Applications of Genetic Programming in Dynamic Scheduling

Domagoj Jakobović and **Marko Đurasević**, University of Zagreb, Croatia

Yi Mei and Mengjie Zhang, Victoria University of Wellington, New Zealand

Su Nguyen, La Trobe University, Australia

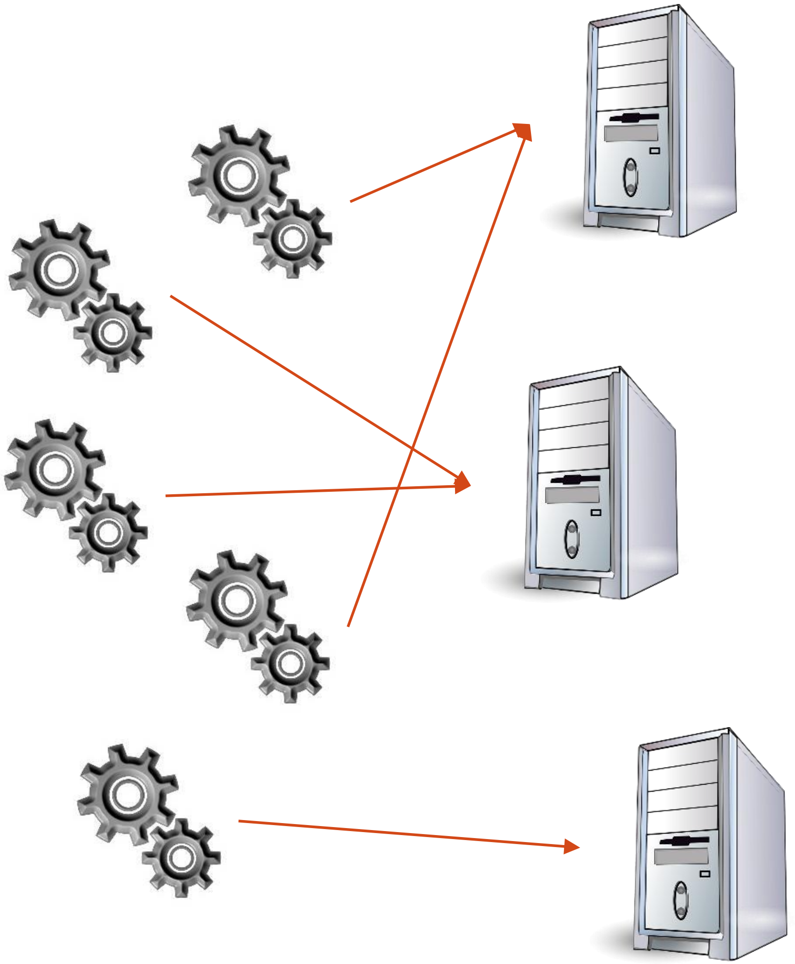
8th September 2018, PPSN-2018, Coimbra, Portugal

Outline

- Introduction
- Scheduling problems
- Genetic programming
- Automatically designing new DRs
- Existing research
- Open issues and topics
- Conclusion and outlook

Scheduling problems

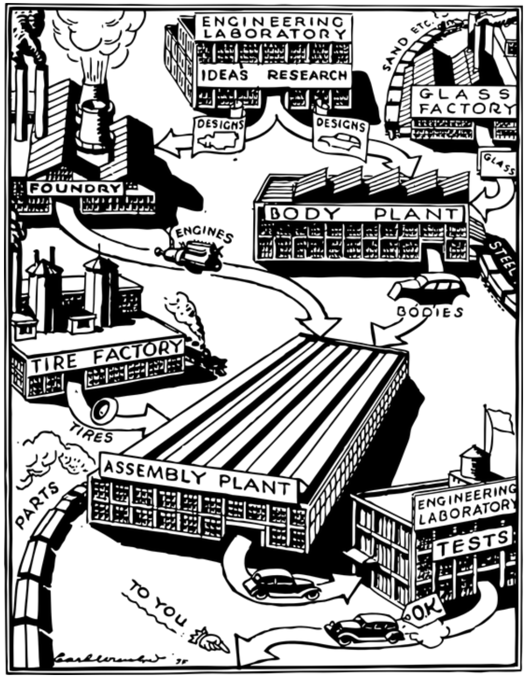
- Allocating tasks (jobs) to certain resources (machines) [1]
- Goal: create a scheudule which optimises certain user defined criteria



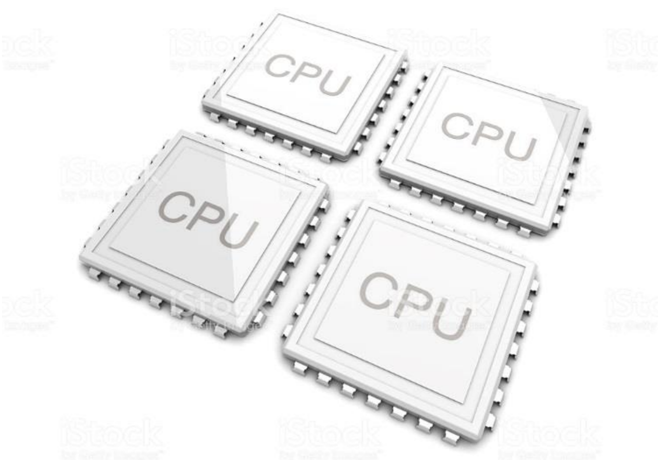
Scheduling problems

Scheduling in grids/clusters

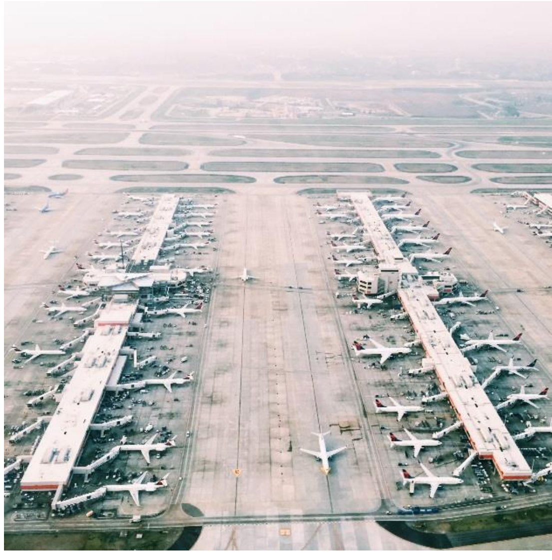
Manufacturing



Scheduling on multiprocessors



Airplane scheduling



Scheduling classification

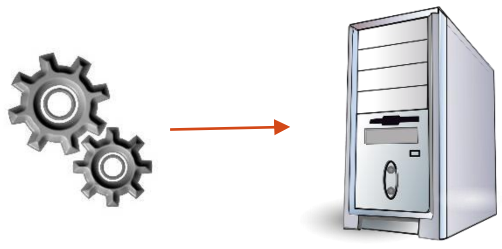
- Due to a large amount of problems a special scheduling classification was designed [1]:
  - Machine environment
  - Optimised criteria
  - Additional constraints
  - Scheduling conditions

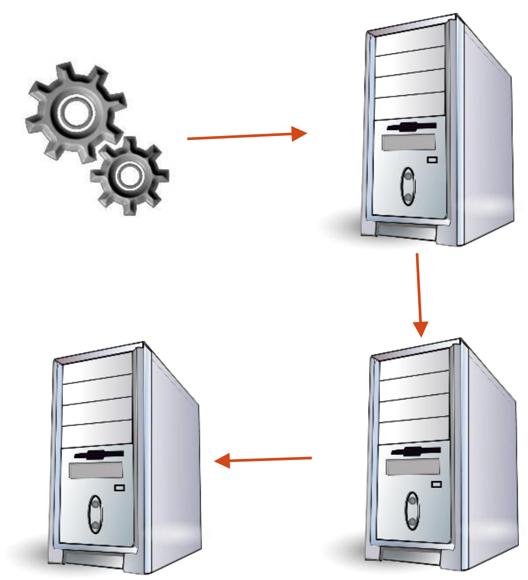
Input of scheduling problems

- m machines and n jobs
- Job properties:
  - Processing time
  - Release time
  - Due date
  - Weight
  - Deadline
  - …

Machine environments

- Single stage
  - Job is executed on a single machine
  - Single machine, unrelated machines
- Multi stage
  - Job is executed on a series of machines
  - Job shop, flow shop, open shop





Additional constraints

- Machine breakdowns
- Machine eligibility
- Precedence constraints
- Setup times
- Batch processing
- Release times

Optimisation criteria

- Total duration of the schedule
- Tardiness of all jobs
- The time jobs spent in the system
- Maximum tardiness
- Number of tardy jobs
- ....

Scheduling conditions

- Parameter reliability:
  - Deterministic
  - Stochastic
- Parameter availability:
  - Offline
  - Online
- Schedule construction:
  - Static
  - Dynamic

Dynamic scheduling

- No information about jobs is known beforehand
- Information about jobs becomes available as they are released
- The schedule needs to be constructed simultaneously with the system execution
- Speed is of the essence (decisions must be performed quickly)

Solving scheduling problems

- Exact algorithms [2]
  - Can obtain the optimal solution, but computationally expensive
- Approximation algorithms [3]
  - Obtain solutions worse than the optimal solution by a certain factor, but difficult to design
- Heuristic algorithms
  - Applicable for various criteria and conditions
  - Do not necessarily obtain the optimal solution
  - Improvement heuristics – iteratively improve a schedule [4]
  - Constructive heuristics – incrementally create a schedule

Dispatching rules (DRs)

- Simple scheduling heuristics [5]
- At each scheduling decision they determine which job should be scheduled on which machine
- To determine which job should be used a priority function is used:
  - EDD: $\frac{1}{d_j}$
  - WSPT: $\frac{w_j}{p_j}$
  - ERD: $\frac{1}{r_j}$

Example of a DR

Priority rule:

$$\pi_j = \frac{p_j * (d_j - time)}{w_j}$$

Job 1:
- $p = 10$
- $d = 17$
- $w = 0.8$

$\pi_1 = 212.5$

Job 2:
- $p = 7$
- $d = 30$
- $w = 0.5$

$\pi_2 = 420$

Schedule:

Machine 1

Time = 0

Example of a DR

Priority rule:

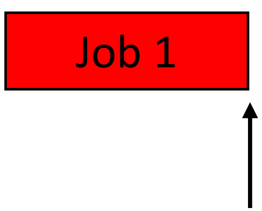
$$\pi_j = \frac{p_j * (d_j - time)}{w_j}$$

Job 2:
- $p = 7$
- $d = 30$
- $w = 0.5$

$\pi_2 = 280$

Schedule:

Machine 1



Time = 10

Job 3:
- $p = 13$
- $d = 25$
- $w = 0.7$

$\pi_3 = 278.6$

Example of a DR

Priority rule:

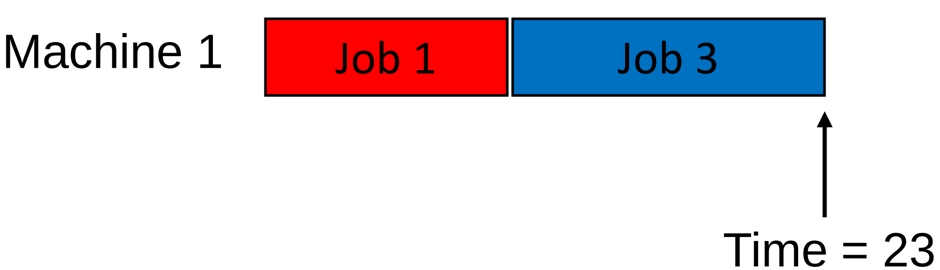
$$\pi_j = \frac{p_j * (d_j - time)}{w_j}$$

Job 2:

- $p = 7$
- $d = 30$
- $w = 0.5$

$\pi_2 = 98$

Schedule:

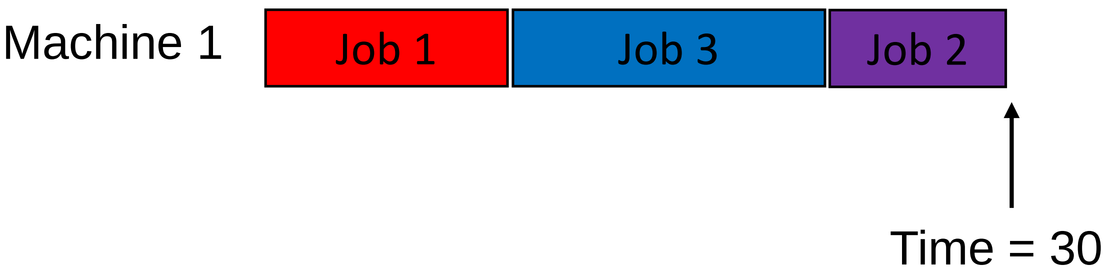


Example of a DR

Priority rule:

$$\pi_j = \frac{p_j * (d_j - time)}{w_j}$$

Schedule:

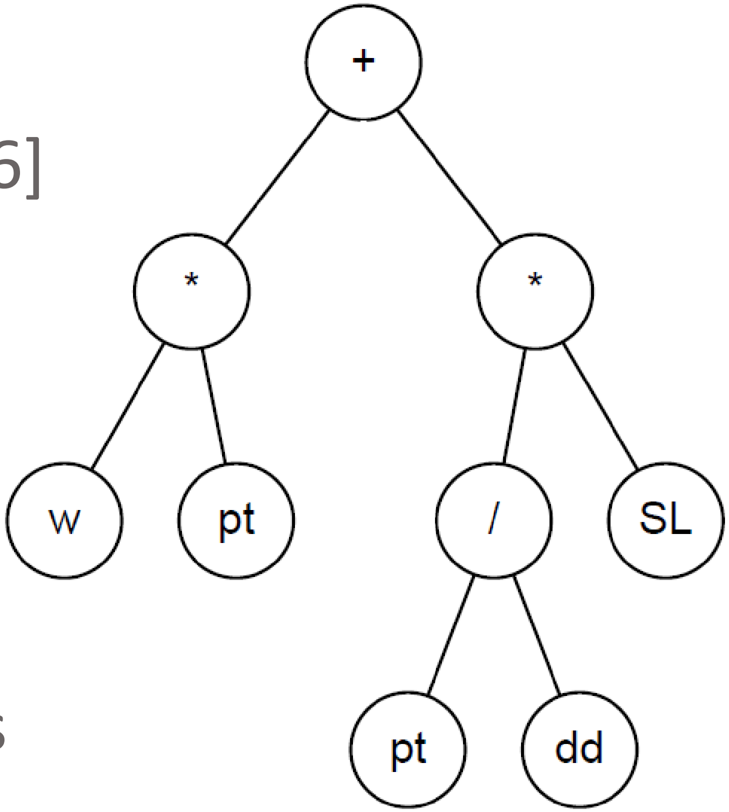


Issues with manually designed DRs

- Performance
  - How to improve their performance to obtain better schedules?
- Difficult to design manually
  - How to easily design new, efficient DRs?
- Which DRs to use
  - Which of the many DRs to use for a certain problem and criteria?

Genetic programming (GP)

- An evolutionary algorithm for solving optimisation problems [6]
- Individuals represent mathematical functions and expressions
- Leaf nodes represent job and system parameters
- Inner nodes represent functions
- Expression: $w * pt + \frac{pt}{dd} * SL$



How to design DRs with GP?

- DRs consist of two parts
  - A priority function:
    - $\pi_j = \frac{1}{r_j}$
  - A schedule generation scheme:
    - Schedule the job with the largest priority on the machine if it is free

How to design the priority function?

- Select appropriate terminal and function nodes, e.g.
  - Function nodes: +, -, *, /, min, max, pos
  - Terminal nodes:
    - Simple: processing time, release time, due date
    - Complex: slack of a job
  - Performance of the generated rules depends on the selected terminals

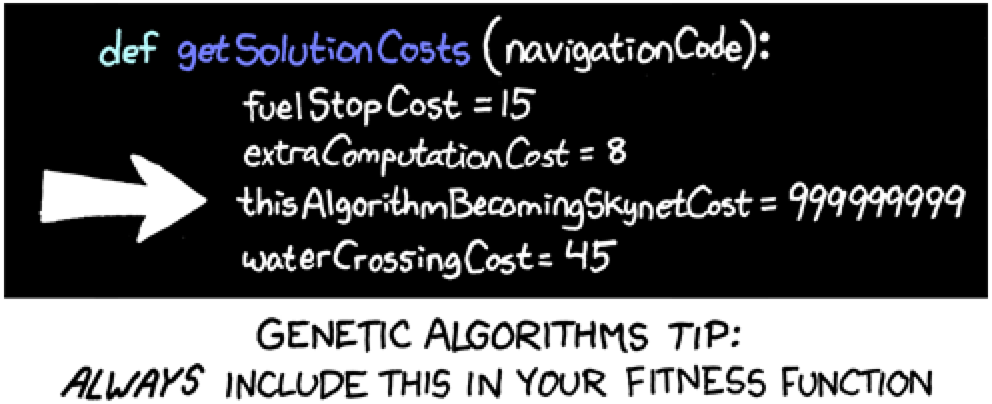
Schedule generation scheme

- Decide how the schedule should be constructed
- Should idle times be inserted?
- Are other constraints present?

```
while (unscheduled jobs exist) {
     wait until a machine becomes ready
     determine the priorities πi of all unscheduled jobs
     schedule the job with the best priority
}
```

How to evolve priority functions?

- Training set:
  - Predefined problem instances
  - Stochastic generation during execution
- Test set:
  - A new set for evaluation purposes



- Fitness function:
  - Ensure that all instances have the same influence!

Comparison with manually designed DRs

Criterion	Manually designed DR	Automatically designed DR (GP)	Improvement
Cmax	37.85	38.02	-0.4%
Ft	154.1	154.0	0%
Fmax	14.03	13.60	**3.1%**
Nwt	6.686	6.384	**4.5%**
Tmax	2.418	2.376	**1.7%**
Twt	13.30	12.96	**2.6%**

What has been done until now?

- Applied on various problems [7, 8, 9]
- Comparing different representations [10, 11, 12]
- Multi objective optimisation [13, 14, 15, 16, 17]
- Ensemble learning [18, 19, 20, 21]
- Due date assignment rules [22, 23, 24, 25, 26]
- Etc.

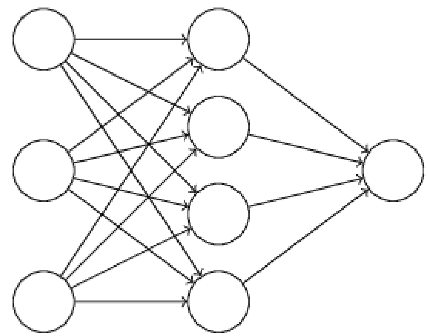
Why use GP?

Linear representation

$$\sum_i w_i s_i$$

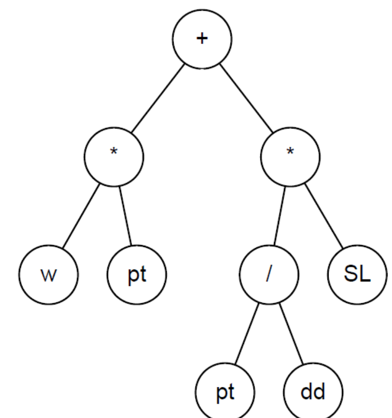
VS

Neural nettwork



VS

GP

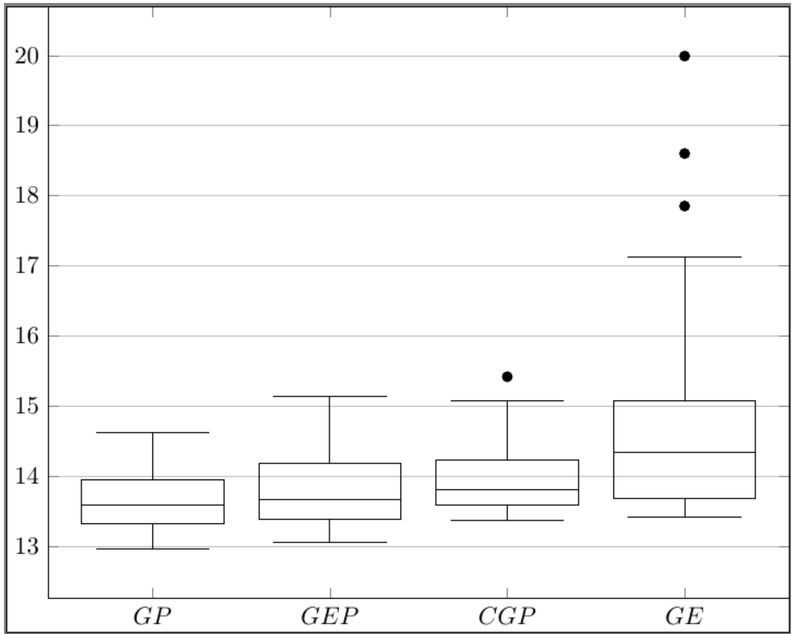


- GP and neural network obtain best results [11]
- GP trees are easier to interpret!

Performance of different GP methods

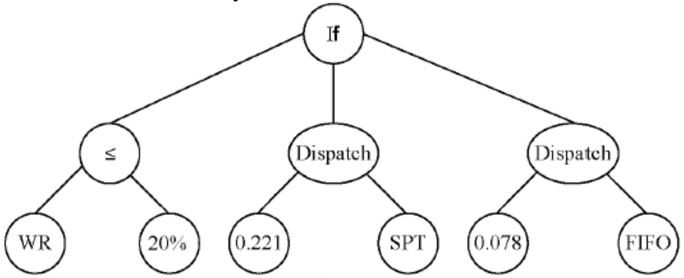
- Several options:
  - Genetic programming (GP)
  - Gene expression programming (GEP)
  - Cartesian genetic programming (CGP)
  - Grammatical evolution (GE)
- Similar performance (GE usually the worst)
- GEP and CGP generate smaller DRs

Method	Twt	DR size
GP	12.96	40.24
GEP	13.06	29.66
CGP	13.38	18.77
GE	13.41	17.40

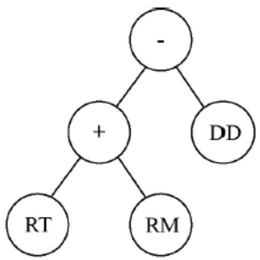


GP based DR representations [12]

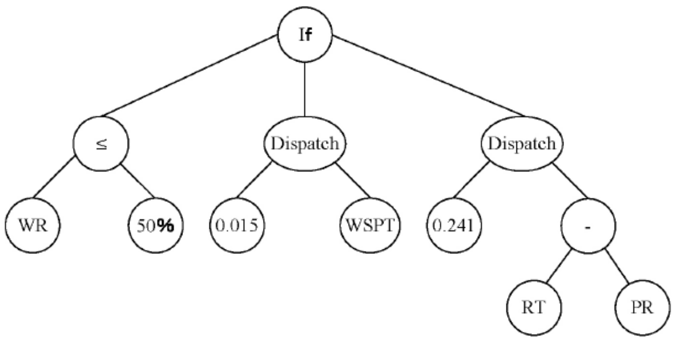
Decision-tree like representation



Arithmetic representation



Mixed representation



- The best representation depends on the problem
- Arithmetic representation most commonly used

Multi-objective optimisation

- Requirements for optimising several criteria simultaneously
- Application of various MOGP algorithms: NSGA-II, NSGA-III, HaD-MOEA, MOEA/D, SPEA2
- Considered from 3 to 9 criteria

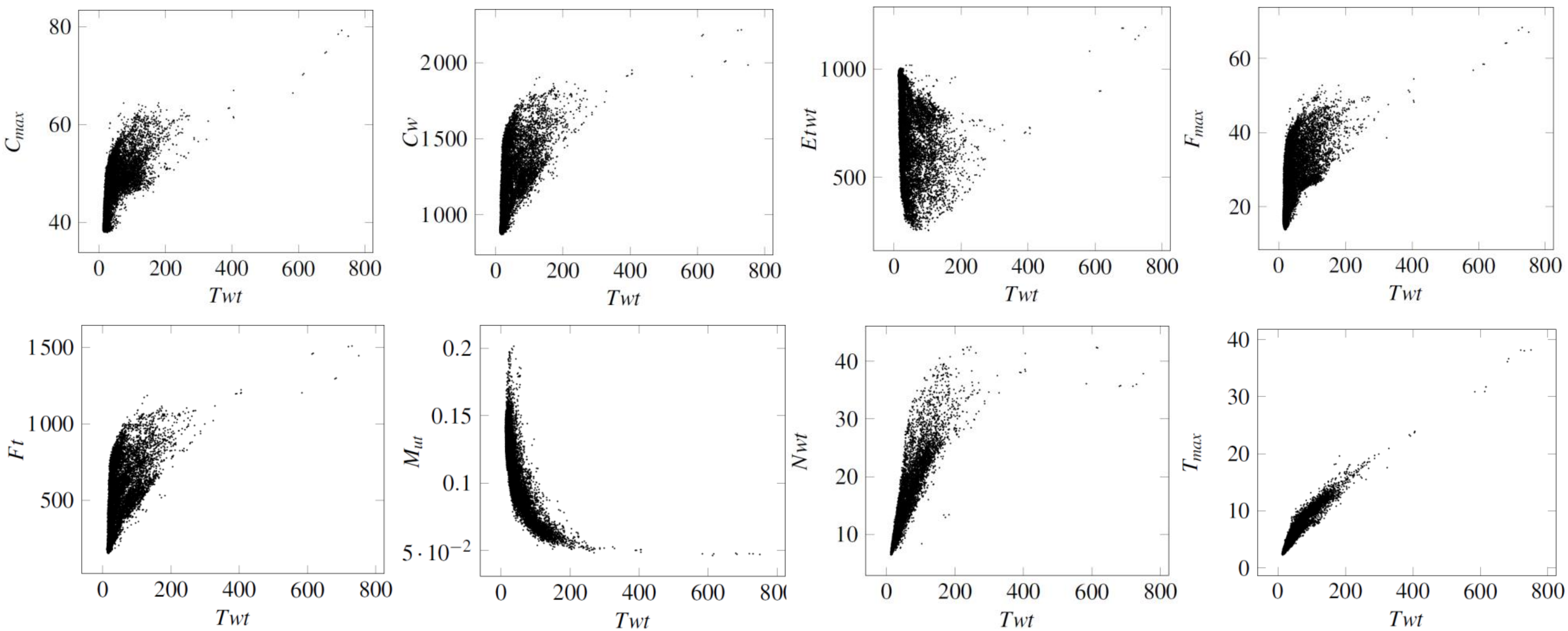
Multi-objective optimisation

- Comparison of automatically designed DRs with the manually designed ATC rule [14, 16]
- ATC defined as:
  - $\pi_j = \frac{w_{T_j}}{p_{ij}} * \exp\left(-\frac{\max(d_j - p_{ij} - time, 0)}{k * \bar{p}}\right)$

Method	Criteria								
	C_{max}	Cw	$Etwt$	F_{max}	Ft	M_{ut}	Nwt	T_{max}	Twt
ATC	38.26	901.5	968.5	14.27	195.9	0.127	6.686	2.418	13.30
Evolved DRs - three objectives									
R1	-	**893.4**	-	-	**173.5**	-	-	-	**12.79**
R2	-	-	-	-	-	-	**6.566**	**2.249**	**12.28**
Evolved DRs - five objectives									
R3	-	**900.5**	**968.5**	-	**179.3**	-	**6.333**	-	**13.00**
R4	-	-	-	17.00	**173.5**	-	**6.440**	**2.401**	**12.68**
Evolved DRs - six objectives									
R5	**38.22**	-	-	16.45	**178.1**	-	**6.306**	**2.410**	**12.85**
Evolved DRs - seven objectives									
R6	**38.08**	903.1	-	15.22	**182.9**	-	**6.650**	**2.390**	**13.22**
Evolved DRs - nine objectives									
R7	39.31	904.0	**967.9**	17.65	**182.5**	0.145	**6.566**	2.477	**13.08**

Multi-objective optimisation

- Correlation of the Twt criterion with other scheduling criteria [16]

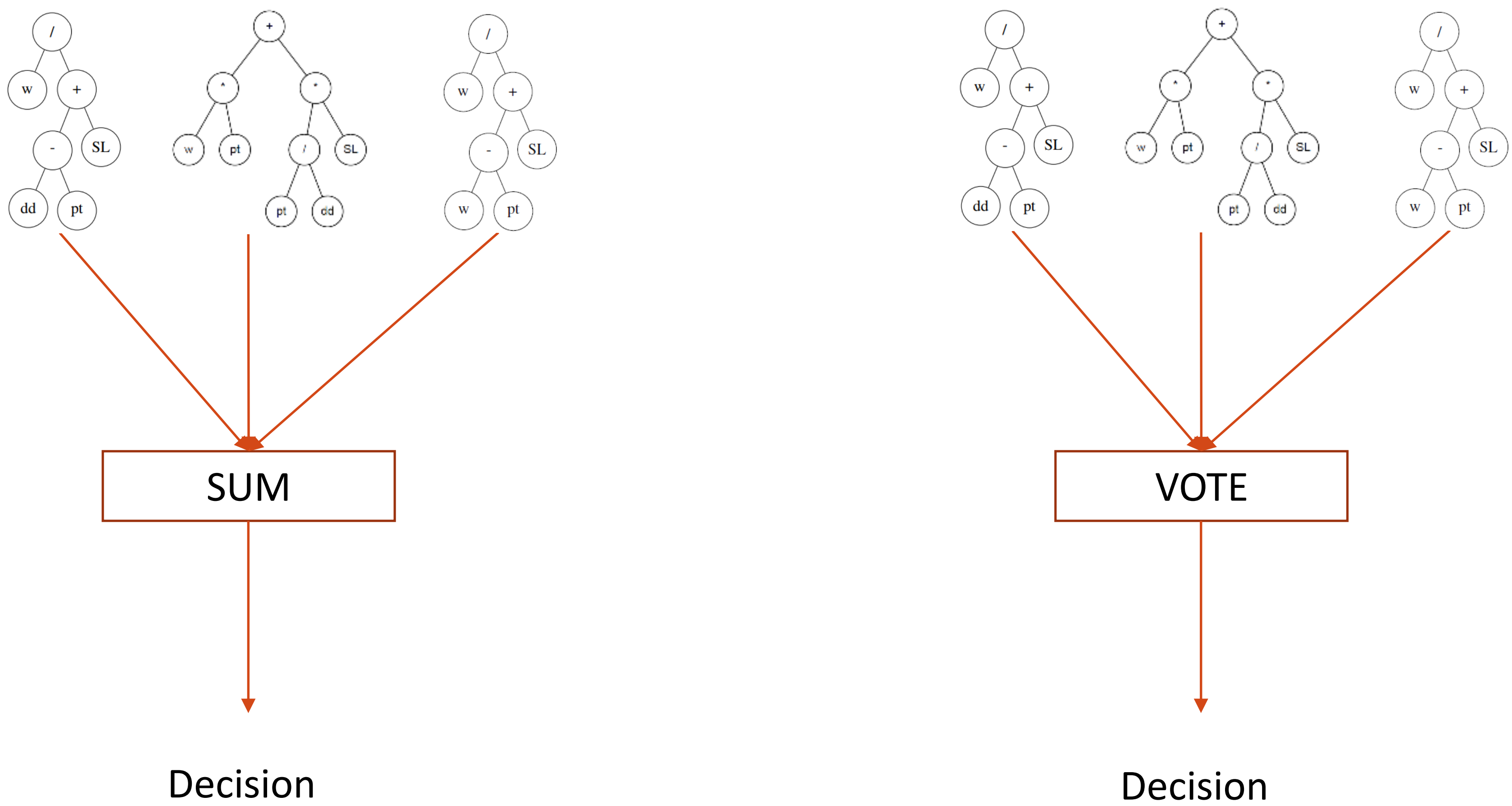


DRs with ensemble learning

- Designed DRs may make suboptimal decisions in certain situations, although they generally perform well

- Using several DRs to perform the decision could possibly lead to better results

DRs with ensemble learning



DRs with ensemble learning

- Various methods
  - Existing: BoostGP [20], BagGP [20], cooperative coevolution [18, 20]
  - New: SEC [20], Nelli-GP [19]

- Efficiency tested in the job shop and unrelated machines environment

- Tests performed for various criteria

DRs with ensemble learning

Criterion	Manually designed DR	Automatically designed DR	Ensemble of DRs	Improvement over manually designed DR	Improvement over atomatically designed DR
Cmax	38.44	38.23	38.20	0.6%	0%
Ft	159.6	158.1	157.6	1.3%	0.3%
Nwt	8.148	7.674	7.505	**7.9%**	**2.2%**
Twt	16.63	15.23	14.81	**11%**	**2.8%**

DRs with ensemble learning

DRs with ensemble learning

- Ensembles show great potential in improving the performance of DRs

- Generated ensembles are less dispersed than individual DRs
  - Smaller standard deviation between the results
  - Better distribution of the results
  - Better results are obtained more often than with GP

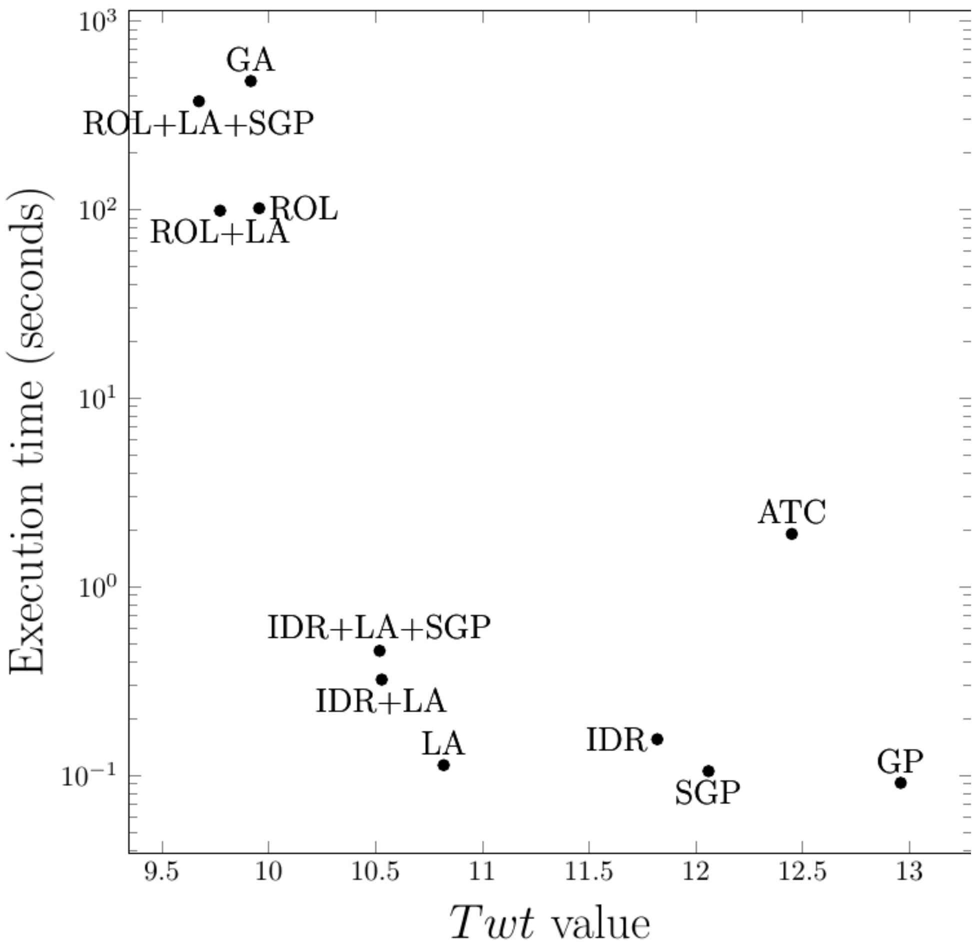
Static scheduling

- Adaptation of DRs to static scheduling conditions

- Various methods of adaptation:
  - Static terminal nodes
  - Iterative DRs [27]
  - Look-ahead [27]
  - Rollout algorithm

Static scheduling

- Different trade-offs between execution speed and schedule quality
- More complex methods are competitive with GA
- Benefit:
  - Generating schedules incrementally



Other important topics

- Due date assignment rules [22, 23, 24, 25, 26]
- Order acceptance and scheduling [28, 29, 30, 31, 32]
- Applications for different problems and criteria

Open issues and topics?

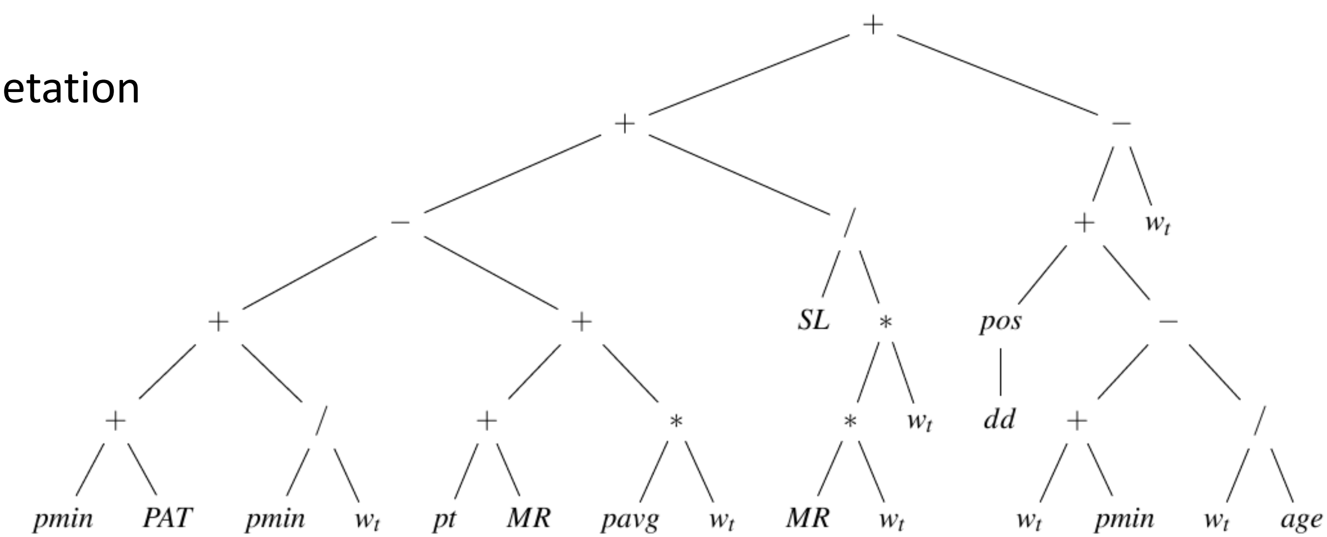
- Still, many open issues and topics:
  - Interpretability
  - Stochastic scheduling
  - Evolution speed
  - Selecting the appropriate DRs
  - Real world application

Interpretability

- Generated rules tend to be complex
- Quite often contain noise and redundant elements
- Difficult to interpret them and gain knowledge on how they work

Interpretability

Tree represnetation



Arithmetic representation:

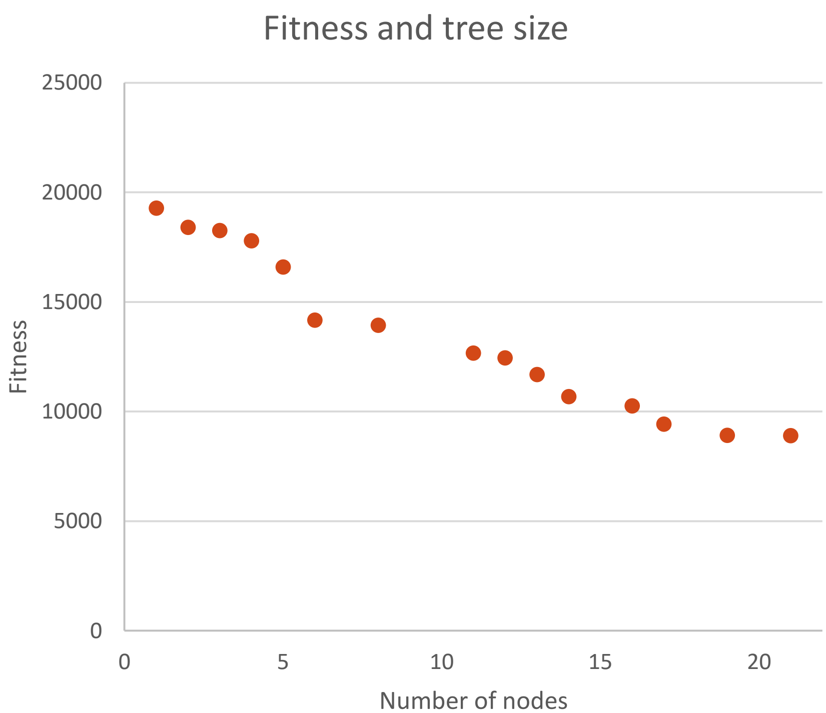
$$\boxed{pmin} + PAT + \frac{pmin}{w_t} - (pt + MR \boxed{+ pavg * w_t}) + \frac{SL}{MR * w_t^2} + dd \boxed{+ w_t + pmin - \frac{w_t}{age} - w_t}$$

Arithmetic representation (after simplification):

$$PAT + \frac{pmin}{w_T} - pt - MR + \frac{SL}{MR * w_T^2} + dd$$

Interpretability

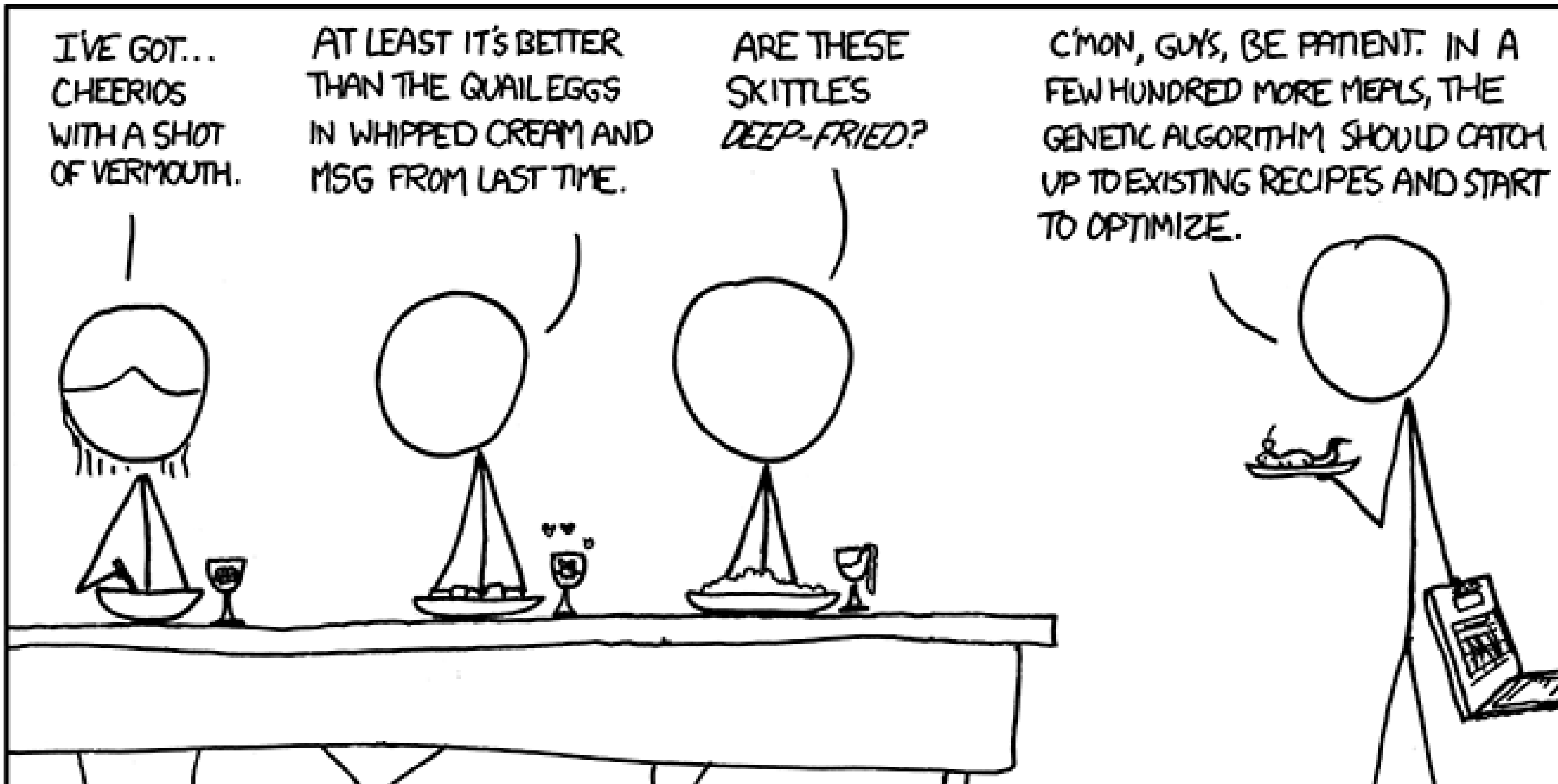
- Priority function (minimisation) size has an influence on its performance [33]



Interpretability

- Use methods for expression simplification
- Use different GP methods to improve interpretability
  - Dimensionally aware GP [34]
  - Methods which generate smaller expressions
- Further analyse the correlation between fitness and expression size

Evolution speed



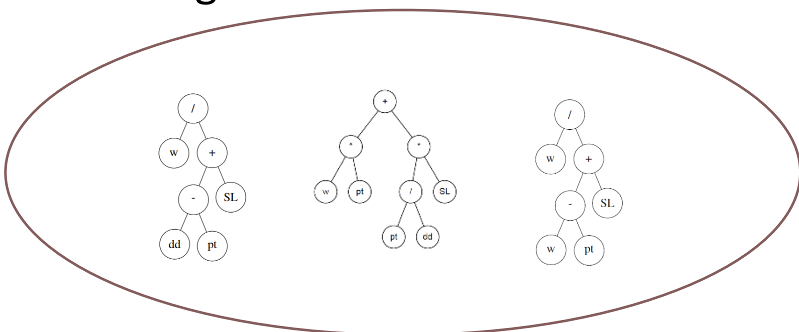
WE'VE DECIDED TO DROP THE CS DEPARTMENT FROM OUR WEEKLY DINNER PARTY HOSTING ROTATION.

Evolution speed

- Learning is performed offline
- A significant amount of time required for generating new DRs [35]
- Most time spent on evaluation of individuals
  - Due to their evaluation on many problems
- Surrogate models [36]

Evolution speed

During evolution

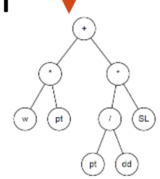


Evaluate

Simple problem instances

Problem instance	p_0	r_0	dd_0	...	p_m	r_m	dd_m
0	54	78	104	...	55	41	103
1	75	37	192	...	24	12	74
2	10	25	55	...	88	17	150
3	12	18	67	...	27	10	55
4	11	7	35	...	75	19	112
5	77	14	115	...	47	27	78

End of evolution

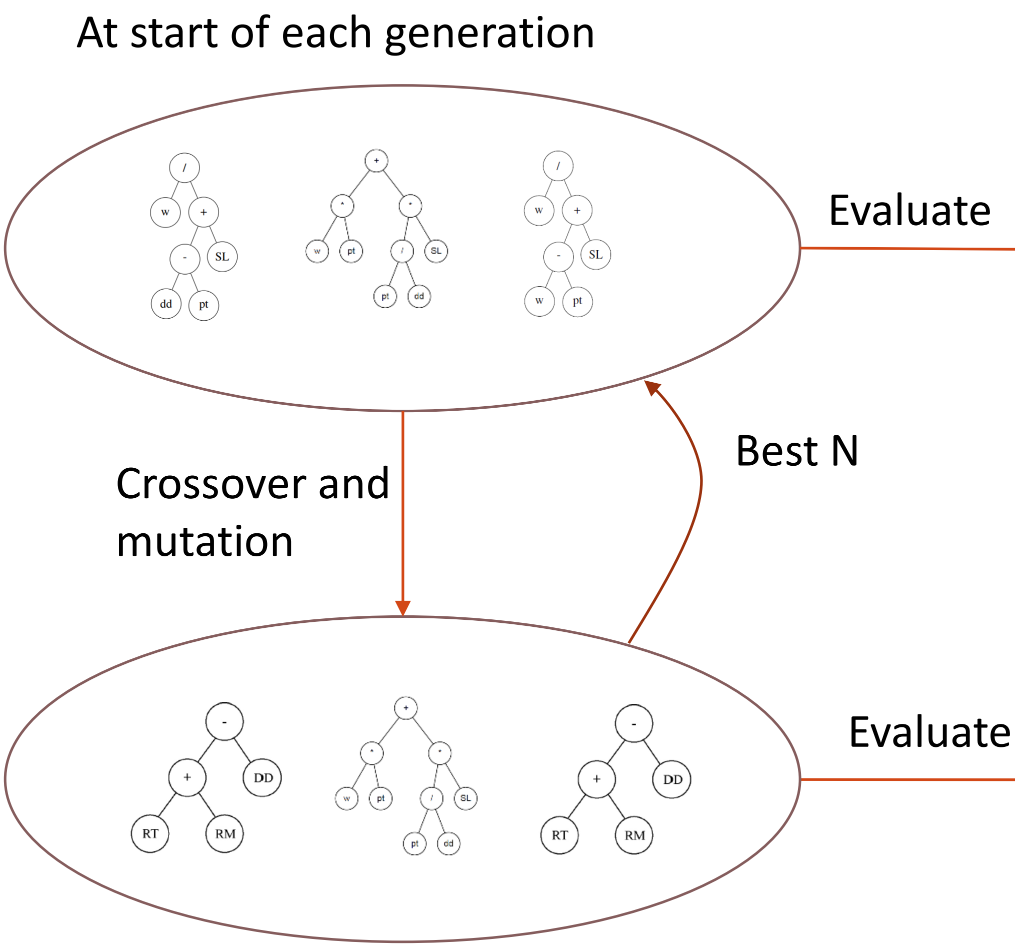


Solve

Complex problem instances

Problem instance	p_0	r_0	dd_0	...	p_n	r_n	dd_n
0	15	15	150	...	12	35	45
1	98	104	170	...	78	64	5
2	1	43	79	...	66	89	9
3	25	76	151	...	55	31	24
4	47	96	137	...	85	75	47
5	31	70	255	...	13	82	19
6	59	84	173	...	22	93	92
7	42	78	130	...	78	24	62
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
k	13	24	60	...	88	35	178

Evolution speed



Complex problem instances

Problem instance	p_0	r_0	dd_0	...	p_n	r_n	dd_n
0	15	15	150	...	12	35	45
1	98	104	170	...	78	64	5
2	1	43	79	...	66	89	9
3	25	76	151	...	55	31	24
4	47	96	137	...	85	75	47
5	31	70	255	...	13	82	19
6	59	84	173	...	22	93	92
7	42	78	130	...	78	24	62
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
k	13	24	60	...	88	35	178

Simple problem instances

Problem instance	p_0	r_0	dd_0	...	p_m	r_m	dd_m
0	54	78	104	...	55	41	103
1	75	37	192	...	24	12	74
2	10	25	55	...	88	17	150
3	12	18	67	...	27	10	55
4	11	7	35	...	75	19	112
5	77	14	115	...	47	27	78

Evolution speed

- Surrogate models achieved a better performance than standard GP
- The execution times are still similar to GP
- Important that the smaller problem instances have similar characteristics as the larger

Stochastic scheduling

- Most research focused on deterministic scheduling
- Information like processing times, release times, due dates, set-up times might not be completely accurate
- DRs should be able to deal with such problems

Stochastic scheduling

- Uncertainties of processing times were considered [37, 38, 39]
- Uncertainty of processing time was incorporated into terminal nodes and evolution
  - Using true processing times
  - Using processing times with uncertainty
- GPs were able to generate good DR
- Still room for improvement

Scheduling constraints

- Several research projects on individual constraints:
  - Breakdowns [40, 41]
  - Set-up times [42]
  - Precedence constraints [42]

- No research on larger combinations of the aforementioned constraints

Selecting appropriate DRs

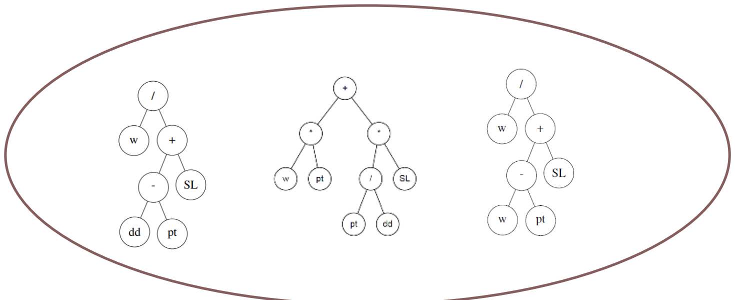
- Many DRs can be generated, but not all will work on all problem instances

- A lot of research with manually designed DRs [43, 44]

- Can we somehow learn which DR is suitable for which situations?
  - Is this even required with automatically designed DRs

Selecting appropriate DRs

Automatically generated DRs



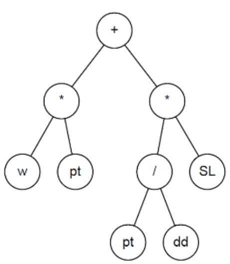
Problem instances for learning

Problem instance	p_0	r_0	dd_0	...	p_n	r_n	dd_n
0	15	15	150	...	12	35	45
1	98	104	170	...	78	64	5
2	1	43	79	...	66	89	9
3	25	76	151	...	55	31	24
4	47	96	137	...	85	75	47
5	31	70	255	...	13	82	19
6	59	84	173	...	22	93	92
7	42	78	130	...	78	24	62
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
k	13	24	60	...	88	35	178

Learn

Classifier
(k-nn, ANN, Bayes, C4.5, etc)

Appropriate DR



New problem instance

p_0	r_0	dd_0	...	p_m	r_m	dd_m
54	78	104	...	55	41	103

Selecting appropriate DRs

Experiment number	*Twt* value on the validation set	Improvement on the validation set	*Twt* value on the test set	Improvement on the test set
1	4.901	12.92%	5.450	10.70%
2	5.009	11.00%	5.395	11.60%
3	5.177	8.01%	5.545	9.14%
4	4.975	11.60%	5.292	13.29%
5	5.328	5.33%	5.895	3.41%
Manually selected DR	5.628	-	6.103	-

Selecting appropriate DRs

- Testing more problem features
- Automatising the process
  - Evolving the rules and the classifier simultaneously
  - Automatically determine when DRs need to be switched
  - Use more information available during the execution
- This approach would lead to a more automatised scheduling system

Application on real problems

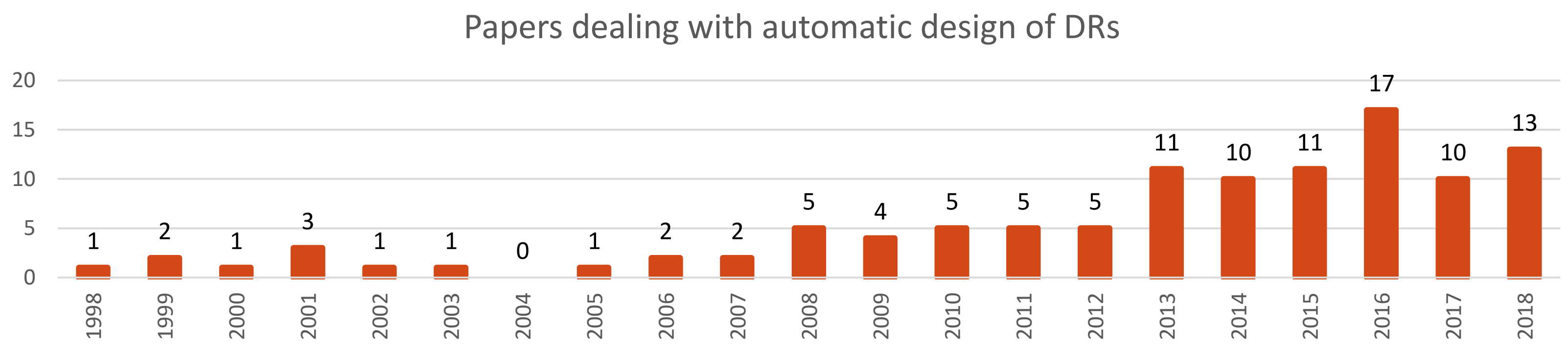
- Most research is performed on synthetic problems
- Test the methods in real environments and real problems
- Create a base of problems for researchers to use to make results comparable (OR library)

Conclusion and future outlook

- State of the art results of automatically generated DRs significantly better than that of manual DRs
- Automatically generating DRs has shown immense potential for various environments and conditions
- Is there still room for further research?

Conclusion and future outlook

- Many results were obtained only recently



- Many topics are still open or not researched thoroughly

- Potential of applying the obtained results for other problems:
  - Vehicle routing problem?

THE END!

- Thank you for your attention
- Questions?
- Discussion?
- Feel free to contact me at: marko.durasevic@fer.hr

General Co-Chairs
Mengjie ZHANG and Kay Chen TAN

Te Papa (New Zealand's National Museum)

IEEE Congress on Evolutionary Computation

10 - 13 June 2019, Wellington, New Zealand

IEEE Computational Intelligence Society

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

Wellington
Absolutely Positively

Business Events Wellington
WellingtonNZ.com

References

[1] Pinedo, M. L., Scheduling: Theory, algorithms, and systems: Fourth edition.

Boston, MA: Springer US, 2012

[2] Rocha, P. L., Ravetti, M. G., Mateus, G. R., Pardalos, P. M., “Exact algorithms for a scheduling problem with unrelated parallel machines and sequence and machinedependent setup times”, Computers & Operations Research, Vol. 35, No. 4, Apr. 2008, pp. 1250–1264.

[3] Graham, R. L., Lawler, E. L., Lenstra, J. K., Kan, A. H. G. R., “Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey”, Annals of Discrete Mathematics, Vol. 5, No. C, 1979, pp. 287–326.

[4] Hart, E., Ross, P., Corne, D., “Evolutionary Scheduling: A Review”, Genetic Programming and Evolvable Machines, Vol. 6, No. 2, Jun. 2005, pp. 191–220, available at: http://link.springer.com/10.1007/s10710-005-7580-7

[5] Đurasevic, M., Jakobovic, D. “A survey of dispatching rules for the dynamic unrelated machines environment”, Expert Systems with Applications, 2018 113 , 555 – 569. URL: http://www.sciencedirect.com/science/article/pii/S0957417418304159. doi:https://doi.org/10.1016/j.eswa.2018.06.053.

[6] Poli, R., Langdon, W. B., McPhee, N. F., A field guide to genetic programming. Published via http://lulu.com and freely available at http://www.gp-fieldguide.org.uk, 2008, (With contributions by J. R. Koza), available at: http: //www.gp-field-guide.org.uk

[7] Miyashita, K., “Job-shop scheduling with genetic programming”, in Proceedings of the 2Nd Annual Conference on Genetic and Evolutionary Computation, ser. GECCO’00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 505–512, available at: http://dl.acm.org/citation.cfm?id=2933718.2933809

[8] Jakobovic, D., Budin, L., “Dynamic scheduling with genetic programming”, in Genetic ́ Programming: 9th European Conference, EuroGP 2006, Budapest, Hungary, April 10-12, 2006. Proceedings, Collet, P., Tomassini, M., Ebner, M., Gustafson, S., Ekárt, A., (ed.). Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 73–84, available at: https://doi.org/10.1007/11729976_7

References

[9] Tay, J. C., Ho, N. B., “Designing Dispatching Rules to Minimize Total Tardiness”, in Evolutionary Scheduling, Dahal, K. P., Tan, K. C., Cowling, P. I., (ed.). Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 101–124, available at: http://link.springer.com/10.1007/978-3-540-48584-1_4

[10] Ðurasevic, M., Jakobovi ´ c, D., Kneževi ´ c, K., “Adaptive scheduling on unrelated ´ machines with genetic programming”, Applied Soft Computing, Vol. 48, Nov. 2016, pp. 419–430, available at: http://linkinghub.elsevier.com/retrieve/pii/S1568494616303519

[11] Branke, J., Hildebrandt, T., Scholz-Reiter, B., “Hyper-heuristic Evolution of Dispatching Rules: A Comparison of Rule Representations”, Evolutionary Computation, Vol. 23, No. 2, Jun. 2015, pp. 249–277, available at: http://www.mitpressjournals.org/doi/10. 1162/EVCO_a_00131

[12] Nguyen, S., Zhang, M., Johnston, M., Tan, K. C., “A Computational Study of Representations in Genetic Programming to Evolve Dispatching Rules for the Job Shop Scheduling Problem”, IEEE Transactions on Evolutionary Computation, Vol. 17, No. 5, Oct. 2013, pp. 621–639, available at: http://ieeexplore.ieee.org/document/6353198/

[13] Tay, J. C., Ho, N. B., “Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems”, Computers & Industrial 413 Bibliography Engineering, Vol. 54, No. 3, Apr. 2008, pp. 453–473, available at: http: //linkinghub.elsevier.com/retrieve/pii/S0360835207002008

[14] Nguyen, S., Zhang, M., Johnston, M., Tan, K. C., “Dynamic multi-objective job shop scheduling: A genetic programming approach”, in Automated Scheduling and Planning: From Theory to Practice, Uyar, A. S., Ozcan, E., Urquhart, N., (ed.). Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 251–282, available at: https://doi.org/10.1007/978-3-642-39304-4_10

[15] Nguyen, S., Zhang, M., Tan, K. C., “Enhancing genetic programming based hyper-heuristics for dynamic multi-objective job shop scheduling problems”, in 2015 IEEE Congress on Evolutionary Computation (CEC). IEEE, May 2015, pp. 2781–2788, available at: http://ieeexplore.ieee.org/document/7257234/

References

[16] Đurasevic, M., Jakobovic, D. „Evolving dispatching rules for optimising manyobjective criteria in the unrelated machines environment“, Genetic Programming and Evolvable Machines (2017). DOI 10.1007/s10710-017-9310-3. URL https://doi.org/10.1007/s10710-017-9310-3

[17] Karunakaran, D., Chen, G., Zhang, M., "Parallel Multi-objective Job Shop Scheduling Using Genetic Programming", in Artificial Life and Computational Intelligence: Second Australasian Conference, ACALCI 2016, Canberra, ACT, Australia, February 2-5, 2016, Proceedings, Ray, T., Sarker, R., Li, X., (ed.). Springer International Publishing, 2016, pp. 234–245, available at: http://link.springer.com/10.1007/978-3-319-28270-1_20

[18] Park, J., Nguyen, S., Zhang, M., Johnston, M., "Evolving ensembles of dispatching rules using genetic programming for job shop scheduling", in Genetic Programming: 18th European Conference, EuroGP 2015, Copenhagen, Denmark, April 8-10, 2015, Proceedings, Machado, P., Heywood, M. I., McDermott, J., Castelli, M., GarcíaSánchez, P., Burelli, P., Risi, S., Sim, K., (ed.). Cham: Springer International Publishing, 2015, pp. 92–104, available at: https://doi.org/10.1007/978-3-319-16501-1_8

[19] Hart, E., Sim, K., "A Hyper-Heuristic Ensemble Method for Static Job-Shop Scheduling", Evolutionary Computation, Vol. 24, No. 4, Dec. 2016, pp. 609–635, available at: http://www.mitpressjournals.org/doi/10.1162/EVCO_a_00183

[20] Đurasevic, M., Jakobovic, D. "Comparison of ensemble learning methods for creating ensembles of dispatching rules for the unrelated machines environment". Genetic Programming and Evolvable Machines pp. 1–40 (2017). DOI 10.1007/s10710-017-9302-3

[21] Park, J., Mei, Y., Nguyen, S., Chen, G., Zhang, M., "An investigation of ensemble combination schemes for genetic programming based hyper-heuristic approaches to dynamic job shop scheduling", Applied Soft Computing 63, 72 – 86 (2018). DOI https://doi.org/10.1016/j.asoc.2017.11.020. URL http://www.sciencedirect.com/science/article/pii/S156849461730683X

[22] Baykasoglu, A., Gocken, M., "Gene expression programming based due date assignment in a simulated job shop", Expert Systems with Applications, Vol. 36, No. 10, Dec. 2009, pp. 12 143–12 150, available at: http://linkinghub.elsevier.com/retrieve/pii/ S095741740900311X

References

[23] Nguyen, S., Zhang, M., Johnston, M., Tan, K. C., "Evolving reusable operation-based due-date assignment models for job shop scheduling with genetic programming", in Genetic Programming: 15th European Conference, EuroGP 2012, Málaga, Spain, April 11-13, 2012. Proceedings, Moraglio, A., Silva, S., Krawiec, K., Machado, P., Cotta, C., (ed.). Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 121–133, available at: https://doi.org/10.1007/978-3-642-29139-5_11

[24] Nguyen, S., Zhang, M., Johnston, M., Tan, K. C., "Genetic Programming for Evolving Due-Date Assignment Models in Job Shop Environments", Evolutionary Computation, Vol. 22, No. 1, Mar. 2014, pp. 105–138, available at: http: //www.mitpressjournals.org/doi/10.1162/EVCO_a_00105

[25] Nguyen, S., Zhang, M., Johnston, M., Tan, K. C., "A coevolution genetic programming method to evolve scheduling policies for dynamic multi-objective job shop scheduling problems", in 2012 IEEE Congress on Evolutionary Computation. IEEE, Jun. 2012, pp. 1–8, available at: http://ieeexplore.ieee.org/document/6252968/

[26] Nguyen, S., Zhang, M., Johnston, M., Tan, K. C., "Automatic Design of Scheduling Policies for Dynamic Multi-objective Job Shop Scheduling via Cooperative Coevolution Genetic Programming", IEEE Transactions on Evolutionary Computation, Vol. 18, No. 2, Apr. 2014, pp. 193–208, available at: http://ieeexplore.ieee.org/document/6468087/

[27] Nguyen, S., Zhang, M., Johnston, M., Tan, K. C., "Automatic Programming via Iterated Local Search for Dynamic Job Shop Scheduling", IEEE Transactions on Cybernetics, Vol. 45, No. 1, Jan. 2015, pp. 1–14, available at: http: //ieeexplore.ieee.org/document/6807725/

[28] Nguyen, S., Zhang, M., Johnston, M., Tan, K. C., "Learning Reusable Initial Solutions for Multi-objective Order Acceptance and Scheduling Problems with Genetic Programming", in Genetic Programming: 16th European Conference, EuroGP 2013, Vienna, Austria, April 3-5, 2013. Proceedings, Krawiec, K., Moraglio, A., Hu, T., Etaner-Uyar, A. ¸S., Hu, B., (ed.). Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 157–168, available at: http://link.springer.com/10.1007/978-3-642-37207-0_14

References

[29] Park, J., Nguyen, S., Johnston, M., Zhang, M., “Evolving Stochastic Dispatching Rules for Order Acceptance and Scheduling via Genetic Programming”, in AI 2013: Advances in Artificial Intelligence: 26th Australasian Joint Conference, Dunedin, New Zealand, December 1-6, 2013. Proceedings, Cranefield, S., Nayak, A., (ed.). Cham: Springer International Publishing, 2013, pp. 478–489, available at: http://link.springer.com/10.1007/978-3-319-03680-9_48

[30] Nguyen, S., Zhang, M., Johnston, M., “A sequential genetic programming method to learn forward construction heuristics for order acceptance and scheduling”, in 2014 IEEE Congress on Evolutionary Computation (CEC). IEEE, Jul. 2014, pp. 1824–1831, available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6900347

[31] Nguyen, S., “A learning and optimizing system for order acceptance and scheduling”, The International Journal of Advanced Manufacturing Technology, Vol. 86, No. 5-8, Sep. 2016, pp. 2021–2036, available at: http://link.springer.com/10.1007/s00170-015-8321-6

[32] Nguyen, S., Zhang, M., Johnston, M., “Enhancing Branch-and-Bound Algorithms for Order Acceptance and Scheduling with Genetic Programming”, in Genetic Programming: 17th European Conference, EuroGP 2014, Granada, Spain, April 23-25, 2014, Revised Selected Papers, Nicolau, M., Krawiec, K., Heywood, M. I., Castelli, M., García-Sánchez, P., Merelo, J. J., Rivas Santos, V. M., Sim, K., (ed.). Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 124–136, available at: http://link.springer.com/10.1007/978-3-662-44303-3_11

[33] Pitzer, E., Beham, A., Affenzeller, M., Heiss, H., Vorderwinkler, M., “Production fine planning using a solution archive of priority rules”, in 3rd IEEE International Symposium on Logistics and Industrial Informatics. IEEE, Aug. 2011, pp. 111–116, available at: http://ieeexplore.ieee.org/document/6031130/

[34] Yi Mei, Su Nguyen, Mengjie Zhang, "Constrained Dimensionally Aware Genetic Programming for Evolving Interpretable Dispatching Rules in Dynamic Job Shop Scheduling," Simulated Evolution and Learning (SEAL), LNCS vol. 10593, pp. 435-447, 2017, Springer.

References

[29] Park, J., Nguyen, S., Johnston, M., Zhang, M., “Evolving Stochastic Dispatching Rules for Order Acceptance and Scheduling via Genetic Programming”, in AI 2013: Advances in Artificial Intelligence: 26th Australasian Joint Conference, Dunedin, New Zealand, December 1-6, 2013. Proceedings, Cranefield, S., Nayak, A., (ed.). Cham: Springer International Publishing, 2013, pp. 478–489, available at: http://link.springer.com/10.1007/978-3-319-03680-9_48

[30] Nguyen, S., Zhang, M., Johnston, M., “A sequential genetic programming method to learn forward construction heuristics for order acceptance and scheduling”, in 2014 IEEE Congress on Evolutionary Computation (CEC). IEEE, Jul. 2014, pp. 1824–1831, available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6900347

[31] Nguyen, S., “A learning and optimizing system for order acceptance and scheduling”, The International Journal of Advanced Manufacturing Technology, Vol. 86, No. 5-8, Sep. 2016, pp. 2021–2036, available at: http://link.springer.com/10.1007/s00170-015-8321-6

[32] Nguyen, S., Zhang, M., Johnston, M., “Enhancing Branch-and-Bound Algorithms for Order Acceptance and Scheduling with Genetic Programming”, in Genetic Programming: 17th European Conference, EuroGP 2014, Granada, Spain, April 23-25, 2014, Revised Selected Papers, Nicolau, M., Krawiec, K., Heywood, M. I., Castelli, M., García-Sánchez, P., Merelo, J. J., Rivas Santos, V. M., Sim, K., (ed.). Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 124–136, available at: http://link.springer.com/10.1007/978-3-662-44303-3_11

[33] Pitzer, E., Beham, A., Affenzeller, M., Heiss, H., Vorderwinkler, M., “Production fine planning using a solution archive of priority rules”, in 3rd IEEE International Symposium on Logistics and Industrial Informatics. IEEE, Aug. 2011, pp. 111–116, available at: http://ieeexplore.ieee.org/document/6031130/

[34] Yi Mei, Su Nguyen, Mengjie Zhang, "Constrained Dimensionally Aware Genetic Programming for Evolving Interpretable Dispatching Rules in Dynamic Job Shop Scheduling," Simulated Evolution and Learning (SEAL), LNCS vol. 10593, pp. 435-447, 2017, Springer.

References

[35] Mei, Y., Zhang, M., “A comprehensive analysis on reusability of GP-evolved job shop dispatching rules”, in 2016 IEEE Congress on Evolutionary Computation (CEC). IEEE, Jul. 2016, pp. 3590–3597, available at: http://ieeexplore.ieee.org/document/7744244/

[36] Nguyen, S., Zhang, M., Tan, K. C., “Surrogate-Assisted Genetic Programming With Simplified Models for Automated Design of Dispatching Rules”, IEEE Transactions on Cybernetics, 2016, pp. 1–15, available at: http://ieeexplore.ieee.org/document/7473913/

[37] Karunakaran, D., Mei, Y., Chen, G., Zhang, M., “Dynamic Job Shop Scheduling Under Uncertainty Using Genetic Programming”, in Intelligent and Evolutionary Systems: The 20th Asia Pacific Symposium, IES 2016, Canberra, Australia, November 2016, Proceedings, Leu, G., Singh, H. K., Elsayed, S., (ed.). Cham: Springer International Publishing, 2017, pp. 195–210, available at: http://link.springer.com/10.1007/978-3-319-49049-6_14

[38] Karunakaran, D., Yi Mei, Gang Chen, Mengjie Zhang, “Evolving dispatching rules for dynamic Job shop scheduling with uncertain processing times”, in 2017 IEEE Congress on Evolutionary Computation (CEC). IEEE, Jun. 2017, pp. 364–371, available at: http://ieeexplore.ieee.org/document/7969335/

[39] Karunakaran, D., Mei, Y., Chen, G., Zhang, M., “Toward evolving dispatching rules for dynamic job shop scheduling under uncertainty”, in Proceedings of the Genetic and Evolutionary Computation Conference on - GECCO ’17. New York, New York, USA: ACM Press, 2017, pp. 282–289, available at: http: //dl.acm.org/citation.cfm?doid=3071178.3071202

[40] Wen-Jun Yin, Min Liu, Cheng Wu, “Learning single-machine scheduling heuristics subject to machine breakdowns with genetic programming”, in The 2003 Congress on Evolutionary Computation, 2003. CEC ’03., Vol. 2. IEEE, 2003, pp. 1050–1055, available at: http://ieeexplore.ieee.org/document/1299784/

References

- [41] Park, J., Mei, Y., Nguyen, S., Chen, G., Zhang, M., “Investigating the generality of genetic programming based hyper-heuristic approach to dynamic job shop scheduling with machine breakdown”, in Artificial Life and Computational Intelligence: Third Australasian Conference, ACALCI 2017, Geelong, VIC, Australia, January 31 – February 2, 2017, Proceedings, Wagner, M., Li, X., Hendtlass, T., (ed.). Cham: Springer International Publishing, 2017, pp. 301–313, available at: https://doi.org/10.1007/978-3-319-51691-2_26
- [42] Jakobovic, D., Marasović, K., “Evolving priority scheduling heuristics with genetic ´ programming”, Applied Soft Computing, Vol. 12, No. 9, Sep. 2012, pp. 2781–2789, available at: http://linkinghub.elsevier.com/retrieve/pii/S1568494612001780
- [43] Mouelhi-Chibani, W., Pierreval, H., “Training a neural network to select dispatching rules in real time”, Computers & Industrial Engineering, Vol. 58, No. 2, Mar. 2010, pp. 249–256, available at: http://linkinghub.elsevier.com/retrieve/pii/S0360835209000953
- [44] Heger, J., Hildebrandt, T., Scholz-Reiter, B., “Dispatching rule selection with Gaussian processes”, Central European Journal of Operations Research, Vol. 23, No. 1, Mar. 2015, pp. 235–249, available at: http://link.springer.com/10.1007/s10100-013-0322-7