Computing multiple running times for railway timetabling: a speed-level based model for constructing alternative speed profiles

Remy Chevrier, Paola Pellegrini, Joaquin Rodriguez

Computing multiple running times for railway timetabling: a speed-level based model for constructing alternative speed profiles

Rémy Chevrier, Paola Pellegrini, Joaquín Rodriguez
Université Lille Nord de France, IFSTTAR
20 rue Élisée Reclus, 59666 Villeneuve d'Ascq Cedex, France
e-mail: {remy.chevrier; paola.pellegrini; joaquin.rodriguez}@ifsttar.fr

Abstract
In railway timetabling, the train running times are often known as the shortest durations between two stations increased of some percent of time. The running times are estimated by constructing the speed profiles that the trains must hold along the track. In this paper, an approach is proposed to compute several speed profiles for a same travel. The optimization approach involves two objectives optimized concurrently: the minimization of the journey duration and the minimization of the energy spent for the travel. Based on an evolutionary bi-objective algorithm, the optimization approach is applied to three instances and results are reported and analyzed.

Keywords
Speed profile, Energy saving, Running time, Railway timetabling, Multi-objective, Evolutionary algorithm, Optimization

1 Introduction

The current policies in transportation insist on the need of reducing the emission of pollutants but also on energy savings. Even if railway transportation is far more energy-efficient than other transportation modes, it is still concerned by the need of reducing the energy consumption. Several ways are possible for reducing this consumption, e.g. by improving train aerodynamics or rolling stock, or by improving the train control. In this research, we intend to develop a different approach which consists in constructing more energy-efficient timetables. Since timetabling is based on the fastest running times increased of a percentage of time used as time-margin [1], we aim to build alternative running times in such a way that the timetables produced are energy-compromises. In other words, they reduce the global energy consumption by using longer running times involving energy-friendly driving regimes, and also by reducing the trains' speeds and hence their tractive effort.

Indeed, to reduce the energy consumption we can use running times longer than the running times normally used. Using longer running times allows to introduce energy-friendly driving regimes during the trip. Introducing such driving regimes modifies the train's speed profile that indicates the speed that the train has to reach at each position.

In this paper, we address the problem of constructing multiple alternative train speed profiles minimizing both running time and energy consumption. 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.

As mentioned above, for the same trip the approach under consideration must be capable of providing a set of speed profiles for the decision-makers. In such a way, they will be able to choose a running time adapted to their needs in the timetabling process. Since two objectives are involved in this work, we propose a bi-objective optimization of speed tuning with energy saving. Given that evolutionary algorithms (EA) are well-suited to multi-objective optimization [2], our approach is based on a state-of-the-art multi-objective EA: the Indicator-Based Evolutionary Algorithm (IBEA) [3].

This paper extends and improves works presented in [4, 5]. In these previous works, the authors proposed to build speed profiles according to a set of rules assumed to keep a good diversity of solutions. However, the main drawbacks encountered in their approach were that a lot of solutions produced by the evolutionary algorithm were infeasible because of the rules themselves. In this paper, we propose an approach involving fewer decision variables and using mechanisms of speed profiling that limit the production of infeasible solutions during the optimization.

The paper is organized as follows. Section 2 concerns train dynamics and explains how the driving regimes work and are evaluated. Then, Section 3 presents the problem under study and its formulation. The solution assessment as well as the algorithms for building a speed profile are presented and detailed in Section 4. In Section 5, we present the specific components of the evolutionary algorithm. In Section 6, two case-studies (including one real railway line) are presented. Results of speed profile optimization obtained on these instances are reported and analyzed. Finally, Section 7 concludes the paper.

2 Driving regimes

In this section, we explain how the running times are estimated. Due to the lack of space, all the formulas of train dynamics used in this work are not presented but an explanation can be found in [6, 7]. The formulas used correspond to the point dynamics and the train length is not yet integrated. They may be modified or replaced without changing the nature of optimization problem defined in the following.

The symbols used for the train dynamics are summarized hereafter. Let T [s] be the journey duration and E [J] the mechanical energy. Let v [m/s] be the train's speed; $F_T(v)$ [N] the tractive effort produced by the train, function of speed v; $F_R(v)$ [N] the sum of resistances, function of speed v.

The mechanical energy needed to move the train can be calculated as the integral of the mechanical power over the running time T [8]. For convenience, let $F(t)$ and $v(t)$ be the tractive effort and the train speed at instant t, respectively: $E = \int_0^T F(t)\ v(t)\ dt$.

2.1 Setting sequences of driving regimes

The definition of accurate running times requires building 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 [9, 8] 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 i always have a different maximal speed $v_{m,i}$. Usually, a one-section journey can be divided in four steps 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 [10, 11]. The immediate consequence is the increase of journey duration but also the reduction of the use of mechanical energy. Indeed, 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 (see Points Co(1), C0(2)). 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.

The speed profile including no coasting allows the train to arrive at destination at the earliest possible time. However, it is in this case that the train will need the greatest quantity of energy.

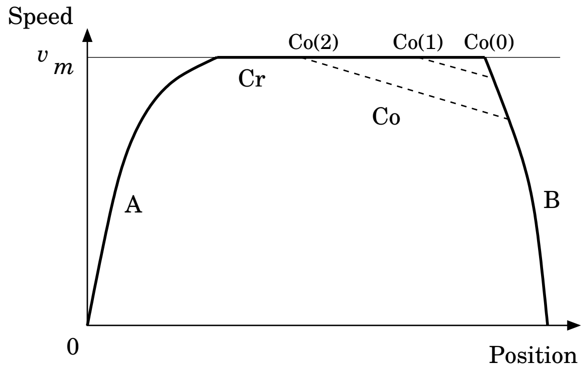


Figure 1: Usual speed profile over one section in four steps (assuming no slope): acceleration (A), cruising (Cr), coasting (Co) and Braking (B).

2.2 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 [12]: acceleration, cruising phase, coasting and braking. Energy consumption and running time evolve differently during each of them.

Table 1: 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

Acceleration
During this phase, the train accelerates at full power. Let F_m be the maximal tractive effort that the train can produce according to speed v. Hence, during an acceleration: $F_T(v) = F_m(v)$.

Cruising phase
This regime consists in maintaining the speed constant, i.e. the acceleration γ 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).

Coasting
The coasting corresponds to an inertia motion, while the engine is stopped. The tractive effort is therefore nil: $F_T(v) = 0$. As a consequence, the energy consumption during coasting is nil.

Braking
The braking is computed using the maximum service braking b_m. When braking, the tractive effort is therefore nil: $F_T(v) = 0$. Like in the coasting regime, the energy consumption during coasting is nil.

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 1.

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.

$$\Phi = (\min T, \min E). \tag{1}$$

Objective values (T_u, E_u) of solution u are assessed by `eval_solution`, which is a function defined by Algorithm 2 in Section 4.2.

3.2 Speed-based decomposition of a section

Using target-speeds as decision variables
The train path is decomposed into a set of n sections, in such a way that the speed profile is successively built in each section. This construction is based on the use of target-speeds, which allows 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 $v_{a,i}, v_{b,i}$ are the decision variables searched for by the algorithm and they are at the basis of the decomposition of the section. The speeds $v_{e,i}$ and $v_{x,i}$ are, respectively, the consequent entrance and exit speed of section i.

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 the determination of 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 the following constraints to the decision variables for each section: $v_{a,i} \geq v_{b,i}, \forall i = 1, ..., n$; $v_{m,i} \geq v_{a,i}, \forall i = 1, ..., n$; $v_{b,i} > 0, \forall i = 1, ..., n$. 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, at first, we provide algorithms for calculating train dynamics corresponding to a driving regime. These algorithms are essential to compute distance traveled, 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 the function `apply_regime`, which determines these driving regimes depending on 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` (not presented here for sake of brevity) 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 2. Within this function, T and E are calculated for each section consecutively by the function `eval_section`.

Algorithm 3 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 4 and 5, 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 4, 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: $l_{f,i} = 0, t_{f,i} = 0, e_{f,i} = 0$.

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 5 describes the construction 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

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 F_R as function of v_t, p_t
    if r == { *br* ***or*** *co* } **then**
      $F_T = 0$
      if r == br **then**
        – – Setting the braking b to the maximum service braking b_m
        $b = b_m$
      else
        Setting the braking $b = 0$
    else
      if r == *cr* **then**
        $F_T = \max(0, \min(F_m, F_R))$
        Calculating braking b if necessary
      else
        – – $r ==$ ac
        $F_T = F_m$
        Setting the braking $b = 0$
    Computing acceleration $a = \frac{F_T - F_R}{\rho\ m} + b$, $\rho = 1.07$
    $v_t = v_t + a\ \Delta t$
    $p_t = p_t + v_t\ \Delta t$
    $l = l + v_t\ \Delta t$
    $e = e + F_T v_t\ \Delta t$
    $R = R \cup (p_t, v_t)$
    $t = t + \Delta t$
end

Algorithm 2: 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 3: 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

begin
  (t_a, l_a, e_a, R_a) = first_part(v_e, v_a, l, p) ;
  $(t_{co}, t_{cr}, t_{br}, e_{co}, e_{cr}, e_{br}, l_{co}, l_{cr}, l_{br}, R_{co}, R_{cr}, R_{br})$ = second_part(v_a, v_b, p, l_a, v_m, v_n) ;
  $t = t_a + t_{co} + t_{br} + t_{cr}$;
  $e = e_a + e_{co} + e_{br} + e_{cr}$;
end

Algorithm 4: 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.

```
begin
    if v_e < v_a then
        (t_a, l_a, e_a, R_a)=apply_regime(v_e,v_a,l,p,ac) ;
    else
        (t_a, l_a, e_a, R_a)=apply_regime(v_e,v_a,l,p,br) ;
end
```

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 begin-point is known for the driving regime to use, while 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 $v_{a,i}$ 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 of 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 insert a coasting phase. Let q be the gradient of the track and σ the threshold above which the coasting results in an slowdown. 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 $v_l = v_{b,i}$. A number of cases must be distinguished 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 5: 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.

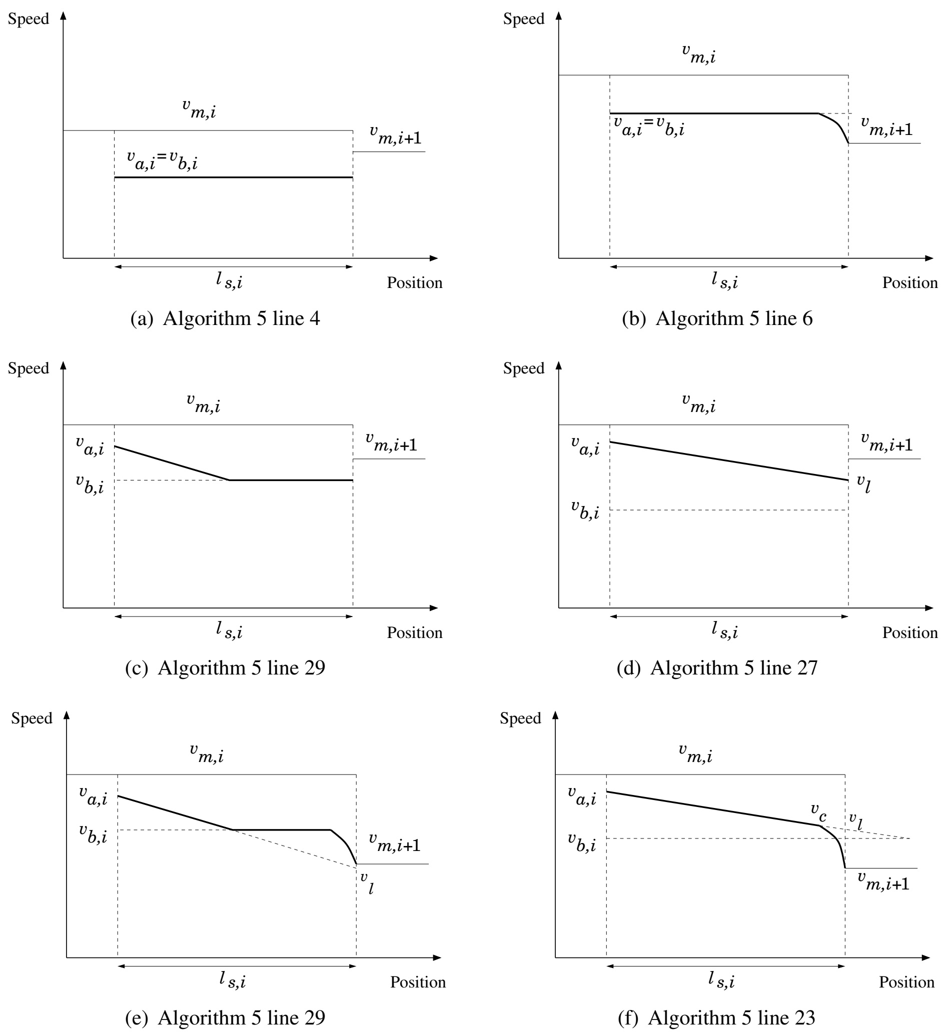
```
begin
    if v_a == v_b then
        if v_a < v_n then
            (t_cr, l_cr, e_cr, R_cr)=apply_regime(v_a,v_b,l - l_a,p + l_a, cr) ;
        else
            (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_b,v_n,l - l_a,p + l,br) ;
            (t_cr, l_cr, e_cr, R_cr)=apply_regime(v_a,v_b,l - l_a - l_br,p + l_a, cr);
    else
        // v_a > v_b
        v_l = exit_coast(v_a, v_b, l - l_a,p + l_a);
        (t_co, l_co, e_co, R_co)=apply_regime(v_a,v_b,l - l_a,p + l_a, co);
        if v_l > v_a then
            if v_n <= v_b then
                (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_m,v_n, l - l_a,p + l, br) ;
                (t_co, t_br, l_br, l_co, e_co, e_br, R_br, R_co)= search_for_intersection(R_co,R_br) ;
                v_b = v_n ;
            else
                (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_m,v_b, l - l_a,p + l, br) ;
                (t_co, t_br, l_co, l_br, e_co, e_br, R_co, R_br)= search_for_intersection(R_co,R_br) ;
        else
            if v_l > v_b then
                if v_n <= v_l then
                    (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_m,v_n, l - l_a,p + l, br) ;
                    (R_co, R_br, v_c)= search_for_intersection(R_co,R_br) ;
                    v_b = v_c
                else
                    v_b = v_l ;
            else
                v_t = min(v_b, v_n) ;
                (t_br, l_br, e_br, R_br)=apply_reverse_regime(v_b,v_t, l - l_a,p + l, br) ;
                (t_cr, l_cr, e_cr, R_cr)=apply_regime(v_b,v_b,l - l_a - l_co - l_br,p + l, cr);
    t = t_a + t_co + t_br + t_cr;
    e = e_a + e_co + e_br + e_cr;
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 $v_{m,i+1}$: $v_{b,i} = v_{m,i+1}$ (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 guaranteeing that it is always



(a) Algorithm 5 line 4

(b) Algorithm 5 line 6

(c) Algorithm 5 line 29

(d) Algorithm 5 line 27

(e) Algorithm 5 line 29

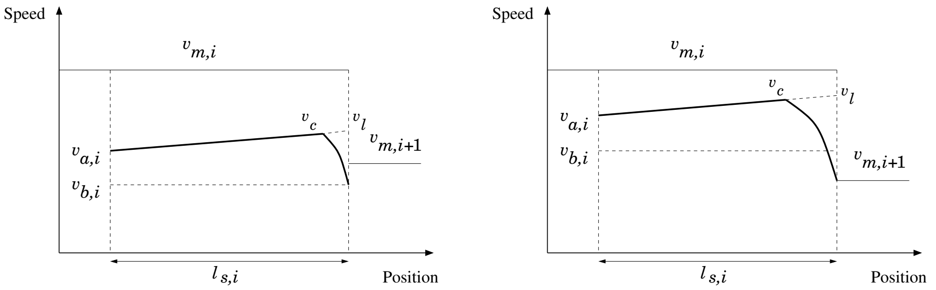
(f) Algorithm 5 line 23

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

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 may 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 the 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).

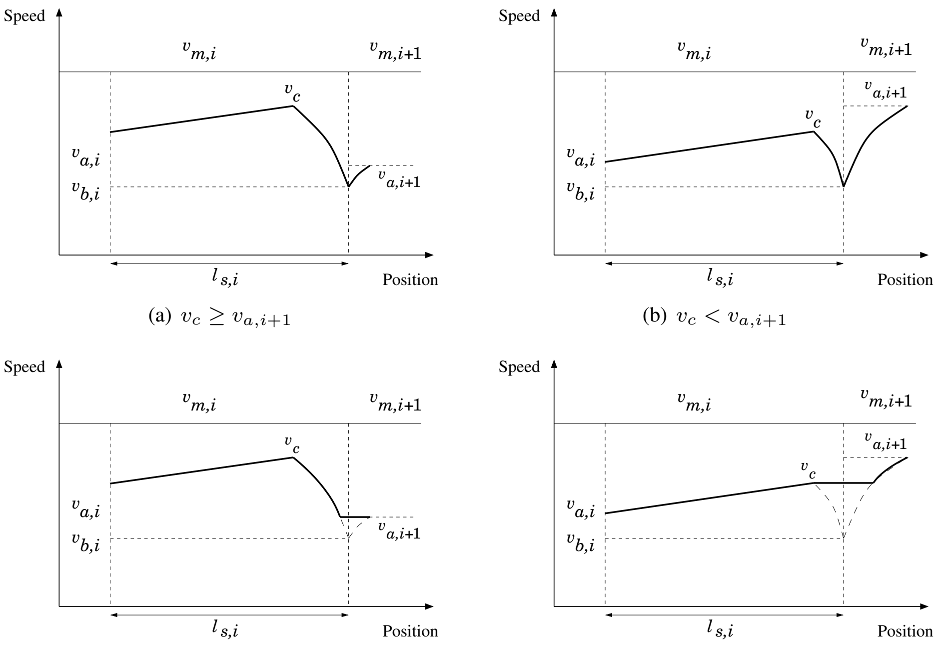
All along this inserted cruising phase, it is necessary to compute the effort necessary to



(a) Algorithm 5 line 18 (b) Algorithm 5 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.

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.



(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.

5 Evolutionary Multi-objective Optimization

The problem under study involves two concurrent objectives. Contrary to methods dealing with an objective function which is a sum of weighted values, we prefer instead to consider a multi-objective framework, which concurrently optimizes each objective. Such an optimization implies a set of distinct possible solutions. Given that the evolutionary algorithms are well suited to this kind of problem, we choose a state-of-the-art evolutionary algorithm: the Indicator Based Evolutionary Algorithm (IBEA) [3], which uses indicators to assess solutions and their contribution in terms of quality, contrary to other evolutionary algorithms which directly deal with dominance notion to rank the solutions.

Whatever the evolutionary algorithm chosen, it is a scheme of optimization using components to be implemented: solution pattern, initialization, evaluation and variation operators described below.

5.1 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 below.

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 allowed on the track.

A complete description of this calculation is given in [7]. 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 $x > 1$.

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.

5.2 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 cannot be built because no rule of construction complies to the speeds under consideration, 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.

5.3 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 [13], which generates two new solutions from two belonging to the current population. 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 [2, 13]. 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 [14] and its implementation of IBEA. 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 the ParadisEO framework.

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 5(a), which reports the length and the maximal speed of each section. The track slopes are depicted in Figure 5(b, c) and given as gradient [‰]. Instance 1 corresponds to a line described and used in [12, 15]. It is 2.2 km long and includes five sections. The second instance is the *Saint-Étienne–Rive de Giers* line in France and the roundtrip is studied as two separate instances denoted I2a for the outward journey and I2b for the inward. It is 20.2 km long and includes five sections. It is interesting to note that the gradient is mostly negative for the outward journey and thus it is in descent for the most part of the line. In the inward journey, the situation is inverted and the train climbs on the most part of the line.

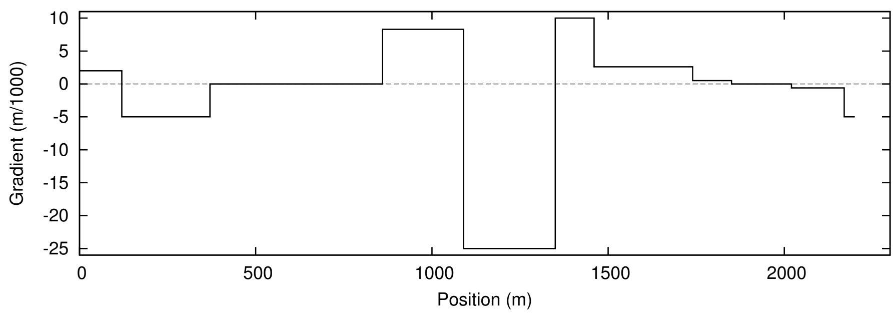
Instance 1

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

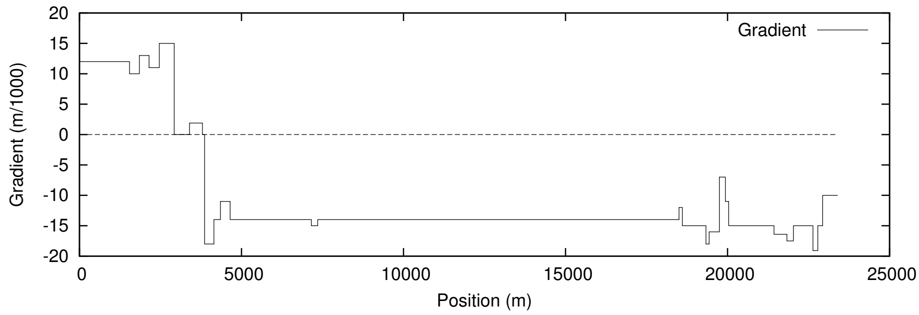
Instance 2

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



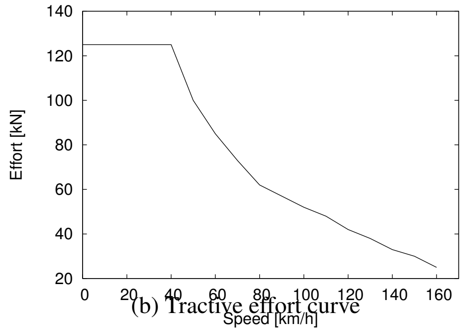
(c) Description of the instance 2a gradient

Figure 5: Description of the studied instances

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. 6.

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 6: Technical parameters of train AGC

6.3 Results

Case study 1

Figure 7(a) depicts the objective space of Instance I1 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 improve 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(a), 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 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 2.

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 1,100 m and 1,300 m due to the steep descent of the track (gradient $q < \sigma$), 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 5 line 18 (Fig. 3(a)). In the second section (from 1,350 m to 1,510 m), a braking covers the whole available

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

Table 2: Numerical results of three solutions obtained on instance I1: 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

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. Speeds $v_{a,3} = 11$ m/s and $v_{b,3} = 9.76$ m/s imply the situation described in Algorithm 5 line 27 (Fig. 2(e)). The cruising distance equals 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 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.

Case study 2

The main difference compared to the case study 1 is the presence of steep descents (resp. climbs) in the track profile for the outward (resp. inward) journey.

Figures 7(b,c) represent the sets of Instance I2 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 solutions I2a* and I2b* are also represented to have basis of comparison in both cases. The journey duration is limited to $1.5 \times T_{I2a*}$ (resp. $1.5 \times T_{I2b*}$). 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 solutions and the others can be explained by the particular topology of the track. For the outward journey, 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. For the inward journey, the gap can be explained by the use of slower speeds which need weaker tractive effort to be maintained constant. In order to highlight this explanation, we can focus on the speed profile of solution I2a_1 in Figure 8(b). 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 (gradient $q < \sigma$). 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 I2a* the train has to accelerate at full power to reach the maximum speed. It is the reason for which solution I2a* 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 I2a_1s, is depicted in Figure 8(b).

The results reported in Table 3(a) 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-support is shown in so far as it could help train-practitioners to build timetables.

Figure 8(c) represents the speed profiles of the reference and one compromise solutions for the inward journey. Since the gradient is mostly positive ($q > 0$), maintaining the speed constant is very energy-consuming as we can see in Table 3(b). The energy consumption reported confirm that the inward journey needs far bigger effort to cover the line. The reference solution I2b* needs slightly fewer seconds than the reference solution I2a* whereas the times should be either equal or the solution I2b* should be longer than solution I2a*. We can assume that it is due to the approximations during the estimation. The energy saved in solution I2b_1 compared to solution I2b* reaches more than 20% with an increase of less than 5% of time. The saving is essentially realized by the introduction of a coasting in the fourth section followed by a cruising regime.

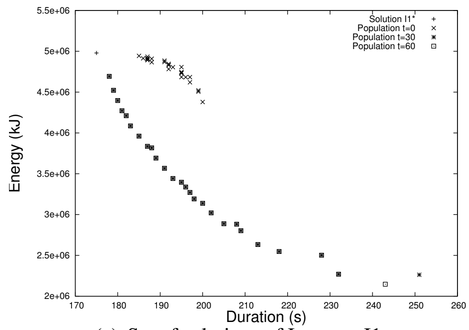
Table 3: Numerical results of two solutions obtained on instances I2a and I2b.

(a) Instance I2a

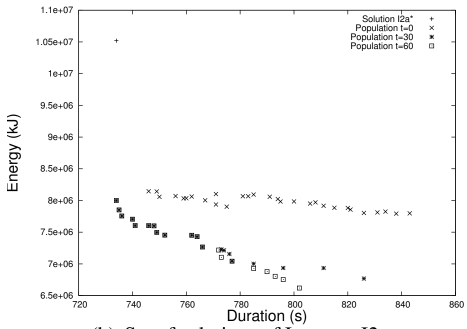
Solution	Duration [s]	Dev. [%]	Energy [kJ]	Dev. [%]
I2a*	734		10.5194×10^6	
I2a_1	773	+5.31	7.10466×10^6	-32.47
I2a_1s	761	+3.67	6.80466×10^6	-33.31

(b) Instance I2b

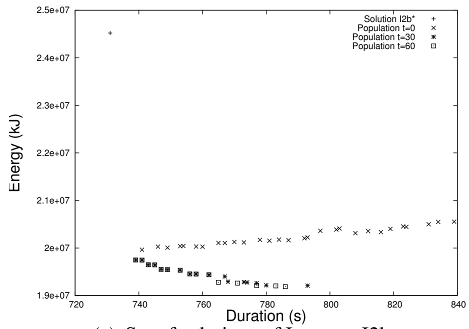
Solution	Duration [s]	Dev. [%]	Energy [kJ]	Dev. [%]
I2b*	731		24.5206×10^6	
I2b_1	765	+4.65	19.2933×10^6	-21.31



(a) Set of solutions of Instance I1



(b) Set of solutions of Instance I2a



(c) Set of solutions of Instance I2b

Figure 7: Fronts of solutions at times t={0, 30, 60} on instances 1 (a), 2a (b) and 2b (c)

7 Conclusion

The estimation of the train running times is directly related to the construction of the speed profiles, indicating the speed to hold at each position. In this paper, we have proposed an approach to build speed profiles based on a set of rules and by using the driving regimes defined according to the Maximum Principle. 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 suitable 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. For illustrating the relevance of the approach, three case studies have been tackled with the multi-objective algorithm, which has produced a set of solutions in a short period of time (from 30 to 60 seconds).

In the future, an improvement of the dynamics model will be proposed to have a more accurate estimation of the durations. The next step will concern the production of alternative timetables optimizing notably the energy consumption.

References

[1] 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.

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

[3] 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.

[4] 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.

[5] R. Chevrier, G. Marlière, B. Vulturescu, J. Rodriguez, Multi-objective evolutionary algorithm for speed tuning optimization with energy saving in railway: Application and case study, in: RailRome 2011, the 4th International Seminar on Railway Operations Research, Rome, Italy, 2011.

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

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

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

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

[10] 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.

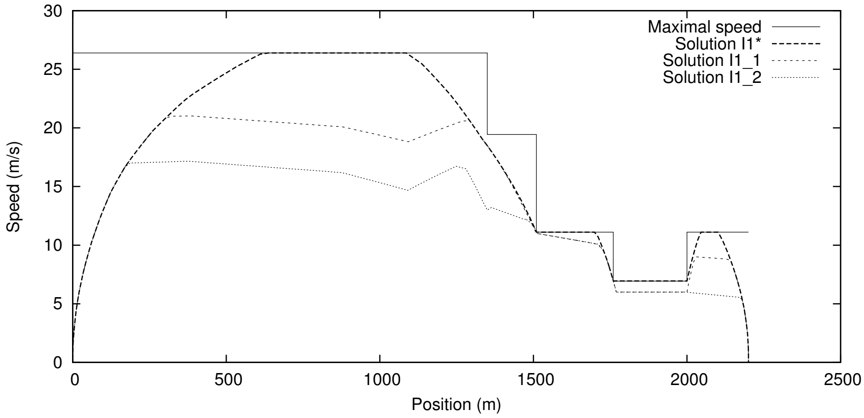
[11] S. Iwnicki (Ed.), Handbook of Railway Vehicle Dynamics, Taylor and Francis Group, 2006, 535 p.

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

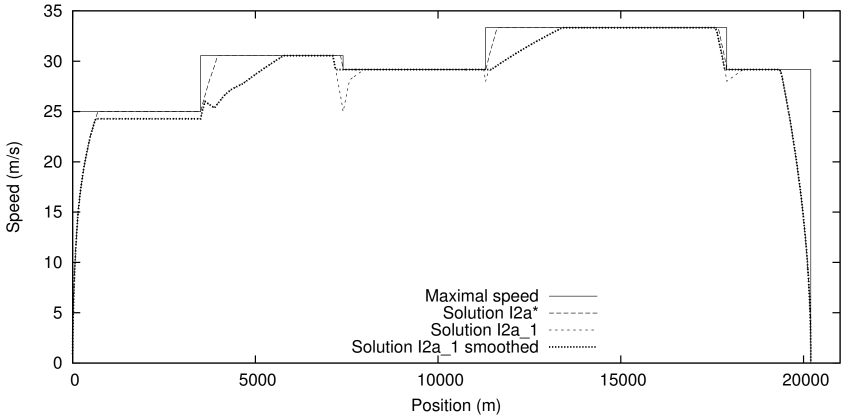
[13] K. Deb, R. B. Agrawal, Simulated binary crossover for continuous search space, Complex Systems 9 (1995) 115–148.

[14] 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.

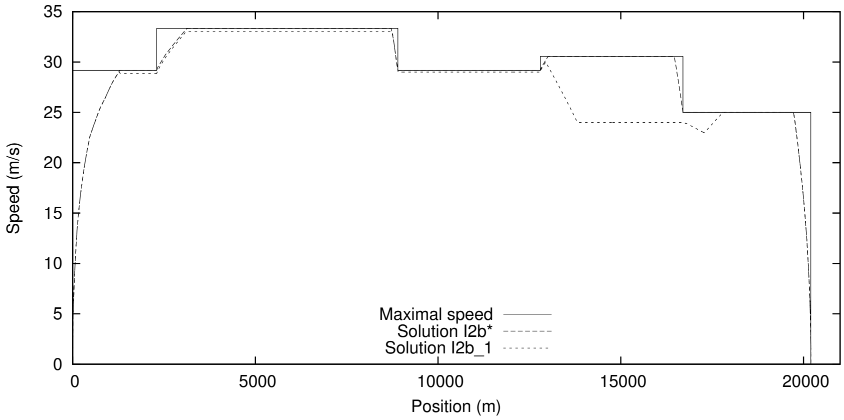
[15] H. Ko, T. Koseki, M. Miyatake, Application of dynamic programming to optimization of running profile of a train, in: Computers in Railways IX, 2004, pp. 103–112.



(a) Instance I1: Reference and two compomise solutions



(b) Instance I2a: Reference and one compromise (original and smoothed) solutions



(c) Instance I2b: Reference and one compromise solutions

Figure 8: Examples of speed profiles on instances I1 (a), I2a (b) and I2b (c)