ATLANTIS PRESS
Atlantis Studies in Uncertainty Modelling, volume 3
Joint Proceedings of the 19th World Congress of the International Fuzzy Systems Association (IFSA), the 12th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT), and the 11th International Summer School on Aggregation Operators (AGOP)

Learning of Consistent Preference Relations for Decision Making and Optimization in Context of Interacting Goals

Rudolf Felix

PSI FLS Fuzzy Logik & Neuro Systeme GmbH, Joseph-von-Fraunhofer Straße 20, 44227 Dortmund
Tel. +49 231 9700921, Fax. +49 231 9700929, Email: felix@fuzzy.de

Abstract

In former papers a decision and optimization algorithm based on interactions between goals was introduced and it was shown how it solves relevant real-world problems. However, the algorithm assumed preference information on the goals as initially given. In this paper we describe a learning algorithm that derives such preference information from decision and optimization input data. It is shown how the former algorithm is improved and how this improvement links the algorithm to reinforcement-based machine learning optimization.

Keywords: Learning of preference relations, Fuzzy interactions between decision goals, Real-world optimization problem, Resource planning, Conflicting goals, Reinforcement-based optimization.

1 Introduction

In a former work [7] it was shown in which way an explicit reasoning on interactions between goals in aggregation, decision-making and in optimization (DMIG) helps to handle complex decision and optimization problems compared to summation-based approaches [2], [15]. It was also shown that even inconsistent preferences expressed given as input [8] can be balanced out according to these interactions [6].

1.1 Introduction of a Learning Algorithm

In this paper we introduce an algorithm for automated learning of (at least partly) consistent preference relations directly from analyzed partly inconsistencies mined from the initial input decision data (as proposed in [6]). We also discuss aspects of using the learned preferences as labeling information for better controlling an existing optimization algorithm and moving it towards reinforcement-based optimization.

1.2 Organization of the Paper

In the subsequent sections first, for better readability of this paper as a whole, we start with a repetition of the description of how interactions between decision goals are modelled and how decision-making and optimization based on these interactions may be organized [6], [7]. Then we introduce the learning algorithm and show how it is used in order to improve an already existing optimization algorithm for active resource planning [12]. We also discuss how the learned preference labels may be used in a novel way as a kind of reinforcement information in optimization -based machine learning. This is a new application of [5] and [7] previously not yet considered neither in [4], [5] nor in [6], [8]. It does not deal with probabilistic models like [9], either. We close with some concluding remarks and with an outlook to future work.

2 Decision Making based on Interactions between Goals (DMIG)

In the following it is shown how an explicit modelling of interactions between decision goals that are defined as fuzzy sets of decision alternatives helps to manage the complexity of the decision making and aggregation process. This modeling of the decision-making and aggregation process significantly differs from the related approaches and the way they manage complex decision situations. First the notion of positive and negative impact sets is introduced. Then different types of interaction between goals are defined. After this it is shown how interactions between goals are used in order to aggregate pairs of goals to the so-called local decision sets. Then it is described how the local decision sets are used for the aggregation of a final decision set. The complexity of the different steps is discussed.

2.1 Positive and Negative Impact Sets

Before we define interactions between goals as fuzzy relations, we introduce the notion of the positive

impact set and the negative impact set of a goal. A more detailed discussion can be found for instance in [5], [6] and [7].

Def. 1a) Let A be a non-empty and finite set of decision alternatives, G a non-empty and finite set of goals, $A \cap G = \emptyset,\ a \in A,\ g \in G,\ \delta \in (0,1]$.

For each goal g we define the two fuzzy sets S_g and D_g each from A into [0, 1] by:

i. Positive impact function of the goal g: $S_g(a) := \delta$, if a affects g positively with degree δ, $S_g(a) := 0$ else.
ii. Negative impact function of the goal g: $D_g(a) := \delta$, if a affects g negatively with degree δ, $D_g(a) := 0$ else.

Def. 1b) Let S_g and D_g be defined as in Def. 1a). S_g is called the positive impact set of g and D_g the negative impact set of g.

The set S_g contains alternatives with a positive impact on the goal g and δ is the degree of the positive impact. In other words, S_g could also be called the set of decision alternatives that are good for the goal g. The set D_g contains alternatives with a negative impact on the goal g and δ is the degree of the negative impact. In other words, D_g could also be called the set of decision alternatives that are bad for the goal g.

2.2 Interactions between Goals

Let P(A) be the set of all fuzzy subsets of A. Let X, Y ∈ P(A), x and y the membership functions of X and Y respectively. Assume now that we have a binary fuzzy inclusion I: P(A)×P(A)→[0,1] a fuzzy non-inclusion N:P(A)×P(A)→[0,1], such that N(X,Y):=1-I(X,Y). In such a case the degree of inclusions and non-inclusions between the impact sets of two goals indicates the degree of the existence of interaction between these two goals. The higher the degree of inclusion between the positive impact sets of two goals, the more cooperative the interaction between them. The higher the degree of inclusion between the positive impact set of one goal and the negative impact set of the second, the more competitive the interaction. The non-inclusions are evaluated in a similar way. The higher the degree of non-inclusion between the positive impact sets of two goals, the less cooperative the interaction between them. The higher the degree of non-inclusion between the positive impact set of one goal and the negative impact set of the second, the less competitive the relationship. The pair (Sg, Dg) represents the whole known impact of alternatives on the goal g. Then Sg is the fuzzy set of alternatives which satisfy the goal g. Dg is the fuzzy set of alternatives, which are rather not recommendable from the point of view of satisfying the goal g.

Based on the inclusion and non-inclusion between the impact sets of the goals as described above, 8 basic fuzzy types of interaction between goals are defined. The different types of interaction describe the spectrum from a high confluence between goals (analogy) to a strict competition (trade-off) [5].

Def. 2) Let S_{g_1}, D_{g_1}, S_{g_2} and D_{g_2} be fuzzy sets given by the corresponding membership functions as defined in Def. 1). For simplicity we write S_1 instead of S_{g_1} etc. Let $g_1, g_2 \in G$ where G is a set of goals. T is a t-norm.

The fuzzy types of interaction between two goals are defined as relations which are fuzzy subsets of $G \times G$ as follows:

1. g_1 *is independent of* g_1: <=> $T(\mathrm{N}(S_1,S_2),\mathrm{N}(S_1,D_2),\mathrm{N}(S_2,D_1),\mathrm{N}(D_1,D_2))$
2. g_1 *assists* g_2: <=> $T(\mathrm{I}(S_1S_2),\mathrm{N}(S_1,D_2))$
3. g_1 *cooperates with* g_2: <=> $T(\mathrm{I}(S_1,S_2),\mathrm{N}(S_1,D_2),\mathrm{N}(S_2,D_1))$
4. g_1 *is analogous to* g_2: <=> $T(\mathrm{I}(S_1,S_2),\mathrm{N}(S_1,D_2),\mathrm{N}(S_2,D_1),\mathrm{I}(D_1,D_2))$
5. g_1 *hinders* g_2: <=> $T(\mathrm{N}(S_1,S_2),\mathrm{I}(S_1,D_2))$
6. g_1 *competes with* g_2: <=> $T(\mathrm{N}(S_1,S_2),\mathrm{I}(S_1,D_2),\mathrm{I}(S_2,D_1))$
7. g_1 *is in trade-off to* g_2: <=> $T(\mathrm{N}(S_1,S_2),\mathrm{I}(S_1,D_2),\mathrm{I}(S_2,D_1),\mathrm{N}(D_1,D_2))$
8. g_1 *is unspecified dependent from* g_2: <=> $T(\mathrm{I}(S_1,S_2),\mathrm{I}(S_1,D_2),\mathrm{I}(S_2,D_1),\mathrm{I}(D_1,D_2))$

The interactions between goals are crucial for an adequate orientation during the decision-making process because they reflect the way the goals depend on each other and describe the pros and cons of the decision alternatives with respect to the goals. For example, for cooperative goals a conjunctive aggregation is appropriate. If the goals are rather competitive, then an aggregation based on an exclusive disjunction is appropriate.

Please note that the complexity of the calculation of every type of interaction between two goals in terms of the O calculus is O(card(A)∗card(A)) [7].

Please note also that the relationships are not symmetrical in general because the degree of the relationship depends on the (fuzzy) cardinality of both the positive and the negative impact sets that is not necessarily equal for two different goals.

2.3 Two Goals Aggregation based on the Type of their Interaction

The assumption, that cooperative types of interaction between goals imply conjunctive aggregation and conflicting types of interaction between goals rather lead to exclusive disjunctive aggregation, is easy to accept from the intuitive point of view. It is also easy to accept that in case of independent or unspecified dependent goals a disjunctive aggregation is appropriate. For a more detailed formal discussion see for instance [5] or [7]. Knowing the type of interaction between two goals means to recognize for which goals rather a conjunctive aggregation is appropriate and for which goals rather a disjunctive or even exclusively disjunctive aggregation is appropriate. This knowledge then in connection with information about goal preferences (numerically modelled as priorities) is used in order to apply interaction-dependent aggregation policies which describe the way of aggregation for each type of interaction. The aggregation policies define which kind of aggregation operation is the appropriate one for each pair of goals. The aggregation of two goals g_i and g_j leads to the so-called local decision set $L_{i,j}$. For each pair of goals there is a local decision set $L_{i,j} \in P(A)$, where A is the set of decision alternatives (see Def 1 a)) and P(A) the power set upon A. For conflicting goals, for instance, the following aggregation policy is given:
if (g_1 is in trade-off to g_2) and (g_1 is preferred to g_2)
then $L_{1,2} := S_1/D_2$.
In case of very similar goals (analogous or cooperative goals) preferences in both directions are partly admissible:
if (g_1 cooperates with g_2) then $L_{1,2} := S_1 \cap S_2$
because $S_1 \cap S_2$ is able to satisfy both goals.
if (g_1 is independent of g1) then $L_{1,2} := S_1 \cup S_2$
because g_1 and g_2 do not interact neither positively nor negatively and preferences may be considered as arbitrary. For more details please see [7].

2.4 Multiple Goal Aggregation as Final Aggregation based on the Local Decision Sets

The next step of the aggregation process is the final aggregation. The final aggregation is performed based on a sorting procedure of all local decision sets $L_{i,j}$. Again, the priority information is used to build a semi-linear hierarchy of the local decision sets by sorting them. The sorting process sorts the local decision sets with respect to the priorities of the goals. Subsequently an intersection set of all local decision sets is built. If this intersection set is empty, then the intersection of all local decision sets except the last one in the hierarchy is built. If the resulting intersection set again is empty, then the second last local decision set is excluded from the intersection process. This process iterates until the intersection is not empty (or more generally speaking until its fuzzy cardinality is big enough with respect to a given threshold). The first nonempty intersection in the iteration process is the final decision set and the membership values of this set give a ranking of the decision alternatives that is the result of the aggregation process (for more details see for instance [5] or [7]).

3 A Learning Algorithm for Calculation of Consistent Preferences

The process goal aggregation described in the sections 2.3 und 2.4 requires as input information which goals are higher prioritized compared to others. This input information assumes that there exists a preference relation for every single goal. It is known that from the perspective of the whole goal set a general and fully consistent preference relation rather does not exist if the complexity of the preferences increases (Arrow´s impossibility and its generalizations) [13]. However, the question is to which extent for some part (subset or subsets) of the goals partly consistent preference relations do exist and how to calculate them.

Such partly consistent preference relations may be obtained by an algorithmic analysis of the interactions between this particular goal and the remaining goals (see section 2.2). The learning algorithm newly further developed from an idea started in [4] generates for every single goal a partly consistent preference order on the set of the remaining other goals as follows here.

3.1 The Learning Algorithm

The input of the algorithm consists of a set of goals G, which at the beginning are all contained in the list L_G. Further on we will construct the preference relations by ordering the goals on a scale, composed of n positions. The scale starts with position 1 and ends with position n, where, as before, n is the number of goals. The algorithm assigns the individual goals positions on the scale. The goals which are assigned to positions at the beginning of the scale are the ones with high preference. The goals which are placed at the end of the scale have low preference.

At the beginning of the execution of the algorithm the scale is empty, meaning that no goal is assigned to the positions of the scale. The algorithm starts with an arbitrary goal and puts this goal at the first position of the scale. Then goals that have a positive interaction to the first goal are step by step put to the next positions on the scale and grouped close to the begin of the scale. Goals with negative interactions to the first goal are

grouped at the end of the scale so that conflicting goals are apart from each other. In order to distinguish stronger (higher) degrees of interactions between the goals from less strong ones the so-called interaction value (*I-value* for short) is used. The *I-value* is the number of predicates the given type of interaction between the goals according Def 2 is composed of. For instance, the *I-value* of the "*cooperates with*" type of interaction is 3 whereas the *I-value* of "*is in trade-off to*" is 4.

The algorithm starts with the initial (nonempty) list of goals L_G where G is the set of goals. Iterating over L_G it assigns goals to the scale until L_G is empty and all positions of the scale have associated goals.

One iteration (here the first one) of the algorithm consists of the following steps:

1. Select an arbitrary goal g from L_G and put this goal into the first free position of the scale. Remove this goal from L_G.
2. Determine all goals g_i for which there is gBg_i or g_iBg where B symbolizes all the negative interaction relation types and sort these goals g_i in an auxiliary list L_-, in such a way that at the higher positions of the list goals are located for which
    a. *I-value* is bigger than the *I-values* of the remaining entries,
    b. When the *I-values* are equal, then sort according the value of μ_B from high to low,
    c. When both the *I-value* and the value of μ_B are equal, then the sorting is arbitrary.

    Using the sorting we take the first goal from L_- and put it at the last free position of the preference scale, then the second goal from L_- is put on the second-last free position of the scale, and so on, until L_- is entirely checked. Finally, remove from L_G all the goals that are in L_- and dissolve L_-.
3. Determine all the goals g_i which are not yet positioned on the scale and for which there is either gBg_i or g_iBg where B symbolizes all the positive relationships and order these goals g_i in an auxiliary list L_+, in such a way that at the higher positions of the list the goals are put for which
    a. *I-value* is greater than the remaining entries,
    b. When the *I-values* are equal, then the goal according to the value of μ_B is taken,
    c. When both the *I-value* and the value of μ_B are equal, then the sorting is arbitrary.
4. Look in L_+ for the goals forming pairs (g_i, g_j) for which g_iBg_j where B is a negative relationship.
5. When there are no such pairs of goals from L_+ then put the goals from L_+, in accordance with the sequence in this list, at the first n_+ positions of the scale, where n_+ is the number of elements of L_+. Remove from L_G all the goals contained in L_+ and finally dissolve L_+.
6. In case we find the pairs of goals in L_+ as given by step 4, determine this pair of goals (g_i, g_j) from L_+ for which there is g_iBg_j and
    a. B has the maximum *I-value*,
    b. In case B has the same *I-value* its value of μ_B is maximum,
    c. In case both the *I-value* and the values of μ_B are the same, an arbitrary choice of the pair can be done.

    For the pair (g_i, g_j) ask the user which of the two goals is more important. Let, out of the two goals in the pair, g_i be the less important goal. Put g_i at the last free position of the scale, and g_j at the first free position of the scale. Remove g_i and g_j from the sets (lists) L_+ and L_G. When L_+ after this operation is not empty, go back to step 4.
7. If L_G is not empty, go to step 1.

The algorithm generates in one (outer) iteration step (steps 1 to 6) one preference relation according to which the goals are grouped (sorted) starting with the selected goal as the most preferred one. As a result, the goals are sorted in consistently with respect to the interactions between the goals: Goals that are in positive interaction to the first goal get preferences close to the selected goal. Those goals which are negative to the selected goal are rather close to the end of the scale of the preference positions. The result of one iteration step is at least one consistent preference relation of the form $g_{i1} > g_{gi2} > g_{i3} \ldots > g_{in-2} > g_{n-1} > g_n$., where $g_i > g_j$ means that g_i *is preferred to* g_j.

After the algorithm has been applied iterating over all the goals as starting goal preference relations for all goals are obtained. Please note that in all cases the obtained preference relations are best possibly

consistent with the interactions between the goals for a given input data set that is evaluated by the impact set functions.

In this sense, the obtained preference relations give compared to the decision-oriented understanding of the input data a comparably deep insight into the structure of the decision situation. This deep insight is something additional that is rather not explicitly known before the preference relations are calculated. Therefore, the algorithm can be viewed as a learning algorithm that derives (mines) consistent preference information originally indirectly hidden in the input data and makes the consistent preferences explicit. In this sense it can even be viewed as a kind of mining [10] of preferences from raw decision data. Please also note that the algorithm is neither supervised [14] nor initial explicit knowledge about the preferences is given [11].

3.2 An Illustration Example of the Learning Algorithm

The behavior of the algorithm is illustrated using an example of a situation with given interactions between some decision goals. Let us assume that a set of goals $G = \{g_1, g_2, g_3, g_4, g_5, g_6\}$ is given. Thus, the list L_G consists of goals g_1 through g_6.

Let us assume that a decision situation with the following goal interactions is given:

g_1 cooperates with g_2, g_2 cooperates with g_1,
g_2 cooperates with g_3, g_3 cooperates with g_2,
g_4 assists g_3, g_4 assists g_5, g_1 assists g_5,
g_3 competes with g_6, g_6 competes with g_3,
g_2 hinders g_4, g_5 hinders g_4

Let us also say that for speeding up the algorithm the membership values of all types of interactions may be (and are) set to the membership value 1.

Please note that only one outer iteration of the learning algorithm is illustrated and that the arbitrarily selected goal WLOG in this iteration step is g_3.

The above interactions between the goals lead to the following course of the algorithm:

Step 1
L_G is initiated as $\{g_1, g_2, g_3, g_4, g_5, g_6\}$. Let goal g_3 be the selected goal. Thus, g_3 is placed on the first free position of the preference scale, which in this situation is position number 1. The goal g_3 is removed from the list L_G.

$g_3 > .. > .. > .. > .. > ..$

Step 2
The only goal which is in negative interaction to goal g_3 is the goal g_6. This latter goal is put on the list L_- and assigned the lowest free position of the scale, in this case the last one, position number 6. The (one-element) list L_- is dissolved.

$g_3 > .. > .. > .. > .. > g_6$

Step 3
The goal g_3 is involved in the following positive interactions:

g_2 cooperates with g_3
g_3 cooperates with g_2 and
g_4 assists g_3.

Consequently, the auxiliary list L_+ consists of the entries g_2 and g_4 and has the form $L_+ = (g_2, g_4)$.

Step 4
Since we have the interaction g_2 hinders g_4, the algorithm branches into the step 6 and continues in this iteration WLOG with g_4 as the more important of the two. Hence, goal g_4 is put on the position 2 of the scale, and goal g_2 on position 5.

$g_3 > g_4 > .. > .. > g_2 > g_6$

The two goals are removed from both L_+ and L_G. Since after this operation L_+ becomes empty, the algorithm passes over to step 7.

Step 7
The list L_G is not yet empty. The algorithm returns to Step 1. (Note: the fact that we now proceed with the second iteration is denoted by the apostrophe: 1').

Step 1'
Again, WLOG the algorithm continues in this iteration with g_1 as the more important one out of g_1 and g_5. Thus, goal g_1 will be assigned position 3 of the scale and removed from the list L_G.

$g_3 > g_4 > g_1 > .. > g_2 > g_6$

Step 2'
No action is undertaken.

Step 3'
Since we have the relationship g_1 assists g_5, the goal g_5 is put on the (again: one-element) list L_+, so that $L_+ = (g_5)$.

Step 4'
No action is undertaken.

Step 5'
According to the contents of the current list L_+, the goal g_5 is put on the first free position of the scale, namely position 4:

$g_3 > g_4 > g_1 > g_5 > g_2 > g_6$

Goal g_5 is removed from both L_G and L_+ (L_+ is becoming empty) and we continue with Step 7'.

Step 7
Since the list of goals L_G is empty, the algorithm stops (in a non-pseudo code implementation the algorithm goes back to the not traversed branches) and the result of the first iteration is the following learned preference relation ($g_i > g_j$ means that g_i *is preferred to* g_j):

$g_3 > g_4 > g_1 > g_5 > g_2 > g_6$.

Please note that the branching may result in more than one preference relation per outer iteration. In a future version of the algorithm the branching will possibly be replaced be generation of preference subclasses and permutating within these subclasses.

3.3 Preference Relations as Labeling Information

Please note that both the interactions between the goals and consequently the learned preference relations are driven by the impact sets (see Def 1) that interpret the raw input data stemming from the domain the decision or the optimization is applied to. This means that the calculation of the preference relations is data driven, too.

Let us assume that impact sets are derived from some measurable values of some key performance indicators (KPIs) that are used to evaluate the decision alternatives with respect the goals. Then for a given set of KPI measurements we obtain both the interactions between the KPIs as goals and sets of consistent KPI preferences. If we then use the preference relations as input for the DMIG decision algorithm consistent decisions are calculated [8] and both a KPI-oriented selection of decision alternatives and at least their partly consistent ranking is the decision-making result.

These consistent preference relations directly reflect how the particular set of measurements of the input data evaluated in terms of the impact sets are converted into consistently learned preference patterns that are (compared to [3]) related to the goals. The obtained preferences are therefore a kind of forecast of how to set the DMIG goal priorities with respect to the possibility to then select and rank decision alternatives (to decide via DMIG) in a best possible, at least partly consistent, way (also in the sense of [13]) for the given input data set from the perspective of the goals to be achieved.

Therefore, the preference relations are a kind of condensed (fused [2]) representation of the initial data set and we can use them as labels that qualify the input data based on the KPI measurements and help to forecast the expected decision output. If a given input data set is similar to an already considered one then the preference relations for the new data set will be similar, too. The preference labels are so to say mined from raw decision data somehow in the sense of [10].

4 The Process of Active Resource Management as Optimization of Key Performance Indicators

In the process of active resource management (ARM), we assume that there is a set of tasks TS to be done across a time line. For each task $t \in$ TS there is a first possible point of time when performing the task may be started (called ASAP for 'as soon as possible') and the latest point of time when the task has to be finished (ALAP for 'as late as possible'). The active resources consist of a set of active entities AE and have ability profiles AP. Furthermore, the AEs are physically located at certain positions in the activity area at its locations L. Each AE is initially located at its initial location IL. Each AE is able to move in the activity area (for instance being connected with some automated guided vehicles AGV). After a task t has been finished, every active entity $a_e \in AE$ may move from its initial location IL (for instance the place where an AE has stopped after its last activity SL) to the place of the first task and then to the next task and then to the next and so on.

The possibility to work on a task is restricted by a set of different conditions. One condition is the capacity limitation C-L of each AE. The capacity may be limited for instance by the time that an active entity AE disposes of. Another limitation is that there may be a maximum movement distance M-D. Another one is that only active entities with particular qualifications P-Q may be associated to particular tasks with the limitation T-Q. All the activities may be organized in activity periods AP, for instance hours, being part of a planning horizon PH, for instance one day or one month. Each task t has to be done in time I-T. 'In time' means that each task t has to be done within the ASAP-ALAP interval and within the planning horizon PH. Each task t may consist of subtasks which may be done in different activity periods AP so that a task may take more than one AP. Every AP may be considered as a limited time sub-horizon and may be associated with the limitations C-L, M-D, P-Q, T-Q etc.
The optimization problem is defined as follows: Attach as many tasks as possible to a number of active entities AE as small as possible in a way that all ASAP-ALAP conditions are kept and all other limitations are possibly not violated and exploited to the maximum possible extent. The more the limitations C-L, M-D, P-Q, T-Q etc. are exploited, the higher the number N-T of tasks $t \in$ TS that are attached to active entities $a_e \in AE$ and the smaller the number N-AE of active entities needed, the better the quality of the optimization result and the better the performance of the whole active resource reservoir. In fact, the performance exploitation of the limitations C-L, M-D, P-Q, T-Q etc. and the extent to which N-T is maximized and N-AE is

minimized are key performance indicators (KPIs) of the resource management problem. From the point of view of the optimization algorithm the KPIs described above are the optimization goals. Many of the KPIs to be optimized may be partly conflicting due to their limitations and are therefore negatively correlated. For instance, the qualification KPIs and both the movement distance KPIs and the capacity KPIs are usually partly conflicting. Also, the movement distance KPI and the termination time KPIs that express that the tasks are done in time are conflictive. On the other hand, the minimization of the movement distance KPI *M-D* and the KPI that expresses that the teams shall perform tasks close to their initial location *I-L* help each other and therefore correlate positively. There are some other positively correlated KPI goals, of course. A more detailed discussion of such optimization goals in a real-world optimization solution is given in [7] and [12] where special cases of ARM have been discussed.

5 Optimization of ARM using DMIG based on Example KPIs

The decision space of the resource management optimization consists of all possible assignments of tasks $t \in$ TS to active entities $a_e \in AE$ in the planning horizon PH within the planning periods PP. This means that the decision space is defined as TS×AE. Therefore we have k=card(TS)∗card(AE) and consequently an efficient heuristic DMIGk when we apply the optimization concept based on DMIG as described in section 4. The optimization algorithm is organized as an iteration over the set of possible assignments of active entities to tasks and terminates when all tasks have been assigned.

Compared to previous versions of this algorithm as described in [7] the learning of preference relations is a new relevant extension that is used to reduce the number of iterations without losing optimization quality.

In order to illustrate this, let us first repeat the description of the algorithm and then extend the description of how preference learning is integrated into it now. For both let us consider as example the KPI expressing the exploitation of the capacity limitation *C-L* and its modeling as optimization goal using DMIG as follows.

Let a_{ip}:=(t_{ip},t_{miq}) ∈ TS×AE := A (for A see definition *1a*)) be the decision alternative to attach the task t_p to the active a_{eq} in the iteration step i of DMIGk, $1 \leq i \leq k$, $i,k \in$ N, where k initially is limited by the maximum possible assignments of possible active entities to tasks and is decreasing with each iteration step and each assignment.

In order to evaluate whether or not a_{ip} is a good decision in the iteration step i we estimate that in case that *C-L* is still very far from the limit the evaluation shall be highly positive (*hp*). In case that it is yet significantly over the limit then the evaluation shall be highly negative (*hn*). Close to the limit the evaluation is positive with a rather small value (*sp*). Close behind the limit the evaluation is rather small negative (*sn*). Please note that *hp*, *sp*, *hn*, *sn* are some fuzzy values the definition of which is omitted here without losing generality of the concept.

With such an interpretation of the capacity exploitation KPI we are able to evaluate *C-L* by applying both the positive and the negative impact functions as defined in Def *1a)* and Def *1b)* and assign to the corresponding δ-values in $S_{C\text{-}L}$ and $D_{C\text{-}L}$ appropriate membership values for *hp*, *sp* and hn, *sp* etc.

In this way *C-L* is interpreted as a goal in the sense of DMIG (please note that all KPIs and therefore all limitations *M-D*, *P-Q*, *T-Q* etc. now can be interpreted in a similar way according to the semantic of the KPIs by defining the corresponding $S_{M\text{-}D}$, $D_{M\text{-}D}$, $S_{P\text{-}Q}$, $D_{P\text{-}Q}$, $S_{T\text{-}Q}$, $D_{T\text{-}Q}$, etc.).

Having done this, now DMIG is applied for the iteration steps i of DMIGk $1 \leq i \leq k$, $i,k \in$ N and the decision a_i=(t_i,a_{ei}) is made by selecting it out of the set A of decision alternatives a_{ip}=(t_{ip},a_{eiq}) ∈ TS×TM. The result for the i^{th} iteration step is the i^{th} component of the total solution o_d = (o_{d1}, … a_i=(t_i,a_{ei}) ..., o_{dk}) ∈ A*.

As extension of the optimization algorithm for each iteration step i the preference learning algorithm is applied in order to call the DMIG decision kernel only with adequate preference settings. The number of learned preference relations adequately limits (and reduces) the combinatorically possible number of calls of the DMIG decision kernel. Since in tendency only the learned preference settings are consistent with the interactions between the decision goals the risk of exploring inadequate regions of the decision space and loosing optimization quality is minimized. The risk (or better said the degree of certainty being on the right course) can be estimated depending on both the degree of membership the interactions between the goals have and the number of preference relations obtained. Thus, the risk can even be made explicit and numerically represented.

DMIGk terminates if a decision is made for all i, $1 \leq i \leq k$ or if the KPI limitations are exceeded.

Please note that k is an upper boundary for the number of iterations. In practical cases the number of iterations may be significantly lower than k, for instance by using information on existing constraints that reduce the optimization space. There are some similarities to [1].

However, in contrast to the latter in KPI oriented optimization there is no fixed hierarchy on the KPIs.

6 Learning of Preferences as a Link to Reinforcement-based Optimization

As described in section 5 the number of preference relations derived in any iteration step of the DMIG optimization algorithm reflects to which extent no risk of the limitation of the number of calls of the DMIG decision kernel with respect to the certainty of exploring the optimization goal space in the right direction is taken.

If these numbers are stored and associated with corresponding input data sets for every iteration step a kind of non-risk-risk trajectory of every run of the optimization algorithm is obtained. If we iterate over a sufficiently high number of sufficiently different input data sets with a sufficient number of optimization runs then a reinforcement information is learned. As such, the preference relations may be considered as goal-oriented reinforcement labels. These labels reflect during the optimization runs in which direction to go with respect to which optimization goals. If we extend the algorithm by a training component (for instance using a neural network) which learns these labels, then we obtain a novel trainable optimization algorithm.

In this sense the extended DMIG-based optimization algorithm becomes a reinforcement-based machine learning optimization algorithm that can be further developed towards a learning system similar to systems mastering games [16], for instance, but designed for optimization purposes.

7 Conclusions and Future Work

In this paper we described a learning algorithm that derives (at least partly) consistent preference relations from decision and optimization goals. It is shown how a former version of the algorithm is extended by integrating the learning of preference relations into the iteration process of the optimization algorithm.

We also discussed how learned preference relations help to adequately limit the number of iteration steps of the optimization and how this improvement links the algorithm to reinforcement-based machine learning optimization. Both to consider and to validate this will be the next steps and future work.

8 References

[1] M.C. Cooper, Reduction operations in fuzzy or valued constraint satisfaction, In: Fuzzy Sets and Systems, 2003, Vol.134, pp. 311-342.

[2] D. Dubois and H. Prade, On the use of aggregation operations in information fusion processes, In: Fuzzy Sets and Systems, 2004, Vol.142, pp. 143-161.

[3] A. Fallah Tehrani, W. Cheng, E. Hüllermeier, Preference Learning using the Choquet Integral: The Case of Multipartite Ranking. IEEE Transactions on Fuzzy Systems, 2012.

[4] R. Felix, Unscharfe Entscheidungen bei qualitativen Zielen Forschungsbericht Nr. 412 Universität Dortmund, Fachbereich Informatik, 1992.

[5] R. Felix, Relationships between goals in multiple attribute decision making. In: Fuzzy Sets and Systems, 1994, Vol.67, pp. 47-52.

[6] R. Felix, Aggregation of Partly Inconsistent Preference Information. In: Proceedings of the 13th International IPMU Conference, Dortmund, Germany, 2010, pp. 178 ff.

[7] R. Felix, Optimization of Partly Conflicting Goals in Complex Resource Planning. In: Proceedings of the 8th EUSFLAT Conference, Milano, Italy, 2013, pp. 669-674.

[8] R. Felix, On Separability of Preferences and Partly Consistent Induced Interacting Goals. In: Proceedings of the IFSA-EUSFLAT International Joint Conference, Gijón, Spain, 2015.

[9] S. Hamzeloo, M.Z. Jahromi, Decentralized Incremental Fuzzy Reinforcement Learning for Multi-Agent Sytems. In: International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems Vol. 28, No. 01, pp. 79-98, 2020.

[10] E. Hüllermeier, Fuzzy-Methods in Machine Learning and Data Mining: Status and Prospects. Fuzzy Sets and Systems, 2005, 156(3), 387-407.

[11] E. Hüllermeier, J. Fürnkranz, W. Cheng, K. Brinker, Label Ranking by Learning Pairwise Preferences. In: Artificial Intelligence, Vol. 172, Issues 16-17, Nov. 2008, pp. 1897-1916.

[12] E. Jaeker, R. Felix, D. Seifert and T. Epler, Multi-Criteria Optimization in Workforce Management. In: Proceedings of the 21st International Conference on Electricity Distribution, Frankfurt, Germany, 2011, Paper 0362.

[13] R. Jain, A Note on Arrow´s Impossibility Theorem, ECONOMIC ANNALS, Volume LX, No.207 / October-December 2015, 2015, pp. 39-48

[14] K. Kowsari, N. Bari, R. Vichr, F. Goodarzi, FSL-BM: Fuzzy Supervised Learning with Binary Meta-Feature for Classification. Proceedings FICC 2018, Singapore.

[15] F. Modave and M. Grabisch, Preference representation by a Choquet integral: Commensurability hypothesis. In: Proceedings of the 7th International IPMU Conference, Paris, France, 1998, pp. 164-171.

[16] D. Silver, A. Huang, et al.: Mastering the Game of Go with Deep Neural Networks and Tree Search. Nature 259, January 2016, 484-489.