A Comparison of Evolutionary Crossover Operators for Real-World Vehicle Routing Problems with Simultaneous Pickup and Delivery with Time Windows

Elya Denysova[1], Ethan Gibbons[1], Alex Bailey[2], Beatrice Ombuki-Berman[1]

1.Department of Computer Science
Brock University, St Catharines, Ontario, Canada

2.RideCo Inc, Waterloo, Ontario, Canada

Introduction

- A variant of a well-established optimization task
- Industries involving transportation of goods
- The goal is to schedule the delivery routes that would optimize the objectives and comply with constraints.

Objectives:

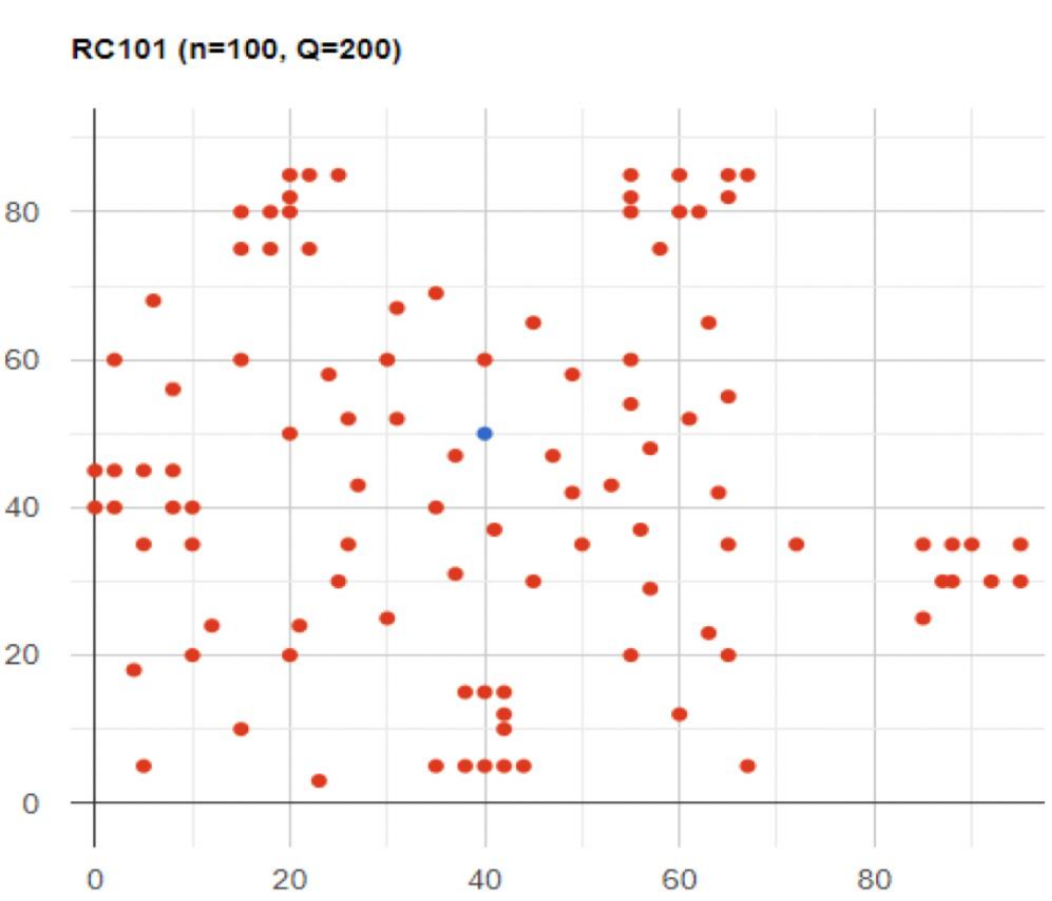
1. Minimize vehicles
2. Minimize distance

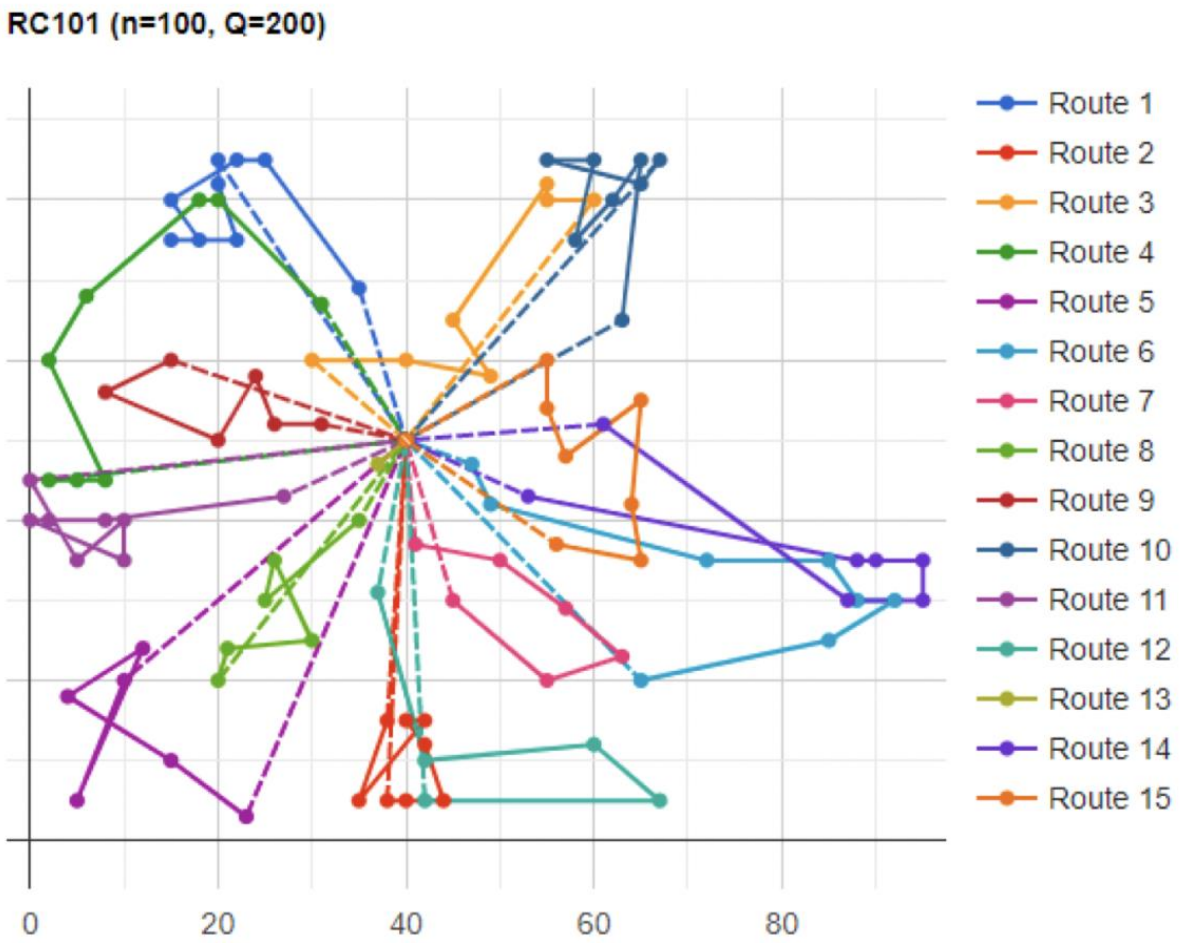
Two objectives are combined in one score

Constraints:

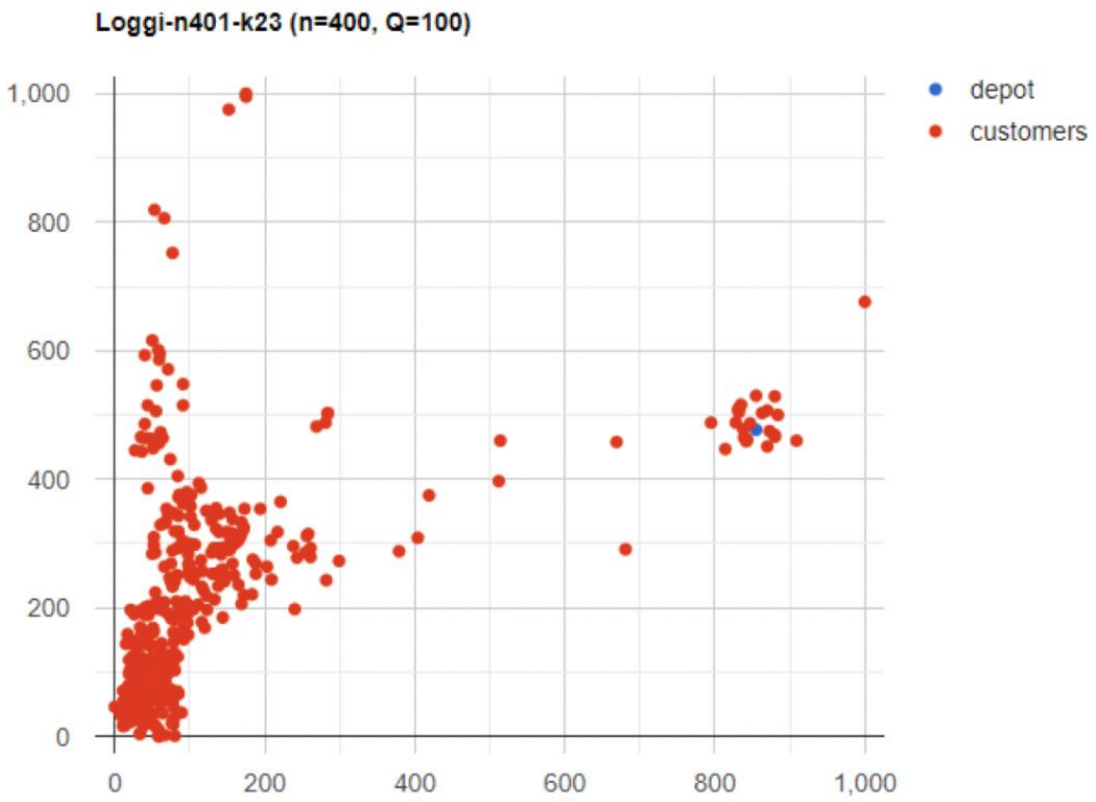
1. Visit each customer exactly once
2. Vehicle's capacity
3. Time Windows
4. Depot working hours

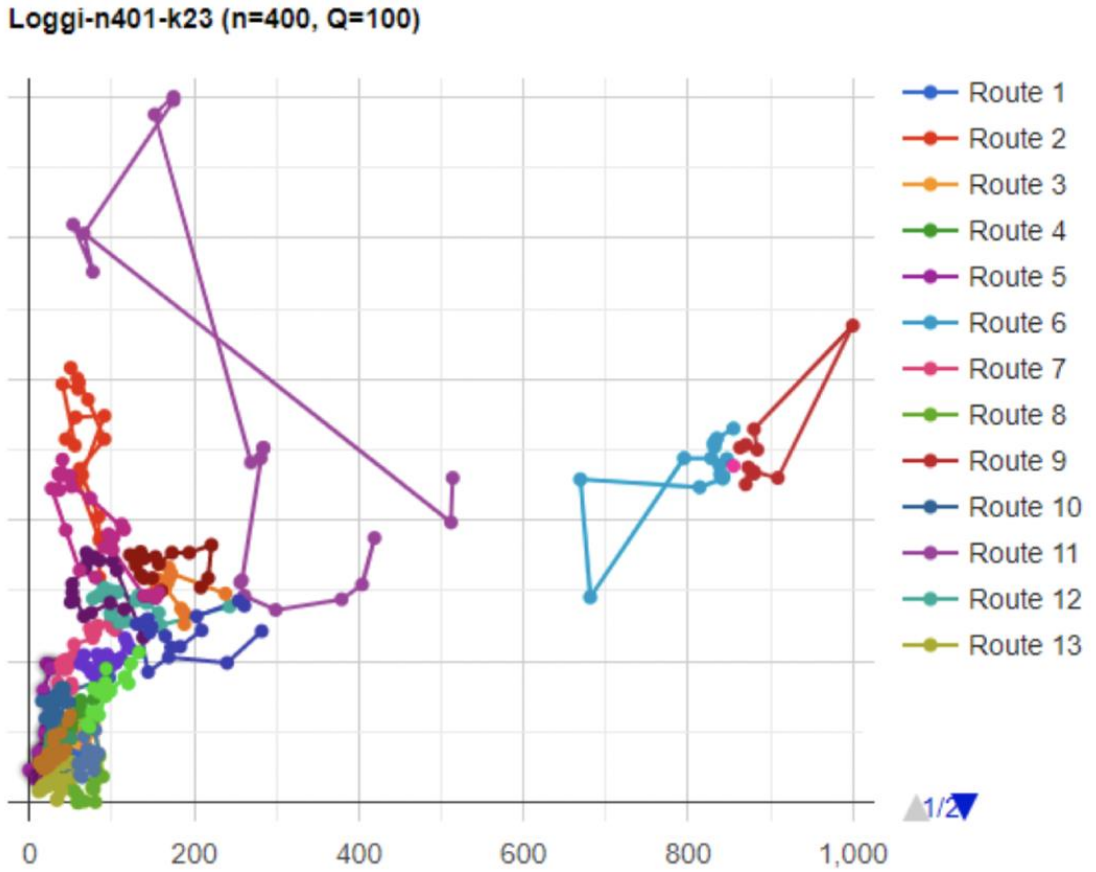
VRPSPDTW





VRPSPDTW





Problem Formulation

$$N = \{1, 2, 3, \ldots, n\}$$

$$V = \{0\} \cup \mathbb{N}$$

$$S = \{R_1, R_2, R_3, \ldots, R_K\}$$

$$R_k = \{i_0, i_1, i_2, \ldots, i_{H_k+1}\}$$

$$C_R = \sum_{h=0}^{H} c_{i_h, i_{h+1}}$$

Minimize $\left(u_1 \cdot K + u_2 \cdot \sum_{k=1}^{K} C_{R_k}\right)$

Problem Formulation

Constraints:

$$K \leq J$$

$$i_0 = i_{H_R+1} = 0 \qquad \forall R \in S$$

$$\sum_{i \in V} \sum_{k=1}^{K} x_{i,j}^{k} = 1 \qquad \forall j \in N$$

$$D_{R_k} \leq Q \qquad \forall k \in \{1, 2, \ldots, K\}$$

$$l_{R_k, i_h} \leq Q \qquad \forall k \in \{1, 2, \ldots, K\}, \; \forall h \in \{1, 2, \ldots, H_{R_k}\}$$

$$a_{i_h} \leq B_{R_k, i_h} \leq b_{i_h} \qquad \forall k \in \{1, 2, \ldots, K\}, \; \forall h \in \{1, 2, \ldots, H_{R_k}\}$$

Related Works and Data

- VRPSPDTW have attracted substantial research interest in the recent years
- Popularized with the release of synthetic VRPSPDTW problem set by Wang and Chen (2012)
- Large-scale real-world VRPSPDTW instances were released by Liu et al. (2021)
- 20 instances of sizes: 200, 400, 600, 800, and 1000 customers were derived from real customers from the company JD Logistics

Strategies

VRPs are NP-Hard problems, implying that finding the optimal routing solution cannot be guaranteed except through the use of exact algorithms on small problem instances.

Heuristic Methods:

- Nearest Neighbor Method,
- Cheapest Insertion Method,
- Clarke and Wright Savings.

Local Search-based metaheuristics:

- Simulated Annealing,
- Tabu Search,
- Variable Neighborhood Descent.

Population-based metaheuristics:

- Genetic Algorithms,
- Any Colony Systems,
- Particle Swarm Optimization.

Predictive methods:

- Machine Supervised Learning,
- Markov Decision Processes,
- Reinforcement Learning.

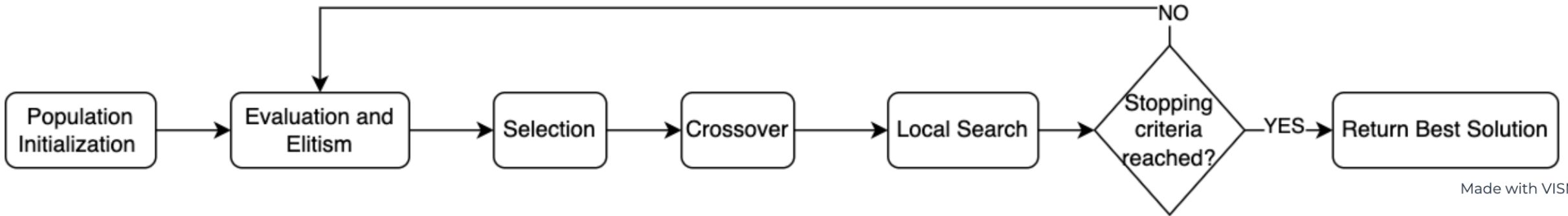
Memetic Algorithm

- A Memetic Algorithm (MA) is a Genetic Algorithm (GA) with a local search
- GAs take inspiration from the natural selection in the real world

Algorithm 1: Memetic Algorithm for the VRPSPDTW

Input: VRPSPDTW Instance, n_{pop}, f_{search}, p_{search}.

$U \leftarrow$ set of n_{pop} empty solutions.
Initialize solutions in U.
Perform local search on $n_{pop} * p_{search}$ solutions in U.
Maintain elitism.
$g \leftarrow 1$.
while *termination condition is not met* **do**
  $U_{new} \leftarrow$ perform k-tournament selection on U $(k = 3)$.
  for *each pair of solutions* (P_1, P_2) *in* U_{new} **do**
    $O_1, O_2 \leftarrow$ perform crossover using solutions (P_1, P_2).
    O_1 and O_2 replace P_1 and P_2 in U_{new}.
  end
  $U \leftarrow U_{new}$.
  if g **mod** $f_{search} = 0$ **then**
    Perform local search on $n_{pop} * p_{search}$ solutions in U.
  end
  Maintain elitism.
  $g \leftarrow g + 1$.
end
return *Best solution found.*

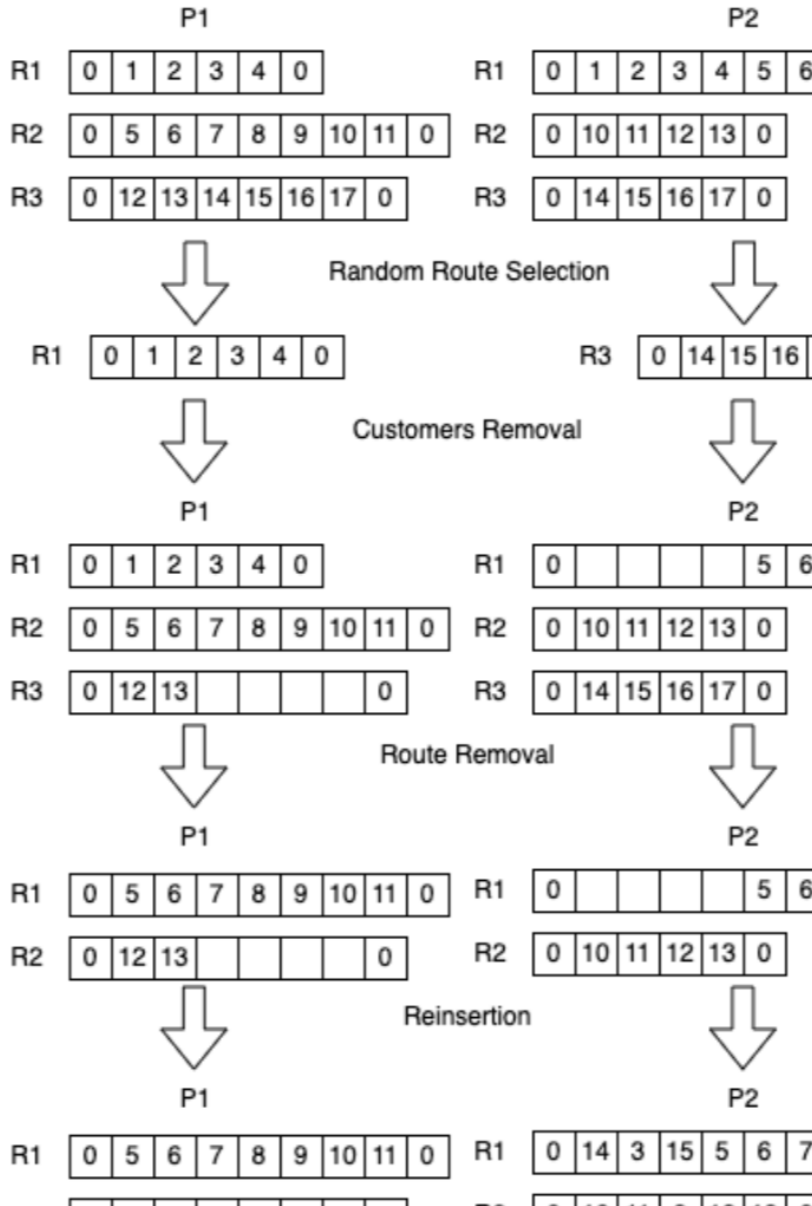


Crossovers

- Best-Cost-Route-Crossover with Route Destruction (BCRCD)
- Route-Assembly-Regret-Insertion Crossover (RARI)
- Route-Copy-Based Crossover (RCX)
- Two-Stage Crossover (2STAGE)
- Exchange Route Crossover (ERX)
- Exchange Route Crossover Regret Insertion (ERXR)
- Pheromone-Based Crossover (PB)
- Pheromone-Based Crossover Extra (PBE)

Best-Cost-Route-Crossover with Route Destruction (BC

Given 2 parent solutions P1 and P2, the routes R1 and R2 are chosen at random from P1 and P2 respectively. The customers in R1 are removed from P2 and vice versa. Then, the routes R1 and R2 are removed from P1 and P2 respectively. In the end, unassigned customers are reinserted via the cheapest insertion.



Route-Assembly-Regret-Insertion Crossover (RARI)

Given two parent solutions P1 and P2, a new empty offspring solution O is created. Routes R1 and R2 are repeatedly chosen from P1 and P2 respectively and inserted into O until there are no more inheritable routes. Remaining unassigned customers are reinserted via regret-based insertion. The regret value represents what is the cost of inserting a node in the future iteration

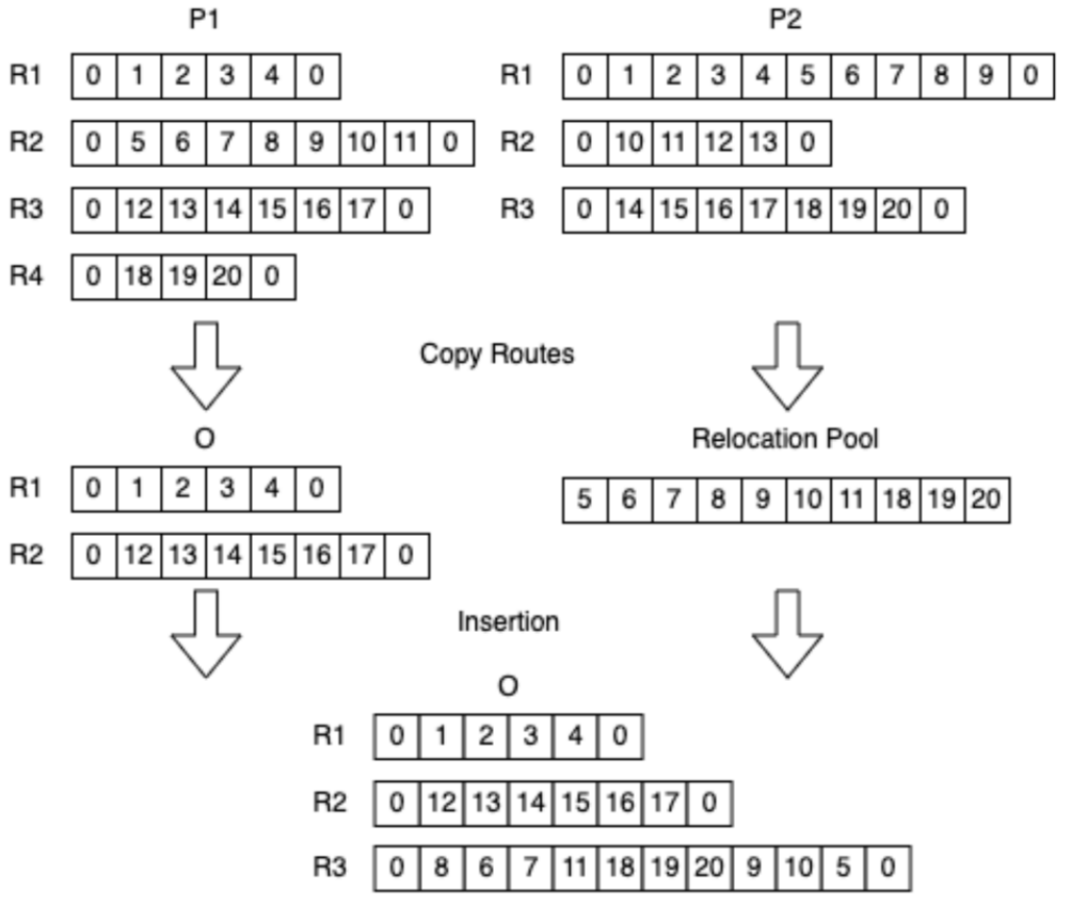
Algorithm 3: The RARI Crossover Operator.

input : parent solutions p_1, p_2
output: x_{child}
$x_{child} \leftarrow$ initialize an empty solution with no routes;
repeat
  copy random route from p_1 to x_{child};
  copy random route from p_2 to x_{child};
until *no more inherited routes are feasible*;
`/* --- regret-based insertion ---`
$U \leftarrow$ all the remaining unassigned customers;
while $U \neq \emptyset$ **do**
  foreach *node* $v \in U$ **do**
    calculate $regret(v)$ according to Eq.~(5);
  end
  $v^* \leftarrow \text{argmax}_{v \in U} regret(v)$;
  Insert v^* into x_{child} at its best insertion position;
  $U \leftarrow U \backslash \{v^*\}$;
end
return x_{child}

Source: [1]

Route-Copy-Based Crossover (RCX)

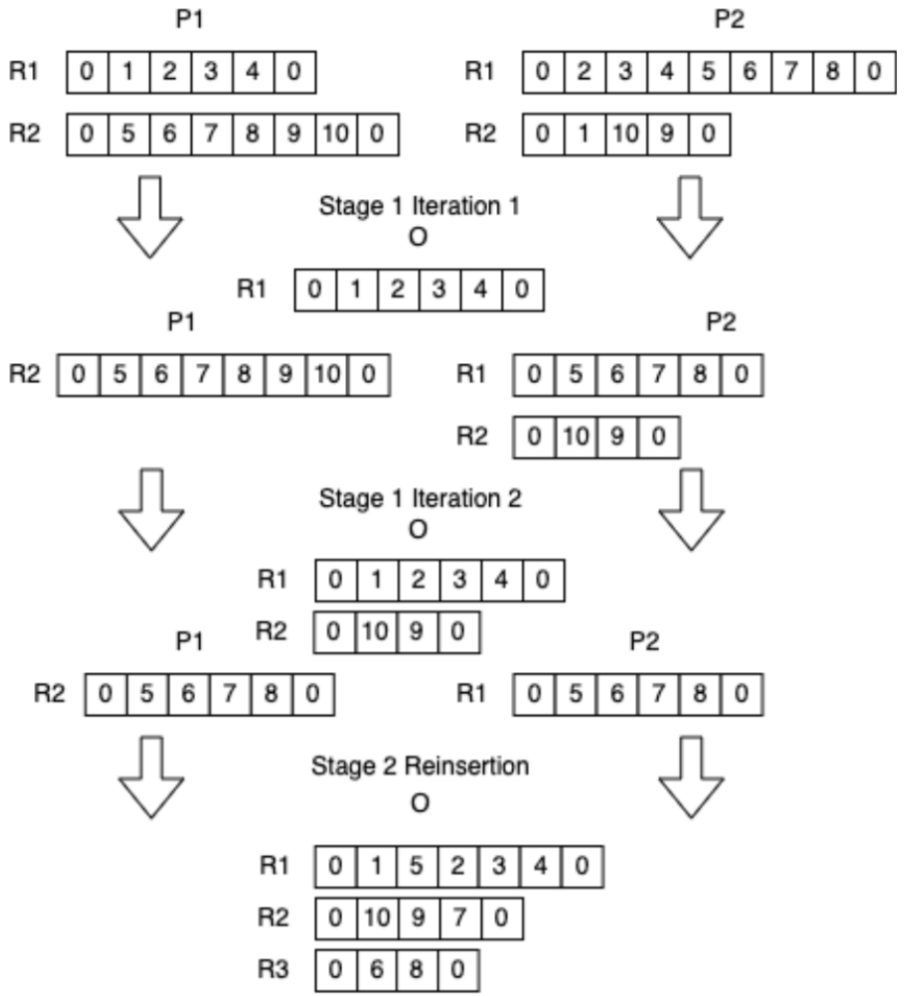
Given two parent solutions P1 and P2, each route R in P1 is copied over to the new empty offspring solution O with 0.5 probability. Unassigned customers are pooled in order from P2 and inserted in solution O via the cheapest insertion



Source: [4]

Two-Stage Crossover (2STAGE)

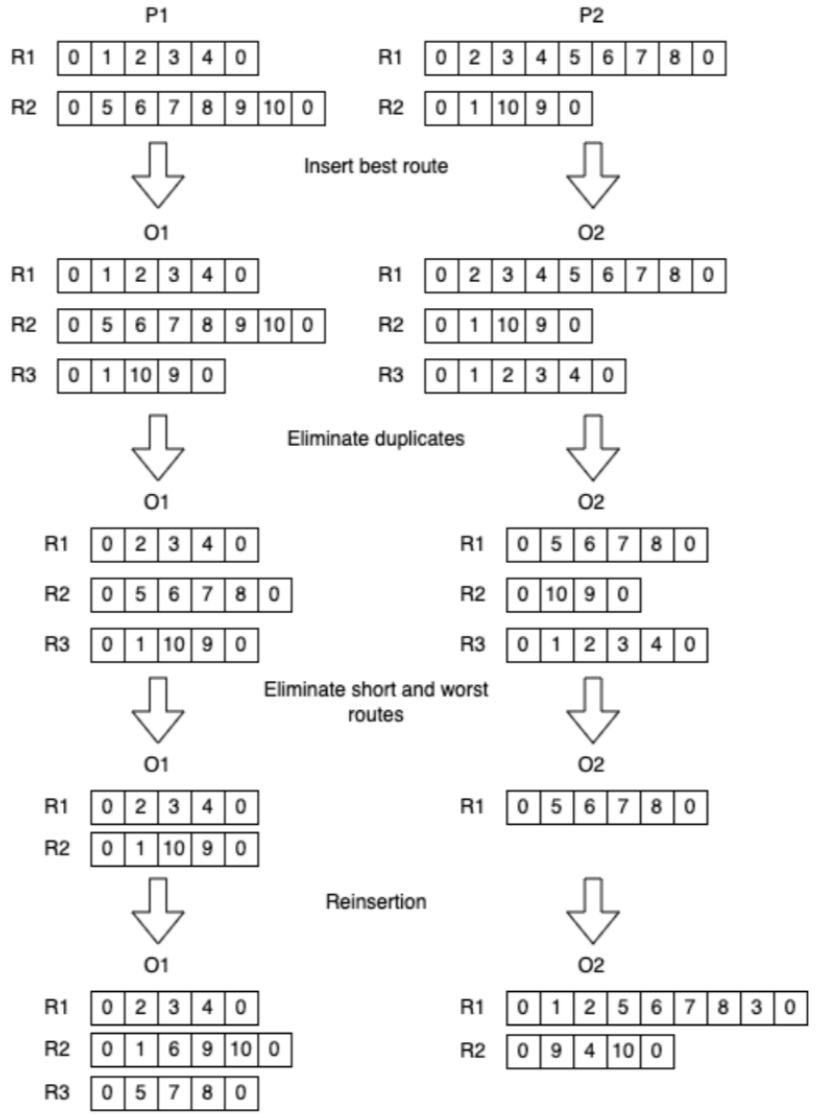
Given two parent solutions P1 and P2, an empty offspring solution is created. In the first stage, parent P is chosen at random repeatedly and the most promising route R is copied to O, then the route R is discarded from P and customers in R are discarded from the other parent. In the second stage, unallocated customers are reinserted via the cheapest insertion



Source: [5]

Exchange Route Crossover (ERX)

Given two parent solutions P1 and P2, two offspring O1 and O2 created by copying P1 and P2 respectively. The best route in P1 is inserted in O2 and vice versa for P2 and O1. Duplicate customers are removed. Then the routes with few customers are removed as well as the route with the worst score. In the end unassigned customers are reinserted via the cheapest insertion



Source: [6]

Exchange Route Crossover Regret Insertion (ERXR)

- Alteration of ERX crossover
- Regret-based insertion instead of the cheapest insertion

Pheromone-Based Crossover (PB)

Given two parent solutions P1 and P2, new offspring O is constructed by utilizing local neighbourhood information in P1 and P2 as well as global neighbourhood information in the pheromone arcs tracking paths from all the best solutions

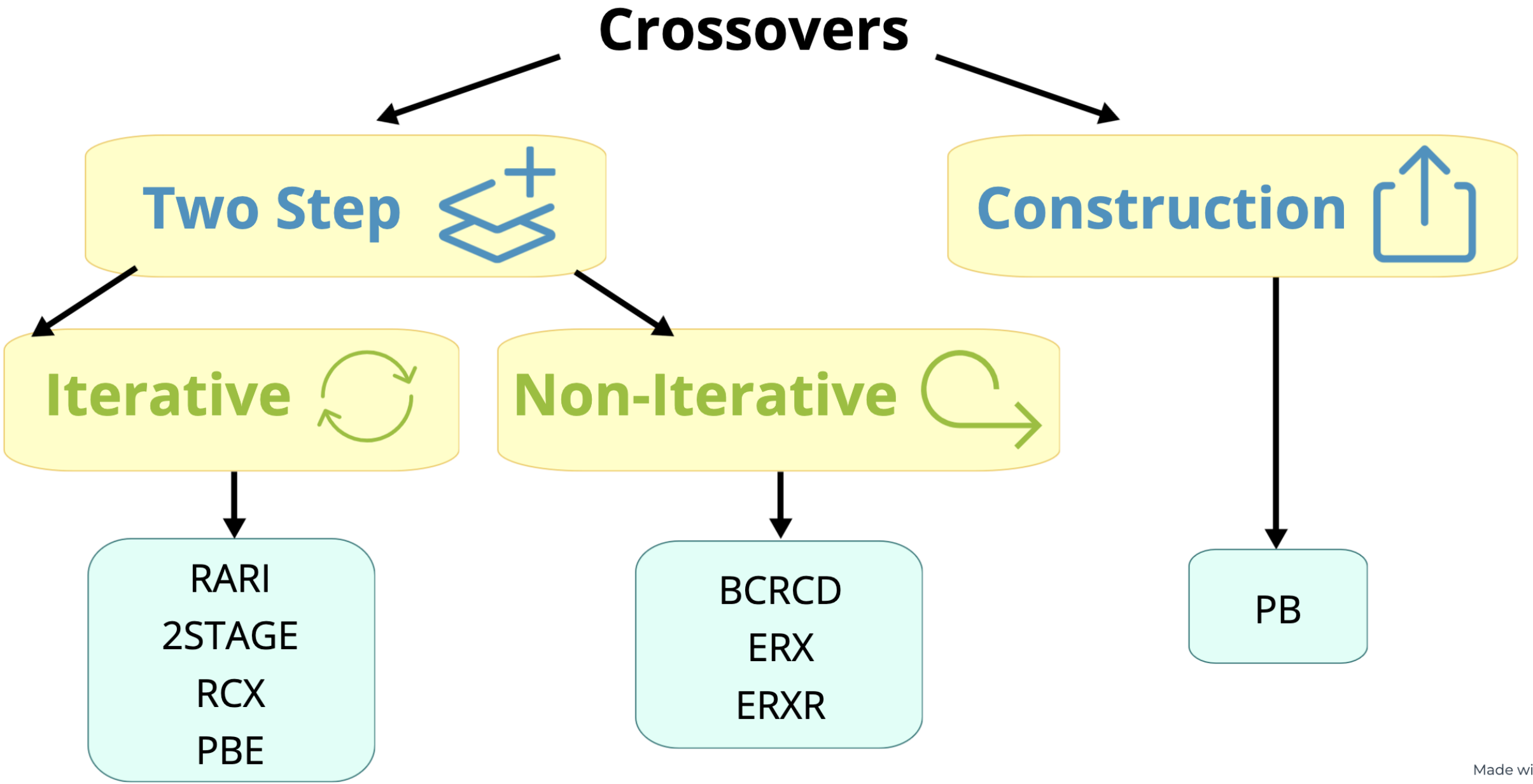
```
begin
   i = 1, k =1;
   select a random customer as initial customer, o(i), of the o;
   while ( i< n+k ) do
      search o(i) in parents p1 and p2 respectively;
      if ( all neighbors of o(i) in p1 and p2 have appeared in o, or all
            neighbors can not be feasibly added to the end of o)
         generate a random decimal fraction q;
         select the next customer j according to formula 1;
         if ( no feasible customer is found)
            i = i+1;
            k = k+1;
            o(i) ← 0 (depot);
            randomly select an unvisited customer j;
            i = i+1;
            o(i) ← j;
         end if;
      else
         select the nearest customer that has not appeared in o from the
          feasible neighbors of o(i) as the next customer j;
         i = i+1;
         o(i) ← j;
      end if;
   end while;
end.
```

Source: [7]

Pheromone-Based Crossover Extra (PBE)

- Alteration of PB crossover
- Some customers are inserted via cheapest insertion
- Adds the ability to insert customers in any position (not just last place on current route or in the new route)

Categories

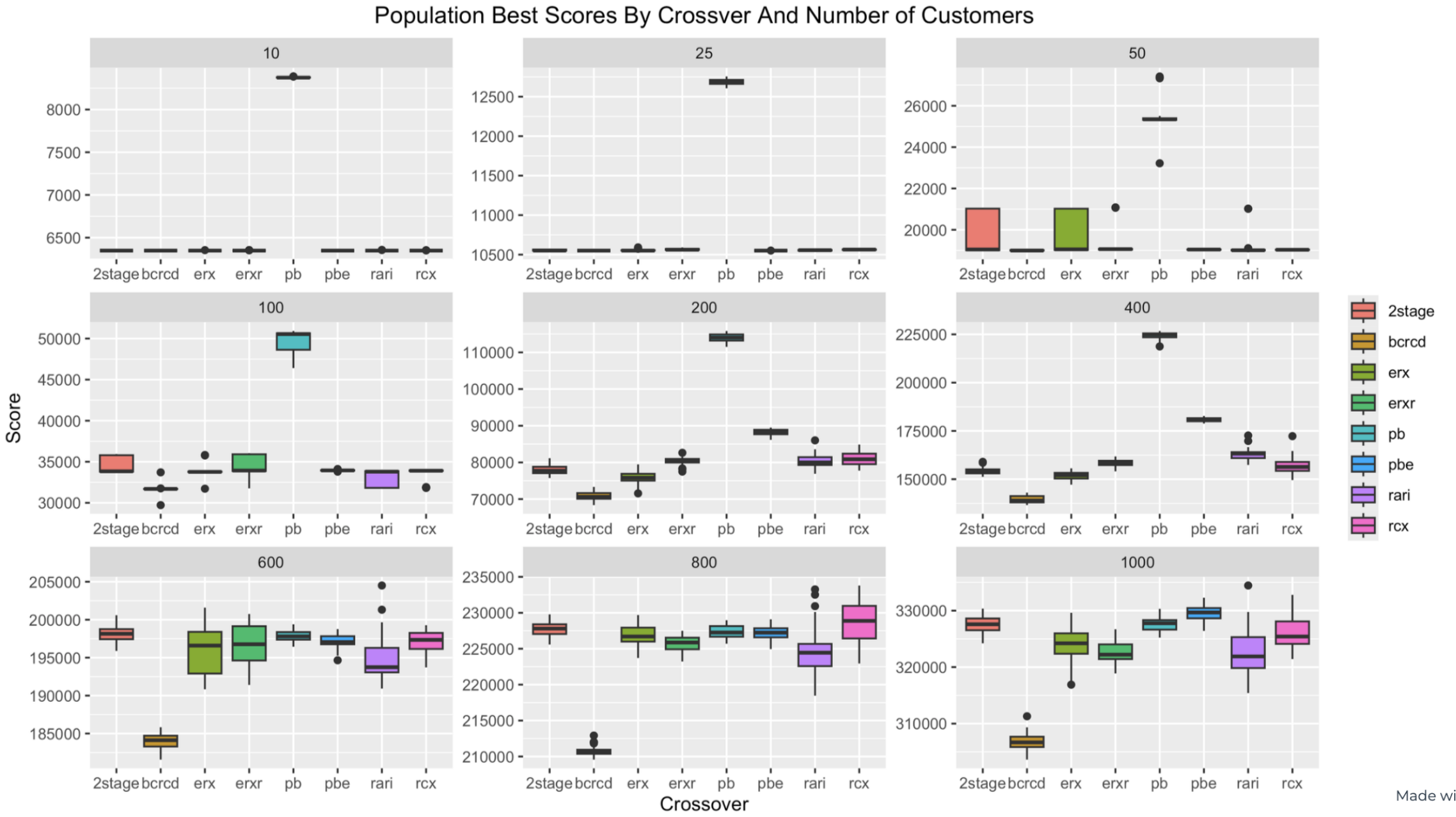


Experimental Setup

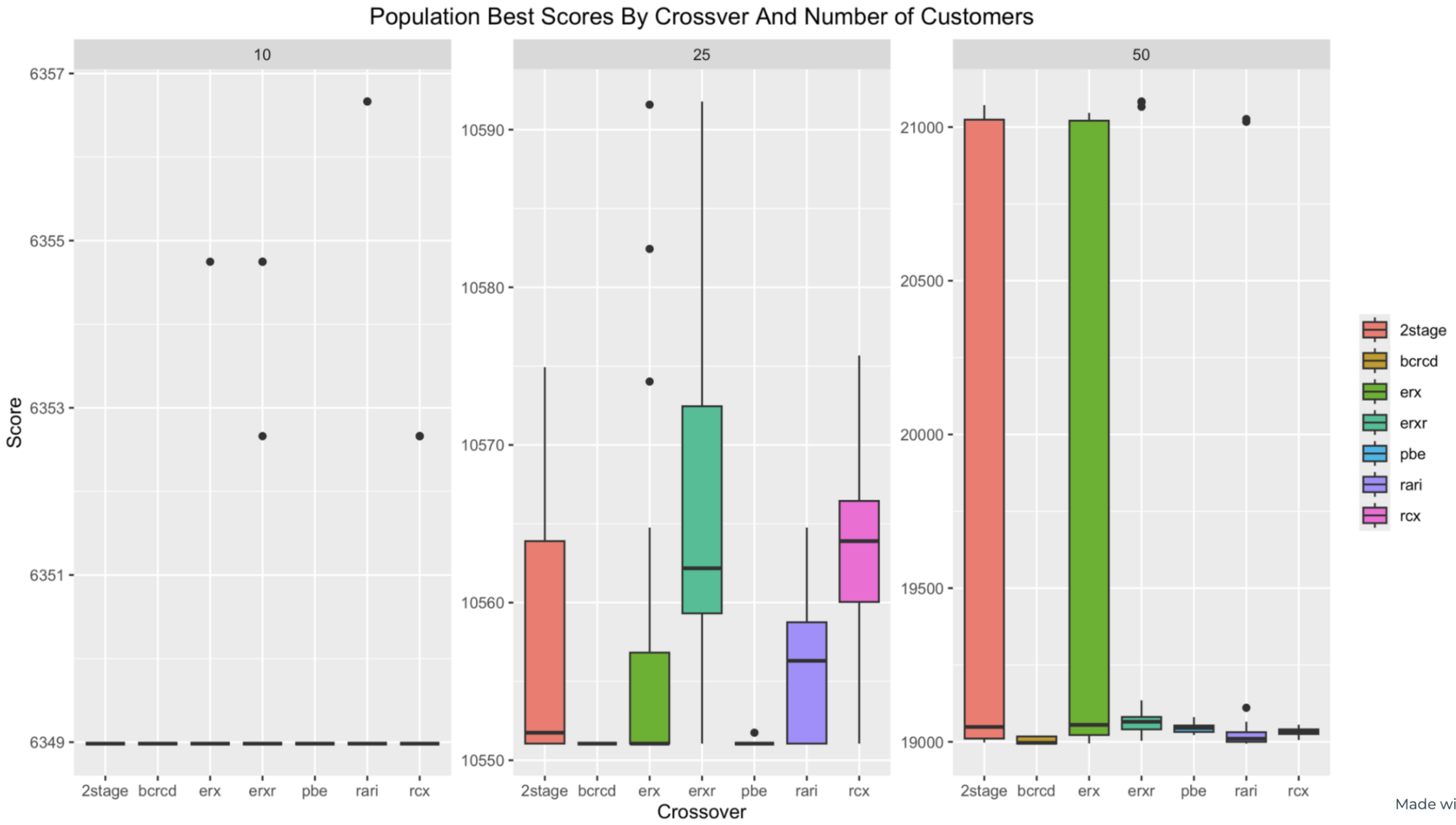
Global parameters were the same for all the crossover operators

- Generation size: 50
- Local search step: every 20 generations
- Local search rate: 0.75 (0.1 for large instances)
- Run times: 30
- Time-out: 2 hours
- Selection: k-tournament, k=3
- Crossover probability: 1

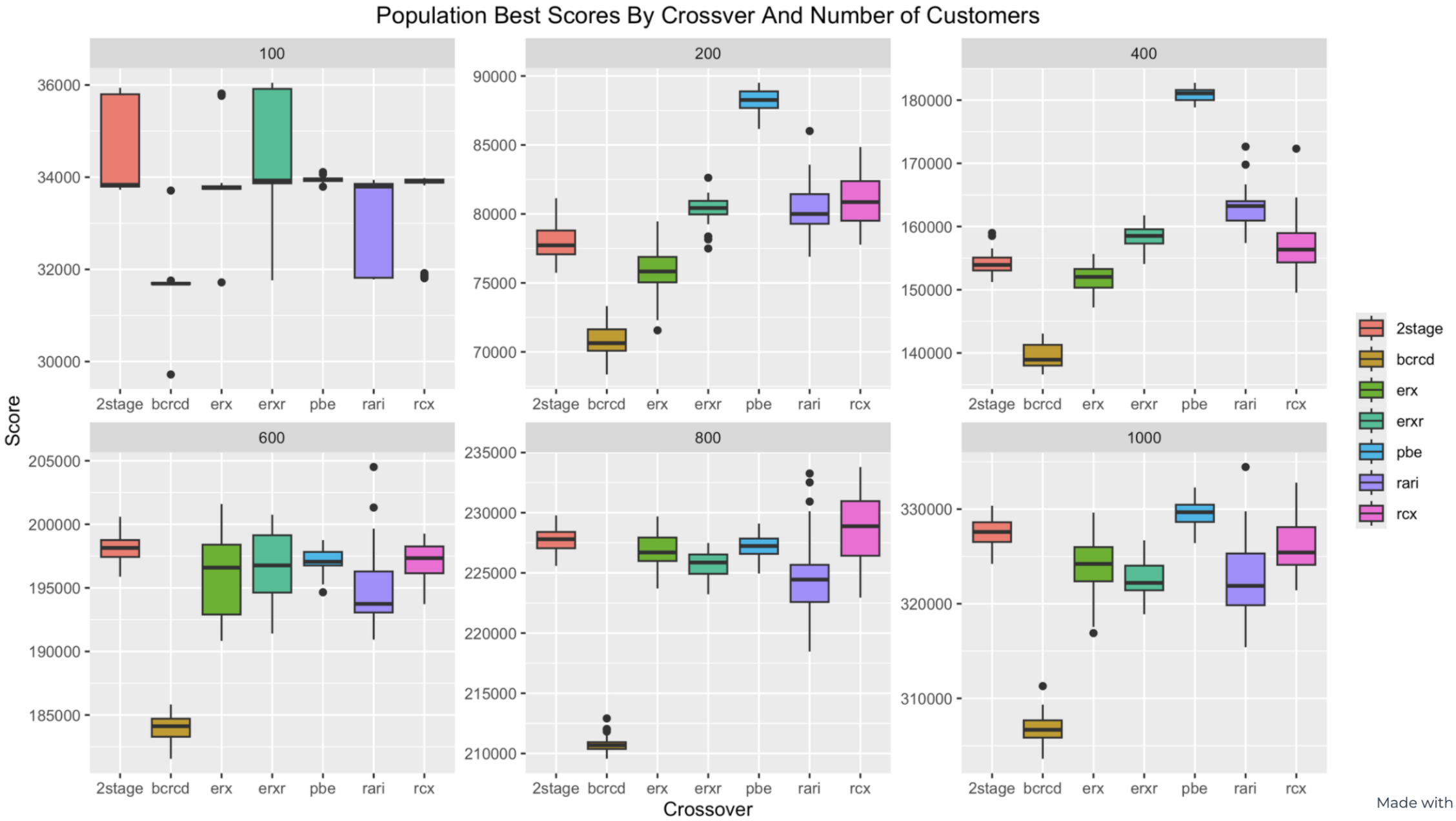
Results



Results



Results



Results

gen: 0	gen: 0
gens without improvement: 0	gens without improvement: 0
numSearchesWithoutImprovement: 0	numSearchesWithoutImprovement: 0
best score: 145992.772000	best score: 145992.772000
Average: 212053.909360	Average: 212053.909360
Standard Deviation: 7037.781247	Standard Deviation: 7037.781247
consumed seconds this time period: 0	consumed seconds this time period: 0
0 seconds	3 seconds
----------	----------
gen: 1	gen: 1
gens without improvement: 0	gens without improvement: 0
numSearchesWithoutImprovement: 0	numSearchesWithoutImprovement: 0
best score: 142677.402000	best score: 100924.626000
Average: 201349.092760	Average: 117305.667650
Standard Deviation: 13731.055664	Standard Deviation: 4440.418072
consumed seconds this time period: 0	consumed seconds this time period: 3

BCRCD VS PB

small changes and drastic turns

Summary

- Each crossover showcased a significant improvement from the first best solution through generations
- A dependence on the number of customers
- BCRCD is a clear choice as it outperformed all other crossovers in all cases
- Two step crossovers have a clear advantage over constructive
- Crossovers with regret-based insertion are often better for higher number of customers

References

[1]Shengcai Liu, Ke Tang, Xin Yao, Memetic search for vehicle routing with simultaneous pickup delivery and time windows

[2]A Memetic Algorithm for Large-Scale Real-World Vehicle Routing Problems with Simultaneous Pickup and Delivery with Time Windows. Ethan Gibbons and Beatrice M. Ombuki-Berman

[3]An Improved Memetic Algorithm for Large-Scale Real-World Vehicle Routing Problems with Simultaneous Pickup and Delivery with Time Windows, Ethan Gibbons, Beatrice Ombuki-Berman

[4]Yongliang Lu, Una Benlic, Qinghua Wu, An effective memetic algorithm for the generalized bike-sharing rebalancing problem

[5]Gaurav Srivastava, Alok Singh, Two evolutionary approaches with objective-specific variation operators for vehicle routing problem with time windows and quality of service objectives

[6]T. M. Stehling and S. R. de Souza, ”A Comparison of Crossover Operators Applied to the Vehicle Routing Problem with Time Window”

[7]Fanggeng Zhao, Dong Mei, Jiangsheng Sun and Weimin Liu, ”A hybrid genetic algorithm for the vehicle routing problem with simultaneous pickup and delivery”