STABILITY OF 2−D DISTRIBUTED PROCESSES WITH TIME-VARIANT COMMUNICATION DELAYS

Peter H. Bauer and Mihail L. Sichitiu

Dept. of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556

Kamal Premaratne

Dept. of Electrical and Computer Engineering
University of Miami
Coral Gables, Fl. 33124

ABSTRACT

Motivated by the applications of parallel and distributed computing in m−D systems, this paper analyzes the effect of time-variant communication delays on the stability of 2−D systems. The time-variant communication delays are at £rst modeled and then used in the formulation of the 2−D Roesser local state space model. The arising overall dynamic system model is of uncertain, shift-variant form and it is described by £nite uncertainty sets. The result is a suf£cient condition for stability of the described process.

1. INTRODUCTION

The complexity of computing the output of a system increases with the dimensionality of the process, since both, the volume of the data set and and the complexity of the process grows exponentially with the dimensionality. Therefore, in m-D processes , parallelizing or at least pipe-lining the computation process is often an important consideration. With the wide availability of computer networks, distributed computing is an obvious solution to the problem. However, due to network congestion, £xed, nominal (and small) communication delays cannot always be assured. Hence, the occurrence of time-variant communication delays [1, 2] must be included in such a model, especially if the system under consideration works in real time and 'waiting for a delayed sample' is not an option. In such a situation, an available sample can be substituted in place of the delayed (or even lost) sample in order to satisfy real time constraints [3]. The exact choice of the substituted sample is of secondary importance for this paper, but several obvious solutions (such as nearest available neighbor) exist. Obviously, there are two main questions that arise from such a real time scheme:

1. What is the effect on the computed solution, i.e how much does it differ from the ideal solution?

2. Under what conditions can stability of the original m−D process (which is assumed) be maintained?

This paper will shed some light on the second question.

2. MAIN RESULT

2.1. Process Model

We limit our discussion to 1st quadrant causal nominally stable 2−D systems that can be described by difference equations of the general form:

$$
\begin{aligned}
y(n_1, n_2) &= \sum_{(i,j) \in \mathcal{M}_0} a_{ij} y(n_1 - i, n_2 - j) + \\
&+ \sum_{(i,j) \in \mathcal{M}_i} b_{ij} x(n_1 - i, n_2 - j)
\end{aligned}
\tag{1}
$$

where $(n_1, n_2) \in \mathcal{N}_0^2$, $a_{ij}, b_{ij} \in \mathcal{R}$ and the input and output signals are denoted by $x(n_1, n_2)$ and $y(n_1, n_2)$ respectively. $\mathcal{N}_0$ and $\mathcal{R}$ denotes the set of non-negative integers and the set of real numbers respectively.

The sets of non-negative integer pairs $\mathcal{M}_i$ and $\mathcal{M}_0$ de£ne the input and output masks. In particular $\mathcal{M}_i \subset \mathcal{N}_0^2$ and $\mathcal{M}_0 \subset \mathcal{N}_0^2 \backslash \{(0,0)\}$. By assumption the coef£cients a_{ij} and b_{ij} are the nominal system coef£cients that result in BIBO stability of (1), which can be determined using a variety of available methods [4].

If equation (1) is to be computed in parallel on a number of processors, let's say C of them, the following conditions need to be satis£ed in order to ensure computability:

(i) Each of the C processors compute the output for a certain prede£ned subregion S_i of $\mathcal{N}_0^2$.

(ii) These subregions are a partition of the 1st quarterplane or a subregion $\mathcal{S}$ of $\mathcal{N}_0^2$. (A generalization to the 1st hyper-quadrant or even other causality regions of m−D processes is straight forward.)

(iii) All (previously computed) outputs within the output masks and inputs within the input mask must be available to each of the processors in time, i.e. the processor can not wait for a sample to arrive.

Since the S_i, $i = 1, \ldots, C$ partition the quarter plane (or a subset $\mathcal{S}$ of it) we have:

$$
\bigcup_{i=1}^{C} S_i = \mathcal{S} \subset \mathcal{N}_0^2; \quad S_i \bigcap_{i \neq j} S_j = \phi
\tag{2}
$$

Furthermore, for simplicity we assume that the entire input signal $x(n_1, n_2)$ is known to all C processors before the output is computed. (This condition can be relaxed in stages, requiring each processor to know only input samples that are required to compute $y(n_1, n_2)$ in S_i or one could view transmitting the input samples to each processor in the same way as we view communicating output samples between processors.)

We now need to introduce a few de£nitions relating the order of computation, communication delays between processes and input/output masks:

Definition 1

Define the mapping

$$f : (n_1, n_2) \rightarrow \nu; (n_1, n_2) \in \mathcal{S} \subset \mathcal{N}_0^2, \ \nu \in \{1, \ldots, C\} \quad (3)$$

which assigns one out of C processors to every output sample in the quarterplane. This function is called processor assignment function. Furthermore $f(n_1, n_2) = \nu$ for $(n_1, n_2) \in S_\nu, \nu = 1, \ldots, C$.

Definition 2

Define the index mapping function

$$I : (n_1, n_2) \rightarrow i; (n_1, n_2) \in \mathcal{S} \subset \mathcal{N}_0^2, \ i \in \mathcal{N}_0 \quad (4)$$

$I(n_1, n_2)$ symbolizes the time instant at which the sample $y(n_1, n_2)$ is computed (on machine $f(n_1, n_2)$).

Definition 3

Define the inter-process delay function

$$d : (\nu_1, \nu_2) \rightarrow i, \ i \in \mathcal{N}_0 \quad (5)$$
$$(\nu_1, \nu_2) \in \{1, 2, \ldots, C\} \times \{1, 2, \ldots, C\}$$

which expresses the maximum communication delay between processors ν_1 and ν_2, measured as an integer time instant. Furthermore $d(\nu, \nu) = 0$, since in this degenerate case, no communication between processors is necessary.

Comments:

- The above defined framework requires the assumption of "synchronized time" on all processors. Even though this assumption could be lifted, a tremendously complicated notation and analysis would result.

  All processors compute the difference equation (1) with the same speed. The time it takes to compute one sample is defined as and normalized to 1 according to Definition 2.

- Determining the maximum communication delay is a nontrivial task and depends on its definition. A statistical measure such as the 90 percentile delay mark could be used to determine d, even though d in practice is also a function of time.

- Unlike the single processor index mapping function $I(n_1, n_2)$ in [5], in the multiprocessor case, $I(n_1, n_2)$ does not need to be invertible. However, if we define I only over $S_i, i = 1, \ldots, C$ then $I(n_1, n_2)$ needs to be invertible due to the sequential computation of the outputs in S_i.

- If the response of system (1) is to be computed in parallel on C processors, and all processors are networked, in the most general case there need to be $(C - 1)C$ (directional) communication paths. (Only in the case where the number of processors is of the order of the cardinality of $\mathcal{M}_0$ or larger will it be possible to rely on locally connected processors.)

In-time recursive computability is then satisfied if for every $(n_1, n_2) \in \mathcal{S}$, the following inequality is satisfied:

$$I(n_1, n_2) > I(n_1 - i, n_2 - j) + d(f(n_1, n_2), f(n_1 - i, n_2 - j))$$

$$\forall (i, j) \in \mathcal{M}_0 \quad (6)$$

Of course, if the sample $y(n_1 - i, n_2 - j)$ was computed on the same processor as $y(n_1, n_2)$, then $f(n_1 - i, n_2 - j) = f(n_1, n_2) = \nu_0$ and by definition, $d(\nu_0, \nu_0) = 0$, which yields the

classical index mapping relationship for a (single processor implemented) partially ordered computation of 1st quadrant causal [5] sequences:

$$I(n_1, n_2) > I(n_1 - i, n_2 - j), \ \forall (i, j) \in \mathcal{M}_0 \quad (7)$$

Example: Consider the case $C = 4$ and $\mathcal{S} = \{(n_1, n_2)|0 \leq n_2 \leq 10, n_1 \geq 0\}$. Further assume $\mathcal{S}$ is partitioned as follows:

$$
\begin{aligned}
S_1 &= \{(n_1, n_2)|0 \leq n_2 \leq 10, n_1 = 0, 4, 8, \ldots\} \\
S_2 &= \{(n_1, n_2)|0 \leq n_2 \leq 10, n_1 = 1, 5, 9, \ldots\} \\
S_3 &= \{(n_1, n_2)|0 \leq n_2 \leq 10, n_1 = 2, 6, 10, \ldots\} \\
S_4 &= \{(n_1, n_2)|0 \leq n_2 \leq 10, n_1 = 3, 7, 11, \ldots\}
\end{aligned}
$$

The processor assignment function f is then defined as:

$$f(n_1, n_2) = \begin{cases} 1 & \text{for } (n_1, n_2) \in S_1 \\ 2 & \text{for } (n_1, n_2) \in S_2 \\ 3 & \text{for } (n_1, n_2) \in S_3 \\ 4 & \text{for } (n_1, n_2) \in S_4 \end{cases}$$

Let the first quadrant causal BIBO stable difference equation be given by:

$$
\begin{aligned}
y(n_1, n_2) &= a_{01}y(n_1, n_2 - 1) + a_{10}y(n_1 - 1, n_2) + \\
&\quad + a_{11}y(n_1 - 1, n_2 - 1) + b_{00}x(n_1, n_2).
\end{aligned}
$$

Assuming $d(f(n_1, n_2), f(n_1 - i, n_2 - j)) = 1$ for $(i, j) \in \{(1, 1), (1, 0)\}$ and zero otherwise, and defining the multi-processor index mapping function as

$$I(n_1, n_2) = \begin{cases} n_2 + 11n_1 & \text{for } (n_1, n_2) \in S_1 \\ n_2 + 2 + 11(n_1 - 1) & \text{for } (n_1, n_2) \in S_2 \\ n_2 + 4 + 11(n_1 - 2) & \text{for } (n_1, n_2) \in S_3 \\ n_2 + 6 + 11(n_1 - 3) & \text{for } (n_1, n_2) \in S_4 \end{cases}$$

we obtain a process for computing the $2-$D system response as shown in Figure 1.
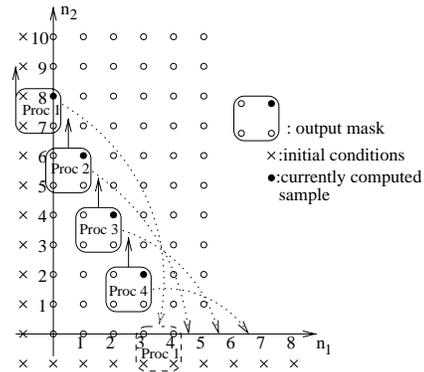

Figure 1: Order of computation using 4 processors.

As the example shows, large delay uncertainties, i.e. large upper bounds on $d(\cdot, \cdot)$, reduce the amount of parallelism that can be brought to bear, at least for a finite data set $\mathcal{S}$. Therefore, small and tight bounds on the interprocess communication delays are advantageous, which confirms previous results in distributed/parallel

computing [6]-[8]. Unfortunately, there are network types (such Ethernet and/or TCP/IP) that do not allow to construct an upper delay bound. If one still assumes a interprocess delay function d with an upper bound, this upper bound will be violated with a certain probability p. In order to ensure a parallel, synchronized, real time implementation of equation (1) despite the fact that no maximum delay exists, available outputs have to be substituted for the missing output samples. In order to illustrate that consider the output mask shown in Figure 2.
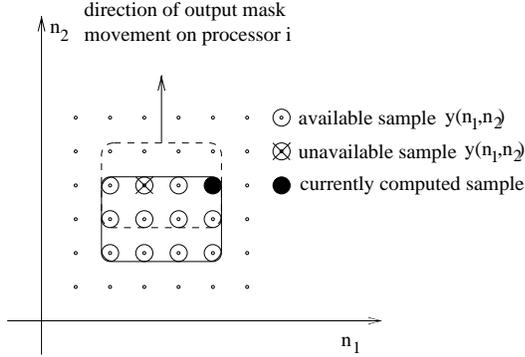

Figure 2: Relationship of available output samples between two consecutive mask positions.

Comparing the elements in the mask to compute $y(n_1, n_2)$ and $y(n_1, n_2 + 1)$, in Figure 2, the following can be observed: If a subset of $y(n_1 - i, n_2 - j)$ with $i, j \in \mathcal{M}_0$ is available for the computation of $y(n_1, n_2)$ then this will have implications on availability of $y(n_1 - i, n_2 + 1 - j)$; $i, j \in \mathcal{M}_0$ for the computation of $y(n_1, n_2 + 1)$. In other words, in the process of computing $y(n_1, n_2)$, $(n_1, n_2) \in S_i$, the available outputs needed to compute $y(n_1, n_2)$ cannot change arbitrarily when going from one point to the next. Hence we have:

If $y(n_1 - i, n_2 - j)$, $(i, j) \in \tilde{\mathcal{M}}_{n_1, n_2} \subset \mathcal{M}_0$ are known at time instant $I(n_1, n_2)$ at processor $f(n_1, n_2)$ and $f(n_1', n_2') = f(n_1, n_2)$, then $y(n_1' - i', n_2' - j')$ are known if $(n_1' - i', n_2' - j') \in \{(m_1, m_2) | (m_1, m_2) = (n_1 - i, n_2 - j), (i, j) \in \tilde{\mathcal{M}}_{n_1, n_2}\}$. $\tilde{\mathcal{M}}_{n_1, n_2}$ is the subset of the output mask index set $\mathcal{M}_0$ for which output values are known at (n_1, n_2). So $\tilde{\mathcal{M}}_{n_1, n_2}$ is the new output mask for computation of $y(n_1, n_2)$, resulting from missing output values in the original mask $\mathcal{M}_0$. Of course, the question that arises now is how can equation (1) be evaluated if $y(n_1 - i, n_2 - j)$, $(i, j) \in \mathcal{M}_0 \backslash \tilde{\mathcal{M}}_{n_1, n_2}$ are not available for computation.

Several solutions exist, all of which are based on computing $y(n_1 - i, n_2 - j)$, $(i, j) \in \mathcal{M}_0 \backslash \tilde{\mathcal{M}}_{n_1, n_2}$ from available values in $\tilde{\mathcal{M}}_{n_1, n_2}$. These methods range from interpolation, extrapolation, to simple substitution techniques using nearest neighbor concepts. For the remainder of this paper, we assume that some substitution technique is used since this has the advantage of a fast implementation and it results in a uncertainty set with £nite cardinality in the system theoretic description.

2.2. A Spatially Varying Roesser 2−D State Space Model

Alternatively to (1), a 2−D 1st quadrant causal system can also be described by the Roesser state space model [9]. (Even though there are a number of different 2−D state space models available,

we choose to use the Roesser model since results on the stability of shift varying Roesser models are already available [10].)

For a shift varying formulation we have:

$$\left[\begin{array}{c} \underline{x}^h(n_1 + 1, n_2) \\ \underline{x}^v(n_1, n_2 + 1) \end{array} \right] = A(n_1, n_2) \underline{x}(n_1, n_2)$$

$$= \left[\begin{array}{cc} A_{HH}(n_1, n_2) & A_{HV}(n_1, n_2) \\ A_{VH}(n_1, n_2) & A_{VV}(n_1, n_2) \end{array} \right] \left[\begin{array}{c} \underline{x}^h(n_1, n_2) \\ \underline{x}^v(n_1, n_2) \end{array} \right] \quad (8)$$

where $\underline{x}(n_1, n_2) \in \mathcal{R}^{(H+V)}$ is the state vector in a 2−D system of order H in n_1, and V in n_2. This vector may also be written as:

$$\underline{x}(n_1, n_2) = \left[\begin{array}{c} \underline{x}^h(n_1, n_2) \\ \underline{x}^v(n_1, n_2) \end{array} \right] \quad (9)$$

$A(n_1, n_2)$ is the uncertain system matrix for a shift-variant 2−D system, with

$$\begin{array}{cc} A_{HH}(n_1, n_2) \in \mathcal{R}^{H \times H} & A_{HV}(n_1, n_2) \in \mathcal{R}^{H \times V} \\ A_{VH}(n_1, n_2) \in \mathcal{R}^{V \times H} & A_{VV}(n_1, n_2) \in \mathcal{R}^{V \times V} \end{array}$$

The matrix $A(n_1, n_2)$ is taken from a set $\mathcal{A}$, that corresponds to the set of all matrices $A(n_1, n_2)$ arising from all possible incomplete output masks $\tilde{\mathcal{M}}_{n_1, n_2}$ [1], i.e. $A(n_1, n_2) \in \mathcal{A} \subset \mathcal{R}^{(H+V) \times (H+V)}$. Denote the cardinality of $\mathcal{A}$ with κ. This £nite set $\mathcal{A}$ describes all possible system matrices that arise from the nominal system matrix A by substituting one or more missing samples by available samples in $\mathcal{M}_0$. This results in a sparse matrix $A(n_1, n_2)$.

De£ne a decomposition [10] of $A(n_1, n_2)$ by:

$$J(n_1, n_2) = \left[\begin{array}{cc} A_{HH}(n_1, n_2) & A_{HV}(n_1, n_2) \\ 0 & 0 \end{array} \right] \quad (10)$$

$$K(n_1, n_2) = \left[\begin{array}{cc} 0 & 0 \\ A_{VH}(n_1, n_2) & A_{VV}(n_1, n_2) \end{array} \right] \quad (11)$$

De£ne the variable dimension supermatrix [10] by

$$A_{cc}(n) = \left[\begin{array}{ccccc} J(n, 0) & 0 & \cdots & 0 \\ K(n, 0) & J(n-1, 1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J(0, n) \\ 0 & 0 & \cdots & K(0, n) \end{array} \right] \quad (12)$$

Note that $A_{cc}(n) \in \mathcal{R}^{(n+2)(H+V) \times (n+1)(H+V)}$. Let $A_{i,cc}(n)$, $i = 1, \ldots, \kappa^{n+1}$ be an element of the set of all possible supermatrices $\mathcal{A}_{cc}(n)$ at time instant n. The system supermatrix $\mathcal{A}_{cc}(n)$ describes the 2−D process as a 1−D process with increasing dimensionality. This is achieved by relating the state along two consecutive diagonals, i.e. for $n_1 + n_2 = n$ and $n_1 + n_2 = n + 1$.

2.3. Stability

Using equation (12), the dynamics of the 2−D process under zero input conditions are uniquely de£ned using the sequence of supermatrices $A_{cc}(0)$, $A_{cc}(1), \ldots, A_{cc}(n), \ldots$ where each $A_{cc}(n) \in \mathcal{A}_{cc}(n)$. Due to the fact that $\tilde{\mathcal{M}}_{n_1, n_2} \subset \mathcal{M}_0$ and $\tilde{\mathcal{M}}_{n_1, n_2}$ for $n_1 + n_2 = c$ and $\tilde{\mathcal{M}}_{n_1', n_2'}$, $n_1' + n_2' = c + 1$ are related (by using

---

[1]We are not considering the case where the incomplete output mask needs to be chosen such that $\tilde{\mathcal{M}}_{n_1, n_2} \not\subset \mathcal{M}$. This requires non minimal realizations and only arises in network connections that frequently violate the delay bound, possibly due to loss of data.

some substitution technique for the missing samples) the supermatrices $A_{cc}(n)$ and $A_{cc}(n+1)$ are also related. In particular:

If $A_{cc}(n) = A_{i,cc}(n)$ then

$$A_{cc}(n+1) \in \mathcal{A}_{cc}^{(i)}(n+1) \tag{13}$$

where $\mathcal{A}_{cc}^{(i)}(n+1)$ is a non empty subset of $\mathcal{A}_{cc}(n+1)$ that depends solely on $A_{cc}(n) = A_{i,cc}(n)$. Hence given $A_{cc}(n)$, the cardinality of the uncertainty set for $A_{cc}(n+1)$ is reduced.

Now we can formulate the main Theorem on stability.

Theorem:

The system in (8) with spatially varying uncertainties as described in (13) is asymptotically stable, if there exists a £nite n such that:

$$\max \| A_{cc}(n) \cdot A_{cc}(n-1) \cdot \ldots \cdot A_{cc}(0) \|_1 \leq \gamma < 1$$

where $A_{cc}(j+1) \in \mathcal{A}_{cc}^{(i)}(j+1)$ if $A_{cc}(j) = A_{i,cc}(j)$.

Proof: The proof follows directly from the main Theorem in [10] with minor modi£cations.

Unlike the $1-D$ case, the theorem only holds for the $1-$norm and necessity cannot be shown. Unfortunately, the computational complexity of Theorem 1 is NP, even when using relationship (13).

3. CONCLUSION

This paper investigates the effect of communication delays (and possibly lost samples) on the stability of $2-D$ systems, if the $2-D$ system is implemented as a distributed process via a communication network. A distributed system model as well as stability results are presented. Generalizations to the $m-D$ case are straightforward, even though computationally complexity for the stability check is expected to become prohibitively high. It is expected that the stability condition in the Theorem provides only conservative results since it allows for the occurrence of missing output samples at every cycle. Stochastic or additive error driven models will probably yield less restrictive results.

4. REFERENCES

[1] P. H. Bauer, M. L. Sichitiu, and K. Premaratne, "Closing the loop through communication networks: The case of an integrator plant and multiple controllers," in *Proc. 38th IEEE CDC*, pp. 2180–2185, Dec. 1999.

[2] P. H. Bauer, M. L. Sichitiu, and K. Premaratne, "Controlling an integrator through data networks: Stability in the presence of unknown time-variant delays," in *Proc. 1999 IEEE ISCAS*, vol. 5, pp. 491–494, May 1999.

[3] C. Lorand, M. L. Sichitiu, P. H. Bauer, and G. Schmidt, "Stability of £rst order discrete time systems with time-variant communication delays in the feedback path," in *IEEE APC-CAS*, pp. 283–287, Dec. 2000.

[4] E. I. Jury, *Multidimensional Systems: Techniques and Applications*, ch. 3. Stability of multidimensional systems and related problems, New York: Marcel Dekker, Inc., 1986.

[5] M. Manry and J. K. Aggarwal, "Picture processing using one-dimensional implementations of discrete planar £lters," *IEEE Trans. on ASSP*, vol. 22, pp. 164–173, June 1974.

[6] A. Agarwal, "Limits on interconnection network performance," *IEEE Trans. on Parallel and Distributed Systems*, vol. 2, pp. 398–412, Oct. 1991.

[7] M. Lin, J. Hsieh, D. H. Du, J. P. Thomas, and J. A. MacDonald, "Distributed network computing over local atm networks," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 733–748, May 1995.

[8] B. K. Schmidt and V. S. Sunderam, "Empirical analysis of overheads in cluster environments," *Concurrency Practice and Experience*, vol. 6, pp. 1–32, Feb. 1994.

[9] R. P. Roesser, "Discrete state-space model for linear image processing," *IEEE Trans. on Automatic Control*, vol. 20, pp. 1–10, Feb. 1975.

[10] P. Bauer and E. Jury, "A stability analysis of two-dimensional nonlinear digital state-space £lters," *IEEE Trans. on ASSP*, vol. 38, pp. 1578–1586, Sept. 1990.