An Experimental Investigation of Reoptimization for Shift Scheduling

David Meignan

Universität Osnabrück
Institut für Informatik, Albrechtstraße 28, 49069 Osnabrück, Deutschland
dmeignan@uni-osnabrueck.de

Abstract

This paper presents an experimental study conducted with subjects on an interactive reoptimization method applied to a shift scheduling problem. The studied task is the adjustment, by a user, of candidate solutions provided by an optimization system in order to introduce a missing constraint. Two procedures are compared on this task. The first one is a manual adjustment of solutions assisted by a software that dynamically computes the cost of the current solution. The second procedure is based on reoptimization. For this procedure, the user defines some desired changes on a solution, and then a reoptimization method is applied to integrate the changes and reoptimize the rest of the solution. This process is iterated with additional desired changes until a satisfactory solution is obtained. For this interactive approach, the proposed reoptimization procedure is an iterated local search metaheuristic. The experiment, conducted with 16 subjects, provides a quantitative evaluation of the manual and reoptimization approaches. The results show that, even for small local adjustments, the manual modification of a solution has an important impact on the quality of the solution. In addition, the experiment demonstrates the efficiency of the interactive reoptimization approach and the adequacy of the iterated local search method for reoptimizing solutions. Finally, the experiment revealed some limitations of interactive reoptimization that are discussed in this article.

1 Introduction

Optimization-based decision support systems are tools that can support a decision maker to solve a complex optimization problem [13]. Such a system provides the decision maker with at least a model of the optimization problem, an optimization procedure for solving it, and means of instantiating the optimization model as well as analyzing solutions [4]. However, in many cases, the optimization model does not capture all aspects of the real problem. For instance, some aspects of the problem may be too difficult to express in mathematical terms (e.g. a robustness criterion), or the optimization model may have been simplified for being tractable. This gap between the real problem and the optimization model can result in the computation of inadequate or unrealistic solutions. In such cases, the decision maker can adjust the solutions to introduce the missing aspects. The adjustment can be performed manually which can be complex and potentially impair the quality of the solutions. Alternatively, the adjustment process can be done interactively using a reoptimization procedure that introduces local modifications in a solution while maintaining the quality of the entire solution [10].

Reoptimization procedures are used in two different contexts, namely dynamic optimization and interactive optimization. In dynamic optimization, the problem data change over time and a reoptimization procedure can be used to adapt a solution according to the perturbations (see for instance [2, 12, 14]). In interactive optimization, the perturbations do not correspond to modifications of the real problem, but are adjustments of a solution requested by a user of the optimization system. Our study investigates the latter context, where a reoptimization procedure is used in an interactive process for adjusting a solution. We use the term *interactive reoptimization* for this process.

In the research literature there are relatively few studies on interactive reoptimization. In [11] the author reviews different optimization methods for scheduling problems and suggests the use of a reoptimization procedure when a user modifies a solution. In the interactive reoptimization method proposed in [11], the user can edit manually a solution and then reoptimize the solution while keeping the manually modified parts *frozen*. Another interactive reoptimization approach is proposed in [5] for solving linear optimization problems. The authors study in particular the *stability* and *responsiveness* of different reoptimization algorithms. The stability is the ability of a reoptimization procedure to minimize the changes it induced on the initial solution. The responsiveness is related to the computation time required for reoptimizing a solution. Finally, in [10] an interactive reoptimization method is proposed for a shift scheduling problem. The work presented in the current paper is based on this latter study.

To the best of our knowledge, no experimental study with real users on interactive reoptimization is reported in the research literature. Results presented in previous works, such as in [5] and [10], are computational evaluations of reoptimization procedures. It should be noted that for other interactive optimization approaches, such as human-guided search [8], experimental studies with test subjects have been performed (see for instance [1, 8]) but the purpose of the interaction is different from that of interactive reoptimization. Looking at the interactive aspect of reoptimization, it appears important to validate such an approach through an experimental study with real users, which is the principal objective of the work presented in this article. The proposed experiment quantitatively evaluates an interactive reoptimization approach with real interactions and realistic datasets. In addition, we compare the reoptimization approach with manual edition in order to determine the gain of interactive reoptimization for adjusting solutions. Finally, we identify the limitations of the proposed interactive reoptimization approach by examining the gap between reoptimized and ideal solutions. In summary, the presented experiment aims at providing experimental evidences for promoting the use of interactive reoptimization and also aims at encouraging further experimental investigations of interactive optimization methods.

In the next section, the interactive reoptimization process is detailed and the optimization problem on which it is applied is presented. In Section 3 the goals and the method of the experimental evaluation are explained. The results of the experiment are presented and discussed in Section 4. Finally, concluding remarks are given in Section 5.

2 Interactive reoptimization for shift scheduling

2.1 Interactive process

Interactive reoptimization aims at adjusting solutions when inaccuracies in an optimization model result in inadequate computed solutions. Although the enrichment of an optimization model is preferable to the adjustment of solutions, in many cases it is not possible to integrate all aspects of the real problem in an optimization model. In this context, it is necessary to rely on the user to adjust solutions according the real problem. The interactive reoptimization approach therefore assumes that the user has an expertise in the application domain, but in turn, it does not require knowledge in modelling or optimization.

The studied interactive reoptimization process starts with an initial solution computed by an optimization procedure. We consider the case where this initial solution has been computed with an incomplete or inaccurate optimization model and consequently may require some modifications. The solution is presented to the user who can check whether the solution is valid or needs to be adjusted. If some adjustments have to be made on the solution, the user specifies the changes that are required and runs a reoptimization procedure for integrating them. In the proposed reoptimization approach, the changes requested by the user are specified in terms of preferred values on decision variables (i.e. a set of preferred or inadequate values can be assigned to each decision variable). The reoptimization procedure aims at integrating the requested changes and reoptimizing globally the solution in order to maintain the quality of the solution. Since the reoptimization can modify some parts of the solution where no changes have been requested, it may be necessary to perform several iterations of reoptimization with additional changes' requests before obtaining a satisfactory solution. For these subsequent iterations, the last reoptimized solution is used as the starting solution and additional requested changes are combined with previous ones to avoid recurrence of inadequate components in the solution.

The main computational component of this interactive process is the reoptimization procedure. In comparison to the optimization procedure used for generating the initial solution, the reoptimization procedure has particular features. First, in addition to the initial constraints and objectives, the optimization model used for the reoptimization contains two supplementary objectives. The first one aims at integrating the changes requested by the user. In the proposed reoptimization model, this first additional objective ensures that decision variables are set with preferred values (when such a preference exists). The second additional objective minimizes the distance between the initial solution and the solution reoptimized. This objective ensures the stability of the reoptimization which is essential for the convergence of the interactive process toward a satisfactory solution. Besides these differences between the optimization model and the reoptimization model, the requirements in terms of computation time are more important for reoptimization than for the initial optimization process. Due to the interaction, the reoptimization should only take a few seconds to keep the user focused. To meet this requirement, the initial solution can be used as a starting point for the reoptimization. However, it should be noted that even with a good initial solution the complexity of a reoptimization problem generally remains the same as the initial optimization problem [2].

2.2 Shift scheduling

The proposed interactive reoptimization approach has been evaluated on a staff scheduling problem. This application domain is promising for applying interactive reoptimization and more generally it is an interesting domain for interactive optimization approaches due to the difficulty in modeling and solving the related optimization problems [3]. The fact that the solutions to staff scheduling problems directly impact employees' activities reinforces the need for interactions between the decision maker and the optimization system in order to adjust and validate solutions. In a real context, an optimization model for staff scheduling is likely to present some simplifications and omissions which could be solved using an interactive reoptimization approach.

The shift scheduling problem consists in assigning shifts (e.g. Early shift 7:00-15:00, Late shift 15:00-22:00) to employees for a given planning period. The result of the optimization is a roster defining the schedule of each employee. The constraints retained for the problem model are drawn from a problem proposed for the International Nurse Rostering Competition held in 2010 (INRC2010) [7]. The initial INRC2010 model contains about 20 different constraints but it has been simplified for the purpose of the experiment.

The problem model used for the generation of initial solutions contains three types of constraints, namely a hard constraint (H), work regulations ($R_1 - R_5$), and soft constraints ($S_1 - S_5$). The hard constraint ensures that the roster is complete:

- H *(Complete roster)* For each day in the planning horizon, the number of employees assigned to a shift must be equal to the demand. An employee can only have one shift assigned per day.

Work regulations are requirements specified by the contracts of the employees and that must be satisfied in a roster:

- R_1 *(Maximum number of assignments)* The total number of assignments of an employee within the whole planning period must not exceed a given maximum value.
- R_2 *(Minimum number of assignments)* The total number of assignments of an employee within the whole planning period must not be less than a given minimum value.
- R_3 *(Maximum number of consecutive working days)* The number of consecutive working days of an employee must not exceed a given maximum value.
- R_4 *(Incompatible shift sequences)* Some pairs of shifts cannot be worked on two consecutive days. For instance, an *Early* shift cannot be worked the day after an *Night* shift.
- R_5 *(Days-off)* No shift must be assigned to an employee for the days he has requested days-off.

Soft constraints are rules for improving the quality of employees' schedule and are not mandatory:

- S_1 *(Minimum number of consecutive working days)* The number of consecutive working days of an employee should not be less than a given minimum value.
- S_2 *(Maximum number of consecutive days-off)* The number of consecutive days-off of an employee should not exceed a given maximum value.
- S_3 *(Minimum number of consecutive days-off)* The number of consecutive days-off of an employee should not be less than a given minimum value.

S_4 *(Complete weekends)* Weekends should be either entirely worked or completely free.

S_5 *(Identical shifts during weekend)* During weekends completely worked, an employee should have the same shift assigned.

As we mentioned in the previous section, two constraints are added to the problem model for the reoptimization procedure. The first constraint (P) aims at integrating the changes requested by the user, the second constraint (D) minimizes the distance between the initial solution and the reoptimized solution.

P *(Preferences on assignments)* For adjusting a solution the user can specify preferred assignments (e.g. the user can indicate that *Early shift* and *Day-off* are preferred assignments for a given day and employee). A preference is satisfied when the assignment corresponds to one of the preferred assignments.

D *(Distance to initial solution)* The distance constraint penalizes any deviation from the initial solution that is reoptimized. An assignment in the solution that is different to the initial assignment corresponds to a penalty of one.

For the optimization and the reoptimization procedures, an order between these different sets of constraints is defined. The global objective, reported in Equation 1, is the minimization of the number of unsatisfied constraints using a lexicographic order. This means that the satisfaction of the hard constraint (H) takes priority over all other constraints. Then, the satisfaction of work regulations ($R_1 - R_5$) comes in second rank. If some preferences on assignments (P) have been defined by the user, they are less important than work regulations but have priority over soft constraints. The soft constraints ($S_1 - S_5$) are rank four, before the distance constraint (D) which is the less important objective.

$$minimize_{LEX}\left(f_H, \sum f_{R_i}, f_P, \sum f_{S_i}, f_D\right) \tag{1}$$

The use of a lexicographic order between different sets of constraints is justified by its simplicity and the fact that it is easily understandable by users. Unlike the INRC2010 problem model, it is not necessary to specify weights for each constraint. Thus, in the proposed model, one occurrence of an unsatisfied constraint corresponds to a cost of one unit in the related rank. The lexicographic order, which may appear complicated in the objective function, is in fact accessible for users without requiring particular knowledge in optimization.

2.3 Optimization procedures

The procedure used for the reoptimization of solutions is the same as the procedure that provides the initial solutions. It is an Iterated Local-Search (ILS) [9] which basically alternates between an improvement phase and a perturbation step for exploring the solution space with one solution. In this section we provide a general description of the optimization procedure and we refer the reader to [10] for additional details on the implemented ILS procedure.

ILS is a trajectory metaheuristic, which means that the exploration is made with one solution that “moves” in the search space. One iteration consists of, a perturbation step applied on the current solution to diversify the search and escape local-optima, then an improvement of the perturbed solution by local-search, and finally the application of an acceptance criterion to determine if the next iteration starts from the newly improved solution or from the last accepted solution. In the implemented ILS, the improvement phase is a Variable Neighborhood Descent (VND) [6]. The neighborhood structures used for the VND are based on block-swap moves. As illustrated in the left hand side of Figure 1, a move is an exchange between two employees of a block of consecutive assignments. The size of the blocks varies between 1 and 7 during the VND. The perturbation step applies random moves to the solution using a different neighborhood structure to that of VND. A perturbation move is illustrated in the right hand side of Figure 1. The moves for the perturbation are rotations of assignments' blocks between three employees. The advantage of such a rotation between three employees is that a perturbation move cannot be easily undone using block-swap moves. Concerning the acceptance criterion, a simple rule that only accepts improving solution has been adopted.

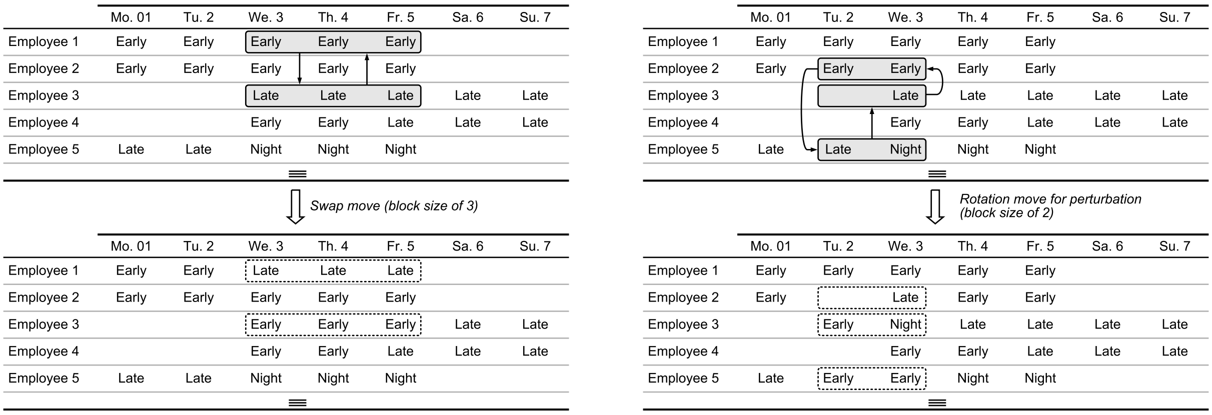

Figure 1: Illustration of moves used in the ILS procedure. On the left, block-swap moves are used in the VND procedure, and on the right, rotation moves are used in the perturbation procedure.

For the reoptimization, the performance of the procedure is a key element. As we mentioned previously, the reoptimization should last only a few seconds to be appropriate for an interactive context. For the experiment the duration of the reoptimization is fixed to 5 seconds. In order to meet this requirement, the implemented ILS procedure makes extensive use of delta-evaluation of solutions, and also exploits high-level parallelism. The parallel version of ILS is obtained by running multiple concurrent ILS procedures that exchange their best found solutions. This simple parallel strategy only aims at speeding up computation by using multiple threads. On the INRC2010 benchmark the results of the implemented ILS procedure are comparable to the results of state of the art metaheuristics [10].

3 Experiment

3.1 Objectives

The proposed experiment aims at evaluating the interactive reoptimization procedure presented in the previous section with real user interactions. For this evaluation, the results obtained by interactive reoptimization are compared to results of a manual adjustment of solutions, and also compared with ideal solutions which require no adjustments. This comparison between reoptimized solutions, manually adjusted solutions and ideal solutions should provide a measure of the cost gain obtained by the interactive reoptimization. In addition, the experiment should assess the impact of manual adjustment of solutions when it is not assisted by a reoptimization procedure. Finally, the comparison between reoptimized and ideal solutions should provide information on the limits of the proposed interactive reoptimization approach.

In a real decision context it would be difficult to evaluate quantitatively the interactive reoptimization approach. The inaccuracies of the optimization model that necessitate to adjust solutions are generally not clearly specified (otherwise these inaccuracies would be resolved in the optimization model). Therefore, it is difficult to determine ideal solutions, and the comparison between interactive reoptimization and manual adjustment would be delicate without knowing exactly if the compared solutions satisfy the same criteria. These obstacles are overcome in the experiment by controlling which aspects of the solutions need to be adjusted. More precisely, during the experiment, the users are asked to modify solutions according to given constraints. Thus, it is possible to determine if the expressed constraints are successfully integrated, and it becomes meaningful to compare solutions obtained by interactive reoptimization with manually adjusted solutions and ideal solutions.

The task analyzed during the experiment is the modification of an *initial roster* to satisfy a given constraint, called *missing constraint*. Subjects completed this task with different missing constraints and initial rosters, each representing a *scenario*. For instance, one of the scenarios consists in modifying a roster so that a particular employee has no shift assigned on Wednesdays. The scenarios are completed by subjects with two different methods, namely manual edition and interactive reoptimization. For both

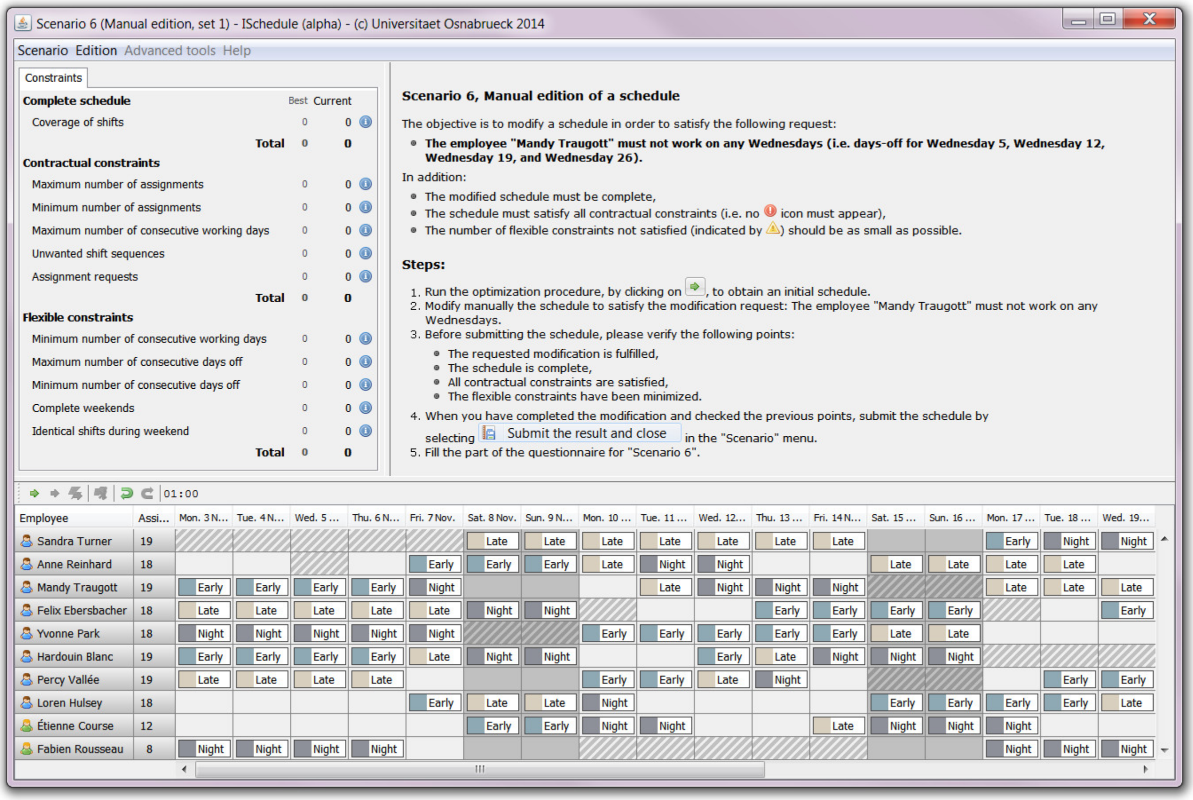

Figure 2: Graphical user interface of the software implemented for the experiment.

approaches, the Graphical User Interface (GUI) is the same (see Figure 2) but the means of interaction are limited to either manual adjustment actions or the use of interactive reoptimization tools. In order to compare the results of the two modes of interaction, subjects are asked to satisfy the missing constraints with the following priorities: First, the modifications must not impact work regulations constraints, then the missing constraint must be satisfied, and finally the number of unsatisfied soft constraint should be minimized. Thus, for scenarios where the missing constraint is satisfied, solutions can be compared on the basis of the cost of soft constraints.

3.2 Method

3.2.1 Subjects and procedures

The subjects for the experiment are voluntary undergraduate and graduate students of the University of Osnabrück. 16 subjects took part in the experiment. The course of the experiment is summarized in Figure 3. Each subject completed a session on manual edition and another session on interactive reoptimization. To minimize possible order effects, a group of subjects started with the interactive reoptimization task and the remaining subjects started with the manual edition task. For the two sessions, the task was explained to the subjects and two training scenarios were completed and verified by the experimenter before starting the evaluated scenarios. Then, the subjects completed 10 scenarios. For each scenario, the subject had 10 minutes for adjusting a provided roster according to a missing constraint. Then, the subject filled out a short questionnaire in which it is asked if the task has been understood and also how the subject evaluate his/her own performance.

3.2.2 Details of scenarios

The 10 scenarios that are completed during the manual edition task are the same as the scenarios completed using interactive reoptimization. The list of the respective missing constraints is given in Table 1. To simplify the task for the subjects, all scenarios are based on similar shift scheduling problems with

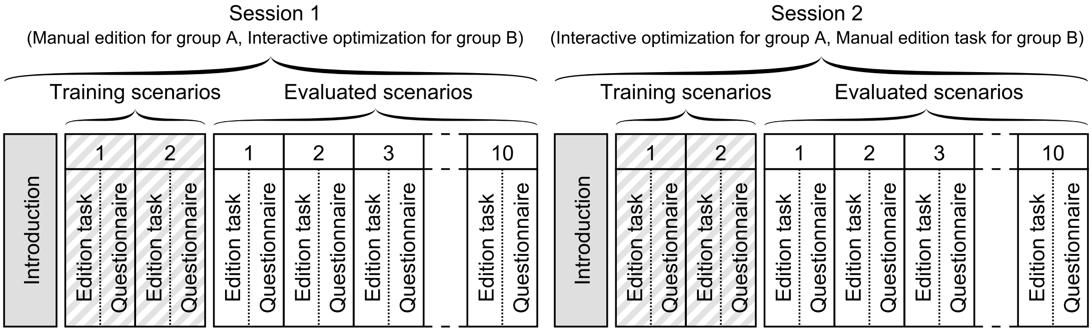

Figure 3: Course of the experiment.

small variations in day off requests and in initial rosters. However, it should be noted that scenarios completed by a user are paired between the manual edition session and the interactive reoptimization session (i.e. for a scenario completed by a subject with manual edition the exact same problem and initial roster is used for the related scenario in interactive reoptimization). The variations in problems and in initial rosters should ensure that average results per scenario are not altered by specific configurations of initial rosters.

Since the subjects are not experts in scheduling, the problems have been simplified for being tractable without prior knowledge in shift scheduling. Only 10 employees are considered, with three possible shifts (*Early*, *Late* and *Night*) and two types of contract (*Full-time* and *Part-time*). Also for simplifying the task, the initial rosters that are optimal pre-computed rosters satisfy all work regulations and only one or no soft constraint is unsatisfied. Thus, subjects do not have to put much effort on the satisfaction of work regulation constraints and can essentially focus on the satisfaction of the given missing constraint and the minimization of unsatisfied soft constraints.

Scenario	Missing constraint
1	The employee *E1*[a] must not work on the first weekend.
2	The employees *E2* and *E3* must work only on Early shifts for the whole planning period.
3	The employee *E4* must not work on *D1* and *D2*.
4	The employee *E5* must not work the second and fourth weekends.
5	The employee *E6* must work five consecutive days from *D3*.
6	The employee *E7* must not work on any Wednesdays.
7	Part-time employees must not work on Night shifts.
8	For part-time employees, the Monday must be free after a working Sunday.
9	For all employees, no Night shift must be assigned on Friday when the weekend is free.
10	For all employees, a Night shift must not be assigned the day after an Early shift.

[a]For the experiment, the codes for employees (*E#*) are replaced by generated names, and days (*D#*) by real dates.

Table 1: List of the missing constraints that have to be integrated by subjects in initial rosters.

3.2.3 Interactions

As previously mentioned, the same GUI presented in Figure 2 is used for both the manual edition task and the interactive reoptimization task. It is composed of three panels. The upper right panel displays the instructions. When the subject loads a scenario, this panel indicates the missing constraint to introduce and recalls the steps for modifying the initial roster. The upper left panel gives a summary of the constraints unsatisfied for the current roster. The subject can also obtain a description of the constraints and their parameters from this panel. When the roster is modified, the values for unsatisfied constraints are instantaneously updated. Finally, the lower panel display the current roster. All unsatisfied constraints are represented directly on the affected assignments and the subject can obtain a precise description of them by selecting the assignments. When a modification of the roster is made, by manual edition or reoptimization, the roster and all indications concerning unsatisfied constraints are instantaneously updated. In addition, the last modifications made on the roster are highlighted to easily keep track of the changes. For modifying the roster, the subject directly interacts with the assignments of the roster. The actions available for the manual edition task and the reoptimization task are as follows.

For manually editing a roster, the subject can swap assignments by drag-and-drop between any employee. In addition, assignments can be removed and then reassigned to any employee using drag-and-drop. The list of shifts for which the demand is not fulfilled appears above the corresponding days. Any modification made on the roster can be undone and redone using buttons in the toolbar of the roster.

For the interactive reoptimization task, only the reoptimization tools are enabled and the roster cannot be directly modified using drag-and-drop of assignments. The subject can set assignment preferences using a contextual menu. A right mouse-click on an assignment opens a menu in which the subject can select the preferred and unwanted assignments (that includes the possibility to set a preference for a day-off). In addition, it is possible to change or clear assignment preferences at any time. The assignments for which a preference has been defined are indicated on the roster. To introduce the desired changes (i.e. reflect assignment preferences) the subject clicks on a reoptimization button in the toolbar of the roster. The roster is then reoptimized for 5 seconds and the changes are dynamically displayed. As for the manual edition, the user can undo and redo any change, including the definition of assignment preferences and the reoptimization, using buttons in the toolbar of the roster.

4 Results

During the experiment, the rosters obtained by subjects using manual edition and interactive reoptimization are compared to ideal solutions. These ideal solutions correspond to rosters that optimally express the missing constraints (i.e. satisfy the work regulations, the missing constraint and have the minimum possible number of unsatisfied soft constraints). Ideal solutions are computed for each scenario using the ILS procedure applied on a model that contains the missing constraint. Optimality has been verified using global lower-bound values.

For the results obtained by the subjects, only the solutions that satisfy the work regulations and the missing constraints are considered. When the missing constraint is not satisfied in a roster adjusted by a subject, it is not possible to determine if it is due to a misunderstanding of the subject, the result of inattention, or related to the difficulty of the task. Thus, we removed those results to only compare successful adjustments of rosters.

Since all of the compared solutions satisfy the work regulations and the missing constraints, the *cost* of a solution is defined as the number of soft constraints unsatisfied. Table 2 reports the average cost difference between solutions obtained by subjects and ideal solutions. In addition, the average duration of the adjustment process is reported. This duration correspond to the time from the first interaction with the initial roster to the end of the last action performed by the subject. For the interactive reoptimization task, this duration includes the time taken by the reoptimization procedure.

	Average cost difference to optimal[b]		Average duration[b] (*seconds*)	
Scenario	Manual edition	Interactive reoptimization	Manual edition	Interactive reoptimization
1	5.20	**0.81**	396	**36**
2	5.90	**1.62**	393	**294**
3	5.00	**0.67**	502	**51**
4	6.29	**1.71**	510	**90**
5	2.38	**0.33**	291	**63**
6	9.00	**2.47**	501	**80**
7	0.33	**0.00**	**93**	98
8	5.33	**1.87**	332	**88**
9	**0.44**	0.67	**148**	161
10	**0.50**	0.70	**234**	242
Average	4.04	**1.08**	340	**120**

[b]Excluding results for which the missing constraint has not been successfully introduced in the initial roster.

Table 2: Average results for the manual edition task and the interactive reoptimization task.

The results in Table 2 confirm that the interactive reoptimization approach is globally more efficient than manual edition for adjusting solutions. But more importantly, these results provide a quantitative evaluation of the impact of manual adjustment of solutions. The manual edition of solutions introduces on average 4.04 unsatisfied soft constraints where the interactive optimization produces on average 1.08 unsatisfied soft constraint. The worst scenario for both manual edition and interactive reoptimization is the scenario 6 for which the average cost differences to optimal values are respectively 9 for manual edition and 2.47 for interactive reoptimization. For scenarios 1 and 3, where only two assignments have to be changed in the initial roster, the manual edition introduces respectively 5.2 and 5 unsatisfied constraints. These results show the substantial impact of manual adjustment of solutions on their quality. In addition, these comparative results illustrate the gains in terms of cost and duration achieved by the interactive reoptimization approach.

A closer look at the results reveals few cases where the interactive reoptimization approach provides worse results than the manual edition setting. For scenarios 9 and 10, the average costs of solutions obtained by manual edition are better than the average costs achieved by interactive reoptimization. It should be noted, however, that these average costs are below 1 for both manual edition and interactive reoptimization. Regarding the average durations, the average times for completing scenarios 7, 9 and 10 are slightly better for manual edition than for interactive reoptimization. These cases where the interactive reoptimization approach has similar or inferior results than the manual edition setting does not question the overall effectiveness of the interactive reoptimization but expose some limits of the implemented approach. These limits are summarized below.

Computational performance of ILS: The reoptimization of rosters is a challenging optimization problem, in particular with the limited computation time. Metaheuristics such as ILS appear to be appropriate for reoptimization thanks to their capacity to provide good solutions in a reasonable time. However, there is some room for improving the implemented ILS and obtaining better solutions.

Expressiveness of preferences: In the proposed interactive reoptimization method, the user can only define preferences related to single assignments. When, in scenarios 9 and 10, it is asked to adjust a roster according to preferred or unwanted sequences of shifts, the preferences on assignments become less efficient. In fact for these two scenarios, when a preference is set for adjusting an inadequate sequence of shifts, the reoptimization affects the rest of the solution and may reintroduce the inadequate sequence at another place of the roster. In this case, it is necessary to proceed with multiple iterations of preference definition and reoptimization, although the distance constraint tends to reduce the number of iterations. This could result in the definition of assignment preferences that overconstrain the problem (e.g. the initial assignment preferences may no longer be necessary after several iterations) and thus impair the quality of solutions. The design of additional tools for defining preferences seems to be necessary to address this problem.

Interactions: Finally, it should be noted that for the interactive reoptimization task the subjects had no means to manually adjust the roster, and in some cases the visualization of the roster allows the subjects to identify efficient moves that are hardly made by the ILS. The good performances on the manual edition task for scenarios 7, 9 and 10 shows that in particular cases it could be interesting to exploit user heuristics for improving solutions. A possible approach would be to combine both manual and interactive approaches for adjusting solution, but it raises the problem that a user may be reluctant to use the interactive reoptimization approach. In this direction, it seems necessary to study the usability and acceptance of methods that combines multiple interaction mechanisms.

5 Conclusion and perspectives

In this paper, we proposed an interactive reoptimization method for adjusting solutions when some inaccuracies in an optimization model need to be solved by the user of the optimization system. This interactive process is studied for a shift scheduling problem. The method proposed for reoptimizing solutions and integrating changes requested by a user is an Iterated Local Search (ILS) procedure. We proposed an experiment to evaluate this interactive reoptimization approach. The experiment was conducted with 16 subjects and 10 different scenarios. The results of the interactive reoptimization approach were compared with solutions obtained by manual adjustment and with ideal solutions. This comparison shows the value of a global optimization method such as the ILS procedure for integrating efficiently some changes in a solution. In addition, the results of the experiment demonstrate the impact of manual adjustment of solutions on their quality. Finally, this experiment revealed some limits of the implemented interactive reoptimization method. The perspectives of this work are directly connected to the observations made on the results of the experiment. Further works will concern the improvement of performance of the reoptimization procedure, the investigation of additional interaction means, and also will address usability and acceptance of the proposed interactive approach.

Acknowledgments: This work was supported by the Deutsche Forschungsgemeinschaft (DFG), under grant ME 4045/2-1, for the project "Interactive metaheuristics for optimization-based decision support systems".

References

[1] David Anderson, Emily Anderson, Neal Lesh, Joe Marks, Brian Mirtich, David Ratajczak, and Kathy Ryall. Human-guided simple search. In *AAAI 2000*, pages 209–216, 2000.

[2] Giorgio Ausiello, Vicenzo Bonifaci, and Bruno Escoffier. *Computability in Context: Computation and Logic in the Real World*, chapter Complexity and Approximation in Reoptimization, pages 101–129. Imperial College Press, 2007. ISBN: 978-1-84816-245-7.

[3] A. T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1):3–27, 2004.

[4] Guisseppi A. Forgionne. *Decision Making Support Systems: Achievements and Challenges for the New Decade*, chapter An Architecture for the Integration of Decision Making Support Functionalities, pages 1–19. 2002.

[5] Simon Hamel, Jonathan Gaudreault, Claude-Guy Quimper, Mathieu Bouchard, and Philippe Marier. Human-machine interaction for real-time linear optimization. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 673–680, 2012.

[6] Pierre Hansen and Nenad Mladenović. *Handbook of Metaheuristics*, chapter Variable Neighborhood Search, pages 145–184. Kluwer Academic, 2003.

[7] Stefaan Haspeslagh, Patrick De Causmaecker, Andrea Schaerf, and Martin Stølevik. The first international nurse rostering competition 2010. *Annals of Operations Research*, 218(1):221–236, 2014.

[8] Gunnar W. Klau, Neal Lesh, Joe Marks, and Michael Mitzenmacher. Human-guided search. *Journal of Heuristics*, 16(3):289–310, 2010.

[9] Helena R. Lourenço, Olivier C. Martin, and Thomas Stützle. *Handbook of Metaheuristics*, chapter Iterated Local Search: Framework and Applications, pages 363–397. Springer, 2010.

[10] David Meignan. A heuristic approach to schedule reoptimization in the context of interactive optimization. In *Genetic and Evolutionary Computation Conference*, pages 461–468. ACM, 2014.

[11] Michael L. Pinedo. *Scheduling Theory, Algorithms, and Systems*, chapter Design and Implementation of Scheduling Systems: Basic Concepts, pages 459–483. Springer, fourth edition, 2012.

[12] Hadas Shachnai, Gal Tamir, and Tami Tamir. A theory and algorithms for combinatorial reoptimization. In *LATIN 2012: Theoretical Informatics*, volume 7256 of *Lecture Notes in Computer Science*, pages 618–630. Springer Berlin Heidelberg, 2012.

[13] J.P. Shim, Merrill Warkentin, James F. Courtney, Daniel J. Power, Ramesh Sharda, and Christer Carlsson. Past, present, and future of decision support technology. *Decision Support Systems*, 33(2):111–126, 2002.

[14] Anna Zych. *Reoptimization of NP-hard problems*. PhD thesis, Eidgenössische Technische Hochschule ETH Zürich, 2012. Nr. 20257.