Energy and Buildings 82 (2014) 57–64

Contents lists available at ScienceDirect

Energy and Buildings

journal homepage: www.elsevier.com/locate/enbuild

Application of dynamic programming to study load shifting in buildings

Bérenger Favre*, Bruno Peuportier

MINES ParisTech – CES (Center for Energy efficiency of Systems), 60, Boulevard Saint Michel, 75272 Paris Cedex 06, France

ARTICLE INFO

Article history:
Received 16 September 2013
Received in revised form 2 July 2014
Accepted 8 July 2014
Available online 15 July 2014

Keywords:
Dynamic programming
Load shifting
Carbon emissions
Building
Insulation
Thermal mass

ABSTRACT

Being increasingly insulated, new buildings are more and more sensitive to variations of solar and internal gains. Due to an important use of electrical heating systems, especially in housing, France is facing a growing problem of peak load on its electricity grid. Controlling the heating system often constitutes an efficient solution to shift heating loads while maintaining indoor comfort. The proposed energy management is a predictive set of optimal commands issued from a dynamic programming optimization knowing in advance the weather, occupancy and internal gains for the next 7 days. This method is tested on a low energy house situated in France with an annual heating demand of 14 kWh/m^2. In this paper, load shifting according to utility rate incentives and carbon emissions are studied. The importance of building models as well as envelope insulation and thermal mass on energy management results is shown.

1. Introduction

The main objectives for control systems in buildings are to save energy and increase comfort. During a summer period, control systems are used to reduce the energy consumption of air conditioning or to maintain comfort using passive cooling. Previous studies concerned the control of shutters [1–3], ventilation [4,5], and active cooling [6,7]. Another objective is addressed here: to reduce environmental impacts and costs thanks to an improved integration in a larger scale electrical system, according to a smart grid approach. During a cold winter period, control systems can be used to decrease the energy consumption of the heating system [8] or to reduce peak demand [9] for example. Peak electrical demand is a serious problem for the stability of the electrical network [10]. Load shifting is an efficient method to face this problem, it allows moving electrical loads from peak to off-peak times. One simple way to shift the load is to store energy during off-peak time in order to decrease energy demand during peak time. There are several methods to store energy. The first is to use the thermal mass of the building envelope. The higher the thermal mass, the more energy can be stored if the insulation is adequate, which corresponds to a passive storage [11,12]. It is also possible to use Phase Change Material to store more energy in the thermal mass of the building [13]. An active storage like a battery is a third way to store energy [14]. The storage and discharge of heat at the right time benefits from a predictive controller able to anticipate the variation of ambient temperature, solar irradiance and internal loads. Many advanced control systems are reviewed in Ref. [15]. Some of them are Model Predictive Controllers (MPC). An optimization is done at each time step (online), the results is an optimal sequence of controls for several time steps on a finite horizon. Only the first control is applied, a new optimization is done for the next time step. This is a relevant method when the system response is greater than the duration of a full optimization because a new optimization is possible at each time step. Linear or quadratic programming [16,17], Shell Multivariable optimizing Control [18] and Generalized Predictive Control [19] are some examples of MPC. For predictive controllers, a thermal model of the building is required [20–22]. Due to the time step of this model, a combinatorial optimization is well suited. Among these methods, the A* [23], and the Branch and Bound algorithms [24], need an assumption of the lower or upper bound which is not available in the case of heating control problems. In this case, using these optimization algorithms would be a lot more time consuming than using dynamic programming. Dynamic programming is also chosen because of its exact optimization character. It has served in a building context mainly for winter operation of the heating system [21,25] but not for load shifting. In this publication, a dynamic programming optimization is used to set up a predictive controller knowing in advance ambient temperature, solar gains and internal loads. The objective is to shift the

* Corresponding author. Tel.: +33 687610666.
E-mail addresses: berenger.favre@mines-paristech.fr, berenger.favre@gmail.com (B. Favre), bruno.peuportier@mines-paristech.fr (B. Peuportier).

http://dx.doi.org/10.1016/j.enbuild.2014.07.018
0378-7788/

heating load in the building, and dynamic programming is tested to control the heating system of a house during a cold week. The load shifting is done according to utility rate incentives in a first step then according to carbon emissions. The controller depending on the building model, the influence of number of thermal zones is studied. The optimal set of commands issued from optimizations with different building envelope insulation levels and thermal masses are then compared.

2. Models

In this paper, a predictive controller using dynamic programming is studied to shift heating loads from peak periods to off-peak time. The prediction is based upon thermal simulation. The corresponding model is presented hereunder, along with the cost functions.

2.1. Thermal model of the building

The building is modeled as zones of homogenous temperature. In each zone, each wall is divided in meshes small enough to also have a homogeneous temperature. There is one more mesh for the air and furniture of the zone. A thermal balance is done on each mesh within the building [26]:

$$C_{mesh}T'_{mesh} = \text{Gains} - \text{Losses} \quad (1)$$

C_{mesh} being the thermal capacity of the mesh, T_{mesh} its temperature, Gains and Losses including heat transfer by conduction, radiation and convection but also possible internal heating and cooling from equipment and/or appliances, occupants and solar gains.

For each zone, repeating Eq. (1) for each mesh and adding an output equation leads to the following continuous linear time-invariant system, non linear terms being included in the driving forces (i.e. ventilation losses) [26]:

$$CT'(t) = AT(t) + EU(t) \quad (2)$$

$$Y(t) = JT(t) + GU(t)$$

where T is the mesh temperature vector; U is the driving forces vector (climate parameters, heating, etc.); Y is an outputs vector (indoor temperatures accounting for air and wall surfaces); C is the diagonal thermal capacity matrix and A, E, J, G matrices relating temperatures and driving forces vectors.

In order to perform simulation, it is important to know the occupancy of the building, which defines the emission of heat by inhabitants and appliances, the thermostat set point influencing the heating/cooling equipment, and possible actions regarding ventilation and shutters. Another important aspect is the weather model, influencing heat losses and solar gains. All the data of the occupancy and weather models are contained in the driving forces vector U.

A high order linear model is thus constituted. But its state dimension is too large to allow a fast convergence of an optimization algorithm. A modal reduction is then applied in order to lower the state dimension and to make the algorithm faster [27].

2.2. Optimization algorithm

The dynamic programming algorithm is a sequential optimization method which provides an optimal set of commands over a period. A control vector u is defined with N_c dimensions:

$$u(t) = u_t \in U_t, \quad U_t \subset R^{N_c} \quad (3)$$

where U_t is the set of possible controls. A state variable describing the system evolution is used and discretized temporally:

$$\mathrm{x}(t) = \mathrm{x}_t \in X_t, \quad X_t \subset R^{N_e} \quad (4)$$

where X_t is the set of possible states, N_e is the dimension of X_t. The state equation at each time step t relates the state variable at the following time step to the variables at t using a function f:

$$x(t) = xt, \quad x(t+1) = f(x(t), u(t), t) \quad (5)$$

A value function v_t is now defined which is the cost to progress from $x(t)$ to $x(t+1)$:

$$v_t(x_t, x_{t+1}), \quad x_{t+1} \in T_t(x_t) \quad (6)$$

T_t being the set of possible state variables at time t. The cost function is then the sum of all value functions at each time step:

$$V_0^t = \sum_{j=0}^{t-1} V_j(x_j, x_{j+1}) \quad (7)$$

The optimization seeks to maximize the following objective function over N time steps corresponding to the period from 0 to t:

$$J = \max[V_0^N] \quad (8)$$

This equation gives us a set of controls to go from x_0 to x_t. Bellman's principle of optimality is applied to accelerate this optimization by breaking this decision problem into smaller subproblems [28]:

> "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."

Eq. (8) becomes then:

$$J = \text{Max}[V_0(x_0, x_1) + \text{Max}[V_1^N]] \quad (9)$$

So dynamic programming operates as shown in the following figure (Fig. 1):

To summarize, a set of commands $U_N = (u_0, u_1, \ldots, u_N)$ maximizing (9) for a system described by (5) with constraints on state variables (4) and on controls (3) is the result of the optimization. For the application of dynamic programming in a building context, the chosen state variable is described in the section 2.3.

2.3. Cost functions

Two value functions (see Eq. (6)) are used in this paper. The first value function emphasizes the utility rate incentives while the second emphasizes the carbon emissions due to the electric consumption of the heating system. The first one is used when the load shifting is done according to utility rate incentives. A higher price

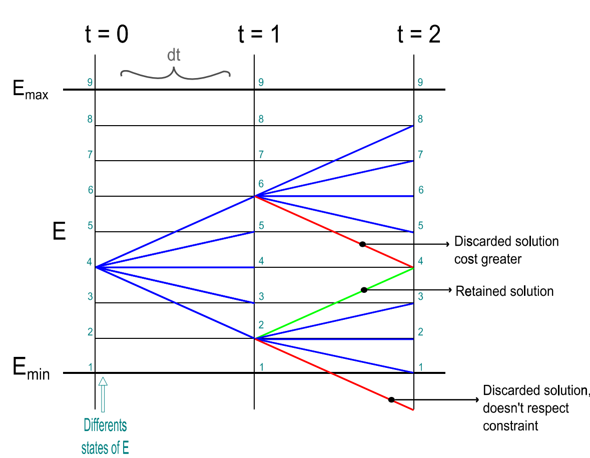

Fig. 1. Dynamic programming description.