

Notes Concerning bad instances for PH

David L. Woodruff

dlwoodruff@ucdavis.edu

1 Introduction

This note is *not* intended to be clear, self-contained or easy to read. Assume two stages. See `cooprc/src/coopr.pysp/coopr/pysp/tests/examples/test_model/twovarslack`

1.1 The Issue

There are instances with binaries for which PH using a static vector ρ that depends on the variable (and not on the scenario) fails to converge for a long time. The fix is not at all obvious. Magical ρ values depending on the variable and scenario could, of course, fix the problem but that is not helpful.

1.2 Background

- We are interested in problems with tens of thousands of variables and tens of thousands of constraints. Many of the variables are binary. We are interested in scores or hundreds of scenarios.
- We try to set ρ_i for variable x_i to be somehow proportional to the rate of change of the objective function with respect to x_i , $\delta f(x)/\delta x_i$. However, *we never directly compute this rate of change, as as a function of x* because that would be impractical for so many x over such large ranges (bear in mind that the rate of change for one x may depend on the value of many x).
- We use the language “scenario s wants a high (low) value of x_i ” to mean that in an optimal solutions that consider only scenario s , x_i will take values that are above (below) the average for all scenarios.

- The variables interact in complicated ways through the constraints, many of which span stages.

1.3 Slack Penalties

A particular form of the troubles concerns second stage slack variables with high penalties. Each scenario wants to configure its solutions so as to avoid paying the high slack cost and each scenario can do that if solved alone. However, the optimal solution of the SP involves some scenarios paying the high slack cost.

If you solve the SP, then you can see which scenarios need to pay the cost (i.e., have a high value of the slack variable) and you can give those scenarios high rho values on the primary variables and (with the right magical rho values) force convergence. However, you have to have the solution to make this work.

Remember the slack is in the second stage and has no W or ρ itself. In real examples, a lot of primary first stage variables contribute to each second stage constraint with a slack variable.

1.4 A simple example

The only first variables are binaries x_1 and x_2 . Second stage variables are scalars y and α .

1.4.1 Scenario Subproblem

Here is the model for each scenario s a very simple example:

$$\begin{aligned}
 & \text{minimize} && cy + M\alpha && \text{(P)} \\
 & \text{subject to:} && y \geq x_i && i = 1, 2 \\
 & && y \leq \sum_{i=1}^2 x_i \\
 & && ay + \alpha \leq b \\
 & && \alpha \in \{0, 1\} \\
 & && x_i \in \{0, 1\}, i = 1, 2
 \end{aligned}$$

Constraints 1 state that a scenario can get $y=one$ with anything, but for $y \leq zero$ it needs all x zero. Constraint 1 states that a scenario can get $y=zero$ with anything, but for

$y=1$, it needs at least one one; however, if it doesn't have at least one one, it has to have $y=0$.

1.4.2 Scenario 1 data

```
param c := 10 ;
param a := -1 ;
param b := 0 ;
param M := 1000 ;
```

Scenario 1 wants y to be small because it has positive cost. Also, $y < 0$ is infeasible, so $y = 0$ is best, which means it wants both x to be zero. Having $x > 0$ is not infeasible, but creates slack, so it is very expensive.

1.4.3 Scenario 2 data

```
param c := -10 ;
param a := 1 ;
param b := 1 ;
param M := 1100 ;
```

So scenario 2 really wants at least one x to be one so it can have $y = 1$ to avoid slack payments, but it is indifferent between the x_i for this purpose. (this is typical in UC problems where there are many generators with very similar costs).

The difference in M between scenarios is not important, but does make 1 for either or both x the optimal.

1.4.4 PH Performance

Assume equally likely scenarios (with just the wrong probabilities, things can be even worse). Without the slack penalties, ρ should be a fraction of $|c|$. So $\rho = 1$ would probably be OK.

With modest ρ : for scenario 2, PH oscillates between having one x_i be one and then the other. With a global ρ above 1000 PH solves the problem quickly (instantly for ρ of say 1100), but with anything 1000 or less, there is not convergence after 100 iterations.

2 Discussion

This is a highly stylized version of behavior that we have seen “in the wild.” This is not an emergency for UC because we can avoid this trouble by modeling around it; however, it is urgent because slack penalties are an extremely common modeling “trick” and anyway, it is not clear that requiring zero slack (i.e., have infeasibility instead of a large penalty) would make things much better for PH.

2.1 My Current Thinking

For full scale instances, we need to try to do something. The things we have done so far (high ρ) result in badly sub-optimal convergence. Thank goodness for the bounds!

1. I think that through a combination of annotations supplied by the modeler and diagnostics on W values, we can isolate families of first stage variables that are “part of the problem.”
2. We could then go to higher ρ values, perhaps guided by annotations or by diagnostics. However, it is not clear that this will work because often only a few scenarios need to force a slack payment.
3. Or we might have to branch on some of these variables (we could obtain pretty good W vectors then let DDSIP do some branching for us).
4. Want a wild and crazy idea? Try “branching” on scenario-specific ρ values (assume massive parallelism).

We are presently working on W diagnostics.