PA184 - Heuristic Search Methods

Lecture 10 – New Ideas and Future Research

- Self-adaptation
- Asynchronous Cooperation
- Self-assembly

Learning outcomes:

- Get an insight into some recent new ideas proposed in the literature of heuristic search methods.
- Identify some examples of self-adaptation in meta-heuristic algorithms.
- Understand the main principles of asynchronous local search as a methodology to extend existitng single-solution approaches.
- Get an insight into the use of self-assembly as a tool for the automated design of heuristic methods. This is still a very new research direction with only promising results.

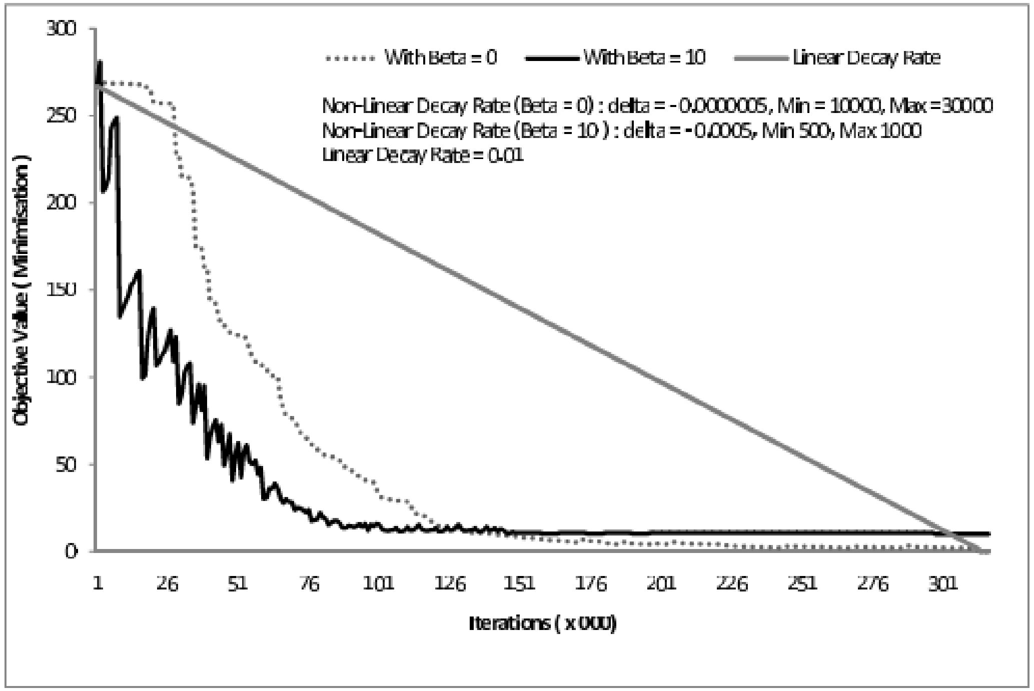
SELF-ADAPTATION

Non-linear Great Deluge with Reinforcement Learning

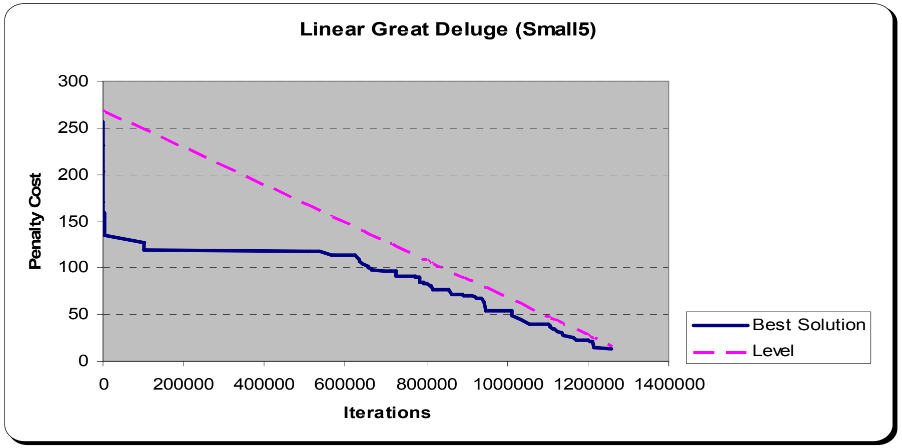
Contrary to the traditional linear form, the non-linear decay rate of the water level reacts to the value of the current best solution, i.e. the decay rate is driven by the search.

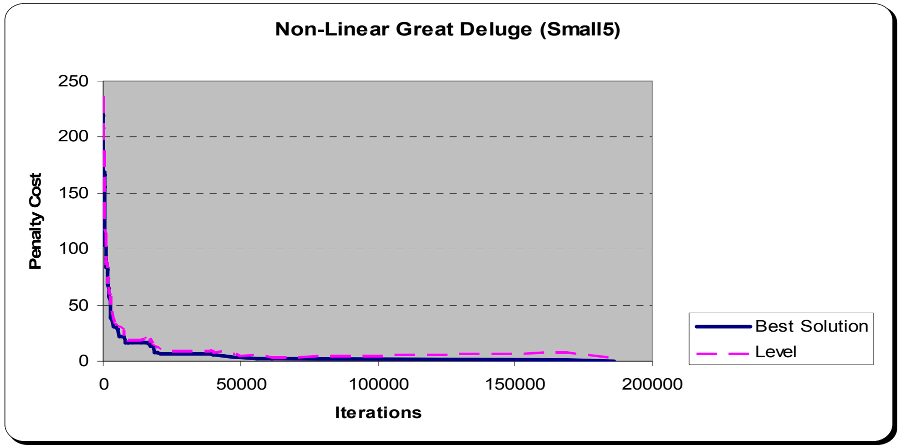
$$B = B \times \left(\exp^{-\delta(rnd[\min,\max])}\right) + \beta$$

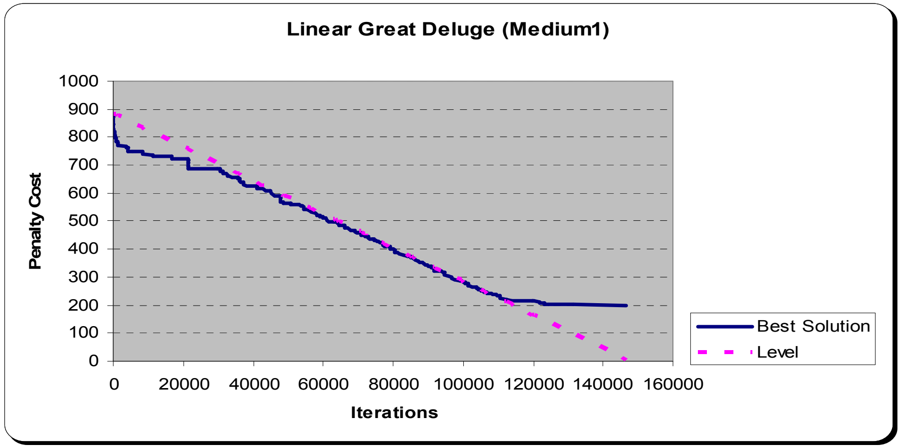
For a minimisation problem, candidate solutions are accepted if the objective function value is below the current water level.

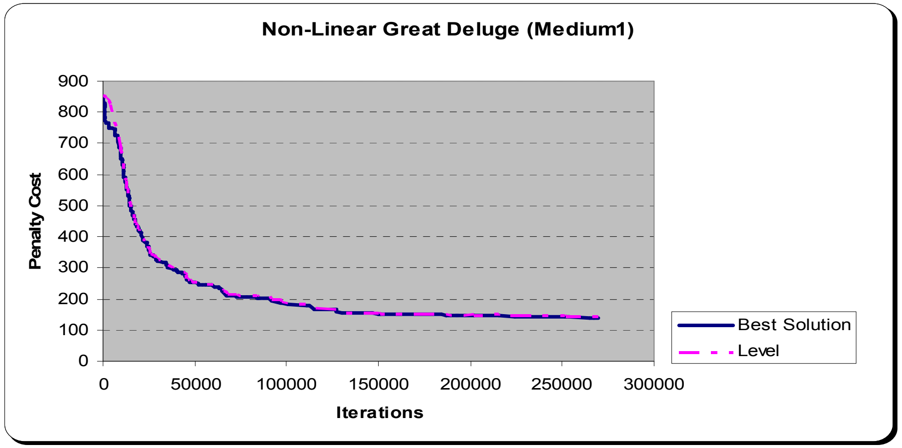


The adaptive water level allows a more effective search for various difficult instances of the UCTT problem.









A simple learning mechanism guides the selection of low-level heuristics (local moves) during the search.

The learning mechanism tunes the priorities of the low-level heuristics as the search progresses in an attempt to learn the low-level heuristic to use at each point of the search.

Static Memory

$$p_i = \frac{w_i}{\sum_{i \in H} w_i}$$

Dynamic Memory

$$p_i = \frac{w_i + w_{\min}}{\sum_{i \in H} w_i + w_{\min}}$$

$$w_{\min} = \min\{0, w_i\}$$

$$w_{ih} = \sum_{j=k}^{h} \sigma^i \Re_{ij}$$

$$\Re_{ij} \begin{cases} r & \text{if } \Delta < 0 \\ -r & \text{if } \Delta > 0 \\ \Im & \text{if } \Delta = 0 \text{ and new solution} \\ \Im & \text{if } \Delta = 0 \text{ and no new solution} \\ 0 & \text{if not selected} \end{cases}$$

The non-linear and floating decay rate and the learning mechanism for low-level heuristic selection, improve the performance of the Great Deluge algorithm when solving instances of the UCTT problem.

Instance	NLGDHH-SM		NLGDHH-DM		EGD		NLGD	ENLGD	GD
	Best	Avg	Best	Avg	Best	Avg	Best	Best	Best
S1	0	0.5	0	2.5	0	0.8	3	0	17
S2	0	0.65	0	1.9	0	2	4	1	15
S3	0	0.20	0	2.05	0	1.3	6	0	24
S4	0	1.5	0	2.85	0	1	6	0	21
S5	0	0	0	0.85	0	0.2	0	0	5
M1	51	60.1	54	139	80	101.4	140	126	201
M2	48	59.05	67	78.2	105	116.9	130	123	190
M3	60	83.9	84	115.45	139	162.1	189	185	229
M4	47	54.9	60	72.05	88	108.8	112	116	154
M5	61	84.15	93	112.8	88	119.7	141	129	222
L	731	888.65	917	1035.25	730	834.1	876	821	1066

The best results so far for most of these instances of the UCTT problem are obtained by the NLGD with Reinforcement Learning.

	NLGDHH-SM LP=1000	NLGDHH-SM LP=2500	NLGDHH-SM LP=5000	NLGDHH-DM	Best Known
S1	**0**	**0**	**0**	**0**	**0** (VNS-T)
S2	**0**	**0**	**0**	**0**	**0** (VNS-T)
S3	**0**	**0**	**0**	**0**	**0** (CFHH)
S4	**0**	**0**	**0**	**0**	**0** (VNS-T)
S5	**0**	**0**	**0**	**0**	**0** (MMAS)
M1	51	**38**	42	54	80 (EGD)
M2	48	**37**	44	67	105 (EGD)
M3	60	61	**60**	84	139 (EGD)
M4	47	41	**39**	60	88 (EGD)
M5	61	61	**55**	93	88 (EGD)
L1	731	638	713	915	**529**(HEA)

Restricted Assortative Mating

- Incorporate some degree of restriction when recombining parents in evolutionary algorithms
- Use mating radius to control the exploration and/or exploitation

Natural Selection

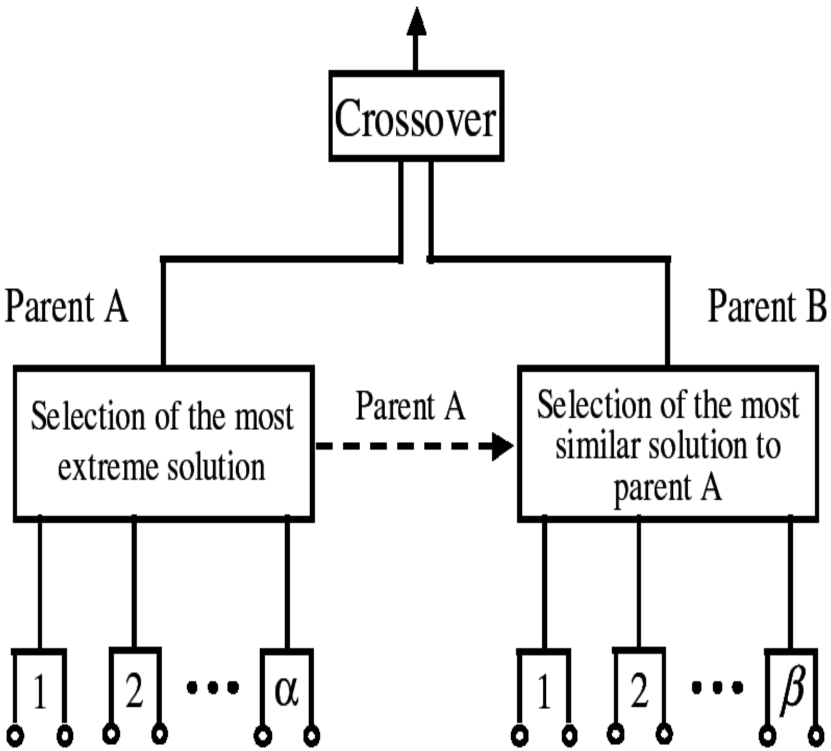
- Selection schemes:
  - Fitness proportionate selection
  - Rank-based selection
  - Tournament selection
- Parents chosen based on their fitness
- Passively assigned mates
- Operate in objective space

Sexual Selection

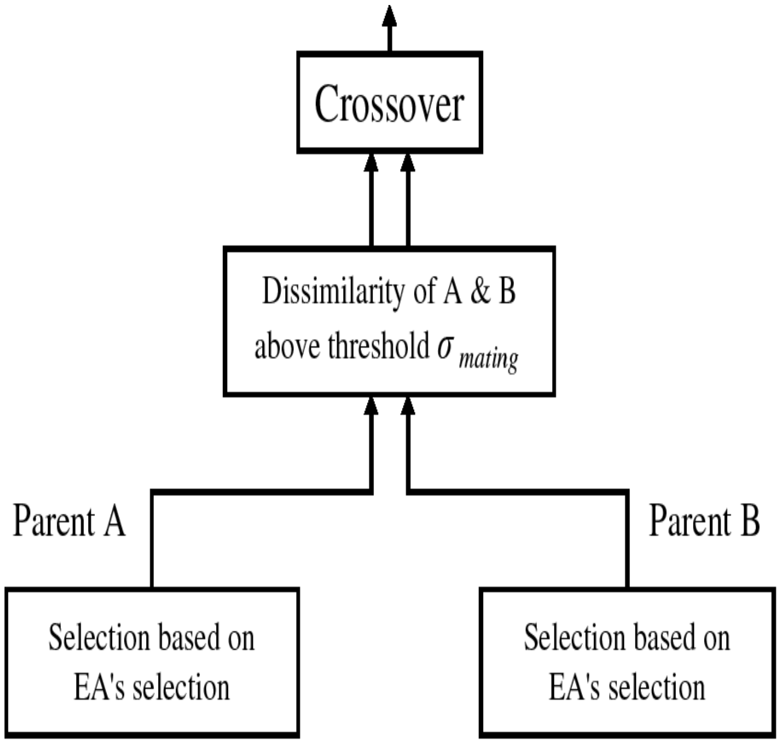
- Selection schemes:
  - Ancestry selection
  - Assortative mating
  - Gender selection
- 2nd parent chosen w.r.t 1st parent
- Actively choose mates
- Operate in objective/decision space

The proposed restricted assortative mating strategy adapts to the similarity between potential parents and also to the diversity of the current population. The method can be incorporated into other EAs.

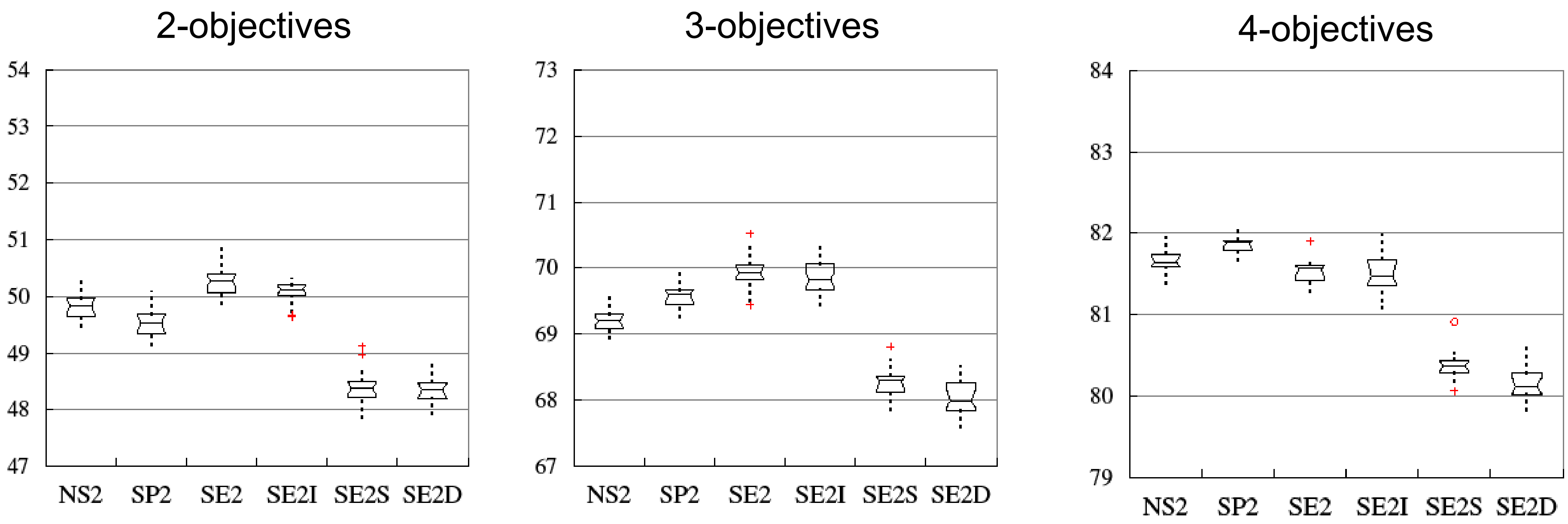
Ishibuchi's strategy



Proposed strategy



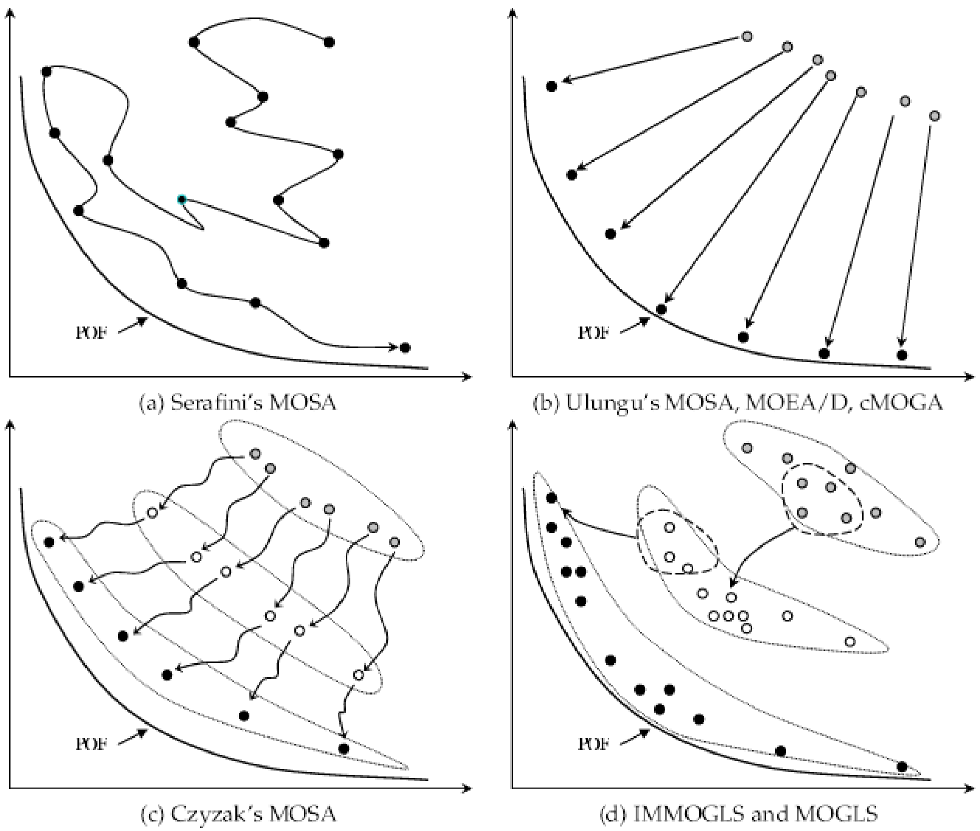
- Experiments on 2-, 3- and 4-objecitve knapsack problems
- NSGA2, SPEA2, SEAMO2
- Ishibuchi mating strategy and restricted assortative mating strategy incorporated into SEAMO2
- The mating ratio σ_{mating} is fixed or automatically adjusted according to the population diversity (decision space)



EMOSA – an Adaptive Algorithm for MOCO

Employs simulated annealing for the optimisation of each sub-problem and adapts search directions for diversifying non-dominated solutions.

Various strategies used to define search directions in existing multi-objective meta-heuristics based on decomposition.



EMOSA Approach

Step 0: Initialisation

Produce *pop* well-distributed weight vectors. For each weight vector, an initial solution is generated randomly. Update the external population (EP) with the nondominated solutions in the initial population. Set T:=T0.

Step 1: Local Search and Competition

For each individual x in the current population, repeat the follow steps K times.

1.1 find a neighbouring solution y
1.2 update the EP with y if it is not dominated by x
1.3 replace x with the probability P(w, x, y, T)

Compete with the other solutions with similar weight vectors to that of x

Step 2: Temperature Change

Lower the temperature value by using T:=T – alpha. If T>=Tc, adapt the weight vector of each individual in the population, otherwise, go to Step 4.

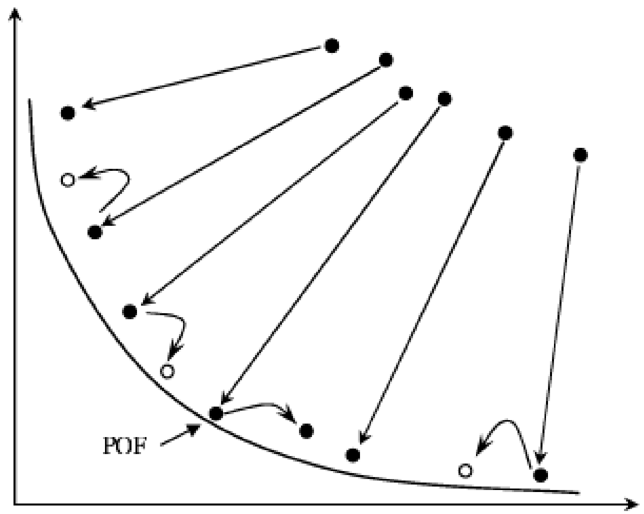
Step 3: Direction Adaptation

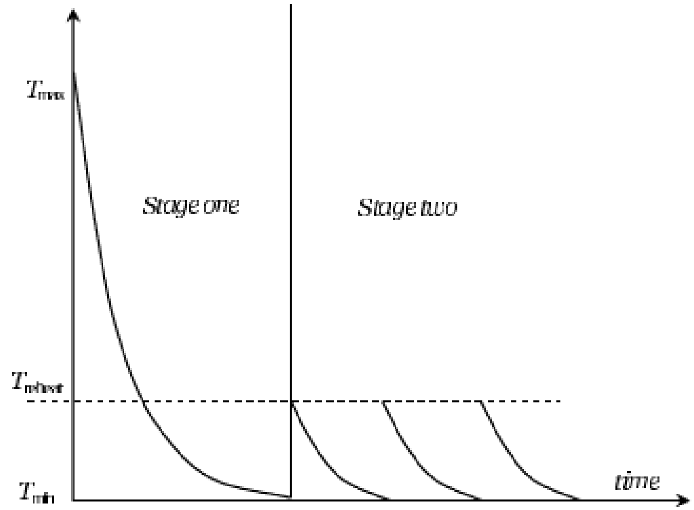
Modify each weight vector to move the current solution away from its nearest nondominated neighbours in the population.

Step 4: Stopping Criteria

If T<Tmin, then stop and return EP, Otherwise go to Step 1.

Faster annealing schedule with re-heating when searching close to the Pareto front and strategy to adapt weight vectors when simulated annealing enters the only improvement phase.

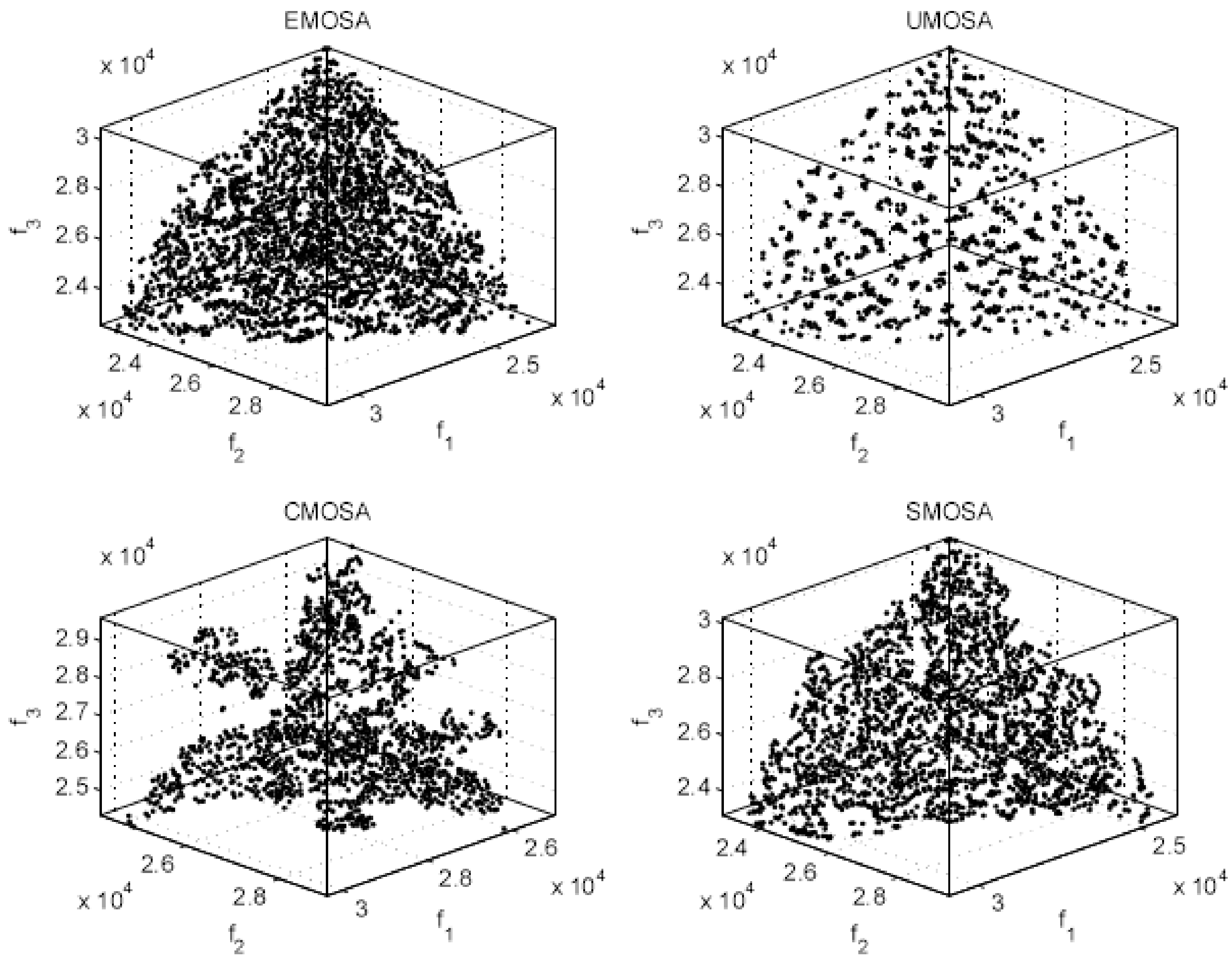




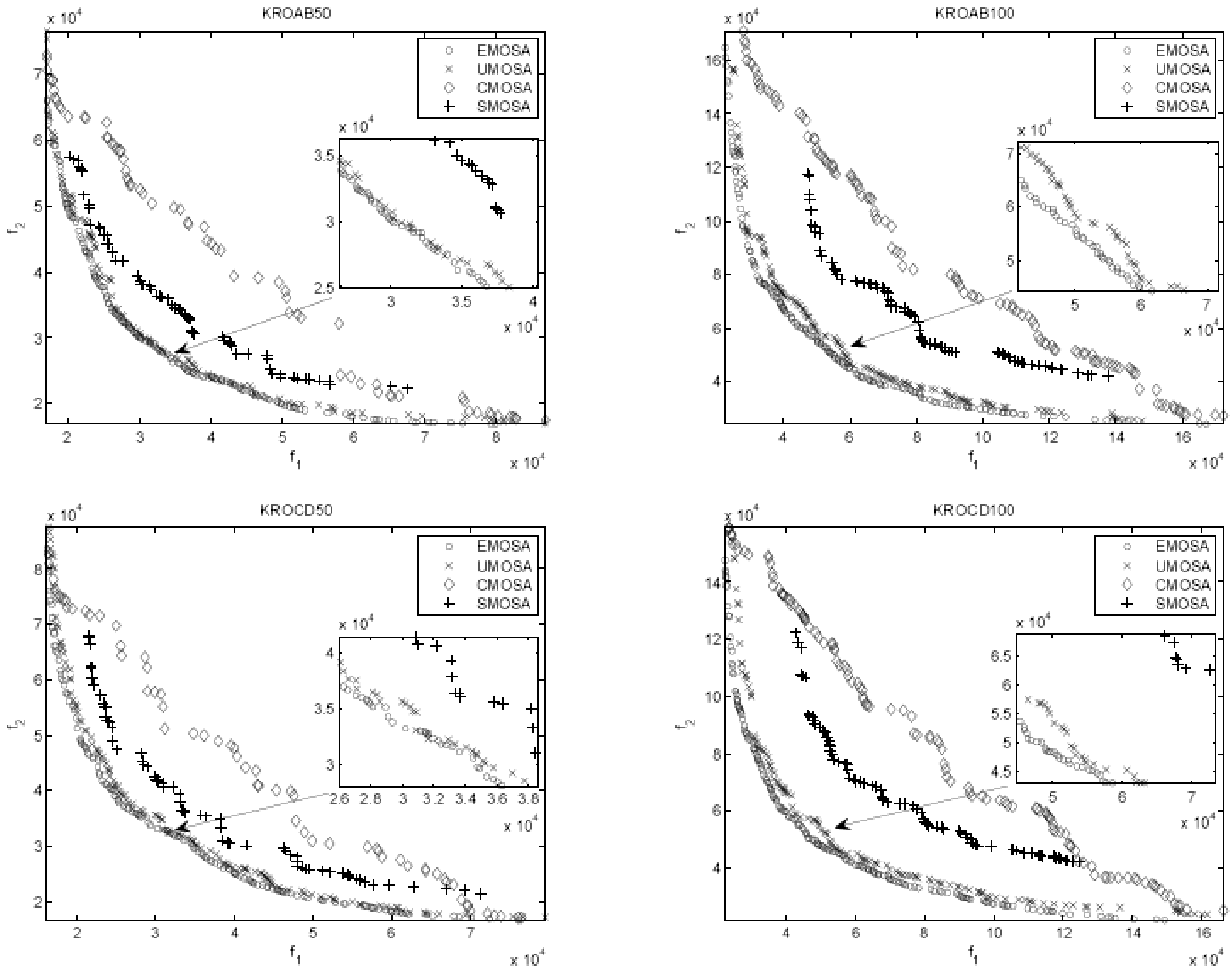
Procedure 3 AdaptWeightVector(s)

find the closest non-dominated neighbor $x^{(t)}, t \in \{1, \ldots, Q\}$ to $x^{(s)}$
$A \leftarrow \{\lambda \in \Theta | \text{dist}(\lambda^{(s)}, \lambda^{(t)}) < \text{dist}(\lambda, \lambda^{(t)}) \text{ and } \text{dist}(\lambda, \lambda^{(s)}) \leq \text{dist}(\lambda, \Lambda)\}$
if A is not empty **then**
    $\lambda^{(s)} \leftarrow \text{argmax}_{u \in A} \text{dist}(u, \lambda^{(s)})$;
end if
return $\lambda^{(s)}$

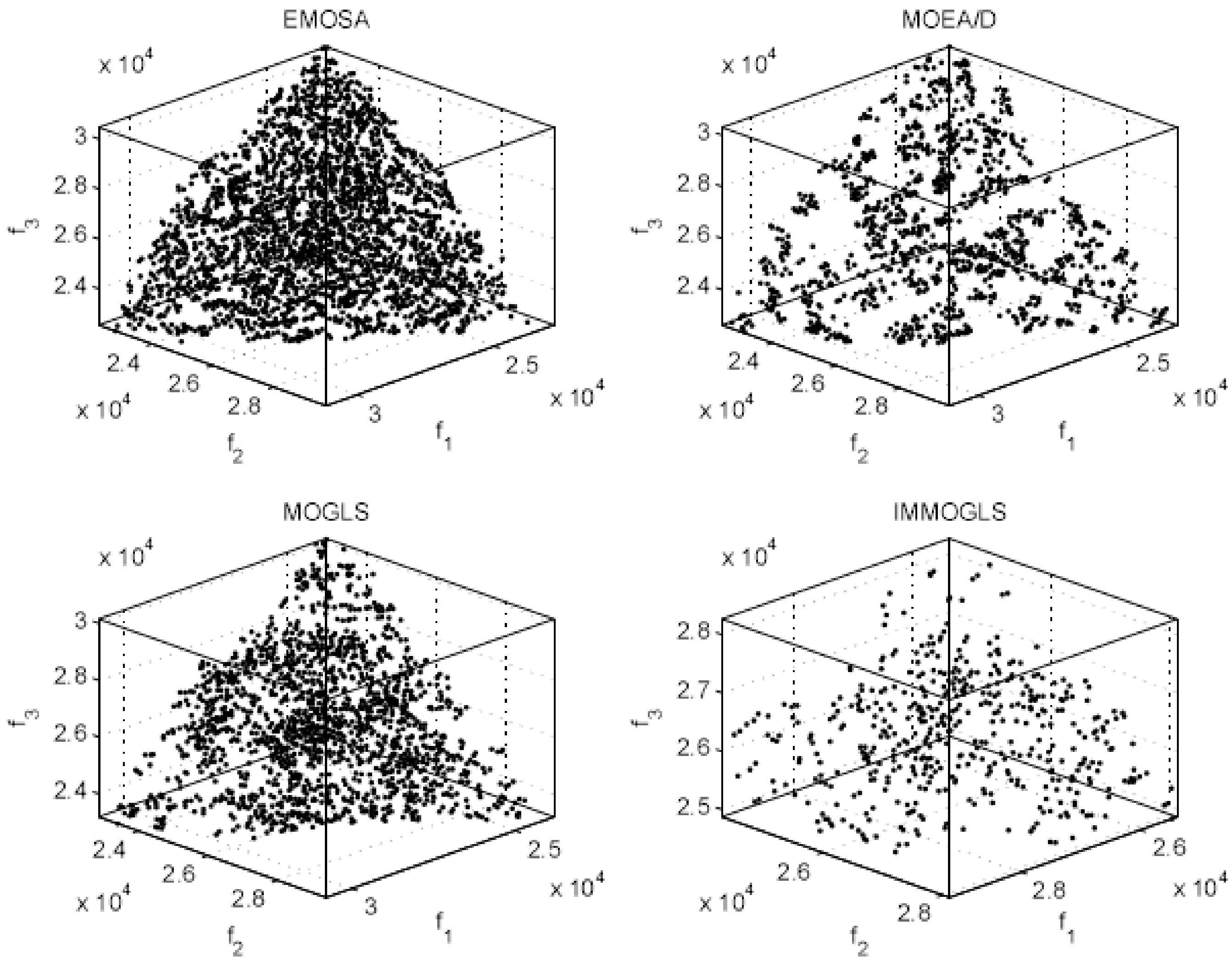
- Some results on 3-bjective knapsack problem instances
- EMOSA outperforms other MOSA-like algorithms



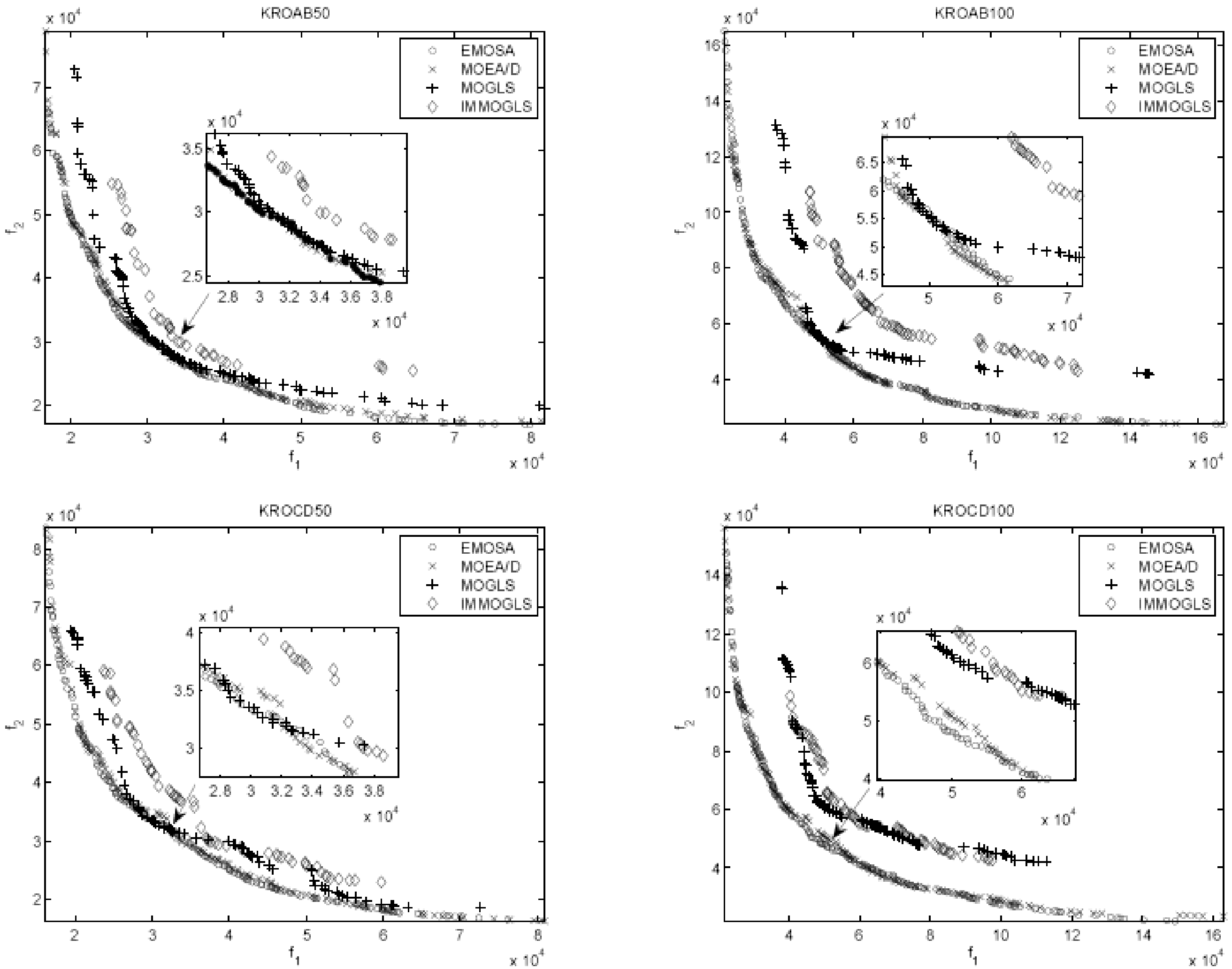
- Some results on 2-bjective TSP problem instances
- EMOSA outperforms other MOSA-like algorithms



- Some results on 3-bjective knapsack problem instances
- EMOSA outperforms other MOMA-like algorithms



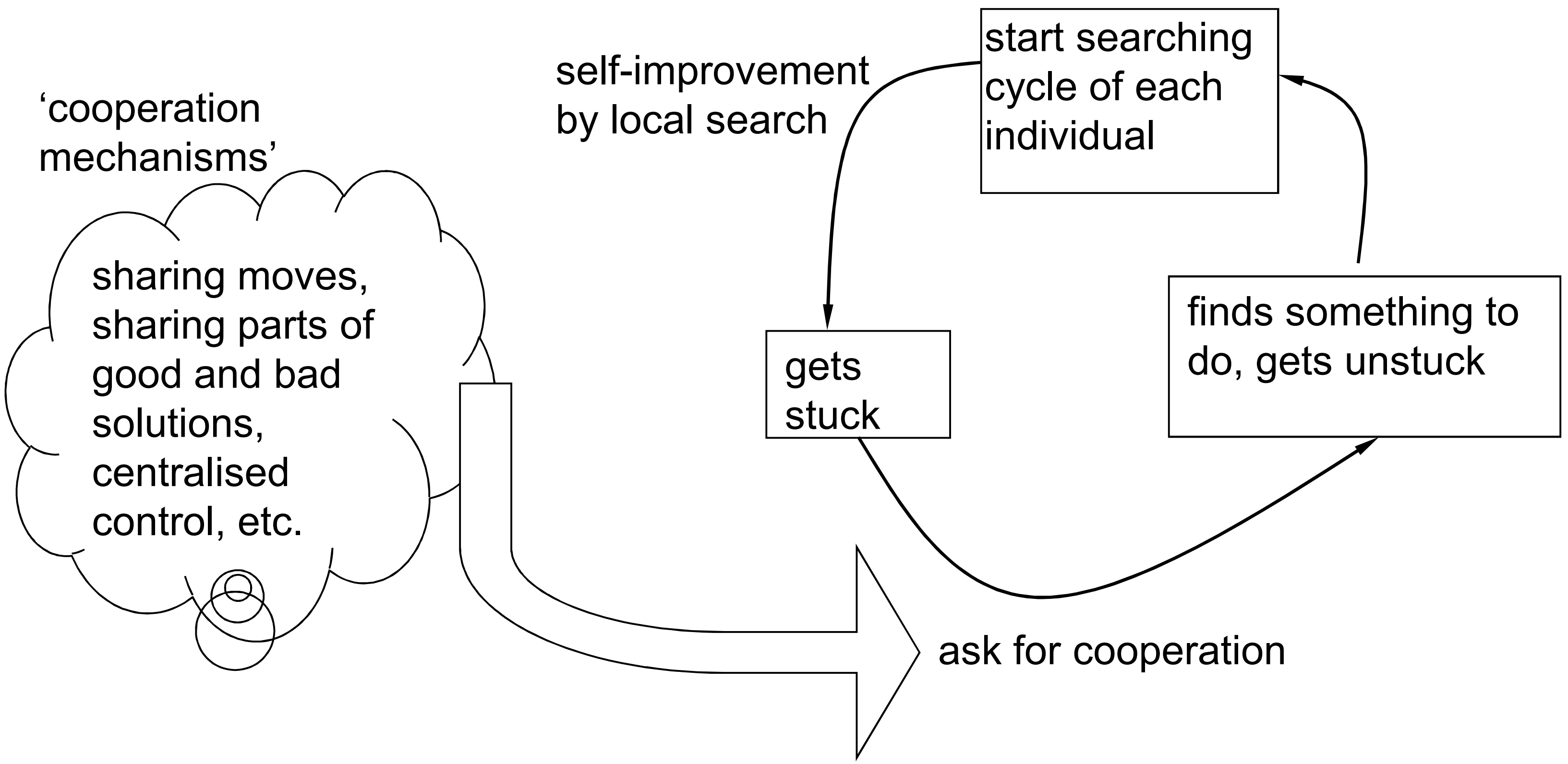
• Some results on 2-bjective TSP problem instances
• EMOSA outperforms other MOMA-like algorithms.



ASYNCHRONOUS COOPERATION

Cooperative Local Search

Simple cooperative strategy based on memory to improve local search procedures.



[Memory-based cooperation mechanism]() based on matrices of genes:

Matrices M_A (attractive) and M_T (tabu) of size n x m

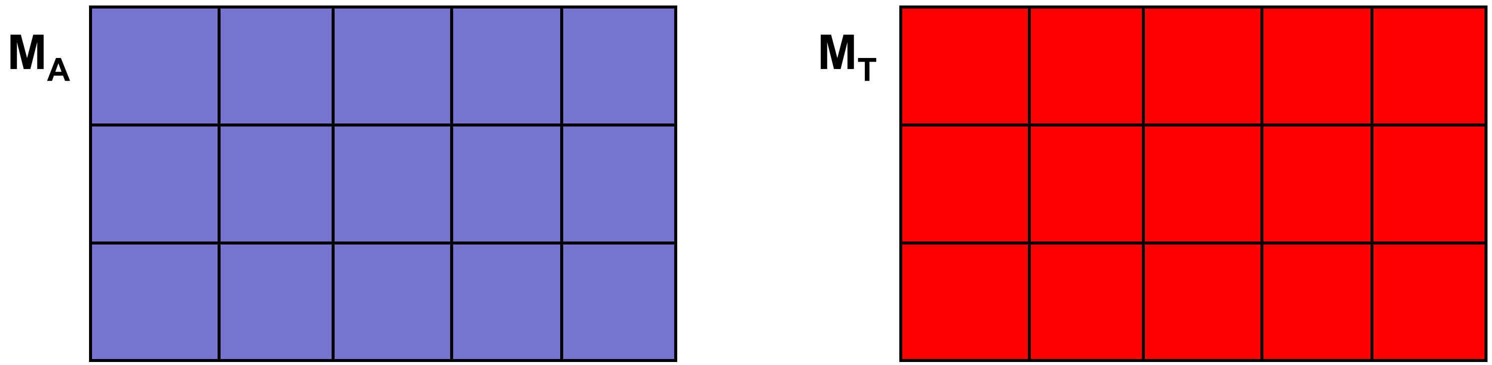
Instead of memorising move attributes as in tabu search, these shared matrices keep track of the frequency of each 'gene' seen during search.

M_A is updated when a move improves a solution

$$M_A(i,j) = M_A(i,j) + 1$$

M_T is updated when a move worsens a solution

$$M_T(i,j) = \textit{current iteration} + \textit{tenure}$$



The information stored in M_A and M_T is used in two ways:

• **Information sharing**. All explorers update the matrices of genes. When an explorer is stuck, it uses the information stored in M_A to modify the current solution (maximum *n*/20 changes).

• **Heuristic heavy mutation**. A heavy disruption of the solution is carried out. It removes the most penalised entities from their assigned locations (maximum *n*/5 removals). Then, each of these entities is assigned a new location avoiding those marked tabu in M_T.

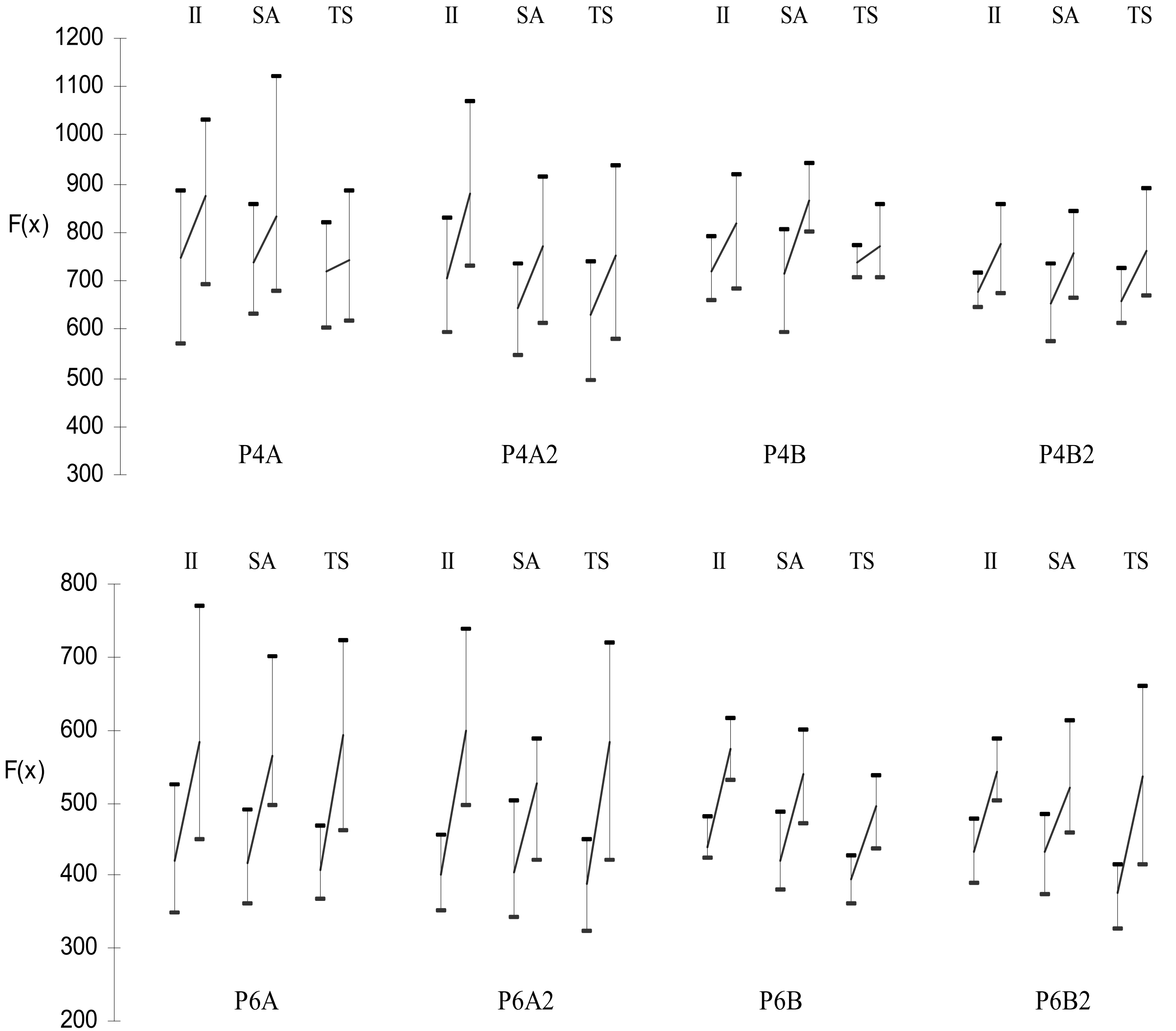
Step 1. Generate population of ‘explorers’ LS_{SS}

Step 2. Self-improvement phase using ‘information sharing’

Step 3. Random variation of population using ‘heavy mutation’

Step 4. If terminate condition Stop, otherwise go to Step 2.

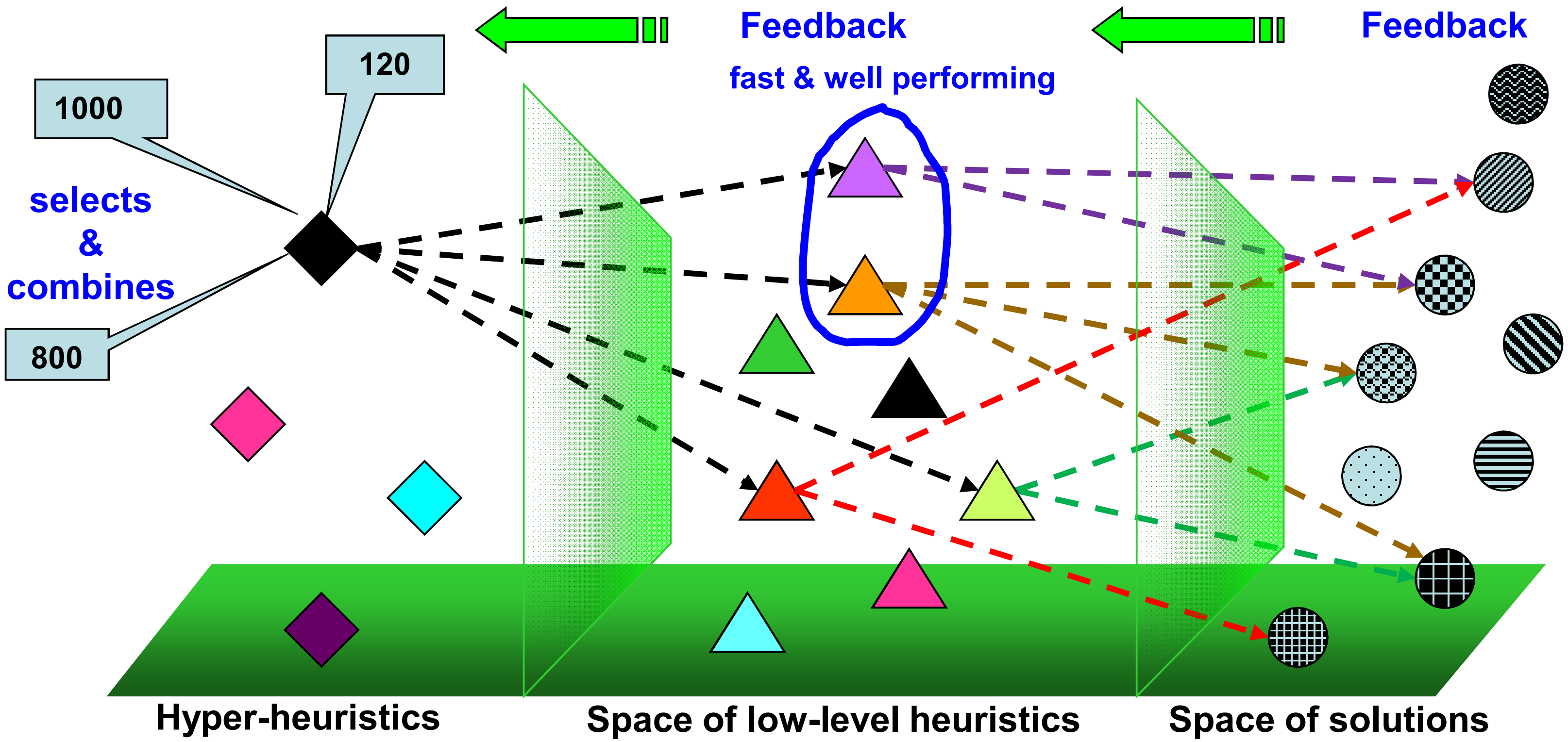
Some results on the Office Space Allocation Problem



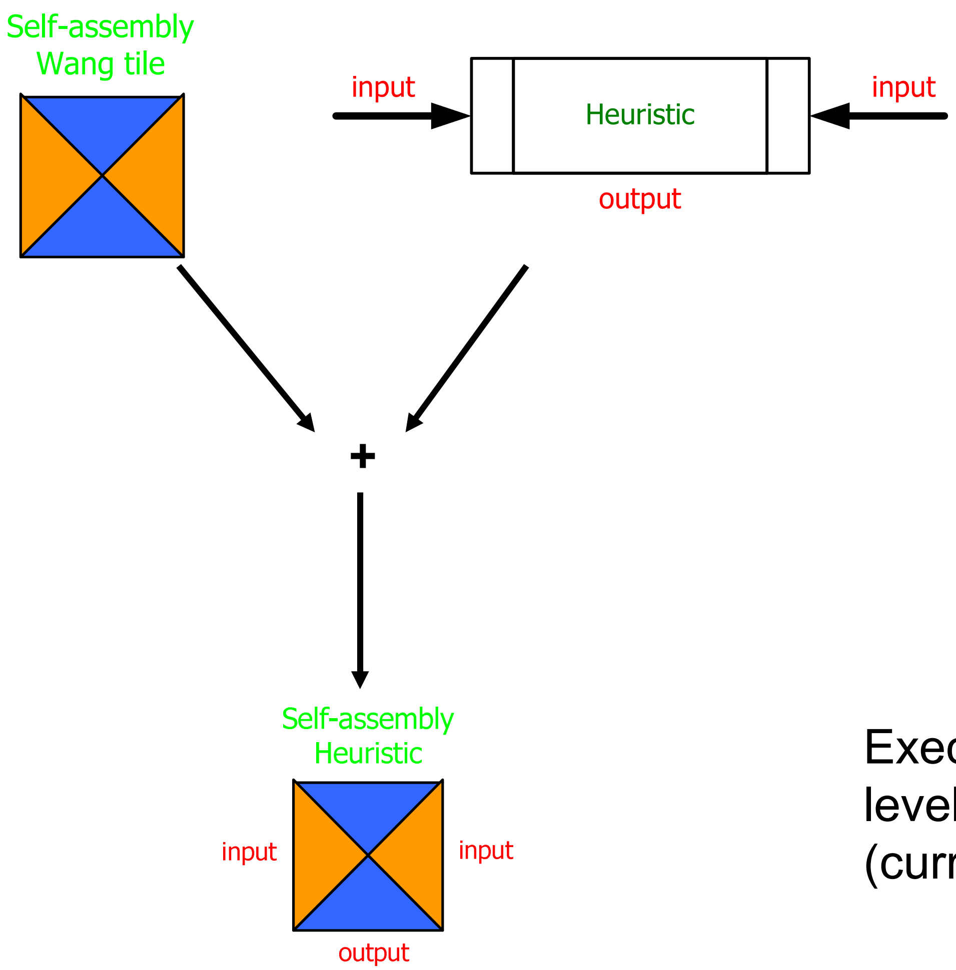
SELF-ASSEMBLY

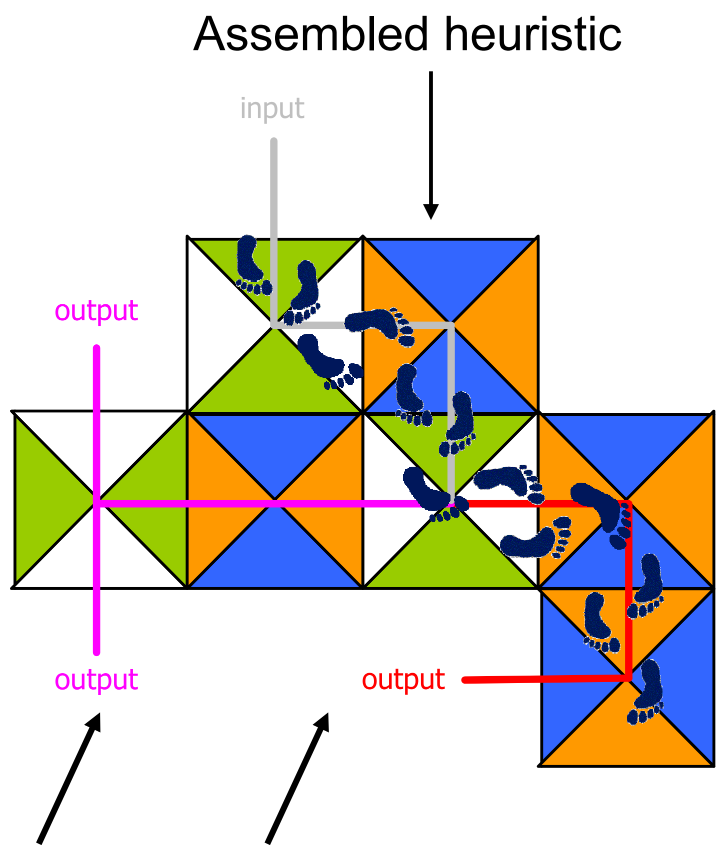
Hyper-heuristics

- Search methodologies that select and combine low-level heuristics to solve hard computational problems.
- Domain-independent strategies that operate in the space of heuristics.
- Manufacture unknown heuristics which are fast and well performing.



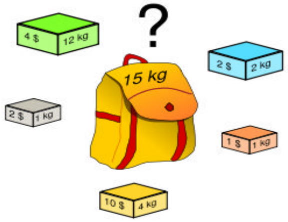
Heuristics Self-assembly System

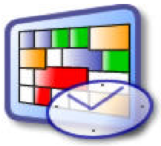




Execution threads (sequences of low level heuristics) by random walk (currently).

Combinatorial Optimisation Problems



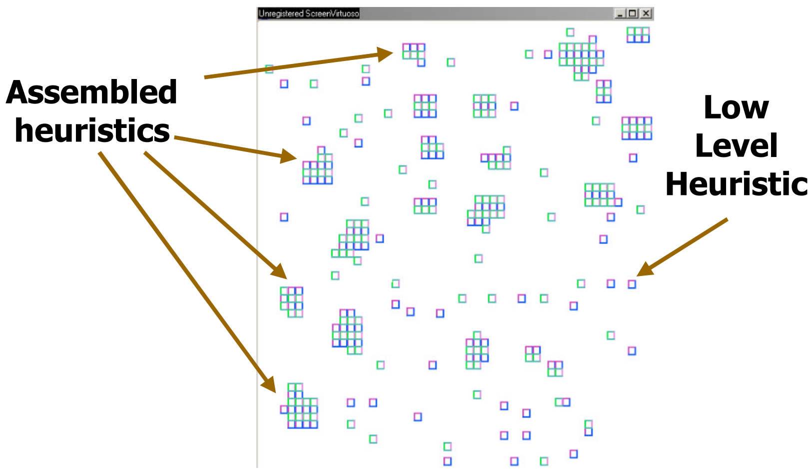


$$(X1 \text{ or } X2 \text{ or } \overline{X3})$$
$$(\overline{X1} \text{ or } \overline{X2} \text{ or } X3)$$
$$(\overline{X1} \text{ or } \overline{X2} \text{ or } \overline{X3})$$
$$(\overline{X1} \text{ or } X2 \text{ or } X3)$$

Is it possible to automatically design the [correct assembly of a heuristic](), the execution threads of which optimise a given problem instance?

If the answer is YES, is it possible to apply the same methodology to a different problem?

Self-assembly Heuristics



Methodology

Execution threads analysis

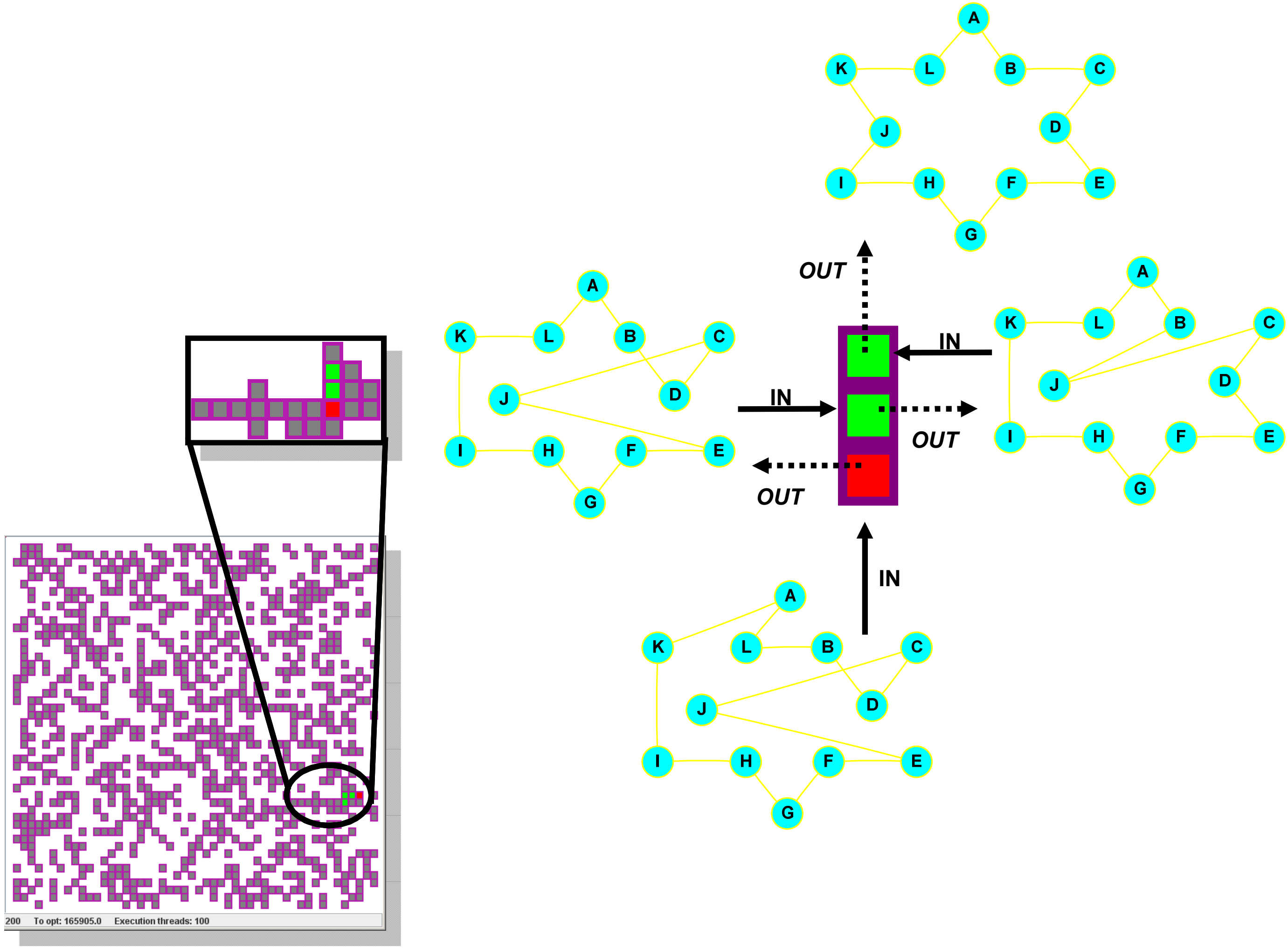
+

Assembled heuristics characterisation

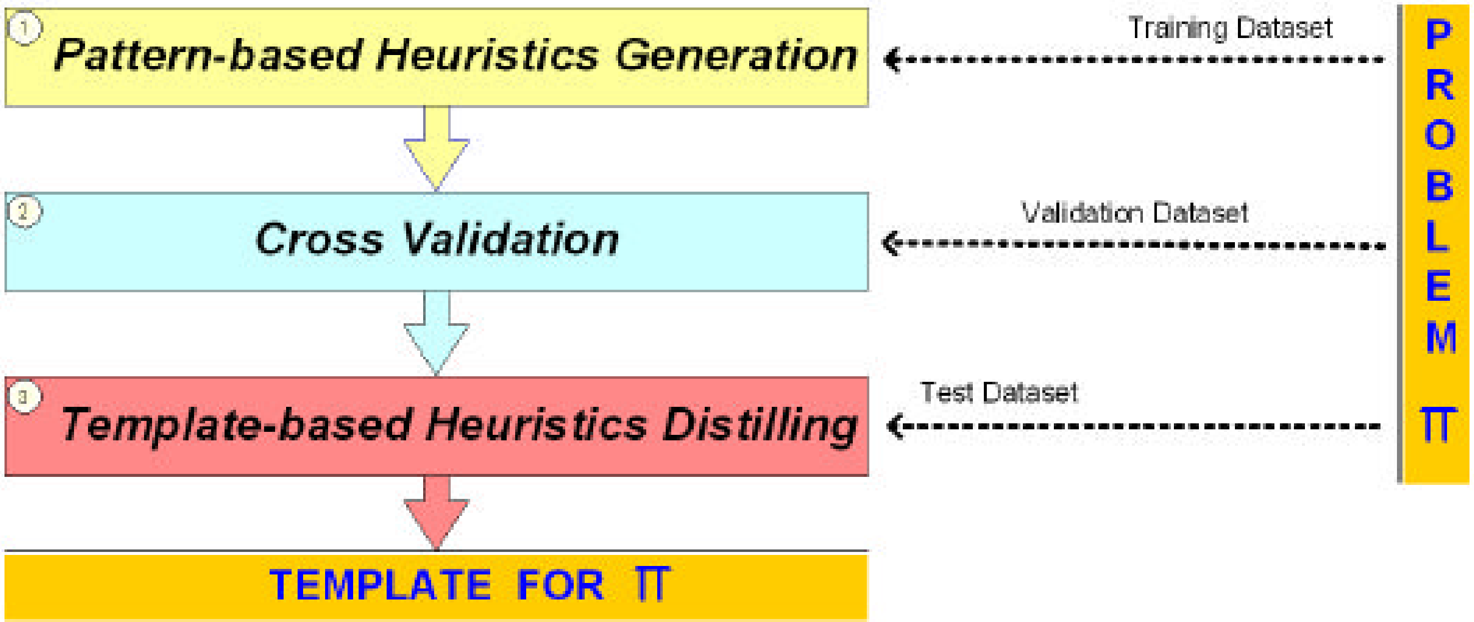
+

Evolutionary design

An execution thread represents a sequence of low-level heuristics that can be applied to a given tour in order to produce another tour.

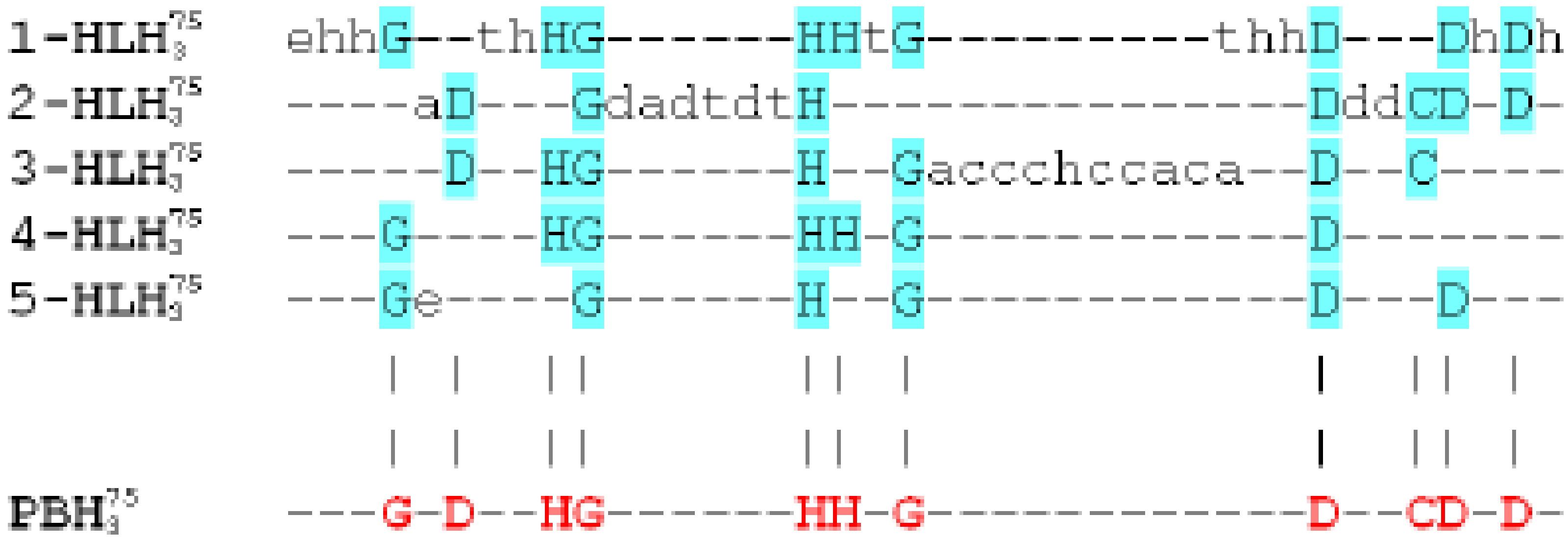


Methodology that generates a [low-level heuristics template]() for then manufacturing good performing heuristics.

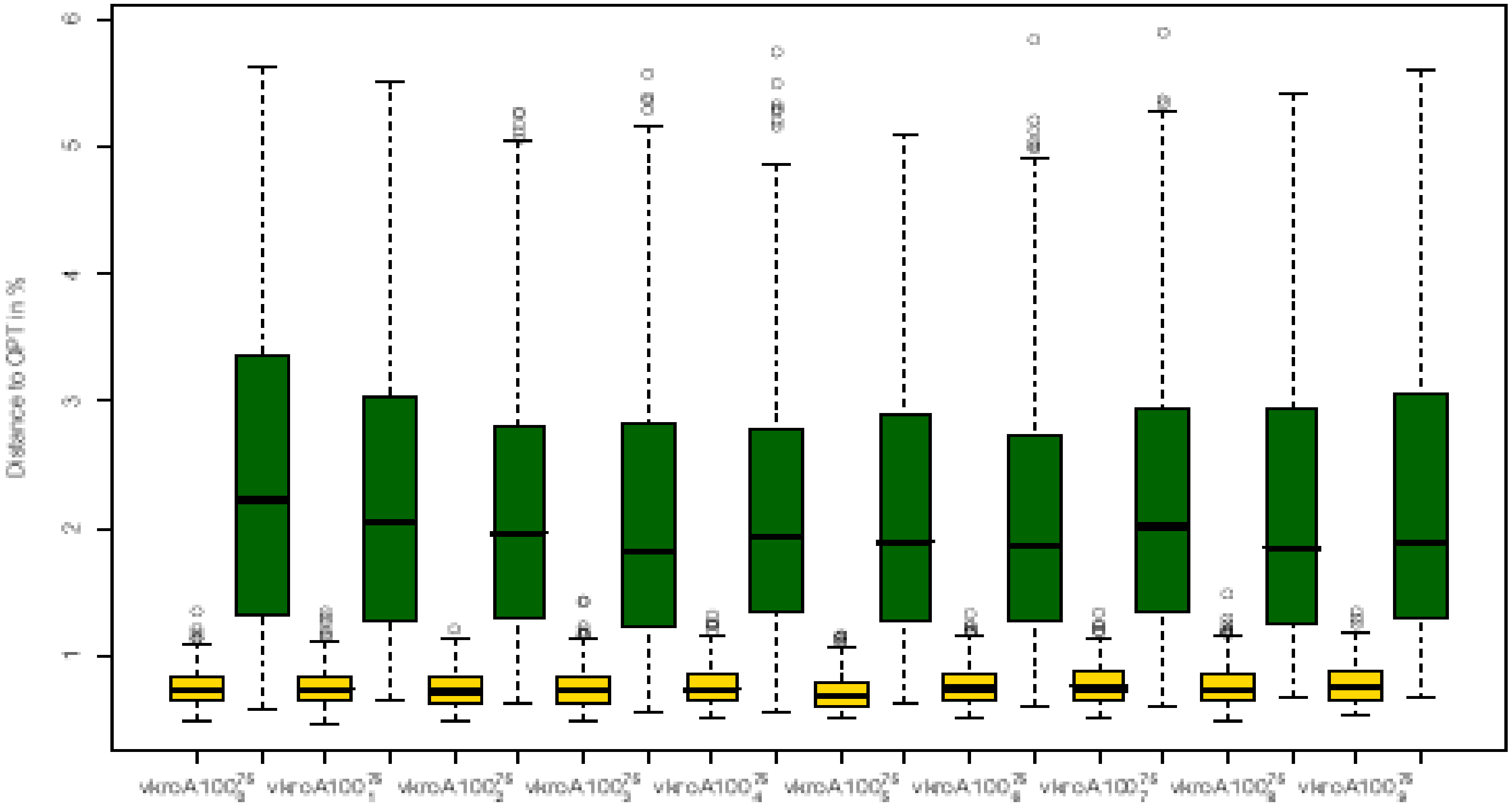


Other researchers are exploring GP and CBR for this purpose.

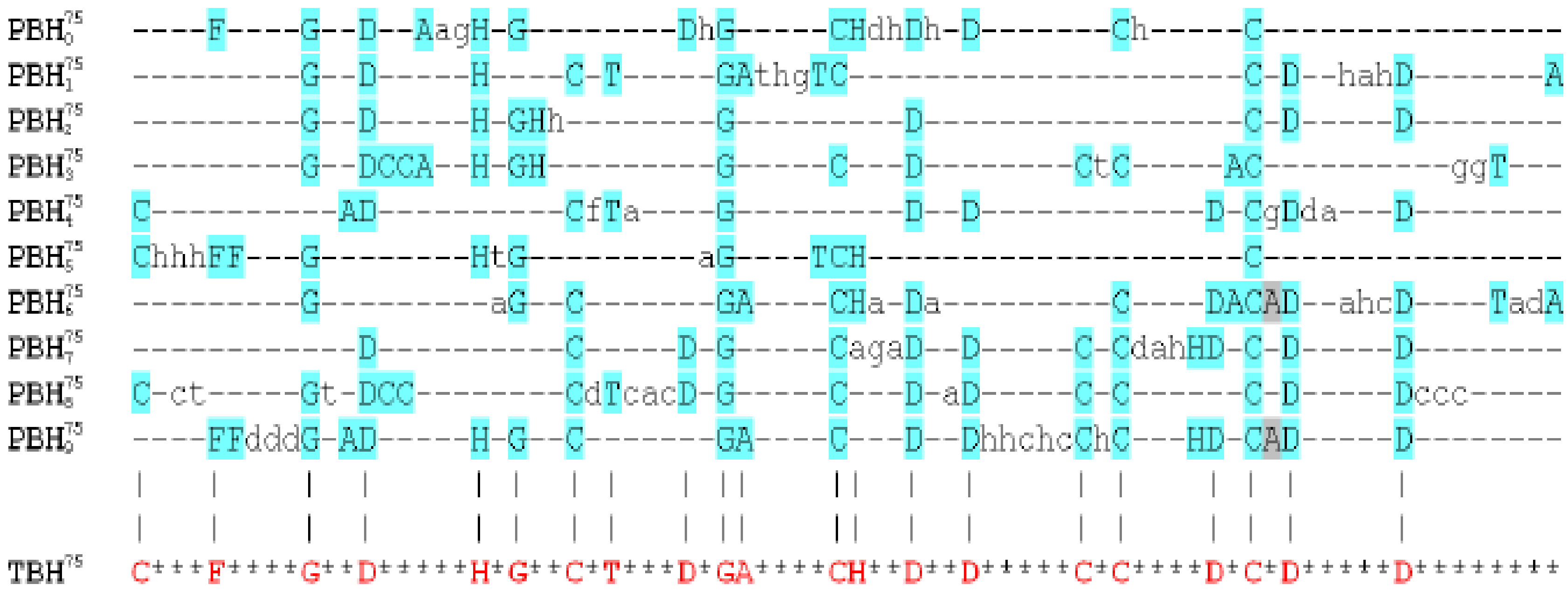
From top performing heuristic sequences, common combinations (patterns) of low-level heuristics are identified, e.g. GDHGHHGDCDD.



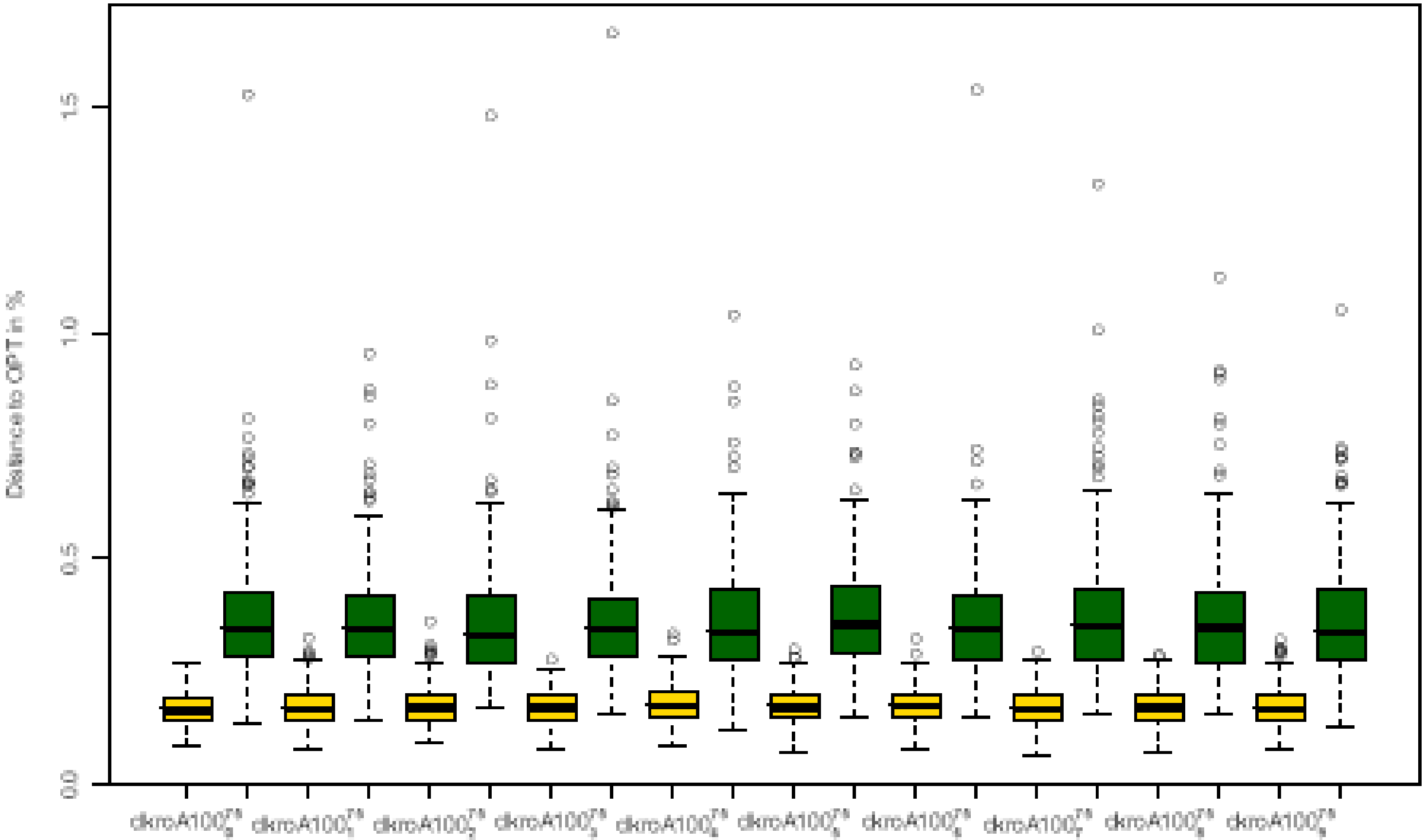
[Performance of pattern-based heuristics](#) is compared against other non-pattern-based heuristic sequences.



Common structures among the pattern-based heuristics are identified. Then, a heuristic template (e.g. TBH^{75}) is constructed in terms of building blocks.



The sequences of heuristics are generated based on the heuristic template have a better and more robust performance when solving other ‘unseen’ instances of the problem.



FINAL REMARKS

Key issues in the design of heuristic search methods for combinatorial optimisation problems include:

- define objective function
- initialise solutions
- search neighbourhood
- define basic strategies to escape local optima
- deal with infeasibility

Known strategies become self-adaptive by following simple principles:

- acceptance of non-improving solutions according to search progress
- use of memory for tracking visited solutions and past performance of neighbourhood search
- restricted mating adapting to diversity in decision space
- weight vectors that adapt for competition and diversification/intensification
- extend single-point local search to population-based local search

Self-assembly and self-generation of heuristics is under investigation.

SOME REFERENCES

Dario Landa-Silva, Edmund K. Burke. **Asynchronous Cooperative Local Search for the Office Space Allocation Problem**. *INFORMS Journal on Computing*, 19(4), pp. 575-587, **2007**.

Khoi Nguyen Le, Dario Landa-Silva. **Adaptive and Assortative Mating Scheme for Evolutionary Multi-objective Algorithms**. Proceedings of the *2007 Evolution Artificielle Conference (EA 2007)*, Tours France, LNCS, Vol. 4926, Springer, pp. 172-183, **2008**.

Joe Henry Obit, Dario Landa-Silva. **Computational Study of Non-Linear Great Deluge for University Course Timetabling**. To Appear in: V. Sgurev, M. Hadjiski (eds). *Intelligent Systems - From Theory to Practice*. Post Conference IEEE-IS 2008 Volume, Studies in Computational Intelligence, Springer-Verlag, **2009**.

Hui Li, Dario Landa-Silva. **An Adaptive Evolutionary Multi-objective Approach Based on Simulated Annealing**. To appear in the MIT Evolutionary Computation Journal. **2010**.

Joe Henry Obit, Dario Landa-Silva, Marc Sevaux, Djamila Ouelhadj. **Non-Linear Great Deluge with Reinforcement Learning for University Course Timetabling**. Under review. **October 2009**.

German Terrazas, Dario Landa-Silva, Natalio Krasnogor. **Discovering Beneficial Cooperative Structures for the Automatic Construction of Heuristics**. To Appear in NICSO 2010. **May 2010.**