Tutorial IV.IV - Advanced Solver Options with HiGHS in JuMP

Applied Optimization with Julia

Introduction

This Julia script is an interactive tutorial to introduce you to advanced solver options in HiGHS using JuMP. You'll learn about setting solver parameters to control the optimization process and improve performance. Follow the instructions and confirm your understanding with the examples. Make sure to have the JuMP package and the HiGHS solver installed to follow this tutorial.

```
using JuMP, HiGHS
```

Section 1: Setting HiGHS Solver Options

HiGHS offers various advanced options that can be set to control the solving process. To change some advanced settings, we first need to create a model with HiGHS.

```
model = Model(HiGHS.Optimizer)
```

```
A JuMP Model
  solver: HiGHS
  objective_sense: FEASIBILITY_SENSE
  num_variables: 0
  num_constraints: 0
  Names registered in the model: none
```

Different options for the HiGHS solver can be found here: HiGHS Options.

Note, that some options are solver specific and some options apply to all solvers, as JuMP itself can work with many different solvers just by changing the model definition and the solver-specific options.

```
# Set options for the solver
set_optimizer_attribute(model, "presolve", "on")
# Enables presolve, which transforms the model to potentially reduce its size or
 ↪  complexity.

set_optimizer_attribute(model, "time_limit", 60.0)
# Sets the time limit for the solver to 60 seconds. This is useful for large problems or
 ↪  when you need a solution within a specific timeframe.

set_optimizer_attribute(model, "mip_rel_gap", 0.0)
# Sets the relative MIP gap tolerance to 0.0, aiming for the exact optimal solution. In
 ↪  practice, a small non-zero value is often used as a trade-off between solution
 ↪  quality and solving time.

println("HiGHS solver options set successfully!")
```

```
HiGHS solver options set successfully!
```

Section 2: Understanding the Impact of Solver Options

Let's explore how these options can affect the solving process with a simple example:

```
# Create a simple mixed-integer programming model
model = Model(HiGHS.Optimizer)

@variable(model, 0 <= x <= 10, Int)
@variable(model, 0 <= y <= 10)

@objective(model, Max, x + y)

@constraint(model, x + 2y <= 15)

# Solve with default options
optimize!(model)
println("Default solution: x = ", value(x), ", y = ", value(y))
println("Solve time: ", solve_time(model), " seconds")

# Now, let's adjust some options
set_optimizer_attribute(model, "mip_rel_gap", 0.1)
set_optimizer_attribute(model, "time_limit", 1.0)

# Solve again with new options
optimize!(model)
println("Solution with new options: x = ", value(x), ", y = ", value(y))
println("Solve time: ", solve_time(model), " seconds")
```

```
Running HiGHS 1.7.2 (git hash: 5ce7a2753): Copyright (c) 2024 HiGHS under MIT licence terms
Coefficient ranges:
  Matrix [1e+00, 2e+00]
  Cost   [1e+00, 1e+00]
  Bound  [1e+01, 1e+01]
  RHS    [2e+01, 2e+01]
Presolving model
1 rows, 2 cols, 2 nonzeros  0s
0 rows, 0 cols, 0 nonzeros  0s
Presolve: Optimal

Solving report
  Status            Optimal
  Primal bound      12.5
```

```
  Dual bound        12.5
  Gap               0% (tolerance: 0.01%)
  Solution status   feasible
                    12.5 (objective)
                    0 (bound viol.)
                    0 (int. viol.)
                    0 (row viol.)
  Timing            0.00 (total)
                    0.00 (presolve)
                    0.00 (postsolve)
  Nodes             0
  LP iterations     0 (total)
                    0 (strong br.)
                    0 (separation)
                    0 (heuristics)
Default solution: x = 10.0, y = 2.5
Solve time: 0.00016887497622519732 seconds
Coefficient ranges:
  Matrix [1e+00, 2e+00]
  Cost   [1e+00, 1e+00]
  Bound  [1e+01, 1e+01]
  RHS    [2e+01, 2e+01]
Assessing feasibility of MIP using primal feasibility and integrality tolerance of       1e-06
Solution has               num          max          sum
Col     infeasibilities      0            0            0
Integer infeasibilities      0            0            0
Row     infeasibilities      0            0            0
Row     residuals            0            0            0
Presolving model
1 rows, 2 cols, 2 nonzeros  0s
0 rows, 0 cols, 0 nonzeros  0s
Presolve: Optimal

Solving report
  Status            Optimal
  Primal bound      12.5
  Dual bound        12.5
  Gap               0% (tolerance: 10%)
  Solution status   feasible
                    12.5 (objective)
                    0 (bound viol.)
                    0 (int. viol.)
                    0 (row viol.)
  Timing            0.00 (total)
                    0.00 (presolve)
                    0.00 (postsolve)
  Nodes             0
  LP iterations     0 (total)
                    0 (strong br.)
                    0 (separation)
                    0 (heuristics)
Solution with new options: x = 10.0, y = 2.5
Solve time: 0.00014408299466595054 seconds
```

This example demonstrates how changing solver options can affect the solution process, even for a simple problem. For more complex models, these options can have a significant impact on solve time and solution quality.

Conclusion

Well done! You've completed the tutorial on advanced solver options with HiGHS in JuMP. You've learned how to set advanced solver options. Continue to the next file to learn more.

Solutions

You will likely find solutions to most exercises online. However, I strongly encourage you to work on these exercises independently without searching explicitly for the exact answers to the exercises. Understanding someone else's solution is very different from developing your own. Use the lecture notes and try to solve the exercises on your own. This approach will significantly enhance your learning and problem-solving skills.

Remember, the goal is not just to complete the exercises, but to understand the concepts and improve your programming abilities. If you encounter difficulties, review the lecture materials, experiment with different approaches, and don't hesitate to ask for clarification during class discussions.

Later, you will find the solutions to these exercises online in the associated GitHub repository, but we will also quickly go over them in next week's tutorial. To access the solutions, click on the Github button on the lower right and search for the folder with today's lecture and tutorial. Alternatively, you can ask ChatGPT or Claude to explain them to you. But please remember, the goal is not just to complete the exercises, but to understand the concepts and improve your programming abilities.