A Solution Framework for Linear PDE-Constrained Mixed-Integer Problems

Fabian Gnegel, Armin Fügenschuh, Michael Hagel, Sven Leyffer, Marcus Stiemer

Contents

ii

A Solution Framework for Linear PDE-Constrained Mixed-Integer Problems *

Fabian Gnegel · **Armin Fügenschuh** · **Michael Hagel** ·
Sven Leyffer · **Marcus Stiemer**

Abstract We present a general numerical solution method for control problems with PDE-defined state variables over a finite set of binary or continuous control variables. We show empirically that a naive approach that applies a numerical discretization scheme to the PDEs (and if necessary a linearization scheme) to derive constraints for a mixed-integer linear program (MILP) leads to systems that are too large to be solved with state-of-the-art solvers for MILPs, especially if we desire an accurate approximation of the state variables. Our framework comprises two techniques to mitigate the rise of computation times with increasing discretization level parameters: First, the linear system is solved for a basis of the control space in a preprocessing step. Second, certain constraints are just imposed on demand via the IBM ILOG CPLEX feature of a lazy constraint callback. These techniques are compared with an approach where the relations obtained by the discretization of the continuous constraints are directly included in the MILP. We demonstrate our approach on two examples: modeling of the spread of wildfire and the mitigation of water contamination. In both examples the computational results demonstrate that the solution time is significantly reduced by our methods. In particular, the dependence of the computation time on the size of the spatial discretization of the PDE is significantly reduced.

F. Gnegel
Brandenburg University of Technology Cottbus - Senftenberg, Platz der Deutschen Einheit 1, 03046 Cottbus, Germany
Tel.: +49-355-693150
Fax: +49-355-692769
E-mail: gnegel@b-tu.de

A. Fügenschuh
Brandenburg University of Technology Cottbus - Senftenberg, Platz der Deutschen Einheit 1, 03046 Cottbus, Germany

M. Hagel
Helmut Schmidt University / University of the Federal Armed Forces Hamburg, Holstenhofweg 83, 22043 Hamburg, Germany

S. Leyffer
Mathematics and Computer Science Division, Argonne National Laboratory, 9700 Cass Avenue, Lemont, Illinois 60439, USA

M. Stiemer
Helmut Schmidt University / University of the Federal Armed Forces Hamburg, Holstenhofweg 83, 22043 Hamburg, Germany

1 Introduction

When a model in engineering or other applied sciences yields multiple feasible parameter choices, one naturally asks, what the optimal choice is for the given application in mind. Often these problems are formulated as optimization problems, where the parameters are used as variables and optimality is specified as the minimum (maximum) of an objective function. Problems of very different nature, such as the optimal control of a chemical plant or the search of the shortest path to the closest train station on a map, can be formulated as optimization problems. The former requires the control of a complicated engineering problem, sometimes modeled by some form of differential equation, while the latter is purely combinatorial. Theoretically, one can list all paths, calculate their length, and choose the shortest path. However, because the number of choices grows exponentially with the number of crossings on the map, the best supercomputer in the world would be unable to find the optimal solution with this naive approach, even for a medium-sized map.

The problems considered in this paper comprise both combinatorial decisions, modeled by using integer variables, and physical phenomena, modeled by a partial differential equation (PDE). More precisely, some of the variables of a mixed-integer program are constrained by the solution of a PDE (in the role of a continuous state variable), and the PDE in turn is influenced by the discrete variables. Hence the control variables are partly continuous time-dependent vector-valued functions and partly discrete time-dependent vector-valued functions. Since the feasibility of the PDE-defined state acts as an additional constraint on the mixed-integer program, such problems are called "mixed-integer PDE-constrained optimization" (MIPDECO) (see, e.g., [25]). PDE-constrained optimization without integer variables is already an ambitious field of research (see, e.g., [21] and the literature cited therein), which is reflected by the large number of scientific contributions (e.g., [2]). Continuous control spaces typically fail to be convex with respect to the norm topology; hence their analysis requires more sophisticated topological concepts. In addition, well-posedness is often not naturally present, and a regularization of the problem is required to guarantee convergent discretization schemes. For MIPDECO problems, the integer nature of some of the control variables imposes additional difficulties, including convexity issues and the disconnectedness of the control space.

Two approaches exist for the solution of optimization problems constrained by differential equations: *discretize-then-optimize* and *optimize-then-discretize* (see, e.g., [24]). Here, we follow the *discretize-then-optimize* approach, because our basic idea is to transform the given semi-continuous problem into an approximating mixed-integer linear program (MILP). This approach was also chosen in recent publications that required discrete decision variables in PDE-constrained optimization problems, such as [20], where a finite-volume method is applied. Moreover, we believe that it is unlikely that an *optimize-then-discretize* approach can be developed for MIPDECOs, because it would imply the existence of algebraic optimality conditions for finite-dimensional general MILPs, which seems unlikely.

To study MIPDECO problems, one must combine knowledge from two fields of mathematics, namely, numerics of PDEs and discrete optimization, which historically have little in common and generally use completely different models and mathematical tools. The study of the numerics of PDEs began with the work on finite-difference methods by Richardson in 1911. The examination of discrete or linear optimization was fundamentally influenced by the work of Dantzig in 1947; see also [6]. Research on problems requiring PDEs to be integrated into discrete optimization problems has only recently begun. Most of these studies focus on problems where the PDEs are defined on the arcs of the network, such as in traffic flow problems [12], in production networks [8, 13, 17], or in the coolest path problem [10]. In contrast the problems studied here are constrained by a time- and space-dependent PDE, while their control is constrained by binary variables. To our knowledge publications considering those kinds of problems consider only a one-way influence, meaning that only the PDE is influenced by the flow of the network [4] or the PDE is an additional constraint for the flow on the network [18]. The model described here, however, considers an interdependent bilateral influence: the decision variables may be constrained by the state described by the PDE, and the control of the PDE may be simultaneously constrained by the decision variables.

A critical point in this work is to identify a discretization strategy of the continuous states given by the solution of a PDE that yields an MILP whose complexity still allows for an effective solution by the simplex method. An investigation of this issue revealed that an MILP formulated with the linear constraints directly obtained from a discretization of the PDE by finite differences led to an unacceptably long computation time of the tested state-of-the-art branch-and-bound solver, even though the resulting discretized model was sparse. Although we have not carried out similar computational studies for the finite-element method, we conjecture that the same runtime behavior can be expected. Hence, here we derive an effectively treatable MILP through an additional preparation step that greatly enhances the performance of the subsequent branch-and-bound computation: In our approach a basis of the control space is chosen, and all states corresponding to these basis control functions are computed in advance by solving the continuous state equation in each case. By linearity of the problem the state corresponding to a particular control can quickly be determined by superposition of the previously determined basis states. We note that the dimension of the basis depends linearly on the number of controls and does not grow with the size of the discretization or with the exponential complexity of the problem. Exploiting the time invariance of the linear system defined by the PDE allows for an additional significant reduction of the computational efforts in this approach.

The remainder of this paper is organized as follows. After the formulation of the general model (Section 2), we provide some basic concepts both of the discrete optimization and of the numerics of partial differential equations that are required for deriving our solution scheme (Section 3). Next, two applications are presented that are inspired by real-world problems: First, the organization of a firefighter mission is discussed (Section 4.1). The goal in this application is to minimize damages caused by a spreading wildfire using firefighters operating on the roads inside the forest. While the spread of the fire is modeled by using appropriate PDEs, which also take into account the effect of the firefighters trying to extinguish the fire, the movement of the firefighters is modeled by a dynamic network flow, which is constrained by the temperature function defined via the PDE. The second problem is the prevention of the contamination of a river by wastewater due to mixing (Section 4.2). The idea here is that an underground flow is going from a contaminated site to a nearby river. In order to minimize the contamination of the river, a fixed number of cleaning facilities that can filter the flow have to be installed at optimal locations. For these two examples three different kinds of implementation are presented, and their numerical effectiveness in providing optimal solutions is analyzed (Section 6). In Section 7 we discuss our conclusions derived from the computational experiments and present ideas for future work.

2 General Framework for Mixed-Integer PDE-constrained Problems

The class of problems studied in this work fits into the framework of mixed-integer PDE-constrained optimization problems. As the name suggests, MIPDECO problems are a superclass of PDE-constrained optimization problems allowing for integer restrictions on a subset of the variables. Hence, in addition to the general challenges of PDE-constrained optimization, these problems include combinatorial restrictions that have to be taken into account when developing solution methods. Additionally, even if sufficiently regular boundary and initial conditions guarantee the existence of a unique function satisfying the PDE, this solution function may not possess a representation in a closed form. Therefore, one cannot replace the PDE system by its solution. As is common practice for the numerical treatment of PDEs, finite-dimensional systems of (linear) equations are used in this work to approximate the PDEs. These finite-dimensional systems scale with the resolution of the approximation, and even a relatively coarse discretization scheme may lead to a system that cannot be handled by state-of-the-art MILP solvers. This work carries out computational studies for a specific type of MIPDECO problem in order to analyze which kind of methods are the most promising with regard to the poor scaling behavior as the resolution of the approximation is refined.

In the problem class considered here, we assume that the PDE defining the state function u is linear over a squared domain[1] $\Omega \subset \mathbb{R}^2$ and a time interval $[0,T]$. The vector-valued function $z : [0,T] \to \{0,1\}^{d_z}$, $d_z \in \mathbb{N}$, contains all binary variables and $w : [0,T] \to \mathbb{R}^{d_w}$, $d_w \in \mathbb{N}$, is the vector of the continuous variables used in the control of the PDE and other continuous variables needed to formulate control restrictions. The continuous variables are assumed to be square integrable, that is, functions in $L^2([0,T])$ in each component. The objective function is denoted by $\phi : \mathcal{H} \times \mathcal{V} \to \mathbb{R}$, where $\mathcal{H}$ is the space of regular functions in which the state variable u is searched in (usually $\mathcal{H} = L^2(0,T,W)$, which is a space of functions defined on the time interval $[0,T]$ with values in a suitable Sobolev space $W = W(\Omega)$ of functions defined on Ω to express regularity in space (see e.g. [1]), with W-norms that are L^2-integrable over the time interval $[0,T]$, and $\mathcal{V}$ is the set comprising the control variables w. All constraints besides the PDE are collected in an operator H, which is assumed to be linear and bounded. If the partial differential operator of the PDE is given by $\mathcal{D}$, then the continuous versions of the optimization problems studied here are of the type

$$
\begin{aligned}
\min_{u,w,z} \quad & \phi(u,w) \\
\text{s.t.} \quad & \mathcal{D}(u) = y(w) \\
& u|_{t=0} = f \\
& \alpha\, u|_{\partial\Omega} + \beta \left.\frac{\partial u}{\partial n}\right|_{\partial\Omega} = g \\
& H(u,z,w) \geq 0.
\end{aligned}
\tag{1}
$$

The operator $\mathcal{D}$ provides a mapping from the state-space $\mathcal{H}$ to the space $\mathcal{Y}$ comprising all right-hand sides $y(w)$ that can be adjusted by the control variables w. By choosing the functions $\alpha, \beta : \partial\Omega \to \mathbb{R}$, different boundary conditions such as Dirichlet, Neumann, or Robin boundary conditions (as employed in the wildfire benchmark problem in Section 4.1) can be realized. While every choice of w defines the corresponding state u, the constraints expressed by H may include restrictions on w depending on u. Therefore, the PDE cannot be solved independently of the flow variables, nor can feasibility of the flow variables be guaranteed without taking the solution of the PDE into account.

For any kind of model that involves a PDE a usual requirement is that the PDE be "well posed", meaning that a unique solution exists that continuously depends on the data [19]. Well-posedness is a natural condition for any system that is supposed to model real-world problems. While existence and uniqueness of the solution are required for a plausible interpretation of the solution, only continuous dependence on the data allows for a meaningful approximate approach for solving the problem. For this reason the methods should be applied only if the PDE is guaranteed to be well posed. For the PDEs considered in this paper, which are convection-diffusion equations, this is the case if the initial and boundary conditions are sufficiently regular [23].

3 Background

Because methods from two separate fields of mathematics are used in this work, we give a short introduction to the relevant concepts required in order to follow the later sections with more emphasis on methods for handling PDE constraints. A more in-depth study of the methods and concepts can be found in the references given in the particular sections.

3.1 Methods for Solving Mixed-Integer Linear Programming Problems

Many combinatorial optimization problems can be modeled as mixed-integer linear programs such as

$$
\min\{c^\mathsf{T} x \mid Ax \leq b, x \geq 0, x \in \mathbb{R}^p \times \mathbb{Z}^q\},
\tag{2}
$$

---

[1] The theory can equivalently be formulated for general domains – a square is employed here for notational simplicity.

where $n = p + q$; $A \in \mathbb{R}^{m \times n}$ is the constraint matrix; $b \in \mathbb{R}^m$ is the right-hand side; $c \in \mathbb{R}^n$ are the objective function coefficients; and $x \in \mathbb{R}^n$ are the unknowns or variables of the problem partitioned into continuous variables, x_C, and integer variables, x_I, such that $x = (x_C, x_I)$. The term $c^\mathsf{T}x$ is called the objective function, $Ax \leq b$ is the constraint system, and $x \geq 0$ are the bound constraints. More general constraints such as finite lower and upper bounds or equality constraints are easily included.

MILPs are usually solved by solving a number of related, easier-to-solve linear programming (LP) subproblems. LPs are an important subclass of MILPs in which all variables are continuous. In particular, the LP relaxation of (2) is defined as the MILP in which all integrality restrictions have been relaxed:

$$\min\{c^\mathsf{T}x | Ax \leq b, x \geq 0, x \in \mathbb{R}^n\}. \tag{3}$$

The classical method for solving LPs is the revised simplex method (RSM), whose origins date back to [5]. The RSM is based on the following fundamental observation: Because of the linearity of the constraint system and the objective function, one can show that provided an optimal solution with a finite objective function value exists, it must lie at a vertex of the polyhedron $\{Ax \leq b, x \geq 0\}$. Hence it is sufficient to search over the set of these feasible vertices. The RSM computes a vertex from a subset of m nonzero variables (which form a basis, B, in simplex terminology), written as $x_B = A_B^{-1}b$, where we have partitioned $x = (x_B, x_N)$ and $A = [A_B : A_N]$ and where $x_N = 0$. The RSM moves from one feasible vertex (or basic feasible solution) to another by exchanging exactly one column of A_B with one of A_N, which is referred to as pivoting. Each move of the RSM aims to reduce the objective function, until no such improvement is possible, at which point an optimal solution has been found. Rather than operating with inverse information, the RSM computes and updates LU factors of the basis matrix, A_B. The required rank-one updates to the LU factors can be computed efficiently and robustly for sparse matrices, hence making the RSM an efficient method. In addition, the RSM requires techniques for finding an initial feasible vertex, which can be achieved by applying the RSM to a so-called phase I problem with additional variables. Robust implementations also require safeguards against (approximate) degeneracy which corresponds to a situation where a pivot does not reduce the objective and the RSM could otherwise cycle. The simplex algorithm can have an exponential runtime behavior [22], although it works surprisingly well in practice. The most successful simplex method in the context of MILP is the dual simplex method, which is equivalent to the RSM applied to the dual of (3). Because the dual simplex method starts from a primal infeasible vertex and maintains dual feasibility, it is especially well suited to the reoptimization needed in MILP methods that are described next.

All methods for solving MILPs build on efficient and robust LP solvers. The branch-and-bound method starts by solving the continuous relaxation of (2) in which all integrality restrictions have been relaxed, resulting in (3). The solution of this LP provides a lower bound on the optimal solution of the MILP. If the solution of the LP relaxation is integer feasible, that is, if $x_I \in \mathbb{Z}^q$, then we have solved the MILP. Otherwise, we choose an integer variable, z_i that takes a nonintegral value, $\hat{z}_i$, and we branch on this value by introducing two LPs with added constraints,

$$z_i \leq \lfloor \hat{z}_i \rfloor \quad \text{and} \quad z_i \geq \lceil \hat{z}_i \rceil,$$

respectively, where $\lfloor a \rfloor$ is the floor (rounded down integer) of a and $\lceil a \rceil$ is the ceiling (rounded up integer) of a. The branch-and-bound method continues by solving one of these new LPs, branching again, if the solution is nonintegral. This process creates the branch-and-bound tree, whose nodes are LP relaxations and whose edges correspond to branching on variables. The branch-and-bound method continues to search this tree, solving LPs until no unexplored node is left. A node is explored if we (1) have branched on on variable, (2) have determined that the LP relaxation is infeasible (in which case all LPs in the corresponding subtree are infeasible), (3) have found an integral solution (which provides an upper bound, or (4) have determined that its lower bound is larger than the current upper bound.

An alternative approach to the branch-and-bound method is the cutting-plane method, which are motivated by the observation that we can solve an MILP in a single-shot LP solve if we know the convex hull of the feasible set of the

MILP (2). Unfortunately, determining the convex hull of a general mixed-integer set is as hard as solving an MILP. However, we can often easily find a hyperplane that separates a current point, $z^{(k)}$, from the convex hull of (2). By adding such cutting planes to the LP relaxation, we successively tighten this relaxation until we obtain an integer point.

Modern MILP solvers combine these two approaches into a branch-and-cut approach that adds a possible cut-generation step before each node of the branch-and-bound tree is solved. Often, we can find classes of cutting planes that are large (exponential in the dimensions n or m), and such classes are handled by a cut pool that adds them one at a time as violated cuts are found. The use of the dual simplex method in the solution of the LPs means that the reoptimization needed in the branch-and-cut approach can be implemented efficiently. See, for example, [11, 27], for a discussion of branch-and-cut schemes.

While MILP solvers have achieved a great degree of success, their intrinsic properties make them less suitable for MIPDECOs, as we demonstrate below. In particular, the following two aspects of their implementation, which is tailored to "normal MILPs" (i.e., MILPs without PDE-type constraints), make solving MIPDECOs difficult:

- The constraint matrices from 3D PDEs are often difficult to factorize, and instead, we use iterative solvers for the solution of the discretized PDE. Unfortunately, these iterative solvers are fundamentally incompatible with rank-one updates, and hence the fast reoptimization cannot be readily implemented for MIPDECOs.
- Before actually solving linear programs and performing branch-and-bound, the solver tries to shrink the problem size by eliminating redundancies in the formulation and by strengthening bounds on variables. Dittel et al. [8] demonstrated that these routines tend to accumulate small numerical errors, so that after several time steps an infeasibility of the initial problem is wrongfully detected.

3.2 Finite-Difference Methods

The general idea of finite-difference methods is to replace a (partial) differential equation defined on a continuous domain by a finite-dimensional system of equations of real variables, which approximate the solution of the differential equation on a grid. The grid used for finite-difference methods is often equidistant in each coordinate direction. Here, we consider nodal finite-difference methods, which for every point of the grid define a variable that is associated with the function value of the solution of the PDE. With these variables a system of equations is defined, which are obtained by replacing the differential operators of the PDE by divided differences centered at the grid points. In the case of linear PDEs the resulting system is also linear. Finite differences have a consistency error that vanishes with increasing grid resolution (i.e., $h \to 0$), such as

$$u'(x) = \frac{u(x+h) - u(x)}{h} + O(h) \qquad \text{forward difference,}$$

$$u'(x) = \frac{u(x) - u(x-h)}{h} + O(h) \qquad \text{backward difference,}$$

$$u'(x) = \frac{u(x+h/2) - u(x-h/2)}{h} + O(h^2) \qquad \text{central difference,}$$

$$u''(x) = \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} + O(h^2) \qquad \text{second-order central difference,}$$

which can be derived from the Taylor expansions for sufficiently smooth functions. Because we are interested in phenomena that depend on space and time, we distinguish between the spatial domain Ω, which is assumed to be a square of side length l so that $\Omega = (0, l) \times (0, l)$, and the time domain $[0, T]$ representing the modeled period of time.

For functions $u : \Omega \times [0, T] \to \mathbb{R}$ we use the following notations for its partial derivatives:

$$\nabla u(x,t) := \begin{pmatrix} \frac{\partial u}{\partial x_1}(x,t) \\ \frac{\partial u}{\partial x_2}(x,t) \end{pmatrix},$$

$$\Delta u(x,t) := \frac{\partial^2}{\partial x_1^2} u(x,t) + \frac{\partial^2}{\partial x_2^2} u(x,t).$$

To illustrate how the idea of finite-difference methods is applied in practice we consider linear convection-diffusion equations of type

$$\frac{\partial u}{\partial t}(x,t) - \mathbf{c} \cdot \nabla u(x,t) - D\Delta u(x,t) = y(x,t,w), \qquad x \in \Omega, t \in (0,T], \tag{4a}$$

$$\frac{\partial}{\partial n} u(x,t) = h_L(x)\left(U_L(x) - u(x,t)\right), \qquad x \in \partial\Omega, t \in (0,T], \tag{4b}$$

$$u(x,0) = g(x), \qquad x \in \Omega, \tag{4c}$$

where $n = n(x)$ is the outer normal at $x \in \partial\Omega$. We further assume that the parameters satisfy $c \in \mathbb{R}^2$ and $D \in \mathbb{R}^+$ and that $U_L, h_L : \partial\Omega \to \mathbb{R}$, and $g : \Omega \to \mathbb{R}$ are continuous functions. All PDEs considered in our applications are of this type with varying parameters and domains. To reduce the continuous domain to a finite set of points, we define a grid. Let $p_x + 1$ be the number of points in each spatial coordinate direction (which, for simplicity, are chosen equal in both spatial directions), and let $p_t + 1$ be the number of points in the time coordinate direction. Throughout this paper, we use the notations $\mathbb{T} := \{0, 1, \ldots, p_t\}$ and $\mathbb{X} := \{0, 1, \ldots, p_x\}$ for the index sets. Using these notations, we can define a linear system with variables $u_{i,j}^t$ for each $i, j \in \mathbb{X}$ and $t \in \mathbb{T}$ in such a way that these variables approximate the real values $u(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t)$ of the solution to (4a)–(4c).

This linear system is initialized by the values of g evaluated on the grid, which serve as a discrete version of the initial condition (4c), namely,

$$u_{i,j}^0 = g(i \cdot \Delta x, j \cdot \Delta x), \qquad i, j \in \mathbb{X}. \tag{5}$$

In order to find a suitable replacement for the boundary condition (4b), the so-called ghost point method is used. In this method additional variables $u_{-1,i}^t, u_{p_x+1,i}^t, u_{i,-1}^t, u_{i,p_x+1}^t$ for all $i \in \mathbb{X}$ and $t \in \mathbb{T}$ are included, so that central differences can be used for the points lying on the boundary of Ω. The derivatives in the direction of the outer normal can therefore be approximated by

$$\begin{array}{ll} \frac{\partial}{\partial n} u(0, i \cdot \Delta x, t \cdot \Delta t) \approx \frac{u_{-1,i}^t - u_{1,i}^t}{\Delta x}, & \frac{\partial}{\partial n} u(l, i \cdot \Delta x, t \cdot \Delta t) \approx \frac{u_{p_x+1,i}^t - u_{p_x-1,i}^t}{\Delta x} \\ \frac{\partial}{\partial n} u(i \cdot \Delta x, 0, t \cdot \Delta t) \approx \frac{u_{i,-1}^t - u_{i,1}^t}{\Delta x}, & \frac{\partial}{\partial n} u(i \cdot \Delta x, l, t \cdot \Delta t) \approx \frac{u_{i,p_x}^t - u_{i,p_x-1}^t}{\Delta x} \end{array}, \qquad i \in \mathbb{X}, t \in \mathbb{T}. \tag{6}$$

Substituting (6) into (4b) and replacing $u(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t)$ by $u_{i,j}^t$ in (4b), we obtain the discrete boundary conditions

$$\begin{aligned} u_{-1,i}^t - u_{1,i}^t &= \Delta x \cdot h_L(0, i \cdot \Delta x)(U_L(0, i \cdot \Delta x) - u_{0,i}^t), & i \in \mathbb{X}, t \in \mathbb{T}, \\ u_{p_x+1,i}^t - u_{p_x-1,i}^t &= \Delta x \cdot h_L(l, i \cdot \Delta x)(U_L(l, i \cdot \Delta x) - u_{p_x,i}^t), & i \in \mathbb{X}, t \in \mathbb{T}, \\ u_{i,-1}^t - u_{i,1}^t &= \Delta x \cdot h_L(i \cdot \Delta x, 0)(U_L(i \cdot \Delta x, 0) - u_{i,0}^t), & i \in \mathbb{X}, t \in \mathbb{T}, \\ u_{i,p_x+1}^t - u_{i,p_x-1}^t &= \Delta x \cdot h_L(i, l)(U_L(i, l) - u_{i,p_x}^t), & i \in \mathbb{X}, t \in \mathbb{T}. \end{aligned} \tag{7}$$

We use a Crank-Nicolson-type implicit method for approximating the differential operators in (4a). Therefore, the differential operators in (4a) are replaced by the approximating difference quotients

$$
\frac{\partial u}{\partial t}(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t) \approx \frac{u_{i,j}^t - u_{i,j}^{t-1}}{\Delta t}, \qquad\qquad i,j \in \mathbb{X}, t \in \mathbb{T},
$$

$$
\Delta u(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t) \approx \frac{u_{i-1,j}^{t-1} + u_{i,j-1}^{t-1} - 4u_{i,j}^{t-1} + u_{i+1,j}^{t-1} + u_{i,j+1}^{t-1}}{2\Delta x^2}
$$
$$
+ \frac{u_{i-1,j}^t + u_{i,j-1}^t - 4u_{i,j}^t + u_{i+1,j}^t + u_{i,j+1}^t}{2\Delta x^2}, \qquad i,j \in \mathbb{X}, t \in \mathbb{T},
$$

$$
\frac{\partial u}{\partial x_1}(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t) \approx \frac{u_{i+1,j}^{t-1} - u_{i-1,j}^{t-1}}{4\Delta x} + \frac{u_{i+1,j}^t - u_{i-1,j}^t}{4\Delta x}, \qquad i,j \in \mathbb{X}, t \in \mathbb{T},
$$

$$
\frac{\partial u}{\partial x_2}(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t) \approx \frac{u_{i,j+1}^{t-1} - u_{i,j-1}^{t-1}}{4\Delta x} + \frac{u_{i,j+1}^t - u_{i,j-1}^t}{4\Delta x}, \qquad i,j \in \mathbb{X}, t \in \mathbb{T}.
$$

After simplification we obtain the equations

$$
\begin{aligned}
&(2R + r_2)(u_{i,j-1}^{t-1} + u_{i,j-1}^t) + (2R + r_1)(u_{i-1,j}^{t-1} + u_{i-1,j}^t) - (4 + 8R)u_{i,j}^{t-1} \\
&+ (2R - r_1)(u_{i+1,j}^{t-1} + u_{i+1,j}^t) + (4 - 8R)u_{i,j}^t + (2R - r_2)(u_{i,j+1}^{t-1} + u_{i,j+1}^t) \\
&= \; y(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t), \qquad\qquad\qquad i,j \in \mathbb{X}, t \in \mathbb{T}, \qquad (8)
\end{aligned}
$$

with $R = D\frac{\Delta t}{\Delta x^2}, r_1 = \mathbf{c}_1 \frac{\Delta t}{\Delta x}$ and $r_2 = \mathbf{c}_2 \frac{\Delta t}{\Delta x}$. The linear system hence is given by (5), (7) and (8). The resulting system can be shown to be second-order accurate; and it is also a so-called unconditionally stable method, which implies that it converges for small step sizes for all grid ratios (see, e.g., [28]).

3.3 Galerkin and Finite-Element Methods

The finite-element method (FEM) can be considered a particular Galerkin approach adapted to the discretization of PDEs. Galerkin methods can be applied to approximate stationary points of general variational problems. In the approach presented here it is employed to determine an approximation of the continuous linear state equations, which are thus altered to linear equation constraints for the MILP approximating the originally given MIPDECO.

If $\mathcal{D} : \mathcal{H} \to \mathcal{F}$ denotes a linear operator mapping from a space of (generalized) functions $\mathcal{H}$ defined in Ω to a space of generalized functions $\mathcal{F}$, both being a subset of the space of square-integrable real-valued functions $L^2 = L^2(\Omega)$, and if $y \in \mathcal{F}$ is given, a Galerkin approximation to the solution $u \in \mathcal{H}$ of the general problem

$$
\mathcal{D}u = y, \tag{9}
$$

which is assumed to be well posed, can be constructed as follows: Choose two finite-dimensional linear function spaces V with basis $v_1, \ldots, v_p$ and W with basis $w_1, \ldots, w_q$ (usually we assume $p = q$). The accuracy of the method is governed by the error estimate of the best approximation of a given function in $\mathcal{H}$ by discrete functions from V and by the error estimate of the best approximation of a given function in $\mathcal{F}$ by discrete functions from W (see below). If further we define the standard scalar product in L^2,

$$
\langle f, g \rangle = \int_\Omega fg \, \mathrm{d}x, \tag{10}
$$

then the Ansatz

$$
u_h = \sum_{k=1}^p a_k v_k \in V \tag{11}
$$

with unknown coefficients $a = (a_k)_{k=1,\dots p} \in \mathbb{R}^p$ can in the case $p = q$ be uniquely identified by solving the linear system

$$\sum_{k=1}^{p} a_k \langle v_k, w_j \rangle = \langle y, w_j \rangle, \qquad j = 1, \dots q \tag{12}$$

with system matrix $S = (\langle v_k, w_j \rangle)_{1 \le k \le p, 1 \le j \le q}$ if $\operatorname{rank} S = p = q$. Galerkin methods provide wide freedom in designing discretization schemes for linear operator equations. Moreover, the quality of the Galerkin approximation can be controlled by the best approximation properties of chosen test-and-trial spaces, which allow for rigorous control of accuracy. In the context of FEM spaces (see below) the quality of the best approximation is often estimated via the interpolation error; see [3].

An FEM is now characterized by a particular choice of V and W: These spaces are constructed with the help of a partitioning of the underlying domain into simple geometric elements such as triangles or quads in a planar case. This partitioning defines a mesh with elements, edges, and vertices (see [3] for details). The test-and-trial spaces are now defined as finite-dimensional function spaces containing functions of a certain overall (Sobolev) regularity (see [1] for all questions related to Sobolev spaces) whose restriction to a particular mesh element is a multivariate polynomial of a defined degree. The required overall regularity is usually guaranteed by demanding that certain functionals associated with mesh entities be continuous over element boundaries. In the simplest case, a triangle mesh can be chosen with globally continuous test-and trial-functions whose restriction to any individual triangle is a linear polynomial (particularly, $V = W$). Continuity is then algorithmically guaranteed by requiring continuity in all vertices of the mesh. In this case, the degrees of freedom are given by the values associated with the vertices of the mesh, and a basis of $V = W$ can be obtained by choosing "hat functions", that is, continuous functions that are linear on each triangle with value 1 in one node and 0 in all others. The numerical efficiency of the FEM is connected with the fact that these basis functions are almost local; that is, they share their support only with basis functions in their direct environment. Hence S is sparse.

We will now develop a discrete formulation of the state equation (4a)–(4c) employed in the two benchmark computations presented in this paper. Basically two ways could be chosen that would lead to a discretization of the state equation. The first way involves a two step approach where first a purely spatial FEM is applied, yielding a system of ordinary differential equations (ODEs) with continuous time variable. The latter can than be treated by any absolutely stable standard method for ODEs. In the second approach a time-space-Galerkin scheme is directly applied, specifically, a set of test-and-trial functions are used that vary with respect to both time and space (see, e.g. [30]). In this case the underlying function space is the space $L^2(0, T, L^2(\Omega))$ of functions mapping the time interval to $L^2(\Omega)$ such that the time-dependent $L^2(\Omega)$-norms are square-integrable over $[0, T]$ with scalar product

$$\langle f, g \rangle = \int_0^T \left(\int_\Omega f(x,t) g(x,t) \, \mathrm{d}x \right) \mathrm{d}t \, . \tag{13}$$

In this case, basis functions of the type $v(x_1, x_2) w(t)$ can be chosen, where v and w are both piecewise polynomials glued together in such a way that a particular global regularity arises. Good results with respect to long-term stability have been achieved with discontinuous Galerkin methods for the temporal test-and-trial functions (see [30]). The latter approach seems to be, in general, more adequate for deriving equality constraints to be used within a MILP, since it offers more freedom for algorithm design and, particularly, the opportunity to adjust spatial and temporal discretization parameters simultaneously. Nevertheless, the classical approach via the spatially semi-discrete problem is chosen here. This choice results from the efficient strategy proposed in this work to decompose the space of feasible states in advance, in the sense of a prepossessing step, making a time-space-Galerkin approach obsolete (see Section 4.1). Moreover, in many cases both approaches lead to equivalent algorithmic schemes.

For the semi-discretization step the domain Ω is triangulated, yielding a set of elements $\mathcal{T}$, edges $\mathcal{E}$, and vertices $\mathcal{V}$. As test-and-trial space we chose

$$V = W = \{v \in \mathcal{C}(\Omega) \,:\, v(x_1, x_2)|_T = a_T x_1 + b_T x_2 + c_T, \quad T \in \mathcal{T}, a_T, b_T, c_T \in \mathbb{R}\}, \tag{14}$$

where $\mathcal{C}(\Omega)$ is the space of functions that are continuous on Ω. It is well defined because equality of the values assigned to any vertex in $\mathcal{V}$ for all adjacent elements is already sufficient for global continuity. This is a finite-dimensional vector space of functions possessing Sobolev H^1-regularity whose dimension $p = \dim V = \operatorname{card} \mathcal{V}$ is given by the cardinality of the set of vertices $\mathcal{V}$. This choice of discretization would fail to yield good results for convection-diffusion problems in three spatial dimensions, which is not the case for the problems considered here. Next, equation (4a) is brought into its so-called weak form by multiplying an arbitrary test function $\varphi \in H^1(\Omega)$, integrating over Ω, and applying Green's identity (see, e.g., [3]).

The resulting equation is

$$\left\langle \frac{\partial u}{\partial t}(\cdot, t), \varphi \right\rangle - \langle c \cdot \nabla u(\cdot, t), \varphi \rangle + \langle D \nabla u(\cdot, t), \nabla \varphi \rangle + \int_{\partial \Omega} h_L(x) u(x, t) \varphi(x) \, dx \tag{15}$$

$$= \langle y, \varphi \rangle + \int_{\partial \Omega} h_L(x) U_L(x) \varphi(x) \, dx$$

and is required to hold for all $\varphi \in H^1(\Omega)$. This formulation already contains the required Robin boundary conditions. An adjustment of the test- and trial space as required in the case of Dirichlet boundary conditions is not necessary here. Inserting now the discrete Ansatz $u_h = \sum_{k=1}^p a_k v_k$ and testing with all basis functions v_k, $1 \le k \le p$, we obtain the linear ODE system

$$(S_D + D_c + N_{h_L}) a + M \frac{\partial a}{\partial t} = b_{\mathrm{i}} + b_{\mathrm{bd}} \tag{16}$$

of first order for the time-dependent vector $a = (a_k)_{1 \le k \le p}$ of degrees of freedom corresponding to approximations to the function values in the vertices with the stiffness matrix $S_D = (\langle D \nabla v_i, \nabla v_k \rangle)_{i,k}$, the mass matrix $M = (\langle v_i, v_k \rangle)_{i,k}$, the damping matrix $D_c = (\langle c \cdot \nabla v_i, v_k \rangle)_{i,k}$, the boundary mass matrix $N_{h_L} = (\langle h_L \nabla v_i, v_k \rangle_{\partial \Omega})_{i,k}$, the vector of inner loads $b_{\mathrm{i}}(\langle y, v_k \rangle)_k$, and the vector of boundary loads $b_{\mathrm{bd}} = (\langle h_L U_L, v_k \rangle_{\partial \Omega})_k$. The subscript $\partial \Omega$ indicates that the scalar product in $L^2(\partial \Omega)$ must be used.

In order to obtain a fully discrete scheme, the ODE system (16) has to be discretized with respect to time. In the simplest case, a backward Euler approach can be applied approximating $\frac{\partial a}{\partial t}(t \cdot \Delta t) \approx (a(t \cdot \Delta t + \Delta t) - a(t \cdot \Delta t))/\Delta t \cdot \Delta t$ with a discretization of the time interval in equidistant steps, as described in the beginning of this section. We arrive at the large set of equality constraints for the degrees of freedom of the discretized state at discrete time $t \in \mathbb{T}$ of the form

$$\left(S_D + D_c + N_{h_L} + \frac{1}{\Delta t} M\right) a^{t+1} = b_{\mathrm{i}} + b_{\mathrm{bd}} + \frac{1}{\Delta t} M a^t \tag{17}$$

for all $t \in \mathbb{T}$, beginning with a projection of the initial values on the spatial discrete space a^0 in $t = 0$. Since the backward Euler scheme is absolutely stable, the time step size Δt is limited only because of accuracy requirements as long as $S_D + D_c + N_{h_L}$ is sufficiently well conditioned, which is always the case for diffusion-dominated problems. If convection is dominant, however, numerical problems can be mitigated by choosing a small Δt. From the fully discrete function $u_h \in V$ with $u_h(x_k, t \cdot \Delta t) = a_k^t$ with k being the number of the node $x_k \in \mathcal{V}$, approximated states can be computed in arbitrary points $x \in \Omega$ by first identifying that the element x is contained in and then inserting its barycentric coordinates in a representation of the restriction of u_h to the particular element.

3.4 Dynamic Flow on a Graph

Because the wildfire application (see Section 4.1) requires the modeling of network dynamics, we briefly introduce flows on directed graphs; for a formal definition we refer to [7]. Let $G = (N, A)$ be a *directed graph*, *digraph*, or *network* $G = (N, A)$, which is a pair of two finite nonempty sets N, the *nodes*, and A, the *arcs*, where the elements of A are ordered pairs of elements from N. Let c_a be the capacity, δ_a the transit time, and L_a the length of arc $a \in A$. Furthermore, let there be two distinguished nonempty subsets: $N^+ \subset N$, the *sources*, and $N^- \subset N$, the *sinks*. A *dynamic flow on G of duration T* is a set of functions given by

$$f_a : [0, T] \times [0, L_a] \mapsto \mathbb{R}^+$$

$$(\theta, x) \to \begin{cases} 0, & \theta \leq x \frac{\delta_a}{L_a}, \\ h_a(\theta - x \frac{\delta_a}{L_a}), & \theta > x \frac{\delta_a}{L_a}, \end{cases}$$

where $h_a : (0, T] \mapsto \mathbb{R}^+$ are weakly differentiable functions[2]. The *excess* at a node $j \in N$ is given by

$$\mathrm{ex}(j, \theta) := \sum_{i \in N : (i,j) \in A} \int_0^\theta f_{(i,j)}(t, L_{i,j}) \, \mathrm{d}t - \sum_{i \in N : (j,i) \in A} \int_0^\theta f_{(j,i)}(t, 0) \, \mathrm{d}t \qquad \theta \in [0, T],$$

and the *external flow* f^j at a node $j \in N$ is defined as

$$f^j(\theta) := \frac{d}{d\theta} \mathrm{ex}(j, \theta) = \sum_{i \in N : (i,j) \in A} f_{(i,j)}(\theta, L_{i,j}) - \sum_{i \in N : (j,i) \in A} f_{(j,i)}(\theta, 0) \qquad \theta \in [0, T].$$

Note that if we identify the tail of a with 0 and the head with L_a, the functions f_a of a dynamic flow are the solutions of the aforementioned linear transport equation on a, namely,

$$\frac{\partial}{\partial t} f_a(t, x) + \frac{L_a}{\delta_a} \frac{\partial}{\partial x} f_a(t, x) = 0, \qquad\qquad t \in (0, T], x \in (0, L_a],$$

$$f_a(0, x) = 0, \qquad\qquad x \in (0, L_a],$$

$$f_a(t, 0) = h_a(t), \qquad\qquad t \in (0, T],$$

with h_a as the boundary condition in 0. Furthermore, the excess of a node j at time θ is the difference between the flow that has arrived at j and the flow that has started in j. Now, because any set of weakly differentiable functions $h_a : (0, T] \mapsto \mathbb{R}^+, a \in A$ can be used to define a dynamic flow, we have to impose additional constraints in order to actually link the flows on the individual arcs.

We say that a dynamic flow of duration T is *flow conserving* if

$$f^j(\theta) \geq 0, \qquad\qquad j \in N \setminus N^+, \theta \in [0, T],$$
$$f^j(\theta) \leq 0, \qquad\qquad j \in N \setminus N^-, \theta \in [0, T].$$

From the definition of a dynamic flow, it directly follows that the external flows are given by

$$f^i(\theta) = \sum_{k \in N : (k,i) \in A} h_{(k,i)}(\theta - \delta_{k,i}) - \sum_{k \in N : (i,k) \in A} h_{(i,k)}(\theta), \qquad \theta \in [0, T], i \in N. \tag{18}$$

Hence, although defined for the f_a, flow conservation is best explained as a restriction of the h_a. They have to be chosen in such a way that the external flow is negative at sources, positive at sinks, and zero at all other nodes, coinciding with the physical interpretation of a flow passing through the network. Depending on the application, a variety of other approaches can be found in the literature to model flows in networks. One possibility for expressing dynamic

---

[2] In order to simplify notation it is assumed that these functions extend to $(-\infty, 0)$ with value zero.

changes of a network flow was given by Göttlich et al. [16]. They derive flow models from PDEs, thus allowing for the inclusion of physical properties of the flow occurring, for example, inside the pipes of gas networks.

Another possibility is given by Skutella [29] via the concept of *flow over time*, conveyed by a set of positive-valued Lebesgue integrable functions for each arc: Note that equation (18) and a change of variables can further be used to show the following identity for the excess:

$$
\begin{aligned}
\mathrm{ex}(j, \theta) = \int_0^\theta f^j(t)\,\mathrm{d}t &= \sum_{i \in N:(j,i) \in A} \int_0^\theta h_{(j,i)}(t - \delta_{j,i})\,\mathrm{d}t - \sum_{i \in N:(i,j) \in A} \int_0^\theta h_{(i,j)}(t)\,\mathrm{d}t \\
&= \sum_{i \in N:(j,i) \in A} \int_0^{\theta - \delta_{j,i}} h_{(j,i)}(t)\,\mathrm{d}t - \sum_{i \in N:(i,j) \in A} \int_0^\theta h_{(i,j)}(t)\,\mathrm{d}t.
\end{aligned}
\tag{19}
$$

Furthermore, if $N^+ = \{s\} \subset N$ and $N^- = \{t\} \subset N$, any choice of the h_a defining a flow conserving dynamic flow is called a *strongly flow conserving (s, t) flow over time*.

4 Examples of MIPDECO Problems

We consider two MIPDECO problems that are motivated by real-world applications. The first application considers the optimal response to wildfires, and the second application computes optimal controls to prevent contaminants in subsurface flow from reaching a river.

4.1 Wildfire Containment

Our goal in the first example is to reduce the impact of wildfires, and we consider the following situation that fits the framework of MIPDECO. In a forest close to a populated area with a firefighting department a wildfire in its early stages is reported. Leaving it burning uncontrolled not only might endanger the local population and their properties but also might lead to a major hazard for the whole region. Of utmost importance therefore is that the firefighters plan their response in an optimal way. However, the safety of the units taking part in the operation must be ensured. While most of the forest cannot be crossed with heavy equipment, there is a road network inside the forest that can be used for firefighting operations. All movements are restricted to this road network. In addition no movement should take place on roads leading through or too close to burning territory, since this might endanger the firefighters themselves. Also, the available resources for controlling the fire (water, equipment, and manpower) are limited; therefore an optimal resource allocation and proper scheduling might make the difference between either getting the fire under control or a major disaster.

Two types of dynamics are interacting in this kind of problem and have to be taken into account for optimal planning: (1) the spreading of the fire, where physical experience allows for predicting its spread direction and its velocity, and (2) the movement of the firefighters and their extinguishing agents (water). Since the ultimate goal of any firefighter mission is to limit the spread of the fire; doing so, however, might restrict the movement of the firefighters. Therefore, these two dynamics have to be described in a joint model. We use a time-dependent PDE to model the fire and a dynamic network flow to model the movements of the firefighters or, more precisely, the water that is used. In order to express the interdependencies, the flow variables of the network are used as control variables for the PDE, and additional constraints are imposed on the network flow, including the state variable defined via the PDE. We assume that the road network is given in the form of a directed graph $G := (N, A)$ with capacities $C_{i,j}$ and transit times $\hat{\delta}_{i,j}$ for all arcs $(i, j) \in A$. Each node $i \in N$ is endowed with a coordinate $x_i \in \Omega \subset \mathbb{R}^2$, a square area of interest, and each arc $a \in A$ is associated with a straight line connecting the coordinates of its incident nodes. We further assume that the transportation of water in the firefighting mission can be described approximately by a flow conserving dynamic flow

on G. We further distinguish two subsets of N: The source nodes, denoted by $N^+ \subset N$, represent locations where the firefighters can release or pick up stored water into the network; and the demand nodes (sinks) denoted by $N^- \subset N$ are nodes suitable for extinguishing the fire. We introduce the time horizon $[0, T]$ as the duration of the mission. As shown in the preliminaries, it suffices to specify the boundary functions expressing the flow entering the arcs $a \in A$, which we denote by $w_a : [0, T] \mapsto \mathbb{R}^+$. The flows entering at source nodes $s \in N^+$ are denoted by $w_s : [0, T] \mapsto \mathbb{R}^-$, and the flow exiting at demand nodes $d \in N^-$ are denoted by $w_d : [0, T] \mapsto \mathbb{R}^+$. For all other nodes k we set $w_k = 0$. We further introduce binary-valued functions $z_i : [0, T] \mapsto \{0, 1\}$ for all nodes $i \in N$. These functions are used to assess whether a node is safe or not and are required to be 0 when the temperature at time $t \in [0, T]$ at the respective coordinate x_i exceeds a threshold U_B, and 1 otherwise. Because we require the flow to be feasible and flow conserving, and in order to ensure the safety of the firefighters, the following conditions are imposed:

$$w_{i,j}(0) = 0, \qquad\qquad (i,j) \in A, \qquad (20a)$$

$$\sum_{i:(i,k)\in A} w_{i,k}(t - \hat{\delta}_{i,k}) = w_k(t) + \sum_{j:(k,j)\in A} w_{k,j}(t), \qquad k \in N, t \in (0, T], \qquad (20b)$$

$$w_{i,j}(t) \leq C_{i,j} z_j(t), \qquad\qquad (i,j) \in A, t \in [0, T], \qquad (20c)$$

$$u(x_i, t) - (1 - z_i(t))M \leq U_B, \qquad\qquad i \in N, t \in [0, T], \qquad (20d)$$

$$z_i(t) \in \{0, 1\}, \qquad\qquad i \in N, t \in [0, T], \qquad (20e)$$

where $M > 0$ is an upper bound on the temperature. Here (20a) requires that initially no flow be present in the network, and (20b) ensures the conservation of the flow. Constraints (20c) enable flow only on those arcs $(i, j) \in A$ where the respective head node j is considered as being safe; if $z_j(t) = 1$ for time $t \in [0, T]$ and node $j \in N$, then a flow is possible on any arc (i, j) up to the capacity limit $C_{i,j}$. Constraint (20d) forces $z_j(t) = 0$ if $u(x_i, t) > U_B$; thus the capacity is reduced to zero when the temperature at the head node j exceeds the threshold U_B. For modeling the dynamics of the fire the following PDE system is considered:

$$u_t(x,t) - \mathbf{c} \cdot \nabla u(x,t) - D\Delta u(x,t) = y(x,t,w), \qquad x \in \Omega, t \in (0, T], \qquad (21a)$$

$$\frac{\partial}{\partial n} u(x,t) = h_L(U_L - u(x,t)), \qquad x \in \partial\Omega, t \in (0, T], \qquad (21b)$$

$$u(x,0) = g(x), \qquad x \in \Omega, \qquad (21c)$$

$$u(x,t) \geq U_L, \qquad x \in \Omega, t \in [0, T]. \qquad (21d)$$

Here the squared area Ω and time horizon $[0, T]$ are used respectively as the spatial and time domains for a convection-diffusion equation with Robin-type boundary conditions. With this type of model for the fire we are able to express the effect of the wind and the diffusive behavior of the fire, while keeping the PDE linear. In (21a) the parameter D, the coefficient of the diffusion term, models the speed of the spreading of the fire, and parameter $\mathbf{c}$ is the velocity field of the wind. In the following computational studies these parameters are independent of space and time, but the methods work also in the dependent case. The boundary condition (21b) imposes that the normal derivative at the boundary is directly proportional to the difference of the temperature on the boundary and the ambient temperature U_L. Equation (21c) is the initial condition, which in this case is the initial temperature distribution. Condition (21d) keeps the temperature everywhere above the ambient temperature level. This prevents the control function y from affecting the state in areas that are not burning.

As expressed by the dependency of y on the vector w of flow functions, this function connects the PDE to the dynamic flow. Various choices for y are plausible, and the computational methods remain meaningful as long as y is regular enough to guarantee a unique solution of the PDE system. Because the controls at different nodes do not interact, $|N^-|$ functions that individually scale with the intensity of the control functions w_i, $i \in N$, are chosen as y.

For our computational studies the following functions are chosen:

$$y(x, t, w) = -\gamma \sum_{i \in N^-} w_i(t) \exp\left(-\frac{\|x - x_i\|^2}{\sigma^2}\right).$$

(22)

On the one hand, these function guarantee that each outgoing flow function has a similar effect, shifted to its respective coordinate. On the other hand, they restrict the individual control functions to a local effect, which has its maximum at the respective coordinate x_i and decreases according to a Gaussian distribution. The shape and the scaling with the value of these functions is visualized in Figure 1. The parameters λ and σ can be used to further tune the height and width of the Gaussian, respectively.

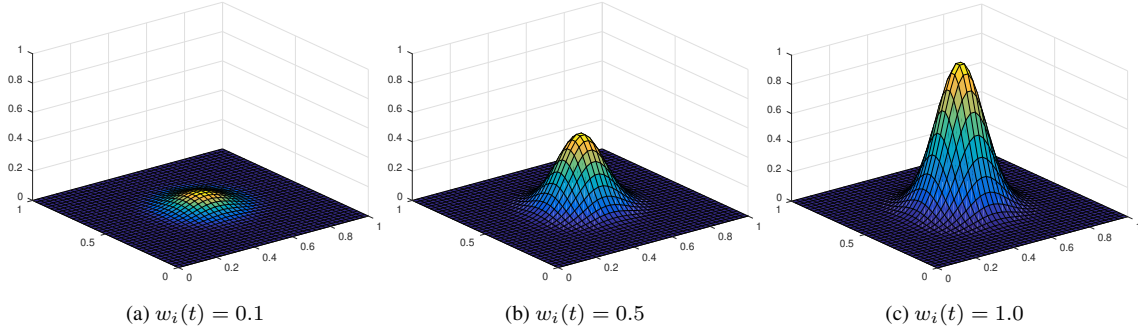(a) $w_i(t) = 0.1$ (b) $w_i(t) = 0.5$ (c) $w_i(t) = 1.0$

Fig. 1: Summands of y for different flow intensities.

The objective of any firefighting mission is to minimize the damage caused by the fire. We assume that the damage is proportional to the integral of the temperature $u(x, t)$ in Ω over the time horizon $[0, T]$ weighted by a space-dependent non-negative function ω. The use of ω guarantees that some regions can be favored over others, for example, if they are inhabited. With this idea in mind we define the infinite-dimensional problem as

$$\min_{u,w,z} \int_0^T \int_\Omega \omega(x) u(x, t) \, \mathrm{d}x \, \mathrm{d}t,$$

(23)

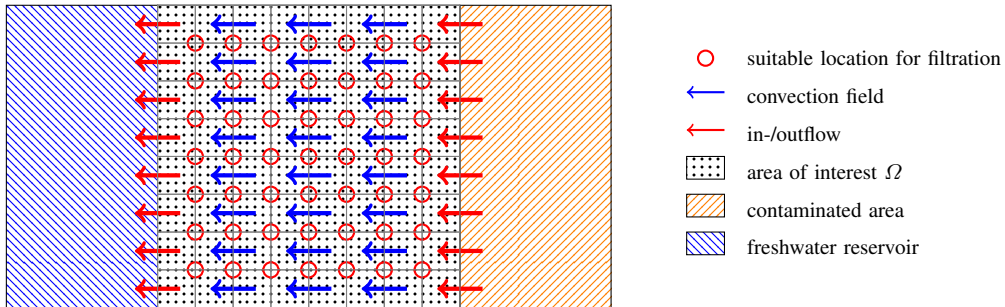s.t. (20a) − (20e), (21a) − (21d).

Fig. 2: Sketch of the problem (differently scaled in $x-$ and $y-$direction).

4.2 Minimization of River Contamination

The application in this section is illustrated in Figure 2. We assume that there is some contaminated area, illustrated by the orange-striped area, and some freshwater reservoir, illustrated by the blue striped area. Between those two areas is a region of porous rocks with a passing underground flow leading to an exchange of water between the two areas. This inner area we denote Ω; it is the domain of the PDE. For simplicity (and especially to reduce the computation times) Ω is considered to be only two-dimensional and squared. Since the flow is going from the contaminated area in the direction of the freshwater reservoir, it is important to prevent any hazardous substances from reaching the freshwater reservoir. To this end, cleaning facilities can be constructed inside Ω, but the budget allows only for the building of K facilities. Thus, an optimal allocation of the cleaning facilities is crucial. To model this problem we derive a model in the framework given by (1). For modeling the underground flow we use a convection-diffusion equation defined on the squared area $\Omega := (0,l) \times (0,l)$ represented by the dotted region in Figure 2:

$$u_t(x,t) - c(x) \cdot \nabla u(x,t) - D(x)\Delta u(x,t) \;=\; y(x,t,w), \qquad\qquad x \in \Omega, t \in (0,T], \qquad (24a)$$

$$\frac{\partial}{\partial n}u((x_1,0),t) \;=\; \frac{\partial}{\partial n}u((x_1,l),t) \;=\; 0, \qquad\qquad x_1 \in (0,l), t \in (0,T], \qquad (24b)$$

$$\frac{\partial}{\partial n}u((0,x_2),t) \;=\; h_L(U_R - u((0,x_2),t)), \qquad\qquad x_2 \in (0,l), t \in (0,T], \qquad (24c)$$

$$\frac{\partial}{\partial n}u((l,x_2),t) \;=\; h_L(U_L - u((l,x_2),t)), \qquad\qquad x_2 \in (0,l), t \in (0,T], \qquad (24d)$$

$$u(x,t) \;\geq\; U_L, \qquad\qquad x \in \Omega, t \in (0,T], \qquad (24e)$$

where u of (24a) represents the quantity of a pollutant in the water (a lower value indicates better water quality). Since the geology may not be uniform inside Ω, we assume that the convection field $c = c(x) \in \mathbb{R}^2$ and the diffusion coefficient $D = D(x) \in \mathbb{R}^+$ are space dependent. The flow on the two vertical sides is set to be proportional to the difference of its level of pollution and the level on the outside, U_L and U_R, respectively, and no flow is passing through the top or bottom boundary. Therefore, we impose homogeneous Neumann-type boundary conditions (24b) on the top and the bottom and Robin-type boundary conditions (24c), (24d) on the borders between the contaminated area and the freshwater reservoir. Similarly to the wildfire problem, constraint (24e) is imposed in order to exclude physically impossible states, namely, those states where the water quality would be better than that of unpolluted water. The cleaning facilities filtrate the water and therefore act like sinks at the distributed n possible locations $x_i \in \Omega$ for $i \in \{1,\ldots,n\}$. Each cleaning facility has a function w_i associated with it that models the filtration for each instant. With these functions we can define the control function y as

$$y(x,t,w) = -\gamma \sum_{i=1}^{n} w_i(t) \exp\left(-\frac{\|x - x_i\|^2}{\sigma^2}\right). \qquad (25)$$

Since only K facilities can be built, we define binary variables $z_i \in \{0,1\}$ for $i \in \{1,\ldots,n\}$, indicating whether a facility will be built or not, and formulate the additional restrictions as follows:

$$0 \leq w_i(t) \leq M z_i, \qquad\qquad i \in \{1,\ldots,n\}, t \in [0,T], \qquad (26a)$$

$$\sum_{i=1}^{n} z_i \leq K, \qquad (26b)$$

$$z_i \in \{0,1\}, \qquad\qquad i \in \{1,\ldots,n\}. \qquad (26c)$$

Here (26a) sets the filtration $w_i(t)$ to 0 if a facility has not been built or bounds it by the maximal possible filtration M, and (26b) bounds the number of facilities by K. With this we can formulate the continuous optimization problem

as

$$\min_{u,w,z} \quad \int_0^T \int_0^l \frac{\partial}{\partial n} u(0, x_2, t) \, dx_2 \, dt, \tag{27}$$
$$\text{s.t.} \quad (24a) - (24e), (26a) - (26c).$$

The objective function is an integral over the flow that is entering the freshwater reservoir.

5 Solution Methods

To solve problems (23) and (27), we first find an approximating finite-dimensional system and then solve this system to optimality. Because all operators in (23) and (27) are linear, the discretized optimization problem is in both cases a finite-dimensional MILP. For the approximation a variety of different approaches exist, two of which were introduced in Section 3. We use these methods to derive two approximations to (23). Approximations to (27) can be derived analogously.

5.1 Direct Replacement of Continuous State Equations

In the first approach to solve the wildfire MIPDECO presented in this work, the continuous state equations given by the PDE are discretized and directly used as constraints in the resulting MILP. To this end, we approximate the PDE using a finite-difference method. However, the conclusions drawn from this approach also apply to an FEM discretization.

For the approximation using finite differences, we use the continuous variables $u_{i,j}^t$ defined in Section 3.2; for ease of notation we assume that the coordinates x_i of the nodes $i \in N$ lie on the grid, and we refer to the temperature at these grid points by u_i^t. In the computational studies they are interpolated by a linear combination of the temperature at the nearest grid points.

For the discretization of the dynamic flow the characterization given by (18) is used. Therefore, the discretization along the spatial dimension of the flow f_a on the arcs can be avoided leading to a more compact approximation. For the discrete models, the functions w_a are replaced with variables $w_{i,j}^t \in \mathbb{R}^+$ for all $(i, j) \in A$ and $t \in \mathbb{T}$ approximating the flow entering arc (i, j) at time $t \cdot \Delta t$. If p_t (the number of time-discretization points) is chosen independently of the transit times, then $\hat{\delta}_{i,j}$ is not necessarily a multiple of Δt for some arcs $(i, j) \in A$. We therefore round down the true transit times to the nearest multiple of Δt by letting

$$\delta_{i,j} := \min_{t \in \mathbb{T}: t \cdot \Delta t \geq \hat{\delta}_{i,j}} t \cdot \Delta t,$$

For all $t \in \mathbb{T}$ we further introduce the discrete variables w_i^t for all $i \in N$ and require $w_i^t \in \mathbb{R}^-$ for all $i \in N \setminus N^-$ and $w_i^t \in \mathbb{R}^+$ for all $i \in N \setminus N^+$ to ensure that flow can enter (or leave) only at sources and sinks. Also the binary-valued functions z are replaced by variables $z_i^t \in \{0, 1\}$ for all $i \in N$ and $t \in \mathbb{T}$. For the solution u of the PDE the goal is to adapt the constraints of the continuous model such that $z_i^t \approx z_i(t \cdot \Delta t)$ and $w_i^t \approx w_i(t \cdot \Delta t)$ for all $i \in N, t \in \mathbb{T}$. Therefore, we approximate the dynamic flow conditions (20a)–(20e) with the conditions

$$w_{i,j}^0 = 0, \qquad\qquad\qquad (i,j) \in A, \tag{28a}$$

$$\sum_{i \in N:(i,k) \in A \wedge \delta_{i,k} \leq t} w_{i,k}^{t-\delta_{i,k}} = w_k^t + \sum_{j \in N:(k,j) \in A} w_{k,j}^t, \qquad\qquad k \in N, t \in \mathbb{T}, \tag{28b}$$

$$u_i^t - (1 - z_i^t)M \leq U_B, \qquad\qquad i \in N, t \in \mathbb{T}, \tag{28c}$$

$$w_{i,j}^t \leq C_{i,j} z_j^t, \qquad\qquad (i,j) \in A, t \in \mathbb{T}, \tag{28d}$$

$$z_i^t \in \{0, 1\}, \qquad\qquad i \in N, t \in \mathbb{T}, \tag{28e}$$

where $M > 0$ is an upper bound on the temperature. While the state constraint (21d) are directly adapted to the discrete setting by requiring

$$u_{i,j}^t \geq U_L, \qquad\qquad i, j \in \mathbb{X}, t \in \mathbb{T} \setminus \{0\}, \tag{29}$$

the conditions (5), (7), and (8) with control given by (22) are added as a replacement for (21a)-(21c). After the integral in the objective function is replaced by a trapezoidal rule on the grid points, the discrete model becomes

$$\min_{u,w,z} \quad \Delta x^2 \, \Delta t \sum_{t=0}^{p_t} \sum_{i=0}^{p_x} \sum_{j=0}^{p_x} \omega(i \cdot \Delta x, j \cdot \Delta x) \lambda_t \mu_i \nu_j u_{i,j}^t, \tag{30}$$

$$\text{s.t.} \qquad\qquad (5), (7), (8), (28a) - (28e), (29),$$

where the objective coefficients are

$$\lambda_t = \begin{cases} 0.5, \ t = 0 \\ 1, \ \ t > 0 \end{cases}, \ \mu_i = \begin{cases} 0.5, \ i \in \{0, p_x\} \\ 1, \ \ \text{otherwise} \end{cases}, \ \text{and} \ \nu_i = \begin{cases} 0.5, \ i \in \{0, p_x\} \\ 1, \ \ \text{otherwise}. \end{cases}$$

In the approach discussed in this section, the discrete relations obtained by the finite-difference method are directly encoded as constraints in the resulting MILP.

5.2 Computing Discrete States for a Basis of the Control Space

For the second approach we use the principle of superposition for linear PDEs. By this principle, the state (i.e., the evolution of temperature) can be determined in any critical point of the infrastructural network for a particular chosen control by superposition of the states determined for a basis of the control space. Here, approximations to the state PDE are computed by the FEM. This is not a principal requirement; however, the FEM facilitates several technical steps in this approach. A discrete FEM solution, as a function defined in whole of Ω, can readily be evaluated in any point where information on the state is needed. Hence the mesh to compute the FEM solution can be chosen independently of the set of points, where the state must be computed. In particular, this approach allows adaptive meshing to increase the accuracy of the FEM approximation exactly in those points where it is required.

Because the operators are linear and the control function is a linear combination of functions, it follows for the continuous state u that

$$u(x, t) = u_{\text{inh}}(x, t) + \sum_{i \in N} w_i(t) u_i(x, t), \tag{31}$$

where u_{inh} is the solution of (21a)–(21c) for $w = \mathbf{0}$ and u_i are the solutions of (21a)–(21c) for each summand in (22), that is for each

$$-\gamma \exp\left(-\frac{\|x - x_i\|^2}{\sigma^2}\right)$$

and homogeneous boundary and initial conditions (i.e., set to zero). The second step is to find a suitable approximation of each function w_i in terms of the discrete variables w_i^t. We therefore define the piecewise constant control

$$w_i(t) \approx \tilde{w}_i(t) = \sum_{\tau \in \mathbb{T}} w_i^\tau \chi_{[\tau \cdot \Delta t, \tau \cdot \Delta t + \Delta t)}(t) \tag{32}$$

as such an approximation, where χ_I is the characteristic function for the interval I (i.e., $\chi_I(t) = 1$ for $t \in I$, and 0 otherwise). With this approximation we can use the principle of superposition in order to approximate the u_i. Denoting the respective basis functions by $\tilde{u}_i^\tau$, we have

$$u_i(x, t) \approx \sum_{\tau \in \mathbb{T}} w_i^\tau \tilde{u}_i^\tau(x, t) \tag{33}$$

and

$$u(x,t) \approx u_{\text{inh}}(x,t) + \sum_{i \in N} \sum_{\tau \in \mathbb{T}} w_i^\tau \tilde{u}_i^\tau(x,t), \tag{34}$$

where the approximation is due to the fact that $\tilde{u}_i^\tau(x,t)$ approximates $u(x,t)$. Using this approximation, however, would require the solution of $p_t(|N|+1)$ PDEs, which is prohibitive because it scales with the number of time steps. For fixed $x \in \Omega$, however, we can exploit the time invariance of the solutions and obtain $\tilde{u}_i^\tau(x,t)$ by shifting the second argument of $\tilde{u}_i^0(x,t)$ to the right:

$$\tilde{u}_i^\tau(x,t) = \begin{cases} 0, & 0 \leq t \leq \tau \cdot \Delta t, \\ \tilde{u}_i^0(x, t - \tau \cdot \Delta t), & \tau \cdot \Delta t < t \leq T, \end{cases} \qquad i \in N.$$

Hence, only $|N|+1$ PDEs need to be solved in order to approximate u, and (34) can be used to replace (21a)-(21c) in the continuous model[3]. This approach also enables us to separate the solution of the PDE from the optimization process, which opens up the possibility to use adaptive FEM instead of finite differences. In addition, finite-element methods in contrast to finite-difference methods define a linear combination of basis functions and thus can be used to derive values anywhere in Ω and not only on a grid. We can therefore use the same variables $u_{i,j}^t$, independent of the finite-element mesh, defined as

$$u_{i,j}^t = u_{\text{inh}}(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t) + \sum_{\tau \in \mathbb{T}} \sum_{k \in N} w_k^\tau \tilde{u}_k^\tau(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t), \qquad i,j \in \mathbb{X}, t \in \mathbb{T}. \tag{35}$$

With this expansion we define the MILP for the second approach as

$$\min_{u,w,z} \quad \Delta x^2 \Delta t \sum_{t \in \mathbb{T}} \sum_{i,j=0}^{p_x} \lambda_t \mu_i \nu_j \omega(i\Delta x, j\Delta x) u_{i,j}^t, \tag{36}$$

$$\text{s.t.} \qquad (28a) - (28e), (29), (35).$$

In practice, this scheme is realized by a table of points for which solutions computed by an FEM have to be evaluated. To do so quickly, an efficient point location algorithm based on a hierarchical substructuring of the finite-element mesh is implemented in the employed software oFEM [9]. In this way, the constraints of the MILP obtained after discretization are computed in advance.

6 Implementation Details and Computational Experiments

In this section we study the performance of the two discretization techniques when applied to the system (23) modeling a wildfire spread and to the system (27) modeling the flow of contaminated groundwater. More precisely we study the impact of the discretization parameters Δx and Δt on the overall solution time of an MILP solver. All computations were carried out on a 2018 Mac mini with a 3.2 GHz Intel Core i7 CPU and 64 GB RAM. All occurring MILPs were solved by the branch-and-cut solver IBM ILOG CPLEX 12.8.0.0, and the PDE solutions needed in the preprocessing step of the model based on finite elements were obtained by employing the object-oriented software package oFEM [9]. While the IBM ILOG CPLEX can be used as a black box for solving MILPs, it also provides "callbacks", which are interfaces that can be used to include customized subroutines in the branch-and-cut process. Taking a closer look at the discretized state constraints in (29), one can see that each outflow variable has an influence on every point of the grid. Therefore, condition (29) and the respective discrete variant of (24e), which are sparse when $u_{i,j}^t$ is included as variables, become dense after substituting (35). These are the only conditions that scale with the discretization

---

Algorithm 1: Lazy Constraint Callback (CB)

---

1 **for** $t \in \mathbb{T}$ **do**

2   **for** $i \in \mathbb{X}$ **do**

3   **for** $j \in \mathbb{X}$ **do**

4               $u_{i,j}^t \leftarrow u_{\text{inh}}(i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t) + \sum_{\tau \in \mathbb{T}} \sum_{k \in N} w_k^\tau \tilde{u}_k^\tau (i \cdot \Delta x, j \cdot \Delta x, t \cdot \Delta t)$;

5       $u_{\min} \leftarrow \min\limits_{(i,j) \in \mathbb{X} \times \mathbb{X}} u_{i,j}^t$;

6       $(\bar{i}, \bar{j}) \leftarrow \arg\min\limits_{(i,j) \in \mathbb{X} \times \mathbb{X}} u_{i,j}^t$;

7   **if** $u_{\min} < U_L$ **then**

8   add Constraint $u_{\bar{i}, \bar{j}}^t \geq U_L$;

---

resolution in space and time. Thus, these conditions might make the LP-relaxations in the branch-and-cut scheme more complex and their solution more time consuming. Many of these state constraints are redundant because neighboring grid points do not differ much in their function values, and hence one might be able to enforce only a few of them during the branch-and-cut process to find an optimum that already fulfills them all. If this is the case, it is also an optimal solution for the original problem. Constraints that are unlikely to cut off feasible solutions found in the branch-and-cut process are often referred to as lazy constraints and can be treated specially in the solution process by using a callback. The so-called lazy constraint callback allows adding these constraints to the model on demand, speeding up the solution of the LP relaxations at the cost of making branching decisions based on incomplete information. The addition of these constraints has to be done carefully. If all constraints that are violated are added in every callback, the amount of additional constraints might not be much less than the original one, and the speedup would be negligible. Therefore, we implemented it as given in the pseudocode of Algorithm 1, which is executed whenever an incumbent is found.

In this callback the temperatures (the concentrations of the pollutant) at the grid points are calculated, and the minimum for each time step and the corresponding indices are stored. If the minimum is less than the lower bound U_L, the incumbent is not feasible, and the respective constraint is added to the model. This approach adds at most one constraint per timestep to the model. If no constraint is added, the incumbent does not violate any of the lazy constraints and is accepted as the new optimal solution. This callback might be called many times, and the sum of its execution times might outweigh the potential gains from reducing the number of active constraints. Furthermore, it is likely that the branching order is changed, which might also increase or decrease the total computational effort. Therefore, computational studies of the FEM-based model are carried out twice, once with the callback switched on and once with the callback switched off.

Computational Experiments for the Wildfire Application: The model (30) based on finite differences contains significantly more constraints and variables than does the second model (36) based on an FEM. Yet, this by no means can be seen as evidence that the second model can always be solved faster than the first one. To study the impact of the discretization resolutions p_x and p_t, we fixed all other parameters to the values given in Table 1.

Snapshots and a link to the full video of a visualization of an optimal solution of this configuration can be found in Figure 3.

The computation times for the two approaches are depicted in the Figures 4a and 4b. The time limit was set to 20,000 seconds. If it was reached for an instance, no computation time is reported and there is a gap in the plot. The different graphs show the runtimes for different levels of time and space discretization with a logarithmic scale on the

---

[3] A similar approach has independently be found and applied by Anna Thünen (unpublished).

Table 1: Parameter configuration for the wildfire model.

Parameter	$\lvert N\rvert$	$\lvert A\rvert$	$C_{i,j}$	T	U_L	h_L	U_B	c	D	γ	σ	M	$\omega(x)$
Value	19	18	0.15	1.0	0.0	0.0	0.8	$\begin{pmatrix}-0.01 & -0.01\end{pmatrix}$	0.02	25.0	0.18	5.0	1.0

Parameter	$g(x)$
Value	$0.3 + \begin{cases} 1.2, & \text{if } \left\| x - \begin{pmatrix}1 & 1\end{pmatrix}^{\mathsf{T}} \right\|^2 \le 0.4 \\ 0.0, & \text{else} \end{cases}$

(a) *First timestep* (b) *Median timestep* (c) *Final timestep* (d) *Video*
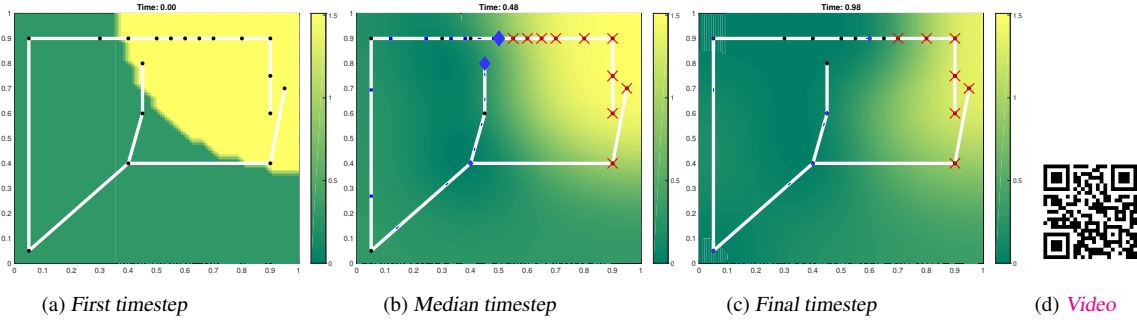
Fig. 3: Visualization of an optimal solution.

vertical axis. One of the difficulties in comparing the two approaches is that the presolve routine of the IBM ILOG CPLEX runs into numerical difficulties on many of the instances of the finite-difference approach and cuts off all feasible solutions. Therefore the graphs in Figure 4b are the results of a run in which bound-strengthening and the variable aggregator are switched off. The difference in computation time is still clearly visible since the first method could not solve any instance with a spatial resolution of over 20×20 grid points, while the FEM-based approach was able to solve all instances with a spatial resolution of up to 160×160. The results suggest that even larger resolutions can be solved, which we could not verify because of memory limitations.
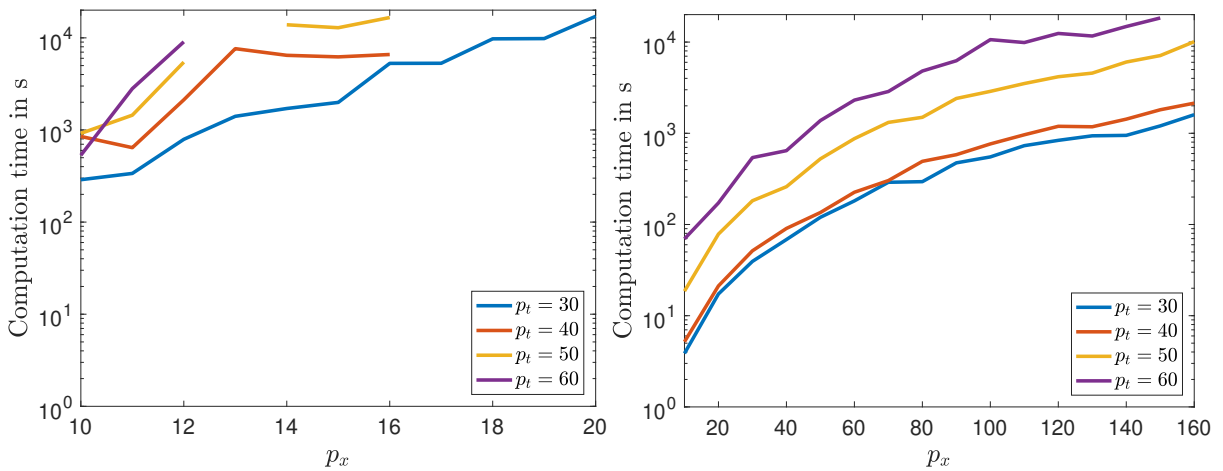
(a) Finite-difference-based approach with direct use of the discretized constraints.

(b) Finite-element-based approach with precomputed basis of the states.

Fig. 4: Computational results for different numbers of time steps p_t for the two approaches applied to the wildfire model.

The best computational results, however, were achieved with the lazy constraint callback switched on, given in Figure 5a. All instances could be solved for a spatial resolution of up to 160×160. Switching on the callback, as visualized by the graph in Figure 5b showing the ratios of the computation times, leads to a speedup by a factor of up to 200 compared with the computation times with the callback switched off.
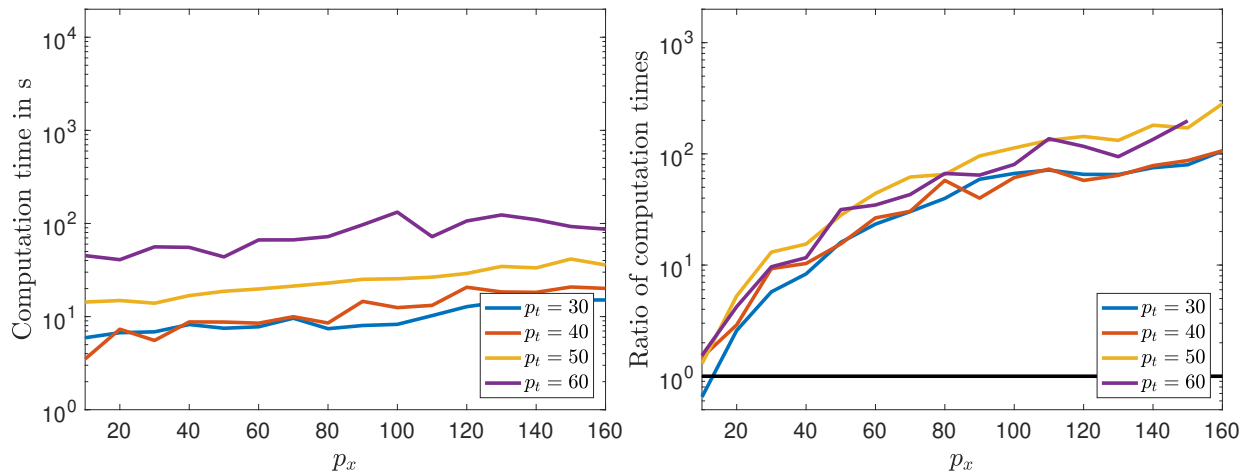
(a) FEM-based approach with precomputed basis states and constraint callback on.

(b) Ratio of CPU times without/with constraint callback.

Fig. 5: Computational results for constraint callback.

Computational Experiments for the River Contamination Application: A visualization of the parameter configuration for the river contamination problem given in Table 2 and its optimal solution are depicted in Figure 6.

Table 2: Parameter configuration for the river contamination model.

Parameter	n	K	l	T	U_L	U_R	h_L	γ	σ	M	$c(x)$	$D(x)$	$\omega(x)$
Value	30	5	1.0	1.0	0.0	5.0	1.0	30.0	0.18	5.0	$\begin{pmatrix} -1.0 & 0.1 \end{pmatrix}$	0.07	1.0

Parameter	$g(x)$
Value	$0.5 + \begin{cases} 2.2, & \text{if } \left\| x - \begin{pmatrix} 1.0 & 0.5 \end{pmatrix}^{\mathsf{T}} \right\|^2 \leq 0.4 \\ 0.0, & \text{else} \end{cases}$

Because the IBM ILOG CPLEX timed out for many instances with an almost negligible remaining gap between proven lower and upper bounds, the tolerance for an optimal solution was set to 1% for all instances of the river contamination model. Analogously to the graphs in Figure 4, the results of the two approaches applied to (27) are depicted in Figure 7. The advantages of the FEM-based approach are again clearly visible because the first method could not solve any instance with a spatial resolution of over 21×21 grid points, while the FEM-based approach was able to solve instances with up to 90×90 grid points.

As shown in Figure 8, switching on the lazy constraint callback again leads to a large reduction in computation time, and therefore resolutions of up to 90×90 grid points could be solved for all time discretizations.
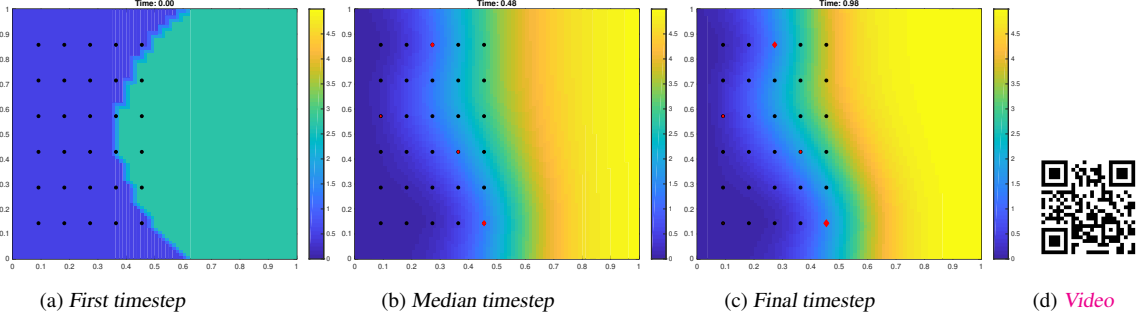
(a) *First timestep*   (b) *Median timestep*   (c) *Final timestep*   (d) *Video*

Fig. 6: Visualization of an optimal solution.

(a) Finite-differences-based approach with discretized PDE constraints.   (b) FEM-based approach with precomputed basis states.
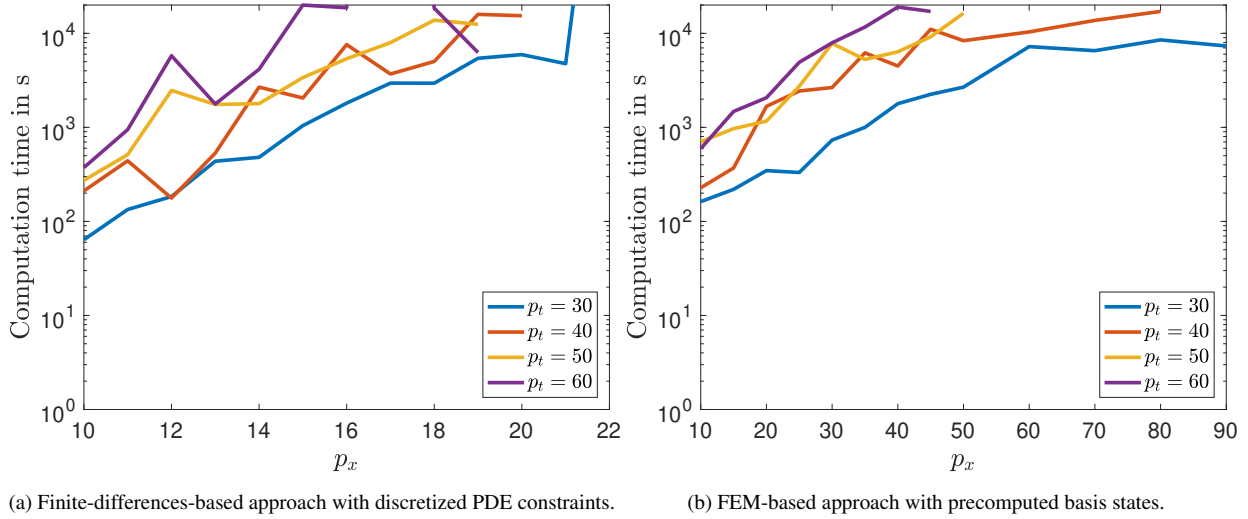
Fig. 7: Computational results for different numbers of timesteps p_t for the two approaches applied to the river contamination model.

7 Conclusions and Future Work

The combination of PDE-constrained and discrete optimization results in a new class of MIPDECO problems that allow for the modeling and mathematical treatment of a variety of important problems in science and logistics that could not have been solved before. This is a challenging class of problems, and we demonstrate empirically that a simple discretize-then-optimize approach using a finite-difference scheme for MIPDECO results in MILPs that cannot be solved in a reasonable amount of time for sufficiently fine discretizations.

We propose a general scheme to tackle such problems and demonstrate its application in two example problems. Our approach constructs PDE solutions for a basis of the control space, and we show how the superposition principle can be exploited to reduce the number of PDE solves required to construct a full basis. Our approach has been implemented based on the FEM discretization, and we obtain a much better performance. We note that this advantage is not inherent in the use of the FEM, because the linear system defined by finite-difference methods could also be solved in advance. The advantage of using the FEM in the preprocessing step is that its solution can be used to directly obtain values anywhere in the domain and that the accuracy of the picked-up values can be increased and controlled by local mesh refinement. Finite-difference methods, in contrast, approximate the solution only on the underlying grid points that are usually regularly distributed and require a particular interpolation technique. Moreover, grid adaptation and error control are much more difficult in this case. We further conclude that the second approach allows
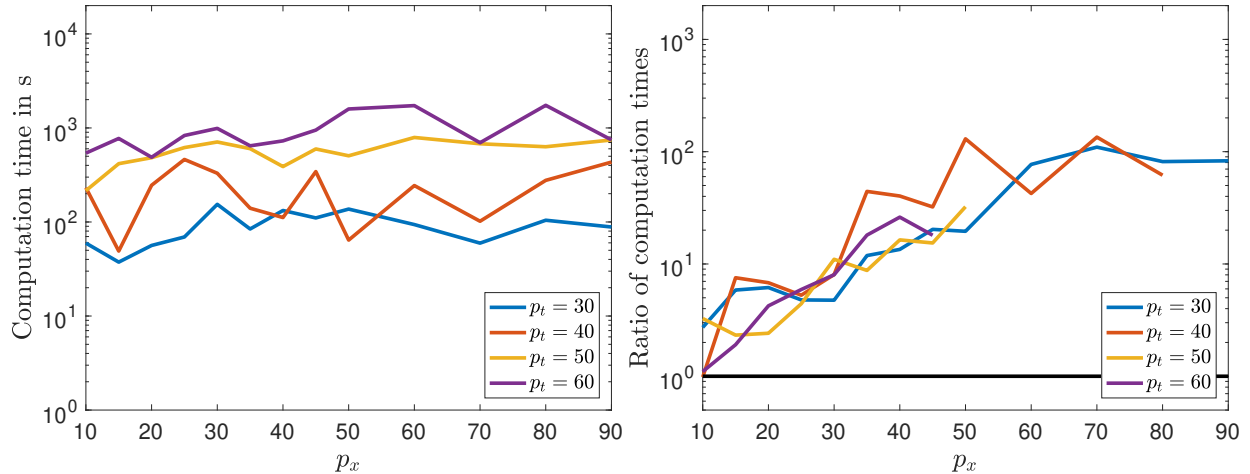
(a) FEM-based approach with precomputed basis states and constraint (b) Ratio of CPU times without/with constraint callback.
callback on.

Fig. 8: Computational results for constraint callback.

for knowledge about the peculiarities of the constraints derived from the PDE discretization (i.e., the redundancy of many state constraints) to be incorporated in the solution method. This observation reduces the runtimes especially for approximations with fine spatial resolutions. Our approach demonstrates that eliminating the PDE constraints and variables whose number scales with the discretization size seems to be the only viable method for solving MIPDECO approximations with high accuracy.

Further investigations into these methods are required to illuminate the analytical background of the proposed methods and to discuss convergence properties on a formal level. In particular, the importance of regularization has not been discussed in this paper, and only nonregularized problems are solved. Regularization could prevent the algorithmic scheme from convergence if the discretization level is increased.

Additionally, an enormous potential exists to enhance the algorithmic formulations proposed in this work. The flexibility of the FEM with respect to, for example, geometric adaptability and rigorous error control, should be further exploited in future approaches. The FEM offers the particular advantage of providing methods that allow for spatial discretization in such a way that a given functional of interest, namely, a figure that is determined from the solution of the PDE, can be approximated as well as possible with the numerical resources at hand. An efficient way to achieve this goal is mesh adaptation based on a posteriori error control, for example, with the method of dual weighted residuals or so-called goal-oriented error estimators. The constraints to be evaluated for the approximating MILPs refer to just a small number of degrees of freedom of the FEM problem defining the continuous state variables. Only these must be known and computed accurately in the context of an active-set strategy. The results presented in this work further suggest that a general framework for a highly efficient treatment of MIPDECO problems could be based on a time-space-Galerkin method relying on sets of tensor-product basis functions with a discontinuous Galerkin method in time. Also of interest is how far the utilized FEM basis functions and meshes can be adapted to guarantee a fast computation of all states related to a suitable basis of the control space, so that the discretization of the PDE-defined constraints can be turned into discrete versions in such a way that IBM ILOG CPLEX (or another MILP solver) can deal with them in the most efficient way.

The proposed method works only for linear PDEs, which often do not capture all the important properties of a real-world application. For example, an exact model for the physics of fire requires a PDE system including nonlinear terms (see, for example, [26], where the PDE modeling the fire is coupled to a fuel term with a nonlinear expression).

Because our method explicitly relies on linearity, it does not apply for nonlinear models. It might, however, be useful for approaches in which the occurring nonlinear terms are approximated by linear or piecewise linear functions.

Acknowledgments

Conflict of Interest

The authors declare that they have no conflict of interest.

References

1. Adams RA (1975) Sobolev Spaces. Academic Press
2. Antil H, Kouri DP, Lacasse MD, Ridzal D (2018) Frontiers in PDE-Constrained Optimization, vol 163. Springer
3. Braess D (1997) Finite Elemente: Theorie, schnelle Löser und Anwendungen in der Elastizitätstheorie. Springer-Verlag
4. Buchheim C, Meyer C, Schäfer R (2018) Combinatorial optimal control of semilinear elliptic PDEs. Computational Optimization and Applications 70(3):641–675
5. Dantzig G (1948) Linear Programming. In: Symposium on Numerical Methods, July 1948, Los Angeles, California, National Bureau of Standards
6. Dantzig G, Fulkerson D, Johnson S (1954) Solution of a large scale traveling salesman problem. Tech. rep., RAND Corporation, Santa Monica, California, USA
7. Diestel R (2012) Graph Theory, 4th edition, Graduate Texts in Mathematics, vol 173. Springer
8. Dittel A, Fügenschuh A, Göttlich S, Herty M (2009) MIP presolve techniques for a PDE-based supply chain model. OMS 24(3):427–445
9. Dudzinski M, Rozgič M, Stiemer M (2018) oFEM: An object oriented package in MATLAB. Appl Math Comput 334:117–140
10. Frank M, Fügenschuh A, Herty M, Schewe L (2010) The coolest path problem. Networks and Heterogenous Media 5(1):143–162
11. Fügenschuh A, Martin A (2005) Computational integer programming and cutting planes. In: K Aardal RW G Nemhauser (ed) Handbooks in Operations Research and Management Science, Vol. 12, Discrete Optimization, Elsevier, Amsterdam, pp 69–121
12. Fügenschuh A, Herty M, Klar A, Martin A (2006) Combinatorial and continuous models and optimization for traffic flow on networks. SIOPT 16(4):1155–1176
13. Fügenschuh A, Göttlich S, Kirchner C, Herty M, Martin A (2009) Efficient reformulation and solution of a nonlinear PDE-controlled flow network model. Computing 85(3):245–265
14. Fügenschuh A, Gnegel F, Hagel M, Leyffer S, Stiemer M (2019) Mixed-Integer Partial Differential Equation Constrained Optimization. In: Liberti L, Sager S, Wiegele A (eds) Oberwolfach Reports, vol 16(2), to appear
15. Gnegel F, Dudzinski M, Fügenschuh A, Stiemer M (2018) Mixed integer pde constrained optimization for the control of a wildfire hazard. In: Operations Research Proceedings 2017: Selected Papers of the Annual International Conference of the German Operations Research Society (GOR), Freie Universiät Berlin, Germany, September 6-8, 2017, Springer, p 483

16. Göttlich S, Herty M, Klar A (2005) Network models for supply chains. Communications in Mathematical Sciences 3(4):545–559

17. Göttlich S, Fügenschuh A, Herty M, Klar A, Martin A (2008) A discrete optimization approach to large scale supply networks based on partial differential equations. SISC 30(3):1490–1507

18. Göttlich S, Kühn S, Ohst JP, Ruzika S, Thiemann M (2011) Evacuation dynamics influenced by spreading hazardous material. Networks and Heterogenous Media 6(3):443–464

19. Hadamard J (1902) Sur les problèmes aux dérivés partielles et leur signification physique. Princeton University Bulletin 13:49–52

20. Hahn M, Leyffer S, Zavala VM (2017) Mixed-integer PDE-constrained optimal control of gas networks. Preprint ANL/MCS-P7095-0817

21. Hinze M, Pinnau R, Ulbrich M, Ulbrich S (2009) Optimization with PDE Constraints. Springer

22. Klee V, Minty GJ (1972) How good is the Simplex algorithm. In: Shisha O (ed) Inequalities III, Academic Press, New York, NY, USA, pp 159–175

23. Ladyzhenskaya OA (1963) The Boundary Value Problems of Mathematical Physics. Springer-Verlag New York

24. Leugering G, Engell S, Griewank A, Hinze M, Rannacher R, Schulz V, Ulbrich M, Ulbrich S (2012) Constrained Optimization and Optimal Control for Partial Differential Equations, vol 160. Birkhäuser Basel

25. Leyffer S (2015) Mixed-integer PDE-constrained optimization. In: Liberti L, Sager S, Wiegele A (eds) Mixed-integer Nonlinear Optimization: A Hatchery for Modern Mathematics, Mathematisches Forschungsinstitut Oberwolfach, vol 46/2015, pp 2738–2740

26. Mandel J, Bennethum LS, Beezley JD, Coen JL, Douglas CC, Kim M, Vodacek A (2008) A wildland fire model with data assimilation. Mathematics and Computers in Simulation 79(3):584–606

27. Nemhauser G, Wolsey L (1999) Integer and Combinatorial Optimization. Wiley, Hoboken, NJ, USA

28. Noye BJ, Tan HH (1989) Finite difference methods for solving the two-dimensional advection–diffusion equation. International Journal for Numerical Methods in Fluids 9(1):75–98

29. Skutella M (2008) An introduction to network flows over time. In: Cook W, Lovász L, Vygen J (eds) Research Trends in Combinatorial Optimization, Springer-Verlag, Berlin, Heidelberg

30. Thomée V (1997) Galerkin Finite Element Methods for Parabolic Problems. Springer Series in Computational Mathematics