Research issues and challenges in fuzzy scheduling

Roger M. Kerr **Wolfgang Slany**[⋆]

[⋆]Christian Doppler Laboratory for Expert Systems
E184/2, TU Wien, A-1040 Vienna, Austria, Europe
Phone: +43–1–58801–6141 Fax: +43–1–5055304
URL: http://www.dbai.tuwien.ac.at/staff/slany.html
E-Mail: wsi@vexpert.dbai.tuwien.ac.at

CD–Technical Report 94/68

Christian Doppler Laboratory
for Expert Systems

Technische Universität Wien
Institut für Informationssysteme
Abteilung für Datenbanken und Expertensysteme

Research issues and challenges in fuzzy scheduling

Roger M. Kerr
Department of Industrial Engineering
University of New South Wales
P.O. Box 1, Kensington
N.S.W. 2033, Australia
E-mail: R.Kerr@unsw.edu.au

Wolfgang Slany*
Institut für Informationssysteme
Technische Universität Wien
Paniglgasse 16 (E184/2)
A-1040 Vienna, Austria
E-mail: wsi@vexpert.dbai.tuwien.ac.at

December 14, 1994

Abstract

Scheduling is an activity fraught with fuzziness. The prevalence of Murphy's Law in real life situations, noise infected measurements, vague objectives of different importance, where compromises between antagonistic optimization criteria are generally allowed, and the difficulty of estimating process times all contribute to make the construction of predictive schedules a complex task. Both authors have gained in-depth practical experience in the field of fuzzy scheduling. Application domains of implemented systems ranged from production scheduling of mechanical and hydraulic presses [18, 19] to fine-grained scheduling of high-grade steel production [25]. In addition to fundamental and advanced theoretic research on fuzzy scheduling, Slany's Ph.D. thesis [25] also contains an annotated bibliography[1] with more than 400 references on fuzzy scheduling. The present paper takes a step back from actual implementations to introduce the main aspects of fuzzy scheduling to a broader audience, to compare pros and cons of various approaches, and to discuss some hot research topics of the field.

1 What is scheduling?

Scheduling has been examined in the operation research literature since the early fifties [8]. It has been defined by Baker [2] as

*Please send correspondance to Wolfgang Slany.

[1]Online-versions of the Ph.D. thesis [25] and the bibliography can be retrieved from ftp://mira.dbai.tuwien.ac.at/pub/slany/{thesis.ps.Z|fuzzy-scheduling.bib.Z}.

"the allocation of resources over time to perform a collection of tasks."

Scheduling has an important role in the field of production and operations management. Therefore, terminology relevant to scheduling derives mostly from that field. Variations in problem types are typically illustrated using the manufacturing domain. For example, a *job* is a term used to designate a single item or batch of items that require processing on the *machines*. The processing of a particular job through a particular machine is called an *operation*.

Usually, a job is identified with a deliverable product that has to meet a certain quality. A job may have a release and a due date, the combination of which being often called a delivery date. Associated with each job is a formal specification of the item to be produced. Resources are typically those tools, units, materials, and personnel which are used or consumed in the production process. Associated with each resource is some formal specification of its characteristics and capabilities.

A planner considers the specifications of the jobs and the resources to generate a set of operations called a process plan. The execution of this plan produces the desired result with a set of explicit ordering constraints on the operations and a set of resource requirements. These process plans are often fixed for certain products. In contrast to a job shop, the sequence of operations is fixed in a flow shop.

When several jobs are to be executed together, the composition of their resource requirements implies additional ordering constraints that prohibit simultaneous demands on non sharable resources. A scheduler must satisfy both the explicit ordering constraints imposed by the plans and the implicit ordering constraints derived from the availabilities of the resources. The scheduler has also to consider release dates, expected due dates, setup times, and maintenance intervals.

Thus, the order or sequence in which jobs are processed gives rise to a common problem classification [4]:

- General job shop scheduling — where every job may have a different routing through the machines.
- Flow shop scheduling — where every job has the same routing through the machines.
- Permutation scheduling — where the same job-sequence applies to all the machines.

It should be noted that the second and third classes are really special cases of the general job-shop problem. It is easy to relate these concepts to other areas where scheduling is necessary. For example, in the management of a hospital, patients can be viewed as *jobs*, and beds, doctors, nurses, etc., as the *resources* that correspond to machines. In the case of the Hubble space telescope scheduling problem [22], astronomical experiments would be the jobs, and energy as well as optical and various other instruments the corresponding machines.

Dorn and Froeschl [10] proposed another possible classification of scheduling problems by methods used to solve them:

- Mathematical-analytical methods developed in the context of operations research can be characterized theoretically sound and yielding optimal solutions. These methods do have their limitations, particularly in the field of modeling nasty real world applications. However, they have also their success stories [33], and have successfully been combined with fuzzy methods to solve scheduling problems. We refer here only to work of Zimmermann [31, 28, 32, 30, 29], but [25] lists several hundreds references relevant to the subject, which should be consulted by interested readers.

- Knowledge based methods try to represent all constraints of a scheduling problem *explicitly*, even if some of them are only vaguely known. The main advantage is that the knowledge base is separated from the inference engine, and that knowledge can be manipulated on its own. Several techniques of artificial intelligence have been developed to master knowledge representation issues. These methods have also been combined with fuzzy methods to solve scheduling problems, albeit to a much smaller degree.

The two approaches are often interpreted as antagonistic [33], and there are differences of fundamental nature. For instance, while constraints are used in both fields, artificial intelligence researchers use them in a quite different setting, also exemplified by the meaning of the abbreviation 'LP' common to both fields: in the mathematical-analytical case, it stands for 'linear programming', where constraints are quite simple mathematical inequalities, whereas in the artificial intelligence case, it stands for 'logic programming', where constraints can be used to unify variables, and where constraints represent knowledge that can be updated and manipulated in many ways. The question "Which of these approaches is the better one?" is a hot topic in many discussions. We believe that both approaches, rather, are complementary to each other, and techniques from one field often inspire new ideas to the other one. Pragmatically, this means that both fields could start tackle problems together instead of fighting with each other.

2 Why fuzzy scheduling?

In manufacturing industries such as steelmaking, the distinction between commercial viability and failure often lies in the ability to control the production process through efficient scheduling. Government as well as industry require practical approaches to a diverse set of complex scheduling and planning problems. While scheduling has been studied in isolation for many years, recent advances in artificial intelligence and operations research indicate a renewed interest in the area [10]. In addition, the scheduling problem is being defined more generally, and work is beginning to consider the closed-loop use of scheduling systems in operational contexts. However, a primary source of difficulty in constructing good schedules stems from the conflicting nature of the objectives.

As with many real life decision-making situations, it is usually not possible to fulfill perfectly all objectives when building new schedules. This applies to class-room schedules, staff-roostering, as well as production schedules in manufacturing. Existing approaches to scheduling have tended to reduce the complexity of the problem by considering only a *small subset* of objectives. In real world situations, it would often be more realistic to find *viable compromises* between the objectives. For many problems, it makes sense to partially satisfy objectives. The satisfaction degree can then be used to evaluate the achieved compromise. In addition, real objectives are often prioritized, therefore it is necessary to weight their satisfaction with importance factors. One especially straightforward way to achieve these two aspects of scheduling problems — to satisfy constraints to a certain degree, and to take into account relative importances — is the modeling of these constraints through fuzzy constraints. Fuzzy constraints are particularly well suited for modeling, since constraints can be written in a format easily understood by human experts, and because they feature a robust behavior which needs almost no tuning to yield reasonable control. In addition, the evaluation of their gradual satisfaction can be very efficiently used to guide heuristic search methods as introduced in [26], in order to find approximate 'good' solutions while at the same time greatly simplifying the complexity of the scheduling problem. Real world descriptions naturally contain vaguely formulated relations, because further details are simply not known or would anyway not lead to better results as they would be canceled out through noise in the data. These uncertain data values can be well modeled through the use of possibility distributions, which are special interpretations of a part of fuzzy set theory. The combination of fuzzy constraints and possibility distributions has first been applied to fuzzy scheduling by Prade [23]. The down-to-earth reason behind our choice of fuzzy logic as a basis for knowledge representation is that it allows straightforward modeling of typical scheduling problems and is perfectly combinable with heuristics that find 'good' solutions in acceptable time.

The respected reader may still ask "Why aren't probabilities used instead?" The reason for choosing fuzzy logic and not probabilities as the fundamental knowledge representation technique for uncertainty and vagueness is that, while probabilities and possibilities (fuzzy logic) express different concepts, they can be used to simulate each other, as has been shown by Kosko [20], and acknowledged by Cheeseman [6] already earlier. So, basically, they are equivalent. Then, why do we prefer fuzzy models? On the one hand, even Cheeseman agrees that modeling with subjective probabilities can be a very tedious task. On the other hand, methods based on fuzzy logic have been accepted very well by users all over the world because of the easiness to model in a human-like way many types of complex relationships, i.e., they capture well the vagueness in such everyday expressions as when describing that, for instance, a car is running *fast*, possibly requiring an appropriate reaction. While methods based on probabilities do have valid application domains where the underlying physical relation is known to be of stochastic nature, these methods require, depending on the exact formalism that is used, unrealistic assumptions such as the independency of random variables, or the judgemental estimate of a large number of parameters for which no empirical support would be available for many domains. Therefore, the fuzzy approach seemed much more natural to us, and we have chosen to investigate its potential to help solve scheduling problems as encountered in the real world. It is a bit like, though not so extreme, as when programming a computer in machine language versus programming in a high level language more suitable for humans. Both have their advantages, but often it is much simpler to solve a problem in PROLOG than to write an assembler program for it.

Other non-standard logics are not further considered because fuzzy logic seems to be the best fit for the general real world scheduling problem in terms of easiness to model the inherent properties of the problem description and the easiness to combine it with available heuristics.

3 Complexity issues

Scheduling is a hard problem both in theory and practice. Theoretical scheduling problems, which are concerned with searching for optimal schedules subject to a limited number of constraints, suffer from excessive combinatorial complexity: The general job-shop scheduling problem with n jobs and m machines has an infinite number of of feasible solutions since idle times between operations can vary. A *semi-active* schedule [8] minimizes idle times, but the number of possible solutions is still $(n!)^m$. Simply put, the number of feasible schedules grows exponentially along each dimension (machines, tools, orders, etc.). Many of the most commonly encountered scheduling problems have been proven to belong to the NP hard problems. This implies that these problems are 'at least as hard' as NP complete problems. According to Garey and Johnson [17], NP complete problems are known to be

> *"... 'just as hard' as a large number of other problems that are widely recognized as being difficult and that have been confounding the experts for years.*
> *... the knowledge that [a problem] is NP complete does provide valuable information about what lines of approach have the potential of being most productive. Certainly the search for an efficient, exact algorithm should be accorded low priority."*

The main result is that an exact and efficient solution for NP hard problems has eluded many researchers until now. In particular, algorithms finding exact solutions are more or less irrelevant and useless since they will normally not scale to real world problems unless trivial. These problems have therefore been termed intractable. It is however possible to relax one of two criteria, either exactness or efficiency, in which case the other criterion can be fulfilled in many cases. One suggestion could be to relax the problem somewhat in its unimportant characteristics, i.e., to model a simplified version that does yield an acceptable solution efficiently. This is often sufficient for real world scheduling situations. Another suggestion is indicated by the second sentence in above quotation, which is worth some more investigation. In particular, it is interesting to know that NP complete problems can be solved in polynomial time by a nondeterministic computer. The scenario is that a solution has to be 'guessed', for instance by consulting an 'oracle', followed by calling a polynomial algorithm to check whether the guessed solution is correct. This would suggest that an algorithm which intelligently 'guesses' a complete instantiation and then checks whether it is a solution could be used to construct an algorithm that finds a 'reasonably good' solution for 'almost all' problems.

The following paragraph provides a little more background about the introduced notions. It is an open problem of complexity theory whether NP is equal to P, P being the problems solvable in polynomial time, i.e., the tractable problems. However, most researchers believe that P and NP are different. This would mean that NP complete problems would remain, at least in the worst case, intractable, i.e., their execution time grows more than polynomially when the structural parameters grow linearly. In such a case, doubling the speed of the computer does not really help since only negligible larger problems will be solvable by that computer, which is usually by far not enough. The NP complete problems are characterized by the fact that they are NP problems and that *all* other NP problems can be polynomially reduced to these NP complete problems. This means that NP complete problems are at least as difficult as any other NP problem. To prove that a problem A is NP complete, it is sufficient to show that A belongs to NP and that one other problem B known to be NP complete can be polynomially transformed into A. A general search problem H belongs to the NP hard problems if and only if there exists a *polynomial time algorithm* for some decision problem C known to be NP complete, assuming that H could be used arbitrarily often for further computations at a computational cost of one unit-time interval by the polynomial time algorithm solving C, i.e., C must be polynomial time Turing-transformable into H.

Practical scheduling problems, although more highly constrained, are complex due to the number and variety of the constraints themselves, many of which are 'soft', e.g., potentially relaxable human preference constraints, rather than 'hard' physical constraints. In addition, a 'good' schedule often needs to be evaluated against a number of potentially conflicting goals which themselves may not be precisely defined. Use of analytic techniques to solve practical scheduling problems has in the past been limited due to the lack of suitably expressive languages for constraint and goal representation. The main application area of fuzzy set theory to scheduling is in the systematic framework it provides for the representation, application, propagation and relaxation of imprecise constraints, and the evaluation of schedules with respect to vaguely defined goals. Thus, fuzzy scheduling may essentially be regarded as a class of fuzzy multiple criteria optimization problems in which symmetry exists between goals and constraints, essentially both being treated in exactly the same way. In this connection, the result of the 'optimization' must not necessarily be the globally optimal solution to the problem. Instead, optimization refers here pragmatically to the search for the best solution that can be found using all the available resources such as *available* computers, *available* time, and *available* algorithms. The task is to search for a schedule which has the maximum degree of satisfaction of the specified goals and constraints, both of which may be subject to imprecision. Because of the symmetry between goals and constraints in fuzzy scheduling problems, we will henceforth use the term constraint to cover both, as well as for all other overloaded terms for side-conditions, such as 'criteria', 'objectives', 'aims', 'aspiration levels', 'domains' of variables, or 'importances' of certain objects. On a conceptual level, there is a difference between these notions. For instance, 'criteria' more or less specify what a solution *must* look like, while 'aspirations' specify what a solution *should* look like. However, in an engineering context all these 'side-conditions' are usually formulated in one and the same framework, namely by overloading the term 'constraint' with all these notions. In an operations research context, Zimmermann [30] does not distinguish between 'constraints' and 'objectives', arguing that it empirically models the behavior of decision makers quite well. In [26], Slany has developed a sound theoretic basis for working with semantically well-behaving constraints in the sense definded by Dubois et al. [13], i.e., where violated constraints imply that a corresponding instantiation is not valid, while allowing trade-offs between gradually satisfied constraints.

Most current approaches to automated scheduling organize problem solving into two components, a decision making component responsible for choosing amongst scheduling decisions in order to reach an acceptable schedule, and a constraint management component which uses deductive constraint propagation techniques to combine pre-defined constraints with new constraints introduced by the decision making component to determine new constraints on the decisions remaining to be made. The main impact of introducing fuzzy set theory into scheduling would appear to be in terms of its implications for the constraint management system, and this is indeed where the main thrust of research has been.

4 Types of imprecision in scheduling

In scheduling, there are three main types of imprecise constraint that are suitable for being handled with fuzzy sets. Imprecision stemming from constraints that are blurred in definition includes vaguely defined release and due dates, approximately specified constraints on elapsed times between successive operations, desirable values of WIP[2] etc. Such constraints are often expressed by human schedulers using linguistic variables definable operationally by mapping the preferred range of values of the usually continuous parameters on to corresponding values of a fuzzy membership function representing that constraint. The mappings may represent the subjective preferences of a human scheduler or may be derived from a hierarchical constraint management system to reflect current preferences between possibly conflicting strategic organizational goals.

A closely related class of imprecision arises from ill defined preferences between a limited number of disjunctive alternatives that are themselves crisp constraints — a common example being the relative ordering of two sequential operations. Preferences for different alternatives may again be mapped on to fuzzy sets. However, it may be that there are several different situational arguments both for and against each alternative which may have varying degrees of both match and importance with respect to the current situation. A further application of fuzzy set theory is to the combination of different and possibly antagonistic arguments which may be expressed by weighted fuzzy rules whose conditional parts are fuzzily matched to the current situation and which give varying degrees of support to the alternatives. The OPAL system [3] provides a means of combining the preferences or 'pieces of advice' inherent in such rules using a weighted voting scheme in which advice for alternative decisions is accumulated from different rules through the weighted cardinality of the fuzzy set. A similar method is used by Dorn et al. [12, 27]. In [26], Slany presents a mathematically correct way to aggregate several fuzzy constraints. In particular, the presented method allows to compromise between antagonistic constraints in a well-defined way. Slany [26] develops the idea to allow constraints of different priorities and shows that these different 'weighting schemes' do make sense only in combination with a certain method to aggregate the satisfaction degrees of the fuzzy constraints. How constraints and their weights can be elicited and further fine-tuned for a certain problem is also explained in [26].

The third type of imprecision stems from uncertainty about the values of crisply defined scheduling parameters such as process times, material arrival times, etc. The constraint management system should be capable in such circumstances of propagating the uncertainty to dependent events so that it accumulates them in a way that reflects how knowledge of event timings becomes increasingly imprecise as the number of uncertain dependencies increases. This would potentially allow a 'graceful' degradation of precision as the schedule extends into the future. It can also allow future

[2]WIP: work in progress.

events whose timings are precisely known, e.g., scheduled maintenance periods, to act as 'islands of certainty' from which to plan. Various methods have been proposed for propagating temporal uncertainty including probabilistic approaches [24, 5], and interval logic, e.g. [1, 9]. However, fuzzy set theory provides an alternative and convenient framework for handling these issues if temporal parameters are expressed in terms of fuzzy numbers, and constraints are propagated according to the rules of fuzzy arithmetic. Tests for satisfaction of temporal constraints expressed as temporal inequalities will involve comparisons between fuzzy numbers, so any schedule constructed with fuzzy arithmetic will satisfy each individual temporal constraint, including crisp constraints, to some degree between 0 and 1. This again leads to schedules in which degrees of constraint satisfaction are possible. An important area of research thus lies in the development of temporal constraint management systems based on fuzzy arithmetic, exemplified by the work of Kerr and Walker [19] and Dubois and Prade [15].

Systems for temporal constraint propagation ensure that whenever an operation is scheduled, temporal constraints elsewhere in the system are modified accordingly. A major advantage of fuzzy temporal constraint propagation is in its potential for 'decoupling' different regions of the schedule which, because of imprecision in the parameters, may be regarded as non-interacting. This is a very important issue in reactive scheduling in which the effects of usually frequent but stochastic, unexpected events and occurrences can be localized to that part of the schedule where it is reasonably certain they will have a dominant impact, rather than being propagated out to the limits of the known horizon where in reality they would be swamped by uncertainties arriving from other sources. A complementary application lies in use of this approach to protect schedules against the effects of uncertainty as has been investigated by Chiang and Fox [7] for hedging against machine failures.

5 Fuzzy schedule construction

In the general case, the construction of a fuzzy schedule will involve:

1. establishing a knowledge base of empirical mapping functions, fuzzy rules and weighting factors from which mapping functions can be derived that express the decision makers preferences and state of knowledge and from which the degree of constraint satisfaction of any given schedule can be computed.

2. searching for a schedule with maximal satisfaction of the fuzzy constraints.

The 'desirability' of a particular schedule is given by the degree to which it simultaneously satisfies all constraints, which may be interpreted as the schedule's degree of membership of the fuzzy conjunction of the constraints.

In fuzzy schedule construction, the combination of vaguely defined constraints, coupled with lack of precise predictive knowledge of the extent to which these will be satisfied by any given schedule, implies there will be a much larger class of schedules about which the decision maker is 'indifferent' or which are indistinguishable within the accuracy of the time estimates, than would exist in the crisp case. This can drastically cut down the size of the search space, and a current challenge in fuzzy scheduling is to exploit the representation it provides of varying relative degrees of precision to concentrate search effort in regions of maximum certainty, and conversely, to avoid becoming 'bogged down' seeking improvements in parts of the schedule where the decision maker is indifferent, or adequate predictive knowledge is lacking.

Approaches to schedule construction tend to rely heavily on finding appropriate means of problem decomposition, e.g., using hierarchical approaches, distinguishing between resource-based and order-based perspectives, or the identification and scheduling of critical activities. Fuzzy scheduling does not so much represent an alternative to these but rather a means of enriching the way in which the constraints in such systems can be expressed and propagated. Thus, any existing approach to scheduling which currently uses crisp parameters could potentially benefit from 'fuzzification'. This could also extend to iterative improvement scheduling techniques such as simulated annealing, genetic algorithms, and neural network techniques, as it was done in [26] and in [11] for genetic algorithms and other iterative improvement techniques. Only a limited number of approaches have so far been fuzzified. These include include branch and bound (Dubois et al. [14]), schedule generation and repair techniques, in which an initial schedule is generated to which 'improvements' are then sought (Dorn et al. [12], Slany et al. [27], Slany [26]), and progressive narrowing of time windows by making sequencing decisions between pairs of jobs (Bensana et al. [3], Kerr and Walker [19]).

6 Research challenges in fuzzy scheduling

One of the most important research challenges in fuzzy scheduling lies in finding appropriate ways to operationalize the intersection relation between different fuzzy constraints, by which the degree of constraint satisfaction of the overall schedule is established. Simply to take the minimum value of the membership functions of the individual constraints as is conventional in classical fuzzy set theory allows no possibility of trade-off between constraints, which in reality may be a significant feature of the decision space. One of the advantages of fuzzy set theory is the relative flexibility available in operational definitions subject to certain key axioms, and of course to the fact that the schedule will equate to the corresponding crisp case when membership functions are restricted to $\{0, 1\}$. Most systems developed so far recognize the difference between 'hard' and 'soft' constraints, and it is usually between the 'soft' or potentially relaxable constraints that trade-offs can occur. Slany [26] has developed a different approach in which, basically, every constraint is considered soft, with the important addition that all instantiations to variables getting evaluation scores above zero are considered as possibly *satisfying* the constraint to that degree, whereas all instantiation to variables evaluating to zero *absolutely violate* the constraint. This approach allows to specify 'hard barriers' that should never be crossed when relaxing a soft constraint. The concept of these hard barriers has emerged from the actual need to specify hard limits to ranges for certain parameters, up to which trade-offs can be allowed, while still being able to grade the possible instantiations. Soft constraints *without* hard barriers can easily be specified, too. The membership functions of such a soft constraint without hard barriers must simply be defined such that it never reachs zero, though it can approach zero up to any coefficient $\varepsilon > 0$, always indicating that the respective instantiation is inferior compared to others with larger evaluation scores. This definition of soft constraints with hard barriers allows compensation of *partially satisfied* constraints by other constraints being satisfied to a higher degree, while *violated* constraints cannot be counterbalanced by the satisfaction of other constraints. Therefore, it is in accordance with the remarks by Dubois et al. [13] about what can be correctly termed a constraint satisfaction problem. The hard barrier should never be crossed when constraint relaxation occurs. Thus, a violated constraint will not propagate through to the evaluation function for a complete instantiation of the problem. A hard constraint can be viewed as a special case that requires no further attention. It is necessary to devise a means of combining individual soft constraint membership functions in such a way that important trade-off preferences between constraints are not lost. The use of weighting factors, as in conventional multiple criteria optimization, is one possible approach and is explored, among other techniques, in [26], but this method basically resumes to a weighted averaging which cannot be combined with other aggregation operators such as for instance t-norms, and where even absolutely *violated constraints* can be compensated unless complicated countermeasures are taken. This method is therefore not in accordance with what can be correctly termed a constraint satisfaction problem as specified by Dubois et al. [13]. Slany [26] investigates weighting schemes having mathematically attractive properties and allowing their combination with more and-like aggregation operators. Indeed, a crisp decision making problem requires a solution to satisfy all constraints, i.e., corresponding to a conjunction of all constraints into one logical formula. In many real world problems, trade-offs are allowed. Thus, pure and-like operators (t-norms) are not really the best choice for most applications. Slany presents in [26] soft-and operators embracing most of the nice properties requested to model real world problems, notoriously including the hard barrier feature, the capability to allow compromises within the hard barriers of constraints, symmetry with regard to the arguments, and an intuitive but mathematically correct way to specify weights to prioritize constraints among each other.

A closely related problem is the establishment of systematic means of logically combining different arguments for and against the satisfaction of individual crisp constraints, as in Dubois et al. [14], and choosing empirical values for subjective weights, preferences and degrees of knowledge, which when combined according to the chosen framework, adequately reflect the cognitive processes of a rational human decision maker. Clearly, in any realistic system, a degree of 'tuning' would be required both of the subjective preferences and uncertainty values, and of the system for combining them. This is a difficult and contentious issue which applies equally to the field of decision analysis where it has received considerable attention in the literature. As an answer to this problem of finding an adequate 'configuration', which would also embrace such aspects as to find adequate models for the sets of fuzzy linguistic terms and variables with their associated membership functions, as well as weights of constraints and even appropriate aggregation operators, Slany has proposed in [26] a method to elicit the priorities for constraints from human experts through an intuitively understandable relative scale of priorities which can be mathematically transformed through matrix calculations into a ratio scale usable in a weighting scheme. [26] also develops a complementary test of general configuration changes that checks in a very pragmatic way whether changes in any dimension of the problem description, such as weights or membership functions, are consistent with a set of reference decisions. The knowledge base containing the reference decisions is extended or updated each time changes in the problem configuration are made. The knowledge base thus forms a set of meta constraints hedging in a flexible and transparent way the problem's parameters.

The consistency test is very general in nature, and could therefore be applied to any decision making problem where configuration changes have to be done from time to time. Reasons to modify the configuration could be the introduction of new machines, or simply individual preferences of different human experts. The consistency test is used in such a way that, if the human experts are dissatisfied with a ranking produced by the optimizing system, they can slightly change the weights of some constraints, or the exact form of some membership function (e.g., to specify that the hard barrier is actually located slightly higher), or any other parameter of the problem, such as the aggregation operator used. The consistency test will then check whether the new configuration is consistent with the rankings for a set of reference pairs of instantiations. This is done by applying the new configuration, e.g., the set of new weights, to all the remembered ordered pairs of instantiations, and calculating their evaluation scores with this new configuration. If for each reference pair the order between the two reference instantiations remains unchanged, this indicates that the new configuration does not invalidate any previous reference ordering. It is compatible with all decisions made in the past that became reference ranking pairs. Through this method, it is possible to lead *several* human experts to agree on a common, undisputed subset of some reference ranking pairs of instantiations, or at least to establish several different sets that correspond to configurations which can be further characterized by (and saved for later reuse under) such names as 'risky/cost-cutting', 'highest-quality', 'observe-temporal-constraints', 'standard-mix', etc., indicating their general tendency for decision making. Whether this consistency test can be further developed to allow automatic learning of problem descriptions is a question open for future investigations.

The development of search strategies which are explicitly designed to exploit the characteristics of fuzzy constraint representation is also an area that requires attention. As outlined above, this representation can assist in pruning the search space and in focusing search in more profitable areas where preferences are stronger and predictive accuracy expected to be higher. Care must however be taken not to prune away the branch where the optimum lies. Indeed, almost all fuzzy constraint satisfaction algorithms found in the literature rely on search space pruning to achieve better performance, but often explicitly do not look at possibly better compromise solutions (in particular methods that prune all paths where α-cuts fall below a certain level), implying that a solution featuring an unimportant subconstraint with very low satisfaction but constituting nevertheless a global optimum because of the other, more important constraints being satisfied to a higher degree than other instantiations, could be neglected forever. This danger is not present when combining fuzzy constraints with repair based or iterative heuristics as proposed in [26], since candidates are selected stochastically. No part of the search space has to be pruned for efficiency reasons, and no constraints need to be explicitly relaxed in an unflexible way.

New forms of problem decomposition might be envisaged which focus for example on individual parts of the schedule surrounding 'islands of certainty', between which only weak coupling exists. This is an important consideration in the context of reactive scheduling and the construction of schedules which are robust with respect to unexpected contingencies.

Another potentially very interesting approach to fuzzy constraints used for scheduling has so far attracted only very limited attention. We refer to unification based constraint satisfaction with fuzzy constraints as proposed by Matyska [21]. The scope of this approach is still limited, since only finite fuzzy constraints (no continuous membership functions) are supported. The reason is that unification on continuous domains is a conceptually difficult problem. So far, only toy problems can therefore be handled by this approach, but it is a line of thought that deserves further investigation as fuzzy unification could provide a powerful means to solve combinatorial problems with vague descriptions. Complexity reduction will of course be an important issue, and it might well be that iterative repair based methods again show a valid path to solve fuzzy unification problems.

Finally, an important issue in fuzzy scheduling lies in the problem of quantifying the benefits of the approach. Although the performance of fuzzy and non-fuzzy schedules may easily be compared in terms of the time taken to compute an 'acceptable' schedule, it is more difficult to compare the actual schedules produced without either field trials or controlled simulation experiments containing parameters and events whose probability distributions are unknown to the scheduler. So far, there appears to be a dearth of studies which address this issue.

7 Conclusion

Some recently implemented systems as well as a wealth of published papers and studies on fuzzy scheduling indicate a thorough interest in the field. Methods developed to solve fuzzy scheduling problems have a realistic potential to scale up to other real world decision making situations as well. In the present paper, we gave a broad introduction to the major issues currently under discussion in the field. Additionally, we highlighted some recent successful achievements and pointed out areas that need further investigations.

References

[1] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):823–843, 1983.

[2] K.R. Baker. *Introduction to Sequencing and Scheduling.* John Wiley & Sons, Inc., 1974.

[3] E. Bensana, G. Bel, and D. Dubois. OPAL, a multi knowledge-based system for industrial job-shop scheduling. *International Journal of Production Research*, 26(5):795–815, 1988.

[4] Pauline M. Berry. The PCP: A predictive model for satisfying conflicting objectives in scheduling problems. *AI in Engineering*, 7:227–242, 1992.

[5] Pauline M. Berry. Scheduling: A problem of decision-making under uncertainty. In Bernd Neumann, editor, *Proceedings of the European Conference on Artificial Intelligence (ECAI-92)*, pages 638–642, August 1992.

[6] Peter Cheeseman. In defense of probability. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 1002–1009, Los Angeles, Ca., August 1985. Morgan Kaufmann Publishers, Inc.

[7] W. Y. Chiang and M. S. Fox. Protection against uncertainty in a deterministic schedule. In *Proceedings 4th International Conference on Expert Systems and the Leading Edge in Production and Operations Management*, pages 184–197, Hilton Head Island, 1990.

[8] R. W. Conway, W. L. Maxwell, and L. W. Miller. *Theory of Scheduling*. Addison Wesley, 1967.

[9] Jürgen Dorn. Supporting scheduling with temporal logic. In *IJCAI 93 Workshop on Knowledge-Based Production Planning, Scheduling, and Control*, pages 113–124, Chambery, August 1993.

[10] Jürgen Dorn and Karl A. Froeschl, editors. *Scheduling of Production Processes*. Ellis Horwood Series in Artificial Intelligence. Ellis Horwood Limited, 1993.

[11] Jürgen Dorn, Mario Girsch, Günther Skele, and Wolfgang Slany. Comparison of iterative improvement techniques for schedule optimization. CD-Technical Report 94/61, Christian Doppler Laboratory for Expert Systems, Technical University of Vienna, 1994.

[12] Jürgen Dorn, Wolfgang Slany, and Christian Stary. Uncertainty management by relaxation of conflicting constraints in production scheduling. In Marc Drummond, Mark Fox, Austin Tate, and Monte Zweben, editors, *Practical Approaches to Scheduling and Planning*, Working Notes AAAI Spring Symposium Series, pages 62–66, Stanford, CA, March 1992. Published by the American Association of Artificial Intelligence.

[13] Didier Dubois, Hélène Fargier, and Henri Prade. Handling flexibility and uncertainty in job-shop scheduling. Rapport IRIT 93-30-R, Institut de Recherche en Informatique de Toulouse (IRIT), November 1993.

[14] Didier Dubois, Hélène Fargier, and Henri Prade. Handling flexibility and uncertainty in job-shop scheduling. In Roger Kerr, editor, *Workshop Notes of the Workshop on Fuzzy Scheduling Systems*, Linz, Austria, June 1993. University of Linz, Department of Mathematics.

[15] Didier Dubois and Henri Prade. Processing fuzzy temporal knowledge. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(4):729–744, August 1989.

[16] Didier Dubois, Henri Prade, and Ronald R. Yager, editors. *Readings in Fuzzy Sets for Intelligent Systems*. Morgan Kaufmann, 1993.

[17] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman and Co., 1979.

[18] Roger M. Kerr. Implementation of expert systems for production scheduling - three case studies. In *Proceedings of the 1st World Congress on Expert Systems*, pages 405–414. Pergamon Press, December 1991.

[19] Roger M. Kerr and R. N. Walker. A job shop scheduling system based on fuzzy arithmetic. In *Proceedings of the 2nd International Conference on Expert Systems and Leading Edge in Production and Operations Management*, pages

433–450, Hilton Head Island, S.C., May 1989. Management Science Dept., University of S. Carolina.

[20] Bart Kosko. *Neural Networks and Fuzzy Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1992.

[21] Luděk Matyska. Logic programming with fuzzy sets. Technical Report TCU/CS/1993/4, Dept. of Computer Science, City Univ., London, UK, December 1993. URL: ftp://ftp.cs.city.ac.uk/papers/93/cs93-4.ps.Z.

[22] Steven Minton, Mark D. Johnston, Andrew B. Philips, and Philip Laird. Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, 58:161–205, 1992.

[23] Henri Prade. Using fuzzy set theory in a scheduling problem: A case study. *Fuzzy Sets and Systems*, 2(2):153–165, 1979.

[24] Norman Sadeh and Mark S. Fox. Variable and value ordering heuristics for job shop scheduling. In *Proceedings 4th International Conference on Expert Systems and the Leading Edge in Production and Operations Management*, pages 134–144, Hilton Head Island, 1990.

[25] Wolfgang Slany. *Fuzzy Scheduling.* PhD thesis, Technical University of Vienna, 1994. URL: ftp://mira.dbai.tuwien.ac.at/pub/slany/thesis.ps.Z.

[26] Wolfgang Slany. Scheduling as a fuzzy multiple criteria optimization problem. CD-Technical Report 94/62, Christian Doppler Laboratory for Expert Systems, Technical University of Vienna, 1994. A revised version has been accepted for publication in Fuzzy Sets and Systems. URL: ftp://mira.dbai.tuwien.ac.at/pub/slany/cd-tr9462.ps.Z.

[27] Wolfgang Slany, Christian Stary, and Jürgen Dorn. Vague data management in production process scheduling applied to high-grade steelmaking. In *Proceedings of the First International Conference on Artificial Intelligence Planning Systems*, pages 214–221, University of Maryland, June 1992. Morgan Kaufmann Publishers, Inc.

[28] Hans-Jürgen Zimmermann. Applications of fuzzy set theory to mathematical programming. *Information Sciences*, 36:29–58, 1985. Republished in Dubois et al. [16, pp. 795–809].

[29] Hans-Jürgen Zimmermann. Approximate reasoning in manufacturing. In A. Kusiak, editor, *Intelligent Design and Manufacturing*, pages 701–722. John Wiley & Sons, Inc., 1992.

[30] Hans-Jürgen Zimmermann. Fuzzy mathematical programming. In Stuart C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, volume 1, pages 521–528. John Wiley & Sons, Inc., 2nd enlarged and revised edition, 1992.

[31] H.J. Zimmermann. Optimization in fuzzy environment. In *Proceedings of the XXI Int. TIMS and 46th ORSA Conference*, San Juan (Puerto Rico), 1974.

[32] H.J. Zimmermann. *Fuzzy Sets, Decision Making and Expert Systems.* Kluwer Academic Publishers, 1987.

[33] Steven Zoraster and Ronald Sawey. Optimization-based vs. rule-based. *Communications of the ACM*, 35(6), June 1992.