Using AMPL inside C

Johnathan Rhyne

CU Denver

March 12, 2023

Table of Contents

What is AMPL?

AMPL Stands for **A** **M**athematical **P**rogramming **L**anguage.
In short, it is a relatively human readable way of asking a computer to solve optimization problems for us.

What is Sudoku?

Insert image of sudoku grid

Sudoku as an Integer Program

$$\sum_{i=1}^{9} x_{ijk} = 1, \qquad 1 \leq jk \leq 9 \qquad \text{(each number } k \text{ appears in each column)}$$

$$\sum_{j=1}^{9} x_{ijk} = 1, \qquad 1 \leq ik \leq 9 \qquad \text{(each number } k \text{ appears in each row)}$$

$$\sum_{q,r=0}^{2} x_{i+q,j+r,k} = 1, \qquad i,j = 1,4,7,\ 1 \leq k \leq 9 \qquad \text{(each } k \text{ appears once in each box)}$$

$$\sum_{k=1}^{9} x_{ijk} = 1, \qquad 1 \leq ij \leq 9 \qquad \text{(each cell contains exactly one number)}$$

$$x_{ijk} \in \{0,1\}, \qquad 1 \leq ijk \leq 9$$

$$x_{ijk} = 1, \qquad \text{When the initial board has number } k \text{ in cell } (i,j).$$

Use Case For External File Creation

- Hard problems
- Running on a cluster with no GUI or very high latency
- Only part of what you are trying to solve

Requirements

- AMPL installation accessible to your user
- A list of AMPL commands inside a text file

Example AMPL Files

```
set x := {1, 2, 3, 4, 5, 6, 7, 8, 9}; # x-coordinate inside our grid
set y := {1, 2, 3, 4, 5, 6, 7, 8, 9}; # y-coordinate inside our grid
set value := {1, 2, 3, 4, 5, 6, 7, 8, 9}; # value that is taken at coordinate (i,j)

# 3D variable where the value at grid[i,j,k] = 1
# if and only if the number k is present at the
# coordinate (i,j) inside our 9x9 grid
var grid {x,y,value} binary;
# Sudoku doesn't require an objective function
# So, we just add a function to let
# AMPL run. This can be anything
minimize dummyFunc: 0;

# Constraint that ensures we only have 1 number
# per column
subject to onePerColumn {(j,k) in {y,value}}:
    sum {i in x} grid[i,j,k] = 1;

# Constraint that ensures we only have 1 number
# per row
subject to onePerRow {(i,k) in {x,value}}:
    sum {j in y} grid[i,j,k] = 1;

# Set that helps with our onePerBox constraint
set offset = {1, 4, 7};

# Set that helps with our onePerBox constraint
set boundaries = {0,1,2};

# Constraint that ensures we only have 1 number
# per 3x3 box
subject to onePerBox {(i,j,k) in {offset, offset, value}}:
    sum {(a,b) in {boundaries,boundaries}} grid[i + a, j + b, k] = 1;

# Constraint that ensures we only have 1 number
# per cell in our grid
subject to onePerCell {(i,j) in {x, y}}:
    sum {k in value} grid[i,j,k] = 1
```

```
data;
# These following lines
# will be the given state of a sudoku grid
let grid[1,1,8] := 1;
let grid[1,5,2] := 1;
let grid[1,6,6] := 1;
let grid[2,7,7] := 1;
let grid[2,9,4] := 1;
let grid[3,4,7] := 1;
let grid[3,9,5] := 1;
let grid[4,4,1] := 1;
let grid[4,8,3] := 1;
let grid[4,9,6] := 1;
let grid[5,2,1] := 1;
let grid[5,5,8] := 1;
let grid[5,8,4] := 1;
let grid[6,1,9] := 1;
let grid[6,2,8] := 1;
let grid[6,6,3] := 1;
let grid[7,1,3] := 1;
let grid[7,6,1] := 1;
let grid[8,1,7] := 1;
let grid[8,3,5] := 1;
let grid[9,4,2] := 1;
let grid[9,5,5] := 1;
let grid[9,9,8] := 1;

end;
```

All Inside C: Key Things to Consider

- No official API like Java, Python, C++
- However, can do most of it through basic File I/O