The validation stage used two optimization strategies crucial to this study. The first strategy is called *just_time* where the optimization process is based only on the time objective, and therefore the objective function only evaluates the solutions with respect to the makespan variable. While the second optimization strategy is based on the Pareto Optimal (*pareto*) approach and is oriented simultaneously on the time (makespan) and cost objectives, in this case, the optimization of plans based on equation (3.1) was used.

For a fair comparison, the following common parameters are established for all the algorithms:

(i) The initial population was built randomly with a size of 100 individuals.
(ii) Thirty percent of the individuals were selected for the generation of the new population.
(iii) Elitism was applied.
(iv) In the case of the GA, the crossover probability is 0.8, while the mutation probability is 0.2.
(v) The stop condition was to reach 100 generations.

The experiments have been performed by running each algorithm 20 times for every instance of the four datasets. The results of the algorithms were evaluated with respect to the following variables:

- *Mean_Makespan:* average makespan considering the 20 runs for each dataset instance.
- *StdDev_Makespan:* standard deviation to the optimum value over the 20 runs for each dataset instance.
- *%Optimum:* percent of times where the algorithm finds the optimal makespan of the dataset over the 20 runs.
- *Execution_time:* average time used by the algorithm to execute 100 iterations.

In order to compare the algorithms, authors used *SPSS version 25* and the *Wilcoxon's non-parametric test* for two samples related, with 95% of confidence interval and 0.05 significance level.

In the comparison, the groups of algorithms were organized according to the quality of the results, that means "Group A results" > "Group B results" > "Group C results" > "Group D results". The algorithms in the same group did not have significant differences between them.

Table 1, 2, 9 and 10 show the descriptive analysis and the results of comparisons using Wilcoxon test for the *Mean_Makespan* variable.

For *StdDev_Makespan* variable, comparison results are summarized in Table 3, 4, 11 and 12. Findings for the *%Optimum* variable are presented in Table 5, 6, 13 and 14.

Finally, Tables 7, 8, 15 and 16 show the same statistical analysis for the *Execution_time* variable.

Table 1. Descriptive analysis, variable *Mean_Makespan* for j30_17 dataset

Algorithm	N Statistic	Minimum Statistic	Maximum Statistic	Mean Statistic	Std. Error	Std. Deviation Statistic
CL_{FDA}_just_time	10	26.20	39.15	31.4150	1.21262	3.83464
Reported_Bibliography	10	26.00	39.00	31.7000	1.24766	3.94546
UMDA_just_time	10	26.40	39.05	31.8100	1.18528	3.74817
CL_{FDA}_pareto	10	28.65	39.85	33.2450	1.00657	3.18307
UMDA _pareto	10	29.10	40.55	33.5300	1.05998	3.35196
GA_just_time	10	28.10	39.45	33.8300	1.02746	3.24912
GA_pareto	10	29.65	40.40	35.0850	0.93634	2.96095

Table 2. Comparison results using Wilcoxon test, variable *Mean_Makespan* for j30_17 dataset

Algorithm1	Algorithm2	Signification *(p-value)*	Groups
CL_{FDA}_just_time	Reported_Bibliography	0.33	*Group 1:* CL_{FDA}_just_time, Reported_ Bibliography
CL_{FDA}_just_time	UMDA_just_time	**0.014**	*Group 2:* UMDA_just_time
UMDA_just_time	CL_{FDA}_pareto	**0.037**	*Group 3:* CL_{FDA}_pareto, UMDA_pareto, GA_just_time
CL_{FDA}_pareto	UMDA_pareto	0.240	*Group 4:* GA_pareto
CL_{FDA}_pareto	GA_just_time	0.374	
CL_{FDA}_pareto	GA_pareto	**0.005**	

Regarding variable *Mean_Makespan* in j30_17 dataset, there were not significant differences between CL_{FDA}*_just_time* and results reported in the bibliography. The best results were obtained by CL_{FDA}*_just_time,* whereas the worst results were obtained by *GA_pareto.*

Table 3. Descriptive analysis, variable *StdDev_Makespan* for j30_17 dataset

Algorithm	N Statistic	Minimum Statistic	Maximum Statistic	Mean Statistic	Std. Error	Std. Deviation Statistic
Reported_Bibliography	10	0.00	4.00	1.3000	0.36667	1.15950
CL$_{FDA}$_just_time	10	0.39	3.27	1.4998	0.30655	0.96940
UMDA_just_time	10	0.22	3.13	1.8599	0.28301	0.89495
CL$_{FDA}$_pareto	10	1.16	6.36	3.2956	0.44275	1.40008
UMDA _pareto	10	1.66	6.59	3.5752	0.45077	1.42547
GA_just_time	10	0.81	5.88	3.7808	0.48026	1.51872
GA_pareto	10	1.76	7.34	5.1077	0.49409	1.56244

Table 4. Comparison results using Wilcoxon test, variable *StdDev_Makespan* for j30_17 dataset

Algorithm1	Algorithm2	Signification (*p-value*)	Groups
Reported_Bibliography	CL$_{FDA}$_just_time	0.646	
Reported_Bibliography	UMDA_just_time	**0.037**	*Group 1:* Reported_ Bibliography, CL$_{FDA}$_just_time
UMDA_just_time	CL$_{FDA}$_pareto	**0.005**	*Group 2:* UMDA_just_time
CL$_{FDA}$_pareto	UMDA_pareto	0.202	*Group 3:* CL$_{FDA}$_pareto, UMDA_pareto, GA_just_time
CL$_{FDA}$_pareto	GA_just_time	0.241	*Group 4:* GA_pareto
CL$_{FDA}$_pareto	GA_pareto	**0.005**	

As regards variable *StdDev_Makespan* in dataset j30_17, there are not significant difference between *CL$_{FDA}$_just_time* and results reported in the bibliography. These algorithms reported the best results; whereas the worst results were obtained by *GA_pareto* algorithm.

Table 5. Descriptive analysis, variable *%Optimum* for j30_17 dataset

Algorithm	N Statistic	Minimum Statistic	Maximum Statistic	Mean Statistic	Std. Error	Std. Deviation Statistic
CL$_{FDA}$_just_time	10	15.00	85.00	56.0000	8.81287	27.86874
UMDA_just_time	10	5.00	95.00	37.5000	7.82624	24.74874
GA_just_time	10	0.00	65.00	10.0000	6.32456	20.00000
CL$_{FDA}$_pareto	10	0.00	40.00	6.0000	3.92994	12.42757
UMDA _pareto	10	0.00	15.00	3.5000	1.97906	6.25833
GA_pareto	10	0.00	25.00	2.5000	2.50000	7.90569

Table 6. Comparison results using Wilcoxon test, variable *%Optimum*, j30_17 dataset

Algorithm1	Algorithm2	Signification (*p-value*)	Groups
CL$_{FDA}$_just_time	UMDA_just_time	**0.028**	*Group 1*: CL$_{FDA}$_just_time
UMDA_just_time	GA_just_time	**0.005**	*Group 2*: UMDA_just_time
GA_just_time	CL$_{FDA}$_pareto	0.161	*Group 3*: GA_just_time, CL$_{FDA}$_pareto, UMDA_pareto
GA_just_time	UMDA_pareto	0.572	
GA_just_time	GA_pareto	**0.042**	*Group 4*: GA_pareto

In dataset j30_17, concerning variable *%Optimum,* the best results were obtained by the algorithm *CL$_{FDA}$_just_time* and the worst results were obtained by *GA_pareto* algorithm. In this dataset, *just_time* optimization strategy reports better results than *pareto* optimization.

Table 7. Descriptive analysis, variable *Execution_time*, j30_17 dataset

Algorithm	N Statistic	Minimum Statistic	Maximum Statistic	Mean Statistic	Std. Error	Std. Deviation Statistic
GA_just_time	10	16.55	20.42	18.7630	0.35353	1.11797
GA_pareto	10	16.92	20.95	19.2350	0.37585	1.18853
UMDA_just_time	10	24.02	29.05	26.9220	0.47009	1.48657
UMDA _pareto	10	26.88	33.48	29.4730	0.56849	1.79773
CL_{FDA}_just_time	10	37.49	42.91	40.2670	0.48567	1.53584
CL_{FDA}_pareto	10	39.98	46.25	42.4040	0.57165	1.80771

Table 8. Comparison results using Wilcoxon test, variable *Execution_time* (j30_17 dataset)

Algorithm1	Algorithm2	Signification (*p-value*)	Groups
GA_just_time	GA_pareto	**0.017**	*Group 1*: GA_just_time
GA_pareto	UMDA_just_time	**0.005**	*Group 2*: GA_pareto
UMDA_just_time	UMDA_pareto	**0.005**	*Group 3*: UMDA_just_time
UMDA_pareto	CL_{FDA}_just_time	**0.005**	*Group 4*: UMDA_pareto
CL_{FDA}_just_time	CL_{FDA}_pareto	**0.005**	*Group 5*: CL_{FDA}_just_time
			Group 6: CL_{FDA}_pareto,

In dataset J30_17, respect to variable *Execution_time,* the best results were obtained by the *Genetic Algorithms* approach. In particular, *GA_just_time* was the fastest algorithm; whereas *CL_{FDA}* were the highest time consume algorithms.

Table 9. Descriptive analysis, variable *Mean_Makespan* (c15_9, c15_10 and c15_12 datasets)

Algorithm	N Statistic	Minimum Statistic	Maximum Statistic	Mean Statistic	Std. Error	Std. Deviation Statistic
Reported_Bibliography	30	12.00	39.00	22.1333	1.23338	6.75550
CL_{FDA}_pareto	30	14.70	38.45	22.4333	1.06220	5.81789
CL_{FDA}_just_time	30	14.30	40.10	22.5567	1.05235	5.76397
UMDA _pareto	30	15.90	40.90	23.0517	1.09634	6.00489
UMDA_just_time	30	16.20	42.05	23.3283	1.10486	6.05156
GA_just_time	30	17.10	42.10	23.5250	1.06103	5.81147
GA_pareto	30	16.50	41.40	23.8117	1.05626	5.78540

Table 10. Comparison results using Wilcoxon test, variable *Mean_Makespan* (c15_9, c15_10 and c15_12 datasets)

Algorithm1	Algorithm2	Signification (p-value)	Groups
CL_{FDA}_just_time	Reported_Bibliography	0.33	
CL_{FDA}_just_time	UMDA_just_time	**0.014**	*Group 1*: CL_{FDA}_just_time, Reported_ Bibliography
UMDA_just_time	CL_{FDA}_pareto	**0.037**	*Group 2*: UMDA_just_time
CL_{FDA}_pareto	UMDA_pareto	0.240	*Group 3*: CL_{FDA}_pareto, UMDA_pareto, GA_just_time
CL_{FDA}_pareto	GA_just_time	0.374	*Group 4*: GA_pareto
CL_{FDA}_pareto	GA_pareto	**0.005**	

As for variable *Mean_Makespan* in dataset c15 with all 30 instances from c15_9, c15_10 and c15_12, there were not significant differences between *CL_{FDA}_just_time* and results reported in the bibliography. The best results were obtained by *CL_{FDA}_just_time and* the worst results were obtained by *GA_pareto.* EDA algorithms (*CL_{FDA}* and *UMDA*) using *just_time* strategy reported better results than the same algorithms with the *pareto* optimization strategy.

Table 11. Descriptive analysis, variable *StdDev_Makespan* (c15_9, c15_10 and c15_12 datasets)

Algorithm	N Statistic	Minimum Statistic	Maximum Statistic	Mean Statistic	Std. Error	Std. Deviation Statistic
Reported_Bibliography	30	0.00	8.00	1.2000	0.33356	1.82700
CL_{FDA}_pareto	30	0.00	4.73	1.4962	0.23377	1.28044
CL_{FDA}_just_time	30	0.00	4.24	1.6584	0.26565	1.45501
UMDA _pareto	30	0.00	5.27	2.0655	0.31213	1.70962
UMDA_just_time	30	0.00	6.32	2.3699	0.36208	1.98317
GA_just_time	30	0.00	6.24	2.7220	0.31062	1.70133
GA_pareto	30	0.00	6.48	2.8390	0.29116	1.59476

Table 12. Comparison results using Wilcoxon test, variable *Stddev_Makespan* (c15_9, c15_10 and c15_12 datasets)

Algorithm1	Algorithm2	Signification (p-value)	Groups
Reported_Bibliography	CL_{FDA}_pareto	0.13	*Group 1*: Reported_ Bibliography, CL_{FDA}_pareto, CL_{FDA}_just_time
Reported_Bibliography	CL_{FDA}_just_time	0.092	
Reported_Bibliography	UMDA_pareto	**0.016**	*Group 2*: UMDA_pareto
UMDA_pareto	UMDA_just_time	**0.014**	
UMDA_just_time	GA_just_time	**0.003**	*Group 3*: UMDA_just_time
GA_just_time	GA_pareto	0.846	*Group 4*: GA_just_time, GA_pareto

In these datasets, for variable *StdDev_Makespan*, the algorithms *Reported_ Bibliography, CL_{FDA}_pareto, CL_{FDA}_just_time* did not have significant differences, whereas the worst results were obtained by *GA* algorithms. In this case, differences between *just_time* and *pareto* optimization strategies were not found to be significant.

Table 13. Descriptive analysis variable *%Optimum* (c15_9, c15_10 and c15_12 datasets)

Algorithm	N Statistic	Minimum Statistic	Maximum Statistic	Mean Statistic	Std. Error	Std. Deviation Statistic
CL_{FDA}_pareto	30	0.00	100.00	52.6667	6.45379	35.34884
CL_{FDA}_just_time	30	0.00	100.00	50.0000	6.84391	37.48563
UMDA _pareto	30	0.00	100.00	41.3333	6.96846	38.16781
UMDA_just_time	30	0.00	100.00	38.0000	8.07508	44.22903
GA_just_time	30	0.00	100.00	24.3333	5.63616	30.87051
GA_pareto	30	0.00	100.00	17.8333	4.30639	23.58708

Table 14. Comparison results using Wilcoxon test, variable *%Optimum* (c15_9, c15_10 and c15_12 datasets)

Algorithm1	Algorithm2	Signification (p-value)	Groups
CL_{FDA}_pareto	CL_{FDA}_just_time	0.173	*Group 1*: CL_{FDA}_pareto, CL_{FDA}_just_time
CL_{FDA}_pareto	UMDA_pareto	**0.000**	*Group 2*: UMDA_ pareto, UMDA_ just_time
UMDA_pareto	UMDA_just_time	0.314	
UMDA_pareto	GA_just_time	**0.000**	*Group 3*: GA_just_time
GA_just_time	GA_pareto	**0.023**	*Group 4*: GA_pareto

In c15_9, c15_10 and c15_12 datasets and in relation to variable *%Optimum,* the best results were obtained by the algorithms *CL_{FDA}_pareto, CL_{FDA}_just_time,* whereas the worst results were obtained by *GA_pareto* algorithm. In this variable, as in *Mean_Makespan,* there were not significant differences between *just_time* and *pareto* strategies. The worst results were obtained by *GA* approach.

Table 15. Descriptive analysis variable *Execution_time* (c15_9, c15_10 and c15_12 datasets)

Algorithm	N Statistic	Minimum Statistic	Maximum Statistic	Mean Statistic	Std. Error	Std. Deviation Statistic
GA_pareto	30	7.29	12.22	9.0757	0.20911	1.14536
GA_just_time	30	7.56	12.79	9.2257	0.21902	1.19963
UMDA _pareto	30	10.37	16.57	12.4103	0.25598	1.40203
UMDA_just_time	30	10.35	17.47	12.4783	0.30701	1.68155
CL_{FDA}_just_time	30	16.08	24.12	18.9783	0.32382	1.77365
CL_{FDA}_pareto	30	16.31	23.27	19.0227	0.28145	1.54156

Table 16. Comparison results using Wilcoxon test, variable *Execution_time* (c15_9, c15_10 and c15_12 datasets)

Algorithm1	Algorithm2	Signification (p-value)	Groups
GA_pareto	GA_just_time	0.064	*Group 1*: GA_pareto, GA_just_time
GA_pareto	UMDA_pareto	**0.000**	
UMDA_pareto	UMDA_just_time	0.51	*Group 2*: UMDA_ pareto, UMDA_ just_time
UMDA_pareto	CL_{FDA}_just_time	**0.000**	
CL_{FDA}_just_time	CL_{FDA}_pareto	0.959	*Group 3*: CL_{FDA}_just_time, CL_{FDA}_pareto

In c15_9, c15_10 and c15_12 datasets, regarding variable *Execution_time,* the best results were obtained by *Genetic Algorithms* approach, whereas the CL_{FDA} approach used a higher time for the same stop criterion (100 generations). This is because CL_{FDA} algorithms spend more time to detect the dependency relation among variables, but it is able to find solutions which have never been found by *GA* or *UMDA*. Furthermore, the objective is to minimize the makespan of the project, not to minimize the execution time of the algorithms.

5. Conclusion

In this paper, a new approach on Estimation of Distribution Algorithms with constraints handling inside the probabilistic model to solve the Multi-mode Resource Constrained Project Scheduling Problems was developed. The proposal was applied to four datasets of PSPLIB in its multi-mode variant, which have several complexity degrees (task numbers, number of modes, and number of resources). The cost component to be optimized along with time was added, always looking for a balance between them.

The obtained results prove to be very effective on the benchmark instances and improved others reported in the bibliography, especially in j30_17, c15_9 and c15_12 datasets.

Overall, CL_{FDA}*_just_time* has been selected as the best algorithm. It achieved the best results for the *Mean_Makespan, StdDev_MakeSpan* and *%Optimum* variables.

Towards further improvement, the flexibility of adding different components like project quality to the model makes the procedure particularly useful. Moreover, further research on different strategies to diversify the search process can lead to a superior performance of the algorithm.

References

[1] INFANTE, A. (2008) Teamsoft: Sistema Para La Gestión Del Trabajo En Equipo En El Desarrollo de Proyectos de Software, Version 2.0, Universidad de las Ciencias Informáticas, La Habana.

[2] AWASTHI, A. (2016) Optimization of NP-Hard Scheduling Problems by Developing Timing Algorithms and Parallelization, PhD Thesis, Carl von Ossietzky Universität Oldenburg, Germany 8–48 pp.

[3] DUNBAR, G. (2015) Project Management Failures - Standish (Chaos) Reports (1994-2015), https://www.linkedin.com/pulse/project-management-failures-standish-chaos-report-2015-dunbar.

[4] JOHNSON, J. (2018) CHAOS Report: Decision Latency Theory: It Is All About the Interval, Second Edition., The Standish Group 70 pp.

[5] VILLAVICENCIO, N. (2016) Modelo Integrado Para La Mejora de La Productividad En Organizaciones Orientadas a Proyectos de Tecnologías de La Información, PhD Thesis, Tesis para optar al grado de: Máster en Diseño, Gestión, Funiber.

[6] AMOUI, M., DERAKHSHANMANESH, M., EBERT, J., TAHVILDARI, L. (2012) Achieving dynamic adaptation via management and interpretation of runtime models, Journal of Systems and Software **85** 12 2720.

[7] PIÑERO, P.Y. (2013) Paquete para la Dirección Integrada de Proyectos y ayuda a la toma de decisiones: GESPRO, Informática 2223.

[8] MOSSALAM, A., ARAFA, M. (2016) The role of project manager in benefits realization management as a project constraint/driver, HBRC Journal **12** 3 305.
[9] BERMÚDEZ, N.V., ABREU, M.P., VALARESO, S.B., PUPO, I.P. (2016) Experiencias en la integración de procesos en las organizaciones orientadas a proyectos de software, Revista Cubana de Ciencias Informáticas **10** 171.
[10] PACELLI, L. (2004) Grandes errores en la gestión de proyectos, Finantial Times Prenice Hall, ISBN **131490478** 167.
[11] EQUIPO DEL PRODUCTO CMMI, CMMI® Para Desarrollo (2010) Versión 1.3, Technical Report CMU/SEI-2010-TR-033 ESC-TR-2010-033, Editorial Universitaria Ramón Areces, Software Engineering Process Management Program, Carnegie Mellon University.
[12] CHAUDHARY, M., CHOPRA, A. (2017) CMMI for Development - Implementation Guide, Apress.
[13] PROJECT MANAGEMENT INSTITUTE, INC (2017) A Guide to the Project Management Body of Knowledge (PMBOK Guide), Sixth edition., Project Management Institute, Newtown Square, Pennsylvania 19073-3299 USA.
[14] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, ISO 21500 (2012): Guidance on Project Management, Geneva.
[15] BRIOSO, X. (2015) Integrating ISO 21500 guidance on project management, lean construction and PMBOK, Procedia Engineering **123** 76.
[16] VACACELA, R.G., PUPO, I.P., VILLAVICENCIO, N., PEREZ, P.Y.P., LUIS, S.B. (2016) Experiencias usando algoritmos genéticos en la planificación de proyectos, Revista Cubana de Ciencias Informáticas **10** 71.
[17] BRUCKER, P., DREXL, A., MÖHRING, R., NEUMANN, K., PESCH, E. (1999) Resource-constrained project scheduling: Notation, classification, models, and methods, European journal of operational research **112** 1 3.
[18] BROWNING, T.R., YASSINE, A.A. (2010) Resource-constrained multi-project scheduling: Priority rule performance revisited, International Journal of Production Economics **126** 2 212.
[19] AYODELE, M., MCCALL, J., REGNIER-COUDERT, O. (2016) BPGA-EDA for the Multi-Mode Resource Constrained Project Scheduling Problem, Evolutionary Computation (CEC), 2016 IEEE Congress On, IEEE 3417–3424.
[20] AYODELE, M., MCCALL, J., REGNIER-COUDERT, O. (2017) Estimation of Distribution Algorithms for the Multi-Mode Resource Constrained Project Scheduling Problem, 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE, Donostia, San Sebastián, Spain 1579–1586.
[21] PINHA, D.C., AHLUWALIA, R.S., SENNA, P. (2016) The Combinatorial Multi-Mode Resource Constrained Multi-Project Scheduling Problem, International Journal of Supply and Operations Managment **3** 3 1391.
[22] ZHANG, H., LI, H., TAM, C.M. (2006) Permutation-based particle swarm optimization for resource-constrained project scheduling, Journal of Computing in Civil Engineering **20** 2 141.
[23] MEDRANO, B.E. (2013) Planificación de múltiples proyectos de desarrollo de software utilizando métodos metaheurísticos, Master Thesis, Universidad de La Habana, La Habana, 16–50 pp.
[24] BAAR, T., BRUCKER, P., KNUST, S. (1999) "Tabu Search Algorithms and Lower Bounds for the Resource-Constrained Project Scheduling Problem", Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization (VOß, S., MARTELLO, S., OSMAN, I.H., ROUCAIROL, C., Eds), Springer US, Boston, MA 1–18.
[25] ARTIGUES, C., MICHELON, P., REUSSER, S. (2003) Insertion techniques for static and dynamic resource-constrained project scheduling, European Journal of Operational Research **149** 2 249.
[26] DONG, N., GE, D., FISCHER, M., HADDAD, Z. (2012) A genetic algorithm-based method for look-ahead scheduling in the finishing phase of construction projects, Advanced Engineering Informatics **26** 4 737.
[27] WANG, W., WANG, X., GE, X., DENG, L. (2014) Multi-objective optimization model for multi-project scheduling on critical chain, Advances in Engineering Software **68** 33.
[28] KUEHN, M., ZAHID, T., VOELKER, M., ZHOU, Z., ROSE, O. (2016) Investigation Of Genetic Operators And Priority Heuristics for Simulation Based Optimization Of Multi-Mode Resource Constrained Multi-Project Scheduling Problems (MMRCMPSP), ECMS 2016 Proceedings Edited by Thorsten Claus, Frank Herrmann, Michael Manitz, Oliver Rose, ECMS 481–487.
[29] AYODELE, M., MCCALL, J., REGNIER-COUDERT, O. (2016) RK-EDA: A Novel Random Key Based Estimation of Distribution Algorithm, Springer 849–858.
[30] AYODELE, M., MCCALL, J., REGNIER-COUDERT, O. (2017) Estimation of Distribution Algoprithms for the Multi-Mode Resource Constrained Project Scheduling Problem, 1579.
[31] TALBOT, F.B., Resource-Constrained Project Scheduling with Time-Resource Trade offs: The Nonpreemptive Case, Management Science **28** 10 (1982) 1197.
[32] KOLISCH, R., SPRECHER, A. (1997) PSPLIB - A project scheduling problem library: OR Software - ORSEP Operations Research Software Exchange Program, European Journal of Operational Research **96** 1 205.
[33] MÜHLENBEIN, H., MAHNIG, T., OCHOA-RODRÍGUEZ, A. (1999) Schemata, Distributions and Graphical Models in Evolutionary Optimization, J. Heuristics **5** 2 215.
[34] CHEN, Y. et al. (2017) Personalized Search Inspired Fast Interactive Estimation of Distribution Algorithm and Its Application, IEEE Transactions on Evolutionary Computation **21** 4 588.
[35] ECHEGARAY, T.M. (2010) Avances en Algoritmos de Estimación de Distribuciones. Alternativas en el Aprendizaje y Representación de Problemas, Ph. D. dissertation, Universidad del País Vasco, España.
[36] RODRIGUEZ, R.P. et al. (2016). Un algoritmo de estimación de distribuciones para el problema de ruteo de autobuses escolares con selección de paradas, DYNA New Technologies **3** 1.
[37] MÜHLENBEIN, H., MAHNIG, T. (1999) Convergence Theory and Applications of the Factorized Distribution Algorithm, Journal of Computing and Information Technology **7** 19.
[38] EMMERICH, M.T.M., DEUTZ, A.H. (2018) A tutorial on multiobjective optimization: fundamentals and evolutionary methods, Natural Computing **17** 3 585.
[39] Multi Mode Data Sets, http://www.om-db.wi.tum.de/psplib/getdata.cgi?mode=mm.