Optimal train speed profiles by dynamic programming with parallel computing and the fine-tuning of mesh

G. Matsuura & M. Miyatake
Sophia University, Japan

Abstract

In this paper the authors investigated an algorithm optimizing a train speed profile by the Bellman's Dynamic Programming (DP). The DP-based method has substantial advantages of coping with complicated conditions easily, e.g. speed limitation, non-linear tractive effort and running resistance, effects of regenerative electric energy and so on. One of the major drawbacks of DP is that it requires a lot of computation time. If high accuracy of solutions is required, computation time for the optimization will increase. In this paper, the authors introduce the parallel computing technique for DP. The parallel computing technique will shorten computation time sharply and succeed in both raising accuracy of simulation and shortening of computation time. While distance between stations for the profile optimization is 1000m at the longest in our previous work, it will be prolonged significantly, keeping a comparable computation time. In this paper the computation times with and without parallel computing will be compared. DP has a further advantage in its use as a real time control algorithm to which the optimal profile can be easily reconfigured against some disturbances such as signalling.
Keywords: train speed profile, dynamic programming, optimization, energy-saving, parallel computing technique, computation time.

1 Introduction

Energy saving technology to reduce CO_2 emission and air pollutant from the railway sector has been developed for more green transportation. Other than improving traction hardware efficiency, some attempts to develop energy

WIT Transactions on The Built Environment, Vol 135, © 2014 WIT Press
www.witpress.com, ISSN 1743-3509 (on-line)
doi:10.2495/CR140641

efficient operation and driving can be found. It is advantageous because it needs less cost for achieving enough energy reduction.

It is known that operation of eco-friendly trains without allowing for regenerative power is done by accelerating maximum, coasting as long as possible, and then decelerating maximum. However, it cannot be said that operation of conventional eco-friendly trains is the most optimal way since railcars with DC/AC inverter control, which function with regenerative brake, are popular on a wide scale.

Some of the authors proposed the application of the DP to optimize energy-saving speed profile of a train ten years ago [1]. After the proposal applicable to actual train operation, some research groups have proposed various types of methods to solve the speed profile optimization problem [2–3], however the DP based method still has substantial advantages of coping with complicated conditions easily, e.g. speed limitation, non-linear tractive effort and running resistance, effects of regenerative electric energy and so on.

One of the major drawbacks of the DP is that it requires a lot of computation time. If high accuracy of solutions is required, computation time for the optimization will increase. In this paper, the authors introduce the parallel computing technique for DP. The parallel computing technique will shorten computation time sharply and succeed in both raising accuracy of simulation and shortening of computation time. While distance between stations for the profile optimization was 1000m at the longest in our previous work, it will be prolonged significantly with keeping comparable computation time. The computation time with/without parallel computing will be compared.

2 Mathematical formulation

Let us define the following notation regarding the train movement between stations in order to analyse the optimal train speed profiles.

m, L	total mass of train and total running distance
$t, T, x(t), v(t)$	current time, trip time, train position and velocity
$u(t)$	control input to determine acceleration, coasting or deceleration
$p(t), J$	input/output power and total consumed energy of train
$\xi(u)$	motor/generator efficiency
$f(u, v)$	acceleration/deceleration force with motor/generator
$r(x, v)$	deceleration force with running and incline resistance
$C(x, v)$	objective function about speed constraints depending on train position

The control input u decides the values of accelerating/decelerating force. The control input corresponds to the notch level of acceleration/brake handles in actual driving. The force is proportional to u, namely, maximum acceleration at u = 1, coasting at u = 0 and maximum deceleration at n = -1. Table 1 shows difference of operation modes in each notch.

Table 1: Relationship between notch values and operation modes.

Notch value	n = 1	0 < n < 1	n = 0	-1 < n < 0	n = -1
Operation mode	Maximum acceleration	Acceleration	Coasting	Deceleration	Maximum deceleration

The following differential equations (1), (2), (3) describe state the equation of train motion and consumed energy.

$$\frac{dx}{dv} = v \tag{1}$$

$$\frac{dv}{dt} = f(u,v) - r(x,v) \tag{2}$$

$$J = \int_0^T \xi(u)\, m\, f(u,v)\, v\, dv \tag{3}$$

Then optimal energy-saving control problem is formulated as follows.

$$\begin{aligned} &\min_u J \\ &\textit{subj to. eqns } (1), (2), (3) \\ &\qquad x(0) = 0,\ v(0) = 0,\ x(T) = L,\ v(T) = 0 \\ &\qquad C(x,v) \le 0, \quad -1 \le u \le 1. \end{aligned} \tag{4}$$

3 Application of Bellman's dynamic programming (DP)

3.1 Dynamic Programming

Dynamic Programming (DP) [4] is a useful method to find optimal solutions of discrete phenomena based on optimal principle. The optimal principle is defined as "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decision must continue an optimal policy with regard to the state from the first decision".

3.2 Basic DP algorithm for optimal control problem

When applying DP algorithm to an optimal control problem, it is necessary to transform original problem into multistage decision process. Generally, this conversion is accomplished by linearization and time-uniform discretization as shown in Figure 1. It is also indispensable to divide objective state space into lattice. One can obtain discretized linearized state equations (5), (6) using first-order Taylor expansion and trapezoidal rule for approximation of integral.

$$\psi_k = \left(\mathbf{I} - \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\left(\mathbf{I} + \frac{\mathbf{A}\Delta t}{2}\right)\psi_{k-1} + \left(\mathbf{I} - \frac{\mathbf{A}\Delta t}{2}\right)^{-1}\mathbf{B}f_0\Delta t \tag{5}$$

$$J_k = J_{k-1} + \xi(u)m\frac{\Delta t}{2}(f(u,v_k)v_k + f(u,v_{k-1})v_{k-1}) \tag{6}$$

where

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ \alpha & \beta \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad \psi = \begin{pmatrix} x \\ v \end{pmatrix}$$

$$f_0 = f(u_{k-1}, v_{k-1}) - r(x_{k-1}, v_{k-1}) - \alpha v_{k-1} + \beta x_{k-1}$$

$$\alpha = f_v(u_{k-1}, v_{k-1}) - r_v(x_{k-1}, v_{k-1})$$

$$\beta = -r_x(x_{k-1}, v_{k-1}).$$

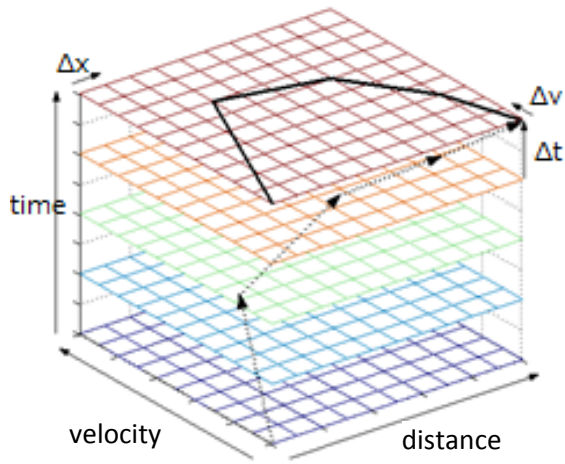

Figure 1: Spatial and time discretization.

In addition, DP needs to transform terminal boundary conditions to penalty function eqn (7) which gives the square of error of each state variable at the final stage.

$$\phi(x(T), v(T)) = \lambda_1(x(T) - L)^2 + \lambda_2(v(T))^2 \tag{7}$$

Thus, optimal problem eqn (4) is approximately converted into the following N-stage decision process eqn (8) without the terminal boundary conditions.

$$\begin{aligned} &\min_{\{u_k\}_{k=1}^N} \{J + \phi(\psi_N)\} \\ &\textit{subj to.}\ (5),\ C(\psi_k) \le 0,\ -1 \le u_k \le 1. \end{aligned} \tag{8}$$

Finally, DP process can be implemented into a digital computer as the following steps.

(a) A penalty value $\phi(x_N, v_N)$ is given for each lattice point on the N-th phase.
(b) The phase k is set to N-1.
(c) The optimal policy is determined on every lattice point of state space with solution of eqn (5) and local valuation which is calculated by bilinear interpolation. If $k = 0$, go to (e).
(d) The phase k is decreased by 1. The step goes back to (c) until $k > 0$.
(e) The forward search is done from an origin along the trajectory already created by the backward-search procedure described above.

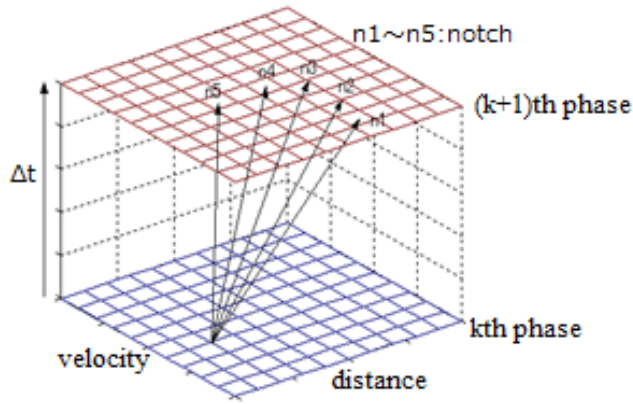

Figure 2: Finding the optimal control input at each lattice point.

The advantage of DP applied to this formulation is the easiness of handle some constraints of state variables and possibility of real time control. Details are listed as follows:

(1) It is possible to combine track conditions, such as discontinuity of multistage notches, velocity limitation, and gradient resistance, which are difficult to be combined as solution method into simulations easily.
(2) Even if the actual trajectory becomes away from the optimal one due to some disturbances by signaling and so on, it is able to recover the trajectory to the optimal one. If the closest lattice point from the point which can detect its position and velocity can be given as primary value and notch value which optimizes cost function can be read out. Here, notch value gives command for powering and brake in the case of train operation.

3.3 Separation method in state space

Figure 3 shows the discretized state space by non-uniform mesh in order to reduce computation cost and keep enough accuracy. The symbols x and v are boundary values of different density of lattice points. It needs to optimize with fine width of separation to adjust penalty value around terminal point. Thus, it is necessary to have minor width of separation for $\Delta x2$, $\Delta v1$ around terminal point.

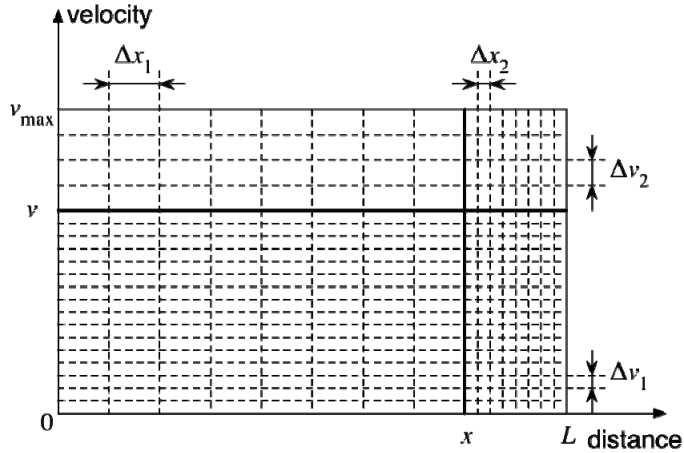

Figure 3: Non-uniform mesh in the state space.

3.4 Application of the parallel computing technique to DP

Although calculation for each lattice point is simple, DP requires a lot of computation time because there are quite a lot of lattice points. If high accuracy of solutions is required, computation time for the optimization will increase. Therefore, accuracy of simulation and shortening computation time has a trade-off relation. The authors proposed to apply the parallel computing technique to DP since the entire computation by DP contains many independent processes. It can shorten computation time sharply and succeed in both raising accuracy of simulation and shortening of computation time.

In this study, the authors implemented the DP with parallel computing on the MATLAB program. Figure 4 is an image of parallel computing technique. The worker is a MATLAB computational engine.

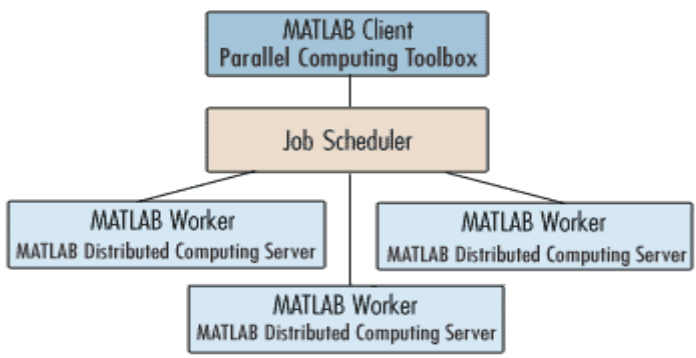

Figure 4: Image of parallel computing technique [5].

4 Demonstration through computer simulation

4.1 Conditions of simulation

The authors made simulation of optimizing the speed profile with parallel computing techniques in order to demonstrate the advantage of computation time and possibility of applying to the long-distance interstations. The conditions of simulations are set as the following items.

(1) Energy conversion efficiencies of motors in acceleration/regenerating are assumed as 90% and 80% respectively referring from a real railcar. Auxiliary power consumption is not considered in this study.
(2) The accelerating/decelerating force is independent on catenary voltage.
(3) No consideration with regenerating squeezed control.
(4) Set regenerative power is returned to substations with 2 inverter functions, consumed without regeneration cancelled.
(5) Braking force obtained in deceleration is always constant in each notch. Ideal mechanical brake compensates shortage of the braking force in the case of not being able to cover all braking force by regenerative braking power.
(6) The mesh parameters shown in Figure 3 are set as Table 2 for each length of interstation. It is determined by trials and errors in order to balance the terminal point errors and energy consumption.

Table 2: The value of width of separation in Figure 3.

Distance [m]	⊿x1[m]	⊿x2[m]	x[m]	⊿v1[km/h]	⊿v2[km/h]	v[km/h]
1000	5	0.5	930	0.5	1	40
10000	25	1	9500	0.5	1	40

4.2 Simulation results

4.2.1 1000[m] run in a straight line

Table 3 and Figure 5 are the optimization results in straight 1000m line. From Table 3, it can be found that the terminal condition errors are quite small. If the intestation is short as this case, the optimized speed profile is the same as well-known one. It is composed of maximum acceleration, coasting and maximum deceleration. Increasing the coasting is regarded as the most important point of view.

Table 3: Optimization results in straight 1000m line.

Running Time [s]	Last Point [m]	Final Speed [km/h]	Consumption energy [kWh]
70	999.988202	-0.109356	8.487945

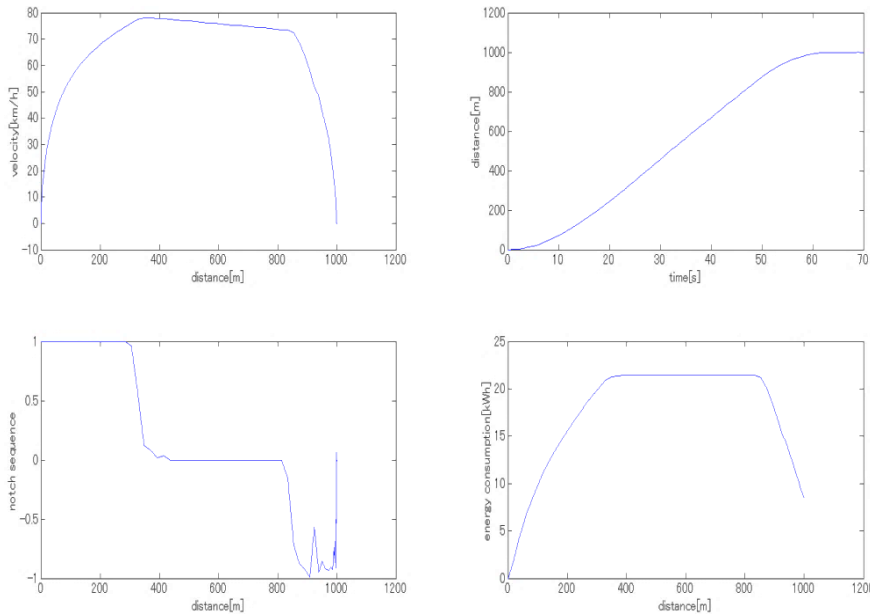

Figure 5: Optimal speed profile in straight 1000m line.

4.2.2 10000m in a straight line

The next condition is that the interstation is ten times longer than the previous condition. Figure 6 is the optimized speed profile which is quite different from Figure 5. Since coasting time cannot be long due to influence of mechanical loss, constant speed mode must be inserted between acceleration and coasting.

The conventional speed profile using long constant speed mode plotted in Figure 7 are compared. The conventional profile is created by giving severe maximum speed limitation 120km/h. Table 4 shows comparison of constant-speed operation and optimization in straight 10000m line. The total consumption energy of optimization is less than constant-speed operation.

peed operation and optimization in straight 10000m line. The total consumption energy of optimization is less than constant-speed operation.

Table 4: Comparison of constant-speed operation and optimization in straight 10000m line.

	Running Time [s]	Last Point [m]	Final Speed [km/h]	Consumption energy [kWh]
Optimization	400	9999.961556	-0.086787	86.284084
Constant-velocity operation	400	9999.956555	-0.081028	90.061799

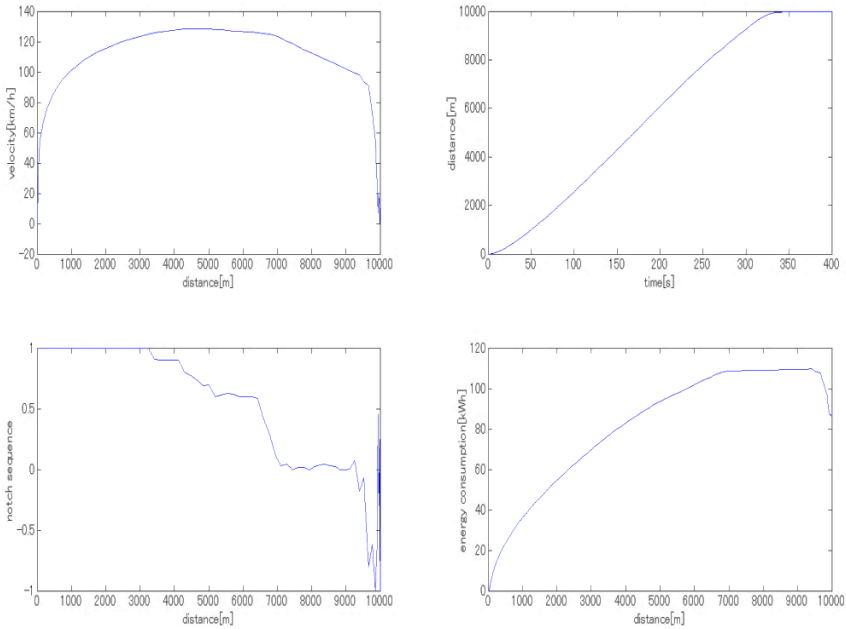

Figure 6: Optimal speed profiles in straight 10000m line.

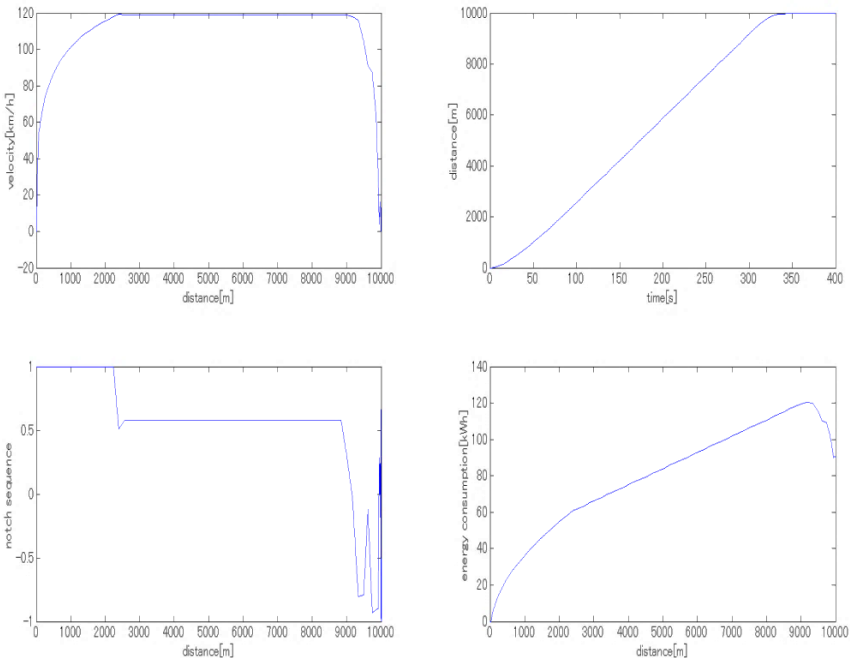

Figure 7: Constant-speed operation in straight 10000m line.

4.2.3 Effect of computation time by the parallel computing technique

Figure 8 shows the computation times with/without parallel computing. Computation time is almost inversely proportional to number of workers in each length of interstation. The maximum available number of workers is 12. In this study the authors used 1, 2 and 4 workers, because of the limitation of computer specification. If GPU computing is available, the computation time is expected to be drastically reduced.

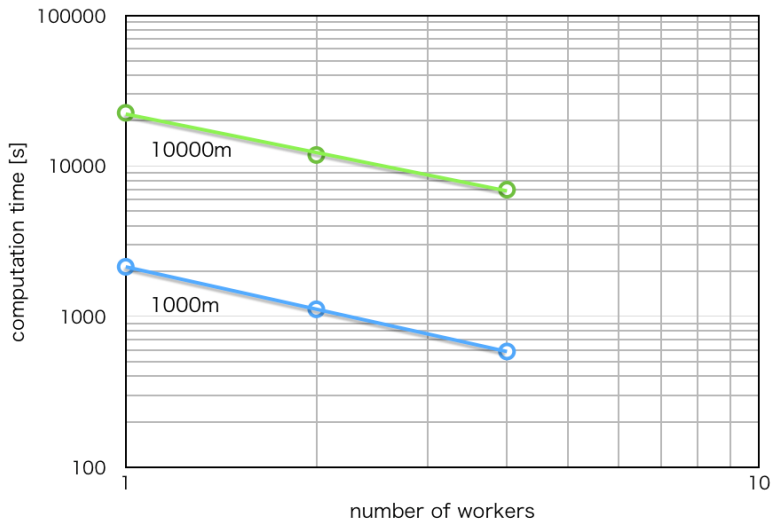

Figure 8: Relation between number of workers and computation time.

5 Conclusion

DP requires a lot of computation time, however, introducing the parallel computing technique, computation time was shortened sharply. It was basically inversely proportional to the number of core. The reduced computation time enables to apply the DP based optimization of speed profiles to the long-distance interstations to 10km for the first time. The solvability was checked by some simulations.

References

[1] Ko, H., Koseki, T. & Miyatake, M., Application of Dynamic Programming to Optimization of Running Profile of a Train, *Computers in Railways IX*, 2004.

[2] Miyatake, M., & Ko H., Optimization of Train Speed Profile for Minimum Energy Consumption, *IEEJ Transactions on Electrical and Electronics Engineering*, Vol. 5, No. 3, pp. 263-269, 2010.

[3] Wang, Y., Ning, B., Cao, F., Schutter, B.D., Boom, Ton J.J. van den, A Survey on Optimal Trajectory Planning for Train Operations, *IEEE International Conference on Service Operations, Logistics and Informatics*, pp. 589-594, 2011.
[4] Bellman, R., Kalaba, R., Dynamic Programming and Modern Control Theory, *Academic Press*, 1964.
[5] MathWorks Website, Installing and Configuring Parallel Computing Toolbox and MATLAB Distributed Computing Server, http://www.mathworks.co.jp/support/product/DM/installation/ver_current/.