# Practical Static Methods for Exact Deadlock Prediction in Message Passing Concurrent Processes

Christina Claudia Wuzik

Diplomarbeit

Fakultäten für Mathematik und Wirtschaftswissenschaften
an der Universität Bielefeld

February 9, 1998

**Abstract**

This paper is a diploma thesis in *Wirtschaftsmathematik* done in the faculties of *Mathematik* and *Wirtschaftswissenschaften* of the University of Bielefeld. It is an interdisciplinary work in computer science and mathematics.
Mathematical methods were developed for finding deadlocks in message-passing concurrent processes and two theorems were proven. The theorems deal with the specific case when communication processes are arranged as a tree. My methods consist of reducing the quantity of data which must be examined for deadlock-finding, and then using a matrix-multiplication-model for deadlock-finding.

# Contents

**1 Introduction** **1**

**2 Introduction to the Mathematical Model** **3**

**3 Simons' Observations I and II** **15**

**4 An Inequality in Cycle-Finding** **20**

**5 Factorisation for 3 Traces** **26**

**6 Enlarging for k Traces** **32**

6.1 Enlarging for four Traces . . . . . 32

6.2 Enlarging for k Traces . . . . . 39

**7 An Equality in Cycle-Finding** **43**

**8 Finding a Cycle by Matrix-Multiplication** **48**

**9 Parallel Discrete Matrix-Multiplication-Algorithms** **51**

9.1 Matrix-Multiplication-Algorithm on a 2-dimensional Array . . 51

9.2 Matrix-Multiplication-Algorithm on a Torus . . . . . 54

9.3 Matrix-Multiplication-Algorithm on a 3-dimensional Array . . 56

9.4 Matrix-Multiplication-Algorithm on a 3-dimensional Mesh of Trees . . . . . 59

**10 Conclusions** **62**

# 1 Introduction

The main question in this paper is how to find a deadlock in "message passing concurrent processes". Take two actors, give each his own cue and ask them only to react when they hear it from each other. If each actor waits for his cue to come from the other actor, they will remain silent forever. Hopefully they are intelligent human beings, and will realize that it does not work.

If you take two communication processors, and let them communicate by a complex system, that gives them a certain order to wait for a cue like the actors above, they will also wait forever. This is known as a deadlock. Deadlocks involve at least two processors.

To detect such a deadlock seems to be difficult, especially if there are more than two communication processors, because the problem is NP hard. For practical reasons, we need efficient methods to find the deadlocks to obtain practical bounds for checking the communication process.

A background in computer science is presented which gives a detailed description of the problem together with a transcription into a mathematical model in section 2.

In section 3 we prove two observations which were originally stated by Barbara Simons without explicit proof. In her approach she considers the communication graph as a tree. The main problem of deadlock finding is simplified by using this approach.

In section 4 we analyse "EMFG's" which communication graph is a cycle. Here the possibility for finite periodical parts in the communication system

is found.

The next two sections contain a method which is used to shorten the part of the communication process which is examined for cycles by removing certain repeated parts. The problem remains for what to do after such reductions with an irreducible communication process. Section 7 presents an upper bound for the parts which are to be examined for cycles in accordance with the results of sections 5 and 6.

We develop the matrix-model of static deadlock detection in section 8 in order to find cycles by matrix-multiplication. In section 9 four algorithms are introduced to illustrate how the cycles are detected by parallel discrete matrix-multiplication-algorithms.

We discover and develop methods for finding deadlocks in message-passing concurrent processes. The conclusions in Section 10 contain matrix-multiplication-algorithms which can be used to examine the communication processes as well as provide an upper bound for the examination. These methods are practically the fastest known methods yet.

I would like to thank Prof. Ladkin PhD. and Prof. Dr. Deuber for their supervision and support during this diploma thesis.

# 2 Introduction to the Mathematical Model

In this section the mathematical model for the communication system of processors is introduced. In order to understand the course of formation, we choose to look at three levels:
- the real world,
- computer science,
- the mathematical model.

In the real world, we look at communication systems in which "persons" or "processors" communicate with each other. A typical example is seen in the theater in which a number of actors communicate with each other in a well defined order. Due to technical reasons we are not interested in the content, but in the order. Every actor complies with an individual form of a protocol which states the order the actor must communicate. Such communications, as a rule, go from one actor to another and contain a direction of "message flow". In a direction protocol a typical communication instruction is $w(i, j)$ which should mean that actor $i$ conveys a message to actor $j$.

In computer science communication processes between $k$ communication processors are controlled by communication programs. The basic elements of such communication programs are thus called COMMUNICATION STATEMENTS. These are labelled with "$w(i, j)$", $i, j \in \{1 \cdots k\}$, $i \neq j$, and are read like the instruction "processor $i$ should wire processor $j$". $w(i, j)$ encodes the ordered pair $(i, j)$ and the notation is chosen to agree with the usual terminology in computer science (see [LS9x]). We emphasize again that $w(i, j)$ contains a communication direction as an ordered pair without contents. The communication statement $w(i, j)$ does not symbolize a function of two variables, but merely the directed ordered pair (i,j). With regard to the archi-

tecture of the computer we can suppose that every communication statement involved in the process of performing communication statements can happen.

With this in mind, the undirected COMMUNICATION GRAPH belonging to the communication process can be introduced. The $k$ communication processors are chosen as the vertices. The two communication processors $i$ and $j$ are connected by an edge, if the communication statements $w(i,j)$ or $w(j,i)$ truely occur in the communication process. With regard to the architecture of the computer we can only suppose that the communication graph belonging to the communication system is realized.

In the real world every actor keeps in mind an individual protocol controlling the succession of the various directed communication statements as occuring during the communication process. In computer science, we assign to every communication processor $P_i$, a sequential list $L_i$ of communication statements. Every list $L_i$ is considered as a program for the communication processor $P_i$ which successively controls the communication process.

In terms of computer science the sequential list $L_i$ associated by the communication processor $P_i$, is a non-interleaved linear LOOP PROCESS. A loop process consists of an initializing PREAMBLE followed by an iterated linear LOOP (see figure 1). The preamble $\mathbf{P}_i$ of communication processor $P_i$ - as well as the loop $\mathbf{L}_i$ - are finite sequences of communication statements of the form $w(i,j)$ or $w(j,i)$ with $j \in \{1 \cdots k\} \setminus \{i\}$. As stated, the idea is that the communication process of the communication processor $P_i$ is initialized by the preamble and then the loop is iterated (see figure 2). $|\mathbf{P}_i|$ and $|\mathbf{L}_i|$ are the length of the corresponding sequences.

Now, we consider the following mathematical model for communication

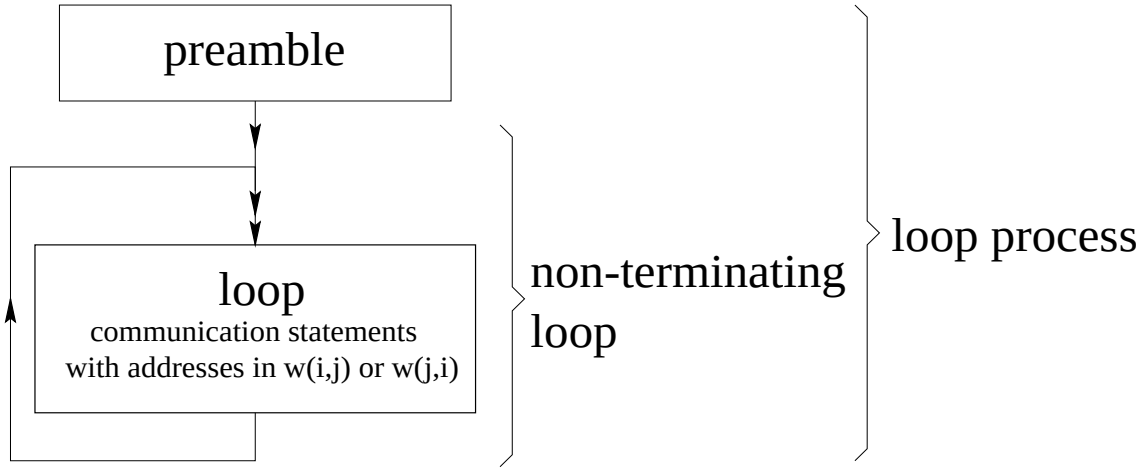



Figure 1: The Sequential List of Communication Processor i

processes and follow [LS9x]:
Let $\Gamma$ be a communication process for $k$ communication processors and $L_i$ be the lists of the communication program. First of all, we assign to every communication processor $P_i$, a one-sided infinite directed path (linear graph) and call it the TRACE $T_i$. We intend to the vertices of the trace $T_i$ with the communication statements from the list $L_i$, as forced by the iteration of the communication process. Formally we label the vertices in the trace $T_i$ with four-tupels:

- The first entry is the number $i$ of the trace $T_i$.
- The second entry is the number of the vertex inside the trace $T_i$.
- The third entry is the communication statement $w(i,j)$ or $w(j,i)$, how it is set according to the iteration of the list $L_i$. $i$ is the number of the trace $T_i$ and $j \in \{1, \cdots, k\} \setminus \{i\}$.
- The fourth entry tells us how often the communication statement which

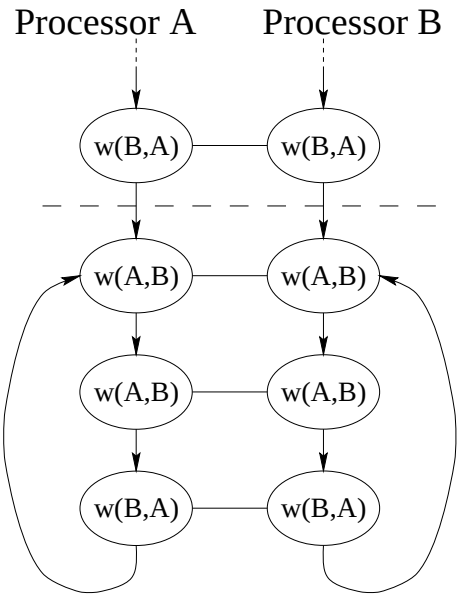



Figure 2: Example for Loop Processes in Computer Science

stands in the third entry has occurred in the trace $T_i$ before.

There is a natural order on every trace (representing the time of the occurence of the communication statement) by the numbering in the second entry of the labelling. We say that the vertex $x$ with the number $n+1$ occurs "later" in the trace $T_i$ than the vertex $y$ with the number $n$ and denote it by $y \prec x$. Thus call the vertex with the number 0 the "initial vertex". We define the third entry in the four-tupel for all vertices in the trace $T_i$ in the order of the time of their occurence, as the subsequence $\mathcal{W}_i$.

For the communication process $\Gamma$ we construct the "Extended Message Flow Graph" EMFG on the set of the union of the traces $T_1 \cdots T_k$ of all communication processors $P_1 \cdots P_k$. To set the edges we remember that the communication process suggests the following procedure in the mathematical model: If we are at a fixed vertex $x$ in trace $T_i$ in which the com-

munication statement $w(i,j)$ or $w(j,i)$ occurs, we know by the labelling how often this communication statement has already occured in the trace $T_i$; for example, $m$-times. We take the vertex $y$ in the corresponding trace $T_j$ at which the mentioned communication statement also occurs for the $m^{th}$ time, and connect the vertices $x, y$ by an undirected edge. This rule defines all edges in the EMFG.

**Remark 2.1** *By the description it is clear that this mathematical model intends to state that in every vertex, there starts one undirected edge. Otherwise, a communication process would not happen.*

It should be clear that the mathematical model is the quintessence of the direction instructions of the theater in the real world and also models the communication process in computer science. The following can be noted:
In trace $T_i$, the first $|\mathbf{P}_i|$ vertices are isomorph to the preamble of the communication process for the communication processor $P_i$. We call this set $\overline{\mathbf{P}_i}$ of trace $T_i$. The next $|\mathbf{L}_i|$ many vertices of the trace $T_i$ are called the loop $\overline{\mathbf{L}_i}$.

**Definition 2.2** *In an EMFG $G$ we call a sequence of vertices $v_1 v_2 v_3 \cdots v_n v_1$, which are pairwise distinct, a* CYCLE, *if for all $i = 1 \cdots n$:*
*Either $v_i$ and $v_{(i+1) \bmod n}$ are in the same trace and there is a directed edge from $v_i$ to $v_{(i+1) \bmod n}$, or $v_i$ and $v_{(i+1) \bmod n}$ are in different traces and connected by an undirected edge.*

**Definition 2.3** *Let $G$ be an EMFG with $k$ communication processors. Let $S$ be a subgraph.*
*1) $S \subset G$ is a* SEGMENT *of $G$, if $S \cap T_i$ is an interval for every trace $T_i \subset G$ with respect to the order on $T_i$.*

*2) $A \subset G$ is an* INITIAL PART *of $G$ if $A$ is segment of $G$ and $A$ contains the initial vertices on every trace.*

A graph theoretical question is now if a given EMFG $G$ contains a cycle (Figure 3 shows this case in computer science and figure 4 in the mathematical model.)

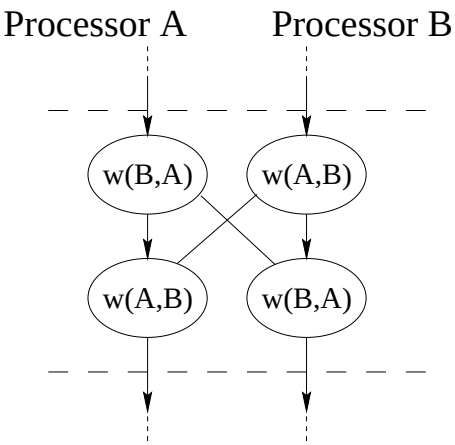



Figure 3: Picture of the most trivial Case of a Deadlock in Computer Science

Now, we shall discuss what the existence of a cycle in the EMFG $G$ means in a communication process or in the real world. A typical situation in the real world would be as follows: $k$ politicans in a coalition dispute are each waiting for a positive signal from the other to find an agreement in further negotiations (communication process) and so to stay in the coalition. If these signals do not appear - a common problem in the real world -, then we have what we call a "deadlock". It is clear that a deadlock in a communication system in the real world or in computer science occurs if the corresponding EMFG of the communication process contains a cycle.

**Definition 2.4** *Let $G$ be an EMFG.*

*We define the matrix* $\Sigma(G) = \sigma(i,j)$*:*
*Let* $\sigma(i,j)$ *with* $i,j \in \{1 \cdots k\}$, $i \neq j$, *be the number of occurrences of the communication statement w(i,j) plus the number of occurrences of w(j,i) in the loop* $\overline{\mathbf{L}_i}$ *in the trace* $T_i$.

**Remark 2.5** *We assign a matrix* $\Sigma(G) = \sigma(i,j)$ *to every communication process in computer science with the EMFG G. Because* $\sigma(i,j)$ *and* $\sigma(j,i)$ *are differently counted on different traces, the matrix in general is not symmetrical in and the diagonal contains only zeros.*

**Definition 2.6** *An EMFG G is* FINALLY PERIODICAL *with the period* $\mathcal{P}$*, if:*
*1)* $\mathcal{P}$ *is a segment of G and there is no undirected edge between* $\mathcal{P}$ *and* $G \setminus \mathcal{P}$.
*2) There is a partition of* $G = A \dot{\cup} \dot{\bigcup}_i^{\infty} B_i$, *where A is an initial part of G and* $B_i$ *is a segment isomorphic to* $\mathcal{P}$.
*3)* $\mathcal{P}$ *is (inclusion) minimal with respect to (1) and (2).*

It is an interesting question, whether for a given matrix $\Sigma$ there is a universal "length" $\overline{\Sigma}$ with the following property: "Let $G$ be an EMFG with $\Sigma(G) = \Sigma$. In case $G$ contains a cycle there are already such cycles in an initial part of $G$ of length atmost $\overline{\Sigma}$". This question is not interesting for aperiodical EMFG's, as under mild restrictions these always contain cycles (see [LS9x]). An example for an aperiodical graph according to definition 2.6 is the EMFG in figure 5. We only need to consider finally periodical graphs.

To measure the "length" of the initial parts $A_\Sigma$ we divide the EMFG into levels. Then the "length" of the initial parts are measured in levels.

**Definition 2.7** *Let G be an EMFG. We divide* $G = \mathcal{L}_0 \dot{\cup} \mathcal{L}_1 \dot{\cup} \mathcal{L}_2 \dot{\cup} \cdots$ *into the following segments with the properties:*
*1) Level* $\mathcal{L}_0$ *is - with respect to inclusion - the minimal initial part of G which*

*contains all preambles and undirected edges which start in these preambles. 2)* $V(G) = \mathcal{L}_0 \dot{\cup} \mathcal{L}_1 \dot{\cup} \mathcal{L}_2 \dot{\cup} \cdots$ *is the partition into* LEVELS. *For* $m > 0$ $\mathcal{L}_m$ *is a segment satisfying* $|\mathcal{L}_m \cap T_i| = |\mathrm{S}_i|$, $i \in \{1 \cdots k\}$.

**Definition 2.8** *Let* $G$ *be an EMFG. Let* $S \subset G$ *be a initial part. The* LENGTH *of a initial part is defined as* $g(S) = \min\{m \mid S \subset \mathcal{L}_0 \dot{\cup} \cdots \dot{\cup} \mathcal{L}_m\}$

Level $\mathcal{L}_0$ contains all vertices and edges which are affected by the initialisation process and level $\mathcal{L}_n$, $n > 0$ respects the length of the loops of the communication process. Levels and periods are not the same. In the EMFG there may be undirected edges between two levels, but not between two periods (see figure 6 and 7).

One question is how many levels have to be examined in order to cover the initial part that contains a cycle. This is answered in section 5. In the next section we will first examine communication graphs without cycles.

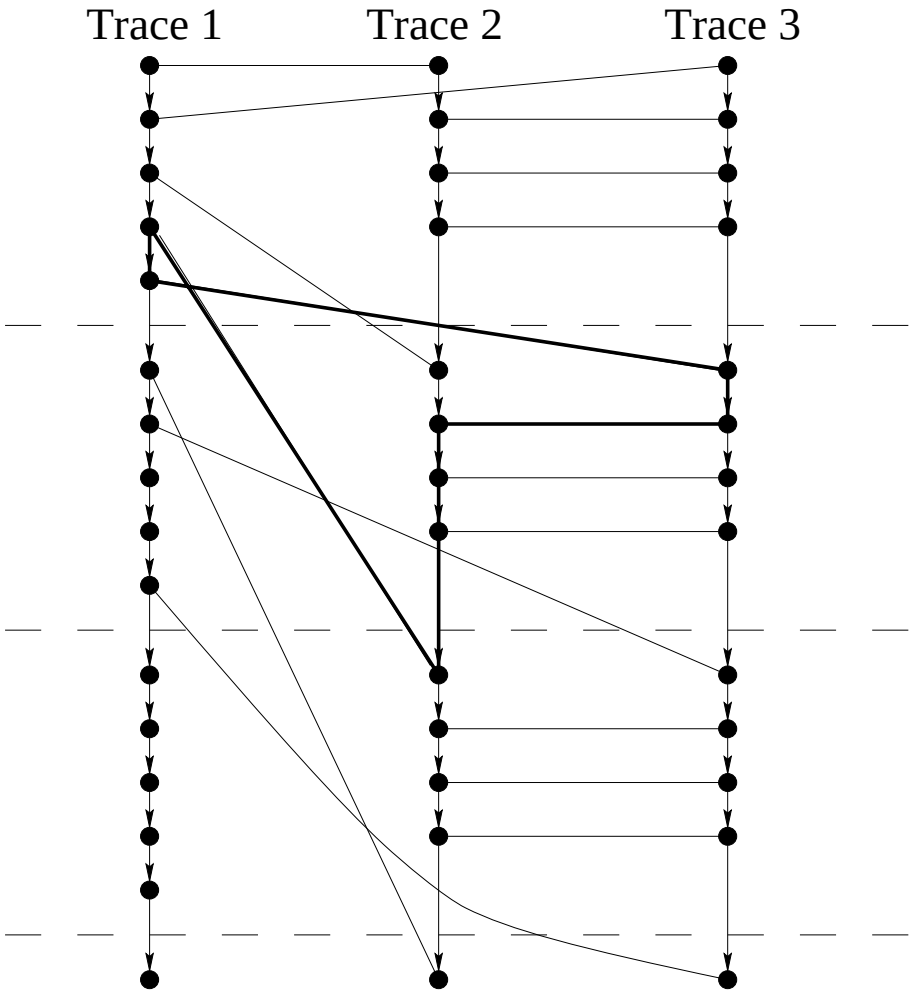



Figure 4: An EMFG with a Cycle

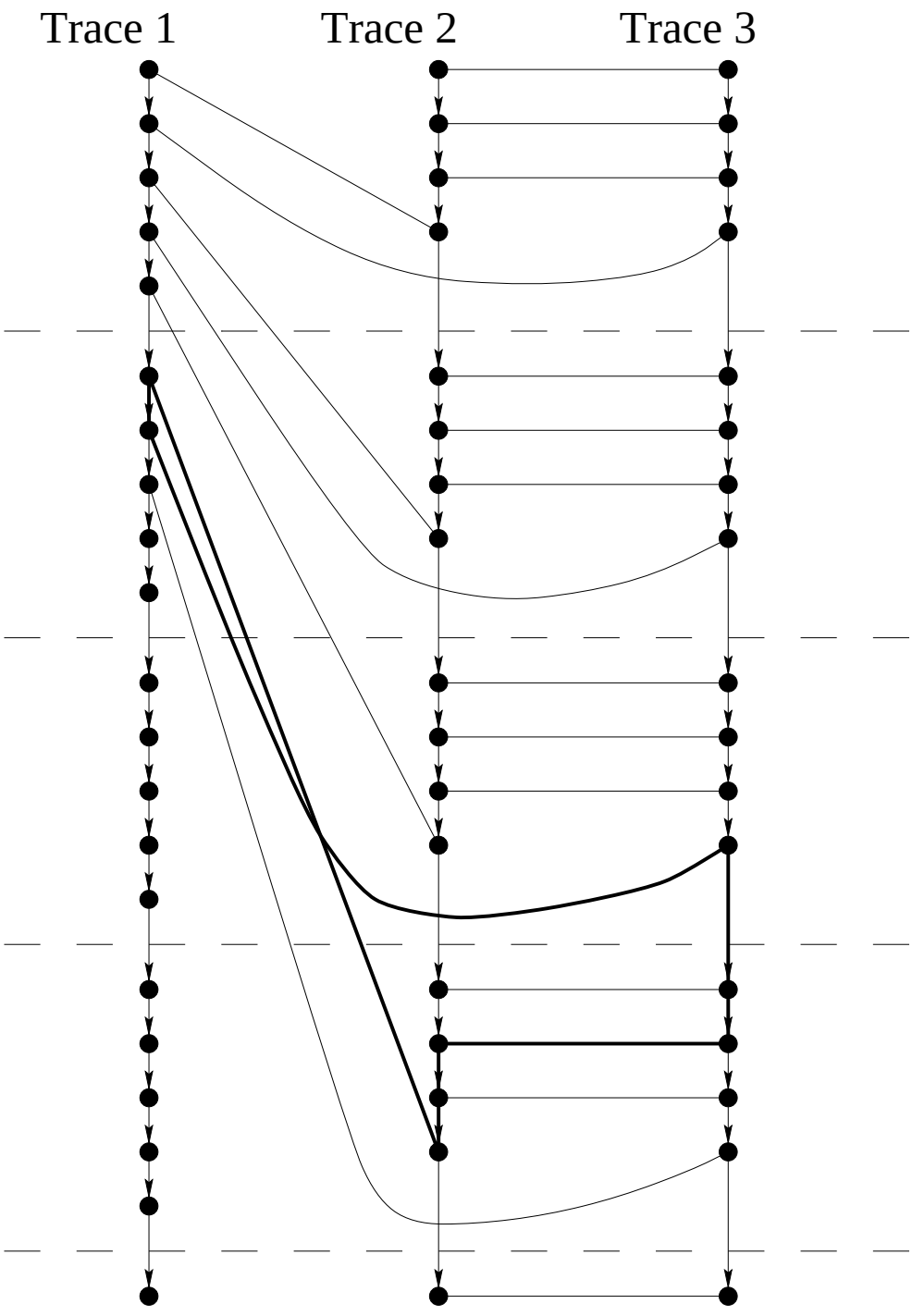



Figure 5: Example of an Aperiodical EMFG

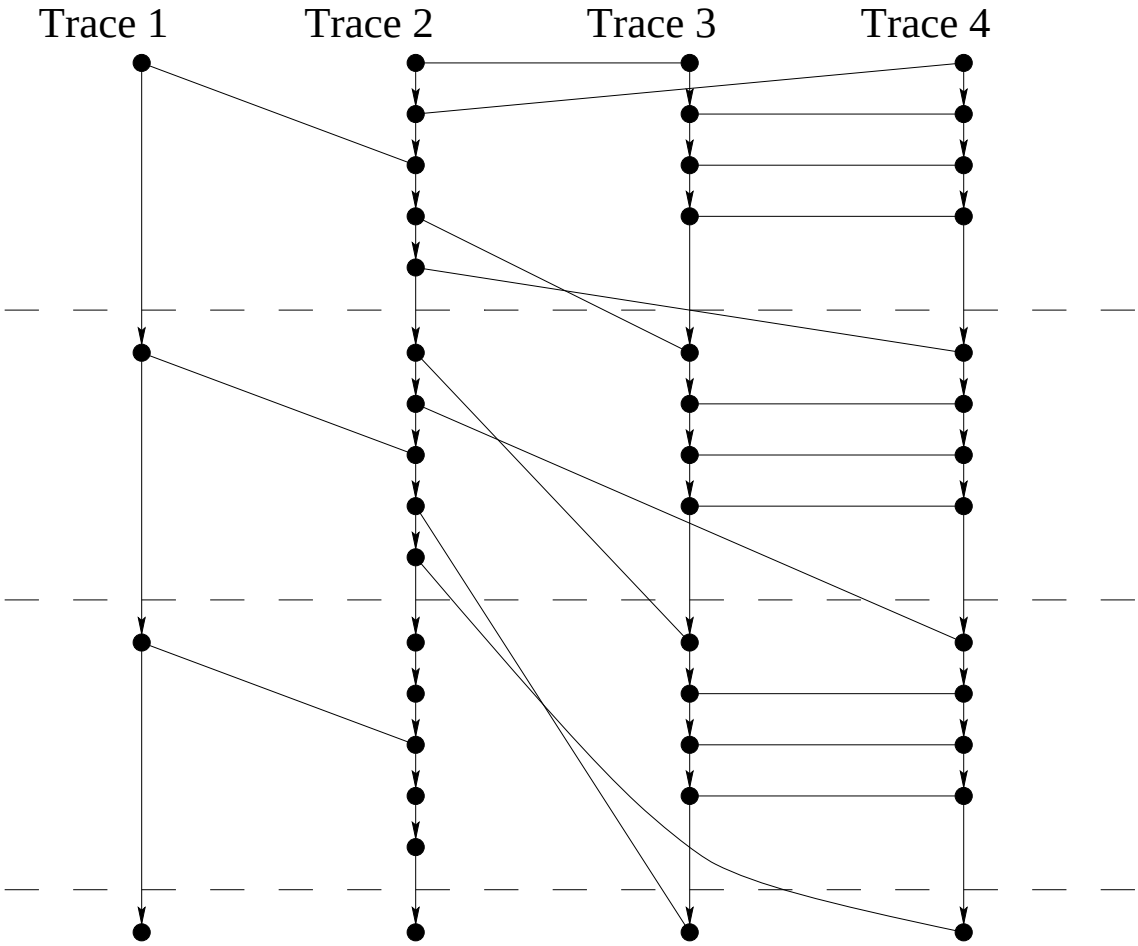



Figure 6: EMFG in which some Undirected Edges lie between two Levels

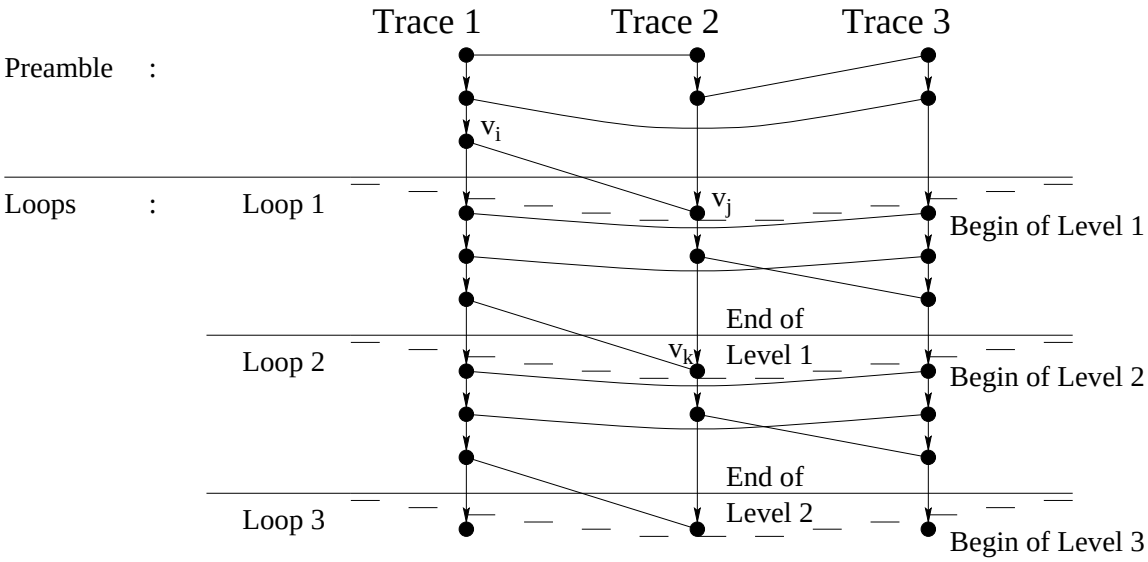



Figure 7: EMFG where Loops and Levels are not the same

# 3 Simons' Observations I and II

In this section we consider communication graphs without cycles and analyse how many traces are required by cycles in the EMFG.

**Definition 3.1 ([LS9x])** *Let $G$ be an EMFG. Let $s \in \mathbb{N}$ and $C$ be a cycle in $G$.*
*We say that $s(C) = s$ i.e. the* PROCESSNUMBER *of $C$ is $s$ or $C$ is an $s$-*PROCESS CYCLE*, if the following hold:*
*1) $C$ intersects exactly $s$ traces in $G$.*
*2) $C$ is segment of $G$.*

Because the traces are directed paths, there is no cycle in a single trace. Thus the case $s(C) = 1$ is trivially excluded, and a cycle has the process-number $s(C) \geq 2$.

We analyse if it is possible to draw conclusions from the process-number $s$ to the communication graph which corresponds to the EMFG $G$. We prove the following lemma:

**Lemma 3.2** *Let $G$ be an EMFG.*
*If there exists a cycle $C$ in $G$ with $s(C) > 2$ then the communication graph associated to the EMFG $G$ contains at least one cycle.*

*Proof:*

Let $v_i$ be the vertex in the trace $T_i$, $i \in \{1 \cdots k\}$. Without any restrictions $(v_1, v_1', v_2, v_2' \cdots v_s', v_1)$ would be a directed $s$-process cycle in the EMFG $G$ in which the traces are crossed traces in their natural order. To reconstruct a

cycle in a subgraph in the EMFG we look at the undirected edges between the traces. If there is an undirected edge between the two traces $T_i$ and $T_{i+1}$ in the mathematical model, then there is a communication statement $w(i, i+1)$ respectively $w(i+1, i)$ in the model for computer science. But the communication graph is so defined that the communication statement $w(i, i+1)$ respectively $w(i+1, i)$ forces an undirected edge between the vertices $i$ and $i+1$. Thus every undirected edge $(v'_i, v_{i+1 \bmod s})$, $i \in \{1 \cdots s\}$, in the EMFG $G$ forces an undirected edge $i, i+1 \bmod s$, $i \in \{1 \cdots s\}$, in the communication graph. Therefore we get the cycle $(1, 2, 3 \cdots s, 1)$ in the communication graph. *QED*

Now we analyse which consequences occur if the communication graph does not contain cycles. The theorem 3.3 immediately results from the lemma 3.2:

**Theorem 3.3 (Simons' Observation I, [Lad, pers.])** *Let $G$ be an EMFG and the corresponding communication graph be without cycles.*
*Then $G$ contains no cycles or 2-process cycles only.*

**Definition 3.4** *Let $G$ be an EMFG.*
*1) The communication processors $P_i$ and $P_j$* MATCH*, if and only if the two subsequences in the traces $T_i$ and $T_j$ with the entries $\{w(i,j), w(j,i)\}$ coincide.*
*2) An EMFG $G$ obeys the* SEQUENCE CONDITION *(SC), if and only if all communication processors match pairwise.*

**Lemma 3.5** *Let $G$ be an EMFG.*
*The SC holds in $G$, if and only if $G$ does not contain 2-process cycles.*

*Proof:*

*Assertion I:* If $s = 2$ then the SC does not hold in $G$.

We recall (cf. page 6):
1) $y \prec x$, $x, y \in T_i \Leftrightarrow$ The vertex $y$ in the EMFG $G$ is earlier in the trace $T_i$ than the vertex $x$.
2) $x^{\prec} := \langle y \mid y \prec x \rangle$
3) $x \lessdot y \Leftrightarrow y$ is the immediately successor of $x$.

Any two traces in $G$ may be arranged as in figure 8. Without any restrictions we can assume that the SC holds in the initial part of $G$ generated by $v_i^{\prec} \cup v_j^{\prec}$. So we write:

$$v_i^{\prec} \cong v_j^{\prec} \tag{1}$$

Because of the undirected edges in the EMFG we get:

$$v_i^{\prec} \lessdot v_i \lessdot v_i' \cong v_j^{\prec} \lessdot v_j' \lessdot v_j \tag{2}$$

But this does not correspond with the subsequence $\mathcal{W}_j$ that force this arrangement:

$$v_j^{\prec} \lessdot v_j \lessdot v_j' \tag{3}$$

But this contradicts with (2):

$$v_j^{\prec} \lessdot v_j' \lessdot v_j \neq v_j^{\prec} \lessdot v_j \lessdot v_j' \tag{4}$$

Therefore, assertion I is true.

*Assertion II:* If $G$ does not hold the SC then $s = 2$.

If $G$ does not hold the SC then there are communication statements in the subsequences which do not match. This is concluded directly from the

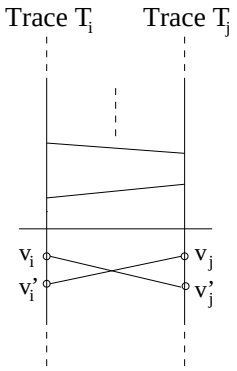



Figure 8: Figure of a EMFG which does not hold SC

definition of the SC.
In order to get a 2-process cycle we need communication statements which do not match.
If $G$ does not hold the SC, then there are unmatched communication statements and therefore the EMFG contains 2-process cycles.

I.e. the two assertions are proven and the lemma holds. *QED*

**Theorem 3.6 (Simons' Observation II, [Lad, pers.])** *Let $G$ be an EMFG which satisfies the SC.*
*If the corresponding communication graph is without cycles, then the EMFG $G$ does not contain any cycles.*

*Proof:*

Concerning theorem 3.3, a cycle can only be a 2-process cycle in the EMFG $G$, if the corresponding communication graph does not contain cycles. Because $G$ satisfies the SC, it is impossible for $G$ to contain 2-process cycles (Lemma 3.5). Thus the EMFG $G$ does not contain any cycles. *QED*

We considered the case of the communication graphs that do not contain cycles and the resulting consequences for the EMFG. However this is a special case and so we therefore want to analyse the results for cycles as communication graphs.

# 4 An Inequality in Cycle-Finding

First it is useful to define the class of communication graphs considered in this section.

**Definition 4.1** *Let $P_1 \cdots P_k$ be the communication processors in the communication process $\Gamma$.*
*The communication process $\Gamma$ is* OF CIRCULAR TYPE*, if the communication graph is the standard cycle $1 - 2 - \cdots - k - 1$.*

Let $\mathcal{C}_k$ be the class of EMFG's of communication processes of circular type and with $k$ communication processors. The corresponding matrices are generalized bandmatrices which have non-vanishing entries only at the positions $\sigma(i, (i+1) \bmod k)$ and $\sigma((i+1) \bmod k, i)$.

**Definition 4.2 (Numerical Matching Condition)** *A matrix $\Sigma$ obeys the* NUMERICAL MATCHING CONDITION *(NMC), if:*
$\sigma(1,2) \cdot \sigma(2,3) \cdots \sigma(k-1,k) \cdot \sigma(k,1) = \sigma(1,k) \cdot \sigma(k,k-1) \cdots \sigma(2,1) =: N(\Sigma)$

If $G$ is of circular type the matrix $\Sigma(G)$ obeys the NMC, if the products of the non-vanishing entries of the matrix above and below the diagonal are equal.

**Theorem 4.3 (see also [LS9x], page 67ff)** *Let $G \in \mathcal{C}_k$ be an EMFG with matrix $\Sigma(G)$ which obeys the NMC.*
*Then the graph which is underlying to $G$ is finally periodical.*

*Proof:*

Without any restrictions let $\mathcal{L}_0 = \emptyset$. The segment $L_1 \cap T_i$ contains $\sigma(i, (i+k-1) \bmod k)$ many vertices which are connected with undirected edges between the traces $T_{(i+k-1) \bmod k}$ and $T_i$. The segment $L_1 \cap T_i$ also contains $\sigma(i, (i+1) \bmod k)$ many vertices which are connected with undirected edges between the traces $T_i$ and $T_{(i+1) \bmod k}$. Thus $\sigma(i, (i+k-1) \bmod k)$ many levels of trace $T_{(i+k-1) \bmod k}$ and $\sigma((i+1) \bmod k, i)$ many levels of trace $T_{(i+1) \bmod k}$ have to be repeated until we are sure that one level of trace $T_i$ is repeated. Because $\Sigma(G)$ obeys the NMC, the number of undirected edges between $T_i$ and $T_{(i+1) \bmod k}$ and of $T_i$ and $T_{(i+k-1) \bmod k}$ are equal. So the graph is periodical. If $\mathcal{L}_0 \neq \emptyset$ then the graph is finally periodically. *QED*

**Remark 4.4** *Let $G \in \mathcal{C}_k$ be an EMFG with matrix $\Sigma(G)$ which satisfies the NMC.*
*The length of a minimal period $P$ of the EMFG $G$ is atmost*
$N(\Sigma) = \prod_{i=1}^{k} \sigma(i, (i+1) \bmod k)$

This also immediately results from the proof of theorem 4.3.

It is clear that the size of the upper bound $N(\Sigma)$ depends on the matrix $\Sigma(G)$ of the EMFG $G$. It is then interesting to consider the particular instance of the matrix as symmetrical.

**Remark 4.5** *Let $\Sigma(G)$ be the symmetrical matrix of the EMFG $G$. The NMC holds in $G$.*
*Then any level is a possibly nonminimal period which satisfies the conditions 1) and 2) of definition 2.6.*

*Proof:*

There are $\sigma(i,j)$ vertices with undirected edges between the traces $T_i$ and $T_j$ in the segment $L_1 \cap T_i$. There are also $\sigma(j,i)$ vertices with undirected edges between the traces $T_i$ and $T_j$ in the segment $L_i \cap T_j$. Therefore $\sigma(i,j) = \sigma(j,i)$, because the matrix is symmetrical. I.e. there are no undirected edges between two levels. Because this is true for all traces in the EMFG, a period is equal to a level. *QED*

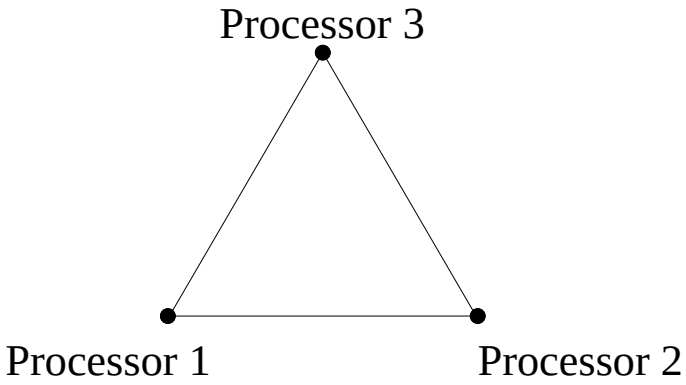



Figure 9: A Communication Graph

As a result, theorem 4.6 is proven. In the figures 9 and 10 the case for $k = 3$ is illustrated.

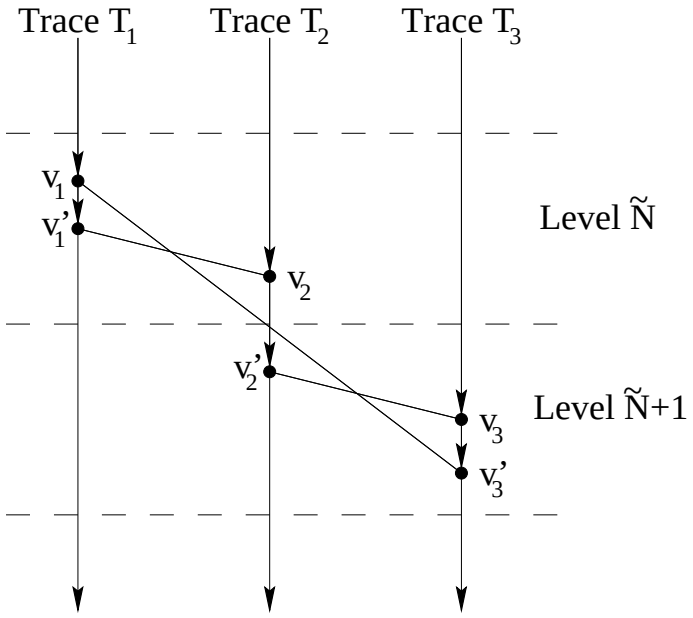



Figure 10: A Picture of the EMFG with a Cycle corresponding to the Communication Graph in Figure 9

**Theorem 4.6** *Let $G \in \mathcal{C}_k$, $k \geq 3$, be an EMFG with matrix $\Sigma(G)$. The NMC holds in $G$. Let $N(\Sigma) \neq 0$.*
*If the inequality*

$$\left\lceil \left\lceil \cdots \left\lceil \left\lceil \tilde{N} \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} + 1 \right\rceil \cdots \right\rceil \cdot \frac{\sigma(k-1,k)}{\sigma(k,k-1)} \right\rceil < \left\lceil \tilde{N} \cdot \frac{\sigma(1,k)}{\sigma(k,1)} \right\rceil \tag{5}$$

*is true for at least one $\tilde{N} \in \{1 \cdots N(\Sigma)\}$, then there is a $k$-process cycle $C$, $C \subset \mathcal{L}_1 \dot{\cup} \cdots \dot{\cup} \mathcal{L}_{N(\Sigma)}$, in the EMFG $G$.*

*Proof:*

Due to the communication process being circular, an undirected edge in the EMFG $G$ connects a vertex $v_i$ in the trace $T_i$ with a vertex $v_{(i+1) \bmod k}$ in the trace $T_{(i+1) \bmod k}$, $i \in \{1 \cdots k\}$.
Without restrictions, we assume that there is a vertex $v_1'$ labelled in the subsequence $\mathcal{W}_1$ with $w(1,2)$ or $w(2,1)$ in the segment $T_1 \cap \mathcal{L}_{\tilde{N}}$. So there is an undirected edge $(v_1', v_2)$. We know that there are $\sigma(1,2)$ undirected edges between trace $T_1$ and $T_2$ lying in a segment $T_1 \cap \mathcal{L}_i$, $i > 0$, and also that there are $\sigma(2,1)$ undirected edges between trace $T_1$ and $T_2$, lying in a segment $T_2 \cap \mathcal{L}_i$, $i > 0$. We look at the numerical proportion from $\sigma(1,2)$ to $\sigma(2,1)$. It specifies how many levels later or earlier the vertex $v_2$ occurs compared to the vertex $v_1'$. More specifically, that the vertex $v_2$ is in level $\mathcal{L}_{\left\lceil \tilde{N} \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \right\rceil}$. It depends on the numerical proportion from $\sigma(1,2)$ to $\sigma(2,1)$, whether $\mathcal{L}_{\left\lceil \tilde{N} \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \right\rceil}$ is later than or earlier than or equal to $\mathcal{L}_{\tilde{N}}$.
There is a vertex $v_2'$ in the trace $T_2$ labelled in the subsequence $\mathcal{W}_2$ with $w(2,3)$ or $w(3,2)$ that has occured later than $v_2$. To be sure that the vertex $v_2'$ is later than the vertex $v_2$, the vertex $v_2'$ is in the next level. In this case, the vertex $v_2'$ is in level $\mathcal{L}_{\left\lceil \tilde{N} \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil}$. Again there is an undirected edge $(v_2', v_3)$ connecting trace $T_2$ and trace $T_3$. The vertex $v_3$ is in level $\mathcal{L}_{\left\lceil \left\lceil \tilde{N} \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil}$, because we look at the relation between $\sigma(2,3)$ and $\sigma(3,2)$.
We use the same technique as above until we determine the vertex $v_k$ in the segment $T_k \cap \mathcal{L}_{\left\lceil \left\lceil \cdots \left\lceil \left\lceil \tilde{N} \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} + 1 \right\rceil \cdots \right\rceil \cdot \frac{\sigma(k-1,k)}{\sigma(k,k-1)} \right\rceil}$.
If there is an undirected edge $(v_k', v_1)$ with $v_k'$ it is later than $v_k$ and $v_1'$ is also later than $v_1$ then there is a cycle $C$ in the EMFG $G$. This means if the inequality

$$\left\lceil \left\lceil \cdots \left\lceil \left\lceil \tilde{N} \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} + 1 \right\rceil \cdots \right\rceil \cdot \frac{\sigma(k-1,k)}{\sigma(k,k-1)} \right\rceil < \left\lceil \tilde{N} \cdot \frac{\sigma(1,k)}{\sigma(k,1)} \right\rceil \tag{6}$$

is true for some $\tilde{N}$, then there is a $k$-process cycle. *QED*

By using the inequality the upper bound $N(\Sigma)$ can sometimes decrease, this is to say if the non-vanishing elements are not relatively prime. We will now show this in the next section.

# 5 Factorisation for 3 Traces

In this section we reduce the natural number $\tilde{N}$ until the inequality 5 of the theorem 4.6 is examined to ensure if cycles exist or not.

**Theorem 5.1** *Let $G_3 \in \mathcal{C}_3$ be an EMFG and matrix $\Sigma(G_3)$. Let $\Sigma(G_3)$ satisfy the NMC. Let $N(3) \neq 0$.*
*If one of the pairs $(\sigma(i,j), \sigma(i',j'))$ with $\sigma(i,j), \sigma(i',j') \in \{\sigma(1,2),\ \sigma(2,3),\ \sigma(3,1)\}$ is not relatively prime, then there is a $\mathcal{N}(3) < N(3)$ with the following property:*
*If there is a 3-process cycle in the EMFG $G_3$ then there is such a 3-process cycle in or before level $\mathcal{L}_{\tilde{N}}$, for some $\tilde{N} \in \{1 \cdots \mathcal{N}(3)\}$.*

*Proof:*

Here the NMC is:

$$\sigma(1,2) \cdot \sigma(2,3) \cdot \sigma(3,1) = \sigma(1,3) \cdot \sigma(3,2) \cdot \sigma(2,1) \tag{7}$$

We take the greatest common divisor gcd and define:

$$\begin{aligned} D_1(1) &\triangleq \gcd(\sigma(1,2), \sigma(2,1)) \\ D_1(2) &\triangleq \gcd(\sigma(2,3), \sigma(3,2)) \\ D_1(3) &\triangleq \gcd(\sigma(1,3), \sigma(3,1)) \end{aligned}$$

Consequently it can be introduced that $r_1(1,2)$, $r_1(2,1)$, $r_1(2,3)$, $r_1(3,2)$, $r_1(1,3)$ and $r_1(3,1)$ and by the following equalities "$r_1(1,2)$ does not divide $r_1(2,1)$" is written by $r_1(1,2) \nmid r_1(2,1)$:

$$
\begin{aligned}
\sigma(1,2) &= r_1(1,2)\cdot D_1(1) \quad \text{with } r_1(1,2) \not| \, r_1(2,1)\\
\sigma(2,1) &= r_1(2,1)\cdot D_1(1)\\
\sigma(2,3) &= r_1(2,3)\cdot D_1(2) \quad \text{with } r_1(2,3) \not| \, r_1(3,2)\\
\sigma(3,2) &= r_1(3,2)\cdot D_1(2)\\
\sigma(1,3) &= r_1(1,3)\cdot D_1(3) \quad \text{with } r_1(1,3) \not| \, r_1(3,1)\\
\sigma(3,1) &= r_1(3,1)\cdot D_1(3)
\end{aligned}
$$

It can be defined:

$$
\begin{aligned}
D_2(3) &\triangleq \gcd(r_1(2,1), r_1(3,1))\\
D_2(1) &\triangleq \gcd(r_1(1,2), r_1(3,2))\\
D_2(2) &\triangleq \gcd(r_1(1,3), r_1(2,3))
\end{aligned}
$$

Therefore, it is true that:

$$
\begin{aligned}
&D_2(3) \not| \, D_2(1)\\
&D_2(1) \not| \, D_2(2)\\
&D_2(2) \not| \, D_2(3)
\end{aligned}
$$

Now we can write:

$$
\begin{aligned}
&\sigma(1,2) = r_2(1,2)\cdot D_2(1)\cdot D_1(1)\\
&\sigma(2,1) = r_2(2,1)\cdot D_2(3)\cdot D_1(1)\\
&\sigma(2,3) = r_2(2,3)\cdot D_2(2)\cdot D_1(2)\\
&\sigma(3,2) = r_2(3,2)\cdot D_2(1)\cdot D_1(2)\\
&\sigma(1,3) = r_2(1,3)\cdot D_2(2)\cdot D_1(3)\\
&\sigma(3,1) = r_2(3,1)\cdot D_2(3)\cdot D_1(3)\\
&\quad \text{and} \quad r_2(1,2) \not| \, r_2(2,1)\\
&\qquad\quad\;\; r_2(2,3) \not| \, r_2(3,2)\\
&\qquad\quad\;\; r_2(1,3) \not| \, r_2(3,1)
\end{aligned}
$$

Next we show that the following claim is true:

$$r_2(1,2) \cdot r_2(2,3) \cdot r_2(3,1) = r_2(1,3) \cdot r_2(3,2) \cdot r_2(2,1) \tag{8}$$

In order to prove this, we will use the NMC and replace the variables by the expressions derived above:

$$\sigma(1,2) \cdot \sigma(2,3) \cdot \sigma(3,1) \;=\; \sigma(1,3) \cdot \sigma(3,2) \cdot \sigma(2,1) \tag{9}$$

$$\Longrightarrow$$

$$\begin{aligned} & r_2(1,2) \cdot D_2(1) \cdot D_1(1) \cdot r_2(2,3) \cdot D_2(2) \cdot D_1(2) \cdot r_2(3,1) \cdot D_2(3) \cdot D_1(3) \\ = & r_2(1,3) \cdot D_2(2) \cdot D_1(3) \cdot r_2(3,2) \cdot D_2(1) \cdot D_1(2) \cdot r_2(2,1) \cdot D_2(3) \cdot D_1(1) \end{aligned}$$

$$\Longrightarrow$$

$$r_2(1,2) \cdot r_2(2,3) \cdot r_2(3,1) \;=\; r_2(1,3) \cdot r_2(3,2) \cdot r_2(2,1) \tag{10}$$

The following equations are also true:

$$r_2(1,2) = r_2(1,3)$$

$$r_2(2,1) = r_2(2,3)$$

$$r_2(3,1) = r_2(3,2)$$

This result follows from an examination of the inequality 5. All factors are elements of $\mathbb{N} \setminus \{0\}$. This means that $r_2(1,2) \mid r_2(1,3) \cdot r_2(3,2) \cdot r_2(2,1)$. But it is also known that $r_2(1,2) \nmid D_2(1)$. $D_2(1)$ contains all common parts of $r_2(3,2)$. $r_2(1,2) \nmid D_1(1)$ and $D_1(1)$ contain those of $r_2(2,1)$. Therefore, only $r_2(1,2) = r_2(1,3)$ can be true because there are no relations between $r_2(1,2)$ and $r_2(3,2)$ and between $r_2(1,2)$ and $r_2(2,1)$. The rest follows analogously. In consequence, it can be written:

$$r_2(1,2) = r_2(1,3) = r_2(1)$$

$$r_2(2,1) = r_2(2,3) = r_2(2)$$

$$r_2(3,1) = r_2(3,2) = r_2(3)$$

Therefore it is true that:

$$\sigma(1,2) = r_2(1) \cdot D_2(1) \cdot D_1(1)$$

$$\sigma(1,3) = r_2(1) \cdot D_2(2) \cdot D_1(3)$$

$$\sigma(2,1) = r_2(2) \cdot D_2(3) \cdot D_1(1)$$

$$\sigma(2,3) = r_2(2) \cdot D_2(2) \cdot D_1(2)$$

$$\sigma(3,1) = r_2(3) \cdot D_2(3) \cdot D_1(3)$$

$$\sigma(3,2) = r_2(3) \cdot D_2(1) \cdot D_1(2)$$

Thus, the inequality 5 of the theorem 4.6 in $N(3)$ can be reduced to the following:

$$\left\lceil \left\lceil N(3) \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil < \left\lceil N(3) \cdot \frac{\sigma(1,3)}{\sigma(3,1)} \right\rceil \tag{11}$$

$$\Longrightarrow$$

$$\left\lceil \left\lceil N(3) \cdot \frac{r_2(1) \cdot D_2(1)}{r_2(2) \cdot D_2(3)} + 1 \right\rceil \cdot \frac{r_2(2) \cdot D_2(2)}{r_2(3) \cdot D_2(1)} \right\rceil < \left\lceil N(3) \cdot \frac{r_2(1) \cdot D_2(2)}{r_2(3) \cdot D_2(3)} \right\rceil \tag{12}$$

So we can write $N(3)$ as follows:

$$N(3) = \mathcal{N}(3) + r_2(2) \cdot r_2(3) \cdot D_2(3) \tag{13}$$

$$\Longrightarrow$$

$$\left\lceil \left\lceil (\mathcal{N}(3) + r_2(2) \cdot r_2(3) \cdot D_2(3)) \cdot \frac{r_2(1) \cdot D_2(1)}{r_2(2) \cdot D_2(3)} + 1 \right\rceil \cdot \frac{r_2(2) \cdot D_2(2)}{r_2(3) \cdot D_2(1)} \right\rceil$$

$$< \left\lceil (\mathcal{N}(3) + r_2(2) \cdot r_2(3) \cdot D_2(3)) \cdot \frac{r_2(1) \cdot D_2(2)}{r_2(3) \cdot D_2(3)} \right\rceil \tag{14}$$

$$\Longrightarrow$$

$$\left\lceil \left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(1)}{r_2(2) \cdot D_2(3)} + 1 + r_2(1) \cdot r_2(3) \cdot D_2(1) \right\rceil \cdot \frac{r_2(2) \cdot D_2(2)}{r_2(3) \cdot D_2(1)} \right\rceil$$
$$< \left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(2)}{r_2(3) \cdot D_2(3)} + r_2(1) \cdot r_2(2) \cdot D_2(2) \right\rceil \tag{15}$$
$$\Longrightarrow$$
$$\left\lceil (\left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(1)}{r_2(2) \cdot D_2(3)} + 1 \right\rceil + r_2(1) \cdot r_2(3) \cdot D_2(1)) \cdot \frac{r_2(2) \cdot D_2(2)}{r_2(3) \cdot D_2(1)} \right\rceil$$
$$< \left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(2)}{r_2(3) \cdot D_2(3)} \right\rceil + r_2(1) \cdot r_2(2) \cdot D_2(2) \tag{16}$$
$$\text{because } r_2(1) \cdot r_2(3) \cdot D_2(1) \in \mathbb{N}$$
$$\Longrightarrow$$
$$\left\lceil \left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(1)}{r_2(2) \cdot D_2(3)} + 1 \right\rceil \cdot \frac{r_2(2) \cdot D_2(2)}{r_2(3) \cdot D_2(1)} + r_2(1) \cdot r_2(2) \cdot D_2(2) \right\rceil$$
$$< \left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(2)}{r_2(3) \cdot D_2(3)} \right\rceil + r_2(1) \cdot r_2(2) \cdot D_2(2) \tag{17}$$
$$\Longrightarrow$$
$$\left\lceil \left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(1)}{r_2(2) \cdot D_2(3)} + 1 \right\rceil \cdot \frac{r_2(2) \cdot D_2(2)}{r_2(3) \cdot D_2(1)} \right\rceil + r_2(1) \cdot r_2(2) \cdot D_2(2)$$
$$< \left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(2)}{r_2(3) \cdot D_2(3)} \right\rceil + r_2(1) \cdot r_2(2) \cdot D_2(2) \tag{18}$$
$$\text{because } r_2(1) \cdot r_2(2) \cdot D_2(2) \in \mathbb{N}$$
$$\Longrightarrow$$
$$\left\lceil \left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(1)}{r_2(2) \cdot D_2(3)} + 1 \right\rceil \cdot \frac{r_2(2) \cdot D_2(2)}{r_2(3) \cdot D_2(1)} \right\rceil$$
$$< \left\lceil \mathcal{N}(3) \cdot \frac{r_2(1) \cdot D_2(2)}{r_2(3) \cdot D_2(3)} \right\rceil \tag{19}$$

Now we have removed all parts which are repeated and which lead to the same result as the sequence of natural numbers from 1 to $\mathcal{N}(3)$. So we find out if 3-process cycles exist by checking the inequality 5 of the theorem 4.6

for $\tilde{N} \in \{1 \cdots \mathcal{N}(3)\}$. *QED*

However, if all pairs of factors of the NMC are relatively prime, the inequality must be checked for all $\tilde{N} \in \{1 \cdots N(3)\}$.

# 6 Enlarging for k Traces

## 6.1 Enlarging for four Traces

In this section we extend the results of section 5 to communication processes with four communication processors and then to communication processes with $k$ communication processors. The corresponding communication graph is a cycle for each case which we consider in this section.

For a communication process with four communcation processors the NMC turns to:

$$\sigma(1,2) \cdot \sigma(2,3) \cdot \sigma(3,4) \cdot \sigma(4,1) = \sigma(1,4) \cdot \sigma(4,3) \cdot \sigma(3,2) \cdot \sigma(2,1) \qquad (20)$$

**Theorem 6.1** *Let $G_4 \in \mathcal{C}_4$ be an EMFG and matrix $\Sigma(G_4)$. Let $\Sigma(G_4)$ holds the NMC. Let $N(4) \neq 0$.*
*If one of the pairs $(\sigma(i,j), \sigma(i',j'))$ with $\sigma(i,j), \sigma(i',j') \in \{\sigma(1,2),\ \sigma(2,3),\ \sigma(3,4),\ \sigma(4,1)\}$ is not relatively prime, then there is a $\mathcal{N}(4) < N(4)$ with the following property:*
*If there is a 4-process cycle in the EMFG $G_4$ then there is such a 4-process cycle in or before level $\mathcal{L}_{\tilde{N}}$, for some $\tilde{N} \in \{1 \cdots \mathcal{N}(4)\}$.*

*Proof:*

Next the greatest common divisor is taken from the pairs of the NMC:

$$D_1(1) = \gcd(\sigma(1,2), \sigma(2,1))$$

$$D_1(2) = \gcd(\sigma(2,3), \sigma(3,2))$$

$$D_1(3) = \gcd(\sigma(3,4), \sigma(4,3))$$

$$D_1(4) = \gcd(\sigma(1,4), \sigma(4,1))$$

Now it can be defined:

$$\sigma(1,2) = r(1,2) \cdot D_1(1)$$

$$\sigma(2,1) = r(2,1) \cdot D_1(1)$$

$$\sigma(2,3) = r(2,3) \cdot D_1(2)$$

$$\sigma(3,2) = r(3,2) \cdot D_1(2)$$

$$\sigma(3,4) = r(3,4) \cdot D_1(3)$$

$$\sigma(4,3) = r(4,3) \cdot D_1(3)$$

$$\sigma(4,1) = r(4,1) \cdot D_1(4)$$

$$\sigma(1,4) = r(1,4) \cdot D_1(4)$$

The greatest common divisor is then taken again:

$$D_2(4) = \gcd(r(2,1), r(4,1))$$

$$D_2(1) = \gcd(r(1,2), r(3,2))$$

$$D_2(2) = \gcd(r(2,3), r(4,3))$$

$$D_2(3) = \gcd(r(1,4), r(3,4))$$

So we can write:

$$\sigma(1,2) = r_2(1,2) \cdot D_2(1) \cdot D_1(1)$$

$$\sigma(2,1) = r_2(2,1) \cdot D_2(4) \cdot D_1(1)$$

$$\sigma(2,3) = r_2(2,3) \cdot D_2(2) \cdot D_1(2)$$

$$\sigma(3,2) = r_2(3,2) \cdot D_2(1) \cdot D_1(2)$$

$$\sigma(3,4) = r_2(3,4) \cdot D_2(3) \cdot D_1(3)$$

$$\sigma(4,3) = r_2(4,3) \cdot D_2(2) \cdot D_1(3)$$

$$\sigma(4,1) = r_2(4,1) \cdot D_2(4) \cdot D_1(4)$$
$$\sigma(1,4) = r_2(1,4) \cdot D_2(3) \cdot D_1(4)$$

Next defined:

$$D_3(1) = \gcd(r_2(1,2), (r_2(4,3))$$
$$D_3(2) = \gcd(r_2(2,3), (r_2(1,4))$$
$$D_3(3) = \gcd(r_2(3,4), (r_2(2,1))$$
$$D_3(4) = \gcd(r_2(4,1), (r_2(3,2))$$

This turns into:

$$\sigma(1,2) = r_3(1,2) \cdot D_3(1) \cdot D_2(1) \cdot D_1(1)$$
$$\sigma(2,1) = r_3(2,1) \cdot D_3(3) \cdot D_2(4) \cdot D_1(1)$$
$$\sigma(2,3) = r_3(2,3) \cdot D_3(2) \cdot D_2(2) \cdot D_1(2)$$
$$\sigma(3,2) = r_3(3,2) \cdot D_3(4) \cdot D_2(1) \cdot D_1(2)$$
$$\sigma(3,4) = r_3(3,4) \cdot D_3(3) \cdot D_2(3) \cdot D_1(3)$$
$$\sigma(4,3) = r_3(4,3) \cdot D_3(1) \cdot D_2(2) \cdot D_1(3)$$
$$\sigma(4,1) = r_3(4,1) \cdot D_3(4) \cdot D_2(4) \cdot D_1(4)$$
$$\sigma(1,4) = r_3(1,4) \cdot D_3(2) \cdot D_2(3) \cdot D_1(4)$$

*Assertion 6.1.1:*

$$r_3(1,2) = r_3(1,4) = r_3(1)$$
$$r_3(2,1) = r_3(2,3) = r_3(2)$$
$$r_3(3,2) = r_3(3,4) = r_3(3)$$
$$r_3(4,1) = r_3(4,3) = r_3(4)$$

If we substitute these expressions into the NMC, it turns into:

$$
\begin{aligned}
& r_3(1,2)\cdot D_3(1)\cdot D_2(1)\cdot D_1(1) \;\cdot\; r_3(2,3)\cdot D_3(2)\cdot D_2(2)\cdot D_1(2) \\
\cdot\; & r_3(3,4)\cdot D_3(3)\cdot D_2(3)\cdot D_1(3) \;\cdot\; r_3(4,1)\cdot D_3(4)\cdot D_2(4)\cdot D_1(4) \\
= \quad & r_3(1,4)\cdot D_3(2)\cdot D_2(3)\cdot D_1(4) \;\cdot\; r_3(4,3)\cdot D_3(1)\cdot D_2(2)\cdot D_1(3) \\
\cdot\; & r_3(3,2)\cdot D_3(4)\cdot D_2(1)\cdot D_1(2) \;\cdot\; r_3(2,1)\cdot D_3(3)\cdot D_2(4)\cdot D_1(1)
\end{aligned}
$$

$$\Longrightarrow$$

$$
\begin{aligned}
& r_3(1,2) \quad r_3(2,3) \quad r_3(3,4) \quad r_3(4,1) \\
= \quad & r_3(1,4) \quad r_3(4,3) \quad r_3(3,2) \quad r_3(2,1)
\end{aligned}
$$

The greatest common divisor is taken from the following pairs:

$$
\begin{aligned}
\sigma(1,2) \quad \text{with} \quad \sigma(i,j) \quad &\in \quad \{\sigma(2,1), \sigma(3,2), \sigma(4,3)\} \\
\sigma(2,1) \quad \text{with} \quad \sigma(i,j) \quad &\in \quad \{\sigma(1,2), \sigma(3,4), \sigma(4,1)\} \\
\sigma(2,3) \quad \text{with} \quad \sigma(i,j) \quad &\in \quad \{\sigma(1,4), \sigma(3,2), \sigma(4,3)\} \\
\sigma(3,2) \quad \text{with} \quad \sigma(i,j) \quad &\in \quad \{\sigma(1,2), \sigma(2,3), \sigma(4,1)\} \\
\sigma(3,4) \quad \text{with} \quad \sigma(i,j) \quad &\in \quad \{\sigma(1,4), \sigma(2,1), \sigma(4,3)\} \\
\sigma(4,3) \quad \text{with} \quad \sigma(i,j) \quad &\in \quad \{\sigma(1,2), \sigma(2,3), \sigma(3,4)\} \\
\sigma(4,1) \quad \text{with} \quad \sigma(i,j) \quad &\in \quad \{\sigma(2,1), \sigma(3,2), \sigma(1,4)\} \\
\sigma(1,4) \quad \text{with} \quad \sigma(i,j) \quad &\in \quad \{\sigma(2,3), \sigma(3,4), \sigma(4,1)\}
\end{aligned}
$$

Therefore it is possible that only:

$$r_3(1,2) \mid r_3(1,4)$$

$$r_3(2,1) \mid r_3(2,3)$$

$$r_3(3,2) \mid r_3(3,4)$$

$$r_3(4,1) \mid r_3(4,3)$$

and

$$r_3(1,4) \mid r_3(1,2)$$

$$r_3(2,3) \mid r_3(2,1)$$

$$r_3(3,4) \mid r_3(3,2)$$

$$r_3(4,3) \mid r_3(4,1)$$

are true. But this means that it must be:

$$r_3(1,2) = r_3(1,4)$$

$$r_3(2,1) = r_3(2,3)$$

$$r_3(3,2) = r_3(3,4)$$

$$r_3(4,1) = r_3(4,3)$$

So the assertion 6.1.1 is true and we may use it for factorisation. Again, like in Section 5, the inequality in $N(4)$ is reduced:

$$\left\lceil \left\lceil \left\lceil N(4) \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} + 1 \right\rceil \cdot \frac{\sigma(3,4)}{\sigma(4,3)} \right\rceil < \left\lceil N(4) \cdot \frac{\sigma(1,4)}{\sigma(4,1)} \right\rceil \tag{21}$$

$$\Longrightarrow$$

$$\begin{aligned} &\left\lceil \left\lceil \left\lceil N(4) \cdot \frac{r_3(1) \cdot D_3(1) \cdot D_2(1) \cdot D_1(1)}{r_3(2) \cdot D_3(3) \cdot D_2(4) \cdot D_1(1)} + 1 \right\rceil \right. \\ &\left. \cdot \frac{r_3(2) \cdot D_3(2) \cdot D_2(2) \cdot D_1(2)}{r_3(3) \cdot D_3(4) \cdot D_2(1) \cdot D_1(2)} + 1 \right\rceil \cdot \\ &\left. \frac{r_3(3) \cdot D_3(3) \cdot D_2(3) \cdot D_1(3)}{r_3(4) \cdot D_3(1) \cdot D_2(2) \cdot D_1(3)} \right\rceil \\ < &\left\lceil N(4) \cdot \frac{r_3(1) \cdot D_3(2) \cdot D_2(3) \cdot D_1(4)}{r_3(4) \cdot D_3(4) \cdot D_2(4) \cdot D_1(4)} \right\rceil \end{aligned} \tag{22}$$

$$\Longrightarrow$$

$$\begin{aligned} &\left\lceil \left\lceil \left\lceil N(4) \cdot \frac{r_3(1) \cdot D_3(1) \cdot D_2(1)}{r_3(2) \cdot D_3(3) \cdot D_2(4)} + 1 \right\rceil \right. \\ &\left. \cdot \frac{r_3(2) \cdot D_3(2) \cdot D_2(2)}{r_3(3) \cdot D_3(4) \cdot D_2(1)} + 1 \right\rceil \\ &\left. \cdot \frac{r_3(3) \cdot D_3(3) \cdot D_2(3)}{r_3(4) \cdot D_3(1) \cdot D_2(2)} \right\rceil \end{aligned}$$

$$< \left\lceil N(4) \cdot \frac{r_3(1) \cdot D_3(2) \cdot D_2(3)}{r_3(4) \cdot D_3(4) \cdot D_2(4)} \right\rceil \tag{23}$$

Set:

$$N(4) = \mathcal{N}(4) + r_3(2) \cdot r_3(3) \cdot r_3(4) \cdot D_3(3) \cdot D_3(4) \cdot D_2(4)$$

If this is put into the inequality, it turns into:

$$\lceil \lceil \lceil (\mathcal{N}(4) + r_3(2) \cdot r_3(3) \cdot r_3(4) \cdot D_3(3) \cdot D_3(4) \cdot D_2(4))$$

$$\cdot \frac{r_3(1) \cdot D_3(1) \cdot D_2(1)}{r_3(2) \cdot D_3(3) \cdot D_2(4)} + 1 \right\rceil \cdot \frac{r_3(2) \cdot D_3(2) \cdot D_2(2)}{r_3(3) \cdot D_3(4) \cdot D_2(1)} + 1 \right\rceil \cdot \frac{r_3(3) \cdot D_3(3) \cdot D_2(3)}{r_3(4) \cdot D_3(1) \cdot D_2(2)} \right\rceil$$

$$< \left\lceil (\mathcal{N}(4) + r_3(2) \cdot r_3(3) \cdot r_3(4) \cdot D_3(3) \cdot D_3(4) \cdot D_2(4)) \cdot \frac{r_3(1) \cdot D_3(2) \cdot D_2(3)}{r_3(4) \cdot D_3(4) \cdot D_2(4)} \right\rceil \tag{24}$$

$$\Longrightarrow$$

$$\left\lceil \left\lceil \left\lceil \mathcal{N}(4) \cdot \frac{r_3(1) \cdot D_3(1) \cdot D_2(1)}{r_3(2) \cdot D_3(3) \cdot D_2(4)} \right.\right.\right.$$

$$+ r_3(1) \cdot r_3(3) \cdot r_3(4) \cdot D_3(1) \cdot D_3(4) \cdot D_2(1) + 1 \rceil$$

$$\left.\left. \cdot \frac{r_3(2) \cdot D_3(2) \cdot D_2(2)}{r_3(3) \cdot D_3(4) \cdot D_2(1)} + 1 \right\rceil \cdot \frac{r_3(3) \cdot D_3(3) \cdot D_2(3)}{r_3(4) \cdot D_3(1) \cdot D_2(2)} \right\rceil$$

$$< \left\lceil \mathcal{N}(4) \cdot \frac{r_3(1) \cdot D_3(2) \cdot D_2(3)}{r_3(4) \cdot D_3(4) \cdot D_2(4)} + r_3(1) \cdot r_3(2) \cdot r_3(3) \cdot D_3(2) \cdot D_3(3) \cdot D_2(3) \right\rceil \tag{25}$$

$$\Longrightarrow$$

$$\left\lceil \left\lceil \left\lceil \mathcal{N}(4) \cdot \frac{r_3(1) \cdot D_3(1) \cdot D_2(1)}{r_3(2) \cdot D_3(3) \cdot D_2(4)} + 1 \right\rceil \cdot \frac{r_3(2) \cdot D_3(2) \cdot D_2(2)}{r_3(3) \cdot D_3(4) \cdot D_2(1)} \right.\right.$$

$$\left.\left. + r_3(1) \cdot r_3(2) \cdot r_3(4) \cdot D_3(1) \cdot D_3(2) \cdot D_2(2) + 1 \rceil \cdot \frac{r_3(3) \cdot D_3(3) \cdot D_2(3)}{r_3(4) \cdot D_3(1) \cdot D_2(2)} \right\rceil \right.$$

$$< \left\lceil \mathcal{N}(4) \cdot \frac{r_3(1) \cdot D_3(2) \cdot D_2(3)}{r_3(4) \cdot D_3(4) \cdot D_2(4)} \right\rceil + r_3(1) \cdot r_3(2) \cdot r_3(3) \cdot D_3(2) \cdot D_3(3) \cdot D_2(3) \tag{26}$$

$$\Longrightarrow$$

$$\left\lceil \left\lceil \left\lceil \mathcal{N}(4) \cdot \frac{r_3(1) \cdot D_3(1) \cdot D_2(1)}{r_3(2) \cdot D_3(3) \cdot D_2(4)} + 1 \right\rceil \cdot \frac{r_3(2) \cdot D_3(2) \cdot D_2(2)}{r_3(3) \cdot D_3(4) \cdot D_2(1)} + 1 \right\rceil \right.$$

$$\cdot\frac{r_3(3)\cdot D_3(3)\cdot D_2(3)}{r_3(4)\cdot D_3(1)\cdot D_2(2)} + r_3(1)\cdot r_3(2)\cdot r_3(3)\cdot D_3(2)\cdot D_3(3)\cdot D_2(3)\Bigg\rceil$$

$$< \left\lceil \mathcal{N}(4)\cdot\frac{r_3(1)\cdot D_3(2)\cdot D_2(3)}{r_3(4)\cdot D_3(4)\cdot D_2(4)}\right\rceil + r_3(1)\cdot r_3(2)\cdot r_3(3)\cdot D_3(2)\cdot D_3(3)\cdot D_2(3) \quad (27)$$

$$\Longrightarrow$$

$$\Bigg\lceil\Bigg\lceil\Bigg\lceil\mathcal{N}(4)\cdot\frac{r_3(1)\cdot D_3(1)\cdot D_2(1)}{r_3(2)\cdot D_3(3)\cdot D_2(4)}+1\Bigg\rceil\cdot\frac{r_3(2)\cdot D_3(2)\cdot D_2(2)}{r_3(3)\cdot D_3(4)\cdot D_2(1)}+1\Bigg\rceil$$

$$\cdot\frac{r_3(3)\cdot D_3(3)\cdot D_2(3)}{r_3(4)\cdot D_3(1)\cdot D_2(2)}\Bigg\rceil + r_3(1)\cdot r_3(2)\cdot r_3(3)\cdot D_3(2)\cdot D_3(3)\cdot D_2(3)$$

$$< \left\lceil \mathcal{N}(4)\cdot\frac{r_3(1)\cdot D_3(2)\cdot D_2(3)}{r_3(4)\cdot D_3(4)\cdot D_2(4)}\right\rceil + r_3(1)\cdot r_3(2)\cdot r_3(3)\cdot D_3(2)\cdot D_3(3)\cdot D_2(3) \quad (28)$$

$$\Longrightarrow$$

$$\Bigg\lceil\Bigg\lceil\Bigg\lceil\mathcal{N}(4)\cdot\frac{r_3(1)\cdot D_3(1)\cdot D_2(1)}{r_3(2)\cdot D_3(3)\cdot D_2(4)}+1\Bigg\rceil$$

$$\cdot\frac{r_3(2)\cdot D_3(2)\cdot D_2(2)}{r_3(3)\cdot D_3(4)\cdot D_2(1)}+1\Bigg\rceil$$

$$\cdot\frac{r_3(3)\cdot D_3(3)\cdot D_2(3)}{r_3(4)\cdot D_3(1)\cdot D_2(2)}\Bigg\rceil$$

$$< \left\lceil \mathcal{N}(4)\cdot\frac{r_3(1)\cdot D_3(2)\cdot D_2(3)}{r_3(4)\cdot D_3(4)\cdot D_2(4)}\right\rceil \quad (29)$$

Similar to the proof of theorem 5.1 we removed all repeated parts and have therefore shown that we only check the inequality 5 of theorem 4.6 for $\tilde{N} \in \{1\cdots\mathcal{N}(4)\}$ to prove, whether 4-process cycles exist. *QED*

Therefore, $N(4)$ can be reduced to $\mathcal{N}(4)$ with $N(4) = \mathcal{N}(4) + r_3(2)\cdot r_3(3)\cdot r_3(4)\cdot D_3(3)\cdot D_3(4)\cdot D_2(4)$, unless all factors of the NMC are relatively prime.

## 6.2 Enlarging for k Traces

Now we extend the results of the subsection 6.1 to communication processes with $k$ communication processors. We use the same techniques as in section 5 and subsection 6.1.

**Theorem 6.2** *Let $G_k \in \mathcal{C}_k$ be an EMFG and matrix $\Sigma(G_k)$. Let $\Sigma(G_k)$ obeys the NMC. Let $N(k) \neq 0$.*
*If one of the pairs $(\sigma(i,(i+1) \bmod k), \sigma((i+1) \bmod k, i))$, $i \in \{1 \cdots k\}$, is not relatively prime, then there is a $\mathcal{N}(k) < N(k)$ with the following property: If there is a $k$-process cycle in the EMFG $G_k$ then there is such a $k$-process cycle in or before level $\mathcal{L}_{\tilde{N}}$, for some $\tilde{N} \in \{1 \cdots \mathcal{N}(k)\}$.*

*Proof:*

First we take the greatest common divisors of the $\sigma(i,j)$, $i,j \in \{1 \cdots k\}$, $i \neq j$, and write:
$\sigma(i,(i+1) \bmod k) =$

$$r(i,(i+1) \bmod k) \cdot \prod_{\substack{j=1, \\ j \neq (k+i-1) \bmod k}}^{k} \gcd(\sigma(i,(i+1) \bmod k), \sigma((j+1) \bmod k, j))$$

and
$\sigma((i+1) \bmod k, i) =$

$$r((i+1) \bmod k, i) \cdot \prod_{\substack{j=1, \\ j \neq (i+1) \bmod k}}^{k} \gcd(\sigma((i+1) \bmod k, i), \sigma(j,(j+1) \bmod k))$$

We insert this into the NMC:

$$\prod_{i=1}^{k}(r(i,(i+1) \bmod k)$$
$$\cdot \prod_{\substack{j=1,\\ j\neq(k+i-1) \bmod k}}^{k} \gcd(\sigma(i,(i+1) \bmod k), \sigma((j+1) \bmod k, j)))$$
$$=$$
$$\prod_{i=1}^{k}(r((i+1) \bmod k, i)$$
$$\cdot \prod_{\substack{j=1,\\ j\neq(i+1) \bmod k}}^{k} \gcd(\sigma((i+1) \bmod k, i), \sigma(j,(j+1) \bmod k)))$$
$$\Longleftrightarrow$$
$$\prod_{i=1}^{k} r(i,(i+1) \bmod k) = \prod_{i=1}^{k} r((i+1) \bmod k, i)$$

It is true that:

$$\sigma(i,(i+1) \bmod k) \mid \sigma((j+1) \bmod k, j)$$
$$\forall i,j \in \{1 \cdots k\}, j \neq (k+i-1) \bmod k$$
and
$$\sigma((i+1) \bmod k, i) \mid \sigma(j,(j+1) \bmod k))$$
$$\forall i,j \in \{1 \cdots k\}, j \neq (i+1) \bmod k$$

Because $\Sigma(G_k)$ obeys the NMC we obtain:
$r(i,(i+1) \bmod k) = r(i,(k+i-1) \bmod k)$
$= \gcd(\sigma(i,(i+1) \bmod k), \sigma(i,(k+i-1) \bmod k))\ \forall i \in \{1 \cdots k\}$

$$\text{Let } N(k) = \mathcal{N}(k) + (\prod_{i=2}^{k} r(i,i-1))$$
$$\cdot(\prod_{i=2}^{k}\prod_{j=i}^{k} \gcd(\sigma(i,i-1), \sigma((j+1) \bmod k, j))).$$

We substitute the $\sigma(i,j)$ and the $N(k)$ in the inequality and simplify:

$$\left\lceil \left\lceil \cdots \left\lceil \left\lceil N(k) \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} + 1 \right\rceil \cdots \right\rceil \cdot \frac{\sigma(k-1,k)}{\sigma(k,k-1)} \right\rceil < \left\lceil N(k) \cdot \frac{\sigma(1,k)}{\sigma(k,1)} \right\rceil$$
$$\Longrightarrow$$
$$\left\lceil \left\lceil \cdots \left\lceil \left\lceil N(k) \cdot \frac{\prod_{j=2}^{k} \gcd(\sigma(1,2), \sigma((j+1) \bmod k, j))}{\prod_{j=2}^{k} \gcd(\sigma(2,1), \sigma(j,(j+1) \bmod k))} + 1 \right\rceil \right.\right.$$

$$\cdot\frac{\prod_{j=1,j\neq 2}^{k}\gcd(\sigma(2,3),\sigma((j+1)\bmod k,j))}{\prod_{j=1,j\neq 2}^{k}\gcd(\sigma(3,2),\sigma(j,(j+1)\bmod k))}+1\Bigg\rceil\cdots\Bigg\rceil$$

$$\cdot\frac{\prod_{j=1,j\neq k-1}^{k}\gcd(\sigma(k-1,k),\sigma((j+1)\bmod k,j))}{\prod_{j=1,j\neq k-1}^{k}\gcd(\sigma(k,k-1),\sigma(j,(j+1)\bmod k))}\Bigg\rceil$$

$$<\left\lceil N(k)\cdot\frac{\prod_{j=1}^{k-1}\gcd(\sigma(1,k),\sigma(j,(j+1)\bmod k))}{\prod_{j=1}^{k-1}\gcd(\sigma(k,1),\sigma((j+1)\bmod k,j))}\right\rceil \tag{30}$$

$\Longrightarrow$

$$\Bigg\lceil\Bigg\lceil\cdots\Bigg\lceil\Bigg\lceil(\mathcal{N}(k)+(\prod_{i=2}^{k}\prod_{j=i}^{k}\gcd(\sigma(i,i-1),\sigma(j,(j+1)\bmod k))))$$

$$\cdot\frac{\prod_{j=2}^{k}\gcd(\sigma(1,2),\sigma((j+1)\bmod k,j))}{\prod_{j=2}^{k}\gcd(\sigma(2,1),\sigma(j,(j+1)\bmod k))}+1\Bigg\rceil$$

$$\cdot\frac{\prod_{j=1,j\neq 2}^{k}\gcd(\sigma(2,3),\sigma((j+1)\bmod k,j))}{\prod_{j=1,j\neq 2}^{k}\gcd(\sigma(3,2),\sigma(j,(j+1)\bmod k))}+1\Bigg\rceil\cdots\Bigg\rceil$$

$$\cdot\frac{\prod_{j=1,j\neq k-1}^{k}\gcd(\sigma(k-1,k),\sigma((j+1)\bmod k,j))}{\prod_{j=1,j\neq k-1}^{k}\gcd(\sigma(k,k-1),\sigma(j,(j+1)\bmod k))}\Bigg\rceil$$

$$<$$

$$\Bigg\lceil(\mathcal{N}(k)+(\prod_{i=2}^{k}\prod_{j=i}^{k}\gcd(\sigma(i,i-1),\sigma(j,(j+1)\bmod k))))$$

$$\cdot\frac{\prod_{j=1}^{k-1}\gcd(\sigma(1,k),\sigma(j,(j+1)\bmod k))}{\prod_{j=1}^{k-1}\gcd(\sigma(k,1),\sigma((j+1)\bmod k,j))}\Bigg\rceil \tag{31}$$

$\Longrightarrow$

$$\Bigg\lceil\Bigg\lceil\cdots\Bigg\lceil\Bigg\lceil\mathcal{N}(k)\cdot\frac{\prod_{j=2}^{k}\gcd(\sigma(1,2),\sigma((j+1)\bmod k,j))}{\prod_{j=2}^{k}\gcd(\sigma(2,1),\sigma(j,(j+1)\bmod k))}+1\Bigg\rceil$$

$$\cdot\frac{\prod_{j=1,j\neq 2}^{k}\gcd(\sigma(2,3),\sigma((j+1)\bmod k,j))}{\prod_{j=1,j\neq 2}^{k}\gcd(\sigma(3,2),\sigma(j,(j+1)\bmod k))}+1\Bigg\rceil\cdots\Bigg\rceil$$

$$\cdot\frac{\prod_{j=1,j\neq k-1}^{k}\gcd(\sigma(k-1,k),\sigma((j+1)\bmod k,j))}{\prod_{j=1,j\neq k-1}^{k}\gcd(\sigma(k,k-1),\sigma(j,(j+1)\bmod k))}\Bigg\rceil$$

$$<\left\lceil\mathcal{N}(k)\cdot\frac{\prod_{j=1}^{k-1}\gcd(\sigma(1,k),\sigma(j,(j+1)\bmod k))}{\prod_{j=1}^{k-1}\gcd(\sigma(k,1),\sigma((j+1)\bmod k,j))}\right\rceil \tag{32}$$

Here we extended the results of removing repeated parts to EMFG's with $k$ traces as in theorem 5.1 and 6.1 and thus the inequality 5 is examined for $\tilde{N} \in \{1 \cdots \mathcal{N}(k)\}$ to prove whether $k$-process cycles exist or not. *QED*

It is possible to obtain a smaller upper bound because of theorem 6.2. However, it is clear that $\mathcal{N}(k)$ grows still exponentially in the number of communication processors and communication statements.

# 7 An Equality in Cycle-Finding

The set of natural numbers for which the inequality 5 of the theorem 4.6 must be examined were determined in the previous sections. In this section, we prove an equality which can also serve as a tool for finding cycles.

Let $\Gamma_3$ be a communication process of circular type with three communication processors. Let $G$ be an EMFG associated to $\Gamma_3$ with the cycle $C$. Let $v_i$, $i \in \{1,2,3\}$, be in the trace $T_i$ and $v_i'$ be later than $v_i$. The cycle $C$ is defined by its undirected edges $(v_1, v_3')$, $(v_1', v_2)$, $(v_2', v_3)$ and the directed pathes $(v_i, v_i')$, $i \in \{1,2,3\}$. The vertices of the cycle in the EMFG are labelled similar to figure 10 on page 23.

**Theorem 7.1** *Let $G_3 \in \mathcal{C}_3$ be an EMFG with matrix $\Sigma(G_3)$. Assume that the NMC holds in $\Sigma(G_3)$. Let $N(3) \neq 0$.*
*Then a 3-process cycle $C$ with the undirected edges $(v_1, v_3')$, $(v_1', v_2)$ and $(v_2', v_3)$ exists in $G_3$, if and only if for some $N \in \{1 \cdots \mathcal{N}(3)\}$ the following condition is satisfied.*

*Condition 1)* $v_1 \in \mathcal{L}_N$

$v_1' \in \mathcal{L}_N$

$v_2 \in \mathcal{L}_{\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$

$v_2' \in \mathcal{L}_{\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$

$v_3 \in \mathcal{L}_{\lceil \lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil \frac{\sigma(2,3)}{\sigma(3,2)} \rceil}$

$v_3' \in \mathcal{L}_i$ *with*

$$i \geq \left\lceil \left\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \right\rceil \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil .$$

*Proof:*

Without any restrictions there is an edge $(v_1, v_3')$ in the EMFG $G$ with $v_1 \in T_1 \cap \mathcal{L}_N$ and $v_3' \in T_3$. Because the communication graph is a cycle there are edges between the trace $T_1$ and the trace $T_2$. The vertex $v_1'$ is later than the vertex $v_1$. So we look at the first edge between the trace $T_1$ and the trace $T_2$ and call it $(v_1', v_2)$. If this edge is in level $\mathcal{L}_N$, the vertex $v_1'$ is in level $\mathcal{L}_N$ and the vertex $v_2$ is in level $\mathcal{L}_{\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$. The level $\mathcal{L}_{\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$ is determined by the numerical proportions like in theorem 4.6. Otherwise the vertex $v_1'$ is in level $\mathcal{L}_N + 1$ because in every level $\mathcal{L}_i$, $i > 0$, there is an edge between the trace $T_1$ and the trace $T_2$. In this case the vertex $v_2$ is in level $\mathcal{L}_{\lceil (N+1) \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$. Now we look at the vertex $v_2'$. The edge $(v_2', v_3)$ lies between the trace $T_2$ and the trace $T_3$. The vertex $v_2'$ is later than the vertex $v_2$. So the vertex $v_2'$ is in the same level as the vertex $v_2$ or one level later. As with the numerical proportions in theorem 4.6, we know in which level the vertex $v_3$ exists.
If the EMFG contains a cycle then the vertex $v_3'$ of the edge $(v_1, v_3')$ is later than the vertex $v_3$. The vertices are in the same levels as above and we exclude the cases *2),3)* and *4)*:

*2)* $v_1 \in \mathcal{L}_N$
$v_1' \in \mathcal{L}_N$
$v_2 \in \mathcal{L}_{\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$
$v_2' \in \mathcal{L}_{\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil + 1}$
$v_3 \in \mathcal{L}_{\lceil \lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \rceil \frac{\sigma(2,3)}{\sigma(3,2)} \rceil}$
$v_3' \in \mathcal{L}_i$ with
$i \geq \left\lceil \left\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil$

*3)* $v_1 \in \mathcal{L}_N$
$v_1' \in \mathcal{L}_N + 1$

$v_2 \in \mathcal{L}_{\lceil (N+1)\cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$

$v_2' \in \mathcal{L}_{\lceil (N+1)\cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$

$v_3 \in \mathcal{L}_{\lceil \lceil (N+1)\cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil \frac{\sigma(2,3)}{\sigma(3,2)} \rceil}$

$v_3' \in \mathcal{L}_i$ with

$i \geq \left\lceil \left\lceil (N+1) \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \right\rceil \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil$

*4)* $v_1 \in \mathcal{L}_N$

$v_1' \in \mathcal{L}_N + 1$

$v_2 \in \mathcal{L}_{\lceil (N+1)\cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$

$v_2' \in \mathcal{L}_{\lceil (N+1)\cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil} + 1$

$v_3 \in \mathcal{L}_{\lceil \lceil (N+1)\cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \rceil \frac{\sigma(2,3)}{\sigma(3,2)} \rceil}$

$v_3' \in \mathcal{L}_i$ with

$i \geq \left\lceil \left\lceil (N+1) \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil$

First we examine case *2)*. By using the numerical proportions we get the inequality 33 like in theorem 4.6:

$$\left\lceil \left\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil \leq \left\lceil N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} \right\rceil \tag{33}$$

By removing the gauss-functions we get:

$$(N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 + x) \cdot \frac{\sigma(2,3)}{\sigma(3,2)} + y \leq N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} + z$$

with $x, y, z \in [0, 1[$

Here the NMC is:

$$\sigma(1,2) \cdot \sigma(2,3) \cdot \sigma(3,1) = \sigma(1,3) \cdot \sigma(3,2) \cdot \sigma(2,1)$$
$$\Longrightarrow \frac{\sigma(1,3)}{\sigma(3,1)} = \frac{\sigma(1,2) \cdot \sigma(2,3)}{\sigma(3,2) \cdot \sigma(2,1)}$$

By using NMC we get:

$$\frac{\sigma(2,3)}{\sigma(3,2)} \cdot (1+x) + y \leq z$$

Without loss of generality we can assume $\frac{\sigma(2,3)}{\sigma(3,2)} > 1$. So there is no cycle. Now the vertices are in the levels of case *3)*. We used the numerical proportions to get the inequality like in theorem 4.6. So we get the following inequality:

$$\left\lceil \left\lceil (N+1) \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil \leq \left\lceil N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} \right\rceil \tag{34}$$

By removing the gauss-functions we get:

$$((N+1) \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + x)\frac{\sigma(2,3)}{\sigma(3,2)} + y \leq N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} + z$$

with $x, y, z \in [0, 1[$

$$\Longrightarrow (N+1) \cdot \frac{\sigma(1,3)}{\sigma(3,1)} + x \cdot \frac{\sigma(2,3)}{\sigma(3,2)} + y \leq N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} + z$$

$$\Longrightarrow \frac{\sigma(1,3)}{\sigma(3,1)} + x \cdot \frac{\sigma(2,3)}{\sigma(3,2)} + y \leq z$$

Without loss of generality we can assume $\frac{\sigma(1,3)}{\sigma(3,1)} > 1$. So there is no cycle. For the case *4)* we get the inequality 35 and we exclude it analogically to the cases *2)* and *3)*.

$$\left\lceil \left\lceil (N+1) \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil \leq \left\lceil N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} \right\rceil \tag{35}$$

The cases *2),3)* and *4)* do not appear. So there is only a cycle if the vertices are in the levels of condition *1)*. *QED*

**Remark 7.2** *Using the same techniques and the same restrictions as in the theorem 4.6, we obtain the inequality 36 from case 1) and the inequality 37*

*from case 2).*

*1)*

$$\left\lceil \left\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil \leq \left\lceil N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} \right\rceil \tag{36}$$

*2)*

$$\left\lceil \left\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} + 1 \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil \leq \left\lceil N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} \right\rceil \tag{37}$$

**Assertion 7.3** *Let $G_3 \in \mathcal{C}_3$ be an EMFG with matrix $\Sigma(G_3)$. Assume $\Sigma(G_3)$ obeys the NMC. Let $N \in \{1 \cdots \mathcal{N}(3)\}$, $\mathcal{N}(3) \neq 0$.*
*Let $v_1 \in T_1 \cap L_N$, $v_1' \in T_1 \cap L_N$, $v_2 \in T_2 \cap L_{\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$, $v_2' \in T_2 \cap L_{\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil}$ and $v_3 \in T_3 \cap L_{\lceil \lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \rceil \frac{\sigma(2,3)}{\sigma(3,2)} \rceil}$.*
*If there is a 3-process cycle then the equality 38 is true for one $N \in \{1 \cdots \mathcal{N}(3)\}$.*

$$\left\lceil \left\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil = \left\lceil N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} \right\rceil \tag{38}$$

*Proof:*

Here NMC is:
$\sigma(1,2) \cdot \sigma(2,3) \cdot \sigma(3,1) = \sigma(1,3) \cdot \sigma(3,2) \cdot \sigma(2,1)$
$\iff \frac{\sigma(1,3)}{\sigma(3,1)} = \frac{\sigma(1,2)}{\sigma(2,1)} \cdot \frac{\sigma(2,3)}{\sigma(3,2)}$
So it is true that:

$$\left\lceil N \cdot \frac{\sigma(1,3)}{\sigma(3,1)} \right\rceil = \left\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil \leq \left\lceil \left\lceil N \cdot \frac{\sigma(1,2)}{\sigma(2,1)} \right\rceil \cdot \frac{\sigma(2,3)}{\sigma(3,2)} \right\rceil$$

We know by the remark 7.2 that the inequality 36 is true. As a result, we obtain the equation and know it is false if no cycles exist. *QED*

Now we have developed the equality 38 for finding cycles in an EMFG. However, $\mathcal{N}(3)$ grows still exponentially in the number of communication processors and communication statements.

# 8 Finding a Cycle by Matrix-Multiplication

Up to the point, upper bounds for searching cycles were developed and the specialized case that the communication graph is a tree was also examined. In this section the matrix-multiplication-model, which is used for the concrete search for cycles, is developed.

The EMFG is transformed into a relation-matrix and by multiplying the relation-matrix with itself it is found out whether there are cycles in the EMFG. The results of Alan Mackworth concerning the use of relation-matrices in [Mac77] are basically used for the complete matrix-model.

**Definition 8.1** *Let $G$ be an EMFG with matrix $\Sigma(G)$. $\Sigma(G)$ satisfies the NMC. The upper bound has the size $N(\Sigma)$. Let $\bar{N}$ be the number of vertices in the subgraph $\mathcal{L}_0 \dot{\cup} \cdots \dot{\cup} \mathcal{L}_{N(\Sigma)}$.*
*The relation-matrix $\mathcal{M}$ is a $\bar{N} \times \bar{N}$ matrix with the entries 1 and 0. Every vertex $v_i$ in the set $(v_1, \cdots, v_{\bar{N}})$ is assigned to one row and one column in the relation-matrix. The value 1 is entered at $m_{ij}$, if there is a directed edge $(v_i, v_j)$. An undirected edge $(v_k, v_l)$ forces the entry 1 at $m_{kl}$ and $m_{lk}$.*

The size of the relation-matrix depends on the size of $\bar{N}$, due to the controlled part of the EMFG being only big enough if there is no undirected edge which starts and does not end in the examined part.

**Definition 8.2** *The product $C$ of two Matrices $A$ and $B$ are defined by the multiplication of the rows with the columns:*

$$C = A \cdot B \text{ with}$$
$$c_{ij} = \sum_{k=1}^{\bar{N}} a_{ik} \cdot b_{kj}$$

The matrix, which is defined by Alan Mackworth, is based on relations between vertices. If the matrix is multiplied with itself and it converges, the problem of the existence of cycles is solved. In this case, this means that if the matrix converges, the EMFG contains no cycles. If the $k$-times multiplicated matrix $\mathcal{M}$ with $k \to \infty$ does not converge, the EMFG contains a cycle and the corresponding communication process contains a deadlock.

The model is illustrated with two examples to clarify it. In the first one a EMFG with three traces and with eight vertices is examined (example 1). In order to create the accompanying $8 \times 8$ relation-matrix $\mathcal{M} = (m_{ij})_{i=8,j=8}$, a $9 \times 9$ table is taken. In the uppermost row and in the first left column the names of the vertices are written. The EMFG contains the following directed edges:

$$(v_1, v_4), (v_2, v_5), (v_3, v_6), (v_5, v_7), (v_6, v_8) \tag{39}$$

And it contains the following undirected:

$$(v_1, v_6), (v_2, v_3), (v_4, v_5), (v_7, v_8) \tag{40}$$

Hence the entry is 1 for

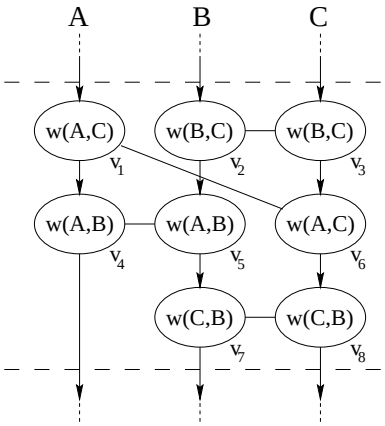
$$m_{14}, m_{25}, m_{36}, m_{57}, m_{68}, \tag{41}$$

$$m_{16}, m_{23}, m_{32}, m_{45}, m_{54}, m_{61}, m_{78}, m_{87} \tag{42}$$

and 0 besides. So, the table at the right side in example 1 is constructed. If the relation-matrix which belongs to the table, is taken and multiplied with itself an infinite amount of times, it converges.

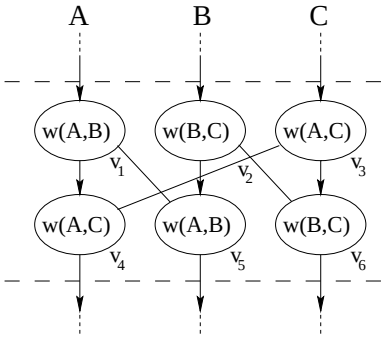
In the second example the EMFG is based on a communication graph which causes a deadlock. This means that the EMFG contains a cycle (example 2). The table (the right side of example 2) is constructed like the first example, and the relation-matrix is multiplied with itself, going to infinity.

Now this matrix-model can be used for finding deadlocks.




| | $v_1$ | $v_2$ | $v_3$ | $v_4$ | $v_5$ | $v_6$ | $v_7$ | $v_8$ |
|---|---|---|---|---|---|---|---|---|
| $v_1$ | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| $v_2$ | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| $v_3$ | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| $v_4$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $v_5$ | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| $v_6$ | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| $v_7$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| $v_8$ | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

The EMFG and the Matrix of Example 1




| | $v_1$ | $v_2$ | $v_3$ | $v_4$ | $v_5$ | $v_6$ |
|---|---|---|---|---|---|---|
| $v_1$ | 0 | 0 | 0 | 1 | 1 | 0 |
| $v_2$ | 0 | 0 | 0 | 0 | 1 | 1 |
| $v_3$ | 0 | 0 | 0 | 1 | 0 | 1 |
| $v_4$ | 0 | 0 | 1 | 0 | 0 | 0 |
| $v_5$ | 1 | 0 | 0 | 0 | 0 | 0 |
| $v_6$ | 0 | 1 | 0 | 0 | 0 | 0 |

The EMFG and the Matrix of Example 2

# 9 Parallel Discrete Matrix-Multiplication-Algorithms

Here, an application for the model which was decribed in the previous section is presented. Now algorithms for matrix-multiplication are introduced. They are from [Lei92].

Although the matrix-exponentiation can be simplified by using Jordan matices, we limit it to matrix-multiplication.

## 9.1 Matrix-Multiplication-Algorithm on a 2-dimensional Array

The first algorithm is computed on a two dimensional array or a mesh. An array is a structured datatype on a uniform element type. The elements of the array are indexed and the type of index of the array is isomorph to the type of index of the matrix.

The $i^{th}$ row of A is taken as the input to the $i^{th}$ column of the array. It starts from the top with $a_{i1}$ at step $i$. The $j^{th}$ column of B is handled in a similar way. It is the input to the $j^{th}$ row of the array that starts from the left with $b_{1j}$ at step $j$. The values of $a_{ik}$ and $b_{kj}$ arrive both at cell $(i, j)$ of the array at step $i + j + k - 2$. There, they are multiplied and the product is added to the local memory, which is first set to zero. The values itself are sent downward initially and rightward respectively. After $i + j + m - 2$ steps the cell $(j, i)$ of the array contains

$$\sum_{k=1}^{m} a_{ik} \cdot b_{kj} = c_{ij} \tag{43}$$

To write the algorithm in a pseudo-code, the processor $p_{ij}$ contains the variables $x_{ij}$, $y_{ij}$ and $z_{ij}$. The inputs are the values of $a_{ij}$ and $b_{ij}$, the outputs are the values of $c_{ij}$. Now it looks like this:

$\backslash *$ init $z_{ij}$ $\backslash *$

*step1* : *cobegin*

*forall* $i, j \in 1 \,..\, m$

$z_{ij} \leftarrow 0$

*coend* ;

$\backslash *$ multiplies the matrices $A$ and $B$ $\backslash *$

*step2* : *for* $l = 1$ *until* $3m - 2$ *do*

*cobegin*

*forall* $i, j \in 1 \,..\, m$ {

$\backslash *$ Every value of the matrix $A$ which has already entered a processor is sent to the next one. $\backslash *$

*if* $(\max\{0; l - m + 1\} < j < \min\{l; m + 1\}$
$\wedge \max\{1; l - m + 1\} < i < \min\{l + 1; m + 1\})$
*then* $\{x_{ij} \leftarrow x_{(i-1)j}\}$

$\backslash *$ Some of the values of the matrix $A$ which are necessary to compute the next value for $z_{ij}$ are entered. $\backslash *$

*if* $(\max\{0; l - m\} < j < \min\{l + 1; m + 1\} \wedge i = 1)$
*then* $\{x_{ij} \leftarrow a_{j(l-j+1)}\}$

$\backslash *$ Every value of the matrix $B$ which has already entered a processor is sent to the next one. $\backslash *$

*if* $(\max\{0; l - m + 1\} < i < \min\{l; m + 1\}$
$\wedge \max\{1; l - m + 1\} < j < \min\{l + 1; m + 1\})$
*then* $\{y_{ij} \leftarrow y_{i(j-1)}\}$

$\backslash *$ Some of the values of the matrix $B$ which are necessary to compute the next value for $z_{ij}$ are entered. $\backslash *$

*if* $(\max\{0; l - m\} < i < \min\{l + 1; m + 1\} \wedge j = 1)$

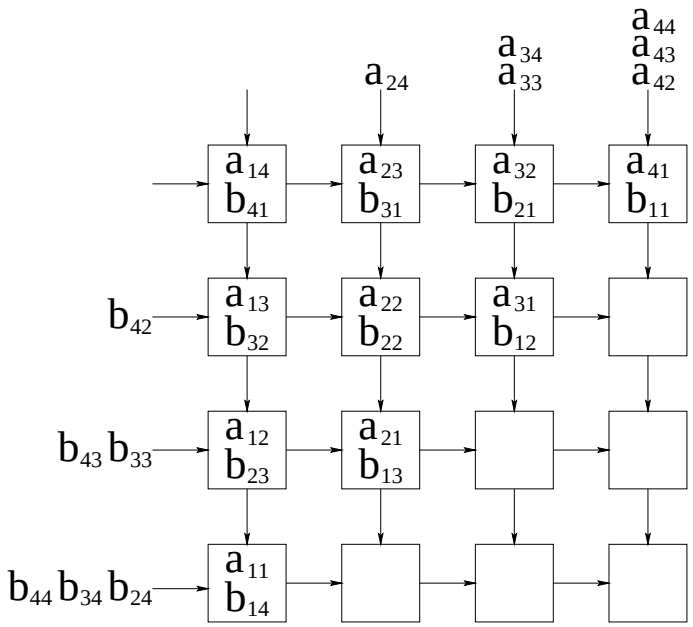



Figure 11: The Matrix-Multiplication-Algorithm on a 2-dimensional Array

$$\textit{then} \ \{y_{i1} \leftarrow b_{(l-i+1)i} \ \}$$

* The local values of the matrices $A$ and $B$ are multiplied with each other and added to the local memory. *

$$\textit{if} \quad (\max\{0; l-m+1\} < i, j < \min\{m+1; l+1\})$$
$$\textit{then} \ \{z_{ij} \leftarrow z_{ij} + x_{ij} \cdot y_{ij}\}$$
$$\}$$

*coend* ;

This algorithm is shown in figure 11. In this case $m$ is set on a specific value: $m = 4$. Each processor or cell is able to compute and to store at least three values.

In the first step of the algorithm the cell in the first row and in the first column receives the values of $a_{11}$ and of $b_{11}$. Both values are multiplied and

the result is saved as the third value named $c_{11}$. Then $a_{11}$ is sent to the next row and $b_{11}$ to the next column. At the same time the values $a_{12}$, $a_{21}$, $b_{12}$ and $b_{21}$ enter the cells like in figure 11. This is done four times until the diagonal is full. Then the values up to three are passed down like before, but those in row four and in column four are thrown away after they have been used for the multiplication. After seven steps of multiplication, the algorithm is finished. This is one possible algorithm, but if the array is replaced by a torus where the values can move in cycles, less steps are necessary.

## 9.2 Matrix-Multiplication-Algorithm on a Torus

On the torus the values are still multiplied and added at each step. The cell $(j, i)$ contains the values of $a_{ik}$ and $b_{kj}$ with $k = m - i - j + 2$ or $k = 2 \cdot m - i - j + 2$ and after $m$ steps it contains $c_{ij}$.

Formally, it looks like this:

$\backslash *$ The values of the matrices A and B are entered. $\backslash *$
*step1* *cobegin*
*forall* $i, j \in 1..m\{$
$\backslash *$ Their cells in the torus are computed. $\backslash *$
*if* $(m + 2 - i - j > 0)$
*then* $\{k = m + 2 - i - j\}$
*else* $\{k = 2m + 2 - i - j\}$
$x_{ij} \leftarrow a_{ik}; y_{ij} \leftarrow b_{kj}; z_{ij} \leftarrow x_{ij} \cdot y_{ij}$
$\}$
*coend* ;
$\backslash *$ The values of the matrices A and B are multiplicated and added. $\backslash *$
*step2* *for* $l = 1$ *until* $m - 1$ *do*
*cobegin*

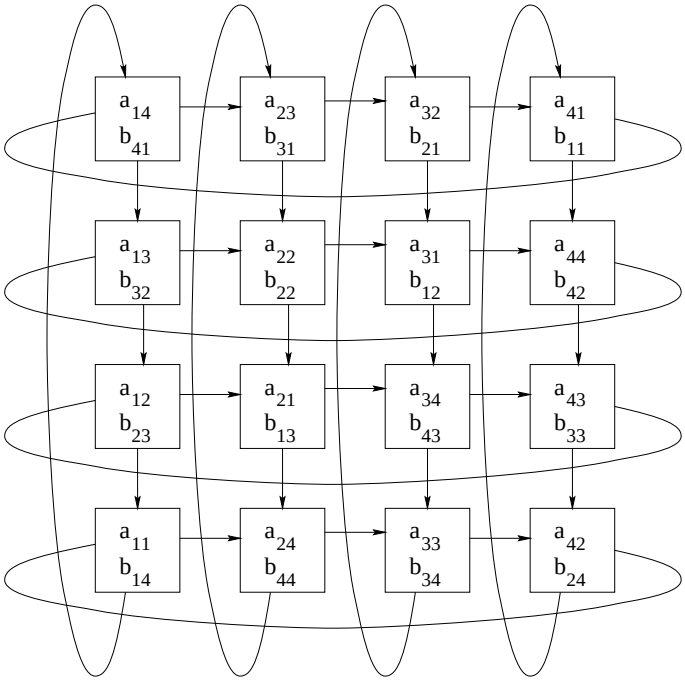



Figure 12: The Matrix-Multiplication-Algorithm on a Torus

$$
\begin{aligned}
&\mathit{forall}\quad i,j \in 1\,..\,m\{\\
&\backslash* \text{ The values of the matrix A are sent to the next cell. } \backslash*\\
&\qquad \mathit{if}\quad (i \neq 1)\\
&\qquad\qquad \mathit{then}\ \ \{x_{ij} \leftarrow x_{(i-1)j}\}\\
&\qquad \mathit{else}\ \ \{x_{1j} \leftarrow x_{mj}\}\\
&\backslash* \text{ The values of the matrix B are sent to the next cell. } \backslash*\\
&\qquad \mathit{if}\quad (j \neq 1)\\
&\qquad\qquad \mathit{then}\ \ \{y_{ij} \leftarrow y_{i(j-1)}\}\\
&\qquad \mathit{else}\ \ \{y_{i1} \leftarrow y_{im}\}\\
&\backslash* \text{ The values of the matrices A and B are multiplicated and added. } \backslash*\\
&\qquad z_{ij} \leftarrow z_{ij} + x_{ij} \cdot y_{ij}\\
&\qquad \}\\
&\mathit{coend}\ \ ;
\end{aligned}
$$

It is easy to see that the algorithm on the torus is a variation of the algorithm on the array. The difference is that now all values of the matrices are entered in the first step and multiplicated and added in the second, instead of being entered at their first use.

## 9.3 Matrix-Multiplication-Algorithm on a 3-dimensional Array

Another algorithm is given by taking sub-matrices and multiplying them. Here the submatrix of A is defined as

$$A_{rs} \text{ with } (1 \leq r, s \leq m^{\frac{1}{4}})$$

It consists of the intersection of the rows, which starts at $(r-1) \cdot \left\lceil m^{\frac{1}{4}} \right\rceil^3 + 1$ and ends at $r \cdot \left\lceil m^{\frac{1}{4}} \right\rceil^3$, and of the columns, which starts at $(s-1) \cdot \left\lceil m^{\frac{1}{4}} \right\rceil^3 + 1$ and ends at $s \cdot \left\lceil m^{\frac{1}{4}} \right\rceil^3$. $B_{rs}$ corresponds to the submatrix of $B$ and $C_{rs}$ to the one of $C$. In both cases the same principle is used as for the submatrix of A.

Hence

$$C_{rs} = \sum_{t=1}^{m^{\frac{1}{4}}} A_{rt} \times B_{ts} \tag{44}$$

The remaining part of this algorithm is analogous to one of the two variations of the first algorithm in this section.

* Init $x_{mn}^{rst}$ and $y_{mn}^{rst}$ *
*step1*
*cobegin*
*forall* $r, s, t \in 1 .. \left\lceil m^{\frac{1}{4}} \right\rceil$
$m, n \in 1 .. \left\lceil m^{\frac{1}{4}} \right\rceil^3$ {

$x_{mn}^{rst} \leftarrow 0;$

$y_{mn}^{rst} \leftarrow 0\}$

*coend* ;

* The matrices A and B are divided into sub-matrices and the values are sent into the corresponding array. *

*step2*

*cobegin*

*forall* $i, j \in 1..m\{$

*if* $(i \ mod \ \lceil m^{\frac{1}{4}} \rceil^3 \neq 0)$

*then* $\{m = i \ mod \ \lceil m^{\frac{1}{4}} \rceil^3\}$

$else\{m = \lceil m^{\frac{1}{4}} \rceil^3\}$

*if* $(j \ mod \ \lceil m^{\frac{1}{4}} \rceil^3 \neq 0)$

*then* $\{n = j \ mod \ \lceil m^{\frac{1}{4}} \rceil^3\}$

$else\{n = \lceil m^{\frac{1}{4}} \rceil^3\}$

*if* $(\lceil m^{\frac{1}{4}} \rceil^3 + 2 - m - n > 0)$

*then* $\{k = \lceil m^{\frac{1}{4}} \rceil^3 + 2 - m - n\}$

$else\{k = \lceil 2 \cdot m^{\frac{1}{4}} \rceil^3 + 2 - m - n\}$

$r = \left\lceil \left( \frac{i}{\lceil m^{\frac{1}{4}} \rceil^3} \right) \right\rceil ;$

$s = \left\lceil \left( \frac{j}{\lceil m^{\frac{1}{4}} \rceil^3} \right) \right\rceil ;$

*forall* $t \in 1..\lceil m^{\frac{1}{4}} \rceil \{$

$x_{mn}^{rst} \leftarrow a_{ij};$

$y_{mn}^{rst} \leftarrow b_{ij};$

$z_{mn}^{rst} \leftarrow x_{mn}^{rst} \cdot y_{mn}^{rst}\}\}$

*coend* ;

* The matrix-multiplication-algorithm on a torus is used to multiply

the sub-matrices. *

*step3* *for* $l = 1$ *until* $\left\lceil m^{\frac{1}{4}} \right\rceil^3 - 1$ *do*

*cobegin*

*forall* $r, s, t \in 1 \,..\, \left\lceil m^{\frac{1}{4}} \right\rceil \{$
$m, n \in 1 \,..\, \left\lceil m^{\frac{1}{4}} \right\rceil^3 \{$
*if* $(m \neq 1)$
*then* $\{x_{mn}^{rst} \leftarrow x_{(m-1),n}^{rst}\}$
*else*$\{x_{mn}^{rst} \leftarrow x^{rst}_{\left\lceil m^{\frac{1}{4}} \right\rceil^3,n}\}$
*if* $(n \neq 1)$
*then* $\{y_{mn}^{rst} \leftarrow y_{m,(n-1)}^{rst}\}$
*else*$\{y_{mn}^{rst} \leftarrow y^{rst}_{m,\left\lceil m^{\frac{1}{4}} \right\rceil^3}\}$
$z_{mn}^{rst} \leftarrow z_{mn}^{rst} + x_{mn}^{rst} \cdot y_{mn}^{rst}\}\}$

*coend* ;

* The sub-matrices are added to get the sub-matrices of C. *

*step4*

*cobegin*

*forall* $r, s, t \in 1 \,..\, \left\lceil m^{\frac{1}{4}} \right\rceil$
$m, n \in 1 \,..\, \left\lceil m^{\frac{1}{4}} \right\rceil^3 \{$
$i = (r-1) \cdot \left\lceil m^{\frac{1}{4}} \right\rceil^3 + m;$
$j = (s-1) \cdot \left\lceil m^{\frac{1}{4}} \right\rceil^3 + n;$
$c_{ij} \leftarrow \sum_{t=1}^{\left\lceil m^{\frac{1}{4}} \right\rceil^3} z_{mn}^{rst}\}$

*coend* ;

## 9.4 Matrix-Multiplication-Algorithm on a 3-dimensional Mesh of Trees

The last algorithm which is introduced is based on a $m \times m \times m$ mesh of trees.

A three-dimensional mesh of trees is a three-dimensional array, in which the vertices along the lines, which are parallel to the three axes of the coordinate system, are connected by binary trees (see figure 13).

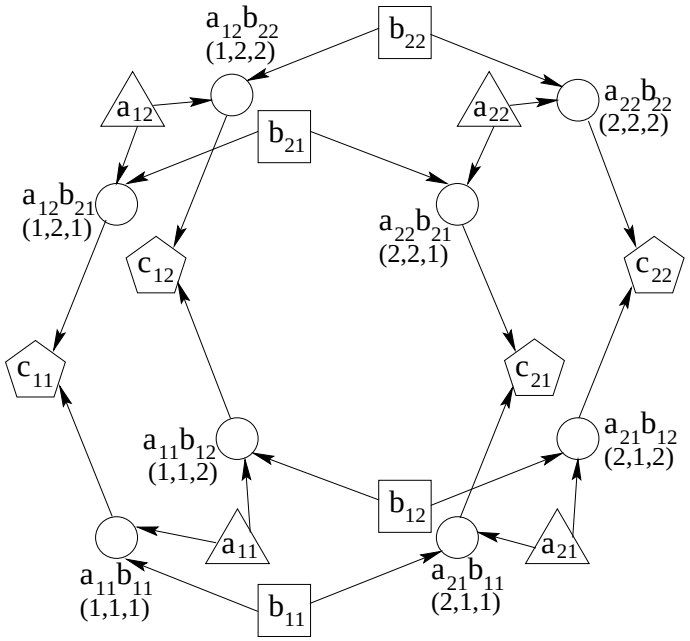



Figure 13: The Matrix-Multiplication-Algorithm on a 3-dimensional Mesh of Trees

At first the value of $a_{ij}$ is entered in the root of the i,j dimension 3 tree for $1 \leq i, j \leq m$ and $b_{jk}$ into the root of the j,k dimension 1 tree for $1 \leq j, k \leq m$.

Then the $A$ and $B$ values are passed downward through the trees so that

at step $\log m + 1$, leaf $(i, j, k)$ receives $a_{ij}$ and $b_{jk}$ for $1 \leq i, j, k \leq m$. This takes $\log m$ steps.

Upon receiving the $A$ and $B$ values from above, each leaf multiplies them and passes the result to its parent in the dimension 2 tree. This means that the value of $a_{ij}b_{jk}$ is sent up the i,k dimensions tree.

In the remaining $\log m$ steps, each dimension 2 tree sums the products computed by its leaves. This is the usual way producing the sum at the root.

The sum at the i,k dimension root is

$$c_{ik} = \sum_{j=1}^{m} a_{ij}b_{jk} \tag{45}$$

which is equal to the $i, k$ entry in the matrix-product

$$C = A \times B \tag{46}$$

* The values of the matrices A and B are entered into the roots. *
*step1*

*cobegin*

*forall* $i, j, k \in 1..m\{$

$r_{ij}^{A} \leftarrow a_{ij}$

$r_{jk}^{B} \leftarrow b_{jk}$

$\}$

*coend* ;

* The values are sent to the leaves and multiplicated. *
*step2*

*cobegin*

*forall* $i, j, k \in 1..m$

$l_{ijk} \leftarrow r_{ij}^{A} \cdot r_{jk}^{B}$

*coend* ;

* The results of the multiplications in the leaves are sent to the roots and added. *

*step3*

*cobegin*

*forall* $i, j, k \in 1 \,..\, m$

$r_{ik}^{C} \leftarrow \sum_{j=1}^{N} l_{ijk}$

*coend* ;

All these are algorithms which can be used for deadlock-finding, but they quickly become quite large and difficult to handle.

# 10 Conclusions

In this paper, three different methods for cycle-finding in the EMFG were introduced and verified.

In one particular case the communication graph is without cycles. The EMFG can only contain 2-precess cycles if the sequence condition is not held. This was proven by theorem 3.3 and theorem 3.6 in section 3.

The next method consists in reducing the size of the part of the EMFG which must be examined. This is done by using the inequality 5 and its generalisation to k traces by the theorem 4.6. On the one hand every common divisor of the fractions on both sides of the inequality was cancelled down to get a lower bound until the inequality 5 must be examined. If the greatest common divisor has to be checked. If the greatest common divisor is a very high number, then this is a good way to shorten the time which is needed for verification. If the numbers are relatively prime, the main problem is still unsolved. On the other hand, the problem is reduced by finding a new upper bound which is lower than the one which is given by inequality 5. Therefore, in section 7, a new bound is found and proven. The number of levels rise exponentially with the number of the subsequences in the communication process. Both methods act in the same way. So we still have a NP-problem.

The third and last method is a common practical suggestion to transfer the problem into a matrix-multiplication problem of relation-matrices, which can be solved by using [Mac77] and [Lei92]. This has been done in sections 8 and 9. As there are many of effective matrix-multiplication-algorithms, an efficient one can be easily and quickly used to obtain results while at the same time using an acceptable amount of memory. Unfortunately, its drawback is

that the size of the matrix is equal to the number of vertices in the levels to be checked. $m$ rises exponentially, so the matrix also rises exponentially as well as the algorithm itself.

In this paper, it can be concluded that I have discovered and developed methods for discovering loop processes by evaluation of the EMFG. These methods have been developed through adapting existing parallel algorithms and arriving at mathematical bounds through the use of the input to these algorithms. This was done by considering the conbinatory structure of EMFG's. We have not yet proven, and we do not yet know whether this input is always polonomial in the size of the original loop processes.
However, let this not be a hindrance to practical use of these methods which are practically the fastest known yet for evaluating deadlocks in loop processes.

# References

[Holt72] Richard C. Holt. Some Deadlock Properties of Computer Systems. *ACM Computing Surveys*, 4[3]:179–196, September 1972.

[Lad, pers.] P. B. Ladkin. Personal communication.

[Lei92] F. Thomson Leighton. *Introduction to Parallel Algorithms and Architectures*. Morgan Kaufmann Publishers, 1992.

[LS9x] P. B. Ladkin and B. B. Simons. *Static Analysis of Interprocess Communications*. Lecture Notes in Computer Science. Springer Verlag, 199x. To appear.

[Mac77] Alan K. Mackworth. Consistency in network relations. *Artifical Intelligence*, 8:99–118, 1977.