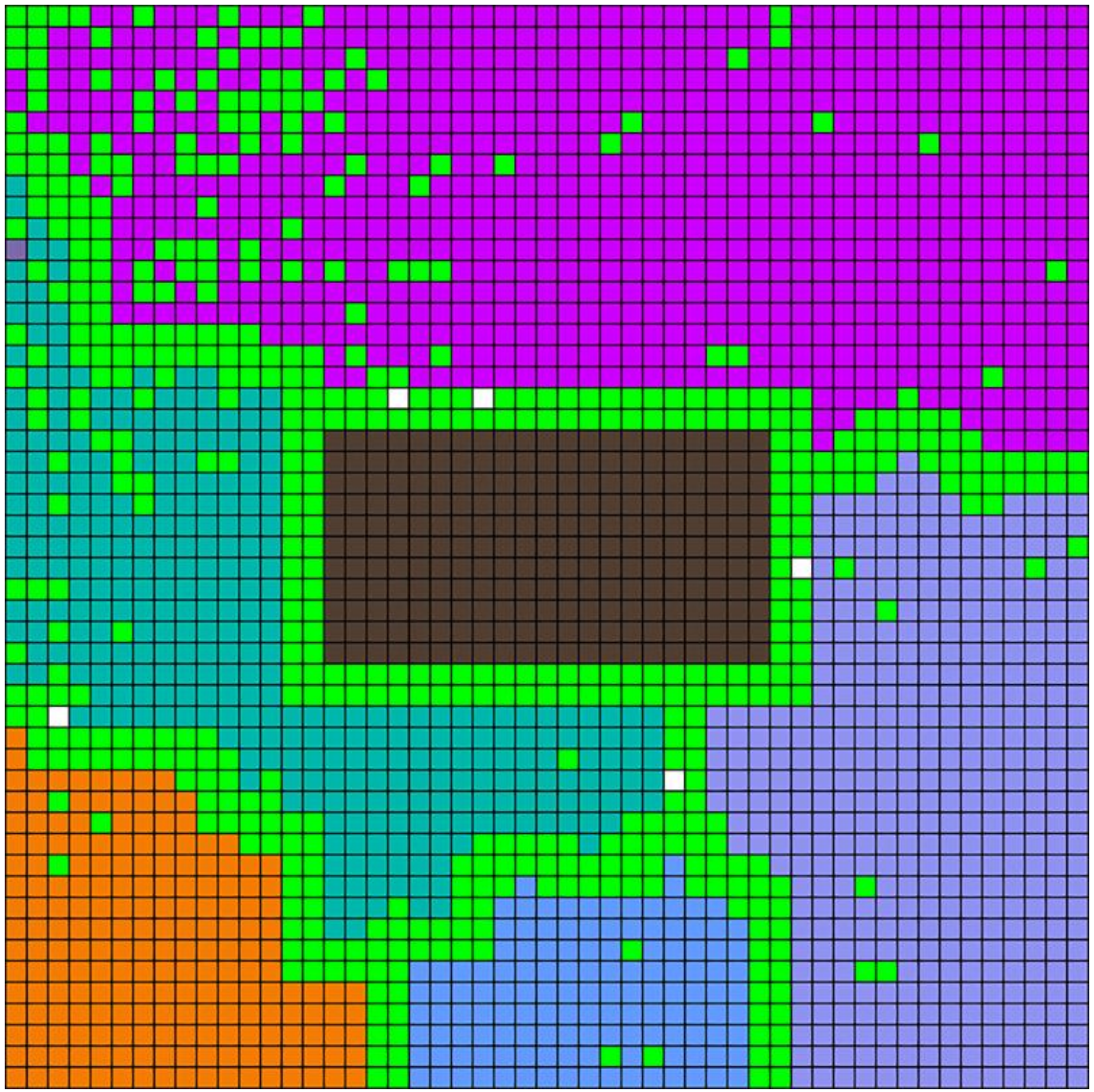
The following examples are results obtained by the program on an urban block of 102 meters by 102 meters tiled by squares with sides of 2 meters. The objective is minimize the porosity of the urban block. Furthermore, we add the following constraints: having between 6 and 10 buildings in our block, the buildings must have a perimeter between 75 and 500 meters and facades must be a 4 meters away from each other (or must have two empty squares between them).

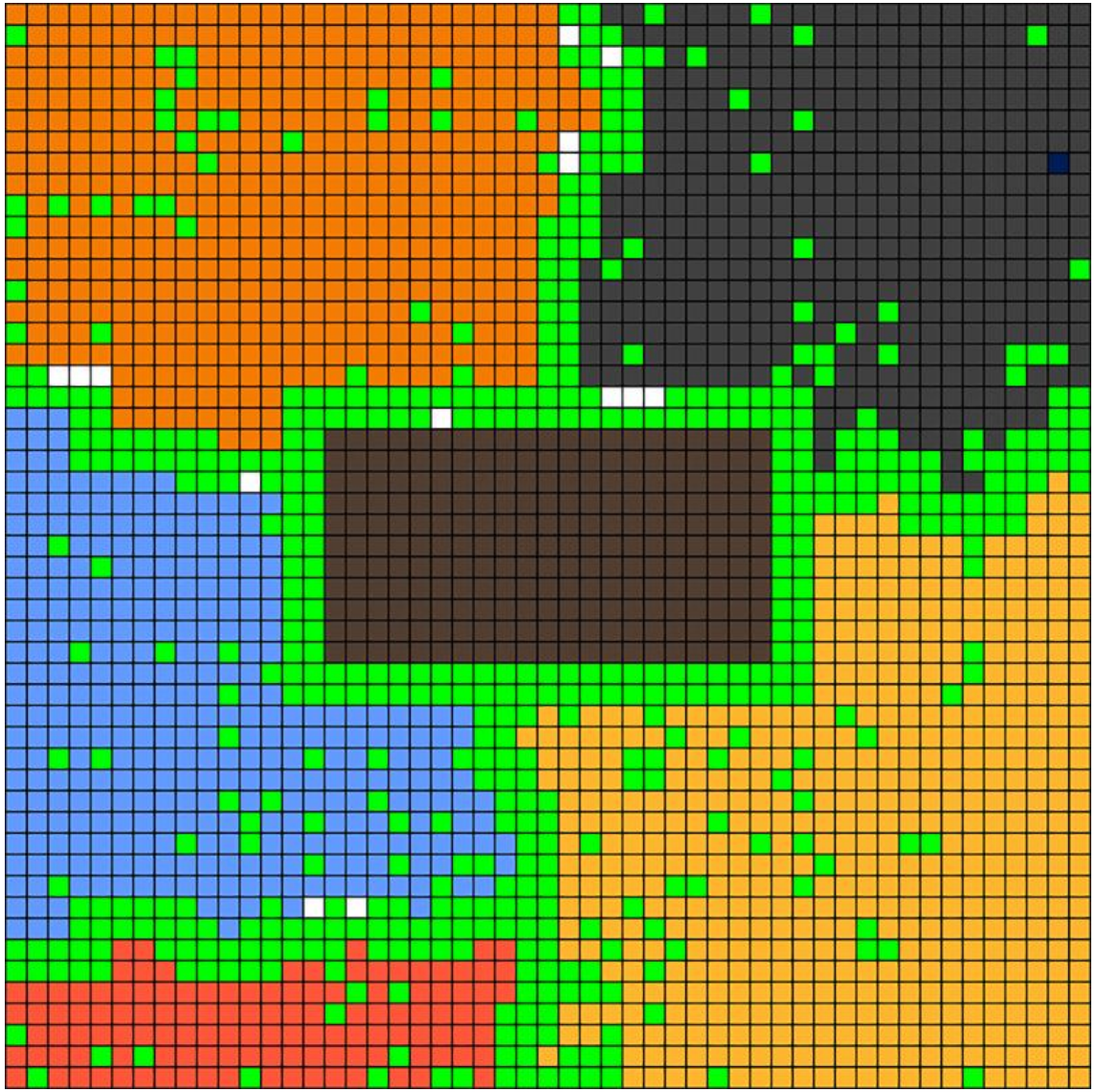


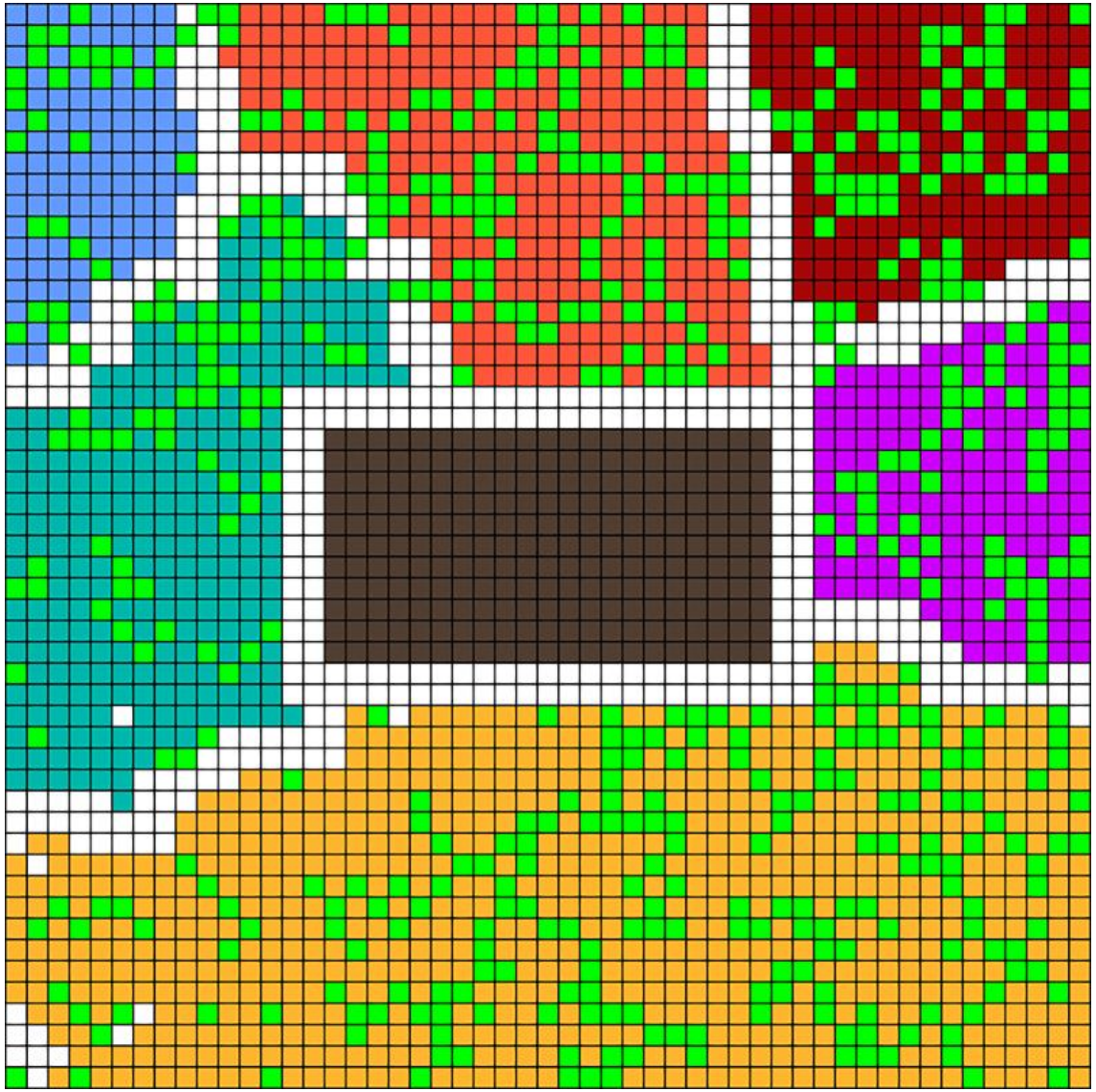












- Our tool to assist in the design of compact urban blocks is still in its beginning.

- Our tool to assist in the design of compact urban blocks is still in its beginning.
- The obtained results are promising but can be improved.

- Our tool to assist in the design of compact urban blocks is still in its beginning.
- The obtained results are promising but can be improved.
- Although we have neglected the notion of compactness in a first time, we are aware that the compactness is our main objective.

- Our tool to assist in the design of compact urban blocks is still in its beginning.
- The obtained results are promising but can be improved.
- Although we have neglected the notion of compactness in a first time, we are aware that the compactness is our main objective.
- The model is frequently enriched with new constraints and objectives to attain the compactness of the urban block.

- We plan to constraint the depth of a building.
  - How formally define the notion of depth of a more omplex polygonal building?
  - How calculate efficiently the depth?

- We plan to constraint the depth of a building.
  - How formally define the notion of depth of a more complex polygonal building?
  - How calculate efficiently the depth?

- The program currently tries to minimize an objective and to be penalised the least possible.
  - To do that, the program sorts the constraints and the objective. The bigger is the penalty, the more “important” the constraint is.
  - We would like to set the constraints and the objective on equal footing and have a multi-objective model.
  - An idea to do that is to use the mathematical models from the game theory.