Decomposition strategies for stochastic integer programming: challenges and perspectives

Fabricio Oliveira

Systems Analysis Laboratory
Department of Mathematics and Systems Analysis

Aalto University
School of Science

April 16, 2018

Collaborators

Fabricio Oliveira

Natashia Boland

Jeffrey Christiansen

Brian Dandurand

Andrew Eberhard

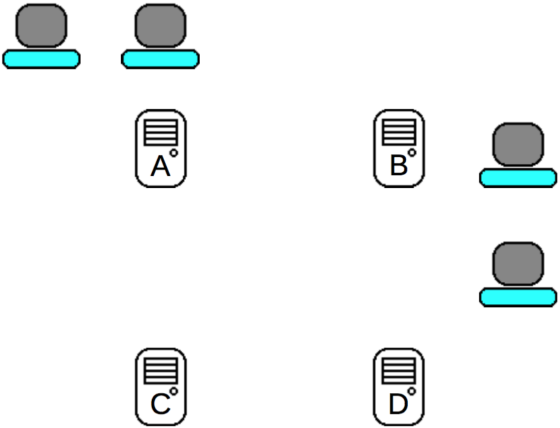
Jeff Linderoth

James Luedtke

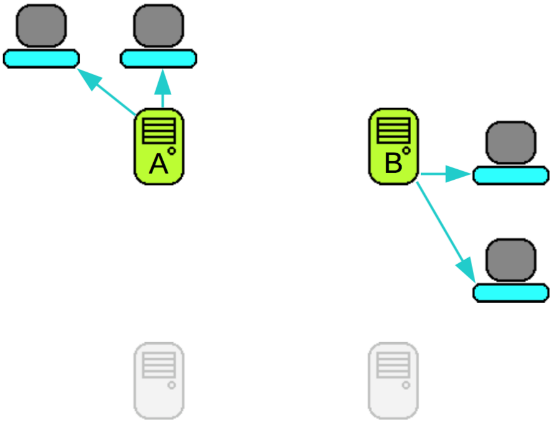
Outline of this talk

Outline of this talk

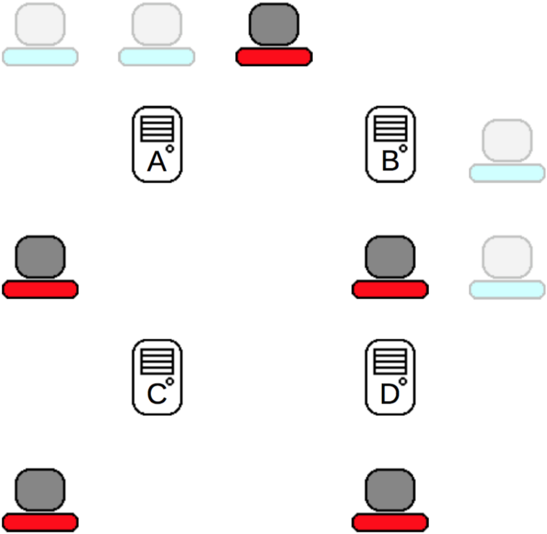
Introduction - Server Location Problem



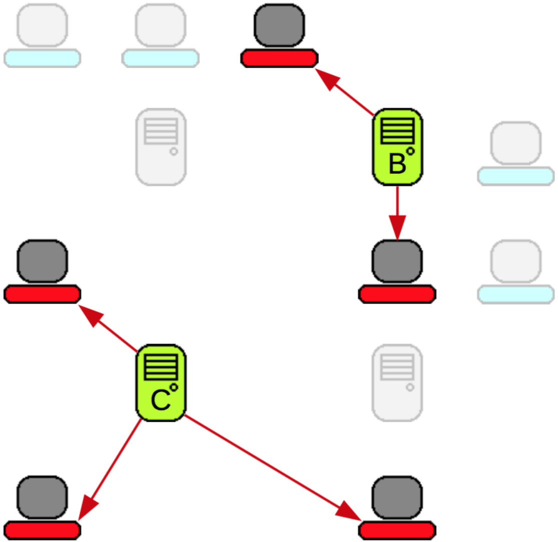
Introduction - Server Location Problem



Introduction - Server Location Problem

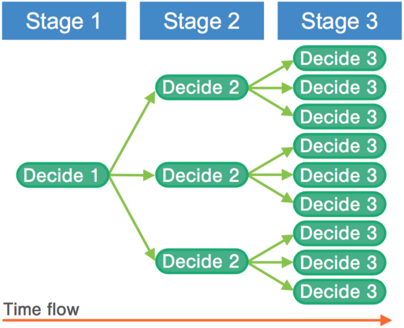


Introduction - Server Location Problem



Introduction

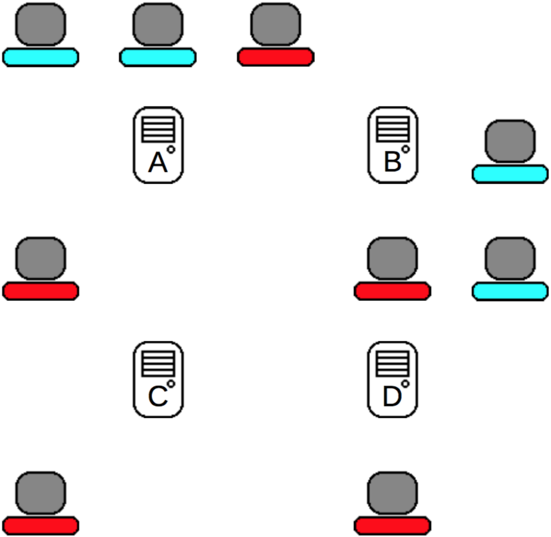
Stochastic Programming is a ramification of mathematical programming in which input parameters are considered uncertain.



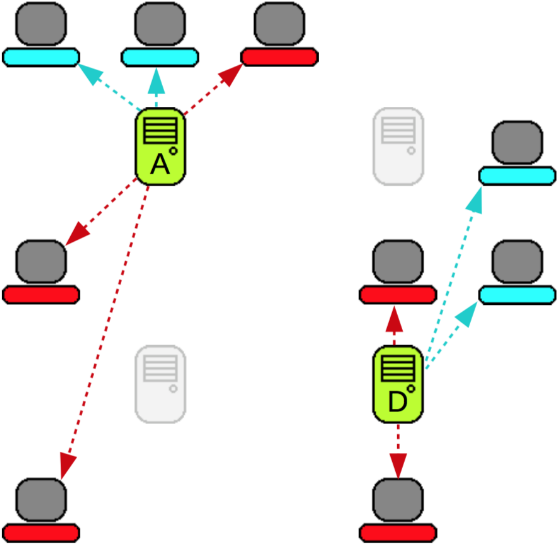
Modelling in this case involves:

1. Break time flow into points of interest; (stages)
2. Gather decisions that must be made at each point;
3. Explicitly represent possible realisation via scenarios

Introduction - Server Location Problem



Introduction - Server Location Problem



$$z^{SIP} := \min_{x}. \left\{ c^\top x + \mathcal{Q}(x) : \; x \in X \right\},$$

where

$$\mathcal{Q}(x) := \mathbb{E}_{\xi} \left[\min_{y}. \left\{ q(\xi)^\top y : \; W(\xi) y = h(\xi) - T(\xi) x, y \in Y(\xi) \right\} \right].$$

Introduction

Stochastic Integer Programming - Formulation

$$z^{SIP} := \min_x . \left\{ c^\top x + \mathcal{Q}(x) : \; x \in X \right\},$$

where

$$\mathcal{Q}(x) := \mathbb{E}_\xi \left[\min_y . \left\{ q(\xi)^\top y : \; W(\xi) y = h(\xi) - T(\xi) x, y \in Y(\xi) \right\} \right].$$

X is a mixed-integer linear set consisting of linear constraints and integer restrictions on x. $\mathcal{Q} : \mathbf{R}^{n_x} \mapsto \mathbf{R}$ is the expected recourse value

Introduction

Stochastic Integer Programming - Formulation

$$z^{SIP} := \min_x . \left\{ c^\top x + \mathcal{Q}(x) : \; x \in X \right\},$$

where

$$\mathcal{Q}(x) := \mathbb{E}_\xi \left[\min_y . \left\{ q(\xi)^\top y : \; W(\xi) y = h(\xi) - T(\xi) x, y \in Y(\xi) \right\} \right].$$

X is a mixed-integer linear set consisting of linear constraints and integer restrictions on x. $\mathcal{Q} : \mathbf{R}^{n_x} \mapsto \mathbf{R}$ is the expected recourse value

ξ is approximated by the discrete finite set S, consisting of the realisations, $\xi_1, \dots, \xi_{|S|}$, with probabilities $p_1, \dots, p_{|S|}$.

$$z^{SIP} := \min_x . \left\{ c^\top x + \mathcal{Q}(x) : x \in X \right\},$$

where

$$\mathcal{Q}(x) := \mathbb{E}_\xi \left[\min_y . \left\{ q(\xi)^\top y : \; W(\xi)y = h(\xi) - T(\xi)x, y \in Y(\xi) \right\} \right].$$

Each ξ_s of ξ is called a scenario and encodes the realisations for each $(q(\xi_s), h(\xi_s), W(\xi_s), T(\xi_s), Y(\xi_s)) \equiv (q_s, h_s, W_s, T_s, Y_s)$.

$$z^{SIP} := \min_{x}. \left\{ c^\top x + \mathcal{Q}(x) : x \in X \right\},$$

where

$$\mathcal{Q}(x) := \mathbb{E}_\xi \left[\min_{y}. \left\{ q(\xi)^\top y : \; W(\xi) y = h(\xi) - T(\xi) x, y \in Y(\xi) \right\} \right].$$

Each ξ_s of ξ is called a scenario and encodes the realisations for each $(q(\xi_s), h(\xi_s), W(\xi_s), T(\xi_s), Y(\xi_s)) \equiv (q_s, h_s, W_s, T_s, Y_s)$.

$Y_s \subset \mathbf{R}^{n_y}$ contains linear and integrality constraints on y_s.

$$z^{SIP} = \min_{x,y}. \left\{ c^\top x + \sum_{s \in S} p_s q_s^\top y_s : \ (x, y_s) \in K_s, \forall s \in S \right\},$$

where $K_s := \{(x, y_s) : W_s y_s = h_s - T_s x, x \in X, y_s \in Y_s\}$.

Introduction

Stochastic Integer Programming - Formulation

$$z^{SIP} = \min_{x,y}. \left\{ c^\top x + \sum_{s \in S} p_s q_s^\top y_s : \ (x, y_s) \in K_s, \forall s \in S \right\},$$

where $K_s := \{(x, y_s) : W_s y_s = h_s - T_s x, x \in X, y_s \in Y_s\}$.

This formulation is the deterministic equivalent.

Introduction

Stochastic Integer Programming - Formulation

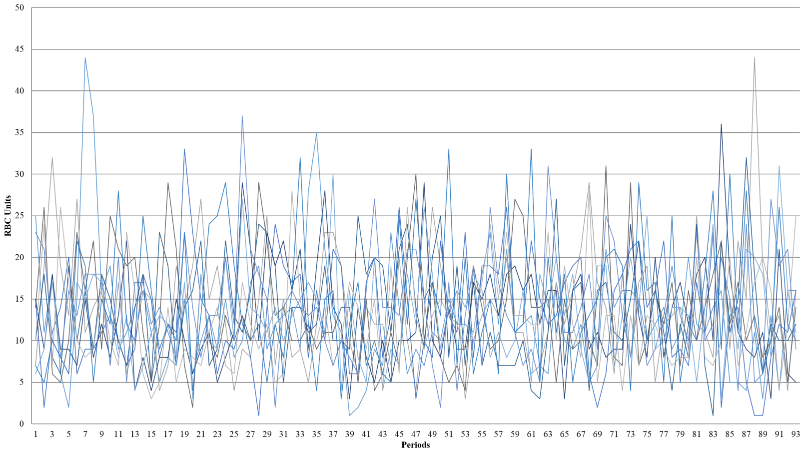
$$z^{SIP} = \min_{x,y}. \left\{ c^\top x + \sum_{s \in S} p_s q_s^\top y_s : \ (x, y_s) \in K_s, \forall s \in S \right\},$$

where $K_s := \{(x, y_s) : W_s y_s = h_s - T_s x, x \in X, y_s \in Y_s\}$.

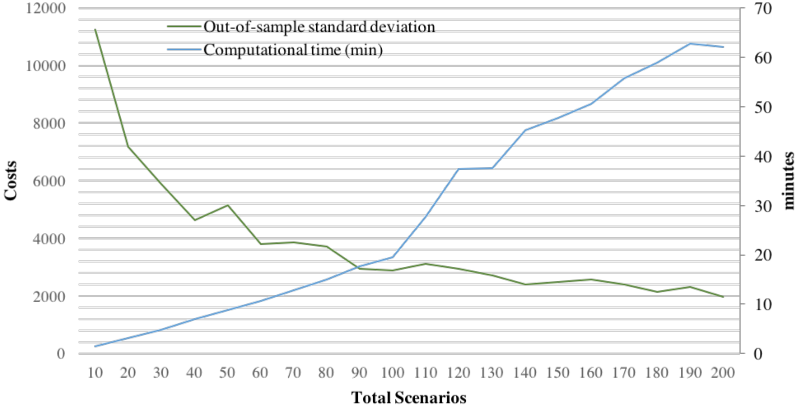
This formulation is the deterministic equivalent.

The challenge: as the quality of the uncertainty representation increases, the size of the problem grows.

This is an even more prominent issue in the context of Stochastic Integer Programming (SIP) problems.



A two-stage stochastic programming model for inventory management in the blood supply chain. M Dillon, F Oliveira, B Abbasi, *International Journal of Production Economics*, 2741, 2017.



A two-stage stochastic programming model for inventory management in the blood supply chain. M Dillon, F Oliveira, B Abbasi, *International Journal of Production Economics*, 2741, 2017.

Introduction

Stochastic Integer Programming - Reformulation

$$z^{SIP} = \min_{x,y,\overline{x}} \left\{ \begin{array}{l} \displaystyle\sum_{s \in S} p_s \left(c^\top x_s + q_s^\top y_s \right) : \\ (x_s, y_s) \in K_s,\ x_s = \overline{x},\ \forall s \in S,\ \overline{x} \in \mathbf{R}^{n_x} \end{array} \right\}.$$

Variable splitting is used to explicitly represent nonanticipativity (NAC) conditions.

$$z^{SIP} = \min_{x,y,\overline{x}} \left\{ \begin{array}{l} \sum_{s \in S} p_s \left(c^\top x_s + q_s^\top y_s \right) : \\ (x_s, y_s) \in K_s, \ x_s = \overline{x}, \ \forall s \in S, \ \overline{x} \in \mathbf{R}^{n_x} \end{array} \right\}.$$

Variable splitting is used to explicitly represent nonanticipativity (NAC) conditions.

The consensus variable $\overline{x}$ enforce nonanticipativity conditions on the first-stage variables.

$$z^{SIP} = \min_{x,y,\overline{x}} \left\{ \begin{array}{l} \displaystyle\sum_{s \in S} p_s \left(c^\top x_s + q_s^\top y_s \right) : \\ (x_s, y_s) \in K_s, \; x_s = \overline{x}, \; \forall s \in S, \; \overline{x} \in \mathbf{R}^{n_x} \end{array} \right\}.$$

This formulation exposes its separable structure.

$$z^{SIP} = \min_{x,y,\overline{x}} \left\{ \begin{array}{l} \displaystyle\sum_{s \in S} p_s \left(c^\top x_s + q_s^\top y_s \right) : \\ (x_s, y_s) \in K_s, \; x_s = \overline{x}, \; \forall s \in S, \; \overline{x} \in \mathbf{R}^{n_x} \end{array} \right\}.$$

This formulation exposes its separable structure.

Our goal: exploit this structure using decomposition techniques.

- numerous small problems $>$ single large-scale problem;
- allows for parallelisation.

Outline of this talk

Lagrangian duality

To obtain separability, we relax the NAC. First, let

$$z^{LR}(\omega) = \min_{x,y,\overline{x}} \left\{ \begin{array}{l} \displaystyle\sum_{s \in S} p_s L_s(x_s, y_s, \overline{x}, \omega_s) : \\ (x_s, y_s) \in K_s, \ \forall s \in S, \ \overline{x} \in \mathbf{R}^{n_x} \end{array} \right\}.$$

where $L_s(x_s, y_s, \overline{x}, \omega_s) := c^\top x_s + q_s^\top y_s + \omega_s^\top (x_s - \overline{x})$.

Lagrangian duality

To obtain separability, we relax the NAC. First, let

$$z^{LR}(\omega) = \min_{x,y,\overline{x}} \left\{ \begin{array}{l} \sum_{s \in S} p_s L_s(x_s, y_s, \overline{x}, \omega_s) : \\ (x_s, y_s) \in K_s, \ \forall s \in S, \ \overline{x} \in \mathbf{R}^{n_x} \end{array} \right\}.$$

where $L_s(x_s, y_s, \overline{x}, \omega_s) := c^\top x_s + q_s^\top y_s + \omega_s^\top (x_s - \overline{x})$.

Then, we focus on solving the Lagrangian Dual problem

$$z^{LD} := \max_{\omega \in \Omega} z^{LR}(\omega),$$

where $\omega := (\omega_s)_{s \in S} \in \Omega := \{\omega \mid \sum_{s \in S} p_s^\top \omega_s = 0\}$.

Lagrangian duality

Note that $\omega \in \Omega := \{\omega \mid \sum_{s \in S} p_s^\top \omega_s = 0\}$ implies that $\overline{x}$ vanishes.

$$z^{LR}(\omega) = \sum_{s \in S} p_s z_s^{LR}(\omega_s), \text{ where}$$

$$z_s^{LR}(\omega_s) = \min._{x,y} \left\{(c + \omega_s)^\top x + q_s^\top y : (x, y) \in K_s, \forall s \in S\right\}.$$

Lagrangian duality

Note that $\omega \in \Omega := \{\omega \mid \sum_{s \in S} p_s^\top \omega_s = 0\}$ implies that $\overline{x}$ vanishes.

$$z^{LR}(\omega) = \sum_{s \in S} p_s z_s^{LR}(\omega_s), \text{ where}$$

$$z_s^{LR}(\omega_s) = \min._{x,y} \left\{ (c + \omega_s)^\top x + q_s^\top y : (x, y) \in K_s, \forall s \in S \right\}.$$

Notice that $z^{LR}(\omega)$ is piecewise-affine and thus nonsmooth optimisation can be employed to obtain z^{LD}.

Some examples include:

- subgradient method;
- cutting plane method;
- bundle methods.

Lagrangian duality

It is a well-known result that $z^{LP} \leq z^{LD} \leq z^{SIP}$, where z^{LP} is the optimal value of the linear relaxation of z^{SIP}.

[1] polyhedral set formed by linear constraints - the LP (continuous) relaxation

Lagrangian duality

It is a well-known result that $z^{LP} \leq z^{LD} \leq z^{SIP}$, where z^{LP} is the optimal value of the linear relaxation of z^{SIP}.

This is associated with a powerful result that provides the equivalent primal characterisation

$$z^{LD} = \min_{x,y,\overline{x}} \left\{ \begin{array}{l} \sum_{s \in S} p_s (c^\top x_s + q_s^\top y_s) : \\ (x_s, y_s) \in \mathbf{conv}(K_s),\ x_s = \overline{x},\ \forall s \in S \end{array} \right\}.$$

and the fact that, typically, $\mathbf{conv}(K_s) \subset \mathbf{poly}(K_s)$[1], for $s \in S$.

Despite the inevitable duality gap, this result is key for what we develop next.

[1] polyhedral set formed by linear constraints - the LP (continuous) relaxation

Augmented Lagrangian duality

Alternatively, we define the (augmented) Lagrangian relaxation as

$$z_\rho^{LR+}(\omega) = \min_{x,y,\overline{x}} \left\{ \sum_{s \in S} p_s L_s^\rho(x_s, y_s, \overline{x}, \omega_s) : (x_s, y_s) \in K_s, \forall s \in S \right\},$$

where $L_s^\rho(x_s, y_s, \overline{x}, \omega_s) := (c + \omega_s)^\top x_s + q_s^\top y_s + \psi_\rho^s(x_s - \overline{x})$.

$\psi_\rho^s : \mathbf{R}^{n_x} \mapsto \mathbf{R}$ is an appropriate penalty function that depends on the penalty parameter ρ.

Augmented Lagrangian duality

Alternatively, we define the (augmented) Lagrangian relaxation as

$$z_\rho^{LR+}(\omega) = \min_{x,y,\overline{x}} \left\{ \sum_{s \in S} p_s L_s^\rho(x_s, y_s, \overline{x}, \omega_s) : (x_s, y_s) \in K_s, \forall s \in S \right\},$$

where $L_s^\rho(x_s, y_s, \overline{x}, \omega_s) := (c + \omega_s)^\top x_s + q_s^\top y_s + \psi_\rho^s(x_s - \overline{x})$.

$\psi_\rho^s : \mathbf{R}^{n_x} \mapsto \mathbf{R}$ is an appropriate penalty function that depends on the penalty parameter ρ.

The main motivation is that typically $z^{LD} \leq z^{LD+} \leq z^{SIP}$, where

$$z^{LD+} = \max_{\omega \in \Omega} z_\rho^{LR+}(\omega).$$

Notice, however, that separability is lost.

Progressive Hedging (PH)

The most widespread paradigm is to set $\psi_\rho^s(x_s - \overline{x}) = \frac{\rho}{2}||x_s - \overline{x}||_2^2$ and use the alternating direction method of multipliers (ADMM).

This idea was originally proposed in Rockafellar and Wets (1991)[2] and framed as the progressive hedging (PH) method.

[2] https://doi.org/10.1287/moor.16.1.119

Progressive Hedging (PH)

The most widespread paradigm is to set $\psi_\rho^s(x_s - \overline{x}) = \frac{\rho}{2}||x_s - \overline{x}||_2^2$ and use the alternating direction method of multipliers (ADMM).

This idea was originally proposed in Rockafellar and Wets (1991)[2] and framed as the progressive hedging (PH) method.

The strategy behind PH to solve $z^{LD+} = \max_{\omega \in \Omega} z_\rho^{LR+}(\omega)$ is:

- use nonlinear Gauss-Seidel/coordinate descent to optimise in x and $\overline{x}$ separately;
- perform a dual ascent step (method of multipliers).

[2] https://doi.org/10.1287/moor.16.1.119

Progressive Hedging (PH)

Algorithm 1 Progressive Hedging

initialise $\rho, (\omega_s^0)_{s \in S}, (x_s^0)_{s \in S}, \overline{x}^0 \leftarrow \sum_s p_s x_s^0, \epsilon, k_{\max}$
for $k = 1, \dots, k_{\max}$ **do**
  for $s \in S$ **do**
    $$(x_s^k, y_s^k) \leftarrow \operatorname{argmin}_{x,y} \left\{ \begin{array}{l} (c + \omega_s)^\top x_s + q_s^\top y_s + \frac{\rho}{2} \|x_s - \overline{x}^{k-1}\|_2^2 : \\ (x_s, y_s) \in K_s \end{array} \right\}$$
  end for
  $\overline{x}^k \leftarrow \sum_s p_s x_s^k$
  if $\sqrt{\sum_{s \in S} p_s \|x_s^k - \overline{x}^{k-1}\|_2^2} < \epsilon$ **or** $k = k_{\max}$ **then**
    return $((x_s^k, y_s^k)_{s \in S}, \overline{x}^k)$
  else
    $\omega_s^k \leftarrow \omega_s^{k-1} + \rho(x_s^k - \overline{x}^k), \ \forall s \in S$
  end if
  $k \leftarrow k + 1$
end for

Progressive Hedging (PH)

In the context of SIP, as K_s is not convex, PH becomes a heuristic. However, it has a positive features:

- Highly parallelisable, with few reduce-type steps;
- Simple implementation, despite relying on the parameter ρ;
- Can be modified to provide dual bounds at any point (solve $z_{\rho}^{LR+}(\omega^k)$) at cost of solving an additional MIP.

Progressive Hedging (PH)

In the context of SIP, as K_s is not convex, PH becomes a heuristic. However, it has a positive features:

- Highly parallelisable, with few reduce-type steps;
- Simple implementation, despite relying on the parameter ρ;
- Can be modified to provide dual bounds at any point (solve $z_\rho^{LR+}(\omega^k)$) at cost of solving an additional MIP.

Next, we present developments to address the following issues:

- Premature convergence of PH when applied to z^{LD+};
- The need to solve an additional MIP to obtain bounds;
- Susceptibility to poorly chosen ρ.

Outline of this talk

Using PH to obtain Lagrangian dual bounds

$$z^{LD+} = \min_{x,y,\overline{x}} \left\{ \begin{array}{l} \displaystyle\sum_{s \in S} p_s(c^\top x_s + q_s^\top y_s) + \frac{\rho}{2}||x_s - \overline{x}||_2^2 : \\ (x_s, y_s) \in \mathbf{conv}(K_s),\ x_s = \overline{x},\ \forall s \in S \end{array} \right\}.$$

The primal characterisation of z^{LD+} has the properties required for PH to converge optimally. This, however, would require a explicit description of $\mathbf{conv}(K_s)$.

Using PH to obtain Lagrangian dual bounds

$$z^{LD+} = \min_{x,y,\overline{x}} \left\{ \begin{array}{c} \sum_{s \in S} p_s(c^\top x_s + q_s^\top y_s) + \frac{\rho}{2} ||x_s - \overline{x}||_2^2 : \\ (x_s, y_s) \in \mathbf{conv}(K_s), \; x_s = \overline{x}, \; \forall s \in S \end{array} \right\}.$$

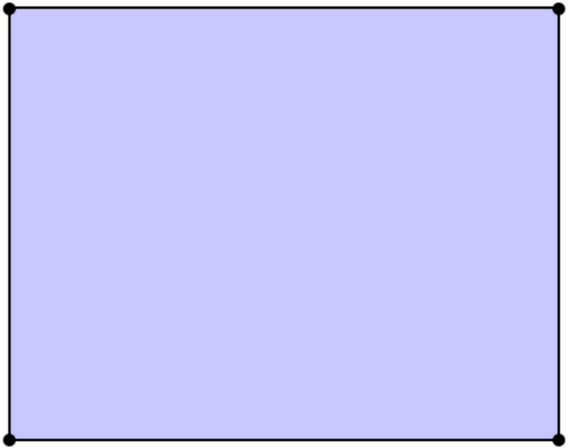
The primal characterisation of z^{LD+} has the properties required for PH to converge optimally. This, however, would require a explicit description of $\mathbf{conv}(K_s)$.

How we circumvent this: we create an inner approximation of $\mathbf{conv}(K_s)$ iteratively.

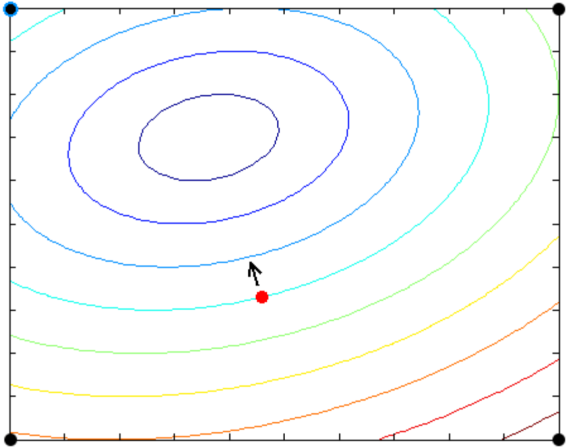
We use a modified Frank-Wolfe method that "keeps track" of the visited points - the Simplicial Decomposition Method (SDM).

Simplicial Decomposition Method

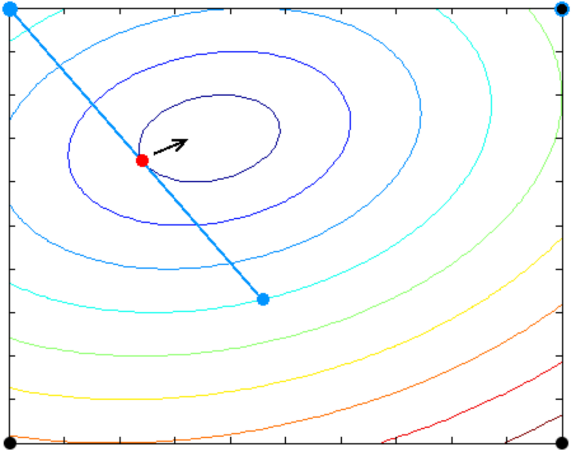
Simplicial Decomposition Method



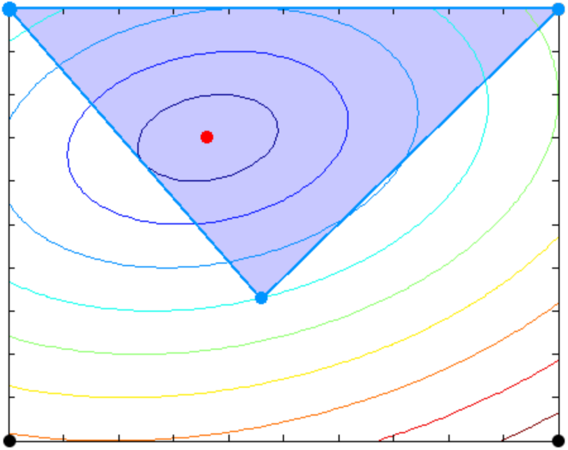
Simplicial Decomposition Method



Simplicial Decomposition Method



Simplicial Decomposition Method



Using PH to obtain Lagrangian dual bounds

Algorithm 2 Simplicial Decomposition Method

preconditions: $V_s^0 \subset \textbf{conv}(K_s)$ and $\overline{x} = \sum_{s \in S} p_s x_s^0$

function $\mathrm{SDM}(V_s^0, x_s^0, \omega_s, \overline{x}, t_{max}, \tau)$

  for $t = 1, \dots, t_{max}$ **do**

    $\widehat{\omega}_s^t \leftarrow \omega_s + \rho(x_s^{t-1} - \overline{x})$

    $(\widehat{x}_s, \widehat{y}_s) \in \mathrm{argmin}_{x,y} \left\{ (c + \widehat{\omega}_s^t)^\top x + q_s^\top y : (x, y) \in \textbf{conv}(K_s) \right\}$

    if $t = 1$ **then**

      $\phi_s \leftarrow (c + \widehat{\omega}_s^t)^\top \widehat{x}_s + q_s^\top \widehat{y}_s$

    end if

    $\Gamma^t \leftarrow -[(c + \widehat{\omega}_s^t)^\top (\widehat{x}_s - x_s^{t-1}) + q_s^\top (\widehat{y}_s - y_s^{t-1})]$

    $V_s^t \leftarrow V_s^{t-1} \cup \{(\widehat{x}_s, \widehat{y}_s)\}$

    $(x_s^t, y_s^t) \in \mathrm{argmin}_{x,y} \left\{ L_s^\rho(x, y, z, \omega_s) : (x, y) \in \textbf{conv}(V_s^t) \right\}$

    if $\Gamma^t \leq \tau$ **or** $t = t_{\max}$ **then**

      return $\left(x_s^t, y_s^t, V_s^t, \phi_s\right)$

    end if

  end for

end function

Using PH to obtain Lagrangian dual bounds

Important remarks to clarify the SDM:

1. Observe that

$$\nabla_{(x,y)} L_s^{\rho}(x, y, \overline{x}, \omega_s)|_{(x_s^{t-1}, y_s^{t-1})} = \begin{bmatrix} c + \omega_s + \rho(x_s^{t-1} - \overline{x}) \\ q_s \end{bmatrix} = \begin{bmatrix} c + \widehat{\omega}_s \\ q_s \end{bmatrix}.$$

2. We can replace $\mathbf{conv}(K_s)$ with K_s in Line 5 due to the linear objective function.

3. $\sum_{s \in S} p_s \phi_s$ provides a valid dual bound.

Using PH to obtain Lagrangian dual bounds

Important remarks to clarify the SDM:

1. Observe that

$$\nabla_{(x,y)} L_s^\rho(x, y, \overline{x}, \omega_s)|_{(x_s^{t-1}, y_s^{t-1})} = \begin{bmatrix} c + \omega_s + \rho(x_s^{t-1} - \overline{x}) \\ q_s \end{bmatrix} = \begin{bmatrix} c + \widehat{\omega}_s \\ q_s \end{bmatrix}.$$

2. We can replace $\mathbf{conv}(K_s)$ with K_s in Line 5 due to the linear objective function.

3. $\sum_{s \in S} p_s \phi_s$ provides a valid dual bound.

4. Notice that Line 11 is a simple QP problem.

$$(x_s^t, y_s^t, \lambda) \in \operatorname*{argmin}_{x,y,\lambda} \left\{ \begin{array}{l} L_s^\rho(x, y, z, \omega_s) : (x, y) = \sum_{(\widehat{x}^i, \widehat{y}^i) \in V_s^t} \lambda_i (\widehat{x}^i, \widehat{y}^i), \\ \sum_{i=1,\dots,|V_s^t|} \lambda_i = 1, \text{ and } \lambda_i \geq 0 \text{ for } i = 1, \dots, |V_s^t| \end{array} \right\}.$$

Using PH to obtain Lagrangian dual bounds

Algorithm 3 FW-PH

function $\mathrm{FW\text{-}PH}((V_s^0)_{s\in S},\ (x_s^0, y_s^0)_{s\in S},\ \omega^0,\ \rho,\ \alpha,\ \epsilon,\ k_{max},\ t_{max})$
    $\overline{x}^0 \leftarrow \sum_{s\in S} p_s x_s^0$
    $\omega_s^1 \leftarrow \omega_s^0 + \rho(x_s^0 - \overline{x}^0)$, for $s \in S$
    for $k = 1, \ldots, k_{max}$ **do**
        for $s \in S$ **do**
            $[x_s^k, y_s^k, V_s^k, \phi_s^k] \leftarrow \mathrm{SDM}(V_s^{k-1}, x_s^{k-1}, \omega_s^k, \overline{x}^{k-1}, t_{max}, 0)$
        end for
        $\phi^k \leftarrow \sum_{s\in S} p_s \phi_s^k$
        $\overline{x}^k \leftarrow \sum_{s\in S} p_s x_s^k$
        if $\sqrt{\sum_{s\in S} p_s \|x_s^k - \overline{x}^{k-1}\|_2^2} < \epsilon$ **or** $k = k_{\max}$ **then**
            return $((x_s^k, y_s^k)_{s\in S}, \overline{x}^k, \omega^k, \phi^k)$
        end if
        $\omega_s^{k+1} \leftarrow \omega_s^k + \rho(x_s^k - \overline{x}^k)$, for $s \in S$
    end for
end function

Computational experiments

Experiments were performed on three distinct problems (instances available in the literature):

1. the capacitated facility location problem (CAP);
2. the dynamic capacity allocation problem (DCAP);
3. the server location under uncertainty problem (SSLP).

We performed computations using a C++ implementation of Algorithms PH and FW-PH using CPLEX 12.5 as the solver.

Computational experiments

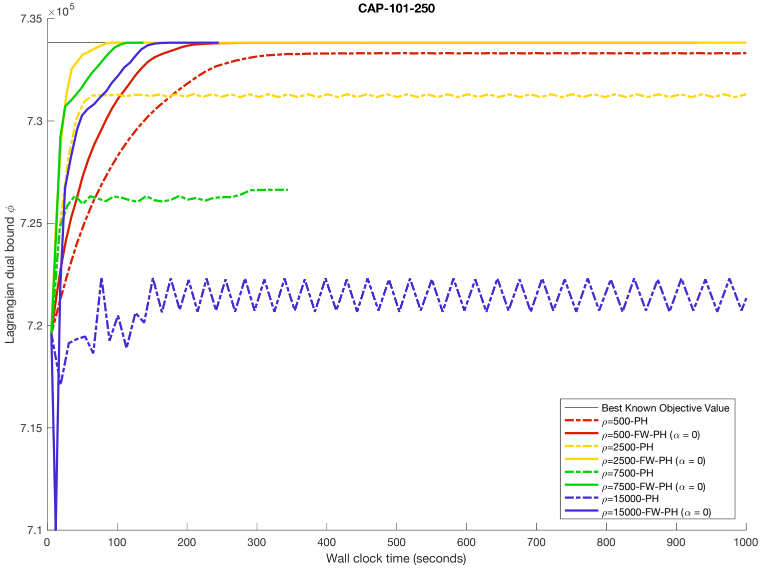
Experiments were performed on three distinct problems (instances available in the literature):

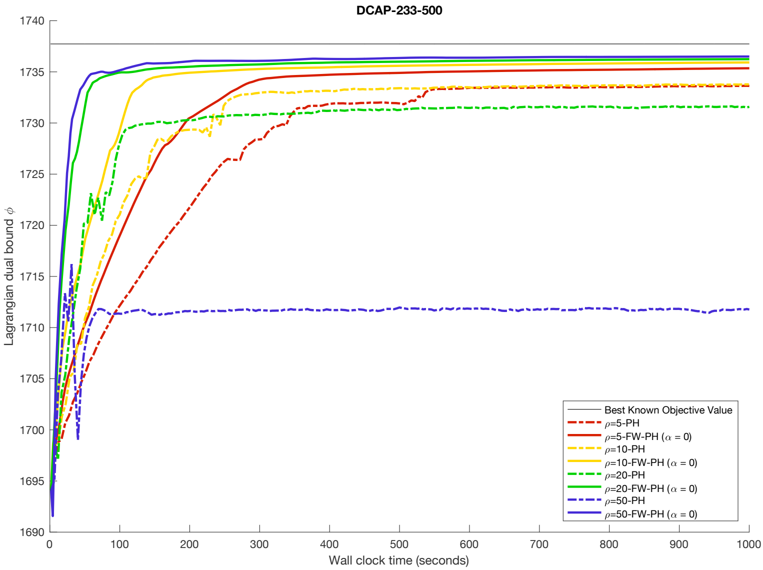
1. the capacitated facility location problem (CAP);
2. the dynamic capacity allocation problem (DCAP);
3. the server location under uncertainty problem (SSLP).

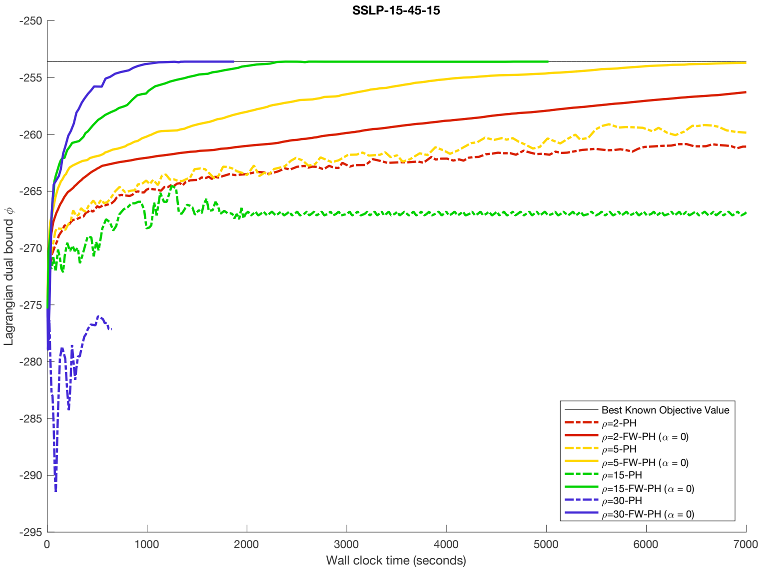
We performed computations using a C++ implementation of Algorithms PH and FW-PH using CPLEX 12.5 as the solver.

Computations run on the Raijin cluster:

- high performance computing (HPC) environment;
- maintained by Australia's National Computing Infrastructure (NCI) and supported by the Australian Government;
- 3592 nodes, 57472 cores of Intel Xeon E5-2670 processors with up to 8 GB PC1600 memory per core (128 GB per node).


CAP-101-250
$\times 10^5$
7.35
7.3
7.25
7.2
7.15
7.1
Lagrangian dual bound ϕ
0
100
200
300
400
500
600
700
800
900
1000
Wall clock time (seconds)
Best Known Objective Value
ρ=500-PH
ρ=500-FW-PH (α = 0)
ρ=2500-PH
ρ=2500-FW-PH (α = 0)
ρ=7500-PH
ρ=7500-FW-PH (α = 0)
ρ=15000-PH
ρ=15000-FW-PH (α = 0)


DCAP-233-500
Lagrangian dual bound ϕ
Wall clock time (seconds)
1740
1735
1730
1725
1720
1715
1710
1705
1700
1695
1690
0
100
200
300
400
500
600
700
800
900
1000
Best Known Objective Value
ρ=5-PH
ρ=5-FW-PH (α = 0)
ρ=10-PH
ρ=10-FW-PH (α = 0)
ρ=20-PH
ρ=20-FW-PH (α = 0)
ρ=50-PH
ρ=50-FW-PH (α = 0)


SSLP-15-45-15
Lagrangian dual bound ϕ
Wall clock time (seconds)
-250
-255
-260
-265
-270
-275
-280
-285
-290
-295
0
1000
2000
3000
4000
5000
6000
7000
Best Known Objective Value
ρ=2-PH
ρ=2-FW-PH (α = 0)
ρ=5-PH
ρ=5-FW-PH (α = 0)
ρ=15-PH
ρ=15-FW-PH (α = 0)
ρ=30-PH
ρ=30-FW-PH (α = 0)

On the selection of penalty parameters

The previous results highlight the dependance of the method to an adequate penalty term ρ.

Bundle methods are variants of proximal algorithms that:

- combine stabilisation from proximal terms and linearisation;
- rely on either outer or inner approximations.

On the selection of penalty parameters

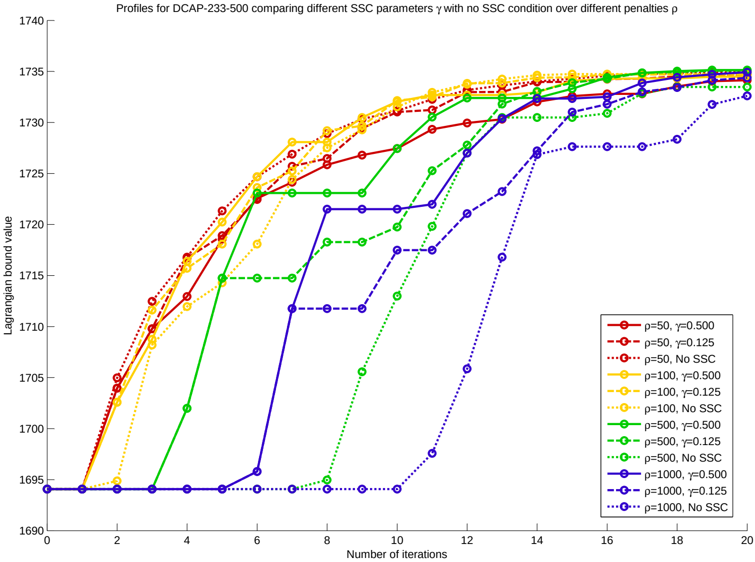
The previous results highlight the dependance of the method to an adequate penalty term ρ.

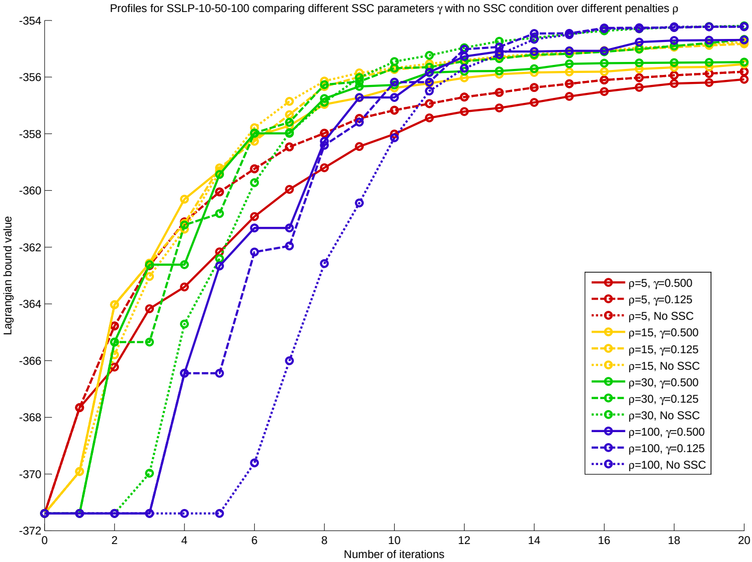
Bundle methods are variants of proximal algorithms that:

- combine stabilisation from proximal terms and linearisation;
- rely on either outer or inner approximations.

The idea is somewhat simple: we perform SDM a few steps (instead of a single one) until a serious step condition is met.

Serious step condition: $\frac{\phi(\tilde{\omega})-\phi(\omega^k)}{\phi^k-\phi(\omega^k)} \leq \gamma$, with $\tilde{\omega} = \omega^k + \rho(x^k - \overline{x}^k)$


Profiles for DCAP-233-500 comparing different SSC parameters γ with no SSC condition over different penalties ρ
Lagrangian bound value
Number of iterations
ρ=50, γ=0.500
ρ=50, γ=0.125
ρ=50, No SSC
ρ=100, γ=0.500
ρ=100, γ=0.125
ρ=100, No SSC
ρ=500, γ=0.500
ρ=500, γ=0.125
ρ=500, No SSC
ρ=1000, γ=0.500
ρ=1000, γ=0.125
ρ=1000, No SSC



Profiles for SSLP-10-50-100 comparing different SSC parameters γ with no SSC condition over different penalties ρ
Lagrangian bound value
Number of iterations
-354
-356
-358
-360
-362
-364
-366
-368
-370
-372
0
2
4
6
8
10
12
14
16
18
20
ρ=5, γ=0.500
ρ=5, γ=0.125
ρ=5, No SSC
ρ=15, γ=0.500
ρ=15, γ=0.125
ρ=15, No SSC
ρ=30, γ=0.500
ρ=30, γ=0.125
ρ=30, No SSC
ρ=100, γ=0.500
ρ=100, γ=0.125
ρ=100, No SSC

Parallelisation performance

No. Proc.	OOQP	PIPS-IPM	SDM-GS1-ALM	SDM-GS5-ALM
	Speedup for SSLP 10-50-500			
1	1.00	1.00	1.00	1.00
8	2.64	2.80	6.87	6.95
16	2.70	2.92	12.95	12.84
32	2.98	3.40	21.67	20.98
Lagr. Value	-349.14	-349.14	-349.48	-349.14

No. Proc.	SDM-GS1-ALM	SDM-GS5-ALM
	Speedup for SSLP 10-50-2000	
1	1.00	1.00
2	2.34	2.34
4	4.81	4.83
8	9.29	9.25
16	18.69	18.48
32	34.63	35.10
64	60.59	60.93
Lagr. Value	-348.35	-347.75

Table: SSLP: Comparing speedup and final best Lagrangian bound

Parallelisation performance

	Speedup for DCAP 233-500			
No. Proc.	OOQP	PIPS-IPM	SDM-GS1-ALM	SDM-GS5-ALM
1	1.00	1.00	1.00	1.00
8	2.44	5.32	6.88	8.11
16	2.81	8.15	13.28	15.65
32	1.63	10.25	23.42	27.40
Lagr. Value	1736.68	1736.68	1734.99	1736.02

	Speedup for DCAP 342-500			
No. Proc.	OOQP	PIPS-IPM	SDM-GS1-ALM	SDM-GS5-ALM
1	1.00	1.00	1.00	1.00
8	2.45	3.78	7.16	8.25
16	2.71	4.36	12.95	15.49
32	1.84	4.64	22.41	26.93
Lagr. Value	1902.84	1903.21	1900.81	1901.90

Table: DCAP: Comparing speedup and final best Lagrangian bound

Outline of this talk

Summary and conclusions

We develop an improved version of the Progressive Hedging (PH) algorithm for Stochastic Integer Programming (SIP) problems, addressing:

1. Premature (suboptimal) convergence;
2. Influence of poorly chosen penalty terms.

Summary and conclusions

We develop an improved version of the Progressive Hedging (PH) algorithm for Stochastic Integer Programming (SIP) problems, addressing:

1. Premature (suboptimal) convergence;
2. Influence of poorly chosen penalty terms.

These are addressed by:

1. Developing FWPH, a combination of a Frank-Wolfe method (Simplicial Decomposition Method) and PH.
2. Proposing a bundle-method inspired dual update control (serious step condition)

Summary and conclusions

The development of an efficient PH-based algorithm allows for

1. Calculation of optimal Lagrangian dual bounds;
2. Very efficient parallelisation;
3. Open a new avenue for research in efficient methods for SIP.

Summary and conclusions

The development of an efficient PH-based algorithm allows for

1. Calculation of optimal Lagrangian dual bounds;
2. Very efficient parallelisation;
3. Open a new avenue for research in efficient methods for SIP.

Future work include:

1. Embed FWPH in a highly paralellisable B&B algorithm for SIP;
2. Allow for suboptimal solutions from the SDM step;
3. Real time adjustment of penalty term (from serious step condition test);
4. Consider alternative penalty functions.

Main references

1. **Original FWPH.**

Combining Progressive Hedging with a Frank-Wolfe method to compute Lagrangian dual bounds in stochastic mixed-integer programming. N Boland, J Christiansen, B Dandurand, A Eberhard, J Linderoth, F Oliveira. *SIAM Journal of Optimization* (in press).

2. **FWPH under a proximal point method perspective.**

A parallelizable augmented Lagrangian method applied to large-scale non-convex-constrained optimization problems N Boland, J Christiansen, B Dandurand, A Eberhard, F Oliveira; *Mathematical Programming*, 1-34, 2018.

Decomposition strategies for stochastic integer programming: challenges and perspectives

Fabricio Oliveira

Systems Analysis Laboratory
Department of Mathematics and Systems Analysis

Aalto University
School of Science

April 16, 2018