Computers and Chemical Engineering 106 (2017) 544–558

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

A novel algorithm for fast representation of a Pareto front with adaptive resolution: Application to multi-objective optimization of a chemical reactor

I. Hashem, D. Telen, P. Nimmegeers, F. Logist, J. Van Impe*

KU Leuven, Chemical Engineering Department, BioTeC & OPTEC, Gebroeders De Smetstraat 1, 9000 Ghent, Belgium

ARTICLE INFO

Article history:
Received 22 February 2017
Received in revised form 15 June 2017
Accepted 16 June 2017
Available online 20 July 2017

Keywords:
Multi-objective optimization
Pareto front representation
Divide and conquer strategy
Dynamic optimization

ABSTRACT

Solving a multi-objective optimization problem yields an infinite set of points in which no objective can be improved without worsening at least another objective. This set is called the Pareto front. A Pareto front with adaptive resolution is a representation where the number of points at any segment of the Pareto front is directly proportional to the curvature of this segment. Such representations are attractive since steep segments, i.e., knees, are more significant to the decision maker as they have high trade-off level compared to the more flat segments of the solution curve. A simple way to obtain such representation is the a posteriori analysis of a dense Pareto front by a smart filter to keep only the points with significant trade-offs among them. However, this method suffers from the production of a large overhead of insignificant points as well as the absence of a clear criterion for determining the required density of the initial dense representation of the Pareto front. This paper's contribution is a novel algorithm for obtaining a Pareto front with adaptive resolution. The algorithm overcomes the pitfalls of the smart filter strategy by obtaining the Pareto points recursively while calculating the trade-off level between the obtained points before moving to a deeper recursive call. By using this approach, once a segment of trade-offs insignificant to the decision maker's needs is identified, the algorithm stops exploring it further. The improved speed of the proposed algorithm along with its intuitively simple solution process make it a more attractive route to solve multi-objective optimization problems in a way that better suits the decision maker's needs.

1. Introduction

Biochemical processes are usually dynamic systems that are described by a set of differential equations. The optimization of such processes is carried out using control variables to achieve either one objective, yielding a single objective optimization problem (SOOP) or multiple objectives, yielding a multi-objective optimization problem (MOOP). Finding the optimal control trajectories as a function of time is a situation that is frequently encountered in chemical engineering applications, e.g., finding the time optimal feed rate for a fed-batch reactor. The problem is typically solved by discretizing the continuous control variable into a large number of discrete variables over the time/space interval making it a relatively computationally expensive problem to solve. This makes a multi-objective setting particularly challenging as the control problem has to be repeatedly solved during the solution process. For MOOPs, a set of mathematically equivalent optimal points exists, called the Pareto front. There are two classes of techniques to obtain an approximation of the Pareto set: vectorization and scalarization methods. Vectorization methods (Deb, 2001) are stochastic techniques that tackle the multi-objective optimization problem directly. However, their time consuming nature and the difficulty of incorporating state constraints make them less attractive to be applied in optimal control problems (Logist et al., 2012). On the other hand, scalarization methods (Miettinen, 1999) have been frequently implemented to solve Multi-Objective Optimal Control Problems (MOOCPs). They work by parameterizing the original MOOP into a series of SOOPs. Solving each SOOP yields a point on the Pareto front such that a Pareto front representation can be obtained for the Decision Maker (DM) to examine. It is natural to assume that not all segments of the Pareto front are equally important to a potential DM (Mattson et al., 2004; Antipova et al., 2015). Examples of techniques for the a posteriori analysis of obtained solutions are the order of efficiency filter (Antipova et al., 2015), which ranks

* Corresponding author.
E-mail address: jan.vanimpe@kuleuven.be (J. Van Impe).

http://dx.doi.org/10.1016/j.compchemeng.2017.06.020
0098-1354/

solutions according to how balanced their overall performance is, and the smart filter (Mattson et al., 2004). The motivation for using a smart filter is to emphasize the segments more attractive to the DM in the final representation at the expense of the less significant segments of the curve. The steeper segments, the "knees" of the representation, have higher trade-off level than the more flat "plateau" segments. For a prespecified trade-off level, the filter removes the solutions deemed insignificant to the DM, keeping only solutions which have significant trade-offs between each other. However, the main disadvantage of this approach is the need to produce a dense representation with excess of insignificant solutions for the filter to act on. Considering the large computational cost for solving an instance of a MOOCP, this hinders applying the smart filter strategy to this class of computationally intensive problems. In this paper, an alternative approach is introduced to obtain a Pareto front with adaptive resolution. The novel algorithm utilizes a recursive paradigm in exploring the Pareto front. This way, once a segment of insignificant trade-off level to the decision maker is identified, the algorithm stops generating more solutions within this segment. The paper is structured as follows: in Section 2, the mathematical formulations for solving a MOOCP are introduced. The algorithm's concept of operation is developed in Section 3. Several numerical problems as well as a dynamic benchmark example are presented in Section 4 while the obtained simulation results are discussed in Section 5. Finally, Section 6 summarizes the paper's conclusions.

2. Mathematical formulations

This section is structured as follows: first, the general formulation of multi-objective optimal control problems is discussed. Then, an overview of multi-objective solution algorithms is presented as well as the formulation of a smart filter for the a posteriori analysis of the Pareto front. Finally, Pomodoro, an in-house library used in this paper for solving dynamic optimization problems is introduced.

2.1. Multi-objective optimization formulation

A multi-objective optimal control problem (MOOCP) can be formulated as a minimization problem as follows (Logist et al., 2010):

$$\min_{u(\epsilon),x(\epsilon),p,\epsilon_f} \{J_1, J_2, \ldots, J_m\} \tag{1}$$

subject to:

$$\frac{dx}{d\epsilon} = F(x(\epsilon), u(\epsilon), p, \epsilon_f) \; \epsilon \in [0, \epsilon_f] \tag{2}$$

$$0 = b_i(x(0), p) \tag{3}$$

$$0 = b_t(x(\epsilon_f), p) \tag{4}$$

$$0 \geq c_p(x(\epsilon), u(\epsilon), p, \epsilon) \tag{5}$$

$$0 \geq c_t(x(\epsilon_f), p, \epsilon_f) \tag{6}$$

where m is the number of objectives, ϵ is the independent variable, usually time and typically ranging from 0 to ϵ_f, x are the state variables, u represents the control variables and p the time-invariant parameters of the process. The (nonlinear) model equations are denoted by F. The vectors b_i and b_t represent the initial and terminal conditions, respectively. The vectors c_p and c_t denote the path and terminal inequality constraints. In this work, an individual objective function J_i is generally formulated as follows:

$$J_i = M_i(x(\epsilon_f), p, \epsilon_f) + \int_0^{\epsilon_f} L_i(x(\epsilon), u(\epsilon), p, \epsilon) d\epsilon \tag{7}$$

with $M_i(x(\epsilon_f), p, \epsilon_f)$ the Mayer term, which represents the terminal cost, e.g., the final conversion at the end of the process and $\int_0^{\epsilon_f} L_i(x(\epsilon), u(\epsilon), p, \epsilon) d\epsilon$ the Lagrange term, representing the integral cost over the interval $[0, \epsilon_f]$, e.g., total fuel consumption during the process.

Finally, for conciseness, a vector of the optimization variables is defined as $y = [x(\cdot)^\top, u(\cdot)^\top, p^\top, \epsilon_f]^\top$. The individual objective functions are grouped in a vector as $J(y) = [J_1(y), J_2(y), \ldots, J_m(y)]^\top$ and the set of feasible solutions S is defined as all vectors y that satisfy the imposed constraints (2)–(6) (Logist et al., 2010).

In multi-objective optimization no single optimal solution exists so the notion of Pareto optimality is adopted. As formulated in Miettinen (1999), a vector y^* is said to be Pareto optimal if there exists no other $y \in S$ such that $J_i(y) \leq J_i(y^*)$ for $i = 1, 2, \ldots, m$ and $J_i(y) < J_i(y^*)$ for at least one J_i. A Pareto point is said to be not dominated by any other point in the objective space. This means that y^* is a Pareto optimal point if there exists no other feasible point that would improve a certain objective without causing a simultaneous increase in another objective. Unless all the objectives are not conflicting, an infinite set of solutions will exist. The complete set of Pareto solutions is called the Pareto front, (Miettinen, 1999; Logist et al., 2010).

2.2. Multi-objective optimization solution algorithms

According to a review by Marler and Arora (2004), two major classes of methods exist to obtain a Pareto front: vectorization methods and scalarization methods. Vectorization methods (Deb, 2001) work by solving the multi-objective optimization problem directly using stochastic algorithms. The drawbacks of these methods, as explained in Logist et al. (2010) are their inability to handle complex constraints, being time consuming and being limited to low dimensional search spaces. On the other hand, scalarization methods (Miettinen, 1999) are deterministic and can handle a large number of decision variables and constraints. However, they are prone to converging to local optima. In this class of methods, the multi-objective optimization problem is converted to a series of parametrized single objective optimization problems. This set is typically generated by varying a parameters/weights vector. This way, solving each sub-problem gives a point on the Pareto front. Since, dynamic optimization problems usually involve a high number of constraints, scalarization methods are more suited to solve them, several successful applications can be found in de Hijas-Liste et al. (2014); Telen et al. (2012); Nimmegeers et al. (2016). Three of the most widely used scalarization methods are discussed in this section: *weighted sum* method, *normal boundary intersection* and *(enhanced) normalized normal constraint*.

2.2.1. Weighted sum method (WS)

The (convex) weighted sum method is one of the most widely applied scalarization techniques in practice, mainly due to its simplicity. It is based on combining the multiple objectives into a single convex function composed of their weighted sums as follows Logist et al. (2010):

$$\min_y \sum_{i=1}^{m} w_i J_i(y) = w^\top J(y) \tag{8}$$

where the weights w_i can be grouped in w. Furthermore, $w_i \geq 0$ with $i = 1, 2, \ldots, m$ and $\sum_{i=1}^{m} w_i = 1$. The solution of this minimization problem is obtained at y^*. Since the obtained point is a Pareto optimal solution, it lies on the Pareto front of the feasible objectives space. The procedure of the weighted sum method is solving the minimization problem repeatedly using different combinations of w to obtain multiple points on the Pareto front. However, despite