Logic-based Benders Decomposition

A short Introduction

Martin Riedler

AC Retreat

Contents

Basic idea

Benders Decomposition (BD) is applicable to LPs having the following shape (S is the feasible set and D_x, D_y are the domains of x and y):

$$\min f(x, y) \tag{1}$$

$$(x, y) \in S \tag{2}$$

$$x \in D_x \tag{3}$$

$$y \in D_y \tag{4}$$

First compute a solution $\bar{y}$ w.r.t. the y-variables. Then solve the subproblem solely defined on the x-variables:

$$\min f(x, \bar{y}) \tag{5}$$

$$(x, \bar{y}) \in S \tag{6}$$

$$x \in D_x \tag{7}$$

Basic Technique

1. Fix part of the variables (master-problem)

$\Rightarrow$ split larger problem into easier solvable sub-problem(s)

2. Solve sub-problem(s)
3. Add Benders cut(s) to master-problem
4. Go to 1 if cuts have been added
5. Optimal solution found

Logic-based Benders Decomposition (LBBD)

- Classical BD [Benders62]
  - − Sub-problems need to be LPs
  - + Cuts can be derived via duality theory

Logic-based Benders Decomposition (LBBD)

- Classical BD [Benders62]
  - − Sub-problems need to be LPs
  - \+ Cuts can be derived via duality theory
- Logic-based BD [Hooker&Ottosson03]
  - \+ Sub-problems can also be IPs, CPs, ...
  - − No systematic way to identify Benders cuts
    - $\Rightarrow$ Requires problem specific cuts

Contents

A small example

Machine Scheduling (minimizing makespan) [Hooker07]

Definition

We consider:

- machines $I = \{1, \ldots, m\}$ with capacities C_i
- jobs $J = \{1, \ldots, n\}$ with
  - processing times p_{ij}
  - resource consumption c_{ij}
  - due dates d_j
- goal: Schedule all jobs on the machines s.t. due dates are not exceeded, resource limits are respected while minimizing the time by which the last job finishes (makespan).

A small example: ILP

Machine Scheduling (minimizing makespan) [Hooker07]

$$\min M \tag{8}$$

$$\sum_{i \in I} \sum_{t \in \mathbb{N}_0 : t \leq d_j - p_{ij}} (t + p_{ij}) x_{ijt} \leq M \qquad \forall j \in J \tag{9}$$

$$\sum_{i \in I} \sum_{t \in \mathbb{N}_0 : t \leq d_j - p_{ij}} x_{ijt} = 1 \qquad \forall j \in J \tag{10}$$

$$\sum_{j \in J} \sum_{t' \in \mathbb{N}_0 : t - p_{ij} < t' \leq t} c_{ij} x_{ijt'} \leq C_i \qquad \forall i \in I, \forall t \in \mathbb{N}_0 : t < T_{max} \tag{11}$$

$$x_{ijt} \in \{0, 1\} \qquad \forall i \in I, \forall j \in J, \; \forall t \in \mathbb{N}_0 : t \leq d_j - p_{ij} \tag{12}$$

A small example: Benders master problem

Machine Scheduling (minimizing makespan) [Hooker07]

$$\min M \tag{13}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{14}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{15}$$

- We only assign jobs to machines
- The rest is done in the subproblems (for each machine individually)

A small example: Benders subproblem

Machine Scheduling (minimizing makespan) [Hooker07]

Let J_i^h be the set of jobs assigned to machine i in iteration h. Then, the subproblem reads as follows:

$$\min M_{ih} \tag{16}$$

$$\sum_{t \in \mathbb{N}_0 : t \leq d_j - p_{ij}} (t + p_{ij}) x_{jt} \leq M_{ih} \qquad \forall j \in J_i^h \tag{17}$$

$$\sum_{j \in J_i^h} \sum_{t' \in \mathbb{N}_0 : t - p_{ij} < t' \leq t} c_{ij} x_{jt'} \leq C_i \qquad \forall t \in \mathbb{N}_0 : t < T_{max} \tag{18}$$

$$x_{jt} \in \{0, 1\} \qquad \forall j \in J_i^h, \; \forall t \in \mathbb{N}_0 : t \leq d_j - p_{ij} \tag{19}$$

A small example: Benders master problem revisited

Machine Scheduling (minimizing makespan) [Hooker07]

In iteration H the master problem becomes the following:

$$\min M \tag{20}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{21}$$

$$M^*_{ih} - \left(\sum_{j \in J^h_i} (1 - x_{ij}) p_{ij} + \Delta^h_i \right) \leq M \qquad \forall i \in I, h = 1, \ldots, H-1 \tag{22}$$

$$\frac{1}{C_i} \sum_{j \in J} c_{ij} p_{ij} x_{ij} \leq M \qquad \forall i \in I \tag{23}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{24}$$

$(\Delta^h_i = \max_{j \in J^h_i} \{d_j\} - \min_{j \in J^h_i} \{d_j\})$

Contents

When to consider LBBD?

- Sometimes, such formulations are somehow natural
  - Vehicle Routing (cluster-first route-second)
  - Scheduling (first assign jobs to machines, then solve per machine)
  - Cutting and Packing (first select items to pack, then find the packing)
- By decomposition the individual problems usually get smaller
  $\Rightarrow$ otherwise too large problems can be handled
- Sometimes decomposition allows to apply other algorithmic techniques for a certain part of the problem (e.g. CP)
- By dealing with the problem in two stages one can avoid modeling difficulties (e.g. quadratic constraints)

Metaheuristics and LBBD

- Metaheuristics have been applied with great success for solving the master problem.
- Metaheuristics can also be used for solving difficult subproblems, helping in deriving Benders cuts.
- Utilize a two-stage approach:
  - first solve master or subproblems heuristically
  - then complete verification steps to preserve optimality

Thank you for your Attention!

Questions?