A Web Application for Solving Multi-objective Integer Programming Problems

Gokhan Ceyhan, Murat Koksalan and Banu Lokman

October 23, 2017

Department of Industrial Engineering, Middle East Technical University

Table of contents

The Research Project

The research project

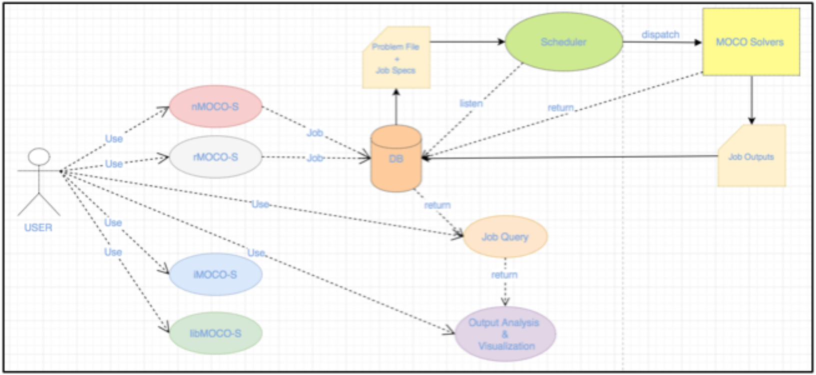
- **Title:** Nondominated Points of Multi-objective Integer Programming Problems: Analysis, Approaches and Applications
- **Funding:** Scientific and Technological Research Council of Turkey (TUBITAK)
- **Members:** Murat Koksalan, Banu Lokman, Gokhan Ceyhan, Sami Ozarik, Ilgin Dogan
- **Goals:**
  - Develop *generic* algorithms to find all or representative nondominated points of *any* MOIP with *any* number of objective function
  - Provide *cloud services* to research communities by embedding these algorithms in a web application
  - Provide tools for visualization of nondominated points
- The project is **on-going**.

www.onlinemoco.com/MOIP/

- **In development phase**
- Applications
  - **nMOCO-S:** Finds all nondominated points
  - **rMOCO-S:** Finds a representative set of nondominated points
  - **iMOCO-S:** Integrates user preferences
  - **libMOCO-S:** A library for MOIP instances
- Technology
  - Client side technologies: **html5, javascript, jQuery**
  - Server side technologies: **javaservlet**
  - Web server: **Apache Tomcat**
  - Database: **Apache Derby** and **JDBC**
  - Solvers: coded in **C++**, use **IBM ILOG CPLEX** API

Use case

How a user intereacts with the system:



Generating all nondominated points

nMOCO-S

MOIP:

$$\begin{aligned} \text{“Max”} \quad & \mathbf{z(x)} = \{z_1(\mathbf{x}), z_2(\mathbf{x}), ..., z_m(\mathbf{x})\} \\ \text{s.t.} \quad & \mathbf{x} \in X \end{aligned}$$

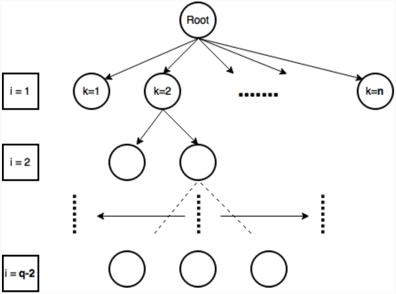
where $z_i(\mathbf{x})$ is a continuous function of $\mathbf{x}$, $X \subseteq \mathbb{R}^n$ is compact and x is a vector of **integer-valued** decision variables. We denote the image of X in the objective space as Z.

- Generates all nondominated points for MOIPs
- Can also be applied as an approximate method for MOMIPs
- Based on Lokman and Koksalan (2013)
- **The idea:** The search region is partitioned and reduced progressively by removing the regions that are dominated by previously found nondominated points.

nMOCO-S

Enhancement:

- An effective data structure to handle any number of objective functions
- A smarter way of detecting the status of a search region



- **N-ary tree**
- **Tree height** = number of objectives (q) - 2
- **Number of child nodes** $\leq$ number of generated nondominated points (N)

nMOCO-S

Tree elements

- **Node:** defines a search space
- **Branching point:** a nondominated point already generated
- **Branching criterion:** a criterion index of the branching point
- **Bound vector:** the *branching criterion* values of *branching points* of the nodes on the path from the root node to the current node

So, at node **k**:

- k_i is the *branching point* index at level i, $i = 1, 2, \ldots, q-2$
- $z_i^{k_i}$ is the *branching criterion* value at level i
- $b^k = (z_1^{k_1}, z_2^{k_2}, \ldots, z_i^{k_i}) \quad \forall i = 1, 2, \ldots (q-3)$ and
  $b^k = (z_1^{k_1}, z_2^{k_2}, \ldots, z_i^{k_i}, z_{i+1}^{k_{i+1}})$ for $i = q-2$ where
  $z_{i+1}^{k+1} = \max_{j \in N} \left\{ z_{i+1}^j : z_l^j \geq b_l^k, l = 1, 2, \ldots, i \right\}$

nMOCO-S

Special structure

- $z_q^1 \geq z_q^2 \geq, \ldots, z_q^N$
- Let C^k be the child nodes of node k. Then,

$$S^k = \bigcup_{c \in C^k} S^c$$

- Let L^k and R^k be the left and right siblings of node k. For nodes except the leaf nodes:

$$S^l \subseteq S^k \subseteq S^r, l \in L^k, r \in R^k$$

- At leaf nodes,

$$b_{q-1}^l \geq b_{q-1}^k \geq b_{q-1}^r, l \in L^k, r \in R^k$$

What kind of problems can nMOCO-S solve?

- For MOIPs, any problem for which there are effective algorithms to solve single objective case
- We use IBM ILOG CPLEX as the single objective solver
  - **MILP**
  - **MIQP**
  - **MIQCP**
- For MOMIPs, our algorithm can generate a set of nondominated points which $\epsilon-$ dominates the nondominated set

Generating representative nondominated points

rMOCO-S

- **The idea:** Find a subset of nondominated points that *represents* the nondominated frontier well enough.
- Based on Ceyhan et al. (2014)
- **How to represent:** *Masin ve Bukchin (2008): Coverage gap*

$$\alpha = \max_{z \in ND} \left\{ \min_{y \in R} \left\{ \max_{1 \leq i \leq p} z_i - y_i \right\} \right\}$$

Corollary

If the coverage gap of subset R is α^*, then R α^*-dominates all nondominated points. $\forall z \in ND, \exists y \in R : z_i \leq y_i + \alpha^*, \forall i = 1, ..., p$.

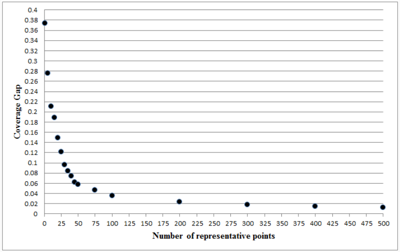
rMOCO-S

Approach 1:

- Find worst covered nondominated point at each iteration which gives the coverage gap of the available subset
- Stop when the desired coverage gap is achieved

The procedure:

- Define coverage gap as an additional objective function to be maximized
- Use nMOCO-S with $q + 1$ objective functions



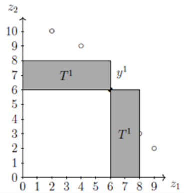
rMOCO-S

Approach 2:

- Identify the desired coverage gap, Δ
- Find a set of nondominated points such that $\alpha \leq \Delta$

The procedure:

- Utilize Δ actively throughout the algorithm
- Construct territories around the previously generated points that are inadmissible for the new point

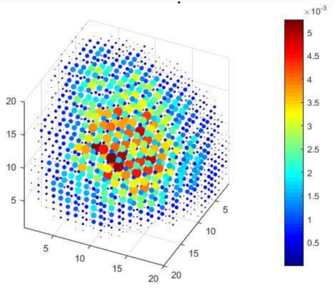


- Tighten the bounds in the leaf nodes of nMOCO-S tree with Δ

rMOCO-S

New features:

- Minimize density weighted coverage gap



- Try to achieve a coverage gap with the smallest possible set
  - Criteria for sorting the search spaces
  - Scalarization of objective functions for the search problem

Visualization of nondominated points

gMOCO-S

The needs:

- To capture the global characteristics of the nondominated frontier

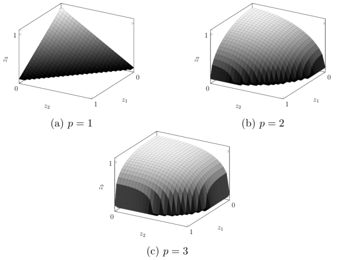


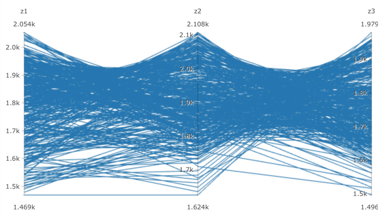
Figure 7: Three dimensional L_p surfaces with different p values

- To investigate the local trade-offs

gMOCO-S

The challenges:

- There can be too many points to visualize



- More than three criteria

gMOCO-S

- We developed *shiny* web application in *R* language
- We use a graphics library offered by *plotly*
- Source code is available in our *github* account
- Users should create an account in *shiny.io* and deploy our source code

Demo

Demo: How to solve a MOKP instance with nMOCO-S?

Summary

Summary

- A need for publicly available multi-objective solvers
- There are recent studies on this direction (e.g. *vOpt-Solver*)
- We try to contribute to this need by providing our MOIP solvers on the web as a *cloud-service*
- A collection of MOIP instances that could be used for benchmarking
- A source code for tools of visualization

Questions?

`http://onlinemoco.com/MOIP/`

`https://github.com/gokhanceyhan`

`https://onlinemoco.shinyapps.io/gMOCO/`