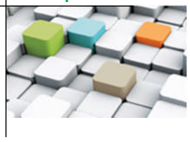
Fundamentals of Hyper-heuristics

Ender Özcan

School of Computer Science

The University of Nottingham
UNITED KINGDOM · CHINA · MALAYSIA



AI-2014

Thirty-fourth SGAI International Conference on Artificial Intelligence

Workshop Stream 2

http://www.cs.nott.ac.uk/~exo/AI2014-HH-slides.pdf

Outline

- Basics – Heuristics
- Hyper-heuristics
  - Definition, Classification, Origins
- Generation Hyper-heuristics
  - Case Studies
- Selection Hyper-heuristics
  - HyFlex and Cross-domain Heuristic Search Challenge 2011
  - Case Studies

2

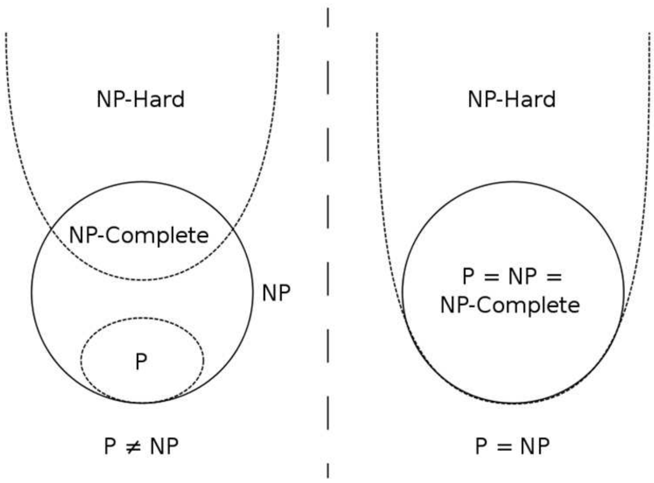
Need for Search Methodologies (Heuristics/Metaheuristics/ Hyper-heuristics, etc...) – Example

- Travelling salesman problem
- N=3, 6
- N=5, 120
- N=7, 5 040
- N=10, 3 628 800
- N=81, ~5.8 x10^{120}

- Number of particles in the universe is in between 10^{72} – 10^{87}
- Tianhe-2: 30.65 PF (10^{15}), ~6 x10^{96} years

3

Problem Classes



4

Heuristic Search

- Heuristics are rule of thumb methods
- They are informal, judgmental knowledge of area which can be used to arrive at "good" enough solutions to some “hard” problems.
- Good for solving
  - ill-structured problems, or
  - complex well-structured problems (large-scale combinatorial problems that have many potential solutions to explore)

5

Search Paradigms

- Single point based search vs. Multi-point (population) based search
- Constructive
  - partial candidate solutions
- Perturbative
  - complete solutions

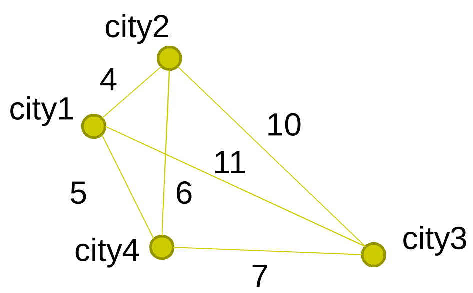
6

Examples – Heuristics for TSP

- **The nearest neighbour (NN) algorithm**
  - Constructive
- **Pairwise exchange (2-opt)**, or Lin–Kernighan heuristics
  - Perturbative

7

The nearest neighbour (NN) algorithm



8

The nearest neighbour (NN) algorithm

<city2>

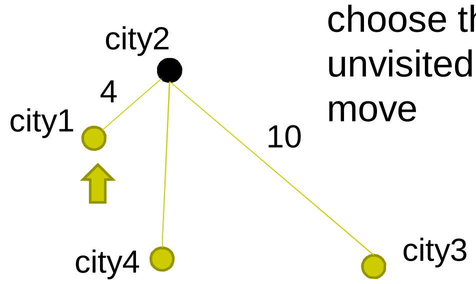
Select a starting city

9

The nearest neighbour (NN) algorithm

<city2, >

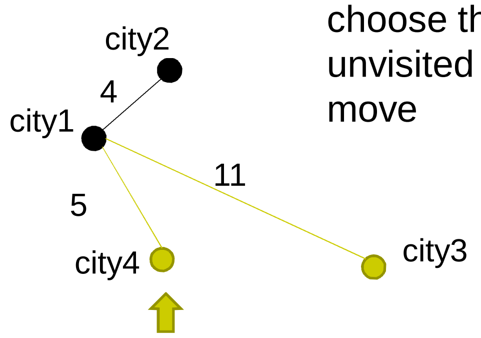
choose the nearest unvisited city as the next move



10

The nearest neighbour (NN) algorithm

<city2, city1, >

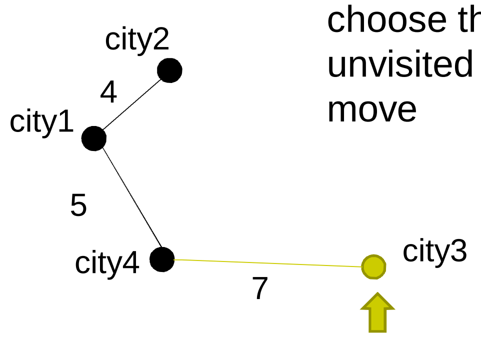


choose the nearest unvisited city as the next move

11

The nearest neighbour (NN) algorithm

<city2, city1, city4, >



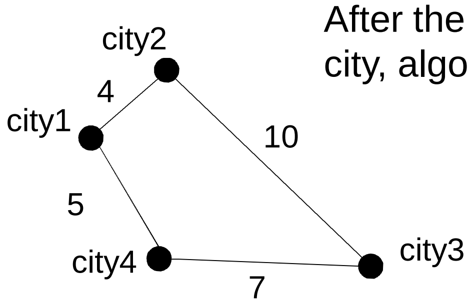
choose the nearest unvisited city as the next move

12

The nearest neighbour (NN) algorithm

<city2, city1, city4, city3> : 26

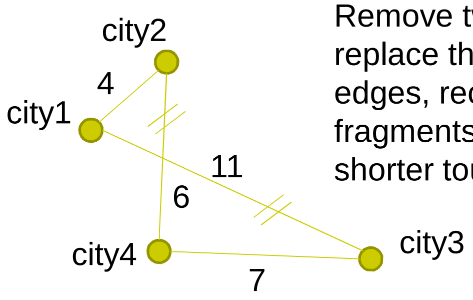
After the choice of the last city, algorithm terminates



Pairwise exchange (2-opt)

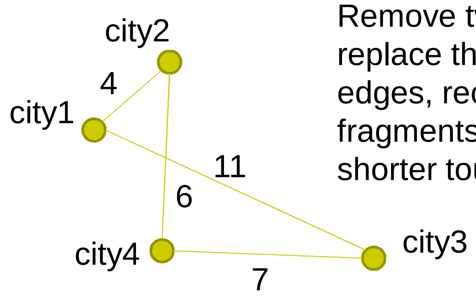
<city2, city1, city3, city4> : 28

Remove two edges and replace them with two different edges, reconnecting the fragments into a new and shorter tour.



Pairwise exchange (2-opt)

<city2, city1, city3, city4> : 28

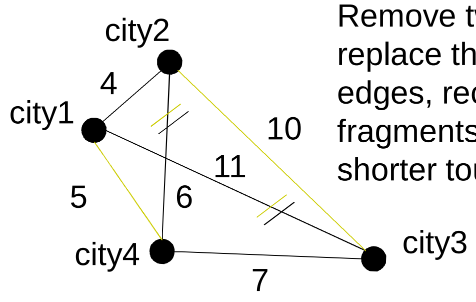


Remove two edges and replace them with two different edges, reconnecting the fragments into a new and shorter tour.

15

Pairwise exchange (2-opt)

<city2, city1, city3, city4> : 28

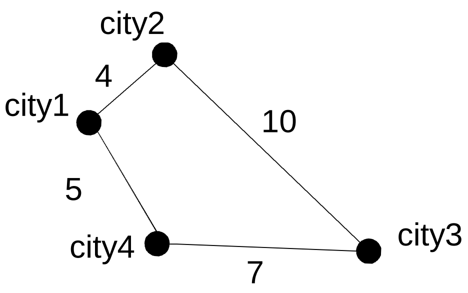


Remove two edges and replace them with two different edges, reconnecting the fragments into a new and shorter tour.

16

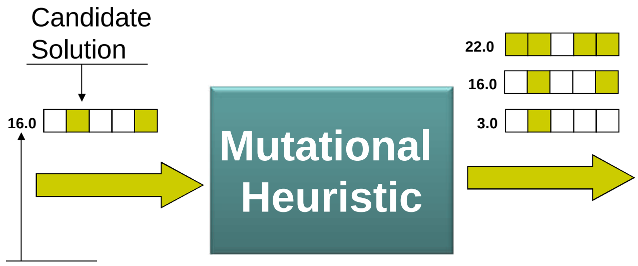
Pairwise exchange (2-opt)

<city2, city1, city3, city4> : 26



Mutational Heuristic

Processes a given candidate solution and generates a solution which is not guaranteed to be better than the input



Minimising Fitness /Cost/Penalty/…

e.g., total number of constraint violations or a weighted sum of violations

Hill Climbing Heuristic

Processes a given candidate solution and generates a better or equal quality solution

Minimising Fitness /Cost/Penalty/…

e.g., total number of constraint violations or a weighted sum of violations

19

Hyper-heuristics

Problem with Heuristics? – Bin Packing

- Place a set of *N* items with given sizes {e.g., *N*=33 items: 1x85, 1x442, 6x10, 7x252, 2x9 , 5x127, 4x106, 3x12, 1x84, 1x46, 2x37} into minimal number of bins, each having a fixed capacity of *C* (e.g., *C*=524)

How would you do it?

21

Sort Items → First Fit Heuristic

	Bin1	Bin2	Bin3	Bin4	Bin5	Bin6	Bin7
442	442						
252		252					
252		252					
252			252				
252			252				
252				252			
252				252			
252					252		
127					127		
127					127		
127						127	
127						127	
127						127	
106						106	
106							106
106							106
106							106
85							85
84							84
46 – removed							
37						37	
37							37
12	12						
12	12						
12	12						
10		10					
10		10					
10			10				
10			10				
10				10			
10				10			
9					9		
9					9		
	524	524	524	524	524	524	524

Instance#1

Bin1	Bin2	Bin3	Bin4	Bin5	Bin6	Bin7	Bin8
442							
	252						
	252						
		252					
		252					
			252				
			252				
				252			
				127			
				127			
					127		
					127		
					127		
					106		
						106	
						106	
						106	
						85	
						84	
46 – removed							
37							
37							
	12						
		12					
			12				
				10			
					10		
					10		
					10		
						10	
						10	
						9	
							9
516	516	516	516	516	517	516	9

Instance#2

Problem with Heuristics? – Examination Timetabling

Problem	[24]	[19]	[21]	[26]	[20]	[12]	[27]	[28]	ALC
car91	7.1	**4.97**	5.03	*4.97*	5.17	6.6	<u>4.6</u>	4.8	5.12
car92	6.2	4.32	**4.22**	*4.28*	4.32	6.0	<u>3.9</u>	4.1	4.41
ears83 I	36.4	36.16	**36.06**	35.86	*35.70*	<u>29.3</u>	32.8	34.92	36.91
hec92 I	**10.8**	11.61	11.71	11.85	11.93	<u>9.2</u>	10.0	10.73	*11.31*
kfu93	**14.0**	15.02	16.02	*14.62*	15.30	13.8	<u>13.0</u>	<u>13.0</u>	14.75
lse91	**10.5**	10.96	11.15	*11.14*	11.45	<u>9.6</u>	10.0	10.01	11.41
pur93 I	**3.9**	-	-	*4.73*	-	<u>3.7</u>	-	4.73	5.87
rye92	**7.3**	-	9.42	9.65	-	<u>6.8</u>	-	9.65	*9.61*
sta83 I	161.5	161.90	158.86	158.33	159.05	158.2	<u>156.9</u>	158.26	***157.52***
tre92	9.6	8.38	**8.37**	*8.48*	8.68	9.4	7.9	<u>7.88</u>	8.76
uta92 I	3.5	**3.36**	3.37	3.40	*3.30*	3.5	<u>3.2</u>	<u>3.2</u>	3.54
ute92	**25.8**	27.41	27.99	28.88	28.00	<u>24.4</u>	24.8	26.11	*26.25*
yor83 I	41.7	40.77	**39.53**	40.74	40.79	36.2	<u>34.9</u>	36.22	*39.67*

S. Abdul-Rahman, A. Bargiela, E. K. Burke, E. Özcan, B. McCollum and P. McMullan, Adaptive Linear Combination of Heuristic Orderings in Constructing Examination Timetable, European Journal of Operational Research, 232 (2), pp. 287-297, 2014

Metaheuristic – Definition

A high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimisation algorithms

<u>Source</u>: Glover, F. And Sorensen, K. In: Encyclopaedia of OR/MS, Springer Verlag, Berlin (to appear)

Sophisticated Metaheuristics

- Simulated annealing
- Tabu search
- Iterated Local Search
- GRASP
- Evolutionary computation
  - Evolutionary strategies, Genetic algorithms, Memetic algorithms, Genetic programming
- Ant colony optimization

and more…

25

Random Mutation Hill Climbing vs. Iterated Local Search

Algorithm 1: Pseudocode of the iterated local search method

```
Let S represent the candidate solution;
S_initial ← CreateInitialSolution();
S ← LocalSearch(S_initial);
repeat
    S' ← Perturbation(S);
    S'' ← LocalSearch(S');
    if Accept(S, S'') then
        S ← S'';
    end
until TerminationCriteriaSatisfied();
```

H. R. Lourenco, O. C. Martin, and T. Stutzle. Iterated local search: framework and applications. In M. Gendreau and J.-Y. Potvin (eds), *Handbook of Metaheuristics*, vol. 146 of *International Series in Operations Research and Management Science*, pp. 363–397, 2010.

26

Genetic Algorithm vs. Memetic Algorithm

Algorithm 2: Pseudocode of memetic algorithm

```
CreateInitialSolutions(); // create initial population of solutions
LocalSearch();
repeat
    Evaluate(); // calculate fitness of each solution in the population
    SelectParents(); // select solutions from the population to breed
    Crossover(); // apply crossover operator with a given probability
    Mutate(); // apply mutation operator with a given probability
    LocalSearch();
    Evaluate();
    ReplaceSolutions(); // generate new population of solutions
until TerminationCriteriaSatisfied();
```

Moscato, P.: On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms, Caltech Concurrent Computation Program Report 826, California Institute of Technology (1989)

27

OBSERVATIONS

- Most of the real-world problems are proven to be NP-hard
- The current state of the art in search methodologies (i.e., metaheuristics) tend to focus on bespoke systems
  - In general, these systems are expensive to build, but provide successful results
  - Unfortunately, their application to new problem domains or even new problem instances from a known domain or a slight change in the problem definition could still require expert involvement.

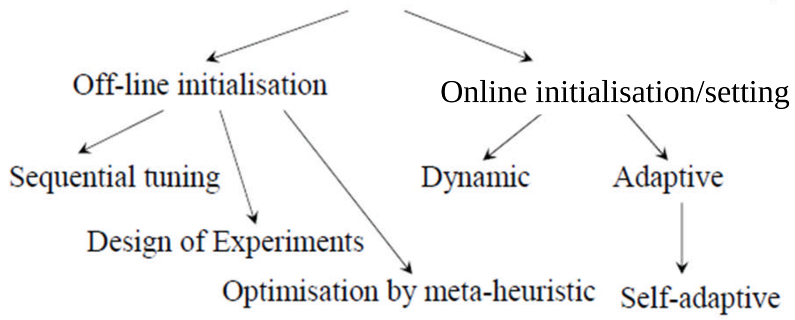
28

Drawbacks of (meta)heuristic search

- There is no guarantee for the optimality of the obtained solutions.
  - May give a poor solution.
- Usually can be used only for the specific situation for which they are designed.
- Often, (meta)heuristics have some parameters
  - Performance of a heuristic could be sensitive to the setting of those parameters

29

Parameter Tuning Parameter Control



- **ParamILS:** F. Hutter, D. Babic, H. H. Hoos, and A. J. Hu, "Boosting verification by automatic tuning of decision procedures," in Proc. of the Formal Methods in Computer Aided Design, ser. FMCAD '07. IEEE Computer Society, 2007, pp. 27–34.
- **iRace:** M. Lopez-Ibanez, J. Dubois-Lacoste, T. Stutzle, and M. Birattari, "The irace package, iterated race for automatic algorithm configuration," IRIDIA, Universite Libre de Bruxelles, Belgium, Tech. Rep. TR/IRIDIA/2011-004, 2011

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 general purpose search methodologies with the common goal of automating the design and tuning of heuristic methods

31

Motivation – Grand Challenge

- Automating the search/heuristic design process
  - Motivated by raising the level of generality.

More General

A B C

Problem Specific Solvers

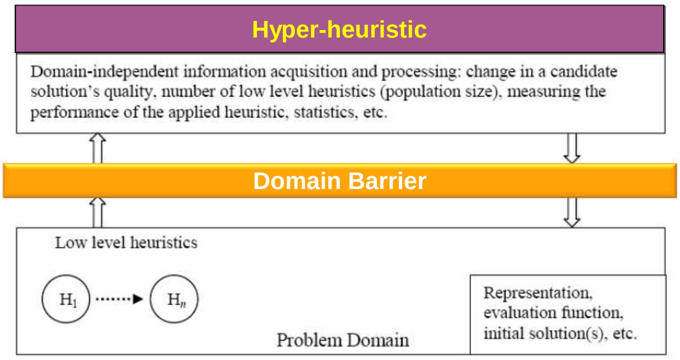
Significant scope for future research

The General Solver

Doesn't exist….

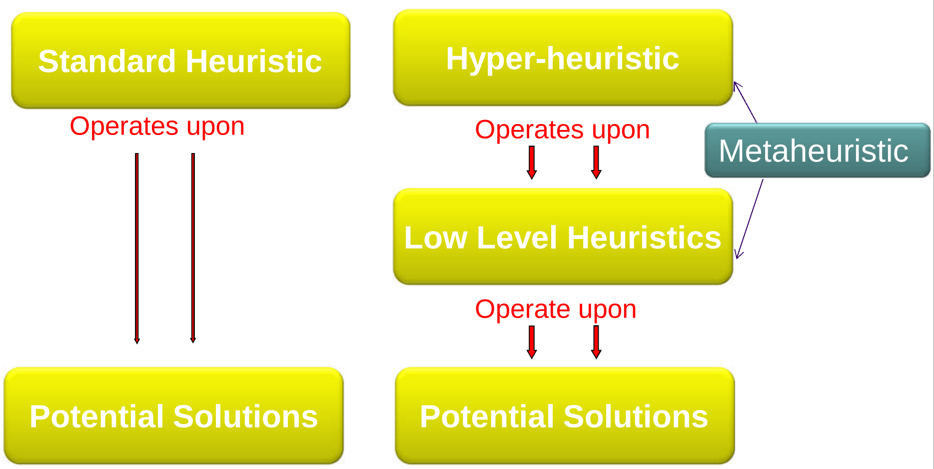
32

A Hyper-heuristic Framework



33

Different Search Spaces



34

Characteristics of Hyper-heuristics

- Operate on a **search space of heuristics** rather than directly on a search space of solutions
- Existing (or computer generated) heuristics can be used within hyper-heuristics
- Aims to take advantage of strengths and avoid weaknesses of each heuristic
- No problem specific knowledge is required during the search over the heuristics space (and so hyper-heuristic components are reusable)
- Easy to implement/deploy/use (easy, cheap, fast)
- Applicable to a range of real-world problems
- Extremely desirable: Employs data science (i.e., machine learning) techniques

35

Related Areas

- Reactive search
- Algorithm portfolios
- Co-evolution, multimeme memetic algorithms
- Adaptive operator selection
- Parameter tuning
- Parameter control in EAs
- Variable Neighbourhood Search
- Meta-learning
- Algorithm configuration
- Cooperative Search

…

36

A Classification of Hyper-heuristics

Feedback

- Online learning
- Offline learning
- No-learning

Hyper-heuristics

Nature of the heuristic search space

Heuristic generation Methodologies to generate	constructive heuristics
	perturbative heuristics
Heuristic selection Methodologies to select	constructive heuristics
	perturbative heuristics

37

Hyper-heuristics: Origins

1961-63 — 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. Probabilistic Learning Combinations of Local Job-shop Scheduling Rules. Ch 15,:225-251, Prentice Hall, New Jersey, 1963

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

MANAGEMENT SCIENCE
Vol. 38, No. 10, October 1992
Printed in U.S.A.

NEW SEARCH SPACES FOR SEQUENCING PROBLEMS WITH APPLICATION TO JOB SHOP SCHEDULING*

ROBERT H. STORER, S. DAVID WU AND RENZO VACCARI
Department of Industrial Engineering, Lehigh University, Bethlehem, Pennsylvania 18015

A Promising Hybrid GA/Heuristic Approach for Open-Shop Scheduling Problems

Hsiao-Lan Fang[1] and Peter Ross[1] and Dave Corne[2]
In Proceedings of the 11th European Conference on Artificial Intelligence. John Wiley and Sons. 1994. pages 590–594.

High Performance ATP Systems by Combining Several AI Methods

Jörg Denzinger, Matthias Fuchs

reproduce this run. So $\mathcal{C}$ can be seen as the description of a "hyper-heuristic" and is used instead of a single heuristic $\mathcal{H}$ when storing data regarding distributed proofs.

67653 Kaiserslautern
Germany
E-mail: {denzinge|fuchs}@informatik.uni-kl.de

Marc Fuchs
Fakultät für Informatik
TU München
80290 München
Germany
E-mail: fuchsm@informatik.tu-muenchen.de

October 29, 1996

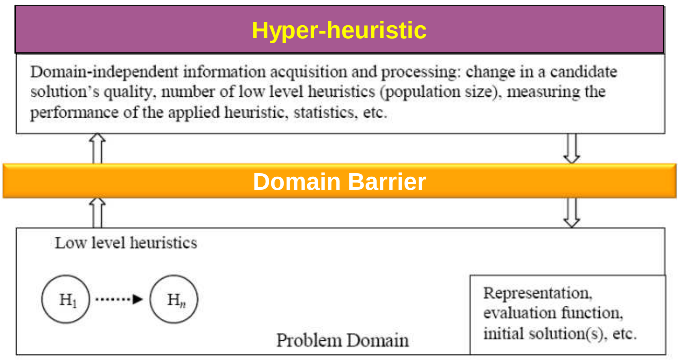
38

Generation Hyper-heuristics

The University of Nottingham

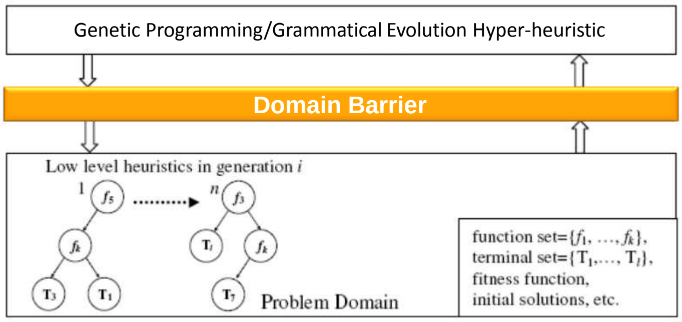
UNITED KINGDOM · CHINA · MALAYSIA

A Hyper-heuristic Framework – revisited



44

A Generation Hyper-heuristic Framework



45

Some Java based Software Libraries

- ECJ: http://cs.gmu.edu/~eclab/projects/ecj/
- TinyGP: http://cswww.essex.ac.uk/staff/rpoli/TinyGP/
- GEVA (grammatical evolution): http://ncra.ucd.ie/Site/GEVA.html
- Cartesian GP resources: http://www.cartesiangp.co.uk/resources.html

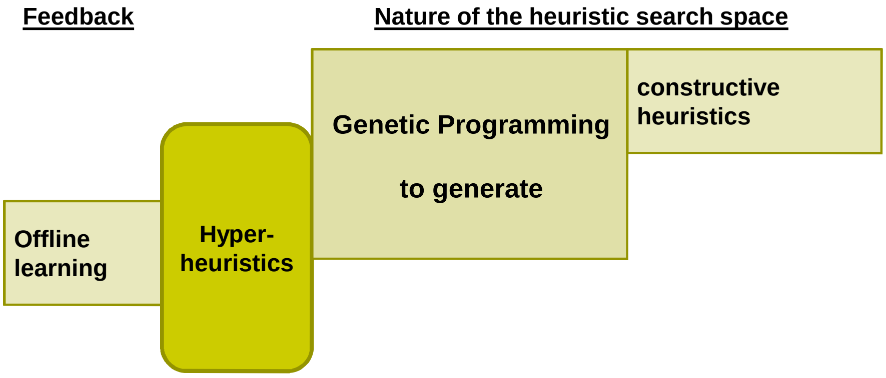
46

Case Study: Genetic Programming Hyper-heuristic for Packing

from the PhD Thesis (2010) of Matthew Hyde

The University of Nottingham
UNITED KINGDOM · CHINA · MALAYSIA

Classification of the Approach



48

1D Offline Bin Packing

Pack a **set** of items of sizes s_i for $i = 1, \ldots, n$

- Sizes are integer values and $s_i \in [1, C]$
- C is the fixed capacity of each bin

in such a way that

- Never exceed bin capacity
- Minimise number of bins used

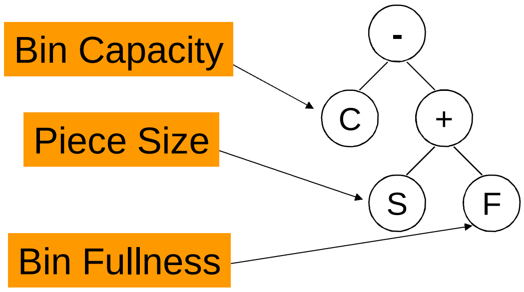
Standard NP-hard problem

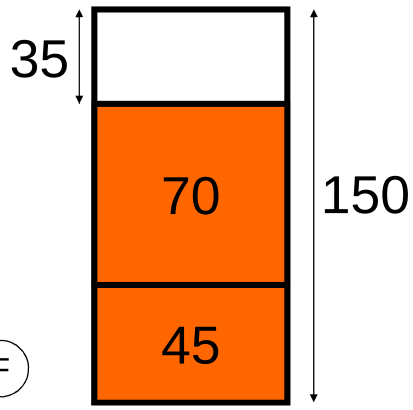
49

Genetic Programming 101

- Evolves trees representing a program
- Following tree is a program that calculates the space left at the top of the bin
- Train and test

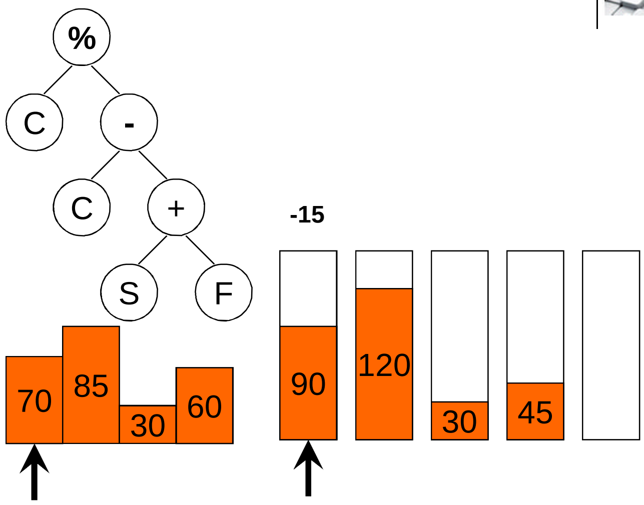
Terminals: {C, S, F}
Non terminals:{%,+,*,-}





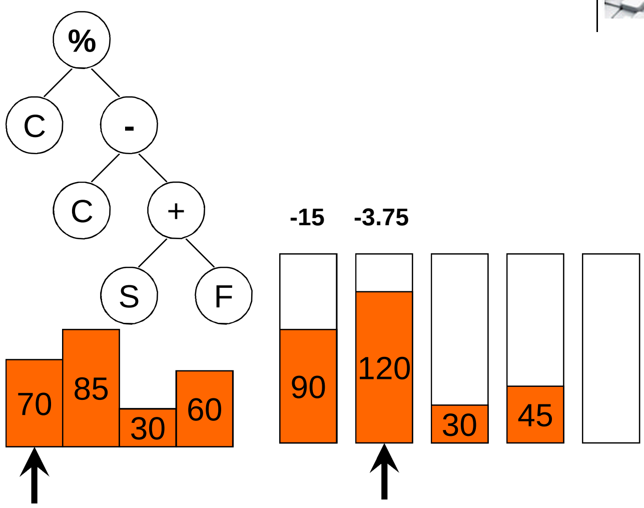
50

Genetic Programming Heuristics – Bin Packing



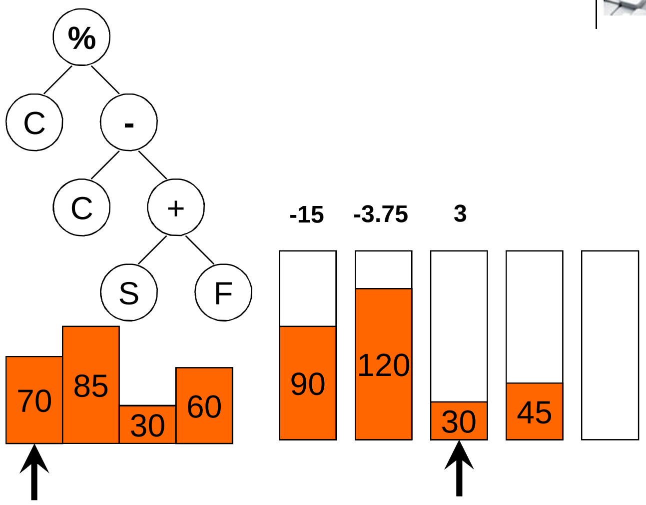
51

Genetic Programming Heuristics – Bin Packing



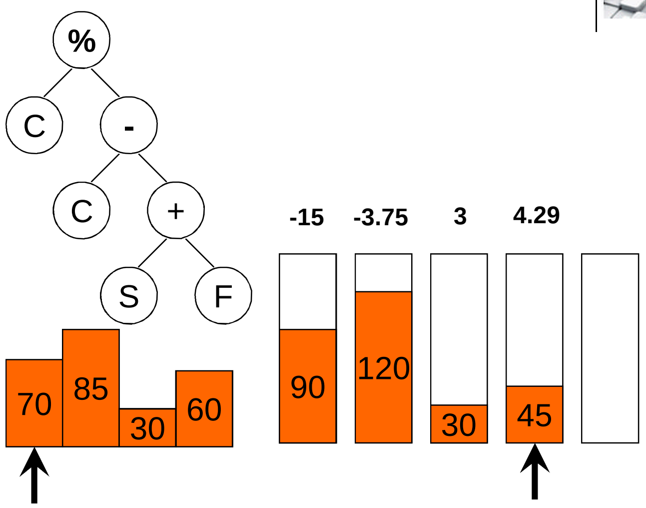
52

Genetic Programming Heuristics – Bin Packing



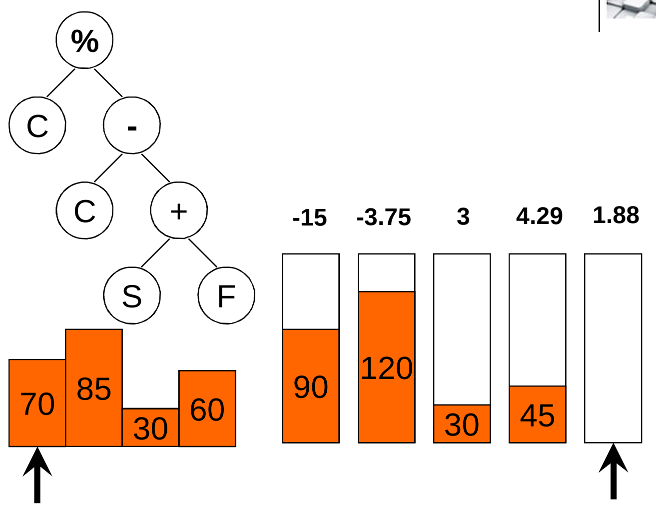
53

Genetic Programming Heuristics – Bin Packing



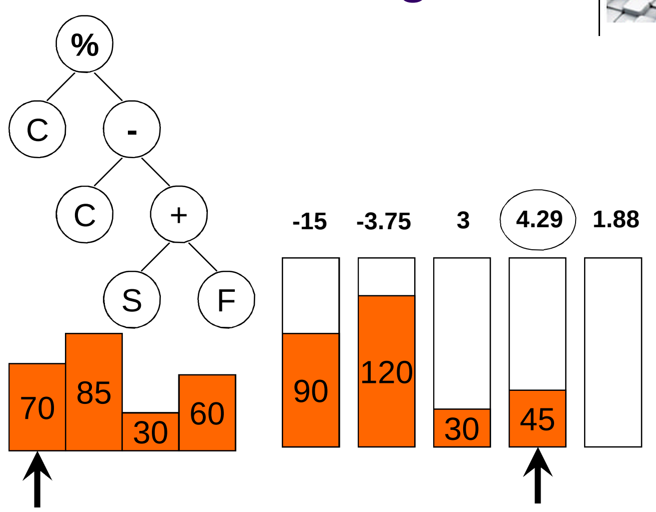
54

Genetic Programming Heuristics – Bin Packing



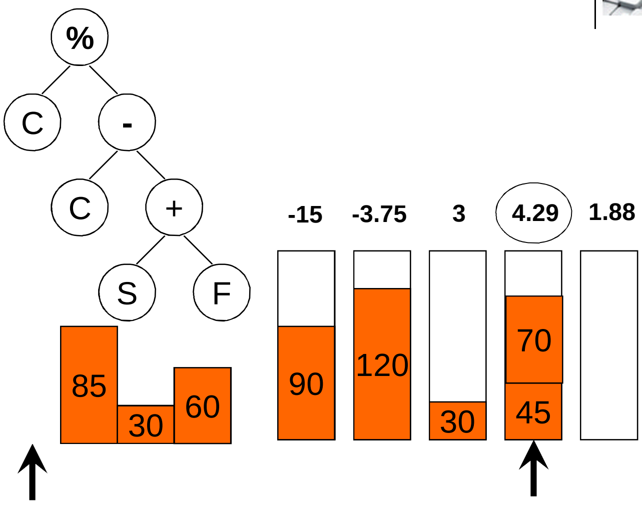
55

Genetic Programming Heuristics – Bin Packing



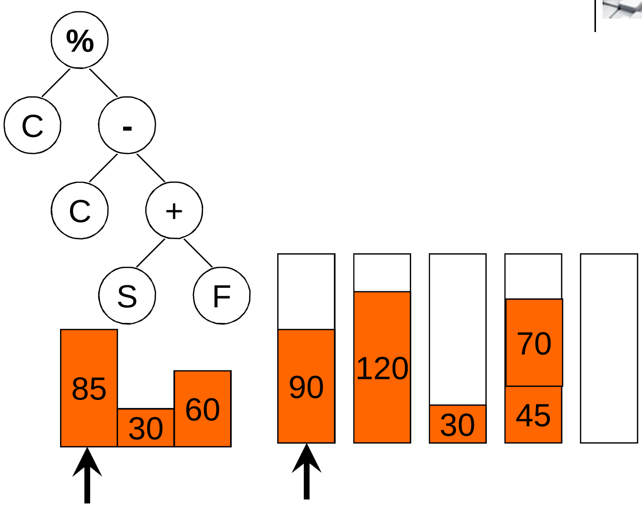
56

Genetic Programming Heuristics – Bin Packing



57

Genetic Programming Heuristics – Bin Packing



58

GP Hyper-heuristic for packing – Conclusions

- A **more general** packing methodology for 1D, 2D and 3D bin packing and knapsack problems
- **Achieved generality without the loss of solution quality**

Policy Matrix Evolution for Generation of Heuristics

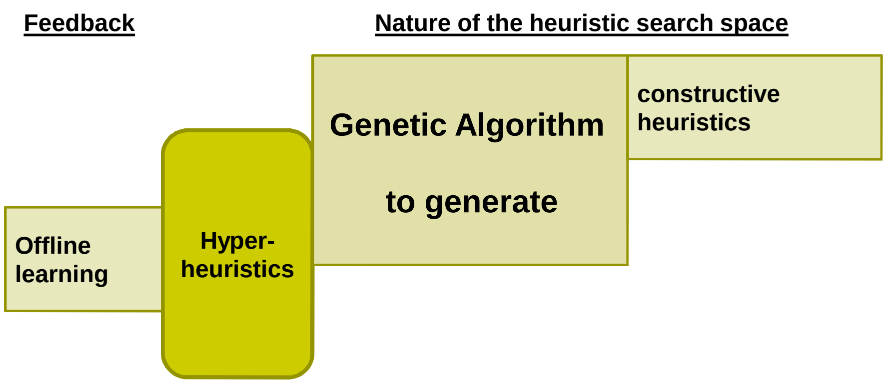
Ender Özcan
Joint work with
Andrew J. Parkes

Best paper Award:

E. Özcan, and A. J. Parkes, Policy Matrix Evolution for Generation of Heuristics, Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO '11), Natalio Krasnogor (Ed.). ACM, New York, NY, USA, pp. 2011-2018, 2011

The University of Nottingham
UNITED KINGDOM · CHINA · MALAYSIA

Classification of the Approach



61

Policy Generation

- Vast O.R. literature on finding policies for **stochastic processes**. Potential usages
  - Customer service centres
    - Call centres
    - Health services
  - Distribution centres
    - items onto trucks
    - etc
- In some cases analytical solutions are possible
- Generally, will need "numerical" methods for complex situations

62

1D Offline Bin Packing

Pack a **set** of items of sizes s_i for $i = 1, \ldots, n$

- Sizes are integer values and $s_i \in [1, C]$
- C is the fixed capacity of each bin

in such a way that

- Never exceed bin capacity
- Minimise number of bins used

Standard NP-hard problem

63

1D Online Bin Packing

Pack a **stream** of items of sizes s_i for $i = 1, \ldots$

- Sizes are integer values and $s_i \in [1, C]$
- C is the fixed capacity of each bin

upon their arrival (one item at a time)

in such a way that

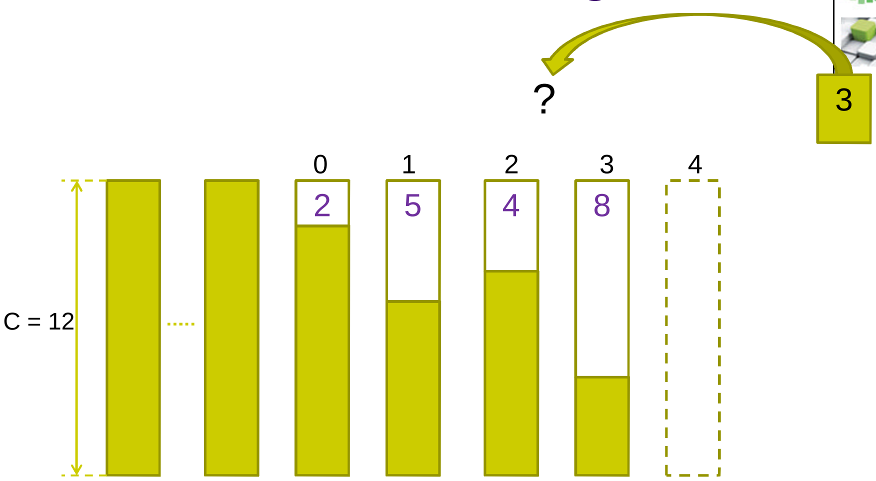
- Never exceed bin capacity
- Minimise number of bins used (Maximise the average bin-fullness) **at the end**

64

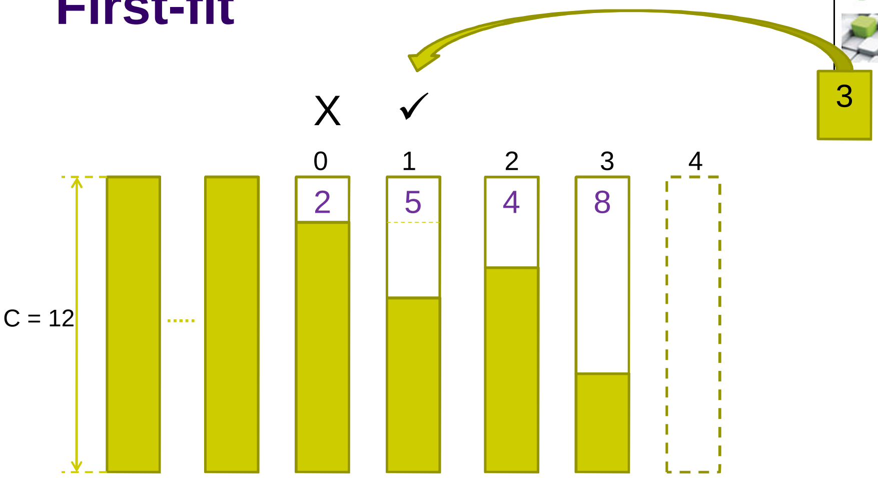
1D Online Bin Packing

- A new empty bin is always available (**open**)
- A bin is **closed** if it can take no more items
  - e.g. if **residual space** is smaller than size of any item
- We need a *good* “policy”, i.e. a method to assign a new item upon its arrival to one of the *open* bins

1D Online Bin Packing



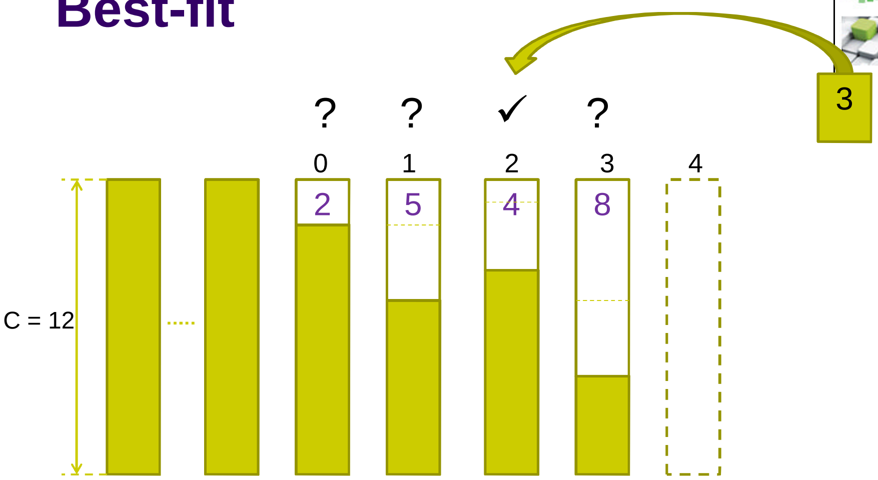
Standard Heuristic Policies: First-fit



Pack new item into the leftmost (first) bin that can take it

67

Standard Heuristic Policies: Best-fit



Pack new item into the fullest bin that can take it

68

Potential General Method for 1D Online Bin Packing

- On arrival of new item of size s_i
  - Inspect the current set of open bins
  - Simultaneously use the entire set of residual spaces in the open bins to pick where to place the new item
- This is difficult and expensive (in general)

"Index Policies"

- "index policy": each choice option is given a score, or "index value" independently of the other options
  - The option with the highest index value is taken
  - Also need a rule to break ties
- Although index policies are a special case, in some situations, they can be optimal, or at least very good
- Index policies occur in bandit problems, with use in search control
  - OR has lots of work in this area, e.g. Gittins/Whittle indices

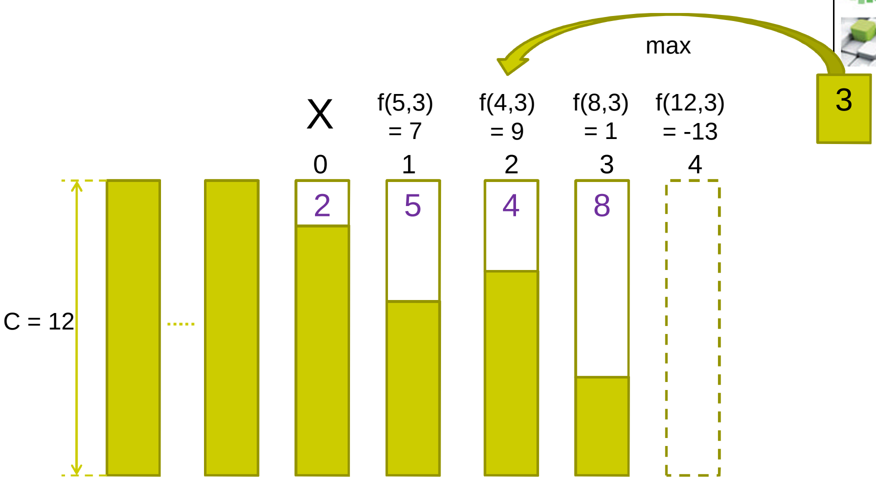
Index Policies for 1D Online Bin Packing

- Given
  - r : remaining capacity of bin (residual space)
  - s : item size
- score of bin is f(r,s) for some function f

- Given a new item of size then place into bin with largest value of f(r,s)
- We will break any ties using FF:
  - **place item in earliest bin with the best available score**

71

1D Online Bin Packing

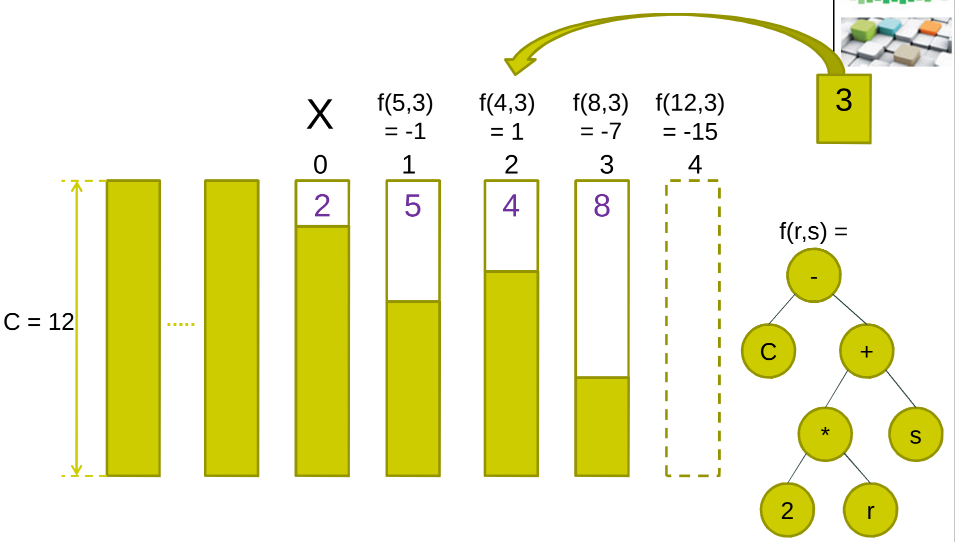


72

Generating Heuristics

- Within search methods, often have score functions, "index functions" to help make some choice
  - difficult to invent successful ones; want to automate this
- GP approach: evolve arithmetic score functions
- Burke, Hyde, Kendall, Woodward (GECCO 2007)
  (and other papers, also on other problem domains, please see http://www.cs.nott.ac.uk/~mvh/)
  - Use Genetic Programming to learn f(r,s)
  - f(r,s) is represented as arithmetic function tree
  - Automatically creates functions that at least match FF, BF

GP – 1D Online Bin Packing



Generating Heuristics

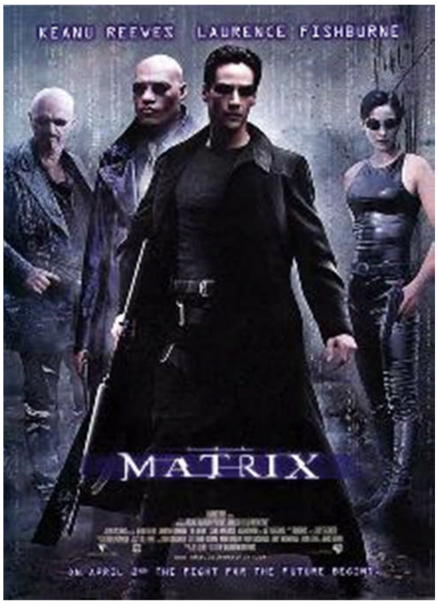
Challenge:

- Space of functions, as used in GP,
  - is hard to understand
  - potentially biased because of the choice of representation
  - some perfectly good functions might have “bloated” representations

Can one do even better?

75

Is there another way to find policies?



Source: http://en.wikipedia.org/wiki/The_Matrix

76

Matrix View of Policies/Heuristics

- Since all item sizes (s) and residual capacities (r) are integer, then f(r,s) is simply a large (*C*x*C*) matrix M(r,s) of **parameter values**

M(r,s)

r \ s	1	2	3	4	5	6
1	.	.	.	.	.	.
2	.	2	.	.	.	.
3	.	1	2	.	.	.
4	.	2	1	.	.	.
5	.	.	.	.	.	.
6	.	2	2	.	.	.

C

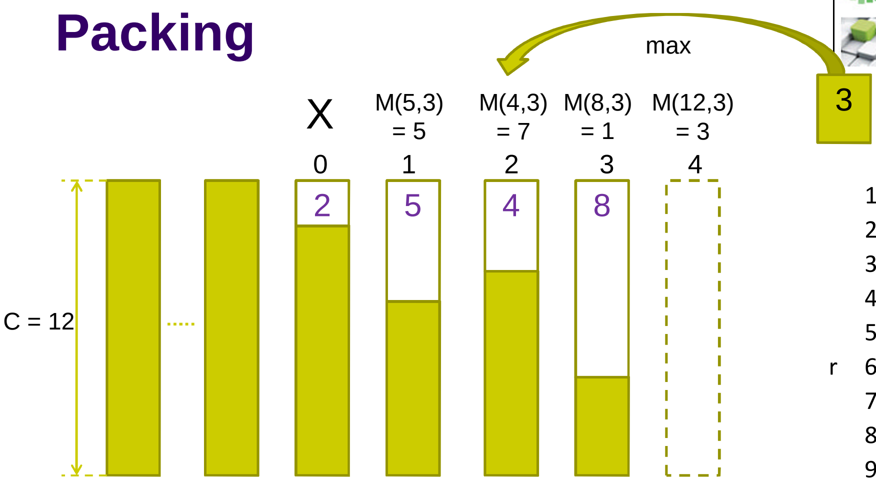
NOT USABLE
r < s

r ≥ s

C

77

Policy Matrix – 1D Online Bin Packing



s

r	2	3	4
1	*	*	*
2	*	*	*
3	*	*	*
4	*	7	*
5	*	5	*
6	*	*	*
7	*	*	*
8	*	1	*
9	*	*	*
10	*	*	*
11	*	*	*
12	*	3	*

78

Uniform (random) Instances

We empirically studied matrix policies on Uniform Bin Packing problems

UBP(C, s_{min}, s_{max}, N)

- Bin capacity C
- N items are generated with integer sizes independently taken uniformly at random from the range [s_{min}, s_{max}]
  - N is usually taken to be large: 100k

79

UBP(6,2,3)

- (Bin capacity 6, items are size 2 or 3 only.)
- The only perfect packings are
  - 2+2+2
  - 3+3
- Hence the 'obvious' policy is …
- … "never mix even and odd sizes"

80

UBP(6,2,3)

- … "never mix even and odd sizes"
- Index policy as a matrix:
  - rows: residual capacity of the bin
  - columns: item size

resid \ item	1	2	3	4	5	6
1	.	.	.	.	.	.
2	.	2	.	.	.	.
3	.	1	2	.	.	.
4	.	2	1	.	.	.
5	.	.	.	.	.	.
6	.	2	2	.	.	.

- Ties are broken using First-Fit (FF)
- Grey entries "." are never usable

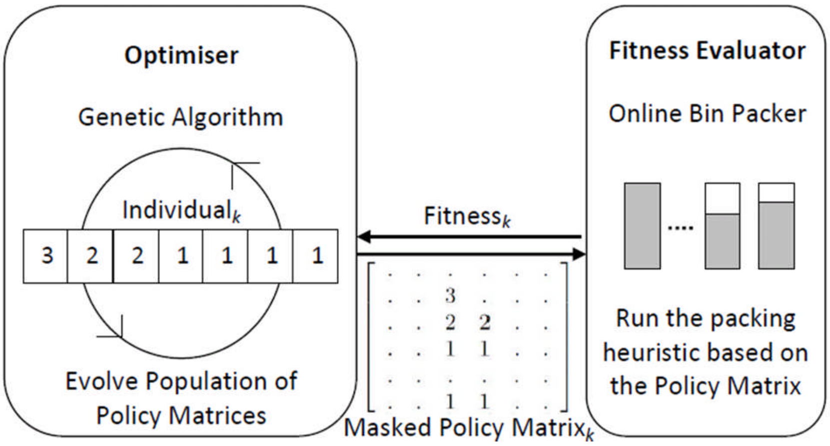
Creating Heuristics viA Many Parameters - CHAMP

- Basic idea:
  - Take values in matrix M(r,s) to be integers
  - Do (meta-)heuristic search to find good choices for M(r,s): Evaluation is by simulation
- Our Original Expectation:
  - the matrix will tweak the functions from GP and might slightly improve performance
- Potential expected disadvantages:
  - matrices can be much more verbose than functions
  - they fail to take into account of the good structure captured by functions

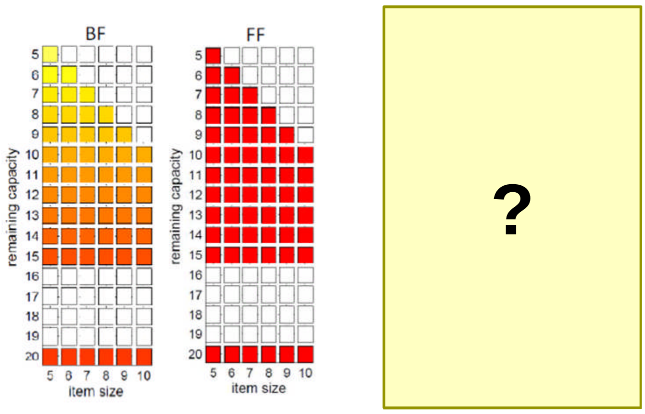
Implementation Details

- Apply a standard Genetic Algorithm
  - Trans-generational (with weak elitism), Uniform Crossover, standard mutation
- Only the active members of the matrix are stored as integer values in the chromosome
- Evaluation:
  - write matrix to a file
  - use matrix as input for a program that packs many items

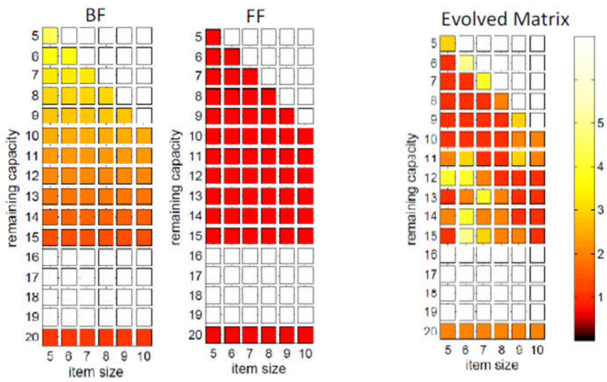
CHAMP-GA Architecture



UBP(20,5,10) – Example of a good evolved matrix



UBP(20,5,10) – Example of a good evolved matrix



- Does not look like a smooth function
  - "Weird"
  - Seems to have spikes

UBP(20,5,10)

- Empirical results

Heuristic	%-Avg. Fullness
First-Fit	91.55
Best Fit	91.54
“Best run” Evolved Matrix	**98.18**
“Worst run” Evolved Matrix	97.00

  - Even the worst run of the GA outperforms FF
  - The gap is quite large – the wasted space is reduced by a factor of ~7

87

Results – Best of runs for GA

Alg.	UBP(6.2.3)	UBP(15.5.10)	UBP(20.5.10)	UBP(30.4.20)	UBP(30.4.25)	UBP(40.10.20)	UBP(60.15.25)	UBP(75.10.50)	UBP(80.10.50)	UBP(150.20.200)
BF	92.30	99.62	91.55	96.84	98.38	90.23	92.55	96.08	96.39	95.82
FF	92.30	99.55	91.54	96.68	97.93	90.22	92.55	95.91	96.29	95.64
GA1	**99.99**	**99.63**	98.18	99.41	98.39	**96.99**	**99.68**	98.22	**98.54**	**97.88**
GA2	**99.99**	99.61	**98.42**	**99.58**	**99.55**	96.75	96.96	**98.45**	98.46	97.63

88

Conclusions

- Can use standard metaheuristics to create policies expressed in matrix representation
  - Policies exist that out-perform standard heuristics
  - Finding the policies is easier than expected
  - There are many different policies with similar performance
  - The policies are "weirder" than expected, even after smoothing.
    - The good policies could have "random" structures
    - Not necessarily easy to capture with an algebraic function of GP
  - The results can be "analysed" (inspected) to produce simple policies that out-perform standard ones
    - and that then scale to larger problems

89

Recent Work: Genetic Programming Hyper-heuristics for Scheduling

The University of Nottingham
UNITED KINGDOM · CHINA · MALAYSIA

Job Shop Scheduling

- Single objective: Rachel Hunt, Mark Johnston, Mengjie Zhang, **Evolving "less-myopic" scheduling rules for dynamic job shop scheduling with genetic programming**, Proc. of the 2014 conference on Genetic and evolutionary computation, pp. 927-934, 2014
- Multi-objective: Su Nguyen, Mengjie Zhang, Johnston, M., Kay Chen Tan, **Automatic Design of Scheduling Policies for Dynamic Multi-objective Job Shop Scheduling via Cooperative Coevolution Genetic Programming**, *Evolutionary Computation, IEEE Transactions on* , vol.18, no.2, pp.193,208, 2014

91

How does an evolved rule look like?

```
Rule #1 – Objectives(757.16, 3520.19, 0.17, 164.52, 1811.77)
(((IF(SJ,RJ,max(PR,WT)) + (max(RO,RT) + (RJ/IF(SJ,PR,rJ)))) − WINQ) + (((max(RO,RT)+
IF(SJ,IF(SJ,PR,rJ),rJ)) + (−1 × (IF(SJ,PR,rJ)) + IF(SJ,DD/PR,rJ))) − min(SJ,(WINQ × min(PR,WINQ)))))
−Abs((rJ − RT) + min(min(SJ,IF(SJ,PR,rJ)),(rJ − RT)))
Rule #2 – Objectives(828.45, 2322.88, 0.19, 165.04, 1931.40)
(−rJ − SJ + max(RO,RT) + (((((RJ/PR) + max(RO,RT)) + max(PR,max(RO,RT))) + (−PR − RT)) − 0.8968051))
−Abs(IF(min(SJ,WINQ),WINQ,DD/PR) + Abs(min(SJ,WINQ)))
Rule #3 – Objectives(720.28, 4383.52, 0.09, 105.67, 2401.59)
((max(RM,(Abs(min(WT,SJ)) × (RT × PR)))/PR)/Abs(PR + RO))/Abs(max(((PR × max(RT,max(APR,SJ))) × (PR
+WINQ)),((Abs(PR) × (RT × PR)) × DD)%Abs(min(WT,(SJ/APR)))))
Rule #4 – Objectives(716.52, 3842.36, 0.11, 82.67, 1714.88)
((max((PR × APR),(Abs(min(WT,SJ)) × WINQ))/PR)%WINQ)/Abs(max(((PR × PR) × (max(Abs(RT),max(APR,SJ))
+min(WT,(SJ/APR)))),((PR × WINQ) × DD)%Abs(min(WT,(SJ/APR)))))
Rule #5 – Objectives(708.05, 4141.63, 0.13, 109.08, 1977.63)
(((((RT/rJ) + rJ)/max(min(DD,SJ),RT)) − min(−(IF(SJ,RJ,NPR)/(SJ + WINQ)),DD)) + (−WINQ + (−RO−
min(min(SJ,WINQ),rJ)))) + ((max(SJ,rJ) + ((IF(SJ,RJ,−RO)/PR) − (rJ + max((WINQ + PR),0.371)))) − NPR)
Rule #6 – Objectives(687.85, 5708.02, 0.16, 134.13, 4046.06)
Abs((((RJ/SJ)/PR)/PR)/max(APR,WINQ)) × Abs(((((SJ/APR) − SJ)/min(RT,SJ)) × min(RT,SJ))
/min(((RJ/SJ) × (RJ/SJ)),RT))
```

S. Nguyen, M. Zhang, M. Johnston, and K-C. Tan, Dynamic Multi-objective Job Shop Scheduling: A Genetic Programming Approach, Automated Scheduling and Planning, Studies in Comp. Intelligence vol. 505, 2013, pp 251-282

92

Flexible Job Shop Scheduling

- Single objective: Beham, A; Winkler, S.; Wagner, S.; Affenzeller, M., **A genetic programming approach to solve scheduling problems with parallel simulation**, *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, pp.1-5, 2008
- Multi-objective: Joc Cing Tay, Nhu Binh Ho, **Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems**, Computers & Industrial Engineering, Vol. 54, Issue 3, 2008, pp. 453-473

93

Single Machine Scheduling

- Single objective: C. Dimopoulos, A.M.S. Zalzala, **Investigating the use of genetic programming for a classic one-machine scheduling problem**, Advances in Engineering Software, Volume 32, Issue 6, June 2001, Pages 489-498
- Multi-objective: S. Nguyen, M.Zhang, M. Johnston, K. C. Tan, **Learning Reusable Initial Solutions for Multi-Objective Order Acceptance and Scheduling Problems with Genetic Programming**, Proc. of the 16th European Conference on Genetic Programming, EuroGP 2013, pp 157-168, LNCS 7881, 2013

94

Others

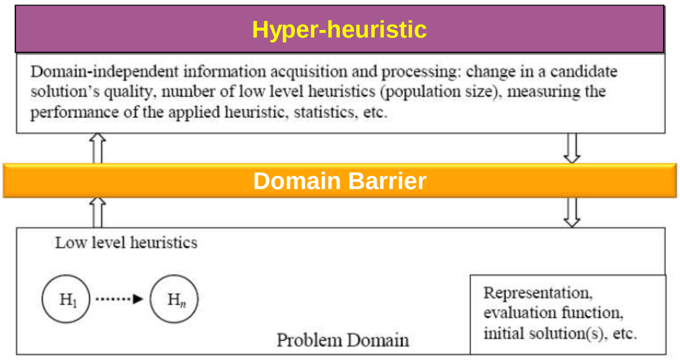
- Parallel Machine: Domagoj Jakobović, Leonardo Jelenković, Leo Budin, **Genetic Programming Heuristics for Multiple Machine Scheduling**, LNCS 4445, 2007, pp 321-330
- Flow shop scheduling: Franco Mascia, Manuel López-Ibáñez, Jérémie Dubois-Lacoste, Thomas Stützle, **From Grammars to Parameters: Automatic Iterated Greedy Design for the Permutation Flow-Shop Problem with Weighted Tardiness**, LNCS 7997, pp 321-334, 2013

95

Selection Hyper-heuristics

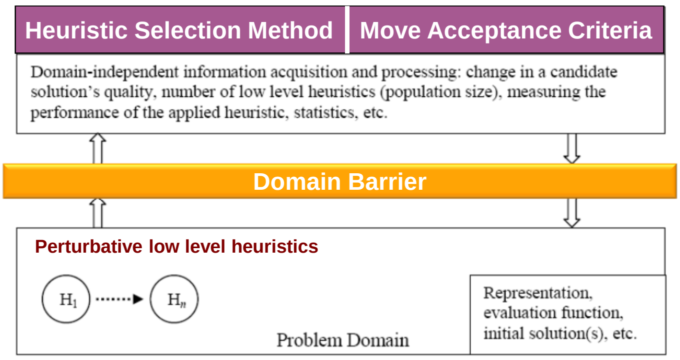
UNITED KINGDOM · CHINA · MALAYSIA

A Hyper-heuristic Framework – revisited



97

A Selection Hyper-heuristic Framework – Single Point Search



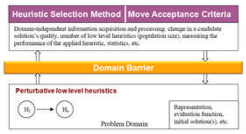
98

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 {H1, ..., Hn}
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;
```

99

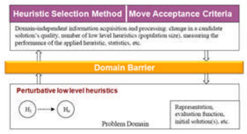
Heuristic Selection



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)
Reinforcement Learning	Nareyek (2003); Pisinger and Ropke (2007); Bai et al (2007a)
Reinforcement Learning with Tabu Search	Burke et al (2003b); Dowsland et al (2007)
Learning Automata	Mısır et al. (2009)
Quality Index and Tabu based Learning Heuristic Selection	Mısır et al. (2009)

100

Move Acceptance

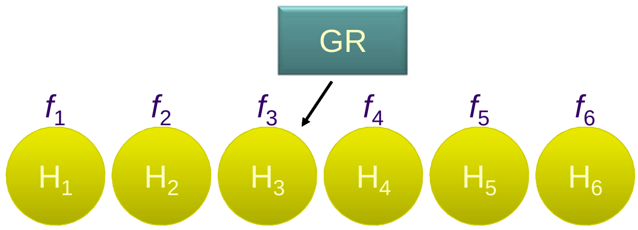


Component name	Reference(s)
Deterministic move acceptance	
All Moves	Cowling et al (2000, 2002b)
Only Improvements	Cowling et al (2000, 2002b)
Improving and Equal	Cowling et al (2000, 2002b)
Non-deterministic move acceptance	
Monte Carlo	Ayob and Kendall (2003)
Great Deluge	Kendall and Mohamad (2004a); Bilgin et al (2006)
Record to Record Travel	Kendall and Mohamad (2004b)
Tabu Search	Chakhlevitch and Cowling (2005)
Simulated Annealing	Bai and Kendall (2005); Bilgin et al (2006); Pisinger and Ropke (2007); Antunes et al (2009)
Simulated Annealing with Reheating	Dowsland et al (2007); Bai et al (2007a)
Late Acceptance	Özcan et al (2009)
Iteration Limited Threshold Accepting (ILTA)	Mısır et al. (2009)
Adaptive ILTA	Mısır et al. (2009)

101

Heuristic Selection – Greedy (GR)

- Apply each low level heuristic to the candidate solution and choose the one that generates the best objective value



$f_3 < f_1, f_2, f_4, f_5, f_6$ at step t

102

Heuristic Selection – Reinforcement Learning (RL)

- A machine learning technique
- Inspired by related psychological theory
  - Reward and punishment
- Concerned with how an agent ought to take actions in an environment to maximize some notion of long-term reward
- Maintains a score for each heuristic
  - If an improving move then increase, otherwise decrease the score of the heuristic

103

Heuristic Selection – Choice Function (CF)

- The choice function maintains a record of the performance of each heuristic. Three criteria are maintained:

  1) Its individual performance

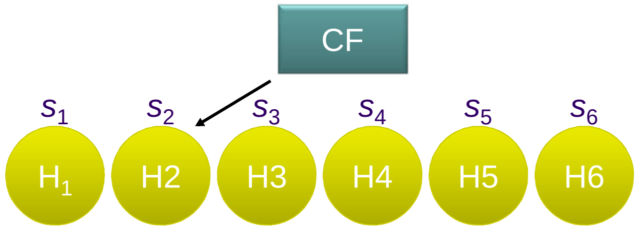
  2) how well it has performed with other heuristics

  3) the elapsed time since the heuristic has been called

$$F_t(h_j) = \alpha_t\, f_1(h_j) + \beta_t\, f_2(h_k, h_j) + \gamma_t\, f_3(h_j)$$

104

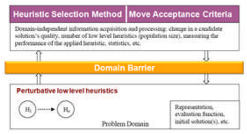
Heuristic Selection – Choice Function (CF)



$s_2 > s_1, s_3, s_4, s_5, s_6$ at step t

105

Move Acceptance



Component name	Reference(s)
Deterministic move acceptance	
All Moves	Cowling et al (2000, 2002b)
Only Improvements	Cowling et al (2000, 2002b)
Improving and Equal	Cowling et al (2000, 2002b)
Non-deterministic move acceptance	
Monte Carlo	Ayob and Kendall (2003)
Great Deluge	Kendall and Mohamad (2004a); Bilgin et al (2006)
Record to Record Travel	Kendall and Mohamad (2004b)
Tabu Search	Chakhlevitch and Cowling (2005)
Simulated Annealing	Bai and Kendall (2005); Bilgin et al (2006); Pisinger and Ropke (2007); Antunes et al (2009)
Simulated Annealing with Reheating	Dowsland et al (2007); Bai et al (2007a)
Late Acceptance	Özcan et al (2009)
Iteration Limited Threshold Accepting (ILTA)	Mısır et al. (2009)
Adaptive ILTA	Mısır et al. (2009)

106

Move Acceptance – Simple Criteria

- **AM**: All Moves Accepted
- **OI**: Only Improving Moves accepted
- **IE**: Improving or Equal moves are accepted.

Move Acceptance – Great Deluge (GD)

- Improving and equal moves are accepted
- Non-improving moves resulting in a fitness value less than a threshold are accepted.
- The threshold is decreased to global minimum with time.
  - N : initial fitness – minimum fitness
  - t : time passed
  - D : Duration

$$\underbrace{f_t}_{\text{current fitness}} < \underbrace{f_{\min} + N \times \left(1 - \frac{t}{D}\right)}_{\text{threshold}}$$

Move Acceptance – Simulated Annealing

- All improving moves are accepted while the non-improving are accepted based on Metropolis criterion ($e^{-\delta/\tau}$), where τ represents temperature, being decreased at each iteration using a *cooling schedule*, and δ is the change in the solution quality.
- Previous studies show that simulated annealing is one of the best move acceptance criterion

109

Some Tools for Heuristic Selection

- SATzilla: algorithm portfolio oriented data-driven framework
- Simple Neighborhood-based Algorithm Portfolio in PYthon (snappy)
- Hyper-heuristics Flexible Interface (HyFlex)
- ParHyFlex extends MPI
- Hyperion provides a white-box framework giving a metaheuristic/hyper-heuristic full access to the problem domain

110

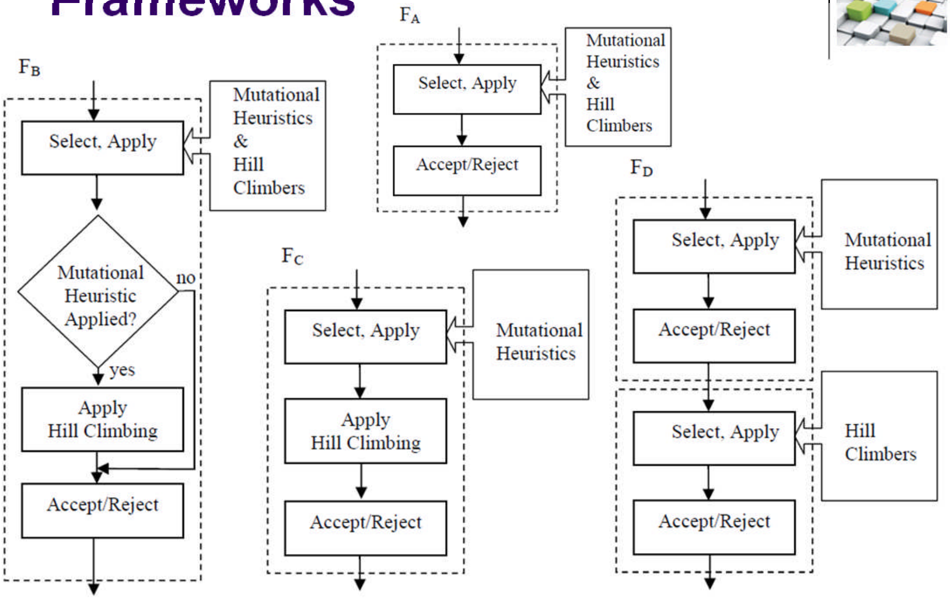
A Comprehensive Analysis of Hyper-heuristics

Ender Özcan, Burak Bilgin, Emin Erkan Korkmaz

Intelligent Data Analysis, 12:1, pp. 3-23, 2008

The University of Nottingham
UNITED KINGDOM · CHINA · MALAYSIA

Selection Hyper-heuristic Frameworks



Results

Label	F_A	F_B	F_C	F_D
F1	1.00	1.00	1.00	1.00
F2	0.00	0.00	0.00	0.00
F3	1.00	1.00	1.00	0.00
F4	0.00	0.02	0.02	0.02
F5	0.76	1.00	1.00	0.54
F6	0.08	1.00	1.00	0.00
F7	0.92	0.98	1.00	0.00
F8	0.00	0.30	0.90	0.90
F9	1.00	1.00	1.00	0.96
F10	0.02	0.44	0.54	0.02
F11	0.00	1.00	1.00	0.06
F12	1.00	1.00	1.00	0.00
F13	0.00	1.00	1.00	0.00
F14	0.82	1.00	1.00	0.06
Avr.	0.47	0.77	0.82	0.25

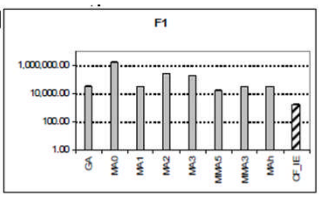
Succes rate = (# of runs achieving expected objective value)/(total # of runs)

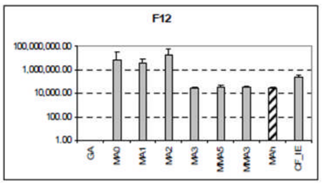
- Binary representation
- 3 mutational, 3 hill climbing heuristics
- F_B and F_C employ DBHC.
- F_D uses CF-AM (mutational) and CF-IE (hill climbing)

113

Results

- GD, MC and IE performs well with CF and SR
- CF-IE (under F_C) delivers a “similar” performance to multimeme algorithm





- Choice of low level heuristics influences the overall performance of a hyper-heuristic

114

[Hyper-heuristics Flexible Interface] HyFlex

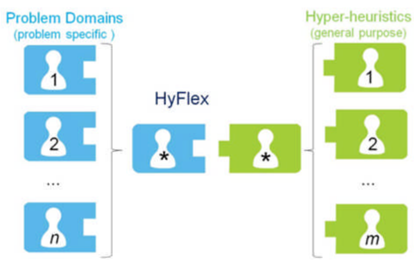
The University of Nottingham

UNITED KINGDOM · CHINA · MALAYSIA

HyFlex

Hyper-heuristics Flexible Interface

- Defines behaviours of components and arranges the interaction between them



Separation between the problem-specific and the general-purpose parts, both of which are reusable and interchangeable through the HyFlex interface

http://www.hyflex.org/

116

HyFlex v1.0 Java Implementation

- Currently there are 6 problem domain implementations
- heuristic types: mutational, ruin-recreate, local search, crossover
- parameters: intensity, depth of search

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	$\mathbf{HC}_2$						

http://www.hyflex.org/

117

CHeSC 2011 CROSS-DOMAIN HEURISTIC SEARCH CHALLENGE

http://www.hyflex.org/

asap research: automated scheduling optimisation & planning

CHeSC 2011 benchmark based on HyFlex v1.0

The University of Nottingham

MAX-SAT
Bin Packing
Flow Shop
Personnel Scheduling
TSP
VRP

Hidden

- 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

Sponsor:

École Polytechnique Montréal

EVENTMAP

Cardiff University Prifysgol Caerdydd

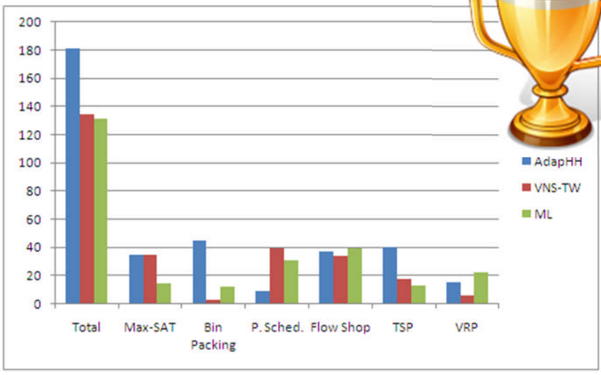
CHeSC 2011 INTERNATIONAL CROSS-DOMAIN HEURISTIC SEARCH CHALLENGE

CHeSC 2011
CROSS-DOMAIN HEURISTIC SEARCH CHALLENGE

http://www.hyflex.org/

ASAP research group

And the winner is...



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

CHeSC 2011 INTERNATIONAL CROSS-DOMAIN HEURISTIC SEARCH CHALLENGE

AdapHH – Overview

ASAP research group

$$p_i = w_1\left[\left(C_{p,best}(i)+1\right)^2\left(t_{remain}/t_{p,spent}(i)\right)\right] \times b + w_2\left(f_{p,imp}(i)/t_{p,spent}(i)\right) - w_3\left(f_{p,wrs}(i)/t_{p,spent}(i)\right) + w_4\left(f_{imp}(i)/t_{spent}(i)\right) - w_5\left(f_{wrs}(i)/t_{spent}(i)\right)$$

$$b = \begin{cases} 1, & \sum_{i=0}^{n} C_{p,best}(i) > 0 \\ 0, & otw. \end{cases}$$

$$avg = \left\lfloor \left(\sum_i^n QI_i\right)/n \right\rfloor$$

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

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

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

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

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

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

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

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

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
```

$$l = l_{base} + (l_{initial} - l_{base} + 1)tf^3$$

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
```

120

AdapHH – Heuristic Selection

- A multi-phase approach adaptively deciding on the subset of low level heuristics to use at a phase and its duration
- Computes quality index for each heuristic based on a weighted average of performance measure based on (i) the number of new best solutions found, the total number of (ii) improvement and (iii) worsening until a given time and (iv, v) during a single phase, (vi) overall remaining time, the time spent by a heuristic (vii) until that moment and (viii) during a phase

  and excludes the one below the average at a stage
- Excludes relatively slow heuristics
- Uses a probability vector to select a heuristic based on (i), (vi), overall time and time spent

121

AdapHH – Heuristic Selection

- A multi-phase approach adaptively deciding on the subset of low level heuristics to use at a phase and its duration
- Computes quality index for each heuristic based on a weighted average of performance measure based on (i) the number of new best solutions found, the total number of (ii) improvement and (iii) worsening until a given time and (iv, v) during a single phase, (vi) overall remaining time, the time spent by a heuristic (vii) until that moment and (viii) during a phase

  and excludes the one below the average at a stage
- Excludes relatively slow heuristics
- Uses a probability vector to select a heuristic based on (i), (vi), overall time and time spent

122

AdapHH – Heuristic Selection

- Relay hybridisation: Keeps track of performance of successive application of heuristic pairs and applies a pair of heuristics with a certain probability. The first heuristic is chosen using a learning automaton which maintains the probability of selecting a given heuristic.
- Heuristic Parameter Adaptation: A reinforcement learning based mechanism is used

AdapHH – Move acceptance AILLA

- Maintains a list of fitness values for the recently visited new best solutions
- A worsening solution is accepted:
  - If a new best solution cannot be found after a <u>certain number of iterations</u> with consecutive worsening solutions (<u>adapted during search</u>)
  - If its fitness is better than the fitness of the top solution in the list which acts like a threshold level

CHeSC Results

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

125

Limitations of CHeSC/Hyflex

- Deficiencies of standard CHeSC/Hyflex:
  - Pure Blackbox Interface: Hyflex
    - Only allows access to the objective value of current state
    - Many suggestions for extensions to permit more information to be passed
  - Fixed 10mins
  - Independent instances
    - The HH is restarted for each instance and so cannot pass on anything it has learned

126

Limitations of CHeSC/Hyflex

- Fixes to deficiencies of standard CHeSC/Hyflex:
  - Blackbox "hyflex" interface
    - Many people have suggested extensions to permit more information to be passed
  - Fixed 10mins
    - Easy to change
  - Independent instances
    - **Batched mode**

127

Batched Mode (CHeSC 2014)

- Simple extension to Hyflex/CHeSC
- "Batched mode":

  HH is given N instances and a total time T
- Advantages:
  - **Load Balancing**:
    - Allocate more time to harder instances, by stopping earlier on "easy" ones
  - **Inter-instance learning**:
    - Allowed to keep information learned from previous instances

128

Potential Future Directions

- Better annotations
- Instance features
- Solution features
- Distance metrics
- Multi-objective support
- Extensions to support generative hyper-heuristics

and more…

These are currently being explored and Hyflex being extended to match them.

129

Case Study: A Tensor-based Selection Hyper-heuristic for Cross Domain Search

Shahriar Asta, Ender Özcan

Information Sciences, *to appear*

School of Computer Science

The University of Nottingham

UNITED KINGDOM · CHINA · MALAYSIA

Classification of the Approach

Feedback

Nature of the heuristic search space

Online learning

Hyper-heuristics

Heuristic selection

Methodologies to select

perturbative heuristics

131

Two Simple Hyper-heuristics Mixing Heuristics (Stochastic Local Search)

- Simple Random Heuristic Selection – Improving and Equal Move Acceptance (IE)
  - Reject any worsening move
- Simple Random Heuristic Selection – Naïve Move Acceptance (NA)
  - Accept a worsening move with a fixed probability of p (0.5 in this study)

132

Proposed Approach – Ideas

- The balance between diversification and intensification is crucial (e.g. ILS)

Intensify	Diversify	Intensify	Diversify	....	Intensify

time

t_s $2t_s$ $3t_s$

? **IE** **?** **NA**

***h*: set of low level heuristics (MU+RC+LS)**

$h_{IE} \cup h_{NA} = h \quad (h_{IE} \cap h_{NA} = \varnothing)$

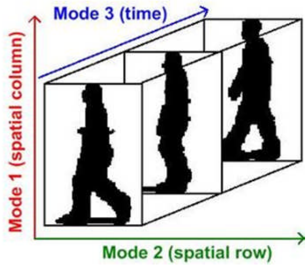
- Mix move acceptance methods
- Use machine learning to partition the low level heuristics associated with each method

133

Tensors

- Many real-world data are multidimensional
  - Very high-dimensional (big) with a large amount of redundancy
- Multi-dimensional arrays representing such data describe a tensor

Many applications in signal processing, psychometrics, and more



SOURCE:http://en.wikipedia.org/wiki/File:Video_represented_as_a_third-order_tensor.jpg

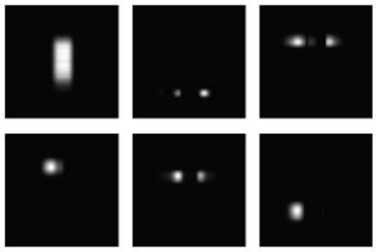
134

Tensor Factorisation

- There are different decomposition methods, we use Canonical Polyadic (CP) factorisation

$$\hat{\mathcal{T}} = \sum_{k=1}^{K} \lambda_k \mathbf{a}_k \circ \mathbf{b}_k \circ \mathbf{c}_k$$

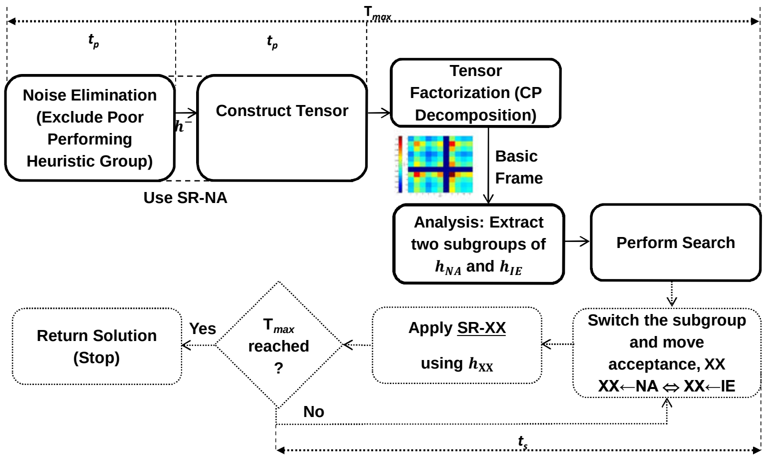
  - This gives a projection of 3D data onto 1D vectors
  - Helps to discover latent structures in data, quantifying the relationship between pairs of different components



SOURCE: B. Krausz, C. Bauckhage, Action recognition in videos using nonnegative tensor factorization., in: ICPR, IEEE, 2010, pp. 1763–1766.

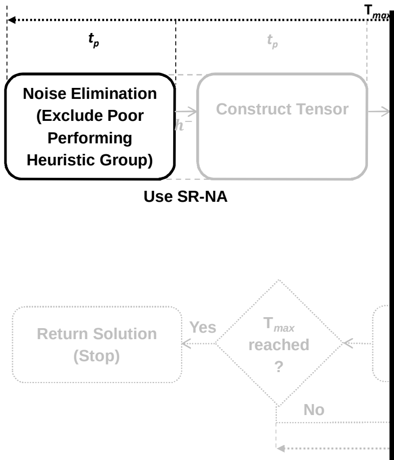
135

Proposed Approach – TeBHA-HH



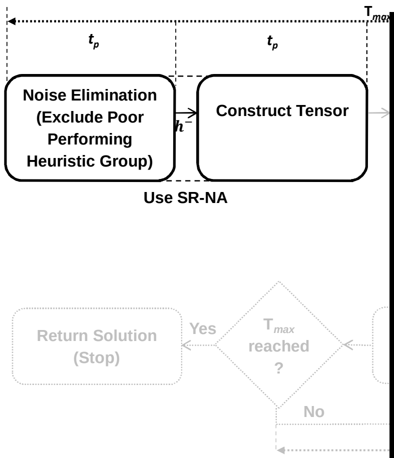
136

TeBHA-HH:
1. Noise Elimination Phase

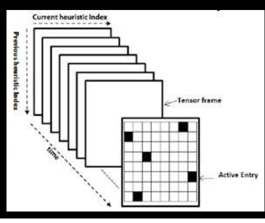


[redacted]

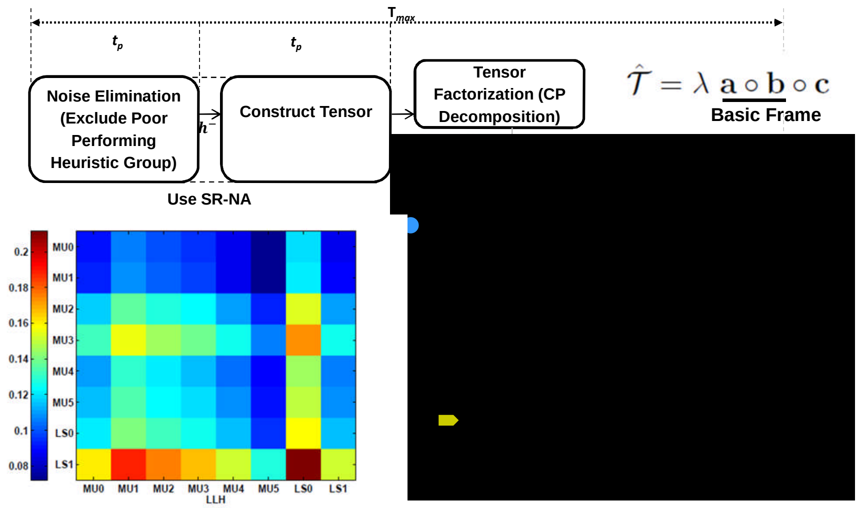
TeBHA-HH:
2. Tensor Construction Phase



[redacted]

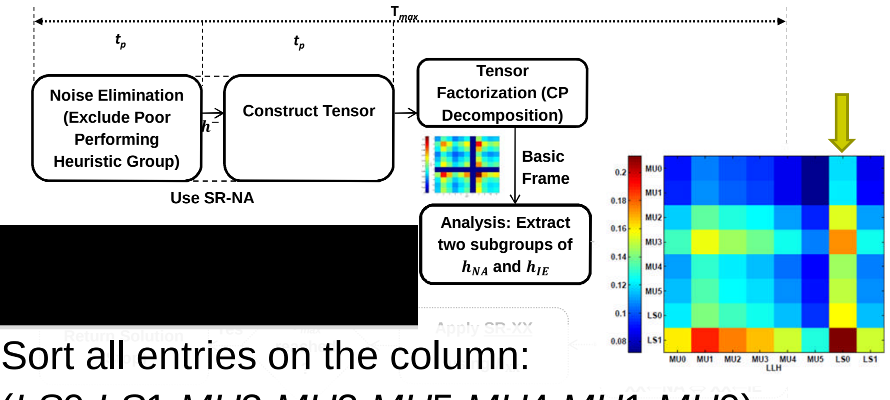


TeBHA-HH:
3. Tensor Factorization



[redacted]

TeBHA-HH:
4. Tensor Analysis

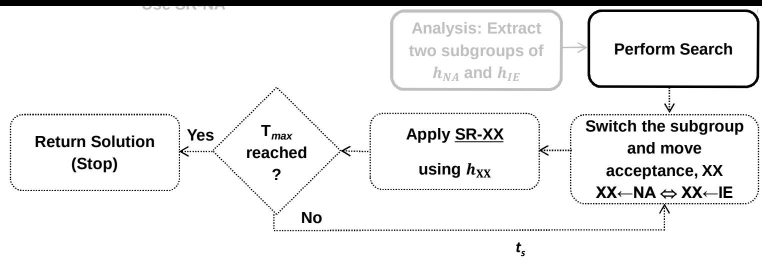


[redacted]

- Sort all entries on the column: $(LS0, LS1, MU3, MU2, MU5, MU4, MU1, MU0)$
- Top half goes to h_{NA}, the rest to h_{IE}

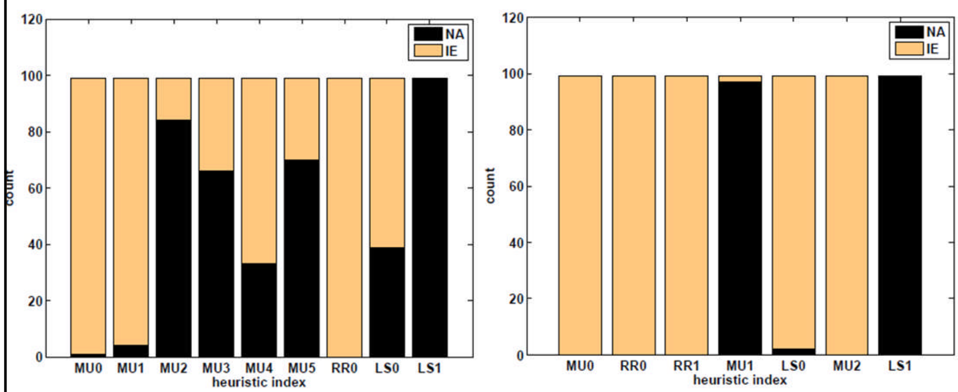
TeBHA-HH:
5. Final Phase: Perform Search

[redacted]



h_{IE} vs. h_{NA}

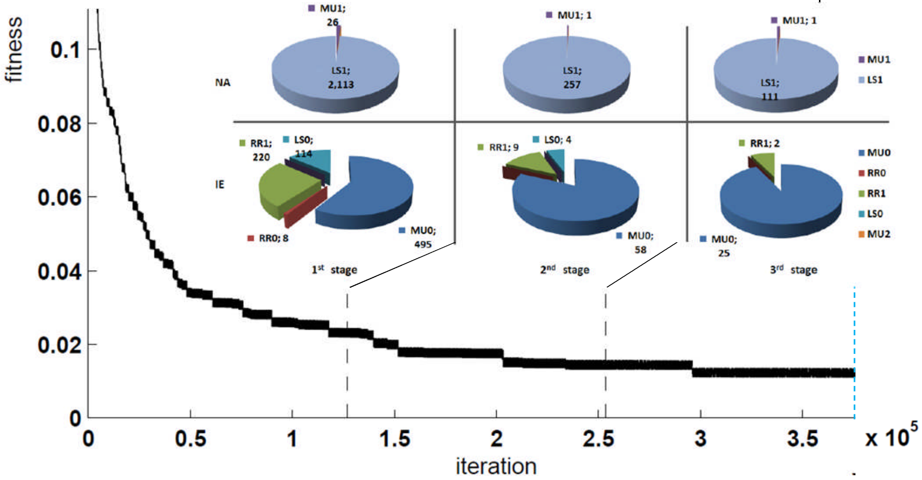
- Histograms



SAT BP

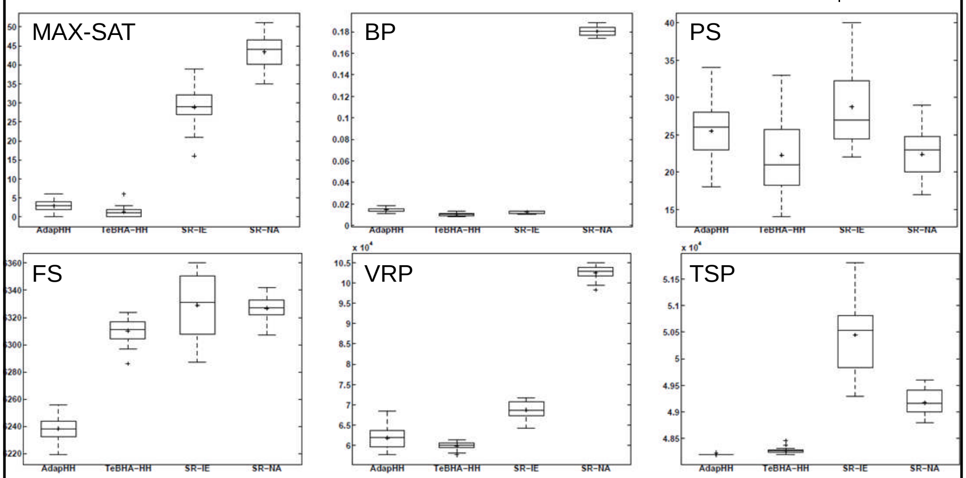
142

Interaction Between Hyper-heuristic Components

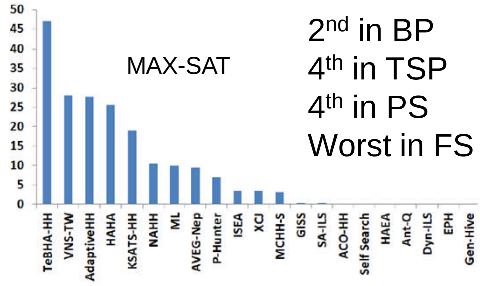


BP

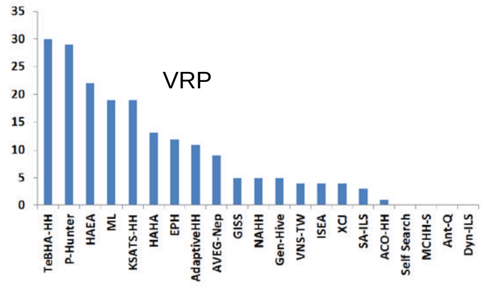
Performance Comparison of Hyper-heuristics



Results–CHeSC2011



2nd in BP
4th in TSP
4th in PS
Worst in FS



Rank	Name	Score
1	AdaptiveHH	162.83
2	**TeBHA-HH**	**148.85**
3	VNS-TW	118.83
4	ML	117.50
5	P-Hunter	84.75
6	EPH	83.25
7	NAHH	68.50
8	HAHA	65.58
9	ISEA	62.50
10	KSATS-HH	52

145

Case Study: A Data Mining Embedded Hyper-heuristic

Sahriar Asta, Ender Ozcan

ASAP research group

ASAP research group

The University of Nottingham

UNITED KINGDOM · CHINA · MALAYSIA

Classification of the Approach

Feedback

Nature of the heuristic search space

Offline learning

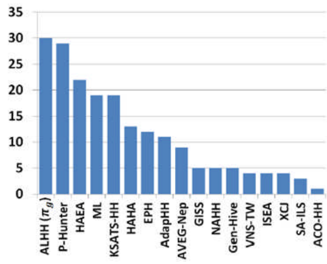
Hyper-heuristic

to generate

a hyper-heuristic

An Apprenticeship Learning Hyper-Heuristic for Vehicle Routing in HyFlex (SSCI 2014, to appear)

- **Basic idea**: Learn from an expert (AdapHH – winner of CHeSC 2011) how to make decisions on heuristic selection, parameter setting and move acceptance for building a selection hyper-heuristic
- C4.5 to construct decision trees



		Solomon Instances					Homberger Instances				
		0	1	2	3	4	5	6	7	8	9
AdapHH (π_e)	μ	5093.4	**20656.9**	13388.0	**5321.9**	14293.0	**146791.0**	62809.4	161638.5	153164.3	**147360.5**
	min	4230.2	*20651.6*	13296.9	*5275.5*	14270.7	144040.9	58521.6	*160074.4*	*146584.7*	*145139.3*
	median	5125.7	20655.4	13349.9	5320.8	14291.1	146906.7	61985.8	161596.3	153083.7	147550.3
	σ	161.7	4.2	183.8	24.5	13.9	1369.0	4609.4	982.0	1841.3	1047.6
Apprentice (π_g)	μ	**4954.6**	20792.8	**13266.7**	5365.2	**14113.8**	147017.6	**60101.9**	**161491.5**	**153132.2**	147414.9
	min	*4178.8*	20653.3	*12300.2*	5305.2	*13277.0*	*144037.7*	*58352.6*	160084.5	149227.1	145478.3
	median	5156.4	20661.2	13365.5	5366.7	14294.0	146988.0	60163.0	161529.8	153000.2	147480.9
	σ	394.2	340.9	310.9	29.4	481.3	1780.5	790.0	842.7	1663.1	956.8
P-Hunter [30]	min	-	20650.8	12263.0	-	-	143663.9	61139.3	-	-	146472.9
	median	-	20650.8	12290.0	-	-	146944.4	64717.8	-	-	148659.0
AdOr-ILS [31]	μ	5281.7	21291.9	13605.0	6564.4	14280.8	155305.5	77302.7	163177.7	158941.9	149447.7
	σ	334.614	482.56	451.64	554.77	319.54	6154.24	3384.83	2100.09	2460.71	1500.9

Case Study: A Multi-stage Selection Hyper-heuristic

Ahmed Kheiri, Ender Ozcan

EJOR, in review

The University of Nottingham
UNITED KINGDOM · CHINA · MALAYSIA

Classification of the Approach

Feedback

Nature of the heuristic search space

Online learning

Hyper-heuristics

Heuristic selection

Methodologies to select

perturbative heuristics

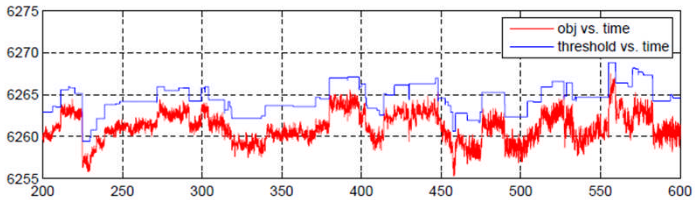
150

Stage 1 Hyper-heuristic

- 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



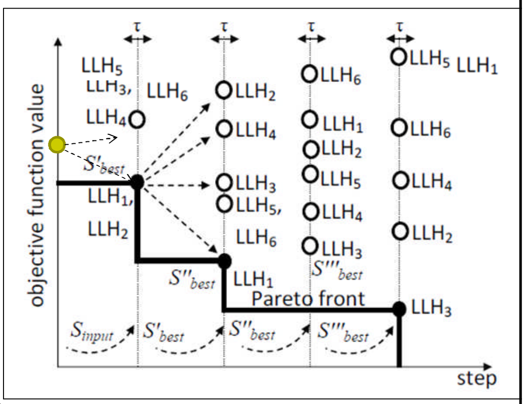
151

Stage 2 Hyper-heuristic

- Uses relay hybridisation

  Given LLH_1 and LLH_2:
  $LLH_3=LLH_1+LLH_1, \ldots,$
  $LLH_6=LLH_2+LLH_1$

- Reduces the set of low level heuristics
- Adjusts heuristic scores according to a Greedy and dominance based approach



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

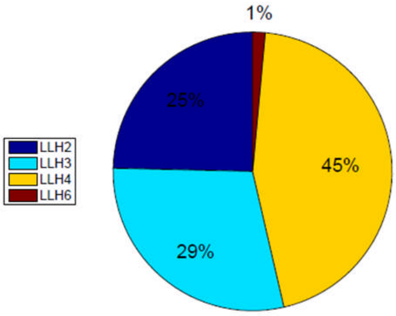
2+4 LLHs → **3 LLHs**

MSHH

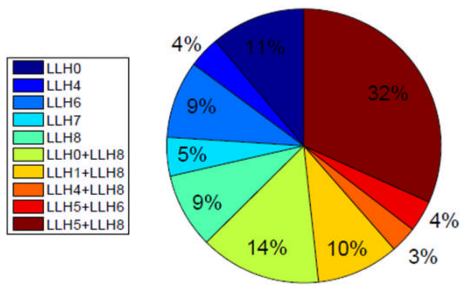
Domain	Instance	MSHH avg.	MSHH std.	MSHH median	MSHH min.	vs.	S1HH avg.	S1HH std.	S1HH min.	vs.	S2HH avg.	S2HH std.	S2HH min.
SAT	Inst1	**0.9**	0.7	1.0	**0.0**	>	6.4	4.5	1.0	>	15.0	4.6	3.0
	Inst2	**3.1**	3.9	2.0	**1.0**	>	21.3	13.3	3.0	>	44.9	9.8	18.0
	Inst3	**0.7**	0.5	1.0	**0.0**	>	7.1	7.7	**0.0**	>	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**	≥	0.0054	0.0012	0.0027	>	0.0128	0.0011	0.0104
	Inst4	0.1084	0.0000	0.1084	**0.1083**	≤	**0.1084**	0.0000	0.1083	>	0.1084	0.0000	0.1084
	Inst5	**0.0050**	0.0019	0.0044	**0.0032**	≥	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**	≤	**9645.3**	159.6	9334.0	≤	9645.8	106.7	9391.0
	Inst3	**3283.7**	93.3	3270.0	**3132.0**	≥	3304.8	99.6	3134.0	≥	3309.9	110.2	3172.0
	Inst4	**1786.3**	172.1	1760.0	1545.0	≥	1801.0	142.3	1570.0	≥	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	≤	**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**	≤	**70223.0**	2960.2	64273.2	>	84103.9	7225.8	68958.3
	Inst2	**13421.8**	251.6	13359.6	**13303.9**	≥	13658.0	471.4	13319.6	>	13695.8	473.9	13320.0
	Inst3	148498.2	1625.8	148436.2	145466.5	≤	**148232.6**	1935.3	145426.5	≥	149553.2	2377.8	**145362.7**
	Inst4	21016.4	488.2	20671.4	**20650.8**	≤	**20991.3**	478.0	20653.5	>	21131.9	510.3	20657.5
	Inst5	**148813.7**	1272.5	149193.7	**146334.6**	≥	148999.1	1217.1	146844.9	>	150282.6	1616.3	146666.9

Relay Hybridisation

PS



TSP



Result

Label	SAT	BP	PS	PFS	TSP	VRP	Overall
MSHH	48.00	38.00	6.00	25.00	42.60	4.00	**163.60**
AdapHH	27.58	44.00	8.00	33.00	34.60	14.00	161.18
VNS-TW	27.08	2.00	39.50	30.00	13.60	6.00	118.18
ML	10.00	8.00	31.00	36.50	10.00	22.00	117.50
PHUNTER	7.00	2.00	11.50	6.00	21.60	33.00	81.10
EPH	0.00	6.00	10.50	18.00	30.60	12.00	77.10
HAHA	25.58	0.00	24.50	2.83	0.00	14.00	66.92
NAHH	10.50	16.00	2.00	19.50	9.00	6.00	63.00
ISEA	3.50	25.00	14.50	3.50	7.00	4.00	57.50
KSATS-HH	19.00	7.00	8.50	0.00	0.00	22.00	56.50
HAEA	0.00	1.00	1.00	7.33	8.00	27.00	44.33
GenHive	0.00	10.00	6.50	7.00	2.00	6.00	31.50
ACO-HH	0.00	17.00	0.00	6.33	6.00	1.00	30.33
SA-ILS	0.25	0.00	18.50	0.00	0.00	4.00	22.75
AVEG-Nep	9.50	0.00	0.00	0.00	0.00	9.00	18.50
XCJ	3.50	10.00	0.00	0.00	0.00	5.00	18.50
DynILS	0.00	9.00	0.00	0.00	8.00	0.00	17.00
GISS	0.25	0.00	10.00	0.00	0.00	6.00	16.25
SelfSearch	0.00	0.00	3.00	0.00	2.00	0.00	5.00
MCHH-S	3.25	0.00	0.00	0.00	0.00	0.00	3.25
Ant-Q	0.00	0.00	0.00	0.00	0.00	0.00	0.00

155

A Memetic Algorithm for Solving a Project Scheduling Problem

S. Asta,D. Karapetyan, A. Kheiri, E. Özcan, and A.J. Parkes, Combining Monte-Carlo and Hyper-heuristic methods for the Multi-mode Resource-constrained Multi-project Scheduling Problem, Journal of Scheduling, in review.

TeamID#3

School of Computer Science

The University of Nottingham
UNITED KINGDOM · CHINA · MALAYSIA

MISTA 2013 Challenge

- Aim: Develop an algorithm that produces the best possible solution to any given problem in 300 sec.
  - Problem instances are not known in advance.
- 21 teams registered, 16 teams qualified after the first round, 9 teams qualified after the final round.
- We designed a memetic algorithm – construct and improve

157

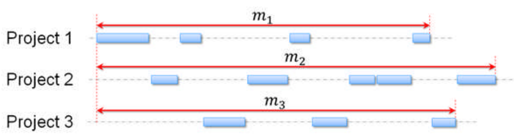
Problem Description

Resource-Constrained Project Scheduling

- Schedule given jobs
- Limited resources
- Precedence relations
- Minimise makespan

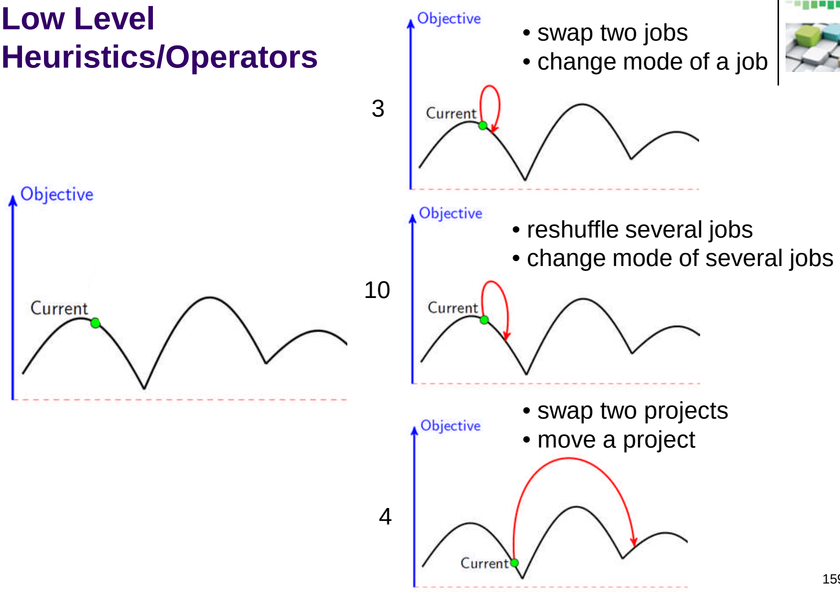
Multi-mode Resource-constrained Multi-project Scheduling

- Multiple modes for each job
- Multiple projects
- Local and global resources
- Minimise the sum of makespans



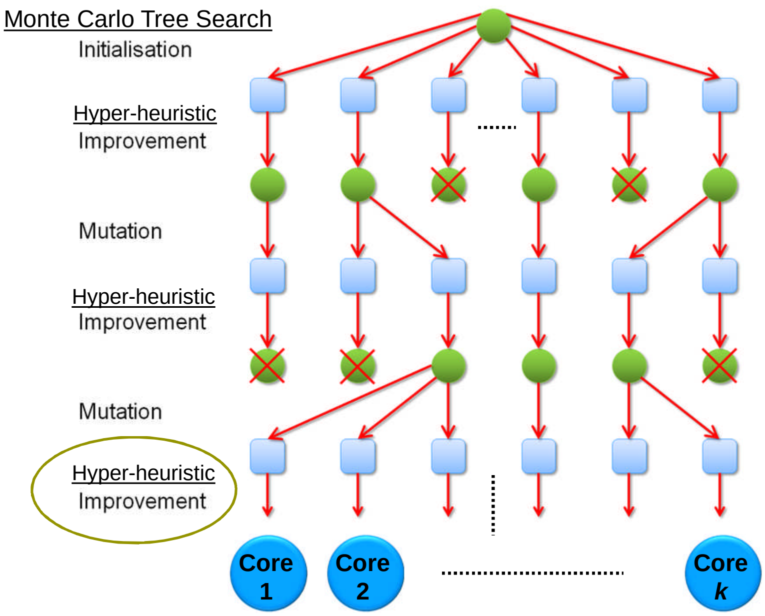
A Multi-stage Hyper-heuristic

Low Level Heuristics/Operators



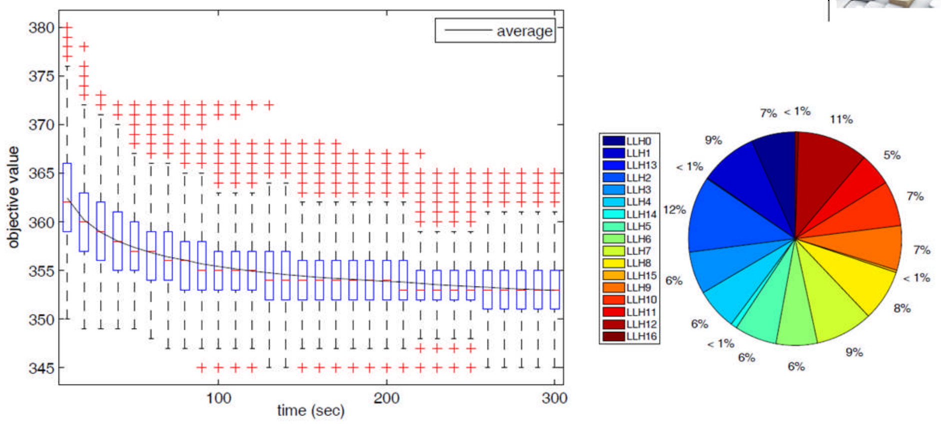
159

A Memetic Algorithm



160

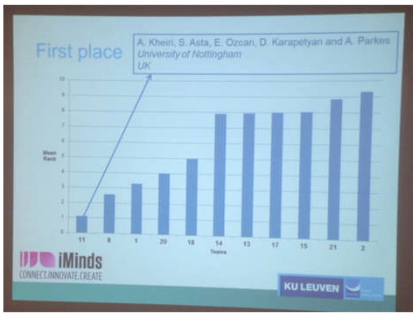
Results



161

MISTA 2013 Challenge – Result

Competition results			
Instance	TPD	TMS	Found by
B-1	349	127	3
B-2	434	160	1
B-3	545	210	3
B-4	1274	289	2
B-5	820	254	3
B-6	912	227	3
B-7	792	228	3
B-8	3176	533	3
B-9	4192	746	3
B-10	3249	456	3
X-1	392	142	1
X-2	349	163	3
X-3	324	192	3
X-4	955	213	3
X-5	1768	374	3
X-6	719	232	3
X-7	861	237	3
X-8	1233	283	3
X-9	3268	643	3
X-10	1600	381	3



- We produced the best solutions for 17 out of the 20 instances
- On the 12th second our algorithm becomes the winner

162

Summary (and Potential Future Research Directions)

Summary

- Hyper-heuristic research originated from a job shop scheduling application and has been rapidly growing since then.
- Generation hyper-heuristics are commonly used in the area
  - Train and test fashion
    - Does the selected subset of training instances is sufficiently representative of the test set?
    - Training is time-consuming (delta/incremental evaluation, surrogate functions)
  - The evolved heuristics might not be easy to interpret, yet they can outperform human designed heuristics

164

Summary (cont.)

- There is empirical evidence that machine learning/analytics/ data science help to improve the hyper-heuristic search process
  - Problem features vs solution/state features
  - Offline versus online learning – Life long learning
- There is still a lack of benchmarks
  - Problem domains are needed
- Multi-criteria, multi-objective and dynamic problems

165

Summary (cont.)

- Domain barrier issues
  - What constitutes as domain independent information
- More/less number of heuristics
  - Minimal heuristic set
- Multistage hyper-heuristics
  - Which hyper-heuristics to combine?
  - How to switch from one to another?
  - How to decide on the low level heuristic set?
  - Is there an end to the recursion/levels? (hyper^{N}heuristic)

166

Summary (cont.)

- Finding common representations or description formats that unify different but related problems
  - Example: grouping hyper-heuristics
  - Design a solver for the problems with binary/permutation/integer packed representation
- How do we compare hyper-heuristics?
  - Fairness issues: Termination criteria
  - If we test a hyper-heuristic on
    - new problem instances
    - new problem domains

167

Summary (cont.)

- Automated design of search methodologies is extremely challenging
  - Addressed in almost complete absence of a mathematical and theoretical understanding
- Heuristic Understanding
  - How can we analyse the search space of heuristics?
  - How can we visualise the search space of heuristics?
  - Is it possible to learn from small examples and apply to large instances?

168

Summary (cont.)


c|net
Search CNET
Reviews
News

Intel reportedly eyes future 18-core 'Broadwell' chip

What kind of high-performance silicon is Intel planning in the not-too-distant future? A site that covers chips says that future includes an 18-core processor.

the INQUIRER
We like to think of ourselves as the Microsoft of the energy world - Enron CEO Kenny Lay
Home | News | Reviews | Video | INQdepth | Jobs | Down
Software > Developer

Java SE 8 lands with Lambda expressions making coding easier for multi-core processors

169

References

- 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.
- S. A. Etaner-Uyar, E. Özcan, N. Urquhart (editors), Automated Scheduling and Planning From Theory to Practice, Studies in Computational Intelligence, Volume 505, ISBN: 978-3-642-39303-7 (Print) 978-3-642-39304-4 (Online), 2013.
- E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. Woodward (2010). A Classification of Hyper-heuristic Approaches. In Gendreau, Michel and Potvin, Jean-Yves (eds.), Handbook of Metaheuristics, International Series in Operations Research & Management Science, Volume 146, pp. 449-468. Springer.
- E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. Woodward (2009). Exploring hyper-heuristic methodologies with genetic programming. In C. Mumford and L. Jain (eds.), Computational Intelligence, Intelligent Systems Reference Library, pp. 177-201. Springer
- E. Özcan, B. Bilgin, E. E. Korkmaz, A Comprehensive Analysis of Hyper-heuristics, Intelligent Data Analysis, 12:1, pp. 3-23, 2008.

Q&A

Thank you.

Ender Özcan
ender.ozcan@nottingham.ac.uk

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