

An Evolutionary Multi-objective Approach for Speed Tuning Optimization with Energy Saving in Railway Management

Rémy Chevrier

Université Lille Nord de France – INRETS ESTAS
 20 rue Élisée Reclus – 59650 Villeneuve d'Ascq, France
 Email: remy.chevrier@inrets.fr

Abstract—An approach for speed tuning in railway management is presented for optimizing both travel duration and energy saving. This approach is based on a state-of-the-art evolutionary algorithm with Pareto approach. This algorithm provides a set of diversified non-dominated solutions to the decision-maker. A case study on Gonesse junction (France) is also reported and analyzed.

Index Terms—Multi-objective, Evolutionary algorithm, Energy saving, Railway management, Optimization

I. INTRODUCTION

For recent years, the concern due to pollution and global warming led to develop more eco-aware transportation systems. Energy spent in the railway management is used to move the trains and if speed tuning is well suited the energy consumption will be lower than in other cases. Indeed, since the acceleration phases consume a huge quantity of energy, it is necessary to tune the speeds according to the distances and the allowed durations in order to avoid to brake a lot and accelerate just after. Moreover, given that the trains have a big inertia, it is judicious to use this physical property by stopping the engine and letting the train advance just thanks to the initial force [1], [2].

The railway management involves multiple objectives which are often antagonist such as the travel duration and the energy consumption. Although the travel duration had often preference of the decision makers, for recent years the criterion of energy consumption is considered from an equivalent point of view. Indeed, global energy saving is becoming the new challenge of the transportation systems including railway.

Works have been led for analytically computing ST solutions according to several levels of delay tolerance [3], [4], [5]. But to our knowledge, there is few multi-objective approach yet. These are based on Differential Evolution [6] for mass transit system [7] or evolutionary algorithms hybridized or not [8]. Nevertheless, to our knowledge few approaches propose to optimize the energy consumption which becomes the new challenge of the decade. Thus, in this paper we deal with a bi-objective optimization of the speed tuning (ST) with energy saving. These concurrently optimized criteria are on the one hand the minimization of the travel duration and on the other hand the minimization of the energy consumption. The main

goal consists in designing ST solutions diversified enough to help decision makers to choose the solution the most adapted to their needs. In order to solve the problem, we propose a new approach based on a Pareto evolutionary algorithm.

After defining the problem we propose to solve in Section II, we present our model of speed tuning in Section III. The evolutionary computation principles are presented in Section IV by exposing also the Indicator-Based Evolutionary Algorithm we use to compute ST solutions. Experimental results based on Gonesse junction (France) are then provided and discussed in Section V. Finally Section VI concludes the paper.

II. PROBLEM OVERVIEW

The main goal consists in designing the most suited speed profile over space. The space corresponds to a sequence of intervals I (block sections) in which the speeds can be changed. Figure 1 represents the decomposition of a one-section journey in four steps. A maximum speed v_{max} limits the train speed. According to this limit and the train parameters the speed can be defined in each step. The first step (A) corresponds to the train acceleration when the speed grows from 0 to v_{max} (if the train can reach v_{max}). Before dealing with the cruising and coasting phases it is necessary to compute the braking phase (B) to be sure that the needed braking distance will not exceed the remaining distance before the end. The cruising phase (Cr) corresponds to the speed maintaining, that is, a null acceleration when the traction effort equals the resistance to the train advance. The coasting phase (Co), depicted by the dashed lines on Figure 1, is engaged when the engine is stopped and the train moves thanks to its inertia. During this phase, no energy is consumed and hence in order to reduce the energy consumption it is interesting to vary the instant (or position) from which the engine is stopped and the coasting phase is started (see points 1, 2, 3 in Fig. 1). The sooner the coasting phase starts the greater the economy but the later the train will arrive. Thus the goal of the problem solving is to determine a good tradeoff between energy consumption and delay occurred.

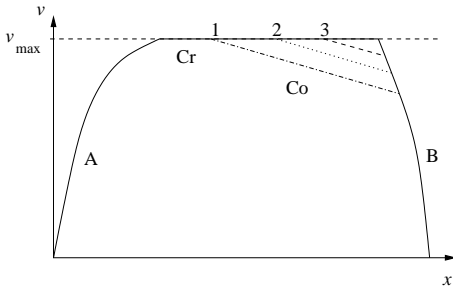


Fig. 1. Speed tuning over space in four steps: acceleration (A), cruising (Cr), coasting (Co) and Braking (B)

III. SPEED TUNING MODEL

Naturally, the four-steps model explained above cannot be applied everywhere and the look of speed profile depends on the entrance speed v_0 (position 0) and the exit speed v_X (position X). Between these two positions 0 and E it is necessary to determine the speeds according to a chosen policy. In this way, we introduce two intermediate speeds v_1 and v_2 which help us to build the speed profile.

A train path is composed of n sections. Therefore, for a section S , we have a set of four speeds: $v_0^S, v_1^S, v_2^S, v_X^S$. When the train starts its journey, speed v_0^1 is null for section 1 ($v_0^1 = 0$), while in the arrival section speed $v_X^n = 0$. When the train leaves a section S and enters in the following ($S + 1$), the exit speed of section S equals the entrance speed of section $S + 1$: $v_X^S = v_0^{S+1}$ in such a way that: $v_X^S \leq \min(v_{max}^S, v_{max}^{S+1})$. Speeds v_1, v_2 to be determined are limited by the maximum speed of the section S : $v_1^S \leq v_{max}^S$ and $v_2^S \leq v_{max}^S$.

Taking these elements into account, we generalize that three speeds have to be determined per section: v_1, v_2 and v_X . These values will be searched by the evolutionary algorithm we propose in Section IV. Speed profile is determined according to a three steps model:

- 1) the entrance phase tunes speed for accelerating/decelerating from v_0 to v_1 ;
- 2) the exit phase is assessed before the intermediate phase for varying speed from v_2 to v_X ;
- 3) the intermediate phase tunes the speed from v_1 to v_2 according to our policy depending on $v_1 > v_2$ or not.

Now, we introduce a set of basic definitions useful for the remainder of the paper:

- T a travel duration which corresponds to the set of intermediate durations (unit [s]);
- E the energy consumed by the train to move over time (unit [J]);
- $P(t)$ the power delivered at instant t (unit [W]);
- $F_T(t)$ the traction effort at instant t (unit [N]);
- $F_R(t)$ the resistance to the advance at instant t (unit [N]);
- $v(t)$ the train speed at instant t (unit [m/s]);
- v_0 the entrance speed of a train (unit [m/s]);
- v_X the exit speed of a train (unit [m/s]);
- $a_b(t)$ the braking at instant t (unit [m/s²]);
- $a(t)$ the acceleration at instant t (unit [m/s²]);

- m the train mass (unit [kg]).

A. Objectives

The problem can be represented as a set Φ of two objective functions to be minimized. The first function φ_1 represents the minimization of the travel duration whereas the energy consumption reduction is illustrated by function φ_2 . The amount of durations corresponds to the sum of all durations needed to travel within the sections.

$$\Phi = (\varphi_1, \varphi_2) \quad (1)$$

$$\varphi_1 = \min T \quad (2)$$

$$\varphi_2 = \min E \quad (3)$$

$$E = \int P(t) dt \quad (4)$$

$$P(t) = F(t) v(t) \quad (5)$$

B. Elements of railway dynamics

The fundamental equation of dynamics states that the relation between the forces, mass and acceleration:

$$F_T(t) - F_R(t) = \rho m a(t)$$

Note that ρ is a mass correction factor usually set to $\rho = 1.04$ [2], [5]. The train data also depict the traction effort profile which indicates effort $F_T(t)$ according to a speed $v(t)$. Resistance $F_R(t)$ is defined according the speed $v(t)$: $F_R(v) = A + Bv + Cv^2$ where A, B, C are defined constants specific to the train. The braking at instant t is calculated as follows: $a_b = D + E \times F^{v(t)}$ where D, E, F are also defined constants related to the train.

Now, with these elements we can determine the acceleration, cruising, coasting and braking phases. We note that only acceleration and cruising phases need energy.

1) *Acceleration*: This can be defined as follows:

$$a(t) > 0 \quad \Leftrightarrow \quad F_T(t) > F_R(t) \quad (6)$$

$$a(t) = \frac{F_T(t) - F_R(t)}{\rho m} \quad (7)$$

$F_T(t)$ and $F_R(t)$ are calculated according to speed $v(t)$ as explained before. The acceleration phase from speed v_A to v_B ($v_A < v_B$) is iterated each second (instant i) and updates the acceleration, the speed and the position.

Algorithm 1: Calculation of an acceleration phase

```

v(i) = v_A;
while v(i) < v_B do
    Update position x(i+1) = 0.5a(i) + v(i);
    Update acceleration a(i+1) according to v(i);
    Update speed: v(i+1) = v(i) + a(i);
    Update energy: E = E + E(i) with E(i) = F(i)v(i);
    Update duration: T = T + 1
end

```

2) *Cruising*: The cruising phase maintains the train speed v along a distance d without accelerating:

$$a(t) = 0 \Leftrightarrow F_T(t) = F_R(t)$$

The cruising phase can be computed as follows:

Algorithm 2: Calculation of a cruising phase

Calculate $F_R(v)$, set $F_T(v) = F_R(v)$;
 Calculate duration: $T = \frac{d}{v}$;
 Calculate energy: $E = T \times F_T(v) \times v$

3) *Coasting*: During a coasting phase the engine is stopped: $F_T(t) = 0$ and the speed decreases because $a(t) < 0$. The calculation of this phase is very close to an acceleration except for the energy consumption which stays null. The coasting between speed v_A and v_B ($v_A > v_B$) is iterated each second.

Algorithm 3: Calculation of a coasting phase

$v(i) = v_A$;
while $v(i) > v_B$ **do**
 Update position $x(i+1) = 0.5a(i) + v(i)$;
 Update acceleration $a(i+1) = \frac{-F_R(i)}{\rho \times m}$;
 Update speed: $v(i+1) = v(i) + a(i)$;
 Update duration: $T = T + 1$
end

4) *Braking*: The braking phase combines two resistance forces: the resistance to the train advance and the service braking force. So the calculation is identical as in a coasting phase except for determining the acceleration: $a(i+1) = \frac{-F_R(i)}{\rho \times m} - a_b(i)$ with $a_b(i) = D + E \times F^{v(i)}$.

C. Entrance and exit phases

Before evaluating the intermediate phase it is necessary to determine on the one hand the entrance phase and on the other hand the exit phase. For each one, duration, distance and energy are calculated according to v_0 compared with v_1 and v_2 compared with v_X . Figure 2 depicts the possible cases described below.

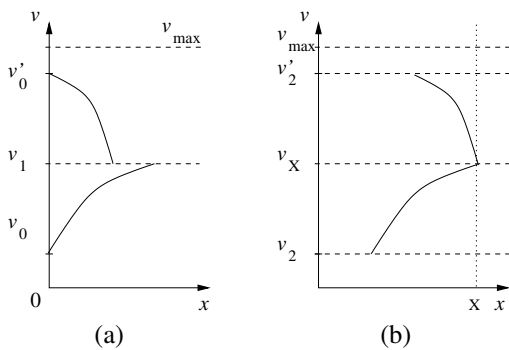


Fig. 2. Scheme of (a) entrance and (b) exit phases

1) *Entrance*: The entrance phase is determined according to v_0 and v_1 . Two cases may arise (Fig. 2(a)):

- 1) if $v_0 < v_1$ then an acceleration occurs, increasing speed from v_0 to v_1 ;
- 2) if $v_0 > v_1$ then a braking is done, decreasing speed from v_0 to v_1 (illustrated by v'_0 and v_1).

In all these cases, a distance d_0 is needed to vary the speed during T_0 . The consumed energy E_0 is null if it is a braking phase, otherwise $E_0 > 0$.

2) *Exit*: The exit phase is done in the same way by using v_2 and v_X and two cases may also arise (Fig. 2(b)):

- 1) if $v_2 < v_X$ then an acceleration occurs, increasing speed from v_2 to v_X ;
- 2) if $v_2 > v_X$ then a braking is done, decreasing speed from v_2 to v_X (illustrated by v'_2 and v_X).

A distance d_X and a duration T_X are needed for this phase. The consumed energy E_X is positive ($E_X > 0$) if the phase is an acceleration.

D. Intermediate phase

Once the exit phase is computed, the feasibility of the solution must be checked. Indeed the travelled section has an available distance d_S and we must be sure that $d_0 + d_X < d_S$ in so far as an available distance remains to allow varying the speed from v_1 to v_2 during the intermediate phase.

Let $d_I = d_S - d_0 - d_X$ be the available distance to vary the speed from v_1 to v_2 . Two cases may arise:

- 1) if $v_1 > v_2$ then we try to insert a coasting phase to decrease the speed and to save energy. If it is possible, we insert a cruising phase before the coasting for completing all the distance available (Fig. 3(a), the plain line). The only consumed energy ($E_I > 0$) is due to the cruising phase. When the distance is not enough to do a complete coasting then a braking phase from v_1 to v_2 must be calculated and we search for intersection of coasting and braking phases (Fig. 3(a), the dashed line) and in this case $E_I = 0$;
- 2) if $v_1 < v_2$ then an acceleration from v_1 to v_2 will be necessary and we insert it halfway through (travelled distance d_A). Cruising phases are added before and after the acceleration, the first at speed v_1 and the second at speed v_2 (Fig. 3(b)). The cruising phases are done on the same distance $((d_I - d_A)/2)$. The consumed energy corresponds to the amount of energy required for each phase.

Naturally, it is necessary to check whether each phase can be inserted according to the remaining distance or not. If the distances do not allow to insert the chosen phases, the solution is marked as not feasible and penalized during the evaluation in the evolutionary algorithm.

IV. EVOLUTIONARY MULTI-OBJECTIVE OPTIMIZATION

An evolutionary algorithm (EA) is an iterative process of exploratory search. Our choice is led by a will to obtain a set of sufficiently diversified solutions in a single run. Indeed, the

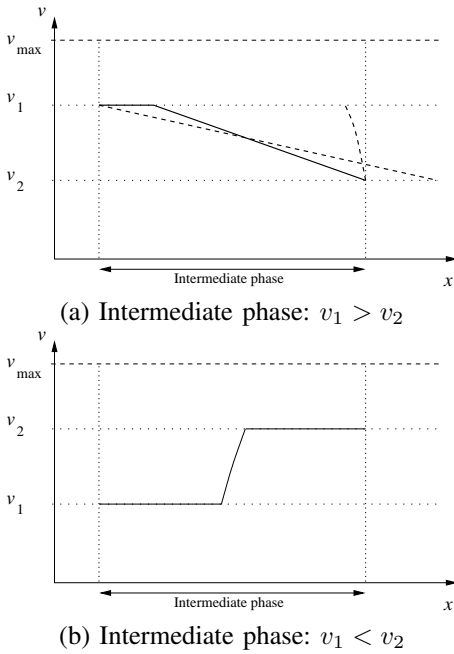


Fig. 3. Two cases of intermediate phase

evolutionary algorithms with Pareto approach are capable to produce well-spread incomparable solutions along the Pareto front. That could be an advantage to help the decision makers in the case of real-life problems [9].

An EA is a nature-inspired metaheuristic gathering a set of solutions (*individuals* or *chromosomes*): a *population*. The latter evolves while recombining pairwise individuals in such a way that new original and improved solutions are produced. A mutation operator allows to diversify the population while randomly modifying solutions. These new solutions are added to a temporary population which will partially or totally constitute the population at the next iteration, depending on the algorithm policy of the population renewal. Algorithm 4 presents the main steps of a general purpose EA.

Algorithm 4: Canonical evolutionary algorithm

Population initialization;
while Stopping criterion not reached **do**
 Evaluate each solution in population \mathcal{P}_i ;
 Select individuals in population \mathcal{P}_i for crossover;
 Cross individuals according to a crossover rate;
 Mutate individuals according to a mutation rate;
 Population \mathcal{P}_{i+1} generation (selection for replacement);
end

A. Multi-objective Optimization

A general Multi-objective Optimization Problem (MOP) can be defined by a set of n objective functions (f_1, f_2, \dots, f_n) , a set X of feasible solutions in the decision space, and a set Z of feasible points in the objective space. Without loss of

generality, we here assume that each objective function is to be minimized. To each solution $x \in X$ is assigned an objective vector $z \in Z$ on the basis of the vector function $f : X \rightarrow Z$ with $z = f(x) = (f_1(x), f_2(x), \dots, f_n(x))$ as illustrated by Figure 4. An objective vector $z \in Z$ is said to *dominate*¹ another objective vector $z' \in Z$ iff $\forall i \in \{1, 2, \dots, n\}, z_i \leq z'_i$ and $\exists j \in \{1, 2, \dots, n\}$ such as $z_j < z'_j$. An objective vector $z \in Z$ is said to be *non-dominated* iff there does not exist another objective vector $z' \in Z$ such that z' dominates z . A solution $x \in X$ is said to be *efficient* if its mapping in the objective space results in a non-dominated point. The set of all efficient solutions is the *efficient set*, denoted by X_E . The set of all non-dominated vectors is the *Pareto front*, denoted by Z_N . A possible approach in MOP solving is to find the minimal set of efficient solutions, i.e. one solution $x \in X_E$ for each non-dominated vector $z \in Z_N$ such as $f(x) = z$. However, generating the entire efficient set is usually infeasible due to the complexity of the underlying problem. Therefore, the overall goal is often to identify a good approximation of it. EAs are commonly used to this end as they are able to find multiple and well-spread non-dominated solutions in a single simulation run [10].

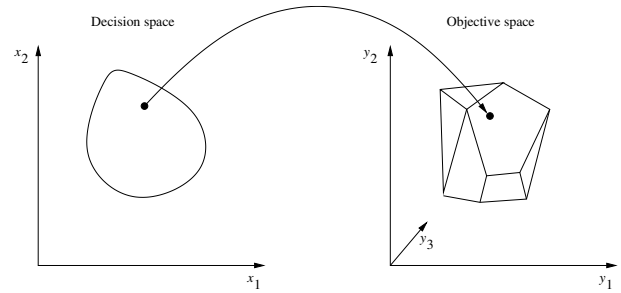


Fig. 4. Representation of a solution (x_1, x_2) in the decision space and the corresponding values in the objective space: $(y_1, y_2, y_3) = f(x_1, x_2)$.

B. Indicator Based Evolutionary Multi-objective Algorithm

Although there exists several state-of-the-art multi-objective EAs (NSGA-II [11], SPEA2 [12]), we use Indicator Based Evolutionary Algorithm [13]. Indeed IBEA is a method more modern than the most widely used EA: NSGA-II. This is a good illustration of the new trend dealing with indicator-based search, and started to become popular for recent years. The main idea behind IBEA is to introduce a total order between solutions by means of a binary quality indicator. Its fitness assignment scheme is based on a pairwise comparison of solutions from the current population with regards to an arbitrary indicator I . To each individual x is assigned a fitness value $F(x)$ measuring the 'loss in quality' if x was removed from the current population P , i.e. $F(x) = \sum_{x' \in P \setminus \{x\}} (-e^{-I(x', x)/\kappa})$, where $\kappa > 0$ is a user-defined scaling factor. Different indicators can be used for such a purpose, and we here choose to use the binary additive ϵ -indicator ($I_{\epsilon+}$) as defined

¹We will also say that a decision vector $x \in X$ *dominates* a decision vector $x' \in X$ if $f(x)$ dominates $f(x')$.

in [13]. $I_{\epsilon+}(x, x')$ gives the minimum value by which a solution $x \in X$ has to or can be translated in the objective space to weakly dominate another solution $x' \in X$. Selection for reproduction consists of a binary tournament between randomly chosen individuals. Selection for replacement consists of iteratively removing the worst solution from the current population until the required population size is reached; fitness information of the remaining individuals is updated each time there is a deletion.

C. Solution encoding and initialization

A solution is defined by a vector of real values. Each value corresponds to one speed. Given that three speeds are necessary to represent a section, we can state the vector length l equals three times the number of sections (n): $l = 3n$.

Section 0			Section n			
v_0^0	v_1^0	v_2^0	...	v_0^n	v_1^n	v_2^n

Fig. 5. Pattern of a solution: a vector of real values for the speeds

Even if the speeds are bounded by maximum entrance, exit and section speeds, three ranges of speeds are defined: slow, middle and high speeds in such a way that one third of the solutions are either slow, middle or high speed tuned. For example, let S_1, S_2 be two sections with respective maximum speeds: 90 km/h and 120 km/h. Three ranges of values are possible: $([0, 30], [0, 40])$, $([30, 60], [40, 80])$ and $([60, 90], [80, 120])$. Such a mechanism is useful to bring a good diversity in the initial population by designing more or less fast solution, i.e. more or less energy expensive. Then, initialization is done by randomly assigning speeds.

D. Operators

1) *Evaluation*: As we mentioned before, our MOP is composed of two objective functions ($\Phi = (\varphi_1, \varphi_2)$). Objective function φ_1 corresponds to the minimization of the amount of the durations T_i needed for each section i : $\varphi_1 = \min T$ with $T = \sum_{i=1}^n T_i$. Objective function φ_2 is in charge of the reduction of the energy consumption: $\varphi_2 = \min E$ with $E = \sum_{i=1}^n E_i$. This global consumption equals the amount of energy required for each section i . Each section is evaluated according to the model presented before in such a way that we know pair (T_i, E_i) of each section i .

2) *Recombination*: The recombination step is done by means of two operators: crossover and mutation. Since we deal with a continuous problem, we use operators specifically designed for this kind of problem: the Simulated Binary crossover (SBX) [14] and similarly, the mutation is based on a polynomial mutation adapted to search over continuous space.

V. EXPERIMENTAL RESULTS

In order to develop our approach, we use framework ParadisEO in which a lot of metaheuristics are implemented [15]. This tool is a white box in which the different steps of the algorithms have to be defined. In the case of IBEA the user has to implement crossover, mutation and evaluation steps and also the problem-related components.

A. Parameter Setting

The population of 50 individuals evolves over 1,000 generations. Crossover (x_r) and mutation (m_r) rates are respectively set to $x_r = 0.9$ and $m_r = 0.5$. In addition, scaling factor κ exists for IBEA and κ is set to 0.05 according to [13].

B. Case study: Gonesse junction

Here is proposed a real-life case study: the Gonesse junction (France) crossed by a passenger train ($m = 180,000$). Eight sections are used for our example whose results are depicted in Figure 6. The plain line indicates the maximum speeds of the sections. The path has 14,285m of length and eight sections are crossed. Each section is limited by a maximum speed. This case study is interesting because of two sections with switch points (sections 2 and 5) which are very short (resp. 150m and 90m) compared with the other sections. Furthermore the switch points are slow speed sections (60 km/h) whereas the other speeds can be very higher (until 200 km/h). So, these sections bring about braking phases to be managed at best.

C. Performance assessment and discussion

Here we propose to discuss three solutions obtained by our approach. Let S^* be a solution to the mono-objective problem consisting in just minimizing the travel duration. Let S_1 and S_2 be two solutions to the bi-objective problem and compared with S^* . The results are provided in Table I and the deviation compared with S^* is also reported.

Sol.	E [kJ]	Deviation	T [s]	Deviation
S^*	507,618		484.13	
S_1	378,683	-24.5%	508.79	+4.9%
S_2	246,245	-51.5%	555.63	+14.7%

TABLE I
ENERGY CONSUMPTION AND DURATION OF SOLUTIONS S^*, S_1, S_2

Figure 6 illustrates the speed profile of each solution. It is interesting to note that solutions S_1, S_2 have big coasting phases, that is why these are less energy expensive than S^* . Besides, we can see that S^* has very long acceleration phases which are very energy expensive. The differences of consumption can also be observed in Figures 7(a,b,c) which depict the produced effort and highlight that Solution S^* consumes more energy than the others. If we focus on the energy saving compared with the delay, we can note that with around 5% of time in more, we can save around 25% of energy. Moreover, if we extend the delay to around 15% of time in more, we can save till around more than half of energy. That proves the interest to take some seconds in more to travel for saving a lot of energy.

Furthermore, the algorithm as well as the underlying method have proved their capability to provide a set of diversified and incomparable solutions.

VI. CONCLUSION AND PERSPECTIVES

In this paper we dealt with a problem of speed tuning in railway management. The solving goal is to optimize both

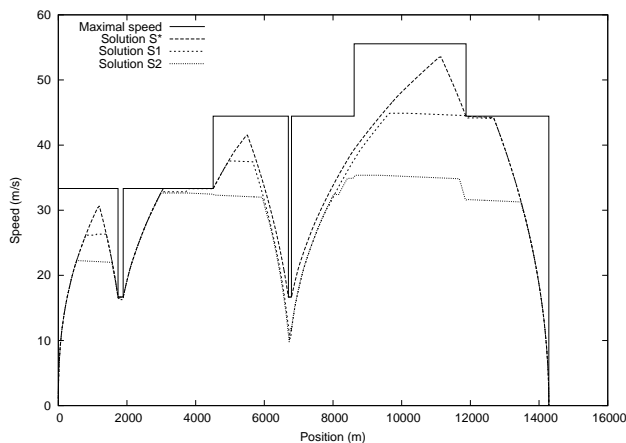


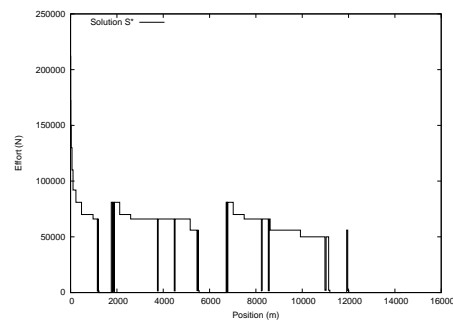
Fig. 6. Example of speed tuning on several sections near Gonesse junction

concurrent objectives: on the one hand by minimizing the travel duration and on the other hand by reducing the energy consumption. To this end we presented an evolutionary multi-objective approach for tuning speeds in order to minimize the durations while saving energy. This algorithm is based on IBEA and uses specific operators well-known in the literature for searching solutions in continuous space. The speed tuning is achieved by our method of speed profile building which introduces two intermediate speeds between the entrance and exit speeds of a section. This method paired with IBEA for searching in a continuous space brings its efficiency to the light.

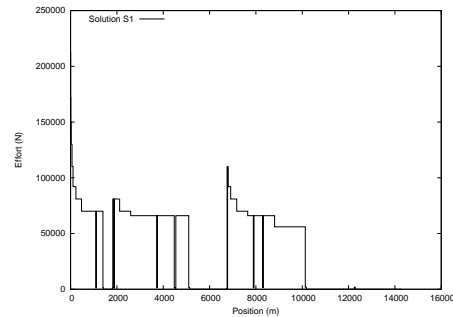
In our opinion, research for optimization in energy saving will intensify and multi-criteria approaches such as evolutionary algorithms are a good way to solve these problems. In the future, we will focus more on train dispatching approaches combining energy saving, conflict resolution and taking uncertainty [16] into account.

REFERENCES

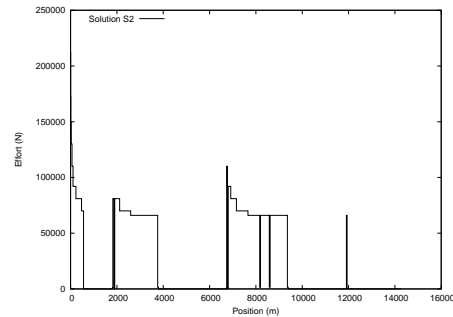
- [1] D. Lancien and M. Fontaine, "Calculs de marches de train économisant l'énergie de traction – le programme MARECO," *Revue Générale des Chemins de Fer*, pp. 679–692, november 1981.
- [2] S. Iwnicki, Ed., *Handbook of Railway Vehicle Dynamics*. Taylor and Francis Group, 2006, 535 p.
- [3] T. Albrecht and S. Oettich, *Computers in Railways VIII*. Southampton: WIT Press, 2002, ch. A new integrated approach to dynamic schedule synchronization and energy saving train control, pp. 847–856.
- [4] R. Liu and I. Golovitcher, "Energy-efficient train control," *Transportation Research part(A)*, vol. 37, pp. 917–932, 2003.
- [5] T. Albrecht, *Railway Timetable & Traffic – Analysis, Modelling, Simulation*. Eurail press, 2008, ch. Energy-Efficient Operation Train Operation, pp. 83–105.
- [6] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, no. 11, pp. 341 – 359, 1997.
- [7] C. Chang, D. Xu, and H. Quek, "Pareto-optimal set based multiobjective tuning of fuzzy automatic train operation for mass transit system," *IEE Proceedings–Electric Power Applications*, vol. 146, no. 5, pp. 577–583, september 1999.
- [8] P. Fleming and R. Purshouse, "Evolutionary algorithms in control systems engineering: A survey," *Control Engineering Practice*, 2002.
- [9] E. G. Talbi, *Metaheuristics: from design to implementation*. Wiley, 2009, 624 p.
- [10] K. Deb, *Multi-objective Optimization using Evolutionary Algorithms*. Wiley, 2001, 517 p.
- [11] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii," in *Parallel Problem Solving from Nature PPSN VI*, vol. 1917/2000. Berlin/Heidelberg: Springer, 2000, pp. 849–858.
- [12] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm," Computer Engineering and Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, Tech. Rep. 103, 2001.
- [13] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Parallel Problem Solving from Nature - PPSN VIII*, vol. 3242/2004. Berlin/Heidelberg: Springer, 2004, pp. 832–842.
- [14] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," Indian Institute of Technology, Kanpur, Tech. Rep. IITK/ME/SMD-94027, 1995.
- [15] A. Liefooghe, L. Jourdan, and E.-G. Talbi, "A unified model for evolutionary multiobjective optimization and its implementation in a general purpose software framework: ParadisEO-MOEO," INRIA, Research Report RR-6906, 2009.
- [16] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments - a survey," *IEEE Transactions on Evolutionary Computation*, vol. 9, pp. 303–317, 2005.



(a) Effort during Solution S^*



(b) Effort during Solution S_1



(c) Effort during Solution S_2

Fig. 7. Train effort according to its position for each solution: S^* , S_1 , S_2