Jul 12th, 10:30 AM - 10:50 AM

Constraint Programming versus MIP for LCA-based Multi-Objective Optimization of Sustainable Potable Water Production Plants

Florin Capitanescu
Luxembourg Institute of Science and Technology (LIST), Environmental Research and Innovation (ERIN), florin.capitanescu@list.lu

Antonino Marvuglia
Luxembourg Institute of Science and Technology (LIST), Environmental Research and Innovation (ERIN), antonino.marvuglia@list.lu

Enrico Benetto
Luxembourg Institute of Science and Technology (LIST), Environmental Research and Innovation (ERIN), enrico.benetto@list.lu

Aras Ahmadi
University of Toulouse, aras.ahmadi@insa-toulouse.fr

Ligia Tiruta-Barna
University of Toulouse, ligia.barna@insa-toulouse.fr

International Environmental Modelling and Software Society (iEMSs)
8th International Congress on Environmental Modelling and Software
Toulouse, France, Sabine Sauvage, José-Miguel Sánchez-Pérez, Andrea Rizzoli (Eds.)
http://www.iemss.org/society/index.php/iemss-2016-proceedings

Constraint Programming versus MIP for LCA-based Multi-Objective Optimization of Sustainable Potable Water Production Plants

Florin Capitanescu [a], **Antonino Marvuglia**[a], **Enrico Benetto**[a], **Aras Ahmadi**[b], **Ligia Tiruta-Barna**[b]

[a]*Luxembourg Institute of Science and Technology (LIST), Environmental Research and Innovation (ERIN), 41 rue du Brill, L-4422, Belvaux, Luxembourg,* {*florin.capitanescu,antonino.marvuglia,enrico.benetto*}*@list.lu*

[b]*University of Toulouse, INSA, UPS, INP, LISBP, 135 Avenue de Rangueil, F-31077 Toulouse, France,* {*aras.ahmadi,ligia.barna*}*@insa-toulouse.fr*

Abstract: Many real-world multi-objective optimization (MOO) problems rely on computationally expensive simulators of industrial processes and require solutions within a limited time budget. In this context, we propose a heuristic approach which aims at building a surrogate problem model, solvable by computationally efficient optimization methods, in order to quickly provide a sufficiently accurate estimation of the Pareto front. The proposed approach generates a multi-objective mixed-integer programming (MO-MIP) proxy model of the MOO problem using sensitivity-based piece-wise linear approximation of objectives and constraints. The approximation of the Pareto front is obtained by applying the ε-constraint method to the multi-objective surrogate problem, transforming it into a desired number of single objective (SO) MIP problems. The paper further explores the pros and cons of three algorithms for the solution of the SO-MIP problems namely constraint programming (CP), MIP, and constraint integer programming (CIP) which integrates CP and MIP methods. In the context of computational sustainability, the proposed methodology is successfully applied to the cost versus life cycle assessment (LCA)-based environmental optimization of potable water production plants (PWPPs). The numerical results obtained indicate that the proposed approach converges much faster to the Pareto front than the state-of-the-art metaheuristic algorithm SPEA2.

Keywords: multi-objective optimization; life cycle assessment; constraint programming; mixed-integer programming; expensive optimization, computational sustainability.

1 Introduction

Computational sustainability is a holistic interdisciplinary[1] framework which aims at managing natural resources as an optimal trade-off between economic, environmental, and societal needs (Gomes, 2009). Quantitative tools need to be developed to achieve this goal as well as to support policy/decision making for sustainable environmental management.

Multi-objective optimization (MOO) algorithms are essential part of the computational sustainability framework. Nowadays there is a wealth of derivative-free meta-heuristic

[1]Computational sustainability framework combines methods from various disciplines such as: environment, economics, mathematics (e.g. optimization, statistics, machine learning), computer science, social science, etc.

MOO algorithms which can be classified into two main categories: evolutionary algorithms (e.g. genetic algorithms, differential evolution, etc.) and swarm intelligence-based algorithms (e.g. particle swarm optimization, ant colony, etc.) (Zhou et al., 2011; Deb, 2014). These algorithms are generic and particularly suitable for black-box MOO, where classical derivative-based mathematical programming methods are impractical because either the problem formulation in analytical form is unavailable or the optimal objectives trade-off (i.e. the Pareto front) features are challenging (e.g. non-convex, discontinuous, etc.). These algorithms are reliable and can explore complex Pareto fronts in a single run. On the other hand, they scale poorly with problem size and converge slowly close to the optimum.

The performance of the meta-heuristic algorithms is usually assessed on computationally inexpensive benchmark MOO problems with complex Pareto fronts and allowing a large number of functions evaluations. However, in many real-life problems, the simulation of the black-box model is computationally intensive while the computational time budget is limited. Under these stringent requirements the meta-heuristic algorithms may not converge and hence they may not provide a satisfactorily accurate and/or evenly spread Pareto front. These computationally expensive MOO problems with limited number of functions evaluations require developing new algorithms with different accuracy/speed trade-offs (Santana-Quintero et al., 2010). In this emerging research field, there exist only a few approaches, which rely on Gaussian stochastic process modelling (Knowles, 2006).

This paper proposes a new approach for expensive MOO problems, which explores functions piece-wise gradients in order to build a MO-MIP surrogate model. A major contribution of the paper consists in the assessment of the pros and cons of three algorithms for the solution of the proxy SO-MIP problems that is CP, CIP, and MIP. In the context of computational sustainability and sustainable infrastructure, the interest of the proposed approach is proven for the cost versus environmental impact optimization of potable water production plants (PWPPs). The environmental impact is modeled adopting a life cycle assessment (LCA) approach. The latter is a standardized methodology (ISO-14040, 2006) which analyzes the lifecycle environmental performance of products and processes (including policies). Although LCA methodology points out potential environmental hot spots, it does not provide per se strategies to reduce the environmental impacts.

Although the extension of various industrial processes optimization to account for environmental impact has been extensively explored, e.g. (Jacquemin et al., 2012), most of the previous works assume that the process can be modeled in closed form allowing thereby resorting to mathematical programming-based methods such as: linear programming (Azapagic and Clift, 1999), NLP (Gebreslassie et al., 2009), MIP (You et al., 2012), mixed-integer nonlinear programming (Guillén-Gosálbez and Grossmann, 2010), etc. As will be explained later on these approaches cannot be directly applied to our MOO problem. Furthermore, the LCA-based MOO of PWPPs has received little attention up to now. This problem has been tackled via e.g. a hybrid algorithm combining NSGA-II and COBYLA in (Ahmadi and Tiruta-Barna, 2015), evolutionary algorithms (SPEA2 and NSGA-II) in (Capitanescu et al., 2015b), surrogate MIP and curve-fitting-based NLP models in (Capitanescu et al., 2015a), and an archive-based evolutionary algorithm with adaptive search space partitioning in (Ahmadi et al., 2016).

The remaining of the paper is organized as follows. Section 2 briefly describes the PWPP simulator, called EVALEAU, formulates conceptually the MOO problem and presents the MOO - process engineering - LCA tool. Section 3 describes the proposed MIP approach for the MOO. Section 4 provides an overview of constraint programming. Section 5 offers the optimization results obtained with the proposed methodology for a real-world PWPP.

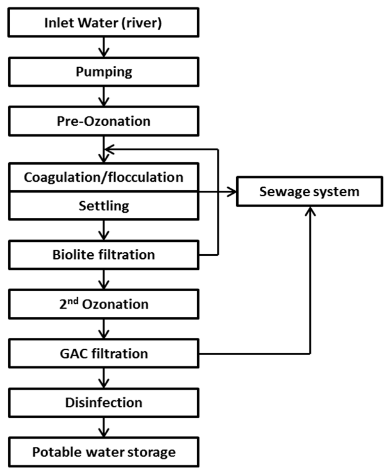

Figure 1. Flowchart of the PWPP used in the case study.

Section 6 concludes.

2 LCA - Process Engineering - Multi-Objective Optimization Problem

2.1 EVALEAU Simulator

The EVALEAU simulator is a state-of-the-art process modelling - LCA tool for prospective and retrospective simulation of PWPPs (Méry et al., 2013). The simulator comprises a certain number of unit processes for water treatment which can be combined to simulate a specific treatment chain (see e.g. Fig. 1). The simulator was developed in the LCA software Umberto® and is linked to the Ecoinvent® database (Weidema et al., 2013) for the life cycle inventory (LCI) of background processes. It relies on the software PHREEQC (Parkhurst and Appelo, 2013) for water chemistry calculation.

2.2 MOO Problem Abstract Formulation

The MOO problem corresponding to a relevant operating scenario of a PWPP can be abstractly formulated as follows:

$$\min_{x_1,\dots,x_n} \quad \{f_1(x_1,\dots,x_n); f_2(x_1,\dots,x_n)\} \tag{1}$$

$$\text{s.t.} \quad g_q(x_1,\dots,x_n) = 0, \quad q = 1,\dots,n_g \tag{2}$$

$$h_r(x_1,\dots,x_n) \geq \underline{h}_r, \quad r = 1,\dots,n_h \tag{3}$$

$$\underline{x}_i \leq x_i \leq \overline{x}_i, \qquad i = 1,\dots,n, \tag{4}$$

where: $\mathbf{x} = [x_1,\dots,x_n]$ is the vector of decision variables (e.g. design and operational parameters of the various unit processes of the treatment chain), the goal f_1 represents the operational cost of the PWPP (e.g. costs of raw materials, chemicals, electricity, etc.), the goal f_2 models the LCA-based environmental impact of the PWPP.

Constraints (2) describe the input-output mass flow for each unit process in the whole chain. Constraints (3) enforce outlet water drinkability quality, according to the best water quality class in France (SEQEau, 2003), modeled by a set of seven major aggregated

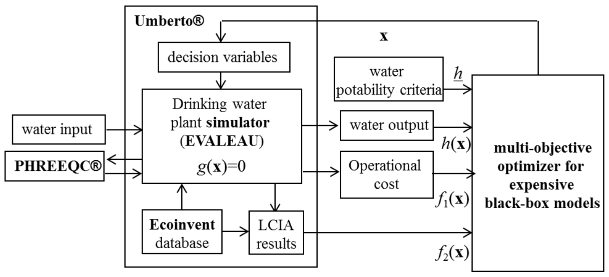

Figure 2. Software architecture of the integrated tool coupling EVALEAU simulator with a MO optimizer.

parameters[2]. Finally, constraints (4) represent physical bounds of the decision variables.

2.3 High Level Software Architecture of the LCA - Process Engineering - Multi-Objective Optimization Tool

Note that, tackling the MOO problem (1)-(4) by means of classical mathematical programming methods is impractical due to several reasons such as: the constraints (2)-(3) possess challenging features (e.g. non-linearity, non-convexity, bi-level as they rely on the expert program PHREEQC for simulating chemical reactions equilibrium in some unit process, being thereby very difficult to fully express in analytical form, large scale size, etc.), the environmental impact goal f_2 depends on the outcome of the simulator run, etc.

To overcome these challenges we adopt a solution, shown in Fig. 2, which couples the EVALEAU simulator with a multi-goal optimizer. The optimization problem is hence decoupled into two manageable sub-problems which are solved iteratively: (a) solution of equality constraints (2) obtained by running the simulator for a given vector of control variables $\mathbf{x}$, and (b) improvement of the MOO problem solutions by the optimizer, which is fed with the values of functions $f_1(\mathbf{x})$, $f_2(\mathbf{x})$, and $h_r(\mathbf{x})$ just evaluated by the simulator, and returns to the latter new promising values for decision variables $\mathbf{x}$.

3 The Proposed Approach for Expensive MOO Problems

3.1 Flowchart of the MO-MIP Approach

Figure 3 shows the flowchart of the proposed MO-MIP approach. The core of the approach is the way in which the surrogate MO optimization problem is built.

The main steps of this approach are the following:

1. Computation of objectives and constraints sensitivities with respect to decision variables, taking a brute force approach.

2. Formulation of the MO-MIP surrogate problem model relying on the sensitivity-based piece-wise linearization of the objectives and constraints.

3. Solution of the MO-MIP problem by the ε-constraint method which transforms the

[2]Namely total coliforms, total trihalomethanes, total organic carbon, Escherichia coli, faecal streptococci, turbidity, and conductivity.

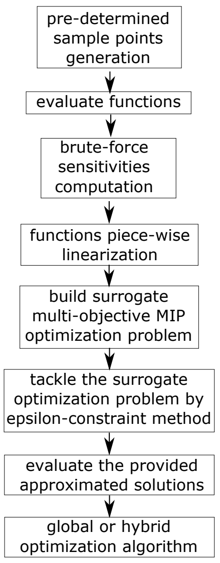
pre-determined sample points generation
evaluate functions
brute-force sensitivities computation
functions piece-wise linearization
build surrogate multi-objective MIP optimization problem
tackle the surrogate optimization problem by epsilon-constraint method
evaluate the provided approximated solutions
global or hybrid optimization algorithm

Figure 3. Flowchart of the proposed methodology.

problem into a desired number of SO-MIP problems where the one of the objectives is modeled as a constraint (Mavrotas, 2009).

Depending on the context and time budget the approach can be used either stand-alone or serving as input for classical (global, hybrid, or local) meta-heuristic algorithms, targeting to improve the accuracy/speed trade-off of the latter.

3.2 Formulation of the MO-MIP Problem Surrogate Model

The underlying assumption of the MIP model is that each function involved in the optimization problem can be approximately expressed, at any point of the search space, as the superposition of n nonlinear functions (which we approximate using piece-wise linearization derived starting from a given base case point) which depend only on one decision variable, where n is the number of decision variables.

The core of the MO-MIP formulation relies on sensitivities information. Sensitivities are computed with respect to a given base case point (e.g. a relevant operating scenario) by altering the value of only one decision variable at the time while freezing the other variables to their initial values, and evaluating the induced change on each function involved in the optimization problem[3]. This process can be repeated for the desired number p of functions' linearization intervals so as to span the whole physical range of each variable.

When seeking for optimal values of decision variables using piece-wise linear functions, the interval to which each optimal value will belong is not known a priori. In order to keep the problem formulation linear and take advantage of existing powerful solvers, we introduce one additional status binary variable b_{ij} per decision variable i and interval j and also split each decision variable i into j variables x_{ij}.

[3]The interested reader is referred to (Igos et al., 2015) for a more comprehensive survey of sensitivity computation methods.

The bi-objective constrained MIP surrogate model of the problem is formulated as follows:

$$\min_{x_{ij},b_{ij}} \{f_1(x_{ij},b_{ij}); f_2(x_{ij},b_{ij})\} \tag{5}$$

$$\text{s.t. } f_1(x_{ij},b_{ij}) = f_1^0 + \sum_{i=1}^{n}\Big[\sum_{j=1}^{p}\big(b_{ij}\underline{f}_{1ij} + s_{ij}^{f_1}(x_{ij} - b_{ij}\underline{x}_{ij})\big) - f_1^0\Big] \tag{6}$$

$$f_2(x_{ij},b_{ij}) = f_2^0 + \sum_{i=1}^{n}\Big[\sum_{j=1}^{p}\big(b_{ij}\underline{f}_{2ij} + s_{ij}^{f_2}(x_{ij} - b_{ij}\underline{x}_{ij})\big) - f_2^0\Big] \tag{7}$$

$$f_1(x_{ij},b_{ij}) \le f_1^{\max} \tag{8}$$

$$f_2(x_{ij},b_{ij}) \le f_2^{\max} \tag{9}$$

$$h_k(x_{ij},b_{ij}) = h_k^0 + \sum_{i=1}^{n}\Big[\sum_{j=1}^{p}\big(b_{ij}\underline{h}_{kij} + s_{ij}^{h_k}(x_{ij} - b_{ij}\underline{x}_{ij})\big) - h_k^0\Big] \ge \underline{h}_k,$$
$$k = 1,\dots,c \tag{10}$$

$$b_{ij}\underline{x}_{ij} \le x_{ij} \le b_{ij}\underline{x}_{ij+1},\ i = 1,\dots,n,\ j = 1,\dots,p \tag{11}$$

$$\sum_{j=1}^{p} b_{ij} = 1,\ i = 1,\dots,n \tag{12}$$

$$b_{ij} \in \{0,1\},\ i = 1,\dots,n,\ j = 1,\dots,p, \tag{13}$$

where p is the number of intervals used in functions' piece-wise linear approximation, n is the number of decision variables, x_{ij} is the value of decision variable i in interval j, b_{ij} is a binary variable indicating whether the decision variable belongs or not to the interval j, f_1 and f_2 are the objectives related to cost and environmental impact respectively, and h_k's are functions describing the water quality.

In this formulation constraints (6) and (7) are piece-wise linearizations of objectives f_1 and f_2, respectively, constraints (10) are piece-wise linearizations of inequality constraints, constraints (8) and (9) impose upper bounds[4] on objectives f_1 and f_2, constraint (11) imposes bounds on decision variables x_{ij} (i.e. if $b_{ij} = 1$ then $\underline{x}_{ij} \le x_{ij} \le \underline{x}_{ij+1}$) and enforces the variable to be set to zero if the optimal value does not belong to its corresponding interval (i.e. $b_{ij} = 0 \rightarrow x_{ij} = 0$), constraint (12) models that fact that each decision variable can belong to only one linearization interval, and constraint (13) models the binary variables.

Note that, compared to the MOO conceptual problem formulation (1)-(4), the above MO-MIP approximation dropped the equality constraints (2), which significantly relieve the MIP formulation. This choice is not only due to the reasons previously mentioned (e.g. unavailability in analytical form) but also as, in our case (as well as many applications[5]), the equality constraints are satisfied whatever the combinations of decision variables values within their physical range.

The MIP optimization problem (5)-(13) contains $n \times p$ binary variables b_{ij} and the same number of continuous variables x_{ij}.

[4]These constraints, which are not necessary for this general MIP formulation, have been included to facilitate the presentation of the MO problem approach via the ε-constraint method.
[5]Admittedly more research is needed to harness potential cases where equality constraints might not be satisfied for some combinations of decision variables values.

3.3 Solution of the MO-MIP Problem by the ε-Constraint Method

The bi-objective MO-MIP problem is tackled using the ε-constraint method (Mavrotas, 2009), which, in order to generate the desired number of solutions m, solves a series of m single objective slightly modified MIP problems, according to the following algorithm:

1. Solve the SO-MIP problem $f_1^{\min} = \min f_1$ subject to (6), (7), (10)-(13). Set $f_2^{\max}$ to the value of the objective f_2 given by (7) at the solution of this optimization problem.
2. Solve the SO-MIP problem $f_2^{\min} = \min f_2$ subject to (6), (7), (10)-(13). Set $f_1^{\max}$ to the value of the objective f_1 given by (6) at the solution of this optimization problem.
3. Assuming, without loss of generality, that the objective f_2 will be modeled as constraint[6]:

   For $l = 3, \ldots, m$, do:

   (a) Set $f_2^{\max} \leftarrow f_2^{\min} + (l-2)(f_2^{\max} - f_2^{\min})/(m-1)$ in constraint (9).

   (b) Solve the SO-MIP problem $\min f_1$ subject to (6)-(13).

4 Overview of Constraint Programming

4.1 Introduction and field of application

Constraint programming (CP) is a logic-based, implicit tree-enumeration search optimization technique which, unlike classical mathematical programming methods (e.g. mixed-integer linear/nonlinear programming), does not rely on integer variables relaxations or derivatives (Rossi, 2006). CP was initially developed for Constraint Satisfaction Problems (CSPs), where only a feasible solution of the problem is sought (i.e. an instantiation of all variables that meets all the constraints). Consequently, CP is also useful in the context of over-constrained (i.e. infeasible) problems, having the ability to find out conflicting constraints (i.e. which cannot be satisfied simultaneously). A survey of industrial applications (Simonis, 1996) reveals that CP has been used to solve a large variety of CSPs of scheduling, routing, resource allocation, car sequencing, and time tabling among others.

Although the major field of application of CP concerns CSPs, it can also be used for optimization problems e.g. by solving a sequence of CSPs which each time looks for a better solution than the best one at hand, until the problem becomes infeasible, the last feasible solution being the optimal one. However, "since CP is purely a search-based technique, it may require considerable amount of computational resource even after the determination of an optimal solution to confirm this optimality" (Kotecha et al., 2010).

4.2 How CP works

An excellent high-level description of how CP works is provided in (Regin, 2004): "Constraint Programming proposes to solve CSPs by associating with each constraint a *filtering algorithm* that removes some values of variables that cannot belong to any solution of the CSP. These filtering algorithms are repeatedly called until no new deduction can be made. This process is called the propagation mechanism. Then, CP uses a *search*

[6]The objective that will be modeled as constraint can be selected as the one with the largest normalized interval variation by comparing the ratios: $f_1^{\max}/f_1^{\min}$ and $f_2^{\max}/f_2^{\min}$ in order to obtain a more even spread of the Pareto front.

procedure (like a backtracking algorithm) where filtering algorithms are systematically applied when the domain of a variable is modified. Therefore, with respect to the current domains of the variables and thanks to filtering algorithms, CP removes once and for all certain inconsistencies that would have been discovered several times otherwise. Thus, if the cost of the calls of the filtering algorithms at each node is less than the time required by the search procedure to discover many times the same inconsistency, then the resolution will be speeded up."

CP relies on powerful constraint propagation algorithms (e.g. for arc consistency, bounds consistency, etc.) to filter out the inconsistent combinations of decision variables for each given constraint and therefore prune the set of possible decisions, squeezing the search space (Van Hentenryck et al., 1992).

CP uses inference in order to reduce the choices to be explored during the search phase. Inference relies on the generation of new problem meta-constraints, called global constraints, which capture relations between a non fixed set of variables encapsulating some existing constraints (e.g. propagation rules). Other two important ingredients to improve efficiency during the search phase are variable ordering (e.g. it is advised starting with variables which are more likely to fail since most time is spent in infeasible sub-trees) and value ordering (e.g. to get earlier upper bounds on an optimization problem) (Haralick and Elliot, 1980). Finally, in order to speed-up computations, the search can be complemented by heuristic local search procedures (Shaw, 1998).

4.3 Pros and cons of CP

The major assets of CP are the expressive and flexible modeling, which allows developing compact models with minimal efforts, the ability to generate global constraints, and the ability to guarantee global optimality irrespective of the problem convexity.

CP is appealing mainly[7] for large scale NP-hard combinatorial problems, for highly constrained problems, and especially in the context of stringent execution requirements (e.g. on-line optimization, limited resources optimization, etc.). Specifically, CP is able to quickly provide a feasible solution if it exists.

On the other hand CP is not suitable for problems where the modeling is not an issue, the optimization is the major difficulty, or "for loosely constrained problems that have limited potential for domain reduction" (Kotecha et al., 2010). The main weakness of CP is the poor global reasoning and thereby objective function estimation as no (ideal) lower bound can be generated.

4.4 CP vs MIP

There is a significant overlap between the field of application of CP and mixed-integer programming (MIP) (Floudas, 1995), both methods being therefore competing and complementary for many problems.

Although both CP and MIP use branching to decompose the problem into tree-search sub-problems, CP prunes the tree to remove infeasible branches using constraint propagation while MIP bounds the tree to remove sub-optimal branches using LP relax-

[7]Another more recent but less explored research stream in CP concerns the optimization problems with continuous variables and nonlinear constraints where contractors-based interval programming analysis is used (Chabert and Jaulin, 2009).

ations (which are solved very efficiently by either the simplex method or the interior point method) and cuts (e.g. Gomory' cutting planes). Then MIP (respectively CP) branches extracting information from LP relaxations (respectively feasibility checks). Next, CP focuses on constrains and feasibility while MIP concentrates on objective and optimality.

CP is reputed to find faster than MIP a feasible solution (or good solutions) for large scale highly constrained combinatorial problems because, if MIP formulation relaxations are inexpressive, MIP may spend a lot of time on solving LP relaxations before actually providing a feasible solution.

In CP the modeling is more natural and easier than in MIP. For instance, compared to MIP, the CP language expressivity allows avoiding painful "big-M" disjunctive constraints (which often have poor relaxations), or increasing artificially the problem size in terms of auxiliary binary variables and constraints. On the other hand MIP has a much better overall perspective on the problem and is able to confirm the solution optimality of the obtained solution or provide the gap to the ideal optimum of the problem.

4.5 Integrating CP and MIP: Constraint Integer Programming (CIP) Framework

Since both methods, MIP and CP, have advantages and drawbacks, in order to take advantage of their strengths and set-off their flaws they can be combined into an integrated approach called constraint integer programming (CIP) (Hooker, 2000; Achterberg et al., 2008). The goal of the latter integrated approach is especially to improve the solution of optimization problems which are unmanageable with either of the two methods alone. Further motivation for this integrated approach is that "in particular, all involved algorithms operate on a single search tree, which enables close interaction amongst these techniques. For example, MIP components can base their heuristic decisions on statistics that have been gathered by CP algorithms or vice versa, and both can use the dual information provided by the LP relaxation of the current subproblem" (Achterberg, 2009). These ideas constitute the foundation of the development of the Solving Constraint Integer Programs (SCIP) solver[8] (Achterberg, 2009). The latter can run in different modes, e.g. full CP mode, full MIP mode, and integrated CIP mode.

5 Numerical Results

5.1 Description of the Case Study, the Simulation Tools, and Experiments Assumptions

The proposed methodology is illustrated using a real-life model of a French PWPP shown in Fig. 1.

The simulations are run on a 2.70GHz/8GB computer. The EVALEAU simulator runs in Umberto®5.6 environment and is linked to the Ecoinvent® v2.2 LCI database for background processes.

The environmental impact objective function focuses only on climate change impact category (Global Warming Potential - GWP - factors based on 100 years time horizon) and is computed using the Midpoint ReCiPe Life Cycle Impact Assessment method (Goedkoop et al., 2009).

[8]SCIP solver user' manual available at http://scip.zib.de/

Table 1. Various MIP surrogate model problems details.

	number of:				
problem	decision variables	linearization intervals	binary variables	continuous variables	constraints
P1	6	2	12	12	38
P2	6	5	30	30	74
P3	6	10	60	60	134
P4	18	2	36	36	98
P5	18	5	90	90	206
P6	18	10	180	180	386

The numerical experiments and comparisons conducted hereafter rely on the following assumptions:

- We consider both a set of 18 decision variables and a subset of it of 6 decision variables.
- We choose approximating the Pareto set by 24 solutions.
- The surrogate model (5)-(13) is solved by SCIP solver (Achterberg, 2009), in two modes CP and CIP, and by the MIP solver GLPK v4.55 (Makhorin, 2014).
- As baseline for performance comparison of the proposed surrogate model we use the benchmark meta-heuristic evolutionary algorithm SPEA2 (Zitzler et al., 2002). The algorithm is run with the default values proposed on PISA platform (Bleuler et al., 2003), using a population of 24 individuals. The comparison criteria used regards the quality of approximated fronts after (ideally) the same number of evaluations.

5.2 CP vs CIP vs MIP

We compare the performances of SCIP solver in both CP and CIP modes with a MIP solver GLPK. In order to enable a fair comparison between the various solver' options the problems are solved in all cases to optimality by assigning a zero value to the integrality gap.

A set of six problems, which differ through the number of decision variables and number of linearization intervals, are used as test bed for this comparison, and Table 1 provides the characteristics of these problems.

Table 2 yields the overall computational time of the three variants. One can observe that, for smaller size problems (i.e. P1 to P4), all variants have comparable computational efforts (here one should keep in mind that this computational effort is negligible compared to the average running time of the simulator, which takes around 70 seconds). However, as the problem size grows (see P5 and especially P6) one can remark that the CP variant scales poorly, the CIP variant scales acceptably, and the MIP variant scales very well. Furthermore, the MIP GLPK solver is the fastest in all six cases.

The poor scalability of the CP is mostly due to the fact that the problem is rather loosely constrained and tough-to-handle continuous variables are present in the formulation.

Table 2. Overall computational time (seconds) to solve 24 different surrogate models according to the ε constraint strategy.

	SCIP		GLPK
problem	CP	CIP	MIP
P1	3.6	2.2	1.2
P2	3.6	3.6	1.2
P3	4.1	2.7	1.3
P4	3.4	2.3	1.2
P5	17.5	4.4	1.5
P6	209.1	5.8	2.4

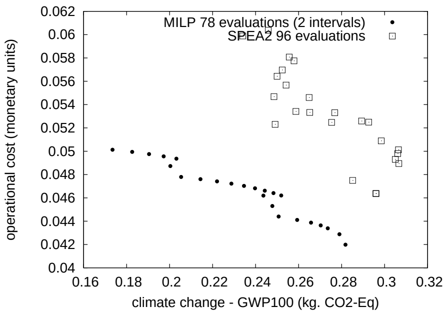

Figure 4. MIP approximation vs. SPEA2: comparison of performances after a close number of evaluations (for 18 decision variables, 2 linearization intervals)

However, one could expect much better performances of CP in cases where the optimization problem is highly constrained.

5.3 Illustration of the proposed approach

The MIP approach depends on the beforehand chosen number of linearization intervals. The latter may reasonably vary between 2 and 10 per decision variable, which means 3 to 11 functions evaluations[9] per variable. In order to assess the influence of the number of linearization intervals on the quality of the front approximation, we performed the MIP approach for the extreme values of 2 and 10 intervals used for linearization. In terms of computational effort the MIP approach with 2 intervals and 10 intervals, respectively, is roughly around 30 and 10 times, respectively, faster than the SPEA2 algorithm for a comparable front approximation quality, which fully justifies it as a stand-alone approach for MOO of computationally intensive black-box model problems.

Note that, compared to one EVALEAU simulator run for functions evaluation which takes in average around 120 seconds, the overall effort of solving to optimality the MO-MIP formulation for optimization part is negligible (i.e. in the order of a few seconds to produce 24 solutions on the front, as shown in Table 2).

We assess now the trade-off between the degree of sub-optimality and the computational effort of the proposed approach. Figures 4 and 5 show for the full set of decision

[9]Recall that e.g. 78 evaluations in the MIP approach (see Fig. 4) correspond to 18 (variables) $\times$ 3 (evaluation points for 2 intervals) + 24 (required Pareto solutions).

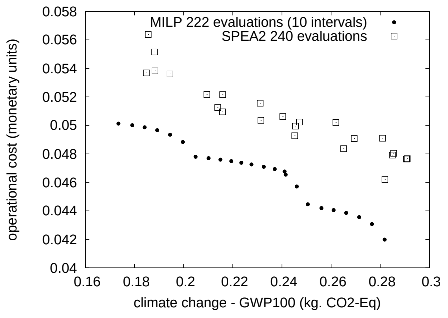

Figure 5. MIP approximation vs. SPEA2: comparison of performances after a close number of evaluations (for 18 decision variables, 10 linearization intervals)

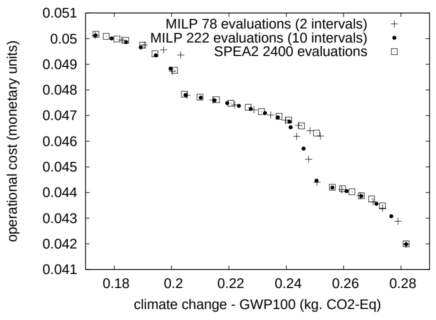

Figure 6. Cost vs. environmental impact optimization for 18 decision variables: MIP approximation vs. SPEA2

variables, the front approximations obtained with both the MIP approach (for 2 and 10 linearization intervals) and SPEA2. Note that, despite a slightly larger number of evaluations, due to its evolutionary algorithm nature, SPEA2 is at an early stage of the search, its solutions looking more like “cloud points” and lagging far away behind the very good front approximations provided by the MIP approach.

Figure 6 shows the Pareto front approximations obtained with both the MIP approach and SPEA2 for the full set of decision variables. One can observe that the MIP approach with 10 linearization intervals produces in both cases only non-dominated solutions which are also well distributed, slightly outperforming SPEA2 in terms of both solutions dominance and spread. The MIP approach with 2 linearization intervals provides mostly Pareto-optimal solutions with a good spread, but it also produces a few dominated solutions. Its performances are slightly inferior to those provided by MIP approach with 10 linearization intervals but comparable with SPEA2. The MIP approach works very well in this case due to for most solutions of the Pareto set most decision variables reach a physical bound.

6 Conclusion

This paper has proposed an approach tailored especially for expensive black-box model multi-objective constrained optimization problems. The approach aims at quickly providing a sufficiently accurate approximation of the Pareto front and relies on MIP surrogate model, taking thereby advantage of powerful existing solvers able to solve the proxy model with negligible computational effort compared to a single simulator run.

The stand-alone MIP approach outperforms the classical state-of-the-art metaheuristic algorithm SPEA2 in terms of front quality approximation with limited computational time budget. Furthermore, in our tests, it provided solutions most of which are either practically Pareto optimal or are located very close to the Pareto front. However, the MIP approach neglects potential nonlinear coupling effects between various decision variables and thereby it may be less efficient in problems where such couplings are strong. Another slight drawback of the MIP approach is the fact it generates trial solutions in a rigid pre-determined way so as to approximate sensitivities in a brute-force fashion. Depending on the computational time budget, the proposed approach can be used as stand-alone or to assist a classical meta-heuristic algorithm serving as an input for the latter aimed to accelerate its convergence.

The paper has also explored the pros and cons of three algorithms for the solution of the MIP surrogate model problems, leading to the conclusion that constraint programming and constraint integer programming algorithms perform less well than MIP algorithms when the problem size grows.

Although the proposed method has been applied, in the context of computational sustainability and sustainable infrastructure, to the bi-criteria (e.g. cost versus life cycle assessment-based environmental impact) optimization of PWPPs at planning stage, it is generic to other application fields addressing (expensive) multi-goal optimization problems, with regard to the problem formulation and operational context (e.g. from operational planning until close to on-line).

Concerning the treatment of the LCA-based environmental metric within MOO problem, the proposed approach represents an important leap forward with respect to the state-of-the-art, since the calculation of the objective scores is based on a fully-fledged LCA (based on the simulation of the operational chain) and not just on the simplified calculation of an environmental metric (e.g. CO2-Eq) based on the link between the optimization algorithm and an inventory table.

Acknowledgments

The authors acknowledge the funding from Luxembourg National Research Fund (FNR) in the framework of the OASIS project (CR13/SR/5871061). The authors also wish to thank Dr. Richard Wallace and Mr. Helmut Simonis, experts in constraint programming from Insight Centre for Data Analytics of the University College Cork, for thoughtful discussions regarding the application of constraint programming to our MOO problem.

References

Achterberg, T. SCIP: solving constraint integer programs. *Mathematical Programming Computation*, pages 1–41, 2009.

Achterberg, T., T. Berthold, T. Koch, and K. Wolter. Constraint integer programming: a new approach to integrate cp and mip. *Proc. of CPAIOR 2008, LNCS 5015*, 2008.

Ahmadi, A. and L. Tiruta-Barna. A process modelling - life cycle assessment - multiobjective optimization (PM-LCA-MOO) tool for the eco-design of conventional treatment processes of potable water. *Journal of Cleaner Production*, 100(1):116–125, 2015.

Ahmadi, A., L. Tiruta-Barna, F. Capitanescu, E. Benetto, and A. Marvuglia. An archive-based multi-objective evolutionary algorithm with adaptive search space partitioning to deal with expensive optimisation problems: application to process eco-design. *Computers & Chemical Engineering*, page in press, 2016.

Azapagic, A. and R. Clift. The application of life cycle assessment to process optimization. *Computers & Chemical Engineering*, 10:1509–1526, 1999.

Bleuler, S., M. Laumanns, L. Thiele, and E. Zitzler. PISA - a platform and programming language independent interface for search algorithms. In *Evolutionary multi-criterion optimization*, pages 494–508. Springer, 2003.

Capitanescu, F., A. Ahmadi, E. Benetto, A. Marvuglia, and L. Tiruta-Barna. Some efficient approaches for multi-objective constrained optimization of computationally expensive black-box model problems. *Computers & Chemical Engineering*, 82:228–239, 2015a.

Capitanescu, F., E. Igos, A. Marvuglia, and E. Benetto. Coupling multi-objective constrained optimization, life cycle assessment, and detailed process simulation for potable water treatment chains. *Journal of Environmental Accounting and Management*, 3(3):213–224, 2015b.

Chabert, G. and L. Jaulin. Contractor programming. *Artificial Intelligence*, 173(11):1079–1100, 2009.

Deb, K. Multi-objective optimization. In *Search methodologies*, pages 403–449. Springer, 2014.

Floudas, C. Oxford university press. In *Nonlinear and Mixed-Integer Programming - Fundamentals and Applications*, 1995.

Gebreslassie, B. H., G. Guillén-Gosálbez, L. Jiménez, and D. Boer. Design of environmentally conscious absorption cooling systems via multi-objective optimization and life cycle assessment. *Applied Energy*, 86(9):1712–1722, 2009.

Goedkoop, M., R. Heijungs, M. Huijbregts, A. De Schryver, J. Struijs, and R. van Zelm. ReCiPe 2008. *A life cycle impact assessment method which comprises harmonised category indicators at the midpoint and the endpoint level*, 1, 2009.

Gomes, C. P. Computational sustainability: Computational methods for a sustainable environment, economy, and society. *The Bridge*, 39(4):5–13, 2009.

Guillén-Gosálbez, G. and I. Grossmann. A global optimization strategy for the environmentally conscious design of chemical supply chains under uncertainty in the damage assessment model. *Computers & Chemical Engineering*, 34(1):42–58, 2010.

Haralick, R. and G. Elliot. Increasing tree search efficiency for constraint satisfaction problems. *Artificial Intelligence*, 14(3):291–313, 1980.

Hooker, J. John wiley & sons. In *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*, 2000.

Igos, E., R. Meyer, E. Benetto, F. Querini, A. Marvuglia, and C. Beaudard. *Abstract book*, chapter Uncertainty and sensitivity analyses in LCA: a review and application to noise characterization. SETAC Europe 25th Annual Meeting, 2015.

ISO-14040. Environmental management - life cycle assessment - principles and framework. Geneva, Switzerland, International Organization for Standardization, 2006.

Jacquemin, L., P.-Y. Pontalier, and C. Sablayrolles. Life cycle assessment (LCA) applied to the process industry: a review. *The International Journal of Life Cycle Assessment*, 17(8):1028–1041, 2012.

Knowles, J. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *Evolutionary Computation, IEEE Transactions on*, 10(1):50–66, 2006.

Kotecha, P., M. Bhushan, and R. Gudi. Efficient optimization strategies with constraint programming. *AIChE journal*, 56(2):387–404, 2010.

Makhorin, A. GLPK (GNU linear programming kit) reference manuel version 4.55, 2014.

Mavrotas, G. Effective implementation of the ε-constraint method in multi-objective mathematical programming problems. *Applied Mathematics and Computation*, 213(2):455–465, 2009.

Méry, Y., L. Tiruta-Barna, E. Benetto, and I. Baudin. An integrated “process modelling-life cycle assessment” tool for the assessment and design of water treatment processes. *Int. J. Life Cycle Assess*, 18(5):1062–1070, 2013.

Parkhurst, D. L. and C. Appelo. Description of input and examples for phreeqc version 3–a computer program for speciation, batch-reaction, one-dimensional transport, and inverse geochemical calculations. Technical report, 2013.

Regin, J. Global constraints and filtering algorithms. *Constraint and Integer Programming Operations Research, Computer Science Interfaces Series*, 27:89–135, 2004.

Rossi, F. In *Handbook of Constraint Programming*. Elsevier, NY, USA, 2006.

Santana-Quintero, L., A. Montano, and C. Coello Coello. A review of techniques for handling expensive functions in evolutionary multi-objective optimization. *Computational intelligence in expensive optimization problems*, pages 29–59, 2010.

SEQEau. Water quality evaluation system in france (version 2). Technical report, http://sierm.eaurmc.fr/eaux-superficielles/fichiers-telechargeables/grilles-seq-eau-v2.pdf, (in French), 2003.

Shaw, P. Using constraint programming and local search methods to solve vehicle routing problems. In *CP98, Principles and Practice of Constraint Programming*, pages 417–431, 1998.

Simonis, H. Problem classification scheme for finite domain constraint solving. In *CP96, Workshop on Constraint Programming Applications: An Inventory and Taxonomy, Cambridge, MA, USA*, pages 1–26, 1996.

Van Hentenryck, P., Y. Deville, and C. Teng. A generic arc-consistency algorithm and its specializations. *Artificial Intelligence*, 57(2-3):291–321, 1992.

Weidema, B. P., C. Bauer, R. Hischier, C. Mutel, T. Nemecek, J. Reinhard, C. Vadenbo, and G. Wernet. Overview and methodology: Data quality guideline for the ecoinvent database version 3. Technical report, Swiss Centre for Life Cycle Inventories, 2013.

You, F., L. Tao, D. J. Graziano, and S. W. Snyder. Optimal design of sustainable cellulosic biofuel supply chains: multiobjective optimization coupled with life cycle assessment and input-output analysis. *AIChE Journal*, 58(4):1157–1180, 2012.

Zhou, A., B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32–49, 2011.

Zitzler, E., M. Laumanns, and L. Thiele. SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization. *Evolutionary Methods for Design, Optimization, and Control*, pages 95–100, 2002.