NOTES ON THE PROPERTIES OF DYNAMIC PROGRAMMING USED IN DIRECT LOAD CONTROL SCHEDULUNG

Isto Aho

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCES
UNIVERSITY OF TAMPERE

REPORT A-2000-12

UNIVERSITY OF TAMPERE
DEPARTMENT OF COMPUTER AND
INFORMATION SCIENCES
SERIES OF PUBLICATIONS A
A-2000-12, AUGUST 2000

NOTES ON THE PROPERTIES OF DYNAMIC PROGRAMMING USED IN DIRECT LOAD CONTROL SCHEDULING

Isto Aho

University of Tampere
Department of Computer and Information Sciences
P.O.Box 607
FIN-33014 University of Tampere, Finland

ISBN 951-44-4894-4
ISSN 1457-2060

Notes on the properties of dynamic programming used in direct load control scheduling

Isto Aho   tyisah@cs.uta.fi
Dept. of Computer and Information Sciences
University of Tampere, Tampere, Finland
Report A-2000-12

9th August 2000

Abstract

We analyze a dynamic programming (DP) solution for cutting the overload in electricity consumption. We are able to considerably improve the earlier DP algorithms presented in the literature. Our improvements make the method practical so that it can be used more often or, alternatively, new state variables can be added into the state space to make the results more accurate. We also propose a way to add a new state variable to the state space. By using different numbers of state variables we can build up a hierarchy of solutions, in which we can trade between rapidity and accuracy. A similar trade-off situation occurs also between low and high number of states a variable is allowed to have.

1 Introduction

Shortage of electricity may cause supplier to use direct load control (DLC); the supplier may turn off the electricity from some of its customers or may start generators to meet the demand. The goal is to minimize the losses by buying minimum amount of the (expensive) electricity from other suppliers to cover the demand after DLC. Some customers have a consumption profile containing payback. A typical example of control group with payback is a residential appliance with electricity heaters or air conditioners. When the supplier controls the devices, a consumption peak appears after the control period when the devices go back to their normal state [1].

DLC has been solved by using various algorithms. Typically, objectives, models and methods differ from paper to paper. In [2] we study a simple model, interactive knapsacks, which is a common simplification to many earlier models. We show that several problems associated to interactive knapsacks are NP-complete [2], i.e. computationally intractable (see [3] for NP-theory). We also show a simplification of the present successive approach to control a group at time to be NP-complete, if the group has payback [2]. Hence, it is reasonable to study successive approaches separately, too. (It is an open question, how the fixed maximum length of payback affects the NP-status.)

Different solution methods include DP [1, 4, 5, 6], linear programming (LP) [7, 8], heuristics [1, 9], enumeration methods and other optimization methods (see [4]), like hybrids of LP and DP [10].

Objectives include load reduction minimization [4, 5, 6], peak load minimization [10, 7], minimizing production costs [4, 6], and maximization of profits [8].

DLC is often combined with unit commitment and economic dispatch, and the applied methods include DP [11] (fuzzy DP [12, 13], stochastic DP [14]), binary network flow formulation [15], and evolutionary strategies [16]. Yan and Luh [17] consider unit commitment with “purchasing emergency power with very high prices”, a similar motivation as we have. See also [18].

Our method is somewhat similar to that of [4, 5], and our model has a “similar spirit” than [8]. We have earlier [1] developed the models and methods of [4, 5] by adding new properties to them. DP solutions of [4] is not optimal, if applied to a group at a time and if the loads are evened out on hourly bases [1]. New state variables improve the results. Moreover, DP of [5] optimizes several groups at a time and therefore needs too large state space to be practical in our case.

The present work further deepens the results of [1] by focusing on DP solution and its properties. Our solution determines the number of controls needed, and a starting time with a duration and resting time for each control (we use 5 minutes precision). Number of controls can also be used as a restriction. Our solution can be used as a successive optimization method.

Our objective is “in between” the minimizing of load reduction and the minimizing of peak load and is different from the objectives presented in other papers. Purchase transactions of electricity and own production give optimum level to be resold at each hour, while load over the optimum level has (very) high price. If demand is higher than our predefined level, we want to cut (expensive) “over loads” and at the same time minimize the losses caused by decreased sales. Optimum level is not usually attainable exactly (because of discretisation). We also use purchasing and reselling prices (time-of-use rates) in the formulation of the objective.

Furthermore, our solution can use different objectives without major modifications of the method. The same holds for the prices, if one needs more complex price structures, and for energy storages of [19, 20].

Subsections 2.1 and 2.2 describe the model in its present augmented form, and Subsection 2.3 derives the DP solution used. The main results concern the "wait states" and "worsening states" (state variables) needed in DP are given in Section 3. We also build up a hierarchy of DP solutions so that it is possible to choose between fast and inaccurate and slow but accurate methods. We show in Subsection 3.1 how the number of wait states needed can be diminished to be about half of the number used in [1]. This fastens the whole optimization process approximately by the same factor. State space can also be decreased with multi-pass DP of [6], but then one should be able to relax some constraints (method is presented for DLC dispatch problem in [6]).

In Subsection 3.2 we show how to add a fourth state variable (worsening states) into the model of [1]. Extensive testing is reported in Section 4, supporting the assumption that we need wait and worsening states in order to get good results.

2 Model

First we describe the model for the problem and give some definitions. The model to be given is a slight simplification of the model used in load clipping [1]. In Subsection 2.1 we give the base of the model and in Subsection 2.2 we discuss on real world restrictions and on the goal functions. Table 1 contains relevant symbols used in this work.

2.1 Basic model

An *interval* $[a, b]$ is the set $a, a+1, \ldots, b$ $(a < b)$ of integers. The *length* of an interval $[a, b]$ is $b - a + 1$. A *clipping situation* $\mathbf{s}$ is a vector $s_0, s_1 \ldots, s_N$ $(N > 0)$ of reals representing the difference between electricity demand and electricity production in time interval $[0, N]$. The domain $[0, N]$ is called the *optimization interval* and values s_i are called either *overload* or *underload*. Overload represents a situation, where demand is higher than combined production and electricity purchases $(s_i \geq 0)$ and underload represents a situation, where combined production and purchases of electricity is above the level of consumption, i.e. demand $(s_i \leq 0)$. The element i of optimization interval $[0, N]$ is called a *time point*. The phrase "time point i" is

Table 1: Used symbols

Clipping situation	$\mathbf{s} = [s_0, s_1, \dots, s_N]$	Set of controls	$\mathbf{C}$
Prices	$\mathbf{p} = [p_0, p_1, \dots, p_N]$	Control capacity	C^c
Revenue	$\mathbf{r} = [r_0, r_1, \dots, r_N]$	Control length	C^l
Length of hour	h^l	Resting time	C^r
Time interval or control	$[a, b]$	Minimum control length	C^m
Loss of incomes	$R(\mathbf{s})$	Maximum control length	C^M
Optimal control plan	$R^*(\mathbf{s})$	Maximum control times	C^T
Dynamic forward recursion	$R'(\mathbf{s}, S', k+1)$	Control time	C^t
Stage change	$R''(\mathbf{s}, S', S, k+1)$	Length of payback	P^l
Wait state	W	Amount of payback[a]	P^c
Worsening state	B	Impact of a control	$I([a,b], \mathbf{s})(k)$
State (3 variable)	$S = (C^t, W, C^l)$	Impacts of all controls	$I'(\mathbf{C}, \mathbf{s})$
State (4 variable)	$S = (C^t, B, W, C^l)$		

[a]Amount of payback corresponds to capacity explaining the c-superscript.

used also for the interval $[i, i]$.

Every time point i with overload has a positive real p_i called the *price factor* (buying price of electricity). If at time point i there is underload, the positive real $r(i)$ is the *revenue factor* (selling price of electricity). The *overload interval* is an interval $[a, b] \subseteq [0, N]$ such that at every time point $i \in [a, b]$ there is overload.

The clipping situation is partitioned into *hours* $0 = a_1, a_2, \dots, a_{n+1} = N$, of equal length, i.e. $a_{i+1} - 1 - a_i + 1 = a_i - a_{i-1}$ $(2 \le i \le n)$. The length $a_{i+1} - a_i$ of an hour is denoted by h^l. Hour i refers to the interval $[a_i, a_{i+1} - 1]$. Overloads (underloads), revenue and price factors do not change during an hour, because of the system used in electricity trading. Thus, we have $s_j = s_{j+1}$, $p_j = p_{j+1}$ and $r_j = r_{j+1}$, where $j \in [a_{i-1}, a_i - 2]$ $(2 \le i \le n)$.

The total loss is

$$R(\mathbf{s}) = \sum_{i \in [0,N]} K(i, s_i), \tag{1}$$

where

$$K(i, s_i) = \begin{cases} -p_i s_i, & \text{if } s_i \ge 0, \\ r_i s_i, & \text{otherwise.} \end{cases} \tag{2}$$

Hence, we always have $K(i, s_i) \le 0$. If there is underload, we lose income (revenue) and if there is overload, we have to pay some extra. The sum (1) counts the money lost, so its best possible value is 0.

A *group* can lower the overload with a *control*, i.e. an interval $[a, b]$. The *controlling capacity* of a group, denoted by C^c, is the amount by which the group can lower the load in an hour. The hours $[a_i, a, a_{i+1}, \ldots, a_{j-1}, b, a_j]$, where $a_i \leq a < a_{i+1}$ and $a_{j-1} < b \leq a_j$ (and $a < b$) have to be taken into account when making a control. The total influence of a control is called the *control amount* and it is the product of the controlling capacity C^c and of the control length $b - a + 1$.

2.2 Payback, restrictions and goal function

Function $P^l : \mathbb{N} \to \mathbb{N}$ maps the control length $b - a + 1$ to the length of a payback and function $P^c : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ describes the amount of the payback of a control of length $b - a + 1$ at time i. Moreover, we always have $P^c(b - a + 1, i) \geq 0$, where $i \in [b, b + P^l(b - a + 1)]$, and otherwise $P^c(b - a + 1, i) = 0$. Further, "in practice" we have

$$\sum_{k \in [b, b+P^l(b-a+1)]} P^c(b - a + 1, k) \leq C^c(b - a + 1)$$

meaning that a payback does not exceed the control amount.

Next we define the impact of a control and its payback to a clipping situation $\mathbf{s}$ as a function I and then we show how one can calculate all controls into the clipping situation, function I'. The hours to be affected are $[a_i, a, b, a_j, b + P^l(b - a + 1), a_l]$. With function

$$I([a,b], \mathbf{s})(k) = \begin{cases} s_k, & \text{when } 0 \leq k < a_i, \\ s_k - C^c(a_{i+1} - a)/h^l, & \text{when } a_i \leq k < a_{i+1}, \\ s_k - C^c, & \text{when } a_{i+1} \leq k < a_{j-1}, \\ s_k - C^c(a_j - b)/h^l + \displaystyle\sum_{k'=b}^{a_j - 1} \frac{P^c(b-a+1, k')}{h^l}, & \text{when } a_{j-1} \leq k < a_j, \\ s_k + \displaystyle\sum_{k''=a_{k'}}^{a_{k'+1}-1} \frac{P^c(b-a+1, k'')}{h^l}, & \text{when } j \leq k' < l \text{ and } a_{k'} \leq k < a_{k'+1}, \\ s_k + \displaystyle\sum_{k'=b}^{a_l - 1} \frac{P^c(b-a+1, k')}{h^l}, & \text{when } a_{l-1} \leq k < a_l, \\ s_k, & \text{when } a_l \leq k \leq N, \end{cases} \tag{3}$$

$(0 \leq k \leq N)$ we obtain the total influence of control $[a, b]$ into the clipping situation. (It would simplify formula (3) a bit if we were not to hourly even out the affects. Another alternative is to let the overloads and underloads vary within the hours and even out the loads when calculating the results.) If the control starts and stops in the same hour, we have to replace the second, third and fourth line of (3) with the

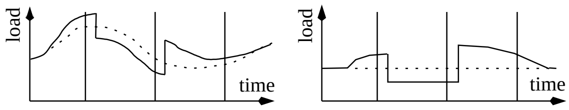

Figure 1: Realistic and theoretic clipping situation with a control. Controlled group can store energy and has payback.

line

$$s_k - C^c(b-a+1)/h^l + \sum_{k'=b}^{a_j-1} P^c(b-a+1,k')/h^l, \tag{4}$$

where $a_{j-1} \leq k < a_j$. If the payback starts and stops in the same hour, we have to make a correction similar to (4) to the calculation of the effects of payback. Energy storage capability is similar to payback: storage appears before control while payback appears after the control.

In function I of (3) the first and last lines leave the uncontrolled hours as they are. The second line calculates the effects for the hour where the control starts (either the overload gets lower or the underload grows). The third line handles hours that are between the starting and stopping hours, if any. The fourth line handles the hour where the control stops. The payback starts here. The payback increases the overload and the rest of the control decreases it. The third last line calculates the payback for hours, where the payback does not end and the second last line deals with the last payback hour.

Figure 1 shows two examples of a control. The left picture describes a control in a realistic situation and the right one in a theoretical situation. The vertical lines are hours. The dotted line is clipping situation without control and the straight line is a clipping situation with control. The left picture shows the advantage of a control: if a group has payback (can store energy), we can "move" the overload (stored energy) to the next (previous) hour where the overload is cheaper.

The effects of all controls $\mathbf{C}$ can be calculated recursively by the function

$$I'(\mathbf{C},\mathbf{s}) = \begin{cases} I'\big(\mathbf{C}-[a,b], I([a,b],\mathbf{s})\big), & \text{if } [a,b] \in \mathbf{C}, \\ \mathbf{s}, & \text{if } \mathbf{C} = \emptyset. \end{cases} \tag{5}$$

When all controls have been calculated, we can use (1) to find out the value of the new clipping situation.

Next we look at the restrictions. First, the controls must be separate such that for all $[a,b],[c,d] \in \mathbf{C}$ we have

$$[a,b] \neq [c,d] \Rightarrow [a,b] \cap [c,d] = \emptyset. \tag{6}$$

Further, during *resting time* it is not allowed to start a new control. Function $C^r : \mathbb{N} \to \mathbb{N}$ is increasing and it maps the length of a control to the length of a resting time. So, for all $[a, b], [c, d] \in \mathbf{C}$ we have

$$[a, b] \cap [c, d] = \emptyset \Rightarrow [b, b + C^r(b - a + 1)] \cap [c, d] = \emptyset. \tag{7}$$

Notice that a new control can be started even if the payback still occurs if the resting time does not overlap with the new control. Usually, the resting time is used to prevent a new control to start in the beginning of the payback, when the need for extra electricity is the largest. If we start a new control at the end of a payback, the change in the payback of the new control is so small that it is not usually taken into account. We could also modify equation (3) to take into account the previous control (or controls) and its (their) possible potency to the payback of the present control, when using too short resting time.

We also need the *minimum* and *maximum* control times C^m and C^M, respectively, and hence

$$c^m \leq b - a + 1 \leq C^M. \tag{8}$$

Sometimes we also restrict the amount of control times $\sum_{[a,b]\in\mathbf{C}} 1$ by C^T, a positive integer.

We can suppose that at every time point k the price factor p_k is (much) larger than the revenue r_k. By making controls we can affect the clipping situation, so the optimization problem can be given in the form

$$\max_{\mathbf{C}} \sum_{k=0}^{N} R(k, I(\mathbf{C}, \mathbf{s})) \tag{9}$$

with restrictions (6)–(8). The objective indicates the income lost and its theoretic maximum is 0.

2.3 Solution with dynamic programming

The problem (9) can be solved with dynamic programming (DP) [1, 4, 5, 18]. One way is to use a very large state space to find optimal control plan $\mathbf{C}$, for example

$$R^*(\mathbf{s}) = \max_{[a,b]} R^*(I'(\mathbf{C} \cup [a, b], \mathbf{s})). \tag{10}$$

In solution (10) different controls form the state space and the number of control times form the stages. This solution enumerates numerous different controls and the state space grows too large. Moreover, this solution is sub-optimal (as well as the method to be presented). The dynamic programming (10) at stage n is equal to the optimization problem

$$R^*(\mathbf{s}) = \max_{[a,b]_1} \cdots \max_{[a,b]_n} R^*(I'([a,b]_1 \cup \cdots \cup [a,b]_n)).$$

At the first stage we have to check approximately $(C^M - C^m)N$ states, where C^m is the minimum length, C^M is the maximum length and N is the number of time points in the optimization interval. At the second stage, when we form the second control, we need to find a connection to all $(C^M - C^m)N$ states. The connection can point to approximately $(C^M - C^m)N$ states. Hence, we should check about $(C^M - C^m)^2N^2$ states, which is too much as both $C^M - C^m$ and N can grow large enough to make this method unpractical.

If we have found optimal five controls first giving clipping situation s' and then find other five controls being optimal against s', the global optimum is not necessarely found. We may achieve better results by using only nine controls having no common control with the previous two sub plans. Principle of optimality (see [21]) is not fulfilled.

Cohen and Wang [4] use two state variables, the control times and control length. Our tests [1] indicated that two state variable systems are fast enough so that we can add (see [21], pp. 30–34) at least one state variable to make results better. In this work we use the state variables *control time* C^t, *wait state* W and *control length* C^l. This way we have a slower but more accurate system than those with two state variables. Each of these variables are defined in finite integer domain. The wait states are used to delay the start of a control while C^t and C^l have obvious meaning.

In the state space we need the control length, so that DP can form the optimal control length and at the same time consider the restriction (8). The variable control length C^l contains the control length and the resting time. Without the control times DP will find only one control, if DP obeys conditions (6)–(8). With these state variables we have one state of stage $k \in [0, N]$ as a triple

$$(C^t, W, C^l)(k). \tag{11}$$

A system in state (C^t, W, C^l) is defined to be a C^l long C^tth control, of which start is delayed W time points. Our tests demonstrate that the three variable (C^t, W, C^l) solution does not give the optimal solution in every situation, especially if there is payback (see Section 4).

The phrase "stage k" refers to a time point. In practice we have to determine upper bound for the wait states W. Theorem 2 gives an upper bound for W when the group does not have payback. If the group does have payback, we assume that W can have $h^l - 1$ (h^l is the length of an hour) different values (we also test other amounts, see Section 4).

In the next equation we use the notations $S = (C^t, W, C^l)$ and $S' = (C'^t, W', C'^l)$. The variables with primes are "new" ones and the variables without the primes are "old" ones, when we form the connections from the "new" stage $k+1$ to the "old" stage k. The function

$$R''(\mathbf{s}, S', S, k+1) = \begin{cases} 0, & \text{when (15)–(18)}, \\ -P', & \text{when (19)}, \\ R(I([k - C^t, k], \mathbf{s})) - R(\mathbf{s}), & \text{when (20)}, \\ -\infty, & \text{otherwise}, \end{cases} \tag{12}$$

gives the change in the value, when we move from the state (C^t, W, C^l) of the stage k into the state (C'^t, W', C'^l) of the stage $k+1$. In other words, R'' equals the value of the connection between states (C^t, W, C^l) and (C'^t, W', C'^l) at the stages k and $k+1$, respectively. The first line is used, when the value does not change. The third line is applied, when we make a decision about the best control judged by (1), and the second line is used, when we start a new control. In these situations, we add to the value the cost of making a control. The last line is used with every other values of the variables S and S'. They are impossible since they do not have any reasonable real world interpretation.

The dynamic forward recursion equation is

$$\begin{aligned} R'(\mathbf{s}, S', k+1) = \max_{S} \big(& R'(\mathbf{s}, S, k) + \\ & R''(\mathbf{s}, S', S, k+1) \big) \end{aligned} \tag{13}$$

and

$$R'(\mathbf{s}, (C^t, W, C^l), 0) = \begin{cases} R(\mathbf{s}), & \text{when } C^t = W = C^l = 0, \\ -\infty, & \text{otherwise}. \end{cases} \tag{14}$$

When

$$\begin{gathered} C'^t = C^t + 1, \quad W' = C'^l = 0, \quad 0 \le W \le h^l - 2, \\ \text{and} \quad C^r(C^m) \le C^l - C^M + 1 \le C^r(C^M), \end{gathered} \tag{15}$$

we think that control at the stage k and in the state (C^t, W, C^l) has stopped, which in turn increases the amount of control times by one ($C'^t = C^t + 1$). The next control starts in the state $(C'^t, 0, 0)$ at the stage $k+1$. We use $C^l - C^M + 1$, because

C^l indexes both the control length and the resting time (see Figure 2). The states corresponding to resting are located next to the control lengths. Notice, that we restrict the number of wait states by $h^l - 2$ (wait states can get $h^l - 1$ different values).

Moreover, it is possible that "old" optimal plan at the stage k does not change (or be better) when we move to the stage $k+1$, and so

$$C'^t = C^t, \quad W' = C'^l = 0, \quad \text{and} \quad W = C^l = 0. \tag{16}$$

This is the only case with conditions (15) and (20), when DP (recursion formula (13)) can make choices about the optimal path. If two paths give the same result, DP (13) chooses the one with a later control. This does not have any impact on the result, but in practice we usually want to do the controls as late as possible. When

$$C'^t = C^t, \quad W' = W + 1, \quad \text{and} \quad C'^l = C^l = 0, \tag{17}$$

we "move some information from the past to the present". With this information we can check what result can be achieved, if we choose the best path W stages ago instead of some other control plan from the interval $[k - W, k]$. When

$$\begin{gathered} C'^t = C^t, \quad W' = W, \quad C'^l = C^l + 1 \\ \text{and} \quad (C'^l \neq 1,\ C'^l \neq C^M + 1), \end{gathered} \tag{18}$$

we increase the control length by one time point (the control started C^l time points ago). When

$$C'^t = C^t \quad W' = W, \quad \text{and} \quad 1 = C'^l = C^l + 1, \tag{19}$$

we have started a new control. In this situation we add to the result the cost of control P'. When

$$\begin{gathered} C'^t = C^t \quad W' = W, \quad C'^l = C^M + 1, \\ \text{and} \quad C^m \leq C^l \leq C^M, \end{gathered} \tag{20}$$

we can calculate the impact of a legal control on a clipping situation. Figure 2 shows the state space.

For each state (C^t, W, C^l) and for each stage $k > 0$, we save the connection pointing to some state of the previous stage. The connections form a *path*. When we have the values

$$R'(\mathbf{s}, (C^t, W, C^l), N)$$

with appropriate values in C^t, W and C^l, we can form the control plan $\mathbf{C}$ by traversing the path formed by the connections. The path is optimal with respect to the state space used (but not with respect to the problem).

We end this section by noticing that the functions R' and R'' in equations (12)–(14) do not violate the conditions (6)–(8).

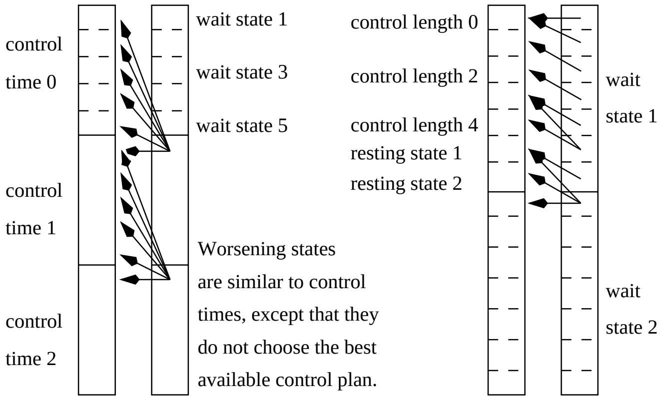

Figure 2: The structure of the state space.

3 The properties of the state space

Next we study the properties of the dynamic recursion formula (13)–(14). In Subsection 3.1 we study how many wait states we need when there is no payback. In Subsection 3.2 we embed worsening state variables into our state space system.

3.1 Paybacks of length zero

Consider a stage i. A *local* control for state $(C^t + 1, 0, 0)$ is a control formed at the control time $C^t + 1$, stopped at the stage $j > i + C^r(C^m)$, and using the control plan formed at the stage i for the state $(C^t, 0, 0)$. The stages $k > i$ do not belong to the local control, provided that we do not use the control plan of the state $(C^t, 0, 0)(i)$ at stage k. This means that the wait states are not considered when forming a local control.

In the next theorem we suppose that all references to wait states have been omitted from the conditions (15), (16), (19) and (20).

Theorem 1 *With the state space* (C^t, C^l) *we will find, for each stage* i*, the best local control following stage* i*.*

Proof. Consider the controls starting after stage i from state $(C^t, 0)$ and using the control plan $\mathbf{C}$ determined by i and $(C^t, 0)$. Conditions (15) and (16) choose the best control for the state $(C^t + 1, 0)(j)$, according to the equation (13). The condition (15) gives the maximum because of the conditions (19) and (20). □

Corollary 1 *State* $(1, 0, 0)(N)$ *gives the best possible control plan having one control.*

Notice that the length or the amount of payback do not have any consequences in the case of Theorem 1. State space (C^t, C^l) gives sub-optimal results [1], which can be improved with wait states (still being sub-optimal).

Let $\mathbf{C}_i, \mathbf{C}_{i+1}, \dots, \mathbf{C}_a$ be the control plans of the control time $C^t = k$ and the stages (time points) $i, i+1, \dots, a$, respectively, i being the first time point of an hour. In the next theorem we show that it is enough to choose between the control plans $\mathbf{C}_i, \mathbf{C}_{i+1}, \dots, \mathbf{C}_a$, when forming $[a, b]$. This refers to the situation in condition (15) of the equation (13), where we check, how well the control plans of states $(C^t, 0, 0)(i), (C^t, 0, 0)(i+1), \dots, (C^t, 0, 0)(a)$ work with the control starting at the time point a.

Intuitively the next theorem is based on the property that if the last control of some control plan stops with resting time in the "previous hour", it will not have any impact on the controls in the "present hour".

Theorem 2 *Suppose there is no payback. Suppose further that from the stage a we start a new control for the control time C^t, which will stop at the stage b locally maximizing the control plan $\mathbf{C}_a$. Let i be the first moment of the hour containing a. When forming a new control $[a, b]$, it is enough to choose (with the wait states) from the set of control plans $\mathbf{C}_i, \mathbf{C}_{i+1}, \dots, \mathbf{C}_a$.*

Proof. We show that it is not necessary to reach time points earlier than the start of the present hour. To do so we consider situations where it is possible to choose between the control plan $\mathbf{C}_{i-1}$ of time point $i-1$, some earlier control plan $\mathbf{C}_{i-n}$ ($n > 1$) and the control plans $\mathbf{C}_i, \mathbf{C}_{i+1}, \dots, \mathbf{C}_a$, when forming $[a, b]$.

Consider the control $[a, b]$ started at the time point a and control time C^t. To derive contradiction we suppose some control plan $\mathbf{C}_{i-n}$ ($n > 0$), when deciding the proper control plan for $[a, b]$. It follows that at least one of the control plans $\mathbf{C}_{i-n+1}, \dots, \mathbf{C}_i$ gives at least as good result at the stage i than $\mathbf{C}_{i-n}$, because DP (13) chooses always maximum. We can suppose that the control plan in question is $\mathbf{C}_i$, since the result of $\mathbf{C}_j$ improves when j increases (not necessarily monotonically). If we choose some of the control plans $\mathbf{C}_{i-n}, \dots, \mathbf{C}_{i-1}$ to be used with a control that starts from a, we obtain better result with the control plan $\mathbf{C}_i$. (Notice that the absence of payback is crucial here.)

The last control moments of the control plans $\mathbf{C}_{i-n}, \dots, \mathbf{C}_{i-1}$ including their resting times are on the earlier hour than a. Suppose next that we choose some control plan $\mathbf{C}' \in \{\mathbf{C}_{i-n}, \dots, \mathbf{C}_{i-1}\}$ and with it a control $[a', b']$, where a' is on the same hour than a. Suppose further, that $\mathbf{C}' \cup [a', b']$ gives better result than $\mathbf{C}_i \cup [a, b]$. According to the previous paragraph, $\mathbf{C}_i \cup [a', b']$ gives better result than $\mathbf{C}' \cup [a', b']$. So, we do not have to check the control plans $\mathbf{C}_{i-n}, \dots, \mathbf{C}_{i-1}$. (If a' is in some earlier hour than a, the control $[a', b']$ can choose between control plans $\mathbf{C}_{i-n}, \dots, \mathbf{C}_{a'}$, so there is no problem.) Hence, it is not necessary to check the control plans $\mathbf{C}_{i-n}, \dots, \mathbf{C}_{i-1}$, when starting to form a new control from time point a. □

Thus, we need one wait state at the first time point of an hour, two at the second time point and finally $h^l - 1$ at the last time point of an hour (h^l is the length of an hour). In other words, we need on the average $(h^l - 1)/2$ wait states at each time point. (In [1] we used $h^l - 1$ wait states at each time point.)

3.2 Worsening states

Even though we showed in Theorem 2 that the results do not improve by increasing the number of wait states, the state space (C^t, W, C^l) does not achieve optimal result when the length of payback is non-zero (a sample case is analyzed in Section 4).

We need at least one more state variable B into the state construction (see [21], pp. 30–34) to be able to form a better path. With variable B we check the paths, which are not maximums according to (13) for the three state variable system.

A *local worsening* of stage i is a control, which stops at the stage i including the resting time and which is not chosen into the control plan by the previous equations and conditions. A three variable system chooses the best alternative among several, as shown in the left side of Figure 2. We set this to be worsening state one. In the worsening state two, we choose the second best path from the control time C^t for the first state of control time $C^t + 1$. The third worsening state uses the third best path found so far and so on.

Now our state is $S = (C^t, B, W, C^l)$. Instead of (12) we use

$$R''(\mathbf{s}, S', S, k+1) = \begin{cases} 0, & \text{when (15)–(18)}, \\ -P', & \text{when (19)}, \\ R(I([k - C^t, k], \mathbf{s})(i)) - R(\mathbf{s}), & \text{when (20) or (22)}, \\ -\infty, & \text{otherwise}, \end{cases} \tag{21}$$

When

$$\begin{aligned} &C'^t = C^t + 1, \quad W' = C'^l = 0 \text{ and} \\ &W \text{ and } C^l \text{ such that} \\ &R'\big(\mathbf{s}, (C^k, B, W, C^l), k\big) \text{ is the } B'th \text{ best} \end{aligned} \tag{22}$$

we choose the $B'th$ best path from control time C^t and set it to the first state of the $B'th$ worsening state of control time $C^t + 1$ (we allow B to vary in its range, when we are looking for the $B'th$ best path). Condition (22) can be taken into account in the implementation of dynamic forward recursion formula (13). When we are looking for the best path, we can easily cater the required amount of paths to find the $B'th$ path. Moreover, the solution (a path) given by condition (16) has also to be checked when we are looking for the number of best paths.

Starting configuration (14) and transition conditions (15)–(20) work with worsening states without major modifications. Starting solution is calculated only for the first worsening state and conditions (15)–(20) work inside a worsening state just as in the case of three variable system.

Now we have applied worsenings to a situation where we choose between different control plans. We could also apply worsenings in the control length decision (condition (20)).

4 Tests

First we made two sets of tests: one for testing the running times and the other for comparing the accuracy of results. In the first test set we had a group with no payback: its control capacity was 0.8 MW, minimum control length 30 minutes while the maximum control length varied from 100 minutes (1 hour 40 minutes) to 1 000 minutes (16 hours 40 minutes).

On the left side of Figure 3 is the clipping situation: each hour has 0.5 MW overload except the two last ones. Each hour is discretized to twelve points. Hence, with 40 minutes control (8 discretized time points) one gets 0.53 MWh cutting capacity. Clipping situation is 25 hours (a day + an extra for current hour). The right side shows the running times for old DP (with full state space) and for the new one (with half state space). DP with new state space is about twice as fast as old one, which is consistent with the theory.

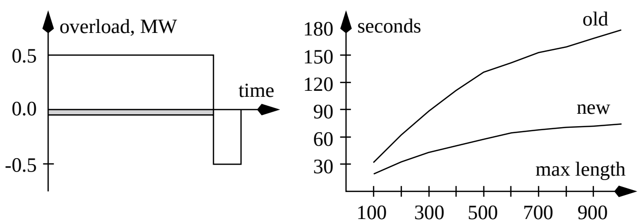

Figure 3: Running time on maximum control length and clipping situation.

Both DP's gave the same control plan. It contained twelve controls each cutting two hours at a time with 80 minutes control. After controls, there remains 0.03 MW underload in each hour (shown as grey area), except the last two. In both cases the obtained result was $-1\,590$. (In the beginning the value was $-1\,139\,400$. If all overloads were to be cut exactly, which is not possible with this group and time discretization, the value would have been -900.) Only the running times differed.

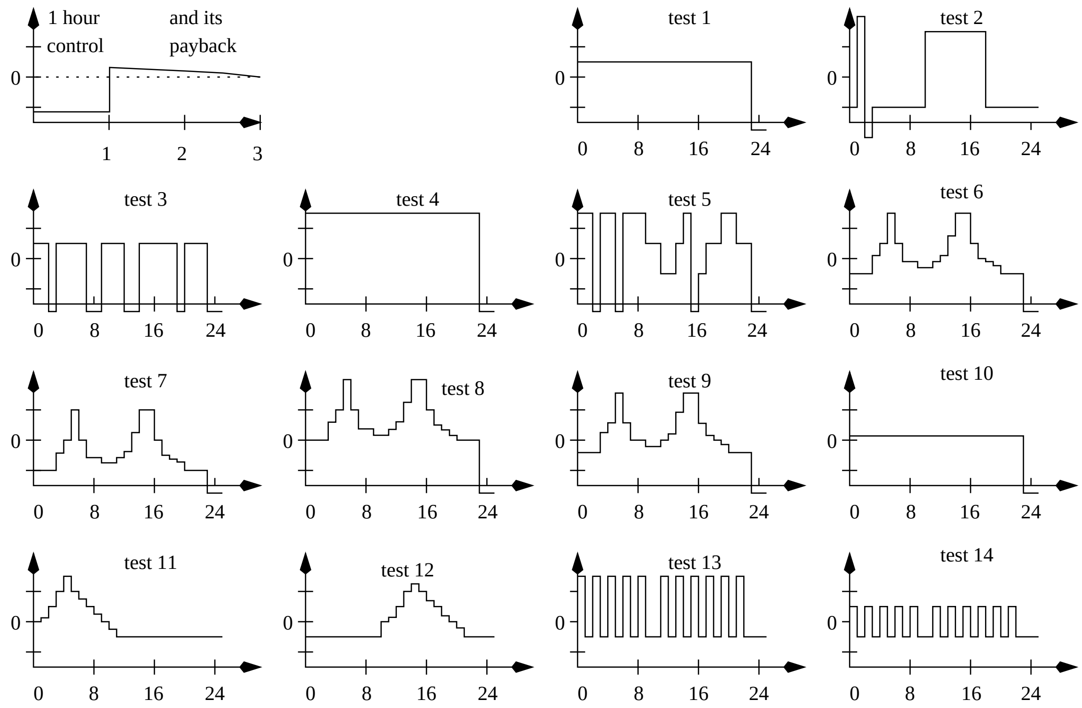

Figure 4: Payback used in the tests and the test cases.

In the next test series we wanted to see whether the paybacks affect the solutions given by the new DP. As we will see, the paybacks affect the results of computations. Because of this, we tried to improve accuracy by increasing the number of wait states.

Intuitively, the wait states starting from point b can only "look up" that far later on at the moment a. So if b is the beginning of an hour, we "don't see" into the previous hour at moment a and cannot affect decision made before, i.e. change the choice between different control plans. When there is no payback, the number of wait states equaling to the number of time points after a start of an hour $a \mod h^l$ is enough, because a control finished in the previous hour does not affect the present hour to be cut. (This does not mean that the system is optimal. All we know is that there is no need to increase the number of wait states.)

However, when we are using payback, we need to be able to look further into the previous hours in order to increase the accuracy. As the number of wait states is $a \mod b$ where b typically is the length of one hour ($b = h^l$) and a is the current time point, we can increase the number of states in two ways. On one hand, we can increase b or, on the other hand, we can directly increase the number of wait states to $c + (a \mod b)$.

We tested both ways and noticed that increasing b does not improve the solutions much. Modulus $a \mod b$ forms a cyclical group and the cycles divide the time line into disjoint intervals of length b. We cannot "see" into the previous interval and basically, our problem remains. The time line is still divided into disjoint areas and the optimum is easily lost, because paybacks can arbitrarily have affect on the next interval. By adding c states we improve the ability to see to earlier hours (or into earlier "cycle intervals"). We are tempted to think that increasing c will improve the solution. Our tests, however, show that while this is mostly true, there are exceptions.

In the tests we used payback shown in the upper left corner in Figure 4. We tested the group with 14 different clipping situations, of which 10 were quite artificial while 4 (tests 6–9) could be normal clipping situations occurring in reality. Tests 6–9 are similarly shaped and contain "morning and afternoon" consumption peaks. The shapes are at different levels giving tests of different difficulty. Test 11 contains only morning peak while test 12 contains afternoon peak. Other tests are artificial. When load curve is above 0, we have overload that should be cut off. One tick stands for 1 MW. (Horizontal axis are for time.)

Again, each hour is discretized into twelve time points. We used a group having 1.2 MW control capacity (i.e. 0.1 for five minutes). For example, in the first test we could cut the overload from each overload hour exactly with the presented group with 25 minutes control (five discretized moments), if there were no payback. Payback, however, affects the underload hours (improving the result at the same time) as well as the next hour to be cut (increasing the amount to be cut), because the payback is two hours long. The minimum control length is 30 minutes and the maximum control length is one hour. One MW overload costs $-99\,000$ and underload -900. Resting time is 10 minutes. Starting values (total losses without controls) are given in Table 2.

Figure 5 contains running times (in seconds) for the old DP solution and for DP's with $c = 0, 1, 2, 3, 5$ and 11 (horizontal axis). Moreover, $b = 12 = h^l$ and a is the moment between hours (or the time point). We see that the running times increase almost linearly on the number of waits states.

Table 3 contains results for the tests used in Figure 5. If a solution is presented only for the old DP, other DP's with different values for c achieved the same result.

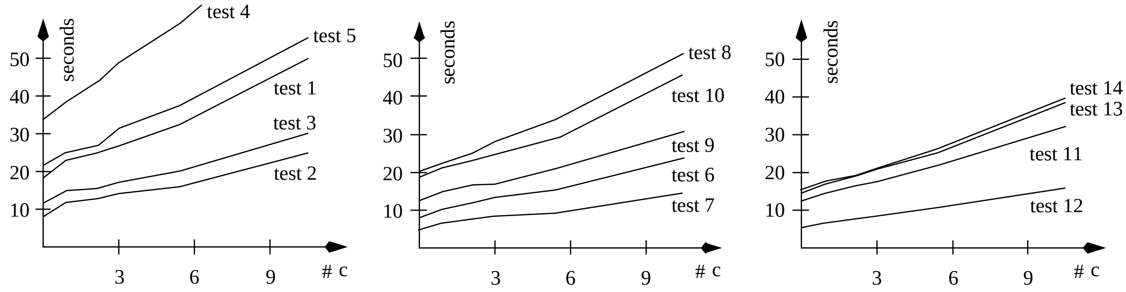

Figure 5: Running times.

Here we see that, in general, the results improve if the number of wait states is increased.

In the seventh test, however, we see that increasing the number of wait states has decreased the result between DP's with $c = 3$ and $c = 5$. This somewhat non-intuitive result follows from the fact that our DP solutions do not fulfill the optimality principle usually stated for dynamic programming solutions, because of paybacks (see [21] p. 16).

The optimality principle is lost because we cannot guarantee that optimal solution at the stage i (time point i) entails optimal solution for the rest of the time line. Reason for this is the payback: it can affect later hours and decisions made later. This information should be available at the moment when we are deciding the length of a control. Similarly, if we can first find the best control, we cannot be sure that the second control — even if optimal after the first one — gives optimal solution for the whole optimization problem.

In the seventh test, DP with $c = 5$ finds in one crucial time point better solution than DP with $c = 3$. It turns out that locally better solution is worse for the rest of the optimization in this case. Larger number of additional states handle the situation correctly.

We did not use any worsening states in the test series reported in Table 3 and Figure 5. We run the same test series using 2, 5, 10 and 15 worsening states. Table 4 contains the best solutions found among all the test series. Results are remarkable: results were improved in the most cases. Moreover, improvements were relatively high (much over 10% in some tests) and even the old DP solution was improved in some cases.

Test 14 demonstrates informatively how results improve as the number of wait states or worsening states (or both) increases. Test 14 has 0.5 MW overload in every other hour and the rest have the same amount of underload (there are two underload hours, 9 to 11, "changing overload phases", see Figure 4). Results and running times of the test 14 are shown in Figures 6 and 7.

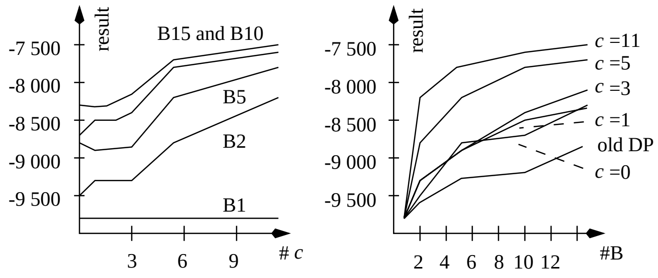

Figure 6: Wait and worsening states, results for test 14.

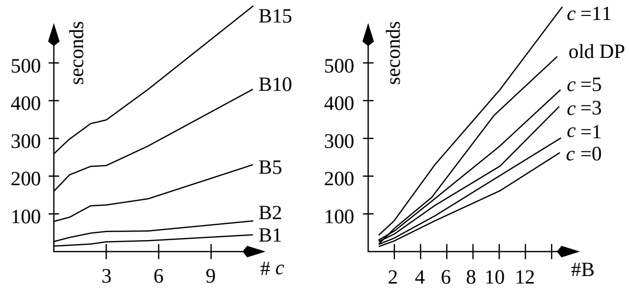

Figure 7: Wait and worsening states, running times for test 14.

Starting solution (no control plan) gives −550 800. DP with one and two worsenings build up similar control plans till the stage 81 (i.e. 6 hours 45 minutes after the beginning). The best result found so far gives −355 171 with controls [0,6], [18,29], [42,53] and [66,77]. At the stage 82, DP with two worsening states found a plan giving −355 089.2 with controls [0,5], [8,13], [22,31], [46,57] and [66,77]. By choosing the second best control plan ([0,5], [8,13], [22,31] and [46,57] with −405 807.7) at the stage 66 we have found better control plan than by using the locally best alternative ([0,6], [18,29], [42,53] with −404 157.9).

The second best plan at stage 66 has cut overload more accurately (there is not so much underload than with the best plan). It also incurs more payback into the seventh hour so that the result is not the best (overload costs much more than underload). This increase caused by the payback is less than the amount of control being one moment longer, which, in turn, causes also the seventh hour to be cut more precisely with the second best path than with the best. By increasing the number of worsenings to 15, the first different stage is 56. Similarly to the previous case, the old plan at phase 42 ([0,6] and [18,29]) is locally better than [4,9], [20,30] (at least $15th$ best) but after control [42,53] at the stage 56 a worser plan at the stage 42 gives better result than the best plan.

We also tried DP's with 30 and 100 worsening states. System with 30 states improves the result first time from 15 state system at stage 80 and system with 100 states improves the result first time from 30 state system at stage 61.

Table 2: Starting values of the tests 1–14.

Test 1	Test 2	Test 3	Test 4	Test 5
−1 141 200	−1 402 200	−852 300	−3 418 200	−1 839 600
Test 6	**Test 7**	**Test 8**	**Test 9**	**Test 10**
−699 480	−341 100	−1 499 400	−797 310	−228 600
Test 11	**Test 12**	**Test 13**	**Test 14**	
−550 980	−550 980	−1 639 800	−550 800	

DP with 15 worsening states gives $-8\,336$, with 30 states $-7\,872$ and with 100 states $-7\,214$, which is better than the solution given by 15 worsenings and 11 additional wait states (see Table 4). Our conclusion is that a clipping situation with many overload intervals most likely benefits from the use of worsening states.

We also studied how worsening states and wait states improve the results together and how they affect the running times. The left hand side of Figure 6 contains results for different worsening state amounts (1, 2, 5, 10 and 15). As the number of wait states is increased, the results improve in general. There are few exceptions, however. We see that few wait states may do worse than DP with $c = 0$ (see lines for B5 and B15). Most of the time 11 additional states to wait states gives better results than less wait states. By using only one worsening state (corresponds to a system, where no worsening state usage is implemented), the number of used additional wait states is irrelevant.

On the right side the same data is plotted for five different additional wait state amounts as well as for the old DP system. We conclude that the number of worsening states is much more crucial for the results than the number of wait states. Both state variables are needed, though. Worsening states improve also the results of DP system with fixed amount of wait states (old DP, $c = h^l - 1$, no $a \mod b$ part).

We didn't try to find the best combination for the number of worsening and wait states as we wanted to keep the running times tolerable for the test runs. In practise, we need the results in five minutes after the hourly load forecast is obtained. One hour is discretized into five minute intervals and if we are going to fully cut the first

Table 3: Solutions without worsening states (empty means the solution given by the DP old).

Test	old DP	$c = 0$	$c = 1$	$c = 3$	$c = 5$	$c = 11$
1.	−3 091					
2.	−532 670					
3.	−11 288					
4.	−1 323 150					
5.	−483 773	−484 248	−484 248	−483 773	−483 773	−483 773
6.	−144 478					
7.	−15 603	−15 662	−15 662	**−15 662**	**−15 778**	−15 603
8.	−368 665	−371 296	−370 238	−368 665	−368 665	−368 665
9.	−174 490	−175 668	−175 668	−175 668	−175 668	−174 490
10.	−8 874	−9 131	−9 131	−8 990	−8 874	−8 874
11.	−68 059	−70 494	−69 657	−68 059	−68 059	−68 059
12.	−68 059	−70 494	−69 657	−68 059	−68 059	−68 059
13.	−366 669					
14.	−9 860					

Table 4: The best solutions with worsening states (empty means the solution given by the DP old).

Test	old DP	$c = 0$	$c = 1$	$c = 3$	$c = 5$	$c = 11$
1.	−3 091	−3 091	−3 091	**−2 975**	**−2 975**	**−2 975**
2.	**−532 670**					
3.	−10 823	−10 823	−10 823	−10 823	−10 823	**−10 707**
4.	**−1 323 150**					
5.	−483 773	−483 773	−483 773	**−483 705**	**−483 705**	**−483 705**
6.	**−144 478**					
7.	−15 603	−15 662	−15 662	**−15 487**	**−15 487**	**−15 487**
8.	−368 373	−369 069	**−368 046**	−368 162	−368 278	−368 373
9.	−174 490	−175 668	−175 543	**−174 398**	**−174 398**	−174 490
10.	−8 410	**−7 365**	**−7 365**	−7 713	−8 202	−8 177
11.	**−67 827**	−69 242	−68 613	**−67 827**	**−67 827**	**−67 827**
12.	**−67 827**	−68 778	−68 613	**−67 827**	**−67 827**	**−67 827**
13.	**−366 669**					
14.	−8 956	−8 336	−8 351	−8 186	−7 742	**−7 538**

hour, the control plan has to be ready in less than five minutes. Figure 7 contains running times for test 14. Execution decelerates almost linearly as the number of worsening states is increased. The same holds also for the number of wait states.

The number of wait states in one hour is $\sum_{a=0}^{h^l-1}\left(c + (a \mod h^l)\right)$ (where $c = 0, \ldots, h^l - 1$ and a is the moment). Hence, the increase of c by one gives $h^l - 1$ additional wait states for one hour. This is proportionally less than the increase brought by the increase of the number of the worsening states by one, which is the number of used wait states in one hour. This explains why the increase of the number of worsening states decelerates more the running times than the increase of the number of wait states.

5 Conclusions

In this work we have shown a non-straightforward solution for a dynamic programming problem arising in direct load control application. The properties of the state space have been analyzed, and quicker optimization algorithms are formed without sacrificing the accuracy of the results when payback is not used. Moreover, we have found practical ways to improve the results by increasing the state space when payback is used.

We have shown a detailed solutions for three and four state variables. Our solution is sub-optimal. If the result accuracy is not crucial, one can drop wait states away, arriving to a faster two state variable solution of [4].

There are still open problems concerning the properties of state variable C^t. They seem to behave "softly enough" [1], so that we can reduce the number of states used (for details, see [1]). Moreover, the control length C^l may have some properties, by which we can further speed up the algorithms.

If there is enough time to calculate results, it is possible to add a new state variable, called worsening state. With four state variables we achieve better results, as is shown in our extensive tests. Additional wait states as well as additional worsening states improve the results in general. Hence, one can choose between fast inaccurate, and accurate but slow solutions. Similar trade can be made between two, three and four variable state spaces the two variable system being the fastest but also the less accurate.

Most important, worsening states seem to improve the results also in the cases occurring in production systems.

Acknowledgment

Author is grateful to Erkki Mäkinen for his time, patience and comments.

References

[1] Isto Aho, Harri Klapuri, Jukka Saarinen, and Erkki Mäkinen, "Optimal load clipping with time of use rates," *International Journal of Electrical Power & Energy Systems*, vol. 20, no. 4, pp. 269–280, 1998.

[2] Isto Aho, "Interactive knapsacks," To appear in *Fundamenta Informaticae.*

[3] Michael R. Garey and David S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-completeness*, W. H. Freeman, 1979.

[4] Arthur I. Cohen and Connie C. Wang, "An optimization method for load management scheduling," *IEEE Transactions on Power Systems*, vol. 3, no. 2, pp. 612–618, May 1988.

[5] Wen-Chen Chu, Bin-Kwie Chen, and Chun-Kuei Fu, "Scheduling of direct load control to minimize load reduction for a utility suffering from generation shortage," *IEEE Transactions on Power Systems*, vol. 8, no. 4, pp. 1525–1530, Nov. 1993.

[6] Deh-chang Wei and Nanming Chen, "Air conditioner direct load control by multi-pass dynamic programming," *IEEE Transactions on Power Systems*, vol. 10, no. 1, pp. 307–313, Feb. 1995.

[7] C.N. Kurucz, D. Brandt, and S. Sim, "A linear programming model for reducing system peak through customer load control programs," *IEEE Transactions on Power Systems*, vol. 11, no. 4, pp. 1817–1824, Nov. 1996.

[8] Kah-Hoe Ng and Gerald B. Sheblé, "Direct load control — a profit-based load management using linear programming," *IEEE Transactions on Power Systems*, vol. 13, no. 2, pp. 688–695, May 1998.

[9] R. Bhatnagar, J. Latimer, L. J. Hamant, A. A. Garcia, G. Gregg, and E. Chan, "On-line load control dispatch at Florida Power & Light," *IEEE Transactions on Power Systems*, vol. 3, no. 3, pp. 1237–1243, Aug. 1988.

[10] Jean-Charles Laurent, Guy Desaulniers, Roland P. Malhamé, and François Soumis, "A column generation method for optimal load management via control of electric water heaters," *IEEE Transactions on Power Systems*, vol. 10, no. 3, pp. 1389–1400, Aug. 1995.

[11] R. Bhatnagar and S. Rahman, "Dispatch of direct load control for fuel cost minimization," *IEEE Transactions on Power Systems*, vol. PWRS-1, no. 4, pp. 96–102, Nov. 1986.

[12] Yuan-Yih Hsu and Chung-Ching Su, "Dispatch of direct load control using dynamic programming," *IEEE Transactions on Power Systems*, vol. 6, no. 3, pp. 1056–1061, Aug. 1991.

[13] K. Bhattacharyya and M. L. Crow, "A fuzzy logic based approach to direct load control," *IEEE Transactions on Power Systems*, vol. 6, no. 3, pp. 708–714, May 1996.

[14] Douglas W. Caves and Joseph A. Herriges, "Optimal dispatch of interruptible and curtailable service options," *Operations Research*, vol. 40, no. 1, pp. 104–112, January–February 1992.

[15] Jianming Chen, Fred N. Lee, Arthur M. Breipohl, and Rambabu Adapa, "Scheduling direct load control to minimize system operational cost," *IEEE Transactions on Power Systems*, vol. 10, no. 4, pp. 1994–2001, Nov. 1995.

[16] A. J. Gaul, E. Handschin, W. Hoffmann, and C. Lehmköster, "Establishing a rule base for a hybrid es/xps approach to load management," *IEEE Transactions on Power Systems*, vol. 13, no. 1, pp. 86–93, Feb. 1998.

[17] Houzhong Yan and Peter B. Luh, "A fuzzy optimization-based method for integrated power system scheduling and inter-utility power transaction with uncertainties," *IEEE Transactions on Power Systems*, vol. 12, no. 2, pp. 756–763, May 1997.

[18] Allen J. Wood and Bruce F. Wollenberg, *Power Generation, Operation and Control*, John Wiley & Sons, New York, 1984.

[19] Paresh Rupanagunta, Martin L. Baughman, and Jerold W. Jones, "Scheduling of cool storage using non-linear programming techniques," *IEEE Transactions on Power Systems*, vol. 10, no. 3, pp. 1279–1285, Aug. 1995.

[20] B. Rautenbach and I. E. Lane, "The multi-objective controller: a novel approach to domestic hot water load control," *IEEE Transactions on Power Systems*, vol. 11, no. 4, pp. 1832–1837, Nov. 1996.

[21] Dimitri P. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, 1996.