Towards A Learning–based Heuristic Searching Reform Scheme

14 July 2010, Lisbon

Fan Xue, CY Chan, WH Ip, CF Cheung

Department of Industrial & Systems Engineering
Hong Kong Polytechnic University

The Hong Kong Polytechnic University
香港理工大學

Department of Industrial and Systems Engineering
工業及系統工程學系

Outlines

1 Introduction

2 The presented method

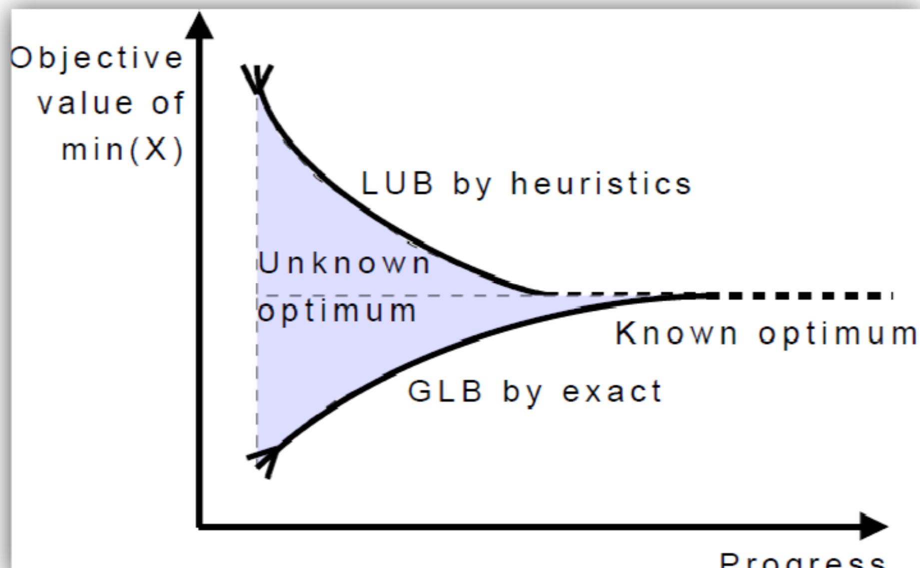
3 Traveling salesman: an example

4 Staff rostering: another example

5 Discussion and conclusion

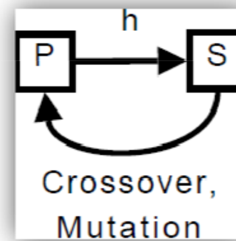
Opportunity and background

- Many combinatorial optimizations are NP-hard
  - "…no good algorithms…" (Edmonds, 1967)
  - The larger, the much more difficult to solve ☹



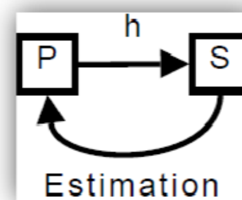
A typical problem solving progress

- Different metaheuristics have been proposed to improve searching (h), e.g.,
  - GA



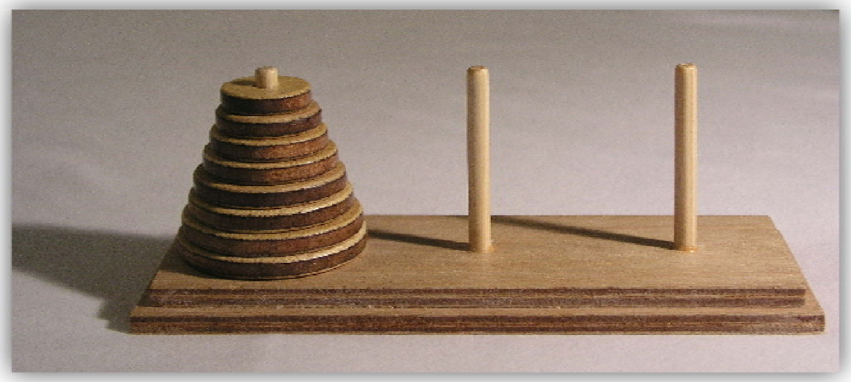
  - LS

  - EDAs



An inspiring game

- The game of *Tower of Hanoi* consists of:
  - Three rods,
  - A number of disks of different sizes.
- The puzzle starts with the disks in a neat stack in ascending order of size on one rod.
- The objective is to move the stack to another rod, obeying:
  - No disk on top of a smaller one
  - One disk at a time.
- To unveil the solving rules, play with 2 or 3 disks at first.
  - *Learn from a small sample*



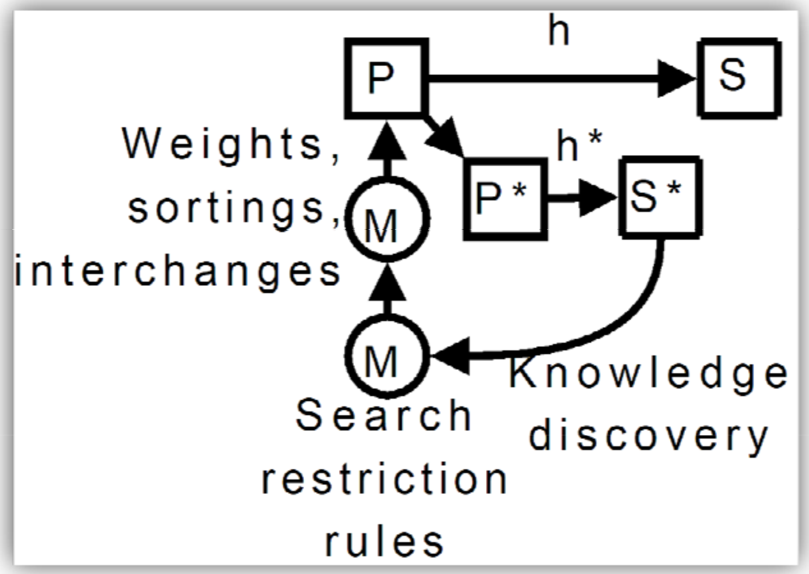
A model of Tower of Hanoi (8 disks, Photo brought from Wikipedia)

Objective and assumptions

- The objective is to improve heuristics through learning-based revisions of assignments of variables
- Assumptions
  - Recognizable problems
  - “Homogeneous” variables
- Notes
  - The smaller, the much easier (NP-hardness ☺)
  - The 1st assumption makes learning possible
  - The 2nd assumption further enables learning from a part of the problem (variables), it implicitly enables learning from near-optimal solutions
  - Large-scale problems are preferred

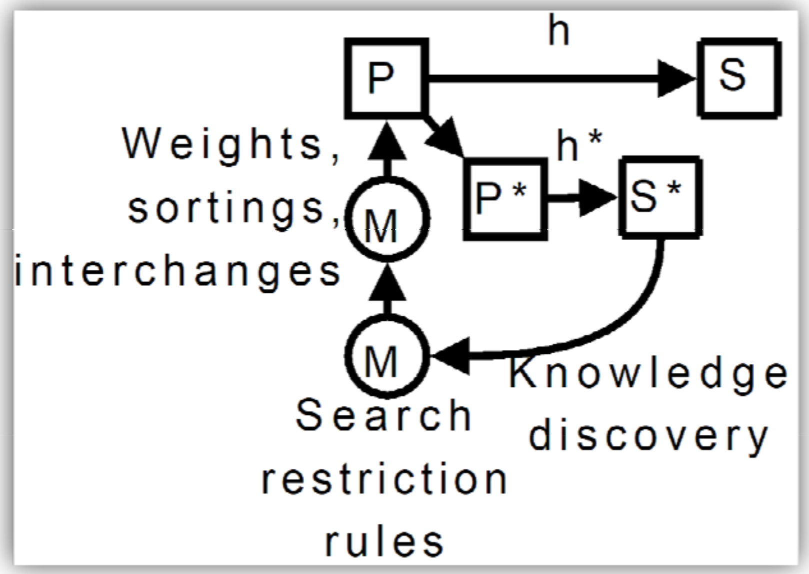
The proposed method

- The phases of the proposed method are:
  - 1. Start with a problem "P"
  - **2**. Find a *small* "representative" part "P*"
  - 3. Obtain a good solution "S*" quickly
  - **4**. Obtain rules about assignments from "S*" as complete as possible
  - 5. Interpret the rules to weights, sorting, or interchanges of possible assignments of the variables
  - 6. Reform the assignment process of heuristic searching "h" in the problem "P"



The proposed method

- Notes
  - Size(P*) << size(P)
  - h* ≠ h (not necessarily same, nor necessarily heuristic)
  - The indirect way of using the learning results
    - Rules with confidences from 100% down to 1% are potentially useful.
  - Interpretations for different heuristics:
    - Weights for value assignments
    - Sorting for tests of local search
    - Interchanges for tests of binaries
    - …

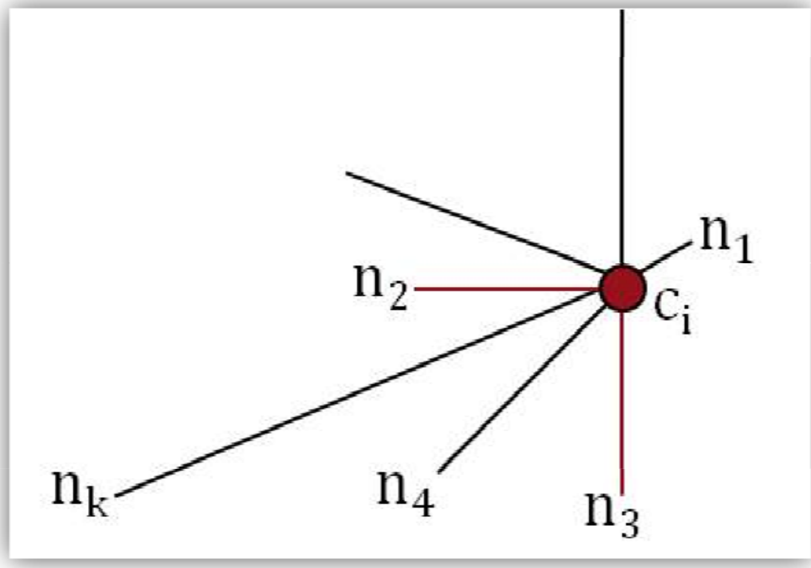


Traveling salesman as an example

- The Euclidean traveling salesman problem (TSP): finding a shortest tour that visits all given spatial points (cities).
  - Hamilton circle: two edges for each city
  - Most of very long edges are not possible to appear in the optimal tour(s)
- How does the method work?
  - Indentify a weight for each edge candidate of each city
  - Reorder and reform the possible candidates by the weights (thus the candidate-set-based neighborhoods).
- How to indentify the weights?
  - Learn from a part of the given problem, with a set of attributes for the edge candidates

Traveling salesman: attributes

- The attributes of an edge (c_i, n_j) for a city c_i
  - G1 Global nearest
  - R1, R2, R3 Length indices comparing to (c_i, n_1), (c_i, n_2), (c_i, n_3)
  - S1, S2 Whether $d(c_i, n_1) \le d(c_i, n_2)/2$, $d(c_i, n_2) \le d(c_i, n_3)/2$
  - P1, P2, P3 R1-R3 of n_j
  - Q1, Q2 S1, S2 of n_j
  - Ag, Ah Minimal / maximal angular gap around c_i
  - An Number of directions around c_i
  - Opt Whether appears in the training sample or not



Traveling salesman: sample data

- Learning samples

G1	R1	R2	R3	S1	S2	P1	P2	P3	Q1	Q2	Ag	Ah	An	Opt
			...					...						...
0	3	1	1	1	0	4	3	2	0	0	3	10	7	0
0	9	3	3	1	0	6	6	2	0	1	3	10	7	1
0	9	3	3	1	0	10	4	2	1	1	3	10	7	1
			...					...						...

- Sample rules ("Opt=1" only)

Id	Rule	Support	Confidence
1	R1=3, S1=1, Q1=1 => Opt=1	0.013	1.000
2	P1=3, S1=1, Q1=1 => Opt=1	0.013	1.000
3	R1=3, S1=1, Q2=0 => Opt=1	0.012	1.000
...	...	...	...
30	G1=1 => Opt=1	0.022	0.913
...	...	...	...
983	R3=8 => Opt=1	0.048	0.010

Traveling salesman: revising the assignments

- Weights of edge candidates
  - Highest confidence of the rule that implies the edge should be in optimal tour (Opt=1)
  - Range [0, 1]
- Reorder (and reorganize) the candidates by
  - The weights descending
  - Distance×(1-weight) (Weighted Distance, WD) ascending
  - Grouping
  - Or other sorting plans…
- For those candidate sets not determine by Euclidean distance, a pseudo-distance could be defined.
  - E.g., a pseudo-distance = ln(α-value) for the α-nearness

Traveling salesman: tests

- Inputs
  - 32 large Euclidean TSPs from industry, geography and random generation, grouped, ranging from 3,000 to 1,000,000 cities.
- Objective algorithm
  - 5-Opt (100 runs) initialized by Greedy, on 5-sized candidate sets
- Parameters (Class Association Rules, CARs)
  - P* = 3,000 cities with a closest density (and same aspect ratio)
  - Min confidence of learning = 0.01
  - Min support of learning = 0.001
  - Learn and reform 50-sized (if applicable) candidate sets
- Optional parameters
  - Length control of rules: |antecedent| < 6 (learns much faster without much loss of rules)

Traveling salesman: tests

- Groups of instances to test

Category	VLSI(BK)	E(BK)	TSPLIB(Optimum)
3k	lsn3119(9114*) lta3140(9517*) fdp3256(10008*)	E3k.0(40634081*)E3k.1(40315287*) E3k.2(40303394*)E3k.3(40589659*) E3k.4(40757209)	pr2392(378032) pcb3038(137694) fnl4461(182566)
10k	dga9698(27724) xmc10150(28387)	E10k.0(71865826)E10k.1(72031630) E10k.2(71822483)	pla7397(23260728) brd14051(469385)
31k	pbh30440(88328) xib32892(96757)	E31k.0(71865826) E31k.1(72031630)	pla33810(66048945)
100k	sra104815(251433)	E100k.0(225787421) E100k.1(225659006)	pla85900(142382641)
316k	ara238025(578775) lra498378(2168067)	E316k.0(401307462)	-
1M	lrb744710(1612132)	E1M.0(713189834)	-

* Also proved optimal

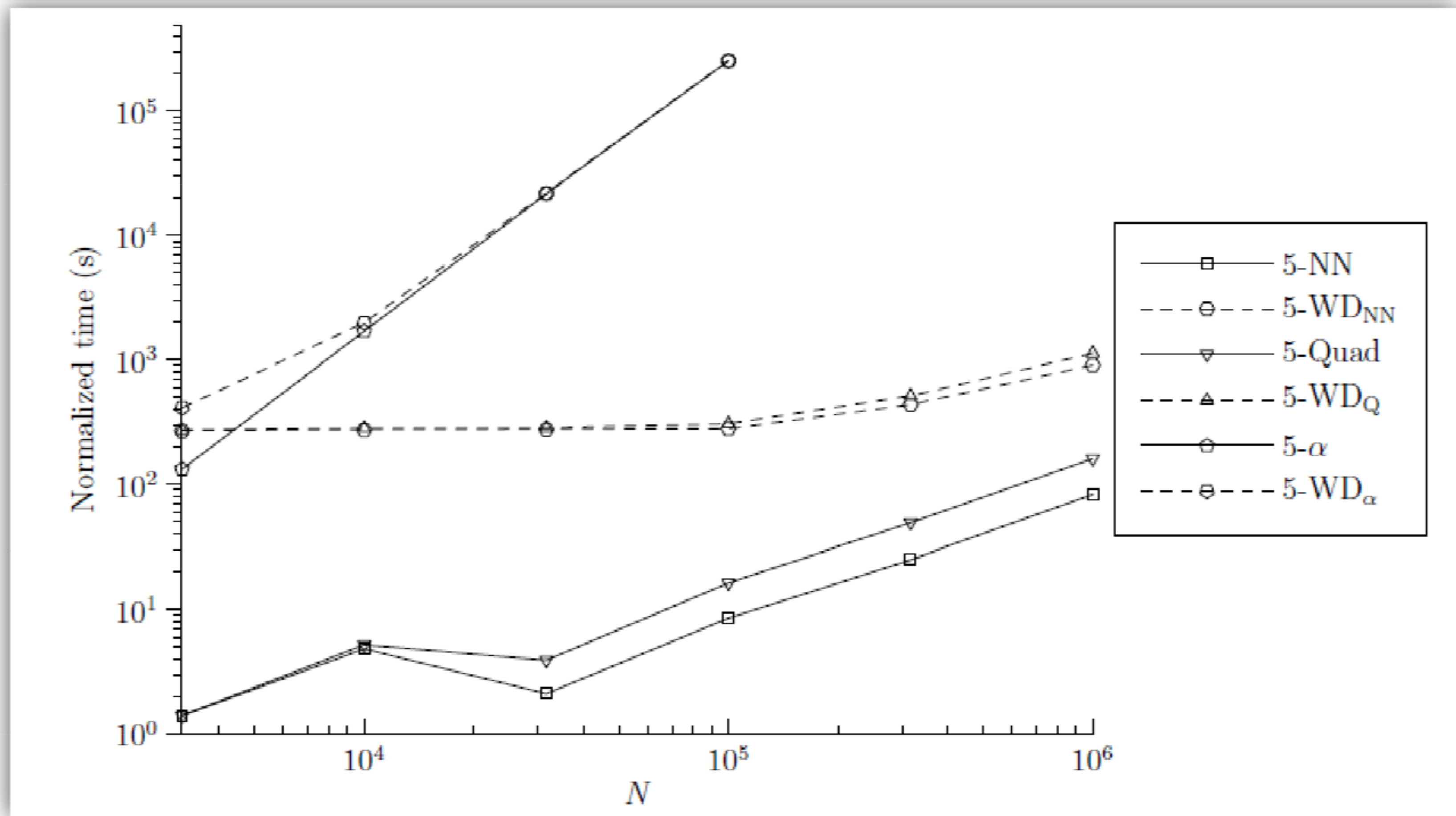
Traveling salesman: results

❖ Average quality (% excess BK) comparison

		G+5-Opt @ NN			G+5-Opt @ Quadrant			G+5-Opt @ α-nearness		
		Avg	Avg/WD	Imp(%)	Avg	Avg/WD	Imp(%)	Avg	Avg/WD	Imp(%)
VLSI	3k	3.889	2.663	**31.5**	0.695	0.649	6.7	0.361	**0.327**	**9.3**
	10k	4.236	3.300	**22.1**	0.863	0.693	**19.7**	0.526	**0.503**	**4.5**
	31k	4.169	2.913	**30.1**	0.814	0.642	**21.2**	0.454	**0.437**	**3.7**
	100k	6.657	6.467	2.9	0.842	0.752	**10.7**	0.339	**0.328**	**3.2**
	316k	9.959	7.950	**20.2**	1.183	0.917	**22.5**	-	-	-
	1M	4.682	4.385	6.3	0.857	0.762	**11.1**	-	-	-
E	3k	0.703	0.487	**30.7**	0.346	0.338	2.3	0.156	**0.156**	0.3
	10k	0.862	0.490	**43.1**	0.375	0.370	1.4	0.179	**0.178**	0.2
	31k	1.262	0.659	**47.8**	0.527	0.526	0.2	0.343	**0.341**	0.6
	100k	1.851	0.646	**65.1**	0.438	0.434	0.9	0.252	**0.250**	0.8
	316k	1.660	0.679	**59.1**	0.430	0.422	1.9	-	-	-
	1M	1.176	0.911	**22.5**	0.381	0.379	0.5	-	-	-
TSPLIB	3k	0.456	0.358	**21.4**	0.340	0.321	5.4	0.143	**0.134**	**6.5**
	10k	2.878	2.234	**22.4**	0.427	0.395	7.6	**0.253**	0.278	-10.1
	31k	2.297	1.677	**27.0**	0.913	**0.517**	**43.4**	0.560	0.617	-10.2
	100k	2.065	1.476	**28.5**	0.761	**0.445**	**41.5**	0.932	0.978	-4.9

Traveling salesman: results

- Set up time costs (dots = weighted distance)



Traveling salesman: results

❖ Average time cost comparison

		G+5-Opt @ NN			G+5-Opt @ Quadrant			G+5-Opt @ α-nearness		
		Avg	Avg/WD	Imp/%	Avg	Avg/WD	Imp(%)	Avg	Avg/WD	Imp(%)
VLSI	3k	2.20	2.27	**-3.1**	1.93	1.48	23.0	2.32	2.21	**4.6**
	10k	8.09	7.84	**3.2**	8.72	6.64	**23.9**	10.32	9.69	**6.1**
	31k	33.20	31.79	**4.3**	37.66	29.40	**21.9**	42.88	46.18	**-7.7**
	100k	88.57	86.00	2.9	147.62	133.05	**9.9**	158.95	169.70	**-6.8**
	316k	479.84	421.65	**12.1**	675.39	649.52	**3.8**	-	-	-
	1M	1123.66	949.97	15.5	1665.11	1500.81	**9.9**	-	-	-
E	3k	2.60	2.47	**5.1**	1.68	1.87	-11.6	1.98	2.08	-5.3
	10k	9.86	10.28	**-4.2**	7.94	7.56	4.8	8.91	8.56	3.9
	31k	41.81	45.40	**-8.6**	37.18	36.69	1.3	47.20	43.73	7.4
	100k	140.30	156.68	**-11.7**	141.62	139.84	1.3	161.85	167.15	-3.3
	316k	503.66	568.11	**-12.8**	601.57	596.53	0.8	-	-	-
	1M	2141.66	2432.96	**-13.6**	2986.15	3033.61	-1.6	-	-	-
TSPLIB	3k	2.71	2.68	**1.3**	2.18	1.84	15.6	1.88	4.07	**-116.7**
	10k	12.88	13.57	**-5.3**	12.60	11.17	11.3	13.40	13.57	-1.3
	31k	97.50	97.02	**0.5**	81.40	65.16	**19.9**	84.44	97.27	-15.2
	100k	149.99	159.61	**-6.4**	174.31	166.20	**4.7**	134.96	150.73	-11.7

Traveling salesman: results interpretation

- ☺
  - The most popular search heuristic, local search, can be significantly benefited on different candidate sets (NN, Quadrant, α-nearness) over different families (especially industrial) of problems
  - The additional time cost is pretty low in very large problems
- ☹
  - Less effective in random than industrial ETSP
  - Less effective for the α-nearness than the NN and the Quadrant candidate sets

Staff rostering as another example

- Staff rostering
  - Determine shifts for demands
  - Construct work timetables*
- Attributes
  - ID, CN   Employee ID, Contract ID (group)
  - S1, S2   Shift on yesterday, on the day before yesterday
  - SQ   Length of current consecutive working days
  - DW   Day of week
  - St, Ed   Level ($\log_2$) of days from the beginning, to the end
  - LD   Absolute difference of the current employee's workload against the average workload (till yesterday, rounded to integer).
  - JB   Shift to determine

+ -	1							2							3					
2005 Jan	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	2
	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T
A																				
B	D1	D1				D1				D1	D1	D1	D1		D1	D1		D1	D1	D
C			D1	D1	D1	D1		D1	D1								D1	L	D1	
D	E	E	N	N		L	N			D2	D2	D2	N		L	L	D2	D2		
E			E	E	E	E		E	E	E	E		E	D2	D2	D2	D2		L	L

Staff rostering: tests

- Inputs
  - Problems (>10 staff, >20 days, fixed number of shifts) from http://www.cs.nott.ac.uk/~tec/NRP/
  - A set of enlarged problems (no day/shift on/off constraints, enlarged to same employees, 3 months)
- Objective algorithm
  - 4-Hybrid VDS (10 runs) initialized by Greedy
- Parameters (CARs)
  - P* = half scheduling period, or those before
  - Min confidence of learning = 0.01
  - Min support of learning = 0.05*

  *: Less training examples (~1,000) than in TSP (~100,000)

Staff rostering: results

- Comparisons on two groups of problems

Problem	BK	4-HVDS			4-HVDS /Weighted			Δ time
		avg	stddev	time(s)	avg	stddev	time(s)	(%)
BCV-2.46.1(46x28)	1572*	**1576**	8.7	631.8	1582	10.8	616.2	-2.47
BCV-3.46.1(46x26)	3280^	3314	7.4	1590	**3307**	11.7	1808	13.7
BCV-3.46.2(46x26)	894*^	**896.1**	1.8	1148	898	1.6	1014	-11.7
BCV-6.13.1(13x30)	768	884.9	101.9	211.1	**833.5**	82.1	204.6	-3.07
BCV-A.12.1(12x31)	1294^	2217	493.5	1678	**1983**	403.2	2003	19.4
BCV-A.12.2(12x31)	1953^	**2440**	188.8	2819	2486	298.5	2160	-23.4
ORTEC01(16x31)	270*^	2254	915.5	29.4	**2128**	1731	26.2	-10.9
QMC-1(19x28)	13*	**31.3**	3	61.6	34.7	2.9	50.1	-18.7
SINTEF(24x21)	0*	9	1.9	12.6	**8.8**	2.3	13.5	6.92
Valouxis-1(16x28)	20*	**422**	7.9	6.2	476	98.3	4.6	-26
* Also proved optimal; ^ found by the Hybrid VDS								
EBCV-4.13.1 (13x3m)	-	155.8	28.6	352.3	**153.9**	98.8	413.6	17.4
EBCV-5.4.1 (4x3m)	-	525.9	132.3	0.8	**462.7**	0.5	1.5	89.6
EGPost-B (8x3m)	-	3223	1939	68	**2599**	1411	63.2	-7.1
EMillar-2Shift-DATA1(8x3m)	-	3650	97.2	8.5	**3640**	51.6	6.9	-18.2
EMillar-2Shift-DATA1.1(8x3m)	-	3640	51.6	1.6	**3620**	42.2	2.7	68.3
EValouxis-1 (16x3m)	-	1656	252.8	109.3	**1632**	161.2	143.8	31.5

Staff rostering: results interpretation

- ☺
  - Fits large-scale problems better
  - According to *limited* evidences, the Hybrid VDS can be benefited in quality, if certain criteria (such as "large-enough") are met
- ☹
  - Although the additional time costs by machine learning are low, the iteration time increases by some percent
  - Preliminary tests only. There might be some other reasons for the quality change (i.e., possibly no improvements by the learning in fact)…

Discussion

- Some characteristics:
  - The parameters of learning (including non-CARs) are easy to determine: set to feasibly minimal
  - The design of decision attributes is the key to a successful application: decentralized, able to borrow the attributes from human heuristics
- Beyond the two tests, more challenges await
  - Heuristics/ CO problems incompatible (not homogeneous)?
  - Problems with *many arbitrary global* constraints (e.g., SAT)
  - Constraint satisfaction methods (e.g., revising backtracks?)
  - Some exact methods (e.g., branch-and-bound ?)
  - An encapsulated general purpose (or a list of purposes) optimization program module

Conclusion and future works

- We present an efficient metaheuristic-like approach
  - Small-problem-oriented learning (thus fast)
  - Enhance problem solving with the rules learnt
  - Transparent to the embedded heuristic
- We find the results of tests encouraging.
- We hope it unveils a direction to take the power of machine learning in large-scale optimization.
- Possible future works
  - An general guide of designing the attributes
  - Special plan guide for special industrial practice
  - Challenges listed on last page

References

- Edmonds, J. (1967). Optimum branchings, Journal of Research of the National Bureau of Standards, 71B: 233-240.
- TSP benchmark data
  - http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/
  - http://www.research.att.com/~dsj/chtsp/
  - http://www.tsp.gatech.edu/vlsi/
  - http://www.akira.ruc.dk/~keld/research/LKH/DIMACS_results.html
- Rostering benchmark data
  - http://www.cs.nott.ac.uk/~tec/NRP/

Thank you for your attention!

E-mail addr.: Dewolf.xue@polyu.edu.hk
Dewolf_matri_x@msn.com

THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學

Department of
Industrial and Systems Engineering
工業及系統工程學系