Branch and price for submodular bin packing

Liding Xu

LIX CNRS, École Polytechnique

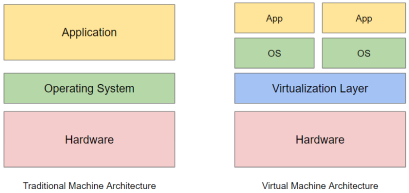
With: Claudia D'Ambrosio, Sonia Haddad Vanier, Emiliano Traversi

02/02/2023

Interests of research

Cloud computing is a over-competitive market.

Interests of research



Resource (memory, CPU, bandwidth, disk) allocation is key to success of cloud computing providers, e.g., AWS, GOOGLE, OVH, SCALEWAY, HETZNER, ALIBABA....

Research problem

On demand allocations of virtual machines: we want to preallocate virtual machines in a physical machine while considering uncertainty.

Some questions

- How to model the uncertainty? Adding some nonlinearity in the classical model.
- Is Dantzig-Wolfe reformulation applicable in mixed-integer nonlinear programming (MINLP)? Yes, it works.
- How is the practical quality of well-known heuristics/approximation algorithms for resource allocation? Good.

Overview

1. Branch-and-price for submodular binpacking
2. Dantzig-Wolfe reformulation
3. Conclusion

Table of Contents

1. Branch-and-price for submodular binpacking
2. Dantzig-Wolfe reformulation
3. Conclusion

Table of Contents

1 Branch-and-price for submodular binpacking

2 Dantzig-Wolfe reformulation

3 Conclusion

Dantzig-Wolfe reformulation and relaxation for MILP

Dantzig-Wolfe (DW): a reformulation technique for Mixed-Integer Linear Program (MILP) with decomposable structures.

- **Decomposition**: a main problem + a pricing sub problem, and the main problem has an huge LP relaxation.

Dantzig-Wolfe reformulation and relaxation for MILP

Dantzig-Wolfe (DW): a reformulation technique for Mixed-Integer Linear Program (MILP) with decomposable structures.

- **Decomposition**: a main problem + a pricing sub problem, and the main problem has an huge LP relaxation.
- **Main LP relaxation**: solved via column generation.
- **Pricing sub problem**: solved via pricing algorithm.
- **Integer DW reformulation**: solved via branch-and-price algorithm.

The performance of DW reformulation

- **Decomposition**: main problem (integer optimization) + sub problem.
- **the LP relaxation** of the main problem is somtimes called configuration LP.
- **Quality of relaxations**: stronger bounds than those of the naive LP relaxations.

Figure: Decomposable structures in coefficients matrix of an MILP

Submodular binpacking

Classical binpacking (BP): packing unsplittable items into a minimal number of bins (capacity usage is linear w.r.t. allocation of items).

- **Practical issue**: sizes of items are uncertain and unprovisioned?

Submodular binpacking

Classical binpacking (BP): packing unsplittable items into a minimal number of bins (capacity usage is linear w.r.t. allocation of items).

- **Practical issue**: sizes of items are uncertain and unprovisioned?
- **Motivating example**: all cloud computing providers function on virtualization techniques:
  - create virtual machines (items) on dedicated servers (bins);
  - receive many random requests for virtual machines;
  - some statistics on demands/data.

Modelling

We consider static but probabilistic models (assumption on data):

- BP with chance constraints;
- Distributionally robust BP.

These models admit an exact MINLP reformulation (nice for us, no need for sampling): **submodular binpacking**.

Submodular functions

Definition

A function $f : \{0,1\}^n \to \mathrm{R}$ is called a submodular function, if for every $x, y \in \{0,1\}^n$, $f(x) + f(y) \geq f(\max(x,y)) + f(\min(x,y))$, where $\max, \min$ are element-wise minimum and maximum.

Econ. explanation: you get less and less utility when paying more resources.

The submodular binpacking formulation

Submodular binpacking:

$$\min \quad \sum_{j \in \mathcal{M}} y_j, \tag{1}$$

$$\sum_{i \in \mathcal{N}} a_i v_{ij} + \sigma \sqrt{\sum_{i \in \mathcal{N}} b_i v_{ij}} \leq c y_j, \qquad \forall j \in \mathcal{M}, \tag{2}$$

$$\sum_{j \in \mathcal{M}} v_{ij} = 1, \qquad \forall i \in \mathcal{N}, \tag{3}$$

$$v_{ij} \in \{0, 1\}, \qquad \forall i \in \mathcal{N}, j \in \mathcal{M}, \tag{4}$$

$$y_j \in \{0, 1\}, \qquad \forall j \in \mathcal{M} \tag{5}$$

Our first contribution: DW reformulation.

DW decomposition: main problem

Let $\mathcal{P}$ be the set of all possible configurations of a bin. DW refomrulation = set cover formulation:

$$\min \quad \sum_{p \in \mathcal{P}} \lambda_p, \tag{6a}$$

$$\sum_{p \in \mathcal{P}} d_{ip} \lambda_p \geq 1, \qquad \forall i \in \mathcal{N}, \tag{6b}$$

$$\lambda_p \in \{0, 1\}, \qquad \forall p \in \mathcal{P}. \tag{6c}$$

DW decomposition: main problem

Let $\mathcal{P}$ be the set of all possible configurations of a bin. DW refomrulation = set cover formulation:

$$\min \quad \sum_{p \in \mathcal{P}} \lambda_p, \tag{6a}$$

$$\sum_{p \in \mathcal{P}} d_{ip} \lambda_p \geq 1, \qquad \forall i \in \mathcal{N}, \tag{6b}$$

$$\lambda_p \in \{0, 1\}, \qquad \forall p \in \mathcal{P}. \tag{6c}$$

$|\mathcal{P}|$ is huge, so solving the LP relaxation via pricing.

DW decomposition: pricing problem

Find a configuration which yields the steepest descent of the main problem:

$$v_{\text{prcing}} = \max \quad \sum_{i \in \mathcal{N}} \pi_i' x_i, \tag{7a}$$

$$\sum_{i \in \mathcal{N}} a_i' x_i + \sigma \sqrt{\sum_{i \in \mathcal{N}} b_i' x_i} \leq c, \tag{7b}$$

$$x_i \in \{0, 1\}, \qquad \forall i \in \mathcal{N}. \tag{7c}$$

DW decomposition: pricing problem

Find a configuration which yields the steepest descent of the main problem:

$$v_{\text{prcing}} = \max \quad \sum_{i \in \mathcal{N}} \pi'_i x_i, \tag{7a}$$

$$\sum_{i \in \mathcal{N}} a'_i x_i + \sigma \sqrt{\sum_{i \in \mathcal{N}} b'_i x_i} \leq c, \tag{7b}$$

$$x_i \in \{0, 1\}, \qquad \forall i \in \mathcal{N}. \tag{7c}$$

A convex MINLP: solvable by CPLEX.

Our second contribution: Tailored MILP-based branch-and-cut.

An exact pricing algorithm: initial MILP relaxation

$$\sum_{i\in\mathcal{N}} a'_i x_i + \sigma\sqrt{\sum_{i\in\mathcal{N}} b'_i x_i} \le c \equiv \sum_{i\in\mathcal{N}} b'_i x_i \le (c - \sum_{i\in\mathcal{N}} a'_i x_i)^2/\sigma^2.$$

1d piece-wise-linear concave overestimator for $q(w) := (c - w)^2$.

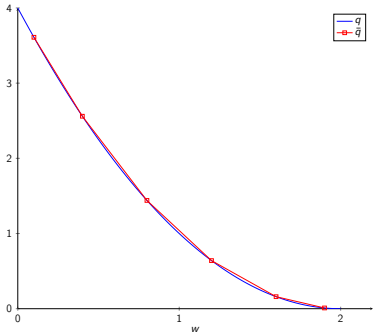


Figure: Graphs of the quadratic and its PWL over-estimator

An exact algorithm for pricing problem: Relaxation refining

Relaxation refining: posterior approximation, lazy cuts separated at infeasible binary points, via linearization of the convex function $\sum_{i\in\mathcal{N}} a'_i x_i + \sigma\sqrt{\sum_{i\in\mathcal{N}} b'_i x_i^2}$.

Combined together, an exact algorithm faster than CPLEX's branch-and-cut algorithm.

Hybrid pricing strategy

Our third contribution: a meta pricing scheduler that replace exact pricing algorithm via heuristic algorithms.
When can we avoid to use the expensive exact pricing algorithm? Some hints from the dual bound.

Dual bounds

Maser LP (MLP) (long time to converge):

$$\textbf{Dual bound } v_{\mathrm{MLP}} = \min \sum_{p \in \mathcal{P}} \lambda_p : \sum_{p \in \mathcal{P}} d_{ip} \lambda_p \geq 1, \forall i \in \mathcal{N}, \lambda_p \in [0,1], \forall p \in \mathcal{P}. \tag{8}$$

Dual bounds

Maser LP (MLP) (long time to converge):

$$\textbf{Dual bound } v_{\mathrm{MLP}} = \min \sum_{p \in \mathcal{P}} \lambda_p : \sum_{p \in \mathcal{P}} d_{ip} \lambda_p \geq 1, \forall i \in \mathcal{N}, \lambda_p \in [0,1], \forall p \in \mathcal{P}. \tag{8}$$

Restricted Maser LP (RMLP) (solved during column generation):

$$v_{\mathrm{RMLP}} = \min \sum_{p \in \mathcal{P}} \lambda_p : \sum_{p \in \mathcal{P}} d_{ip} \lambda_p \geq 1, \forall i \in \mathcal{N}, \lambda_p \in [0,1], \forall p \in \mathcal{P}' \subseteq \mathcal{P}. \tag{9}$$

$v_{\mathrm{MLP}} \leq v_{\mathrm{RMLP}}$.

Farley bound

Recall that v_{prcing} is the pricing objective value (a maximum problem). Then, $v_{\text{RMLP}}/v_{\text{prcing}} \leq v_{\text{MLP}} \leq v_{\text{RMLP}}$. Thus, $v_{\text{RMLP}}/v_{\text{prcing}}$ is also a lower bound (Farley bound) .

When exact pricing cannot yield better Farley bound

Assume that we have a good and fast pricing heuristic (which also generates a column), we run it first

Theorem

Let v_{heur} be the solution value of the pricing heuristic, let v_{RMLP} be the optimum of RMLP, and let v_{ld} be the current local dual bound for the master problem. If $\frac{v_{\mathrm{RMLP}}}{v_{heur}} \leq v_{\mathrm{ld}}$, the exact algorithm cannot yield a better local dual bound than v_{ld}.

Because $v_{heur} \geq v_{price}$, $\frac{v_{\mathrm{RMLP}}}{v_{heur}} \leq v_{\mathrm{ld}}$ implies that $\frac{v_{\mathrm{RMLP}}}{v_{price}} \leq v_{\mathrm{ld}}$ (no need for exact pricing).

Experiment: settings

Settings:

- Compact: compact formulation solved by CPLEX.
- DW: DW reformulation with pricing problem solved by CPLEX.
- DW+: DW reformulation with tailored pricing algorithm.
- DW-hybrid: DW+ with a hybrid strategy using heuristics or exact pricing algorithms.

Experiment: environment

- Randomly generated instances: number of items in $\{100, 400, 1000\}$.
- Intel Xeon @ 3.90GHz and 126GB memory, 3600 CPU second time limit.
- Every setting is given by a solution found by a greedy heuristic..

Experiment: results

Benchmarks	Solvers	Problem statistics			
		time (s)	dual gap (%)	primal improve(%)	#columns
`CloudSmall` ($\|\mathcal{N}\| = 100$)	BSOCP-BC	1452	15.8	0.0	-
	DW-BC	2129	11.4	0.9	1373
	DW-PWL	633	2.4	2.7	1869
	DW-Hybrid	330	2.0	3.4	3485
`CloudMedium` ($\|\mathcal{N}\| = 400$)	BSOCP-BC	3600	100.0	0.0	-
	DW-BC	3600	39.0	0.1	861
	DW-PWL	3600	17.2	0.4	3372
	DW-Hybrid	3600	11.8	0.6	6879
`CloudLarge` ($\|\mathcal{N}\| = 1000$)	BSOCP-BC	3600	100.0	0.0	-
	DW-BC	3600	59.6	0.0	741
	DW-PWL	3600	43.1	0.2	2105
	DW-Hybrid	3600	34.2	0.4	4257

Table: Aggregated statistics of the main computational results

Results

- **Experiment**: provides better dual bounds than CPLEX's bounds for compact formulation (13%+ improvement).
- **Conclusion**: greedy algorithms perform quite well (usually better than theoretical performance guarantee) (5% to optimality).

Related paper: Xu, D'Ambrosio, Haddad Vanier, and Traversi. "Branch and Price for Submodular Bin Packing", European journal on computational optimization, 2023.

Table of Contents

1 Branch-and-price for submodular binpacking

2 Dantzig-Wolfe reformulation

3 Conclusion

Conclusion

- In this paper, we explore DW reformulation for the submodular binpacking (MINLP);
- We propose several methods to generate columns efficiently;
- The results show that the DW reformulation is a promising method for the submodular binpacking (closing duality gap).

The end

Thank you for your attention!