

Advances in Model-Based Optimization with **AMPL**

Filipe Brandão

filipe@ampl.com

AMPL Optimization Inc.

www.ampl.com – +1 773-336-AMPL

INFORMS 2023, Phoenix, Arizona – October 15-18, 2023, CC-North 231B,
Session TC54 – Optimization Modeling Software II, Tuesday, October 17, 12:45 – 1:00 pm

Formulating Models More Like You Think About Them

Describe an optimization problem

- In a form *you find natural or convenient*
- Using readily recognized expressions

Send it to a solver

- In a form *the solver will accept*
- Relying on the modeling software to translate

Get back a result

- In the form you originally used

Typical User Complaint

— — —
Thank you so much for replying.

Let me show my "if-then" constraint in a more clear way as follows:

```
set veh := {1..16 by 1};
```

```
param veh_ind {veh};  
param theory_time {veh};  
param UP := 400000;
```

```
var in_lane_veh {veh} integer >=1, <=2;  
var in_in_time {veh} >=0, <=UP;
```

Note that "in_lane_veh {veh}" are integer variables which equal 1 or 2, and "in_in_time {veh}" are continuous variables.

```
subject to IfConstr {i in 1..card(veh)-1, j in i+1..card(veh):  
  veh_ind[i] = veh_ind[j] and theory_time[i] <= theory_time[j]}:
```

```
  in_lane_veh[i] = in_lane_veh[j] ==> in_in_time[j] >= in_in_time[i] + l_veh/V;
```

When I run my program, there appears the following statement:

CPLEX 20.1.0.0: logical constraint _slogcon[1] is not an indicator constraint.

Typical Reply

To reformulate this model in a way that your MIP solver would accept,
you could define some more binary variables,

```
var in_lane_same {veh,veh} binary;
```

with the idea that $in_lane_same[i,j]$ should be 1 if and only if $in_lane_veh[i] = in_lane_veh[j]$.

Then the desired relation could be written as two constraints:

```
in_lane_veh[i] = in_lane_veh[j] ==> in_lane_same[i,j] = 1  
in_lane_same[i,j] = 1 ==> in_in_time[j] >= in_in_time[i] + l_veh/V;
```

The second one is an indicator constraint, but you would just need to replace the first one by equivalent linear constraints.

Given that in_lan_veh can only be either 1 or 2, those constraints could be

```
in_lane_same[i,j] >= 3 - in_lane_veh[i] - in_lane_veh[j]  
in_lane_same[i,j] >= in_lane_veh[i] + in_lane_veh[j] - 3
```

New Solver Interface Library (MP)

Design

- C++ library for building efficient, configurable solver drivers
- Support for features of current C interface library
- *Extensive toolset for problem recognition and transformation*

Motivation . . .





- AMPL has logical and “not linear” expressions
for *writing models the way you think of them*
- Old interfaces have very limited support for these
- New interfaces, built with MP,
allow these expressions to be used and combined freely

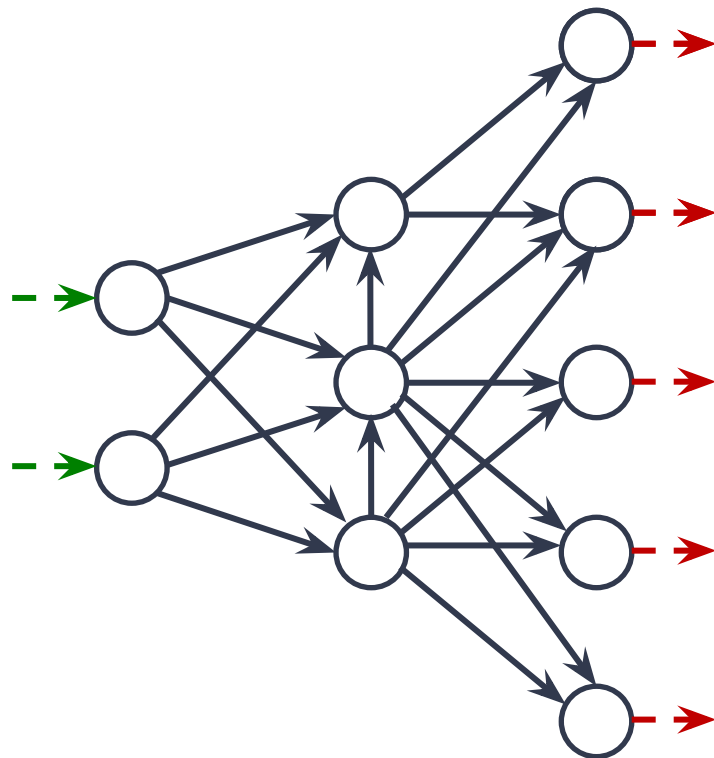
Example: Multi-Product Network Flow

Motivation

- Ship products efficiently to meet demands

Context

- a transportation network
 - Nodes  representing cities
 - arcs  representing roads
- supplies  at nodes
- demands  at nodes
- capacities on arcs
- shipping costs on arcs



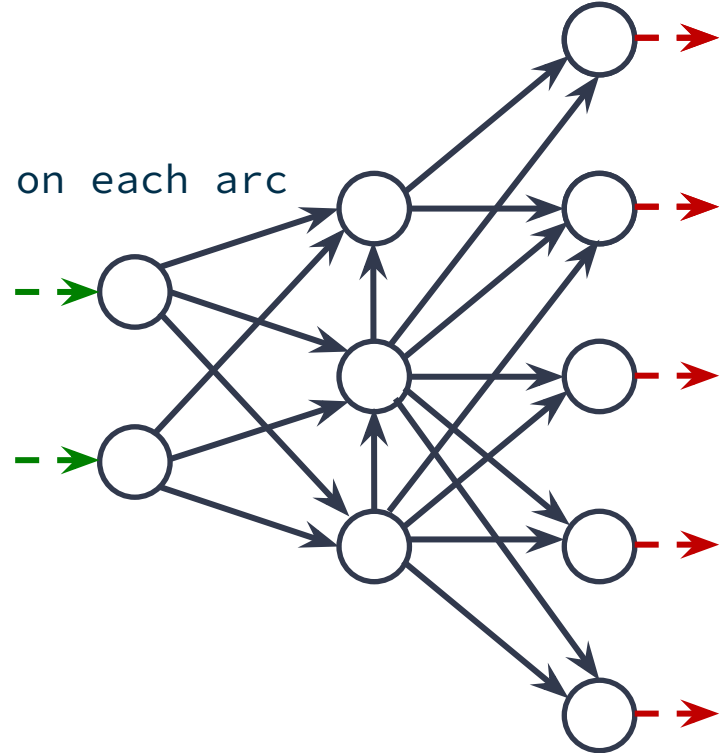
Example: Multi-Product Network Flow

Decide

- how much of each product to ship on each arc

So that

- shipping costs are kept low
- shipments on each arc respect capacity of the arc
- supplies, demands, and shipments are in balance at each node



AMPL Model for Multi-Product Network Flow

```
---  
  
set PRODUCTS;  
set NODES;  
param net_inflow {PRODUCTS,NODES};  
  
set ARCS within {NODES,NODES};  
param capacity {ARCS} >= 0;  
  
param var_cost {PRODUCTS,ARCS} >= 0;  
var Flow {PRODUCTS,ARCS} >= 0;  
  
minimize TotalCost:  
    sum {p in PRODUCTS, (i,j) in ARCS} var_cost[p,i,j] * Flow[p,i,j];  
  
subject to Capacity {(i,j) in ARCS):  
    sum {p in PRODUCTS} Flow[p,i,j] <= capacity[i,j];  
  
subject to Conservation {p in PRODUCTS, j in NODES):  
    sum {(i,j) in ARCS} Flow[p,i,j] + net_inflow[p,j] =  
    sum {(j,i) in ARCS} Flow[p,j,i];
```

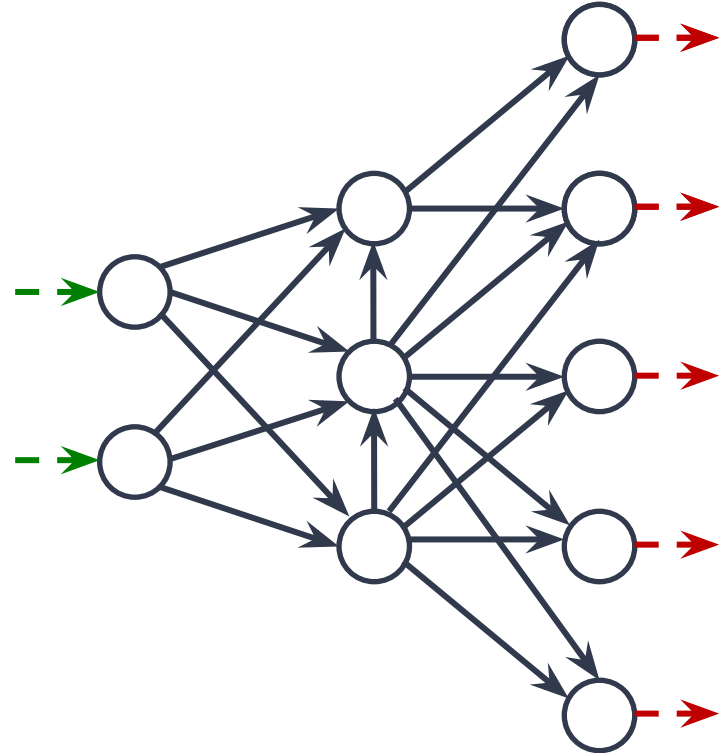

Example with conditions: Multi-Product Network Flow

Decide also

- whether to use each arc

So that

- variable costs plus fixed costs for shipping are kept low
- shipments are not too small
- not too many arcs are used



Positive Shipments Incur Fixed Costs

Linearization

```
param fix_cost {ARCS} >= 0;  
var Use {ARCS} binary;  
  
minimize TotalCost:  
    sum {p in PRODUCTS, (i,j) in ARCS} var_cost[p,i,j] * Flow[p,i,j] +  
    sum {(i,j) in ARCS} fix_cost[i,j] * Use[i,j];
```

How you think about it

```
param fix_cost {ARCS} >= 0;  
  
minimize TotalCost:  
    sum {p in PRODUCTS, (i,j) in ARCS} var_cost[p,i,j] * Flow[p,i,j] +  
    sum {(i,j) in ARCS}  
        if exists {p in PRODUCTS} Flow[p,i,j] > 0 then fix_cost[i,j];
```

Shipments Can't Be Too Small

Linearization

```
subject to Min_Shipment {(i,j) in ARCS}:  
    sum {p in PRODUCTS} Flow[p,i,j] >= min_ship * Use[i,j];  
  
subject to Capacity {(i,j) in ARCS}:  
    sum {p in PRODUCTS} Flow[p,i,j] <= capacity[i,j] * Use[i,j];
```

How you think about it

```
subject to Shipment_Limits {(i,j) in ARCS}:  
    sum {p in PRODUCTS} Flow[p,i,j] = 0 or  
    min_ship <= sum {p in PRODUCTS} Flow[p,i,j] <= capacity[i,j];
```

Can't Use Too Many Arcs

Linearization

```
subject to Max_Used:  
    sum {(i,j) in ARCS} Use[i,j] <= max_arcs;
```

How you think about it

```
subject to Limit_Used:  
    atmost max_arcs {(i,j) in ARCS}  
    (sum {p in PRODUCTS} Flow[p,i,j] > 0);
```

Linearization is Usually Not That Easy!

```
subject to IfConstr {i in 1..card(veh)-1, j in i+1..card(veh):
    veh_ind[i] = veh_ind[j] and theory_time[i] <= theory_time[j]}:
    in_lane_veh[i] = in_lane_veh[j]
    ==> in_in_time[j] >= in_in_time[i] + l_veh/V;
```

```
minimize total_fuelcost:
    sum{(i,j) in A} sum{k in V} X[i,j,k] *
        ((if H[i,k] <= 300 then dMor[i,j] else
            if H[i,k] <= 660 then dAft[i,j] else
            if H[i,k] <= 901 then dEve[i,j])) * 5 +
        (if H[i,k] <= 300 then tMor[i,j] else
            if H[i,k] <= 660 then tAft[i,j] else
            if H[i,k] <= 901 then tEve[i,j])) * 0.0504);
```

```
subject to NoPersonIsolated
    {l in TYPES['loc'], r in TYPES['rank'], j in 1..numberGrps}:
    sum {i in LOCRANK[l,r]} Assign[i,j] = 0 or
    sum {i in LOCRANK[l,r]} Assign[i,j] + sum {a in ADJACENT[r]} sum {i in LOCRANK[l,a]} Assign[i,j] >= 2;
```

Example: N-Queens

How can n queens be placed on an $n \times n$ chessboard so that no two of them attack each other?

Constraint **alldiff** enforces a set of integer variables to take distinct values. Using **alldiff**, we can model N-Queens as follows:

```
param n integer > 0; # N-queens
var Row {1..n} integer >= 1 <= n;
s.t. row_attacks: alldiff ({j in 1..n} Row[j]);
s.t. diag_attacks: alldiff ({j in 1..n} Row[j]+j);
s.t. rdiag_attacks: alldiff ({j in 1..n} Row[j]-j);
```



$$\begin{aligned} \text{Row}[1] &== \text{Row}[2] \\ 1 &== 1 \end{aligned}$$



$$\begin{aligned} \text{Row}[1]+1 &== \text{Row}[2]+2 \\ 3+1 &== 2+2 \end{aligned}$$



$$\begin{aligned} \text{Row}[1]-1 &== \text{Row}[2]-2 \\ 1-1 &== 2-2 \end{aligned}$$

Example: N-Queens (<https://colab.ampl.com>)

The screenshot shows a web browser window displaying a Google Colab notebook. The browser's address bar shows the URL: `colab.research.google.com/github/ampl/amplcolab/blob/master/authors/glebbelow/miscellaneous/nqueens.ipynb#scrollTo=jYDiBDD_7Kba`. The notebook interface includes a menu bar with options like 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. Below the menu, there are buttons for '+ Code', '+ Text', and 'Copy to Drive'. The notebook content is titled 'N-Queens' and includes a description: 'How can N queens be placed on an NxN chessboard so that no two of them attack each other?'. It also lists tags: 'amplpy, constraint-programming, highlights' and the author: 'Gleb Belov <gleb@ampl.com>'. The notebook contains two code cells. The first cell, executed at 8s, contains the command: `!pip install -q amplpy`. The second cell, executed at 12s, contains the code for Google Colab & Kaggle integration: `from amplpy import AMPL, tools; ampl = tools.ampl_notebook(modules=["highs"], # modules to install; license_uuid="default") # license to use`. At the bottom of the notebook, there is a message: 'Using default Community Edition License for Colab. Get yours at: <https://ampl.com/ce>. Licensed to AMPL Community Edition License for the AMPL Model Colaboratory (<https://colab.ampl.com>).'. The status bar at the bottom indicates '2s completed at 11:53 AM'.

Example: N-Queens (<https://colab.ampl.com>)

The screenshot shows a Google Colaboratory notebook titled "nqueens.ipynb". The browser address bar shows the URL: `colab.research.google.com/github/ampl/amplcolab/blob/master/authors/glebbelov/miscellaneous/nqueens.ipynb#scrollTo=jYDiBDD_7Kba`. The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a status bar (Cannot save changes). The notebook content is organized into sections:

- Modeling N-Queens with `alldiff`**

N-Queens: How can N queens be placed on an NxN chessboard so that no two of them attack each other?

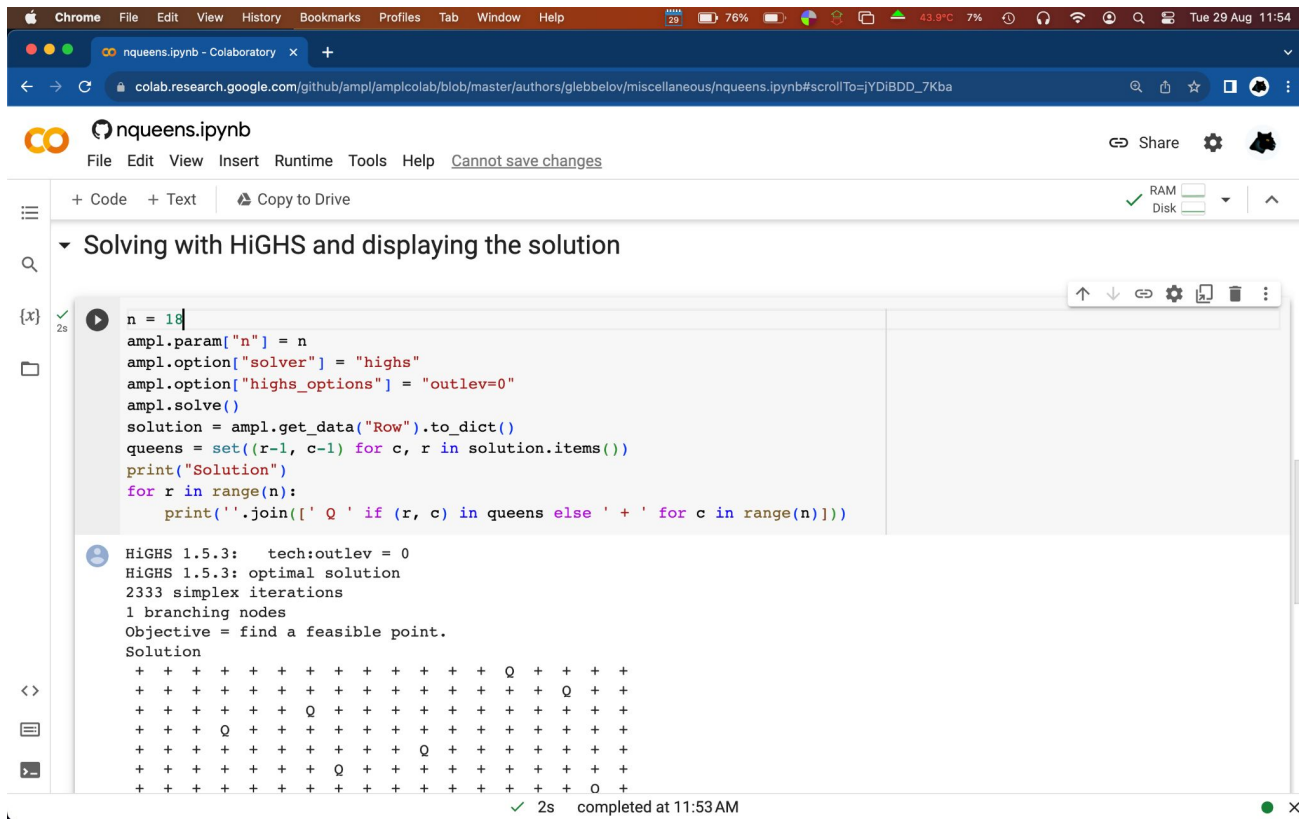
Constraint `alldiff` enforces a set of integer variables to take distinct values. Using `alldiff`, we can model N-Queens as follows:
- Solving with HiGHS and displaying the solution**

```
[3] %%ampl_eval
reset;
param n integer > 0; # N-queens
var Row {1..n} integer >= 1 <= n;
s.t. row_attacks: alldiff {(j in 1..n) Row[j]};
s.t. diag_attacks: alldiff {(j in 1..n) Row[j]+j};
s.t. rdiag_attacks: alldiff {(j in 1..n) Row[j]-j};
```

```
n = 18
ampl.param["n"] = n
ampl.option["solver"] = "highs"
ampl.option["highs_options"] = "outlev=0"
ampl.solve()
solution = ampl.get_data("Row").to_dict()
queens = set((r, c) for r, c in solution.items())
```

2s completed at 11:53 AM

Example: N-Queens (<https://colab.ampl.com>)



The screenshot shows a Google Colab notebook titled "nqueens.ipynb". The browser address bar shows the URL: colab.research.google.com/github/ampl/amplcolab/blob/master/authors/glebbelov/miscellaneous/nqueens.ipynb#scrollTo=jYDiBDD_7Kba. The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a status bar indicating "Cannot save changes".

The notebook content is titled "Solving with HiGHS and displaying the solution". It contains a code cell with the following Python code:

```
n = 14
ampl.param["n"] = n
ampl.option["solver"] = "highs"
ampl.option["highs_options"] = "outlev=0"
ampl.solve()
solution = ampl.get_data("Row").to_dict()
queens = set((r-1, c-1) for c, r in solution.items())
print("Solution")
for r in range(n):
    print(''.join([' Q ' if (r, c) in queens else ' + ' for c in range(n)]))
```

The output of the code cell shows the HiGHS solver results and the solution matrix:

```
HiGHS 1.5.3: tech:outlev = 0
HiGHS 1.5.3: optimal solution
2333 simplex iterations
1 branching nodes
Objective = find a feasible point.
Solution
+ + + + + + + + + + Q + + + +
+ + + + + + + + + + + + + Q + +
+ + + + + + Q + + + + + + + + +
+ + + Q + + + + + + + + + + + +
+ + + + + + + + + Q + + + + + +
+ + + + + + + + Q + + + + + + +
+ + + + + + + + + Q + + + + + +
+ + + + + + + + + + + + + + + O +
```

The status bar at the bottom of the code cell indicates "2s completed at 11:53 AM".

Example: N-Queens (<https://colab.ampl.com>)

The screenshot shows a Google Colaboratory notebook titled "nqueens.ipynb". The browser address bar shows the URL: colab.research.google.com/github/ampl/amplcolab/blob/master/authors/glebbelov/miscellaneous/nqueens.ipynb#scrollTo=jYDiBDD_7Kba. The notebook interface includes a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". A "Share" button and a "Cannot save changes" warning are visible. The code cell contains the following Python code:

```
print("Solution")
for r in range(n):
    print(''.join([' Q ' if (r, c) in queens else ' ' for c in range(n)]))
```

The output cell shows the following text:

```
HiGHS 1.5.3: tech:outlev = 0
HiGHS 1.5.3: optimal solution
2333 simplex iterations
1 branching nodes
Objective = find a feasible point.
Solution
+ + + + + + + + + + + Q + + + +
+ + + + + + + + + + + + + + + Q + +
+ + + + + + Q + + + + + + + + + +
+ + + Q + + + + + + + + + + + + +
+ + + + + + + + + + Q + + + + + +
+ + + + + + + + + + + + + + + + +
+ + + + + + + + + + + + + + + Q +
+ + + + + + + + Q + + + + + + + +
+ Q + + + + + + + + + + + + + + +
Q + + + + + + + + + + + + + + + +
+ + + + + + + + + + + + + + + + Q
+ + + + + + + + + + + + + + + Q + +
+ + + + + Q + + + + + + + + + + +
+ + + + + + + + + + + Q + + + + +
+ + + + + + + + + + + + + + + Q + +
+ Q + + + + + + + + + + + + + + +
```

The bottom status bar indicates the execution completed in 2 seconds at 11:53 AM.

Example: Recharging strategy for an electric vehicle (<https://mo-book.ampl.com/>)

MO-BOOK: Hands-On
Optimization with AMPL in Python

1. Mathematical Optimization
2. Linear Optimization
3. Mixed Integer Linear Optimization
4. Network Optimization

Recharging strategy for an electric vehicle

Contents

- Problem Statement
- Modeling
- Charging Station Information
- Route Information
- Car Information
- AMPL Model
- Suggested Exercises

```
param dist{SEGMENTS};
param t_lost;

# distance traveled
var x{LOCATIONS} >= 0, <= 10000;

# arrival and departure charge at each charging station
var c_arr{LOCATIONS} >= c_min, <= c_max;
var c_dep{LOCATIONS} >= c_min, <= c_max;

# arrival and departure times from each charging station
var t_arr{LOCATIONS} >= 0, <= 100;
var t_dep{LOCATIONS} >= 0, <= 100;

# arrival and departure rest from each charging station
var r_arr{LOCATIONS} >= 0, <= r_max;
var r_dep{LOCATIONS} >= 0, <= r_max;

minimize min_time: t_arr[n + 1];

s.t. drive_time {i in SEGMENTS}: t_arr[i] == t_dep[i-1] + dist[i]/v;
s.t. rest_time {i in SEGMENTS}: r_arr[i] == r_dep[i-1] + dist[i]/v;
s.t. drive_distance {i in SEGMENTS}: x[i] == x[i-1] + dist[i];
s.t. discharge {i in SEGMENTS}: c_arr[i] == c_dep[i-1] - R * dist[i];

s.t. recharge {i in STATIONS}:
# list of constraints that apply if there is no stop at station i
((c_dep[i] == c_arr[i] and t_dep[i] == t_arr[i] and r_dep[i] == r_arr[i])
or
# list of constraints that apply if there is a stop at station i
(t_dep[i] == t_lost + t_arr[i] + (c_dep[i] - c_arr[i])/C[i] and
c_dep[i] >= c_arr[i] and r_dep[i] == 0))
and not
((c_dep[i] == c_arr[i] and t_dep[i] == t_arr[i] and r_dep[i] == r_arr[i])
and
(t_dep[i] == t_lost + t_arr[i] + (c_dep[i] - c_arr[i])/C[i] and
c_dep[i] >= c_arr[i] and r_dep[i] == 0));
```

Example: Recharging strategy for an electric vehicle (<https://mo-book.ampl.com/>)

```
__ minimize min_time: t_arr[n + 1];

s.t. drive_time {i in SEGMENTS}: t_arr[i] == t_dep[i-1] + dist[i]/v;
s.t. rest_time {i in SEGMENTS}: r_arr[i] == r_dep[i-1] + dist[i]/v;
s.t. drive_distance {i in SEGMENTS}: x[i] == x[i-1] + dist[i];
s.t. discharge {i in SEGMENTS}: c_arr[i] == c_dep[i-1] - R * dist[i];

s.t. recharge {i in STATIONS}:
    # list of constraints that apply if there is no stop at station i
    ((c_dep[i] == c_arr[i] and t_dep[i] == t_arr[i] and r_dep[i] == r_arr[i])
    or
    # list of constraints that apply if there is a stop at station i
    (t_dep[i] == t_lost + t_arr[i] + (c_dep[i] - c_arr[i])/C[i] and
     c_dep[i] >= c_arr[i] and r_dep[i] == 0))
    and not
    ((c_dep[i] == c_arr[i] and t_dep[i] == t_arr[i] and r_dep[i] == r_arr[i])
    and
    (t_dep[i] == t_lost + t_arr[i] + (c_dep[i] - c_arr[i])/C[i] and
     c_dep[i] >= c_arr[i] and r_dep[i] == 0));
```

Supported Extensions

Operators and functions

- Conditional: **if-then-else**; **==>**, **<==**, **<==>**
- Logical: **or**, **and**, **not**; **exists**, **forall**
- Piecewise linear: **abs**; **min**, **max**; **<<breakpoints; slopes>>**
- Counting: **count**; **atmost**, **atleast**, **exactly**; **numberof**
- Comparison: **>**, **<**, **!<=**; **alldiff**
- Complementarity: **complements**
- Nonlinear: *****, **/**, **^**; **exp**, **log**; **sin**, **cos**, **tan**; **sinh**, **cosh**, **tanh**
- Set membership: **in**

Expressions and constraints

- High-order polynomials
- Second-order cones
- exponential cones (MOSEK driver!)

Supported Solvers

Solvers

- **Gurobi, Xpress, COPT, MOSEK**
- **HiGHS, CBC, SCIP, GCG**
- **CPLEX** soon

Modeling guide

- <https://mp.ampl.com/model-guide.html>

Examples using MP features

- <https://colab.ampl.com>
- <https://mo-book.ampl.com> (**NEW BOOK!**)

Small promo: Our main talk is right after this session!

Technology Tutorial, Tuesday, October 17, 2:55 – 3:30 pm

Location: **CC-North 120 D**

Python and AMPL: Build Prescriptive Analytics applications quickly with Pandas, Colab, Streamlit, and amply