International Journal of Pure and Applied Mathematics
Volume 70 No. 3 2011, 329-347

WHY PARALLEL COMPUTER PROCESSING SYSTEMS ARE PREFERRED TO SERIAL COMPUTER PROCESSING SYSTEMS: A FORMAL DISCUSSION

Zvi Retchkiman Konigsberg
Instituto Politécnico Nacional
CIC – Centro de Investigacion en Computacion
Mineria 17-2, Col. Escandon, Mexico D.F., 11800, MEXICO

Abstract: Serial computer processing systems are characterized by the fact of executing software using a single central processing unit (CPU) while parallel computer processing systems simultaneously use multiple CPU's at the time.

Parallel computer processing systems are an evolution of serial computer processing systems that attempt to emulate what has always been the state of affairs in the natural world: many complex, interrelated events happening at the same time, yet within a sequence. Today, commercial applications provide an equal or greater driving force in the development of faster computers. These applications require the processing of large amounts of data. Some of the arguments which have been used to say why it is better parallel than serial are: save time and/or money, solve larger problems. Besides that there are physical and economical limits to serial computer processing systems. However we would like to be more precise and give a definitive and unquestionable formal proof to justify the claim that parallel computer processing systems are better than serial processing systems. The main objective and contribution of this paper consists in using a formal and mathematical approach to prove that parallel computer processing systems are better than serial computer processing systems (better related to: saving time and/or money and being able to solve larger problems). This is achieved thanks to the theory of Lyapunov stability and max-plus algebra applied to discrete event systems modeled with time Petri nets.

AMS Subject Classification: 39A30, 15A80, 34H15
Key Words: parallel and serial computer processing systems, Lyapunov methods, max-plus algebra, timed Petri nets

Received: January 15, 2011

1. Introduction

Serial computer processing systems are characterized by the fact of executing software using a single central processing unit (CPU) while parallel computer processing systems simultaneously use multiple CPU's at the time. Parallel computer processing systems are an evolution of serial computer processing systems that attempt to emulate what has always been the state of affairs in the natural world: many complex, interrelated events happening at the same time, yet within a sequence. Today, commercial applications provide an equal or greater driving force in the development of faster computers. These applications require the processing of large amounts of data. Some of the arguments which have been used to say why it is better parallel than serial are:

1. Save time and/or money: In theory, throwing more resources at a task will shorten its time to completion, with potential cost savings.

2. Solve larger problems: Many problems are so large and/or complex that it is impractical or impossible to solve them on a single computer, especially given limited computer memory.

Besides that there are limits to serial processing computer systems:

1. Transmission speeds: The speed of a serial computer systems is directly dependent upon how fast data can move through hardware. Absolute limits are the speed of light (30 cm/nanosecond) and the transmission limit of copper wire (9 cm/nanosecond). Increasing speeds necessitate increasing proximity of processing elements.

2. Limits to miniaturization: Processor technology is allowing an increasing number of transistors to be placed on a chip. However, even with molecular or atomic-level components, a limit will be reached on how small components can be.

3. Economic limitations: It is increasingly expensive to make a single processor faster. Using a larger number of moderately fast commodity processors to achieve the same (or better) performance is less expensive.

However we would like to be more precise and give a definitive and unquestionable formal proof to justify the claim that parallel computer processing systems are better than serial computer processing systems.

In some way, Amdahl's law [1], which is a model for the relationship between the expected speedup of parallelized implementations of an algorithm relative to the serial algorithm, does it however, its derivation is based on observations and plausible arguments.

The main objective and contribution of this paper consists in using a formal and mathematical approach to prove that parallel computer processing systems are better than serial computer processing systems (better related to: saving time and/or money and being able to solve larger problems). This is achieved thanks to the theory of Lyapunov stability and max-plus algebra applied to discrete event systems modeled with time Petri nets. The paper is organized as follows. In Section 2, Lyapunov theory for discrete event systems modeled with Petri nets is addressed. Section 3, presents max-plus algebra. In Section 4, the solution to the stability problem for discrete event systems modeled with timed Petri nets using a Lyapunov, max-plus algebra approach is given. Section 5, applies the theory presented in the previous sections to formally prove that parallel computer processing systems are better than serial computer processing systems. Finally, the paper ends with some conclusions.

2. Lyapunov Stability and Stabilization of Discrete Event Systems modeled with Petri Nets (see [2, 3, 4])

The solution to the stability problem for discrete event systems, whose model is obtained employing timed Petri nets, is achieved thanks to the theory of vector Lyapunov functions and comparison principles. The methodology shows that it is possible to restrict the systems state space in such a way that boundedness is guaranteed.

Notation. $N = \{0, 1, 2, ...\}$, $R_+ = [0, \infty)$, $N_{n_0}^+ = \{n_0, n_0 + 1, ..., n_0 + k, ...\}$, $n_0 \geq 0$. Given $x, y \in R^n$, we usually denote the relation "$\leq$" to mean componentwise inequalities with the same relation, i.e., $x \leq y$ is equivalent to $x_i \leq y_i, \forall i$. A function $f(n, x)$, $f : N_{n_0}^+ \times R^n \to R^n$ is called nondecreasing in x if given $x, y \in R^n$ such that $x \geq y$ and $n \in N_{n_0}^+$ then, $f(n, x) \geq f(n, y)$.

Consider systems of first ordinary difference equations given by

$$x(n+1) = f[n, x(n)], \quad x(n_o) = x_0, \quad n \in N_{n_0}^+, \tag{1}$$

where $n \in N_{n_0}^+$, $x(n) \in R^n$ and $f : N_{n_0}^+ \times R^n \to R^n$ is continuous in $x(n)$.

Definition 1. The n vector valued function $\Phi(n, n_0, x_0)$ is said to be a solution of (1) if $\Phi(n_0, n_0, x_0) = x_0$ and $\Phi(n+1, n_0, x_0) = f(n, \Phi(n, n_0, x_0))$ for all $n \in N_{n_0}^+$.

Definition 2. The system (1) is said to be
i). Practically stable, if given (λ, A) with $0 < \lambda < A$, then

$$|x_0| < \lambda \Rightarrow |x(n, n_0, x_0)| < A, \forall n \in N^+_{n_0}, n_0 \geq 0;$$

ii). Uniformly practically stable, if it is practically stable for every $n_0 \geq 0$.

The following class of function is defined.

Definition 3. A continuous function $\alpha : [0, \infty) \to [0, \infty)$ is said to belong to class $\mathcal{K}$ if $\alpha(0) = 0$ and it is strictly increasing.

Consider a vector Lyapunov function $v(n, x(n))$, $v : N^+_{n_0} \times R^n \to R^p_+$ and define the variation of v relative to (1) by

$$\Delta v = v(n+1, x(n+1)) - v(n, x(n)) \tag{2}$$

Then, the following result concerns the practical stability of (1).

Theorem 4. *Let $v : N^+_{n_0} \times R^n \to R^p_+$ be a continuous function in x, define the function $v_0(n, x(n)) = \sum_{i=1}^{p} v_i(n, x(n))$ such that satisfies the estimates*

$$b(|x|) \leq v_0\left(n, x\left(n\right)\right) \leq a(|x|) \quad \textit{for } a, b \in \mathcal{K} \quad \textit{and}$$

$$\Delta v(n, x(n)) \leq w(n, v(n, x(n))),$$

for $n \in N^+_{n_0}$, $x(n) \in R^n$, where $w : N^+_{n_0} \times R^p_+ \to R^p$ is a continuous function in the second argument.

Assume that: $g(n, e) \triangleq e + w(n, e)$ is nondecreasing in e, $0 < \lambda < A$ are given and finally that $a(\lambda) < b(A)$ is satisfied.

Then, the practical stability properties of

$$e(n+1) = g(n, e(n)), \quad e(n_0) = e_0 \geq 0 \tag{3}$$

imply the practical stability properties of system (1).

Corollary 5. *In Theorem (4):*
i) If $w(n, e) \equiv 0$ we get uniform practical stability of (1) which implies structural stability.
ii) If $w(n, e) = -c(e)$, for $c \in \mathcal{K}$, we get uniform practical asymptotic stability of (1).

Definition 6. A Petri net is a 5-tuple, $PN = \{P, T, F, W, M_0\}$ where:
$P = \{p_1, p_2, ..., p_m\}$is a finite set of places,
$T = \{t_1, t_2, ..., t_n\}$ is a finite set of transitions,

$F \subset (P \times T) \cup (T \times P)$ is a set of arcs,
$W : F \to N_1^+$ is a weight function,
M_0: $P \to N$ is the initial marking,
$P \cap T = \varnothing$ and $P \cup T \neq \varnothing$.

Definition 7. The clock structure associated with a place $p_i \in P$ is a set $\mathbf{V} = \{V_i : p_i \in P\}$ of clock sequences $V_i = \{v_{i,1}, v_{i,2}, ...\}$, $v_{i,k} \in R^+$, $k = 1, 2, ...$

The positive number $v_{i,k}$, associated to $p_i \in P$, called holding time, represents the time that a token must spend in this place until its outputs enabled transitions $t_{i,1}, t_{i,2}, ...,$ fire. Some places may have a zero holding time while others not. Thus, we partition P into subsets P_0 and P_h, where P_0 is the set of places with zero holding time, and P_h is the set of places that have some holding time.

Definition 8. A timed Petri net is a 6-tuple $TPN = \{P, T, F, W, M_0, \mathbf{V}\}$ where $\{P, T, F, W, M_0\}$ are as before, and $\mathbf{V} = \{V_i : p_i \in P\}$ is a clock structure. A timed Petri net is a timed event petri net when every $p_i \in P$ has one input and one output transition, in which case the associated clock structure set of a place $p_i \in P$ reduces to one element $V_i = \{v_i\}$

A PN structure without any specific initial marking is denoted by N. A Petri net with the given initial marking is denoted by (N, M_0). Notice that if $W(p,t) = \alpha$ (or $W(t,p) = \beta$) then, this is often represented graphically by α, (β) arcs from p to t (t to p) each with no numeric label.

Let $M_k(p_i)$ denote the marking (i.e., the number of tokens) at place $p_i \in P$ at time k and let $M_k = [M_k(p_1), ..., M_k(p_m)]^T$ denote the marking (state) of PN at time k. A transition $t_j \in T$ is said to be enabled at time k if $M_k(p_i) \geq W(p_i, t_j)$ for all $p_i \in P$ such that $(p_i, t_j) \in F$. It is assumed that at each time k there exists at least one transition to fire. If a transition is enabled then, it can fire. If an enabled transition $t_j \in T$ fires at time k then, the next marking for $p_i \in P$ is given by

$$M_{k+1}(p_i) = M_k(p_i) + W(t_j, p_i) - W(p_i, t_j). \tag{4}$$

Let $A = [a_{ij}]$ denote an $n \times m$ matrix of integers (the incidence matrix) where $a_{ij} = a_{ij}^+ - a_{ij}^-$ with $a_{ij}^+ = W(t_i, p_j)$ and $a_{ij}^- = W(p_j, t_i)$. Let $u_k \in \{0,1\}^n$ denote a firing vector where if $t_j \in T$ is fired then, its corresponding firing vector is $u_k = [0, ..., 0, 1, 0, ..., 0]^T$ with the one in the j^{th} position in the vector and zeros everywhere else. The matrix equation (nonlinear difference equation) describing the dynamical behavior represented by a PN is:

$$M_{k+1} = M_k + A^T u_k \tag{5}$$

where if at step k, $a_{ij}^{-} < M_k(p_j)$ for all $p_i \in P$ then, $t_i \in T$ is enabled and if this $t_i \in T$ fires then, its corresponding firing vector u_k is utilized in the difference equation to generate the next step. Notice that if M' can be reached from some other marking M and, if we fire some sequence of d transitions with corresponding firing vectors $u_0, u_1, ..., u_{d-1}$ we obtain that

$$M' = M + A^T u, \ u = \sum_{k=0}^{d-1} u_k. \tag{6}$$

Let $(N_{n_0}^m, d)$ be a metric space where $d : N_{n_0}^m \times N_{n_0}^m \to R_+$ is defined by

$$d(M_1, M_2) = \sum_{i=1}^{m} \zeta_i \mid M_1(p_i) - M_2(p_i) \mid ; \zeta_i > 0$$

and consider the matrix difference equation which describes the dynamical behavior of the discrete event system modeled by a PN

$$M' = M + A^T u, \ u = \sum_{k=0}^{d-1} u_k, \tag{7}$$

where $M \in N^m$, denotes the marking (state) of the PN, $A \in Z^{n \times m}$, its incidence matrix and $u \in N^n$, is a sequence of firing vectors. Then, the following results concerns in what to the stability problem means.

Proposition 9. *Let PN be a Petri net. PN is uniform practical stable if there exists a Φ strictly positive m vector such that*

$$\Delta v = u^T A \Phi \leq 0 \tag{8}$$

Moreover, PN is uniform practical asymptotic stable if the following equation holds

$$\Delta v = u^T A \Phi \leq -c(e), \text{ for } c \in \mathcal{K} \tag{9}$$

Lemma 10. *Let suppose that Proposition (9) holds then,*

$$\Delta v = u^T A \Phi \leq 0 \Leftrightarrow A \Phi \leq 0 \tag{10}$$

Remark 11. Notice that since the state space of a TPN is contained in the state space of the same now not timed PN, stability of PN implies stability of the TPN.

2.1. Lyapunov Stabilization

Notice, that in the solution of the stability problem, the u vector does not play any role, so why not to take advantage of it in order to get some specific behavior.

Consider the matrix difference equation which describes the dynamical behavior of the discrete event system modeled by a Petri net

$$M' = M + A^T u.$$

We are interested in finding a firing sequence vector, control law, such that system (7) remains bounded.

Definition 12. Let PN be a Petri net. PN is said to be stabilizable if there exists a firing transition sequence with transition count vector u such that system (7) remains bounded.

Proposition 13. *Let PN be a Petri net. PN is stabilizable if there exists a firing transition sequence with transition count vector u such that the following equation holds*

$$\Delta v = A^T u \leq 0 \tag{11}$$

Remark 14. It is important to underline that by fixing a particular u, which satisfies (11), we restrict the state space to those markings (states) that are finite. The technique can be utilized to get some type of regulation and/or eliminate some undesirable events (transitions).

3. Max-Plus Algebra, see [5, 6]

In this section the concept of max-plus algebra is defined. Its algebraic structure is described. Matrices and graphs are presented. The spectral theory of matrices is discussed. The problem of solving linear equations is addressed. Finally, max-plus recurrence equations for timed Petri nets are introduced.

3.1. Basic Definitions

Notation. $\mathbb{N}$ is the set of natural numbers, $\mathbb{R}$ is the set of real numbers, $\epsilon = -\infty$, $e = 0$, $\mathbb{R}_{max} = \mathbb{R} \cup \{\epsilon\}$, $\underline{n} = 1, 2, ..., n$

Let $a, b \in \mathbb{R}_{max}$ and define the operations $\oplus$ and $\otimes$ by: $a \oplus b = \max(a, b)$ and $a \otimes b = a + b$.

Definition 15. The set $\mathbb{R}_{max}$ with the two operations $\oplus$ and $\otimes$ is called a max-plus algebra and is denoted by $\Re_{\max} = (\mathbb{R}_{max}, \oplus, \otimes, \epsilon, e)$.

Definition 16. A semiring is a nonempty set R endowed with two operations $\oplus_R$, $\otimes_R$, and two elements ϵ_R and e_R such that: $\oplus_R$ is associative and commutative with zero element ϵ_R, $\otimes_R$ is associative, distributes over $\oplus_R$, and has unit element e_R, $\in_R$is absorbing for $\otimes_R$ i.e., $a \otimes_R \epsilon = \epsilon_R \otimes a = a$, $\forall a \in R$..

Such a semiring is denoted by $\Re = (R, \oplus_R, \otimes_R, \epsilon, e)$. In addition if $\otimes_R$ is commutative then R is called a commutative semiring , and if $\oplus_R$ is such that $a \oplus_R a = a$, $\forall a \in R$ then it is called idempotent.

Theorem 17. *The max-plus algebra $\Re_{\max} = (\mathbb{R}_{max}, \oplus, \otimes, \epsilon, e)$ has the algebraic structure of a commutative and idempotent semiring.*

3.2. Matrices and Graphs

Let $\mathbb{R}_{max}^{n\times n}$ be the set of $n \times n$ matrices with coefficients in $\mathbb{R}_{max}$ with the following operations: The sum of matrices $A, B \in \mathbb{R}_{max}^{n\times n}$, denoted $A \oplus B$ is defined by: $(A \oplus B)_{ij} = a_{ij} \oplus b_{ij} = \max\,(a_{ij}, b_{ij})$ for i and $j \in \underline{n}$. The product of matrices $A \in \mathbb{R}_{max}^{n\times l}, B \in \mathbb{R}_{max}^{l\times n}$, denoted $A \otimes B$ is defined by: $(A \otimes B)_{ik} = \bigotimes_{j=1}^{l} a_{ij} \otimes b_{jk} = \max_{j\in \underline{l}} \{a_{ij} + b_{jk}\}$ for i and $k \in \underline{n}$.

Let $\mathcal{E} \in \mathbb{R}_{max}^{n\times n}$ denote the matrix with all its elements equal to ϵ and denote by $E \in \mathbb{R}_{max}^{n\times n}$ the matrix which has its diagonal elements equal to e and all the other elements equal to ϵ. Then, the following result can be stated.

Theorem 18. *The 5-tuple $\Re_{\max}^{n\times n} = (\mathbb{R}_{max}^{n\times n}, \oplus, \otimes, \mathcal{E}, E)$ has the algebraic structure of a noncommutative idempotent semiring.*

Definition 19. Let $A \in \mathbb{R}_{max}^{n\times n}$ and $k \in \mathbb{N}$ then the k-th power of A denoted by $A^{\otimes k}$ is defined by: $A^{\otimes k} = \underbrace{A \otimes A \otimes \cdots \otimes A}_{k-\text{times}}$, where $A^{\otimes 0}$ is set equal to E.

Definition 20. A matrix $A \in \mathbb{R}_{max}^{n\times n}$ is said to be regular if A contains at least one element distinct from ϵ in each row.

Definition 21. Let $\mathcal{N}$ be a finite and non-empty set and consider $\mathcal{D} \subseteq \mathcal{N} \times \mathcal{N}$. The pair $G = (\mathcal{N}, \mathcal{D})$ is called a directed graph, where $\mathcal{N}$ is the set of elements called nodes and $\mathcal{D}$ is the set of ordered pairs of nodes called arcs. A directed graph $G = (\mathcal{N}, \mathcal{D})$ is called a weighted graph if a weight $w(i, j) \in \mathbb{R}$ is associated with any arc $(i,j) \in \mathcal{D}$.

Let $A \in \mathbb{R}_{max}^{n\times n}$ be any matrix, a graph $\mathcal{G}(A)$, called the communication graph of A, can be associated as follows. Define $\mathcal{N}(A) = \underline{n}$ and a pair $(i,j) \in \underline{n} \times \underline{n}$ will be a member of $\mathcal{D}(A) \Leftrightarrow a_{ji} \neq \epsilon$, where $\mathcal{D}(A)$ denotes the set of arcs of $\mathcal{G}(A)$.

Definition 22. A path from node i to node j is a sequence of arcs $p = \{(i_k, j_k) \in \mathcal{D}(A)\}_{k\in\underline{m}}$ such that $i = i_1, j_k = i_{k+1}$, for $k < m$ and $j_m = j$. The path p consists of the nodes $i = i_1, i_2, ..., i_m, j_m = j$ with length m denoted by $\mid p \mid_1 = m$. In the case when $i = j$ the path is said to be a circuit. A circuit is said to be elementary if nodes i_k and i_l are different for $k \neq l$. A circuit consisting of one arc is called a self-loop.

Let us denote by $P(i,j;m)$ the set of all paths from node i to node j of length $m \geq 1$ and for any arc $(i,j) \in \mathcal{D}(A)$ let its weight be given by a_{ij} then the weight of a path $p \in P(i,j;m)$ denoted by $\mid p \mid_w$ is defined to be the sum of the weights of all the arcs that belong to the path. The average weight of a path p is given by $\mid p \mid_w / \mid p \mid_1$. Given two paths, as for example, $p = ((i_1,i_2),(i_2,i_3))$ and $q = ((i_3,i_4),((i_4,i_5)$ in $\mathcal{G}(A)$ the concatenation of paths $\circ : \mathcal{G}(A) \times \mathcal{G}(A) \rightarrow \mathcal{G}(A)$ is defined as $p \circ q = ((i_1,i_2),(i_2,i_3),(i_3,i_4),(i_4,i_5))$.

The communication graph $\mathcal{G}(A)$ and powers of matrix A are closely related as it is shown in the next theorem.

Theorem 23. *Let $A \in \mathbb{R}_{max}^{n\times n}$, then $\forall k \geq 1$: $[A^{\otimes k}]_{ji} = max\{\mid p \mid_w : p \in P(i,j;k)\}$, where $[A^{\otimes k}]_{ji} = \epsilon$ in the case when $P(i,j;k)$ is empty i.e., no path of length k from node i to node j exists in $\mathcal{G}(A)$.*

Definition 24. Let $A \in \mathbb{R}_{max}^{n\times n}$ then define the matrix $A^+ \in \mathbb{R}_{max}^{n\times n}$ as: $A^+ = \bigoplus_{k=1}^{\infty} A^{\otimes k}$. Where the element $[A^+]_{ji}$ gives the maximal weight of any path from j to i. If in addition one wants to add the possibility of staying at a node then one must include matrix E in the definition of matrix A^+ giving rise to its Kleene star representation defined by:

$$A^* = \bigoplus_{k=0}^{\infty} A^{\otimes k}. \tag{12}$$

Lemma 25. *Let $A \in \mathbb{R}_{max}^{n\times n}$ be such that any circuit in $\mathcal{G}(A)$ has average circuit weight less than or equal to ϵ. Then it holds that:*

$$A^* = \bigoplus_{k=0}^{n-1} A^{\otimes k}. \tag{13}$$

Definition 26. Let $G = (\mathcal{N}, \mathcal{D})$ be a graph and $i, j \in \mathcal{N}$, node j is reachable from node i, denoted as $i\mathcal{R}j$, if there exists a path from i to j. A graph G is said to be strongly connected if $\forall i, j \in \mathcal{N}, j\mathcal{R}i$. A matrix $A \in \mathbb{R}^{n\times n}_{max}$ is called irreducible if its communication graph is strongly connected, when this is not the case matrix A is called reducible.

Remark 27. In this paper irreducible matrices are just considered. It is possible to treat the reducible case by transforming it into its normal form and computing its generalized eigenmode, see [4].

3.2.1. Spectral Theory

Definition 28. Let $A \in \mathbb{R}^{n\times n}_{max}$ be a matrix. If $\mu \in R_{max}$ is a scalar and $v \in R^{n}_{max}$ is a vector that contains at least one finite element such that:

$$A \otimes v = \mu \otimes v \tag{14}$$

then, μ is called an eigenvalue and v an eigenvector.

Remark 29. Notice that the eigenvalue can be equal to ϵ and is not necessarily unique. Eigenvectors are certainly not unique indeed, if v is an eigenvector then $\alpha \otimes v$ is also an eigenvector for all $\alpha \in \mathbb{R}$.

Let $\mathcal{C}(A)$ denote the set of all elementary circuits in $\mathcal{G}(A)$ and write:

$$\lambda = \max_{p\in\mathcal{C}(A)} \frac{|p|_w}{|p|_1}, \tag{15}$$

for the maximal average circuit weight. Notice that since $\mathcal{C}(A)$ is a finite set, the maximum of (15) is attained (which is always the case when matrix A is irreducible). In case $\mathcal{C}(A) = \emptyset$ define $\lambda = \epsilon$.

Definition 30. A circuit $p \in G(A)$ is said to be critical if its average weight is maximal. The critical graph of A, denoted by $G^c(A) = (\mathcal{N}^c(A), \mathcal{D}^c(A))$, is the graph consisting of those nodes and arcs that belong to critical circuits in $G(A)$.

Lemma 31. *Let assume that $G(A)$ contains at least one circuit then, any circuit in $G^c(A)$ is critical.*

Definition 32. Let $A \in \mathbb{R}^{n\times n}_{max}$ be a matrix and μ an eigenvalue of A with associated eigenvector v then, the support of v consists of the set of nodes of $G(A)$ which correspond to finite entries of v.

Lemma 33. *Let $A \in \mathbb{R}_{max}^{n\times n}$ be an irreducible matrix then any $v \in R_{max}^{n}$ which satisfies (14) has all components different from ϵ.*

Next, the most important result of this sub-section is given.

Theorem 34. *If $A \in \mathbb{R}_{max}^{n\times n}$ is irreducible, then there exists one and only one finite eigenvalue (with possible several eigenvectors). This eigenvalue is equal to the maximal average weight of circuits in $G(A)$:*

$$\lambda(A) = \max_{p\in\mathcal{C}(A)} \frac{|p|_w}{|p|_1}. \tag{16}$$

3.2.2. Linear Equations

Theorem 35. *Let $A \in \mathbb{R}_{max}^{n\times n}$ and $b \in \mathbb{R}_{max}^{n}$. If the communication graph $G(A)$ has maximal average circuit weight less than or equal to e, then $x = A^* \otimes b$ solves the equation $x = (A \otimes x) \oplus b$. Moreover, if the circuit weights in $G(a)$ are negative then, the solution is unique.*

3.3. Max-Plus Recurrence Equations for Timed Event Petri Nets

Definition 36. Let $A_m \in \mathbb{R}_{max}^{n\times n}$ for $0 \le m \le M$ and $x(m) \in \mathbb{R}_{max}^{n}$ for $-M \le m \le -1$; $M \ge 0$. Then, the recurrence equation: $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \ge 0$ is called an Mth order recurrence equation.

Theorem 37. *The Mth order recurrence equation, given by equation $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \ge 0$, can be transformed into a first order recurrence equation $x(k+1) = A \otimes x(k)$; $k \ge 0$ provided that A_0 has circuit weights less than or equal to zero.*

With any timed event Petri net, matrices $A_0, A_1, ..., A_M \in \mathbb{N}^n \times \mathbb{N}^n$ can be defined by setting $[A_m]_{jl} = a_{jl}$, where a_{jl} is the largest of the holding times with respect to all places between transitions t_l and t_j with m tokens, for $m = 0, 1, ..., M$, with M equal to the maximum number of tokens with respect to all places. Let $x_i(k)$ denote the kth time that transition t_i fires, then the vector $x(k) = (x_1(k), x_2(k), ...x_m(k))^T$, called the state of the system, satisfies the Mth order recurrence equation: $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m)$; $k \ge 0$

Now, assuming that all the hypothesis of theorem (37) are satisfied, and setting $\hat{x}(k) = (x^T(k), x^T(k-1), ..., x^T(k-M+1))^T$, equation $x(k) = \bigoplus_{m=0}^{M} A_m \otimes x(k-m);\ k \geq 0$ can be expressed as: $\hat{x}(k+1) = \hat{A} \otimes \hat{x}(k);\ k \geq 0$, which is known as the standard autonomous equation.

4. The Solution to the Stability Problem for Discrete Event Systems Modeled with Timed Petri Nets

This section defines what it means for a TPN to be stable, then gathering the results previously presented in the past sections the solution to the problem is obtained.

Definition 38. A TPN is said to be stable if all the transitions fire with the same proportion i.e., if there exists $q \in \mathbb{N}$ such that

$$\lim_{k\to\infty} \frac{x_i(k)}{k} = q, \forall i = 1, ..., n \tag{17}$$

This last definition tell us that in order to obtain a stable TPN all the transitions have to be fired q times. However, it will be desirable to be more precise and know exactly how many times. The answer to this question is given next.

Lemma 39. *Consider the recurrence relation* $x(k+1) = A \otimes x(k), k \geq 0$, $x(0) = x_0 \in \mathbb{R}^n$ *arbitrary.* A *an irreducible matrix and* $\lambda \in \mathbb{R}$ *its eigenvalue then,*

$$\lim_{k\to\infty} \frac{x_i(k)}{k} = \lambda, \forall i = 1, ..., n \tag{18}$$

Proof. Let v be an eigenvector of A such that $x_0 = v$ then,

$$x(k) = \lambda^{\otimes k} \otimes v \Rightarrow x(k) = k\lambda + v \Rightarrow \frac{x(k)}{k} = \lambda + \frac{v}{k} \Rightarrow \lim_{k\to\infty} \frac{x_i(k)}{k} = \lambda. \quad \square$$

Now starting with an unstable TPN, collecting the results given by: proposition (13), what has just been discussed about recurrence equations for TPN at the end of subsection 3.3 and the previous lemma (39) plus theorem (34), the solution to the problem is obtained.

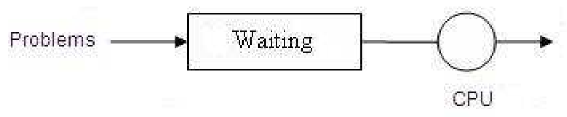

Figure 1: Serial computer processing system

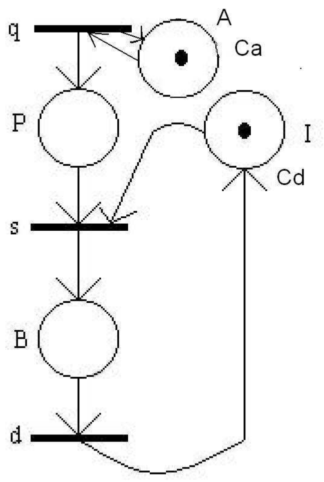

Figure 2: Timed Petri net model

5. Parallel vs Serial Computer Processing Systems

In this section, the main objective of this manuscript which consists in giving a precise and definitive answer to the question why are parallel computer processing systems preferred to serial computer processing systems (better related to: saving time and/or money and being able to solve larger problems), is presented. In Subsections 5.1 and 5.2, both cases are analyzed separately, arriving to some conclusions which are summarized at the end of them then, in Subsection 5.3 both cases are compared.

5.1. Serial Computer Processing System

Consider a serial computer processing system (Fig 1.) whose TPN model is depicted in Fig 2. Where the events (transitions) that drive the system are: q: a problem of size Ca has to be solved, s: the problem starts being executed by the CPU, d: the problem has been solved. The places (that represent the states of the serial computer processing system) are: A: problems loading, P: the problems are waiting for a solution, B: the problem is being solved, I: the CPU of capacity Cd is idle. The holding times associated to the places A and I are Ca and Cd respectively, (with $Ca > Cd$).

Remark 40. Notice that Ca, the size of q, is the time it takes to a problem until is completely loaded in the computer in order to be solved, larger problems will have larger Ca' s, while Cd, the capacity of the CPU, is the time it takes to the CPU to reset.

The incidence matrix that represents the PN model is

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}.$$

Therefore since there does not exists a Φ strictly positive m vector such that $A\Phi \leq 0$ the sufficient condition for stability is not satisfied. Moreover, the PN (TPN) is unbounded since by the repeated firing of q, the marking in P grows indefinitely i.e., the amount of problems that require a solution accumulate. However, by taking $u = [k, k, k]; k > 0$ (but unknown), we get that $A^T u \leq 0$. Therefore, the PN is stabilizable which implies that the TPN is stable. Now, let us proceed to determine the exact value of k. From the TPN model we obtain that:

$$A_0 = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon \\ 0 & \varepsilon & \varepsilon \\ \varepsilon & 0 & \varepsilon \end{pmatrix} \quad \text{and} \quad A_1 = \begin{pmatrix} Ca & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & Cd \\ \varepsilon & \varepsilon & \varepsilon \end{pmatrix}$$

and making the required computations that: $A_0^* = \begin{pmatrix} 0 & \varepsilon & \varepsilon \\ 0 & 0 & \varepsilon \\ 0 & 0 & 0 \end{pmatrix}$, leading to:

$$\hat{A} = A_0^* \otimes A_1 = \begin{pmatrix} Ca & \varepsilon & \varepsilon \\ Ca & \varepsilon & Cd \\ Ca & \varepsilon & Cd \end{pmatrix}.$$

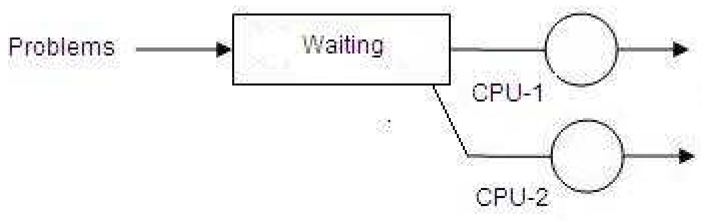

Figure 3: Parallel computer processing system with two CPU's

Therefore, $\lambda(A) = \max\limits_{p\in\mathcal{C}(A)} \frac{|p|_w}{|p|_1} = \max\{Ca, Cd\} = Ca$. This means that in order for the TPN to be stable and work properly the speed at which the serial computer processing system works has to be equal to Ca or being more precise, that all the transitions must fire at the same speed as the problems arrive i.e., they have to be solved as soon as they are loaded into the computer which is attained by setting $k = Ca$. In particular, transition s which is related to the execution time of the CPU has to be fired at a speed equal to Ca.

Summary 41. The serial computer processing system works properly if transition s fires at a speed equal to Ca which implies that the execution frequency of the CPU has to be equal to Ca. Now, if Ca increases due to the fact that the problem to be solved becomes larger then, this will result in an increment on the CPU's execution frequency. However, there is a limit to this increment due to economical and physical limitations. One possible solution is to break this large problem into several smaller problems but this will result in larger execution times.

5.2. Parallel Computer Processing System

Consider a parallel computer processing systems with two CPU's (Fig 3.) whose TPN model is depicted in Fig 4. Where the events (transitions) that drive the system are: q: a problem of size Ca has to be solved, s1, s2: the problem starts being executed by the CPU's, d1,d2: the problem has been solved. The places (that represent the states of the parallel computer processing system) are: A: problems loading, P: the problems are waiting for a solution, B1, B2: the problem is being solved, I1, I2: the CPU's of capacity Cd are idle. The holding times associated to the places A and I1, I2 are Ca and Cd respectively,

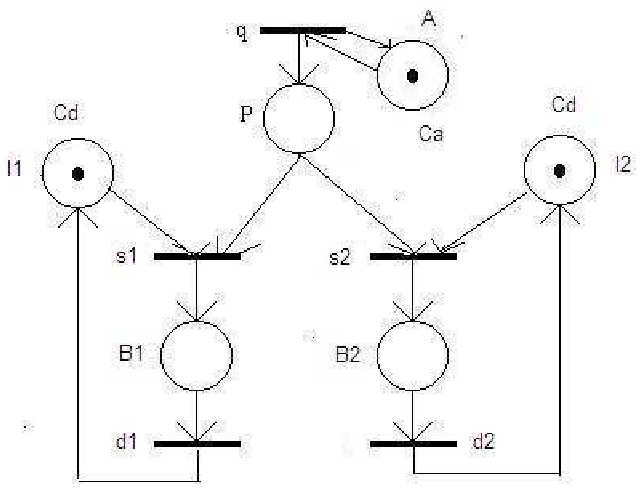

Figure 4: Timed Petri net model

(with $Ca > Cd$). The incidence matrix that represents the PN model is

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}.$$

Therefore since there does not exists a Φ strictly positive m vector such that $A\Phi \leq 0$ the sufficient condition for stability is not satisfied. Moreover, the PN (TPN) is unbounded since by the repeated firing of q, the marking in P grows indefinitely i.e., the amount of problems that require a solution accumulate. However, by taking $u = [k, k/2, k/2, k/2, k/2]; k > 0$ (but unknown) we get that $A^T u \leq 0$. Therefore, the PN is stabilizable which implies that the TPN is stable. Now, let us proceed to determine the exact value of k. From the TPN

model we obtain that:

$$A_0 = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 0 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & 0 & \varepsilon & \varepsilon \end{pmatrix} \quad \text{and} \quad A_1 = \begin{pmatrix} Ca & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & Cd & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & Cd \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{pmatrix}$$

and making the required computations that: $A_0^* = \begin{pmatrix} 0 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 0 & 0 & \varepsilon & \varepsilon & \varepsilon \\ 0 & \varepsilon & 0 & \varepsilon & \varepsilon \\ 0 & 0 & \varepsilon & 0 & \varepsilon \\ 0 & \varepsilon & 0 & \varepsilon & 0 \end{pmatrix}$, leading to:

$$\hat{A} = A_0^* \otimes A_1 = \begin{pmatrix} Ca & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ Ca & \varepsilon & \varepsilon & Cd & \varepsilon \\ Ca & \varepsilon & \varepsilon & \varepsilon & Cd \\ Ca & 0 & \varepsilon & Cd & \varepsilon \\ Ca & \varepsilon & \varepsilon & \varepsilon & Cd \end{pmatrix}.$$

Therefore, $\lambda(A) = \max_{p \in \mathcal{C}(A)} \frac{|p|_w}{|p|_1} = \max\{Ca, Cd\} = Ca$. This means that in order for the TPN to be stable and work properly the speed at which the parallel computer processing system works has to be equal to Ca or being more precise, that all the transitions must fire at the same speed as the problems arrive i.e., they have to be solved as soon as they are loaded into the computer which is attained by setting $k = Ca$. In particular, transitions s1 and s2 which are related to the execution time of the CPU's have to be fired at a speed equal to $Ca/2$.

Remark 42. The previous analysis is easily extended to the case with n CPU's, obtaining that $u = [Ca, Ca/n, Ca/n, ..., Ca/n]$ which translates into the condition that the transitions s1,s2,...,sn, have to be fired at a speed equal to Ca/n.

Summary 43. The parallel computer processing system works properly if transitions s1,s2,...,sn, fire at a speed equal to Ca/n which implies that the execution frequency of the CPU's has to be equal to Ca/n.

5.3. Comparison

As a result of summaries (41) and(43) the following facts are deduced:

1. (Saving time) It is possible to solve a problem of size Ca with one CPU which takes time "Ca", or there is the option of solving a problem of size nCa (or n problems of size Ca each one) using n CPU's which will take the same time as with one CPU.

2. (Saving Money) In order to execute a program, there is the option of purchasing one CPU that costs "Ca" or n CPU's that cost "Ca/n". This is significant for large Ca.

3. (Solving larger problems) If Ca increases due to the fact that the problem to be solved becomes larger then this will result in an increment on the CPU's execution frequency. As a consequence the serial computer processing option becomes expensive and/or slow. This is also true for the parallel computer processing alternative however, by distributing Ca between the n CPU's the economical and/or time impact results to be much lower.

6. Conclusions

The main objective and contribution of this paper consists in using a formal and mathematical approach to prove that parallel computer processing systems are better than serial computer processing systems. This is achieved thanks to the theory of Lyapunov stability and max-plus algebra applied to discrete event systems modeled with time Petri nets

References

[1] A. Gene, Validity of the single processor approach to achieving large-scale computing capabilities, *AFIPS Conference Proceedings*, **30** (1967), 483485; http://www-inst.eecs.berkeley.edu/ n252/paper/Amdahl.pdf

[2] Z. Retchkiman, Stability theory for a class of dynamical systems modeled with Petri nets, *International Journal of Hybrid Systems*, **4**, No. 1 (2005).

[3] V. Lakshmikantham, V.M. Matrosov, S. Sivasundaram, *Vector Lyapunov Functions and Stability Analysis of Nonlinear Systems*, Kluwer Academic Publ., Dordrecht (1991).

[4] Z. Retchkiman, From stability to the stabilization problem of discrete event systems modeled by Petri nets, In: *American Control Conference'99*, San Diego, Cal (June, 1999).

[5] B. Heidergott, G.J. Olsder, J. van der Woude, *Max Plus at Work*, Princeton University Press (2006).

[6] F. Baccelli, G. Cohen, G.J. Olsder, J.P. Quadrat, *Synchronization and Linearity*, Web-Edition (2001).