M U L T I P L E C R I T E R I A D E C I S I O N M A K I N G
Vol. 8 2013

Ignacy Kaliszewski[*]
Janusz Miroforidis[**]
Jarosław Stańczak[***]

DECISION MAKER'S PREFERENCES, AIRPORT GATE ASSIGNMENT PROBLEM AND MULTIOBJECTIVE OPTIMISATION

Abstract

We present an application of a methodology we developed earlier to capture a decision maker's preferences in multiobjective environments to a notorious problem in the realm of Air Traffic Management, namely the Airport Gate Assignment Problem.

The problem has been modelled as an all-integer optimisation problem with two criteria.

We have implemented this methodology into the commercial solver CPLEX and also into an Evolutionary Multiobjective Optimisation algorithm and we have solved with them a numerical instance of the Airport Gate Assignment Problem for a couple of decision making scenarios.

Keywords: preference capture, airport gate assignment, exact optimization computations, evolutionary optimization computations

1 Introduction

Nowadays Multiple Criteria Decision Making (MCDM) problems are most often solved interactively (Miettinen, 1999; Ehrgott, 2005; Kaliszewski, 2006). Interactive decision making processes reflect best the natural dynamics of the Decision Maker (DM) problem recognition, accumulation of knowledge about

[*] Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warszawa, Poland, e-mail: *ignacy.kaliszewski@ibspan.waw.pl*

[**] Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warszawa, Poland, Treeffect Co, Gdów 1028, 32-420 Gdów, Poland, e-mail: *jmiroforidis@gmail.com*

[***] Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warszawa, Poland, e-mail: *jaroslaw.stanczak@ibspan.waw.pl*

interplay of problem driving factors and DM's ability to reveal preferences about outcomes of different factor patterns.

Along this line, a methodology has been proposed to capture DM's preferences in the course of interactive decision making processes (Kaliszewski, 2004, 2006) which subsumes two classic MCDM methods, namely the *weighting method* and the *reference point method*. This methodology is quite general. It is applicable to any MCDM problem which uses Multiobjective Optimisation (MO) as the underlying formal model. Moreover, the methodology is independent of optimisation methods and solvers used to solve MO. Over time this methodology has been coupled with MCDM methods based on approximate calculations of efficient outcomes (Kaliszewski, 2006), evolutionary computations (Miroforidis, 2008, 2010; Kaliszewski, 2008; Kaliszewski, Miroforidis, 2009, 2012b; Kaliszewski et al., 2012), and classical optimisation calculations (Kaliszewski, Miroforidis, 2012a).

This work is an extension of an earlier research reported in Kaliszewski, Miroforidis (2012a). There we presented a model for a notorious problem of Air Traffic Management, namely the Airport Gate Assignment Problem (AGAP) (Dorndorf et al., 2007). The rationale behind the model was to assist air ground services in a small or medium size airport in assigning flights to gates under conflicting criteria. We showed that under the concept of time windows the problem can be effectively decomposed into a series of smaller problems and we argued that under such a decomposition the deterioration of optimality of solution, if any, is not very significant. To illustrate how the preference capture methodology works it was tested on a small instance of AGAP where efficient assignments were derived by enumeration. This, however, raised a question of scalability of the model to practical sizes. Moreover, the ability of optimisation software, academic and commercial, to cope with such problems in the preference capture methodology environment was still an open question.

To perform optimisation calculations, in this paper we use commercial software, namely the CPLEX package – a leader of many benchmarks. By this we attempt to convey the message that the methodology we developed couples well with commercial optimisation solvers able to handle effectively medium- and large-scale problems. This in turn paves the way for scalability of MCDM problems into the realm of such sizes.

We complement this work by mirroring CPLEX computations by an Evolutionary Multiobjective Optimisation (EMO) algorithm implemented specifically for that task. Our intention was to draw preliminary comparability conclusions on the workload and scalability with respect to those two distinct optimisation paradigms.

The outline of the paper is as follows. In the next section we present an adaptation of a model of AGAP adequate for small airports and presented in Kaliszewski, Miroforidis (2012a). In the subsequent section, for the sake of completeness, we give a concise description of our multiple criteria decision

making methodology (for a more detailed treatment cf. Kaliszewski et al., 2012). Lastly, we apply this methodology to a small but illustrative instance of the presented AGAP model with the help of the CPLEX package and our custom-coded EMO solver, and we comment briefly on the suitability of these solvers for medium- and large-scale instances of AGAP. We conclude with some remarks on possible directions of future research.

2 AGAP for a Small Airport

The problem under consideration is to assign incoming flights to airport gates in time horizon Θ. If at a given time there is no gate to serve a flight that flight (the corresponding plane) can be directed to wait for a gate or it can be served at once on the airport apron. Waiting times and the number of flights served on the apron are best if both are equal to zero but in the case of airport overload they are in an obvious conflict.

We assume that the airport under consideration:

1) is small, so gate assignments have no significant impact on passenger walking distance,
2) all gates can serve any flight,
3) there are no constraints on neighbour gate operations,
4) any flight can wait to be served at a gate for time T at most and after that time it is served on the apron.

2.1 The Model

A flight j, $j = 1, \dots, n$,t is characterised by arrival time a_j and ground operation time (for short: ground time) g_j (time needed to serve flight j at a gate). Arrival times, ground times and waiting times are assumed to be discrete with interval δ. Hence, the maximal waiting time is $T = \alpha\delta$ for some $\alpha > 0$ and time horizon is $\Theta = \beta\delta$ for some $\beta > \alpha$.

Let $x_{j,i}^t$ be a binary variable which is equal to 1 if flight j is assigned to gate i at time t and equal to 0 ~~if~~ otherwise. No assignment to a gate can be made before flight j arrives, hence for $t < a_j$ the variables $x_{j,i}^t$ are undefined. Similarly, no assignment to a gate can be made after a flight has waited α time intervals for a gate assignment (after that time this flight is served on the apron), hence for $t > a_j + \alpha$ the variables $x_{j,i}^t$ are undefined.

With m gates there are $m\sum_{j=1}^{n}(\alpha + 1) = mn(\alpha + 1)$ variables $x_{j,i}^t$.

A flight j can be assigned at most once to at most one gate, so

$$\sum_{i=1}^{m}\sum_{t=a_j}^{a_j+\alpha} x_{j,i}^t \leq 1, \text{ for } j = 1, \dots, n. \qquad (1)$$

There are n constraints of type (1).

Let $y_{j,i}^t$ be a binary variable which is equal to 1 if gate i is serving flight j at time t, and equal to 0 if otherwise. No assignment to a gate can be made before flight j arrives, hence for $t < a_j$ the variables $y_{j,i}^t$ are undefined. Similarly, gate i cannot serve flight j after $t > a_j + \alpha + g_j$ since after $t > a_j + \alpha$ flight j is served on the apron. Hence, for $t > a_j + \alpha + g_j$ the variables $y_{j,i}^t$ are undefined.

The number of variables $y_{j,i}^t$ is equal to $m\sum_{j=1}^{n}(\alpha + g_j + 1)$.

If flight j is assigned to gate i, $i = 1, \dots, m$, at time t (i.e. $x_{j,i}^t = 1$) then this gate is not available for another flight assignment for $1 + g_j$ consecutive time intervals starting from interval t, i.e. for interval $t,\ t+1, \dots, t+g_j$. This condition is equivalent to:

$$\begin{aligned} & x_{j,i}^t \le y_{j,i}^t , \\ & x_{j,i}^t \le y_{j,i}^{t+1} , \\ & \vdots \\ & x_{j,i}^t \le y_{j,i}^{t+g_j} , \end{aligned} \tag{2}$$

where $a_j \le t \le a_j + \alpha$.

There are $m\sum_{j=1}^{n}\alpha(g_j + 1)$ constraints of type (2).

As constraints of that type are the most numerous in the model, we propose a more concise formulation resulting from replacing constraints (2) by their logical equivalent:

$$g_j x_{j,i}^t \le y_{j,i}^t + y_{j,i}^{t+1} + \cdots + y_{j,i}^{t+g_j}, \tag{2'}$$

where $a_j \le t \le a_j + \alpha$.

The number of constraints of type (2') is equal to the number of variables $x_{j,i}^t$, hence it is equal to $m\sum_{j=1}^{n}(\alpha + 1) = mn(\alpha + 1)$.

No more than one flight can be assigned to a gate at a time, so:

$$\sum_{j=1}^{n} x_{j,i}^t \le 1,\ t = a_*, \dots, \beta,\ i = 1, \dots, m , \tag{3}$$

where * is the index of the flight scheduled as second. There are at most $m(\beta - a_* + 1)$ constraints of type (3); the exact number of these constraints depends on the flight arrival time structure.

Gate i at time t can serve at most one flight, so:

$$\sum_{j=1}^{n} y_{j,i}^t \le 1,\ t = a_*, \dots, \beta ,\ i = 1, \dots, m . \tag{4}$$

There are at most $m(\beta - a_* + 1)$ constraints of type (4); the exact number of these constraints depends on the flight arrival time structure.

If flight j is assigned to a gate at its arrival time a_j then there is no waiting time. Otherwise, the waiting time for flight j equals:

$$\delta\left(\sum_{i=1}^{m} x_{j,i}^{a_j+1} + 2\sum_{i=1}^{m} x_{j,i}^{a_j+2} + \cdots + \alpha\sum_{i=1}^{m} x_{j,i}^{\alpha}\right)$$

and the total waiting time over all flights is:

$$f_1(x) = \delta \sum_{j=1}^{n}(\sum_{i=1}^{m} x_{j,i}^{a_j+1} + 2\sum_{i=1}^{m} x_{j,i}^{a_j+2} + \cdots + \alpha \sum_{i=1}^{m} x_{j,i}^{\alpha}). \quad (5)$$

If flight j is not assigned to a gate then it is assigned to the apron, so:

$$\sum_{i=1}^{m} \sum_{t=a_j}^{t=a_j+\alpha} x_{j,i}^{t} = 1 - y_j, \text{ for } j = 1, \ldots, n, \quad (6)$$

where y_j is a binary variable equal to 0 if flight j is assigned to a gate and equal to 1 otherwise.

There are n variables y_j and n constraints of type (6).

With $f_2(y)$,

$$f_2(y) = \sum_{j=1}^{n} y_j \,, \quad (7)$$

as the objective function to be minimised, at optimality with respect to $f_1(x)$ (i.e. when variables $x_{j,i}^{t}$, $y_{j,i}^{t}$ and y_j are optimal with respect to $f_1(x)$ first and then with respect to $f_2(x)$, in that order) or at efficiency (i.e. when variables $x_{j,i}^{t}$, $y_{j,i}^{t}$ and y_j are efficient with respect to $f_1(x)$ and $f_2(x)$, for the definition of efficiency see the next section), the number of variables y_j taking value 1 will be minimal but not less than the value dictated by constraints (6).

Objective functions (5) and (7) together with constraints (1), (2'), (3), (4) and (6) constitute a bicriteria model for AGAP at a small airport. Values of objective functions at efficient assignment represent rational compromises between waiting time and the number of apron operations.

The model can accommodate also other objective functions because the multiple criteria decision making methodology we present in the next section can deal with any number of criteria. In Kaliszewski, Miroforidis 2012a, we considered a similar model where instead of the total waiting time the maximal waiting time over all flights was minimised. For that purpose in our earlier paper we used the following form of the first objective function:

$$f_1(x) = \delta \, max_j(\sum_{i=1}^{m} x_{j,i}^{a_j+1} + 2\sum_{i=1}^{m} x_{j,i}^{a_j+2} + \cdots + \alpha \sum_{i=1}^{m} x_{j,i}^{a_j+\alpha}). \quad (5')$$

However, this function works in the context of the AGAP problem correctly only if a solving method (solver) assigns flights to gates as soon as a gate is idle. However, with a general method (solver) this is not always guaranteed and this is the case with the general purpose solver CPLEX. In consequence, when working with objective function (5'), assignments can be derived in which for the minimal value of maximal (over flights) waiting time the minimal individual flight waiting time is not minimal. In other words, assignments can be derived in which, for the minimal value of maximal (over flights) waiting time, flights are not assigned to gates as early as possible (*aeap* assignments). Such assignments cannot be accepted in practice. This situation can be avoided either by adding to the model a significant number of constraints modelling precedence-type relations, which in consequence may hamper the scalability of the model, or by

adding to objective function (5') a penalty term in the form of a small fraction of objective function (5) to eliminate non-*aesp* assignments.

In this paper we have decided to apply a symmetric approach, namely to work primarily with objective function (5). This guarantees that all assignments will be *aeap* but raises, in turn, the question of individual flight waiting times. One can expect, however, that the maximal flight waiting time limited to α and a decomposition of the AGAP problem to a series of time windows (cf. the next section) will moderate the maximal waiting times of AGAP assignments produced. Eventually, to ensure that for a given value of the total waiting time the maximal waiting time is minimal (there can be multiple solutions with the same values of the total waiting time), one can add to objective function (5) a penalty term in the form of a small fraction of objective function (5').

2.2 Time windows

The model presented is all-integer and linear and if a penalty term in the form of a small fraction of objective function (5') is added to objective function (5), the model can be again made linear by a transformation of that term (a transformation analogous to that used in the next subsection to linearize in optimisation problem (9) the *maximum* function).

The problems (instances of the model) to be solved are of considerable size even for modest values of n, m and β (cf. the previous subsection). Although we have no influence on the magnitude of n and m, we can decrease the magnitude of β significantly by employing the concept of *time windows*.

Observe that in the model an apron is a buffer which absorbs all flights which cannot wait sufficiently long for an assignment to a gate. In the previous subsection we have set that threshold to $\alpha\delta$. Hence, in the model any flight is assigned to the apron at the latest at its arrival time plus $\alpha\delta$. Suppose that the time horizon $\Theta = \delta\beta$ of AGAP is divided into time windows of equal size such that window width is not greater than $\alpha\delta$ (see Appendix 1 for an example). Then, in the model a flight whose arrival time is in time window q will never compete for a gate assignment with a flight whose arrival time is in time window $q + 2$. Hence, the model for AGAP can be solved separately in each time window for all flights with arrival times in that window. Gate assignments in a given time window which overlap with the next time window can be represented in time window $t + 1$ by fixing the corresponding variables $y_{j,i}^{t}$ to 1 but this requires that AGAP have to be solved sequentially in separate time windows, starting from the first time window.

Solving AGAP in separate time windows does not guarantee optimality with respect to the whole time horizon Θ but makes the entire problem much more manageable from the computational point of view.

3 The Multiobjective Methodology

Let x denote a (decision) *variant* (solution), X a space of variants, X_0 a set of *feasible variants*, $X_0 \subseteq X$. Then the multiobjective optimisation problem is:

$$\text{"vmax"}\ f(x) \qquad x \in X_0\,, \tag{8}$$

where $f: X \to R^k$, $f = (f_1, \ldots f_k)$, $f_i: X \to R$, $i = 1, \ldots, k$, $k \geq 2$, are *objective functions* (*criteria*); "vmax" denotes the operator of deriving all *efficient* (as defined below) *variants* in X_0.

A variant $\bar{x}$ of X_0 is *efficient* if $f_i(x) \geq f_i(\bar{x})$, $i = 1, \ldots, k$, $x \in X$, implies $f(x) = f(\bar{x})$.

It is a well-established result (cf. Kaliszewski, 2006; Ehrgott, 2005; Miettinen, 1999) that variant $\bar{x}$ is efficient[1] if and only if it solves the optimisation problem:

$$min_{x \in X_0} max_i [\lambda_i (y_i^* - f_i(x)) + \rho e^k (y^* - f(x))]\,, \tag{9}$$

where $\lambda_i > 0$, $i = 1, \ldots, k$, $e^k = (1, \ldots, 1)$, y^* is such that $y_i^* > f_i(x)$, $i = 1, \ldots, k$, $x \in X_0$, and ρ is a positive "sufficiently small" number[2].

By the "only if" part of this result no efficient variant is a priori excluded from being derived by solving an instance of optimisation problem (9). In contrast to that, the maximisation of a weighted sum of objective functions over X_0 does not have, in general (and especially in the case of problems with discrete variables), this property.

Let $\hat{y}_i = max_{x \in X_0} f_i(x)$, $i = 1, \ldots, k$. We can restate the above result saying that a variant $\bar{x}$ such that $f_i(\bar{x}) \neq \hat{y}_i$ for all $i = 1, \ldots, k$, is efficient (cf. footnote 4) if and only if it solves the optimisation problem:

$$min_{x \in X_0} max_i [\lambda_i (\hat{y}_i - f_i(x)) + \rho e^k (\hat{y} - f(x))]\,, \tag{9'}$$

This is an immediate consequence of accounting in the proof of the only if part of Theorem 4.3 in Kaliszewski (1994) for the condition $f_i(\bar{x}) \neq \hat{y}_i$ for all $i = 1, \ldots, k$. Efficient solutions with $f_i(\bar{x}) = \hat{y}_i$ for some $i \in \widetilde{K} \subseteq \{1, \ldots, k\}$, can be derived with the optimisation problem:

$$min_{x \in X_0} max_{i \notin \widetilde{K}} [\lambda_i (\hat{y}_i - f_i(x)) + \rho e^k (\hat{y} - f(x))]\,, \quad f_i(x) = \hat{y}_i\,,\ i \in \widetilde{K}\,. \tag{9''}$$

At the first glance, the objective function in (9') ($max_i [\lambda_i (\hat{y}_i - f_i(x)) + \rho e^k (\hat{y}_i - f(x)]$) seems difficult to handle. However, observe that optimisation problem (9') is equivalent to:

---

[1] Actually, variant $\bar{x}$ is ρ *−properly efficient*, for a formal treatment of this issue cf. e.g. Kaliszewski (2006), Ehrgott (2005), Miettinen (1999).

[2] Kaliszewski (2006), Ehrgott (2005), Miettinen (1999).

$$min\ s\ ,$$
$$s \geq \lambda_i\big(\hat{y}_i - f_i(x)\big) + \rho e^k\big(\hat{y}_i - f(x)\big),\ i = 1, \dots, k\ ,$$
$$x \in X_0\ .$$

An analogous observation applies to optimisation problem (9").

Besides the potential ability to derive each efficient variant, optimisation problem (9') provides an easy and intuitive capture of decision maker's preferences. Observe that an element $\hat{y}$ (recall that $\hat{y}_i = max_{x \in X_0} f_i(x),\ i = 1, \dots, k$) represents maximal values of objective functions which can be attained if they are maximised separately.

To assist the decision maker in the search for *the most preferred variant* one can employ the optimisation problem (9') or (9"). By this we assume a modicum of rationality on the part of the decision maker, namely we assume that the decision maker prefers an efficient variant to any variant dominated by it (variant $\bar{x}$ dominates variant x if $f_i(\bar{x}) \geq f_i(x),\ i = 1, \dots, k,$ and $f_i(\bar{x}) > f_i(x)$ for at least one i.

Suppose that an element $x \in X_0$ such that $\hat{y} = f(x)$ does not exist which is rather a standard situation of conflicting criteria (otherwise, x is clearly the most preferred variant). Then, the decision maker knows that to derive an efficient variant he (or she) has to compromise on values of objective functions f_i with respect to values $\hat{y}_i\ ,\ i = 1, \dots, k$. He can define his acceptable compromises on values $\hat{y}_i\ ,\ i = 1, \dots, k,$ and by this direct of search for an efficient variant which corresponds to these compromises in three ways:

1) providing a *vector of concessions* τ ,
2) providing a reference point y^{ref},
3) providing weights $\lambda_i\ ,\ i = 1, \dots, k$.

Way 1. The components of a vector of concessions τ specify concessions the decision maker is willing to make with respect to $\hat{y}_i\ ,\ i = 1, \dots, k$. The components of the vector τ can be defined in absolute values ("the decision maker is willing to make a concession of τ_i units on the value $\hat{y}_i\ ,\ i = 1, \dots, k$") or in relative values ("the decision maker is willing to make a concession of τ_i per cent on the value $\hat{y}_i\ ,\ i = 1, \dots, k$").

Way 2. A reference point y^{ref} ($y^{ref} \in R^k,\ y_i^{ref} \leq \hat{y}_i\ ,\ i = 1, \dots, k$), (it is irrelevant whether there exists an element $x \in X_0$ such that $y^{ref} = f(x)$ or not) specifies explicitly a compromise between values of objective functions f_i with respect to values $\hat{y}_i\ ,\ i = 1, \dots, k,$ which the decision maker regards as agreeable (Wierzbicki, 1999). A reference point specifies indirectly a vector of concessions:

$$\tau_i = \hat{y}_i - y_i^{ref},\ i = 1, \dots, k\ . \tag{10}$$

Way 3. An experienced decision maker can define a vector of concessions τ in terms of weights $\lambda_i > 0,\ i = 1, \ldots, k$, in optimisation problem (9') or (9'').

For $\tau_i > 0,\ i = 1, \ldots, k$, the vector of concessions τ and the vector of weights λ are related by the formula (Kaliszewski, 2006):

$$\lambda_i = (\tau_i)^{-1},\ i = 1, \ldots, k, \tag{11}$$

If $\tau_i = 0$ for some i then this means that the decision maker is not willing to make any concessions on $\hat{y}_i$ and he (she) is interested only in efficient solutions for which $f_i(x) = \hat{y}_i$ for that i. Then optimisation problem (9') should be replaced by optimisation problem (9'') with $\widetilde{K}$ composed of all indices $i = \{1, \ldots, k\}$ such that $\tau_i = 0$. Hence, from now on we shall assume without loss of generality that $\tau_i > 0,\ i = 1, \ldots, k$.

The optimisation problem (9'), if solved with $\lambda_i,\ i = 1, \ldots, k$, given by formula (11), has the following property:

- it finds an efficient variant x such that $f(x)$ is on the half line $y = \hat{y} - t\tau,\ t \geq 0$, whenever such a variant exists,
- otherwise, it finds an efficient variant x such that $max_i \lambda_i(\hat{y}_i - f_i(x)) + \rho e^k(\hat{y} - f(x)) = max_i \lambda_i(\hat{y}_i - \tilde{y}_i) + \rho e^k(\hat{y} - \tilde{y})$, where $\tilde{y}$ is on the half line $y = \hat{y} - t\tau,\ t \geq 0$.

4 Solving an AGAP Instance

Consider the following instance of AGAP. In the time horizon of 2 hours there are 5 flights scheduled as in Table 1. These flights can be served at two gates or on apron. The discretisation interval is $\delta = 5$ minutes. All ground times are equal to 50 minutes. The upper bound on waiting time α is 6 discretisation intervals, i.e. 30 minutes.

Table 1

TIME WINDOW *i*

FLIGHT	ARRIVAL TIME
1	0:05
2	0:15
3	0:30
4	0:40
5	0:45

We illustrate on this example the MCDM methodology for decision maker's preference capture presented in the previous section. The methodology has been outlined for problems where all objective functions are maximised whereas in our example objective functions, waiting time and the number of apron operations, are to be minimised. Thus, to have in our example both objective

functions in the "max" form, we simply change signs of values of objective functions. For example, if we maximise in the *adapted* problem "minus" the number of apron operations then -1 apron operation is better than -3 apron operations, which is a purely technical convention.

4.1 The AGAP Instance

The instance of our AGAP model to be solved has the following structure.

There are 70 binary variables $x_{j,i}^t$, 170 binary variables $y_{j,i}^t$ and 5 binary variables y_j , 245 binary variables in total.

There are 70 constraints of type (2'), 44 constraints of type (3), 23 constraints of type (4) and 5 constraints of type (6), 142 constraints in total.

In addition to structural variables $x_{j,i}^t$, $y_{j,i}^t$, y_j , to simplify manipulations with objective functions (5) and (7), we add (in this case nonnegative) variables f_1, f_2:

$$f_1 = f_1(x) \text{ and } f_2 = f_2(x).$$

This increases the size of the model to 247 variables, 245 binary and 2 continuous and nonnegative, and 144 constraints among which 137 are of "less than or equal to" type and 7 are of "equal to" type.

For optimisation problem (9) one additional (in this case nonnegative) variable and two additional "greater than or equal to" type constraints are needed. In this case the model has 248 variables, 245 binary and 3 continuous and nonnegative, and 146 constraints among which 137 are of "less than or equal to" type, 2 are of "greater than or equal to" type and 7 are of "equal to" type.

In CPLEX nonnegativity of variables is assumed by default, so there is no need to add such constraints explicitly.

4.2 Solving the AGAP Instance with CPLEX

We used the above model to simulate an AGAP decision making process with calculations performed by CPLEX solver.

To derive $\hat{y}$, first we have calculated the maximal value of the function $-f_1(x)$ (formula (5)). The optimal value of this function is 0. It is worth observing that CPLEX produced this value with 0 gate and 5 apron operations, while clearly for the same value of $-f_1(x)$ only 3 apron operations are feasible (any 2 out of 5 flights can be served by two gates without a delay, so there are $\binom{5}{2} = 20$ solutions with this optimal value), thus the assignment produced is not efficient (it is weakly efficient). The efficiency of assignments is not essential when deriving just $\hat{y}_i$ but to work with efficient assignments only we maximized $-f_1(x) - \varepsilon f_2(x)$ with $\varepsilon = 0.00001$ (cf. a comment in Section 2

on the value of the function $f_2(x)$ at the optimality of the function $f_1(x)$). This modified objective function produced an efficient assignment #1 (Table 2; in tables below WT denotes waiting time and #APRON denotes the number of apron operations).

Next we have calculated the maximal value of the function $-f_2(x)$ (formula (7)). The optimal value of this function is -1. For the same reasons as above we maximized $-\varepsilon f_1(x) - f_2(x)$ with the same value of ε. This modified objective function produced an efficient assignment #2 (Table 2).

Table 2

FLIGHT-GATE/APRON ASSIGNMENTS BY CPLEX

No	Scenario	GATE 1	GATE 2	APRON	- WT	- #APRON
#1	y^{ref} = (-15.00,-1.00) λ = (1.00,1.00)	1	4	2,3,5	0	-3
#1*a*	τ = (5.00,1.00)	2	1	3,4,5	0	-3
#2	-	1,4	2,5	3	-45	-1
#3	τ = (10.00,1.00)	1,5	3	2,4	-15	-2
#3*a*	y^{ref} = (-25.00,-2.00)	3	1,5	2,4	-15	-2
#3b	λ = (1.00,23.00)	4	1,5	2,4	-15	-2

So in the *adapted* problem we have $\hat{y} = (0, -1)$.

The clearly best combination: 0 waiting time and -1 apron operation is not possible in that problem (if it were, the maximization of $-f_1(x) - \varepsilon f_2(x)$ or $-f_1(x) - \varepsilon f_2(x)$ would produce it) so the decision maker has to compromise on $\hat{y}$, i.e. accept assignments which are worse than this combination with respect to at least one objective function.

As presented in the previous section, the decision maker can define his favourable compromises in three ways. Here we show how he can act along each of these ways. Below, in all computations we have used $\rho = 0.00001$ (cf. formulae (9), (9'), (9")).

1. Suppose that the decision maker is willing to make concessions on the (impossible) best combination $\hat{y}$ and he defines such concessions by (the vector of) favourable concessions: (10 minutes waiting time, 1 apron operation). Hence $\tau = (10.00, 1.00)$. By formula (11) $\lambda = (0.10, 1.0)$. With these weights the objective function of optimisation problem (9') has the smallest value for assignment #3 (15 minutes waiting time, 2 apron operations) (see Figure 1).

Suppose now that the decision maker is willing to make concessions on the (impossible) best combination $\hat{y}$ but this time he defines such concessions by (the vector of) favourable concessions: (5 minutes waiting time, 1 apron operation). Hence $\tau = (5.00, 1.00)$. By formula (11) $\lambda = (0.20, 1.00)$. With

these weights the objective function of optimisation problem (9') has the smallest value for assignment #1*a* (0 minutes waiting time, 3 apron operations). Here *#Nx* denotes an assignment which has the same values of objective functions as assignment *#N* but differs from assignment *#N* and possibly other assignments *#Nx* by flight gate/apron assignments.

2. Suppose that the decision maker specifies explicitly a compromise between the number of apron operations and waiting time he would like to achieve or at least to approximate as closely as possible: 15 minutes waiting time, 1 apron operation. Observe that in our problem there is no such assignment, nevertheless the reference point $y^{ref} = (-15.00, -1.00)$ (signs have to be reverted for the *adapted* problem) captures the decision maker's preferences for that point as described in the previous section. By formula (10) and formula (11) $\tau = (15.00, 0.00)$. As $\tau_2 = 0$, this means that the decision maker is not willing to make any concessions on $\hat{y}_2 = -1$. Hence the problem reduces to maximization of $-f_2(x)$ which has been already done when calculating $\hat{y}_1$ and this yielded assignment #1.

Suppose now that the decision maker specifies explicitly another compromise between the number of apron operations and waiting time he would like to achieve or at least to approximate as closely as possible: 25 minutes waiting time, 2 apron operations. Hence, $y^{ref} = (-25.00, -2.00)$ and by formula (10) $\tau = (25.00, 1.00)$ and by formula (11) $\lambda = (0.04, 1.00)$. With these weights the objective function of optimisation problem (9) has the smallest value for assignment #3*a*.

Suppose that the decision maker specifies directly two vectors of weights λ where from his experience with the problem (e.g. the problem was solved many times in the past) he knows that the first vector leads to assignments with a small number of apron operations whereas the second leads to assignments with low waiting times. Let those vectors be: $\lambda^1 = (1.00, 23.00)$ and $\lambda^2 = (1.00, 1.00)$.

In the first case the objective function of optimisation problem (9') has the smallest value for assignment #3*b* (15 minutes waiting time, 2 apron operations).

In the second case the objective function of optimisation problem (9') has the smallest value for assignment #1 (0 minutes waiting time, 3 apron operations).

All scenarios for the problem have been solved with CPLEX on a UNIX platform in less than 0.1 seconds.

4.3 Solving the AGAP Instance with Evolutionary Optimisation

We have mirrored all calculations listed in the previous section with a custom-made evolutionary optimisation algorithm (E−AGAP). A brief description of this algorithm is given in Appendix 2.

As expected, for all the decision making scenarios considered in Section 4.2, E−AGAP has produced the same results as CPLEX with respect to values of objective functions, as shown in Table 3. However, for the same values of objective functions, E−AGAP and CPLEX produced different solutions (assignments).

All scenarios for the problem have been solved with algorithm E−AGAP on a PC computer with Linux operating system in less than 1 second.

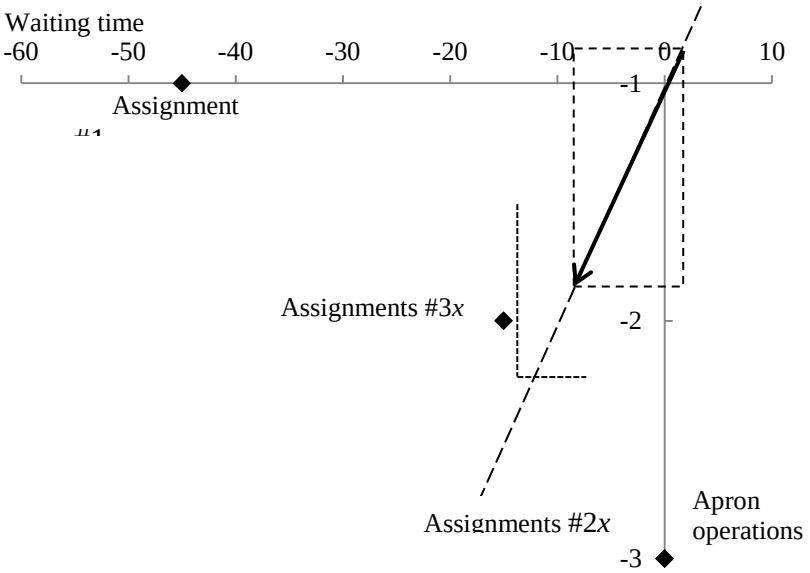

Figure 1. A graphical interpretation of searching for optimal solution in the example problem with the vector of concessions $\tau = (10.00, 1.00)$. Here the smallest value of the objective function in optimisation problem (9') is attained for assignment #3

Table 3

FLIGHT-GATE/APRON ASSIGNMENTS BY E−AGAP

No	Scenario	GATE 1	GATE 2	APRON	- WT	- #APRON
#1*b*	τ = (5.00, 1.00)	4	5	1, 2, 3	0	-3
	y^{ref} = (-15.00,-1.00)	4	5	1,2,3	0	-3
	λ = (1.00, 1.00)	4	5	1,2,3	0	-3
#3c	τ = (10.00, 1.00)	1,5	2	3,4	-15	-2
	y^{ref} = (-25.00, -2.00)	1,5	2	3,4	-15	-2
	λ = (1.00, 23.00)	1,5	2	3,4	-15	-2

5 Concluding Remarks

In principle there is no limit for applicability of the methodology we have developed and we recommend to problems of higher dimensions than those solved in the paper as long as a solver can handle problem (9).

In the paper we have solved a small instance of a multiple criteria decision problem using a commercial optimisation package. Here by solving multiple criteria decision problems we mean the ability to derive any efficient solution (decision variant) which the decision maker implicitly points to by his/her preferences.

We have shown that the methodology for decision makers' preference capture seamlessly works with CPLEX with no need for any adaptation of the package, so in fact it would work in that manner with any commercial solver. As CPLEX is reported to be able to solve problems with thousands of variables and constraints this opens the door for applying MCDM to practical problems of considerable sizes as required in some industries.

We have also shown that heuristics, such as Evolutionary Multiobjective Optimisation, can be easily fitted to our methodology as a potential viable alternative to CPLEX. The lack of guarantee of optimality in such methods is outweighed by their flexibility, adaptability and low cost.

This poses the question of how those two computing paradigms relate to each other in the function of growing size and complexity of decision making problems such as AGAP. In more general terms, it would be of utmost interest and importance for MCDM community to to which extent heuristics such as Evolutionary Multiobjective Optimisation (cf. e.g. Kaliszewski et al., 2012), can be competitive with the exact optimisation. Investigations of that question are the intended topic for our future research.

Appendix 1

Consider a flight schedule as in Table 1, Section 4. As said in Section 4.2, the upper bound for the width of time windows allowing sequential solving of the AGAP is $\alpha\delta$, which in this case is 30 minutes. It is shown in Figure 1 that to this aim time windows of $3\delta = 15$ minutes are not wide enough and in Figure 2 it is shown that time windows of $4\delta = 20$ minutes of the minimal required size.

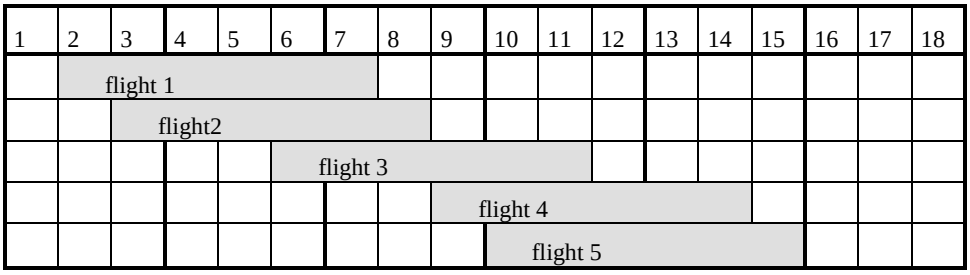

Figure 1. For time windows of minutes some flights extend over more than two windows

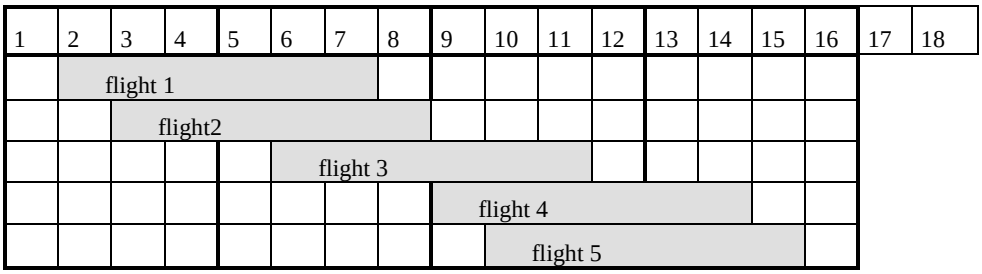

Figure 2. For time windows of minutes no flight extends over more than two windows

Appendix 2

The purpose of solving AGAP with evolutionary optimisation was to get some comparative experience in solving MCDM problems with a commercial software (such as CPLEX) and a sort of heuristics. To that aim we have implemented a heuristic algorithm – E−AGAP − which mimics principles of evolutionary computations. At the current stage, the algorithm solves just the instance of AGAP used in this paper as an illustrative numerical example, but it can be easily parameterised to handle any AGAP instances.

In the algorithm each individual is composed of three queues, two for the gates and one for the apron. In other words, each individual represents a variant of gate/apron assignment. In the initialisation step for each individual flights are randomly scheduled to queues. Then, in consecutive iterations, flights are rescheduled by genetic-like operators. Three operators are used: exchange operator, relocation operator and selection operator.

The exchange operator chooses randomly two flights assigned to different queues and exchanges them. The relocation operator selects randomly a flight from a queue and places it in a randomly selected position in a gate queue (not apron queue). If possible with respect to the fitness function, both operators can be used in the same iteration several times on the same individual.

Following the idea exposed in Stańczak (2003), in the course of the algorithm a record of actions of these two operators is kept for each individual. An

operator with more historical success in improving values of objective functions for a given individual has a better chance to be called to act on that individual.

In contrast to the exchange operator and the relocation operator, which act if called to, the selection operator is called in each iteration to select individuals from the next iteration population.

The fitness function used in our paper is in fact a mechanism which besides computing values of objective functions has a built-in functionality of correcting infeasible solutions, namely: if a flight waits too long for a service, the fitness function assigns it to the apron queue.

To account for multiple criteria interplay the selection operator employed promotes nondominated solutions. If the population of nondominated solutions becomes too small some dominated best-fit solutions also become candidates for selection.

The evolutionary optimisation algorithm composed in this manner has been effective in solving the instance of AGAP problem considered in this paper. For more challenging instances of AGAP the algorithm can be endowed with some other (as many as appropriate) genetic-like operators (cf. Stańczak, 2003).

References

Dorndorf U., Drexel A., Nikulin Y., Pesch E. (2007) *Flight gate scheduling: State-of-the-art and recent development*. Omega 35, p. 326-334.

Ehrgott M., (2005) *Multicriteria Optimization*, Springer.

Kaliszewski I. (2004) *Out of the Mist – Towards Decision-maker-friendly Multiple Criteria Decision Making Support*, European Journal of Operational Research, 158, p. 93–307.

Kaliszewski I. (1994) *Quantitative Pareto Analysis by Cone Separation Technique*, Kluwer Academic Publishers.

Kaliszewski I. (2006) *Soft Computing for Complex Multiple Criteria Decision Making*, Springer.

Kaliszewski I. (2008) *Multiple Criteria Decision Making: Outcome Assessments with Lower and Upper Shells*, Systems Research Institute Report RB/9/2008, Warszawa.

Kaliszewski I., Miroforidis J. (2009) *Multiple Criteria Decision Making: Efficient Outcome Assessments with Evolutionary Optimization*, Communications in Computer and Information Science 35, p. 25–28.

Kaliszewski I., Miroforidis J. (2012a) *On Interfacing Multiobjective Optimisation Models – the Case of the Airport Gate Assignment Problem*, Proceedings of the 2nd International Conference on Application and Theory of

Automation in Command and Control Systems (ATACCS'2012), IRIT Press, p. 93-97.

Kaliszewski I., Miroforidis J. (2012b) *Real and virtual Pareto set upper approximations*, in: *Multiple Criteria Decision Making'11*, Ed. T. Trzaskalik, T. Wachowicz, The Publisher of University of Economics in Katowice, p. 121-131.

Kaliszewski I., Miroforidis J., Podkopaev D. (2012) *Interactive Multiple Criteria Decision Making Based on Preference Driven Evolutionary Multiobjective Optimization with Controllable Accuracy*, European Journal of Operational Research, 216, p. 293–307.

Miroforidis, J. (2008) Private communication.

Miroforidis J. (2010) *Decision Making Aid for Operational Management of Department Stores with Multiple Criteria Optimization and Soft Computing*, PhD Thesis, Systems Research Institute, Warsaw.

Miettinen K.M. (1999) *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers.

Stańczak J. (2003) *Biologically inspired methods for control of evolutionary algorithms*, Control and Cybernetics 32(2), p. 411-433.

Şeker, M. (2011) *Stochastic optimization models for the airport gate assignment problem*, Transportation Research Part E, 48, p. 438-459.

Yan S., Huo Ch-M. (2001) *Optimization of multiple objective gate assignments*, Transportation Research Part A, 35, p. 413-432.

Wierzbicki A.P. (1999) *Reference point approaches*, in: *Multicriteria Decision Making – Advances in MCDM: Models, Algorithms, Theory and Applications*, Ed. T. Gal, Th. Stewart, Th. Hanne, Kluwer Academic Publishers.