Machine Learning meets Selection Hyper-heuristics

Prof Ender Özcan
School of Computer Science

12/06/2024

Outline

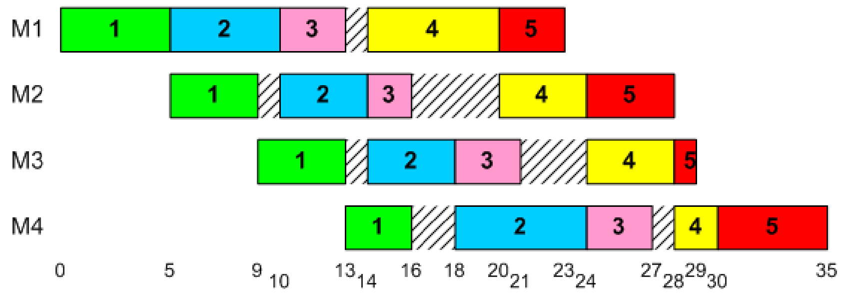
- Hyper-heuristics – Definition, Origins, Motivation, Classification
- Selection Hyper-heuristics Controlling Perturbative Heuristics
  - HyFlex,Cross-Domain Heuristic Search Competition (CHeSC 2011)
  - AdapHH, MSHH
- Automated Design/Generation of Selection Hyper-heuristics
- An Apprenticeship Learning Hyper-heuristic for OVRP
  - Experts: AdapHH, MSHH, Apprentice: TDNN
- Concluding Remarks

State-of-the-art in Meta/heuristic Optimisation

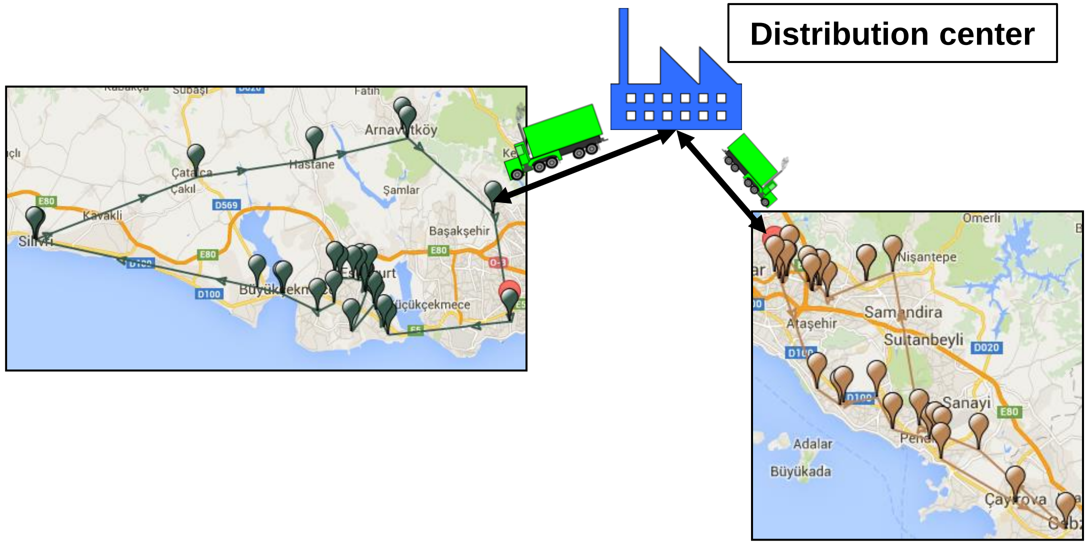
Trial and Error:

- Design and implement algorithmic components
- Configure the algorithm and tune the parameters: Test on selected instances (revisit the design options)
- Performance analyses on unseen instances (revisit the design options)

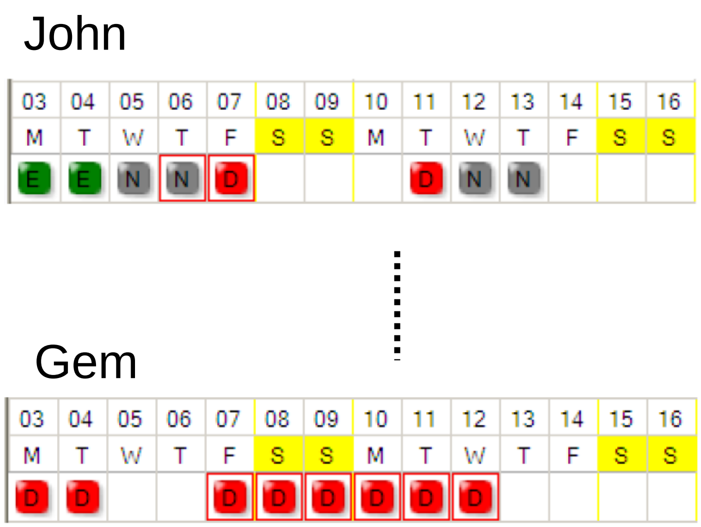
Flowshop Scheduling



Vehicle Routing



Nurse Rostering



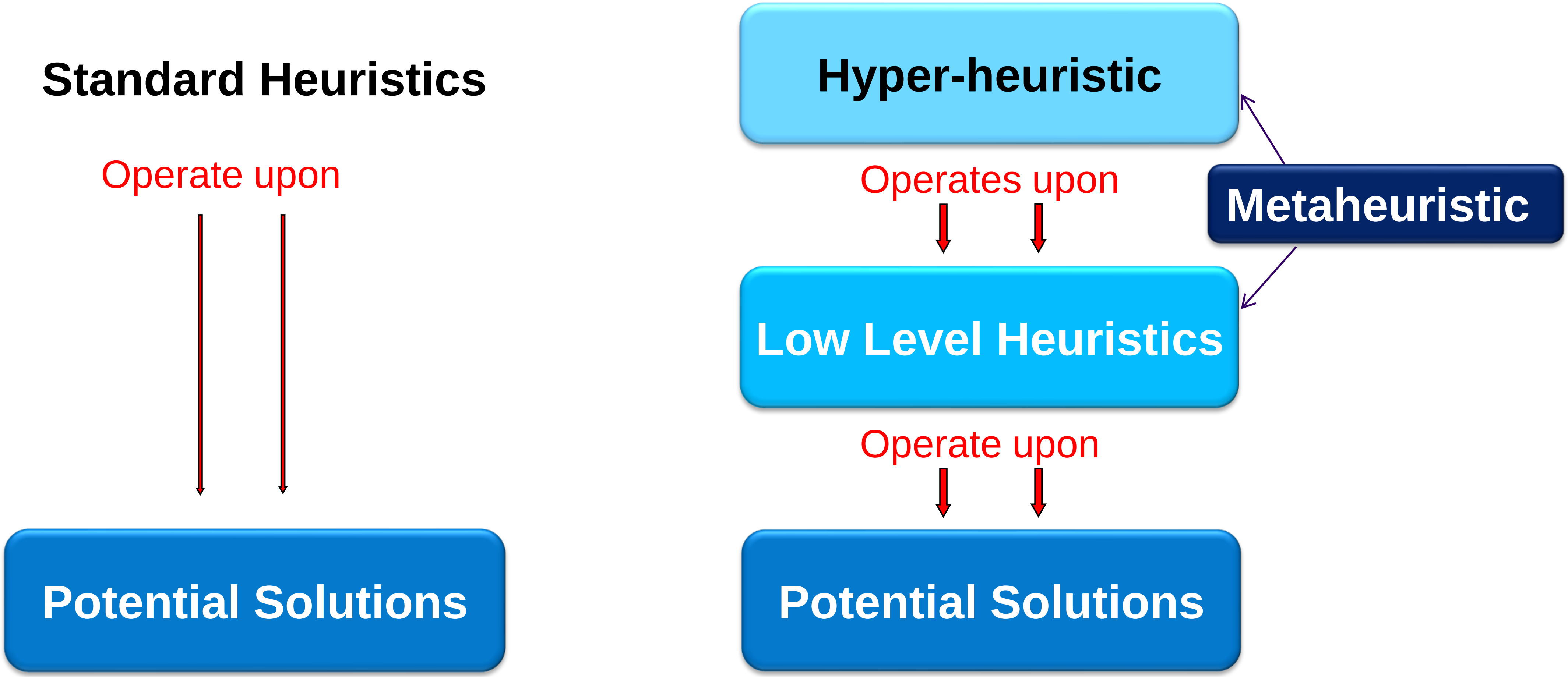
Hyper-heuristics

A hyper-heuristic is a search method or learning mechanism for selecting or generating heuristics to solve computationally difficult problems

- A class of methodologies for cross-domain search
  - search methods with reusable components used for solving characteristically different multiple problems preferably with the least or even no “human” intervention (e.g.,for tuning, applying the approach to a new instance,etc.)

E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, R. Qu, Hyper-heuristics: A Survey of the State of the Art, Journal of the Operational Research Society, 64 (12) , pp. 1695-1724, 2013. [PDF]

Different Search Spaces



Motivation

- Hyper-heuristic research is motivated by raising the level of generality. What are the limits?
- **Grand Challenge**

More General → The General Solver

A B C

Problem Specific Solvers

Significant scope for future research

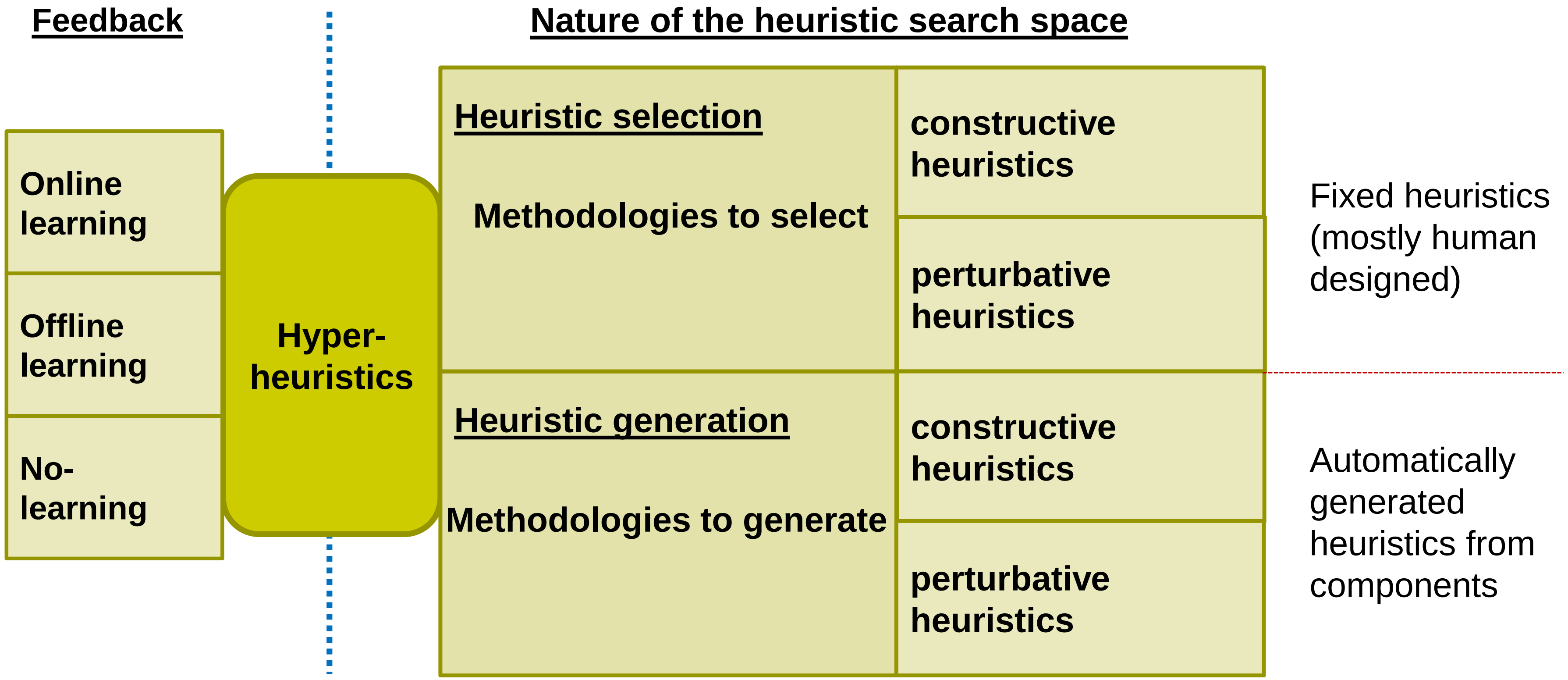
Doesn't exist….

Related Areas

- Adaptive operator selection (Fialho et al., 2008, and 2014)
- Algorithm configuration (López-Ibáñez et al., 2014, and 2016)
- Algorithm selection/portfolios (Kotthoff, 2014)
- Co-evolution/multimeme memetic algorithms/Memetic computing (Ong et al., 2006, Feri et al., 2012)
- Hybrid metaheuristics (Raidl, 2015)
- Meta-learning (Pappa et al., 2014, Blum et al., 2011)
- Parameter control (e.g., in EAs) (Eiben et al., 2007)
- Reactive search (Battiti & Brunato, 2017)
- Variable Neighbourhood Search (Hansen et al., 2010)
- …

A Classification of Hyper-heuristics

E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Özcan and J. Woodward, A Classification of Hyper-heuristic Approaches: Revisited, In Gendreau, M, and Potvin, JY. (eds.), Handbook of Metaheuristics, International Series in Operations Research & Management Science, vol. 272, pp. 453-477. Springer Cham, 2019. [PDF]



J. Branke, S. Nguyen, C. W. Pickardt, and M. Zhang, Automated design of production scheduling heuristics: A review. IEEE Trans. Evol. Comput., vol. 20, no. 1, pp. 110–124, Feb. 2016. [PDF]

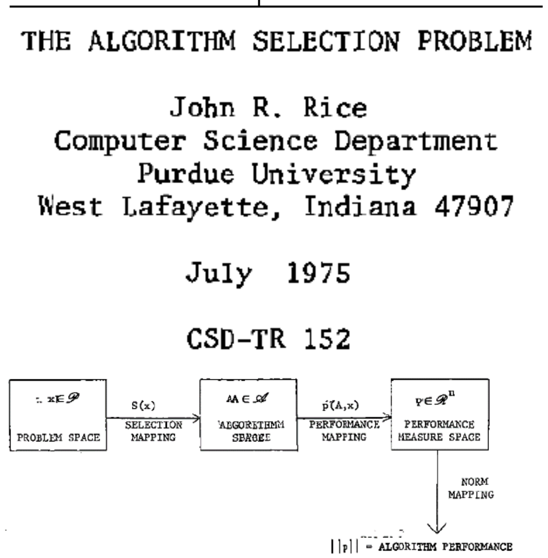
Hyper-heuristics: Origins

1961-63 | 1975 | 1990-95 | 1997 | 2001

Cowling P.I., Kendall G. and Soubeiga E., 2001. A Hyperheuristic Approach to Scheduling a Sales Summit, selected papers from PATAT 2000, Springer, LNCS 2079, 176-190.

Fisher H. and Thompson G.L., 1963. Probabilistic Learning Combinations of Local Job-shop Scheduling Rules. Ch 15,:225-251, Prentice Hall, New Jersey.

Crowston W.B., Glover F., Thompson G.L. and Trawick J.D. Probabilistic and Parameter Learning Combinations of Local Job Shop Scheduling Rules. ONR Research Memorandum, GSIA,CMU, Pittsburgh, (117), 1963


THE ALGORITHM SELECTION PROBLEM

John R. Rice
Computer Science Department
Purdue University
West Lafayette, Indiana 47907

July 1975

CSD-TR 152

Storer R. H., Wu S. D. , Vaccari R., 1992. New Search Spaces for Sequencing Problems with Application to Job Shop Scheduling, INFORMS, 38(10), 1495-1509.

Fang H.-L., Ross P. and Corne D., 1994. A Promising Hybrid GA/Heuristic Approach for Open-Shop Scheduling Problems., in'ECAI' , 590-594.

Denzinger J., Fuchs M. and Fuchs M., 1997. High performance ATP systems by combining several AI methods. In Proc. of the 15th IJCAI, 102-107.

A Selection Hyper-heuristic Framework for Cross-domain Search

- No domain knowledge, other than that embedded in a range of simple knowledge-poor heuristics.
- Robust enough to effectively handle a wide range of problems and problem instances from a variety of domains.

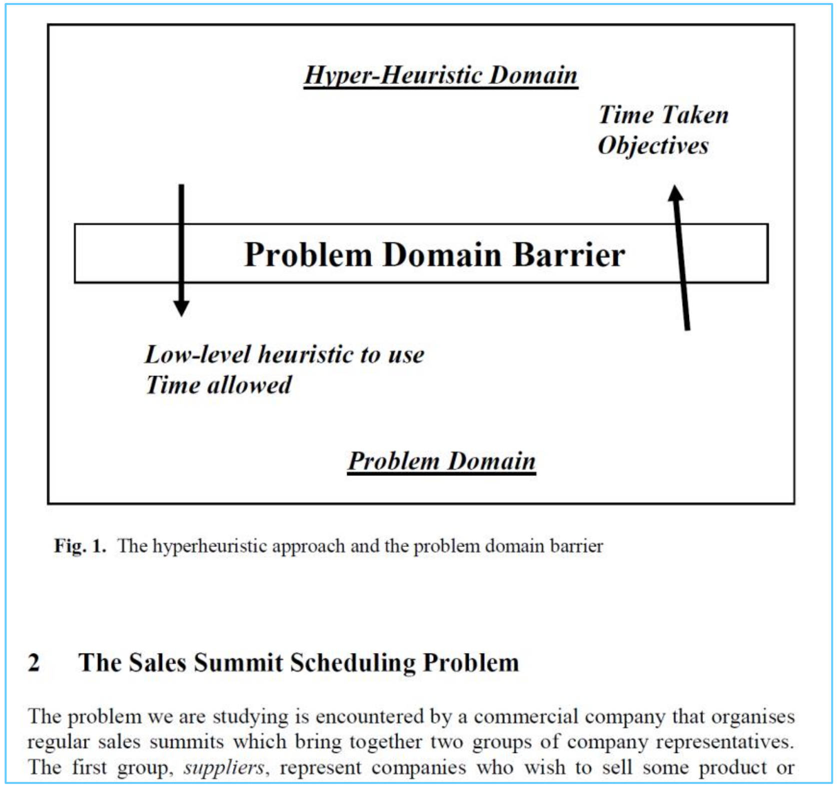


Fig. 1. The hyperheuristic approach and the problem domain barrier

2 The Sales Summit Scheduling Problem

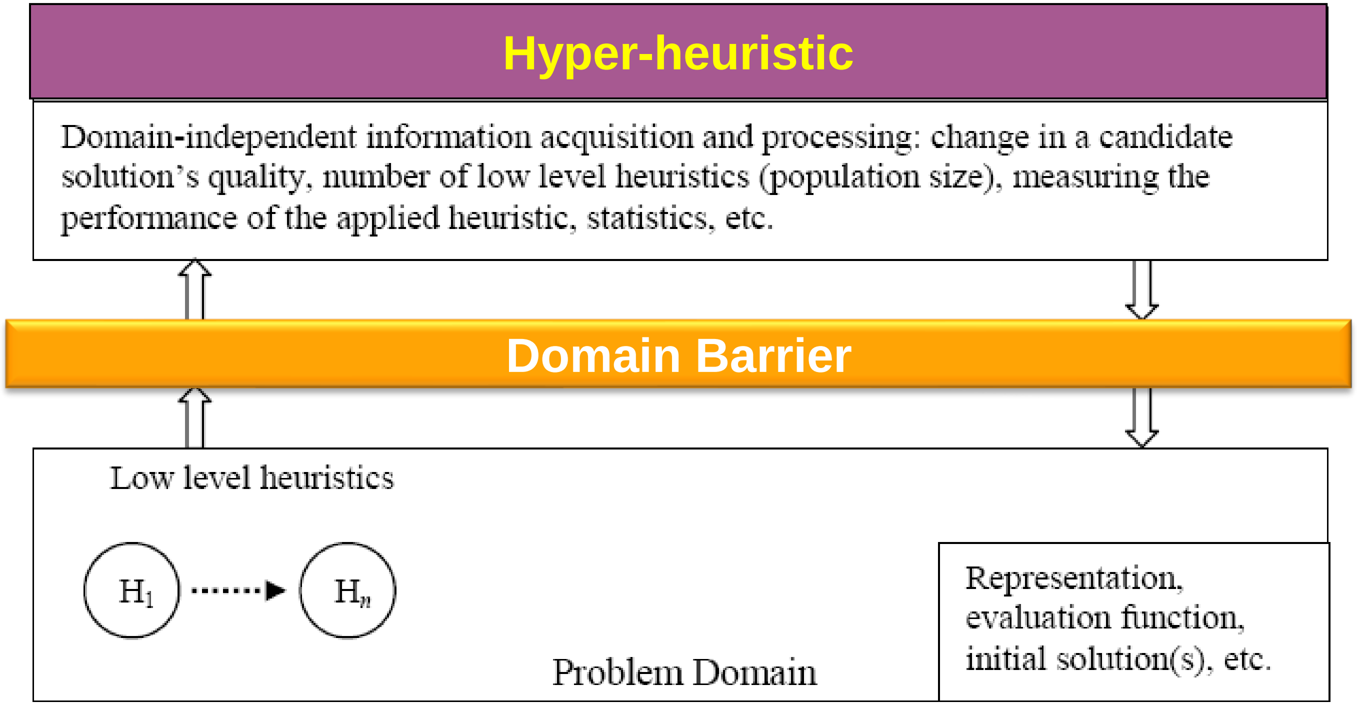
The problem we are studying is encountered by a commercial company that organises regular sales summits which bring together two groups of company representatives. The first group, *suppliers*, represent companies who wish to sell some product or

Cowling P.I., Kendall G. and Soubeiga E., 2001. A Hyperheuristic Approach to Scheduling a Sales Summit, selected papers from PATAT 2000, Springer, LNCS 2079, 176-190. [PDF]

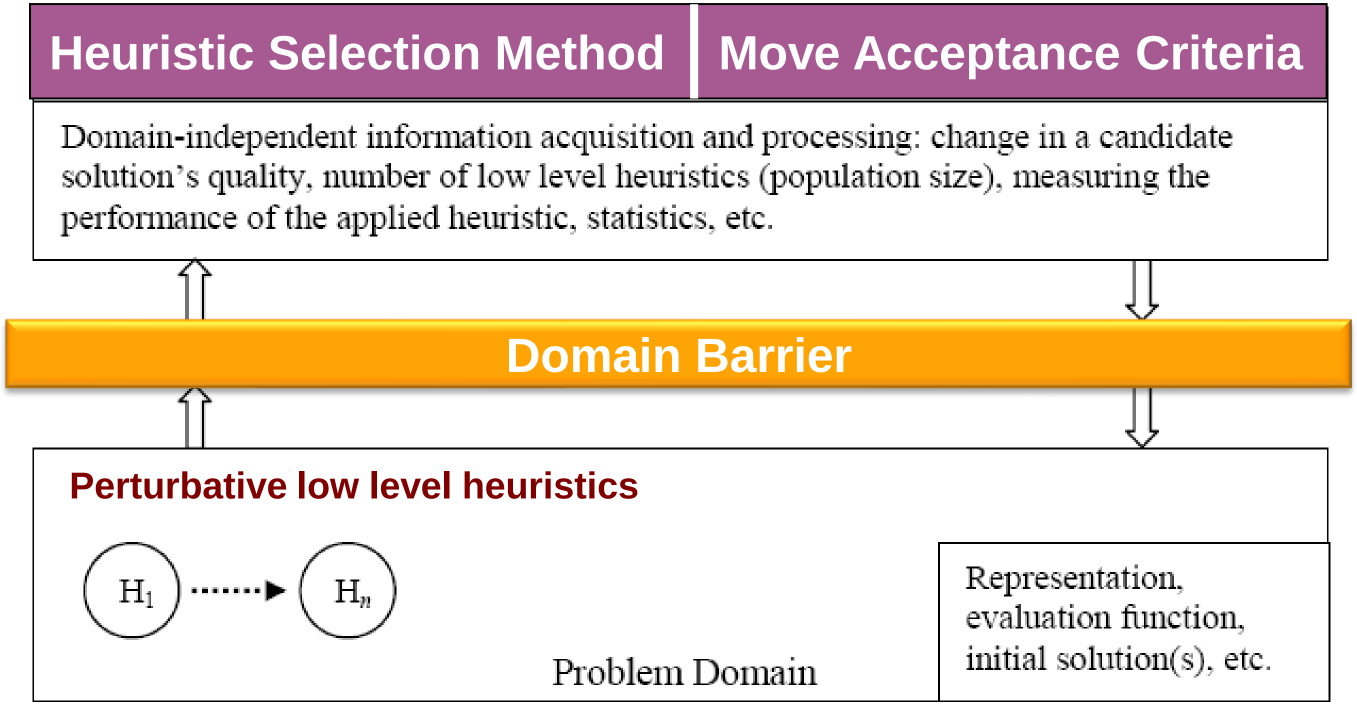
Recent Applications of Selection Hyper-heuristics

Application Domain	*References*
Design problems	Kheiri et al. (2015); Peraza-Vázquez et al. (2016); Allen et al. (2013)
Dynamic environments	Uludağ et al. (2012, 2013); Kiraz et al. (2013a,b); Topcuoglu et al. (2014); van der Stockt and Engelbrecht (2014); Baykasoğlu and Ozsoydan (2017)
Knapsack	Drake et al. (2016, 2014); Soria-Alcaraz et al. (2014a, 2017a); Lassouaoui and Boughaci (2014)
Maximum satisfiability	Jackson et al. (2014); Ferreira et al. (2015)
Puzzles and games	Wauters et al. (2012); Kheiri and Özcan (2014); Li and Kendall (2017)
Real-valued blackbox optimisation	Epitropakis et al. (2014); Grobler et al. (2013, 2014, 2015); Damaševičius and Woźniak (2017); Tinoco and Coello (2013)
Scheduling	Mısır et al. (2012a); Bilgin et al. (2012); Mısır et al. (2013a); Koulinas and Anagnostopoulos (2013); Rajni and Chana (2013); Tsai et al. (2014); Mısır and Lau (2014); Koulinas et al. (2014); Aron et al. (2015); Zheng et al. (2015); Mısır et al. (2015); Monemi et al. (2015); Asta et al. (2016a); Hassan and Pillay (2016); Chen et al. (2016a); Asta et al. (2016b); Wu et al. (2016); Lin et al. (2017); Pour et al. (2017); Chen et al. (2017)
Search based software engineering	Henard et al. (2014); Jia et al. (2015); Zamli et al. (2016, 2017)
Shelf allocation	Bai et al. (2013); Zhao et al. (2016)
Telecommunication	Yang et al. (2014); Hassan and Pillay (2016); Tsai et al. (2017)
Timetabling	Kalender et al. (2012, 2013); Kheiri et al. (2016b); Burke et al. (2014); Soria-Alcaraz et al. (2014b); Ahmed et al. (2015); Kheiri and Keedwell (2017); da Fonseca et al. (2016); Soria-Alcaraz et al. (2016, 2017c,b)
Traveling salesman	Swiercz et al. (2014); Qu et al. (2015); Smith and Imeson (2017); Choong et al. (2017); Martins et al. (2017); El Yafrani et al. (2018)
Vehicle routing	Akar et al. (2014); Marshall et al. (2015); Urra et al. (2015); Sabar et al. (2015c); Yin et al. (2016); Sim and Hart (2016); Chen et al. (2016b); Mourdjis et al. (2016); Tyasnurita et al. (2017); Soria-Alcaraz et al. (2017b)

A Hyper-heuristic Framework



A Selection Hyper-heuristic Framework – Single Point Search



A Selection Hyper-heuristic Framework – Single Point Search

1. generate initial candidate solution *p*
2. while (termination criteria not satisfied) {
3. select a heuristic (or subset of heuristics) *h* from $\{H_1, ..., H_n\}$
4. generate a new solution (or solutions) *s* by applying *h* to p
5. decide whether to accept *s* or not
6. if (*s* is accepted) then
7. *p*=*s* }
8. return *p*;

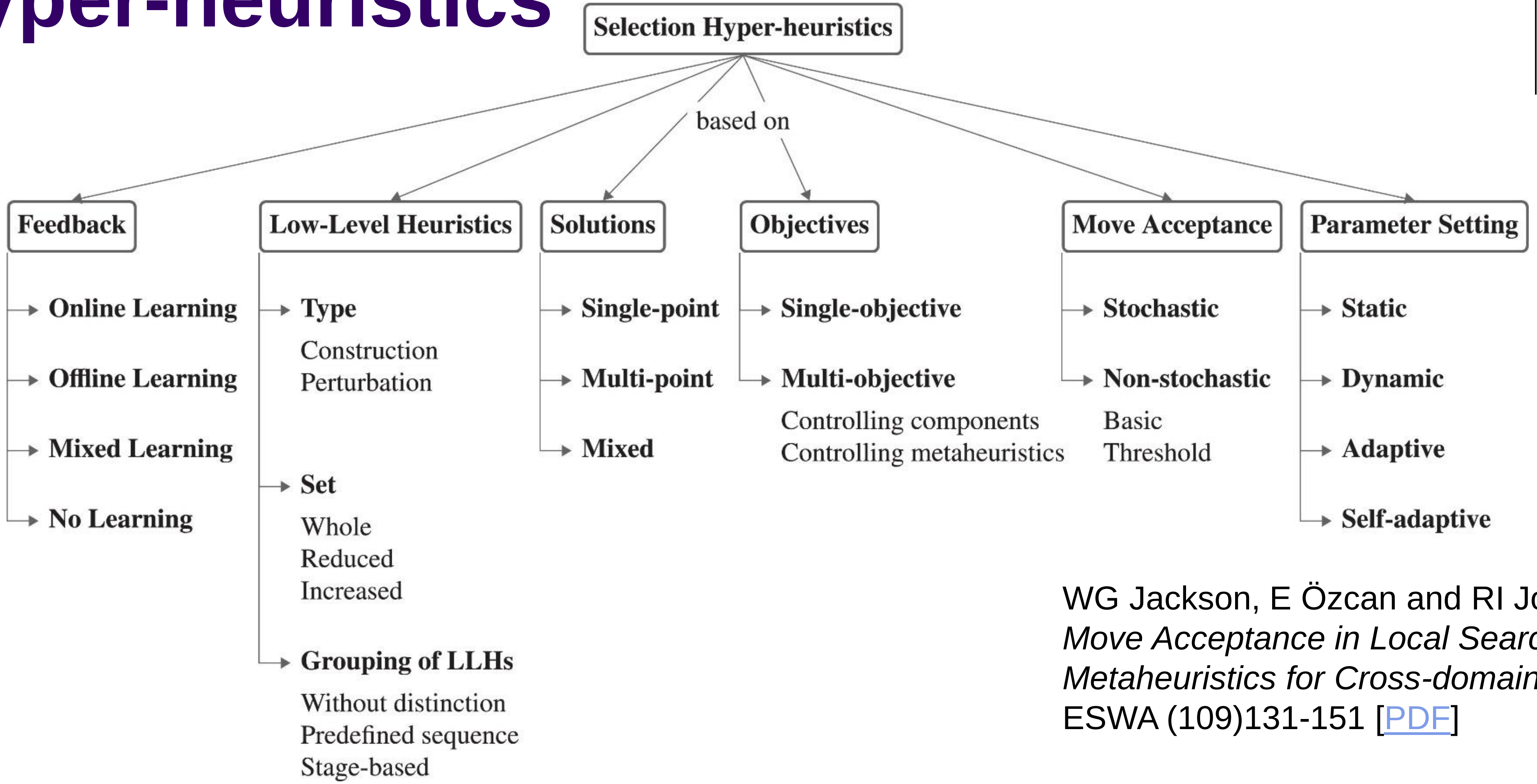
Heuristic Selection

with no learning

with learning

Component name	Reference(s)
Heuristic selection **with no learning**	
Simple Random	Cowling et al (2000, 2002b)
Random Permutation	Cowling et al (2000, 2002b)
Heuristic selection **with learning**	
Peckish	Cowling and Chakhlevitch (2003)
Greedy	Cowling et al (2000, 2002b); Cowling and Chakhlevitch (2003)
Random Gradient	Cowling et al (2000, 2002b)
Random Permutation Gradient	Cowling et al (2000, 2002b)
Choice Function	Cowling et al (2000, 2002b); Maashi et al (2015); Drake et al (2015)
Reinforcement Learning	Nareyek (2003); Pisinger and Ropke (2007)
Reinforcement Learning with Tabu Search	Burke et al (2003); Dowsland et al (2007)
Quality Index and Tabu based Learning Heuristic Selection	Mısır et al (2009, 2012)
Dominance-based Selection	Kheiri and Özcan (2011; 2015)
Probability-based Selection	Lehrbaum and Musliu (2012)
Adaptive pursuit	Walker et al (2012)

Extended Classification of Selection Hyper-heuristics



WG Jackson, E Özcan and RI John, *Move Acceptance in Local Search Metaheuristics for Cross-domain Search*, ESWA (109)131-151 [PDF]

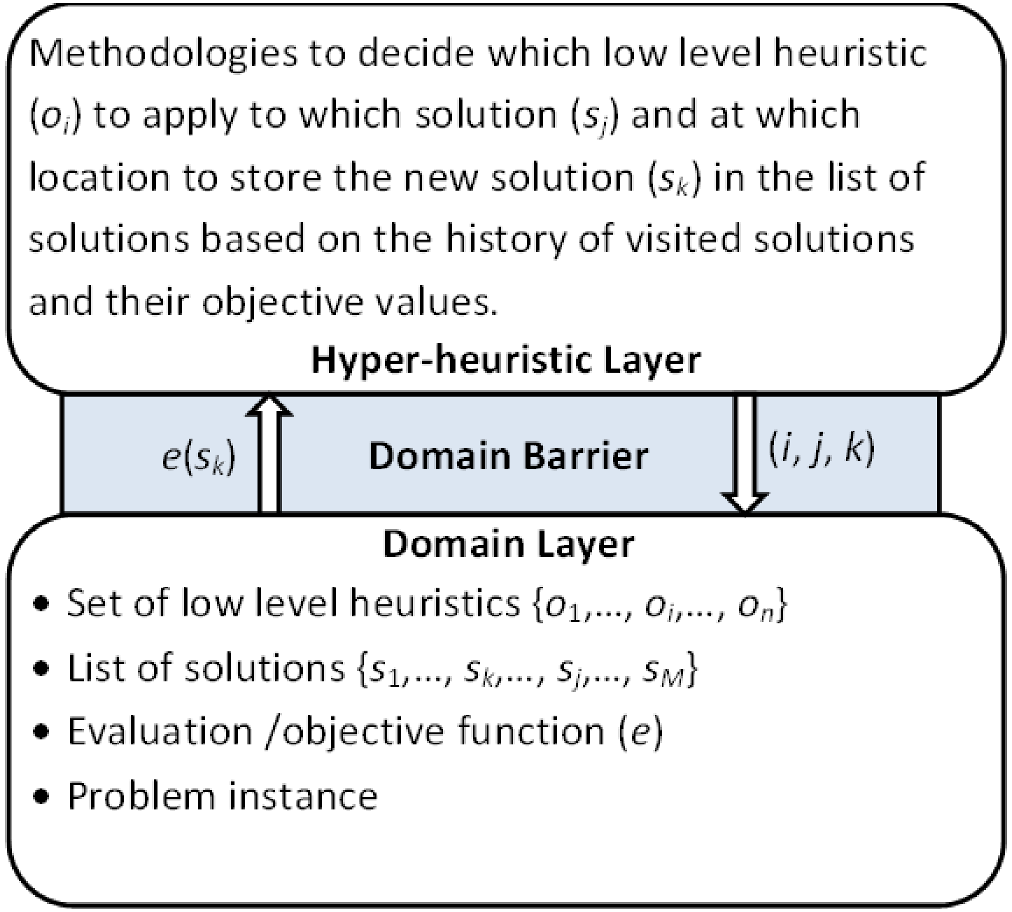
J. H. Drake, A. Kheiri, E. Özcan, and E. K. Burke, *Recent Advances in Selection Hyper-heuristics*, EJOR, 285(2): 405-428, 2020 [PDF].

A Sample of Selection Hyper-heuristics

Source	Search points	Feedback	LLH set	Grouping of LLHs	Accept/reject	Parameter setting in move acceptance
(Chan et al., 2012)	Single	Mixed	Whole	Predefined	Basic, threshold	Static, adaptive
(Di Gaspero & Urli, 2012)	Single	Online	Whole	Predefined	Basic	None
(Drake et al., 2012)	Single	Online	Reduced	Without distinction	Basic	None
(Hsiao et al., 2012)	Mixed	Online	Reduced	Predefined	–	–
(Kubalík, 2012)	Population	Mixed	Reduced	Predefined	–	–
(Lehrbaum & Musliu, 2012)	Mixed	Online	Reduced	Predefined	–	–
(Mascia & Stützle, 2012)	Single	Offline	Reduced	Predefined	Stochastic	Static
(Mısır et al., 2012b)	Single	Online	Whole	Without distinction	Threshold	Adaptive
(Jackson et al., 2013)	Single	Online	Whole	Without distinction	Threshold	Static
(Adriaensen et al., 2014b)	Single	Online	Whole	Predefined	Stochastic	Adaptive
(Kheiri et al., 2016)	Single	Online	Reduced	Predefined	Threshold	Adaptive
(Asta & Özcan, 2015)	Single	Offline	Reduced	Stage-based	Basic	Static
(Drake, 2014)	Single	Online	Whole	Without distinction	Basic	None
(Kheiri & Keedwell, 2015)	Single	Online	Reduced	Without distinction	Threshold	Adaptive
(Asta et al., 2016a)	Single	Online	Reduced	Stage-based	Threshold	Adaptive
(Kheiri & Özcan, 2016)	Single	Online	Increased	Stage-based	Threshold	Adaptive
(Meignan et al., 2016)	Single	Online	Reduced	Predefined	Basic	None
(Chuang & Smith, 2017)	Single	No learning	Reduced	Predefined	Basic	None
(Ferreira et al., 2017)	Single	Online	Whole	Without distinction	Threshold	Dynamic
(Yates and Keedwell 2017)	Single	Offline	Whole	Without distinction	Stochastic	Static

Hyper-heuristics Flexible Interface (HyFlex)

https://www.cs.nott.ac.uk/~pszwj1/chesc2011/ (web archive)



Ochoa G, Hyde M, Curtois T, Vazquez-Rodriguez JA, Walker J, Gendreau M, Kendall G, McCollum B, Parkes AJ, Petrovic S, Burke EK (2012) HyFlex: a benchmark framework for cross-domain heuristic search. In: Evolutionary Computation in Combinatorial Optimization, LNCS 7245, pp 136–147 [PDF]

HyFlex v1.0 Java Implementation

- Heuristic types:
  mutational (MU), ruin-recreate (RC), local search (HC), crossover (XO)
- Parameters: intensity of mutation (MU+RC), depth of search (HC)

Problem Domains

- MAX-SAT
- Bin Packing
- Flow Shop
- Personnel Scheduling
- TSP
- VRP

Heuristic IDs	LLH0	LLH1	LLH2	LLH3	LLH4	LLH5	LLH6	LLH7
MAX-SAT	MU_0	MU_1	MU_2	MU_3	MU_4	$\mathbf{MU}_5$	$\mathbf{RC}_0$	HC_0
Bin Packing	MU_0	RC_0	$\mathbf{RC}_1$	MU_1	HC_0	$\mathbf{MU}_2$	$\mathbf{HC}_1$	$\mathbf{XO}_0$
PS	HC_0	HC_1	HC_2	HC_3	$\mathbf{HC}_4$	RC_0	RC_1	$\mathbf{RC}_2$
PFS	MU_0	MU_1	MU_2	MU_3	$\mathbf{MU}_4$	RC_0	$\mathbf{RC}_1$	HC_0
TSP	MU_0	MU_1	MU_2	MU_3	$\mathbf{MU}_4$	$\mathbf{RC}_0$	HC_0	HC_1
VRP	MU_0	MU_1	RC_0	$\mathbf{RC}_1$	HC_0	XO_0	$\mathbf{XO}_1$	$\mathbf{MU}_2$

Heuristic IDs	LLH8	LLH9	LLH10	LLH11	LLH12	LLH13	LLH14
MAX-SAT	$\mathbf{HC}_1$	XO_0	$\mathbf{XO}_1$				
PS	XO_0	XO_1	$\mathbf{XO}_2$	$\mathbf{MU}_0$			
PFS	HC_1	HC_2	$\mathbf{HC}_3$	XO_0	XO_1	XO_2	$\mathbf{XO}_3$
TSP	$\mathbf{HC}_2$	XO_0	XO_1	XO_2	$\mathbf{XO}_3$		
VRP	HC_1	$\mathbf{HC}_2$					

asap research
automated scheduling optimisation & planning

The University of Nottingham

CHeSC 2011 benchmark based on HyFlex v1.0

Problem Domains

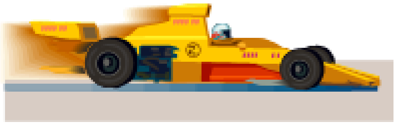
- MAX-SAT
- Bin Packing
- Flow Shop
- Personnel Scheduling
- TSP
- VRP

Hidden: TSP, VRP

- 10 public training instances
- **5 test instances**
  (3 training + 2 hidden/all hidden)

- Set problem instance
- Set time limit (10 min.)
- Perform 31 runs
- Report median

Ranking: Formula 1 scoring system



Rank	Score
1	10
2	8
3	6
4	5
5	4
6	3
7	2
8	1
rest	0

Organising Partners:

Queen's University Belfast

EPSRC Engineering and Physical Sciences Research Council

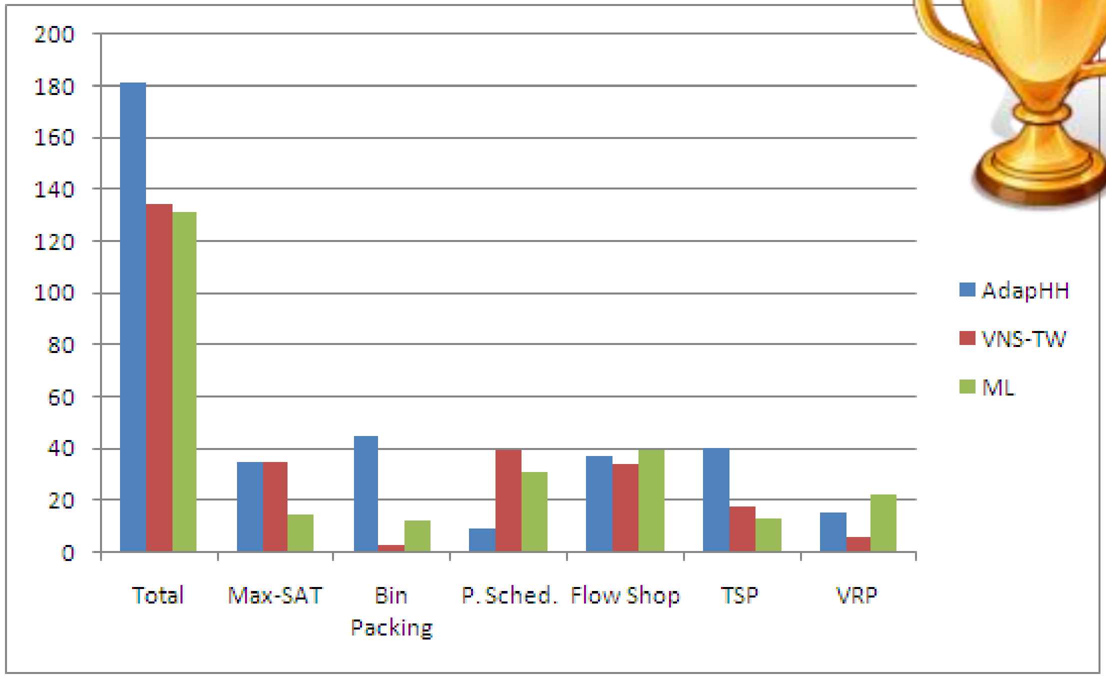
École Polytechnique Montréal

Cardiff University Prifysgol Caerdydd

Sponsor:

EVENTMAP

And the winner is...



AdapHH – M. Mısır
K. Verbeeck
P. De Causmaecker
G. Vanden Berghe

Rank	Hyper-heuristic	Score	Rank	Hyper-heuristic	Score
1	AdapHH	181.00	11	ACO-HH	39.00
2	VNS-TW	134.00	12	GenHive	36.50
3	ML	131.50	13	DynILS	27.00
4	PHUNTER	93.25	14	SA-ILS	24.25
5	EPH	89.75	15	XCJ	22.50
6	HAHA	75.75	16	AVEG-Nep	21.00
7	NAHH	75.00	17	GISS	16.75
8	ISEA	71.00	18	SelfSearch	7.00
9	KSATS-HH	66.50	19	MCHH-S	4.75
10	HAEA	53.50	20	Ant-Q	0.00

AdapHH – Overview

$$
\begin{aligned}
p_i \;=\; & w_1\Big[\big(C_{p,best}(i)+1\big)^2\big(t_{remain}/t_{p,spent}(i)\big)\Big]\times b\,+\\
& w_2\big(f_{p,imp}(i)/t_{p,spent}(i)\big) - w_3\big(f_{p,wrs}(i)/t_{p,spent}(i)\big)+\\
& w_4\big(f_{imp}(i)/t_{spent}(i)\big) - w_5\big(f_{wrs}(i)/t_{spent}(i)\big)
\end{aligned}
$$

$$
b \;=\; \begin{cases} 1, & \sum_{i=0}^{n} C_{p,best}(i) > 0 \\ 0, & otw. \end{cases}
\qquad\qquad avg = \Big\lfloor \Big(\sum_{i}^{n} QI_i\Big)/n \Big\rfloor
$$

$$pl \;=\; ph_{duration}/t_{subset}$$

$$ph_{duration} \;=\; t_{total}/ph_{requested}$$

$$exc(i) = t_{perMove}(i)/t_{perMove}(fastest)$$

$$\sigma > 2.0 \;;\; exc(i) > 2\varpi \;;\; nb > 1$$

$$pr_i = \big((C_{best}(i)+1)/t_{spent}\big)^{(1+3tf^3)}$$

$$
k \;=\; \begin{cases} ((l-1).k + iter_{elapsed})/l, & \text{if } cw = 0 \\ ((l-1).k + \sum_{i=0}^{cw} k.0.5^i.tf)/l, & \text{otherwise} \end{cases}
$$

$$tf = (t_{exec} - t_{elapsed})/t_{exec}$$

$$cw = iter_{elapsed}/k$$

Algorithm AILLA move acceptance

```
Input: i = 1, K ≥ k ≥ 0, l > 0
for i=0 to l-1 do best_list(i) = f(S_initial)
if adapt_iterations ≥ K then
    if i < l − 1 then
        i++
    end
end
if f(S') < f(S) then
    S ← S'
    w_iterations = 0
    if f(S') < f(S_b) then
        i = 1
        S_b ← S'
        w_iterations = adapt_iterations = 0
        best_list.remove(last)
        best_list.add(0, f(S_b))
    end
else if f(S') = f(S) then
    S ← S'
else
    w_iterations++
    adapt_iterations++
    if w_iterations ≥ k and f(S') ≤ best_list(i) then
        S ← S' and w_iterations = 0
    end
end
```

Algorithm Relay hybridisation

```
Input: list_size = 10; γ ∈ (0.02, 50); p, p' ∈ [0 : 1]
γ = (C_best,s + 1)/(C_best,r + 1)
if p ≤ (C_phase/pl)^γ then
    select LLH using a LA and apply to S → S'
    if size(list_i) > 0 and p' <= 0.25 then
        select a LLH from list_i and apply to S' → S''
    else
        select a LLH and apply to S' → S''
    end
end
```

AdapHH (GIHH)

Mustafa Misir, Katja Verbeeck, Patrick De Causmaecker, Greet Vanden Berghe. *A New Hyper-heuristic as a General Problem Solver: an Implementation in HyFlex*. *Journal of Scheduling*, 16(3), 2013 [PDF]

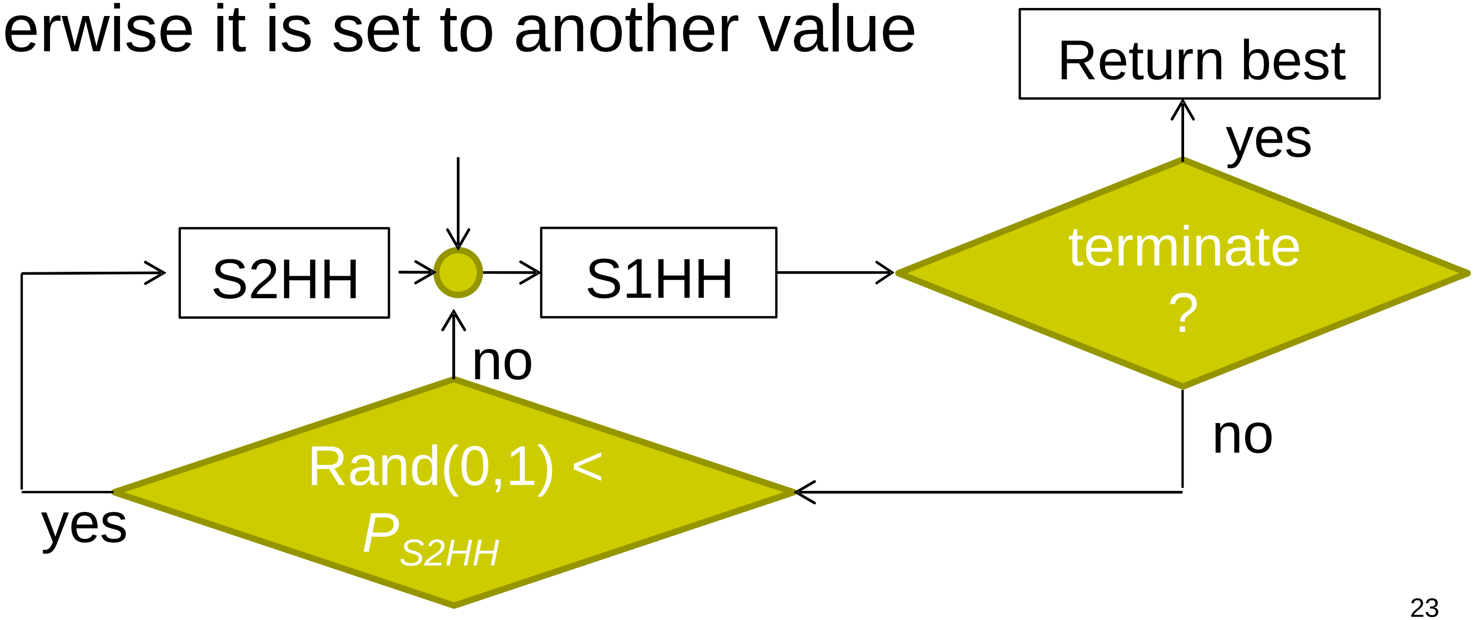
Steven Adriaensen, and Ann Nowé. *Case Study: An Analysis of Accidental Complexity in a State-of-the-art Hyper-heuristic for HyFlex*. 2016 IEEE CEC, 1485-1492 [PDF]

LeanGIHH

An Iterated Multi-stage Selection Hyper-heuristic (MSHH)

A. Kheiri and E. Özcan, *An Iterated Multi-stage Selection Hyper-heuristic*, European Journal of Operational Research (250)1:77–90, 2016 [PDF]

- Single point based search – Crossover operators are ignored
- Parameter setting: IoM and DoS are discretised {0.2, 0.4,…,1.0} and a random value is chosen. The same value is used as long as there is improvement, otherwise it is set to another value randomly.



MSHH – Stage 1 Hyper-heuristic (S1HH)

- A score is maintained for each low level heuristic ($score_i$)
- Select a low-level heuristic i with probability

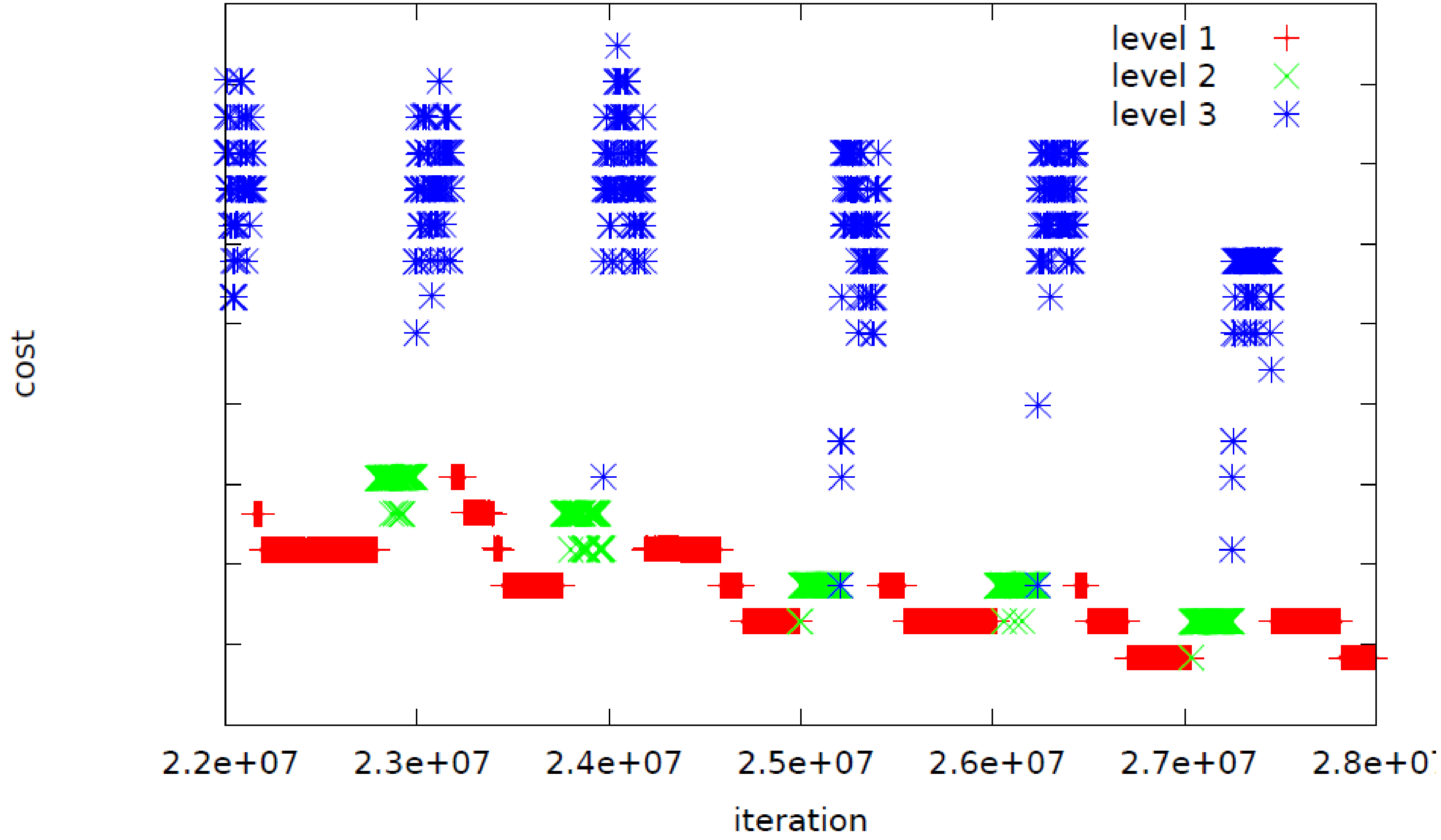
$$score_i / \sum_{\forall k} (score_k)$$

- Apply the chosen heuristic
- Accept/reject based on an adaptive threshold acceptance method
- Stage 1 terminates if a duration of s_1 is exceeded without any improvement

MSHH – An Adaptive Threshold Move Acceptance Method

$\in (c_i , f(S_{beststage}))$

c_i is an integer value in $C = \{c_0, \ldots, c_i, \ldots, c_{(l-1)}\}$ and C is a circular list



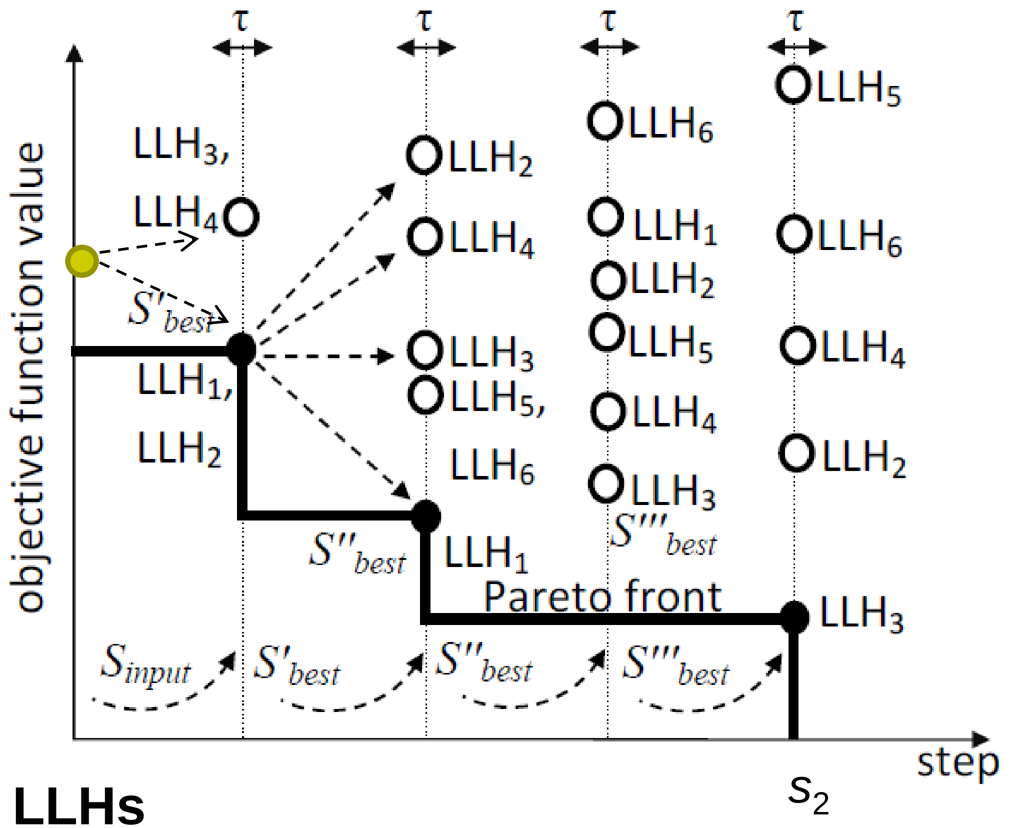
MSHH – Stage 2 Hyper-heuristic (S2HH)

Given *N* LLHs, e.g., LLH_1, LLH_2

Pair up all and increase the number of LLHs to $N+N^2$

$LLH_3 \leftarrow LLH_1 + LLH_1$
$LLH_4 \leftarrow LLH_2 + LLH_2$
$LLH_5 \leftarrow LLH_1 + LLH_2$
$LLH_6 \leftarrow LLH_2 + LLH_1$

Reduce the Number of LLHs ($N \rightarrow n$) + Assign Probabilities



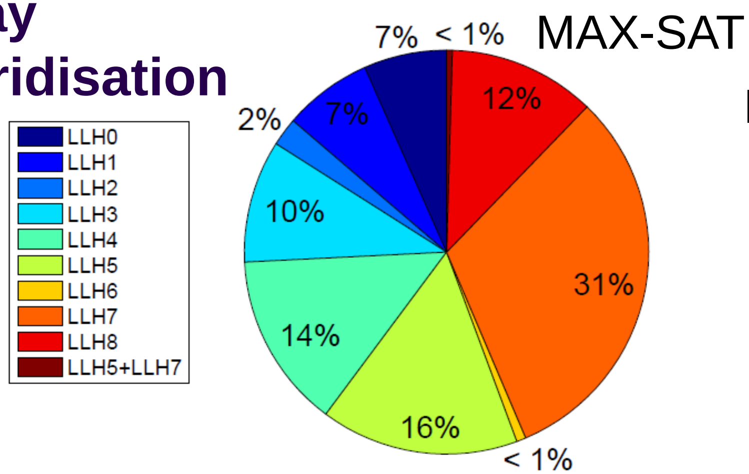
$LLH_1=2$, $LLH_2=1$, $LLH_3=1$
50% 25% 25%

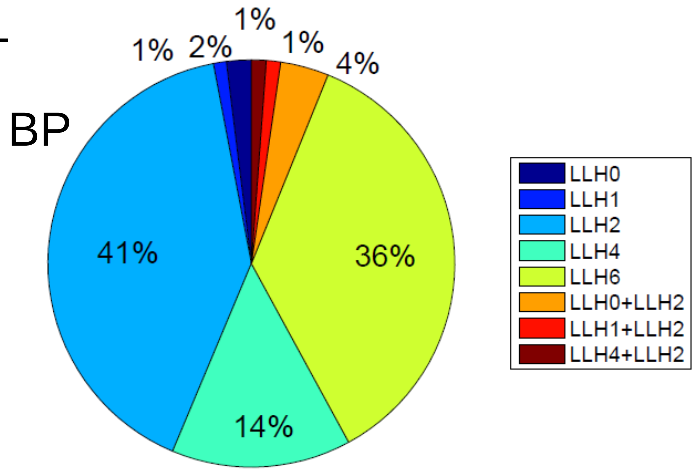
6 LLHs → **3 LLHs**

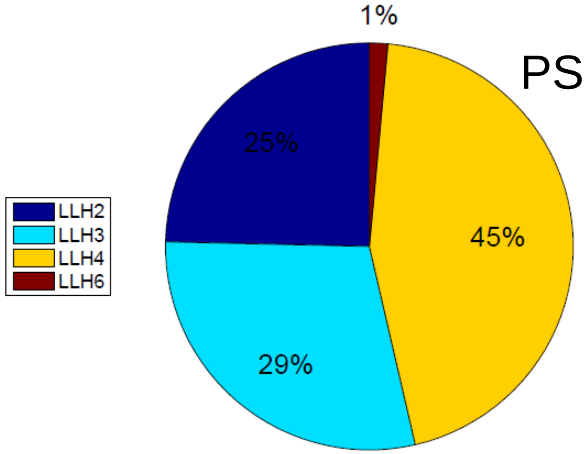
Parameter Tuning

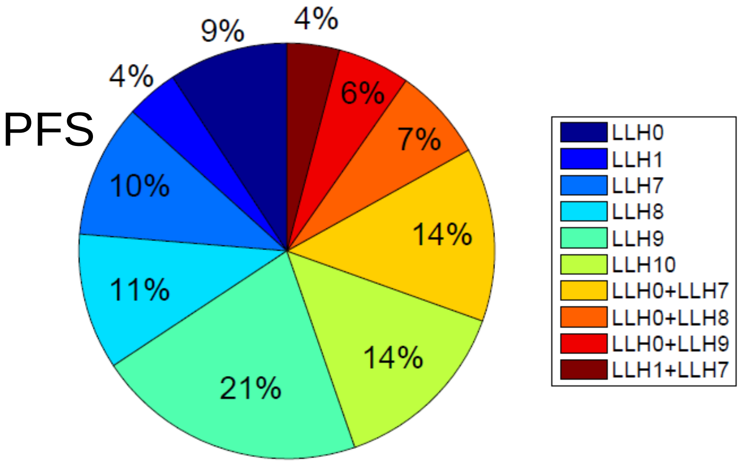
- The proposed approach introduces 6 system parameters to be set
  - $\tau = \{10, \mathbf{15}, 20, 30\}$ (in milliseconds)
  - $d = \{7, \mathbf{9}, 10, 12\}$ (in seconds)
  - $s_1 = \{10, 15, \mathbf{20}, 25\}$ (in seconds)
  - $s_2 = \{3, \mathbf{5}, 10, 15\}$ (in steps/iterations)
  - $P_{S2HH} = \{0.1, \mathbf{0.3}, 0.6, 0.9, 1.0\}$
  - $C = \{\{0\}, \{3\}, \{6\}, \{9\}, \{\mathbf{0,\ 3,\ 6,\ 9}\}\}$

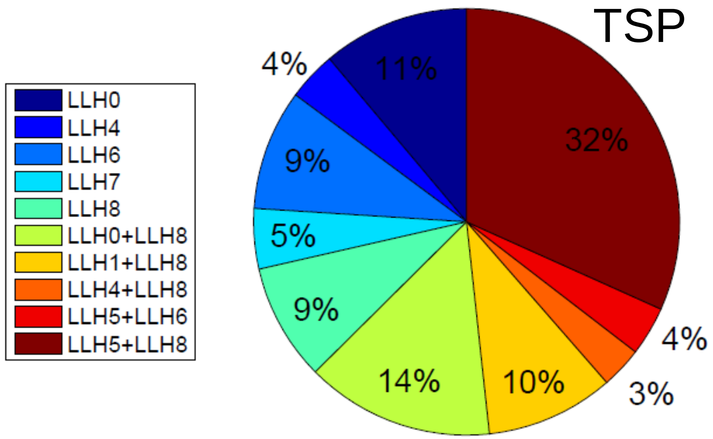
Relay hybridisation

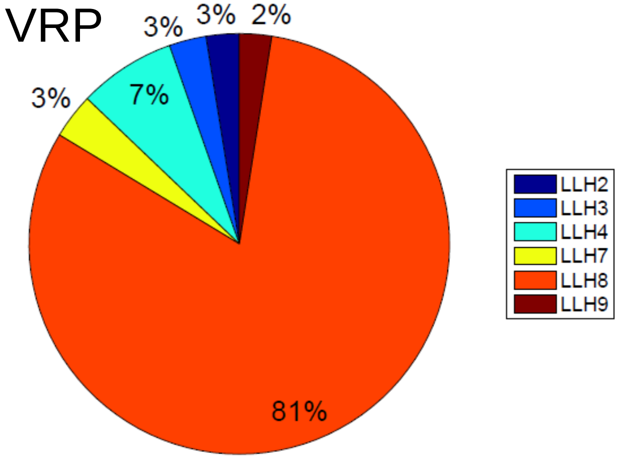










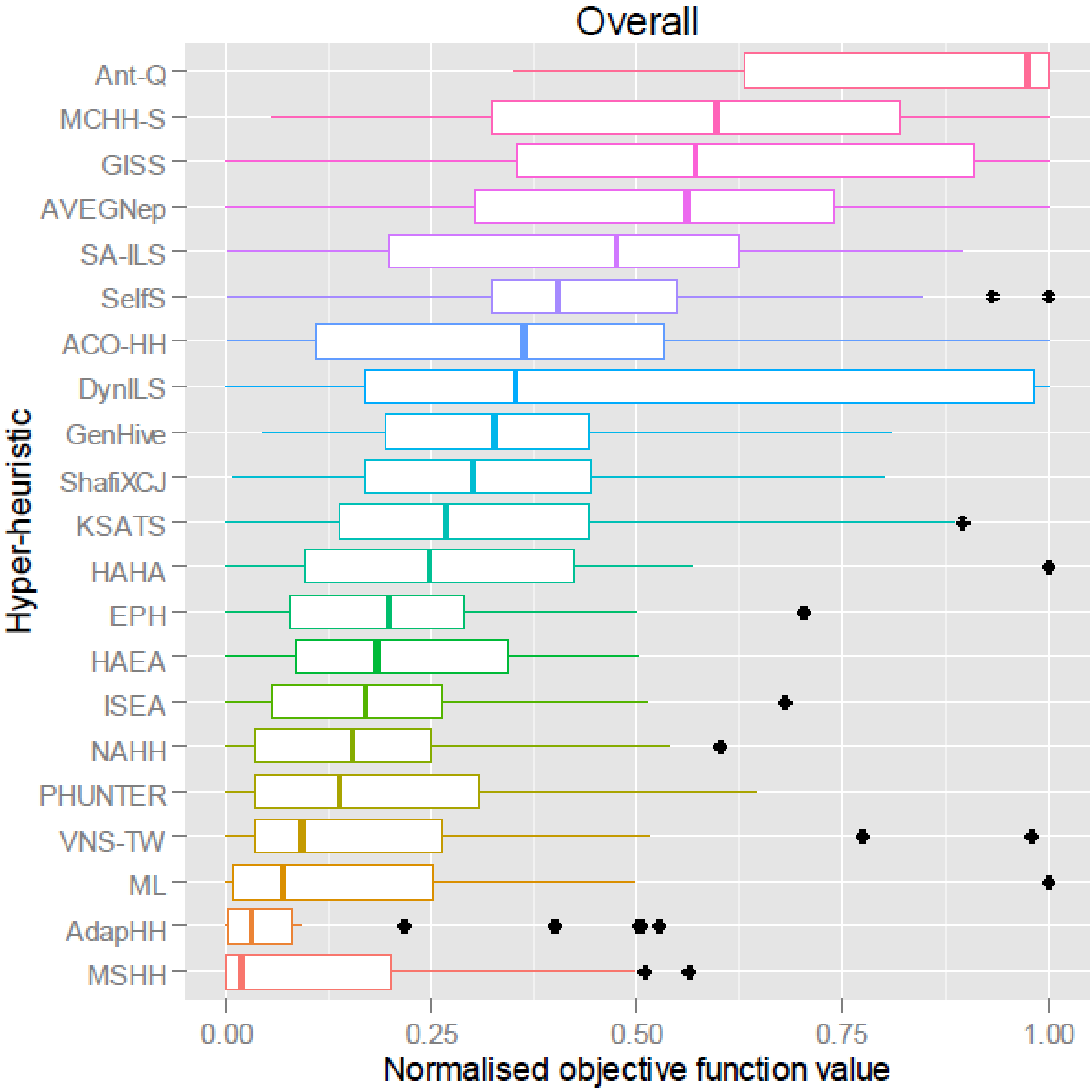


Performance Comparison

		MSHH					S1HH				S2HH		
Domain	Instance	avg.	std.	median	min.	vs.	avg.	std.	min.	vs.	avg.	std.	min.
SAT	Inst1	**0.9**	0.7	1.0	**0.0**	$>$	6.4	4.5	1.0	$>$	15.0	4.6	3.0
	Inst2	**3.1**	3.9	2.0	**1.0**	$>$	21.3	13.3	3.0	$>$	44.9	9.8	18.0
	Inst3	**0.7**	0.5	1.0	**0.0**	$>$	7.1	7.7	**0.0**	$>$	26.3	14.0	1.0
	Inst4	**1.7**	1.0	1.0	**1.0**	$>$	5.7	4.3	**1.0**	$>$	20.0	4.6	12.0
	Inst5	**7.6**	0.9	7.0	**7.0**	$>$	10.4	1.5	**7.0**	$>$	15.4	1.7	13.0
BP	Inst1	0.0163	0.0014	0.0163	**0.0136**	$<$	**0.0159**	0.0010	0.0137	$>$	0.0198	0.0015	0.0160
	Inst2	**0.0037**	0.0015	0.0030	**0.0025**	$>$	0.0061	0.0015	0.0034	$>$	0.0104	0.0021	0.0077
	Inst3	**0.0050**	0.0015	0.0049	**0.0025**	$\geq$	0.0054	0.0012	0.0027	$>$	0.0128	0.0011	0.0104
	Inst4	0.1084	0.0000	0.1084	**0.1083**	$\leq$	**0.1084**	0.0000	0.1083	$>$	0.1084	0.0000	0.1084
	Inst5	**0.0050**	0.0019	0.0044	**0.0032**	$\geq$	0.0055	0.0021	0.0032	$>$	0.0210	0.0015	0.0187
PS	Inst1	**25.5**	4.5	25.0	**16.0**	$>$	28.8	4.7	18.0	$>$	31.6	4.9	22.0
	Inst2	9668.9	217.8	9638.0	**9184.0**	$\leq$	**9645.3**	159.6	9334.0	$\leq$	9645.8	106.7	9391.0
	Inst3	**3283.7**	93.3	3270.0	**3132.0**	$\geq$	3304.8	99.6	3134.0	$\geq$	3309.9	110.2	3172.0
	Inst4	**1786.3**	172.1	1760.0	1545.0	$\geq$	1801.0	142.3	1570.0	$\geq$	1836.0	291.1	**1400.0**
	Inst5	**353.2**	21.2	350.0	**315.0**	$>$	724.4	657.3	320.0	$>$	810.7	621.5	360.0
PFS	Inst1	**6239.8**	14.9	6239.0	**6212.0**	$>$	6287.6	21.9	6249.0	$>$	6353.3	29.8	6301.0
	Inst2	26895.2	55.3	26889.0	**26775.0**	$<$	**26873.2**	30.7	26822.0	$>$	26976.9	54.7	26849.0
	Inst3	**6333.8**	19.0	6325.0	**6303.0**	$>$	6360.5	16.4	6323.0	$>$	6405.5	23.7	6369.0
	Inst4	**11363.8**	32.7	11359.0	**11320.0**	$>$	11429.9	43.8	11357.0	$>$	11529.3	35.9	11436.0
	Inst5	26711.9	47.0	26709.0	26630.0	$\leq$	**26693.1**	40.7	**26608.0**	$>$	26779.1	49.8	26702.0
TSP	Inst1	**48208.1**	31.8	48194.9	**48194.9**	$>$	50032.0	571.1	49263.1	$>$	50326.5	606.6	49221.6
	Inst2	**2.09e^{+7}**	9.05e^{+4}	2.09e^{+7}	**2.07e^{+7}**	$>$	2.14e^{+7}	1.12e^{+5}	2.12e^{+7}	$>$	2.13e^{+7}	1.05e^{+5}	2.11e^{+7}
	Inst3	**6809.1**	7.1	6808.8	**6796.6**	$>$	7012.5	30.4	6964.6	$>$	7040.2	31.3	6988.6
	Inst4	**66840.2**	276.5	66843.6	**66236.8**	$>$	68908.4	382.4	68159.9	$>$	70241.9	704.6	68791.0
	Inst5	**53011.4**	469.7	52910.2	**52341.3**	$>$	54411.1	595.1	53686.0	$>$	55814.8	946.4	53992.4
VRP	Inst1	70998.4	3840.3	70506.5	**63948.2**	$\leq$	**70223.0**	2960.2	64273.2	$>$	84103.9	7225.8	68958.3
	Inst2	**13421.8**	251.6	13359.6	**13303.9**	$\geq$	13658.0	471.4	13319.6	$>$	13695.8	473.9	13320.0
	Inst3	148498.2	1625.8	148436.2	145466.5	$\leq$	**148232.6**	1935.3	145426.5	$\geq$	149553.2	2377.8	**145362.7**
	Inst4	21016.4	488.2	20671.4	**20650.8**	$\leq$	**20991.3**	478.0	20653.5	$>$	21131.9	510.3	20657.5
	Inst5	**148813.7**	1272.5	149193.7	**146334.6**	$\geq$	148999.1	1217.1	146844.9	$>$	150282.6	1616.3	146666.9

Performance Comparison to CHeSC 2011 competitors

- Top with a CHeSC 2011 score of **163.60**

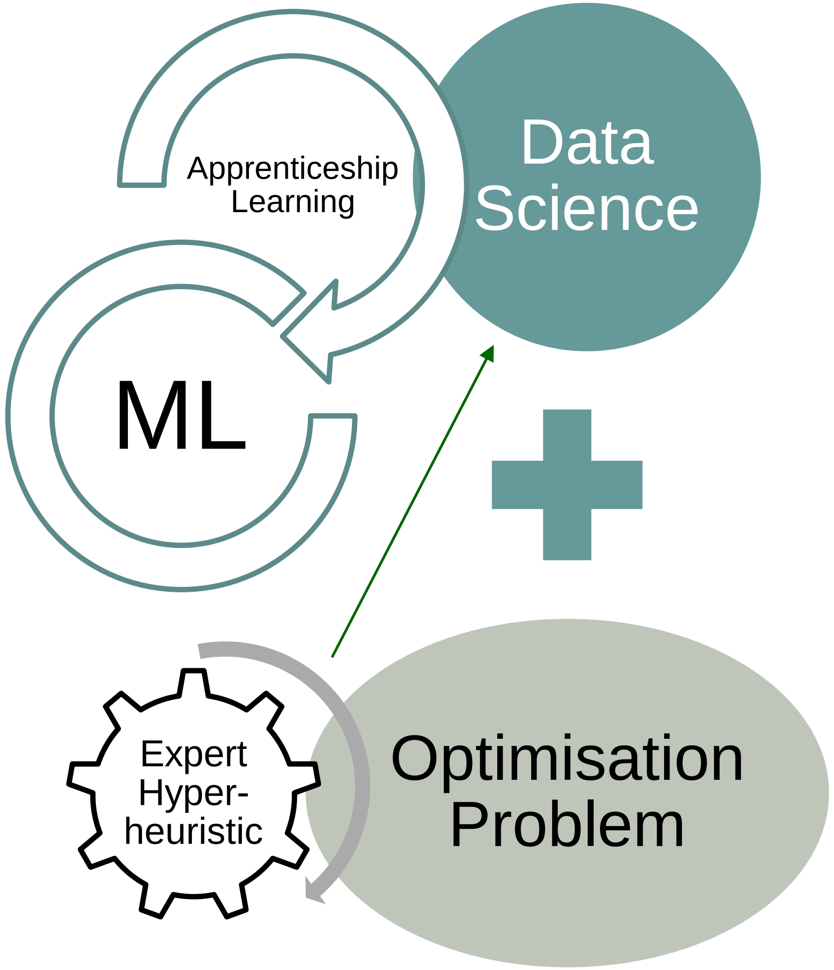


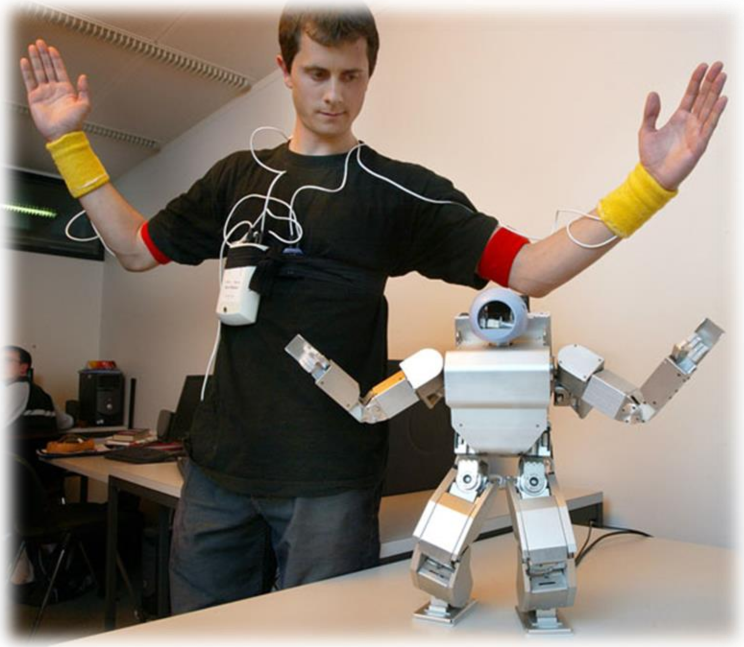
Automated Design of Selection Hyper-heuristics

- Sabar, Ayob, Kendall, and Qu (2013) used Grammatical Evolution for CVRP and ETT
- Adriaensen, Brys, and Nowé (2014a) performed a meta-level search using ILS over a set of design decisions for developing a simple selection hyper-heuristic (HyFlex)
- Sabar and Kendall (2015) used Monte Carlo Tree Search to generate heuristic select.(HyFlex)
- Ayob, Kendall, and Qu (2015a) applied Gene Expression Programming (Hyflex)
- Fontoura, Pozo, and Santana (2017) used Grammatical Evolution for protein structure prediction
- Karapetyan, Punnen, and Parkes (2017) introduced Conditional Markov Chain Search to solve the bipartite boolean quadratic programming problem
- El Yafrani et al. (2018) used Genetic Programming for the traveling thief problem
- Choong, Wong, and Lim (2018) used Reinforcement Learning to choose components of ILS based selection hyper-heuristics

Apprenticeship Learning / Learning by Demonstration Concept

P. Abbeel and A. Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in Proc. of the ICML ’04, 2004 [PDF]





http://www.calinon.ch/images/hoap2-xsens01.jpg

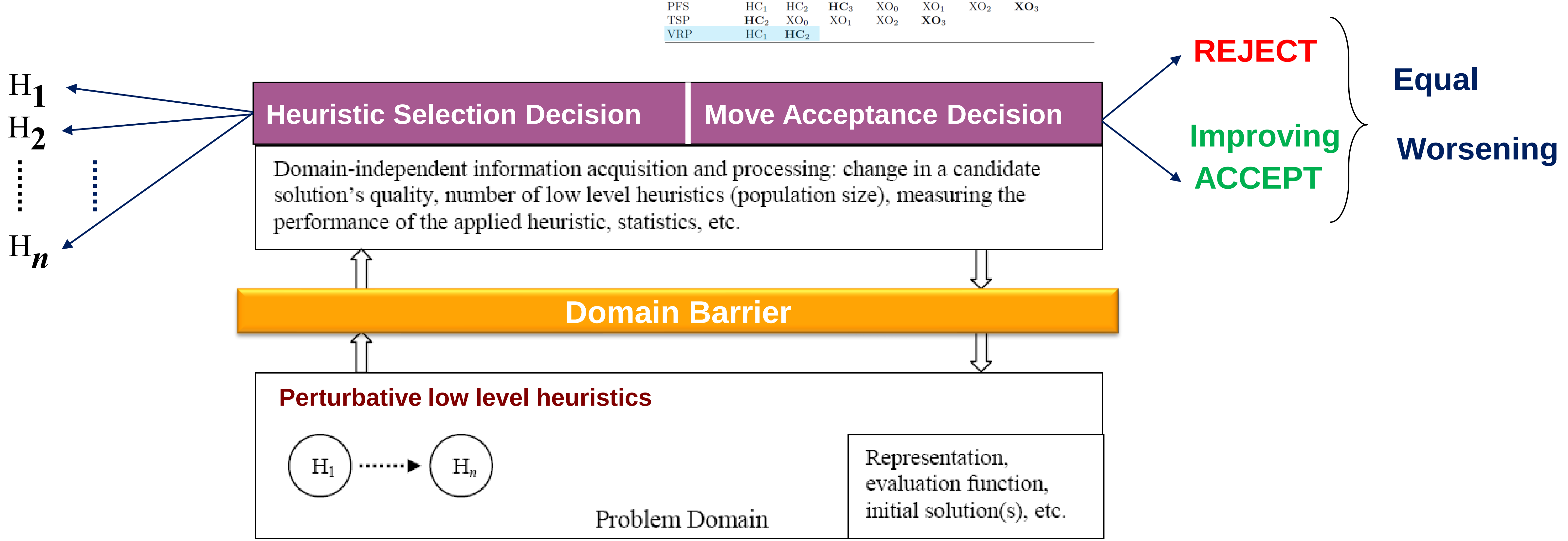
B. D. Argall, S. Chernova, M. Veloso, B. Browning, "A survey of robot learning from demonstration". *Robotics and Autonomous Systems*. **57** (5): 469–483, 2009 [PDF]

Can we automatically design a new selection hyper-heuristic by observing an expert hyper-heuristic in operation?

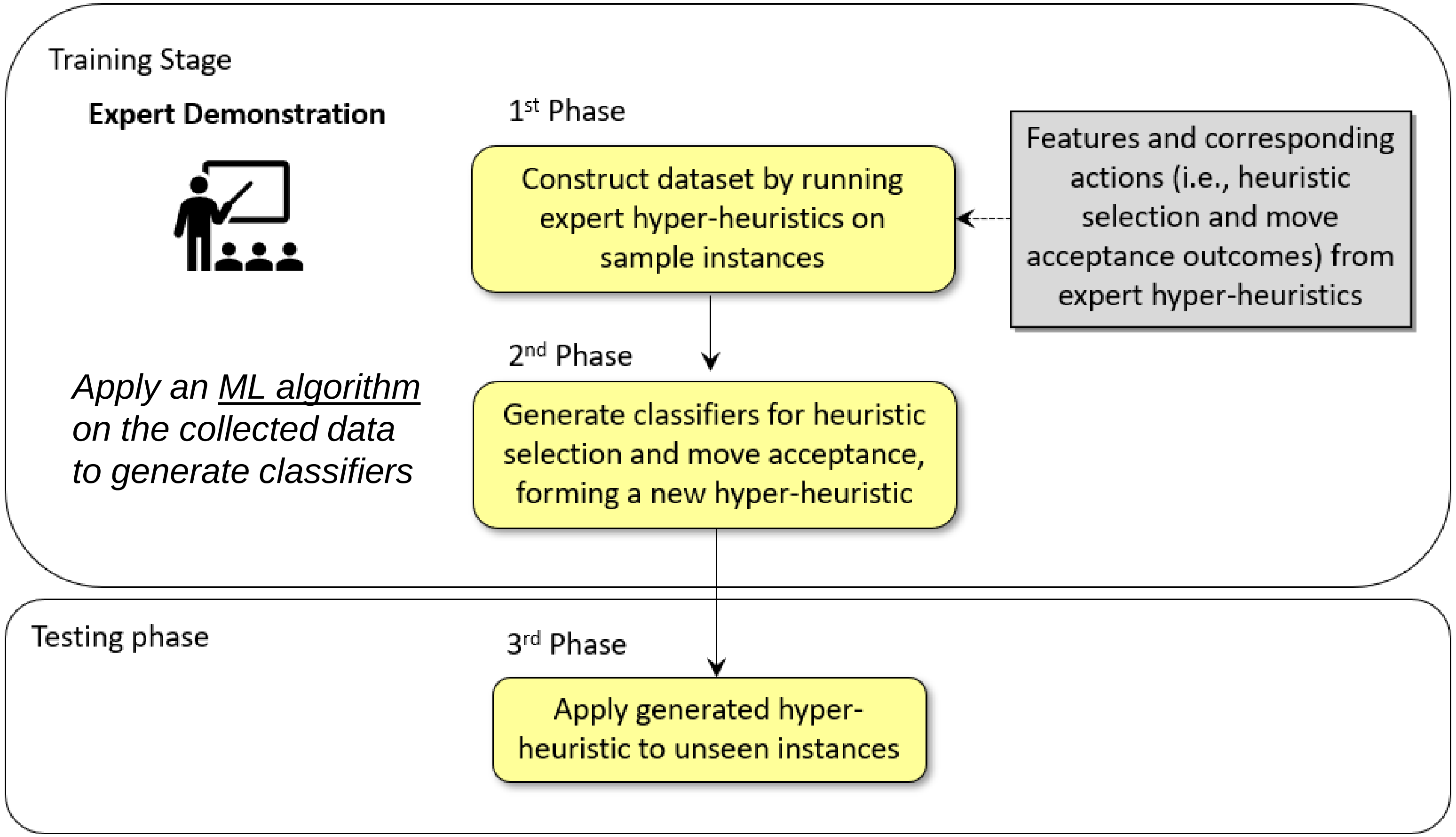
A Selection Hyper-heuristic Framework – Single Point Search

Heuristic IDs	LLH0	LLH1	LLH2	LLH3	LLH4	LLH5	LLH6	LLH7
MAX-SAT	MU_0	MU_1	MU_2	MU_3	MU_4	$\mathbf{MU_5}$	$\mathbf{RC_0}$	HC_0
Bin Packing	MU_0	RC_0	$\mathbf{RC_1}$	MU_1	HC_0	$\mathbf{MU_2}$	$\mathbf{HC_1}$	$\mathbf{XO_0}$
PS	HC_0	HC_1	HC_2	HC_3	$\mathbf{HC_4}$	RC_0	RC_1	$\mathbf{RC_2}$
PFS	MU_0	MU_1	MU_2	MU_3	$\mathbf{MU_4}$	RC_0	$\mathbf{RC_1}$	HC_0
TSP	MU_0	MU_1	MU_2	MU_3	$\mathbf{MU_4}$	$\mathbf{RC_0}$	HC_0	HC_1
VRP	MU_0	MU_1	RC_0	$\mathbf{RC_1}$	HC_0	XO_0	$\mathbf{XO_1}$	$\mathbf{MU_2}$

Heuristic IDs	LLH8	LLH9	LLH10	LLH11	LLH12	LLH13	LLH14
MAX-SAT	$\mathbf{HC_1}$	XO_0	$\mathbf{XO_1}$				
PS	XO_0	XO_1	$\mathbf{XO_2}$	$\mathbf{MU_0}$			
PFS	HC_1	HC_2	$\mathbf{HC_3}$	XO_0	XO_1	XO_2	$\mathbf{XO_3}$
TSP	$\mathbf{HC_2}$	XO_0	XO_1	XO_2	$\mathbf{XO_3}$		
VRP	HC_1	$\mathbf{HC_2}$					



Apprenticeship Learning Framework



+ Problem Domain

Domain: Open Vehicle Routing Problem

Low Level Heuristics

Parameter ‘intensity of mutation’ (IoM)

- **LLH0** - MU_0 swaps randomly selected two adjacent customers within a route
- **LLH1** - MU_1
- **LLH7** - MU_2
- **LLH2** - RC_0 selects a number of customers to remove from the solution based on their proximity to a given location.
- **LLH3** - RC_1

Parameter ‘depth of search’ (DoS)

- **LLH4** - HC_0 accepts the first improvement, repeatedly swapping the current city and the next nearest city to it
- **LLH8** - HC_1
- **LLH9** - HC_2

Heuristic IDs	LLH0	LLH1	LLH2	LLH3	LLH4	LLH5	LLH6	LLH7
MAX-SAT	MU_0	MU_1	MU_2	MU_3	MU_4	$\mathbf{MU_5}$	$\mathbf{RC_0}$	HC_0
Bin Packing	MU_0	RC_0	$\mathbf{RC_1}$	MU_1	HC_0	$\mathbf{MU_2}$	$\mathbf{HC_1}$	$\mathbf{XO_0}$
PS	HC_0	HC_1	HC_2	HC_3	$\mathbf{HC_4}$	RC_0	RC_1	$\mathbf{RC_2}$
PFS	MU_0	MU_1	MU_2	MU_3	$\mathbf{MU_4}$	RC_0	$\mathbf{RC_1}$	HC_0
TSP	MU_0	MU_1	MU_2	MU_3	$\mathbf{MU_4}$	$\mathbf{RC_0}$	HC_0	HC_1
VRP	MU_0	MU_1	RC_0	$\mathbf{RC_1}$	HC_0	XO_0	$\mathbf{XO_1}$	$\mathbf{MU_2}$
Heuristic IDs	LLH8	LLH9	LLH10	LLH11	LLH12	LLH13	LLH14	
MAX-SAT	$\mathbf{HC_1}$	XO_0	$\mathbf{XO_1}$					
PS	XO_0	XO_1	$\mathbf{XO_2}$	$\mathbf{MU_0}$				
PFS	HC_1	HC_2	$\mathbf{HC_3}$	XO_0	XO_1	XO_2	$\mathbf{XO_3}$	
TSP	$\mathbf{HC_2}$	XO_0	XO_1	XO_2	$\mathbf{XO_3}$			
VRP	HC_1	$\mathbf{HC_2}$						

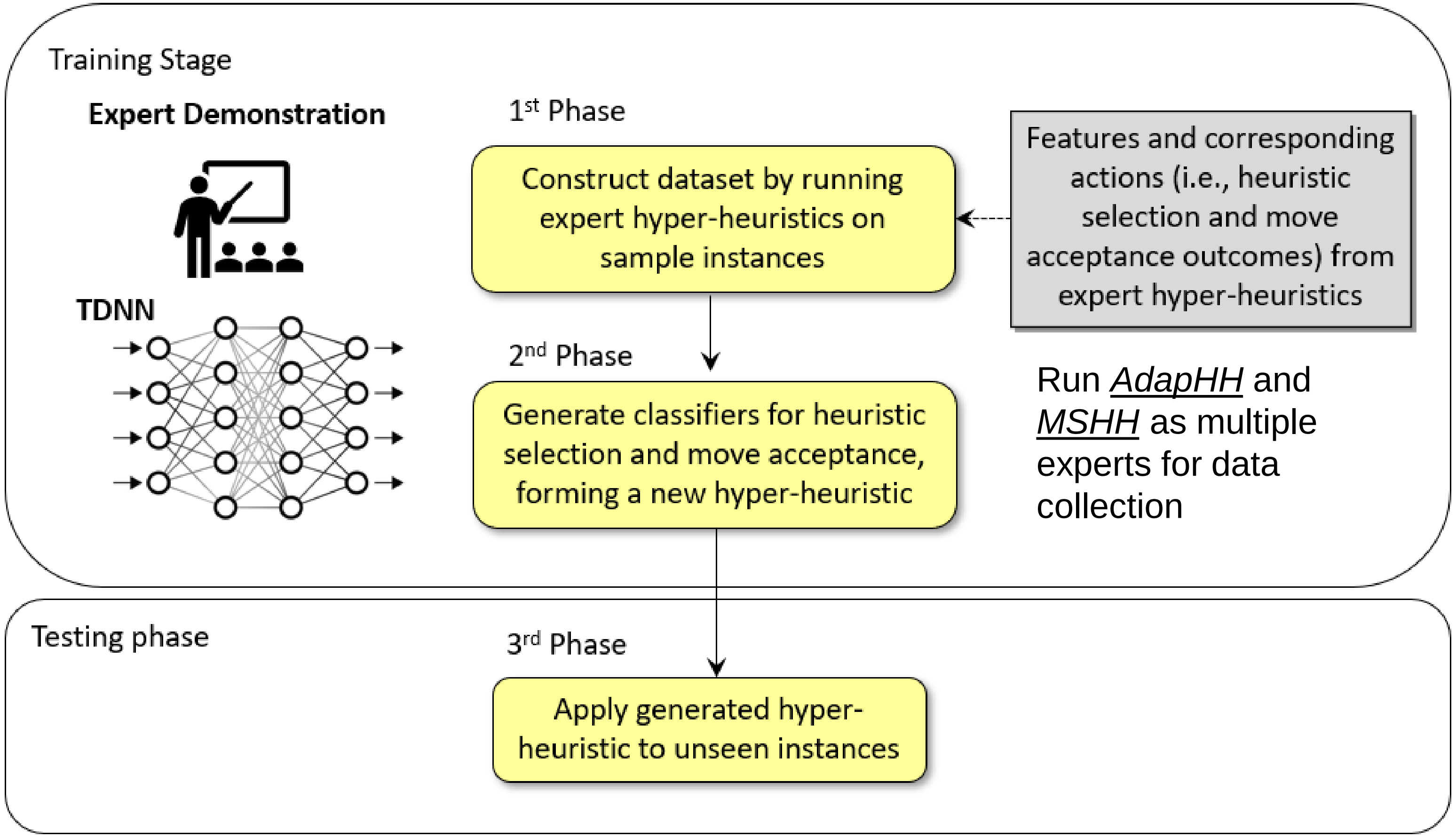
Key Observations from Initial Studies

- Different ML algorithms deliver different performances
  - TDNN, Multilayer Perceptron, C4.5
- Hyper-parameter tuning is required
  - No. of neurons in the hidden layer is the most influential factor on the hyper-heuristic performance in TDNN
  - Taguchi DoE: 24 hidden nodes, 0.07 learning rate, and 0.9 momentum
- ML generated selection hyper-heuristic performs significantly better than the expert (MCF-AM) on majority of the instances

R. Tyasnurita, E. Özcan and R. John, Learning Heuristic Selection using a Time Delay Neural Network for Open Vehicle Routing, Proc. of the 2017 IEEE Congress on Evolutionary Computation (CEC), pp. 1474-1481. [PDF]

Time Delay Neural Network Apprenticeship Learning Using Multiple Experts for OVRP

Apply Time Delay Neural Network on the collected data to generate classifiers

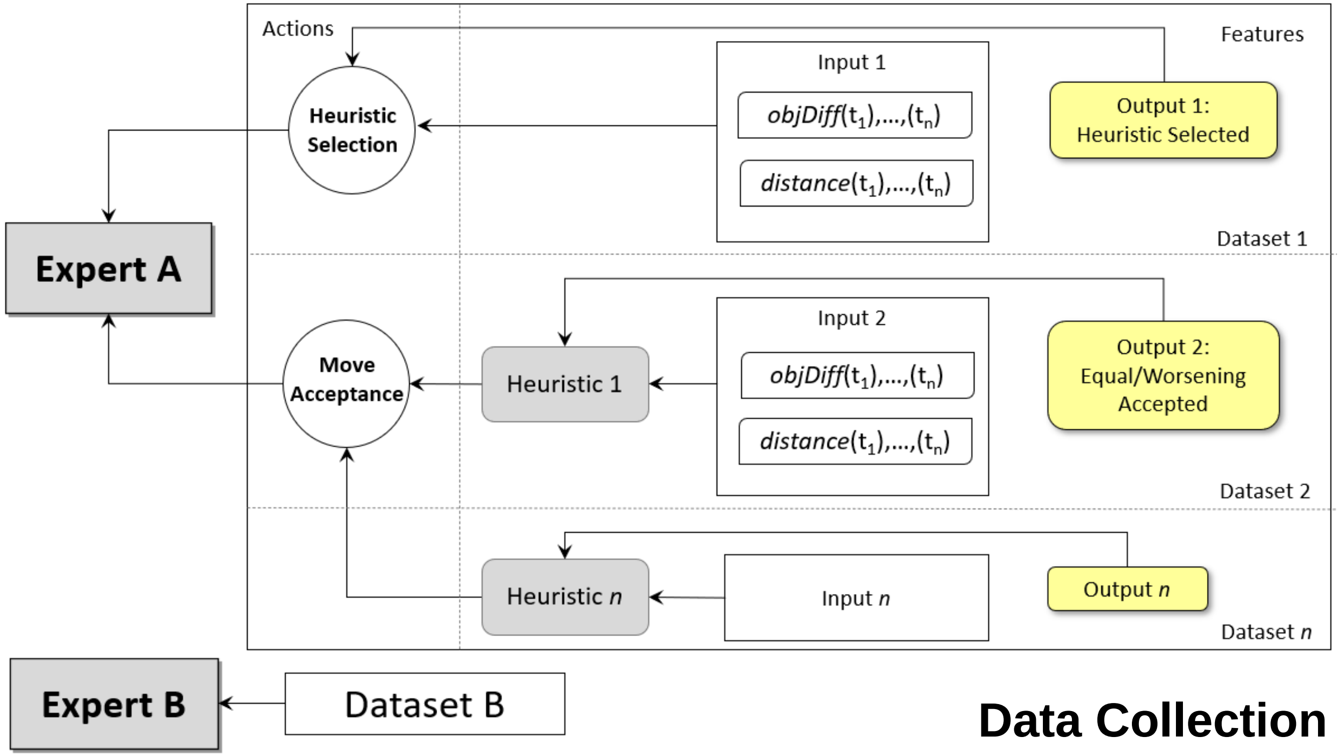


+ OVRP

R. Tyasnurita, E. Özcan, J.H. Drake, S. Asta, *Constructing selection hyper-heuristics for open vehicle routing with time delay neural networks using multiple experts*, Knowledge- Based Systems, vol. 295, 111731, 2024 [PDF]

Apprenticeship Learning for Open Vehicle Routing

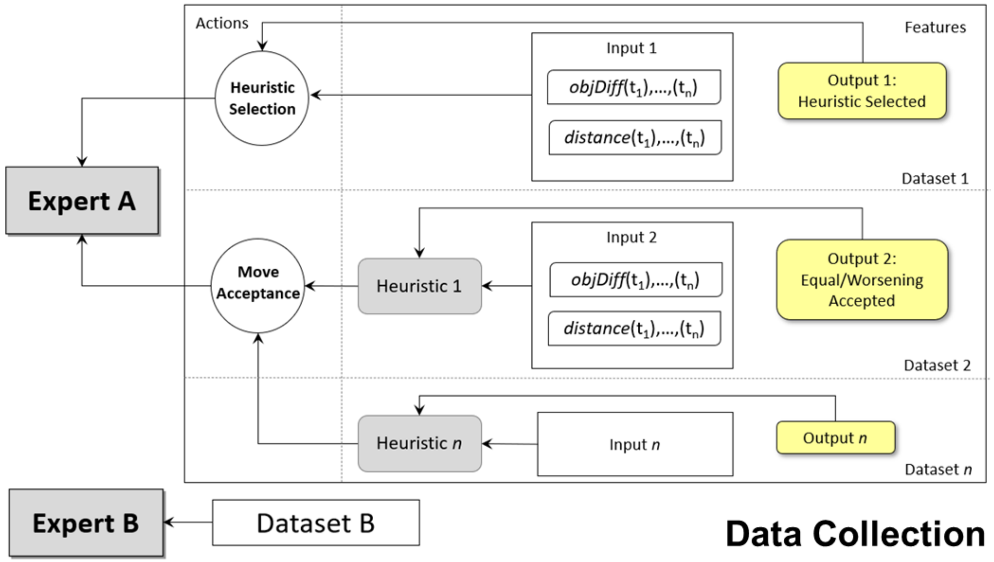
- Time Delay Neural Net
  - Expert A: AdapHH
  - Expert B: MSHH
  - intensity of mutation = 0.4
  - depth of search = 0.8
  - n = 8
  - Generated ALHH
    - ALHH-Both
    - ALHH-AdapHH
    - ALHH-MSHH



Data Collection

OVRP Benchmark

- S: Brandão (2004)
- L: Li et al. (2007)
- VL: Li et al.(2005)



Data Collection

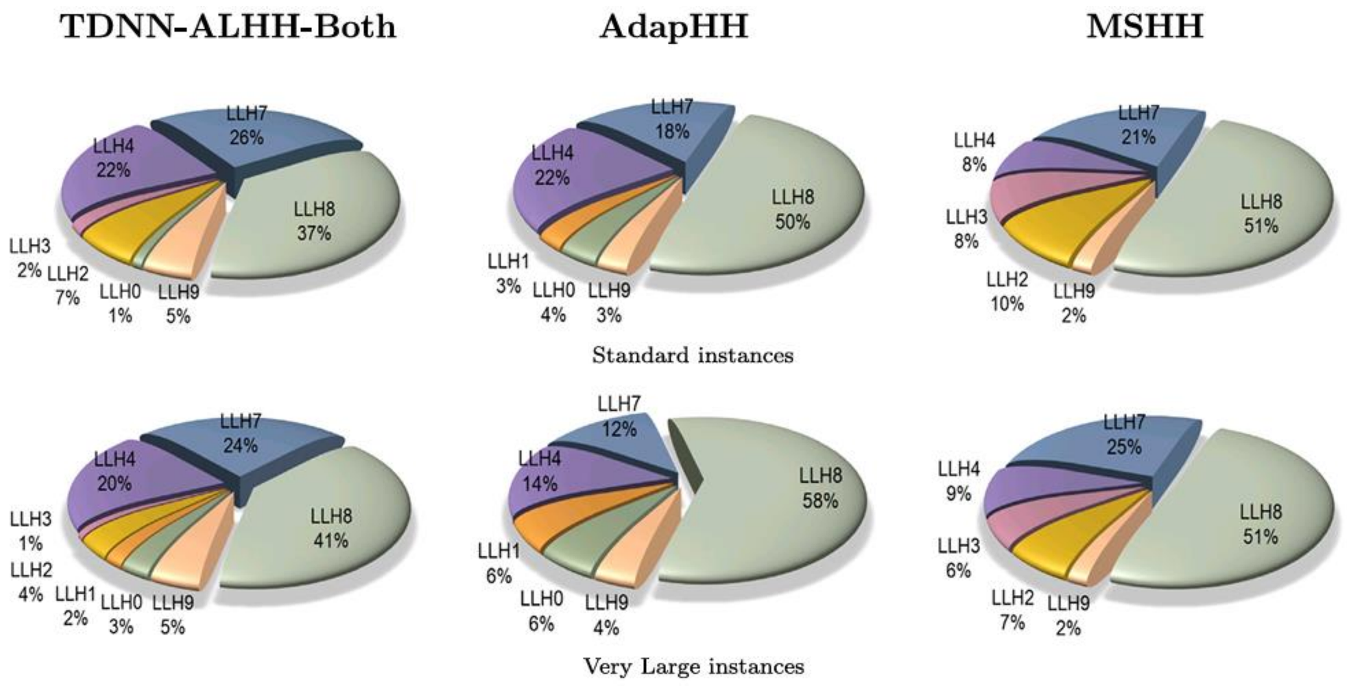
Instance ID	Customers	Vehicles	Vehicle capacity
S1	50	5	160
S2	75	10	140
S3	100	8	200
S4	150	12	200
S5	199	16	200
S6	50	5	160
S7	75	10	140
S8	100	8	200
S9	150	12	200
S10	199	16	200
S11	120	7	200
S12	100	10	200
S13	120	7	200
S14	100	10	200
L1	200	5	900
L2	240	9	550
L3	280	7	900
L4	320	10	700
L5	360	8	900
L6	400	9	900
L7	440	10	900
L8	480	10	1000
VL1	560	10	1200
VL2	600	15	900
VL3	640	10	1400
VL4	720	10	1500
VL5	760	21	900
VL6	800	11	1700
VL7	840	21	900
VL8	880	10	1800
VL9	960	10	2000
VL10	1040	10	2100
VL11	1120	10	2300
VL12	1200	10	2500

Performance of Classifiers and Low Level Heuristics

Class	Exact match ratio (%)	AUC
LLH0	98.25	0.781
LLH1	99.46	0.768
LLH2	91.58	0.864
LLH3	99.18	0.651
LLH4	75.99	0.743
LLH7	77.43	0.956
LLH8	64.32	0.805
LLH9	93.78	0.782

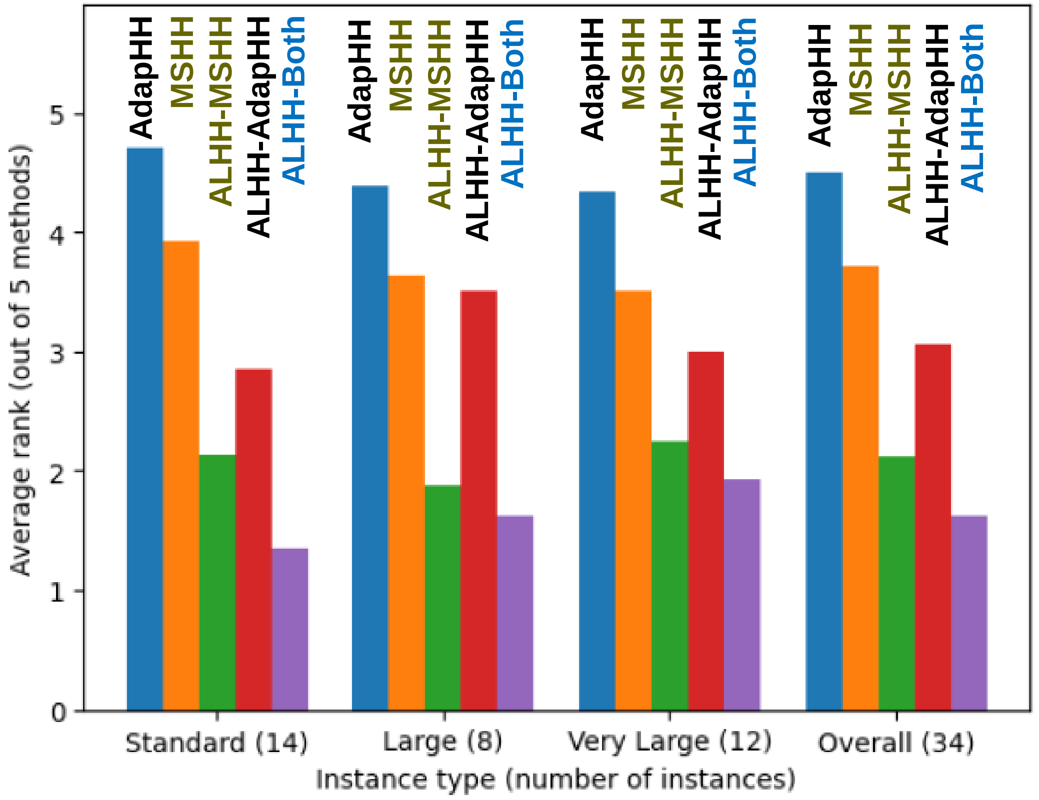
Algorithm	Local search	Mutation	Ruin-recreate
Standard			
TDNN-ALHH-Both	64%	27%	9%
AdapHH	75%	25%	0%
MSHH	61%	21%	18%
Very Large			
TDNN-ALHH-Both	66%	29%	5%
AdapHH	76%	24%	0%
MSHH	62%	25%	13%

LLH0 - MU_0, **LLH1** - MU_1, **LLH7** - MU_2, **LLH2** - RC_0, **LLH3** - RC_1, **LLH4** - HC_0, **LLH8** - HC_1, **LLH9** - HC_2



Experimental Results

- ALHH-Both is the top ranking approach (1.62)
- ALHH-MSHH is the second best (2.12)
- ALHH-Both performs significantly better than all, except ALHH-MSHH
- ALHH-AdapHH and ALHH-MSHH performs significantly better than AdapHH and MSHH
- AdapHH (4.50, no best performance)



Training & Testing

- '>' : *Algorithm 1* performs better than *Algorithm 2* with a statistically significant difference in the performance within a confidence interval of 95% over 30 runs, and '<' indicates vice versa.
- Non-statistically significant differences in performance are denoted by '≈'.

Algorithm 1	Algorithm 2	>	<	≈
Standard (train) to standard (test)				
TDNN-ALHH-Both	TDNN-ALHH-AdapHH	10	1	3
TDNN-ALHH-Both	TDNN-ALHH-MSHH	10	1	3
TDNN-ALHH-Both	AdapHH	11	1	2
TDNN-ALHH-Both	MSHH	10	1	3
TDNN-ALHH-AdapHH	AdapHH	11	0	3
TDNN-ALHH-MSHH	MSHH	10	0	4
Standard (train) to large (test)				
TDNN-ALHH-Both	TDNN-ALHH-AdapHH	5	0	3
TDNN-ALHH-Both	TDNN-ALHH-MSHH	5	0	3
TDNN-ALHH-Both	AdapHH	6	1	1
TDNN-ALHH-Both	MSHH	6	0	2
TDNN-ALHH-AdapHH	AdapHH	5	2	1
TDNN-ALHH-MSHH	MSHH	5	1	2
Standard (train) to very large (test)				
TDNN-ALHH-Both	TDNN-ALHH-AdapHH	5	1	6
TDNN-ALHH-Both	TDNN-ALHH-MSHH	5	0	7
TDNN-ALHH-Both	AdapHH	5	1	6
TDNN-ALHH-Both	MSHH	5	0	7
TDNN-ALHH-AdapHH	AdapHH	6	1	5
TDNN-ALHH-MSHH	MSHH	6	0	6
Large (train) to very large (test)				
TDNN-ALHH-Both	TDNN-ALHH-AdapHH	7	3	2
TDNN-ALHH-Both	TDNN-ALHH-MSHH	7	1	4
TDNN-ALHH-Both	AdapHH	7	0	5
TDNN-ALHH-Both	MSHH	7	0	5
TDNN-ALHH-AdapHH	AdapHH	7	1	4
TDNN-ALHH-MSHH	MSHH	7	2	3
Standard + Large (train) to very large (test)				
TDNN-ALHH-Both	TDNN-ALHH-AdapHH	7	1	4
TDNN-ALHH-Both	TDNN-ALHH-MSHH	7	0	5
TDNN-ALHH-Both	AdapHH	9	2	1
TDNN-ALHH-Both	MSHH	9	1	2
TDNN-ALHH-AdapHH	AdapHH	8	2	2
TDNN-ALHH-MSHH	MSHH	8	1	3

Concluding Remarks I

- HyFlex framework has been in use for hyper-heuristic research and benchmarking for more than a decade
  - Put search methods on a much more experimentally rigorous footing
  - Build an invaluable communal resource that is of benefit to both practitioners and researchers
- The standard mode of using hyper-heuristics

  “One-off with opaque domain barrier and no learning between instances”

  can be greatly extended without loss of domain independence
- The need for more flexible/modular tools with reusable components supporting information rich environments is growing

J. Swan et al., Metaheuristics “In the Large”, EJOR, Vol. 297, Issue 2, pp. 393-406, 2022 [PDF]

Concluding Remarks II

- A novel train-and-test approach to automated generation of selection hyper-heuristics based on Apprenticeship Learning is investigated
  - ML can effectively find hidden patterns in the heuristic selection and move acceptance choices of the multiple experts
  - ‘Student surpasses the teacher’
- There are still many issues/open questions to be explored:
  - Data collection
  - Feature engineering
  - Data balancing
  - Generalisation
  - Which ML method to use for what purpose
    - Mixing different ML approaches (parameter setting)
  - Explainability

Q&A

Thank you.

Ender.Ozcan@nottingham.ac.uk

EWG DSO: EURO working group on Data Science meets Optimization
goo.gl/6Fe1EX

University of Nottingham, School of Computer Science
Jubilee Campus, Wollaton Road, Nottingham
NG8 1BB, UK
http://www.cs.nott.ac.uk/~pszeo/