Modeling Paradigms Applied to the Analysis of European Air Quality

Marek Makowski
International Institute for Applied Systems Analysis, Laxenburg, Austria

RR-00-06
March 2000

Reprinted from *European Journal of Operational Research*, 122 (2000) 219–241.

ELSEVIER

European Journal of Operational Research 122 (2000) 219–241

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/orms

Modeling paradigms applied to the analysis of European air quality

Marek Makowski [*]

International Institute for Applied Systems Analysis, A-2361 Laxenburg, Austria

Received 1 October 1998; accepted 1 April 1999

Abstract

The paper presents an overview of various modeling paradigms applicable to the analysis of complex decision-making problems that can be represented by large non-linear models. Such paradigms are illustrated by their application to the analysis of a model that helps to identify and analyze various cost-effective policy options aimed at improving European air quality. Also presented is the application of this model to support intergovernmental negotiations. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Decision support systems; Model management; Non-linear models; Preprocessing; Robustness; Multiple-criterion optimization; Criterion functions; Object-oriented programming; Air pollution

1. Introduction

Recent developments in Operations Research (OR) provide better opportunities to support various stages of decision-making processes that require the analysis of a complex problem that can be represented by a mathematical model. However, such opportunities can easily be misused, especially in cases of an analysis of a complex large-scale problem. Therefore, it is important to discuss a number of methodological and technical issues pertaining to the specification and analysis of large-scale complex models – which are of a broader interest to OR practitioners – and to illustrate them using a relevant real-world problem. Such a problem is provided by the Transboundary Air Pollution (TAP) Project at IIASA, which has over several years developed the `RAINS` models used to support international negotiations. The models help to identify cost effective measures aimed at improving the air quality assessed by several indicators at approximately 600 receptors throughout Europe. The resulting models are large (over 25,000 variables) and non-linear. Moreover, supporting international negotiations requires various types of analysis of a complex model, as well as interaction with its users. Both help to determine the requirements for the model-based decision support methods applicable in this case. The first

[*] Tel.: +43-2236-8070; fax: +43-2236-71-313.
E-mail address: marek@iiasa.ac.at (M. Makowski);
URL: http://www.iiasa.ac.at

requirement is the domain of the developers, who have to design and implement a mathematical model, relevant data bases and software tools for various types of model analysis. The second involves making the model specification acceptable to the users and the results of various analysis comprehensible to them, as well. Both these domains are closely related. However, due to space limitations and the complexity of the problems related to the second domain, this paper will focus only on the issues pertaining to the first domain.

The paper is organized in the following way. Section 2 provides a descriptive summary of the problem of analyzing cost-effective policies aimed at improving European air quality. Section 3 summarizes basic concepts of model-based decision support. An overview of various modeling paradigms that pertain to the model specification and analysis is presented in Section 4. A more formal discussion of the RAINS core model specification is presented in Section 5. Various problems of a more general interest, and related to model generation and preprocessing are discussed in Section 6. Section 7 illustrates the application of selected methods of model analysis discussed in Sections 3 and 4 to the RAINS model. Finally, Section 8 presents an overview of the role of the RAINS model for supporting international negotiations.

2. Problem description

The interest in the air quality has intensified in recent years because of the increasing evidence that acidification, eutrophication and ground-level ozone can have adverse effects on crops, trees, materials and human health. Moreover, substantial progress has been made in quantifying the environmental sensitivities of various ecosystems. The corresponding threshold values have been determined on a European scale, focusing on acidification and eutrophication, as well as on vegetation damage from tropospheric ozone. In many parts of Europe, the critical levels/loads of air pollution indicators are exceeded, thus requiring measures to improve the air quality in these areas to help protect the relevant ecosystems.

Several international agreements have been reached in Europe over the last decade to reduce emissions. Most of the current agreements determine the required abatement measures solely in relation to the technical and economic characteristics of the sources of emissions, such as the available abatement technologies, costs, historic emission levels, etc. However, to achieve an overall cost-effectiveness of strategies, the environmental benefits of the proposed measures must also be taken into account. To this end the Transboundary Air Pollution (TAP) Project at IIASA has for several years been developing – in collaboration with several European institutions – the RAINS models that have been used to support the negotiations of international agreements on controlling air pollution in Europe. The RAINS model helps to identify cost effective emission control policies aimed at keeping the values of the previously mentioned indices below levels determined for each of approximately 600 receptors across Europe.

The structure of the current version of the RAINS model is outlined in Fig. 1. The decision variables are composed of the levels of emissions of NH_3 (ammonia), SO_x (sulphur oxides), NO_x (nitrogen oxides) and VOC (volatile organic compounds) in each country, which imply the corresponding emission control policies. Each type of emission has for each country an associated cost function that relates the emission level with the corresponding cost of reducing the emission to a certain level. Therefore, cost-effective measures can be calculated by a minimization of the cost function that corresponds to the sum of costs related to reductions of all types of considered emissions in all countries. In order to determine the corresponding environmental impact, emission levels are used as inputs to the three dispersion submodels and to the ozone formation submodel. Studies of the impact of ozone, acidification and eutrophication have resulted in the establishment of critical levels for various air-quality indicators in order to protect agricultural crops and forests. These are determined using a long-term exposure measure, called the *accumulated excess*. Consequently, nine such exposure indices (six for ozone, two for acidification and one for eutrophication) has been defined for each of approximately 600

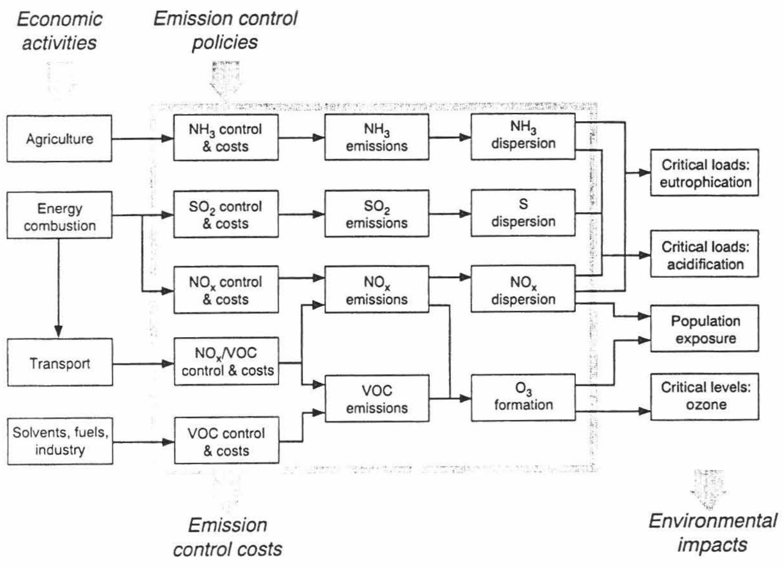

Fig. 1. The structure of the RAINS model of acidification and tropospheric ozone.

grids in Europe, and accumulated excess PWL (piece-wise linear functions) are defined for each grid and for each type of acidification and eutrophication excess.

The atmospheric dispersion processes over Europe for NH_3, SO_x, NO_x and VOC compounds are modeled using results of the European EMEP model, developed at the Norwegian Meteorological Institute and described e.g. by Olendrzyński et al. (2000). However, the EMEP model is far too complex to be used for optimization, or even for many scenario analyses. Therefore, an essential requirement of an integrated assessment of the RAINS model is a simplified but reliable description of the dispersion processes in order to represent the source–receptor relationships involved. It is possible to envisage several ways of condensing the results of more complex models to achieve this. One approach is to use statistical techniques to build a simplified model based on the results obtained from a complex mathematical model for a large number of emission reduction scenarios. Such an approach has been implemented for, and is currently used by, the RAINS model (cf. Eq. 10, p. 232). Another approach to the specification of a simplified ozone model is based on using fuzzy-rules generation methodology and is presented by Ryoke et al. (2000). Of course, using simplified source–receptor relationships between the precursor emissions and the various thresholds of corresponding levels/loads results in a lesser accuracy than that assured by the EMEP photo-oxidants model. Therefore, selected results obtained from the simplified model are compared with results from the EMEP model. This is done by running the EMEP model for the emissions obtained from the RAINS model, and comparing the levels/loads values provided by both models.

The outline presented above illustrates the challenges created by the problem. The corresponding model is complex, large and non-linear. Its implementation triggers a number of methodological and technical issues related to the specification and analysis of such models. In particular:

- In order to adequately meet the requirements for model analysis, a problem-specific generator has been developed and coupled with three non-linear solvers. An object-oriented programming approach to model generation and analysis has been applied.

- The generation of the model requires processing a large amount of data coming from various sources. For efficient and portable handling of data, the public domain library HDF, developed by the National Center for Supercomputing Applications, Illinois, USA (URL: `http://hdf.ncsa.uiuc.edu/HDF5`), has been applied.
- A representation of environmental targets by hard constraints would result in the recommendation of expensive solutions, hence soft constraints (with compensation for the violation of original targets) are specified.
- The resulting optimization problem typically has non-unique solutions. More exactly, it has many very different solutions with almost the same value of the original goal function. These correspond to various instances of the mathematical programming problem that differ very little. Therefore, a technique called regularization was applied to provide a suboptimal solution which has additional properties specified by a user.
- A minimization of costs related to measures needed for improvement of air quality is a main goal; however, other objectives – such as robustness of a solution, trade-offs between costs and violations of environmental standards – are also important. Therefore, a multi-criteria model analysis has been applied to this case study.
- Some instances of the model contain over 25,000 variables and constraints, therefore its preprocessing is essential for a substantial reduction of computation time. Section 6.4 shows how much one can gain by a proper reformulation and preprocessing of a large non-linear model.

These issues are related to various modeling paradigms, and therefore are of a broader interest to OR practitioners.

The main message of this paper is to stress the often forgotten fact that no single modeling paradigm can be successfully used to analyze a complex problem, especially if the results of such an analysis are used to support various elements of real decision-making processes. There is a number of rules that have to be observed during the specification of a model in order to provide useful results. Also, various techniques of model analysis should be used instead of just the classical approaches which are focused and driven either by simulation or optimization paradigms.

3. Model-based decision support

The problem outlined in Section 2 illustrates a situation where making a rational decision requires access to, and the processing of, a large amounts of data and logical relations that cannot be replaced by intuition. Moreover, it is also practically impossible to examine even the possible range of all feasible alternatives. Therefore, in such situations, one develops a mathematical model that can adequately represent the decision problem and a problem specific software, conventionally called a Decision Support System (DSS).

Wright et al. (1998) provide a survey of more than 200 modeling practitioners on the use of various types of models in US organizations. Among the four most desirable requirements, two are related to data processing (automated access to the model data, and automated error and consistency checking), while the other two (model integration and model formulation) are concerned with models. This justifies the need to develop methods and tools for model generation and analysis. However, in order to analyze rational approaches to model development and analysis, it is worth considering some fundamental issues pertaining to the relationship between the needs of decision-making support and the opportunities that can be offered by OR methods and tools.

In this paper, we will consider a model-based DSS, which uses an underlying mathematical model. Such a mathematical model is built for a part of the Decision-Making Process (DMP), where it is possible to implement a mathematical model that is good enough to represent the available (though often quite complex) knowledge and experience of a user in order to support his/her intuition. A user is a Decision Maker (DM), whether an individual analyst or a group of experts that provides advises. In this paper, we will use the terms *user* and DM interchangeably.

Model-based decision-making support is conceptually distinct from the more traditional data-oriented perspective of decision support. We do not claim that the approach discussed in this paper is better than this more traditional approach, rather we simply point out that quite often the DMP requires not only data processing in the traditional sense, but also the analysis of a large number of logical or analytical relations and the processing of – in the sense of solving an underlying mathematical model – large amounts of data. The word *solving* is used here for various approaches to the analysis of mathematical models. In such situations, a properly designed and implemented model-based DSS not only performs cumbersome data processing, but it also provides relevant information that enables a DM to concentrate on those parts of the DMP that cannot be formalized.

3.1. General concepts

The following assumptions are typically adopted for a model-based DSS:

- A well-defined part of a DMP (for which a DSS is to be implemented) can be represented in the form of a mathematical model. Decisions have quantitative characters and therefore can be represented by a set of model variables, hereafter referred to as decisions – for the sake of brevity, we often call decision variables simply decisions – $x \in E_x$, where E_x denotes a space of decisions.
- The model defines a set of feasible decisions $X_0 \subseteq E_x$. Therefore x is feasible, iff $x \in X_0$. The set X_0 is usually defined implicitly by a specification of a set of constraints that corresponds to logical and physical relations between the variables. The feasibility of decisions given by a DM should be assessed. Decisions computed by a DSS should be feasible, if a feasible solution exists.
- The model can be used for predicting the consequences of decisions proposed by a DM or computed by a DSS. The prediction of the consequences is represented by a mapping $y = f(x) \in E_y$, where E_y is a space of consequences (outcomes) of the decisions.
- The consequences of different decisions x are evaluated by values of criteria $q \in E_q$, where E_q is a space of criteria (sometimes referred to as outcomes, goals, objectives, performance indices, attributes, etc.). Usually E_q is a subspace of E_y, that is, the DM might select some criteria q_i between various outcomes y_j. Sometimes some of the decision variables x are also used as criteria. A partial preordering in E_q is usually implied by the decision problem and has obvious interpretations, such as the minimization of costs competing with the minimization of pollution. However, a complete preordering in E_q usually cannot be given within the context of a mathematical model.

The essence of decision-making support is to help a DM to select *the best* decision among all feasible decisions. Such a decision is typically represented by a vector of variables, and is denoted further on by $\hat{x}$. In the case of the `RAINS` model, it is a set of emission levels for each type of pollutant and for each country, optionally it also includes the accepted violations of certain environmental targets, see Section 5.2 for details. Therefore, the key problem for an adequate formulation and analysis of a model aimed to support DMP is to understand what *the best* means to the DM who actually makes a decision. The problem of a *rational choice* of a decision has been extensively discussed in a number of publications, and even a brief summary of this discussion is beyond the scope of this paper. A discussion of different approaches to this problem can be found e.g. in Wierzbicki et al. (1999)

3.2. Different views on DSS

This section concentrates on the different views related to model analysis. This problem was first raised by Ackoff (1979), who stressed that many DSS are driven by optimization techniques, which means that the user has only partial control of the way in which analysis of the model is done.

From the (traditional) OR perspective, it is natural to formulate a mathematical programming problem in the form

$$\hat{x} = \arg\min_{x \in X_0} \mathscr{F}(x) \tag{1}$$

and solve it. A concise formulation of (1) may be misleading for those who do not know that, frequently, solving a mathematical programming problem is a challenging task. One should be aware of both the scientific values and the resources required to find and implement an algorithm that can provide a (correct!) solution $\hat{x}$ from a set X_0 that minimizes the objective $\mathscr{F}(x)$, and uses possibly small computer resources.

However, a DM has a completely different perspective. Let us briefly summarize some elements of this perspective (typical for non-engineering applications of DSS) that differs from the traditional OR way of formulating and solving a mathematical programming problem:

- A unique specification of both a mathematical model and of one criterion that adequately represents a preferential structure of a DM is very difficult, if at all possible, for most real-life situations. Therefore, a series of cycles composed of an analysis of the results provided by a solution of the model and a modification of the preferential structure of the DM is the most typical desired activity of the analysis of any complex problem.
- Models are simplifications of reality, and optimization is limited to models that include an objective that is always a simplification of a preferential structure of the DM. Therefore, the optimal (for a given representation of a preferential structure) solution of a model may not necessarily be optimal (i.e. *the best*) in reality, as perceived by a DM. It may be desirable to modify an "optimal" solution, in order to take into account some factors not yet accounted for in the underlying model (very often some are deliberately not included). Moreover, a DMP is typically composed of subproblems analyzed/solved independently; therefore, the overall optimum is usually not composed of optima computed separately for each sub-problem.
- DSS should support various ways of learning about the problem, in particular an examination of the consequences of the implementation of given values of decision variables. This should include the possibility of fixing the values of variables and/or goals, modification of a set of goals (both treating goals as soft constraints and vice versa, as well as changing the definitions of goals), and looking for a suboptimal solution with certain additional properties. Fixing values of decision variables should not be implemented as constraints; instead, the *regularization* or the so-called *inverse simulation* techniques should be used. Moreover, often selected constraints should be optionally treated as so-called *soft* constraints and their violations should be considered as one of the criteria.
- DM usually prefers to be sure that his sovereignty in making decisions (for which he is responsible) is not jeopardized by a computer. The main reason is psychological. For example, it is a commonly known fact that even the developers of DSSs supporting choice (out of a given set) of an alternative do not necessarily follow optimal solutions suggested by their own DSS when solving a personal problem. However, they do use the DSS to analyze of the problem. Therefore it is important that a DM – who rarely is also a computer guru nor does he/she want to devote a substantial amount of time digging into hundreds of pages of software documentation – clearly understands all assumptions made for the model specification and important functions of the DSS.

In addition, it is clear that optimization in DSS should have quite a different role than the function of optimization in some engineering applications (especially real-time control problems) or in very early implementations of OR for solving well-structured military problems. This point has been clearly made by Ackoff (1979).

Optimization would be better accepted outside the OR community if users would be able to treat optimization as a tool for selecting a number of solutions that have certain properties, and if support for comparing such solutions from various perspectives preferred by a DM would be widely available. This is, however, contrary to the traditional way of using OR methods, as characterized by Chapman (1992), for solving a problem in the following five stages: describe the problem, formulate a model of the problem, solve the model, test the solution, implement the solution. It is also contrary to another traditional OR perspective that implies that an optimal solution is also the best available solution, which in turn implies that there is neither much room nor need for human decision-making. Also, text books on DSS addressed to managers, see e.g. Emery (1987), often treat optimization merely as a tool for providing *the* solution. To make the situation worse, many of these books still present only single-objective optimization, whereas multi-criteria model analysis, when properly used, remarkably softens this perspective.

One should point out another danger related to generate *the best* solution through a formulation of an optimization problem. It is obvious, that one can make (by adding appropriate constraints) any feasible solution to be optimal for a given objective function. Therefore, a common (mis)use of optimization is to generate a sequence of problems based on analysis of previous solutions and to sequentially add constraints that try to reflect dissatisfaction of a user who analyzes presented solutions in a corresponding sequence. In this way the set of feasible solutions of original model is decreased by introducing constraints that correspond to the preferential model of the user. However, such a procedure applied to a complex problem is likely to leave many rational solutions beyond analysis.

4. Modeling paradigms

The organization of this section is in response to Dolk (2000), who spells out an important observation about the paradigm-centric nature of MS/OR community: *Practitioners in the MS/OR modeling world rarely venture outside the particular paradigm in which they were trained. Thus, once an optimization person, always an optimization person. ... The value of seeing beyond one's own modeling discipline to problems of "modeling in the large" does not yet seem to be strongly embraced.* This section tries to link the modeling paradigms that pertain to model specification with two – often perceived as competing and exclusive – paradigms for model analysis, namely simulation and optimization.

From both methodological and practical points of view, it is rational to discuss and implement a model-based DSS in two stages:

- First, develop a core model that implicitly defines a set of feasible solutions X_0. The core model should include all logical and physical relations between variables but should not contain any constraints corresponding to the preferential structure of the user.
- Second, provide a methodology and tools for the analysis of the model. This can be done in different ways, e.g. by simulation, by some extensions of single-criterion optimization, or by multi-criteria model analysis.

These two stages are discussed in the following subsections.

4.1. Model specification

When a model-based DSS is desired, it can only be achieved for a problem that is understood well enough to build a mathematical model that can adequately represent a decision situation. A decision situation is adequately represented when a corresponding core model can be used for predicting and evaluating the consequences of decisions. A more detailed discussion about the structure of a core model can be found in Wierzbicki and Makowski (1992). Here, we will only outline the structure and basic requirements for a core model that implicitly defines a set of feasible solutions X_0 (cf. Section 3.1). Such a model is typically composed of:

- Decision variables that represent actual decisions (alternatives, choices, options, etc.).
- Outcome variables (often called goal functions or performance indices) that can be used for assessing various elements, such as costs and air quality indices, to determine the quality of a solution (i.e. consequences of implementation of given or computed values of decision variables).
- Various intermediate and parametric variables, such as balance and/or state variables, resources, external decisions, i.e. those not directly controlled by the DM.
- Constraining relations between variables (inequalities, equations, etc.) that indirectly

determine the set of feasible decisions. Some of the constraints may reflect the logic of handling events represented by variables (e.g. assuring that exactly one technology is selected for each installation).

Building a mathematical model is a complex task that requires both a good understanding of the problem and an in-depth knowledge of model-building methodology. Moreover, the specification of the model to be used in a model-based DSS should also meet additional requirements. Such requirements are discussed in more details by Makowski (1994b). Here we only summarize the key elements of an appropriate model specification.

A core model implicitly defines a set X_0 that contains all feasible solutions x. Clearly most of the $x \in X_0$ are not rational, therefore, only a small part of X_0 is subject to various analyses that eventually lead to a selection of a small set of "interesting" solutions. However, this part is typically composed of a continuum of *very* different solutions. The key point of an appropriate model specification and analysis is to make it possible to analyze all "interesting" solutions. Whether a solution is "interesting" or not, depends on the preferential structure of the user. However, many practical examples clearly show that such a structure changes substantially during the learning phase of model analysis (therefore, its adequate representation in the core model is hardly possible). This is the main reason why a preferential structure of a user should not be included in the core model, because including acceptability conditions or a preferential structure into a definition of X_0 often results in implicit rejection of a large number of feasible solutions. Such a narrowing of X_0 will mislead the user, because in such a case s/he cannot evaluate all feasible solutions. Therefore, the specification of a model that defines X_0 should not include any relations that represent a preferential structure of a DM.

A core model specification should provide the possibility to treat selected sets of constraints as so-called soft constraints. In order to illustrate this point, let us consider the following trade-off in the RAINS model, a model which helps identify and evaluate different policies of air pollution reductions and the related costs, as well as the corresponding quality indices. It is obvious that a specification of constraints for air quality indices that represents the strict environmental standards for all grids leads to solutions that are too expensive. Moreover, setting values of constraints too tightly would result in restricting the analysis of the problem to a small part of feasible solutions, often resulting in an empty set of feasible solutions. The typical advice in such situations is to specify two types of constraints, so-called hard and soft constraints which correspond to *must* and *should* types of conditions, respectively. For *soft* constraints, any violation of the original constraints is in fact controlled, and helps to determine trade-offs between costs and environment quality, as demonstrated by the RAINS core model formulation presented in Section 5.4.2. In fact, dealing with soft constraints can easily be done within multi-criteria model analysis, as described e.g. by Makowski (1994b).

The specification and parameters of the core model must not be changed after a verification and validation of the model is done. Therefore, each instance of the mathematical programming problem generated and solved during the model analysis is conceptually composed of two parts (see Section 7 for details):

- A large, constant core model. This part is developed and verified before an actual analysis of the problem begins.
- A part that corresponds to the current specification of goals and conditions set by the user. This specification of the preferential structure of the user is changed, often drastically, for each scenario.

A proper implementation of such an approach makes it possible for the DM to analyze feasible solutions that correspond best to various specifications of his/her preferential structure. Changing this structure is the essence of model analysis and of model-based decision support. An additional bonus comes from the fact that there always exists a feasible solution of the underlying mathematical programming problem, which is a prerequisite for an analysis of complex models.

Finally, we should point out that building a model requires collection, processing and verification of data. The famous saying "*garbage in, garbage out*" implies that the problem of data accuracy should not be overlooked. Fortunately, the recently emerging data warehouse technology, see e.g. Dolk (2000), provides much better tools for properly handling data than the well-established DBMS. The user need not worry about the possible range of quantities (which usually have an impact on computational problems) because this should be accounted for by the DSS. However, a designer of a DSS should make sure that only the substantial elements are included in the model, and that all such elements are included. A more detailed discussion on specification of a core model can be found e.g. in Makowski (1994b).

The value of a mathematical model as a decision aid comes from its ability to adequately represent reality. Therefore, there is always a trade-off between the requested accuracy (realism) of the model and the costs (and time) required to develop it, along with providing the model with the data, and resources needed for its analysis. Hence, the requested accuracy of the model should be consistent with the accuracy actually needed for the model analysis and with the quality of the available data. This point is well illustrated by the specification of the RAINS model, which includes simplified dispersions and ozone formation submodels (see Section 2). One should notice, however, that these simplified submodels are based on the corresponding detailed models, which are used for checking the accuracy of results obtained from the RAINS model (because of the resources needed for running the detailed models, such a comparison can only be made for a limited number of selected scenarios; however, this is not a practical limitation for actual application of the simplified submodels).

4.2. Model analysis

One typically distinguishes two types of model-analysis methods, which are conventionally called simulation and optimization. The simulation and optimization methods can be characterized as follows:

- In simulation, decision variables are inputs and goals are outcomes. Therefore this technique is good for exploring the intuition of a DM, not only for verification of the model, but also for providing a DM with information about the consequences – typically represented by values of goals and constraints – of applying certain decisions. One can also consider simulation as an alternative-focused method of analysis that is oriented towards examining given alternatives.
- Optimization can be considered as a goal-oriented (value-focused) approach that is directed towards creating alternatives. Optimization is driven by formulating a single objective in single-criterion optimization, or several objectives in multi-criteria optimization, and looking for values of decision variables that optimize the value of the specified objective(s). Therefore, goals are the driving force and the values of decision variables are the outcomes.

Traditional approaches to model analysis have been based either on simulation or on classical formulations of single-criterion optimization. A summary of these approaches and their limitations is helpful for understanding the advantages of modern decision-support methods which extend and combine these approaches. Therefore, the remaining part of this section is devoted to summarizing the following topics (which are discussed in the following order):

- Relations between descriptive and prescriptive decision support methods and simulation- and optimization-based model analysis methods.
- Basic limitations of model analysis methods based on single-criterion optimization.
- Two types of extensions of classical single-criterion optimization approaches that are helpful in model-based decision support.
- The combined use of simulation and optimization for model analysis.

4.2.1. Descriptive and prescriptive decision support

There is a general agreement, see e.g. Simon (1990), that model-based DSS are generally of two types that correspond to the two ways to analyze a model:

- Descriptive (sometimes called predictive), which are used for prediction of the modeled system's behavior without an attempt to determine desired values of control variables (which in managerial situations are called decisions). For the descriptive type of DSS, the values of decision (or control) variables are defined by the user. The expected consequences of implementing such decisions are evaluated by selected variables, conventionally called outcome variables (or performance indices or objectives or criteria).

- Prescriptive (normative), which are aimed at identifying the values of decision variables that can result in the desired behavior of the modeled system. This desired behavior is usually evaluated with the help of goals (objective values, performance indices, etc.). For the prescriptive type of DSS, optimization techniques are widely considered to be good tools for selecting a solution from an admissible set that is considered *the best*. The term *best* corresponds to the solution that provides the best value of a performance index (goal function, objective, criterion).

In other words, a descriptive DSS helps to answer questions such as "*what would happen if*", whereas a prescriptive DSS supports answers for questions like "*what decisions are likely to be the best*".

A given mathematical model can be considered the kernel of a model-based DSS that is either descriptive or prescriptive, depending on the way the model is analyzed. Obviously, it is desirable to analyze a model in both (i.e. descriptive and prescriptive) ways interchangeably. For example, before even trying to find prescriptions, one should verify the model in the descriptive mode. The model should not only conform to the formal specification, but also all discrepancies must be resolved between the intuitive judgment of the DM and the analytic results obtained from the model. Such inconsistencies show that either the model (the assumptions, specification, data) or the DM's intuition is wrong. Any conflicts between results provided by the model and what is perceived as correct by the DM must be resolved before the DM will trust the model, which is obviously a necessary but often neglected condition for the actual use of a DSS.

4.2.2. Limitations of single-criterion optimization for a model analysis

Model analysis methods based on single-criterion optimization are very appealing because they are simple and have been widely taught, typically using examples that are both simple and well-structured. However, these methods have serious limitations when applied to the analysis of complex problems. For such problems, in a typical decision situation, it is necessary to evaluate the consequences of a decision by more than one criterion. There are two main classical approaches to applying a single-criterion optimization paradigm to the analysis of models that represent multi-criteria decision-making problems:

- Since the classical formulation of optimization problems allows for dealing with only one criterion (or goal function), one objective is selected as the goal function, while the other objectives are converted into constraints whose values are treated as parameters. Although parametric optimization and sensitivity analysis are sound ideas for OR-oriented users, in practical applications – especially those dealing with large complex models – they are hardly applicable.
- All criteria are aggregated into one goal function which is composed of a weighted sum of criteria. This quite popular approach has serious but seldom recognized deficiencies, which are discussed in details e.g. in Wierzbicki et al. (1999).

This brief summary shows that the specification of a single-objective function that adequately reflects the preferences of a model user will remain the major unresolved difficulty in formulating many practical problems into a relevant single-criterion optimization problem. A more detailed discussion of various extensions of the traditional single-criteria optimization useful for model analysis (still within the framework of single-criterion optimization) is presented by Makowski (1994a). Fortunately, multi-criteria model analysis approaches offer attractive alternatives, particularly if they allow for an interactive redefinition of user preferences. The availability of modular tools, such as ISAAP by Granat and Makowski (2000), makes the implementation of such an analysis much easier.

4.2.3. Regularization and soft simulation

A large optimization problem typically has a non-unique optimal solution. Although this is theoretically rare, in practice, many problems actually have a large set of widely varying solutions for which the objective values differ very little, see e.g. Makowski and Sosnowski (1989). In most cases, the optimization algorithm stops when a current solution is recognized as optimal for a given set of tolerances. For problems with a non-unique optimum, the first optimal solution found is accepted, so one may not even be aware of the non-uniqueness of the solution reported as optimal. Thus, in practical applications, we are faced with the problem of choosing an optimal (or, in most cases, to be more accurate, a suboptimal) solution that possesses certain additional properties desired by the user.

In the RAINS model, the stabilization of a solution is an important feature of the model, which is typical for many real-world problems. Namely, for problems that have (practically) non-unique optimal solutions, a small perturbation of parameters results typically in a small change in the value of the goal function (or criteria for multi-criteria model analysis). However, the new solution also tends to differ substantially from that of the non-perturbed problem. The practical implications of this are discussed in Section 7.

This problem may be overcome by applying an approach called *regularization*. We apply Tikhonov's type regularization which provides the (sub)optimal solution having either a minimum norm or a minimum distance from a given reference point. This can be achieved by adding to a minimized goal function the term

$$\varepsilon\|z - \bar{z}\|, \tag{2}$$

where ε is a small positive number, z a vector composed of a subset of variables and $\bar{z}$ a vector composed of the corresponding desired values of these variables. The choice of $\bar{z}$ depends on the desired properties of the solution. If one knows the desired values of decision variables, then those values should be used for defining the vector $\bar{z}$. If such values are not know, then one can set $\bar{z} = 0$, which implies a preference for the minimum norm solution.

One should note that the term (2) with larger values of the parameter ε can be used for various simulation techniques. For example, by using a large value of ε (i.e. one that dominates the other terms of the goal function) and setting $\bar{z}$ equal to desired values of decision variables, one can find a solution that is closest to such values. If these values are feasible, then a solution composed of these values will be found. If they are not feasible, than the closest feasible solution will be found. Note, that in the latter case, a traditional simulation will simply report "*infeasible problem*". Finally, one should point out that for such an approach to soft simulation the original goal function takes the role of the regularizing term.

4.2.4. Simulation and optimization for model analysis

Simulation- and optimization-based approaches, as defined in Section 4.2, are in fact complementary. For simulation, one needs to provide values for all decision variables. For this purpose, one may use random values for variables (as proposed by Goodwin and Wright (1991), who present various techniques and examples), or assign values based either on the DMs intuition or on a heuristic (possibly based on information from a knowledge base). For models having hundreds or even more variables, a specification of values for all decision variables based on intuition is practically unrealistic. However, even for a large model, simulation can be useful for *what if* type of analysis, e.g. for comparing the results from optimizations with the outcomes from values of decision variables set by the user. Of course, there is no way to assure that a given specification of the values of decision variables will result in a feasible solution.

For a single-criterion optimization, one has to specify a goal function, and the values of the decision variables that optimize this function are computed. While the simplicity of this approach is appealing, it has also limitations that have been summarized in Section 4.2.2. Actually, the development of multi-criteria model analysis (MCMA) methods have been motivated by the needs of model analyses that can neither be met by simulation nor by traditional optimization paradigms. A more detailed discussion of MCMA methods is beyond the scope of this paper, but their methodological background and various applications can be found e.g. in Wierzbicki et al. (1999). Here, we will only outline one of MCMA methods that has been applied to the RAINS model analysis described in Section 7.

So-called softly constrained inverse simulation, described e.g. in Wierzbicki et al. (1999), is a very useful technique for examining the trade-offs between the optimized criteria and the desire for having values of (possibly only a subset of) decision variables close to the given, corresponding values. This can be implemented by applying intermediate values (i.e. one that is neither small nor large) of the parameter ε in (2). Obviously, a mixture of both classical and inverse simulation techniques can be used for two sets of variables (i.e. variables whose values are fixed/simulated and those that are subject to further optimization). Note that the concept of stabilized criteria implemented in ISAAP (Granat and Makowski, 2000) provides an easy way for the soft fixing of a variable and it is conceptually close to the regularization mechanism implemented with a large regularizing coefficient. Finally, one should point out that various simulation techniques applied in the descriptive mode may provide information not only for model verification, but also may lead the DM to modify the selected constraints or goals.

The arguments summarized above show that simulation and optimization are complementary paradigms for model analysis. Therefore, model-based decision support should take advantages of a complementary usage of both methods.

5. RAINS core model definition

This section provides a specification of the RAINS model. Due to space limitations this specification has been simplified and is, therefore, not complete. A complete specification of RAINS can be found at `www.iiasa.ac.at/~marek/pubs/prepub.html`.

We should distinguish first between a set I of sources of various types of air pollution, and a set J of areas for which various air quality indicators are assessed. Conventionally, the names *emitter* and *receptor* are used for the elements of such sets.

5.1. Notation

The model definition requires the use of the following indices:

- Index $i \in I$ corresponds to the emitters. The numbers of elements in I corresponds to a number of countries (about 40).
- Index $is \in S_i$ corresponds to a sector that emits either NO_x or VOC or a linear combination of them; S_i is a set of sectors in the i-th country. Sets S_i may have up to five elements.
- Index $j \in J$ corresponds to the receptors. The number of elements in J corresponds to the number of grids (about 600).
- Index $l \in L$ corresponds to a combination of ozone thresholds and a given year. The set L may have up to six elements.
- Index $m \in M$ corresponds to a set of receptors for which the balancing of violations and surpluses of targets is defined.

5.1.1. Emission sectors

Emissions are analyzed for sets of emitters located in a certain area, which is typically a country. However, sets of NO_x and VOC emitters are further subdivided into subsets, called *sectors*, in order to account for measures that can be applied to the emitters that belong to a particular sector. The emitters that belong to a particular sector emit either NO_x or VOC, or a linear combination of them. In the latter case, the relation between the amount of VOC emission and the corresponding emission of NO_x is defined by

$$v_{is} = \lambda_{is} n_{is} + \mu_{is}, \tag{3}$$

where the parameters λ_{is} and μ_{is} are given.

5.2. Decision variables

The main decision variables are the annual emissions of the following four types of primary air pollutants from either sectors or countries:

- n_{is} – annual emission of NO_x (nitrogen oxides),
- v_{is} – annual emission of VOCs (Volatile Organic Compounds),
- a_i – annual emission of NH_3 (ammonia),
- s_i – annual emission of SO_2 (sulphur dioxide).

Additionally, optional decision variables are considered for scenarios which allow for limited violations of air quality targets. For such scenarios variables corresponding to each type of the considered air quality targets are defined for each receptor. Each variable represents a violation of a given environmental standard. Optionally, violations of targets can be balanced with surpluses (understood as the difference between a target and a corresponding actual concentration/deposition). For efficiency reasons, one variable is used both for violations of targets and for surpluses (positive values represent violations while negative values correspond to the part of a surplus that is used to balance violations of targets).

Therefore, the following decision variables are optionally defined:

- y_{lj} – violation of ozone exposure targets (surplus if $y_{lj} < 0$),
- ya_j – violation of acidification targets (surplus, if $ya_j < 0$),
- ye_j – violation of eutrophication targets (surplus, if $ye_j < 0$).

5.3. Outcome variables

The consequences of the application of computed values of decision variables are evaluated by the values of the outcome variables. However, several auxiliary variables needed for the definitions of outcome variables have to be specified first.

5.3.1. Auxiliary variables

n_i – the annual emission of NO_x (nitrogen oxides) defined by

$$n_i = \sum_{is \in S_i} n_{is}. \tag{4}$$

v_i – the annual emission of VOCs (Volatile Organic Compounds) defined by

$$v_i = \sum_{is \in S_i} v_{is}. \tag{5}$$

en_{lj} – the mean effective emissions of NO_x experienced at *j*th receptor is given by

$$en_{lj} = \sum_{i \in I} e_{lij} n_i + enn_{lj}, \tag{6}$$

where enn_{lj} are given effective natural emissions of NO_x.

nlv_{lj} – the representation of another non-linear term defining ozone exposure at *j*th receptor is defined by

$$nlv_{lj} = \sum_{i \in I} d_{lij} v_i. \tag{7}$$

5.3.2. Definition of outcome variables

One outcome variable represents the sum of the costs of reducing emissions; four sets of other outcome variables correspond to various indices of air quality.

Annual costs related to the reduction of a corresponding emission to a certain level are given by a convex piece-wise linear (PWL) function for each type of emission and for each emitter. Formally, the following PWL functions define the annual costs related to reducing the level of the emission to a level given by argument(s) of the functions: $ca_i(a_i)$ for a_i, $cs_i(s_i)$ for s_i, $c_i(n_i, v_i)$ for n_i and v_i. The term $c_i(n_i, v_i)$ is defined by

$$c_i(n_i, v_i) = \sum_{s \in S_i} c_s(\cdot), \tag{8}$$

where $c_s(\cdot)$ is a cost function for NO_x or for VOC or for a joint reduction of NO_x and VOC.

For the sake of brevity, the sum of costs is defined by

$$\text{cost} = \sum_{i \in I} (ca_i(a_i) + cs_i(s_i) + c_i(n_i, v_i)). \tag{9}$$

Such a function is continuous and convex but not smooth. Therefore, it has to be represented by another function that meets typical requirements of non-linear solvers. Such a modification is outlined in Section 6.3.

For each receptor, the following $(l + 3)$ outcome variables correspond to various indices of air quality:

- aot_{lj} – the long term ozone exposure of l-th type:

$$aot_{lj} = \sum_{i\in I}(a_{lij}v_i + b_{lij}n_i + \gamma_{lij}n_i^2) + \alpha_{lj}en_{lj}^2 + \beta_{lj}en_{lj}nlv_{lj} + k_{lj}, \tag{10}$$

- $ac1_j$ – acidification of type 1:

$$ac1_j = tns_j\left(\sum_{i\in I} tn_{ij}n_i + \sum_{i\in I} ta_{ij}a_i + kn_j\right) + \sum_{i\in I} ts_{ij}s_i + ks_j, \tag{11}$$

- $ac2_j$ – acidification of type 2:

$$ac2_j = \sum_{i\in I} tn_{ij}n_i + \sum_{i\in I} ta_{ij}a_i + tss_j\left(\sum_{i\in I} ts_{ij}s_i + ks_j\right) + kn_j, \tag{12}$$

- eu_j – eutrophication:

$$eu_j = \sum_{i\in I} tn_{ij}n_i + \sum_{i\in I} ta_{ij}a_i + kn_j, \tag{13}$$

where tn_{ij}, ta_{ij}, ts_{ij} are transfer coefficients for NO_x, NH_3 and SO_2, respectively; kn_j and ks_j are constants for nitrogen and sulphur background depositions; tns_{ij}, tss_{ij} are scaling coefficients that convert acidification coefficients of one type into acidification coefficients of another type, for NO_x and NH_3, and for SO_2, respectively. The methodological background for modeling ozone exposure, and loads of acidification and eutrophication is summarized by Amann and Makowski (1999).

Environmental effects caused by the two types of acidification and by eutrophication are evaluated at each receptor by a Piece-Wise Linear function (PWL), which represents an accumulated excess of each type of the air quality index:

- $aac1_j$ – accumulated excess of acidification of type 1:

$$aac1_j = \mathrm{PWL}_j^{ac1}(ac1_j), \tag{14}$$

- $aac2_j$ – accumulated excess of acidification of type 2:

$$aac2_j = \mathrm{PWL}_j^{ac2}(ac2_j), \tag{15}$$

- aeu_j – accumulated excess of eutrophication:

$$aeu_j = \mathrm{PWL}_j^{eu}(eu_j). \tag{16}$$

5.4. Constraints

5.4.1. Bounds

Each of the decision variables declared in Section 5.2 for $i \in I$ or for $is \in S_i$ is implicitly bounded by a corresponding definition of the domain of the corresponding cost function, which represents costs associated with the reduction of the emission, as outlined in Section 5.3.2. This domain may be restricted by specifying the optional bounds.

Violations of targets are constrained at each receptor by corresponding lower and upper limits specified for each target type and for each grid:

$$y_{lj}^{\min} \leqslant y_{lj} \leqslant y_{lj}^{\max}, \tag{17}$$

$$ya_j^{\min} \leqslant ya_j \leqslant ya_j^{\max}, \tag{18}$$

$$ye_j^{\min} \leqslant ye_j \leqslant ye_j^{\max}. \tag{19}$$

5.4.2. Complex constraints

The accumulative excess of long-term ozone exposure of l-th type is constrained at each receptor by

$$aot_{lj} - y_{lj} \leqslant aot_{lj}^{\max}, \tag{20}$$

where aot_{lj} is defined by (10) and $aot_{lj}^{\max}$ is a given maximum ozone exposure for the l-th threshold at the j-th receptor.

Constraint (20) without the term $-y_{lj}$ represents a so-called *hard constraint* for accumulated excess of ozone exposure. Such a formulation is typically used in a traditional formulation of optimization problems. It can also be used in the presented model by selecting an option that does not allow for the generation of variables y_{lj}. However, an assumption of the hard constraints for air quality targets results in forcing more expensive solutions that are caused by constraints active in only one or two receptors. The introduction of the term $-y_{lj}$ converts a hard constraint into a so-called *soft constraint*. This allows a violation of the target air quality. Such a violation is:

- constrained by upper bounds on variables y_{lj};
- compensated by surpluses (i.e. differences between actual exposure and the corresponding target) in other receptors belonging to the same set of receptors (e.g. located in the same country or region);
- controllable by a trade-off between the violation of targets and corresponding costs of reducing emissions.

Constraints for the accumulated excess of the two types of acidification and of eutrophication are defined in a similar way:

$$aac1_j - ya_j \leqslant aac_j^{\max}, \tag{21}$$

$$aac2_j - ya_j \leqslant aac_j^{\max}, \tag{22}$$

$$aeu_j - ye_j \leqslant aeu_j^{\max}. \tag{23}$$

Optionally, violations of targets can be balanced with surpluses of targets over sets of receptors denoted in the following constraints by J_m, where $m \in M$ is the index of a set of receptors. Obviously, lower bounds in conditions (17)–(19) have to be negative in such a case. The balances are represented by the following constraints:

$$\sum_{j \in J_m} wo_{lmj} y_{lj} \leqslant tbo_{lm}, \quad l = 0, \tag{24}$$

$$\sum_{l=1}^{L} \sum_{j \in J_m} wo_{lmj} y_{lj} \leqslant \sum_{l=1}^{L} tbo_{lm}, \tag{25}$$

$$\sum_{j \in J_m} wa_{mj} ya_j \leqslant tba_m, \tag{26}$$

$$\sum_{j \in J_m} we_{mj} ye_j \leqslant tbe_m, \tag{27}$$

where $wo_{lmj}, wa_{mj}, we_{mj}$ are given weighting coefficients, J_m, $m \in M$, are sets of receptors, and tbo_{lm}, tba_m, tbe_m are the target balances for the m-th set of receptors for the l-th type of ozone exposure, for acidification, and for eutrophication, respectively. The sets J_m are defined implicitly by non-zero elements of sparse matrices wo_l, wa and we, respectively.

6. Model generation and preprocessing

The user of a model is typically not interested in issues related to the generation, preprocessing or solving of the model. However, OR specialists, who design and implement large and complex models, will certainly agree that the generation and management of the `RAINS` model is a challenging task from the operations research point of view. Therefore, several methodological and technical issues of a broader interest to the OR community are discussed in the subsequent subsections.

6.1. Generation and solution of the model

There are basically two approaches to the generation and analysis of a mathematical programming problem. These either develop a problem-specific generator or use a modeling system (such as GAMS, AMPL, AIMMS). The following issues should be considered when selecting one of these approaches:

- A modeling system greatly simplifies the task of model specification, especially if compared with the amount of resources needed for the development of a model generator using traditional procedural programming languages like Fortran or C. However, the use of C++ substantially reduces this difference, especially with the Standard Template Library (recently included in the C++ standard), and with other class libraries supporting implementations of mathematical programming type of problems.
- A model generator is more efficient in processing the input data needed for model specification. It is also preferred when a more sophisticated check of data consistency is desired.
- A modeling system has limited possibilities for efficient preprocessing of optimization problems. This is not a serious problem for linear models because preprocessing is a standard

feature of any good LP solver. However, the preprocessing of non-linear models is much more difficult, as demonstrated e.g. by Drud (1997). A model generator can generate a non-linear problem that is much easier to solve than a model generated by a modeling system. Therefore, it might be a better choice for large non-linear problems, where solution time becomes an important issue.

- For a large problem, a good starting point might dramatically decrease the computation time. A computation of such a point is much easier for a problem-specific generator.
- A modeling system typically does not exploit the possibilities of an optimization algorithm used by a solver. This is not an important issue as long as the optimization problem is of a moderate size, but for large problems the exploitation of the problem structure may be necessary for reducing optimization time. In such cases, a modeling system must be augmented by a software tool, e.g. by Fragniere et al. (1997), for exploiting the structure of optimization problems generated by a modeling language. A model generator can easily be adapted for generating a model in a form that allows the solver to exploit the model structure.
- A modeling system greatly simplifies model analysis within the paradigm specific to a given system. However, using different paradigms – such as soft and/or inverse simulation, regularization, soft constraints, multi-criteria model analysis – typically require much additional effort if the particular paradigm is not included into a given modeling system.
- A modeling system releases a modeler from the complex task of providing code for computing the values of non-linear constraints and the non-linear elements of the Jacobian. The latter especially used to require substantial development time. However, recently tools are available that substantially ease this task, see e.g. Bischof et al. (1992). A typical non-linear problem has only a few formulas for the non-linear part. Therefore, one can use, e.g. Mathematica (Wolfram, 1996) for generating C language code for formulas of the Jacobian and for the values of constraints, and then include this code in a C++ class that provides values for particular elements of the Jacobian and for the constraints.
- Finally, for models that are not only run on various platforms but are also widely distributed, a problem-specific generator substantially decreases costs for the users (typically, the cost of a solver plugged into the problem-specific software is a small fraction of the cost of the run-time license for a modeling system).

The decision about an approach to the implementation of the RAINS model has primarily been based on the following issues, which are the most important for this case study:

- Implementation of the RAINS model using the available modeling systems (in a way that would provide the required analyses) would be difficult, if not impossible.
- The resulting optimization problem is computationally very demanding. Therefore, the optimization problems have to be well preprocessed and scaled in a problem specific way (see subsequent subsections). Moreover, a task-specific setting of parameters for non-linear solvers is also required.
- The software for the analysis of the RAINS model should be distributable, also for users who cannot afford even run-time licenses for a modeling system.

Taking into account the above summarized points, the problem generator of the RAINS model has been implemented as a problem-specific C++ classes that use a C++ template class library supporting the generation of mathematical programming problems. The generator is linked with non-linear solvers. Therefore, a user can run only one executable program (corresponding to a selected solver) to solve a given scenario, see Section 7 for more details.

A commonly accepted rule of thumb for the generation and optimization based analysis of large non-linear models is to generate the model in such a way that various solvers can be tried. For the RAINS model, three solvers, namely CFSQP (Lawrence et al., 1996), CONOPT (Drud, 1996) and MINOS (Murtagh and Saunders, 1987) are used to solve the resulting optimization problem. The task of implementing software that uses several non-linear solvers is interesting from the software engineering point of view. Each solver has a different way to specify an optimization problem. However, most of the software components are common to all the solvers. Therefore, an object-oriented programming approach is a natural choice because it greatly simplifies software development by handling common parts in base classes and by providing solver-specific interfaces through inherited classes.

The approach is conceptually very simple. Each of the above mentioned solvers is available as a library of Fortran subroutines. The generator has C++ classes that are specific for each solver. These classes are inherited from the base classes that handle a common part of the generator. A problem specific report writer processes the results into a form that eases their interpretations. Another class supports a portable interface between C++ and Fortran. Hence, three versions of executables can easily be produced, each composed of the generator, preprocessor, postprocessor and one of the solvers.

A non-linear solver requires routines that compute values as well as elements of the Jacobian of the non-linear constraints and the goal function. A large part of the total computation time is used for the execution of these functions, therefore the efficiency of their implementation is important. The code for the Jacobian has been generated by *Mathematica* (Wolfram, 1996) with a prior use of the *FullSimplify* operator, which simplifies the formulas substantially. This is an easy way to generate an efficient and bug-free code.

Finally, one should notice that the dimensions of the model are not fixed. For some scenarios a part of the constraints and/or variables does not need to be generated. Moreover, the dimensions of the matrices and vectors used in the model definition vary substantially for various types of analysis. Fortunately, constructors of C++ template classes handle such problems in a natural and efficient way.

6.2. *Data handling*

The model has a large number of parameters, but this would not be a problem in itself. The challenge comes from the fact that various parts of the parameters are provided as a result of data processing that is performed on various computers. Data handling for the model has to meet the following requirements:

- efficient handling of a large amount of data;
- binary compatibility, at least for Unix and NT;
- easy handling of basic data structures (the sparse and dense matrices having elements of the types used in this application);
- no royalties fees.

The HDF (Hierarchical Data Format) public domain software by Koziol and Matzke (1998) is used for handling the data in this model. The basic data structures are handled by a collection of well-tested template C++ classes that are also used for the LP-DIT. A C++ interface class has been implemented for an easy and efficient handling of the used data structures by the HDF library.

6.3. *Conversion of PWL functions*

Costs of emission reductions, discussed in Section 5.3.2, are given as PWL (Piece-Wise-Linear) functions of the corresponding emission levels. PWL functions are not smooth. Therefore, in order to be able to use efficient non-linear solvers (which require smooth functions), the PWL cost functions are represented by corresponding smooth functions. However, the PWL functions ((14), (15), (16)) are replaced by sets of inequalities. Due to space limitations, these conversions are not presented here; however, they are described by Amann and Makowski (1999).

6.4. *Preprocessing of the optimization problem*

The preprocessing of an optimization problem is aimed at generating another problem that has the same goal function value as the original problem and fulfills its constraints, but is easier to solve. It is a commonly known fact that the preprocessing of a large optimization problem can dramatically reduce computation time and memory requirements. Preprocessing is a standard feature of any good LP solver. However, preprocessing of non-linear models is a much more difficult task, see e.g. Drud (1997). Generally, preprocessing of an optimization problem in a problem generator is much more efficient than an attempt to preprocess a non-linear problem by a solver. Some instances of the model presented in this paper contain over 25,000 variables and constraints, which makes preprocessing essential.

Preprocessing in the generator is composed of the following elements:

- Outcome variables defined by Eqs. (8)–(16) are not generated. The affected constraints are reformulated to equivalent forms without using these outcome variables (auxiliary functions are implemented to provide values of outcome variables for the report writer).
- The variables en_j and nlv_j and Eqs. (6) and (7) are eliminated and Eq. (10) is modified accordingly.
- All linear constraints are combined into the LP-DIT format by Makowski (1999), and the preprocessing implemented in LP-DIT – which is similar to that implemented by Gondzio (1997) – is applied to these constraints. Only those preprocessing methods based on the analysis of the primal problem can be applied (because of the non-linear goal function). Nevertheless, for many types of scenarios even a majority of linear constraints can be removed from the optimization problem.

Preprocessing substantially reduces the computation time and memory requirements needed for solving an instance of the model. This confirms theoretical expectations.

6.5. Scaling

Scaling of non-linear models is an important element of model preprocessing. Experiences from the earlier stages of the RAINS model development show that a badly scaled model creates numerical problems for all the solvers that are used (one should mention that CONOPT provides very helpful diagnostics based on an analysis of the Jacobian and Hessian that also traces improper scaling). A detailed discussion of the scaling implemented in the RAINS model is far beyond the scope of this paper. Therefore, we only mention that each instance of the optimization problem is scaled in the generator in such a way that:

- absolute values of the elements of the Jacobian and Hessian are smaller than 10^5;
- an attempt is also made to achieve a smallest (non-zero) absolute value of the Jacobian to be "not too small".

7. RAINS model analysis

The RAINS model is used extensively for various types of analysis that are needed for supporting international negotiations, as outlined in Section 8. Due to space constraints, we have limited this section to presenting the structure of one cycle of analysis followed by a summary of the implementation of a composite goal function for the RAINS model analysis.

The structure of one cycle of the RAINS model analysis is outlined in Fig. 2. Prior to analysis, a

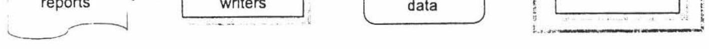

Fig. 2. The structure of one cycle of the RAINS model analysis.

data file is prepared that contains all parameters of the RAINS core model described in Section 5. Another data file with a definition of the parameters for a particular scenario is prepared by specialized software. These two data sets are converted by another specialized program into the HDF format file. Additionally, a user has the possibility to select various options and specify the corresponding parameters (for example, of the composite goal function discussed below) and options (e.g. allowing for soft constraints, requesting the balancing of violations with surpluses) that overwrite the default selections and are used for a definition of a particular instance of the non-linear optimization problem. Such options and parameters are stored in a specification file. Both the HDF data file and the specification file define an instance of the optimization problem.

The optimization problem is generated and solved by the problem specific model generator linked with a selected non-linear solver library. The generator (which functions are described in Section 6) creates the necessary data structures, which are kept in-core and are used for functions that are required by each of the used solvers. Such an approach allows for the efficient generation and solution of the corresponding large non-linear optimization problem. After an optimal solution is found, a postprocessor converts the solution to a form that is convenient for analysis (by "undoing" the actions of the preprocessor and by computing values of variables, which were not generated).

A solution provided by the postprocessor is processed by a specialized report-writer program, which provides various types of information needed for the analysis of a solution. Afterwards, another scenario is designed based on this analysis and on requests from users. This scenario is used as an input to a new cycle of the analysis.

A particular scenario is defined by many parameters. However, we will limit our discussion to the composite goal function (28), which is applied to support the analysis of trade-offs between the following three criteria:

- the minimization of total costs of emissions reduction;
- the minimization of violations of the environmental standards;
- the robustness of solutions.

The first two components have already been discussed; therefore, only the last one requires justification.

A typical problem with the application of optimization techniques for decision support is caused by very different solutions (with almost the same value of the original goal function) corresponding to various instances of a mathematical programming problem that differ very little. The quality of a solution is assessed from the optimization point of view primarily through the value of a goal function; therefore, solutions of slightly perturbed problems may differ substantially. However, from an application point of view, an equally important indication of a solution's robustness is some measure of the closeness of solutions of perturbed problems. Consider, for the sake of illustration, two instances of the RAINS model that differ very little. The values of the goal function for such solutions are typically almost the same. However, it often happens that the optimal solution of the first instance has a high reduction of the emission in country A and a low reduction in country B, while the optimal solution for the second instance has a low reduction in country A and a high reduction in country B. Such solutions would hardly be acceptable. In order to deal with this problem, a technique called regularization, see Section 4.2.3 for a more detailed discussion, is applied.

In order to support the analysis of the above mentioned trade-offs, the criterion function is defined by

$$\text{goal\_function} = \text{cost} + \Theta + \Psi, \tag{28}$$

where the cost term corresponds to the sum of the costs of emission reductions and is defined by (9); the regularizing term Θ and the penalty term Ψ (which also plays a role in regularization) are defined as follows.

The regularizing term Θ is defined by

$$\Theta = \varepsilon \|z - \bar{z}\|^2, \tag{29}$$

where ε is a given positive (not necessarily small) number, z denotes a vector composed of decision variables that correspond to emissions, and $\bar{z}$ is a given vector composed of desired (reference) values of emissions. The role of the term Θ is twofold. First, it helps to avoid large variations of solutions (with almost the same value of the original goal function) for problems that differ very little. Second, it substantially improves the numerical stability of the optimization problem. Additionally, for large values of the parameter ε, the term Θ can be used for the technique called softly constrained inverse simulation that is outlined in Section 4.2.4. Thus, it is possible to analyze trade-offs between minimization of costs and solutions that correspond closely to various given patterns of emissions defined by $\bar{z}$.

The role of the term Ψ is also twofold. First, it serves as a penalty term for optional variables y, ya and ye. Second, it provides regularization for these decision variables, which are not covered by the Θ term. The term Ψ is defined by

$$\Psi = \sum_{l\in L}\sum_{j\in J}\psi(y_{lj}, \rho_0, \sigma_0) + \sum_{j\in J}\psi(ya_j, \rho_a, \sigma_a) + \sum_{j\in J}\psi(ye_j, \rho_e, \sigma_e), \quad (30)$$

where $\rho_0, \rho_a, \rho_e, \sigma_0, \sigma_a, \sigma_e$ are given positive parameters, and the function $\psi(\cdot)$ is defined by

$$\psi(x, \rho, \sigma) = \begin{cases} -\rho\sigma x - \rho\sigma^2/2 & \text{for } x < -\sigma, \\ \rho/2x^2 & \text{for } |x| \leqslant \sigma, \\ \rho\sigma x - \rho\sigma^2/2 & \text{for } x > \sigma. \end{cases} \quad (31)$$

Note that $\psi(x, \rho, \sigma)$ is a smooth function that, depending on the parameters ρ and σ, can be used for both purposes that correspond to the role of the term Ψ outlined above. First, it plays a role of a classical quadratic penalty function, if large values of the parameters ρ, σ are selected. Such a function can be used to examine the trade-offs between violations of air quality standards and minimization of costs. Second, it may not be desirable to apply any penalty function for some scenarios in which the balances between violations of environmental targets and surpluses – given by Eqs. (24)–(27) – are accounted for. However, in such cases, it is still necessary to apply regularization in order to deal correctly with the soft constraints optionally defined by introduction of variables y_{lj}, ya_j, ye_j into constraints Eqs. (20)–(23). A quadratic function may not be suitable for this purpose because often violations and surpluses take small values in some grids and large values in other grids; therefore, it is not possible to find a value of the parameter ρ such that it would allow for large values of violations/surpluses in some grids, while serving as a regularizing term for grids where violations/surpluses may be three orders of magnitude smaller. Therefore, when used for regularization purposes alone, the function ψ is defined using small values of both parameters ρ, σ, which implies using a flat piece-wise linear function with a small quadratic segment needed to make such a function smooth.

We summarize the discussion on the form of the goal function (28) by stressing the fact that a properly defined goal function is the key element for achieving two goals, namely providing a tool for a comprehensive problem analysis and assuring possibly good numerical properties of the corresponding optimization problems. The specific form of this model – in particular, the penalty terms for soft constraint violations, the regularizing terms – make it very similar to a multi-objective formulation, as applied e.g. to softly constrained inverse scenario analysis, see Wierzbicki et al. (1999) for more details.

8. `RAINS` in supporting international negotiations

The short summary of the use of the `RAINS` model presented in this section is based on a more detailed description of its role in international negotiations, which is provided in the Summer'98 issue of *IIASA Options* (available from URL: `www.iiasa.ac.at`). A more technical discussion and examples of interpretations of illustrative results from the `RAINS` model is given by Amann and Makowski (1999).

Modern efforts to control air pollution in Europe began in the 1970s, prompted by concerns over acid rain. The convention on Long-Range Transboundary Air Pollution was signed by all European states, USA and Canada in 1979. The convention was negotiated through the United Nations Economic Commission for Europe (UN-

ECE), and this convention has become a framework for subsequent efforts to improve air quality in Europe. In 1989, when the sulphur protocol was due for renegotiation, the UN-ECE accepted the RAINS model for use in the negotiations. Most probably, this was the first time when all parties to a major international negotiation accepted one computer model as a key tool in their negotiations. Currently, the RAINS model is used not only by UN-ECE, but also by the Council of EU Environment Ministers. There is also a version of RAINS developed for Asia.

However, the acceptance of the model was just the beginning. The negotiators had to trust the model results and to understand how the model works. The scientists had to understand the political realities and modify the model in order to respond better to the requests of the negotiators. In order to illustrate just one element of this process, let us consider an interpretation of the optimality of a solution. From the scientific perspective, a rational optimality criteria is a minimization of the sum of costs of emission reductions subject to constraints on values of the air quality indices. However, this obviously results in solutions that would oblige some countries and/or industries to make larger emission reductions (which also implies substantial costs) than others. Acceptance of such a solution would certainly distort competition; therefore, negotiators cannot accept such solutions. On the other hand, the RAINS model clearly demonstrates that uniform reductions (which are a sound idea from a political point of view) would not only be much more expensive but also would not result in achieving the desired air quality. Another example of this mutual learning process undertaken by the negotiators and scientists is illustrated by the evolution of the understanding of what the desired air quality should be. For example, the results of extensive research have shown that the critical acid loads should vary substantially between various ecosystems. Therefore, there is no justification to apply uniform environmental requirements for all grids in Europe.

From the methodological point of view, the RAINS model of 1989 (which was a small LP model that dealt only with acidification) can be considered as a small pilot prototype of the current version of RAINS described in this paper. The development of several versions of RAINS made in those 10 years were driven by the needs of the negotiators. The first version of RAINS was used for negotiating the sulphur protocol; therefore, it dealt only with a single pollutant. However, it has become clear that a multi-pollutant, multi-effect approach offers substantial environmental and financial advantages. Therefore, to respond to these needs, RAINS has been extended and gradually modified to its current form.

Obviously, RAINS does not provide any "best" solutions. This is simply because, there is a number of problems and trade-offs that are both moral and social. For example:

- Which is more important, protecting forests from acid rain or limiting human exposure to harmful tropospheric ozone?
- Should we put more resources into improving the situation in worst-affected areas, or should we try to spread benefits evenly?
- How do we balance the interests of agriculture versus transport versus electricity production?

No model can actually answer such questions. This remains the domain of negotiations. However, models can help the negotiators to concentrate on those parts of the negotiations that should not be represented by a mathematical model. This assistance is provided by various unbiased analyses, such as computation of the consequences of given policies of emission reductions, or advising the values of emission levels that correspond best to given criteria and constraints.

9. Conclusions

The paper summarizes selected issues of model-based decision-making support and illustrates various modeling paradigms by their application to the generation and analysis of large non-linear models, which are applied to the identification and examination of various policy options aimed at improving air quality in Europe. Various extensions of traditional OR methods that enhance the usefulness of model-based decision support for policy analysis have been presented and illustrated by their application to this complex problem. Software engineering issues pertinent to the generation and analysis of complex and large non-linear models have also been discussed. Finally, a summary of experiences learned from using the RAINS model for supporting international negotiations illustrates both the complexity and usefulness of an application of advanced modeling methods to support the processes of policy-making.

The paper shows that no single modeling paradigm alone is sufficiently good enough to identify and analyze various cost-effective policy options aimed at improving European air quality. Rather, an integration of various modeling methods and tools is needed to provide the best available support possible to analyze this complex problem. Such a conclusion has a more general value. Indeed, the analysis of any complex problem calls for application of various methods and tools to help identify a variety of different policy options and provide ways to compare the consequences of their implementations. Focusing on a particular modeling paradigm considered to be the most appropriate for a given problem was previously justified by limited hardware and software capabilities. However, lessons learned from the applications of various modeling paradigms to very different types of real-world problems, and the recent abundance of computing hardware and software tools makes it possible to integrate several methods of a model specification and analysis, and to apply them to large and complex problems. Such an integration calls for a collaboration of specialists, who have been concentrated – and therefore have substantial experience – in a particular method. Fortunately, recent developments in both research culture and in hardware that supports cooperative work over the Internet has made such a collaboration substantially easier. Therefore, one should expect that various integrations of different modeling paradigms will be used more broadly to improve decision-making support in a wide range of practical problems.

Acknowledgements

The RAINS models have been developed for several years by the Transboundary Air Pollution (TAP) Project at IIASA in collaboration with several European institutions. The author has been collaborating with colleagues of the TAP Project for several years by taking part in various activities related to the development of methodology and software for the generation and the optimization-based analysis of the RAINS models. The author would like to thank all members of the TAP Project, led by Markus Amann, for this collaboration, but especially Chris Heyes and Wolfgang Schöpp for many fruitful discussions. The countless number of various practical modeling problems that have arisen and been solved during this long-term collaboration has also stimulated interesting research developments.

The author would also like to thank Arne Drud of ARKI Consulting and Development A/S, Denmark, for continuously providing the latest versions of the CONOPT libraries (Drud, 1996), and for his consultations on advanced topics related to formulating and solving non-linear optimization problems.

Finally, the author greatfully acknowledges the comments of Tung Bui and Dan Dolk made to an earlier version of this paper. Their suggestions have helped improve this paper considerably.

References

Ackoff, R., 1979. The future of operational research is past. Journal of OR Society 30 (2), 93–104.

Amann, A., Makowski, M., 1999. Assessment of air quality. In: Wierzbicki, A., Makowski, M., Wessels, J. (Eds.), Model-Based Decision Support Methodology with Environmental Applications. Kluwer Academic Publishers, Dordrecht, 122 (2), this issue (to be published).

Bischof, C., Carle, A., Corliss, G., Griewank, A., 1992. Adifor – generating derivative codes for Fortran programs. Scientific Computing 1 (1), 1–29.

Chapman, C., 1992. My two cents worth on how OR should develop. Journal of Operational Research Society 43 (7), 647–664.

Dolk, D., 2000. Integrated model management in the data warehouse era, European Journal of Operational Research.

Drud, A., 1996. CONOPT: A system for large scale non-linear optimization, Reference manual for CONOPT subroutine library, ARKI Consulting and Development A/S, Bagsvaerd, Denmark.

Drud, A., 1997. Interactions between non-linear programming and modeling systems. Mathematical Programming 79 (1–3), 99–123.

Emery, J., 1987. Management Information Systems. The Critical Strategic Resource, Oxford University Press, New York.

Fragniere, E., Gondzio, J., Sarkissian, R., Vial, J., 1997. Structure exploting tool in algebraic modeling languages. Technical Report 1997.2, Department of Management Studies, University of Geneva, Geneva, Switzerland. URL: ecolu-info.unige.ch/~logilab.

Gondzio, J., 1997. Presolve analysis of linear programs prior to applying an interior point method. INFORMS Journal on Computing 9 (1), 73–91.

Goodwin, P., Wright, G., 1991. Decision Analysis for Management Judgment. Wiley, New York.

Granat, J., Makowski, M., 2000. Interactive specification and analysis of aspiration-based preferences, European Journal of Operational Research 122 (2), this issue.

Koziol, Q., Matzke, R., 1998. HDF5 – a new generation of HDF, Reference manual and user guide, National Center for Supercomputing Applications, Champaign, IL. http://hdf.ncsa.uiuc.edu/nra/HDF5/.

Lawrence, C., Zhou, J., Tits, A., 1996. User's guide for CFSQP version 2.4: A C code for solving (large scale) constrained non-linear (minimax) optimization problems, generating iterates satisfying all inequality constraints, Technical report, Institute for Systems Research, University of Maryland, College Park, Maryland.

Makowski, M., 1994a. Design and implementation of model-based decision support systems, Working Paper WP-94-86. International Institute for Applied Systems Analysis, Laxenburg, Austria. Available on-line from http://www.iiasa.ac.at/~marek/pubs/.

Makowski, M., 1994b. Methodology and a modular tool for multiple criteria analysis of LP models. Working Paper WP-94-102. International Institute for Applied Systems Analysis, Laxenburg, Austria. Available on-line from http://www.iiasa.ac.at/~marek/pubs/.

Makowski, M., 1999. LP-DIT++, C++ class library for data interchange tool for linear programming problems, (version 2.06), Interim Report IR-99-xxx, International Institute for Applied Systems Analysis, Laxenburg, Austria (to appear).

Makowski, M., Sosnowski, J., 1989. Mathematical programming package HYBRID. In: Lewandowski, A., Wierzbicki, A. (Eds.), Aspiration Based Decision Support. Theory, Software and Applications, vol. 331 of Lecture Notes in Economics and Mathematical Systems, Springer, Berlin, New York, pp. 106–144 and 376–377.

Murtagh, B., Saunders, M., 1987. MINOS 5.1 user's guide, Technical Report SOL 83-20R, Stanford University, Stanford, CA 94305-4022, USA.

Olendrzyński, K., Berge, E., Bartnicki, J., 2000. EMEP eulerian acid deposition model and its application. European Journal of Operational Research 122 (2), this issue.

Ryoke, M., Nakamori, Y., Heyes, C., Makowski, M., Schöpp, W., 2000. A simplified ozone model based on fuzzy rules generation. European Journal of Operational Research 122 (2), this issue.

Simon, H., 1990. Prediction and prescription in systems modeling. Operations Research 38 (1), 7–14.

Wierzbicki, A., Makowski, M., 1992. Multi-objective optimization in negotiation support, Working Paper WP-92-07, International Institute for Applied Systems Analysis, Laxenburg, Austria. Available on-line from http://www.iiasa.ac.at/~marek/pubs.

Wierzbicki, A., Makowski, M., Wessels, J. (Eds.), 1999. Model-Based Decision Support Methodology with Environmental Applications. Kluwer Academic Publishers, Dordrecht, Boston, London (to be published).

Wolfram, S., 1996. The Mathematica Book, third ed., Mathematica Version 3. Cambridge University Press, Cambridge.

Wright, G., Chaturvedi, A., Mookerjee, R., Garrod, S., 1998. Integrated modeling environments in organizations: An empirical study. Information Systems Research 9 (1), 64–84.