Fuzzy Fusion Fairness Relations for the Evaluation of User Preference

Mario Köppen*, Jun Okamoto† and Aoi Honda*
*Kyushu Institute of Technology
680-4 Kawazu, Iizuka, Fukuoka 820-8502, Japan
Email: mkoeppen@ieee.org, aoi@cse.kyutech.ac.jp
†NTT Service Integration Laboratories, NTT Corporation
Email: okamoto.jun@lab.ntt.co.jp

***Abstract*—In this paper, formal representations for user preferences will be provided, which also take fairness in the achievement of probably conflicting goals into account. The analysis of the Bottleneck Flow Control (BFC) algorithm for congestion avoidance in traffic networks will serve as a base to specify extensions of relations among preference sets. In particular, these relations are the lexicographic minimum, the maxmin fairness, and the newly introduced ordered-ordered weighted averaging operator. All of them are shown to become, in a relational sense, maximized by the BFC algorithm. Their further expansion by introducing fuzzy fusion operators in their formal definitions than establishes a comprehensive sets of relations. The application of these relations to the Multiple Relation Analysis (MRA) of subjective video quality evaluation data is demonstrated and gives the conclusion that also fairness criterions are present in such evaluations, at least as a secondary preference criterion.**

***Keywords*-fairness; maxmin fairness; ordered weighted averaging operator; preference modelling; fuzzy information fusion**

I. Introduction

Preference modelling, and the related preference prediction have become research fields of increasing importance. This tendency has been established by the wide-spread use of the internet, social networks, and vendor systems, all of them comprising systems that receive a strong impact from the user's attitude. So far, the focus of user preference modelling was on either binary systems ("yes-no", "like-neutral") or numerical evaluations given by the user (ratings, stars). However, recent research is already focussing on the limitations of such a point a view. For example, CP-nets were introduced as graphical representations of conditional preferences [1]. Also older works that were not considered for a long time newly came under focus, as for example the Landau theorems on scoring sequences (i.e. necessary and sufficient conditions for scoring systems in a network context) [2]. In this work, we also want to promote the point of view to represent preferences by means of (set-theoretic) relations, in order to allow for a rich analysis of various aspects of user decisions. In particular, the goal is to extend the analysis to additionally represent (possibly hidden) trade-offs that are inherent in user preferences, by a formalization of the concept of user fairness. As a means for such an analysis, a field was choosen were much progress in the modelling of fairness was achieved for a long time, i.e. the introduction of maxmin fairness in the field of data communication [3]. By a rigorous analysis of one of the fundamental algorithms in this field, the so-called Bottleneck Flow Control (BFC) algorithm [4] for congestion avoidance in traffic networks, we could find opportunities for the generalization of relations that are established by this algorithm. The generalization of these relations can be successfully achieved by employing concepts from fuzzy fusion. At the end of our exposition, we will have a suite of formal relations at hand that is capable to issue a Multiple Relation Analysis (MRA) for general data analysis, also taking the attitude of users towards unachievable goals into account. The MRA will be demonstrated on a subjective video quality evaluation dataset.

This paper is organized as follows: the following Section II will recall the BFC algorithm, and discuss the relevant aspects of the state that is achieved at the end of this algorithm. These characteristics of the algorithm will guide to the specification of a suite of formal relations for MRA in Section III, and the application of this suite to the video quality dataset will be given in Section IV. The paper ends with a short conclusion section.

II. Fairness Relations

In this section, we will focus on the formal representations of fairness. In general, fairness, as a daily life concept, can be understood in many ways and fashions. Problems with global optimization easily lead to a motivation to also take fairness into account [5][6]. We can find various attempts to represent fairness in the past, but mostly there was no notable progress. For example, in processor scheduling tasks, often related to hard combinatorial optimization problems, the concept of n-fairness was introduced, with the notion meaning that in a processor regime, each processor should ensure that the processor before can perform at least n tasks [7]. Also in common means like credit granting, fairness can be used as an indicator for a relevant assessment. Here, the criterion could be that two customers that have the same evaluation criterions should receive the grant with the same likelihood.

However, all such specifications do not guide to a formal representation of fairness in such a way that it can be effectively used for the finding of suitable "fair" states. Here, the best progress so far was achieved in the field of data communications. Starting with the proposal of the Bottleneck

Flow Control (BFC) algorithm, an effective way of a fair resource assessment was introduced, probably for the first time [4].

A. The Bottleneck Flow Control Algorithm

We assume a traffic network congestion avoidance problem. A network for carrying traffic is given as an un-directed graph G with nodes N and links L. Also, a *maximum capacity* is assigned to each link. Then, there are a number of users that want to send traffic units through this network. Thus, also a set of n triples (Π_i, s_i, r_i) of sender-receiver pairs and paths connecting them is given, where s_i represents the sending node, r_i the receiver node for user u_i. Paths Π_i are given as a sequence of joined links starting from s_i and ending at r_i. We also consider the union of all links l_i used by all paths, each link with a multiplicity w_i according to the number of paths using the link l_i. Then the Bottleneck Flow Control algorithm assigns traffic amounts t_i to all users u_i in the following way:

Bottleneck Flow Control

---

1) Set the *remaining paths* to the set of all paths. Set the traffics t_i for users u_i along their corr. paths Π_i to 0.
2) While the *remaining paths* set is not empty, perform the following steps:
3) For all links l_i used by the *remaining paths*, get the number w_i of paths that pass through this link.
4) Find the links with minimum value of $m_i = c_i/w_i$.
5) Add m_i to the traffics for all users through the links with minimal m_i.
6) Remove the paths of all users of links, for which m_i is minimal, from the *remaining paths*.
7) Set new capacities of network links $c_i \rightarrow c_i - m_i * w_i$.

An example for the BFC algorithm is given in Fig. 1. It shows a network with 7 nodes and links between these nodes as indicated in the figure. It is assumed that the link connecting nodes 3 and 4 has a maximum capacity of 200 units, and the link connecting nodes 4 and 7 has a maximum capacity of 100 units. All other links will have some higher maximum capacity. Three users wants to send traffic through this network: user 1 sends traffic t_1 via the nodes 1, 3, 4 and 7; user 2 sends traffic t_2 via the nodes 2, 4 and 7; and user 3 sends traffic t_3 via 5, 3, 4 and 6. It means that some users have to share links for their traffic: users 1 and 3 share the link between nodes 3 and 4, and users 1 and 2 share the link between nodes 4 and 7. The BFC algorithm now will assign specific values for the traffic amounts t_1, t_2 and t_3, starting with amount 0, and increasing equally for a subgroup of users. At the beginning (Level 0) the amount will increase for all users. In some later stage, for example at Level 20, the traffic amount assignment of 20 (units) to all users is still feasible. However, at Level 50, the link between nodes 4 and 7 has to transport a total traffic of 100: 50 from user 1, and 50 from user 2. This gives a so-called *bottleneck*. Any attempt to further increase the traffic for either user 1 or user 2 will result in exceeding the maximum capacity of this link. Thus, the BFC algorithm stops to further

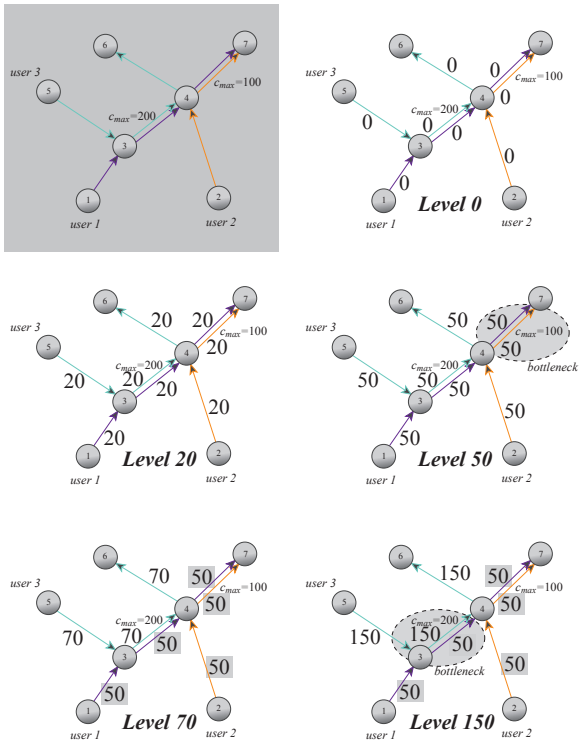

Fig. 1. Example for the Bottleneck Flow Control Algorithm: the algorithm starts traffic assignments for users along given paths at level 0, and gradually increases this level until bottlenecks are appearing. Then, the algorithm stops the further increase for the affected users, but continues for the other users. Thus, it is attained to assign the same traffic to subsets of users as long as possible. This algorithm also utilizes the specific network in the best way.

increase the traffic amount for users 1 and 2, and fixes the assignment $t_1 = 50$ and $t_2 = 50$.

However, user 3 is not affected by this bottleneck, since her traffic is not using the link between nodes 4 and 7. So, the BFC algorithm continues. Later on, for example at Level 70, the traffic assignment $t_3 = 70$ (while keeping $t_1 = t_2 = 50$) is still feasible. This level increase will continue until Level 150. Now, the total traffic for the link between nodes 3 and 4 becomes 200: user 1 was fixed before at the level $t_1 = 50$, and $t_3 = 150$ for user 3.

Any further increase of traffic is not possible, and the BFC algorithm stops. Thus, the final traffic assignment is $t_1 = 50, t_2 = 50, t_3 = 150$. In the implementation of this algorithm, of course, there are no increasing level sets, as the values of bottlenecks can be directly inferred from the network configuration and the values of maximum capacities.

It has to be noted that the BFC algorithm also specifies the feasible space of traffic assignments. We can write the result of the algorithm by a set of linear equalities (bottleneck equations):

$$\begin{aligned}
\lambda_{11}t_1 + \cdots + \lambda_{1n}t_n &= b_1 \\
\lambda_{21}t_1 + \cdots + \lambda_{2n}t_n &= b_2 \\
&\cdots \\
\lambda_{k1}t_1 + \cdots + \lambda_{kn}t_n &= b_k
\end{aligned} \tag{1}$$

where all $\lambda_{ij} \in \{0, 1\}$, t_i is the traffic amount allocated to the sender-receiver pair (s_i, r_i) and k is the number of bottlenecks found by the BFC algorithm, in the same order by which they are found by the algorithm. It can also be seen that the t_i can be numbered (arranged) in such a way that the set of equations fulfills the following properties:

- In row 1, all t_i with $\lambda_{1i} = 1$ are equal.
- In each row $r > 1$ there exists a "border" index i_r such that for all $i < i_r$ and $\lambda_{ri} = 1$ in at least one row $r_1 < r$ also $\lambda_{r_1,i} = 1$, and for $i \geq i_r$, all t_i are equal, larger than any t_j with $j \leq i_r$, and there is an index $r_2 \leq r$ such that $\lambda_{r_1,i} = 0$ for all $r_1 < r_2$ and $\lambda_{r_1,i} = 1$ for all $r_2 \leq r_1 < r$. This means that the traffics comprising a particular bottleneck equation (their λ-values are 1) can be divided into two groups: the first group contains only traffic amounts already assigned by foregoing bottleneck equations. Thus, in the progress of the algorithm, while reaching another maximum capacity (bottleneck), the remaining traffic amount at that bottleneck will be equally shared among another new group of sender-receiver pairs. These traffics then comprise the second group. Note that the first group might be empty.
- If $i > j$ then for the largest k_i and k_j from $\{1, \ldots, n\}$ with $\lambda_{i,k_i} = 1$ and $\lambda_{j,k_j} = 1$ resp. it holds that $t_{k_i} \geq t_{k_j}$. This means the last elements in the equations are sorted in non-decreasing order, according to the flow of the BFC algorithm.

For clarity (we will need this structure of the bottleneck equations later on) here the bottleneck equations for the example given above:

$$\begin{aligned}
t_1|_{=50} + t_2|_{=50} &= 100 \\
t_1|_{=50} + t_3|_{=150} &= 200
\end{aligned}$$

thus, for the second row the index i_2 is 2: all traffics appearing in the equation (i.e. their λ factor is 1 and not 0) with lower index appear in at least one foregoing bottleneck equation (here, t_1 appears in the first equation and was fixed to 50), and the remaining capacity of $200 - 50 = 150$ at the second bottleneck will be equally shared for the third traffic. Since there is only one sender-receiver pair, its value will become 150.

Writing the bottleneck equations in such an index-ordered way has the additional advantage that, after replacing each equal sign in the Eqns. (1) with a lower-or-equal sign specifies the feasible space of all possible traffic assignments.

After this analysis of the BFC algorithm, the question is about the characterization of the final state achieved by the algorithm. The BFC algorithm is effective in the sense that for any given network structure, and given sender-receiver pairs, it will uniquely assign traffic amounts to all sender-receiver pairs, and thus select one and only one element of the feasible space. How is this element be qualified against all other elements of the feasible space?

In the following, we will recall two already known notations of that final state, and provide another (up to our best knowledge) new characteristic. Then, we will discuss the extension of these characteristics into the fuzzy fusion domain.

B. Characterization of the BFC result

There are several ways to characterize the final state of the BFC algorithm. Here, we will focus on three ways. In a less formal way, the BFC algorithm can be seen as a prototype to achieve fairness in the assignment of traffic amounts among the various users (i.e. sender-receiver pairs) in such a network model. The algorithm assigns the same traffic to all user as long as possible. Then, when this increase is no longer possible, as the assignment is reaching a bottleneck, it is no longer possible to give a larger same amount to all users. But, in contrary to other "equalizing" approaches to fairness, the BFC algorithm does not stop here. It acknowledges the fact that a number of users is *not* affected by this bottleneck, and continues to raise their traffic amounts equally — until the next bottleneck is reached.

From this argument, we can see the first characteristic of the BFC algorithm.

(1) **Lexmin relation**. The lexmin relation is defined as a relation between two points from R^n. Given two vectors x and y from R^n, we consider the coordinates of both vectors sorted in increasing order. It is said that $x >_{lexmin} y$ if and only if the first coordinate of x, in that sorting, which is different from y, is larger than the corresponding coordinate of y, in that sorting. For simplicity, if the smallest coordinate of x is larger than the smallest coordinate of y or if both are equal but the second-smallest coordinate of x is larger than the second-smallest coordinate of y etc. - than x is lexmin-related to y.

We also want to recall here that for any relation $>_R$, formally seen as a subset of $A \times A$, where A is an arbitrary but fixed set, we can define a *maximum set* and a *best set*: the maximum set contains all elements x from A such that there is no different $y \neq x$ in A with $y >_R x$. The best set, on the other hand, contains all elements x of A such that for each different $y \neq x$ $x >_R y$ holds.

In this sense, the best set of the feasible space of a network traffic assignment problem under the lexmin relation contains exactly one element, and this is the one found by the BFC algorithm. This can be seen from the index-ordered bottleneck equations: the minimum traffic is given in the first of the equations, and it cannot be increased. Then, for states with the same minimum traffic, the second-smallest traffic will appear in the second equation. Also this one cannot be increased without exceeding the second bottleneck. The argument can be continued for the third-smallest traffic, fourth-smallest traffic etc.

(2) **Maxmin fair dominance relation**. It has already been established [3][8] that the BFC algorithm is selecting the best set element of another relation, the so-called *maxmin fairness dominance*.

Definition 1. *An element x of the feasible space is* maxmin fair dominating *an element y of same space, if for each component y_i of y, which is larger than the corresponding component x_i of x there is another component x_j of x which is (already) smaller than or equal to x_i and such that y_j is smaller than x_j.*

This can be directly seen from the index-ordered bottleneck equations. If one traffic is increased, there will be a bottleneck equation r where this traffic appears for the first time (and there belonging to the second group, i.e. with index $i \geq i_r$, using former convention). To maintain the bottleneck equation, another traffic has to be decreased in order to compensate for the increase (note that the compensation can be larger than the increase). But each other traffic in this equation will be the same (before increase) or smaller (if belonging to the first group with indizes $i < i_r$). So, any increase of a traffic amount can only happen if some other traffic amount, which is already the same or smaller, is decreased. This is exactly the definition of the maxmin fair dominance relation. Note that maxmin fair dominance relation and lexmin relation are different relations.

In the next subsection, we will introduce a generalization of the maxmin fair dominance relation, based on using fuzzy fusion operators. Here, we will already note that *maxmin fairness* (referring to the quality of the best element of this relation) has already found a number of alternatives in the data network domain (where it was first introduced). The most popular one is the so-called *proportional fairness*. It can be defined as follows:

Definition 2. *A point x of feasible space (a subset of R_n^+) is considered proportional fair dominating another feasible point y, if and only if*

$$\sum_{i=1,\ldots,n} \frac{y_i - x_i}{x_i} \leq 0 \tag{2}$$

holds.

The reason to introduce proportional fairness were a number of problems with maxmin fairness, including the negligence of user utility, and the overvaluation of the least element, for example as it was noted by Kelly[9]: "*The maxmin fairness criterion gives an absolute priority to the smaller flows, in the sense that if $x_{s^*} < x_s$ then no increase in x_s, no matter how large, can compensate for any decrease in x_{s^*}, no matter how small.*"

We can find the further extension of proportional fairness to the so-called weighted alpha fairness dominance relation:

Definition 3. *A point x of feasible space (a subset of R_n^+) is considered weighted alpha fair dominating another feasible point y, if and only if for a given fixed set of weights w_i*

$$\sum_{i=1,\ldots,n} w_i \frac{y_i - x_i}{x_i^\alpha} \leq 0 \tag{3}$$

holds.

It can be shown that for $\alpha \to \infty$ alpha-fairness nearly everywhere converges to maxmin fairness, but equals proportional fairness for $\alpha = 1$.

(3) **Exponential OOWA**. While the former two definitions show how to use a relation between points in the feasible space, and to characterize the BFC algorithm as the one that finds the best element for each relation, the question comes up if there is also a scalar function that is directly maximized by the BFC algorithm. As a new result, we will show that there is such a function. We define a special case of the ordered weighted averaging (OWA) operator. As a reminder, the OWA of a point x of R_n, given a weight vector $w \in R_n$, is defined as $\sum w_i x_{(i)}$. In this expression, $x_{(i)}$ indicates the i-th largest element of all coordinates of x. We specialize the OWA by also requiring the weights to be sorted in the opposite order.

Definition 4. *Given a point x from R_n and a set of weights $w \in R_n$, the Ordered-Ordered Weighted Averaging (OOWA) of x by w is defined as*

$$OOWA_w(x) = \sum_{i=1}^{n} w_{(n-i+1)} x_{(i)} \tag{4}$$

Thus, in the OOWA, the largest value is multiplied with the smallest weight, the second-largest value with the second-smallest weight etc. As a special case of the OOWA, we also introduce the exponential OOWA. The additional requirement here is

$$w_{(i)} > \sum_{k=1}^{i-1} w_{(k)} \tag{5}$$

so that the weights itself are exponentially increasing. A possible choice is $w_i = 2^i$ for any n. Then it holds:

Theorem 1. *The BFC algorithm is maximizing any exponential OOWA.*

For an outline of the proof, see the Appendix.

Back to the example from the former subsection, and using the weights $(1, 2, 4)$ to compute

$$OOWA_{exp}(t_1, t_2, t_3) = t_{(1)} + 2t_{(2)} + 4t_{(3)} \tag{6}$$

we can see that any feasible modification of the t_i will reduce this expression for the state $(50, 50, 150)$ assigned by the BFC algorithm. This state has the exponential OOWA value $1 \cdot 150 + 2 \cdot 50 + 4 \cdot 50 = 450$. If for example t_2 is increased, this has to be compensated by a decrease of t_1. The increase can be at most 50 due to the first bottleneck at 100, and t_2 will always become the second largest traffic, and the exponential OOWA is changing to

$$f^* = 1 \cdot 150 + 2 \cdot (50 + \delta) + 4 \cdot (50 - \delta_1) \tag{7}$$

where $\delta_1 \geq \delta$. This expression is clearly smaller than 450, as it reduces the former value by at least 2δ. If t_1 is increased, than

t_2 and t_3 have to be decreased, and the argument is basically the same. An increase of t_3 forces a decrease of t_1. Since the increased t_3 will still be the largest value, and thus weighted by 1, the smaller t_1 now will be weighted by 2 or 4 (which depends on a possible modification of t_2). So, also here the net change in the exponential OOWA expression will be negative.

The exponential condition is also necessary. It suffices to consider a network structure, where one traffic shares links with k other users, all of same maximum capacity and exactly one other user per link to share, and the remaining $n-k$ links will have only one sending user, all with a lower maximum capacity. Then, an increase of all the k sharing users will require the one sharing user to compensate all k traffics. This, its weight must be at least as large as the sum of the weights of all other k increased traffics.

As a last comment, we may note that the exponential condition is a result of link sharing. With regard to fairness, single user sharing links with many other users at once can be considered the worst case, as assigning any traffic to such a user affects the traffic of many other users, in fact enforces their reduction.

III. Suite of Relations

In order to apply the concepts that were presented in the foregoing section, we will have to study the unrevealed relations independently from the BFC algorithm and the underlying network traffic assignment problem. In fact, there are a number of formal ways to expand these definitions, and they will be discussed in the following.

A. Maxmin fairness and fuzzy fusion

The definition of maxmin fair dominance is rather complex. However, its implementation can be much simplified [5]: in general it would need to check the fairness condition for all pairs of components, but for simplification, we define index sets as follows (see Fig. 2). Given two vectors x and y we want to test if x maxmin fair dominates y. Then, we compute

- $A = \{i \,|\, x_i > y_i\}$
- $B = \{i \,|\, x_i = y_i\}$
- $C = \{i \,|\, x_i < y_i\}$
- $min_A = \min_{i \in A} x_i$, or ∞ if A is empty
- $min_C = \min_{i \in C} x_i$, or ∞ if C is empty

The definition of maxmin fairness requires to check the following for every increasing component x_i: is there any $j \neq i$ such that $x_j \leq x_i$ and $y_j < x_j$? It means that $i \in C$ and $j \in A$. We omit the case that either A or C is empty at the moment. Then, for such an index j to exist, it is necessary that for every $i \in C$ there is at least one element j of A with $x_j \leq x_i$, in particular it is necessary that there is a $j \leq min_C$ and thus $min_A \leq min_C$. We can see that this is a sufficient condition as well: for any i from C, already one j with $x_j = min_A$ fulfills the check: the j-th component (min_A) is smaller than or equal to any x_i with $i \in C$, and $x_j > y_j$ since $j \in A$. If we also take formally $z < \infty$ for any real z, we can evaluate maxmin fair dominance as follows: *Given two vectors x and y, then x will maxmin fair dominate y if and only if $min_A \leq min_C$.*

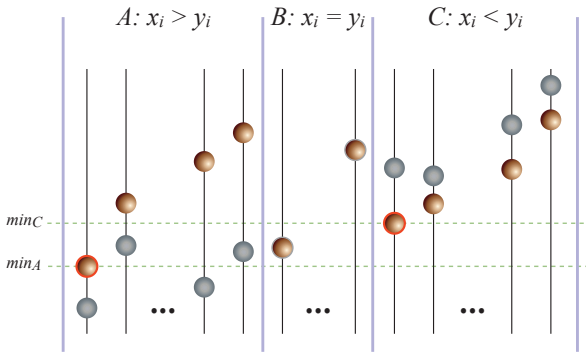

Fig. 2. Maxmin fairness relation can be more easily defined, if we consider the sets of components of x larger (set A) and smaller (set C) than corresponding components of y. Vector x will maxmin fair dominate y if and only if the smallest larger component of x is smaller than or equal to the smallest smaller component of x [5]. In the figure, corresponding components of x and y are placed on the same line, with height indicating their magnitude, and the lines are re-arranged to show first x-components larger than the corr. y-component, then equal components, and third smaller components (each of the three sections are additionally arranged by increasing size of the corr. x-component).

Note that equal components do not have any influence on maxmin fair dominance, and that the definition includes Pareto dominance. If C is empty, but A not, then each component of x is larger than or equal to the corresponding component of y, and at least one is larger.

From this way of testing the maxmin fair dominance relation, we can see that the relation actually refers to a fusion of all smaller and all larger components of a vector, compared to another vector. The fusion used here is the minimum operation. In this regard, we consider an extension of the maxmin fair dominance by formally generalizing on the fusion aspect. Thus, instead of comparing $\min A$ and $\min C$ we use other means to fuse the information of the sets A and C into single values. In present study, we will use other t-norms (incl. Dubois Prade t-norm, and algebraic t-norm), and OWA operators with different weights.

B. OOWA

Also the exponential OOWA that was shown to be maximized by the BFC algorithm offers some straightforward extensions, esp. by requiring a different law of weights increase. Here, we will consider also a linear OOWA, where weight i is choosen to be i.

Such operations will reflect sharing aspects. In the network problem, the exponential increase of weights was necessary in order to ensure that even the situation where one sender shares links with ALL other users is covered. In the simple scenario of no link sharing at all, a linear OOWA would be sufficient to ensure maximization by BFC algorithm. Thus, comparing linear and exponential OOWA in a situation, where the interpretation of "link sharing" is not obvious, it can still refer to a relevant problem of the studied feasible space -

technically, these expressions can be computed without any reference to a networking problem at all.

C. Data processing suite of relations

With the modifications mentioned in the former subsections, we propose a suite of relations that can be used together for a Multiple Relation Analysis.

- Fuzzy fusion fairness relations fff_{op}, where op indicates the used fuzzy fusion operator. To decide for two points x and y whether $x >_R y$ or not, we first compute the set A of all components of x that are larger than the corr. component of y, and the set C of all components of x that are smaller than the corr. component of y. Then, if A is not-empty, op is applied to A, giving a value f_A, and if C is not-empty, op is applied to the set C, giving a value f_C. If either of these sets is empty, the corr. f-value is set to ∞. Then, $x >_R y$ if and only if $f_A \leq f_C$. More specifically, we will consider Dubois-Prade T-norm for fusion (which is for two reals a and b from $(0, 1)$ and a parameter α defined as $ab/\max[a, b, \alpha]$, and for more than two arguments by associativity of the expression for two arguments), the "simple" algebraic t-norm as product of components, and several OWA operators, with larger weights for smaller values, or nearly equal weights, thus focussing on averaging. Also, plain average of the sets A and C will be considered.
- Other fairness relations, as introduced in subsection II-B(2): proportional fair dominance, and alpha fair dominance with various values of α.
- Exponential OOWA, as introduced in section II-B(3), the variant linear OOWA as discussed in the foregoing subsection, and in this context also the lexmin relation.
- Other general relations that are not primarily related to the concept of fairness: more than half of the components of x is larger than the corresponding components of y; the largest component of x is larger than the largest component of y; and the average of the components of x is larger than the average of the components of y in order to x be in relation to y.
- We also consider Pareto dominance relation with regard to maximization and minimization. For maximization, x is larger than (or: in relation to) y iff all components of x are larger or equal to the corresponding components of y, and at least one component is larger (minimization Pareto dominance is defined accordingly).

D. Mathematical properties

For space reasons, we will not provide a thorough analysis of the mathematical properties of the introduced relations. Here we only mention that the introduced fuzzy fusion fairness relations are all not complete (i.e. it is possible that for some x and y neither $x >_R y$ nor $y >_R x$ holds) and not (always) transitive. For example, for the maxmin fair dominance relation, consider the tripel of vectors $x = (70, 50), y = (40, 40), z = (30, 60)$. Here x maxmin fair dominates y, and y maxmin fair dominates z but x does not maxmin fair dominate z. Nevertheless, the relations will not produce conflicts, as from $x >_R y$ and $y >_R z$ always $z >_R x$ can be excluded. On the other hand, the lexmin relation as well as the OOWA are complete and transitive.

IV. Experimental Validation: Subjective Evaluation of Video Quality

A. Used data and recent findings

For the application of the relations, we are considering a dataset of subjective evaluations of video data quality. The dataset contains two kinds of information for 1139 sample videos: five objective measures of video quality, in particular values numerically representing overall noise, degradation caused by block distortion, degradation associated with blurring, local spatial degradation, and freeze degradation, and an average ranking of a number of users for each of these 5-tuples of data. The dataset is part of a larger experiment performed at NTT Corp.[1]

The data have already been analyzed in a recent publication[10]. There, the goal was to use a newly-proposed fuzzy integral, the inclusion-exclusion integral, to derive the user ranking from the quality measures by direct computation. The approach was shown to improve the recommended rule by the Telecommunication Standardization Sector of International Telecommunication Union (ITU-T), and an approach based on the Choquet integral as well. It was also confirmed that the 5-th quality measure shows a logarithmic utility for the user, while the other four correspond directly to the user utility.

B. Data Pre-Processing

Here, we are interested in specifying the user attitude towards increase or decrease of the quality measures in total. While it can be expected that video material with lower degradation measures will be more likely higher ranked, the question that can be handled by the introduced fairness relations is also about the acceptance of trade-offs by the user. So, the question for the analysis is if an increase in quality according to one criterion is less favored, if the cost for this increase is the decrease of another criterion, which is already lower in quality.

For this processing, we have to represent the quality values by their utility. Also we have to consider that the introduced relations are focussing on increasing magnitudes of components. Thus, the following pre-processing of the data was performed:

1) Data sets containing unsafe measurements (about 20) were removed.
2) All 5-th measurements x_{i5} were replaced by the decimal logarithm $\log_{10} x_{i5}$.
3) All datasets were normalized by their average and standard deviation (μ_j is the average of the j-measurement, and σ_j the corresponding standard deviation):
$$x_{ij} \leftarrow \frac{x_{ij} - \mu_j}{2\sigma_j} \tag{8}$$
4) Datasets with strong outliers were removed (about 200), as we are more focussing on the average behaviour.

[1]The dataset itself is not publicly available.

5) The single data were rescaled to the $(0, 1)$ range, such that larger values represent better quality:

$$x_{ij} \leftarrow 1 - \frac{x_{ij} + 1}{2} \tag{9}$$

After preprocessing, the frequency correlation for each proposed relation was computed. Thus, we are considering the normalized measurements for each possible pair (x, y) of datasets in two ways:

- Precision: in case that x is in relation to y, we are counting how often then also $s_x > s_y$ holds, i.e. how often the users also ranked the video higher.
- Recall: in case that $s_x > s_y$, we count how often this is indication for the relation $x >_R y$ to hold as well.

In the processing, equal pairs of values were ignored. It was also noted how often the relation was actually occurring. The total number of pairs used in the comparison for each relation was 938961.

C. Results and Discussion

The result for all used relations can be seen in Table I.

We describe the main properties of the result by the following items, see also Fig. 3.

- Most of the relations achieve precision and recall measures between 0.1 and 0.8. The largest precision value is achieved by the Pareto dominance (maximizing), but it is accompanied by a low recall value. This indicates that an improvement in ALL measures usually also results in a higher rating of the videos, but that it is not required from the user perspective. The low measures for the Pareto dominance minimizing strengthen this conclusion: a higher rating of a video with all lower quality measures is highly unlikely. Note the small number of occurrences of the Pareto dominance relation as well.
- We find the fuzzy fusion fairness relations located in the lower right part of the precision-recall plot, in an area with the maxmin fair dominance relation (as a special case of fuzzy fusion fairness) in its center. This indicates that these relations are focussing on related aspects of user preferences. Remember that these relations are focussing on the modality of accepting a decrease in some quality measures for the benefit of increasing other measures. In a fair state, the decrease should not happen to quality measures that are already lower. The rather low recall values indicates that the user is not considering this a primary criterion, and is more focussing on general improvements. Nevertheless, the recall values can become rather large, esp. when the algebraic t-norm is used for fuzzy fusion. This is a relevant issue, as most t-norms have the tendency to depend (in their numerical range) strongly from the number of arguments, and this effect is most extreme for the product. It can also be seen from the tendency of the Dubois-Prade t-norm to provide larger recall values with increasing α parameter. It is known that this parameter smoothly transits the operation result from the minimum ($\alpha \to 0$) to the product of its arguments for $\alpha \to 1$.
- We can find OOWA and lexmin rather close together, and with comparable large precision and recall values. Especially, there is the same result for exponential and linear OOWA. Recalling the discussion of the introduction of these operators, and their motivation from the maximizing property of the BFC algorithm, this is an indication that there is no strong "sharing" between the quality measures. It means that the increase of a quality measure can be gracefully done without the need to decrease other quality measures. On the other hand, lexmin can also be seen as an OOWA with weights converging to infinity, and the results are rather similar, just a little bit smaller.
- Averaging relations in general do perform less well, compared to other relations. Especially for the fuzzy fusion fairness with nearly equal OWA weights it gives the lowest precision and recall values (except Pareto minimum). Also the largest component alone is not much indicative for the user rating.
- The default suite of relations, esp. the number of larger components relation, confirms the findings from the other relations: the number of larger quality measures can be seen as the primary criterion for the user rating, and the corr. trade-off as secondary criterion. But it seems in this trade-off the user is not much considering the choice of which of the other measures is actually decreased.

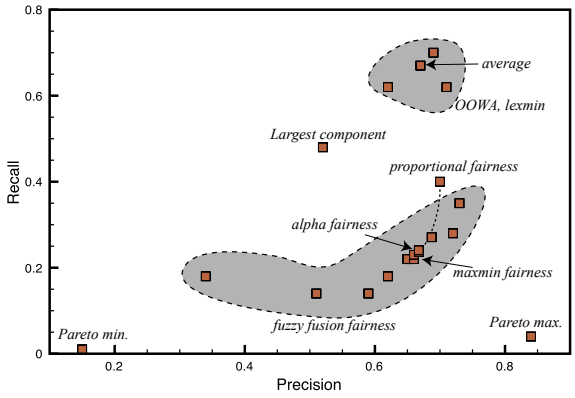

Fig. 3. Precision vs. Recall for the used relations.

V. Conclusions

We have provided formal representations for user preferences, also taking fairness among the achievement of probably conflicting goals into account. Their further expansion by introducing fuzzy fusion operators in their formal definitions than established a comprehensive set of relations. The application of these relations for a Multiple Relation Analysis (MRA) for the analysis of subjective video quality evaluation data is demonstrated and gives the conclusion that also fairness criterions are present in such evaluations, at least as a secondary preference criterion. Further work will have to focus on the

TABLE I
PRECISION AND RECALL FOR VARIOUS RELATION-BASED EVALUATIONS.

Relation	Parameter	Number of related pairs	Total ratio	Precision	Recall
maxMinFairness = fff_{min}	-	159024	0.110257	0.651015	0.223613
fff_{owa}	$w = (0.8, 0.2, 0, 0, 0)$	130073	0.0863007	0.622981	0.175027
fff_{owa}	$w = (0.6, 0.3, 0.1, 0, 0)$	123956	0.0675385	0.511601	0.136976
fff_{owa}	$w = (0.3, 0.2, 0.2, 0.2, 0.1)$	242104	0.0875755	0.339647	0.177613
$\mathrm{fff}_{dubois-prade}$	$\alpha = 0.2$	159010	0.110258	0.651078	0.223616
$\mathrm{fff}_{dubois-prade}$	$\alpha = 0.5$	156492	0.109369	0.656219	0.221812
$\mathrm{fff}_{dubois-prade}$	$\alpha = 0.8$	183208	0.140454	0.719843	0.284857
$\mathrm{fff}_{algebraic}$	-	222989	0.174354	0.734171	0.35361
$\mathrm{fff}_{average}$	-	111732	0.0702915	0.590708	0.142559
proportional fairness	-	265724	0.197693	0.698567	0.400943
alpha fairness	alpha=3	186517	0.136353	0.686425	0.276539
alpha fairness	alpha=7	164952	0.117707	0.670025	0.238722
alpha fairness	alpha=10	161627	0.114446	0.664864	0.232109
exp. OOWA	-	468996	0.325803	0.652278	0.660764
lin. OOWA	-	468996	0.325803	0.652278	0.660764
lexmin	-	468996	0.308115	0.616867	0.624892
Pareto dominance (maximizing)	-	20341	0.0182233	0.841207	0.036959
Pareto dominance (minimizing)	-	20341	0.00316414	0.14606	0.00641722
number of larger components	-	401481	0.304671	0.712547	0.617906
largest component	-	426865	0.237688	0.522835	0.482058
average	-	468996	0.332448	0.665583	0.674242

more elaborate analysis of mathematical properties of these relations, and their employment in other domains, especially preference prediction.

ACKNOWLEDGMENT

This work was partly supported by the Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research (S) (No. 18100001).

REFERENCES

[1] C. Boutilier, R. I. Brafman, C. Domshlak, H. H. Hoos, and D. Poole, "Preference-based constrained optimization with cp-nets," *Computational Intelligence*, vol. 20, no. 2, pp. 137–157, 2004. [Online]. Available: http://dx.doi.org/10.1111/j.0824-7935.2004.00234.x

[2] H. Landau, "On dominance relations and the structure of animal societies: III the condition for a score structure," *Bulletin of Mathematical Biology*, vol. 15, pp. 143–148, 1953, 10.1007/BF02476378. [Online]. Available: http://dx.doi.org/10.1007/BF02476378

[3] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, N: Prentice Hall, 1992.

[4] J. Jaffe, "Bottleneck flow control," *IEEE Trans. Commun.*, vol. COM-29, July 1981.

[5] M. Koeppen, K. Yoshida, and M. Tsuru, "Fuzzification of maxmin fairness relation based on subvector dominance degree," in *Proc. 2nd International Conference on Intelligent Networking and Collaborative Systems (INCoS 2010), Thessaloniki, Greece*, November 2010.

[6] M. Koeppen, R. Verschae, K. Yoshida, and M. Tsuru, "Heuristic maxmin fairness for the wireless channel allocation problem," in *Proc. Fifth International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA 2010), Nov. 2010, Fukuoka, Japan*, 2010.

[7] S. Haldar and D. K. Subramanian, "Fairness in processor scheduling in time sharing systems," *SIGOPS Oper. Syst. Rev.*, vol. 25, no. 1, pp. 4–18, 1991.

[8] M. Koeppen, K. Yoshida, M. Tsuru, and Y. Oie, "Evolutionary routing-path selection in congested communication networks," in *Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics, San Antonio, TX, USA - October 2009*, 2009, pp. 2224–2229.

[9] F. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Telecomm.*, vol. 8, pp. 33–37, Jan./Feb. 1997.

[10] A. Honda and J. Okamoto, "Inclusion-exclusion integral and its application to subjective video quality estimation," in *Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Methods - 13th International Conference, IPMU 2010, Dortmund, Germany, June 28 - July 2, 2010. Proceedings, Part I*, E. Hüllermeier, R. Kruse, and F. Hoffmann, Eds., 2010, pp. 480–489.

APPENDIX

Outline of the proof for Theorem 1

The following lemma indicates a minimizing property of the OOWA against all OWA using the same set of weights:

Lemma 1. *Given a set of n weights w in non-decreasing order, and any permutation $w*$ of these weights. Then for any $x \in R^n$*

$$OOWA_w(x) \leq OWA_{w*}(x) \tag{10}$$

Proof: We will use bracketed subscripts to refer to the different orderings. By $x_{(i)}$ we indicate the i-th largest element of x, by $w_{(i*)}$ the i-th element in the permutation $w*$ of the weights. We also define $h_i = x_{(n-i+1)} - x_{(n-i+2)}$ for $i > 1$ and $h_1 = x_{(n)}$. Then, $x_{(i)} = \sum_{k=1}^{n-i+1} h_k$ and it follows:

$$
\begin{aligned}
OWA_{w*}(x) &= \sum_{i=1}^{n}\left[w_{(i*)}\sum_{k=1}^{n-i+1}h_k\right] \\
= \sum_{i=1}^{n}\left[h_i\sum_{k=1}^{n-i+1}w_{(k*)}\right] &\geq \sum_{i=1}^{n}\left[h_i\sum_{k=1}^{n-i+1}w_k\right] \\
= \sum_{i=1}^{n}\left[w_i\sum_{k=1}^{n-i+1}h_k\right] &= OOWA_w(x)
\end{aligned}
$$

since the weights are sorted in non-decreasing order and thus, the sum of the first k weights will be always smaller or equal to the sum of the first k permuted weights. ■

We indicate the assignment T^* of traffics achieved by the BFC algorithm with a star t_i^*. Now we have to consider any other feasible assignment T of t_i traffic values to users. By T^+ we indicate the set of traffics that become larger in T and by T^- the set of values that become smaller (if needed, arrange the traffic indices such that for equal t^* values, T^+ traffics appear before T^- traffics). We consider a bipartite directed graph with nodes set (T^+, T^-) constructed as follows: each node represents a traffic that was modified when changing from T^* to T and keeps two values: the change δ and the weight of the traffic in the original order O. A directed link goes from a node t_1 in the "right" set of the T^--nodes to a node t_2 in the "left" set of the T^+-nodes if t_1 appears in the same bottleneck equation(s), where t_2 appears for the first time. Thus, we use a link to indicate that a reduced traffic can *compensate* for an increase in traffic in the same bottleneck equation.

Now we generate the following sum S: for each link, we multiply the weight of the incident node (i.e. for an increased traffic) and the δ-value of the outgoing node (i.e. of a decreased traffic) and sum up all products for all links. Then, from the condition for the weight increase of the exponential OOWA we can see that

$$\sum_{i,t_i\in T^-}\delta_i w_i > S \tag{11}$$

since each node on the right side is at most connected to $(k-1)$ nodes to the left, where k is the index of its weight, and all links will have a different weight factor in S with smaller weight index. On the other hand

$$S \geq \sum_{i,t_i\in T^+}\delta_i w_i \tag{12}$$

since S contains, for each increasing node, the compensating terms in the same bottleneck equation (the δ-values of decreasing traffics, to which it is connected) with the same weight as the increase appears in the modified $oowa^*$ expression. Thus, by this modification, the original OOWA expression becomes smaller. By Lemma 1, after re-arranging the weights into the correct order, the value will not increase. Thus, the total OOWA expression becomes smaller.

For assisting these arguments, figure 4 gives an example. Assuming a setup with the following bottleneck equations as a result T^* of the BFC algorithm:

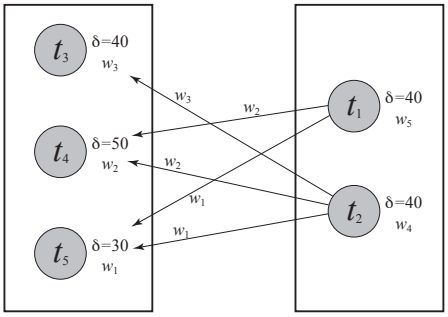

Fig. 4. Example for the bi-graph used in the proof.

$$
\begin{aligned}
t_1|_{=50} + t_2|_{=50} &= 100 \\
t_2|_{=50} + t_3|_{=60} &= 110 \\
t_1|_{=50} + t_2|_{=50} + t_4|_{=70} + t_5|_{=70} &= 240
\end{aligned}
$$

Then, we assume a change to another feasible state T as follows: $t_1 = 10, t_2 = 10, t_3 = 100, t_4 = 120, t_5 = 100$.

The set T^+ contains the elements t_3, t_4 and t_5, and the set T^- the two elements t_1 and t_2. The directed links are as shown in the figure, for example, node t_2 is connected to node t_5, since t_5 appears the first time in the 3rd bottleneck equation, and t_2 as element of T^- appears also in the same bottleneck equation.

Now there are five links and the value of S is computed as follows:

$$40 \cdot w_1 + 40 \cdot w_1 + 40 \cdot w_2 + 40 \cdot w_2 + 40 \cdot w_3 \tag{13}$$

and from $w_4 > w_1 + w_2 + w_3$ and $w_5 > w_1 + w_2 + w_3 + w_4 > w_1 + w_2$ it can be seen that the weighted sum for the right nodes exceeds S: $40 \cdot w_4 + 40 \cdot w_5 > S$.

For the left nodes, we can see that S contains all terms needed to compensate the weighted sum of all T^+ changes as well:

- For t_3, which increases the $oowa^*$ value by $40w_3$, S contains the (necessary) compensation of the increase in the second bottleneck equation from node t_2 ($40w_3$).
- For t_4, which increases the $oowa^*$ value by $50w_2$, S contains a compensation from decreasing t_1 by 40 and t_2 by 40, both weighted by w_2 in S.
- For t_5 with increase $30w_1$, the connecting links from t_1 and t_2 provide a compensation for the increase by a decrease of 40 and are weighted with w_1 in S.

So it can be seen that the sum of all changes for T^+ nodes (smaller or equal to S) increases the original OOWA by a smaller amount than the sum of all changes for T^- nodes (larger than S).