Energy saving in railway timetabling: A bi-objective evolutionary approach for computing alternative running times

Rémy Chevrier, Paola Pellegrini, Joaquin Rodriguez

Energy saving in railway timetabling: A bi-objective evolutionary approach for computing alternative running times

Rémy Chevrier*, Paola Pellegrini, Joaquín Rodriguez

Université Lille Nord de France, IFSTTAR – ESTAS
20 rue Élisée Reclus, 59650 Villeneuve d'Ascq, France

Abstract

The timetabling step in railway planning is based on the estimation of the running times. Usually, they are estimated as the shortest journey duration increased of a short time supplement. Estimating the running time amounts to define the speed profile which indicates the speed that the train driver must hold at each position. The approach proposed in this paper produces a set of solutions optimizing both the running time and energy consumption. The approach is based on an original method of speed profiling performed by a multi-objective evolutionary algorithm. The speed profiles found by the evolutionary algorithm are all compromises between journey duration on the one hand and energy consumption on the other hand. A set of results obtained on two real-life instances are analyzed and discussed to highlight the relevance of such an approach in an operational context.

Key words: Speed profile, Energy saving, Running time, Railway timetabling, Multi-objective, Evolutionary algorithm, Optimization

1. Introduction

Over the years, the increase of traffic volumes in Western Europe and the increase of emission of pollutants have put a stronger and stronger ac-

*Corresponding author
Email addresses: remy.chevrier@ifsttar.fr (Rémy Chevrier), paola.pellegrini@ifsttar.fr (Paola Pellegrini), joaquin.rodriguez@ifsttar.fr (Joaquín Rodriguez)

cent on the development of more eco-aware transportation systems. A direct consequence of this stronger accent is the search for a better management of energy. In railways, this better management may be sought, first of all, in the planning step. Indeed, it is in this step that timetabling is done. As timetabling process is based on the running times between stations, the definition of alternative running times will directly impact, of course, the timetable, but also the whole energy consumption. Estimating alternative running times amounts to modify the train control by defining different sequences of driving regimes. More precisely, defining a sequence of driving regimes consists of the construction of a speed profile. A speed profile indicates the train speed at each position. It is indicated in the roadmap provided for the train driver. The efforts produced by the train to follow the roadmap will directly affect the train's energy consumption. If the acceleration effort is appropriately managed, the energy spent will be reduced while possibly increasing the journey duration. Hence, by modifying the speed profile used for designing the timetable, we may have a more eco-aware system.

For optimizing energy efficiency of train operation, there exist methods searching for a single optimal solution minimizing energy consumption for a limited journey duration. The problem can be solved with an analytical method calculating the sequence of optimal controls and change points such as in [1], or with switch-points and differential equations systems as explained in [2]. Furthermore, due to the critical aspects of real-time railway management, a body of work has been carried out to solve the optimal speed profile according to an available journey time, such as in [3] or in [4] for multi-train scheduling. More recently, a method based on mixed integer linear programming has been proposed to optimize running times [5]. The objective function is a tradeoff between energy consumption and the riding comfort. Trajectory optimization without optimal control theory has been addressed by [6, 7] with dynamic programming. Nevertheless, these methods are designed to provide a single solution to the decision-makers, though they could need more flexibility in the timetabling process.

In order to produce multiple solutions, as far as we know, there are still few multi-objective approaches to optimize speed profiles. Differential Evolution [8] has been used for mass transit systems [9] involving three objectives: punctuality, energy consumption and passenger comfort (by reducing jerks). Another evolutionary method has been developed to perform a speed-based model [10], building speed profiles in a multi-objective way according to a set of predefined rules.

Although estimating the appropriate running time is crucial for the timetabling process, in practice, the running times are usually based on the fastest journey multiplied by an arbitrary factor slightly greater than 1 to have a running time supplement [11]. In general, the time supplement allows the train driver to adapt the speed to traffic. However, the time supplements can be reduced by relaxing the timetables of some specifications (tracks, platforms, time windows, ...) for reducing the use of railway capacity [12].

If perturbations occur, it may happen that the running times have to be modified to take conflicts or delays into account. In [13], the authors propose to forecast delays to produce a distribution of possible running times to be used when the timetable cannot be respected. The authors solve systems of linear differential equations to estimate possible running times. For real-time traffic management, there exist methods for re-optimizing running times at network scale, as in [14].

The time supplement could also be used to save energy. Indeed, we can identify the most energy-friendly speed profile, which benefits of this additional time for using energy-free driving regimes. The speed profile designed in this way differs to the one of the fastest journey in duration (longer) and energy consumption (weaker).

In this paper, we propose an original approach to compute train running times by concurrently minimizing both energy consumption and running time. Usually, the running time is estimated for an entire trip including a set of journeys between stations and the time supplement is spread over the trip [15]. However, as a preliminary work, we consider here the estimation of running time between two stations. Since a trip includes a succession of journeys from one station to another, the approach proposed performs a running time estimation of one journey. This estimation is done by building the speed profile according to a set of rules that we propose to determine the order of driving regimes that the train driver must follow. In particular, a braking must not be followed by an acceleration, because it is absurd from an energy consumption point of view. Moreover, the approach under consideration must be capable of providing a set of tradeoff-solutions for the decision-makers in a single run. In such a way, they will be able to choose a running time adapted to their needs in the timetabling process. Thus, the paper deals with a bi-objective optimization of speed tuning with energy saving. Given that evolutionary algorithms (EA) are well-suited to multi-objective optimization [16], our approach is based on a state-of-the-art multi-objective EA: the Indicator-Based Evolutionary Algorithm (IBEA) [17].

The paper is organized as follows. At first, Section 2 concerns the basic principles of train dynamics and running time estimation used in the optimization model. Then, Section 3 presents the problem under study and its formulation. The algorithms for building a speed profile and evaluating a solution are presented and detailed in Section 4. In Section 5, we present both the principles of multi-objective optimization and IBEA. The specific components of the algorithm are also presented in this section. Section 6 presents two case-studies (including one real railway line), as well as the results of speed profile optimization obtained on these instances. An analysis and a discussion are provided for highlighting the interest of such a method in the planning process. Finally, Section 7 concludes the paper.

2. Running times and train dynamics

In this section, we explain how the running times are estimated and we recall the basic formulas of train dynamics used to perform this estimation. The reader can refer to [18, 19] to have a more detailed explanation of train dynamics. The formulas presented may be modified or replaced without changing the nature of optimization problem defined in the following. Table 1 summarizes the main symbols used in this section.

2.1. Setting sequences of driving regimes

In order to define accurate running times, it is necessary to build speed profiles, which are indicated in the roadmaps that the train driver must follow. According to the theory of optimal control, there are four optimal regimes defined by application of the Maximum Principle (see [20, 2] for details): Acceleration at full power; Cruising at constant speed; Coasting (inertia motion while the engine is stopped); Maximum braking (according to the service braking, softer than emergency braking). Since acceleration is very energy-consuming, the inefficiency of applying unnecessary sequences of braking followed by acceleration is straightforward. Hence, it is a principle of the method that we propose in the paper. In the roadmaps to provide for the drivers, a braking must not be followed by an acceleration.

The problem we deal with consists in designing the most suited speed profile over the path. This path is composed of a sequence of *sections*, in which the speeds have to be tuned. A section is defined by a length and a constant and fixed maximal speed. Consecutive sections always have different maximal speed. Usually, a one-section journey can be divided in four steps

Table 1: Definition of the symbols used in train dynamics

T	journey duration	[s]
E	mechanical energy	[J]
v	train speed	[m/s]
$P(t)$	mechanical power at instant t	[W]
$F_T(v)$	tractive effort, function of speed v	[N]
$F_m(v)$	Maximal tractive, function of speed v	[N]
$F_R(v)$	sum of resistances	[N]
L_R	line resistance	[N]
C_R	curve resistance	[N]
$M_R(v)$	vehicle resistance, function of speed v	[N]
γ	train acceleration	[m/s^2]
b	braking	[m/s^2]
m	train mass	[kg]
ρ	mass correction factor	
β	angle of the slope	
q	gradient of the slope	[‰]
c	radius of the curve	[m]

as depicted in Fig. 1 (we assume there is neither slope nor curve in this example). Let v_m be the maximal speed. First, the train accelerates (A) in order to reach speed v_m as quickly as possible. Then, a cruising phase (Cr) follows during which the acceleration is nil and the traction effort equals the resistance to the train advance. Given that the wheel/rail adhesion is weak, it is common to let the train coast over long distances [21, 22], e.g. points Co(1), Co(2) indicate two positions from which coasting can be started. Coasting from point Co(1) may increase the journey duration a little while reducing the use of mechanical energy. By coasting from point Co(2), the energy consumption may be further decreased with a consequent increase of journey duration. The sooner the coasting starts, the greater the economy, but the longer the journey duration. Point Co(0) indicates the last position from which the train can brake with its normal service braking (B) for being able to stop at the end of the section.

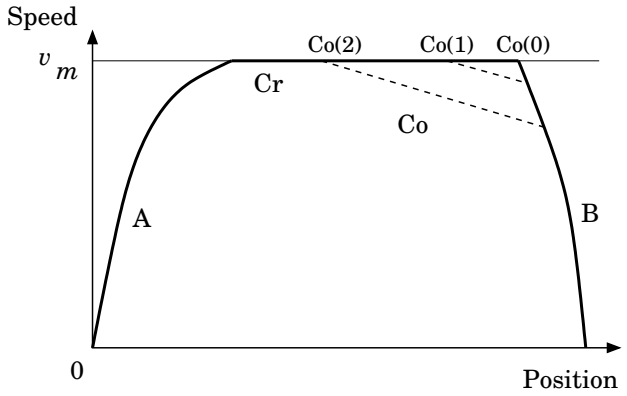


Figure 1: Usual speed profile over one section in four steps (assuming no slope): acceleration (A), cruising (Cr), coasting (Co) and Braking (B).

2.2. Elements of railway dynamics

The fundamental equation of dynamics states the relation between the forces F_T, F_R, speed v, mass m and acceleration γ:

$$F_T(v) - F_R(v) = \rho\ m\ \gamma, \tag{1}$$

ρ being a mass correction factor usually set to $\rho = 1.04$ [22, 2].

2.2.1. Tractive effort

Tractive effort F_T is the effort that the train produces for running, bounded by the maximum tractive effort F_m. The maximum tractive effort that a train can produce is a function of both the train characteristics and the current speed.

2.2.2. Constraints

Train speed, tractive effort and braking are bounded. Let v_m^c be the maximum speed of train c, $F_m(v)$ the maximum tractive effort that the train can exert when traveling at speed v, and b_m the maximum service braking.

$$v \leq v_m^c \tag{2}$$
$$F_T(v) \leq F_m(v) \tag{3}$$
$$b \leq b_m. \tag{4}$$

2.2.3. Resistances

The resistance to the train advance F_R corresponds to the sum of line (L_R), curve (C_R) and vehicle (M_R) resistances:

$$F_R(v) = L_R + C_R + M_R(v). \tag{5}$$

Line resistance L_R depends on the train mass and the slope angle β:

$$L_R = m\ g\ \sin\beta, \tag{6}$$

g being the gravity constant: $g = 9.81$ N/kg. However, line resistance L_R is often approximated as:

$$L_R = m\ g\ q, \tag{7}$$

with $q = \tan\beta$. This is considered a good approximation since, for small values of β as the ones considered, $\sin\beta$ and $\tan\beta$ are very similar. The quantity q is a gradient measured in meters per thousand.

Concerning curve resistance, the value of C_R is approximated by

$$C_R = m\ g\ \frac{700}{c}. \tag{8}$$

Finally, vehicle resistance M_R combines both rolling resistance and air resistance. The former linearly increases as a function of the adhesion and the wheel rims. The latter quadratically increases as a function of the train velocity. Resistance M_R depends on the physical properties of the train and its current speed. In order to simplify its calculation, constants A, B and C, specific to the rolling stock, are introduced to approximate resistance M_R as follows:

$$M_R(v) = A + Bv + Cv^2. \tag{9}$$

2.2.4. Mechanical energy consumption

Let E be the mechanical energy needed to move the train. It can be calculated as the integral of the mechanical power over the running time T [2]. For convenience, let $F(t)$ and $v(t)$ be the tractive effort and the train speed at instant t, respectively:

$$E = \int_0^T P(t)\ dt \text{ with } P(t) = F(t)\ v(t). \tag{10}$$

Power $P(t)$ generated by the train at instant t is calculated in function of the regime adopted, within function `apply_regime` (Algorithm 1 in Section 4.1).

2.3. Description of the driving regimes

As mentioned in Section 2.1, according to the Maximum Principle, four regimes can be adopted by the train, when power recovery (regenerative braking) is not used [7]: acceleration, cruising phase, coasting and braking. Energy consumption and running time evolve differently during each of them.

2.3.1. Acceleration

During this phase, the train accelerates at full power. The delivered power, P, depends on both tractive effort F_T and speed v. Hence, during an acceleration:

$$F_T(v) = F_m(v) \quad (11)$$
$$P(v) = F_T(v)\ v. \quad (12)$$

2.3.2. Cruising phase

This regime consists in maintaining the speed constant, i.e. the acceleration is nil: $\gamma = 0$. In fact, the resistance is counterbalanced by the minimum necessary tractive effort: $F_T(v) = F_R(v)$. In other words, the train must adapt its effort to the resistance either by partially braking or producing an effort according to the gradient and the resistances (line and vehicle).

Since γ results of both the acceleration due to the forces of motion and the braking, we denote a the acceleration due to the forces of motion and b the service braking such that: $\gamma = a - b$.

Let σ be the gradient representing the threshold under which the descent may allow the train to accelerate without effort. We can formalize the effort in the two cases defined below.

1. If $q \geq \sigma$, the train has to produce effort to maintain the speed. First, we determine the resistance $F_R(v)$ and then we set the tractive effort $F_T(v)$ that the train should exert to counterbalance $F_R(v) \geq 0$:

$$F_T(v) = F_R(v) \quad \Leftrightarrow \quad \gamma = 0. \quad (13)$$

   Last but not least, the power delivered can be deduced as follows:

$$P(v) = F_T(v)\ v. \quad (14)$$

2. If $q < \sigma$, the train has to partially brake to maintain the speed. First, we state resistance $F_R(v)$. Given that $F_R(v) < 0$ in this case, we

determine the resulting acceleration $a > 0$ to deduce the braking b necessary to keep $\gamma = a - b = 0$.

$$b = a = \frac{-F_R(v)}{\rho\ m}. \tag{15}$$

Finally, as no power recovery is considered, we set the power delivered as nil:

$$P(v) = 0. \tag{16}$$

2.3.3. Coasting

The coasting corresponds to an inertia motion, while the engine is stopped. The tractive effort is therefore nil:

$$F_T(v) \quad = \quad 0. \tag{17}$$

As a consequence, the energy consumption during coasting is nil and $P(v) = 0$.

2.3.4. Braking

The braking is computed using the maximum service braking b_m. When braking, the tractive effort is therefore nil:

$$F_T(v) \quad = \quad 0. \tag{18}$$

Like in the coasting regime, the energy consumption during coasting is nil and $P(v) = 0$.

3. Problem definition

In order to build a speed profile between two stations, we build the speed profile within each section covered by the train, sequentially. Each section is then decomposed according to a set of speeds for choosing the appropriate driving regimes. The main symbols used in this section are defined in Table 2.

3.1. Objectives formulation

The problem under study can be formulated as a set Φ of two objective functions to be minimized while respecting constraints. The first represents the minimization of journey duration T, and the second the reduction of energy consumption E.

$$\Phi \quad = \quad (\min T, \min E). \tag{19}$$

Objective values (T_u, E_u) of solution u are assessed by `eval_solution`, which is a function defined by Algorithm 3 in Section 4.2.

Table 2: Definition of symbols used in the problem

S	sequence of sections composing the train path
n	number of sections
i	section index: $1 \leq i \leq n$
p_i	starting position of section i with respect to a reference point
l_i	length of section i
$v_{m,i}$	maximum speed in section i (remark that $v_{m,i} > 0$ for all $i = 1, ..., n$)
$l_{f,i}$	length of the first part of section i
$l_{s,i}$	length of the second part of section i
$t_{f,i}$	duration of the first part of section i
$t_{s,i}$	duration of the second part of section i
$e_{f,i}$	energy spent in the first part of section i
$e_{s,i}$	energy spent in the second part of section i

3.2. Speed-based decomposition of a section

3.2.1. Using target-speeds as decision variables

As the train path is decomposed into a set of n sections, the speed profile is successively built in each section. This construction is based on the use of target-speeds, which allow to decompose each section into a sequence of driving regimes. For each section $i = 1, ..., n$ we define the following speeds: $v_{e,i}, v_{x,i}, v_{a,i}, v_{b,i} \in \mathbb{R}$. The speeds $v_{e,i}$ and $v_{x,i}$ are, respectively, the entrance and exit speed of section i and they are determined while building the speed profile. On the other hand, the speeds $v_{a,i}, v_{b,i}$ are the decision variables searched for by the algorithm and they are at the basis of the decomposition of the section.

3.2.2. Principle of decomposition of a section

Speed profiling is done in two phases, each depending on a set of speeds. The main idea consists in splitting the section in two parts: a first part in which the acceleration at full power (the most energy-consuming driving regime) can be used and a second part for using energy-friendly (cruising) or energy-free (coasting, braking) driving regimes.

During the first part, the train enters at speed $v_{e,i}$ and has to reach the first target-speed $v_{a,i}$ by braking or accelerating. Then, during the second part, the train tries to reach speed $v_{b,i}$, initially by coasting. Additional regimes may be used to reach speed $v_{b,i}$ (braking) and complete the rest of the section (cruising). Even if some of these driving regimes are not used, globally, their use follows the order: coasting, cruising, braking. Building the profile between the entrance speed $v_{e,i}$ and the first target-speed $v_{a,i}$ allows to determine the length and the time necessary for the first part: $l_{f,i}$ and $t_{f,i}$, respectively. This building also allows to deduce the length of the second part of the section: $l_{s,i} = l_i - l_{f,i}$. The construction of the second part starts from position $p_i + l_{f,i}$ and depends on target-speeds $v_{a,i}$ and $v_{b,i}$. The produced sequence of driving regimes conducts to determine the exit speed $v_{x,i}$ of section i and, obviously, $v_{e,i+1} = v_{x,i}, i < n$. The details of the construction are explained right below.

Constraints. During the solution construction, we impose Constraints (20) to (22) to the decision variables for each section.

$$v_{a,i} \geq v_{b,i} \qquad \forall i = 1, ..., n \tag{20}$$

$$v_{m,i} \geq v_{a,i} \qquad \forall i = 1, ..., n \tag{21}$$

$$v_{b,i} > 0 \qquad \forall i = 1, ..., n \tag{22}$$

As explained in the following, the value of $v_{b,i}$, $i = 1, ..., n$, may be changed during the evaluation of the objective function, in case the original one results unfeasible.

4. Solution assessment and running time estimation

In this section, the algorithms for building the speed profile and for assessing a solution are given and detailed. But, beforehand, we provide algorithms for calculating train dynamics corresponding to a driving regime. These algorithms are essential to compute distance crossed, energy consumed and time spent during a driving regime. After this description, we will give the algorithms of speed profiling according to the target-speeds defined in each section. In the following, symbols `ac, cr, co, br`, respectively, represent acceleration, cruising, coasting and braking.

4.1. Calculation of driving regime

Based on the description of the possible driving regimes, Algorithm 1 defines function `apply_regime` which determines these driving regimes in function of the characteristics of the track and the train, and also of the speeds given in input. The principle at the basis of this iterative function is to determine efforts, resistances, acceleration, speed, power and energy at each instant t (let Δt be the time-slot).

Since function `apply_regime` needs to be interrupted when changing the driving regime, function `end_reached` (Algorithm 2) indicates when the current regime is implemented. The main reasons to interrupt a regime are either that the target-speed is reached or that the limit position beyond which the regime used must be changed is attained.

4.2. Objectives computation

For computing T and E, we apply the function `eval_solution` described in Algorithm 3. Within this function, T and E are calculated for each section consecutively by the function `eval_section` defined in Section 3.2.

Algorithm 4 describes the function named `eval_section`. Based on the characteristics of a section and the values of the decision variables, this function returns the time and the energy spent in the section itself. Within this function, we use two additional sub-functions `first_part` and `second_part`, which respectively build the speed profile on the first and the second part of the section under consideration (Algorithms 5 and 6 respectively).

Construction of the speed profile in the first part. The first part corresponds to the entrance in the section and depends on two speeds: the entrance speed $v_{e,i}$ and the target-speed $v_{a,i}$. The latter is a decision variable of the problem and it is searched for by the evolutionary algorithm. The construction of the speed profile is carried out through Algorithm 5, which identifies the regime to be used: acceleration if $v_{a,i} > v_{e,i}$, braking otherwise. If the two speeds are equal ($v_{e,i} = v_{a,i}$), length, time and energy spent are nil: $l_{f,i} = 0, t_{f,i} = 0, e_{f,i} = 0$: function `end_reached()` returns 1 in such a case.

Construction of the speed profile in the second part. This part depends on both variables defined for section i, namely $v_{a,i}$ and $v_{b,i}$ and it depends on the gradient, the maximum speed of the following section and the capability to coast all over length $l_{s,i}$. Let l_{co} be the length of coasting, l_{cr} the length of cruising, l_{br} the length of braking. Algorithm 6 describes the construction

Algorithm 1: Function `apply_regime`(v_1, v_2, l, p, r)

Data: v_1: initial speed, v_2: speed to reach, l: distance to cover, p: start position, r: driving regime to use
Result: (t, l, e, R): a vector containing the time spent, the length and the energy used during the motion. R is a vector containing the pairs (p_t, v_t).
Initialization
$t = h$; $l, e = 0$; $R = ()$
$v_t = v_1$; $p_t = p$
begin
  while *not end_reached(v_t, v_1, v_2, p, $p + l$, r)* **do**
    Calculating L_R, C_R, M_R as function of v_t, p_t (Eq. 7, 8, 9)
    $F_R = L_R + C_R + M_R$

    if $r ==$ { *br* ***or*** *co* } **then**
      $F_T = 0$
      if $r ==$ ***br*** **then**
        $b = b_m$
      else
        $b = 0$
    else
      if $r == cr$ **then**
        $F_T = \max(0, \min(F_m, F_R))$
        $b = \max(0, -F_R/\rho.m)$
      else
        $-- r ==$ ac
        $F_T = F_m$
        $b = 0$
    $a = \frac{F_T - F_R}{\rho\ m} + b$
    $v_t = v_t + a\ \Delta t$

    $p_t = p_t + v_t\ \Delta t$
    $l = l + v_t\ \Delta t$

    $e = e + F_T v_t\ \Delta t$

    $R = R \cup (p_t, v_t)$
    $t = t + \Delta t$
end

of the second part. It has to be noted that two additional functions are used in the algorithm. The former is `search_for_intersection` which computes the changing between two regimes by searching for the intersection of the speed curves representing the driving regimes under consideration. The latter is `apply_reverse_regime` which is the counterpart of `apply_regime` but computing the phase from the end-point to the beginning. This function is used when no beginning-point is known for the driving regime to use, but, on the other hand, that the end-point of the phase is known. Given that these two functions can be retrieved easily, they are not defined in this paper.

If $v_{a,i} = v_{b,i}$, the speed profile consists of a cruising phase at speed $v_{a,i}$

Algorithm 2: Function end_reached(v_1,v_2, p_1, p_2, r)

Data: v_1: current speed, v_2: target-speed, p_1: current position, p_2: limit position, r: current driving regime
Result: $reached = \{0|1\}$
begin
  reached = 1;
  switch r **do**
    case *br*
      if $v_1 > v_2$ *or* $p_1 < p_2$ **then** reached = 0
    case *co*
      if $v_1 < v_2$ *or* $p_1 < p_2$ **then** reached = 0
    case *cr*
      if $p_1 < p_2$ **then** reached = 0
    case *ac*
      if $v_1 < v_2$ *or* $p_1 < p_2$ **then** reached = 0
end

Algorithm 3: eval_solution($u = (v_{a,1}, v_{b,1}, ..., v_{a,n}, v_{b,n})$).

Data: for each section $i = 1, ..., n$: $v_{a,i}, v_{b,i}, p_i, l_i, v_{m,i}$
Result: vector (T, E) including the total running time and total energy consumption

$(T, E) = (0, 0)$;
$(T, E) = (T, E)$+eval_section(0, $v_{a,1}$, $v_{b,1}$, $v_{a,2}$, p_1, l_1, $v_{m,1}$, $v_{m,2}$);
for $i = 2, ..., n-1$ **do**
  $(T, E) = (T, E)$+eval_section(min$\{v_{b,i-1}, v_{m,i}\}$, $v_{a,i}$, $v_{b,i}$, $v_{a,i+1}$, p_i, l_i, $v_{m,i}$, $v_{m,i+1}$);
$(T, E) = (T, E)$+eval_section(min$\{v_{b,n-1}, v_{m,n}\}$, $v_{a,n}$, $v_{b,n}$, 0, p_n, l_n, $v_{m,n}$, 0)

Algorithm 4: eval_section($v_e, v_a, v_b, p, l, v_m, v_n$)

Data: v_e: entry speed, v_a: target speed in the first part, v_b: target speed in the second part, p: entry position, l: section length, v_m: maximum speed of the section, v_n: maximum speed of the next section
Result: vector (t, e) including the total running time and total energy consumption in the section

```
1 begin
2   (t_a, l_a, e_a, R_a) = first_part(v_e, v_a, l, p) ;
3   (t_co, t_cr, t_br, e_co, e_cr, e_br, l_co, l_cr, l_br, R_co, R_cr, R_br) = second_part(v_a, v_b, p, l_a, v_m, v_n) ;
4   t = t_a + t_co + t_br + t_cr;
5   e = e_a + e_co + e_br + e_cr;
6 end
```

all along the section if $v_{m,i+1} \geq v_{a,i}$ (Fig. 2(a)), i.e., if the maximum speed of the following section is higher than the current target-speed. Otherwise it consists in a cruising phase at speed $v_{a,i}$ followed by a braking to reach speed $v_{m,i+1}$ (Fig. 2(b)).

Let v_l be the last speed measured at the end of the coasting and returned by function `exit_coast` (not defined in the paper). If $v_{a,i} > v_{b,i}$, we try to

Algorithm 5: first_part(v_e, v_a, l, p)

Data: v_e: entry speed, v_a: target speed in the first part, p: start position, l: section length, v_m: maximum speed of the section
Result: vector (t_a, l_a, e_a, R_a) including the total running time, length, energy consumption and regime used in the section.

```
1 begin
2     if v_e < v_a then
3         (t_a, l_a, e_a, R_a)=apply_regime(v_e,v_a,l,p,ac) ;
4     else
5         (t_a, l_a, e_a, R_a)=apply_regime(v_e,v_a,l,p,br) ;
6 end
```

insert a coasting phase. If $q \geq \sigma$ the train decelerates by coasting, and thus $v_l < v_{a,i}$ because of the slowdown due to the resistive efforts while coasting. Otherwise, there is a steep descent: $q < \sigma$. If the coasting permits to reach $v_{b,i}$, then $v_l = v_{b,i}$. Otherwise, a number of cases must be distinguised to be treated differently. When $v_l < v_{a,i}$, we distinguish four cases depending on the possibility to coast along a distance smaller than or equal to the distance $l_{s,i}$:

- if $v_{b,i} \leq v_{m,i+1}$
  - i. if the train may reach speed $v_{b,i}$, starting at speed $v_{a,i}$, by coasting along the length $l_{s,i}$, the speed profile includes the coasting followed by a cruising regime at speed $v_{b,i}$ in the remaining distance (Fig. 2(c)). The exit speed $v_{x,i}$ equals $v_{b,i}$,
  - ii. if the train covers the section by coasting and it never reaches speed $v_{b,i}$, then we set: $v_{b,i} = v_l$ (Fig. 2(d)). In addition, $v_{x,i} = v_{b,i}$.
- if $v_{b,i} > v_{m,i+1}$
  - iii. if the train may reach speed $v_{b,i}$, starting at speed $v_{a,i}$, by coasting along the length $l_{s,i}$, the same speed profile described in (i) is imposed, but a final braking is necessary to enter the following section at speed $v_{m,i+1}$ (Fig. 2(e)). In this case, $v_{x,i} = v_{m,i+1}$,
  - iv. if the train covers the second part of the section by coasting and it never reaches speed $v_{b,i}$, then a final braking is imposed for attaining this speed (Fig. 2(f)). Let v_c be the speed measured when starting braking, i.e. v_c is obtained after calling function `search_for_intersection`. Since speed $v_{b,i}$ cannot be attained, it is then corrected by replacing it with: $v_{b,i} = v_c$. Last, $v_{x,i} = v_{m,i+1}$.

Algorithm 6: second_part(v_a, v_b, p, l, v_m, v_n)

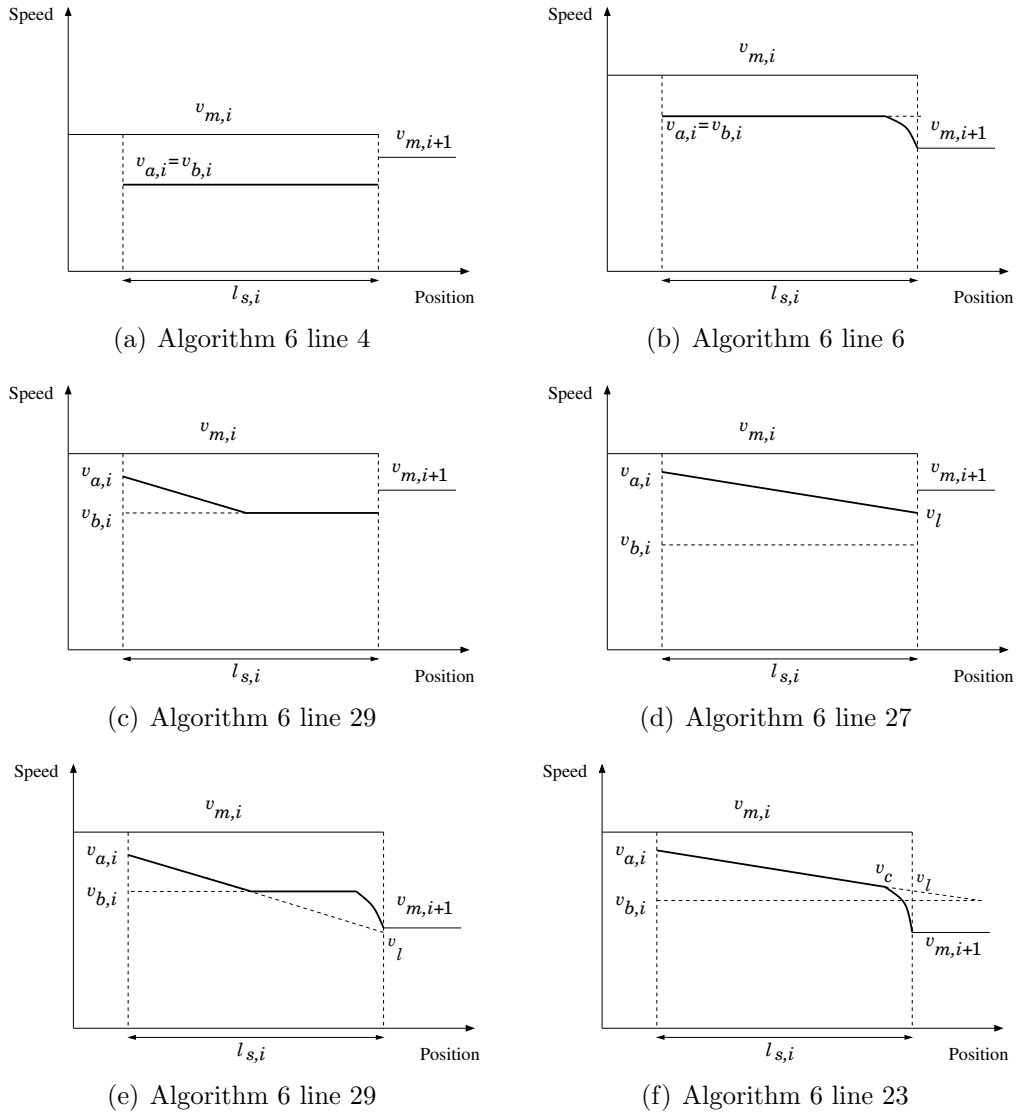
Data: v_a: target speed in the first part, v_b: target speed in the second part, p: start position, l: section length, v_m: maximum speed of the section, v_n: maximum speed of the next section

Result: vector $(t_{co}, t_{cr}, t_{br}, e_{co}, e_{cr}, e_{br}, l_{co}, l_{cr}, l_{br}, R_{co}, R_{cr}, R_{br})$ including the total running time, the total energy consumption and the total run length of each regime used in the second part of the section.

```
1  begin
2      if v_a == v_b then
3          if v_a < v_n then
4              (t_cr, l_cr, e_cr, R_cr)=apply_regime(v_a,v_b,l − l_a,p + l_a, cr) ;
5          else
6              (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_b,v_n,l − l_a,p + l,br) ;
7              (t_cr, l_cr, e_cr, R_cr)=apply_regime(v_a,v_b,l − l_a − l_br,p + l_a, cr);
8      else
9          // v_a > v_b
10         v_l = exit_coast(v_a, v_b, l − l_a,p + l_a);
11         (t_co, l_co, e_co, R_co)=apply_regime(v_a,v_b,l − l_a,p + l_a, co);
12         if v_l > v_a then
13             if v_n ≤ v_b then
14                 (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_m,v_n, l − l_a,p + l, br) ;
15                 (t_co, t_br, l_br, l_co, e_co, e_br, R_br, R_co)= search_for_intersection(R_co,R_br) ;
16                 v_b = v_n ;
17             else
18                 (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_m,v_b, l − l_a,p + l, br) ;
19                 (t_co, t_br, l_co, l_br, e_co, e_br, R_co, R_br)= search_for_intersection(R_co,R_br) ;
20         else
21             if v_l > v_b then
22                 if v_n ≤ v_l then
23                     (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_m,v_n, l − l_a,p + l, br) ;
24                     (R_co, R_br, v_c)= search_for_intersection(R_co,R_br) ;
25                     v_b = v_c
26                 else
27                     v_b = v_l ;
28             else
29                 v_t = min(v_b, v_n) ;
30                 (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_b,v_t, l − l_a,p + l, br) ;
31                 (t_cr, l_cr, e_cr, R_cr)=apply_regime(v_b,v_b,l − l_a − l_co − l_br,p + l, cr);
32     t = t_a + t_co + t_br + t_cr;
33     e = e_a + e_co + e_br + e_cr;
34 end
```

As discussed above, it may happen that a coasting results in an acceleration if $q < \sigma$, in this case:

- if $v_{b,i} \leq v_{m,i+1}$, the coasting is interrupted by a braking to leave the section at speed $v_{b,i}$ (Fig. 3(a)).
- if $v_{b,i} > v_{m,i+1}$, the train stops coasting and brakes to leave the section at speed $v_{m,i+1}$. Speed $v_{b,i}$ is therefore corrected to $v_{m,i+1}$: $v_{b,i} = v_{m,i+1}$



(a) Algorithm 6 line 4

(b) Algorithm 6 line 6

(c) Algorithm 6 line 29

(d) Algorithm 6 line 27

(e) Algorithm 6 line 29

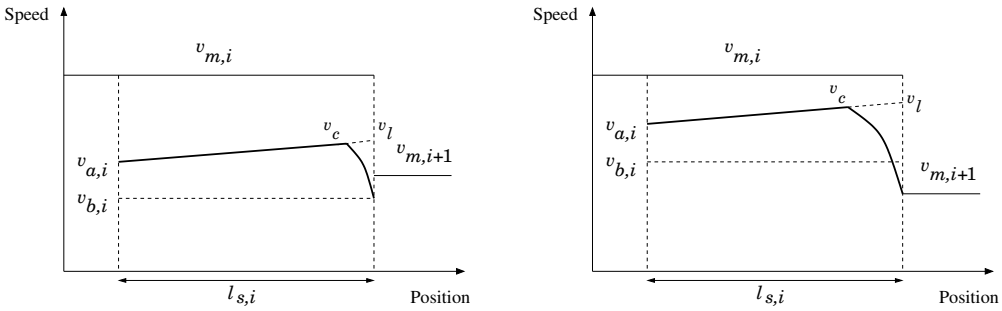
(f) Algorithm 6 line 23

Figure 2: Description of the possible situations in the second part of a section

(Fig. 3(b)).

4.3. Post processing: smoothing the speed profiles

Although the construction of speed profiles aims to avoid sequences composed of braking followed by acceleration, a post-processing is necessary for



(a) Algorithm 6 line 18 (b) Algorithm 6 line 14

Figure 3: Description of particular situations in the second part of a section, when gradient q is negative and the descent is such that a train can accelerate without effort.

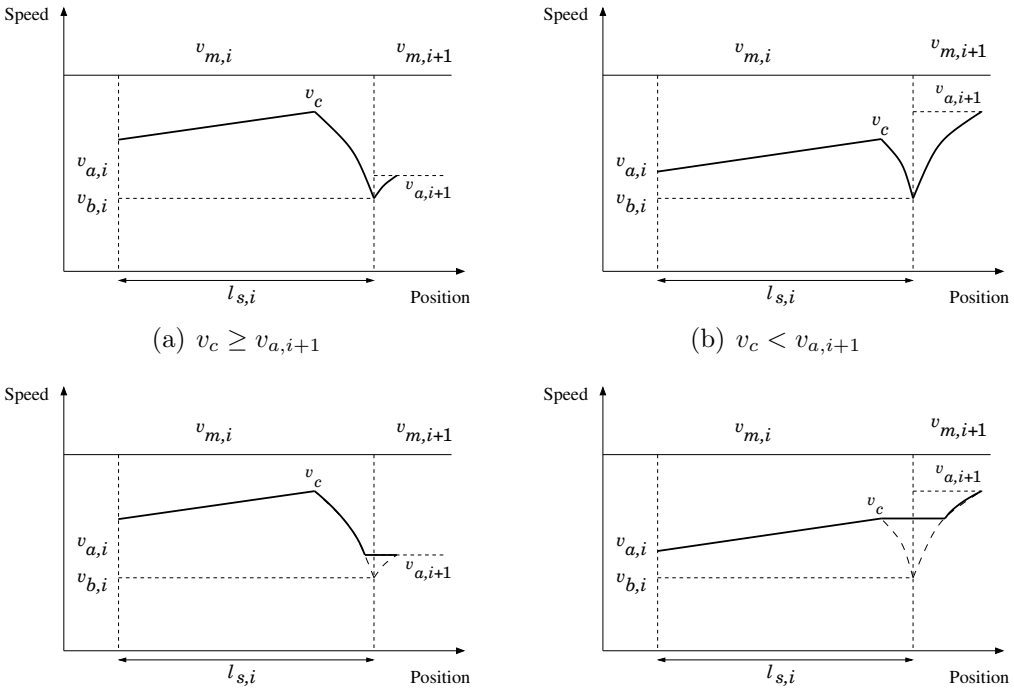
guaranteeing that it is always the case. In fact, if the slope in the second part of the section is sufficiently steep to make the train accelerate while coasting, then a braking is introduced to reach speed $v_{b,i}$. If $v_{b,i} < v_{m,i+1}$, this braking could be followed by an acceleration (if $v_{a,i+1} > v_{b,i}$).

For avoiding this, we use a smoothing post-processing to eliminate sequences: (Braking, Acceleration at full power); (Braking, Acceleration while coasting). Whatever the sequence under consideration, we can distinguish two cases for which we determine a cruising phase replacing one part of the sequence according to speeds v_c (defined as speed measured when starting braking) and $v_{a,i+1}$ (Fig. 4(a, b)). The cruising speed corresponds to the minimum between them: $\min(v_c, v_{a,i+1})$. Finally, Figures 4(c, d) are respectively the smoothed profiles of Figures 4(a, b).

All along this inserted cruising phase, it is necessary to compute the effort necessary to maintain the speed constant. This effort will replace the one previously computed for the acceleration phase in the evaluation of the second objective of the optimization. The same holds for the running time associated to the speed profile. The advantage of inserting a cruising phase instead of an inappropriate sequence is to reduce the journey duration while also reducing the quantity of energy consumed, because acceleration is replaced by a regime far less energy-consuming.

5. Evolutionary Multi-objective Optimization

The problem under study is a multi-objective continuous optimization problem. We propose an evolutionary algorithm to tackle it. This kind of



(a) $v_c \geq v_{a,i+1}$

(b) $v_c < v_{a,i+1}$

(c) Smoothing profile with $v_{cr} = v_{a,i+1}$

(d) Smoothing profile with $v_{cr} = v_c$

Figure 4: Description of smoothing of speed profiles: profiles (a) and (b) have a braking followed by an acceleration (at full power or by coasting in descent); profiles (c) and (d) are the respective smoothed speed profiles.

algorithm is known to be well-suited to multi-objective problems [16]. First, we present multi-objective optimization principles and concepts. Then, we introduce the state-of-the-art evolutionary algorithm which is used in the experimental analysis. Finally, we present the mechanisms specific to our application.

5.1. Multi-objective Optimization

A general Multi-objective Optimization Problem (MOP) can be defined by a set of k objective functions $f = (f_1, f_2, \ldots, f_k)$ and a set U of feasible solutions in the decision space. Let Z be the objective space $Z = f(U)$. Without loss of generality, we assume here that each objective function is to be minimized. To each solution $u \in U$ is assigned an objective vector $z \in Z$ with $z = \{z_1, z_2, ..., z_k\}$ computed on the basis of the vector function $f : U \rightarrow Z$ with $z = f(u) = (f_1(u), f_2(u), \ldots, f_k(u))$. An objective vector $z \in Z$ is

said to *dominate* another objective vector $z' \in Z$ *iff* $\forall i \in \{1, 2, \ldots, k\}$, $z_i \leq z'_i$ and $\exists j \in \{1, 2, \ldots, k\}$ such that $z_j < z'_j$. A decision vector $u \in U$ *dominates* a decision vector $u' \in U$ if $f(u)$ dominates $f(u')$. An objective vector $z \in Z$ is said to be *non-dominated iff* no other objective vector $z' \in Z$ exists such that z' dominates z. A solution $u \in U$ is said to be *efficient*, or *Pareto optimal*, if its mapping in the objective space results in a non-dominated point. Due to the complexity of the underlying problem, the overall goal is often to identify a good approximation of the efficient set. Population-based metaheuristics in general, and evolutionary algorithms in particular, are commonly used to this end, as they are capable of finding multiple and well-spread non-dominated solutions in a single run [16].

5.2. Indicator Based Evolutionary Multi-objective Algorithm

Over the last decades, a very large number of evolutionary algorithms for MOP solving have been proposed in the literature [16, 23]. These approaches can be seen as 'white-boxes' in which problem-related components have to be defined. For this paper, we have used a state-of-the-art evolutionary multi-objective optimization algorithm, namely the Indicator-Based Evolutionary Algorithm (IBEA) [17]. This algorithm follows the main steps illustrated in the flowchart of Fig. 5.

IBEA characterizes the trend in evolutionary computation dealing with indicator-based search which has become popular over recent years. The main idea is to introduce a total order between solutions by means of a binary quality indicator. The fitness assignment scheme is based on a pairwise comparison of solutions from the current population with regard to an indicator I. To each individual u is assigned a fitness value $\phi(u)$ measuring the 'loss in quality' if u was removed from the current population Q, *i.e.* $\phi(u) = \sum_{u' \in Q \setminus \{u\}} (-e^{-I(u',u)/\kappa})$, where $\kappa > 0$ is a user-defined scaling factor. The variation step comprises recombination (crossover) and mutation. The selection for crossover consists of a binary tournament between randomly chosen individuals and the selection for replacement consists in iteratively removing the worst solution from the current population until the required population size is reached. The fitness information of the remaining individuals is updated whenever one is deleted. Moreover, all new non-dominated solutions found during the process are archived in a separate population. This archived population is updated every iteration in function of the new non-dominated solutions for discarding the dominated ones.

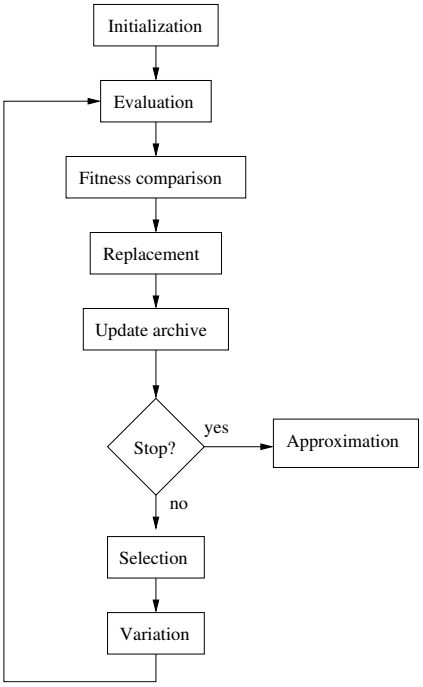


Figure 5: Flowchart of an evolutionary multi-objective optimization algorithm.

5.3. Solution representation and initialization

A solution is defined by a vector of speeds: $\langle v_{a,1}\ v_{b,1}\ ...\ v_{a,n}\ v_{b,n} \rangle$. Given that two speeds are necessary to represent a section, the number of components of solution u equals twice the number n of sections: $\#u = 2n$.

To avoid too many unfeasible solutions at the beginning of the optimization, a specific initialization strategy is developed based on the fastest journey, as described in 5.3.2.

5.3.1. Determination of the reference solution

In order to have a reference solution for further comparisons, we search for the solution which minimizes journey duration, denoted u^*. Concretely, it consists in driving as fast as possible with respect to the maximum speeds of the track.

A complete description of this calculation is given in [19]. In few words, the speed profile is built in three steps. First, the method consists in determining all necessary braking at the end of the sections for respecting the

maximal speed of consecutive sections. Second, it consists in determining the maximal acceleration at the beginning of each section. Third, cruising phases are added between accelerations and brakings to complete the speed profile. The obtained solution represents the upper-bound of energy consumption and the lower-bound $\underline{T}$ of journey duration. The decision-maker will be able to limit the possible range of journey duration by upper-bounding it to a duration equal to $x \times \underline{T}$, by setting parameter $x > 1$.

5.3.2. Initialization of the population

The solutions are based on solution u^* and are successively initialized. The initial population as well as the following ones are composed of a fixed number N of solutions. Within each initialization of solution $\mu \in [1, N]$, the values of $v^{\mu}_{a,i}, v^{\mu}_{b,i}$ are determined from $v^{u^*}_{a,i}, v^{u^*}_{b,i}$ in such a way that the solution initialized is automatically longer and less energy-consuming than the reference solution u^*. At every solution initialization, the solution is longer than the previous one. In our implementation, we have chosen a very easy way to do this. First, we assume that the population is limited to 100 solutions: $N \leq 100$. Then, the interval I_i for decreasing speeds $v^{\mu}_{a,i}, v^{\mu}_{b,i}$ per section i corresponds to half the maximal speed reachable:

$$\forall i \leq n, \quad I_i = \frac{v^{u^*}_{a,i}}{2} \tag{23}$$

$$\forall \mu \in [1, N], \forall i \leq n, \quad v^{\mu}_{a,i} = v^{u^*}_{a,i} - (I_i \times i \times 0.01) \tag{24}$$

$$\tag{25}$$

This strategy is clearly improvable. Nevertheless, it is satisfactory for producing original solutions and the development of a new initialization strategy is out of the scope of the paper. After initialization, the population is composed of a set of N distinct solutions, which will be used to produce new and improved ones.

5.4. Solution evaluation

Each solution u is evaluated during its construction. In particular, for each section i, the running time t_i and the energy consumption e_i are computed as explained in Section 3.

If a speed profile does not satisfy the constraints or cannot be built, then the solution is not feasible and it is discarded. As the objectives have to be minimized, their fitnesses are assigned huge values so that the solution will not appear in the next population.

5.5. Solution variation

Variation step includes two operations: crossover and mutation. Crossover is the mechanism allowing solutions to recombine with each other in order to produce new solutions. As the solution space is continuous, we use an operator adapted to the continuous search: the Simulated Binary Crossover [24], which generates two new solutions from two belonging to the current population. Its mechanism is inspired from the single-point crossover used in the binary-coded genetic algorithm [24]. A crossover rate indicates the percentage of individuals to recombine.

For selecting solutions to cross, a binary tournament is operated. It consists in randomly selecting two solutions in the population and to keep the one with the best fitness. A second binary tournament is operated to select the second solution to cross.

Mutation consists in providing diversity for the population by modifying a solution randomly chosen in the population. For the same reason mentioned for the crossover, we use an operator adapted to the continuous search: a polynomial mutation [16, 24]. A mutation rate indicates the percentage of individuals to mutate.

6. Experimental analysis

6.1. Implementation

We implemented the algorithm by using the ParadisEO framework [25]. The ParadisEO framework is a 'white box' in which several algorithmic components are already implemented, and must be combined and integrated by the user. In addition to the problem-related modules that we have developed, we use the ParadisEO implementations of SBX operator and polynomial mutation. We performed the experiments on a PC (3.0 GHz with 6 GiB) running Linux release of framework ParadisEO.

Parameter settings. The population is composed of 50 solutions and evolves over 60 seconds of computation, which is the stopping criterion. Crossover and mutation rates are respectively set to 0.9 and 0.5. Specific parameter κ for IBEA is set to 0.0001. We selected these values based on some preliminary experiments.

6.2. Instances and rolling stock used in the analysis

The instances that we use for the experiments represent two real lines described in Figure 6(a) which reports the length and the maximal speed of each section. The track slopes are depicted in Figure 6(b, c) and given as gradient [‰].

Instance 1 corresponds to a line described and used in [6, 7]. It is 2.2km long and includes five sections. Instance 2 is the *Saint-Étienne–Rive de Giers* line in France. It is 20.2 km long and includes five sections. It is interesting to note that the gradient is mostly negative and thus the journey is in descent for the most part of the line.

In both instances, the train is an AGC[1]. For computing energy consumption, the relevant parameters about AGC, as well as the tractive effort curve, are reported in Fig. 7.

6.3. Results

Here, we analyze the results obtained on the two instances tackled in this paper, separately detailed and discussed in the cases studied hereafter.

6.3.1. Case study 1

Figure 8(a) depicts the objective space of Instance 1 and the sets of produced solutions at different times. Three populations are represented: the initial one (time t=0), the sets of non-dominated solutions produced at time t=30 and the corresponding set at the end of the process (time t=60). The reference solution I1* corresponds to the fastest journey and is represented to have a basis of comparison. Note that the limit of journey duration is fixed to $1.5 \times \underline{T_{I1}} = 1.5 \times T_{I1^*}$.

Compared to the initial solutions, it clearly appears that the sets of solutions improved during the process. Moreover, according to the stretched shape of the sets, we can say the solutions have been well diversified during the search. Now, if we compare the set at time t=30 with that at time t=60, we can see that the solutions have not been strongly improved during the last 30 seconds. Hence, with the adopted parameter settings, the most part of optimization has been done during the first half of the available time.

In Figure 8(c), three speed profiles are drawn: reference solution I1*, and two others: I1_1 and I1_2. Solutions I1_1 and I1_2 are two alternative

[1] *Autorail Grande Capacité* constructed by Bombardier Inc. http://www.bombardier.com

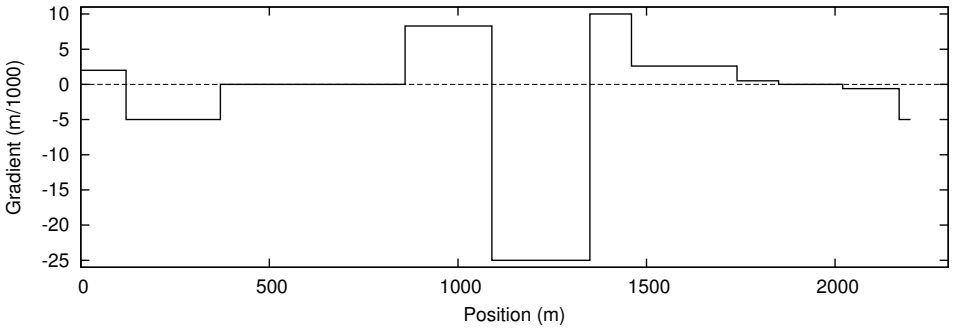
Instance 1

section	1	2	3	4	5
length (m)	1,350	160	250	240	200
maximum speed (km/h)	95	70	40	25	40

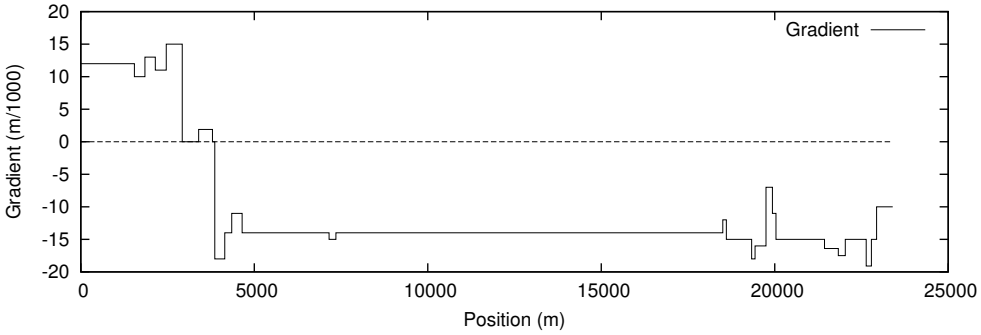
Instance 2

section	1	2	3	4	5
length (m)	3,500	3,900	3,900	6,600	2,300
maximum speed (km/h)	90	110	105	120	105

(a) Length and maximum speed allowed on each section



(b) Description of the Instance 1 gradient

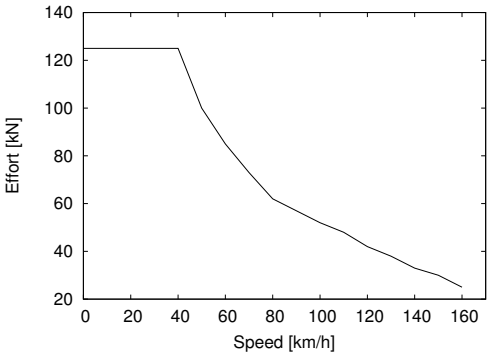


(c) Description of the instance 2 gradient

Figure 6: Description of the studied instances

m	135,000	kg
v_m^c	160	km/h
a_m	1.21	m/s^2
b_m	0.6	m/s^2
A	2540	
B	33.44	
C	0.572	

(a) Numerical data



(b) Tractive effort curve

Figure 7: Technical parameters of train AGC

solutions obtained in the same run. Duration and energy consumption of each solution, as well as deviations from solution I1*, are reported below. As could be assumed, percentage decrease of energy consumption is much higher than the percentage increase of duration, as depicted in Table 3.

Table 3: Numerical results of three solutions obtained on instance 1: reference solution I1* and two alternative solutions I1_1, I1_2.

Solution	Duration [s]	Dev. [%]	Energy [kJ]	Dev. [%]
I1*	175		4.9790×10^6	
I1_1	189	+8.0	3.6917×10^6	-25.85
I1_2	213	+21.7	2.6327×10^6	-47.12

By analyzing the speed profile of solution I1_1 (Figure 2), we can observe that, in the first section, after an acceleration at full power for attaining $v_{a,1} = 21$ m/s, the train coasts for reaching target-speed $v_{b,1}$. Then, in the second part, we can see that the train accelerates between positions 1,100m and 1,300m due to the steep descent of the track even if $v_{b,1} = 18.48$ m/s. In this context, the train cannot reach the target-speed and the situation identified here corresponds to that described in Algorithm 6 line 18 (Fig. 3(a)). In the second section (from 1,350m to 1,510m), a braking covers the

whole available distance. It corresponds in fact to a braking when the train enters in the section, followed by the same situation as before (Fig. 2(e)), but with coasting and cruising distances equal to 0. In the third section, the train enters at speed $v_{e,3} = v_{a,3} = 11$ m/s so that the first part is nil. In the second part, the train coasts then brakes to leave at the maximum speed of the next section. With speeds $v_{a,3} = 11$ m/s and $v_{b,3} = 9.76$ m/s, that corresponds to the situation described in Algorithm 6 line 27 (Fig. 2(e)) with a cruising distance equal to 0. Finally, the train leaves section 3 at speed $v_{x,3} = 6.94$ m/s and enters section 4 at speed $v_{e,4} = v_{x,3}$. In the fourth section, the train drives at constant speed (see Fig. 2). Last, in the fifth section, the train accelerates to reach the first target-speed before coasting until the compulsory braking when arriving at the end of the path (see Fig. 3(b)).

The speed profile of solution I1_2 follows approximately the same driving regimes but with lower target-speeds. Its consumption is weaker compared to solution I1_1 but with, of course, a longer journey duration.

Through this example, the relevance of a multi-objective approach of speed profiling optimization is highlighted. Indeed, in a short computation time, the approach has been capable of producing a set of distinct solutions. Train-practitioners will have the possibility to choose a solution adapted to their needs instead of working with the shortest running time increased of a supplement. In such a way, the proposed approach would help them to decide what a good tradeoff is between time supplement and energy consumption in the planning under consideration.

6.3.2. Case study 2

The main difference compared to case study 1 is the presence of steep descents in the track profile. The interest of using a smoothing method will be highlighted in such a context to produce optimized speed profiles.

Figure 8(b) presents the sets of solutions at different times (t={0, 30, 60}). As for case study 1, the initial population (t=0) is compared to populations at t=30 and t=60. The reference solution I2* is also represented to have a basis of comparison. The journey duration is limited to $1.5 \times T_{I2*}$. Similarly to what observed in case study 1, the comparison of the three sets clearly indicates that the most part of optimization is done during the first half of the computation.

The gap in energy consumption between the reference solution and the others can be explained by the particular topology of the track. In this

example, given that the train can move with low effort by using coasting, the engine can be utilized only little, so that the consumption falls dramatically.

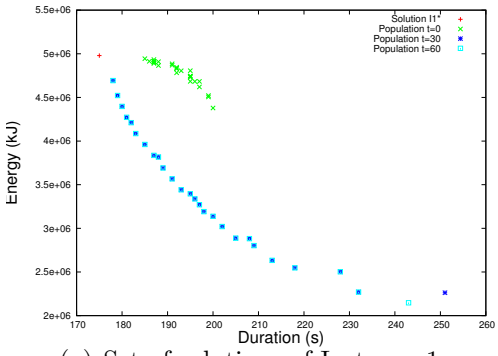
In order to highlight this explanation, we can focus on the speed profile of solution I2_1 in Figure 8(d). A coasting is introduced in all sections except the first. Moreover, still with the exception of the first section, the train does not use very much the acceleration regime to increase speed: indeed it can accelerate by coasting in descent. As soon as the train reaches the maximum speed by coasting in sections 2, 3, 4, 5, it must partially brake to maintain the speed, whereas in solution I2* the train has to accelerate at full power to reach the maximum speed. It is the reason for which solution I2* requires far more energy than the others proposed.

The acceleration occurring while entering in section 3 is due to the model: $v_{e,3} = v_{x,2} < v_{a,3}$, and the sequence (Braking, Acceleration) is smoothed according to the method explained in Section 4.3. Indeed, given that the coasting from speed $v_{a,2}$ to $v_{b,2}$ is accelerating, it is necessary to brake for exiting section 2 at speed $v_{b,2}$. Then, in section 3, speed $v_{a,3}$ is reached after accelerating as the model rules. In such a case, due to the negative gradient, the model produces a sequence of driving regimes which will need to be smoothed: (Acceleration while coasting, Braking, Acceleration). In the same way, the accelerating effect in coasting occurring in sections 4 and 5 can also be smoothed according to the same method. The smoothed speed profile, denoted I2_1s, is depicted in Figure 8(d).

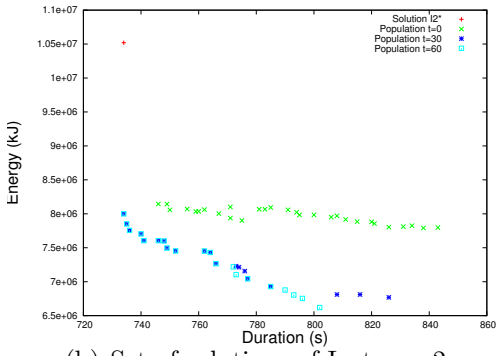
The results reported in Table 4 show the duration increase and energy savings occurring while using energy-free driving regimes such as coasting or cruising with partial braking. As for case study 1, the approach presents a good capacity of producing a set of distinct solutions in a short computation time. Its relevance in decision-helping is shown in so far as it could help train-practitioners to build timetables.

Table 4: Numerical results of two solutions obtained on instance 2: reference solution I2* and one alternative solutions I2_1 and its smoothed couterpart I2_1s.

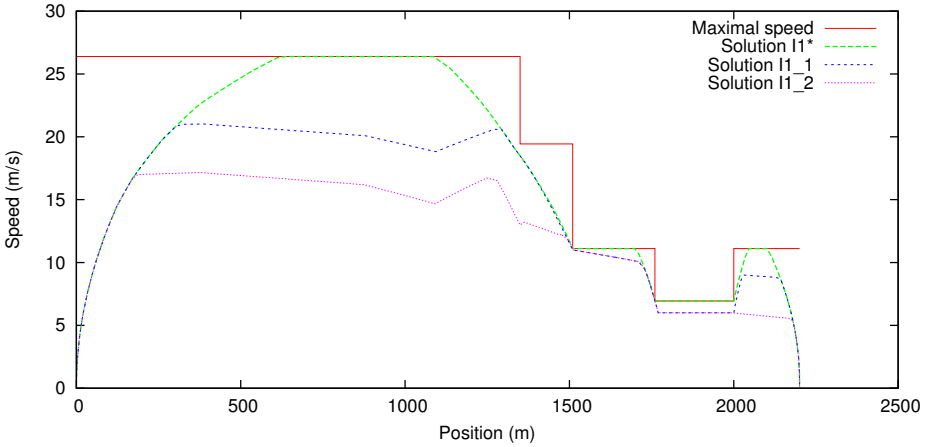
Solution	Duration [s]	Dev. [%]	Energy [kJ]	Dev. [%]
I2*	734		10.5194×10^6	
I2_1	766	+4.35	7.26566×10^6	-30.93
I2_1s	760	+3.54	6.61866×10^6	-37.08



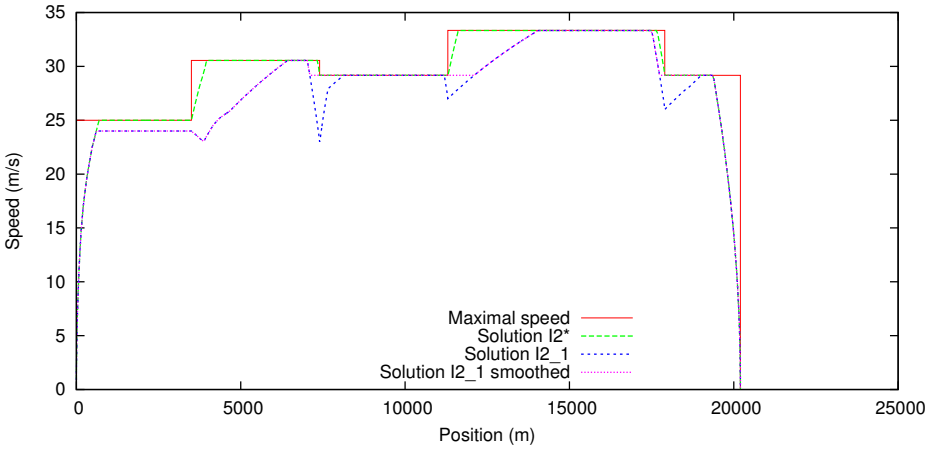
(a) Set of solutions of Instance 1



(b) Set of solutions of Instance 2



(c) Instance 1: Reference solution and two tradeoffs



(d) Instance 2: Reference solution and one tradeoff (original and smoothed)

Figure 8: Results obtained: Fronts of solutions at times t={0, 30, 60} on instances 1 (a) and 2 (b); Examples of speed profiles on instances 1 (c) and 2 (d).

7. Conclusion and perspectives

In this paper, we consider a particular step of the train planning. This step preceding the timetabling concerns the estimation of running times. This estimation is directly related to the construction of speed profiles, which indicate the speed that the train must hold at each position. In this work, as power recovery is not integrated, this problem is addressed with an approach using conventional driving regimes according to the Maximum Principle: acceleration at full power, cruising, coasting and maximum braking.

Since timetabling process uses running times increased of a short time as supplement to prevent disturbances, the approach proposed here consists in defining a running time directly adapted to the needs of planning and optimizing also the energy consumption. Hence, one major contribution of this work is to provide for the practitioners the capability of choosing the solution the most adapted to their needs directly among a set of compromise solutions.

In order to build a set of speed profiles adapted to a track, a specific model has been developed. From two target-speeds defined for each maximal constant speed section of the track, the model builds the speed profile within each section. Then, the optimization is performed by an evolutionary multi-objective algorithm.

The optimization model is independent of the train dynamics model that we implemented. Indeed, due to the controversial aspects of some approximations, anyone can use different train dynamics without changing neither the speed profiling method nor the optimization model. That is clearly an advantage for further comparisons of dynamics models in the future.

For illustrating the relevance of the approach, two real-life case studies have been proposed. They have been tackled with a multi-objective optimization algorithm, which has produced a set of solutions in a short time (30 seconds).

In the future, we will integrate the estimation of alternative running times in a planning context while building timetables. That will imply to consider whole trips and not only point-to-point journeys. This integration would allow the train-practitioners to consider the energy spent per timetable produced with regards to the service (dwell and transfer times, connections). The production of alternative timetables optimizing both energy and quality of service will constitute the next step of this research.

References

[1] R. Liu, I. Golovitcher, Energy-efficient train control, Transportation Research part(A) 37 (2003) 917–932.

[2] T. Albrecht, Railway Timetable & Traffic – Analysis, Modelling, Simulation, Eurail press, 2008, Ch. Energy-Efficient Train Operation, pp. 83–105.

[3] M. Lüthi, Improving the efficiency of heavily used railway networks through integrated real-time rescheduling, Ph.D. thesis, ETH Zurich (2009).

[4] T. Albrecht, S. Oettich, Computers in Railways VIII, WIT Press, Southampton, 2002, Ch. A new integrated approach to dynamic schedule synchronization and energy saving train control, pp. 847–856.

[5] Y. Wang, B. De Schutter, B. Ning, N. Groot, T. J. van den Boom, Optimal trajectory planning for trains using mixed integer linear programming, in: ITSC'2011, 14th Int. IEEE Conf. on Intelligent Transportation Systems, Washington, DC, USA, 2011, pp. 1598–1603.

[6] H. Ko, T. Koseki, M. Miyatake, Application of dynamic programming to optimization of running profile of a train, in: Computers in Railways IX, 2004, pp. 103–112.

[7] M. Miyatake, H. Ko, Optimization of train speed profile for minimum energy consumption, IEEJ Transactions on Electrical and Electronic Engineering 5 (2010) 263–269.

[8] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization (11) (1997) 341 – 359.

[9] C. Chang, D. Xu, H. Quek, Pareto-optimal set based multiobjective tuning of fuzzy automatic train operation for mass transit system, IEE Proceedings–Electric Power Applications 146 (5) (1999) 577–583.

[10] R. Chevrier, An evolutionary multi-objective approach for speed tuning optimization with energy saving in railway management, in: ITSC'2010, 13th Int. IEEE Conf. on Intelligent Transportation Systems, Madeira (Portugal), 2010, pp. 279–284.

[11] L. Kroon, R. Dekker, M. Vromans, Cyclic railway timetabling: A stochastic optimization approach, in: F. Geraets, L. Kroon, A. Schoebel, D. Wagner, C. Zaroliagis (Eds.), Algorithmic Methods for Railway Optimization, Vol. 4359 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2007, pp. 41–66.

[12] A. D'Ariano, D. Pacciarelli, M. Pranzo, Assessment of flexible timetables in real-time traffic management of a railway bottleneck, Transportation Research part(C) 16 (2008) 232–245.

[13] T. Huisman, R. J. Boucherie, Running times on railway sections with heterogeneous train traffic, Transportation Research part(B) 35 (2001) 271–292.

[14] A. D'Ariano, T. Albrecht, Running time re-optimization during real-time perturbations, in: Computers in Railways X, Prague, Czech Republic, 2006, pp. 531–540.

[15] R. Goverde, Punctuality of railway operations and timetable stability analysis, Ph.D. thesis, TU Delft (2005).

[16] K. Deb, Multi-objective Optimization using Evolutionary Algorithms, Wiley, 2001, 517 p.

[17] E. Zitzler, S. Künzli, Indicator-based selection in multiobjective search, in: Parallel Problem Solving from Nature - PPSN VIII, Vol. 3242/2004, Springer, Berlin/Heidelberg, 2004, pp. 832–842.

[18] J. Pachl, Railway Operation and Control, VTD Rail Publishing, 2004, 239 p.

[19] O. Brünger, E. Dahlhaus, Railway Timetable & Traffic – Analysis, Modelling, Simulation, Eurail press, 2008, Ch. Running Time Estimation, pp. 58–82.

[20] P. Howlett, P. J. Pudney, Energy-efficient train control, Springer, 1995, 306 p.

[21] D. Lancien, M. Fontaine, Calculs de marches de train économisant l'énergie de traction – le programme MARECO, Revue Générale des Chemins de Fer (1981) 679–692.

[22] S. Iwnicki (Ed.), Handbook of Railway Vehicle Dynamics, Taylor and Francis Group, 2006, 535 p.

[23] E. G. Talbi, Metaheuristics: from design to implementation, Wiley, 2009, 624 p.

[24] K. Deb, R. B. Agrawal, Simulated binary crossover for continuous search space, Complex Systems 9 (1995) 115–148.

[25] A. Liefooghe, L. Jourdan, E.-G. Talbi, A software framework based on a conceptual unified model for evolutionary multiobjective optimization: ParadisEO-MOEO, European Journal of Operational Research 209 (2) (2011) 104–112.