inovex

Solving Hard Optimization Problems with Pyomo and HiGHS

A Practical Introduction

2025-08-21 EuroScipy Kraków

Florian Wilhelm

Dr. Florian Wilhelm

inovex • HEAD OF DATA SCIENCE

- @FlorianWilhelm
- FlorianWilhelm
- FlorianWilhelm.info
- florian.wilhelm@inovex.de

- Mathematical Modelling
- Modern Data Warehousing & Analytics
- Personalisation & RecSys
- Uncertainty Quantification & Causality
- Python Data Stack
- OSS Contributor & Creator of PyScaffold

inovex

inovex

is an innovation and quality-driven IT project house with a focus on digital transformation.

- Application Development (Web Platforms, Mobile Apps, Smart Devices and Robotics, UI/UX design, Backend Services)
- Data Management and Analytics (Business Intelligence, Big Data, Searches, Data Science and Deep Learning, Machine Perception and Artificial Intelligence)
- Scalable IT-Infrastructures (IT Engineering, Cloud Services, DevOps, Replatforming, Security)
- Training and Coaching (inovex Academy)

Using technology to inspire our clients.
And ourselves.

www.inovex.de

Berlin · Karlsruhe · Pforzheim · Stuttgart · München · Köln · Hamburg · Erlangen

Definition of Linear Programming (LP)

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{c}^T \boldsymbol{x} \\ \text{subject to} \quad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \in \mathbb{R}^n \end{aligned}$$

LPs can be solved with the Simplex algorithm in cubic on average but exponential number of steps in worst case scenario or interior point methods in about $O(n^{2.5})$.

Definition of Mixed Integer Programming (MIP)

$$
\begin{aligned}
\underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{c}^T \boldsymbol{x} \\
\text{subject to} \quad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\
& \boldsymbol{x} \in \mathbb{R}^n \\
& \boxed{x_j \in \mathbb{Z} \text{ for } j \in J}
\end{aligned}
$$

MIPs are NP-hard, i.e. complexity grows exponentially with $\boldsymbol{n}$
Dropping the integrality constraints, i.e. "linear relaxation", leads to an LP problem.

Special Cases of Mixed Integer Programming

The cost functional $f(x)$ can be

- linear, i.e. $c^T x$,
- quadratic, i.e. $x^T Q x + q^T x$,
- $\dots$

If all variables need to be integer, it is a (pure) **integer linear program (ILP, IP)**.

If all variables need to be 0 or 1 (binary, boolean), it is a **0 – 1 linear program**.

Typical fields of application

- **Assignment/Allocation Problem**
  How to schedule some talks in an agenda under some constraints like room sizes to optimize the conference experience?
- **Transportation Problem**
  Which supplier should deliver to which factories given some costs to satisfy each factory with minimal costs?
- **Shortest Path Problem**
  Given some graph with a cost on each edge, what is the shortest path from source to sink?
- **Maximum Flow Problem**
  Given some pipe system with a source and sink, what is the maximum flow through the system?

inovex

Solving a MILP

Two major method classes for solving MILPs:

- *Cutting plane methods*
- *Branch and bound methods*

Often combined as *Branch and Cut methods*

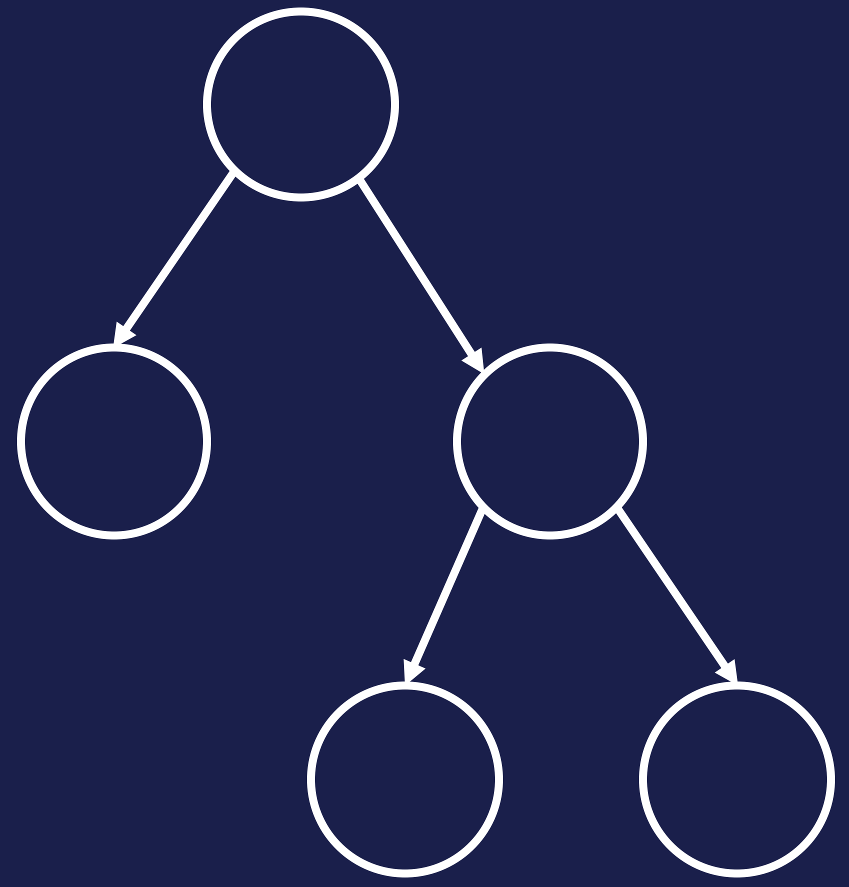
Source: Introduction to Optimization by Laurent Lessard, Spring 2017–18

Cutting Planes

Idea

- solve the LP relaxation problem
- while solution is not integral:
    - add constraint that excludes solution but no integer points
    - solve LP relaxation again

Branch & Bound

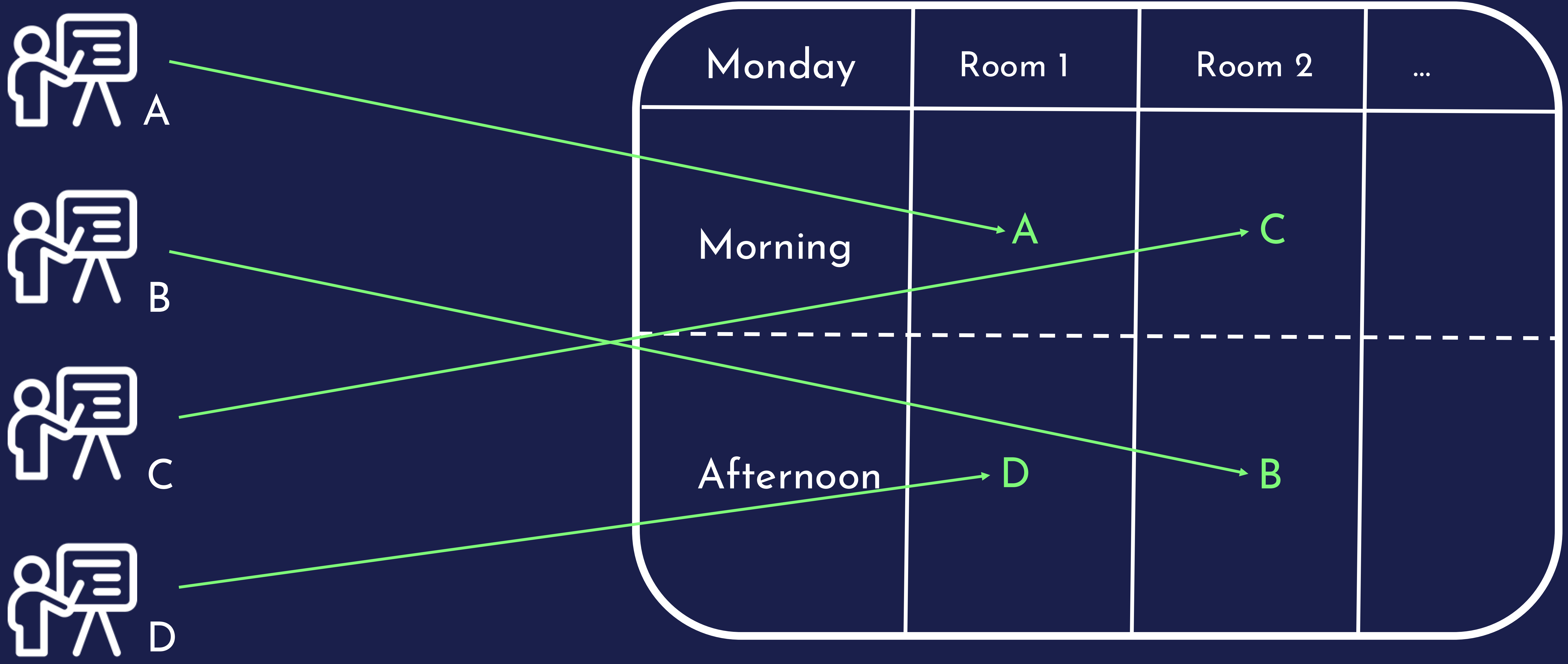


Idea

1. solve the relaxed LP and for a fractional x_i split into two subproblems
   - with constraint $x_i \leq \lfloor x_i \rfloor$
   - and with constraint $x_i \geq \lceil x_i \rceil$
2. repeat step 1. and build a tree of subproblems
3. eliminate branches of the tree using $f(x^{feasible}) \leq f(x^{optimal}) \leq f(x^{relaxed})$

Use-Case

How can we use MILPs for a Conference Scheduling?



Monday	Room 1	Room 2	...
Morning	A	C	
Afternoon	D	B	

Use-Case

What Constraints do we have?

- each talk must be assigned exactly once,
- each room/timeslot combination can only be occupied by one talk at most,
- the length of the timeslot must match the length of the talk
- some tutorials have part 1 & 2, thus need to be consecutive

Use-Case

What is our objective?

1. the **preferences** for day and time of some speakers, e.g. keynotes, need to be considered
2. **popularity** of a talk should be reflected in the room **capacity**,
3. avoid **parallel talks** that attract the same audience,
4. have in the same session, i.e. consecutive block of talks, the **same main track**, e.g. PyData vs. PyCon,
5. or even the **same sub track**, e.g. PyData: Data Handling,

Precedence: 1 > 2 > 3 > 4 > 5

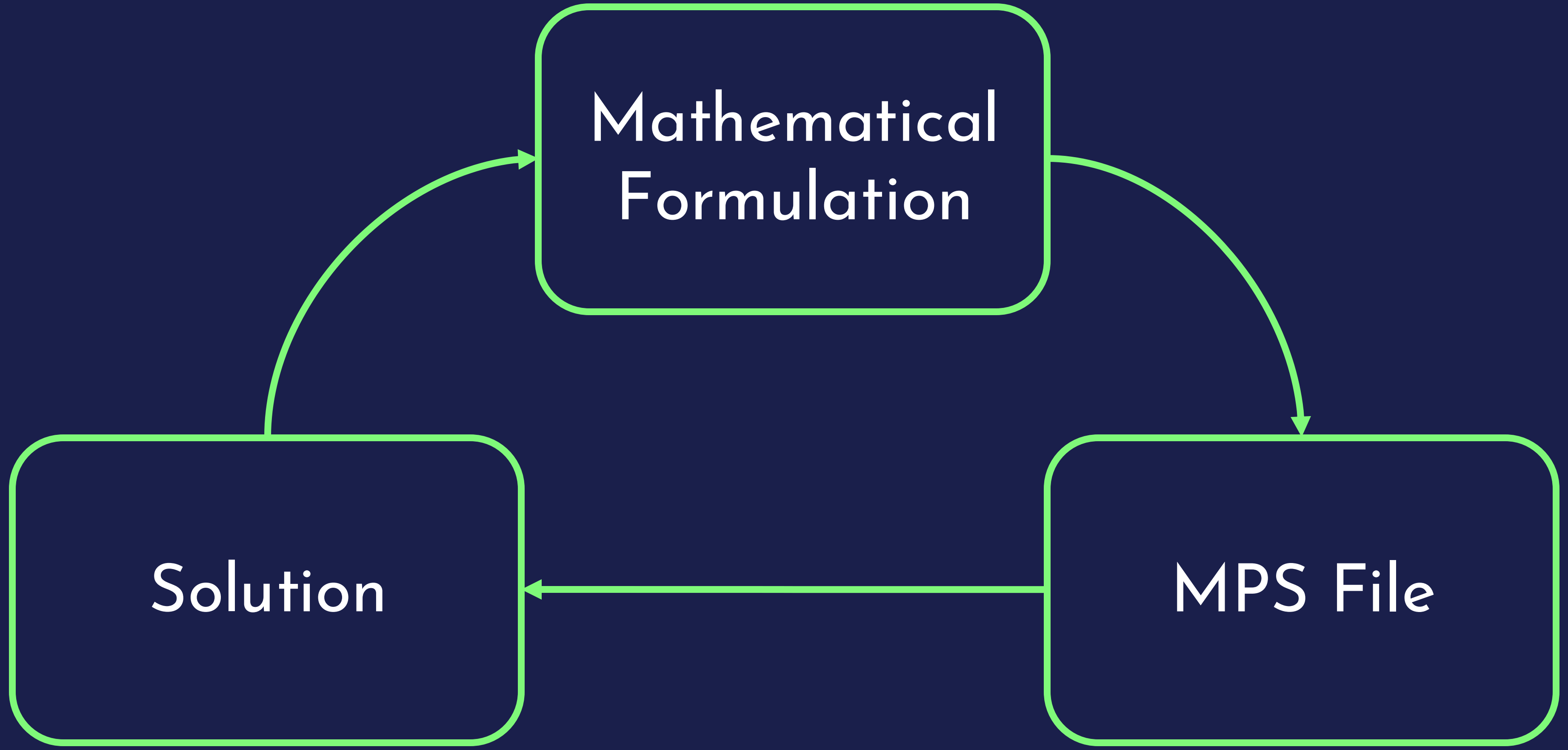
inovex

Use-Case

Solving MILPs in Python

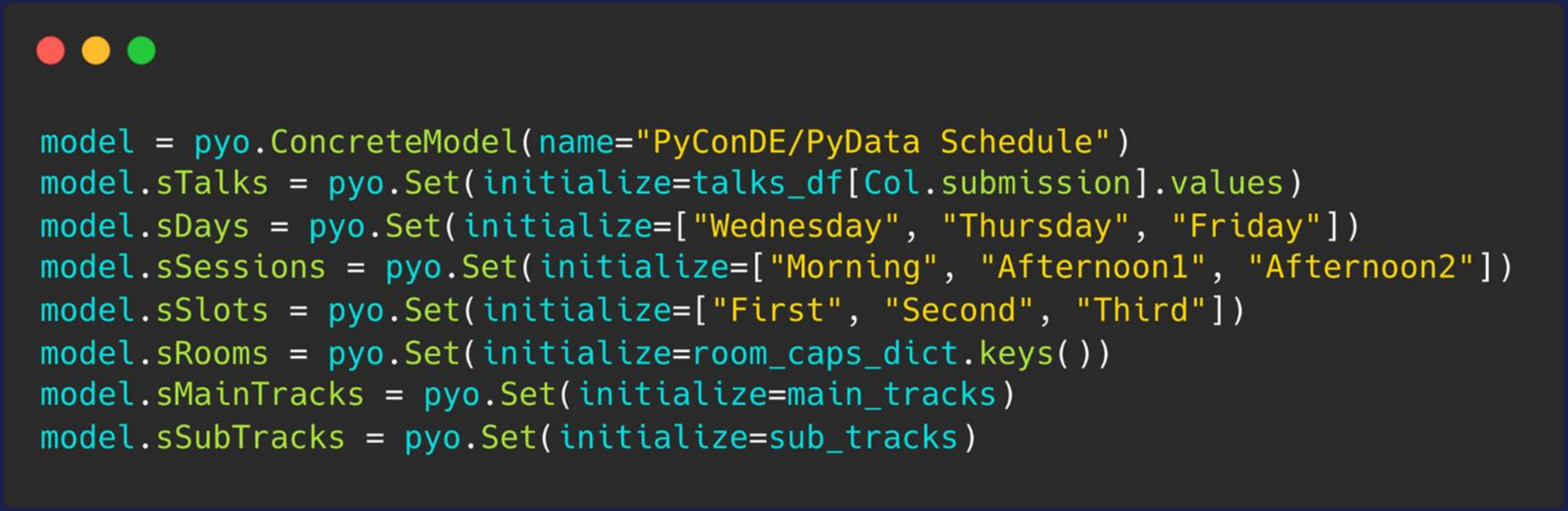
1. Framework to formulate the problem, e.g.
   - **a.Pyomo by Sandia NL**
   - b.PuLP

1. Solver to solve the canonical problem, e.g.
   - **a.HiGHS**
   - b.CBC
   - c.IPOPT



Use-Case

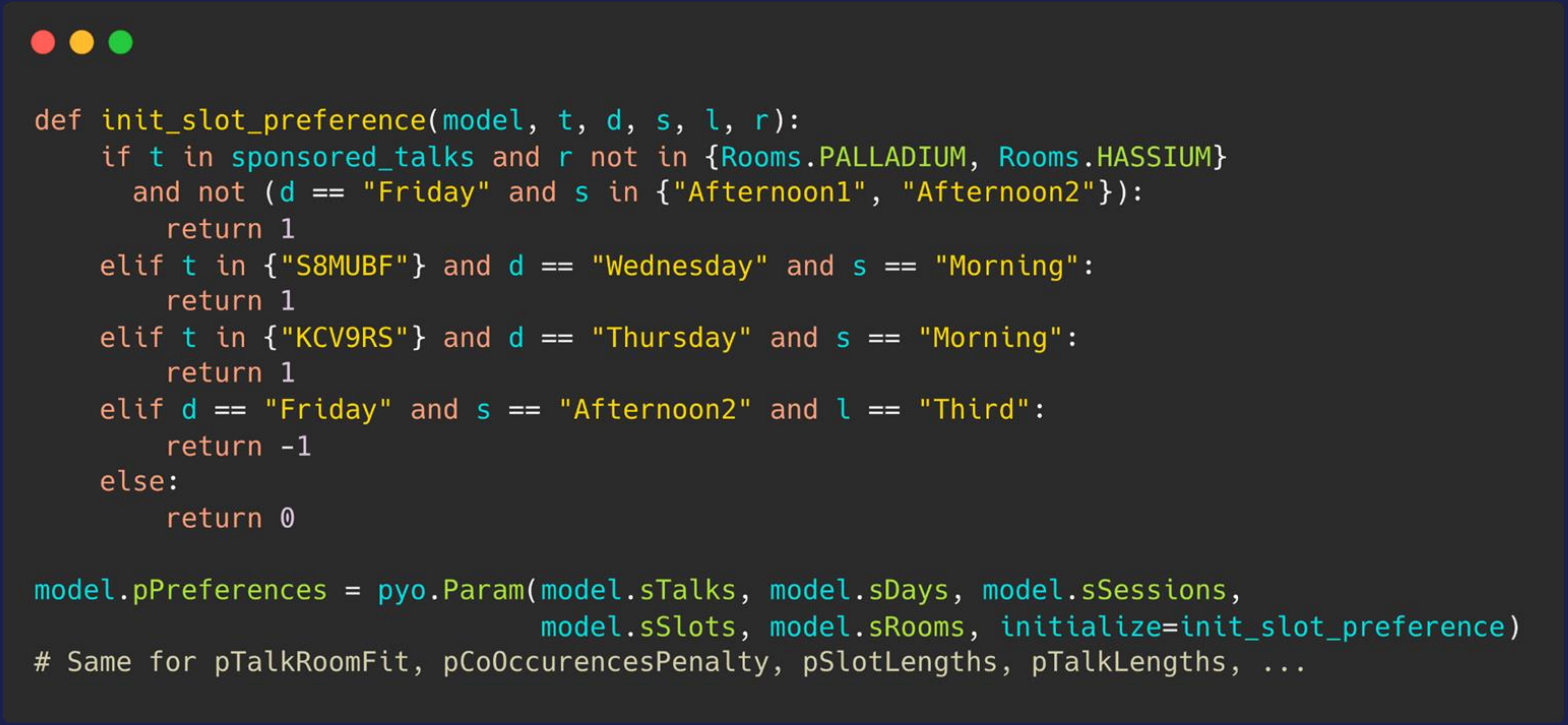
Pyomo: Index Sets



```
model = pyo.ConcreteModel(name="PyConDE/PyData Schedule")
model.sTalks = pyo.Set(initialize=talks_df[Col.submission].values)
model.sDays = pyo.Set(initialize=["Wednesday", "Thursday", "Friday"])
model.sSessions = pyo.Set(initialize=["Morning", "Afternoon1", "Afternoon2"])
model.sSlots = pyo.Set(initialize=["First", "Second", "Third"])
model.sRooms = pyo.Set(initialize=room_caps_dict.keys())
model.sMainTracks = pyo.Set(initialize=main_tracks)
model.sSubTracks = pyo.Set(initialize=sub_tracks)
```

Use-Case

Pyomo: Parameters



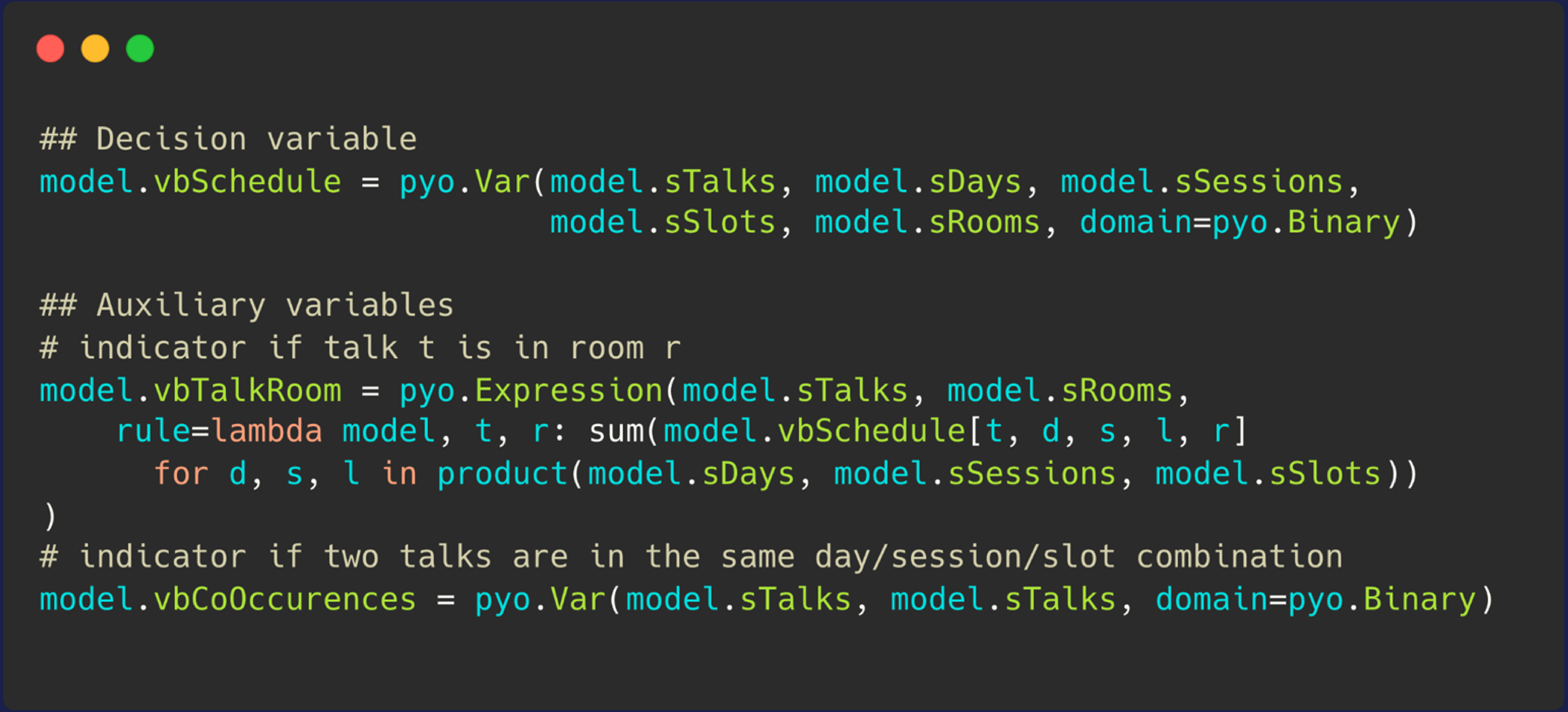
```python
def init_slot_preference(model, t, d, s, l, r):
    if t in sponsored_talks and r not in {Rooms.PALLADIUM, Rooms.HASSIUM}
      and not (d == "Friday" and s in {"Afternoon1", "Afternoon2"}):
        return 1
    elif t in {"S8MUBF"} and d == "Wednesday" and s == "Morning":
        return 1
    elif t in {"KCV9RS"} and d == "Thursday" and s == "Morning":
        return 1
    elif d == "Friday" and s == "Afternoon2" and l == "Third":
        return -1
    else:
        return 0

model.pPreferences = pyo.Param(model.sTalks, model.sDays, model.sSessions,
                               model.sSlots, model.sRooms, initialize=init_slot_preference)
# Same for pTalkRoomFit, pCoOccurencesPenalty, pSlotLengths, pTalkLengths, ...
```

inovex

Use-Case

Pyomo: Variables

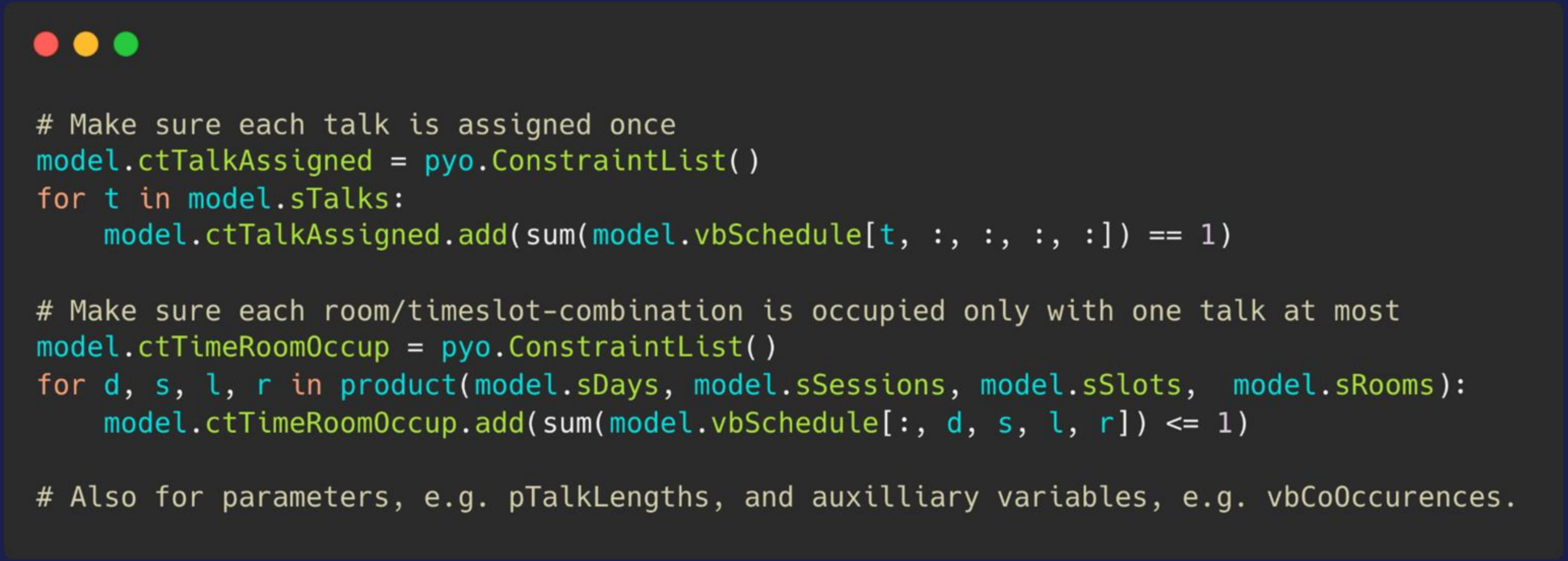


```
## Decision variable
model.vbSchedule = pyo.Var(model.sTalks, model.sDays, model.sSessions,
                           model.sSlots, model.sRooms, domain=pyo.Binary)

## Auxiliary variables
# indicator if talk t is in room r
model.vbTalkRoom = pyo.Expression(model.sTalks, model.sRooms,
    rule=lambda model, t, r: sum(model.vbSchedule[t, d, s, l, r]
      for d, s, l in product(model.sDays, model.sSessions, model.sSlots))
)
# indicator if two talks are in the same day/session/slot combination
model.vbCoOccurences = pyo.Var(model.sTalks, model.sTalks, domain=pyo.Binary)
```

Use-Case

Pyomo: Constraints



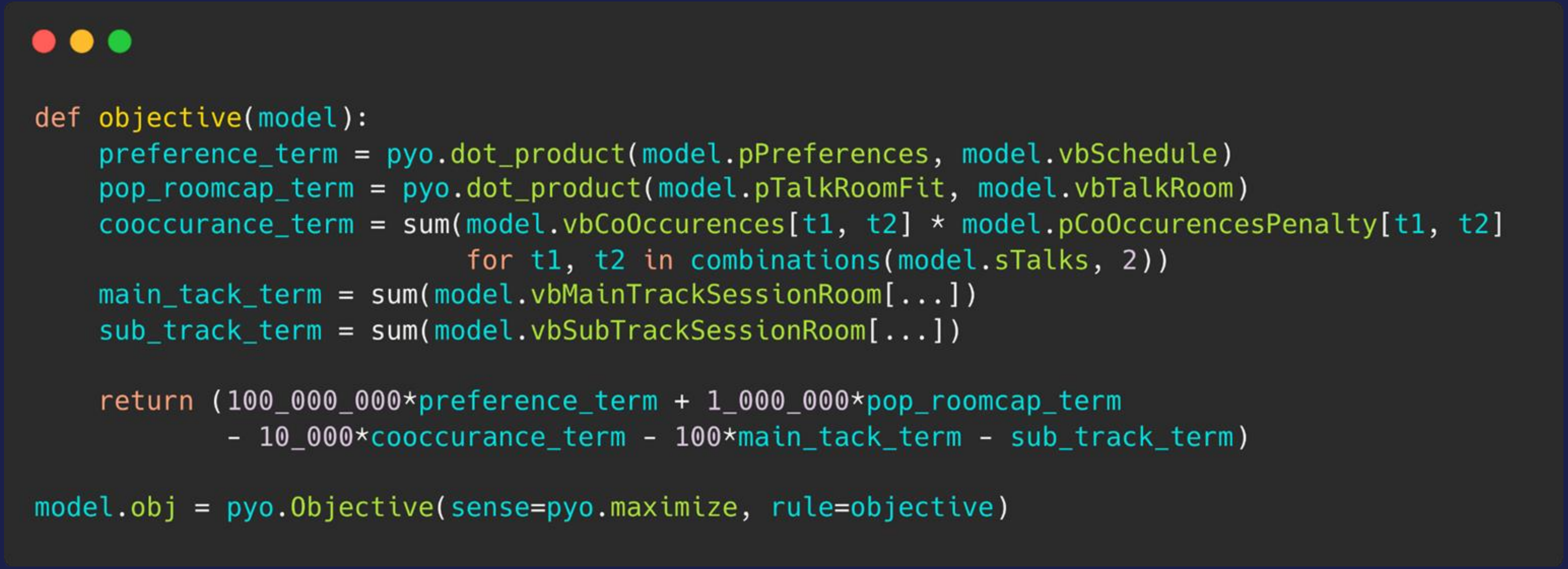
```
# Make sure each talk is assigned once
model.ctTalkAssigned = pyo.ConstraintList()
for t in model.sTalks:
    model.ctTalkAssigned.add(sum(model.vbSchedule[t, :, :, :, :]) == 1)

# Make sure each room/timeslot-combination is occupied only with one talk at most
model.ctTimeRoomOccup = pyo.ConstraintList()
for d, s, l, r in product(model.sDays, model.sSessions, model.sSlots,  model.sRooms):
    model.ctTimeRoomOccup.add(sum(model.vbSchedule[:, d, s, l, r]) <= 1)

# Also for parameters, e.g. pTalkLengths, and auxilliary variables, e.g. vbCoOccurences.
```

Use-Case

Pyomo: Objective



```
def objective(model):
    preference_term = pyo.dot_product(model.pPreferences, model.vbSchedule)
    pop_roomcap_term = pyo.dot_product(model.pTalkRoomFit, model.vbTalkRoom)
    cooccurance_term = sum(model.vbCoOccurences[t1, t2] * model.pCoOccurencesPenalty[t1, t2]
                           for t1, t2 in combinations(model.sTalks, 2))
    main_tack_term = sum(model.vbMainTrackSessionRoom[...])
    sub_track_term = sum(model.vbSubTrackSessionRoom[...])

    return (100_000_000*preference_term + 1_000_000*pop_roomcap_term
            - 10_000*cooccurance_term - 100*main_tack_term - sub_track_term)

model.obj = pyo.Objective(sense=pyo.maximize, rule=objective)
```

Use-Case

Check out the Full Source Code

Pytanis includes a Pretalx client and all the tooling you need for conferences using Pretalx, from handling the initial call for papers to creating the final program.

As the Pretalx API was completely changed, Pytanis needs an update.

Pyomo / HiGHS notebook:
https://github.com/PioneersHub/pytanis/blob/main/notebooks/pyconde-pydata-darmstadt-2025/40_scheduling_v1.ipynb

inovex

Summary

References

- **Introduction to Optimization** by Laurent Lessard
- **Mixed Integer Programming for time table scheduling** by Francisco Espiga
- **Schedule Optimisation using Linear Programming in Python** by Lewis Woolfson
- Some icons taken from **Flaticon.com**

inovex

EuroScipy 2025

Thank you!

Dr. Florian Wilhelm

Head of Data Science & Mathematical Modelling

florian.wilhelm@inovex.de

inovex.de

@inovexlife

@inovexgmbh

inovex