Usage of Constraint Programming for Nurse Rostering Problems
- A literature study

by

Oliver Strömgren

LIU-IDA/LITH-EX-G--15/039--SE

2015-06-22
På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Oliver Strömgren
Usage of Constraint Programming for Nurse Rostering Problems
- A literature study

by

Oliver Strömgren

LIU-IDA/LITH-EX-G--15/039--SE

2015-06-22

Supervisor: Ola Leifler
Examiner: Ola Leifler
Abstract

Constraint Programming can be used to solve many problems and this thesis is about getting an overview on the usage of Constraint Programming for Constraint Satisfaction Problems, both interactive and explorative. Many problems can be mathematically modeled as a Constraint Satisfaction Problem but this thesis will focus on the Nurse Rostering Problem since it is a well-studied area. The problem when creating a schedule for nurses is that it can easily be over-constrained and a solution could be hard to find. This thesis will investigate whether if Constraint Programming is a good technique for solving the Nurse Rostering Problem but also if user interaction is considered when solving the problem. The method for this is a literature study where a number of research articles has been reviewed and categorized, and resulted in 27 different kinds of sources that were used. The conclusion is that there exists better ways to solve these problems than the use of pure Constraint Programming. To answer the second part of the thesis, it seems like the solution for the problem is the main focus and therefore is user interaction something that is given less attention.

Keywords: Constraint Programming; Nurse Rostering Problem; User interaction
# Table of Contents

1. **Introduction** ................................................................................................................. 1  
   1.1 Motivation ..................................................................................................................... 1  
   1.2 Purpose .......................................................................................................................... 1  
   1.3 Problem definition ......................................................................................................... 2  
   1.4 Limitations .................................................................................................................... 2  
   1.5 Background .................................................................................................................. 2  

2. **Theory** ............................................................................................................................. 3  
   2.1 Nurse Rostering Problem .............................................................................................. 3  
      2.1.1 Constraints ............................................................................................................. 3  
      2.1.2 Modeling ................................................................................................................ 4  
   2.2 Constraint Programming .............................................................................................. 4  
      2.2.1 Constraint Logic Programming .............................................................................. 6  

3. **Method** ........................................................................................................................... 7  
   3.1 Limitations .................................................................................................................... 7  
   3.2 Making the study .......................................................................................................... 7  
      3.2.1 Step one – Preliminary research ........................................................................... 8  
      3.2.2 Step two – Primary search .................................................................................... 8  
      3.2.3 Step three – Secondary search ............................................................................ 9  
      3.2.4 Step four – Categorize the results ....................................................................... 9  

4. **Result** ............................................................................................................................. 11  
   4.1 Step one ....................................................................................................................... 11  
   4.2 Step two and three ...................................................................................................... 11  
   4.3 Step four ..................................................................................................................... 12  
      4.3.1 The Constraint Programming approach ............................................................... 13  
      4.3.2 The Constraint Logic Programming approach ................................................... 13  
      4.3.3 The hybrid approach ............................................................................................ 14  

5. **Discussion** ...................................................................................................................... 17  
   5.1 Result ......................................................................................................................... 17  
   5.2 Method ....................................................................................................................... 18  

6. **Conclusion** ...................................................................................................................... 19  

Reference list ......................................................................................................................... 20
List of Figures

Figure 1. Variable and constraints example for the given problem ......................................... 5
Figure 2. Method overview ........................................................................................................ 7
Figure 3. Filtering overview ....................................................................................................... 9
Figure 4. Categorization overview ............................................................................................ 10
Figure 5. Number of articles found ............................................................................................ 11

List of Tables

Table 1. Most commonly used constraints in different papers [4] .............................................. 3
Table 2. Part of a Nurse-day view, M = morning, A = afternoon, N = night O = day off .......... 4
Table 3. Problem example, nurses’ requirements ...................................................................... 5
Table 4. A solution to the example problem .............................................................................. 6
Table 5. Keywords used in digital libraries .................................................................................. 8
Table 6. Result from step one ..................................................................................................... 11
Table 7. Articles overview .......................................................................................................... 12
Table 8. Categorization of articles ............................................................................................. 12
# Glossary

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSP</td>
<td>Constraint Satisfaction Problem, a problem where a set of objects needs to satisfy a number of constraints.</td>
</tr>
<tr>
<td>CP</td>
<td>Constraint Programming, a programming paradigm. The programming is done by creating relations between variables stated as constraints.</td>
</tr>
<tr>
<td>CLP</td>
<td>Constraint Logic Programming, CP integrated with logic programming.</td>
</tr>
<tr>
<td>HCLP</td>
<td>Hierarchical Constraint Logic Programming, a subgroup to CLP. If constraints are weighted, have costs, they can be sorted in groups, i.e. hierarchies.</td>
</tr>
<tr>
<td>NRP</td>
<td>Nurse Rostering Problem, the problem of creating rosters (schedules) for, in this case, nurses. It typically involves solving a lot of constraints.</td>
</tr>
<tr>
<td>NSP</td>
<td>Nurse Scheduling Problem, same as NRP.</td>
</tr>
<tr>
<td>ILP</td>
<td>Integer Linear Programming, in ILP the objective functions and the constraints are linear. The variables used can only be integers.</td>
</tr>
</tbody>
</table>
1. Introduction

In this chapter an introduction of the area and the reason for doing this thesis will be brought up.

1.1 Motivation

Constraint Programming (CP) can be used to solve many problems and this work is about getting an overview on the usage of CP for Constraint Satisfaction Problems (CSPs), both interactive and explorative. One well studied area where this is applicable is the Nurse Rostering Problem (NRP) or Nurse Scheduling Problem (NSP). That is why this work has a health care approach.

In health care the scheduling of nurses has always been a hard task. Because of the many requirements that exist when operating a hospital. One typical way to do this is to make a schedule by hand, which is very time consuming. Another way is to write a program that creates schedules automatically for you. The hard thing in this is to create a good model for the problem and a solver for solving it.

One way to model the NRP is to model it as a CSP and this kind of model is typically good for CP [1][2]. When a problem is modeled as a CSP, there exist a lot of constraints in the problem that need to be fulfilled. The large number of constraints is also the reason why to model it as a CSP. When the problem is being modeled it may happen that the model gets over-constrained and cannot be solved. When this happens in a fully developed system that is in use, the interaction between the system and user has to work in a good way. If the usability of a system is good the satisfaction factor will probably be higher.

Today it seems like these systems that are supposed to solve the scheduling problem at a hospital automatically are not very well thought-out [3]. The user seems to have no control over the model when too many constraints are added. Therefore it would be interesting to investigate this further, by the means of how the results are when CP is used for solving the NRP. It would also be interesting to know how these systems are used in practice. If the nurses, in this case, can have their personal wishes added to the model, or if they need to accept the schedules as they are.

1.2 Purpose

The purpose of this work is to do a literature study on existing work on CP for solving the NRP. Aiming for what the results are and how the interaction part in a working system is solved. This is also done as a preparatory work to ease for, hopefully, a bigger project that will be done later by Master’s Students.
1.3 Problem definition

- Based on the results of existing work on NRP, are CP a good approach for solving the problem?
- Is user interaction considered in the existing work?

With a good approach in the first question is good referring to if CP can find a solution in a feasible amount of time, also if the schedules that are created are of quality for the nurses.

1.4 Limitations

Scheduling problems is a big problem and can be applied in many areas, not just for nurses. Therefore will this work only focus on the NRP. Also when solving the NRP many techniques can be used but the main focus will be the CP technique. If other techniques different from what this work is about is mentioned, the reader will not be able to learn about them in this paper. This decision was made because the purpose of the thesis is not about learning different techniques.

1.5 Background

This thesis was advertised by my examiner, Ola Leifler, at IDA on LiU. I took this work because I found the subject interesting, both in a technical and societal aspect. Another reason was that I had never done a literature review before, I saw a good opportunity to learn this in a good way.
2. Theory

The theory behind NRP, CP and other definitions will be presented and explained in this chapter.

2.1 Nurse Rostering Problem

The NRP is a scheduling problem, the problem to create schedules for a given period. Example for a day, week or longer. In some cases a schedule is created based on tasks or different patterns [4]. It is not only a problem that arises for the scheduling of nurses, it is a problem when scheduling personnel of other types as well. Worth mentioning is that this sort of problem has been addressed in different kind of fields for over 40 years [5] and is classified as a NP-hard problem [6][7]. The solution for NRP's may have the same goal but the constraints in the problem may differ from hospital to hospital, based on their own needs. Therefore is it hard to find a general model and solution.

2.1.1 Constraints

When creating a schedule many necessities need to be considered. The hospital has its requirements for assigning nurses to different shifts. For example, the hospital may want X number of nurses with expertise A and Y number of nurses with expertise B in one shift. A number of shifts need to be filled each day, every day of a week, every week of a month and so on. Depending on the number of patients at the time, the hospital need Z nurses. And, of course, nurses are not allowed to work too long in a row. At the same time the economy needs to be taken into account. While the hospital needs to consider this, the nurses themselves have their requirements. Maybe some nurses may not want to work nights, some might not want to work in the weekends. Sometimes they want to take vacations or they need a day off because they worked too much. These requirements goes on and on, on both sides, and building up a lot of constraints that needs to be satisfied to create a good schedule. It is of high importance to create a good schedule because of the impact it has on the quality on the health care [5][8]. In the paper by Cheang (2003) et al. [4] they map the most commonly used constraints, see Table 1 below. These are just a few examples, but it is easy to see how this problem can be over-constrained and hard to solve.

These constraints that exist can be split up in hard and soft constraints [5][9]. Constraints that belong to the hard category are the ones that usually belong to the hospital. Like policy rules, number of nurses and so on. Hard constraints are often constraints that must be satisfied and cannot be violated. Soft constraints are those that usually belong to the nurses, their personal wishes. These are not needed to be satisfied but they are desirable, they can be violated if not a solution can be found [5].

<table>
<thead>
<tr>
<th>Common constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nurses workload, Consecutive working shift/days, Nurses’ preferences or requirements, Nurses free days, Free time between working shifts, Shift types(s) assignments, Working weekend, Other requirements in a shorter or longer time period other than the planning period, Requirements of (different types of) nurses or staff demand for any shift.</td>
</tr>
</tbody>
</table>

Table 1. Most commonly used constraints in different papers [4]
2.1.2 Modeling

NRP can be mathematically modeled and solved in different ways [4], the models can be big because of the many constraints that exist in a problem. The most common way to describe the problem is by a nurse-day view, or probably the one most people think of when thinking of a schedule, which is a two-dimensional array where the dimensions are days of the week and the nurses names or IDs. Below is an example of a nurse-day view, Table 2, where the shifts are: M = morning, A = afternoon, N = night and O = day off. Other descriptions are nurse-task view which is almost like nurse-day view but instead of shifts to fill on each day, tasks on each day are filled, and nurse-pattern view. The nurse-pattern view is the most different one of these three, as it is built from patterns the nurses are working. An example pattern for a nurse can be: work evening every day except Saturday and Sunday.

NRP can be modeled as a CSP, which is most suitable when there are a lot of constraints. This is almost always the case and is as mentioned earlier desirable when working with CP. A CSP is defined as a set of variables that can take a finite set of values, and all the possible values a variable can take is called the variable’s domain [10]. The relations that exists between the variables are the constraints that are meant to be solved. A CSP is considered solved when one solution or more is found [11], which means that a solution to the NRP does not have to be optimal. Another typical example of a CSP, a bit smaller than the NRP, is the Map-Coloring problem, which is the problem to paint a map where adjacent regions must have different colors.

When a better optimized solution to the NRP should be found, costs added to the constraints in the model is usually the approach used. This is done after how important they are, either the higher the cost the more important it is, it could also be the other way around. In this way the constraints are in some way called weighted, some constraints will weigh more than others. If some constraints are being violated the greater the cost will be, or lower depending on how it is done. In this way different solutions can be compared to each other, the one with the lowest or highest cost will be the best solution.

<table>
<thead>
<tr>
<th>Nurse ID</th>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
<th>Saturday</th>
<th>Sunday</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M</td>
<td>A</td>
<td>N</td>
<td>A</td>
<td>A</td>
<td>M</td>
<td>O</td>
</tr>
<tr>
<td>2</td>
<td>A</td>
<td>N</td>
<td>A</td>
<td>A</td>
<td>O</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>3</td>
<td>N</td>
<td>A</td>
<td>M</td>
<td>M</td>
<td>N</td>
<td>O</td>
<td>A</td>
</tr>
</tbody>
</table>

Table 2. Part of a Nurse-day view, M = morning, A = afternoon, N = night O = day off

2.2 Constraint Programming

CP is a programming paradigm and one field that CP can be applied to is Combinatorial Search Problems [11]. It is applied to many areas, such as scheduling. It is a different way of thinking when coding compared to Imperative Programming for example. In CP the user, programmer, models the problem as a CSP and the computer tries to solve it. When the problem is being modeled the programmer does not have to assign every variable a value, as in Imperative Programming, since the whole idea is to work with relations, i.e. constraints.

To solve these kind of problems, a constraint solver is used to solve it. A constraint solver can take a real-world problem and it will try to find an assignment to all the variables that will satisfy the existing constraints. A constraint solver is generally doing this with a search algorithm and constraint propagation [10]. Constraint propagation is a kind of filtering, the purpose is to remove those values in a variables domain that is not in the solution of a constraint. A simple example would be the constraint $x < y$ and the domain of $x$ is $(2, 3, 4, 5, 6, 7)$ and for the domain of $y$ is $(0, 1, 2, 3, 4, 5)$. After the propagation the domains will be $(2, 3, 4)$ and $(3, 4, 5)$ respectively.
Of course it exists a lot of algorithms, different algorithms are better for different models and problems [10]. This is the hard thing about CP, to create a good model that works with a solver [11]. It is not that easy to create this model from a real world problem, because of the many constraints that exists. CP is often used with rule-based programming languages, e.g. Prolog and Wolfram/Mathematica, but can be used with other languages like C++ and Java as well. For the last two mentioned languages toolkits or libraries are needed, since CP is not a part of the original programming language. It exist different kinds but two of them that seems to be popular are ECLiPSe which is an open-source software system [12] and Gecode which is a toolkit [13].

To give a small example on how this could work, consider just Monday to Wednesday in the nurse-day view from Table 2 above. Let us say that the nurses can work as stated in Table 3 below, where M, A, N and O means the same thing as in Table 2. The goal for this very small example is to assign one nurse for each shift on each day, when having the restrictions: only one nurse can work on a shift at a time, a nurse can only work one shift per day, a nurse cannot be assigned shift M after shift N or shift A after shift N. At the same time the nurses’ requirements should be considered from Table 3. For solving this the backtracking method can be used. The backtracking will try to assign values to the variables as far as it can. When a variable is assigned an invalid value the algorithm will step back and try another possible value for the variable, if a value is left in its domain.

<table>
<thead>
<tr>
<th>Nurse ID</th>
<th>Type of shift per day</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Monday: M, A</td>
</tr>
<tr>
<td></td>
<td>Tuesday: M, A</td>
</tr>
<tr>
<td></td>
<td>Wednesday: A</td>
</tr>
<tr>
<td>2</td>
<td>Monday: M, A</td>
</tr>
<tr>
<td></td>
<td>Tuesday: M, N</td>
</tr>
<tr>
<td></td>
<td>Wednesday: M, N</td>
</tr>
<tr>
<td>3</td>
<td>Monday: N</td>
</tr>
<tr>
<td></td>
<td>Tuesday: A, N</td>
</tr>
<tr>
<td></td>
<td>Wednesday: A, N</td>
</tr>
</tbody>
</table>

Table 3. Problem example, nurses’ requirements

$V_{i,j}$ is the shift type assigned to nurse i on day j.

The shift types are (the variables’ domain):

$M = \text{morning}, A = \text{afternoon} \text{ and } N = \text{night}$

Some constraints from the problem:

- if $V_{i,j} = N \rightarrow V_{i,j+1} \neq M$ or $V_{i,j+1} \neq A$
- if $V_{i,j} = M \rightarrow V_{i,j} \neq A$ or $V_{i,j} \neq N$
- if $V_{i,j} = A \rightarrow V_{i,j} \neq M$ or $V_{i,j} \neq N$
- if $V_{i,j} = N \rightarrow V_{i,j} \neq M$ or $V_{i,j} \neq A$

Figure 1. Variable and constraints example for the given problem
To give an example of an over-constrained problem consider a modification of the presented one. To this example let us add another nurse, now a total of four nurses, and at the same time add the requirement: every nurse needs to work one shift per day. This is clearly not possible because it only exists three shifts per day, therefore will this problem become over-constrained. This is of course a very simplified example but a common way to avoid creating over-constrained problems in real life is by breaking some soft constraints. Another approach is by adding costs to the constraints, as mentioned in 2.1.2. This way a solution will be found but it will be better or worse.

### 2.2.1 Constraint Logic Programming

Constraint technology has been integrated into several programming languages and paradigms [11]. One integration is Constraint Logic Programming (CLP), which is the integration of constraint solving into constraint logic languages. What makes this integration good is that constraint solving and logic programming is both declarative paradigms [11]. This schema gives a generic way of creating a CLP language [10]. All that is needed is the constraints, the solver for them and a rule-based language, for example Prolog.

#### 2.2.1.1 Hierarchical Constraint Logic Programming

A subgroup to CLP is something called Hierarchical Constraint Logic Programming (HCLP) [14]. This practice are focusing on the constraints in another way than CP and CLP does. In HCLP the constraints are grouped in different sets and the sets are sorted in hierarchies, the hierarchies are sorted after importance. In this way a decision on how many levels of hierarchies that a solution should support can be done. This is a preferred way if the constraints have different costs. Usually the solver for these problems tries to maximum or minimum, depending on the formulation of the problem, the costs of the constraints [15].
3. Method

For the method in this work guidelines from the paper by Keele (2007) [16] was used. Although this study is not of the same nature, i.e. literature review, parts could still be used for the research part in this work.

3.1 Limitations

It existed some limitations on when and what to study and search on. Since this work will study, in some way, the state of the art of NRPs solved with CP, existing work after year 2004 was mainly focused. Also, even if NRP is a scheduling problem and therefore searching for scheduling problems in general could be possible, it was not done. The search area would be too big for one person and the time would not be enough.

3.2 Making the study

The method is divided into several steps. One reason for this was to work in an iterative manner, another, or due to work iteratively, to get better results. It is step two and three that are done iteratively.

Figure 2. Method overview
3.2.1 Step one – Preliminary research

In the first step both research over the Internet and research in books were done. This was done to get more knowledge of the field. Since this area was new to me I needed to get a bigger understanding of the main problem of NRP and the CP technique. Most of the research to gain knowledge of CP was mainly done in books. The overview articles were mainly used to get knowledge about the NRP. To find these overviews the keywords “Nurse Rostering Problem” and “Constraint Programming” were used, also mixed together. This step was also done in purpose to get a bigger vocabulary. I needed to get a bigger vocabulary to get a broader and better spread on the search keywords in step two, these are shown in chapter 3.2.2. Also to get synonyms on keywords to work with.

3.2.2 Step two – Primary search

In this second step the real search for answers on the research questions was done. Four digital libraries, or search engines, was chosen: Google Scholar, IEEE Xplore digital library, Springer and the digital library on Linköping University’s website that uses the engine UniSearch [17].

<table>
<thead>
<tr>
<th>Keywords used</th>
</tr>
</thead>
</table>

Table 5. Keywords used in digital libraries

All four libraries were used for this. The same keyword, or combination of keywords, was used in each to get a bigger search result.

The search results needed to be filtered in some way. On the digital libraries modifications was made, if it was doable. The modifications made was the restriction of years that the engine should search between, it was between 2004 and 2015. When this was done, additional filtering needed to be done to get the most relevant results. Taking advice for filtering in a good way from the paper by Keele (2007) [16], some criteria was developed. The results needed to include one or more of these terms:

- Constraint Programming
- Nurse Rostering Problem, or synonyms
- Constraints, either soft, hard or both
- Usability or User interaction
- Real-world case

The filtering was done manually and performed as shown in Figure 3.
First the filtering were done on titles, the most relevant ones were chosen. Of the ones chosen the abstract and conclusion was read. If the paper, or different kind, was still relevant the introduction was read. Last, if it was relevant the whole way, the entire article was considered. If a paper, or whatever it was, went through the whole filtering process, notes were used to write down/document main points of it.

### 3.2.3 Step three – Secondary search

Additional searches was done through the results in step two. Of those which went through the whole process, their own references were used. This was done by checking in the reference section.

Another approach used was Google Scholar’s Cite function. The results from step two was used again. The Cite function shows which articles, or work on other forms that are citing a work. Because of the questions that are meant to be answered, about the usability and user interaction, not the most cited articles were focused. The reason is because this particular area is not so well studied.

The filtering in this step was conducted as in step two, with a gentler filtering by titles. The reason for this was that they simply were quoted in already used articles. Same goes for the limitation on which year it was done or published.

### 3.2.4 Step four – Categorize the results

After the search process, a categorization, or grouping, of the articles was done. The articles were divided into different groups. The categories were of different types, with this I mean that they created some sort of hierarchy, this is shown in Figure 4. An overview of the categories:

- Type of technique for the solution to the NRP
- If personal whishes were included, i.e. soft constraints
- User interaction
- Graphical interface
Usage of Constraint Programming for Nurse Rostering Problems – A literature study

Figure 4. Categorization overview
4. Result

This chapter will be divided in the same subheading as in the previous chapter, in this way the results after each step will be clearly presented. One difference is that the result from step two and three will be presented under the same subheading, simply because the whole result is more interesting.

4.1 Step one

From step one, five mediums in total were used, see Table 6 below. The books helped with the understanding of CP, including one article on HCLP, and the articles helped with the understanding of NRPs. They also helped to build a bigger vocabulary hence better keywords could be used. The keywords were represented in chapter 3.2.2, Table 5.

<table>
<thead>
<tr>
<th>Medium</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Book</td>
<td>[10], [11]</td>
</tr>
<tr>
<td>Article</td>
<td>[4], [5], [18], [14]</td>
</tr>
</tbody>
</table>

Table 6. Result from step one

4.2 Step two and three

These two steps produced an amount of 58 articles after the filtering on titles was done. After the whole filtering process was complete the amount ended up on 21 articles. Some of these articles are used to help build up the introduction- and theory chapter, while others are used in the discussion chapter later.

![Figure 5. Number of articles found](image)
Usage of Constraint Programming for Nurse Rostering Problems – A literature study

The articles are of different natures, some of them are case studies in the real world and others are research articles, done more for academic reasons. Of these 21 articles most of them were of an academic approach, with an academic approach I mean that the articles are about development and experimental evaluation of algorithms for the NRP. A mini goal was to find as many with the real world examples as possible, because of the question on usability this work was meant to try to answer. Worth mentioning is that some of the articles with an academic approach are using benchmark sets or/and real world data when they are trying their solutions.

<table>
<thead>
<tr>
<th>Academic</th>
<th>[19], [7], [2], [20], [21], [22], [23], [6]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use case / Used in the real world</td>
<td>[1], [15], [24], [25], [26], [27]</td>
</tr>
</tbody>
</table>

**Table 7. Articles overview**

Most of the articles were found through Google Scholar’s function Cite and references in the first articles found, from step two.

### 4.3 Step four

In the articles, different techniques were used to solve the NRP. Two approaches was mainly used, namely pure CP or a hybrid approach. A hybrid approach is using CP combined with another technique, which can be a number of various techniques. With pure CP I mean that only the CP technique was used to solve the problem. In some cases different approaches in CP was used, e.g. CLP or HCLP.

<table>
<thead>
<tr>
<th>Categorization</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP</td>
<td>[1], [6], [23]</td>
</tr>
<tr>
<td>CLP/HCLP</td>
<td>[15], [26], [27]</td>
</tr>
<tr>
<td>Hybrid</td>
<td>[2], [7], [19], [20], [21], [22], [24], [25]</td>
</tr>
<tr>
<td>Soft Constraints</td>
<td>[2], [6], [15], [19], [20], [22], [23], [24], [25], [26], [27]</td>
</tr>
<tr>
<td>Soft Constraints + User interaction</td>
<td>[15], [22], [23], [24], [25], [26], [27]</td>
</tr>
<tr>
<td>User interaction + Graphical interface</td>
<td>[15], [22], [24], [25], [26], [27]</td>
</tr>
</tbody>
</table>

**Table 8. Categorization of articles**
4.3.1 The Constraint Programming approach

Common for these articles is that the authors have used CP to solve the problem. However, they differ in some respects to what they want to achieve, while in others they share aspects. One thing they all have in common, except for the CP approach, is that they have an academic approach. Even if they are making a model for a hospital in the work by Pizarro et al. (2011) [1], the purpose was to check the quality of the solution. Therefore none of them are saying anything about a graphical interface.

Soft constraints are commonly used in all the works, except for in the paper where they hope to include that in the future [1], here the soft constraints are referred to as wishes. The constraints used in the papers are not all from real world examples [6], here it is not clearly stated whether they are using real world examples or not.

Two papers are quite alike each other when looking at what they are trying to achieve [1], [6]. Both have kind of the same goal when coming to their solutions. They are focusing on the computing time when creating scheduling, how fast they can do it. In the paper by Métivie el al. (2009) [6] they are also focusing on the quality of the solution, compared to other solutions they mention in their work. They say that they have excellent results on small to medium size problems and very promising on large scale problems. They mention that the solution could be better for large scale problems if a hybrid solution, with heuristics, would be used. In comparison, the work by Pizarro et al. (2011) [1] is not comparing to other solutions. They are reporting their run times though, when creating schedules for different amount of nurses.

In the work by Trilling et al. (2006) [23] they have a slightly different approach to the problem. First of all they have two solutions to the problem, one is a CP approach and the other one is an Integer Linear Programming approach (ILP). The goal they are trying to achieve is to maximize the fairness of the schedule and to compare the two approaches. To get a fair schedule as possible they apply a penalty to shifts for each nurse. The shifts has different score depending on what kind of shift it is and which day it is. The best schedule would have the least total penalty associated to each nurse. They also allow some user interaction, when a possible solution is found additional constraints can be added to find a better solution. Their conclusion is that the ILP is a better solution to this problem, but at the same time they state that the two is not entirely of a comparable nature. They base this on the computing time when creating a schedule and better results when looking at the objective criteria they have. The ILP solution gives a faster solution in almost every case they have and when looking at objective criteria, the CP solution gives worse results when the number of nurses are many or oversized.

None of the articles mention anything on any user interaction when their solution fails to create a schedule. They barely mention anything on failures at all, just that the fail rate is some factor in present. Neither anything on how to modify the constraints so it would be solvable, or if the solution can tell what the failing constraint was.

4.3.2 The Constraint Logic Programming approach

This chapter will also include HCLP solutions, because it is a subgroup to CLP but also because the articles are not that many. There is no reason to separate them.

Each work in these articles are based on a fully built systems, based on soft constraints and real world examples.
The ConPlan project was conducted by the German Research Center for Artificial Intelligence and the SIEDA Software house [15]. This system they created are used in several hospitals in Germany. The system is built with a help of HCLP C++ library they built on their own to solve the problem. A schedule takes from five to twenty minutes to create, depending on the complexity of the problem. Users can define personal requirements in this system, it is not clear however how this is implemented. It also supports user interaction. A user is able to change the problem description when no possible solution could be found, they do not mention how this is done either. A user should also be able to alter the resulting schedule by hand.

The work by Hofe (2000) [27] is a continuation on earlier work by the same author 1997 on ConPlan [15]. It builds on the previous work and experiences from the system. In this work they have changed the way to attack the problem. Instead of model the problem as a constraint satisfaction problem, it is modeled as a constraint optimization problem. Hence it is not possible to satisfy all the constraints when they are defined in a fuzzy way, like they are in this work. This way to handle constraints is a new way to handle them, namely to partially violate them and partially satisfy them. Because of this, a new way of solving the problem is developed. The new solution is a hybridization of iterative improvements.

The next article is quite old [26], older than the year stated in chapter 3.1 but it is still considered worth mentioning.

INTERDIP was a system built and tested on a hospital in Munich. The system presented looked very promising when reading this article. INTERDIP could produce a schedule in a few seconds, not optimal but satisfying. One minor thing was that they could not guarantee it to create a better schedule then a human though, they refer to the problem’s complexity, a NP-hard problem. In the solution they use costs on the constraints to get a fair schedule. They do not state any numbers on results for their solution though. When coming to the usability part, the system looks very promising. Users are supposed to have the ability to add wishes and the system takes these in account when creating a schedule. A user could also alter a finished schedule and then check with INTERDIP if the changes were possible. The system is also able to give hints to the user when the model is over-constrained, on constraints which made it over-constrained. For future work they want the system to be able to find new schedules when a user is adding requirements to an already existing schedule.

This last article about INTERDIP would be interesting to follow up, but nothing else than this paper has been found.

### 4.3.3 The hybrid approach

These articles have a hybrid approach for solving the problem. Some of them share some sub approaches for solving the problem. As mentioned earlier in 1.4, the new techniques that are used in the hybrid solutions will not be explained in this paper. Every article will be presented separately, because of the differences between them.

The approach that is used in the paper by Wong, Chun (2003) [7], is able to detect if the problem is solvable even before the search has begun. They use meta-level reasoning to find erroneous constraints before the CP part will start searching for a solution. In this way they can minimize the problem, hence they can find a complete solution faster. Typically the search for a solution need to search through the whole domain before it can tell if a solution does not exist. They claim that they can solve the problem within one second. Worth mentioning is that their model of the NRP is simplified, they are scheduling twelve nurses on three different shifts over a week and they do not consider wishes, i.e. soft constraints. The hybrid approach is used because they state that an approach based on only CP cannot solve the problem in a reasonable amount of time. When coming to error handling or user interaction, they do not mention anything of it. This article is of an academic approach only, the solution is the only thing considered.
This next solution for solving the problem is done with a combination of CP and meta-heuristics [2]. The heuristics used is iterative forward search and simple variable neighborhood search. In their solution they use CP as an initial solver to find sub solutions, partially working schedules. When this is done they use iterative local search to create complete schedules. The simple variable neighborhood is used to improve a complete schedule. The solution is tested with real world examples, large scale benchmark problems. The hybrid solution can create a complete schedule in thirty minutes, the CP part takes an average time of 370 seconds depending on the amount of constraints and the iterative part is done within a minute. They mention that a longer search time does not produce a better schedule. A comparison of this solution and solutions in literature is done, the comparison shows that their approach is quite good compared to others. A comparison between this hybrid and solutions based on the single paradigms, used in the hybrid, alone is also done. The result was that the hybrid approach is much better than the techniques alone. Nothing about error handling or user interaction is considered in this article.

The approach that sticks out is the one using CP with column generation [20]. The problem is divided in a master problem and sub problems. The columns, that are generated, which satisfies the constraints will enter the master problem. The sub problems are weighted, they have a costs, and on this they use something they call a threshold. This threshold is adaptively updated by bound tightening, the purpose is to get better columns, i.e. lower the price on the constraints. The columns that are low enough will be allowed to enter the master problem. The master problem is formulated as an IP and the sub problems are modeled as weighted Constraint Optimization Problems in the CP paradigm. They claim that this model formulates all the complex real world constraints in several benchmark NRPs. One thing that they do different than others is that they try to solve the night shift assignment before any other shift, the motivation is that it is those which tend to ruin a search run. They mention a failure rate but not how to come back from it, neither do they mention anything of user interaction.

This solution is implemented in a system that is used at a hospital [24]. Therefor their test data is from real world examples, including soft constraints. It is still an ongoing work though, every result in the paper are still preliminary. Their solution(s) is a combination of CP and different local search algorithms. The local search algorithms are used to improve the solution found by CP. They use different local search algorithms simply because they want to test which one is best. According to the paper hill climbing is the best one. An acceptable solution can be found within a couple of minutes, the CP part finds a possible solution within five to ten second. This solution is then used as a starting point for the local search algorithm. The paper is mentioning that users are able to interact with the system, a bit at least. A user has the possibility to choose which local search algorithm to use when a first possible solution is found by CP. The user is also able to choose how long the search will run. Nothing of error handling is mentioned in this paper either.

Another system that is used in the industry is the one presented in the paper by Stølevik et al. (2011) [25]. The system is used in several hospitals in Norway and Sweden and called Gat, therefor real examples are used. Here they are also using CP for finding an initial solution to the problem, variable neighborhood search is used for improvements. They do not compare the solution with others out there, but they present some data on how the result for their solution is. Here they mention that soft constraints cannot be avoided to be violated, they claim that when having these big real world problems, some of them needs to be violated to find a solution. Nothing is mentioned on how the system works when this happens, when looking at the user interaction in such a case. However, they mention that a user can abort the search process whenever he/she wants, when this happens the best schedule found at the time will be chosen. The article does not focus at the usability, just the solution. Some future work is mentioned though, they seem to want to add more flexibility for end users, by adding the possibility to dynamically change parameters, e.g. start and stop times.
Another software was found, called PARPAP, which is an industrial prototype [22]. The software has a graphical user interface but is not mentioned in this paper, because of the focus on the optimization module. The PARPAP software is using a combination of CP, LP and (meta-) heuristic to solve the problem. They use real world constraints, including soft constraints. To fulfill end users requirements they even came up with a new approach. This solution is also using CP as the initial solver and the others to make it better. The solution is creating many good schedules instead of a perfect one, the end user can choose which one of the schedules to use. They compare the paradigms, techniques, used against each other. According to their result, none of the techniques alone can compete with the hybrid they are using. Again, nothing is mentioned on how to come back when the solution gives a failure. An attempt to find out more about PARPAP’s graphical interface was done but without any luck.
5. Discussion

This chapter will first discuss the result of this work, and later, a discussion of the method used will be done.

5.1 Result

Clearly there exists no common solution to the NRP and most works on the problem with the CP technique are of an academic approach. One reason for this can be due to the NRP is a NP-Hard problem [28]. The purpose has been about the modeling and solution, not so much on the interaction and usability part. None of the articles really mention anything about what happens when their solution is not able to create a complete schedule, they just mention a failure rate, if they even do that. Maybe this is due to the academic world, you would probably not present bad results in, instead point at the good things. In those cases when a graphical user interface exists, the focus has still been on the modeling and solution and no discussion about the interaction. An opportunity to try some of these systems hands on has not been possible. It had probably been good to do so, because it should answer the question better.

Another thing that has been noticed is that there exists a lot more of hybrid solutions then pure CP, or CLP, solutions. A lot of discussions about this can probably be done but the main reason is mostly due to the complexity of CP. The learning curve for this technology is pretty high and takes a lot of time to master [28]. When looking at how the problem is being tackled and what conclusions they present in the articles, it seems like a hybrid approach is the preferred one for this problem. In some of the articles, [6][22][23], they claim that CP is not that good at solving the NRP but a hybrid is better. In the paper by Hohe (1997) [15] they mention that CLP is not good at handling soft constraints. Which can be seen in the articles where they use CP as the initial solver and another technique to make it better. The problem does not seem to be that the CP technique is bad at finding a solution, but it cannot find a good solution by itself. It is more of a problem in the optimization part for the CP approach. This is also proven in the articles where they do a comparison between the different techniques. It is also proven that backtracking has a bad performance [18], which is used in CP. An article found, not really belonging to this work, is about an international competition in nurse rostering [29]. This was quite interesting to read about but for my disappointment none of the participants seemed to use CP for solving the problem.

In this paragraph a short discussion will be held about why some results were hard to find. The results referred to is the ones focusing on user interactions. One of the reasons for this was that it was hard to find a good amount of articles or work on this at all. A literature review on personnel scheduling has been done [30], in this they do different kinds of categorizations around the problem. They have searched a lot but they did not find so much on this when CP is used either. I can mention that the articles they found has been reviewed. Another reason can be due the change in attention to the development of black-box constraint solvers [31]. Apparently the CP community did develop open and extensible optimization tools, but in recent years they changed view and focusing more on black-box solutions. A black-box solution in this case is a solution or system where the solution step(s) is hidden for the user. A user is not able to modify the input, output or the solver because the solution step(s) is unknown, the system is just giving the user a solution, if there is one. If the community are going in this direction it can become hard to influence their work on, for example, these problems. In the paper by Kellogg, Walczak (2007) [3] they bring up that only thirty percent of systems that research articles discuss are implemented, of the articles they reviewed, only ten percent was solved with CP and only two systems developed with CP were used in the industry. This is quite a low amount, which also can be a reason for why it was hard to find anything.
Further reasons can be that companies may not publish any paper on their product or announce that the product uses public research, due to not show competitors how they are doing things.

Some thoughts outside the primary work, I agree with many things they say in the paper by Kellogg, Walczak (2007) [3]. The scope on academic models are too small, they may need to expand and not only look at the model and solution. The academic community rather focus on the development of new techniques rather than creating whole systems [9]. But if they ever will focus on something else is a big question, this is maybe something that will never happen [3]. Hence, if you have a good solution you will probably not publish it for competitors to see. Still, this is a problem that has existed a long time and will continue to do so. If this could be better and good schedules could be made faster and automatically, it will improve health for both the working nurses and the patients [5][8]. It favors only the hospitals and the health care generally. Others has also come up with suggestions like sitting down with nurses when creating a system [3], and it should be done early and under the whole development. I share their opinion on this, it is the nurses that are the experts and know best. When developing a model or system I think it is quite easy to think that constraint X is more important than constraint Y, while a nurse with many years of experience knows that it is the other way around. Additionally, it is important with a graphical interface [28]. If there would be one, better solutions could possibly be found. Due to people are good at solving these problems, it has nevertheless been done by hand for many years. If a human could interact with the solver, he or she could help with finding a good solution [28].

5.2 Method

Given the time for this thesis maybe to many keywords for step two was used. They produced a lot of search results, which is why the criteria in 3.2.2 was developed. Another person to do this had not been a negative thing, search results had been good to discuss sometimes and more search results could had been reviewed. Maybe a better or broader preliminary research could had been done. This stage is nevertheless the one that produces the keywords used in step two, maybe other keywords that was not used would give more and better search results. Additionally, the criteria developed in 3.2.2 for the filtering process may be to narrow. This could affect the result in a bad way, but it is not certain. Some articles that maybe was good for the purpose of this work may have been lost in the filtering process and was never reviewed.

If a person should do this in the future maybe some differences will be found in the results. It seems like the research are going in another direction, away from using pure CP and towards more of hybrid solutions. Even other solutions than hybrids seems more attractive for this problem than pure CP solutions [30]. As mentioned earlier from the paper by Michel, Van Hentenryck (2012) [31] the CP community is focusing more on black-box solutions than before, this may also play a role in the future. But of course, I cannot foresee the future.

To find answers on the research questions only articles on the Internet has been used. As mentioned earlier could this work benefit from actually testing a system, or even benefit from talking to nurses about this. The attitude on the reviewed articles is that they should not be taken completely as “true”. With this I mean that a researcher may paint his or her solution as “the best”, of course in some cases this is not done with purpose.
6. Conclusion

In this work, a research on CP for solving NRPs has been done and an attempt to investigate if user interaction is considered when solving these problems. The purpose for this work was not about presenting different techniques, but instead focusing on if CP was a good approach for problems like NRPs, which the answer to is no. Sure, it is a working technique but based on the reviewed papers the CP approach alone is too slow and not optimized enough. The hybrid approaches seems more promising for these problems. When it comes to the user interaction part of this work, the conclusion is that no user interaction is actually considered. On big factor for this was the lack of research results in the area, some of the results are mentioning a few things but for the most part is only the solution considered. Other various reason for why this conclusion can be made can be read more about in chapter 5.1. The only article really found focusing on user interaction was on the system Gat, which was mention in the paper by Stølevik et al (2011) [25], and a review of this was found in a master’s thesis by Nakken, Nuper (2014) [32] (in Norwegian).

If the time frame for this work had been longer I had looked more into other approaches for solving NRPs, since a survey found in the end was showing that other approaches for this problem were more popular than CP [30]. If more time was given I had also changed the search keywords for matching more hybrid solutions. Another thing I would had done is to look at other scheduling problems, not just the NRP. It would also be interesting to see if other approaches than CP is more adaptive to user interaction. Maybe other forums is needed for this, since research articles tend to focus only on the solution. This is what I encourage others to look more into for future work.
Reference list


Usage of Constraint Programming for Nurse Rostering Problems – A literature study


[32] Nakken SR, Nupen JR. I hvilken grad er Gat som elektronisk verktøy for arbeidstidsplanlegging og ressursstyring en suksess for Sørlandet Sykehus Helseforetak?: en kvantitativ studie basert på RAPS-veilederen for“ beste-praksis” og De Lone og McLeans InformasjonsSystem (IS) su. Universitetet i Agder; University of Agder; 2014;