Les Cahiers du GERAD ISSN: 0711–2440

Normalizations of employee preferences in personnel scheduling

L.N. Hoang, G. Desaulniers,
M. Elahipanah, F. Soumis

G–2014–75

October 2014

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Normalizations of employee preferences in personnel scheduling

Lê Nguyên Hoang
Guy Desaulniers
Mahsa Elahipanah
François Soumis

GERAD & Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7

le.nguyen.hoang@gerad.ca
guy.desaulniers@gerad.ca
mahsa.elahipanah@gerad.ca
francois.soumis@gerad.ca

October 2014

Les Cahiers du GERAD
G–2014–75

Abstract: Including employee preferences in a shift-scheduling scheme raises the question of how to aggregate employee satisfactions in a sensible manner. To do so, we first need to model the employees' preferences, which can be done using so-called utility functions and a multi-attribute approach like MACBETH. Even then, though, we still need to make sure that it is sensible to compare or to add two different utility functions that concern different attributes. To do so, an appropriate normalization of utility functions is required. In this paper, we first discuss a naive extreme value normalization (EVN), which has utility values ranging all along two predetermined values, e.g. between 0 and 5. Then, we propose an alternative normalization, called AMACSN, which relies on a description of the "typical outcomes" produced by a shift-scheduling scheme. We then give both the conceptual grounds and the computational results to argue that AMACSN is more relevant and more meaningful than EVN to address the comparison or addition of different utility functions. As a bonus, we observe that AMACSN induces greater fairness.

Key Words: Normalization, preferences, utility, multi-attribute, scheduling.

1 Introduction

Since Dantzig (1954), shift scheduling has become a widely studied problem of operations research. It consists of creating and allocating shifts to employees to best cover a demand for employees at all times, and it can be stated as a set-covering problem. While set-covering problems have been shown to be NP-complete by Karp (1972), efficient exact or heuristic approaches based on column generation (Appelgren (1969); Desrochers and Soumis (1989); Barnhart et al. (1998); Desaulniers et al. (2005)) have yielded very good solutions. The efficiency of these approaches lies in the smallness of integrality gaps.

As computer power has increased, shift-scheduling models have become more realistic by including more constraints. For instance, at first, breaks in shifts were not taken into account (see Moondra (1976)); this was then rectified by Bechtold and Jacobs (1990) with the inclusion of lunch breaks. Later, Aykin (1996) proposed a model with several possible breaks. Rekik et al. (2010) generalized the model to include fractionable breaks. We pursue this effort to make shift-scheduling models more realistic, by including employee preferences.

One difficulty posed by such a goal is that shifts must be personalized. Personalized scheduling is not new though. Indeed, many models of *rostering* require knowledge of employees' planned activities like holidays, training periods or medical appointments. This, in turn, requires shift scheduling to treat employees asymmetrically, as was done for air crew scheduling (Gontier (1985)) or nurse scheduling (Bard and Purnomo (2005)). Using heuristic column generation and today's computer power, good solutions can still be found in a reasonable amount of time.

While this paper does provide a new approach to optimize the satisfaction of employees' preferences, the main contribution of this paper lies rather in the way we include these preferences. For one thing, it is notable that the mere step of mathematically modeling one's preferences is a difficult task. Indeed, it has been the core of active fields of research including conjoint analysis (Green and Srinivasan (1978); Orme (2005)) and multicriteria analysis (Siskos and Spyridakos (1999); Zopounidis and Doumpos (2002)). One approach from the latter category that we have found particularly relevant for our purposes is known as the MACBETH method, introduced by Bana e Costa and Vansnick (1994). In this setting, an employee describes a linear multi-attribute utility function by giving qualitative answers to comparison questions only. An example of such question is "*How much do you prefer starting a work shift at 8:00 a.m. compared to at 10:00 a.m.?*"

Thus, MACBETH models any employee's preferences as a weighted sum of so-called partial utility functions. Each partial utility function corresponds to an attribute (e.g. the time of day worked, the total number of hours worked in the week or the day off in the week). More precisely, for any employee $i \in N = \{1, \ldots, n\}$ and any attribute $k \in K$, the MACBETH method yields a partial utility function u_{ik}, which maps levels l_{ik} of this attribute (e.g. early morning, 40 hours or Sunday) to a real number. This real number $u_{ik}(l_{ik})$ takes on greater values for more preferred levels l_{ik}. Also, there is a weight w_{ik} associated with attribute k, so that the linear multi-attribute utility function of employee i can be written

$$u_i = \sum_{k \in K} w_{ik} u_{ik}(l_{ik}). \tag{1}$$

The MACBETH software allows employee i to determine the partial utility functions $u_{ik}(\cdot)$ and the weights w_{ik} that best match his preferences. We will describe this setting more formally in Section 2, as well as a shift-scheduling program we use to optimize employees' utilities.

It is important to notice, though, that the MACBETH method does not yield any normalization of the employees' linear multi-attribute utility functions. In other words, while it offers a sense of each utility function u_i individually and up to a positive affine transformation, it does not offer the possibility to compare the utility functions u_i and u_j of two different employees i and j. However, since we will be seeking to maximize the sum of all utilities, it is essential for these utility functions to be on the same scale.

A naive approach to doing this is what we call the extreme value normalization (EVN). In this setting, we make sure that the minimal value of a partial utility function is 0, and that its maximum value is 5. The choice of the value 5 is arbitrary, but results would be identical for any other choice. Importantly, partial utilities are normalized with regard to their extreme values. This normalization is used, for instance,

to compute the Human Development Index (HDI). Then, we require the sum of weights to add up to 100. While this naive normalization is widely used and already yields relevant shift allocations, we will show its flaws, both conceptually, through the so-called *busy Christmas paradox* and numerically, with our more advanced normalization called AMACSN.

AMACSN consists of two steps. First, we need to find a meaningful normalization of the partial utility functions u_{ik} to make all their values comparable with one another. To do so, we introduce a new normalization we call the correlated social normalization (CSN), which represents a major contribution of this paper. In essence, for each employee $i \in N$ and each attribute $k \in K$, this normalization consists of comparing the partial utility $u_{ik}(l_{ik})$ of the employee for the level of his schedule to his partial utilities $u_{ik}(l_{jk})$ for the other employees' levels l_{jk}, especially if employees i and j have the same preferences.

Next, note that the normalization of each partial utility function u_{ik} implies a rescaling of the corresponding weight w_{ik} to keep the multi-attribute utility function consistent. Indeed, if the partial utility u_{ik} is stretched by the normalization by a factor of 2, then the corresponding weight w_{ik} should be divided by 2. However, we still may need to normalize the whole vector of weights $(w_{ik})_{k\in K}$, so that two different employees' multi-attribute utilities are comparable. We formalize this aspect by introducing a multiplier α_i of the vector of weights for each employee i. Determining the multipliers α_i is the second step of our normalization procedure. In Section 4, we will present a normalization of these multipliers based on the so-called standard utility functions. At last, we obtain AMACSN.

This paper is divided into 6 sections. In Section 2, we present MACBETH, the description of preferences as linear multi-attribute utility functions and the shift-scheduling formulation as an integer program. Next, Section 3 introduces EVN, and then AMACSN, our main contribution. Section 4 then presents results of the shift-scheduling program, using EVN and AMACSN. These results confirm the greater relevancy of AMACSN. Finally, conclusions are drawn in Section 5.

2 Preferences and scheduling program

In this paper, we consider a shift-scheduling and job-assignment problem over one week. The main goal of this paper is to design an algorithm that also personalizes employees' shifts according to their preferences. To proceed to personalized shift scheduling with preferences, we first need to define a protocol that enables the employees to describe their preferences. Then, we need to include these preferences in the shift-scheduling optimization program. This requires us to provide a quantification of the employees' preferences.

In Subsection 2.1, we briefly discuss the modeling of the preferences according to linear multi-attribute utility functions that shall be used in the shift scheduling program. By using MACBETH, each employee determines his corresponding linear multi-attribute utility function, hence defining weights w_{ik} and partial utilities u_{ik}.

Then, in Subsections 2.2 and 2.3, we present the shift-scheduling optimization program, which is an integer linear program. We first present the program without preferences, which will enable us to derive the minimal shift-scheduling cost. Then, we present the program with preferences, that maximizes the sum of utilities, while guaranteeing a bound on costs defined by the minimal shift-scheduling cost. We also briefly discuss algorithms used to solve heuristically these scheduling programs.

2.1 Linear multi-attribute utility functions

Defining a procedure to help people quantitatively describe their preferences is a difficult problem that represents an active field of research. This is particularly true when faced with a large number of complex possible alternatives, as is the case for preferences about work schedules. A common simplification consists of characterizing *attributes* (also known as criteria), which preferences really depend on. In this paper, we consider four attributes, namely *Hours Per Week* (HPW), *Job Activity* (Job), *Shift-Type* (ShT) and *Day-On* (Day). We denote $K = \{HPW, Job, Day, ShT\}$ the set of attributes.

For a given schedule s, each attribute k takes a value called *level* $l_k(s)$. Depending on attributes, levels can have different forms. They can be vectors, scalars or subsets. To illustrate, the *Day-On* attribute is the subset of the days in the week that are working days. So, for instance, if a schedule s gives Wednesday and Sunday off, the *Day-On* attribute has level $l_{Day}(s) = \{Monday, Tuesday, Thursday, Friday, Saturday\}$. For the purpose of analyzing employees' preferences, a shift s can be regarded as a vector

$$(l_k(s))_{k\in K} = (l_{HPW}(s), l_{Job}(s), l_{Day}(s), l_{ShT}(s)). \tag{2}$$

For the sake of exposition and for confidentiality reasons, we do not explicit all levels.

To proceed, we assume that an employee's preferences about schedules can be fully described by a linear multi-attribute utility function. This means that we assume that the employee i's utility function $u_i : s_i \mapsto u_i(s_i) \in \mathbb{R}$ can be decomposed into a weighted sum of partial utility functions $u_{ik} : l_{ik} \mapsto u_{ik}(l_{ik}) \in \mathbb{R}$, for attributes $k \in K$. Denoting w_{ik} the weight of attribute k in employee i's utility function, then, for any schedule s_i given to employee i, we have

$$u_i(s_i) = \sum_{k\in K} w_{ik} u_{ik}(l_k(s_i)). \tag{3}$$

The description of partial utility functions $u_{ik}(\cdot)$ depends on the structure of the levels of attribute $k \in K$. Once again, for confidentiality reasons, we will not give more details regarding this modeling. Note, though, that the details about the partial utility functions that are skipped are not useful for the sequel of this paper.

Importantly, this decomposition is not unique. Indeed, following Von Neumann and Morgenstern (1944), if we only regard an employee's viewpoint, then only the ordering of shifts matters. Therefore, utility functions should actually be defined up to a positive affine transformation. This means that $u_i : \Omega \to \mathbb{R}$ and $\alpha u_i + \delta$ represent the same utility functions for any $\alpha > 0$, since such a transformation leaves the ordering of preferences unchanged.[1] In other words, for a given utility function, there are two degrees of freedom that need to be fixed by normalization.

Similarly, for each partial utility function $u_{ik}(\cdot)$, we have two degrees of freedom, as we can replace $u_{ik}(\cdot)$ by $\beta_{ik} u_{ik}(\cdot) + \gamma_{ik}$ for any $\beta_{ik} > 0$. Note that if we do multiply a partial utility function $u_{ik}(\cdot)$ by a multiplier $\beta_{ik} > 0$, then we need to divide w_{ik} by β_{ik} simultaneously to maintain the consistency of the multi-attribute utility function $u_i(\cdot)$. Otherwise, our modified multi-attribute utility function might describe a different ordering of shift allocations, and hence describe different preferences. To stick with simple notations though, we will not explicit these normalizable parameters of linear multi-attribute utility functions.

Before getting to the normalization considerations, we first need to determine a procedure that can help employees arrive at a decomposition of their preferences. There are two main areas of research that aim at such a procedure. The first is conjoint analysis (see Green and Srinivasan (1978); Louviere (1988); Green et al. (2001); Netzer et al. (2008)), which consists of analyzing trade-off situations one may be faced with. For instance, one may be asked to rank a small set of alternatives. From these observations, a regression model characterizes the trends to induce a global ordering of all the alternatives.

However, more straightforward approaches have come from multicriteria analysis. Some of the most popular methods from this field are ELECTRE (Benayoun et al. (1966); Maystre et al. (1994); Greco et al. (2011)), PROMETHEE (Brans et al. (1986); Brans and Mareschal (2002)) and MACBETH. This last one is used in this paper. There are two main steps involved in using MACBETH to describe linear multi-attribute utility functions. First, employee i's partial utility functions $u_{ik}(\cdot)$ is defined, and then his weights w_{ik} are determined.

Roughly, an employee's partial utility function is constructed by having the employee comparing *reference levels*. This is done by filling in a half matrix in MACBETH. Then, depending on the considered attribute, MACBETH infers a whole partial utility function $u_{ik}(\cdot)$.

[1] More precisely, for any two shifts s_1 and s_2, $u_i(s_1) \geq u_i(s_2)$ if and only if $\alpha u_i(s_1) + \delta \geq \alpha u_i(s_2) + \delta$. Plus, the same property must hold for probability distributions over shifts.

Remark 1 *The partial utility functions of MACBETH are only normalized such that the maximum utility of an attribute always equals 100. However, there is no normalization of the minimum utility. It can equal 0 or be as small as -600. This is not good for our optimization, as partial utility functions will not all be at the same scale.*

Now, for each attribute $k \in K$, MACBETH determines a default level $l_k^{default}$ and a most preferred level $l_k^{preferred}$. Then, for any two attributes $k_1, k_2 \in K$, employees are asked to compare levels $(l_{k_1}^{default}, l_{k_2}^{preferred})$ and $(l_{k_1}^{preferred}, l_{k_2}^{default})$. By achieving all pairwise comparisons between any two attributes, employees fill in a half matrix in MACBETH, which is then automatically used to compute all weights $w_{ik} \geq 0$ (with at least one non-zero).

2.2 Shift scheduling without preferences

The shift-scheduling problem we face has a one-week horizon. This week is divided into a set $T = \{1, \ldots, |T|\}$ of periods. At each period $t \in T$ and for each job activity $a \in A$, we suppose that there is a known demand d_{at}. Each undercovering (respectively, overcovering) of demand has a cost $\underline{c}_a$ (respectively, $\bar{c}_a$) per period $t \in T$ of time. We denote by U_{at} and O_{at} the number of undercoverings and overcoverings of demand for job activity a at period t. For each employee $i \in N$, we denote by Ω_i the set of his admissible one-week schedules. We define δ_{ats} the binary parameter that equals 1 if and only if schedule $s \in \Omega_i$ requires employee i to work job activity $a \in A$ at period t. Finally, we denote by x_{is} the binary variable that equals 1 when employee $i \in N$ works schedule $s \in \Omega_i$.

The following Shift Scheduling (SS) integer linear program determines the minimal shift-scheduling cost:

$$C_0 = \underset{x,U,O,C}{\text{Minimize}} \quad C \tag{4}$$

$$\text{subject to:} \quad \sum_{s \in \Omega_i} x_{is} = 1, \qquad \forall i \in N, \tag{5}$$

$$U_{at} + \sum_{\substack{i \in N \\ s \in \Omega_i}} \delta_{ats} x_{is} = d_{at} + O_{at}, \qquad \forall a \in A, \forall t \in T, \tag{6}$$

$$C = \underline{c}_a U_{at} + \sum_{\substack{a \in A \\ t \in T}} \bar{c}_a O_{at} \tag{7}$$

$$U_{at} \geq 0, O_{at} \geq 0, \qquad \forall a \in A, \forall t \in T. \tag{8}$$

$$x_{is} \in \{0, 1\}, \qquad \forall i \in N, \forall s \in \Omega_i. \tag{9}$$

Equations (5) assert that each employee must be given one and only one shift. Equations (6) compute the number of undercoverings and overcoverings in each period and for each job. Equations (7) derive costs from undercoverings and overcoverings. Relations (8) ensure that the numbers of undercoverings and overcoverings are computed as non-negative values. Finally, relations (9) are the integrality requirement on the schedule variables.

We shall use this minimal cost C_0 to design our shift-scheduling program with preferences.

2.3 Shift scheduling with preferences

In order to balance shift-scheduling cost with fairness in a reasonable way, we require the personalized shift scheduling to cost no more than $(1 + \alpha)C_0$ for $\alpha > 0$. In our case, we will choose $\alpha = 2\%$.

Moreover, we assume that each employee has determined and revealed his linear multi-attribute utility function for shifts. This means that we are given as inputs the partial utility functions $u_{ik} : l_k \mapsto u_{ik}(l_k) \in \mathbb{R}$ for any employee $i \in N$ and attribute $k \in K$, as well as weights $w_{ik} \in \mathbb{R}_+$. As discussed earlier, these should be normalized before running the shift-scheduling program. Different normalizations will be discussed in Section 3.

Now, in an attempt to maximize the employees' satisfactions, we define the Shift Scheduling with Preferences (SSP) optimization program:

$$\underset{x,U,O,C}{\text{Maximize}} \quad \sum_{i\in N}\sum_{s\in\Omega_i}\sum_{k\in K} w_{ik}u_{ik}(l_k(s))x_{is} \tag{10}$$

$$\text{subject to:} \quad C \leq (1+\alpha)C_0, \tag{11}$$

$$\text{Constraints (5)–(9).}$$

It is noteworthy that SSP does not include any fairness objective. We have found that solving SSP is already quite time-consuming. Yet, a fairness term in the objective function, which, for instance, would minimize some sort of standard deviation, can be expected to yield a much greater integrality gap. For this reason, we expect the addition of such a term to greatly increase the computation time. For this reason, we limit ourselves to simply maximizing the sum of the employees' satisfactions. However, as we shall see, a right normalization of utility functions will naturally guarantee a satisfying amount of fairness.

To solve the SSP program, we propose a heuristic based on column generation (see Appelgren (1969); Desrochers and Soumis (1989); Barnhart et al. (1998); Desaulniers et al. (2005)). We first solve the linear relaxation of the SS program for a set Ω_i that initially contains only a few of the admissible shifts for employees $i \in N$. Then, given dual variables of the SS program, and using a subproblem, we generate other relevant columns $s \in \Omega_i$. This subproblem is solved using a professional software, which we will not dwell on for confidentiality reasons. This is also the reason we do not provide more information about sets Ω_i of schedules. But it is noteworthy that this professional software is used in over 10,000 companies, and each company uses it for different independent groups of employees, e.g. for different stores, factories and departments.

Importantly, columns generated by the subproblem enlarge the set of generated shifts. Once this set is large enough, we use the solver Xpress-MP to solve the SS program with the integrality constraints, hence deriving the minimal shift scheduling cost C_0. Finally, we solve the SSP program with the same set of columns generated in SS, still using the solver Xpress-MP.

One might fear that this set of columns is too restricted. However, as we said, the SSP program is already very time-consuming as is. Moreover, we could argue that our sets Ω_i generated by SS are already large enough to find the shifts the employees ask for. More specifically, in our instances, the SS program generates about $100,000$ columns. Yet, for each attribute, a rough estimate shows that at least one out of 10 levels is precisely what an employee has asked for. Hence, since there are 4 attributes, there are at least one in 10^4 shifts that completely satisfy an employee. Therefore, it is very likely that, for any given employee, many of the shifts we generated match that employee's preferences.

Note that, because the solution value found for the SS program is only used as a parameter in Constraint (11), it is not that important to get an accurate value for C_0. On the other hand, the SS program is so much easier to solve than the SSP program that it is not where most of the computation time is lost. Thus, requiring greater accuracy in the computation of C_0 will not deeply affect the computation time overall. Once again, the SS program is mainly essential to generate relevant columns.

3 Multi-attribute normalizations

In this section, we introduce two normalizations of linear multi-attribute utility functions. We first quickly present the naive EVN. Then, we will define AMACSN, which is the main contribution of this paper.

3.1 Extreme value normalization

As announced in the introduction, we define a naive normalization for linear multi-attribute utility functions. The EVN shares similarities with many common measure indices, like, for instance, HDI.

Definition 1 *A linear multi-attribute utility function is EVN if the extreme partial utilities equal 0 and 5 and such that the sum of weights equal 100. We denote by u_{ik}^{EVN} and w_{ik}^{EVN} the EVN partial utilities and weights.*

Let us verify that this normalization is well-defined. To do so, we must assume that partial utilities are non-degenerate. This means that a partial utility u_{ik} has two different values for some two levels l_{ik}^1 and l_{ik}^2.

Proposition 1 *If partial utilities are bounded and non-degenerate, then the EVN exists and is unique.*

Proof. Let u_{ik}^{min} and u_{ik}^{max} be defined by

$$u_{ik}^{min} = \min_{l_{ik}} u_{ik}(l_{ik}) \quad \text{and} \quad u_{ik}^{max} = \max_{l_{ik}} u_{ik}(l_{ik}). \tag{12}$$

By assumption, we have $-\infty < u_{ik}^{min} < u_{ik}^{max} < \infty$. Given that only positive affine transformations are allowed, the EVN partial utility is then necessarily

$$u_{ik}^{EVN} = 5 \times \frac{u_{ik} - u_{ik}^{min}}{u_{ik}^{max} - u_{ik}^{min}}. \tag{13}$$

For the ordering of shifts induced by u to remain the same, we need to rescale weights, by

$$\hat{w}_{ik} = w_{ik}(u_{ik}^{max} - u_{ik}^{min}). \tag{14}$$

It is then straightforward to see that $\sum w_{ik}u_{ik}$ and $\sum \hat{w}_{ik}u_{ik}^{EVN}$ represent the same utility functions. Finally, we have to set $w_{ik}^{EVN} = 100 \times \hat{w}_{ik} / \sum_{k' \in K} \hat{w}_{ik'}$, in order to guarantee that the sum of the weights amounts to 100. This proves existence and uniqueness. □

While we do not state this fact formally, it is also straightforward to see that no matter which normalization of the linear multi-attribute utility function we start with, applying EVN always yields the same EVN linear multi-attribute utility function.

Also, we will call EVN-SSP the SSP program whose objective function is written with EVN linear multi-attribute utility functions.

3.2 Affinely multi-attribute correlated social normalization

In this subsection, we propose a more meaningful alternative to EVN, which extends ideas by Hoang et al. (2014) to the setting of linear multi-attribute utility functions. But before getting to this alternative, let us criticize EVN conceptually through an example.

3.2.1 The busy Christmas paradox

Consider only one attribute, namely, *Day-On*. Assume that Tuesday is Christmas, and that nearly everyone wants it off. Now, consider employees 1 and 2 who both really care about the day of the week they get off. However, while employee 1 really wants to have Christmas off, employee 2 does not celebrate Christmas, and actually wants Friday off.

Let us compare the two employees' partial utilities for the day-on attribute. Employee 2 basically knows he will be given Friday off, as he's almost surely the only one who asked for it. Thus, he will certainly have the maximal utility of 5 out of 5. Similarly, if employee 1 does get Christmas off, he would have the maximal utility of 5 out of 5 too. However, because it is much less likely for him to obtain what he wants, employee 1 will definitely feel much happier about having Christmas off than employee 2 does about having Friday off.

There is another way of seeing this. By giving employee 2 Friday off, the cost for the other employees and the company is basically zero, if not negative. However, giving Christmas off to employee 1 means sacrificing an opportunity that many other employees would gladly take. In other words, it induces a non-negligible

cost to other employees. For this reason, employee 1 has to feel happier about his day off than employee 2 does about his.

Conversely, if employee 2 were not given Friday off, he would be so surprised that he would actually feel greatly disappointed. Meanwhile, if employee 1 were not given Christmas off, then he would know he probably is one of many other employees who did not get Christmas off. Therefore, employee 1 would not feel as badly about not having Christmas off as employee 2 would about not having Friday off.

What this discussion shows is that the EVN we have been using to normalize partial utilities is not appropriate to compare the different partial utilities of different employees. Rather, a normalization of utility functions should be defined depending on some context that the shift allocation defines. This leads us to one of the main contributions of this paper, which extends the SN introduced by Hoang et al. (2014).

3.2.2 Correlated social normalization

Like SN, correlated social normalization (CSN) consists of normalizing an employee's utility with regard to utilities he would have, were he given other employees' shifts. To do so, it is necessary to consider some typical output of SSP. This output will form a benchmark that employees can compare their schedules to. Thus, throughout this subsection, we need to consider a given solution $\hat{s}$ that assigns a schedule $\hat{s}_i$ to each employee i. Also, for simplicity, we denote $\hat{l}_{ik} = l_k(\hat{s}_i)$.

We shall detail SN in Section 4. Roughly, SN has every employee comparing his schedule to others'. In addition to this, CSN takes into account the fact that employees tend to especially compare themselves with other employees who have similar utility functions. For instance, if you have asked for Christmas off, then it is much more relevant to compare your shift with the shifts of other employees who also asked for Christmas off than with the shifts of those who asked for Friday off.

To formally define CSN, let the utility matrix $\hat{U}^k$ be defined by $\hat{U}^k_{ij} = u_{ik}(\hat{l}_{jk})$. This matrix contains all the information about how each employee feels about the levels of attribute k in his and the others' allocated schedules. Now, the similarity between two partial utility functions u_{ik} and u_{jk} can be characterized by the correlation r_{ijk} between employees i and j's partial utilities $u_{ik}(\hat{l}_{mk})$ and $u_{jk}(\hat{l}_{mk})$ for different levels $\hat{l}_{mk}$ of all third employees $m \in N$. Formally, this correlation is given by

$$r_{ijk} = \frac{Cov(\hat{U}^k_i, \hat{U}^k_j)}{\sqrt{Var(\hat{U}^k_i)Var(\hat{U}^k_j)}} = \frac{\sum_{m\in N}(u_{ik}(\hat{l}_{mk}) - \mu_{ik})(u_{jk}(\hat{l}_{mk}) - \mu_{jk})}{n\sigma_{ik}\sigma_{jk}}, \tag{15}$$

where μ_{ik} and σ_{ik} (respectively, μ_{jk} and σ_{jk}) are the averages and standard deviations of $u_{ik}(\hat{l}_{mk})$ (respectively, $u_{jk}(\hat{l}_{mk})$) for $m \in N$, i.e.,

$$\mu_{ik} = \frac{1}{n}\sum_{m\in N} u_{ik}(\hat{l}_{mk}) \qquad \text{and} \qquad \sigma^2_{ik} = \frac{1}{n}\sum_{m\in N}\big(u_{ik}(\hat{l}_{mk}) - \mu_{ik}\big)^2. \tag{16}$$

Now, the greater r_{ijk} is, the more relevant that is. For this reason, and to have non-negative numbers only, we define the transformed correlation $R_{ijk} = 1 + r_{ijk} \in [0, 2]$. The relevant benchmark to normalize an employee's partial utility is then given by the values $u_{ik}(\hat{l}_{jk})$ for $j \in N$ with weights R_{ijk}. These values define the CSN averages μ^{CSN}_{ik} and standard deviations σ^{CSN}_{ik} by

$$\mu^{CSN}_{ik} = \frac{\sum_{j\in N} R_{ijk}u_{ik}(\hat{l}_{jk})}{\sum_{j\in N} R_{ijk}} \qquad \text{and} \qquad \big(\sigma^{CSN}_{ik}\big)^2 = \frac{\sum_{j\in N} R_{ijk}\big(u_{ik}(\hat{l}_{jk}) - \mu_{ik}\big)^2}{\sum_{j\in N} R_{ijk}}. \tag{17}$$

Finally, we obtain the CSN partial utility

$$u^{CSN}_{ik}(\cdot) = \frac{u_{ik}(\cdot) - \mu^{CSN}_{ik}}{\sigma^{CSN}_{ik}}. \tag{18}$$

If we do that, though, because we need to keep track of the fact that the global multi-attribute utility function should yield the same orderings of allocations after normalization, we need to compensate for the rescaling of partial utilities u_{ik}. Since they have been divided by σ_{ik}^{CSN}, it suffices to multiply the weights w_{ik} by σ_{ik}^{CSN}, yielding $w_{ik}^{CSN} = \sigma_{ik}^{CSN} w_{ik}$. These transformations lead us to rewrite the linear multi-attribute utility function of employee i as

$$u_i = \sum_{k \in K} w_{ik}^{CSN} u_{ik}^{CSN}. \tag{19}$$

In the introduction, we pointed out that this utility function is defined up to a positive affine transformation. However, with no additive constant, our normalization of partial utilities guarantees that, for an employee whose levels are all averages, the linear multi-attribute utility function is 0. Therefore, it makes sense to set the additive constant to 0. Conversely though, let us highlight the degree of freedom corresponding to the positive multiplier α_i, by writing the linear multi-attribute utility function as:

$$u_i = \alpha_i \sum_{k \in K} w_{ik}^{CSN} u_{ik}^{CSN}. \tag{20}$$

To determine appropriate multipliers α_i, we turn to the concept of standard utility functions.

3.2.3 Standard utility functions

To determine a normalization of multipliers α_i, we propose to describe how employees' weights affect their partial utilities. Intuitively, the greater an employee's weight for an attribute, the greater his partial utility for that attribute should be. We formalize this intuition with the concept of the standard utility function. A standard utility function ψ_k for attribute $k \in K$ is an increasing function that maps an employee i's weight $\alpha_i w_{ik}^{CSN}$ to the partial utility $\psi_k(\alpha_i w_{ik}^{CSN})$ he should expect to have for attribute k. In other words, we should have $\psi_k(\alpha_i w_{ik}^{CSN}) \approx u_{ik}^{CSN}(\hat{l}_{ik})$ for any employee $i \in N$ and attribute $k \in K$.

Note that, importantly, the standard utility functions ψ_k do not depend on employees. Rather, standard utility functions are rough descriptions of the properties of the shift-allocation schemes, which employees will be sensible to. As we shall see later, such rough descriptions will give us a natural way to determine the multipliers α_i, hence making multi-attribute utility functions comparable. But first, we discuss how to compute good standard utility functions.

For tractability reasons, we propose to only consider positive affine standard utility functions. More explicitly, we only consider standard utility functions for an attribute k that are functions $\psi_k(\alpha_i w_{ik}) = \gamma_k + \beta_k \alpha_i w_{ik}$ with $\beta_k \geq 0$. Thus, a standard utility function ψ_k for attribute $k \in K$ is fully determined by a pair $(\beta_k, \gamma_k) \in \mathbb{R}_+^* \times \mathbb{R}$. Aggregating all standard utility functions for all attributes then yields an element ψ, which is represented by vector $(\beta_k, \gamma_k)_{k \in K} \in (\mathbb{R}_+^* \times \mathbb{R})^K$.

Now, standard utility functions need to describe how weights affect partial utilities. Then, the affine standard utility function ψ_k that best describes these inputs is then the linear regression of the form

$$\forall i \in N, \quad u_{ik}^{CSN}(\hat{l}_{ik}) = \psi_k(\alpha_i w_{ik}^{CSN}) + \epsilon_i = \gamma_k + \beta_k \alpha_i w_{ik}^{CSN} + \epsilon_i, \tag{21}$$

such that the sum of squares $\sum \epsilon_i^2$ is minimized. This boils down to equalities

$$\beta_k = \frac{Cov(\alpha_i w_{ik}^{CSN}, u_{ik}^{CSN}(\hat{l}_{ik}))_{i \in N}}{Var(\alpha_i w_{ik}^{CSN})_{i \in N}} \qquad \text{and} \qquad \gamma_k = \frac{1}{n} \sum_{i \in N} \Big(u_{ik}^{CSN}(\hat{l}_{ik}) - \beta_k \alpha_i w_{ik}^{CSN} \Big). \tag{22}$$

These equations define the best-fit standard utility function ψ_k to multipliers α. In other words, our linear regression yields a function $\mathsf{Regress}_k$, where $\mathsf{Regress}_k(\alpha)$ is the best-fit standard utility function ψ_k to multipliers α. By combining $\mathsf{Regress}_k$ functions for all attributes k, we obtain the regression function $\mathsf{Regress}$, which maps multipliers to the best-fit affine standard utility function ψ.

3.2.4 Multiplier normalization

Now, recall that standard utility functions $\psi = (\psi_k)_{k\in K}$ for all attributes yield a good description of the relation between weights and partial utilities, that is, $u_{ik}^{CSN}(\hat{l}_{ik}) \approx \psi_k(\alpha_i w_{ik}^{CSN})$. Moreover, recall that since we have defined $\hat{l}_{ik} = l_k(\hat{s}_i)$, we have $u_i(\hat{s}_i) = \sum_{k\in K} \alpha_i w_{ik}^{CSN} u_{ik}^{CSN}(\hat{l}_{ik})$. Then, employee i's linear multi-attribute utility function for his shift s_i can be approximated by

$$u_i(s_i) \approx \sum_{k\in K} \alpha_i w_{ik}^{CSN} \psi_k(\alpha_i w_{ik}^{CSN}). \tag{23}$$

The last quantity represents a rough estimate of what employee i's multi-attribute utility is, given his weights and the rough description of the shift allocation mechanism. Yet, for multi-attribute utilities to be comparable between any two employees, this estimate should be the same for all employees, say $U = 100$. This gives us a natural way to normalize multipliers α_i, by requiring that they satisfy the following equation:

$$\sum_{k\in K} \alpha_i w_{ik}^{CSN} \psi_k(\alpha_i w_{ik}^{CSN}) = U. \tag{24}$$

Now, it is important to note that the normalization of multipliers α we propose here require knowledge of the standard utility functions ψ. This means that we have a function $\mathsf{Multipliers} : \psi \mapsto \alpha = \mathsf{Multipliers}(\psi)$. Let us point out that assuming that standard utilities are affine enables an algebraic computation of $\mathsf{Multipliers}$.

Proposition 2 *Let a standard utility function ψ, with $\beta > 0$. Then, we have the following equality:*

$$\forall i \in N, \quad \mathsf{Multipliers}_i(\psi) = \frac{-\sum\limits_{k\in K} w_{ik}^{CSN}\gamma_k + \sqrt{\left(\sum\limits_{k\in K} w_{ik}^{CSN}\gamma_k\right)^2 + 4U \sum\limits_{k\in K} (w_{ik}^{CSN})^2\beta_k}}{2\sum\limits_{k\in K} (w_{ik}^{CSN})^2\beta_k} \tag{25}$$

Proof. We have the following computation:

$$\sum_{k\in K} \alpha_i w_{ik}^{CSN} \psi_k(\alpha w_{ik}^{CSN}) = \sum_{k\in K} \alpha_i w_{ik}^{CSN}(\gamma_k + \alpha_i w_{ik}^{CSN}\beta_k) = \alpha_i^2 \sum_{k\in K} (w_{ik}^{CSN})^2\beta_k + \alpha_i \sum_{k\in K} w_{ik}^{CSN}\gamma_k. \tag{26}$$

Now, if $\alpha_i = \mathsf{Multipliers}_i(\psi)$, then the left term above must equal U, hence,

$$\alpha_i^2 \sum_{k\in K} (w_{ik}^{CSN})^2\beta_k + \alpha_i \sum_{k\in K} w_{ik}^{CSN}\gamma_k = U, \tag{27}$$

which is a second-degree equation in α_i, with a single positive solution. This solution is given by the formula of the proposition. □

3.2.5 Consistency

For our description of the shift-allocation mechanism by standard utility functions to be consistent, we need to find multipliers α^* and standard utility functions ψ^* that correspond to one another. This leads us to the following definition.

Definition 2 *A multiplier α^* and a standard utility function ψ are consistent if they satisfy*

$$\mathsf{Regress}(\alpha^*) = \psi^* \qquad \textit{and} \qquad \mathsf{Multipliers}(\psi^*) = \alpha^*. \tag{28}$$

We propose to solve these equations by iterations. Namely, at stage $t \in \mathbb{N}$, we assume we are given α^t and ψ^t. We then compute $\psi^{t+1} = \mathsf{Regress}(\alpha^t)$ and $\alpha^{t+1} = \mathsf{Multipliers}(\psi^t)$. Equivalently, this boils down to searching for a fixed point $\psi^* = \mathsf{Regress}(\mathsf{Multipliers}(\psi^*))$ by computing the sequence $\psi^{t+1} = \mathsf{Regress}(\mathsf{Multipliers}(\psi^t))$.

To test the convergence of this sequence, we have computed the slopes β_k^t of ψ_k^t at each iteration t, and for all attributes $k \in K$. Results are displayed in Figure 1. We clearly see a fast convergence after merely 10 iterations. This pattern appears in other computations we have made.

We can finally combine everything we have discussed to determine the normalization we have been searching for.

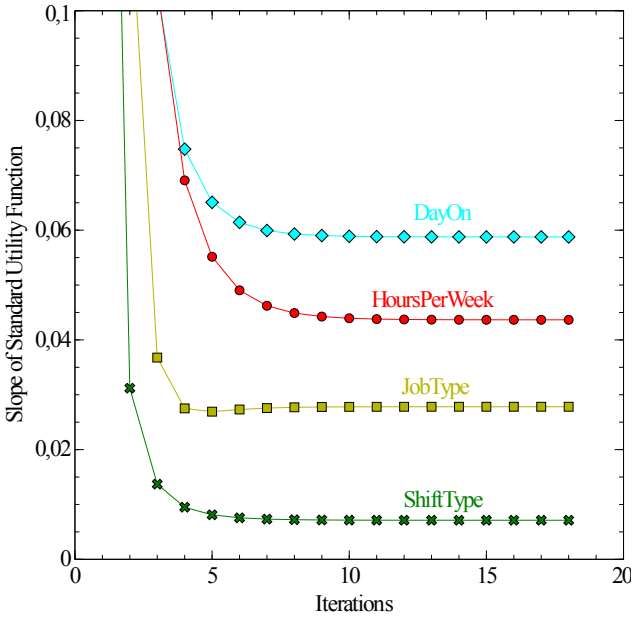

Figure 1: Convergence of slopes as we iterate operators Regress and Multipliers.

Definition 3 *Multi-attribute utility functions u_i^* for $i \in N$ are affinely multi-attribute correlated and socially normalized (AMACSN) if there exist consistent multipliers α^* such that*

$$\forall i \in N, \quad u_i^*(s_i) = \sum_{k \in K} \alpha_i^* w_{ik}^{CSN} u_{ik}^{CSN}(l_k(s_i)). \tag{29}$$

Similarly to EVN, we will call AMACSN-SSP the SSP program whose linear multi-attribute utility functions are normalized by AMACSN.

3.2.6 Typical outcome

Recall that AMACSN depends on the "typical outcome" $\hat{s}$ we consider. To compute AMACSN, we thus need a solution of SSP. Ideally, this solution should be a solution of AMACSN-SSP. Evidently, we cannot do so, since AMACSN-SSP requires AMACSN to be computed in the first place. A good approach to determine AMACSN would consist of computing some first AMACSN_1 based on a solution to, say, EVN-SSP. Then, we would use the solution of AMACSN_1-SSP to compute the more appropriate AMACSN_2, and so on. We would have a sequel of normalizations AMACSN_t, for all $t \geq 1$. Hopefully, the sequence would yield some limit AMACSN_∞. We could then expect to have AMACSN_∞ being consistent with solutions of AMACSN_∞-SSP.

However, for simplicity and because of computational times, we will merely consider the results of two instances of the EVN-SSP program to define "typical outcomes" $\hat{s}$ to compute AMACSN. The solutions of EVN-SSP that we use to define this typical outcome will be analyzed in details in the next section. Interestingly, the fact that we have actually not used the most appropriate "typical outcome" will underline the robustness of AMACSN.

More explicitly, we will use the results of the solutions of EVN-SSP to compute the transformed correlations R_{ijk}, the averages μ_{ik}^{CSN}, standard deviations σ_{ik}^{CSN}, as well as consistent multipliers α^* and consistent standard utility functions ψ^*. These values determine the CSN of partial utility functions and the weight normalization for linear multi-attribute utility functions. The consistent standard utility functions are depicted in Figure 2, which also depicts the relation between AMACSN weights and CSN partial utilities for the different attributes.

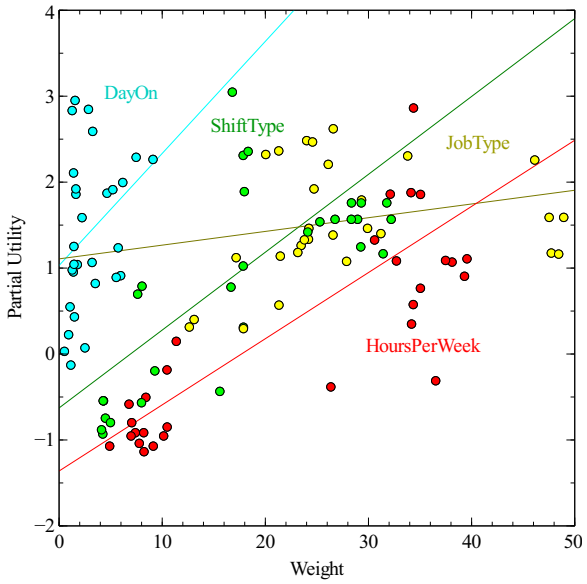

Figure 2: Consistent standard utility functions computed with the cumulative allocations given by EVN-SSP in two instances, which will be analyzed in the next section. These standard utility functions are the ones we use to compute the AMACSN utility functions.

4 Results

The SS program is solved with a tolerance on the optimality gap of 3%. We allowed an optimality gap of 5% for the computation of the SSP program. It is not necessary to be much more accurate, as the uncertainty on employees' preferences can be regarded as being at least 5%, because of assumptions like the linearity of multi-attribute utility functions, or because inaccuracy emerged in the reporting of preferences. In addition, we need to keep in mind that employees may have had incentives to lie about their preferences, and thus there are additional uncertainties due to untruthful preference revelations. The column-generation heuristic was launched on instances with 29 employees and 6 job activities, over a 1-week horizon with periods of 15 minutes (hence $|T| = 4 \times 24 \times 7 = 672$). The instances we consider are real-life instances, except for the preferences that were generated by hand.

We want to compare solutions of EVN-SSP to solutions of AMACSN-SSP. However, it is not clear that it is on the EVN scale that the quality of results should be judged. Naturally, it would not be convincing either if we judged results on the AMACSN scale. To be more fair in the comparison, we will use the SN scale defined by Hoang et al. (2014).

4.1 Social normalization

A more relevant way to judge the quality of the optimization is to study how employees compare one another's shift. Intuitively, if the optimizer has done a good job, each employee should get a better shift than any of the others' shifts. A natural way to model this idea is by involving the SN introduced by Hoang et al. (2014). The idea lies in considering other employees' shifts as a benchmark for an employee to judge the quality of his shift. Let us consider a solution $\hat{s}$ obtained by some SSP. For employee i, we define the average utility μ_i for others' shifts and the standard deviation σ_i by

$$\mu_i = \frac{1}{n-1} \sum_{i' \neq i} u_i(\hat{s}_{i'})), \tag{30}$$

$$\sigma_i^2 = \frac{1}{n-1} \sum_{i' \neq i} (u_i(\hat{s}_{i'}) - \mu_i)^2. \tag{31}$$

We stress the fact that this SN of an employee i's utility function is made with respect to i's utility function and to other employees' shifts. In particular, it does not depend on other employees' utility functions.

The SN utility function of employee i is the positive affine transformation with an average utility for others' shifts of 0 and a standard deviation of 1. More formally, given a non-normalized utility function u_i, with corresponding average μ_i and standard deviation σ_i, the SN utility function u_i^{SN} is given by

$$u_i^{SN}(\cdot) = \frac{u_i(\cdot) - \mu_i}{\sigma_i}. \tag{32}$$

In particular, an employee's SN utility for his shift equals $u_i^{SN}(\hat{s}_i) = (u_i(\hat{s}_i) - \mu_i)/\sigma_i$. This SN utility counts how many standard deviations above average an employee's utility is. In particular, we expect it to be positive for all employees, which means that every employee's utility is above average.

Similarly, we may also use the CSN scale to judge the qualities of two solutions by EVN-SSP and AMACSN-SSP, where CSN is applied to the linear multi-attribute utility function — not to the partial utility functions as is done in AMACSN.

4.2 EVN-SSP

To test the performance of the optimization algorithms alone, let us quickly analyze results of EVN-SSP judged by EVN scales. Figure 3 plots the EVN utilities for the schedules of the 29 employees computed by EVN-SSP for two instances. As often in this paper, we merely present results for these two instances, although we have verified that their features are also revealed in the 10 instances we ran. Interestingly, all utilities range between 250 and 500. This latter bound equals the theoretical maximum utility of an employee.

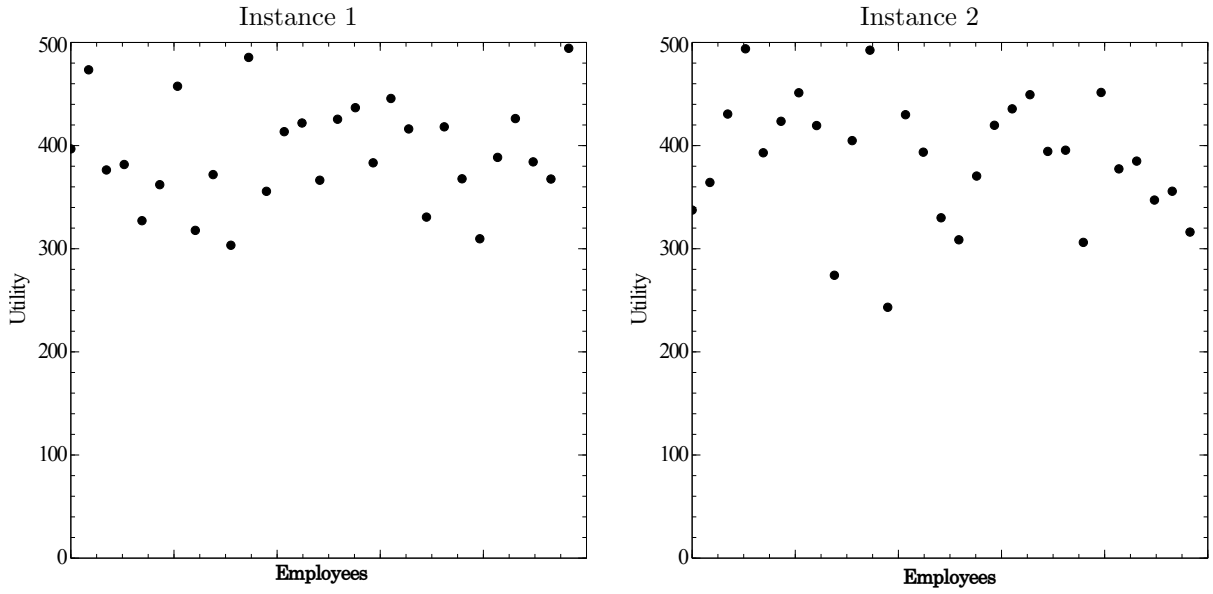

Figure 3: EVN utility values from the solutions of EVN-SSP.

Another way to unveil the quality of the optimization is to look at how the weights w_{ik} of employees i for attributes k affect the corresponding partial utilities u_{ik}. Intuitively, the greater the weight w_{ik}, the more the optimizer should gain by yielding large values of u_{ik}. Thus, u_{ik} should look like an increasing function of w_{ik}. This is what is displayed in Figure 4.

Let us now analyze EVN-SSP through the lens of SN and CSN. Figure 5 displays the SN utilities of all employees for the two instances. Overall, these results show that our algorithm has succeeded in optimizing its objective value in a fairly convincing way. No employee is under 0, which means that no one will feel disadvantage with respect to the average schedule of others.

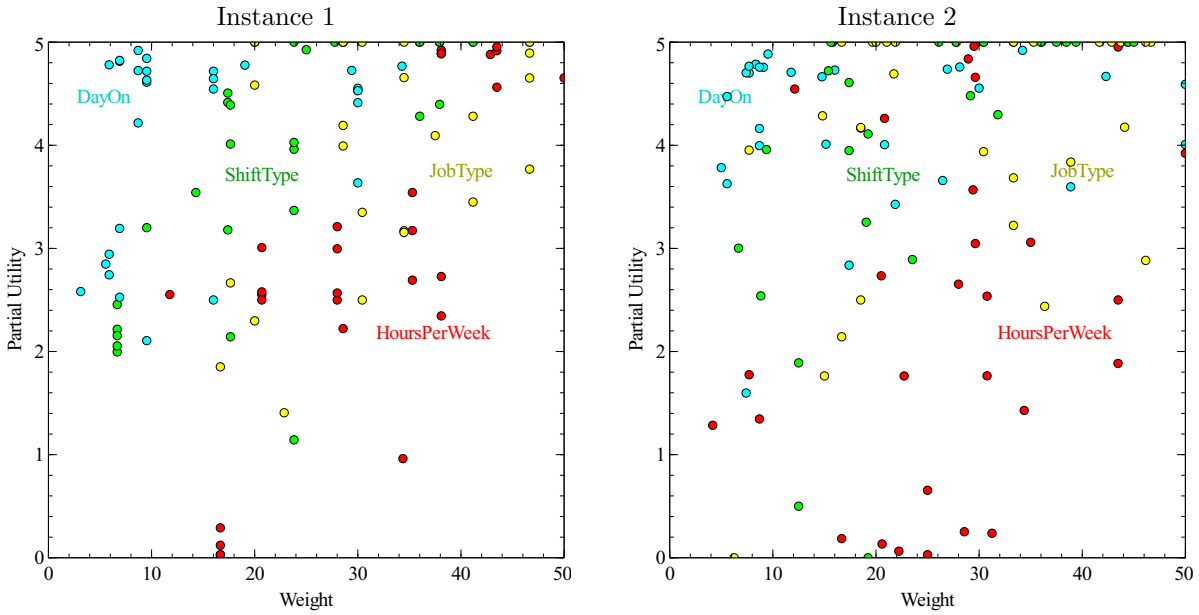

Figure 4: EVN partial utility values as functions of weights.

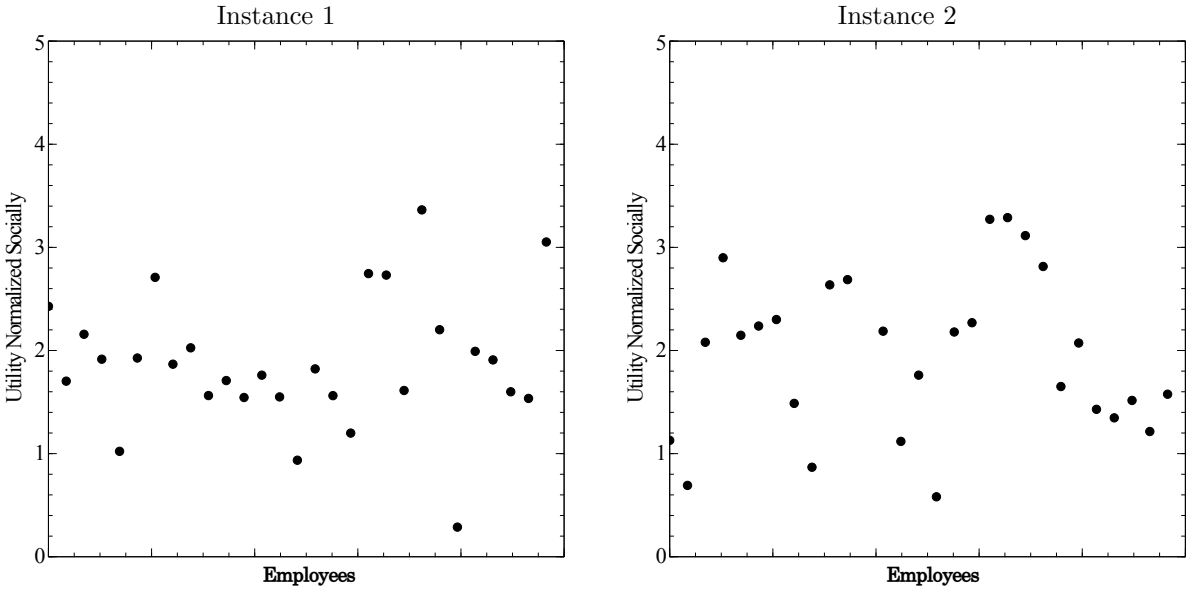

Figure 5: Employees' SN utilities for shifts given by EVN-SSP.

To have an idea of whether the EVN we have been using makes sense, we may compare it directly with SN and CSN. This is what is depicted in Figure 6, where the x-axis stands for the employees' utilities obtained by EVN, while the y-axis represents their SN utilities. What we see is that, although there is a positive correlation between EVN and, respectively, SN and CSN, the correlation is not entirely convincing. In fact, the correlation of EVN and SN is only 0.67, while that of EVN and CSN is 0.68. This indicates that our normalization by extreme values has some weaknesses.

Before analyzing solutions of AMACSN-SSP, let us end this section by studying the solutions of EVN-SSP through the lens of AMACSN. Computations of the consistent multipliers and standard utility functions yield Figure 7, where CSN partial utilities u_{ik}^{CSN} are depicted as functions of normalized weights $\alpha_i w_{ik}^{CSN}$.

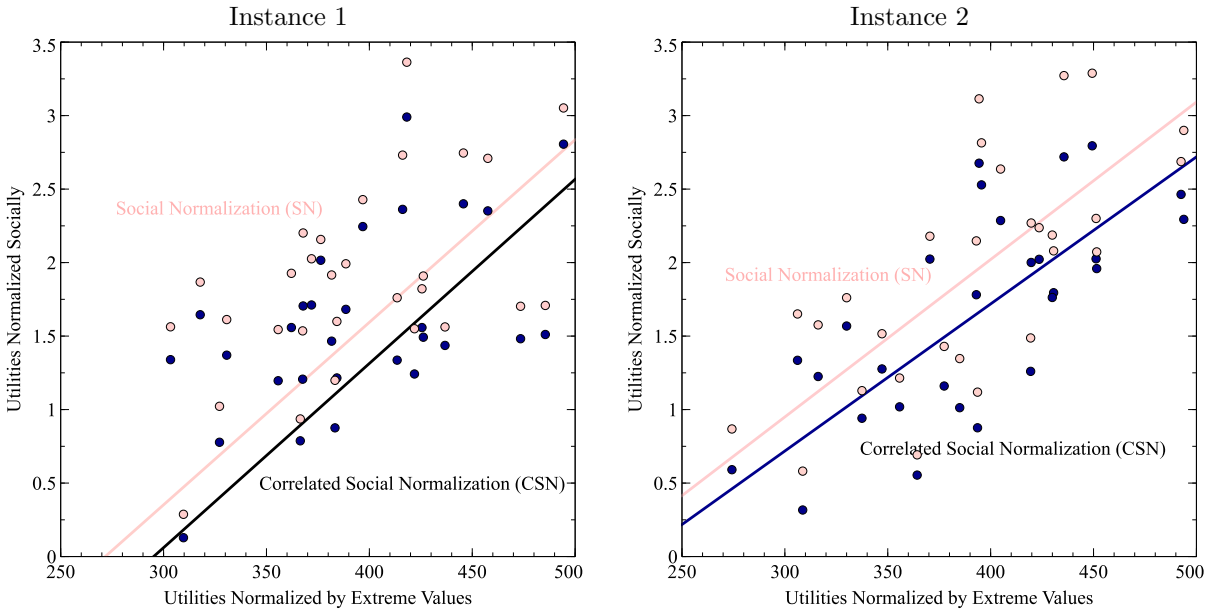

Figure 6: Relation between SN, CSN and EVN.

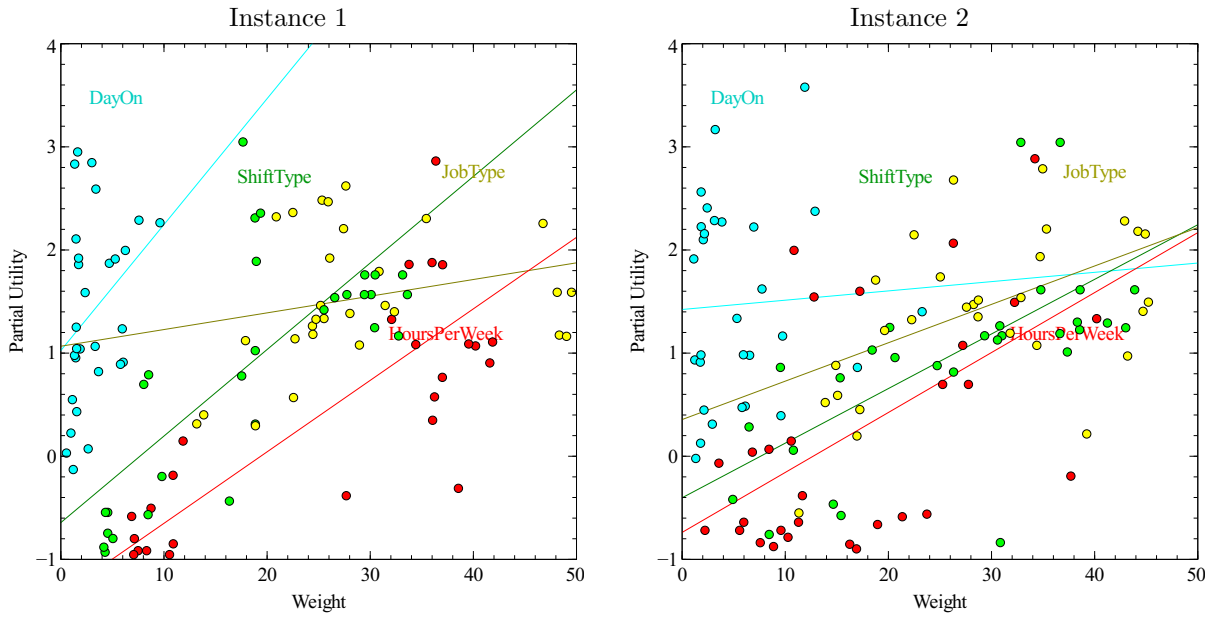

Figure 7: CSN partial utilities as a function of weights. Lines represent linear regressions that make up the best-fit affine standard utility functions.

Interestingly, our normalization enhances the fact that on a normalized scale, the inputs actually correspond to employees claiming they give little importance to the Day-On attribute compared to the Job-Type one. Also, there is clearly a lot of noise in the way that normalized weights affect CSN partial utilities.

4.3 AMACSN-SSP

Our results for AMACSN-SSP are reported in Figures 8 and 9, which describe properties of the outcomes obtained.

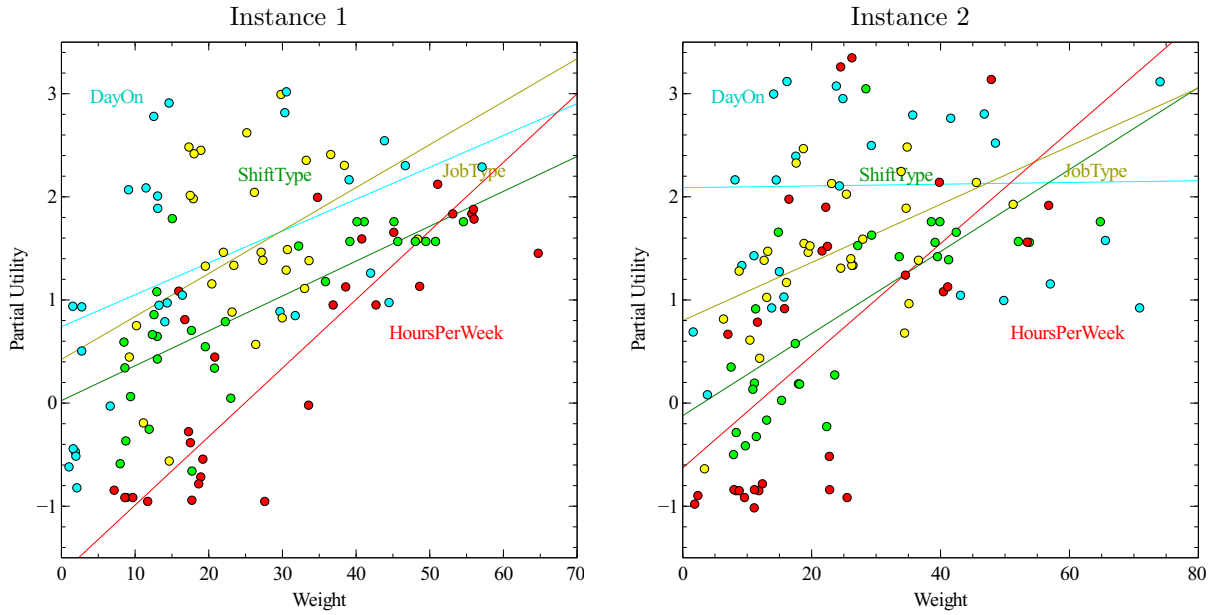

Figure 8: Partial utilities with AMACSN

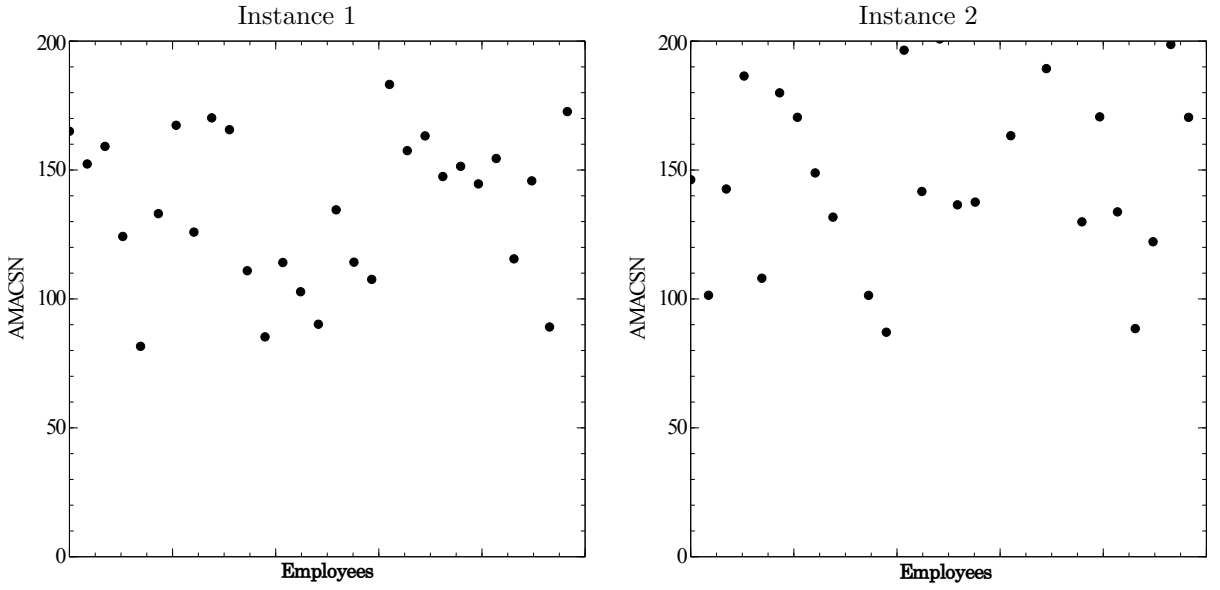

Figure 9: Utilities with AMACSN

Figure 8 displays the relation between AMACSN weights and CSN partial utilities. Interestingly, with the notable exception of the *Day-On* attribute of the second instance, the standard utilities all have fairly the same slope. This is evidence of a better balance between the normalizations of the different attributes. Figure 9 displays the AMACSN utilities of the solution of AMACSN-SSP. Arguably, the figure is fairly similar to Figure 1, even though the scale of the y-axis is different.

The normalization of utility functions, which are inputs of the shift-scheduling program, cannot foresee the outcomes of the shift-scheduling program. It is thus questionable whether this AMACSN fits the actual outcome it aims to describe. To see if this is the case, we may compare AMACSN to SN and CSN defined for the outcomes of the shift scheduling. This yields Figure 10.

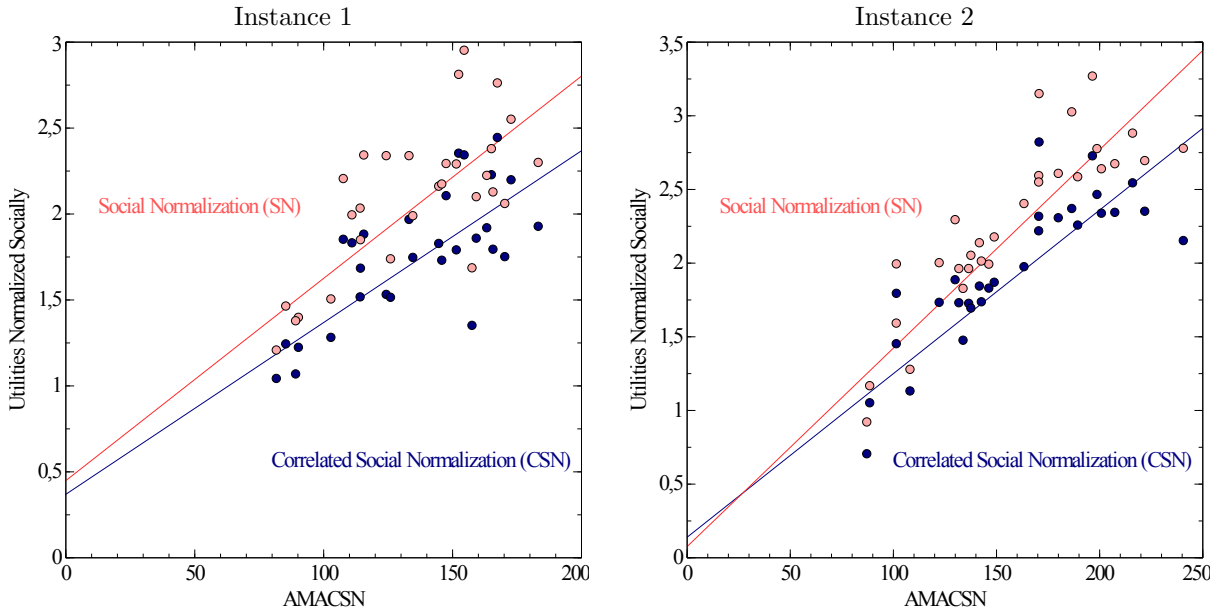

Figure 10: AMACSN compared to CSN and SN

The correlation between AMACSN and SN is 0.86, and that between AMACSN and CSN is 0.84. This is significantly better than the correlations between EVN and the social normalizations SN and CSN. These facts are evidence of the relevancy of AMACSN.

We can also notice that linear regressions nearly pass through the origin, which means that the utility of 0 has nearly the same meaning for all three normalizations. Interestingly, these graphs show that AMACSN has globally the same meaning as other normalizations overall, while it yields a deeper description of utilities by enabling comparisons of partial utilities.

4.4 EVN-SSP versus AMACSN-SSP

To actually judge the quality of AMACSN-SSP, we can compute SN and CSN utilities for this setting and compare the results to those of Figure 5. This is what we have done in Figure 11.

On this figure, we have added pale red and dark blue horizontal lines. They represent the average SN utilities for the solutions of, respectively, AMACSN-SSP and EVN-SSP. The figures display a significant improvement of the SN utilities with AMACSN. The figure also depicts the standard deviations of the SN utilities in the two different settings.

To make our analysis clearer, we have compared the averages and standard deviations of SN and CSN utilities for the two settings. Table 1 displays the results for SN utilities, while Table 2 corresponds to CSN utilities. The tables show a clear improvement by using AMACSN compared to EVN.

Regarding the average SN and CSN utilies, these improvements, depending on instances and SN/CSN, range from 12% to 22%. Note that this is much more than the optimality gap. This plainly justifies our focus on normalizations rather than on optimization algorithms.

Moreover, it is interesting to note that we also have a significant decrease of the standard deviations. This shows that, as a bonus, our normalization also guaranteed a significant increase in fairness. In particular, this hints at the fact that there is little need for concern for fairness in personalized shift scheduling, at least, when hardly any constraint differs between any two employees apart from their preferences. As long as we maximize the sum of employees' (well-normalized) utilities, both social efficiency and (a good amount of) fairness follow.

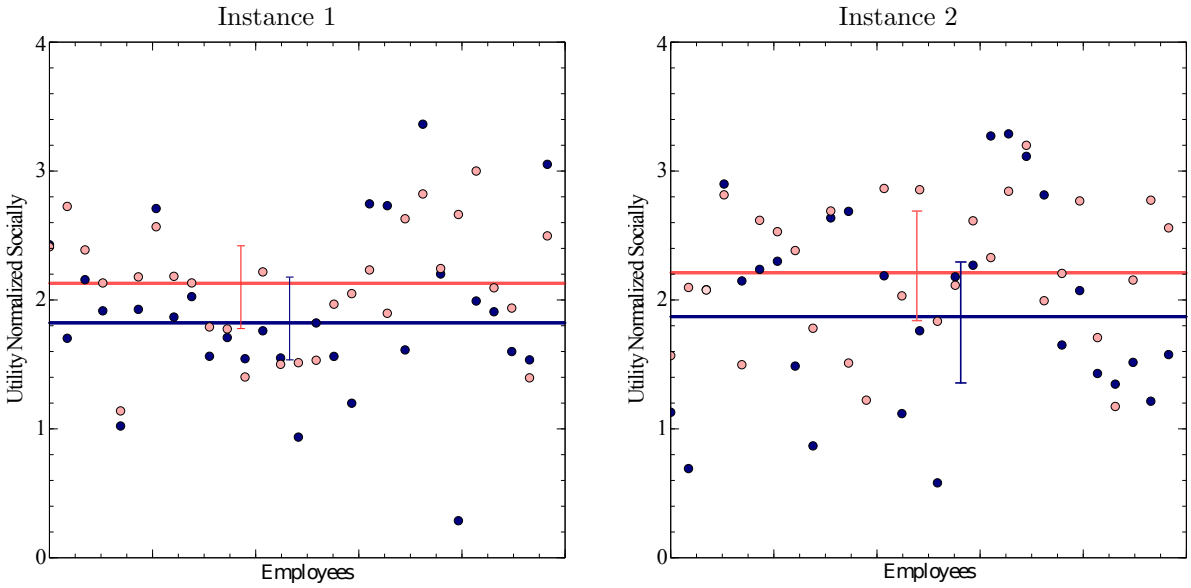

Figure 11: SN utilities of the AMACSN-SSP (pale red) compared to SN utilities of EVN-SSP (dark blue).

Table 1: Averages and standard deviations of SN utilities for AMACSN shift scheduling and EVN shift scheduling.

		AMACSN	EVN
Instance 1	Average of SN utilities	2.104	1.877
	Standard deviation of SN utilities	0.461	0.645
Instance 2	Average of SN utilities	2.235	1.871
	Standard deviation of SN utilities	0.541	0.860

Table 2: Averages and standard deviations of CSN utilities for AMACSN shift scheduling and EVN shift scheduling.

		AMACSN	EVN
Instance 1	Average of CSN utilities	1.788	1.582
	Standard deviation of CSN utilities	0.412	0.618
Instance 2	Average of CSN utilities	1.933	1.579
	Standard deviation of CSN utilities	0.471	0.794

5 Conclusion

In this paper, we have provided a model to perform optimized shift scheduling with employees' preferences. By including a linear multi-attribute setting and involving the state-of-the-art MACBETH method, we have proposed a relevant procedure to include employees' preferences. More importantly, we have characterized a new normalization of employees' linear multi-attribute utility functions, which, crucially, yields a meaningful way to compare any two utility functions, any two partial utility functions and any two weights. Amazingly, using this normalization has induced a significant improvement in the outcomes of the shift-scheduling program. Not only have we ensured a more socially efficient outcome, we have also shown that our approach guarantees even more fairness between employees.

References

L.H. Appelgren. A column generation algorithm for a ship scheduling problem. Transportation Science, 3(1):53–68, 1969.

T. Aykin. Optimal shift scheduling with multiple break windows. Management Science, 42(4):591–602, 1996.

C.A. Bana e Costa and J.-C. Vansnick. MACBETH: An interactive path towards the construction of cardinal value functions. International Transactions in Operational Research, 1(4):489–500, 1994.

J.F. Bard and H.W. Purnomo. Preference scheduling for nurses using column generation. European Journal of Operational Research, 164(2):510–534, 2005.

C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W. Savelsbergh, and P.H. Vance. Branch-and-price: Column generation for solving huge integer programs. Operations Research, 46(3):316–329, 1998.

S.E. Bechtold and L.W. Jacobs. Implicit modeling of flexible break assignments in optimal shift scheduling. Management Science, 36(11):1339–1351, 1990.

R. Benayoun, B. Roy, and B. Sussman. ELECTRE: Une méthode pour guider le choix en présence de points de vue multiples. Note de travail, 49, 1966.

J.-P. Brans and B. Mareschal. Prométhée-Gaia: une méthodologie d'aide à la décision en présence de critères multiples. Éditions de l'Université de Bruxelles, 2002.

J.-P. Brans, P. Vincke, and B. Mareschal. How to select and how to rank projects: The PROMETHEE method. European Journal of Operational Research, 24(2):228–238, 1986.

G.B. Dantzig. A comment on Edie's traffic delays at toll booths. Journal of the Operations Research Society of America, 2(3):339–341, 1954.

G. Desaulniers, J. Desrosiers, and M.M. Solomon. Column Generation, volume 5. Springer, 2005.

M. Desrochers and F. Soumis. A column generation approach to the urban transit crew scheduling problem. Transportation Science, 23(1):1–13, 1989.

T. Gontier. Longhaul cabin crew assignment. In 1985 AGIFORS Symposium Proceedings, volume 25, pages 44–66, 1985.

S. Greco, M. Kadziński, V. Mousseau, and R. Słowiński. ELECTRE GKMS: Robust ordinal regression for outranking methods. European Journal of Operational Research, 214(1):118–135, 2011.

P.E. Green and V. Srinivasan. Conjoint analysis in consumer research: Issues and outlook. Journal of Consumer Research, 5(2):103–123, 1978.

P.E. Green, A.M. Krieger, and Y. Wind. Thirty years of conjoint analysis: Reflections and prospects. Interfaces, 31 (3 supplement):S56–S73, 2001.

L.N. Hoang, F. Soumis, and G. Zaccour. Measuring unfairness feeling in allocation problems. Les Cahiers du GERAD G–2014–70, HEC Montréal, September 2014.

R.M. Karp. Reducibility among combinatorial problems. In R.E. Miller, J.W. Thatcher, and J.D. Bohlinger, editors, *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Springer US, 1972.

J.J. Louviere. Conjoint analysis modelling of stated preferences: A review of theory, methods, recent developments and external validity. Journal of Transport Economics and Policy, 93–119, 1988.

L.Y. Maystre, J. Pictet, J. Simos, and B. Roy. Méthodes multicritères ELECTRE: description, conseils pratiques et cas d'application à la gestion environnementale, volume 8. PPUR Presses Polytechniques, 1994.

S. Moondra. An LP model for work force scheduling for banks. Journal of Bank Research, 7(4):299–301, 1976.

O. Netzer, O. Toubia, E.T. Bradlow, E. Dahan, T. Evgeniou, F.M. Feinberg, E.M. Feit, S.K. Hui, J. Johnson, J.C. Liechty, J.B. Orlin, and V.R. Rao. Beyond conjoint analysis: Advances in preference measurement. Marketing Letters, 19(3-4):337–354, 2008.

B.K. Orme. Getting Started with Conjoint Analysis: Strategies for Product Design and Pricing Research. Research Publishers, LLC, 2005.

M. Rekik, J.-F. Cordeau, and F. Soumis. Implicit shift scheduling with multiple breaks and work stretch duration restrictions. Journal of Scheduling, 13(1):49–75, 2010.

Y. Siskos and A. Spyridakos. Intelligent multicriteria decision support: Overview and perspectives. European Journal of Operational Research, 113(2):236–246, 1999.

J. Von Neumann and O. Morgenstern. Theory of Games and Economic Behavior. Princeton University Press, 1944.

C. Zopounidis and M. Doumpos. Multicriteria classification and sorting methods: A literature review. European Journal of Operational Research, 138(2):229–246, 2002.