Applied Soft Computing 16 (2014) 50–62

Contents lists available at ScienceDirect

Applied Soft Computing

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

Discovering driving strategies with a multiobjective optimization algorithm

E. Dovgan [a,*], M. Javorski [b], T. Tušar [a], M. Gams [a], B. Filipič [a]

[a] Department of Intelligent Systems, Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
[b] Faculty of Mechanical Engineering, University of Ljubljana, Aškerčeva cesta 6, 1000 Ljubljana, Slovenia

ARTICLE INFO

Article history:
Received 17 February 2012
Received in revised form
23 September 2013
Accepted 30 November 2013
Available online 8 December 2013

Keywords:
Driving strategy
Traveling time
Fuel consumption
Driving optimization
Multiobjective optimization
Black-box approach

ABSTRACT

When driving a vehicle along a given route, several objectives such as the traveling time and the fuel consumption have to be considered. This can be viewed as an optimization problem and solved with the appropriate optimization algorithms. The existing optimization algorithms mostly combine objectives into a weighted-sum cost function and solve the corresponding single-objective problem. Using a multiobjective approach should be, in principle, advantageous, since it enables better exploration of the multiobjective search space, however, no results about the optimization of driving with this approach have been reported yet. To test the multiobjective approach, we designed a two-level Multiobjective Optimization algorithm for discovering Driving Strategies (MODS). It finds a set of nondominated driving strategies with respect to two conflicting objectives: the traveling time and the fuel consumption. The lower-level algorithm is based on a deterministic breadth-first search and nondominated sorting, and searches for nondominated driving strategies. The upper-level algorithm is an evolutionary algorithm that optimizes the input parameters for the lower-level algorithm. The MODS algorithm was tested on data from real-world routes and compared with the existing single-objective algorithms for discovering driving strategies. The results show that the presented algorithm, on average, significantly outperforms the existing algorithms.

1. Introduction

Driving a vehicle along a given route is a complex process that consists of a series of control actions that are applied to the vehicle by taking into account the vehicle and route states. A set of connections between the states and control actions is a driving strategy. The cost of driving by applying a driving strategy mainly depends on the fuel consumption; however, when the fuel consumption is reduced, the traveling time increases. Consequently, the two objectives, i.e., the traveling time and the fuel consumption, are in conflict with each other. Improving one of the objectives deteriorates the other. Moreover, the extreme reduction of one objective leads to an unacceptable value for the other objective. Therefore, both objectives have to be taken into account simultaneously when constructing a driving strategy.

The existing techniques for discovering driving strategies use single-objective optimization methods in combination with predictive control [1]. They can be divided into two groups: model-based approaches and black-box approaches. Model-based approaches require a knowledge about the applied vehicle model and are usually analytical, while black-box approaches use vehicle models without any knowledge of vehicle operation and are usually numerical. The selection among them mainly depends on whether the knowledge about the vehicle model is available or not, and whether a black-box simulator is available or not. However, the black-box approach is preferable from the user point of view as the knowledge about the vehicle model is usually unavailable.

Model-based techniques aim to minimize either the weighted sum of the fuel consumption and the traveling time, or the fuel consumption only while considering the traveling time as a constraint. To optimize both objectives simultaneously, Huang et al. [2] used constrained nonlinear programming for predictive control, which is a gradient-based method that finds the global optimum if the optimized function is convex. Ivarsson et al. [3], on the other hand, used an analytical method appropriate only for routes with small gradients. Other approaches presented by Melnik [4], and Howlett et al. [5] aim at optimizing only the fuel, or more generally, energy consumption. Both constructed a set of equations describing the vehicle and the environment, and implemented algorithms that calculate the optimal velocity of either a road vehicle [4] or a train [5]. This velocity is obtained by

* Corresponding author. Tel.: +386 1 477 3393.
E-mail addresses: erik.dovgan@ijs.si (E. Dovgan), matija.javorski@fs.uni-lj.si (M. Javorski), tea.tusar@ijs.si (T. Tušar), matjaz.gams@ijs.si (M. Gams), bogdan.filipic@ijs.si (B. Filipič).
URL: http://dis.ijs.si/Erik/ (E. Dovgan).

1568-4946/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2013.11.014

equation differentiation. Similarly, Khmelnitsky [6] presented an analytical approach to calculate the optimal control actions of a train by minimizing the energy consumption on short segments. In addition, he implemented a numerical approach that combines several optimal control actions for short segments to obtain the optimal control actions for longer routes. Akcelik and Biggs [7] used an analytical method to optimize acceleration and deceleration profiles. These profiles aim at minimizing the fuel consumption and the traveled route, and were optimized on short routes, since only acceleration and deceleration phases were tested. Other researchers focused on obtaining good vehicle models to allow for control of the vehicle. Such approaches are suitable when target vehicle behavior is known, in contrast to the previously presented search approaches suitable when the target behavior is unknown. For example, Strum et al. [8] applied neural and fuzzy systems to learn the driving model and control the vehicle. The goal was to drive the vehicle as close as possible to a predefined speed profile. Tanaka and Sano [9] modeled a vehicle with a fuzzy model, and derived control rules from the model. The goal was to control the trajectory of the vehicle along a given reference trajectory.

The black-box techniques mainly use dynamic programming methods to find the driving strategies, as presented in the following examples. The weighted sum of the traveling time and the fuel consumption was minimized by several authors. Monastyrsky and Golownykh [10], and Hooker et al. [11], on the one hand, searched for the global optimum, which can be done only for limited route lengths, while Hellstrom et al. [12], on the other hand, continuously optimized only a finite route length ahead of the vehicle. Other examples try to minimize only the energy consumption, while the traveling time is considered as a constraint. Hellstrom et al. [13] introduced an algorithm that optimizes the driving by controlling the throttle, brakes and gears of a vehicle. In contrast, Johannesson et al. [14] described an algorithm that schedules the charging and discharging of an energy-storage system for a hybrid bus. Researchers also optimized vehicle position on the route. For example, Diehl and Bjornberg [15] optimized vehicle parking in front of a wall. To that end, they implemented a dynamic programming algorithm that minimized the traveling time and distance to the wall.

The previously presented single-objective methods have several disadvantages. Primarily, they find only one driving strategy and have to be used each time the requirements, e.g., the time constraint, change. Moreover, when a weighted sum is used, the driving strategy very much depends on the selected weights [16]. In addition, it is not clear how to select the weights. Therefore, if the driving strategy is not acceptable or when the requirements change, the algorithm has to be restarted with different weights. To find a set of driving strategies that meet the various requirements, a multiobjective method has to be used.

The multiobjective technique finds a set of driving strategies that are incomparable since no driving strategy is better in both objectives than any other driving strategy. Such driving strategies are nondominated [17]. The set of nondominated driving strategies makes it possible to select a different strategy when the requirements change without restarting the algorithm or whenever the objectives are a matter of choice [18]. This is also suitable for users frequently traveling on the same route since each time they can apply a driving strategy with a different trade-off between the traveling time and the fuel consumption based on current requirements. The multiobjective approach is also more convenient than the single-objective approach since no constraints and/or weights have to be specified [19]. Moreover, Van Willigen et al. [20] presented the idea of deploying nondominated driving strategies in adaptive cruise control of future intelligent vehicles. In this case, a user can enter his/her preferences into the vehicle's cruise control at real time. Setting preferences corresponds to real-time selection of the driving strategy with the preferred values of the objectives. Searching for driving strategies by modeling a real vehicle driving on a real route as a black box and using a multiobjective optimization algorithm has not been proposed and evaluated yet.

In this paper we present a two-level Multiobjective Optimization algorithm for discovering Driving Strategies (MODS) on a given route that minimizes the traveling time and the fuel consumption. The lower-level algorithm is a deterministic multiobjective algorithm based on a breadth-first search [21] and Nondominated Sorting Genetic Algorithm (NSGA-II) [17]. The algorithm searches for driving strategies and minimizes the traveling time and the fuel consumption. The upper-level algorithm is a single-objective evolutionary algorithm that searches for the optimal values of the input parameters for the lower-level algorithm. The initial implementation of the lower-level algorithm was presented in [22].

The paper is further organized as follows. The implemented driving simulation is presented in Section 2. Section 3 describes an initial implementation of MODS, called MODS1, and an enhanced version of the algorithm, called MODS2. Section 4 presents the numerical experiments performed with MODS1, MODS2 and two traditional algorithms, i.e., predictive control and dynamic programming. It also presents the obtained driving strategies and the test cases that show how the vehicle state changes when the found driving strategies are applied to the vehicle. The results are discussed in Section 5. Finally, Section 6 concludes the paper with some ideas for future work.

2. Driving simulation

This section presents the black-box vehicle simulation model that simulates the vehicle driving along a predefined route.

2.1. Route representation

A route R is represented as a vector of segments. Each segment R_{i_R} is defined with a touple $\langle s_{i_R}, r_{i_R}, \alpha_{i_R}, v^{\star}_{\mathrm{lim},i_R}\rangle$, where the components are:

- length s_{i_R},
- turning radius r_{i_R},
- inclination α_{i_R},
- velocity limit $v^{\star}_{\mathrm{lim},i_R}$.

The actual velocity limit v_{lim,i_R} depends on the velocity limit $v^{\star}_{\mathrm{lim},i_R}$ and the maximum turning velocity v_{T,i_R} [23]. The maximum turning velocity v_{T,i_R} is the maximum velocity a vehicle can reach without skidding in a turn, and is calculated as:

$$v_{\mathrm{T},i_R} = \sqrt{r_{i_R} g \cos\alpha_{i_R} c_{\mathrm{s}}}, \qquad (1)$$

where g is the acceleration due to gravity and c_{s} is the static friction coefficient. The actual velocity limit v_{lim,i_R} is:

$$v_{\mathrm{lim},i_R} = \begin{cases} v^{\star}_{\mathrm{lim},i_R}; & \text{if } r_{i_R} = \infty, \\ \min\{v_{\mathrm{T},i_R}, v^{\star}_{\mathrm{lim},i_R}\}; & \text{otherwise.} \end{cases} \qquad (2)$$

More precisely, if $r_{i_R} = \infty$, then the route segment R_{i_R} is straight and there is no maximum turning velocity. Otherwise, the segment is a turn and the velocity limit is the lowest value of the segment velocity limit and the maximum turning velocity.

2.2. One-step simulation

The driving simulation simulates the vehicle driving along a predefined route. The requested input data are: