Mixed-Integer PDE-Constrained Optimization
GIAN Short Course on Optimization: Applications, Algorithms, and Computation

Pelin Cay, Bart van Bloemen Waanders, Drew Kouri, Anna Thuenen and Sven Leyffer

Lehigh University, Universität Magdeburg, Argonne National Laboratory, and Sandia National Laboratories

September 12-24, 2016
Outline

1 Introduction
   • Problem Definition and Killer Application
   • Theoretical and Computational Challenges

2 Early Numerical Results
   • MIPDECO & Branch-and-Bound
   • Eliminating the PDE and States
   • Control Regularization: Not All Norms Are Equal

3 Control of Heat Equation
   • Design and Operation of Actuators
   • Sum-Up Rounding Heuristic for Time-Dependent Controls

4 Conclusions
Mixed-Integer PDE-Constrained Optimization (MIPDECO)

PDE-constrained MIP ... \( u = u(t, x, y, z) \Rightarrow \text{infinite-dimensional!} \)
- \( t \) is time index; \( x, y, z \) are spatial dimensions

\[
\begin{align*}
\minimize_{u,w} & \quad \mathcal{F}(u, w) \\
\text{subject to} & \quad C(u, w) = 0 \\
& \quad u \in \mathcal{U}, \quad \text{and} \quad w \in \mathbb{Z}^P \, (\text{integers}),
\end{align*}
\]

- \( u(t, x, y, z) \): PDE states, controls, & design parameters
- \( w \) discrete or integral variables

MIPDECO Warning

\( w = w(t, x, y, z) \in \mathbb{Z} \) may be infinite-dimensional integers!
Mixed-Integer PDE-Constrained Optimization (MIPDECO)

PDE-constrained MIP ... \( u = u(t, x, y, z) \Rightarrow \text{infinite-dimensional!} \)

- \( t \) is time index; \( x, y, z \) are spatial dimensions

\[
\begin{align*}
\text{minimize} & \quad F(u, w) \\
\text{subject to} & \quad C(u, w) = 0 \\
& \quad u \in \mathcal{U}, \text{ and } w \in \mathbb{Z}^P \text{ (integers),}
\end{align*}
\]

- \( u(t, x, y, z) \): PDE states, controls, \& design parameters
- \( w \) discrete or integral variables

**MIPDECO Warning**

\( w = w(t, x, y, z) \in \mathbb{Z} \) may be infinite-dimensional integers!

The Darth Vader of Optimization!
Design of Ultra-Efficient Solar Cell

Design of non-reciprocal optical metamaterial for solar cells

Choose orientation of atoms and molecules to maximize energy
Design of Ultra-Efficient Solar Cell

Design of non-reciprocal optical metamaterial for solar cells

\[ \nabla \times \mathbf{H} = -i\omega (\chi \mathbf{H} + \epsilon \mathbf{E}) + \mathbf{J}_e, \]

\[ \nabla \times \mathbf{E} = i\omega (\mu \mathbf{H} + \zeta \mathbf{E}) + \mathbf{J}_m, \]

- Maxwell’s equation gives \( \mathbf{E} \) and \( \mathbf{H} \) electric and magnetic field.
- Objective is to maximize power inside solar cell (x space dims)

\[ \frac{1}{2} \int_{\omega} I_{\text{solar}}(\omega) \int_{V} \Re(\epsilon(x, w))|\mathbf{E}(x, w; \omega)|^2 + \Re(\mu(x, w))|\mathbf{H}(x, w; \omega)|^2 \, dV \, d\omega \]

- \( w_{i,j,k} = 1 \) if orientation \( i \) chosen on face \( j \) of molecule \( k \)
- \( w_{i,j,k} \) impact permittivities and permeabilities in Maxwell’s

\[ \tilde{\epsilon}_{j,k} = \sum_{i \in \mathcal{O}} w_{i,j,k} \epsilon_i \]
Mesh-Independent & Mesh-Dependent Integers

**Definition (Mesh-Independent & Mesh-Dependent Integers)**

1. The integer variables are mesh-independent, iff number of integer variables is independent on the mesh.

2. The integer variables are mesh-dependent, iff the number of integer variables depends on the mesh.

---

**Mesh-Independent**
- Manageable tree
- Theory possible

**Mesh-Dependent**
- Exploding tree
- Theory???
Theoretical Challenges of MIPDECO

Denis Ridzal (Yoda): Function space which \( w(x, y) \in \{0, 1\} \) in lies, you think?

- Consistently approximate \( w(x, y) \in \{0, 1\} \) as \( h \to 0 \)?
- Conjecture: \( \{w(x, y) \in \{0, 1\}\} \neq L_2(\Omega) \)
  ... e.g. binary support of Cantor set not integrable
- Likely need additional regularity assumptions

Coupling between Discretization & Integers

Discretization scheme (e.g. upwinding for wave equation) depends on direction of flow (integers).

- Application: gas network models with flow reversals
Computational Challenges of MIPDECO

- Approaches for **humongous branch-and-bound trees**
  ... e.g. 3D topology optimization with $10^9$ binary variables

- **Warm-starts** for PDE-constrained optimization (nodes)
  ... iterative Krylov (PDE) solve vs. rank-one updates (MIP)

- Guarantees for **nonconvex (nonlinear) PDE constraints**
  ... factorable programming approach hopeless for $10^9$ vars!

$$f(x_1, x_2) = x_1 \log(x_2) + x_2^3$$
Outline

1. Introduction
   - Problem Definition and Killer Application
   - Theoretical and Computational Challenges

2. Early Numerical Results
   - MIPDECO & Branch-and-Bound
   - Eliminating the PDE and States
   - Control Regularization: Not All Norms Are Equal

3. Control of Heat Equation
   - Design and Operation of Actuators
   - Sum-Up Rounding Heuristic for Time-Dependent Controls

4. Conclusions
Using My Favorite Lightsaber (Hammer)

Infinite-Dimensional MIPDECO

\[
\begin{align*}
\text{minimize} & \quad \mathcal{F}(u, w) \\
\text{subject to} & \quad \mathcal{C}(u, w) = 0 \\
& \quad u \in \mathcal{U}, \text{ and } w \in \mathbb{Z}^p \text{ integer,}
\end{align*}
\]

Discretize $\Rightarrow$ finite dim. MINLP:

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad c(x) \leq 0 \\
& \quad l \leq x \leq u, \quad x_i \in \mathbb{Z} \text{ for all } i \in I
\end{align*}
\]

Feel the force of AMPL/MINLP!

With abuse of notation ...

- $x$ discretized $(u, w)$
- $f(x)$ discretized $\mathcal{F}(u, w)$
- $c(x)$ discretized $\mathcal{C}(u, w)$
Source Inversion as MIP with PDE Constraints

Simple Example: Locate number of sources to match observation $\bar{u}$

\[
\begin{aligned}
\text{minimize} \quad & \mathcal{J} = \frac{1}{2} \int_{\Omega} (u - \bar{u})^2 d\Omega \\
\text{subject to} \quad & -\Delta u = \sum_{k,l} w_{kl} f_{kl} \quad \text{in } \Omega \\
& \sum_{k,l} w_{kl} \leq S \quad \text{and } w_{kl} \in \{0, 1\}
\end{aligned}
\]

with Dirichlet boundary conditions $u = 0$ on $\partial \Omega$.

E.g. Gaussian source term, $\sigma > 0$, centered at $(x_k, y_l)$

\[
f_{kl}(x, y) := \exp \left( \frac{-\| (x_k, y_l) - (x, y) \|^2}{\sigma^2} \right),
\]

Motivated by porous-media flow application to determine number of boreholes, [Ozdogan, 2004, Fipki and Celi, 2008]
Discretizing Poisson Equation

PDE (Poisson equation) in two dimensions \((x, y)\):

\[-\Delta u = f \iff -\frac{\partial^2 u}{\partial x^2}(x, y) - \frac{\partial^2 u}{\partial y^2}(x, y) = f(x, y)\]

Discretize PDE on finite mesh, e.g. for \((x, y) \in \Omega = [0, 1]^2\):

- Discretize \([0, 1]^2\) using mesh-size \(h > 0\), e.g. \(h = \frac{1}{N}\)
- Meshpoints: \((x_i, y_j) = (ih, jh)\) for \(i = 0, \ldots, N\), \(j = 0, \ldots, N\)
- Approximation of solution \(u_{i,j} \approx u(x_i, y_j) = u(ih, jh)\)

Second derivative central difference approximation

\[
\frac{\partial^2 u}{\partial x^2}(x_i, y_j) \approx \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{h^2}
\]

Similar for \(u_{yy} \approx \frac{(u_{i,j-1} - 2u_{i,j} + u_{i,j+1})}{h^2} \ldots\)
Source Inversion as MIP with PDE Constraints

Consider 2D example with $\Omega = [0, 1]^2$ and discretize PDE:
- 5-point finite-difference stencil; uniform mesh $h = 1/N$
- Denote $u_{i,j} \approx u(ih, jh)$ approximation at grid points

\[
\begin{align*}
\text{minimize} \quad & J_h = \frac{h^2}{2} \sum_{i,j=0}^{N} (u_{i,j} - \bar{u}_{i,j})^2 \\
\text{subject to} \quad & \frac{4u_{i,j} - u_{i,j-1} - u_{i,j+1} - u_{i-1,j} - u_{i+1,j}}{h^2} = \sum_{k,l=1}^{N} w_{kl} f_{kl}(ih, jh) \\
& u_{0,j} = u_{N,j} = u_{i,0} = u_{i,N} = 0 \\
& \sum_{k,l=1}^{N} w_{kl} \leq S \text{ and } w_{kl} \in \{0, 1\}
\end{align*}
\]

... finite-dimensional (convex) MIQP!
Mesh-Independent Source Inversion

Potential source locations (blue dots) on $16 \times 16$ mesh

Create target $\tilde{u}$ using red square sources
Source Inversion as MIP with PDE Constraints

Target (3 sources), reconstructed sources, & error on 32 × 32 mesh
Mesh-Independent Source Inversion: MINLP Solvers

Number of Nodes and CPU time for Increasing Mesh Sizes

- Number of Nodes independent of mesh size!
- MINLP & Minotaur: filterSQP runs out of memory for $N \geq 32$
- BonminOA takes roughly 100 iterations ... quadratic objective
Mesh-Dependent Source Inversion: MINLP Solvers

Number of Nodes and CPU time for Increasing Mesh Sizes

- Number of nodes & CPU time explodes with mesh size!
- OA <BREAK> after 130,000 seconds ... stress test for solvers!
MIPDECO trees are huge ... an overwhelming dark force!
MIPDECO Trick # 1: Eliminating the PDE

Discretized PDE constraint (Poisson equation)

\[
\frac{4u_{i,j} - u_{i,j-1} - u_{i,j+1} - u_{i-1,j} - u_{i+1,j}}{h^2} = \sum_{k,l} w_{kl} f_{kl}(ih,jh), \ \forall i,j
\]

\[\iff \quad Au = \sum w_{kl} f_{kl}, \text{ where } w_{kl} \in \{0, 1\} \text{ only appear on RHS!}\]

Elimination of PDE and states \(u(x,y,z)\)

- \(Au = \sum w_{kl} f_{kl} \iff u = A^{-1} \left( \sum w_{kl} f_{kl} \right) = \sum w_{kl} A^{-1} f_{kl}\)
- Solve \(n^2 \ll 2^n\) PDEs: \(u^{(kl)} := A^{-1} f_{kl}\)
- Eliminate \(u = \sum w_{kl} u^{(kl)}\)
MIPDECO Trick # 1: Eliminating the PDE

Eliminate \( u = \sum_{k,l} w_{kl} u^{(kl)} \) in MINLP:

\[
\begin{align*}
\minimize_{w} & \quad J_h = \frac{h^2}{2} \sum_{i,j=0}^{N} \left( \sum_{k,l} w_{kl} u^{(kl)}_{ij} - \bar{u}_{i,j} \right)^2 \\
\text{subject to} & \quad \sum_{k,l=1}^{N} w_{kl} \leq S \quad \text{and} \quad w_{kl} \in \{0, 1\}
\end{align*}
\]

- Eliminates the states \( u \) \((N^2 \text{ variables})\)
- Eliminates the PDE constraint \((N^2 \text{ constraints})\)

... generalizes to other PDEs (with integer controls on RHS)

Simplified model is quadratic knapsack problem
Elimination of States & PDEs: Source Inversion

CPU Time for Increasing Mesh Sizes: Simplified vs. Original Model

Eliminating PDEs is two orders of magnitude faster!
Control Regularization: Not All Norms Are Equal

Poisson with Distributed Control [OPTPDE, 2014] & [Tröltzsch, 1984]

\[
\begin{align*}
\text{minimize} & \quad \|u - u_d\|_{L^2(\Omega)}^2 + \int_{\Gamma} e_{\Gamma} \ u \ ds + \alpha \|w\|_{L^2}^2 \\
\text{subject to} & \quad -\Delta u + u = w + e_{\Omega} \quad \text{in} \ \Omega \\
& \quad \frac{\partial u}{\partial n} = 0 \quad \text{on boundary} \ \Gamma \\
& \quad w(x, y) \in \{0, 1\}
\end{align*}
\]

\(L^1\) or \(L^2\) regularization term for control \(w(x, y) \in \{0, 1\}\)?

Good Norms for MIPs

MIP’ers prefer polyhedral norms … promote integrality

- Old MIP trick: \(w^2 = |w|\) for \(w \in \{0, 1\}\)
  \(\Rightarrow \ L^1\)-norm same as \(L^2\)-norm on binary variables!
Not All Norms Are Equal

Consider **Distributed Control** for increasing mesh-size

<table>
<thead>
<tr>
<th>Mesh</th>
<th>Minotaur</th>
<th>B-BB</th>
<th>B-Hyb</th>
<th>B-OA</th>
</tr>
</thead>
<tbody>
<tr>
<td>8x8</td>
<td>0.04</td>
<td>0.80</td>
<td>2.54</td>
<td>126.81</td>
</tr>
<tr>
<td>16x16</td>
<td>6.61</td>
<td>72.21</td>
<td>1305.00</td>
<td><strong>Time</strong></td>
</tr>
<tr>
<td>32x32</td>
<td><strong>Time</strong></td>
<td><strong>Time</strong></td>
<td><strong>Time</strong></td>
<td><strong>Time</strong></td>
</tr>
</tbody>
</table>

$L^1$ regularization is equivalent to $L^2$, but faster. Many fewer nodes in tree-searches ⇒ solve up to $256 \times 256$. 

---

23 / 36
Not All Norms Are Equal

Consider Distributed Control for increasing mesh-size

<table>
<thead>
<tr>
<th>Mesh</th>
<th>CPU for $L^2$ Regularization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minotaur</td>
</tr>
<tr>
<td>8x8</td>
<td>0.04</td>
</tr>
<tr>
<td>16x16</td>
<td>6.61</td>
</tr>
<tr>
<td>32x32</td>
<td>Time</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mesh</th>
<th>CPU for $L^1$ Regularization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minotaur</td>
</tr>
<tr>
<td>8x8</td>
<td>0.03</td>
</tr>
<tr>
<td>16x16</td>
<td>0.11</td>
</tr>
<tr>
<td>32x32</td>
<td>0.18</td>
</tr>
</tbody>
</table>

- $L^1$ regularization is equivalent to $L^2$, but faster
- Many fewer nodes in tree-searches $\Rightarrow$ solve up to $256 \times 256$
Outline

1. Introduction
   - Problem Definition and Killer Application
   - Theoretical and Computational Challenges

2. Early Numerical Results
   - MIPDECO & Branch-and-Bound
   - Eliminating the PDE and States
   - Control Regularization: Not All Norms Are Equal

3. Control of Heat Equation
   - Design and Operation of Actuators
   - Sum-Up Rounding Heuristic for Time-Dependent Controls

4. Conclusions
Actuator Placement and Operation [Falk Hante]

Goal: Control temperature with actuators

- Select sequence of control inputs (actuators)
- Choose continuous control (heat/cool) at locations
- Match prescribed temperature profile

... “de-mist bathroom mirror with hair-drier”

Potential Actuator Locations \( l = 1, \ldots, L \)
Actuator Placement and Operation

Find optimal sequence of actuators, $w_l(t)$, and controls, $v_l(t)$:

\[
\begin{aligned}
\text{minimize}_{u,v,w} & \quad \|u(t_f, \cdot)\|_\Omega^2 + 2\|u\|_T \times \Omega^2 + \frac{1}{500}\|v\|_T^2 \\
\text{subject to} & \quad \frac{\partial u}{\partial t} - \kappa \Delta u = \sum_{l=1}^{L} v_l(t)f_l \quad \text{in } T \times \Omega \\
& \quad w_l(t) \in \{0, 1\}, \quad \sum_{l=1}^{L} w_l(t) \leq W, \quad \forall t \in T \\
& \quad Lw_l(t) \leq v_l(t) \leq Uw_l(t), \quad \forall l = 1, \ldots, L, \quad \forall t \in T
\end{aligned}
\]

where

\[
f_l(x, y) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(\frac{-\| (x, y) - (x_l, y_l) \|^2}{2\sigma}\right)
\]

point-source for actuators at $(x_l, y_l)$
Actuator Placement and Operation

Solution of NLP relaxation on $32 \times 32$ mesh ...

... provides no useful information?
Actuator Placement and Operation

Solution of MIPDECO on $32 \times 32$ mesh ...

... implementable discrete control!
NLP Relaxations of MIPDECOs Can Be Useless

Relaxations useless

- Controls smeared out $w_t = \frac{1}{W}$
- No obvious rounding ...
  ... in fact, $\text{round}(w_t) = 0$
- Branch-and-bound is hopeless ...
  ... generate huge search tree
- Many identical subtrees
  $\Rightarrow$ exploit symmetry
  (orbital branching)

... can make heuristics work ...
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control \( w(t) \) independent of \((x, y, z)\) ...

\[
\begin{aligned}
\min_{u, w} & \quad \mathcal{F}(u, w) \\
\text{s.t.} & \quad \mathcal{C}(u, w) = 0, \quad u \in \mathcal{U}, \quad w(t) \in \{0, 1\}^p
\end{aligned}
\]

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let \( \tilde{w}_t \in [0, 1] \) continuous relaxation ... construct integral \( w_t \)
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control $w(t)$ independent of $(x, y, z)$ ...

$$\begin{cases} \text{minimize} & F(u, w) \\ \text{subject to} & C(u, w) = 0, \quad u \in \mathcal{U}, \ w(t) \in \{0, 1\}^p \end{cases}$$

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let $\tilde{w}_t \in [0, 1]$ continuous relaxation ... construct integral $w_t$

for $t = 1, \ldots, T$ do

- Compute rounding residual:

  $$r_t := \tilde{w}_t + \sum_{\tau=0}^{t-1} (\tilde{w}_\tau - w_t)$$

- Round: $w_t = \begin{cases} 1 \text{ if } r_t > \frac{1}{2} \\ 0 \text{ else} \end{cases}$

end
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control $w(t)$ independent of $(x, y, z)$ ...

\[
\begin{aligned}
\begin{cases}
\text{minimize} & \mathcal{F}(u, w) \\
\text{subject to} & C(u, w) = 0, \quad u \in \mathcal{U}, \ w(t) \in \{0, 1\}^p
\end{cases}
\end{aligned}
\]

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let $\tilde{w}_t \in [0, 1]$ continuous relaxation ... construct integral $w_t$

for $t = 1, \ldots, T$ do

Compute rounding residual:

\[
r_t := \tilde{w}_t + \sum_{\tau=0}^{t-1} (\tilde{w}_\tau - w_t)
\]

Round: $w_t = \begin{cases} 1 \text{ if } r_t > \frac{1}{2} \\ 0 \text{ else} \end{cases}$

end
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control $w(t)$ independent of $(x, y, z)$ ...

\[
\begin{aligned}
\minimize_{u, w} & \quad \mathcal{F}(u, w) \\
\text{subject to} & \quad \mathcal{C}(u, w) = 0, \quad u \in \mathcal{U}, \quad w(t) \in \{0, 1\}^p
\end{aligned}
\]

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let $\tilde{w}_t \in [0, 1]$ continuous relaxation ... construct integral $w_t$

\begin{algorithm}
\For{$t = 1, \ldots, T$} {
\begin{align*}
\text{Compute rounding residual:} \\
r_t & := \tilde{w}_t + \sum_{\tau=0}^{t-1} (\tilde{w}_\tau - w_t)
\end{align*}
\begin{align*}
\text{Round: } w_t & = \begin{cases} 
1 & \text{if } r_t > \frac{1}{2} \\
0 & \text{else}
\end{cases}
\end{align*}
end
\end{algorithm}
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control \( w(t) \) independent of \((x, y, z)\) ...

\[
\begin{aligned}
\text{minimize} \quad & \mathcal{F}(u, w) \\
\text{subject to} \quad & C(u, w) = 0, \quad u \in \mathcal{U}, \quad w(t) \in \{0, 1\}^p
\end{aligned}
\]

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let \( \tilde{w}_t \in [0, 1] \) continuous relaxation ... construct integral \( w_t \)

\[
\text{for } t = 1, \ldots, T \text{ do}
\]

Compute rounding residual:

\[
r_t := \tilde{w}_t + \sum_{\tau=0}^{t-1} (\tilde{w}_\tau - w_t)
\]

Round: \( w_t = \begin{cases} 1 & \text{if } r_t > \frac{1}{2} \\ 0 & \text{else} \end{cases} \)
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control $w(t)$ independent of $(x, y, z)$ ...

\[
\begin{aligned}
\begin{cases}
\text{minimize} & \mathcal{F}(u, w) \\
\text{subject to} & C(u, w) = 0, \quad u \in \mathcal{U}, \quad w(t) \in \{0, 1\}^p
\end{cases}
\end{aligned}
\]

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let $\tilde{w}_t \in [0, 1]$ continuous relaxation ... construct integral $w_t$

for $t = 1, \ldots, T$ do

Compute rounding residual:

$$r_t := \tilde{w}_t + \sum_{\tau=0}^{t-1} (\tilde{w}_\tau - w_t)$$

Round: $w_t = \begin{cases} 
1 & \text{if } r_t > \frac{1}{2} \\
0 & \text{else}
\end{cases}$

end
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control $w(t)$ independent of $(x, y, z)$ ...

\[
\begin{aligned}
& \text{minimize} \quad \mathcal{F}(u, w) \\
& \text{subject to} \quad C(u, w) = 0, \quad u \in \mathcal{U}, \quad w(t) \in \{0, 1\}^p
\end{aligned}
\]

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let $\tilde{w}_t \in [0, 1]$ continuous relaxation ... construct integral $w_t$

\begin{verbatim}
for $t = 1, \ldots, T$ do
    Compute rounding residual:
    \[ r_t := \tilde{w}_t + \sum_{\tau=0}^{t-1} (\tilde{w}_\tau - w_t) \]
    Round: $w_t = \begin{cases} 1 & \text{if } r_t > \frac{1}{2} \\ 0 & \text{else} \end{cases}$
end
\end{verbatim}
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control \( w(t) \) independent of \((x, y, z)\) ...

\[
\begin{align*}
\begin{cases}
\text{minimize} & \mathcal{F}(u, w) \\
\text{subject to} & C(u, w) = 0, \quad u \in \mathcal{U}, \; w(t) \in \{0, 1\}^p
\end{cases}
\end{align*}
\]

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let \( \tilde{w}_t \in [0, 1] \) continuous relaxation ... construct integral \( w_t \)

\[
\text{for } t = 1, \ldots, T \text{ do }
\]

Compute rounding residual:

\[
r_t := \tilde{w}_t + \sum_{\tau=0}^{t-1} (\tilde{w}_\tau - w_t)
\]

Round: \( w_t = \begin{cases} 1 \text{ if } r_t > \frac{1}{2} \\ 0 \text{ else} \end{cases} \)

end
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control $w(t)$ independent of $(x, y, z)$ ...

$$\begin{align*}
\min_{u, w} & \quad F(u, w) \\
\text{subject to} & \quad C(u, w) = 0, \quad u \in \mathcal{U}, \ w(t) \in \{0, 1\}^p
\end{align*}$$

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let $\tilde{w}_t \in [0, 1]$ continuous relaxation ... construct integral $w_t$

for $t = 1, \ldots, T$ do

Compute rounding residual:

$$r_t := \tilde{w}_t + \sum_{\tau=0}^{t-1} (\tilde{w}_\tau - w_t)$$

Round: $w_t = \begin{cases} 1 & \text{if } r_t > \frac{1}{2} \\ 0 & \text{else} \end{cases}$

end
Sum-Up Rounding Heuristics for MIPDECOs

MIPDECO with binary control $w(t)$ independent of $(x, y, z)$ ...

\[
\begin{align*}
\begin{cases}
\text{minimize} & \mathcal{F}(u, w) \\
\text{subject to} & \mathcal{C}(u, w) = 0, \quad u \in \mathcal{U}, \quad w(t) \in \{0, 1\}^p
\end{cases}
\end{align*}
\]

Generalize optimal-control sum-up-rounding [Sager et al., 2012] ...

Let $\tilde{w}_t \in [0, 1]$ continuous relaxation ... construct integral $w_t$

\begin{algorithm}
for $t = 1, \ldots, T$ do
    Compute rounding residual:
    \[
    r_t := \tilde{w}_t + \sum_{\tau=0}^{t-1} (\tilde{w}_\tau - w_t)
    \]
    Round: $w_t = \begin{cases} 1 \text{ if } r_t > \frac{1}{2} \\ 0 \text{ else} \end{cases}$
end
\end{algorithm}
Sum-Up Rounding vs. Simple Rounding

Simple rounding: \( w_t = \text{round}(\tilde{w}_t) \)

Sum-Up Rounding: \( 6.31 \rightarrow 6 \)

Simple Rounding: \( 6.31 \rightarrow 7 \)

Simple Rounding arbitrarily poor: \( \tilde{w}_t = 0.500001 \Rightarrow w_t = 1 \)

Sum-Up Rounding has guarantees on quality of bounds!
Consider 2D heat equation with Robin boundary control

- $\Omega = [0, 1]^2 \times [0, 2]$ discretized with $N = 8, 16, 32$ in space and $M = 16, 32, 64$ in time.
- MINLP solvers: Minotaur and Bonmin (BnB, Hyb, OA)
- Sum-Up-Rounding (knapsack): Two NLPs solved with IPOPT
- Two instances per mesh (different initial conditions & forcing)

<table>
<thead>
<tr>
<th>Mesh</th>
<th># Variables</th>
<th># Binary Vars</th>
<th># Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>8x8x16</td>
<td>2873</td>
<td>272</td>
<td>3094</td>
</tr>
<tr>
<td>16x16x32</td>
<td>13497</td>
<td>528</td>
<td>13926</td>
</tr>
<tr>
<td>32x32x64</td>
<td>82745</td>
<td>1040</td>
<td>83590</td>
</tr>
</tbody>
</table>
Results for Sum-Up Rounding

Consider 2D heat equation with Robin boundary control

- $\Omega = [0, 1]^2 \times [0, 2]$ discretized with $N = 8, 16, 32$ in space and $M = 16, 32, 64$ in time.
- MINLP solvers: Minotaur and Bonmin (BnB, Hyb, OA)
- Sum-Up-Rounding (knapsack): Two NLPs solved with IPOPT
- Two instances per mesh (different initial conditions & forcing)

<table>
<thead>
<tr>
<th>Mesh</th>
<th>Minotaur</th>
<th>B-BnB</th>
<th>B-Hyb</th>
<th>B-OA</th>
<th>SUR-k</th>
</tr>
</thead>
<tbody>
<tr>
<td>8x8x16</td>
<td>4660.4</td>
<td>4660.4</td>
<td>4660.4</td>
<td>Time</td>
<td>1.08</td>
</tr>
<tr>
<td>8x8x16</td>
<td>18240.4</td>
<td>18240.4</td>
<td>18240.4</td>
<td>18240.4</td>
<td>1.66</td>
</tr>
<tr>
<td>16x16x32</td>
<td>Time</td>
<td>Time</td>
<td>Time</td>
<td>Time</td>
<td>23.7</td>
</tr>
<tr>
<td>16x16x32</td>
<td>4333.4</td>
<td>Time</td>
<td>4332.6</td>
<td>Time</td>
<td>43.5</td>
</tr>
<tr>
<td>32x32x64</td>
<td>Time</td>
<td>Time</td>
<td>Time</td>
<td>Time</td>
<td>9297.5</td>
</tr>
<tr>
<td>32x32x64</td>
<td>Time</td>
<td>Time</td>
<td>Time</td>
<td>Time</td>
<td>2650.7</td>
</tr>
</tbody>
</table>
Results for Sum-Up Rounding

NLPs solve faster than MINLPs ... what about solution quality?
Results for Sum-Up Rounding

NLPs solve faster than MINLPs ... what about solution quality?

... most solutions within 10% of optimum!

<table>
<thead>
<tr>
<th>$N$</th>
<th>Low Bnd</th>
<th>Upp Bnd</th>
<th>Rel. Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>4660.4</td>
<td>4809.9</td>
<td>3.1%</td>
</tr>
<tr>
<td>8</td>
<td>18240.4</td>
<td>18838.6</td>
<td>3.2%</td>
</tr>
<tr>
<td>16</td>
<td>2483.6</td>
<td>2517.9</td>
<td>1.4%</td>
</tr>
<tr>
<td>16</td>
<td>4332.7</td>
<td>4840.1</td>
<td>10.5%</td>
</tr>
<tr>
<td>32</td>
<td>900.8</td>
<td>976.8</td>
<td>7.8%</td>
</tr>
<tr>
<td>32</td>
<td>1840.8</td>
<td>2560.5</td>
<td>28.1%</td>
</tr>
</tbody>
</table>

...most solutions within 10% of optimum!
Conclusions

Mixed-Integer PDE-Constrained Optimization (MIPDECO)
- Class of challenging problems with important applications
  - Subsurface flow: oil recovery or environmental remediation
  - Design of next-generation solar cells
- Classification: mesh-dependent vs. mesh-independent
- On-going work: Building AMPL library of test problems
  ... formulation matters: interplay of binary and continuous
- Discretized PDEs ⇒ huge MINLPs ... push solvers to limit
- Elimination of PDE and state variables $u(t, x, y, z)$
- Sum-up rounding heuristics can be generalized

Outlook and Extensions
- Consider multi-level in space (network) and time
- Move toward truly multi-level approach similar to PDEs
The Deathstar of Optimization Problems ...

Add nonlinearities, uncertainty, robustness ...


