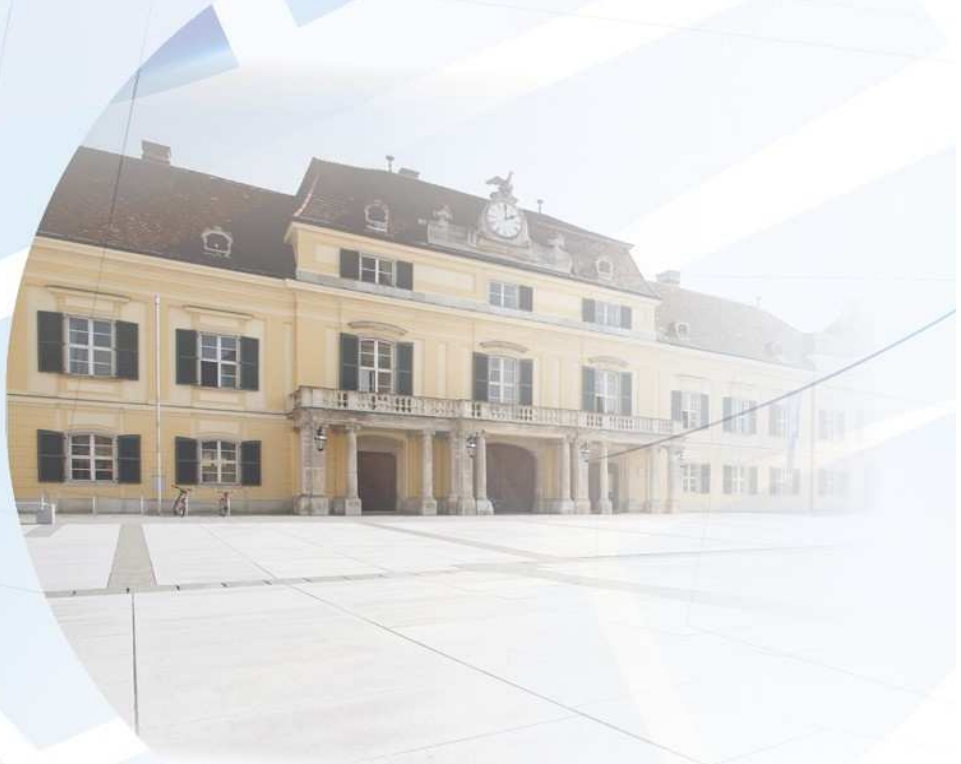
International Institute for Applied Systems Analysis
IIASA www.iiasa.ac.at

Water Quality Management: Problem Formulations and Solution Methods

Ruszczynski, A.

IIASA Working Paper



July 1993

Working Paper

Water quality management: problem formulations and solution methods

Andrzej Ruszczyński

WP-93-36
July 1993

IIASA International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria
Telephone: +43 2236 715210 □ Telex: 079 137 iiasa a □ Telefax: +43 2236 71313

Water quality management: problem formulations and solution methods

Andrzej Ruszczyński

WP-93-36
July 1993

International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria
Telephone: +43 2236 715210 □ Telex: 079 137 iiasa a □ Telefax: +43 2236 71313

Water quality management: problem formulations and solution methods

Andrzej Ruszczyński

1 Introduction

Among the many challenges facing Central and Eastern Europe is the problem of improving the quality of the regions rivers, reservoirs and lakes (see [4] for an extensive discussion of the background). One of the key problems is the choice from among alternative treatment technologies to meet ambient quality standards cost-effectively. The main purpose of this note is to discuss the resulting optimization problems and to develop specialized solution procedures for them.

The model introduced in [4] considers emission sources $i = 1, 2, \ldots, m$, pollutants $l = 1, 2 \ldots, L$ and monitoring points $j = 1, 2, \ldots, n$. The pollutants are transferred from the sources to the monitoring points. In the simplest approach, one can use linear transfer functions which express the ambient water quality Q_j^l for pollutant l at the monitoring point j as

$$Q_j^l = \sum_{i=1}^{m} T_{ij}^l E_i^l + b_j^l, \tag{1}$$

where:

E_i^l - emission of pollutant l at source i;

b_j^l - ambient quality background level.

Transfer coefficients describe the effect of reactions that take place in the water on the way from the source to the monitoring point. In the Streeter-Phelps model

$$T_{ij}^l = \exp\left(-\kappa_l t_{ij}\right), \tag{2}$$

where κ_l is the decay rate and t_{ij} is the travel time from i to j (see [4] for a discussion of other models).

For every source i there is a set $K(i)$ of available treatment technologies. Each technology $k \in K(i)$ is characterized by the following data:

c_{ik} - cost,

E_{ik}^l - emission of pollutant l, $l = 1, 2, \ldots, L$.

A *control policy* is defined as a selection of technologies $k_1, k_2, \ldots, k_m$ such that $k_i \in K(i)$, $i = 1, 2, \ldots, m$. Every control policy is characterized by the following outcomes: the ambient quality levels (1) at monitoring points,

$$Q_j^l = \sum_{i=1}^{m} T_{ij}^l E_{ik_i}^l + b_j^l, \quad j = 1, \ldots, n, \; l = 1, \ldots, L, \tag{3}$$

and the cost

$$c = \sum_{i=1}^{m} c_{ik_i}. \tag{4}$$

Clearly, we would like to have low Q_j^l for all l and j at low cost c, but these goals are contradictory.

In the next section we discuss a number of mathematical programming problems that arise from the above setting: cost minimization, quality optimization and multiobjective formulations. The combinatorial nature of our problem makes the proper problem statement crucial for the efficiency of solution methods. In section 3 we present linear programming formulations of our problems as $0-1$ mixed-integer programs and we briefly discuss available solution methods. In section 4 we further exploit the structure of the problem and we develop dynamic programming statements of our optimization problems. We also discuss specialized algorithms for solving the problems.

2 Basic problems

Before proceeding to detailed problem formulations let us make a simple observation. The physical nature of the problem implies that the transfer coefficients T_{ij} can be different from zero only for sources i located above the monitoring point j. We shall denote the set of such sources by $M(j)$,

$$M(j) = \{i : T_{ij} > 0\}.$$

With no loss of generality we can then restrict the summation in (1) and (3) to $i \in M(j)$.

To simplify the notation, we shall also use vector-valued quality levels

$$Q_j = (Q_j^1, Q_j^2, \ldots, Q_j^L),$$

transfer coefficients

$$T_{ij} = (T_{ij}^1, T_{ij}^2, \ldots, T_{ij}^L)$$

emissions

$$E_{ik} = (E_{ik}^1, E_{ik}^2, \ldots, E_{ik}^L)$$

and background levels

$$b_j = (b_j^1, b_j^2, \ldots, b_j^L).$$

2.1 Cost minimization

The fundamental formulation discussed in [4] assumes that there are quality standards S_j^l for pollutants l at monitoring points j. Then the problem can be stated as follows: find technologies $k_i \in K(i), i = 1, \ldots, m$, so as to

$$\text{minimize}_{\{k_i\}} \left\{ c(k_1, k_2, \ldots, k_m) = \sum_{i=1}^{m} c_{ik_i} \right\} \tag{5}$$

subject to

$$\sum_{i \in M(j)} T_{ij} E_{ik_i} \leq S_j - b_j, \quad j = 1, \ldots, n; \tag{6}$$

where $S_j = (S_j^1, S_j^2, \ldots, S_j^L)$.

2.2 Quality optimization

When the available budget is limited, it may be impossible to fulfill the quality standards (6) at an acceptable cost. Then we have to reformulate the problem and state it in a more realistic way: allocate the resources in such a way that the water quality will be as good as possible. The water quality, however, is given by a collection of outcomes (3) at different monitoring points and for different pollutants. There are many ways to bring them to one integrated quality measure; one possibility is the (scaled) "max" measure

$$Q = \max_{\substack{j=1,\dots,n \\ l=1,\dots,L}} [(Q_j^l - S_j^l)/S_j^l], \tag{7}$$

where S_j^l denote quality standards at points j. In other words, Q is the maximum relative violation of the standards. The problem would be then to minimize this violation subject to the budget constraint $\bar{c}$:

$$\text{minimize}_{\{k_i\}} \left\{ Q(k_1, k_2, \dots, k_m) = \max_{\substack{j=1,\dots,n \\ l=1,\dots,L}} [(\sum_{i \in M(j)} T_{ij}^l E_{ik_i}^l + b_j^l - S_j^l)/S_j^l] \right\} \tag{8}$$

$$\sum_{i=1}^{m} c_{k_i} \leq \bar{c}. \tag{9}$$

This is a typical *worst-case* optimization problem. It may turn out that in some regions where the worst case does not occur the technologies will not be uniquely defined and there will be still a possibility of improving the local quality levels. One way to overcome this drawback is to re-define the quality standards for different regions - we shall discuss this issue together with the multi-objective statement of the problem.

A simple way of forcing all levels downwards (but with the largest weight put on the worst case) is to replace the worst-case function (7) by a quadratic (say) penalty function

$$P = \sum_{j=1}^{n} \sum_{l=1}^{L} [\max(0, (Q_j^l - S_j^l)/S_j^l)]^2. \tag{10}$$

Here, the use of the term $\max(0, (Q_j^l - S_j^l)/S_j^l)$ instead of just $(Q_j^l - S_j^l)/S_j^l$ (as in (7)) is motivated by the necessity to penalize only the violations of the standards, not the improvements. The resulting problem has the form:

$$\text{minimize}_{\{k_i\}} \left\{ P(k_1, k_2, \dots, k_m) = \sum_{j=1}^{n} \sum_{l=1}^{L} [\max(0, (\sum_{i \in M(j)} T_{ij}^l E_{ik_i}^l + b_j^l - S_j^l)/S_j^l)]^2, \right\}. \tag{11}$$

$$\sum_{i=1}^{m} c_{ik_i} \leq \bar{c}. \tag{12}$$

With this formulation we may still have some freedom to choose technologies in the regions where the quality standards can be easily met, but this is not a serious obstacle (see the next subsection). The main difficulty is the nonlinear form of the objective (11) whereas the problem (8)-(9) can be re-formulated as a mixed-integer linear programming problem (see section 3), which allows for the use of powerful large-scale linear programming techniques. On the other hand (11) is *additive* with respect to the measurement points $j = 1, \dots, n$, while (8) is not, which is important for the dynamic programming approach discussed in section 4.

2.3 Multiobjective formulations

The formulations discussed so far assume either stiff quality constraints (6) or budget constraints (9), (12). Since the quality requirements and budget restrictions are in conflict, some compromise is necessary. In our case, one can scan the set of possible compromise solutions by changing the quality requirements S_j^l in (6) or the budget limitations $\bar{c}$ in (9) or (12). There is, however, a more general way of generating attractive compromise solutions. Instead of putting one set of requirements into the objective function and another into the constraints, we can formulate an integrated *achievement function*, which measures the fulfillment of our quality and budget requirements (see [6]). Similarly to (7) the achievement function for our problem can be defined as

$$A_\infty = \max\{\max_{\substack{j=1,\dots,n \\ l=1,\dots,L}} [(Q_j^l - S_j^l)/S_j^l], (c - \bar{c})/\bar{c}\}, \tag{13}$$

where, as before, S_j^l are desired quality standards and $\bar{c}$ is the projected budget. The problem would be then to minimize the achievement function (13) over all policies $k_1, k_2, \dots, k_m$. Let us note that, contrary to earlier formulations, both the quality requirements and the budget are flexible now: we minimize the relative violation of the *reference levels* S_j^l and $\bar{c}$. An acceptable compromise can be achieved by (interactive) setting the reference levels and minimizing (13). Again, we shall see in section 3 that the problem of minimizing (13) can be re-stated as a mixed-integer linear programming problem.

We can also use an additive nonlinear achievement function, similar to (10):

$$A_2 = \sum_{j=1}^{n} \sum_{l=1}^{L} [\max(0, (Q_j^l - S_j^l)/S_j^l)]^2 + [\max(0, (c - \bar{c})/\bar{c})]^2. \tag{14}$$

It may be useful for dynamic programming formulations discussed in section 4.

3 Mixed-integer linear programming formulations

Some of the optimization problems stated in section 2 can be re-formulated as binary mixed-integer linear programs. These reformulations have two advantages: they allow for fast and efficient generation of approximate solutions and also for implementation of advanced techniques for finding optimal solution. The basic concept that leads to mixed-integer programs is the introduction of $0-1$ variables $x_{ik}, i = 1, \dots, m;\ k \in K(i)$, defined as follows:

$$x_{ik} = \begin{cases} 1 & \text{if technology } k \text{ is used at source } i; \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Introducing the constraints

$$\sum_{k \in K(i)} x_{ik} = 1, \tag{16}$$

$$x_{ik} \in \{0, 1\},\ i = 1, \dots, m,\ k \in K(i), \tag{17}$$

we guarantee that at every source i one and only one variable x_{ik} will be equal to 1, i.e., one technology will be selected.

3.1 Cost minimization

The cost minimization problem (5)-(6) can now be stated as follows

$$\text{minimize} \left\{ c = \sum_{i=1}^{m} \sum_{k \in K(i)} c_{ik} x_{ik} \right\} \tag{18}$$

$$\sum_{i \in M(j)} \sum_{k \in K(i)} T_{ij} E_{ik} x_{ik} \leq S_j - b_j, \; j = 1, \ldots, n; \tag{19}$$

$$\sum_{k \in K(i)} x_{ik} = 1, \; i = 1, \ldots, m \tag{20}$$

$$x_{ik} \in \{0, 1\} \; i = 1, \ldots, m; \; k \in K(i). \tag{21}$$

By replacing the integrality constraint (21) with the box constraints

$$0 \leq x_{ik} \leq 1; \quad i = 1, \ldots, m, \; k \in K(i), \tag{22}$$

we obtain a *linear programming relaxation* of the original problem: an approximate problem whose optimal cost function is not larger than the optimal cost function of (18) - (21). Indeed, (22) includes all points defined by (21) so the feasible set of the relaxation is not smaller than the original one. Therefore, the optimal objective function cannot be worse.

It is interesting to interpret the set given by (20), (22) as the set of *mixed technologies* - imaginary weighted combinations of available technologies (with weights $x_{ik}, k \in K(i)$).

3.2 Quality optimization

In order to transform problem (8)-(9) into a linear program let us introduce an additional variable v, which will represent the maximum relative quality violation:

$$Q_j^l - S_j^l \leq v S_j^l, \quad j = 1, \ldots, n, \; l = 1, \ldots, L. \tag{23}$$

We can then state the problem as follows:

$$\text{minimize } v \tag{24}$$

subject to

$$\sum_{i \in M(j)} \sum_{k \in K(i)} T_{ij} E_{ik} x_{ik} - S_j v \leq S_j - b_j, \; j = 1, \ldots, n, \tag{25}$$

$$\sum_{i=1}^{n} \sum_{k \in K(i)} c_{ik} x_{ik} \leq \bar{c}, \tag{26}$$

$$\sum_{k \in K(i)} x_{ik} = 1, \quad i = 1, \ldots, m \tag{27}$$

$$x_{ik} \in \{0, 1\}, \; i = 1, \ldots, m, \; k \in K(i). \tag{28}$$

Again, by replacing the integrality constraint (28) by (22) we get a continuous relaxation of the quality optimization problem.

The problem with the nonlinear penalty (11)-(12) cannot be transformed to a mixed-integer linear program. By using the transformation (15)-(17) we would then obtain a mixed-integer linear-quadratic problem similar to (24)-(28), but with the objective v^2 and an additional constraint $v \geq 0$. Such problems are very difficult to solve as general mixed-integer programs. Instead of using the transformation (15)-(17) there, it seems better to apply special techniques exploiting the combinational structure of the problem in the dynamic programming setting (see section 4).

3.3 Multiobjective formulation

The mixed-integer linear programming formulation of the scalarized multiobjective problem is similar to (24)-(28):

$$\text{minimize} v \tag{29}$$

subject to

$$\sum_{i\in M(j)} \sum_{k\in K(i)} T_{ij}E_{ik}x_{ik} - S_j v \leq S_j - b_j,\ j = 1,\ldots,n, \tag{30}$$

$$\sum_{i=1}^{n} \sum_{k\in K(i)} c_{ik}x_{ik} - \bar{c}v \leq \bar{c}; \tag{31}$$

$$\sum_{k\in K(i)} x_{ik} = 1, \quad i = 1,\ldots,m \tag{32}$$

$$x_{ik} \in \{0,1\} \quad i = 1,\ldots,m,\ k \in K(i). \tag{33}$$

All the remarks following (24)-(28) apply here as well.

3.4 Solution procedures

A general approach to $0-1$ mixed integer linear programming problems is the *branch-and-bound* method (see, e.g. [3]). Its main idea is to search a tree of linear programming relaxations of the original problem. Each node of the tree corresponds to an LP problem which has some of the variables x_{ik} fixed at 0 or 1; other variables are regarded as continuous variables in $[0,1]$. The successor nodes result from fixing a new continuous variable on its bound (0 or 1), etc. The overall number of nodes in the tree can be very large, but special strategies for choosing the node to develop and for cutting the branches can substantially reduce the computational effort. Such strategies are necessary, because the number of nodes in the search tree grows exponentially with the number of sources.

The main idea of directing the search is to bound the optimal values of the nodes in the tree. First, let us observe that if a given node has an integer solution, it provides an upper bound of the optimal value of the whole problem. The best upper bound found so far allows us to cut-off many branches of the tree. Indeed, the minimum value of an LP relaxation at a given node is always a lower bound of the values at all its successors. So, if this value is above the current upper bound, the whole branch can be safely removed from the search tree. It turns out that by a proper choice of the nodes to be processed we can quickly reduce the number of variations to a manageable size (see, e.g. [3] and references therein for an extensive discussion of this issue).

Another important issue of the branch-and-bound methods is the improvement of the lower bound at a given node. It turns out that we can exploit the integrality of the variables to generate new inequalities in the problem which cut-off some parts of the feasible region of the LP relaxation that do not contain integer solutions. Owing to that, the optimum values of the LP relaxations become larger. Consequently, some nodes can be removed earlier from the list of candidates to be processed (see [2, 5], etc.). Finally, there are highly specialized linear programming techniques that can be used to quickly solve the node subproblems. They differ one from another only slightly (some variable is fixed or some cuts added), so their solutions can be updated instead of solving the next problem from the beginning. This, of course, involves complicated updating procedures for sparse matrix factorizations used in the LP solvers at the nodes.

Summing up, the modern branch-and-bound approaches to 0-1 mixed integer linear programming problems are based on a number of advanced techniques of bounding, node

selection, cut generation and LP updating. The resulting highly complicated systems are capable of solving very large problems. Still, to a larger extent than in continuous linear programming, problem re-formulation and problem-oriented modifications of general methods play a crucial role in 0-1 integer programming. It is therefore necessary to work on special methods for our problem, either in the framework of the branch-and-bound method or in other general approaches.

4 Dynamic programming formulations

We already mentioned many times that the key to efficient solution of combinatorial and mixed-integer optimization problems is the proper problem formulation that takes advantage of problem's structure.

In our case, the structure that has not been exploited so far is the sequential nature of the propagation of pollutants. Obviously, they move down the rivers. Consequently, the water quality at a monitoring point j is fully determined by the quality at monitoring points that are immediate predecessors of j and by the emission of sources that are between j and its predecessors.

4.1 The water quality equation

Let us introduce the following notation. For every monitoring point j let $\pi(j)$ denote the set of monitoring points that are immediate predecessors of j in the system, i.e., such points r that r is above j and there are no other monitoring points between r and j. Next, let us define the set $I(j)$ as the set of all sources that are located between j and at least one of its predecessors $r \in \pi(j)$. In other words, $I(j)$ is the set of sources for which j is the first monitoring point ($T_{ij} > 0$ and $T_{ir} = 0$ for $r \in \pi(j)$). Finally, for every $r \in \pi(j)$ we define the transfer coefficients D^l_{rj} (indexed by the pollutant type l) from r to j. Such transfer coefficients for the Streeter-Phelps model (2) have the form

$$D^l_{rj} = \exp(-\kappa_l t_{rj})$$

where t_{rj} is the travel time from r to j. For us, however, it is only important that by the physical nature of the problem such transfer coefficients *must* exist; their form is a secondary issue.

We can now write the fundamental equation for the evolution of water quality levels:

$$Q_j = \sum_{r \in \pi(j)} D_{rj} Q_r + \sum_{i \in I(j)} T_{ij} E_i + \Delta b_j, \; j = 1, \ldots, n. \tag{34}$$

Here Δb_j is the incremental background level at j defined as

$$\Delta b_j = b_j - \sum_{r \in \pi(j)} D_{rj} b_r. \tag{35}$$

Let us note the major difference between (34) and (1). In (34) quality levels are defined recursively, going from the sources to the sinks of the rivers. We shall exploit this feature by the solution procedure. We shall call (34) the *water quality equation*. One can, of course, use it in our earlier models to get alternative problem statements.

4.2 Linear programming formulations with the water quality equation

As an example of the use of the equation (34) in the linear programs of section 3 let us re-formulate the cost minimization problem (5)-(6):

$$\text{minimize} \left\{ c = \sum_{i=1}^{m} \sum_{k \in K(i)} c_{ik} x_{ik} \right\} \tag{36}$$

$$Q_j = \sum_{r \in \pi(j)} D_{rl} Q_r + \sum_{i \in I(j)} \sum_{k \in K(i)} T_{ij} E_{ik} x_{ik} + \Delta b_j, \quad j = 1, \ldots, n, \tag{37}$$

$$Q_j \leq S_j - b_j, \quad j = 1, \ldots, n, \tag{38}$$

$$\sum_{k \in K(i)} x_{ik} = 1, \quad i = 1, \ldots, m \tag{39}$$

$$x_{ik} \in \{0, 1\}, \; i = 1, \ldots, m, \; k \in K(i). \tag{40}$$

Replacing (19) by (37) increased the sparsity of the constraints, which may be of importance for very large sizes of the problem. But the most important feature of the new formulation is that it may provide new hints for strategies within the branch-and-bound method, such as selection of nodes to develop, generation of cutting planes, etc. All these issues need to be investigated in detail.

4.3 Dynamic programming equation for the cost minimization problem

Owing to the recursive dependence of water quality levels by (34), we can use the *principle of optimality* of [1] to develop special dynamic programming equations for some of our problem formulation. The equations can be derived for problems that have additive objectives (such as (5), (11) or (14)). We shall do it for the cost minimization problem (5)-(6). Let us introduce for every monitoring point the set $\Pi(j)$ of *all* monitoring points that lie above j:

$$\Pi(j) = \{j\} \cup \pi(j) \cup \pi(\pi(j)) \cup \ldots \tag{41}$$

Now, we can define the *cumulative cost function* $c_j(Q_j)$ as the optimal value in the following problem

$$\text{minimize}_{\{k_i\}, i \in M(j)} \sum_{i \in M(j)} c_{ik_i} \tag{42}$$

$$\sum_{i \in M(r)} T_{ir} E_{ik_i} + b_r \leq S_r, \quad r \in \Pi(j), \tag{43}$$

$$\sum_{i \in M(j)} T_{ij} E_{ik_i} + b_j \leq Q_j. \tag{44}$$

Problem (42)-(44) has Q_j as a parameter, the optimal value $c_j(Q_j)$ is therefore a function of this parameter. We can interpret $c_j(Q_j)$ as the lowest cost of getting water quality Q_j at point j without violating earlier constraints (for $r \in \Pi(j)$).

It follows from the principle of optimality [1] that the functions $c_j(Q_j)$ are related by the following *dynamic programming equation*:

$$c_j(Q_j) = \min_{\substack{\{k_i\}, i \in I(j) \\ \{Q_r\}, r \in \pi(j)}} \{ \textstyle\sum_{i \in I(j)} c_{ik_i} + \sum_{r \in \pi(j)} c_r(Q_r) | \tag{45}$$

$$\textstyle\sum_{r \in \pi(j)} D_{rj} Q_r + \sum_{i \in I(j)} T_{ij} E_{ik_i} + \Delta b_j \leq Q_j, \tag{46}$$

$$Q_r \leq S_r, r \in \pi(j) \}. \tag{47}$$

In other words, the lowest cost of getting quality Q_j is the minimum sum of costs at getting qualities Q_r at the preceding points $r \in \pi(j)$ and the costs of technologies used in the last section ($i \in I(j)$).

4.4 Dynamic programming equation for the quality optimization problem

In a similar way to the previous section we can develop a dynamic programming equation for the problem (11)-(12) (we choose the quadratic penalty form because it is additive, whilst (8) is not). We define the ***cumulative penalty function*** $P_j(Q_j, c_j)$ as the optimal value of the following problem:

$$\text{minimize}_{\{k_i\}_{i\in M(j)}} \sum_{r\in\Pi(j)} \left(\sum_{l=1}^{L}[\max(0, (\sum_{i\in M(j)} T_{ij}^l E_{ik_i}^l + b_j^l - S_j^l)/S_j^l)]^2\right) \tag{48}$$

$$\sum_{i\in M(j)} T_{ij}E_{ik_i} + b_j \le Q_j, \tag{49}$$

$$\sum_{i\in M(j)} c_{ik_i} \le c_j. \tag{50}$$

This problem has as its parameters the water quality Q_j that has to be reached at the monitoring point j and the sum c_j of costs of technologies used at earlier sources $i \in M(j)$. Therefore its optimal value $P_j(Q_j, c_j)$ is a function of these parameters.

The dynamic programming equation has the form:

$$\begin{aligned} P_j(Q_j, c_j) = \min_{\substack{\{k_i\}_{i\in I(j)} \\ \{Q_r\}_{r\in\pi(j)} \\ \{c_r\}_{r\in\pi(j)}}} \Bigg\{ & \sum_{l=1}^{L}[\max(0, (Q_j^l - S_j^l)/S_j^l)]^2 + \sum_{r\in\pi(j)} P_r(Q_r, c_r)| \\ & \sum_{r\in\pi(j)} D_{rj}Q_r + \sum_{i\in I(j)} T_{ij}E_{ik_i} + \Delta b_j \le Q_j, \\ & \sum_{r\in\pi(j)} c_r + \sum_{i\in I(j)} c_{ik_i} \le c_j \Bigg\}. \end{aligned} \tag{51}$$

Let us observe that we had to increase the "state space" by including to it the cumulative costs c_r.

The dynamic programming equation for the multiobjective formulation (14) is similar to (51), we only add to the last function $P_n(Q_n, c_n)$ the term $[\max(0, (c_n - \bar{c})/\bar{c})]^2$.

The advantage of the dynamic formulation (51) over (24)-(28) and (14) is that it does not contain the budget constraint $\bar{c}$ explicitly. In fact, $P_n(Q_n, c_n)$ provides results for *any* cumulative budget c_n, thus allowing for a selection of a suitable compromise solution.

4.5 Solution procedures

The idea of the solution procedure is already present in the dynamic programming equation: we just have to recursively calculate the cumulative cost functions $c_j(Q_j)$ by (45) starting from the first points j (such that $\pi(j) = \phi$) and moving down the rivers. A standard approach is to introduce a grid in the state space and to round-off Q_j to the nearest grid point. With such an assumption, each Q_r in (45) has only a finite number of different values and the whole problem (45) can be solved by a straightforward enumeration.

Clearly, the number of grid points N has a decisive influence on the number of variations to consider - it will be proportional to $N^{|\pi(j)|+1}$, where $|\pi(j)|$ is the number of immediately preceding monitoring points. Generally, the necessity to deal with all possible values of the state vector is the main drawback of dynamic programming: it is called the "curse of dimensionality". In our case, however, the dimension of the state space is equal to the number of pollutants L (or $L+1$ for (51)) which is usually small (in [4] $L=3$). We can, additionally, exploit the nature of the constraints in (45) to quickly eliminate most of the possibilities.

Together with storing $c_j(Q_j)$ at the grid points we can store also the corresponding solutions of (45): $k_i(Q_j), i \in I(j)$ and $Q_r(Q_j), r \in \pi(j)$. After reaching the last monitoring point n we can choose the lowest value $\hat{c}_n(\hat{Q}_n)$ for $\hat{Q}_n \leq S_n$ - this is the optimal value of the original problem. The solution can be now recovered by the recursive application (moving up the rivers) of the mappings:

$$\hat{k}_i = k_i(\hat{Q}_j),\ i \in I(j), \tag{52}$$

$$\hat{Q}_r = Q_r(Q_j), r \in \pi(j). \tag{53}$$

Solution of (51) is similar - the only difference is that the dimension of the state is larger (the cumulative cost is a state variable here), so the number of grid points will be larger. As a reward, however, we shall obtain the family of solutions for all budget constraints $\bar{c}$ in (12).

To recover the solution for a particular budget constraint $\bar{c}$ we just need to get $\bar{c}_n = \bar{c}$ and find $\hat{Q}_n$ such that

$$P_n(\hat{Q}_n, \hat{c}_n) = \min_{Q_n} P_n(Q_n, \hat{c}_n);$$

this will be our terminal water quality. We move then backwards by applying the mappings:

$$\hat{k}_i = k_i(\hat{Q}_j, \hat{c}_j),\ i \in I(j), \tag{54}$$

$$\hat{Q}_r = Q_r(\hat{Q}_j, \hat{c}_j),\ r \in \pi(j), \tag{55}$$

$$\hat{c}_r = c_r(\hat{Q}_j, \hat{c}_j),\ r \in \pi(j), \tag{56}$$

where, as before, $k_i(Q_j, c_j), Q_r(Q_j, c_j)$ and $c_r(Q_j, c_j)$ are the solution of (51).

Summing up, the dynamic programming algorithm discussed here is relatively easy to implement. Owing to the low dimensionality of the state space (water quality and cost) its costs are moderate. An important feature is that the computation costs grow *lineally* with the number of monitoring points n, which makes the approach attractive for large-scale problems.

It should be stressed that an identical procedure can be used when the linear water quality equation is replaced by a more accurate nonlinear relation.

5 Conclusions

We have discussed various formulations of the problem of cost-effective water quality management strategies: linear programming and dynamic programming formulations. We have also briefly characterized possible solution methods.

The advantage of the mixed-integer linear programming formulations is the availability of efficient general-purpose packages for such problems. They have a number of advanced acceleration techniques implemented, which allows for solving large problems. However,

attempts to solve such problems by some "ad hoc" techniques may fail on larger problems, because of an exponential growth of the computation time when the number of sources and monitoring points increases. Additionally, the mixed-integer approach is practically restricted to linear problems. It may be useful, however, for long term multi-stage planning.

The dynamic programming formulations exploit the physical nature of the problem to a much larger extent than linear programming formulations. They lead to reliable and easy to implement solution methods that can solve linear and nonlinear problems, provided that the objectives are separable. The solution methods can be slower on smaller problems but will prevail on sufficiently large problems, because their cost grows linearly with the number of monitoring points and sources. Another advantage of the dynamic programming approach over linear programming is that it provides in a natural way a family of solutions for varying budget constraints. It is, therefore, easier to use in more complex decision support systems.

Finally, dynamic programming provides a convenient framework for addressing uncertainty of emissions and transfers. We plan to analyze these issues in our further research.

References

[1] R. Bellman, *Dynamic Programming* Princeton University Press, Princeton, N.J., 1957.

[2] J.J. Forrest and J.A. Tomlin, "Optimization Subroutine Library, Guide and Reference", IBM, SC23-0519, 1990.

[3] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinational Optimization*, John Wiley and Sons, 1988.

[4] L. Somlyody and C.M. Paulsen, "Cost-effective water quality management strategies in Central and Eastern Europe", *WP-92-091,* IIASA, 1992.

[5] U. Suhl and L. Suhl, "Computing sparse LU factorizations for large-scale linear programming bases", *ORSA Journal on Computing*, 2(1990), 325-335.

[6] A. Wierzbicki, "A mathematical basis for satisficing decision making, *Mathematical Modeling* 3(1982), 371-405.