University of Antwerp
**Operations Research** Group
ANT/OR

# An Iterated Local Search for the Vehicle Routing Problem with Backhauls

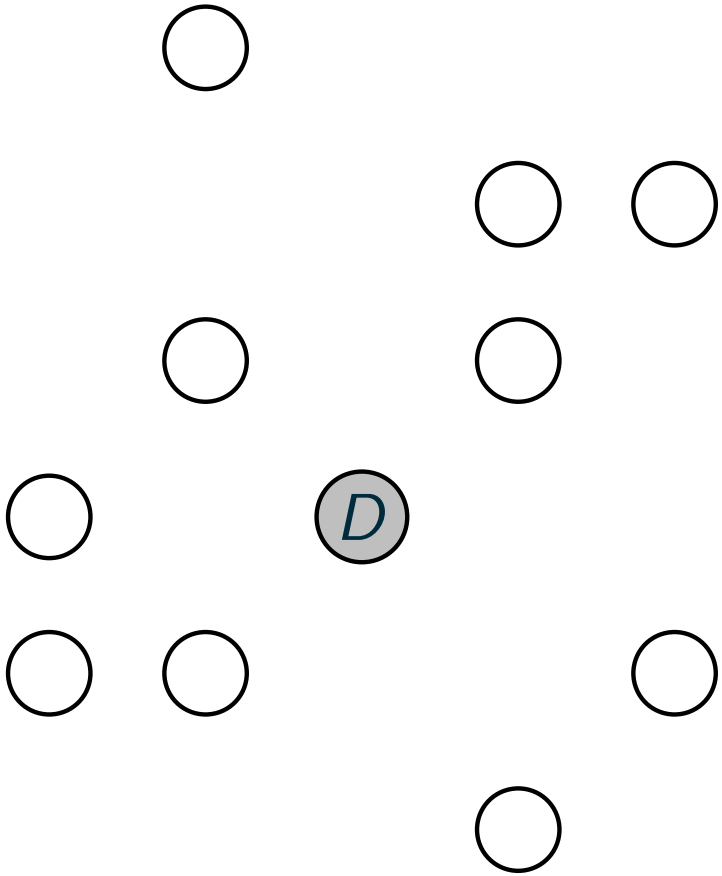
Daniel Palhazi Cuervo

Doctoral Day - Antwerp, November 23rd 2011

# Outline

- The Vehicle Routing Problem (VRP)
- The Vehicle Routing Problem with Backhauls (VRPB)
- Heuristic algorithms - Metaheuristics
  - Local Search (LS)
  - Iterated Local Search (ILS)
- Implementation details
- Computational experiments
- ILS vs. The state of the art algorithms
- Conclusions

# Vehicle Routing Problem

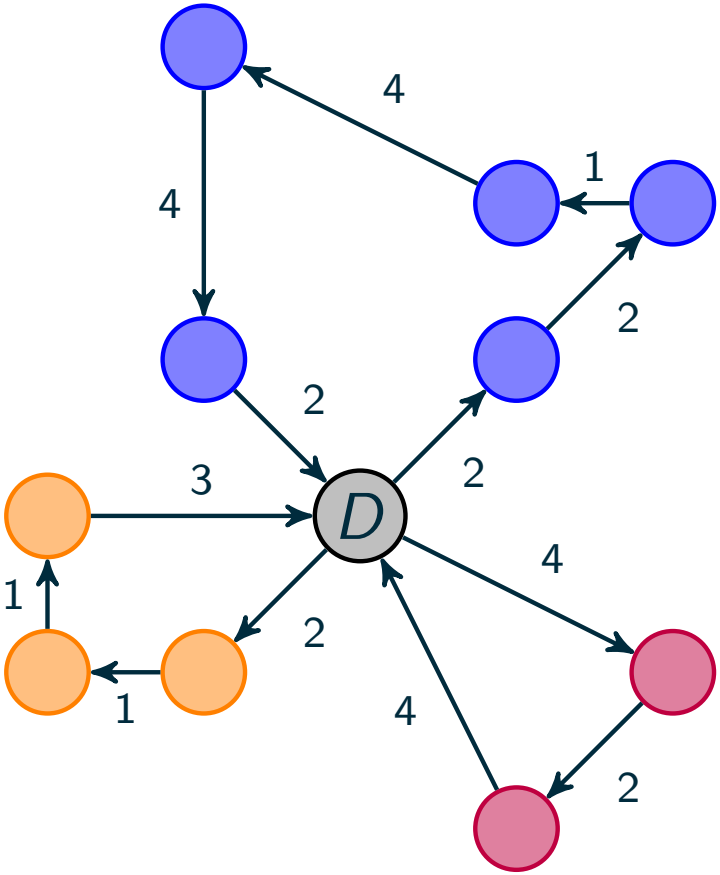



**Elements:**

- A central **depot**
- A set of **customers** that request goods from the depot
- A set of identical **vehicles** with a fixed capacity

**Idea:** Define the route for each vehicle in order to serve every customer

# Vehicle Routing Problem

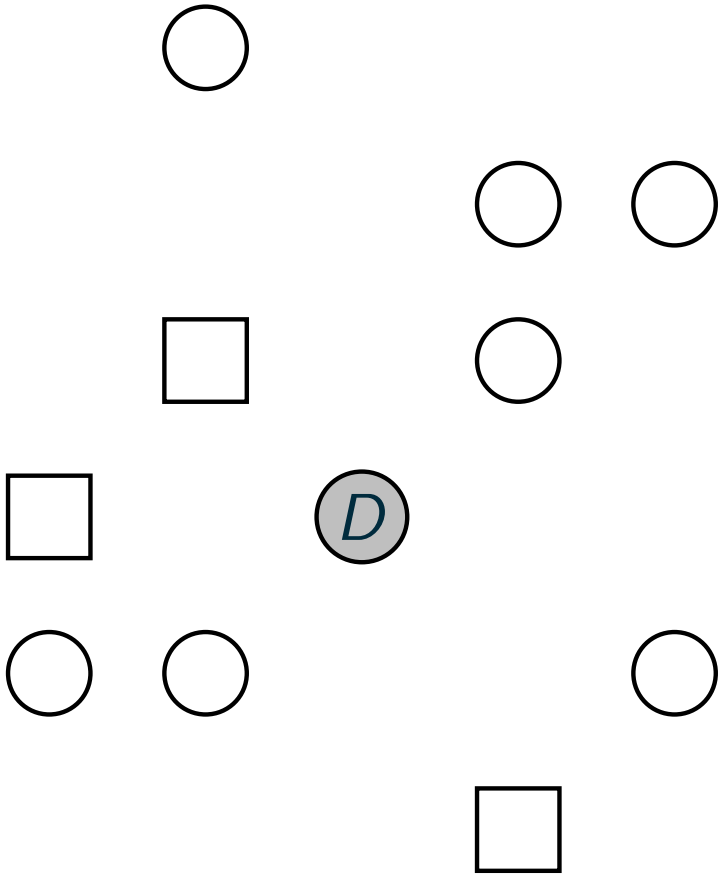



**Constraints:**

- Every route starts and finishes at the depot
- Every customer is visited exactly one time
- The amount of goods transported in each route cannot exceed the vehicle capacity

**Objective:** Minimize the distance traveled by the vehicles

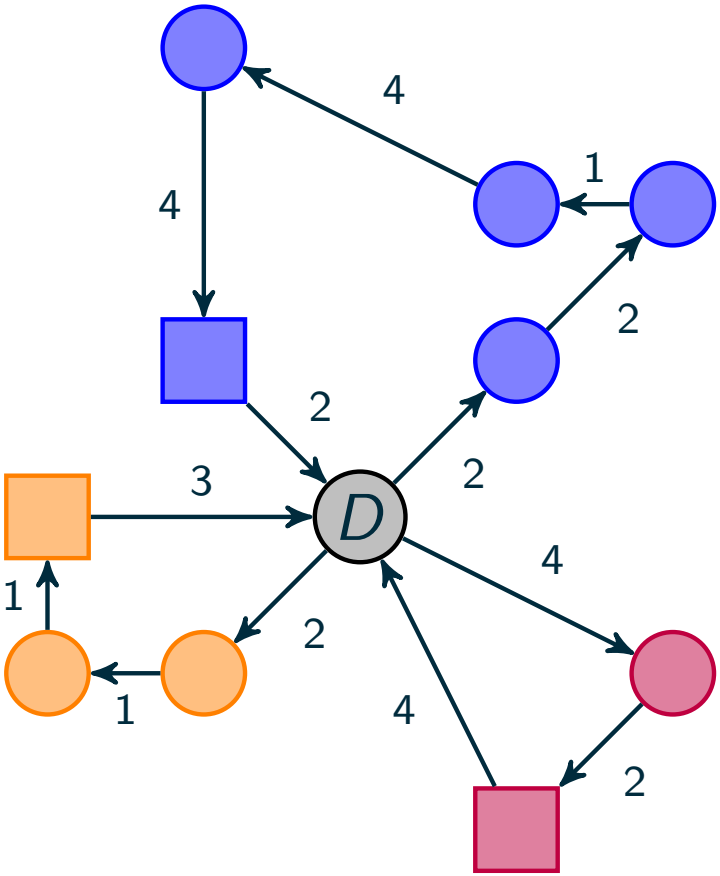
# Vehicle Routing Problem with Backhauls




**Two types of customers:**

- **Consumers (linehauls):** request goods from the depot
- **Suppliers (backhauls):** send goods back to the depot

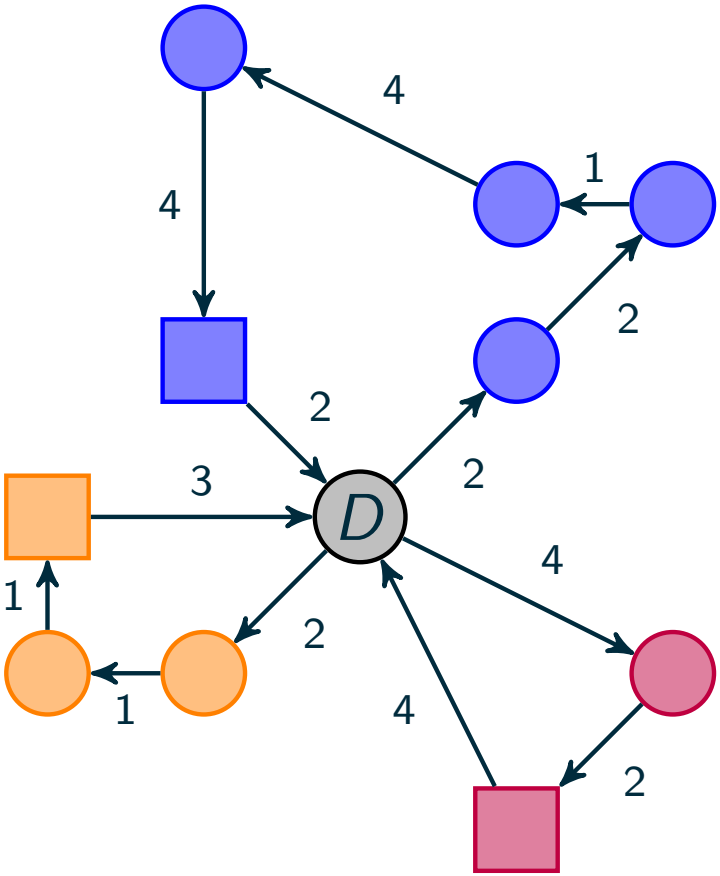
# Vehicle Routing Problem with Backhauls




**Constraints for every route:**

- The consumers must be served before the suppliers
- The load of goods sent to the *consumers* and the load of goods received from the *suppliers* must not separately exceed the capacity of the vehicle

# Vehicle Routing Problem with Backhauls




## Motivation:

- **Main idea:** take advantage of the available capacity in the vehicles during the trip back to the depot
- **Principal assumption:** the rearrangement of the load in the delivery points is not possible or is too expensive

# Heuristic algorithms - Metaheuristics

- **Limitation of exact algorithms:** cannot be used to solve real-life problems (too long execution times)
- **Alternative:** Heuristic algorithms (Metaheuristics)
- **Feature:** Obtain very good solutions in short execution times
- **Drawback:** The obtained solution cannot be proven to be optimal

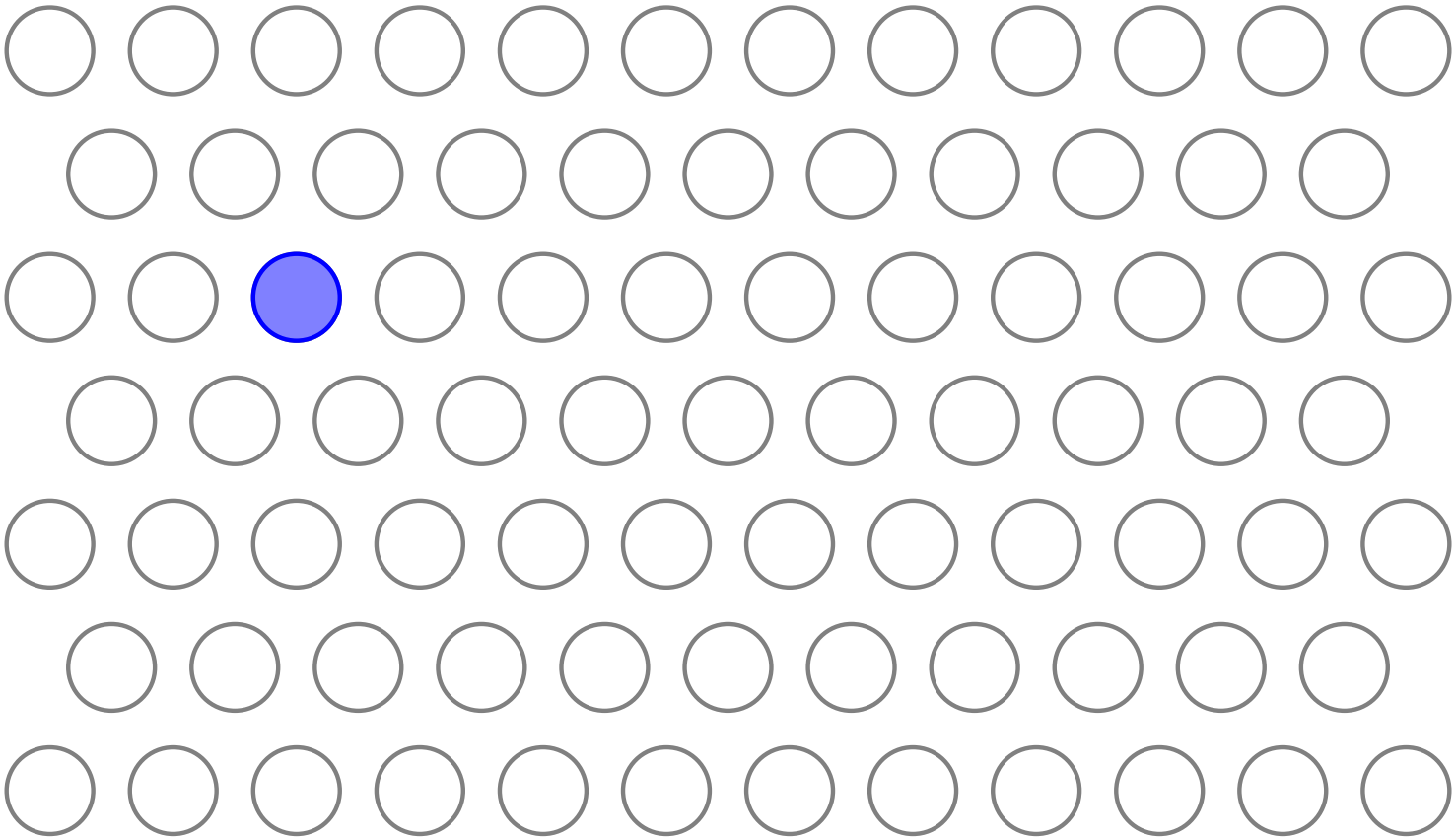
# Local Search

- Iteratively explores adjacent portions of the solution space

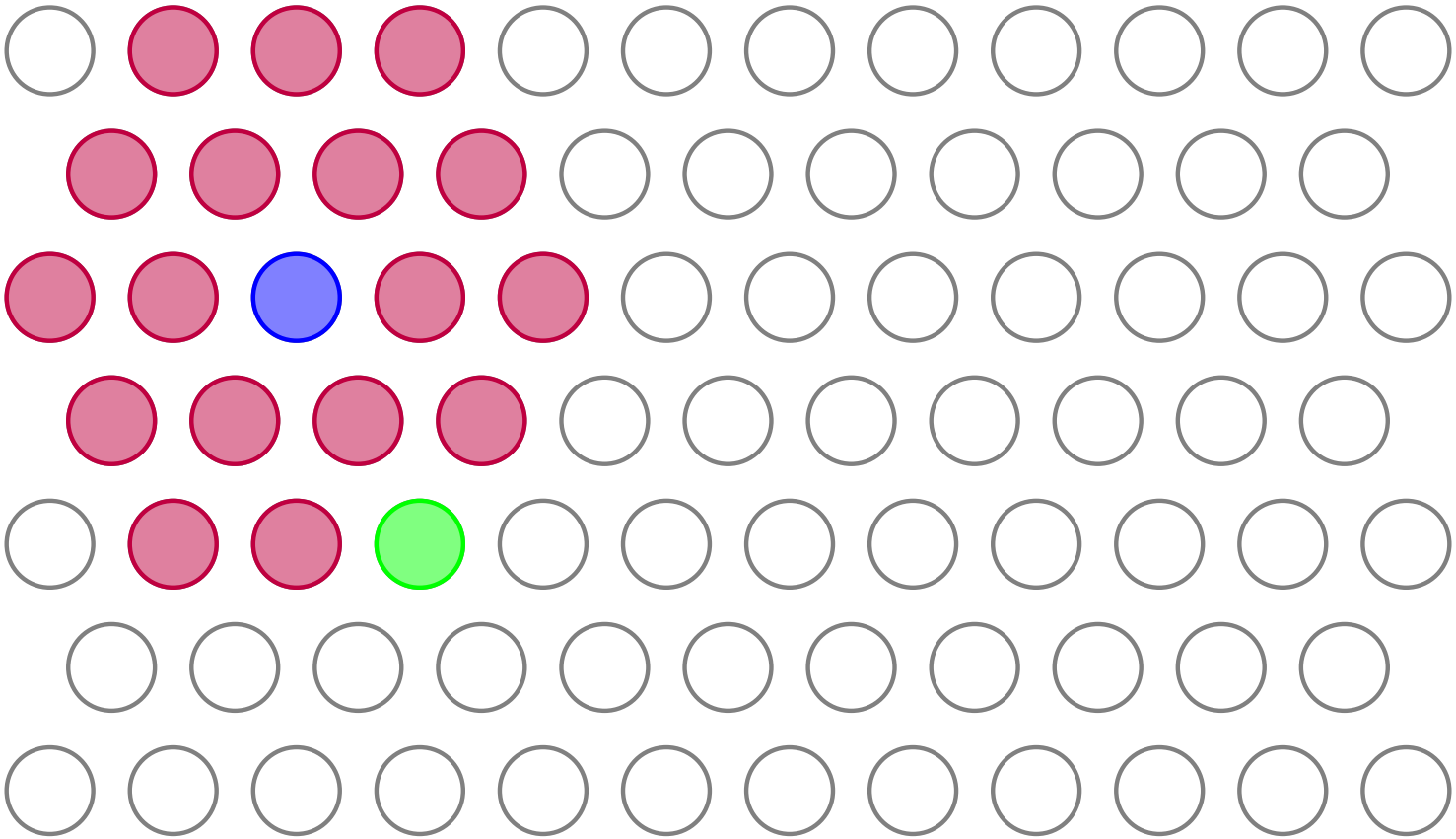
**Pseudo code:**

1. Build an initial solution
2. Generate a set of neighboring solutions as a result of applying different local changes to the current solution
3. Move to a neighbor solution with a better value of the cost function and continue the exploration

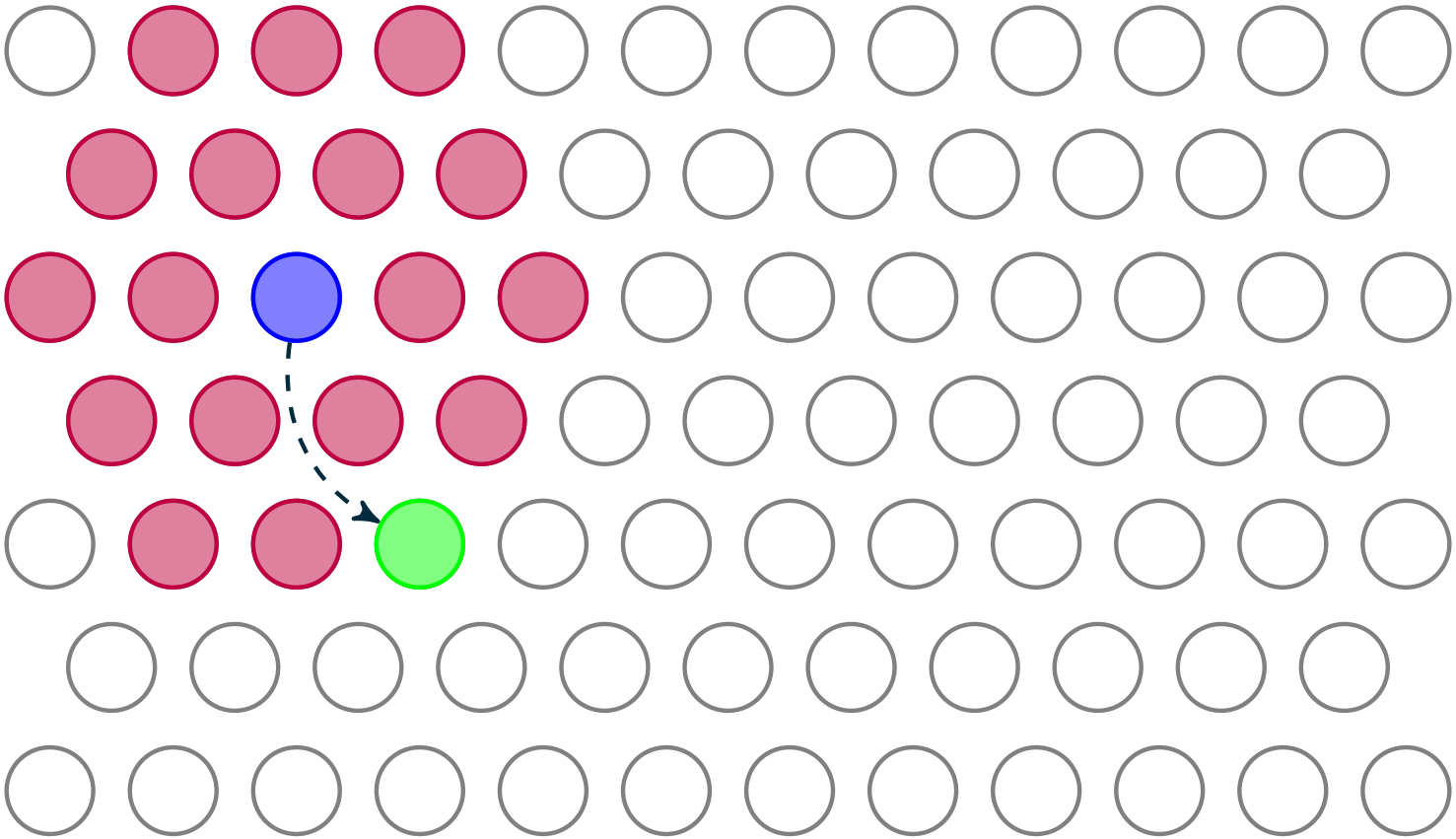
# Local Search



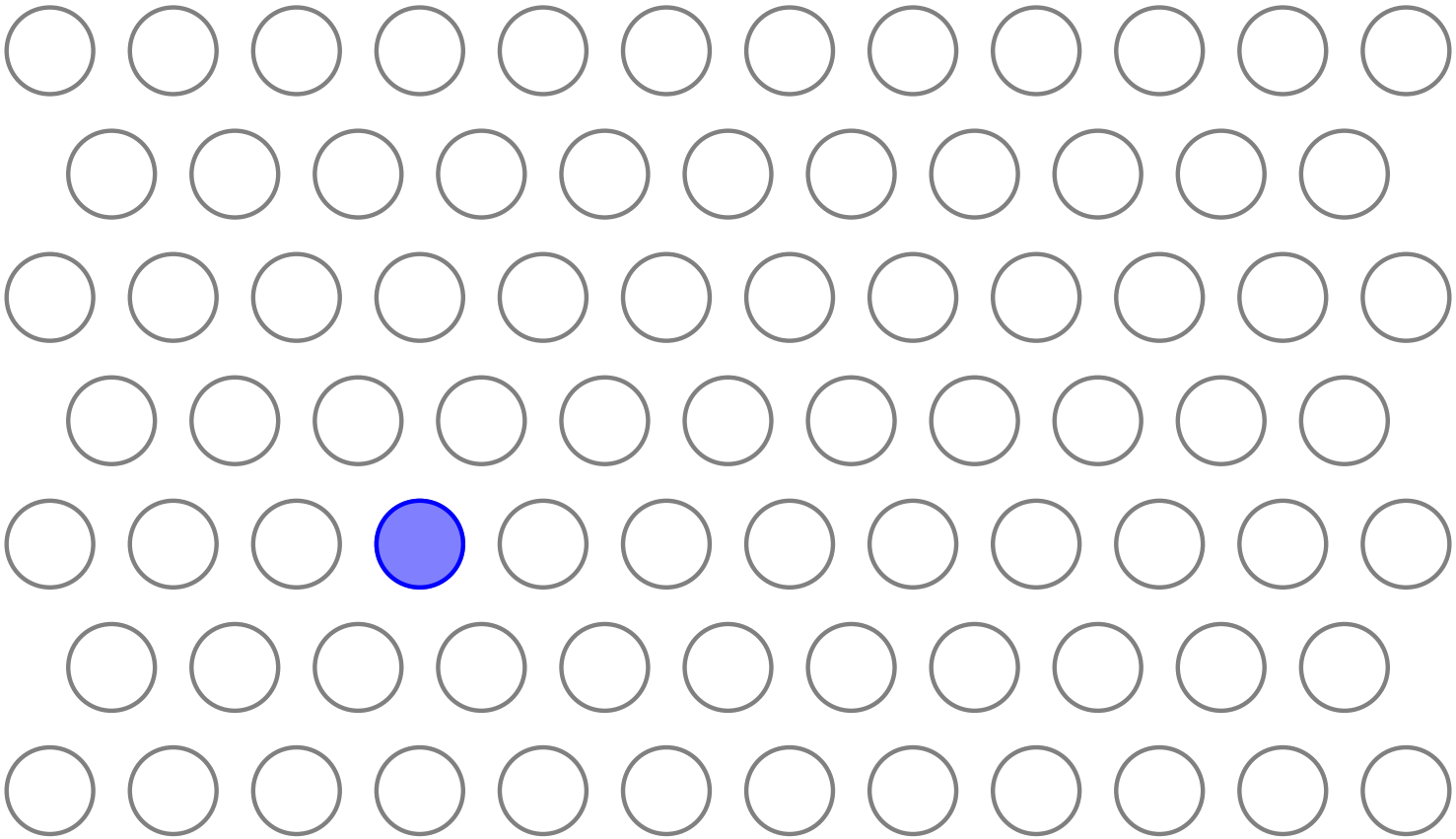
# Local Search



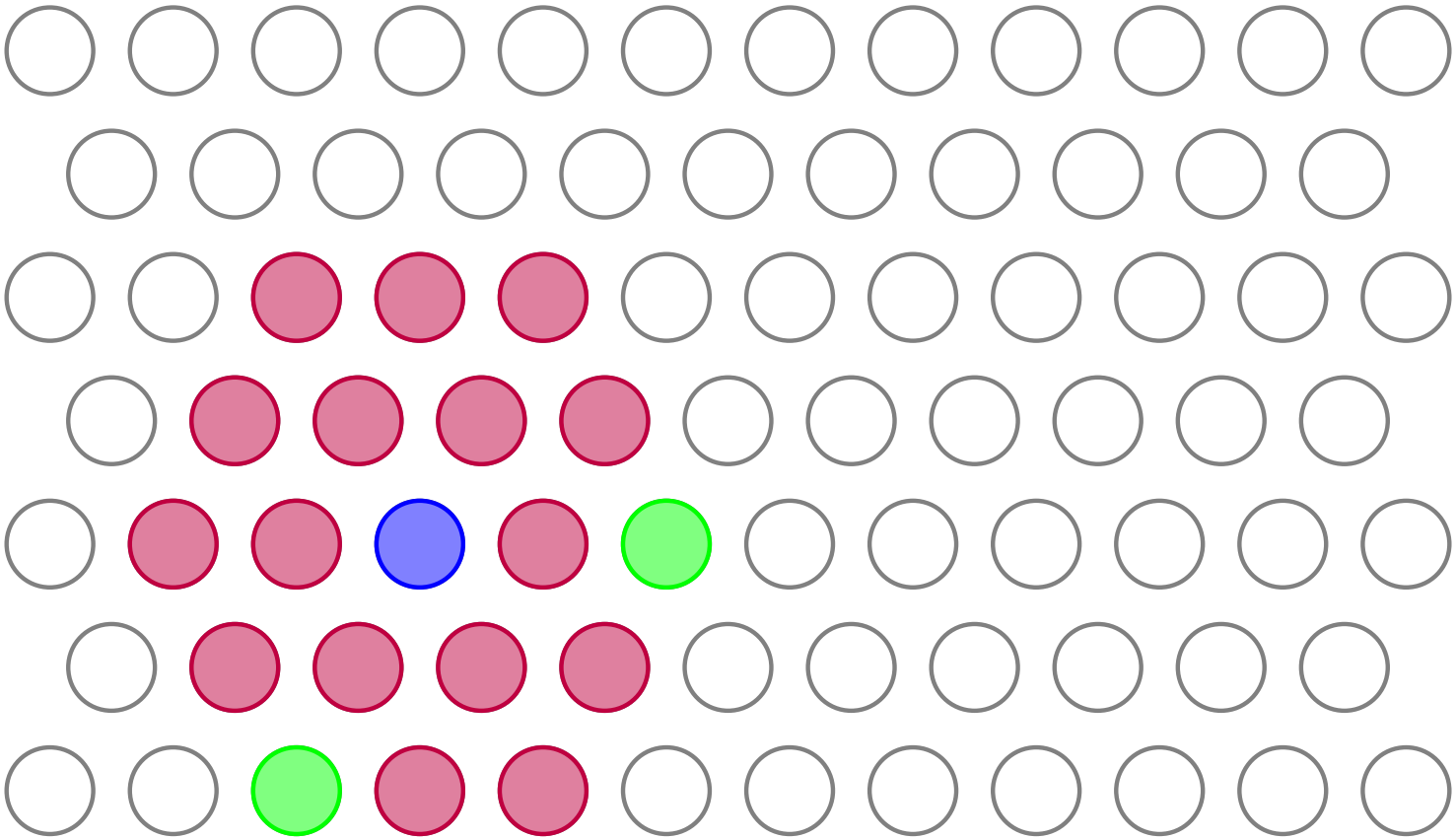
# Local Search



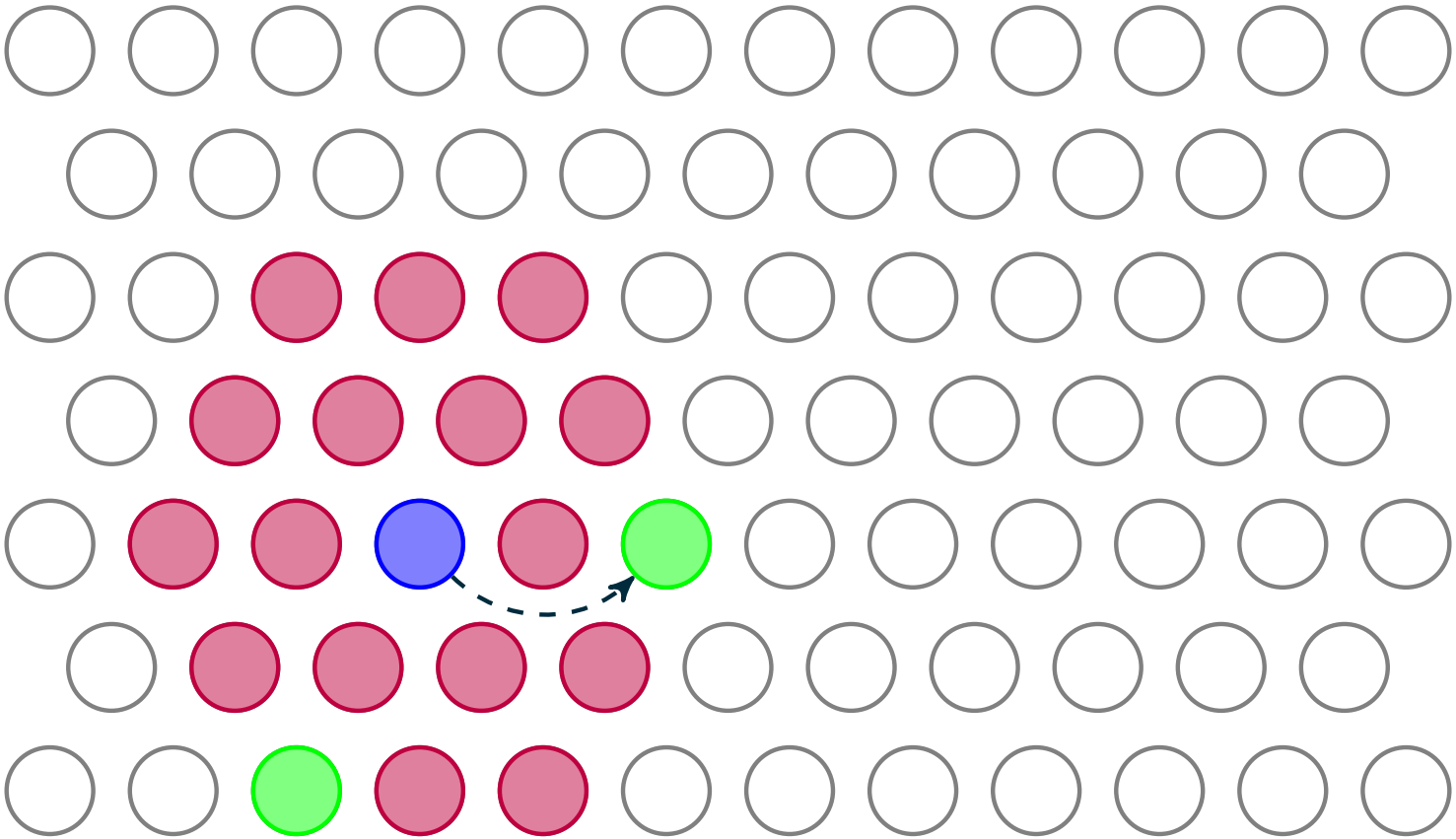
# Local Search



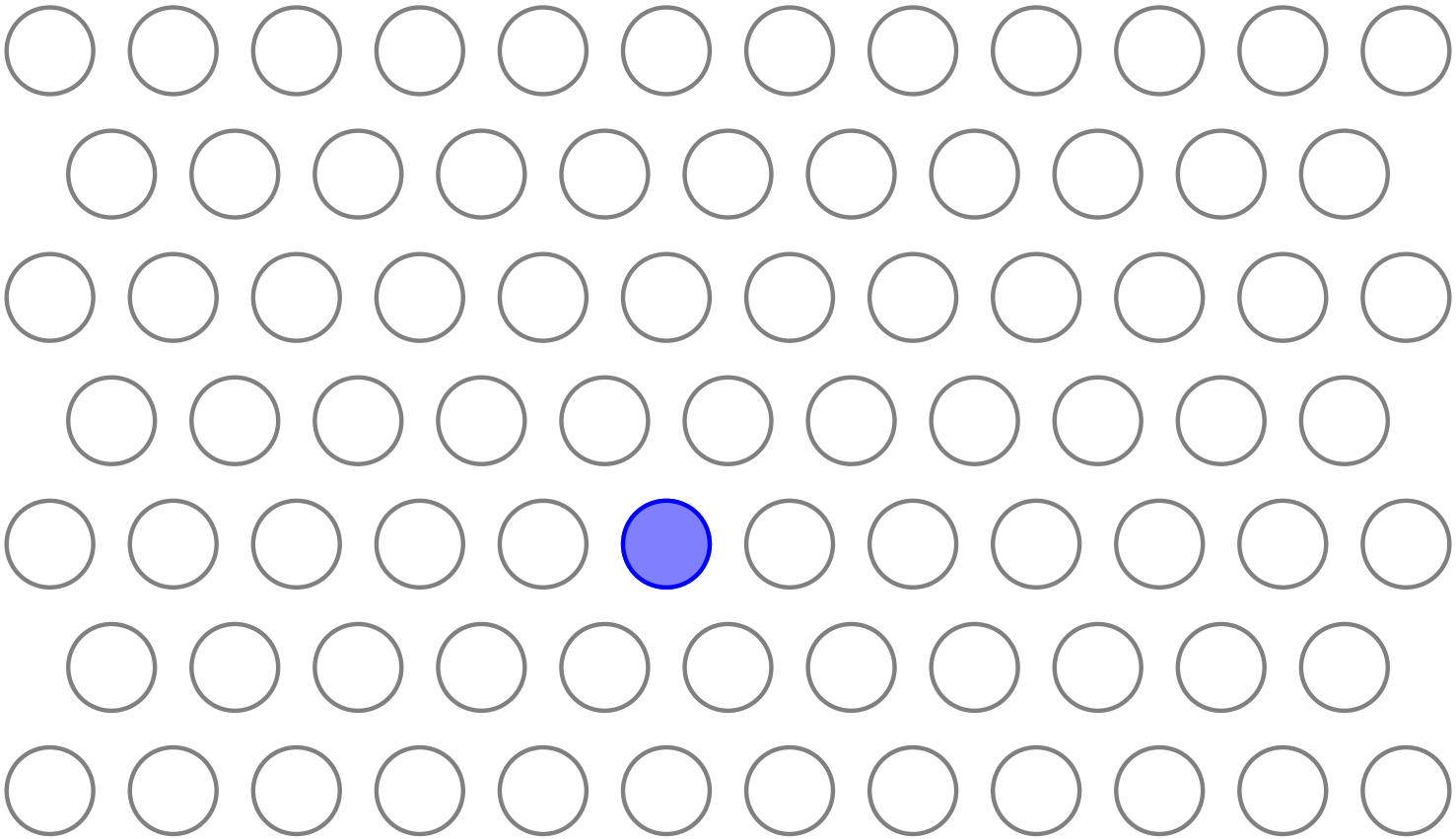
# Local Search



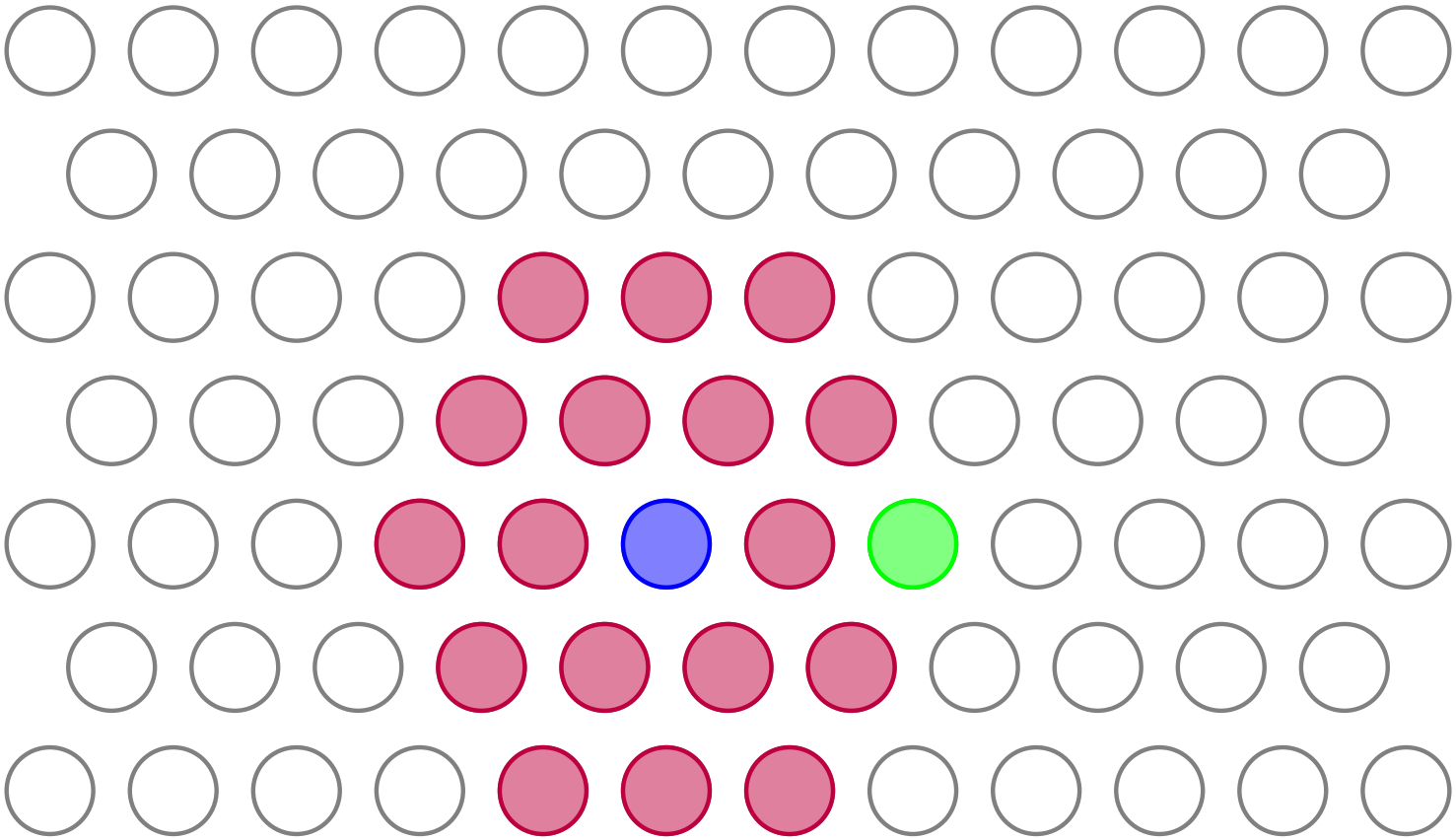
# Local Search



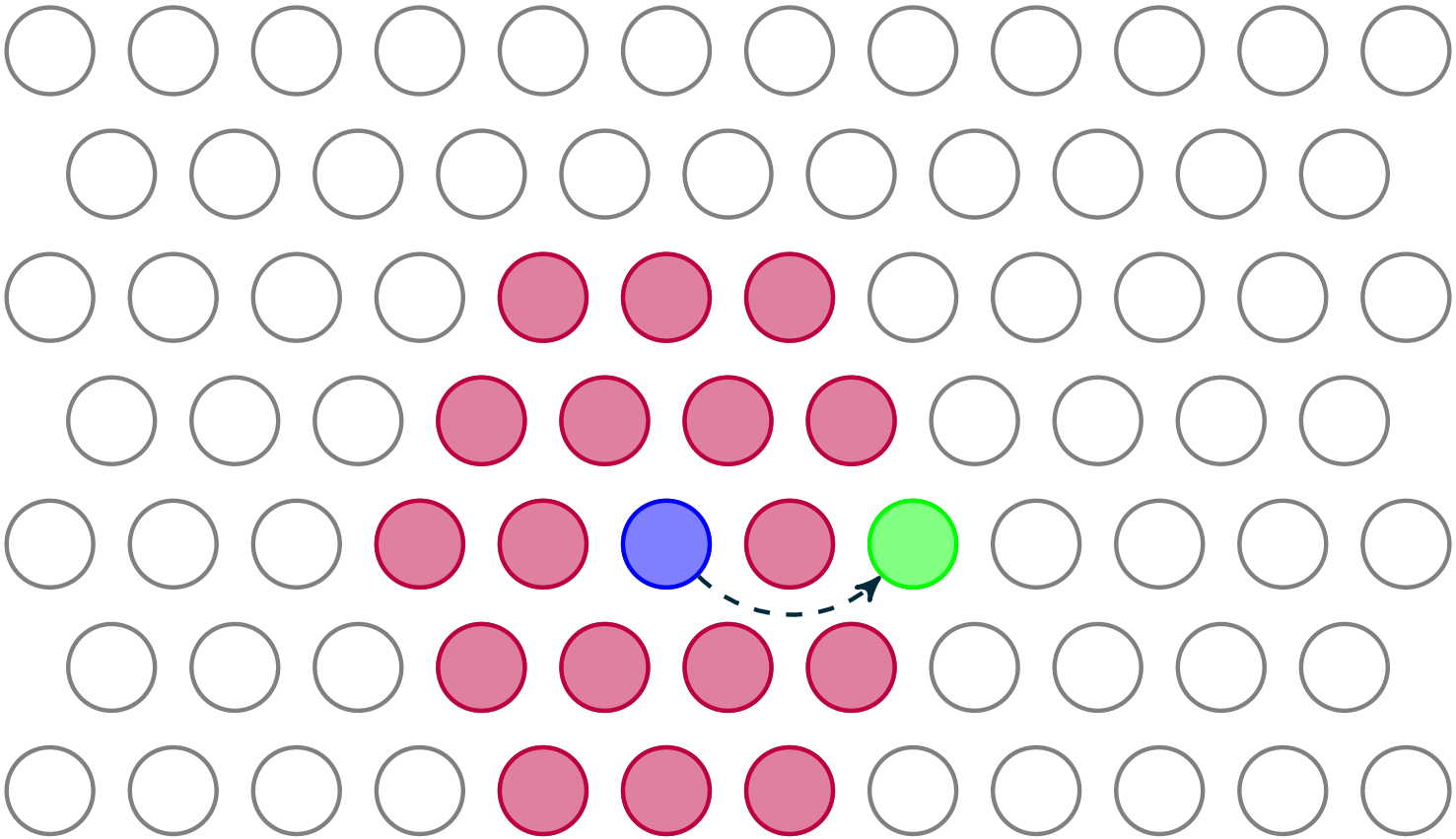
# Local Search



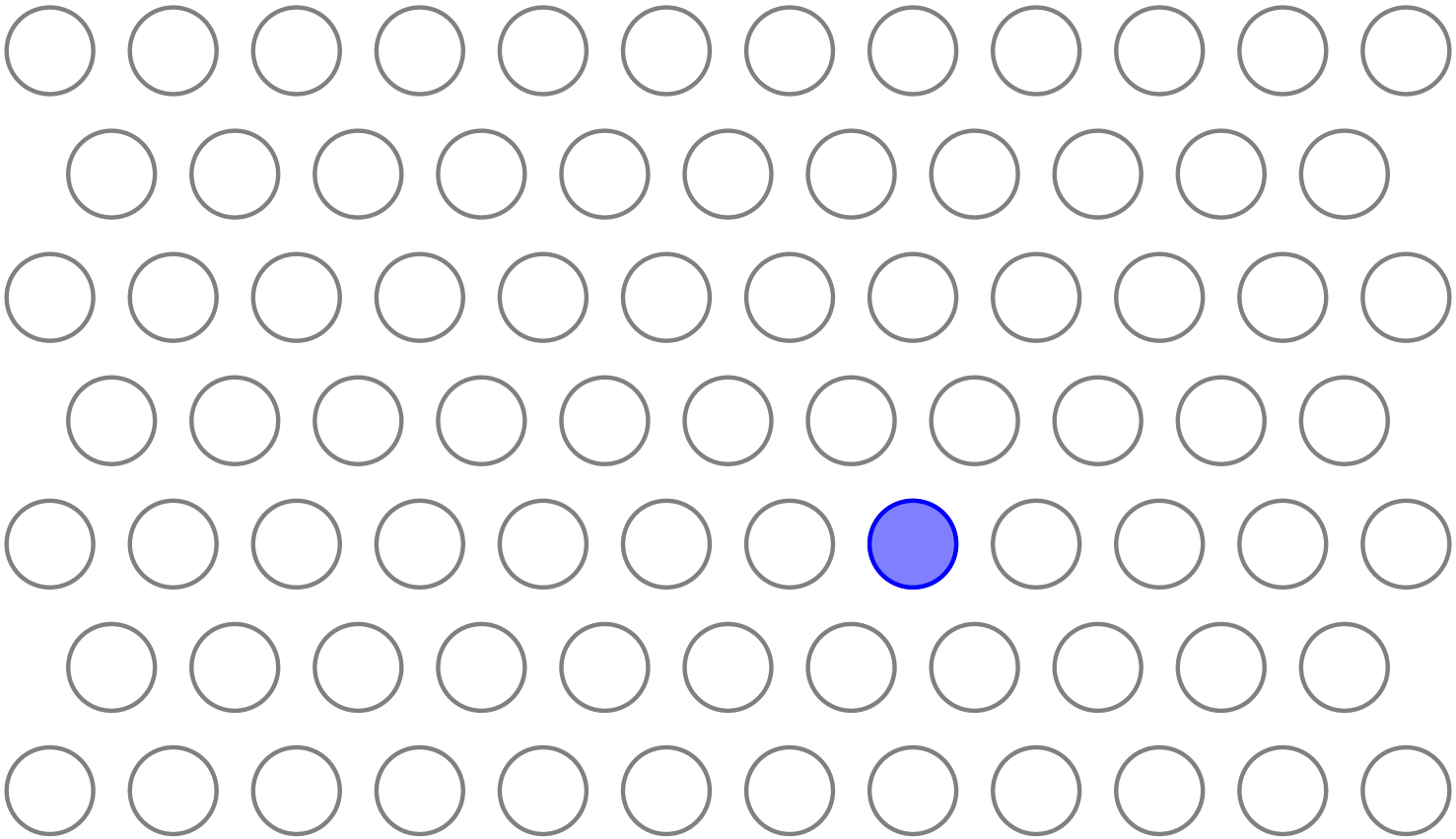
# Local Search



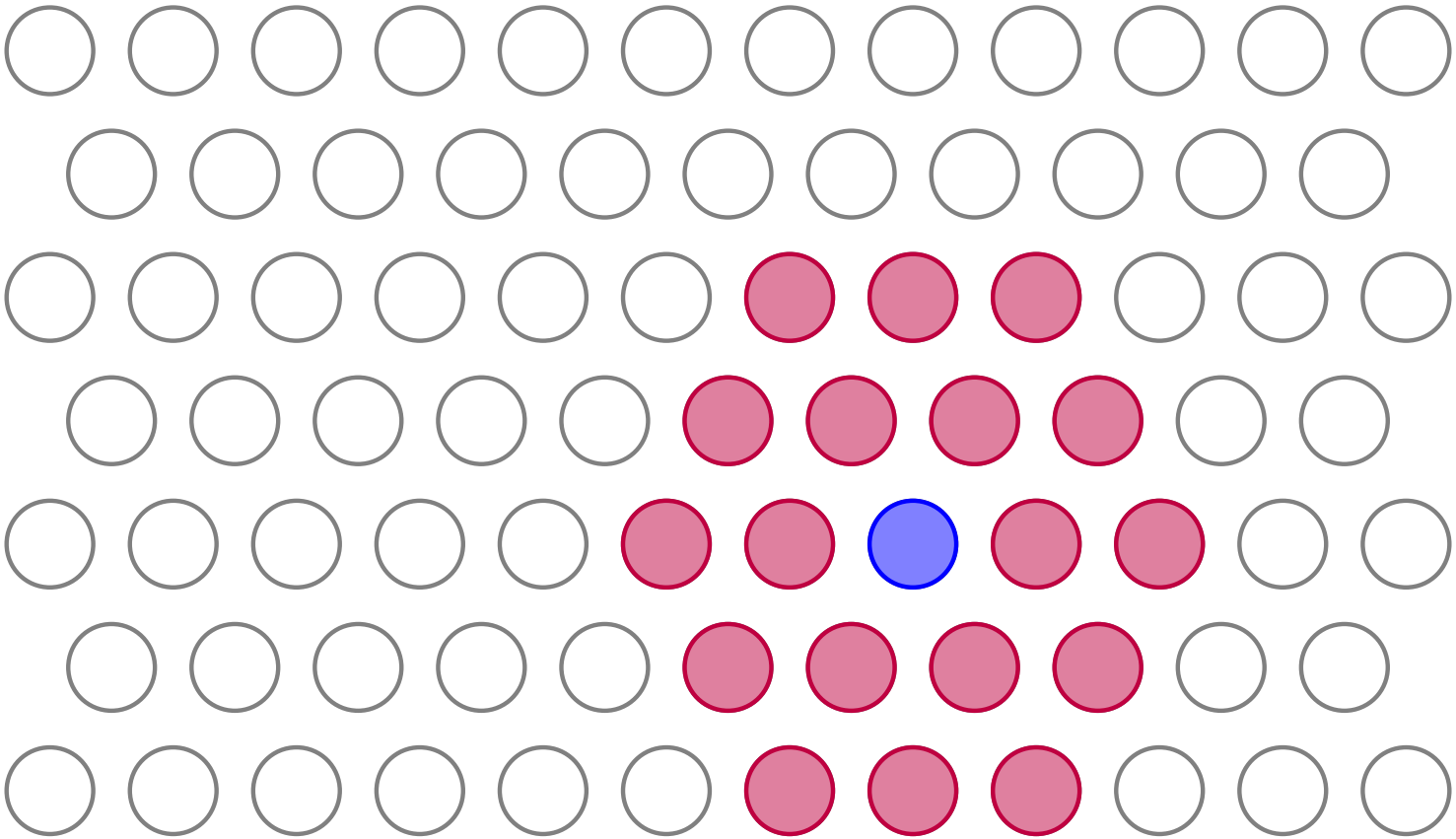
# Local Search



# Local Search



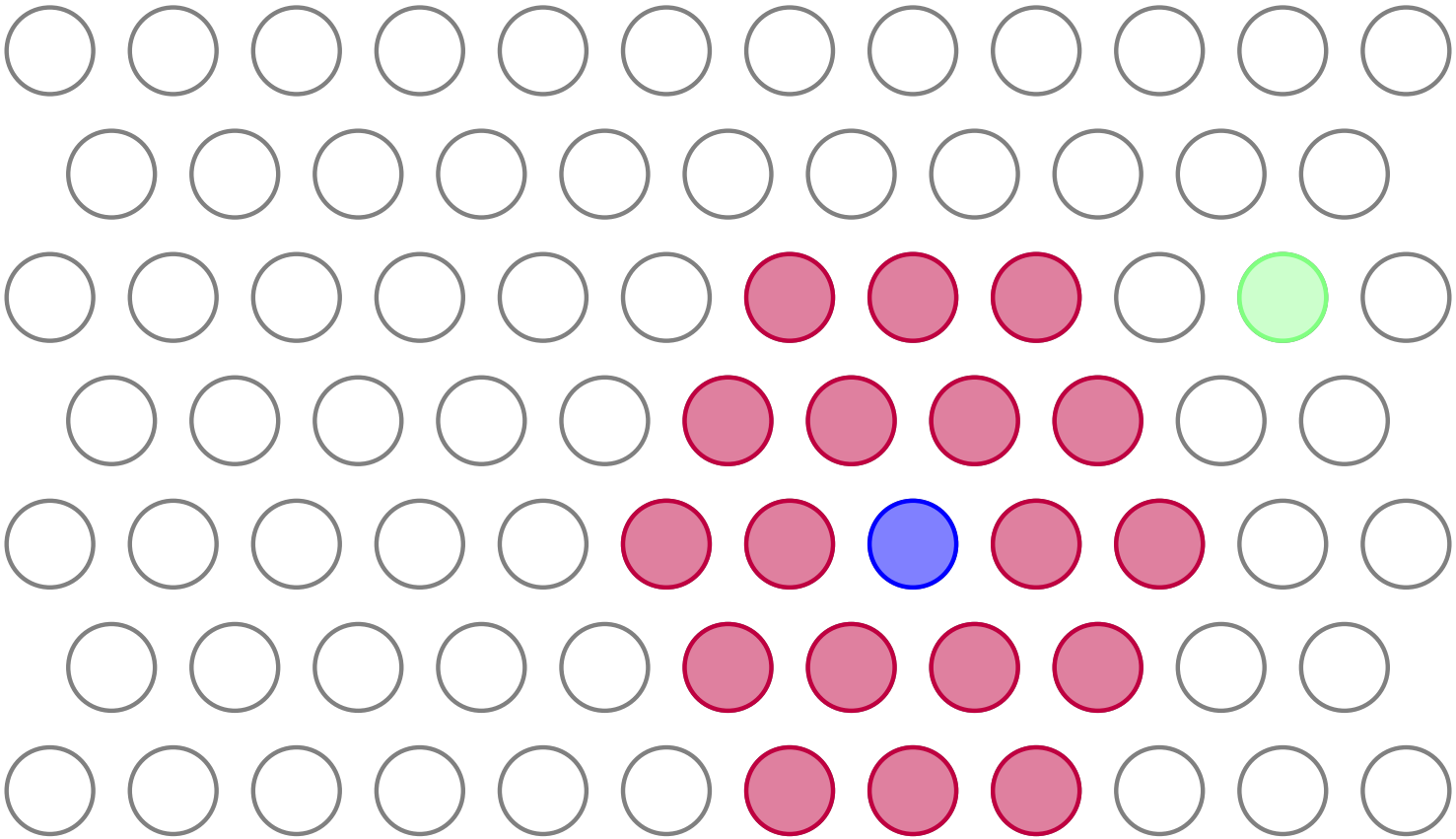
# Local Search



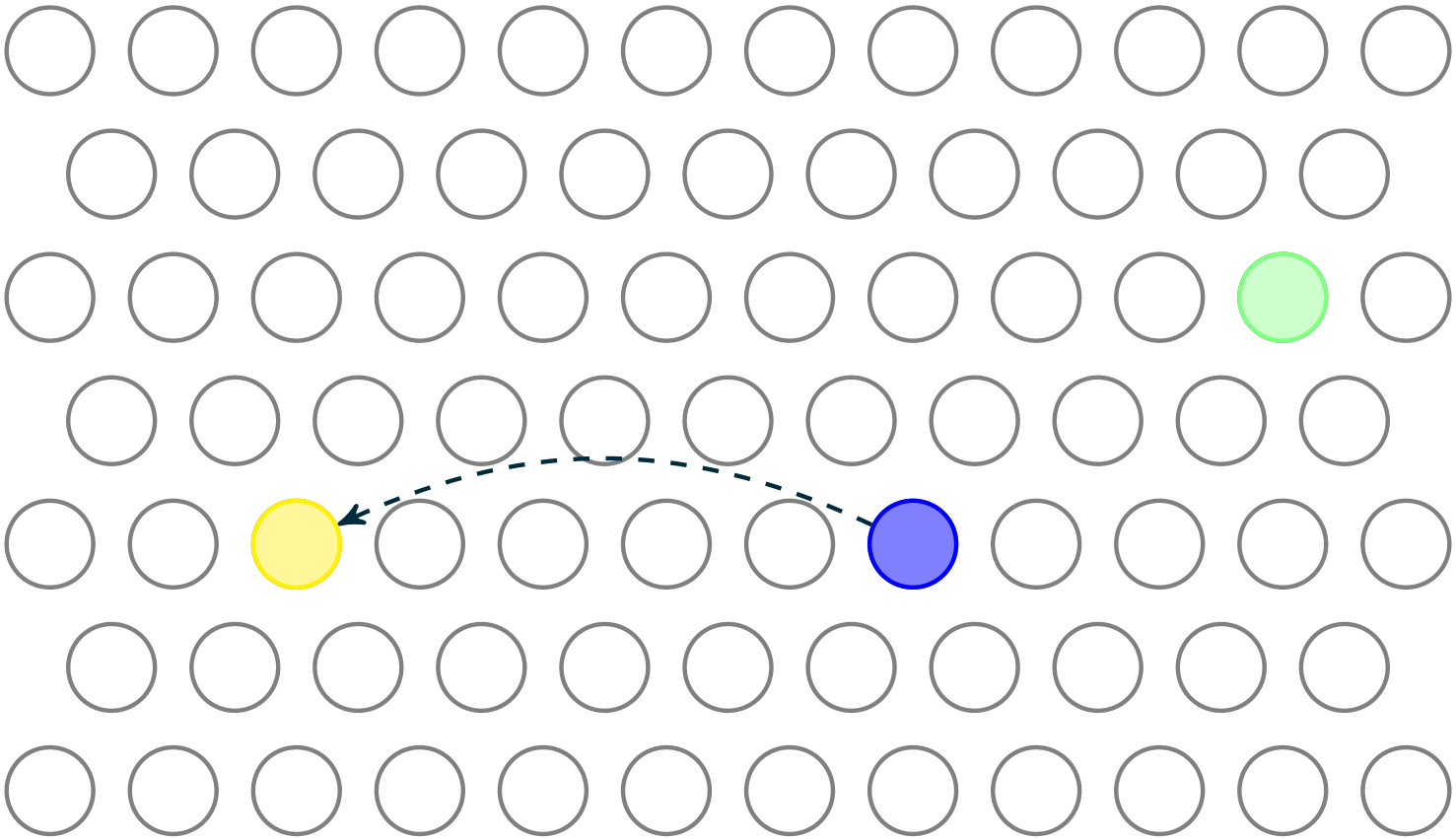
# Iterated Local Search

- **Main drawback of Local Search:** It gets easily stuck in local optimum solutions
- **Intuitive strategy:** apply Local Search several times starting from different initial solutions (random sampling)
- **Assumption:** Better solutions are probably found in the surrounding space of good solutions, but they are not reachable by the local search.
- **Better strategy:** when a local optimum solution is found, randomly apply a perturbation operator and apply Local Search.

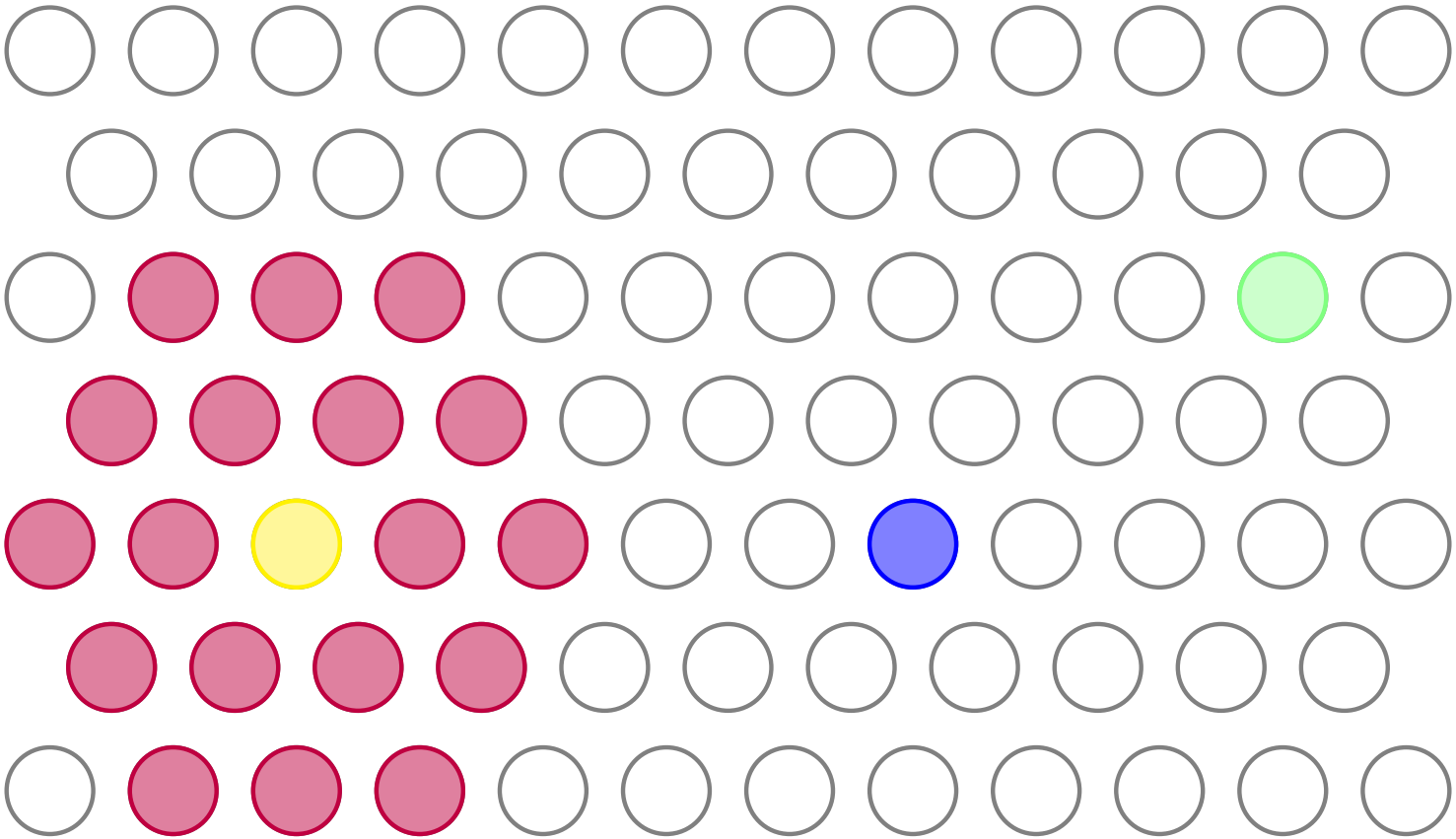
# Iterated Local Search



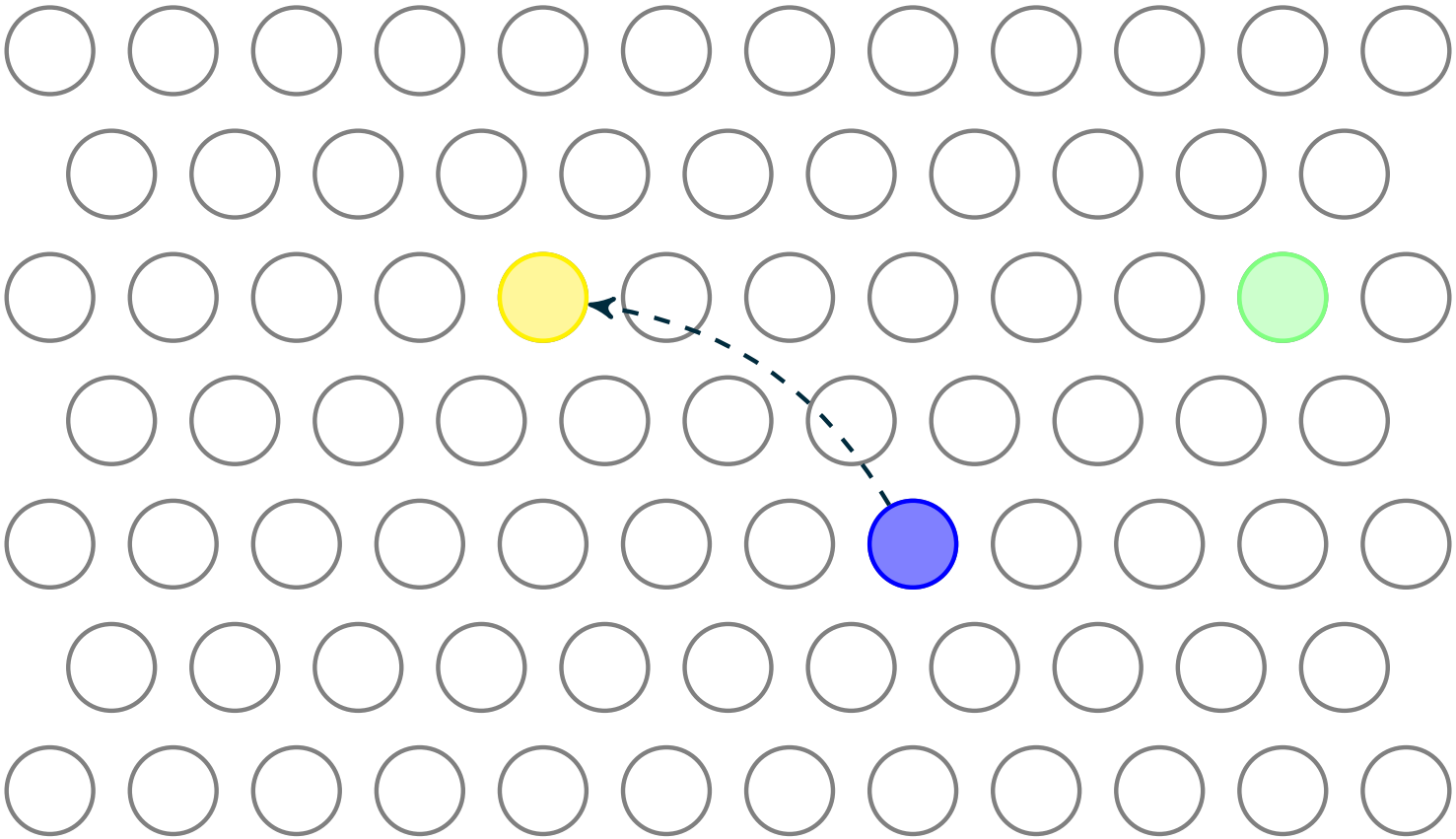
# Iterated Local Search



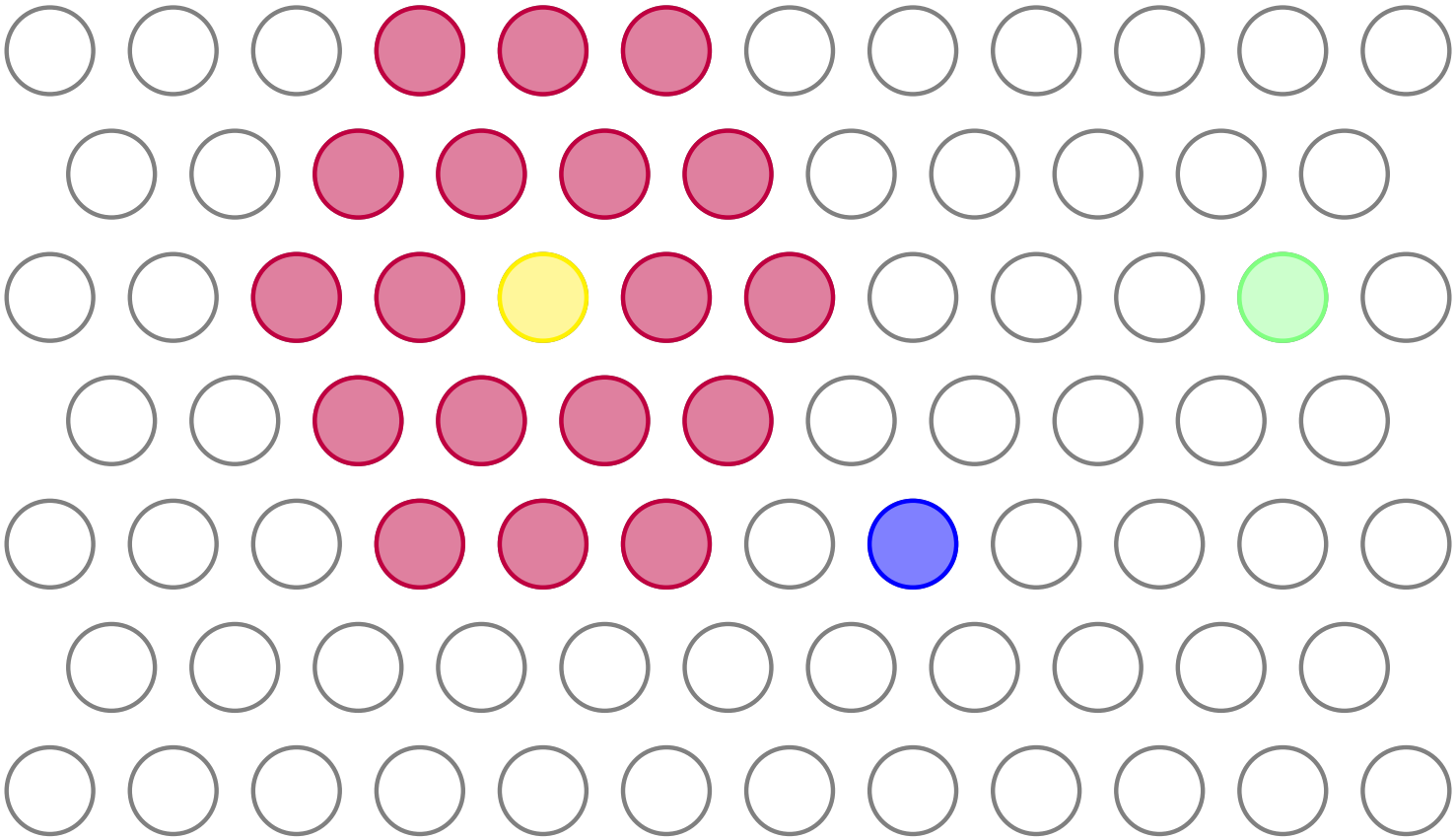
# Iterated Local Search



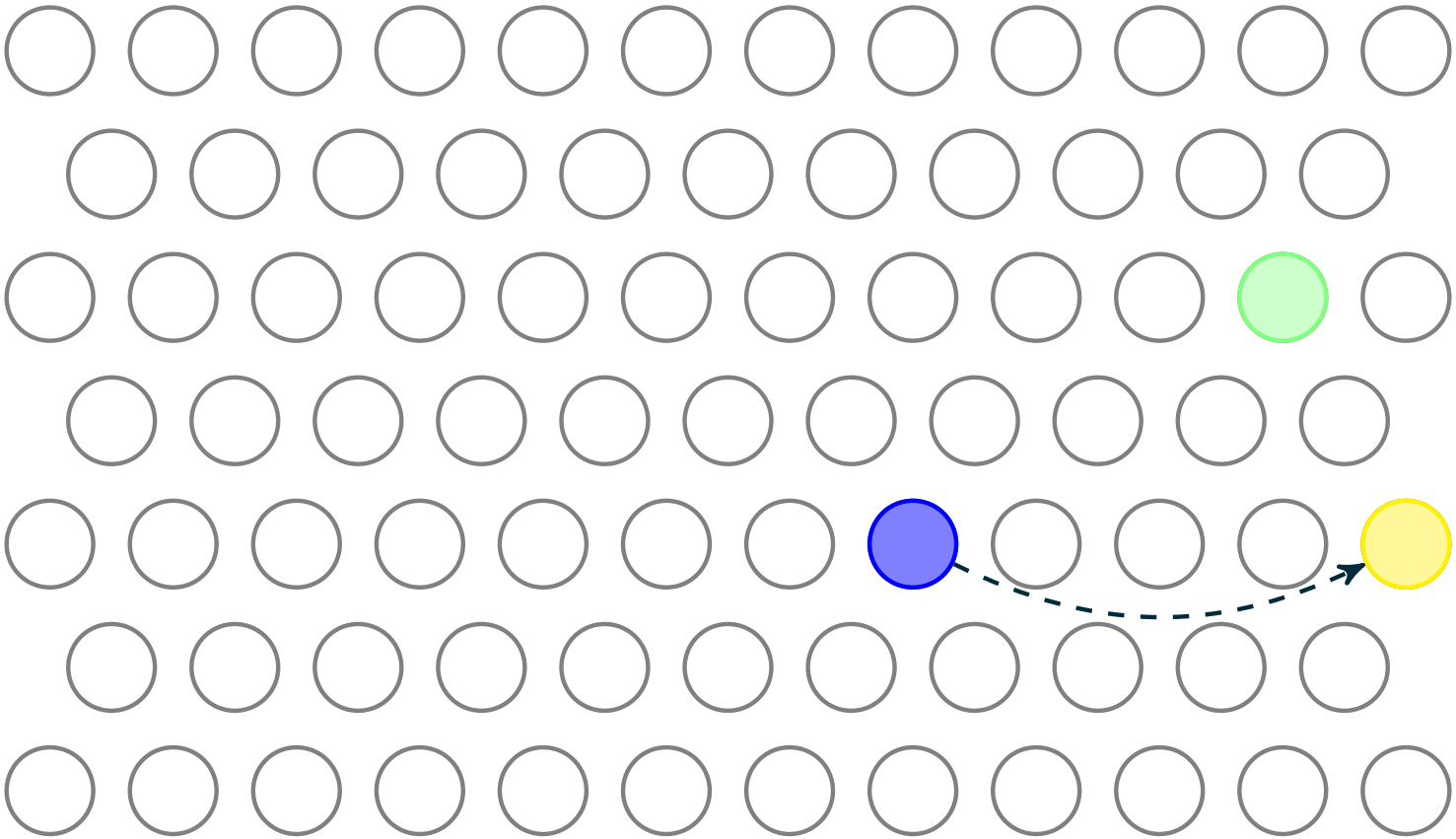
# Iterated Local Search



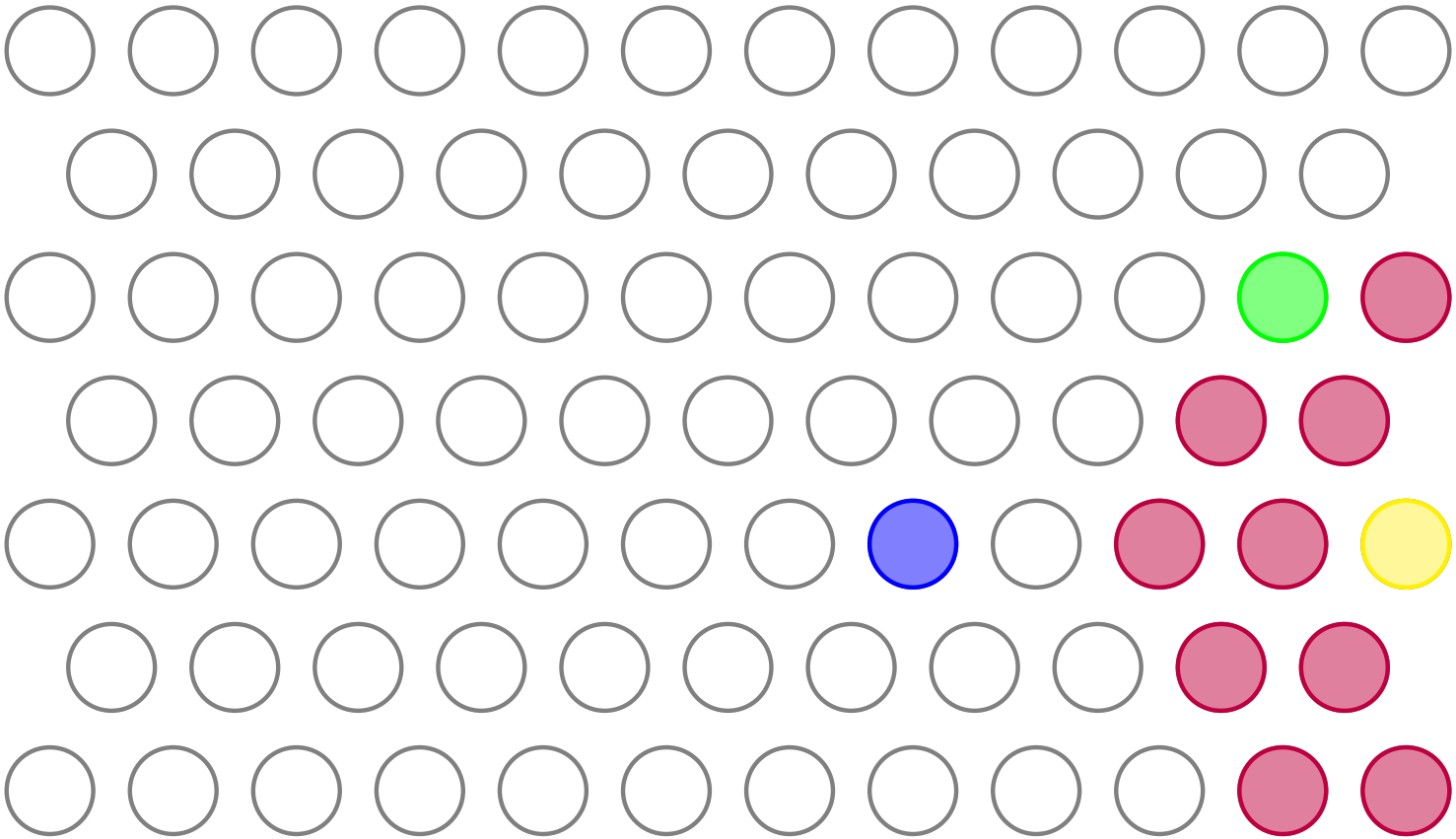
# Iterated Local Search



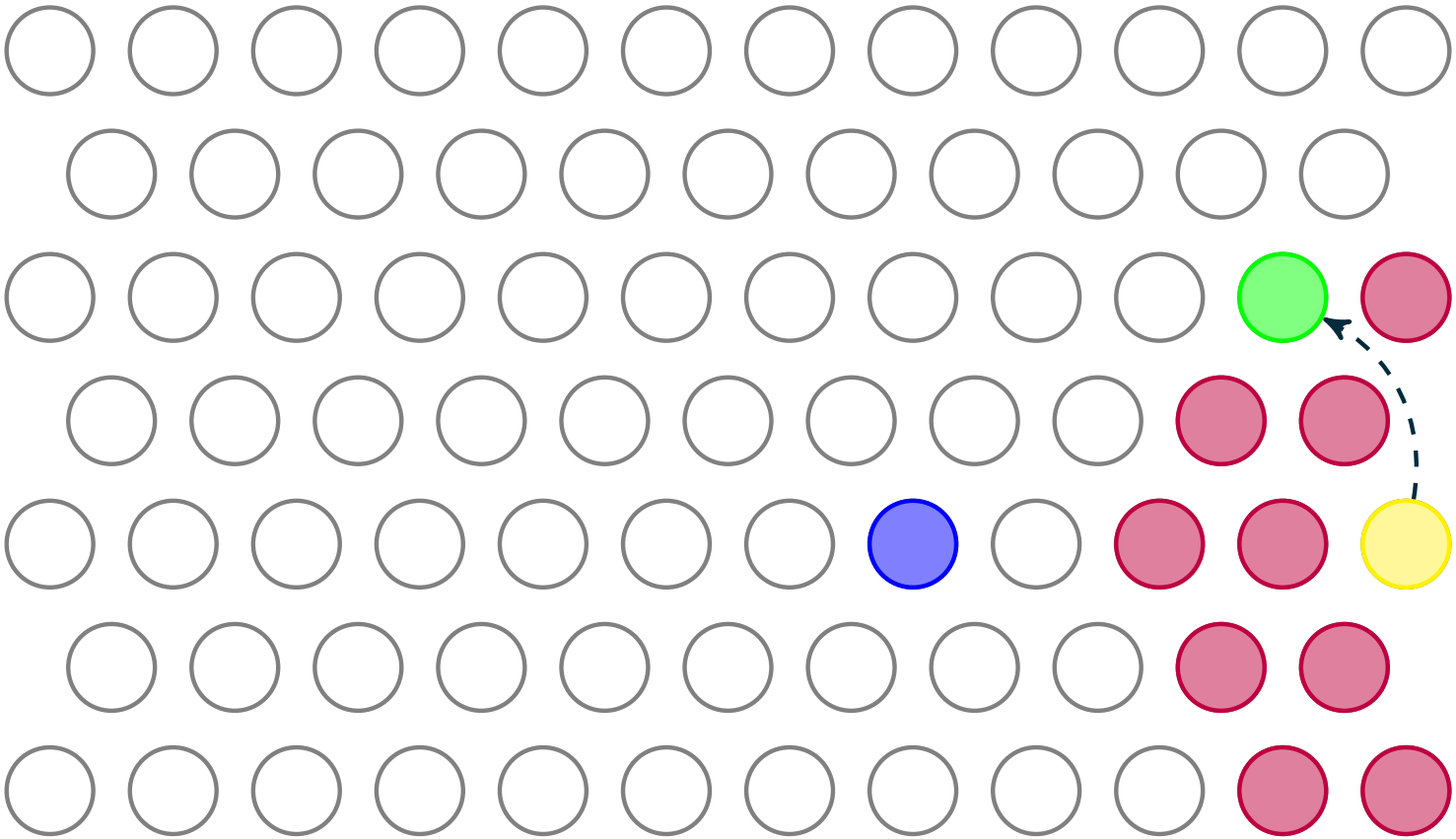
# Iterated Local Search



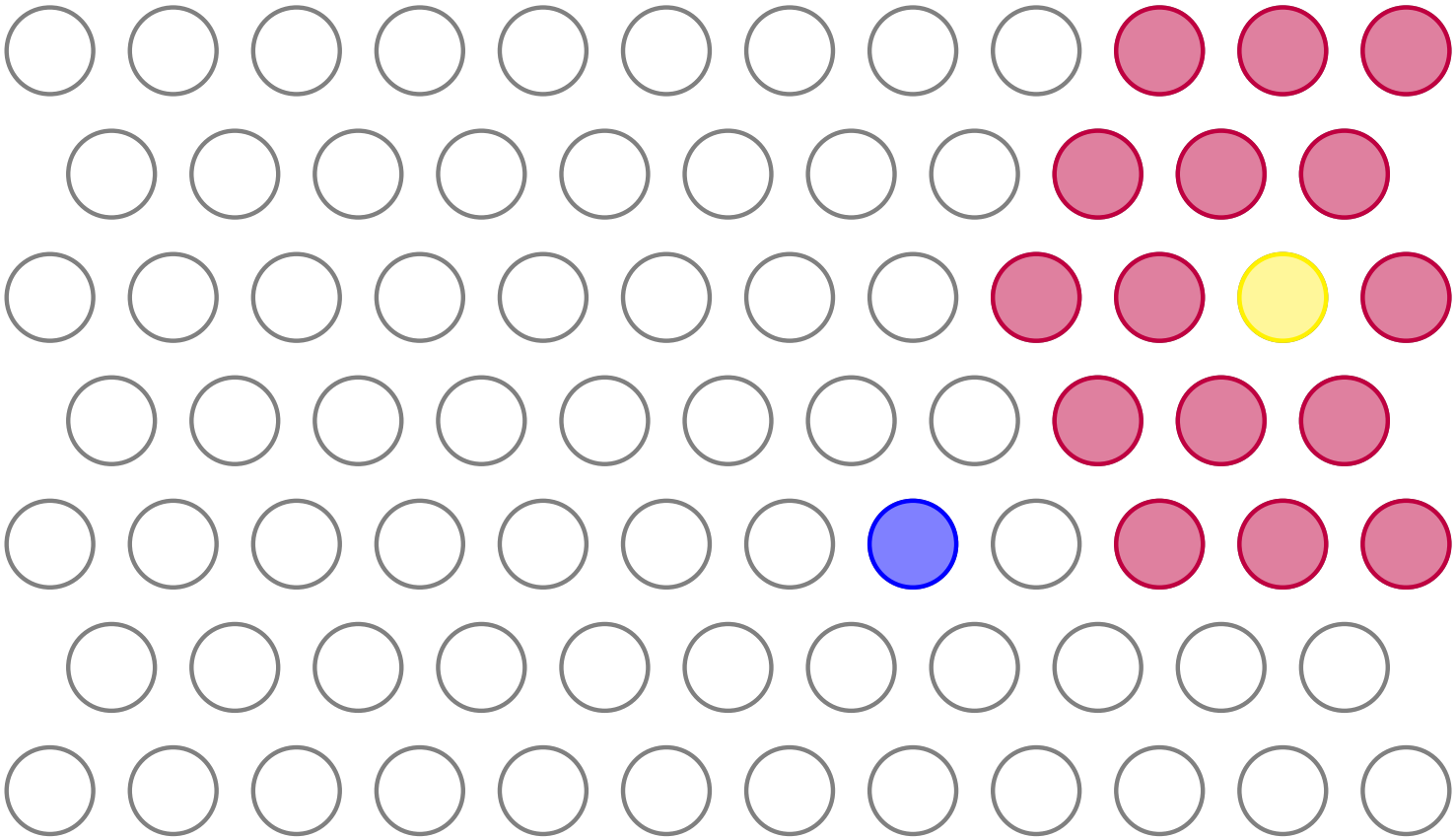
# Iterated Local Search



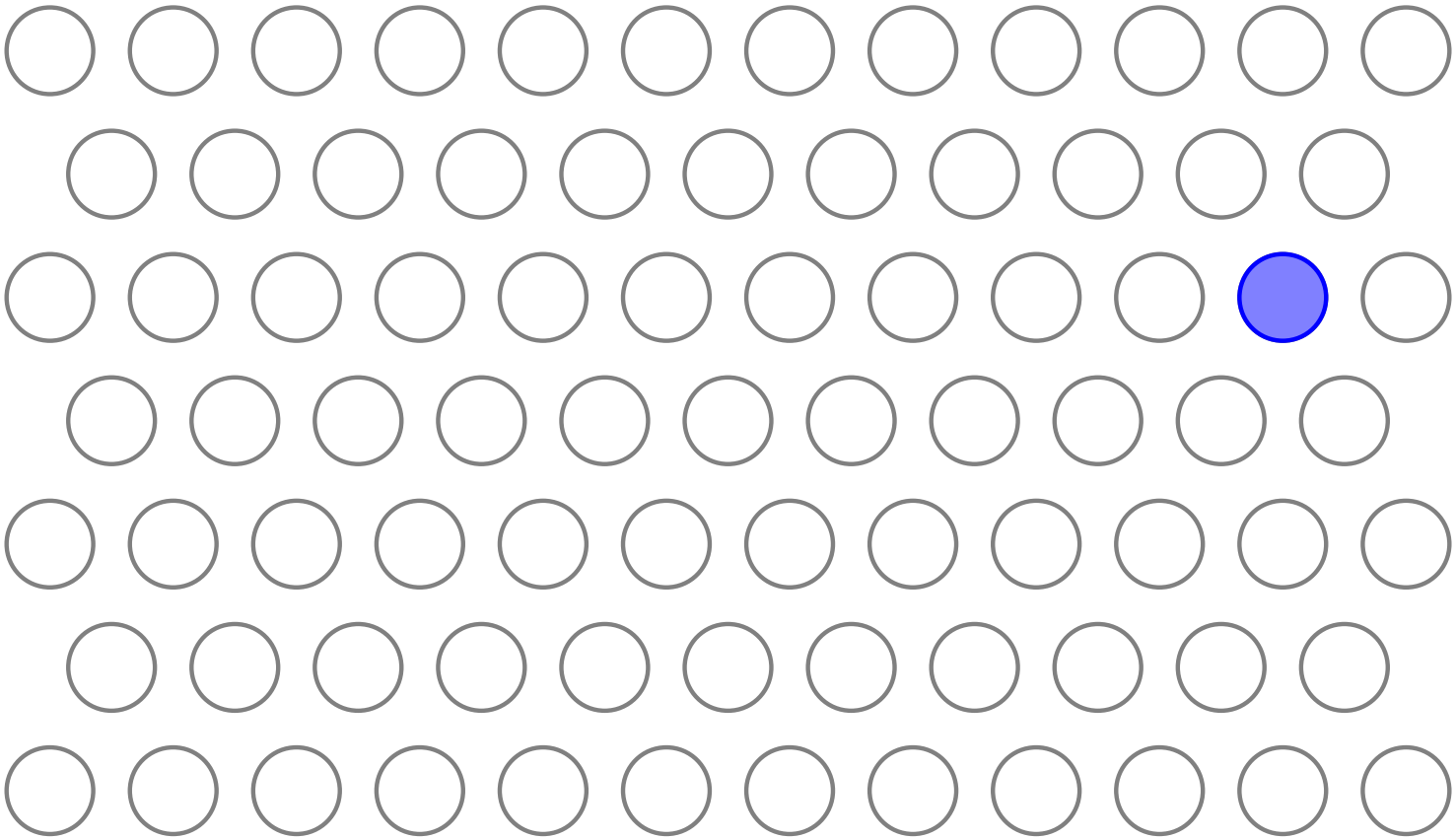
# Iterated Local Search



# Iterated Local Search



# Iterated Local Search



# Implementation details

- Three neighborhood operators: relocate, exchange and crossover
- Efficient calculation of the neighborhood in each iteration.
  - Benefit: Drastic reduction of the execution time
  - Drawback: More complex implementation and lost of modularity in the code
- Infeasible solutions (capacity constraint) are taken into account during the first stages of the algorithm and are handled by the penalization of the objective function.

# Computational experiments

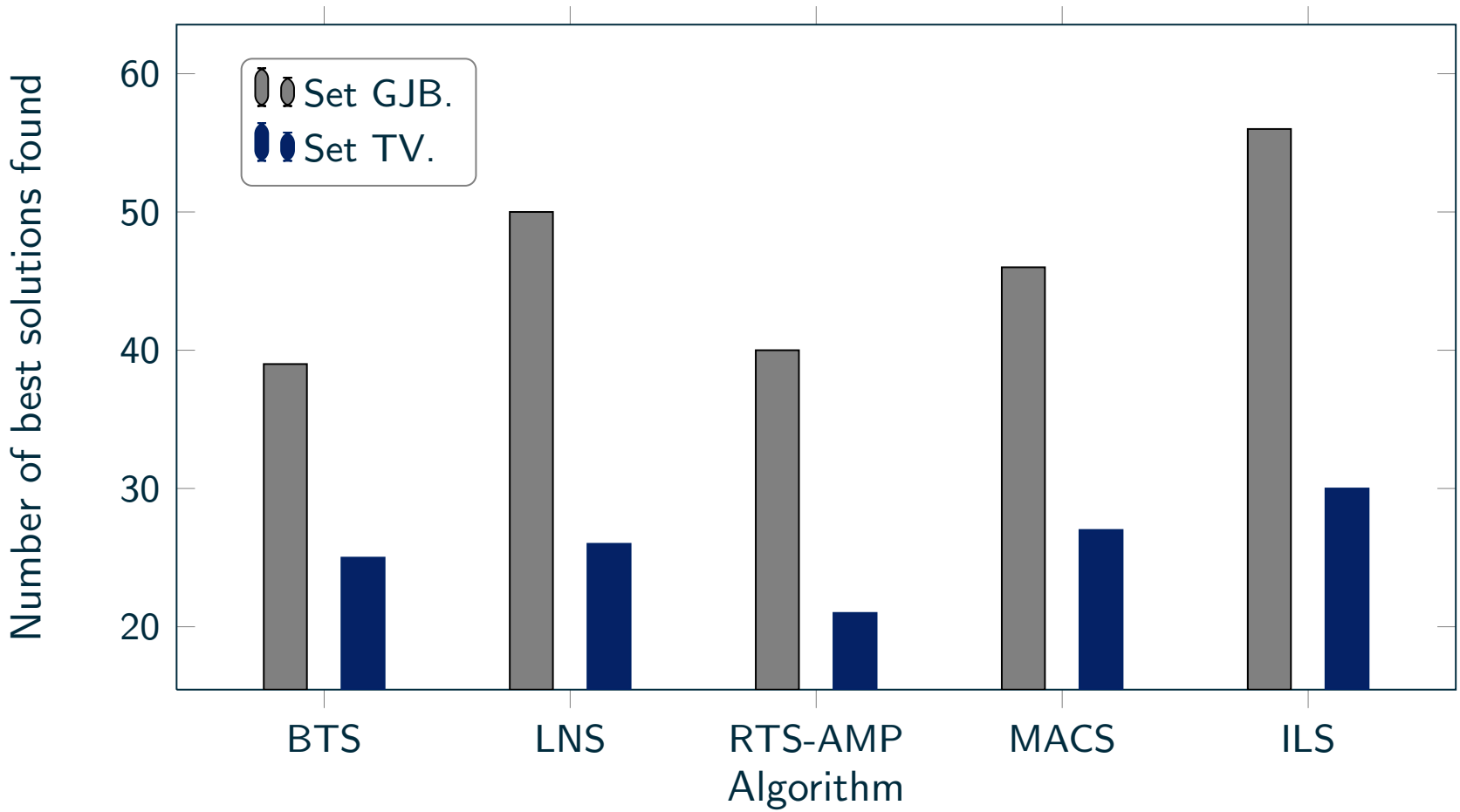
- Two sets of problems:
  - **GJB**: 62 instances, proposed by Goetschalckx and Jabobs-Blecha
  - **TV**: 33 instances, proposed by Toth and Vigo
- 15 runs of the algorithm for each instance
- Computer: 2.13GHz Intel Core 2 Duo P7450

# ILS vs. The state of the art algorithms

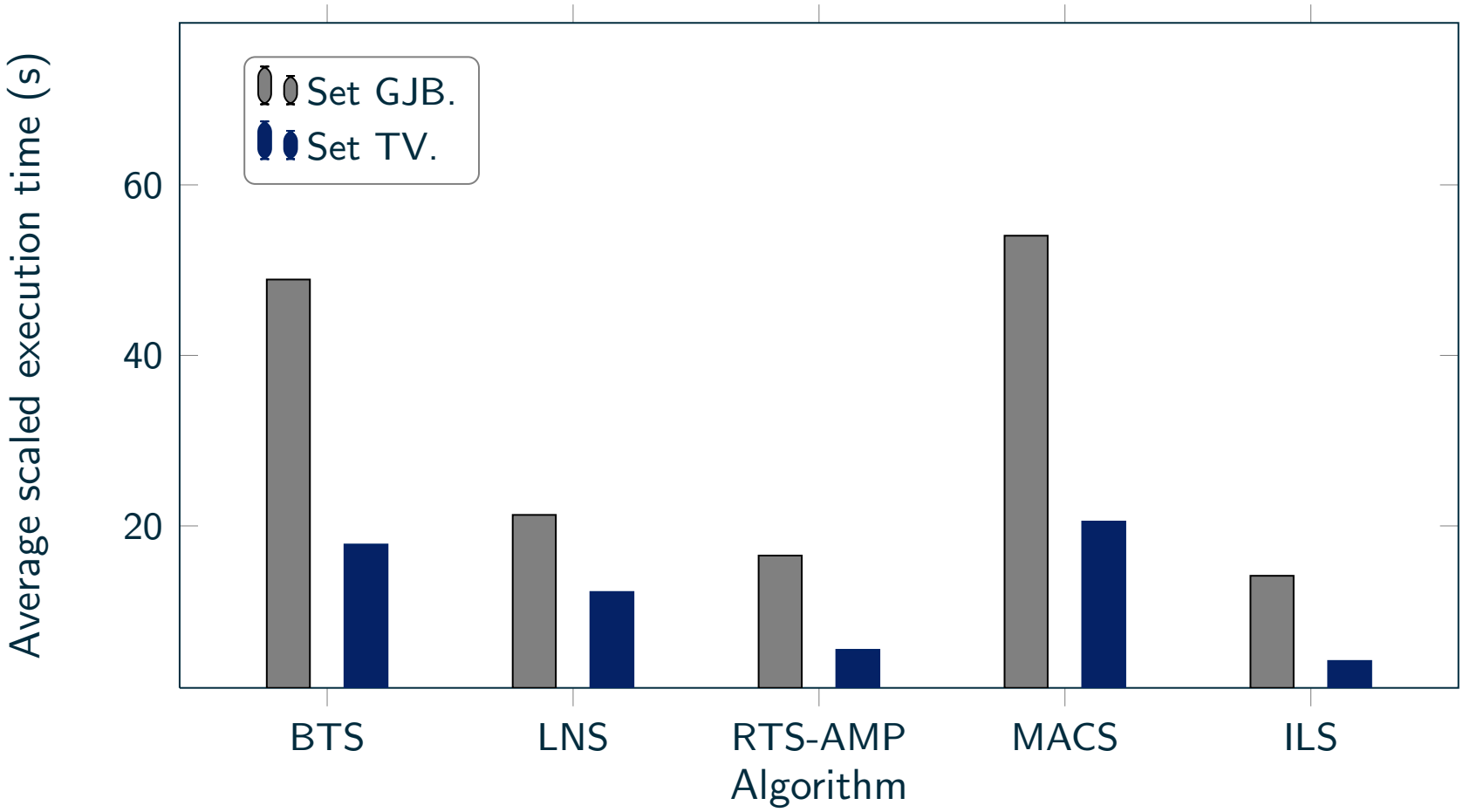
Algorithms compared:

- **BTS**: Brandão's Tabu Search
- **LNS**: Røpke and Pisinger's Large Neighbourhood Search
- **RTS-AMP**: Wassan's Reactive Adaptative Memory Programming Search
- **MACS**: Gajpal and Abad's Multi-ant Colony System
- **ILS**: proposed Iterated Local Search

# ILS vs. The state of the art algorithms



# ILS vs. The state of the art algorithms



# Conclusions

## What we propose:

A simple and competitive algorithm to solve the Vehicle Routing Problem with Backhauls

## What we learned:

Allowing the Local Search to consider infeasible solutions in the early stages, can be helpful to find valuable information about the global optimum solution in fewer iterations

## Future work:

- Extend this algorithm in order to solve other variants of VRP
- Find efficient mechanisms to handle the violation of other constraints through the penalization of the objective function