IEEE WCCI 2020

IEEE World Congress on Computational Intelligence

Virtual Conference – July 19-24, 2020

Feature-independent Hyper-heuristics

For the 0/1 Knapsack Problem

Tecnológico de Monterrey

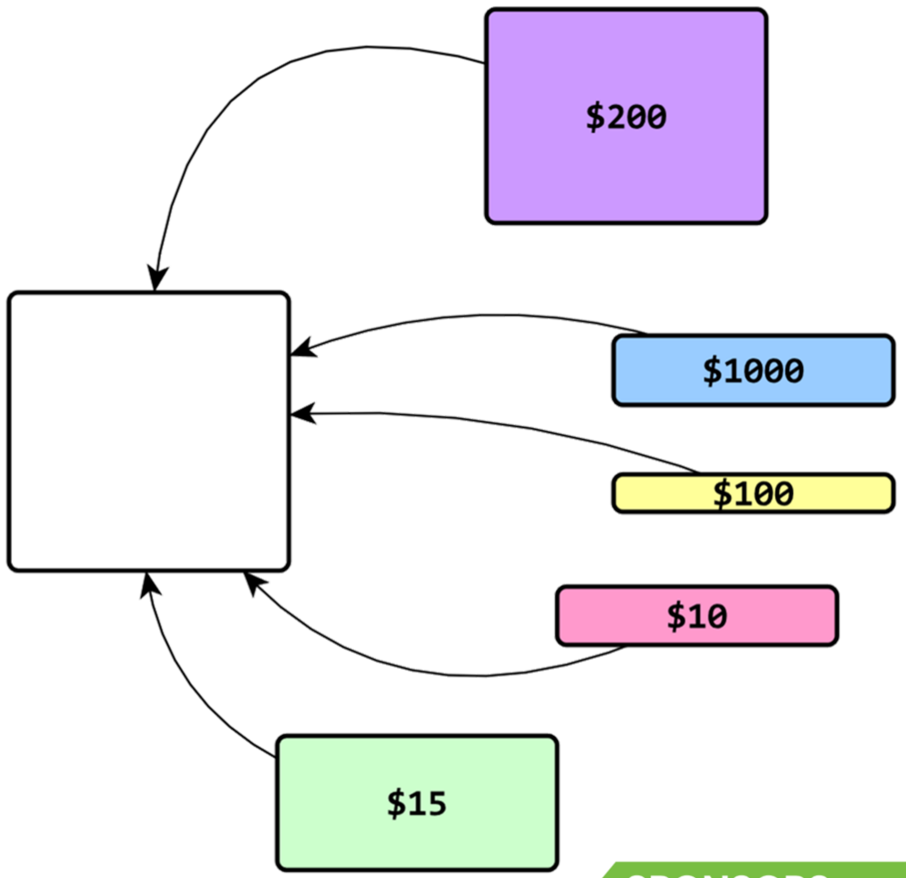
X. Sánchez-Díaz J.C. Ortiz-Bayliss I. Amaya

J. M. Cruz-Duarte S. E. Conant-Pablos H. Terashima-Marín

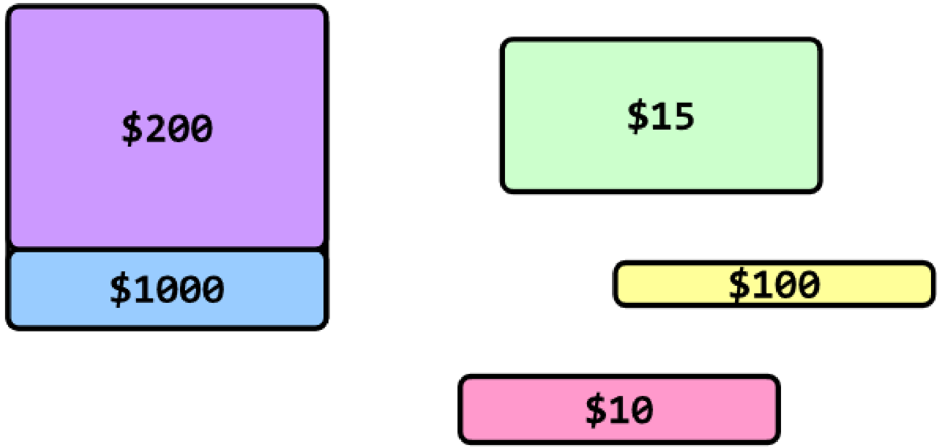
The binary knapsack problem

- Pack a selection of items inside a container with limited capacity.
- Optimisation problem looking for the subset of items which maximises the profit.
- Real-life applications in cargo loading, cutting stock, resource allocation and cryptography.

The binary knapsack problem



The binary knapsack problem



An idea

- Optimal solutions can be found in polynomial time if the knapsack capacity is not very large.

- Heuristic approaches provide good approximations.

- Mixing heuristics tends to yield even better results.

Mixing heuristics

- Browse available operators (heuristics) and select an appropriate one depending on the state of the problem:
  - × Need information about the current state,
  - × Decisions are usually domain-dependent.

A hyper-heuristic approach

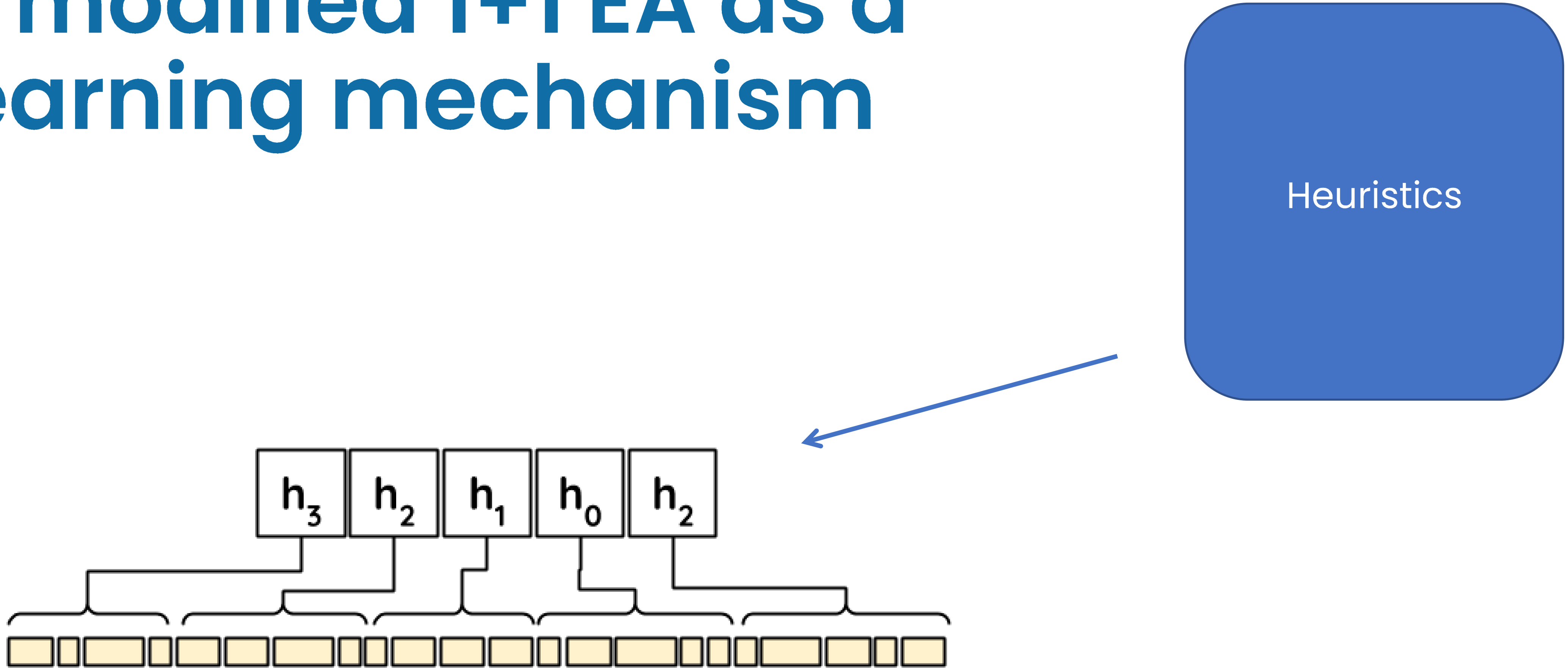
- Generate a sequence of packing heuristics and look for improvements iteratively.

- Use an Evolutionary Algorithm to optimise the profit of the knapsack.

Maximising the profit

1. Generate an individual
2. Clone it
3. Select a mutation operator and mutate the clone
4. Compare their evaluations: if equal or greater, keep the clone (to favour diversity)

A modified 1+1 EA as a learning mechanism



Packing heuristics

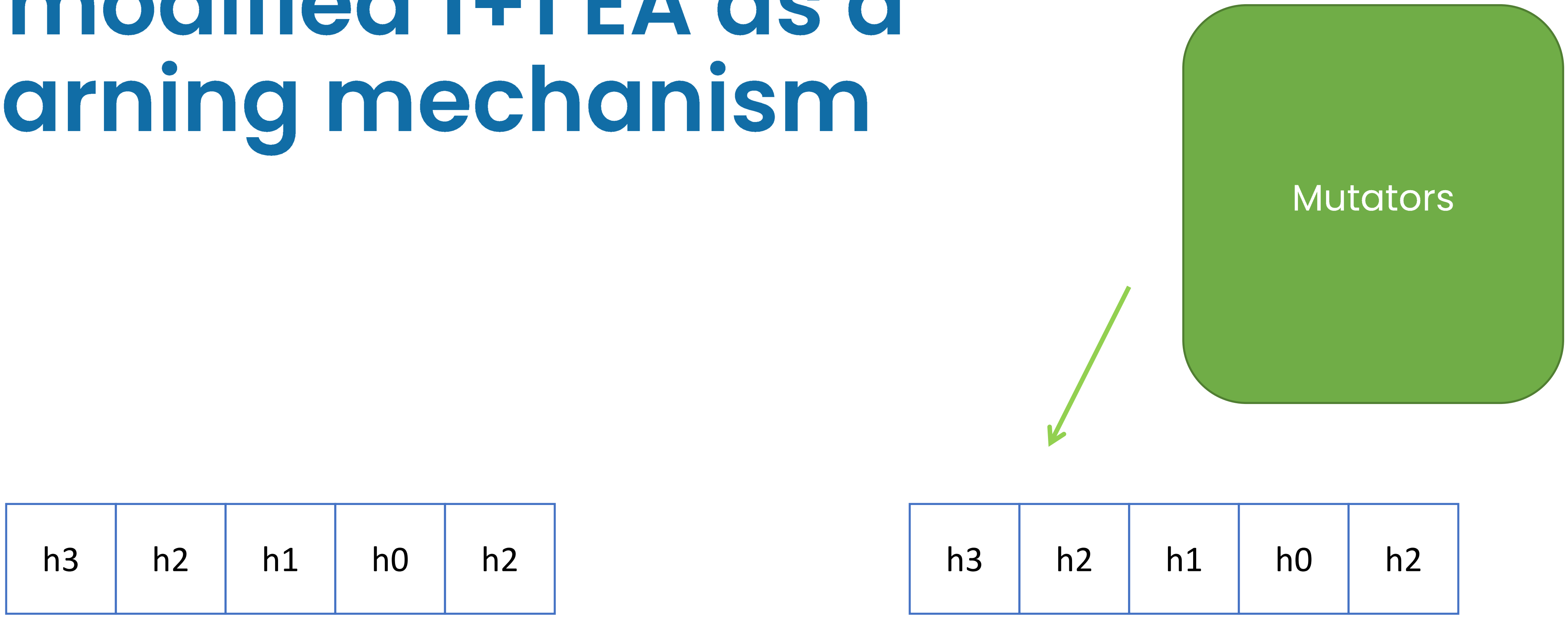
- Maximum profit (MaxP)
- Maximum profit per weight unit (MaxPW)
- Minimum weight (MinW)
- Default order (Def)

A modified 1+1 EA as a learning mechanism

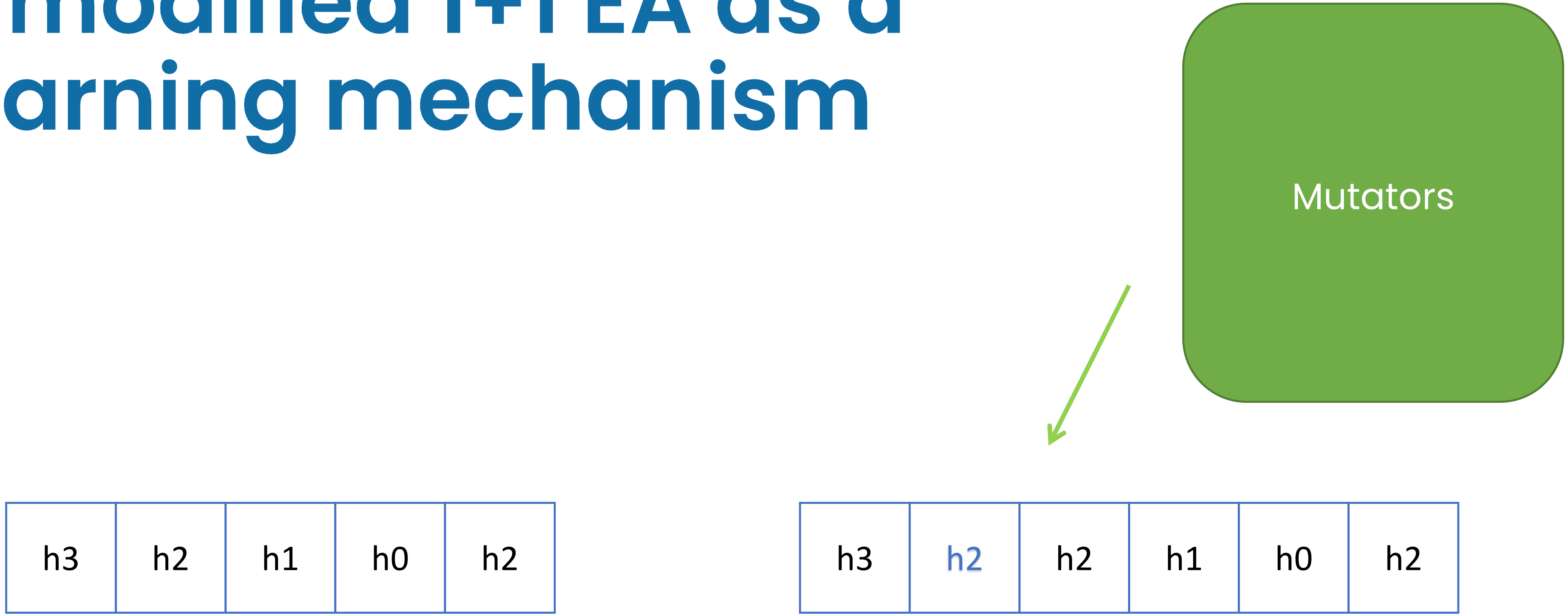
h3	h2	h1	h0	h2

h3	h2	h1	h0	h2

A modified 1+1 EA as a learning mechanism



A modified 1+1 EA as a learning mechanism



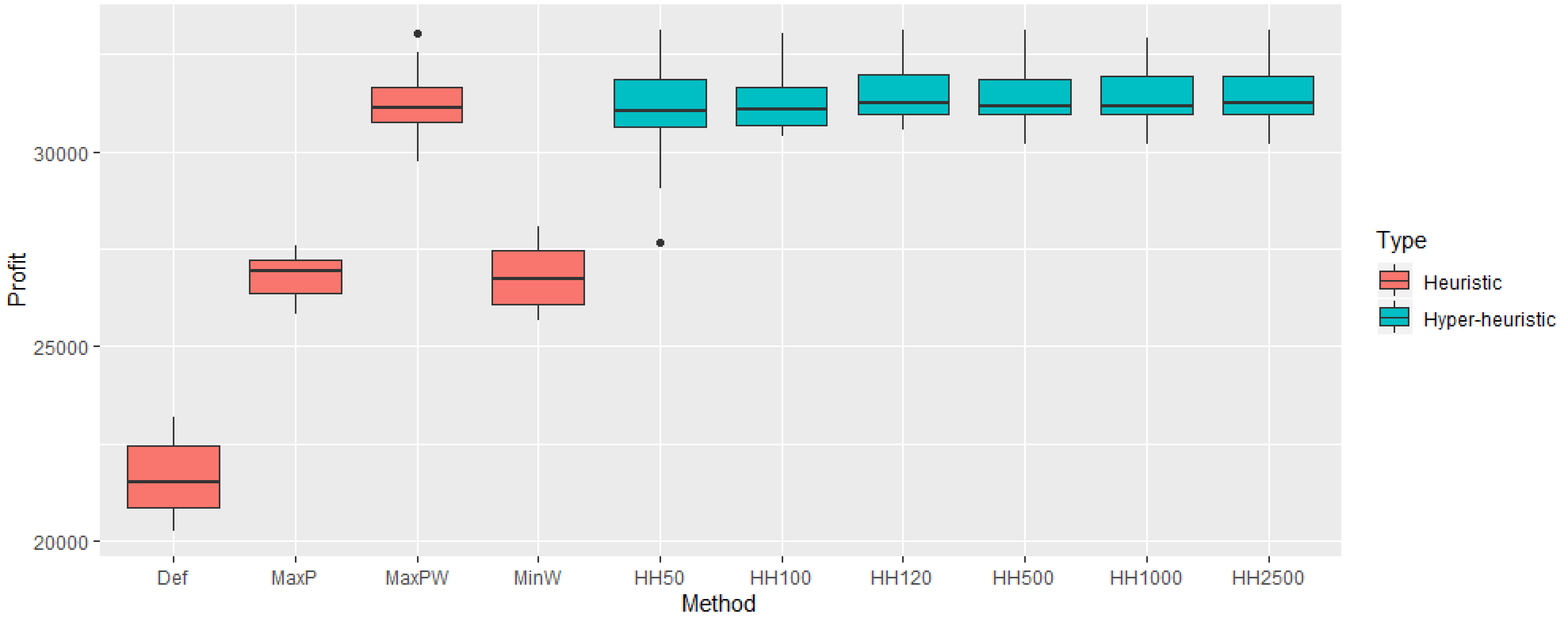
Mutation operators

- Select a random point in the chromosome and:
  - Add a gene
  - Remove a gene
  - Flip a gene
  - Flip a gene depending on neighbours
- Select two random points and:
  - Flip genes
  - Flip genes depending on neighbours
  - Swap Genes

Experiments

- 100 knapsack synthetic instances of 20 items and 50 units of capacity.
- Balanced set: each packing heuristic represents the best option for roughly 25% of the set.
- 60/40 split for training/testing for 10 runs, under six different scenarios (50, 100, 120, 500, 1000 and 2500 iterations.)

Profit per method



Hypothesis testing

- *Is the mean profit of two methods equal?*

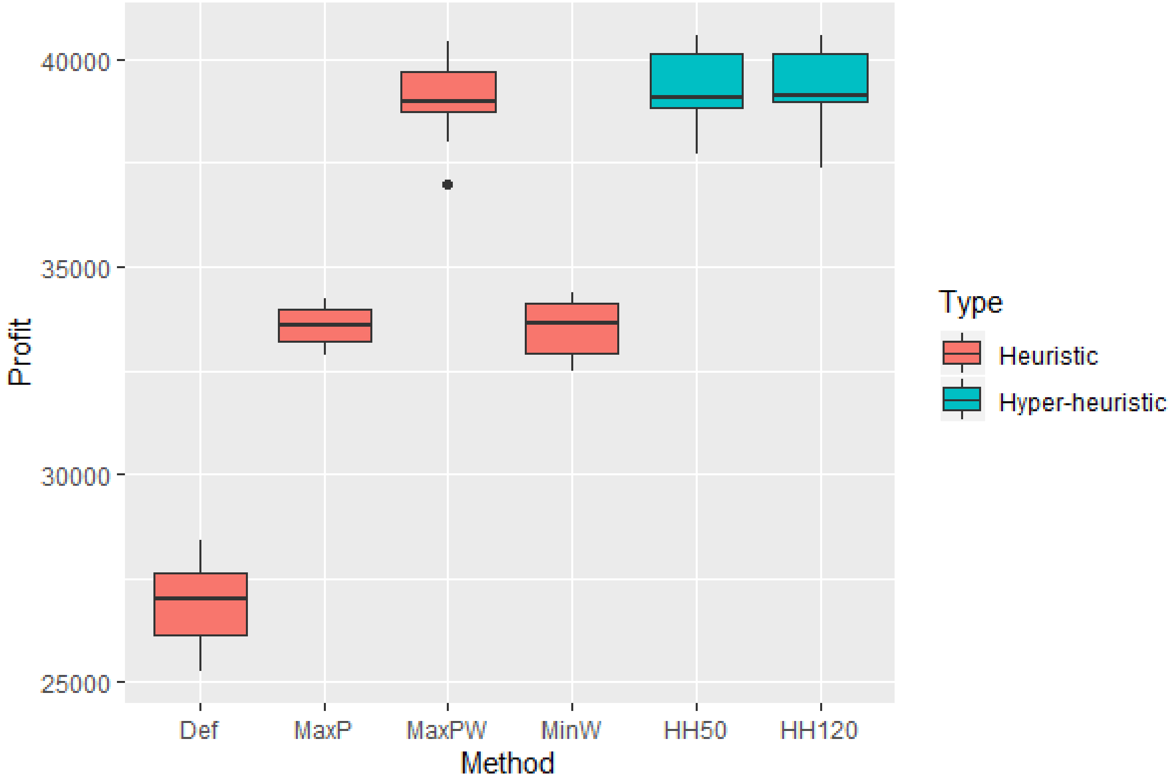
- Two-tail t-test comparing on pairs of Hyper-heuristic methods and best heuristic in isolation (MaxPW.)

- HH120, HH500, HH1000 and HH2500 are in general better than MaxPW for these type of instances.

Additional experiments on 50 & 120 iterations

- Tested both hyper-heuristic methods again, using different train/test splits:
  - 50/50
  - 30/70

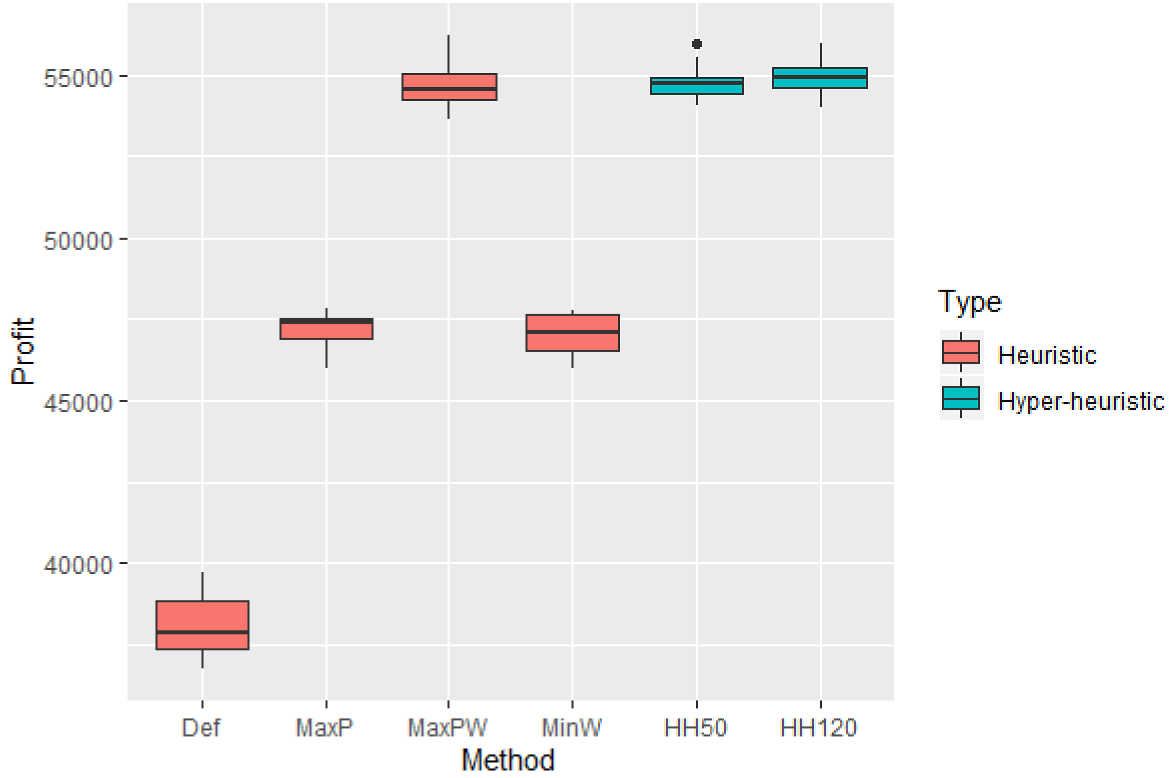
Additional experiments on 50 & 120 iterations



- 50/50 split

Additional experiments on 50 & 120 iterations

- 30/70 split



Frequency analysis

- Throughout the evolution of all 10 hyper-heuristics of 120 iterations we recorded all two-heuristic sub-sequences.

h3	h2	h1	h0	h2

- 2-heuristic sub-sequences:
  - h3, h2
  - h2, h1
  - h1, h0
  - h0, h2

Frequency analysis

- The most common sub-sequence was MaxPW-MaxPW in 90% of the hyper-heuristics.
- MaxPW + MaxP was second place in 8/9 hyper-heuristic.
- Sub-sequences including the default ordering (Def) were the least used: barely 5% of all sub-sequences analysed.

Implications

- Hyper-heuristics obtained, in general, a higher profit than heuristics alone.

- No problem characterisation was required.

- Feeding heuristics and mutation operators into the model allows to explore different domains in optimisation.

Thank you!

Xavier F. C. Sánchez-Díaz <sax@tec.mx>
Research Group with Strategic Focus in Intelligent Systems
School of Engineering and Sciences
Tecnológico de Monterrey