Background

# Learning when to use a decomposition,

Markus Kruber et al.

*original* MIP

$$\min\left\{c^t x,\ Ax \geq b,\ x \in \mathbb{Z}^n_+ \times \mathbb{Q}^q_+\right\}.$$ It can sometimes be re-arranged such that a particular structure:

$$\min c^t x$$
$$\text{s.t.} \begin{bmatrix} D^1 & & & & F^1 \\ & D^2 & & & F^2 \\ & & \ddots & & \vdots \\ & & & D^\kappa & F^\kappa \\ A^1 & A^2 & \cdots & A^\kappa & G \end{bmatrix} \cdot \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^\kappa \\ x^\ell \end{bmatrix} \geq \begin{bmatrix} b^1 \\ b^2 \\ \vdots \\ b^\kappa \\ b^\ell \end{bmatrix}$$
$$x \in \mathbb{Z}^n_+ \times \mathbb{Q}^q_+.$$

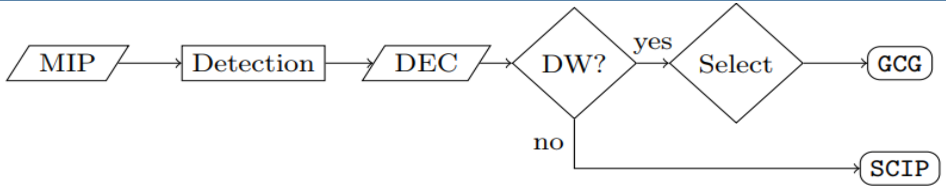
Such a re-arrangement is called a decomposition of the original MIP and finding it is called detection.

motivation

The MIP can be reformulated according to “best suited” decomposition, if the MIP structure can be automatically and finely detected. As a result, the solver may be much faster on the reformulated model than on the original one.

Dantzig-Wolfe (DW) reformulation is very efficient on solving specially structured MIPs. Successfully applying it may require a solid background, experience, and a non-negligible implementation effort. This paper proposes a supervised learning approach to decide whether or not a reformulation should be applied, and which decomposition to choose when several are possible. Preliminary experiments with a MIP solver equipped with this knowledge show a significant performance improvement on structured instances, with little deterioration on others.

algorithm




**SCIP** is a well-established MIP solver, **GCG** is its extension to make DW reformulation more accessible.

Their Supervised Learning Approach: They would like to learn an answer to the question: Given a MIP P, a DW decomposition D, and a time limit τ, will GCG using D optimally solve P faster than SCIP? They define a mapping ϕ that transforms a tuple (P; D; τ) into a vector of sufficient statistics or features ϕ(P; D; τ). Due to this mapping ϕ, the question above becomes a standard binary classification problem. Therefore a standard classifier f : {0,1} can be trained to solve this problem. Given an instance (P; D; τ), the quantity f ∘ ϕ(P; D; τ) is equal to one iff the predicted answer to the question above is positive. Practically, they built a database of SCIP and GCG runs for tuples (P; D; τ), a mapping ϕ, and trained classifiers f from the scikit-learn library on the instances ϕ(P; D; τ). Answers to the probabilistic versions g : [0, 1] of these classifiers can be interpreted as the probability that GCG using D outperforms SCIP if the time limit is τ.