*Belgian Journal of Operations Research, Special Issue Constraint Programming Vol. 35 (2) 95*

# Constraint Satisfaction for Requirement Handling of Football Fixture Lists

Jan A.M. Schreuder

University of Twente
Enschede, The Netherlands

**Abstract**

The Fixture Lists for the Dutch Professional Football League are constructed based on a so called clustering method. This approach consists of three basic steps of which the division of the requirements of the League into hard demands and soft wishes is rather important. Such a division is needed because the scheduling is over-determined due to conflicting interests of the parties involved and quality demands. A computer program for fast and directed switches between those two types of requirements is not yet realised. Research is carried out to use Constraint Satisfaction, more specific Constraint Logic Programming, to this extend based on commercial available software packages such as CHIP (Constraint Satisfaction In Prolog).

The conclusions of our investigations are that these kind of packages supply correct answers for a given set of hard and medium requirements in a reasonable amount of time. However, using such an approach in an overall, fast, robust and reliable Decision Support System for Fixture List construction is still a long way to go. The power of constraint logic programming is that it uses consistency techniques which reduces the solution space. The introduction of the domain concept, techniques and syntax derived from it, is one of the assets of CHIP with respect to Prolog.

**Keywords :** constraint satisfaction, timetabling, sports, logic programming, mathematical programming

## INTRODUCTION

In this paper we consider the construction of the Fixture Lists for the Dutch Professional Football League (two Divisions).

**A FIXTURE LIST is a timetable of sports matches, e.g. football, for a season each match involving two teams playing against each other, one at home and the other away, such that** [Schreuder 1993, pp 3-7)

(i) all teams meet each other twice of which one meeting is in the first half and the second one in the second half of the season,

(ii) the matches are played in rounds such that all teams play at most one match in each round,

(iii) the second half of a season is the complement of the first half (just reversing the home-away assignments),

(iv) a number of requirements are obeyed concerning commercial, sporting, and organisational aspects as well as historical rights.

As shown in the PhD-thesis [Schreuder 1993, pp 44-53] such a problem can be formulated with a Discrete Mathematical Programming model, in fact a Set-Partitioning Problem with an exponential number of columns. Solving such a problem is based on the use of Basic Match Schedules (**BMS**). A BMS gives for each number of clubs in a League for each match the home-away assignment and the opponent. For example, club A has the playing order: +B -C +D -E ... meaning that club A plays the first match at home against club B, the second match away against club C, etc. Despite the use of such BMS's which takes care of a lot of requirements, e.g., not more than two home or away matches in a row for each club, the Fixture List Construction (**FLC**) is still NP-hard.

FLC for the Dutch Professional Football League is based on a so called clustering method. This approach consists of three basic steps in which the division of the requests of the League into hard demands and soft wishes, which is based on conflicts and quality, is rather important. A computerprogram for fast and directed switches between those two types of requirements is not yet realised. Research is carried out to use Constraint Satisfaction for this purpose based on commercial available software packages (solvers) such as CHIP (Constraint Handling In Prolog). The first step of our approach uses quite a lot of CS properties like pruning, combining and clustering, more specific propagation: continuously reduce the constraint set depending on the domains of the variables.

The variables in the kind of timetabling models like FLC fulfil certain requirements (= constraints) and optimise a certain objective (= objective function). As can be expected with such a kind of problems where many parties with conflicting interests are involved, establishing an useful and precise objective function is impossible [Tripathy, 1992]. The standard solution process mostly aims at obtaining a feasible solution or a largest subset which is of little use here. Furthermore, there is a quality aspect to timetabling (= preference for one timetable to another based on experience and 'instinct') which can often not taken care of explicitly. This is due to the fact that quality is largely a perception of the decisionmaker. This necessitates interaction of the decisionmaker during the construction of a timetable through a computer. All this urges to look for methods which could handle constraints fast with minimal interaction of experts.

One of the first approaches which looks appropriate to apply is Constraint Selection [Meyers & Shih, 1988] which consists of methods which identify those constraints most likely to be tight at optimality. This kind of approaches, however, just reduces the number of constraints (very useful in itself) but it leaves the two main problems, conflicts and quality, untouched. Another promising approach which is looked upon is called CONSTRAINT SATISFACTION (**CS**). Given are a number of mathematical equations $R_m = \{ r_1, r_2, ..., r_m \}$ which represent certain requirements on a set of variables denoted by $X_n = \{x_1, x_2, ..., x_n\}$ each of them having a certain Domain. We are looking for a feasible solution within the domain of the variables. The main used technique is propagation which continuously reduces the constraint set depending on the domains of the variables. CS uses techniques and ideas from artificial intelligence more in specific Constraint Logic Programming (CLGP). CLGP is not an actual method or algorithm but a tool through which certain algorithms can be employed. CLGP uses a collection of assertions such as IF-THEN-ELSE-AND-OR-EITHER-NOT. A logic model is built up from a set of axioms in Horn Clause form (either a simple assertion or an implication) which will answer .TRUE. or .FALSE. .The main purpose of LGP is to make the programming of software for mathematical expressions (especially conditions) more easily.

Two main approaches for CS applied to FLC could be considered here. The first one is a more Operational Research (OR) oriented approach based on combinatorial heuristics with backtracking [Bakker et.al., 1993] and the second one is based on CLGP which is more Information oriented. In this paper we will concentrate on the last approach such as could be realised by a package like CHIP. The OR approach of Bakker is in itself quite successful but is for a real decision support system too time-consuming (15 hours for one fixture list run and then quality demands another run!).

CHIP is used to evaluate the requirements of the season 1993-94 in order to find a lot of alternative solutions under which hopefully the one preferred by the League (need not be the optimal one!). Evaluating these solutions is done by assigning a value to each fulfilled requirement and comparing the totals of each alternative keeping in mind that FLC is an overdetermined problem with conflicting interests.

Another possible useful approach in this area could be Linear Programming with Logical Constraints (either ... or, if ... then [de Keyser, 1992}. After all, the Fixture List construction can be considered as an Assignment Problem with side constraints. Keyser's approach is not taken into account yet.

This paper considers a part of the author's research into sports timetabling problems which was carried out by two students ([Brown, 1994] & [van Weert, 1994]) under his supervision. Brown's 4 month Industrial Project was entirely devoted to research ways in which CHIP could be used to deal with FLC. Van Weert's Project gives an overview of available References of CS. Finally, as a consequence of the conference on Automated Timetabling at the Napier University in Edinburgh Scotland August 1995, an EURO Working Group on Automated Timetabling (**WATT**) is founded. Most of the research deals with the scheduling of lecture courses or exams in educational institutions, i.e., the problem is to automatically generate a plan which indicates meetings between students, teachers and subjects at times and places. The timetabling of sports fixtures is also one of the fields of interest.

In the remaining part of this paper we first outline the construction of the Fixture Lists with clustering and explore the problem we would like to solve with a package like CHIP. Next, we describe the way we used CHIP and the experiences we encountered using it. Finally, we will come to some conclusions which will give an impression of the usefulness of CS for these kind of timetabling problems.

## CONSTRUCTION OF FIXTURE LISTS WITH CLUSTERING

In this chapter the general approach used for determining a fixture list (an one-to-one assignment of BMS's to clubs) is outlined. This approach consists of the following three main stages.

i) a directed search in the set of all possible fixture lists in order to reduce the solution space;

ii) fulfilling as many priority based requirements as possible;

iii) solve the remaining problems (unsatisfied requirements) in an ad hoc way.

Especially during recent years, the number of problems in stage iii) has increased which should be contained by the still improving and increased power of the computer algorithms employed for stages i) and ii). The objective of our approach used is to minimise the problems at stage iii). These problems must be solved before a final fixture list can be agreed upon by the League. Also, there is a clear distinction between the rational solutions based on computerprogrammes at stage i) and ii), and the politically and socially inspired compromises at stage iii).

As pointed out in the PhD-thesis [Schreuder, 1993], the kind of problems described here cannot be solved with exact algorithms or local search strategies [Glover & Greenberg, 1989]. In fact, it is not hard to show that these so called Local Directed Search Methods (**LDSM**, changing the actual values of the variables looking for a global solution allowing moving away from such a solution due to a decreasing chance mechanism) always will be dominated by methods which will use the structure of a problem. LSDM, however, have the advantage of fast application. It is important to recognise that in the Thesis approach one is not concerned with reducing the size of the problem, but with reducing the size of the solution space based on clustering such that it can be searched in polynomial time (sorting is O(nlogn)) in order to present the League at least two as different as possible Fixture Lists. A cluster consists of a number of clubs with their possible BMS assignments. The objective is to find at least one cluster which contains all the clubs.

The role of the objective function is, in contrast with the usual OR approach, minimal in this approach. The reason is that the set of constraints of the assignment is not feasible (in general over-determined and conflicting) as well the mutual weights of the objective function are subjective due to conflicting preferences. The first stage i) is realised by a combinatorial heuristic which maximises only the number of demands which can be combined such that at least one cluster can be constructed. In practice is aimed at minimal 10 unto 10,000 alternative clusters. Which of theses clusters should be preferred is carried out by the second stage ii). Here a more OR type of objective function is used: the mutual weights of the different wishes. As empirical evidence showed, these weights are rather insensitive for small changes as long as

the mutual priorities are not damaged. Finally at stage iii), the unsatisfied requirements of the chosen Fixture List are taken care of. This approach is pictured in Fig. 1.

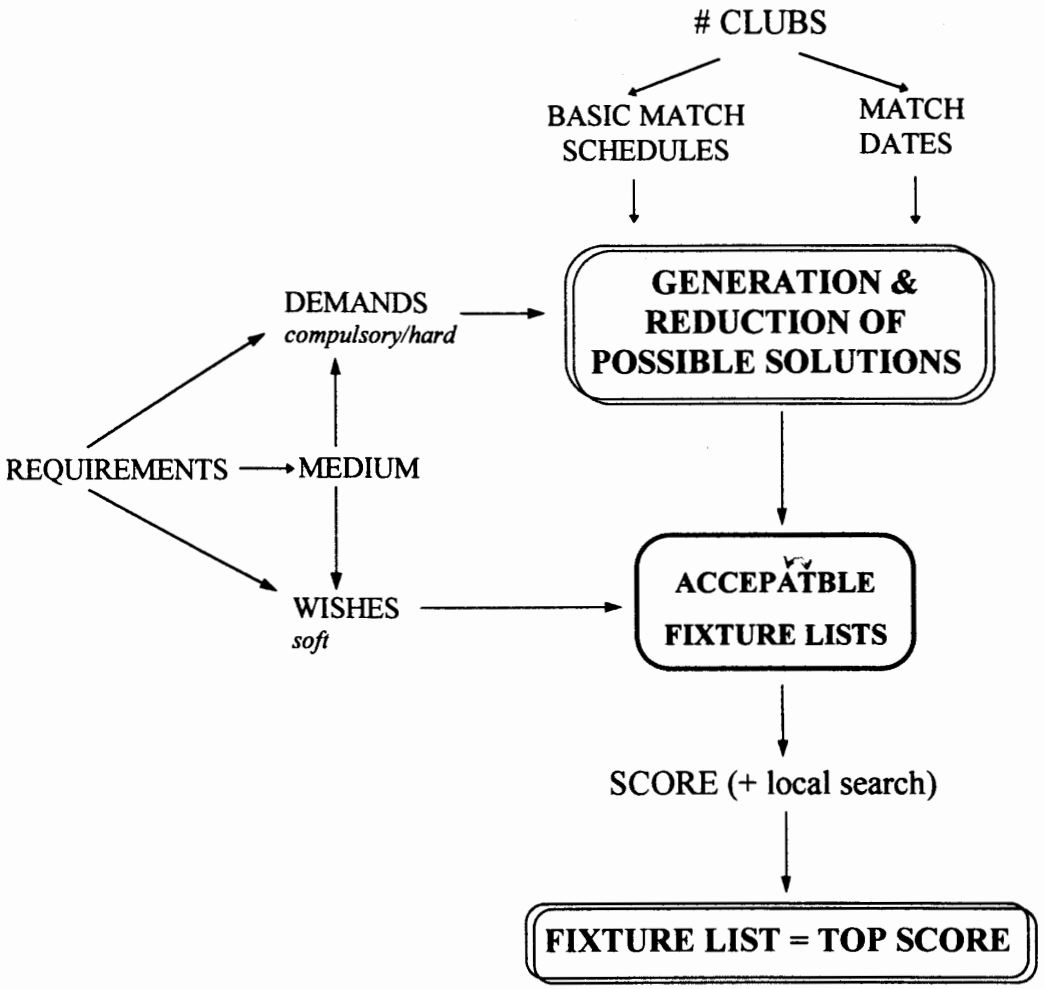


**Fig. 1. APPROACH USED**

The idea behind the use of hard, medium and soft constraints is to find at least one solution which satisfies the most basic requirements. The problem, however, is to know which of the requirements is basic, the League wants to incorporate as much requirements as possible (and more!). In practice a solution has to be found and this approach proved to be most valuable during a lot of years which does not mean it is easy! A lot of negotiation and experimentation has to be done in order to come to one for all parties involved acceptable Fixture List

and this approach really worked in practice. Experience is the key issue here which is needed to divide the medium requirements between the really hard ones and soft ones. The obvious disadvantage is that the demands and wishes are treated separately and the whole procedure has to be repeated several times even if the weights of the wishes are known. As argued before a minimal conflicting set would be ideal (this is the optimal combination of requirements!), but is not practical yet as experiments proved (see [Bakker et.al. 1993]). CS could be the answer here keeping in mind that in general when LGP receives a set of constraints, it has to satisfy them all in contrast to OR approaches which can look for the largest subset.

## APPROACH USED WITH CHIP

CHIP is a solver based on a logic programming language which is supplied by COSYTEC, a firm located at Paris. CHIP is an acronym for Constraint Handling In Prolog. Prolog is in essence a logic program, in which an order is defined for both clauses in the program and goals in the body of the clause. This means that the difference between a Prolog program and a CLGP is that Prolog adopts a stack scheduling policy. CLGP takes the top goal given to it, tries to satisfy this and then any derive goals are stored on a resolving stack. In words of the CHIP manual "*CHIP is not just another Prolog system. While it is compatible with most conventional Prolog implementations, it offers much more in terms of expressive power and efficiency*". It is important to keep in mind that the main issue of this paper is that in view of limited resources packages like CHIP could be an useful alternative for the present use approach or even an improvement.

The power of a language such as Prolog is based on the following three mechanisms.

1. **Relational form**, allows definition of predicates without fixing a priori the input and output arguments.
2. **Non-determinism**, allows a tree search procedure and provides an automatic backtracking mechanism.
3. **Unification**, allows computation to be made.

The power of CLGP is that it uses consistency techniques which means that the search space is pruned a priori by removing combinations which contradict the constraints. The main principle behind CLGP is to replace the idea of equation solving (the unification part) by a more general concept, that of **constraint solving**. This is possible in CHIP and it allows us to solve not only equations, e.g., 3*X+2*Y=17, but also other types of constraints such as

- inequality (e.g. 3*X+3>Y);
- non-equality or disequality (e.g. Day # Monday);
- set-membership (e.g. Colour belongs to [blue, white, red]) etc.

The CHIP package works by importing into its environment a file that has been written in a source file. The general layout of this program is to start with a goal (top goal), then followed by constraints fixed with this goal and finally a labelling procedure that fixes the solution(s) found to the variables declared in the topgoal. Such a procedure determines the search through possible combinations.

We used the Finite Domain of CHIP (others are Herbrand Universe, Rational Arithmetic and Boolean Algebra) as the one most useful for the FLC. The more straight forward demands are easy to incorporate, e.g., 1) the mutual match of two top clubs A and B should not be the last match of the season, and 2) club C does not want to play at home in the third round. First have to be determined which combinations of BMS's satisfy the demands 1) and 2). Demand 1) gives that for each possible BMS - say 11 - assigned to club A club B cannot have two BMS's - say 4 & 10 -.

CHIP: IF A #= 11 THEN B#\=4

IF A #= 11 THEN B#\= 10

Demand 2) gives that club C should have BMS's 4,5,6,8,12.

CHIP: member (C,[4,5,6,8,12])

The kind of constraints for demands 1) and 2) are easy, but constraints containing more variables like the following construction IF (this) AND (this) THEN (this) is not allowed. Of course it can be formulated, but the number of constraints will be explode this way or alternatively the formulation based on BMS's must be dropped as did the CHIP people. So, the more complicated the demands the more the resemblance between the users formulation and the CHIP input will be lost. Experienced CHIP users are needed then or a separate program should be build in order to restore this resemblance. The modelling power of CHIP is nicely demonstrated because it was not difficult to extend the program with constraints like "force/not-force A v B in given round, force/not-force home/away for given round/team, if A v B at A then C and D must be away, for a specific round if A v B then C cannot play D, etc."

Ideally what is desired from a package like CHIP is one single main program, possibly with subprograms to be called up, that will allow details of BMW's for each club, along with the constraints, to be entered. This be doing, the program should run with no human interference and produce the top ten optimal solutions.

The 1993-94 season was the chosen problem. As said before, the second stage of our approach is the more simple one (just a matter of assigning weights to each of the alternative solutions constructed at the first stage: sorting O(nlogn)). Therefore we only considered the first stage which concerns the hard constraints for which a feasible solution must be found. With all this one must still keep in mind the limited time and money available for experiments.

## USEFULNESS OF CONSTRAINT SATISFACTION

The main questions which should be answered when applying a new method like CLGP for FLC are the following. Does a feasible solution exist in view of the requirements, and, if not, how to construct one considering the development and operational aspects of the computer package? Does CLGP perform better than the present used approach both in an effective and efficient way? Finally, what are the effects of using one of the standard computer languages like Fortran, Pascal or C++ (maybe in combination with a database system like Clipper) compared with a CLGP like CHIP? It is certainly not the solution power of CHIP which is at stake here which is without any question and CHIP satisfies the industrial standards. The point is learning to use the already available programs and handling CHIP.

The main problem encountered with using a package like CHIP is the intelligence by which CHIP runs its programs. It is strictly a logic programming environment and as such the idea of putting weights on and perhaps even ignoring certain constraints is not what it was intended for. One of the seemingly inevitable problems with manuals and packages is that you can only understand them if you know them. Considering the package is now up to its fourth version, one would have thought this must be improved quite a lot. The procedure of correcting your own errors however is still rather complicated and time consuming. How to use the labelling procedure correctly is not quite clear. Of course, a lot of improvement can be expected if a well trained and experienced person would handle CHIP.

A maybe smaller but even more significant problem was encountered while asking for support from the firm supplying CHIP. The engineers did a nice job in formulating the problem, but changed the way BMS's should be handled and even relaxed the used bounds on the home-away patterns. It is understandable from their point of view and they are not to blame for it. But doing so, a basic rule in OR approaches is violated which states never to change a problem to fit the method, but adapt the method such that it solves the problem. So, CHIP just considered as a solver cannot do the trick. What is needed is an experienced user of CHIP who is well known with the ins-and-outs of the Football League.

One of the last propositions is that the advantage of using a CLGP in which the modelling part is build in, should outweigh the advantages of using computer languages like Fortran in which all aspects could be build in by the user, is certainly not true yet. Moreover, the speed of solving the problem is not the important issue here. The method developed by the author will give a Fixture List within 5 minutes for any combination of hard and soft requirements. In fact, the system is used for support at meetings between the League officials and the university for establishing the Fixture Lists. The added value for CS, given it can match the required speed, which is not true yet, is in constructing higher quality solutions. These approaches will certainly pursued further for this purpose, but cannot compete yet with the overall performance of the present used approach, keeping in mind that special developed tailor made approaches are always hard to beat for more general operating packages which in turn allow a faster first approach.

## CONCLUSION

One of the most difficult problems in applying a new method like CLGP to these kind of timetabling problems is to establish the rate of success. The more problems it solves or requirements fulfils, the more will be pushed forward. So, these kind of problems are basically over-determined. Inevitable there is a point where questions will be asked which subset of requirements is feasible or even more which is the minimal conflicting set. Methods which just aim at optimality or even feasibility will fail here. Another problem for solvers like CHIP is that CLGP aims at facilitate the user in building a model or programming an algorithm which is only true for an experienced user. It is clear that in our case where the problem is infeasible this is hard to realise without a direct communication between the problem owner and the computer package. Of course, in view of more time (and money!) spend on using CHIP, results could be improved, but that's not the way real world projects are carried out!

Looking for the longest subset in a more OR oriented approach or a minimal conflicting set looks more promising in view of quality improvement, but these approaches are very time consuming at the moment as research suggests.

The use of the clustering method as developed by the author is still appreciated by the Dutch League and supplies acceptable results fast enough such that the League as well as other parties involved can act interactively. As argued before in this paper, the research in solving this kind of problems is certainly not finished yet. Quite a lot of possible approaches of which CS is a rather important one can and hopefully will be investigated in order to improve FLC. Important, however, is that we need practical solutions now too with an approach as effective and efficient as possible in view of limited resources such as manpower, time and money. Altogether, CS approaches for FLC are rather time-consuming or too complicated for inexperienced users.

**REFERENCES**


[1] R.R. Bakker et.al., "Diagnosing and solving over-determined constraint satisfaction problems", PROCEEDINGS IJCAI-93, Chambery, 1993.

[2] Hemant K. Bhargava, "Editor's introduction to the special issue on logic modelling", *DECISION SUPPORT SYSTEMS 16* (1996) 1-2.

[3] Robert Geoffrey Brown, "Industrial Project Glasgow Caledonian University: CONSTRUCTION OF FOOTBALL FIXTURES", Glasgow, october 1994.

[4] Fred Glover & Harvey J. Greenberg, " New approaches for heuristic research: A bilateral linkage with artificial intelligence", EUROPEAN JOURNAL OF OPERATIONAL RESEARCH 39 (1989), 119-130.

[5] W. de Keyser, "Dissertation Vrije Universiteit Brussel: LINEAR PROGRAMMING WITH LOGICAL CONSTRAINTS", 1992.

[6] Ho Geun Lee, Ronald M. Lee and Gang Yu, "Constraint logic programming for quantitative and quantitative constraint satisfaction problems, *DECISION SUPPORT SYSTEMS 16* (1996) 67-83.

[7] Danny C. Myers & Wei Smith, "A CONSTRAINT SELECTION TECHNIQUE FOR A CLASS OF LINEAR PROGRAMS", OPERATIONS RESEARCH LETTERS, Vol 7 Nr 4 (1988) 191-195.

[8] Jan A.M. Schreuder, "PhD Department of Management Science, University of Strathclyde: CONSTRUCTION OF FIXTURE LISTS FOR PROFESSIONAL FOOTBALL LEAGUES", Glasgow, december 1993.

[9] Arabinda Tripathy, "Computerized decision aid for timetabling - a case analysis", DISCRETE APPLIED MATHEMATICS 35 (1992) 313-323.

[10] Arjen van Weert, "References Research Project Faculty of Applied Mathematics, University of Twente: CONSTRAINT SATISFACTION PROBLEMS AND THEIR SOLUTIONS", Enschede, June 1994.

[11] Mike Wright, "Timetabling Country Cricket Fixtures Using a Form of Tabu Search", J.OPL.RES.SOC., Vol.45, No.7 (1994) 758-770.