Parallel Asynchronous Algorithms for Optimal Control of Large Scale Dynamic Systems

S. S. Abdelwahed
Department of Electrical and Computer Engineering, University of Toronto
Toronto, Ontario, Canada M5S 1A4

M. F. Hassan
Professor, Electronics and Communication Department, Faculty of Engineering
Cairo University, Giza, Egypt

M. A. Sultan
Associate Professor, Electronics and Communication Department, Faculty of Engineering
Cairo University, Giza, Egypt

Abstract

This work presents two parallel asynchronous algorithms for the solution of the optimal control problem of linear large scale dynamic systems. These algorithms are based on the prediction concept. The first one adopts the interaction prediction approach and the second is based upon the costate prediction approach. The convergence behavior of the proposed algorithms is thoroughly investigated. The new algorithms are applied on three practical systems and simulation results are presented and compared with those obtained using the well known synchronous algorithms. It is shown that substantial saving in computation time can be achieved by employing the proposed asynchronous algorithms.

List of Figures

1 Introduction

The information processing and requirements for experimenting with large scale systems are usually excessive and many problems may arise when dealing with such systems. Conventional control methods usually fail when applied to such systems. Therefore, new multilevel techniques were established to handle these problems.

We consider two issues while evaluating these techniques, namely the computational time and the storage requirement. It has been shown that multilevel techniques need, in general, a much smaller storage requirement than conventional methods.[1] As for the computation time, practical implementation of these methods have shown that they need less computational time when compared with conventional methods. Another important factor is that the computational structure of these techniques can effectively utilize the existing parallel and distributed computing facilities, from which a great reduction of computational time can be obtained.

The general form of the multilevel algorithms is characterized by their synchronous nature where the iteration time is equal for all subproblems irrespective of their dimensions, with no interaction between the subsystems within the iteration period. However, many practical systems are composed of a number of subsystems with different structures and dimensions. In such situation the asynchronous implementation of multilevel parallel algorithms may result in a better convergence.[2,3]

With a variety of applications, a number of researches have been focused to develop and investigate asynchronous implementation of parallel algorithms.[4–6] In some earlier work,[7,8] a partial asynchronous implementation of the interaction and costate prediction algorithms have been investigated. In these algorithms, information updating takes place within the iteration, however, all processors have to synchronize for a new iteration. It was shown, both theoretically and practically, that a significant reduction of computational time can be achieved using the partially asynchronous algorithms.

In order to enhance the utilization of the computational facilities, we propose a totally asynchronous - or simply asynchronous - algorithm applied to both interaction and costate prediction methods. In this approach, processors perform certain computations and then exchange their informations either by direct communications with each other or by means of a coordinator processor. Each processor performs its steps independent to other processors, that is a processor can proceed with its next cycle without waiting for other processors to finish their iteration. This may result in some processors performing computation faster than others. Consequently, this will allow faster updating of systems components, and hence, it is excepted that the asynchronous method will improve the convergence rate over the synchronous one. However, the communication delay factor will be important since excessive information exchange may take place between the subsystems.

This paper is organized as follows. Section 2 presents the problem formulation and an outline the synchronous algorithm. In section 3, asynchronous algorithms, for both interaction and costate prediction methods, are presented to solve the optimal control problem of linear interconnected dynamical systems. The convergence behavior of the algorithms is analyzed in section 4. The convergence was proven based on the results of El-Tarazi.[9] Section 5 discusses the practical implementation of the algorithms and the effect of communication delay. In section 6, the results of simulation of the proposed algorithms are presented to emphasize the results of the theoretical

investigation. Section 7 concludes the paper.

2 Problem Formulation

Consider a linear large scale time invariant system

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \boldsymbol{x}(t_o) = \boldsymbol{x}_o \tag{1}$$

where $\boldsymbol{x} \in \mathcal{R}^n$ and $\boldsymbol{u} \in \mathcal{R}^m$. It is assumed that the system can be decomposed into N interconnected subsystems given by

$$\dot{\boldsymbol{x}}_i(t) = \boldsymbol{A}_i\boldsymbol{x}_i(t) + \boldsymbol{B}_i\boldsymbol{u}_i(t) + \boldsymbol{C}_i\boldsymbol{z}_i(t), \quad \boldsymbol{x}_i(t_o) = \boldsymbol{x}_{io}, \quad i = 1, \ldots, N \tag{2}$$

The interaction vector $\boldsymbol{z}_i$ is a linear combination of the states of the other $N-1$ subsystems given by

$$\boldsymbol{z}_i(t) = \sum_{j=1}^{N} \boldsymbol{L}_{ij}\boldsymbol{x}_j(t) \tag{3}$$

where $\boldsymbol{x}_i \in \mathcal{R}^{n_i}$ is the state vector, $\boldsymbol{u}_i \in \mathcal{R}^{m_i}$ is the control vector, $\boldsymbol{z}_i \in \mathcal{R}^{q_i}$ is the interconnection vector, $\boldsymbol{A}_i, \boldsymbol{B}_i, \boldsymbol{C}_i$ are the ith subsystem matrices with proper dimensions, and $\boldsymbol{L}_{ij}$ is the interconnection matrix.

The original system optimal control problem is reduced to the optimization of N subsystems, satisfying (2) and (3) while minimizing the cost function

$$J_i = \int_{t_o}^{t_f} \boldsymbol{x}_i^T(t)\boldsymbol{Q}_i\boldsymbol{x}_i(t) + \boldsymbol{u}_i^T(t)\boldsymbol{R}_i\boldsymbol{u}_i(t)dt \tag{4}$$

where $\boldsymbol{Q}_i$ is $n_i \times n_i$ positive semidefinite matrix and $\boldsymbol{R}_i$ is $m_i \times m_i$ positive definite matrix.

As known from the original prediction concept, this problem can be solved by first introducing a set of Lagrange multipliers $\boldsymbol{\pi}_i(t)$ and costate vectors $\boldsymbol{\lambda}_i(t)$ to augment the *interaction* equality constraint (3) and the subsystem dynamic constraint (2) to the cost function and defining the Hamiltonian of the ith subsystem as follows:

$$\begin{aligned} H_i \;=\; & \frac{1}{2}\left(\boldsymbol{x}_i^T(t)\boldsymbol{Q}_i\boldsymbol{x}_i(t) + \boldsymbol{u}_i^T(t)\boldsymbol{R}_i\boldsymbol{u}_i(t)\right) + \boldsymbol{\pi}_i^T(t)\left(-\boldsymbol{z}_i(t) + \sum_{\substack{j=1 \\ j\neq i}}^{N} \boldsymbol{L}_{ij}\boldsymbol{x}_i(t)\right) + \\ & \boldsymbol{\lambda}_i^T\left(\boldsymbol{A}_i\boldsymbol{x}_i(t) + \boldsymbol{B}_i\boldsymbol{u}_i(t) + \boldsymbol{C}_i\boldsymbol{z}_i(t)\right) \end{aligned} \tag{5}$$

then the following set of necessary conditions of optimality can be obtained

$$\frac{\partial H_i}{\partial \boldsymbol{u}_i} = \boldsymbol{R}_i \boldsymbol{u}_i(t) + \boldsymbol{B}_i^T \boldsymbol{\lambda}_i^T(t) = 0 \tag{6}$$

$$\frac{\partial H_i}{\partial \boldsymbol{\lambda}_i} = \boldsymbol{A}_i \boldsymbol{x}_i(t) + \boldsymbol{B}_i \boldsymbol{u}_i(t) + \boldsymbol{C}_i \boldsymbol{z}_i(t) = \dot{\boldsymbol{x}}(t) \tag{7}$$

$$\frac{\partial H_i}{\partial \boldsymbol{x}_i} = \boldsymbol{Q}_i \boldsymbol{x}_i(t) + \boldsymbol{A}_i^T \boldsymbol{\lambda}_i(t) + \sum_{j=1}^{N} \boldsymbol{L}_{ij}^T \boldsymbol{\pi}_i(t) = -\dot{\boldsymbol{\lambda}}_i(t) \tag{8}$$

$$\frac{\partial H_i}{\partial \boldsymbol{z}_i} = \boldsymbol{\pi}_i(t) - \boldsymbol{C}_i^T \boldsymbol{\lambda}_i(t) = 0 \tag{9}$$

$$\frac{\partial H_i}{\partial \boldsymbol{\pi}_i} = \boldsymbol{z}_i(t) - \sum_{j=1}^{N} \boldsymbol{L}_{ij} \boldsymbol{x}_j(t) \tag{10}$$

Substituting $\boldsymbol{u}_i$ from equation (6) into (7) and $\boldsymbol{\pi}_i$ from (9) into (8) we get

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}_i \boldsymbol{x}_i(t) + \boldsymbol{B}_i \boldsymbol{R}_i^{-1} \boldsymbol{B}_i^T \boldsymbol{\lambda}_i + \boldsymbol{C}_i \boldsymbol{z}_i \quad \boldsymbol{x}_i(t_o) = 0 \tag{11}$$

$$\dot{\boldsymbol{\lambda}}(t) = -\boldsymbol{Q}_i \boldsymbol{x}_i(t) - \boldsymbol{A}_i \boldsymbol{\lambda}_i(t) - \sum_{j=1}^{N} \boldsymbol{L}_{ij}^T \boldsymbol{C}_j \boldsymbol{\lambda}_j(t) \quad \boldsymbol{\lambda}_i(t_f) = 0 \tag{12}$$

The above problem can be solved in hierarchical structure by introducing a set of coordination variables. The lower level solves the local optimization problems given by (6)-(10) keeping the coordination variables fixed, and the upper level uses the optimal trajectories received from the lower level to update the coordination vectors then sends them back to the lower level.

According to the coordination parameters we will consider two approaches in this paper, first, the interaction prediction where the coordination vector is given by $\begin{bmatrix} \boldsymbol{z}_i^T & \boldsymbol{\pi}_i^T \end{bmatrix}^T$, and second, the costate prediction approach in which $\begin{bmatrix} \boldsymbol{z}_i^T & \boldsymbol{\lambda}_i^T \end{bmatrix}^T$ is the coordination vector.

2.1 The Interaction Prediction Approach

Consider the lower level and let

$$\boldsymbol{\lambda}_i(t) = \boldsymbol{P}_i \boldsymbol{x}_i(t) + \boldsymbol{s}_i \tag{13}$$

Substituting for $\boldsymbol{\lambda}_i(t)$ in (12) and (9) we get

$$\boldsymbol{P}_i + \boldsymbol{A}^T \boldsymbol{P}_i + \boldsymbol{P}_i \boldsymbol{A}_i - \boldsymbol{P}_i \boldsymbol{B}_i \boldsymbol{R}_i^{-1} \boldsymbol{B}_i^T \boldsymbol{P}_i + \boldsymbol{Q}_i = 0 \quad \boldsymbol{P}_i(t_f) = [0] \tag{14}$$

$$\boldsymbol{s}_i + \boldsymbol{A}_i^T \boldsymbol{s}_i - \boldsymbol{P}_i \boldsymbol{B}_i \boldsymbol{R}_i^{-1} \boldsymbol{B}_i^T \boldsymbol{s}_i + \boldsymbol{P}_i \boldsymbol{C}_i \boldsymbol{z}_i - \sum_{j=1}^{N} \boldsymbol{L}_{ij}^T \boldsymbol{\pi}_j = 0 \quad \boldsymbol{s}_i(t_f) = 0 \tag{15}$$

As a result, the local control $\boldsymbol{u}_i$ is given by

$$\boldsymbol{u}_i(t) = -\boldsymbol{R}_i^{-1}\boldsymbol{B}_i^T\boldsymbol{P}_i\boldsymbol{x}_i(t) - \boldsymbol{R}_i^{-1}\boldsymbol{B}_i\boldsymbol{s}_i \tag{16}$$

The well-known synchronous algorithm is given as follows:[10]

step 1 Solve the N independent differential Matrix Riccati equations (15) and store the values of $\boldsymbol{P}_i(t)$.

step 2 The upper level assigns the iteration number $k = 1$, guesses the initial values of the coordinating variables $\boldsymbol{z}, \boldsymbol{\pi}$ and transmits them to the lower level.

step 3 Using $\boldsymbol{z}^{(k)}, \boldsymbol{\pi}^{(k)}$ received from the upper level and the stored values of $\boldsymbol{P}_i(t)$, each subsystem solves the adjoint equation (17), then uses the obtained values of $\boldsymbol{P}_i, \boldsymbol{s}_i$ to solve for $\boldsymbol{x}_i^{(k+1)}, \boldsymbol{u}_i^{(k+1)}$ then sends them to the upper level.

step 4 The upper level uses the received values of $\boldsymbol{x}, \boldsymbol{\pi}$ from all the subsystems to update the coordination vector, puts $k = k + 1$, and then calculates the Euclidean norm of the error,

$$e^{(k+1)} = \| \boldsymbol{f}^{(k+1)} - \boldsymbol{f}^{(k)} \|, \quad \boldsymbol{f}^{(k)} = \left[\boldsymbol{z}_i^{(k)T}\ \boldsymbol{\pi}_i^{(k)T}\right]^T \tag{17}$$

If $e^{(k+1)} \leq \epsilon$, where ϵ is a small positive number, it stops, otherwise sends the new values of $\boldsymbol{z}^{(k+1)}, \boldsymbol{\pi}^{(k+1)}$ to the lower level and goes to step 3.

2.2 The Costate Prediction Approach

Equations (12) and (14) constitute the core of the costate prediction algorithm. The subsystems solve these equations at the lower level then send the results of the integration to the upper level coordinator who do a very simple job of calculating the coordination vector and sending it back to the subsystems. The synchronous algorithm is given as follows:

step 1 The upper level assigns the iteration number $k = 1$, guesses the initial values of the coordinating variables $\boldsymbol{z}, \boldsymbol{\lambda}$ and transmits them to the lower level.

step 2 Using $\boldsymbol{z}^{(k)}$, $\boldsymbol{\lambda}^{(k)}$ received from the upper level each subsystem solves the state and costate equations then sends the updated values to the upper level.

step 3 The upper level uses the received values of $\boldsymbol{x}, \boldsymbol{\lambda}$ from all the subsystem to update the coordination vector, puts $k = k + 1$, and then calculates the Euclidean norm of the error given by

$$e^{(k+1)} = \| \boldsymbol{f}^{(k+1)} - \boldsymbol{f}^{(k)} \|, \quad \boldsymbol{f}^{(k)} = \left[\boldsymbol{z}_i^{(k)T}\ \boldsymbol{\lambda}_i^{(k)T}\right]^T \tag{18}$$

If $e^{(k+1)} \leq \epsilon$, where ϵ is a small positive number, it stops, otherwise sends the new values of $\boldsymbol{z}^{(k+1)}, \boldsymbol{\pi}^{(k+1)}$ to the lower level and goes to step 2.

3 The Asynchronous Algorithms

In order to present the asynchronous algorithms, we assume that the dimensions of the N subsystems are given by $n_1, n_2, \ldots, n_N$ such that

$$n_1 \leq n_2 \leq n_3 \ldots \leq n_N$$

with at least one dimension different from the others. In this case, at least one of the processors will terminate computations before the others, and the following algorithm can be applied:

Algorithm 1 *(Asynchronous Interaction Prediction)*

step 1 Solve the N independent differential Matrix Riccati equations (15) and store the values of $\boldsymbol{P}_i(t)$.

step 2 The upper level assigns the iteration number $k = 1$, guesses the initial values of the coordinating variables $\boldsymbol{z}, \boldsymbol{\pi}$ and transmits them to the lower level.

step 3 Using $\boldsymbol{z}^{(k)}$, $\boldsymbol{\pi}^{(k)}$ received from the upper level and the stored values of $\boldsymbol{P}_i(t)$, the subsystems at the lower level solve the adjoint equation (16) backward, then use the obtained values of $\boldsymbol{P}_i, \boldsymbol{s}_i$ to solve for $\boldsymbol{x}_i^{(k+1)}$, $\boldsymbol{u}_i^{(k+1)}$, and send their values to the upper level. Since there exists at least one subsystem with smaller dimension than the others then at least one processor will terminate before the others. Here we introduce the following rule:

> *When any processor p_i terminates an iteration, it sends the solution to the upper level to update the values of the coordination variables $\boldsymbol{z}_i(t), \boldsymbol{\pi}_i(t)$. Processor p_i will start a new iteration immediately after receiving updated values of the coordination variables independent of the other subsystems.*

step 4 The upper level updates the coordination vector by using the received values of $\boldsymbol{x}, \boldsymbol{\pi}$ from the lower level, and then calculates the iteration error from equation (19). If $e \leq \epsilon$, where ϵ is a small positive number, it stops, otherwise sends the new values of $\boldsymbol{z}, \boldsymbol{\pi}$ to the lower level and goes to step 3.

Algorithm 2 *(Asynchronous Costate Prediction)*

step 1 The upper level assigns the iteration number $k = 1$, guesses the initial values of the coordinating variables $\boldsymbol{z}, \boldsymbol{\lambda}$ and transmits them to the lower level.

step 2 Using $\boldsymbol{z}^{(k)}, \boldsymbol{\lambda}^{(k)}$ received from the upper level each subsystem solves its state and and costate equations then sends the results to the upper level. Since there is at least one subsystem has smaller dimension than the others, at least one processor will terminate before the others. Here we introduce the following rule:

When any processor p_i terminates an iteration, it sends the solution to the upper level to update the values of the coordination variables $\boldsymbol{z}_i(t), \boldsymbol{\lambda}_i(t)$. Processor p_i will start a new iteration immediately after receiving updated values of the coordination variables independent of the other subsystems.

step 3 The upper level updates the coordination vector by using the received values of $\boldsymbol{x}, \boldsymbol{\lambda}$ from the lower level, and then calculates the iteration error from equation (19). If $e \leq \epsilon$, where ϵ is a small positive number, it stops, otherwise sends the new values of $\boldsymbol{z}, \boldsymbol{\lambda}$ to the lower level and goes to step 3.

Remarks

- At the lower level the subsystems are allowed to communicate their results with each other through the coordinator without timing restriction. The most recent information obtained from each subsystem is used to update the predicted interaction with other subsystems. Moreover, the subsystems are allowed to run independently of each other in a complete asynchronous manner, so that subsystems with lower dimensions will release more updated information about their trajectories compared with the synchronous and the partial asynchronous cases.[7,11] This results in a better computational performance as shown in section 6.
- The asynchronous algorithms still maintain the simple structure of the synchronous algorithms. These algorithms are also directly applicable to large scale discrete time control systems. Moreover, for the interaction prediction algorithm a completely closed-loop control structure can be established by applying the extension made by Singh *et.al.*[12]

4 Convergence Analysis

We define $\boldsymbol{W}$ as the matrix of the interaction variables between the ith subsystem and others. $\boldsymbol{W}$ is, in general, a piecewise continuous function having a finite number of discontinuity (as we have finite number of subsystems performing a finite number of iterations), $\boldsymbol{W}$ can be written in the form

$$\boldsymbol{W} = [\boldsymbol{W}_x \;\; \boldsymbol{W}_\lambda] \tag{19}$$

where $\boldsymbol{W}_x$ is the $\boldsymbol{x}$ components of $\boldsymbol{W}$ and $\boldsymbol{W}_\lambda$ is the $\boldsymbol{\lambda}$ components of $\boldsymbol{W}$ given by

$$\boldsymbol{W}_x = [\boldsymbol{w}_{x1} \mid \boldsymbol{w}_{x2} \mid \ldots \mid \boldsymbol{w}_{xN}], \quad \boldsymbol{W}_\lambda = [\boldsymbol{w}_{\lambda 1} \mid \boldsymbol{w}_{\lambda 2} \mid \ldots \mid \boldsymbol{w}_{\lambda N}] \tag{20}$$

We can write the state and costate equation at any iteration in the form

$$\dot{\boldsymbol{x}}_i = \boldsymbol{A}_i\boldsymbol{x}_i - \boldsymbol{B}_i\boldsymbol{R}_i^{-1}\boldsymbol{B}_i^T\boldsymbol{\lambda}_i + \sum_{\substack{j=1\\ j\neq i}}^{N}\boldsymbol{A}_{ij}\boldsymbol{w}_{xj}, \quad \boldsymbol{x}_i(t_o) = \boldsymbol{x}_{io} \tag{21}$$

$$\dot{\boldsymbol{\lambda}}_i = -\boldsymbol{Q}_i\boldsymbol{x}_i - \boldsymbol{A}_i^T\boldsymbol{\lambda}_i - \sum_{\substack{j=1\\ j\neq i}}^{N}\boldsymbol{A}_{ij}\boldsymbol{w}_{\lambda j}, \quad \boldsymbol{\lambda}_i(t_f) = 0 \tag{22}$$

In the following the convergence of the proposed algorithms will be proved based on the results of El-Tarazi[9] for general asynchronous iterative algorithms.

4.1 The Interaction Prediction Algorithm

In this case equations (22) and (23) constitute a two point boundary value problem *(TPBVP)* which can be combined in matrix form as follows:

$$\begin{bmatrix} \dot{\boldsymbol{x}}_i \\ \dot{\boldsymbol{\lambda}}_i \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_i & -\boldsymbol{B}_i\boldsymbol{R}_i^{-1}\boldsymbol{B}_i^T \\ -\boldsymbol{Q}_i & -\boldsymbol{A}_i^T \end{bmatrix} + \begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{\lambda}_i \end{bmatrix} + \sum_{\substack{j=1\\ j\neq i}}^{N}\begin{bmatrix} \boldsymbol{A}_{ij} & 0 \\ 0 & -\boldsymbol{A}_{ij}^T \end{bmatrix}\begin{bmatrix} \boldsymbol{w}_{xj} \\ \boldsymbol{w}_{\lambda_j} \end{bmatrix} \tag{23}$$

Let $\boldsymbol{\nu}_i^{(k)} = \left[\boldsymbol{x}_i^{(k)} \;\; \boldsymbol{\lambda}_i^{(k)}\right]^T$,

then equation (24) can be written in the form

$$\dot{\boldsymbol{\nu}}_i^{(k+1)} = \boldsymbol{H}_{oi}\boldsymbol{\nu}_i^{(k+1)} + \sum_{j=1}^{N}\boldsymbol{H}_{ij}\boldsymbol{\nu}_j^{(k)} \tag{24}$$

where

$$\boldsymbol{H}_{oi} = \begin{bmatrix} \boldsymbol{A}_i & -\boldsymbol{B}_i\boldsymbol{R}_i^{-1}\boldsymbol{B}_i^T \\ -\boldsymbol{Q}_i & -\boldsymbol{A}_i^T \end{bmatrix}, \;\; \boldsymbol{H}_{ij} = \begin{bmatrix} \boldsymbol{A}_{ij} & 0 \\ 0 & -\boldsymbol{A}_{ij}^T \end{bmatrix}, \boldsymbol{H}_{ii} = 0, \;\; \forall i,j \in [i,N]$$

Let the optimal solution be $\boldsymbol{\nu}_i^* = [\boldsymbol{x}_i^* \;\; \boldsymbol{\lambda}_i^*]^T$, where $\boldsymbol{x}_i^*$, $\boldsymbol{\lambda}_i^*$ are the optimal values of $\boldsymbol{x}$, $\boldsymbol{\lambda}$. Define the interaction error of subsystem i at iteration k to be $\boldsymbol{e}_i^{(k)} = \boldsymbol{\nu}_i^{(k)} - \boldsymbol{\nu}_i^* = \left[\boldsymbol{e}_{ix}^{(k)} \;\; \boldsymbol{e}_{i\lambda}^{(k)}\right]$, and the iteration error to be $\boldsymbol{e}_{wj} = [\boldsymbol{e}_{wxj} \;\; \boldsymbol{e}_{w\lambda j}] = \left[\boldsymbol{w}_{xj} - \boldsymbol{x}_j^* \;\; \boldsymbol{w}_{\lambda j} - \boldsymbol{\lambda}_j^*\right]$. Now we can rewrite equation (25) as follow:

$$\dot{\boldsymbol{e}}_i = \boldsymbol{H}_{oi}\boldsymbol{e}_i + \sum_{j=1}^{N}\boldsymbol{H}_{ij}\boldsymbol{e}_{wj} \tag{25}$$

Solving with respect to time we get

$$\boldsymbol{e}_i(t) = \boldsymbol{\Phi}_i(t, t_o)\boldsymbol{e}_i(t) + \sum_{j=1}^{N} \int_{t_o}^{t} \boldsymbol{\Phi}_i(t, \tau)\boldsymbol{H}_{ij}\boldsymbol{e}_{wj}(\tau)\, d\tau, \quad \boldsymbol{e}_i(t_o) = [\boldsymbol{e}_{ix}(t_o) \;\; \boldsymbol{e}_{i\lambda}(t_o)]^T \tag{26}$$

where $\boldsymbol{\Phi}_i(t, t_o)$ is the state transition matrix for the subsystem i given by

$$\boldsymbol{\Phi}(t, \tau) = \exp \boldsymbol{H}_{oi}(t - \tau) = \begin{bmatrix} \boldsymbol{\Phi}_{i11}(t, \tau) & \boldsymbol{\Phi}_{i12}(t, \tau) \\ \boldsymbol{\Phi}_{i21}(t, \tau) & \boldsymbol{\Phi}_{i22}(t, \tau) \end{bmatrix} \tag{27}$$

Substituting with $t = t_f$, and since $\boldsymbol{e}_{i\lambda}(t_f) = 0$ and $\boldsymbol{e}_{ix}(t_o) = 0$, we can obtain

$$\boldsymbol{e}_i(t) = \sum_{j=1}^{N} \left(\int_{t_o}^{t} \boldsymbol{\Phi}_i(t, \tau)\boldsymbol{H}_{ij}\boldsymbol{e}_{wj}(\tau)\, d\tau - \int_{t_o}^{t_f} \boldsymbol{\Theta}_i(t_o, t_f, t, \tau)\boldsymbol{H}_{ij}\boldsymbol{e}_{wj}(\tau)\, d\tau \right),$$

$$\boldsymbol{\Theta}_i(t_o, t_f, t, \tau) = \begin{bmatrix} \boldsymbol{\Phi}_{i12}(t, t_o) \\ \boldsymbol{\Phi}_{i22}(t, t_o) \end{bmatrix} \boldsymbol{\Phi}_{i22}^{-1}(t_f, t_o) \; [\boldsymbol{\Phi}_{i12}(t_f, \tau) \;\; \boldsymbol{\Phi}_{i22}(t_f, \tau)] \tag{28}$$

To obtain sufficient conditions for the convergence of the global system, we define the norm for the error vector $e(t)$ over the time period $[t_o, t_f]$ to be

$$\text{Norm of } e(t) = \max_{t \in [t_o, t_f]} \| \, e(t) \, \|_2 \tag{29}$$

where $\| \cdot \|_2$ is the Euclidean norm in $\mathcal{R}^{2n}$.

Taking the norm of both sides of equation (30) we get

$$\max_{t \in [t_o, t_f]} \| \, e(t) \, \| \leq \sum_{j=1}^{N} M_{ij} \int_{t_o}^{t_f} \max_{t \in [t_o, t_f]} \| \, \boldsymbol{e}_{wj}(t) \, \| \; d\tau \tag{30}$$

Where

$$M_{ij} = \max_{t, \tau \in [t_o, t_f]} \| \, \boldsymbol{\Phi}_i \boldsymbol{H}_{ij} \, \| + \max_{t, \tau \in [t_o, t_f]} \| \, \boldsymbol{\Theta}_i \boldsymbol{H}_{ij} \, \| \tag{31}$$

From inequality (32), we get

$$\max_{t \in [t_o, t_f]} \| \, e(t) \, \| \leq (t_o - t_f) \left(\sum_{j=1}^{N} M_{ij} \right) \max_{\substack{t \in [t_o, t_f] \\ j \in [1, N], \, j \neq i}} \| \, \boldsymbol{e}_{wj}(t) \, \| \tag{32}$$

and we can conclude that

$$\max_{\substack{t\in[t_o,t_f]\\ i\in[1,N]}} \| \boldsymbol{e}_i(t) \| \leq \alpha_i \max_{\substack{t\in[t_o,t_f]\\ j\in[1,N],\, j\neq i}} \| \boldsymbol{e}_{wj}(t) \|, \quad \alpha_i = (t_o - t_f) \sum_{j=1}^{N} M_{ij} \tag{33}$$

If we choose $(t_f - t_o)$ such that $\alpha_i < 1, \ \ \forall i \in [1, N]$, then inequality (35) defines a property of contraction on the space E defined by

$$E = \prod_{i=1}^{N} E_i, \quad E_i = C([t_o, t_f]; \mathcal{R}^{2n_i}) \tag{34}$$

where $C(\cdot)$ denotes the set of continuous functions. From El-Tarazi,[9] this contraction property guarantees the convergence of the asynchronous iterations.

4.2 The Costate Prediction Algorithm

Here we will solve equations (22) and (23) independently of the other. If we solve equation (22) forward with time we get

$$\boldsymbol{x}_i(t) = \boldsymbol{\Phi}_{ix}(t, t_o)\boldsymbol{x}_i(t) + \int_{t_o}^{t} \boldsymbol{\Psi}_{ix}(t,\tau)\boldsymbol{\lambda}_i(\tau) + \sum_{j=1}^{N} \boldsymbol{\Psi}_{ijx}(t,\tau)\boldsymbol{w}_{xj}(\tau)\, d\tau, \tag{35}$$

where,

$$\begin{aligned} \boldsymbol{\Phi}_{ix}(t,\tau) &= e^{\boldsymbol{A}_i(t-\tau)}, \qquad \boldsymbol{\Psi}_{ix}(t,\tau) = -\boldsymbol{\Phi}_{ix}(t,\tau)\boldsymbol{B}_i\boldsymbol{R}_i^{-1}\boldsymbol{B}_i^T, \\ \boldsymbol{\Psi}_{ijx}(t,\tau) &= \boldsymbol{\Phi}_{ix}(t,\tau)\boldsymbol{A}_{ij}, \quad \boldsymbol{\Psi}_{iix}(t,\tau) = 0 \qquad \forall i, j \in [1, N] \end{aligned} \tag{36}$$

By solving equation (23) backward with time we get

$$\boldsymbol{\lambda}_i(t) = \boldsymbol{\Phi}_{i\lambda}(t_f, t)\boldsymbol{\lambda}_i(t_f) + \int_{t}^{t_f} \boldsymbol{\Psi}_{i\lambda}(t,\tau)\boldsymbol{x}_i(\tau) + \sum_{j=1}^{N} \boldsymbol{\Psi}_{ij\lambda}(t,\tau)\boldsymbol{w}_{\lambda j}(\tau)\, d\tau \tag{37}$$

where,

$$\begin{aligned} \boldsymbol{\Phi}_{i\lambda}(t,\tau) &= e^{-\boldsymbol{A}_i^T(t-\tau)}, \qquad \boldsymbol{\Psi}_{i\lambda}(t,\tau) = -\boldsymbol{\Phi}_{i\lambda}(t,\tau)\boldsymbol{Q}_i, \\ \boldsymbol{\Psi}_{ij\lambda}(t,\tau) &= \boldsymbol{\Phi}_{i\lambda}(t,\tau)\boldsymbol{A}_{ji}^T, \quad \boldsymbol{\Psi}_{ii\lambda}(t,\tau) = 0 \quad \forall i, j \in [1, N] \end{aligned} \tag{38}$$

Substituting, by the final condition given in equation (23) and then by $\boldsymbol{x}(t)$ from (37) into (39), we get

$$\begin{aligned}\boldsymbol{\lambda}_i(t) &= \boldsymbol{\Phi}_{i\lambda}(t_f,t)\boldsymbol{\lambda}_i(t_f) + \int_t^{t_f} \boldsymbol{\Xi}_i(t,\tau,t_o)\boldsymbol{x}_i(t_o) + \sum_{j=1}^{N} \boldsymbol{\Psi}_{ij\lambda}(t,\tau)\boldsymbol{w}_{\lambda j}(\tau)\, d\tau + \\ &\int_t^{t_f}\int_{t_o}^{\tau} \boldsymbol{\Omega}_i(t,\tau,\upsilon)\boldsymbol{\lambda}_i(\upsilon) + \sum_{j=1}^{N} \boldsymbol{\Omega}_{ij}(t,\tau,\upsilon)\boldsymbol{w}_{xj}(\upsilon)\, d\upsilon\, d\tau \end{aligned} \tag{39}$$

where

$$\begin{aligned}&\boldsymbol{\Xi}_i(t,\tau,t_o) = \boldsymbol{\Psi}_{i\lambda}(t,\tau)\,\boldsymbol{\Phi}_{ix}(\tau,t_o), \quad \boldsymbol{\Omega}_i(t,\tau,\upsilon) = \boldsymbol{\Psi}_{i\lambda}(t,\tau)\,\boldsymbol{\Psi}_{ix}(\tau,\upsilon), \\ &\boldsymbol{\Omega}_{ij}(t,\tau,\upsilon) = \boldsymbol{\Psi}_{i\lambda}(t,\tau)\boldsymbol{\Psi}_{ix}(\tau,\upsilon) \qquad \forall i,j \in [1,N]\end{aligned} \tag{40}$$

Note that at the optimal solution, we have

$$\boldsymbol{x}_i^*(t) = \boldsymbol{\Phi}_{ix}(t,t_o)\boldsymbol{x}_i^*(t) + \int_{t_o}^{t} \boldsymbol{\Psi}_{ix}(t,\tau)\boldsymbol{\lambda}_i^*(\tau) + \sum_{j=1}^{N} \boldsymbol{\Psi}_{ijx}(t,\tau)\boldsymbol{x}_j^*(\tau)\, d\tau, \tag{41}$$

Subtracting (43) from (37) and using our error definition in the previous subsection, and from the initial condition in (22) we obtain

$$\boldsymbol{e}_{ix}(t) = \int_{t_o}^{t} \boldsymbol{\Psi}_{ix}(t,\tau)\boldsymbol{e}_{w\lambda i}(\tau) + \sum_{j=1}^{N} \boldsymbol{\Psi}_{ijx}(t,\tau)\boldsymbol{e}_{wxj}(\tau)\, d\tau \tag{42}$$

Following the same procedures for the $\boldsymbol{\lambda}$ component we get

$$\begin{aligned}\boldsymbol{e}_{i\lambda}(t) &= \int_t^{t_f} \sum_{j=1}^{N} \boldsymbol{\Psi}_{ij\lambda}(t,\tau)\boldsymbol{e}_{w\lambda j}(\tau)\, d\tau + \\ &\int_t^{t_f}\int_{t_o}^{\tau} \boldsymbol{\Omega}_i(t,\tau,\upsilon)\boldsymbol{e}_{w\lambda i}(\upsilon) + \sum_{j=1}^{N} \boldsymbol{\Omega}_{ij}(t,\tau,\upsilon)\boldsymbol{e}_{wxj}(\upsilon)\, d\upsilon\, d\tau \end{aligned} \tag{43}$$

Taking the norm of both sides of equation (44) and integrating we get

$$\max_{t\in[t_o,t_f]} \parallel \boldsymbol{e}_{ix}(t) \parallel \leq (t_f - t_o)\left(M_{ix} \max_{t\in[t_o,t_f]} \parallel \boldsymbol{e}_{w\lambda i}(t) \parallel + \sum_{j=1}^{N} M_{ijx} \max_{t\in[t_o,t_f]} \parallel \boldsymbol{e}_{wxj}(t) \parallel \right) \tag{44}$$

where,

$$M_{ix} = \max_{t,\tau\in[t_o,t_f]} \| \boldsymbol{\Phi}_{ix}(t,\tau) \|, \quad M_{ijx} = \max_{t,\tau\in[t_o,t_f]} \| \boldsymbol{\Psi}_{ijx}(t,\tau) \| \tag{45}$$

Similarly, by taking the norm of equation (45), we get

$$\begin{aligned} \max_{t\in[t_o,t_f]} \| \boldsymbol{e}_{i\lambda}(t) \| \quad \leq \quad & (t_f - t_o) \sum_{j=1}^{N} M_{ij\lambda} \max_{t\in[t_o,t_f]} \| \boldsymbol{e}_{w\lambda j}(t) \| + \\ & (t_f - t_o)^2 \left(M_{i\lambda x} \max_{t\in[t_o,t_f]} \| \boldsymbol{e}_{w\lambda j}(t) \| + \sum_{j=1}^{N} M_{ijx\lambda} \max_{t\in[t_o,t_f]} \| \boldsymbol{e}_{wxj}(t) \| \right) \end{aligned} \tag{46}$$

where

$$M_{ij\lambda} = \max_{t,\tau\in[t_o,t_f]} \| \boldsymbol{\Psi}_{ix} \|, \quad M_{i\lambda x} = \max_{t,\tau\upsilon\in[t_o,t_f]} \| \boldsymbol{\Omega}_{i} \|, \quad M_{ijx\lambda} = \max_{t,\tau,\upsilon\in[t_o,t_f]} \| \boldsymbol{\Omega}_{ij} \| \tag{47}$$

Combining inequality (46) and (48) in a matrix form, with $T = (t_f - t_o)I$ is the integration period multiplied by the identity matrix of size $n_i \times n_i$, we obtain

$$\begin{aligned} \begin{bmatrix} \max\limits_{t\in[t_o,t_f]} \| \boldsymbol{e}_{ix} \| \\ \max\limits_{t\in[t_o,t_f]} \| \boldsymbol{e}_{i\lambda} \| \end{bmatrix} \quad = \quad & \begin{bmatrix} 0 & TM_{ix} \\ 0 & T^2 M_{i\lambda x} \end{bmatrix} \begin{bmatrix} \max\limits_{t\in[t_o,t_f]} \| \boldsymbol{e}_{wxi} \| \\ \max\limits_{t\in[t_o,t_f]} \| \boldsymbol{e}_{w\lambda i} \| \end{bmatrix} + \\ & \sum_{j=1}^{N} \begin{bmatrix} TM_{ijx} & 0 \\ T^2 M_{ij\lambda x} & TM_{ij\lambda} \end{bmatrix} \begin{bmatrix} \max\limits_{t\in[t_o,t_f]} \| \boldsymbol{e}_{wxj} \| \\ \max\limits_{t\in[t_o,t_f]} \| \boldsymbol{e}_{w\lambda j} \| \end{bmatrix} \end{aligned} \tag{48}$$

Taking the global maximum of the previous inequality, we get

$$\max_{t\in[t_o,t_f]} \| \boldsymbol{e}_{i}(t) \| \leq \beta_i \max_{t\in[t_o,t_f]} \| \boldsymbol{e}_{wj}(t) \| + \sum_{\substack{j=1 \\ j\neq i}}^{N} \gamma_{ij} \max_{t\in[t_o,t_f]} \| \boldsymbol{e}_{wj}(t) \| \tag{49}$$

where

$$\beta_i = \max(\| TM_{ix} \|, \ \| T^2 M_{i\lambda x} \|), \quad \gamma_{ij} = \max(\| TM_{ijx} \|, \ \| T^2 M_{ij\lambda x} \ TM_{ij\lambda} \|) \tag{50}$$

Taking the maximum over all the subsystems we get the following inequality

$$\max_{\substack{t\in[t_o,t_f] \\ i\in[1,N]}} \| \boldsymbol{e}_{i}(t) \| \leq \delta_i \max_{\substack{t\in[t_o,t_f] \\ j\in[1,N],\, j\neq i}} \| \boldsymbol{e}_{wj}(t) \| \tag{51}$$

Similar to the previous algorithm, if we choose $(t_f - t_o)$ such that $\delta_i < 1, \ \forall i \in [1, N]$, then the last inequality defines a property of contraction on the space E defined in (36), and from El-Tarazi,[9] this contraction property guarantees the convergence of asynchronous point iterations.

5 On The Asynchronous Algorithms

Because of the complex nature of the asynchronous algorithms and the lack of common timing frame [1], it is difficult to represent it mathematically. In the following, we will present some remarks about these algorithms based upon their algorithmic and mathematical structures, pointing out some practical considerations.

5.1 The Algorithm Structure

- In the asynchronous implementation the processors are not required to receive all the results of the previous iteration, rather, each processor is allowed to continue iterating with its own components at its own pace. If the current value of the component updated by some other processor is not available, then the latest update received is used instead. Furthermore, processors are not required to communicate their results after each iteration but only once in a while. In this structure we allow some processors to compute faster and execute more iterations than others. Only the subsystem(s) with the largest dimension will do as many iterations as performed in the synchronous case. Therefore, it is expected that the asynchronous algorithms will show faster convergence.

- Although the algorithm is expected to perform well on any type of multiprocessor, shared memory system is more suitable for it. In a shared memory system all subsystems can have access to the system trajectories simultaneously, which reduces the effect of communication delay, which is a concern for this type of algorithms. Using a shared memory system also reduces the coordination task to calculating the iteration error and terminating the computations.

- The analytical study of the asynchronous algorithms shows that, its convergence depends on two factors: the integration period, and the interaction between the subsystems. These two factors are also dominant factors for the convergence of the synchronous algorithms. However, we saw from the practical applications that asynchronous algorithms (within certain ranges) are more stable with respect to these two factors. Another observation from the convergence analysis is that, in the costate prediction algorithm, the internal structure of the system have effect in the convergence, and this may reduce the efficiency of the costate algorithms in some cases.

- Like synchronous algorithms, we can deal with some cases of divergence by applying a set of adjustments to the optimization horizon and/or the cost matrices.[1,11,12]

[1] In the asynchronous environment, each subsystems performs its computations independent of the others and it is not possible to define a global iteration index.

5.2 The Effect of Communication Delay

It is difficult to measure theoretically the effect of communication delay on the performance of the asynchronous algorithms. Nevertheless, we can see from the algorithm structure and practical implementation that the delay caused by information exchange between the subsystems is more substantial than the synchronous case. Moreover, the communication effect is hard to predict, because it is difficult to estimate the accumulated timing differences between subsystems and to know the effect of updating the information randomly. However, from our practical experience, we propose the following measures that could help to overcome or reduce the effects of this problem:

- Provide a time mechanism inside the coordinator to limit the amount of information exchange between the subsystems. For example, in the case of large variation between the dimension of the subsystems, a small subsystem may perform many iterations within the time of a single iteration of a large subsystem, without significant impact on the total convergence. If the communication overhead is substantial, then it is recommended to minimize unnecessary or less effective communications by limiting the minimum time between two successive iterations for some subsystems.
- Reduce the broadcasting by using either a special purpose communication hardware, or a multiprocessor network that matches the decomposition of the subsystem and their dependency, so that all necessary communications involve directly connected nodes.

6 Simulation Results

To demonstrate the effectiveness of the proposed algorithm, numerical simulations have been done on a single processor computer, using a parallel processing simulation program. In the following we will present the simulation results for three practical systems that were solved using both the synchronous and asynchronous algorithms.

River Pollution Control

The problem for the river pollution control (RPC) is to maintain the instream biomedical oxygen demand (BOD) and the desolved oxygen (DO) at prespecified levels for a river with multiple polluters using the discharges from the sewage stations as the control variables. The control problem for a two-reach system without delay is represented by.[13]

$$\begin{aligned} \min J &= \int_0^8 (\boldsymbol{x}(t)-\boldsymbol{x}_d)^T \boldsymbol{Q}(\boldsymbol{x}(t)-\boldsymbol{x}_d) + \boldsymbol{u}^T(t)\boldsymbol{R}_i\boldsymbol{u}(t)dt, \\ \dot{\boldsymbol{x}}(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{D}, \quad \boldsymbol{x}(0) = \boldsymbol{x}_o, \end{aligned} \tag{52}$$

$$\boldsymbol{A} = \begin{bmatrix} -1.32 & 0 & 0 & 0 \\ -0.32 & -1.2 & 0 & 0 \\ 0.9 & 0 & -1.32 & 0 \\ 0 & 0.9 & -0.32 & -1.2 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0 \\ 0 & 0.1 \\ 0 & 0 \end{bmatrix}, \boldsymbol{D} = \begin{bmatrix} 5.35 \\ 10.9 \\ 4.19 \\ 1.9 \end{bmatrix}$$

This system was split into two subsystems, of orders 1 and 3. The simulation was carried out with the following values,

$$\begin{aligned} \boldsymbol{Q} &= \text{diag}(2, 1, 2, 1) \qquad \boldsymbol{R} = \text{diag}(1, 1), \\ \boldsymbol{x}_d &= [4.06\ 8\ 5.94\ 6]^T, \quad \boldsymbol{x}_o = [10\ 7\ 5\ 7]^T \end{aligned}$$

Figure 1 shows the evolution of error with the iteration for the synchronous and the asynchronous costate and interaction prediction algorithms.

Gas Absorber Tower

A gas absorber tower (GAT) is an important element in several chemical processes. A typical gas absorber system consists of a number of vertically arranged plates enclosed within a chemical tower. The chemical reactions that take place in the tower are affected by the inlet feed compositions corresponding to a downward liquid stream and upward vapor stream.[14] From the material balance of each plate in the tower, a state-variable model has been developed. The system is decomposed into two subsystems with dimensions 2 and 4 respectively. The system is initially stable and its model has the following parameters

$$\boldsymbol{A} = \begin{bmatrix} -1.173 & 0.634 & 0 & 0 & 0 & 0 \\ 0.538 & -1.173 & 0.634 & 0 & 0 & 0 \\ 0 & 0.538 & -1.173 & 0.634 & 0 & 0 \\ 0 & 0 & 0.538 & -1.173 & 0.634 & 0 \\ 0 & 0 & 0 & 0.538 & -1.173 & 0.634 \\ 0 & 0 & 0 & 0 & 0.538 & -1.173 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 0.538 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0.88 \end{bmatrix}$$

$$\boldsymbol{Q} = \begin{bmatrix} 136 & 17 & 54 & 0 & 0 & 0 \\ 17 & 23 & 8 & 0 & 0 & 0 \\ 54 & 80 & 29 & 0 & 0 & 0 \\ 0 & 0 & 0 & 17 & 30 & 13 \\ 0 & 0 & 0 & 30 & 28 & 2 \\ 0 & 0 & 0 & 13 & 28 & 18.5 \end{bmatrix}, \quad \boldsymbol{R} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The system Ricatti equations were integrated over 5 seconds. Figure 2 shows the evolution of error with the iteration for the synchronous and the asynchronous costate and interaction prediction algorithms.

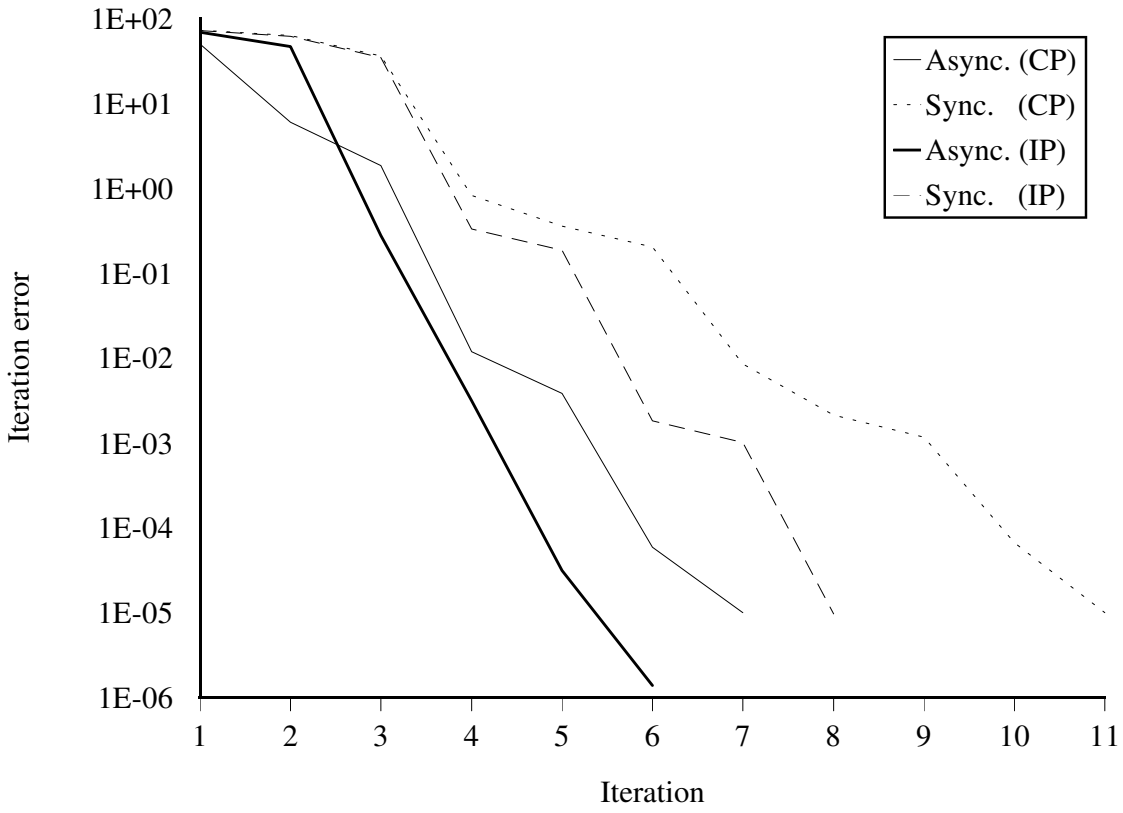

Figure 1: Interaction error against iteration for river pollution control

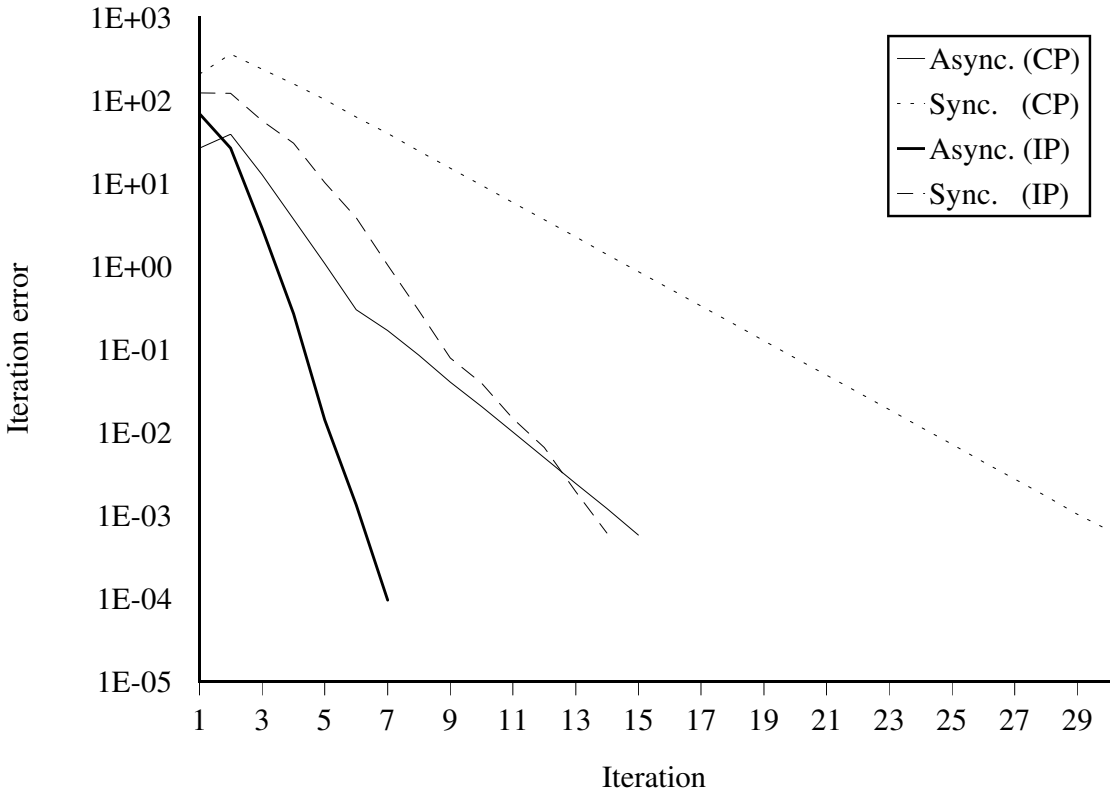

Figure 2: Interaction error against iteration for gas absorber tower

Power System Control

The power system (PSC) problem under consideration consists of 7 machines. The mathematical model of the n-machine systems consists of a set of nonlinear differential equations given in.[15] The linearized model have the following parameters

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -93.931 & 0 & 8.743 & 0 & 5.374 & 0 & 2.626 & 0 & 4.056 & 0 & -1.647 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 15.85 & 0 & -140.367 & 0 & 9.6 & 0 & 14.026 & 0 & 6.684 & 0 & 2.675 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 10.7 & 0 & 10.25 & 0 & -116.06 & 0 & 6.22 & 0 & 4.313 & 0 & 0.416 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 8.515 & 0 & 18.074 & 0 & 7.655 & 0 & -13.97 & 0 & 4.075 & 0 & 1.703 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 11.295 & 0 & 8.604 & 0 & 5.989 & 0 & 4.313 & 0 & -124.46 & 0 & 2.98 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1.969 & 0 & 1.962 & 0 & 1.684 & 0 & 1.212 & 0 & 1.645 & 0 & -99.69 & 0 \end{bmatrix}$$

$$\boldsymbol{B} = I, \quad \boldsymbol{Q} = 0.1I, \quad \boldsymbol{R} = I$$

It is required to control each state variable of the power system such that the quadratic cost function is minimized. In simulation, the system has been split up into two subsystems of orders 4 and 8 respectively. The system Ricatti equations were integrated over 4 seconds. Figure 3 shows the evolution of error with the iteration for the synchronous and the asynchronous costate and interaction prediction algorithms.

Comparing The Computational Efficiency

Table 1 shows computational comparison between the synchronous and asynchronous algorithms for the previous examples. The table shows the result of both the interaction prediction (IP) and the costate prediction algorithm (CP). In this table, e_f is the error at the final step, η is the number of iteration needed to achieve the specified level of error, $T_a/T_s\%$ is the percentage ratio between the computational time (required to reach the specified error level) using the asynchronous algorithm (T_a) and the computational time using the synchronous algorithm (T_s), and G is the efficiency factor of the algorithm given by

$$G = -\frac{\log(e_f)}{\eta} \tag{53}$$

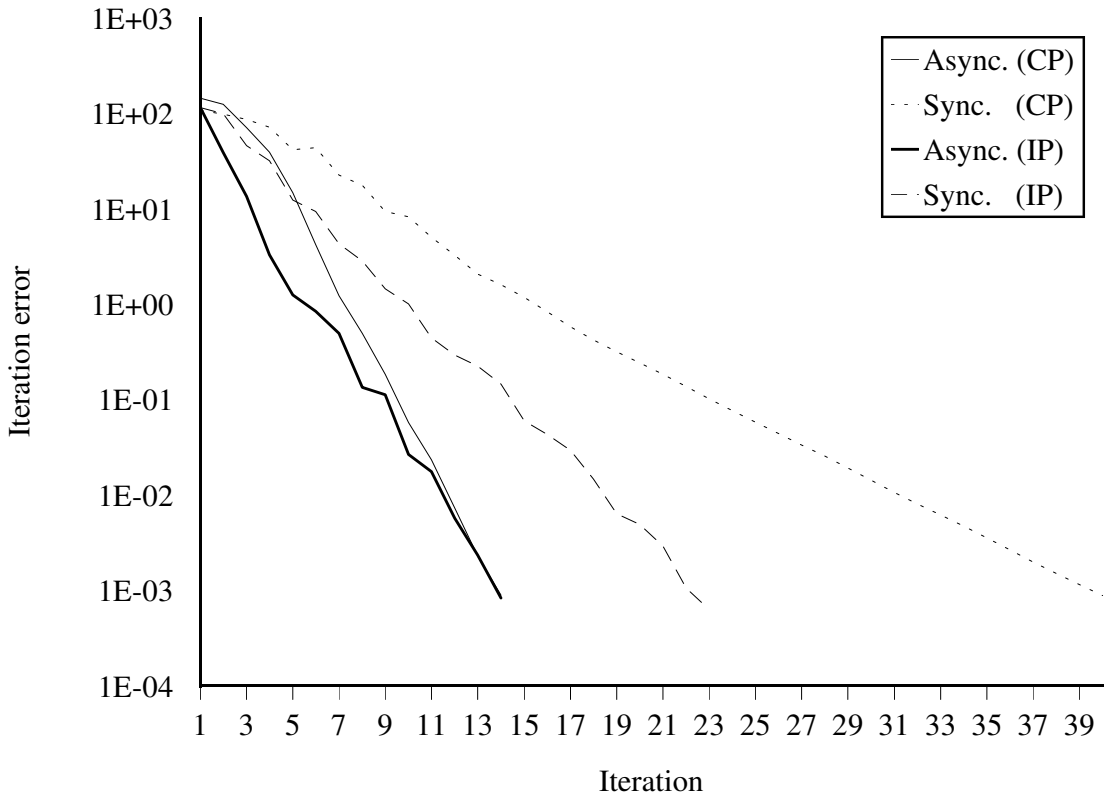

Figure 3: Interaction error against iteration for power system

7 Conclusion

In this paper, we proposed asynchronous algorithms for the costate and interaction prediction methods. A mathematical study of the convergence behavior of the proposed algorithms is presented. It is shown that the new algorithm is convergent under given sufficient conditions which are strongly related to the interaction between the subsystems and the integration period. In the case of the costate prediction, the internal system structure also influences the convergence of the algorithm.

In view of the structural properties and the convergence results of the proposed algorithms, some important comments related to the practical implementation are presented. Also, we suggest specific methods to reduce the effect of the high communication delay resulting from excessive information exchange among the subsystems.

Finally, we present the results of the numerical simulation of the algorithms applied to a number of practical systems. In general, the solutions were identical to the global optimal solution obtained from both the centralized and synchronous algorithms. The comparison with the asynchronous algorithms shows that a substantial reduction in the total calculation time is achieved through the proposed algorithms for all examined cases.

References

[1] Mahmoud, M. S., M. F. Hassan and M. G. Darwish, *Large Scale Control Systems Theories and Techniques*, Dekkar, New York, 1985.

[2] Bertsekas, D. B. and J. N. Tsitsiklis, 'Some aspects of parallel and distributed iterative algorithms-a survey', *Automatica*, **27**, 3–21, (1991).

[3] Kung, H. T., 'Synchronized and asynchronous parallel algorithms for multiprocessors', In J.F. Traub, editor, *Algorithms and complexity :New directions and Recent Results*, pages 217–292. Academic Press, (1976).

[4] Bertsekas, D. B. and J. N. Tsitsiklis, *Parallel and distributed computation: Numerical Methods*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[5] Lang, B., J. C. Miellou, and P. Spiteri, 'Asynchronous relaxation algorithms for optimal control problems', *Mathematics and Computer in Simulation*, **28**, 227–242, (1986).

[6] Mitra, D., 'Asynchronous relaxation for the numerical solutions of differential equations by parallel processors', *Siam J. Sci. Stat. Comput.*, **8**, s43–58, (1987).

[7] Abdelwahed, S. S., M. F. Hassan, and M. A. Sultan, 'Partially asynchronous co-state prediction algorithm', *IEE Proc. Control Theory Appl.*, **142**, 88–94, (1995).

[8] Hassan, M. F., 'New multi-level algorithm for linear dynamical systems', *Int. J. Sys. Sci.*, **19**, (1988).

[9] El-Tarazi, M. M., 'Some convergence results for asynchronous algorithms', *Numreish Mathematik*, **39**, 325–340, (1982).

[10] Jamshidi, M., *Large-scale Systems Modeling and Control*, North-Holland, New York, 1983.

[11] Abdelwahed, S. S., *Development of multi-level asynchronous algorithms for complex systems*, Master's thesis, Dept. of Elec. Eng., Cairo Univ. Egypt, 1993.

[12] Singh, M. G. and A. Titli, *Systems: Decomposition, Optimization, and Control*, Pergamon Press, Oxford, 1978.

[13] Tamura, H., 'Decentralized optimization for distributed-lag models of discrete systems', *IEEE Trans. Sys. Man Cyber.*, **4**, 424–429, (1974).

[14] Darwish, M. and J. Fantin, 'Stabilization and control of absorber tower chemical process', In *Third IFAC/IFIP/IFORs Conf.*, pages 153–158. Rabat,Moroco, (1970).

[15] Darwish, M. and J. Fantin, 'The application of lyapunov method to large power systems using decomposition and aggregation techniques', *Int. J. Control*, **29**, 351–362, (1979).

System	algorithm	Synchronous Algorithms			Asynchronous Algorithms			T_a/T_s%
		η	e_f	G	η	e_f	G	
RPC	IP	8	9.7e-6	0.87	6	1.39e-6	1.02	75 %
	CP	11	9.9e-6	0.64	7	9.2e-6	0.99	64 %
GAT	IP	14	6.1e-4	0.34	7	9.6e-5	0.85	50 %
	CP	30	8.1e-4	0.16	15	5.6e-4	0.32	50 %
PSC	IP	23	6.3e-4	0.21	14	8.3e-4	0.35	61 %
	CP	40	8.6e-4	0.12	14	8.9e-4	0.34	36 %

Table 1: Comparison between the synchronous and asynchronous algorithms