NSPLib – A Nurse Scheduling Problem Library: A tool to evaluate (meta-)heuristic procedures

Mario Vanhoucke[1,2] and Broos Maenhout[1]

Submitted to ORAHS 2005 Proceedings

[1]*Faculty of Economics and Business Administration, Ghent University, Gent, Belgium*
[2]*Operations & Technology Management Centre, Vlerick Leuven Gent Management School, Gent, Belgium*
mario.vanhoucke@ugent.be • broos.maenhout@ugent.be

In this paper, we propose a large set of data instances based on different complexity indicators for the well-known nurse scheduling problem (NSP). The NSP assigns nurses to shifts per day taking both hard and soft constraints into account. The objective is to maximize the nurses' preferences and to minimize the total penalty cost from violations of the soft constraints. The problem is known to be NP-hard.

Due to its complexity and relevance in practice, many researchers have developed (meta-)heuristic procedures to solve a NSP instance heuristically in an acceptable time limit. However, these solution procedures are often very case-specific towards one hospital and hence, cannot be compared with each other. Moreover, lack of data and the many interpretations of how to evaluate solution procedures have contributed to the never-ending amount of newly developed procedures without any effort to benchmark them in literature.

The contribution of this paper is threefold. First, we propose a large set of benchmark instances for the nurse scheduling problem in order to facilitate the evaluation of existing and future research techniques. Secondly, we propose a computer platform independent stop criterion to evaluate and compare meta-heuristic procedures for the NSP. Finally, we propose a newly developed website where the benchmark instances can be downloaded and where the best known solutions can be uploaded.

Keywords: *Nurse scheduling; Benchmark instances; Problem classification*

1 Introduction

The nurse scheduling problem (NSP) is a well-known combinatorial optimization problem in literature and has attracted numerous researchers to develop exact and (meta-)heuristic procedures. The NSP involves the construction of duty rosters for nursing staff and assigns the nurses to shifts per day taking both hard and soft constraints into account. The objective maximizes the preferences of the nurses and minimizes the total penalty cost from violations of the soft constraints. The problem is known to be NP-hard (Osogami and Imai (2000)).

Due to its complexity and relevance in practice, the operations research literature has been overwhelmed by different procedures to solve the problem. The complexity has resulted in the development of several (meta-)heuristic procedures, able to solve a NSP instance heuristically in an acceptable time limit. The practical relevance has resulted in a never-ending amount of different NSP versions, taking practical, case-specific constraints into account. Despite the numerous procedures for the NSP, no state-of-the-art results have been presented in literature. The main reason is that comparison between procedures is very difficult, since problem descriptions and models vary drastically and depend on the need of the particular hospital. Due to the huge variety of hard and soft constraints, and the several objective function possibilities, the nurse scheduling problem has a multitude of representations which resulted in the wide variety of different solution procedures. The comparison between procedures is further hindered by the lack of benchmark problem instances and the unavailability of source code of the different procedures. Moreover, there is no general agreement of how to evaluate

and compare procedures in terms of solution comparison, stop criterion, etcetera. Consequently, a fair comparison between procedures seems to be an impossible idea, which undoubtedly limits the efficient development of future algorithms. Note that both Cheang et al. (2003) and Burke et al. (2004) express the need for a benchmark database to facilitate comparisons of the various algorithms and to motivate future researchers to develop better solution procedures for the NSP.

In this paper, we come towards this need of benchmarking by presenting a large set of data instances for the NSP. The outline of this paper is as follows. In section 2, we highlight the necessary conditions under which a benchmark dataset can be used in the operations research community. In section 3, we present the details of our nurse scheduling problem library NSPLIB. In this section, the complexity of the NSP is discussed, followed by the settings and details of our data instances. Moreover, we propose a computer platform independent stop criterion such that our data instances allow a fair evaluation and comparison between different meta-heuristic procedures for the NSP. Section 4 makes conclusions and highlights avenues for future research.

2 Benchmark libraries in the OR community

One of the most important applications of a library is to provide a common set of problem instances on which different researchers can easily benchmark their algorithms. In doing so, the library provides a competitive environment among researchers, stimulates progress in the development of algorithms and contributes to gain insights in the performance of these algorithms. Hooker (1995) calls our attention to potential pitfalls of *competitive testing*, and argues that benchmark instances should stimulate scientific testing rather than only create a competitive environment where algorithms are purely evaluated on computational performance. *Scientific testing*, on the contrary, still recognizes that fast and efficient code should be written, but puts more focus on the development of algorithms to yield insight into their performance. As a consequence, we believe a benchmark dataset should satisfy the following conditions (these four conditions, among others, have also been mentioned by Gent and Walsh (1999)):

- Diversity: the problem set should contain instances that are as diverse as possible. Hoos and Stützle (2000) argue that benchmark sets should contain a large variety of different types of problem instances, such that they can be used for evaluating different types of algorithms in an as unbiased as possible way. Elmaghraby and Herroelen (1980) argue that diversity of problem instances (which are, in their case, project networks) is indispensable in order to span *the full range of complexity*. Hooker (1995) argues that a controlled experimentation begins with a selection of *n* factors that could affect performance, in order to have a diverse set of problem instances which are homogeneous with respect to characteristics that are likely to affect performance.
- Realism: Since the ultimate goal of research is to develop algorithms for solving real-life problems, a benchmark library should reflect real-life problems and hence should be built as realistic as possible.
- Size: Ideally, a benchmark dataset should be as large as possible. However, testing algorithms often involves a trade-off between problem instance size and solvability (due to the heavy burden of required CPU time). Hence, we have chosen to vary the size of our problem instances, from rather small to sufficiently large instances, in order to allow the researcher to draw general conclusions on his/her selected data instances.

- Extensibility: the problem set should be easy to extend to other features. A dynamic benchmark dataset is preferable above a static one (Hooker (1994, 1995), since the latter assumes that a problem description (like the NSP) never changes. However, new features can show up as progress is made, coming from new research avenues or as a result of practical needs. The data set should allow the incorporation of these new features in an easy and comprehensible way.

In the remainder of this paper, we will propose different data instances that fulfil the requirements above. In the next section, we propose both a diverse set and a realistic set with varying size and that are easily extendable to other features.

3 NSPLIB: the Nurse Scheduling Problem LIBrary

3.1 Complexity of the NSP

In the basic nurse scheduling problem, a set of nurses needs to be assigned to one of a number of possible shifts in order to meet the minimal coverage constraints and other case-specific constraints and to maximize the quality of assigned working shifts. According to Warner (1976), quantifying *preferences* in the objective function maintains fairness in scheduling nurses over the scheduling horizon. Hence, the quality of a schedule is a subjective judgment of the nurses depending on how well the assigned schedule is conform to his/her desires to be off or on duty and to other schedule properties such as work stretch, rotation patterns, etc…. The *coverage constraints* determine the required nurses per shift and per day, and are inherent to each NSP instance. However, many other constraints are very *case-specific*, and are determined by personal time requirements, specific workplace conditions, national legislation, etc…. Consequently, each data instance should contain information of the following three classes in order to be useful for a NSP procedure

- Preference matrix: the preference or aversion of nurses to work on a shift/day
- Coverage constraints: the required number of nurses
- Other (often very case-specific) constraints

The first two classes describe a two-dimensional nurse/day roster matrix, which is inherent to any NSP instance. To that purpose, Vanhoucke and Maenhout (2005) have proposed 9 complexity indicators in order to describe a NSP roster matrix and have presented a generator to construct NSP instances based on these 9 input parameters. These indicators can be used to generate data under a controlled design to measure the size and the structure of the preference matrix and the corresponding coverage requirements. Hence, these indicators do not describe the case-specific constraints, which will be discussed in section 3.2.

The size of the NSP instance under study depends on the size of the duty roster matrix. Hence, the size of a NSP instance can be measured by three indicators, as follows:

- Number of nurses
- Number of days
- Number of shifts

Three indicators measure the structure of the preference matrix, as follows:

- Nurse-preference distribution (*NPD*): Distribution of preferences among the nurses
- Shift-preference distribution (*SPD*): Distribution of nurse's preferences among shifts
- Day-preference distribution (*DPD*): Distribution of nurse's preferences among days

The last three indicators measure the way the coverage requirements are distributed among shifts and days, as follows:

- Total coverage constrainedness (*TCC*): Total number of nurses required
- Shift-coverage distribution (*SCD*): Distribution of coverage requirements among shifts
- Day-coverage distribution (*DCD*): Distribution of coverage requirements among days

For more information about the calculation and interpretation of these indicators, we refer to the working paper of Vanhoucke and Maenhout (2005).

3.2 The library

The development of exact and (meta-)heuristic procedures for the nurse scheduling problem arose the need for benchmark instances (Cheang et al. (2003) and Burke et al. (2004)). Rather than testing a newly developed algorithm on one real-life NSP instance (which has high practical relevance), one can additionally test its procedure on a dataset and report results. In doing so, a benchmark dataset that will be shared among the research community will facilitate the systematic evaluation and comparison of the performance of the different procedures.

In the following of this subsection, we discuss the nurse scheduling problem library (NSPLIB) containing three main categories. In section 3.2.1, two sets of problem instances are presented accessible by the research community. In section 3.2.2, we show how to incorporate classes of case-specific constraints. Finally, in section 3.2.3, we propose a way to test and evaluate procedures independent of the computer platform.

3.2.1 Benchmark instances

The benchmark instances have been grouped in two different sets, a diverse and a realistic set, each containing four and two sub-sets, respectively. Each sub-set is characterized by systematically varied levels of the complexity indicators. Table 1 contains the values for the different levels of the 9 complexity indicators.

The diverse set has been constructed using three levels for the last six complexity indicators, in order to guarantee that the problem instances span the full range of complexity. Although we varied the number of nurses from 25 to 100, we kept the planning horizon rather small (7 days). Using 10 instances per setting, each sub-set contains 7,290 problem instances. This large number of benchmark instances has been constructed to test meta-heuristic procedures in an acceptable time (small planning horizon) but prevents the over-fitting of the tested procedures (due to the large number of instances, i.e. 29,160 in total).

The realistic set has been constructed with only two levels for the last six complexity indicators, but within a planning horizon of 28 days. Since many procedures in literature have been developed for and tested within a planning horizon of 28 days, we believe that this set is crucial and indispensable to compare and benchmark the current state-of-the-art NSP procedures[1]. Using 10 instances per setting, each sub-set contains 960 instances.

[1] We would like to thank Prof. Dr. Marion Rauner for drawing our attention on this issue during 31st meeting of the EURO Working Group "Operational Research Applied to Health Services" at the University of Southampton (UK).

Table 1. Test settings for our benchmark instances

Diverse Set		Realistic Set	
Problem size		**Problem size**	
N	25, 50, 75 or 100	***N***	30 or 60
S	4 (including the free shift)	***S***	4 (including the free shift)
D	7	***D***	28
Preference distribution		**Preference distribution**	
NPD	0.25, 0.50 or 0.75	***NPD***	0.3 or 0.7
SPD	0.25, 0.50 or 0.75	***SPD***	0.3 or 0.7
DPD	0.25, 0.50 or 0.75	***DPD***	0.3 or 0.7
Coverage constraints		**Coverage constraints**	
TCC	0.20, 0.35 or 0.50	***TCC***	0.20, 0.35 or 0.50
DCD	0.25, 0.50 or 0.75	***DCD***	0.3 or 0.7
SCD	0.25, 0.50 or 0.75	***SCD***	0.3 or 0.7

The problem instances can be downloaded from www.projectmanagement.ugent.be/nsp.php under the names N25.zip, N50.zip, N75.zip and N100.zip (diverse set) and N30.zip and N60.zip (realistic set). Note that, to the best of our knowledge, only two other NSP data instances are available on the internet by Burke et al. (2004) and Musliu et al. (2004). The first set contains real world data and can be downloaded from http://ingenieur.kahosl.be/vakgroep/it/nurse/archive.htm. The second set contains a randomly generated dataset for the rotating workforce scheduling problem, and can be downloaded from http://www.dbai.tuwien.ac.at/staff/musliu/benchmarks/.

3.2.2 Case-specific constraint

In order to solve nurse scheduling problem instances, many more case-specific constraints can be incorporated in the problem description. Cheang et al. (2003) identified nine constraint types as appearing frequently in the literature. We have extended the NSPLIB with case constraint files (CC-files) which incorporates a subset of these nine constraints with specific input parameters. Each file can be used as an additional set of constraints of the proposed benchmark instances. The daily coverage requirements are an inherent characteristic to all personnel rostering problems, and are incorporated in three of the aforementioned indicators. Moreover, two other constraints (i.e. minimal free time between working shifts and the fact that nurses only can take one assignment per day) occur almost always in literature (Cheang et al. (2003)). These three constraints are supposed to characterize all nurse scheduling problems and are applicable in all cases (Koop (1998)). Furthermore, some constraints could be distinguished which frequently typify nurse scheduling practices but are not always incorporated. To that purpose, we have classified the case-specific constraints as follows:

- Number of assignments, i.e. non-free shift assignments per scheduling period
- Number of assignments per shift, i.e. identical shift assignments per scheduling period
- Consecutive working shifts, i.e. non-free shift assignments per scheduling period
- Consecutive same working shifts, i.e. identical shift assignments per scheduling period

Not only the presence of these constraints is an indicator of the restrictiveness of nurse scheduling instances, but also the parameter setting for each constraint contributes to the restrictiveness of this problem. For each of these cases, some settings are defined based on

specific cases described in literature or on interesting settings not mentioned in literature. An overview of the sixteen cases containing a mix of case-specific constraints and their corresponding settings are given in table 2. The minimal value and maximal value is given between brackets and can possibly be different for each shift. As an example, CC-file 7 requires minimal 2 consecutive assignments and maximum 3 for the three working shifts (no requirements are given for the free shift), denoted by [2,3], [2,3], [2,3]. The detailed format of each CC-file and the specific settings of each constraint are given on the website and are outside the scope of this paper.

Table 2. The case-specific constraints per CC with [min, max] settings

	Number of assignments	Number of assignments per shift	Consecutive working shifts	Consecutive same working shifts
	Diverse Set: N25, N50, N75 and N100 instances			
Case 1	[5,5]	x	x	x
Case 2	[4,6]	x	x	x
Case 3	[5,5]	[1,3], [1,3], [1,2]	x	x
Case 4	[4,5]	[0,5], [0,5], [0,4]	x	x
Case 5	[5,5]	x	[2,5]	x
Case 6	[4,6]	x	[1,5]	x
Case 7	[5,5]	[0,5], [0,5], [0,3]	[2,5]	[2,3], [2,3], [2,3]
Case 8	[2,6]	[0,6], [0,6], [0,3]	[2,4]	[1,4], [1,4], [2,4]
	Realistic Set: N30 and N60 instances			
Case 9	[20,20]	x	[1,7]	[1,7], [1,7], [1,7]
Case 10	[16,24]	x	[1,7]	[1,7], [1,7], [1,7]
Case 11	[20,20]	[4,12], [4,12], [4,8]	[1,7]	[1,7], [1,7], [1,7]
Case 12	[16,20]	[0,20], [0,20], [0,16]	[1,7]	[1,7], [1,7], [1,7]
Case 13	[20,20]	x	[2,5]	[1,7], [1,7], [1,7]
Case 14	[16,24]	x	[1,5]	[1,7], [1,7], [1,7]
Case 15	[20,20]	[0,20], [0,20], [0,12]	[2,5]	[2,3], [2,3], [2,3]
Case 16	[16,24]	[0,24], [0,24], [0,12]	[2,4]	[1,4], [1,4], [2,4]

3.2.3 Evaluation of (meta-)heuristic procedures for the NSP

We would like to express that researchers who test their specific procedure on the proposed test design contribute to the development of best-known solutions which can be used for future comparison purposes. However, in order to make a fair comparison between procedures, one need a clear and easily applicable stop criterion that is independent of the computer platform and coding skills. In recent research papers, many algorithms have been developed, but little effort has been done in standardizing the test approach. Table 3 displays the different meta-heuristic approaches for the NSP. For each method, we have listed the stop criterion, the required computational time, the test instances used as well as information about the problem size and the computer platform.

Table 3. Test settings for the meta-heuristic procedures in literature

We have copied the original terminology of the papers (e.g. the term ‘generations’ is often used for genetic algorithms while ‘moves’ are used in tabu search algorithms), without explaining its details. Inoue et al. (2003), for example, clearly describe a generation as a combination of crossover operations, heuristics and mutation. In other papers, the stop criteria

are somewhat more vague. Therefore, we would like to propose a test design based on a clear and easily applicable stop criterion. This test design is very much inspired on the critical remarks on the use of the test design written by Kolisch and Hartmann (2005). These authors use the number of created schedules as a stop criterion for meta-heuristic procedures in project scheduling. In doing so, the criterion is independent of the computer platform, and hence, allows a fair comparison between procedures. Inspired on these ideas and given the competitive environment where researchers develop project scheduling procedures which outperform previous ones, we would like to call for using the *number of visited solutions* as the one and only stop criterion for testing meta-heuristic nurse scheduling problem procedures. Although this stop criterion can also be criticised for its potential pitfall, we would like to minimise the confusion about when a solution can be considered as visited (and hence, accounts for an extra solution closer to the stop criterion). Therefore, from the moment the algorithm evaluates the objective function of the new solution, an additional solution has been generated by the algorithm. This involves that mutation operations and local search algorithms only contribute to the solution generation, only from the moment an evaluation takes places. Consecutive (small) changes to a nurse roster (e.g. a series of shift-pattern swaps between nurses) only results in a new solution when the effect on the objective function is known. This is completely in line with Kolisch and Hartmann (2005) who use the use of a so-called schedule generation scheme, which basically involves the construction and evaluation of a project schedule, as a criterion to count an extra schedule.

The data instances, the CC-files and the solutions can be downloaded from our website (www.projectmanagement.ugent.be/nsp.php). The solution files (in excel) contain the individual solution of each problem instance for 1000 and 5000 schedules, as well as the best known solutions for the test instances (without a given stop criterion). For each problem instance the total solution quality, the CPU-time, and the number of violations regarding the minimal coverage requirements have been reported. Up to now, the solutions are found by the procedure of Maenhout and Vanhoucke (2005) found by their electromagnetism approach.

The website will be regularly extended, and we call upon researchers to report their solutions of their procedures when this leads to an improvement (both within a stop criterion of 1,000 and 5,000 schedules and without a stop criterion). Moreover, newly tested (real-life) nurse scheduling problem instances and new case constraint files are welcome and contribute to the dynamic extension of our library. Finally, researchers who wish to generate other problem instances to adapt them to their specific settings can download the generator from the same web address. We are confident that we have created a competitive environment to stimulate researchers to outperform these known results by the development of novel procedures.

3.3 Extensibility to other problem types

In the previous section, we argued that the proposed library is open for extensions, and researchers are free to submit new (real-life) data instances as well as new CC-files. We believe that our suggested data sets can be used for a wide variety of nurse scheduling problems, without altering the data, and might be good benchmark sets for a broader class of employee scheduling problems.

The incorporation of *grades among nurses* (i.e. employee skill level or employee status) implies day/shift coverage requirements for each grade, and assumes that coverage requirements for nurses with a particular grade can be fulfilled by nurses of a higher grade. The extension to different *types of nurses* (e.g. full-time nurses, part-time and nurses working

four days a week, etc…) can also be easily incorporated in algorithms and does not require drastic changes in the data. Each nurse of our data instances needs to be assigned to one of the possible types or grades of nurses defined by the user. In both cases (types and grades) it might seem necessary to define new CC-files to incorporate these extra features.

Employee scheduling problems arise in a variety of environments and hence, we believe that our data instances can be used for other problem types outside hospitals. Glover and McMillan (1986) mention telephone operators scheduling, airline and hotel reservation personnel scheduling, bank personell scheduling and many more applications in the service delivery settings. In their paper, they mention availability preferences that can be specified by each employee, which resemble very much on the nurses' preferences of the current paper. Hence, we believe that our current data instances are general and can be used in a wide variety of settings.

4 Conclusions

In this paper, we have presented a benchmark dataset for the well-known nurse scheduling problem. A nurse scheduling problem instance is characterised by a two-dimensional roster matrix, describing the nurses' preferences and related coverage requirements, as well as by other, often very case-specific constraints. In our dataset, we have relied on 9 indicators developed by Vanhoucke and Maenhout (2005) to control the two-dimensional roster matrix, and have presented so-called CC-files to incorporate case-specific constraints.

The proposed problem sets fulfil four important criteria of a benchmark library, i.e. diversity, realism, varying size and extensibility. First, our testset guarantees *diversity* in order to span the full range of complexity (Elmaghraby and Herroelen, 1980). More precisely, we have generated a large diverse set, containing 4 sub-sets with 6 varying input parameters measuring the structure of the roster matrix. All data instances have been constructed for a scheduling horizon of 7 days and 4 shifts. Secondly, a second data set, serves as our *realistic* set, since we have generated data instances that assume a four-week planning horizon and contains a much smaller amount of data instances. Thirdly, each sub-set of the previously mentioned dataset has varying *sizes*: the diverse sub-sets contain instances with 25, 50, 75 and 100 nurses while the realistic sub-sets contain instances with 30 and 60 nurses. Last, our library fulfils the *extensibility* criterion since it is one of our aims to dynamically update the library with newly developed best known solutions, new real-life data instances as well as newly created case-specific constraint files.

All data instances, as well as the obtained solutions and the CC-files can be downloaded from the website www.projectmanagement.ugent.be/nsp.php. The website will be regularly extended, and we strongly motivate researchers to report their new benchmark solutions on this site. Adding new CC-files as well as real-life problems also belongs to our update intensions and contributes to the competitive environment among researchers in solving different versions of the nurse scheduling problem. We are confident that we have created a competitive environment to stimulate all researchers to make progress and benchmark their results with the best known results in literature. We, at least, will spend time and effort in the maintenance of the website in order to facilitate the accessibility and hence the progress in research.

References

Aickelin, U., and Dowsland, K.A., 2000, "Exploiting problem structure in a genetic algorithm approach to a nurse rostering problem", *Journal of Scheduling*, 3, 139-153.

Aickelin, U., and Dowsland, K.A., 2004, "An indirect Genetic Algorithm for a nurse-scheduling problem", *Computers and Operations Research*, 31, 761-778.

Bellanti, F., Carello, G., Della Croce, F., and Tadei, R., 2004, "A greedy-based neighborhood search approach to a nurse rostering problem", *European Journal of Operational Research*, 153, 28-40.

Burke, E.K., De Causmaecker, P., and Vanden Berghe, G., 1998, "A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem", *Lecture Notes in Computer Science*, 1585, 187-194.

Burke, E., Cowling, P., De Causmacker, P. and Vanden Berghe, G., 2001, "A memetic approach to the nurse rostering problem", *Applied Intelligence*, 3, 199-214

Burke, E.K., De Causmaecker, P., Vanden Berghe, G. and Van Landeghem, H., 2004, "The state of the art of nurse rostering", *Journal of Scheduling*, 7, 441-499.

Burke, E., De Causmacker, P., Petrovic, S., and Vanden Berghe, G., 2004, "Variable neighbourhood search for Nurse Rostering Problems", p153-172 in Resende M.G.C, and Pinho de Sousa, J., 2004, Metaheuristics: Computer Decision-Making, Kluwer Academic Publishers, 736p

Cheang, B., Li, H., Lim, A., and Rodrigues, B., 2003, "Nurse rostering problems – a bibliographic survey", *European Journal of Operational Research*, 151, 447-460.

Cowling, P., Kendall G., and Soubeiga, E., 2002, "Hyperheuristics: A Robust Optimisation Method Applied to Nurse Scheduling", *Lecture Notes in Computer Science*, 2439, 851-860.

Dias, T.M., Ferber, D.F., de Souza, C.C., and Moura, A.V., 2003, "Constructing nurse schedules at large hospitals", *International Transactions in Operational Research*, 10, 245-265

Dowsland, K.A., 1998, "Nurse scheduling with Tabu Search and strategic oscillation", *European Journal of Operational Research*, 106, 393-407.

Elmaghraby, S.E., and Herroelen, W.S., 1980, "On the measurement of complexity in activity networks", *European Journal of Operational Research*, 5, 223-234.

Gent, I.P., and Walsh, T., 2004, "CSPLIB: A Benchmark Library for Constraints", *Lecture Notes in Computer Science*, 1713, 480-481.

Glover, F., and McMillan, C., 1986, "The General Employee Scheduling Problem: An integration of MS and AI", *Computers and Operations Research*, 13, 563-573.

Hooker, J.N., 1994, "Needed: An empirical science of algorithms", *Operations Research*, 42, 201-212.

Hooker, J.N., 1995, "Testing Heuristics – We Have It All Wrong", *Journal of Heuristics*, 1, 33-42.

Hoos, H.H., and Stützle, T., 2000, SATLIB: An Online Resource for Research on SAT, in Gent, I., van Maaren, H., and Walsh, T., "SAT20000: Highlights of Satisfiability Research in the year 2000", *Frontiers in Artificial Intelligence and Applications*, Kluwer Academic, 2000, 283-292.

Inoue, T., Furuhashi T., Maeda, H., and Takaba, M., 2003, "A Proposal of Combined Method of Evolutionary Algorithm and Heuristics for Nurse Scheduling Support System", *IEEE Transactions on Industrial Electronics*, 50, 833-838.

Jan, A., Yamamoto M., and Ohuchi, A., 2000, "Evolutionary Algorithms For Nurse Scheduling Problem", *Proceedings of the 2000 Congress on Evolutionary Computation*, 196-203.

Kolisch, R. and Hartmann, S., 2005, "Experimental investigation of heuristics for resource-constrained project scheduling: An update", *European Journal of Operational Research*, to appear.

Koop, G.J., 1988, "Multiple shift workforce lower bounds", *Management Science*, 34, 1221-1230.

Li, J., and Aickelin, U., 2004, "The Application of Bayesian Optimization and Classifier Systems in Nurse Scheduling", *Lecture Notes in Computer Science, Proceedings of PPSN VIII: 8th International Conference: Parallel Problem Solving from Nature*, 3242, 851.

Louw, M.J., Nieuwoudt, I., and van Vuuren, J.H., 2005, "Finding Good Nursing Duty Schedules: a Case Study", *Journal of Scheduling,* to appear.

Maenhout, B. and Vanhoucke, M., 2005, "An Electromagnetism meta-heuristic for the nurse scheduling problem", *working paper 05/316*, Ghent University.

Musliu, N., Schaerf, A., Slany, W., 2004, "Local search for shift design", *European Journal of Operational Research*, 153, 51-64.

Osogami, T., and Imai, H., 2000, "Classification of Various Neighbourhood Operations for the Nurse Scheduling Problem", *Lecture Notes in Computer Science*, 1969, 72-83.

Vanhoucke, M., and Maenhout, B., "Characterisation and Generation of Nurse Scheduling Problem Instances", *working paper 2005*, Ghent University.

Warner, H.W., 1976, "Scheduling Nursing Personnel According to Nursing Preference: A Mathematical Approach", *Operations Research*, 24, 842-856.

Table 3. Test settings for the meta-heuristic procedures in literature

Reference[1]	Stop Criterion	CPU	Data set	Problem Size[2] N, D, S	Computer Platform
Aickelin and Dowsland, 2000 (GA)	No improvement during 30 generations	14.9s	52 real data sets	30, 7, 4	Pentium II 200
Aickelin and Dowsland, 2004 (GA)	100 generations	9.3s	52 real data sets	30, 7, 4	Pentium II
Bellanti et al., 2004 (TS)	No improvement during 50 neighbourhoods scans	2s up to 6h47m36s	4 real data instances / 12 random instances	20 to 60, 30, 4	Pentium IV, 2Ghz
Burke et al., 1998 (TS)	No improvement during x moves	1m28s up to 28m8s	2 real data instances	20, 28, 8	IBM Power PC RS6000
Burke et al., 2001 (TS/GA/MA)	No improvement during 2 generations	49s up to 2h31m51s	4 real data instances	20, 28, 8	IBM Power PC RS6000
Burke et al., 2005 (TS)	No improvement during x moves	3m41s up to 23m19s	1 real data instance	20, 28, 8	IBM Power PC RS6000
Cowling et al., 2003 (TS/GA)	6,000 iterations	44s up to 60s	52 real data instances	30, 7, 4	Pentium II, 1000 Mhz
Dias et al., 2003 (GA/TS)	×	ca 120s	12 real data instances	30, 30, 4	AMD Athlon 600
Dowsland, 1998 (TS)	No improvement during 1,000 moves	60s up to 150s	real data sets	Up to 30, 7 to 42, 4	Pentium 60
Inoue et al., 2003 (GA)	50 generations	×	1 real data instance	20, 30, 6	×
Jan et al., 1999 (GA)	10,000; 100,000; 200,000 generations	49s	1 real data instance	15, 30, 4	Pentium II 300
Kragelund and Mayoh, 1999 (SA)	No improvement during 1,000,000 moves or when penalty costs are zero	×	2 real data instances	45, 30, 11	×
Li and Aickelin, 2003 (BA)	2,000 generations	10s up to 20s	52 real data sets	30, 7, 4	Pentium 4
Louw et al., 2005 (TS)	No improvement >2% during x moves	1s up to 2h18m09s	1 real data instance	22, 28, 3	667 Mhz

[1]We use the abbreviations GA for Genetic Algorithms, TS for Tabu Search algorithms, MA for Memetic Algorithms and BA for Bayesian optimization Algorithm

[2]We use the notation N (number of nurses), D (number of days) and S (number of shifts) to describe the size of the roster matrix

× = no information available