Event Control for Deadlock Avoidance in Production Systems With Multiple Capacity Resources

Maria Pia Fanti, Bruno Maione and Biagio Turchiano
Dipartimento di Elettrotecnica ed Elettronica
Politecnico di Bari
Via Re David 200,
70125 Bari
ITALY
e-mail: fanti@poliba.it

Abstract: Deadlock is a critical problem in the control of Automated Flexible Manufacturing Systems. Namely, when a deadlock occurs, the flow of jobs is permanently inhibited so that operations on parts may not be performed. That is why an efficient control policy must avoid deadlock without imposing unnecessary restrictions on part loading and routing.

In this paper we generalize the results of a previous companion work dealing with deadlock avoidance in systems containing single capacity resources. Using the same theoretic framework, we rigorously characterize the deadlock occurrence in multiple resource systems. Digraph theory is still effective for deriving necessary and sufficient conditions characterizing highly undesirable situations (second level deadlocks) that inevitably evolve to circular waits in the near future. The results of the analysis let us introduce some control laws (named restriction policies) that avoid deadlocks in multiple-resource systems by inhibiting or enabling some properly identified events. Finally, the paper discusses and compares the computation costs of the proposed restriction policies.

Keywords: Deadlock Avoidance, discrete event dynamical systems, FMS control

Maria Pia Fanti was born at Siena , Italy , in 1957. She received the degree in Electronic Engineering from University of Pisa, Italy in 1983. She is currently Assistant Professor at the Department of Electrical and Electronic Engineering of the Polytechnic of Bari ,Italy. Her research focuses on structural properties of linear systems and on FMS control and modelling.

Bruno Maione was born in Napoli, Italy, on April 30, 1940. He received the degree in Electrical Engineering with honours from the University of Napoli in 1964. Professor Maione is currently Full Professor of Automatic Control at the Department of Electrical and Electronic Engineering of the Polytechnic of Bari (Italy). He held the position of faculty dean from.1986 to 1992. In 1983 and 1985 he was visiting professor at University of Florida (Gainesville). His primary areas of research and teaching are intelligent control, discrete event dynamical system modeling, systems and control theory.

Biagio Turchiano was born at Bitetto, Bari, Italy, on July 25, 1953. He received the degree in Electrical Engineering with honours from the University of Bari, Italy, in 1979. He joined the Department of Electrical and Electronic Engineering of the Polytechnic of Bari in 1984 as Assistant Researcher. Professor Turchiano is currently Associate Professor of Automatic Control at the Polytechnic of Bari. His research and teaching interests are in the areas of production automation, systems and control theory, modelling and control of discrete event systems.

1. Introduction

Agile and Flexible Manufacturing Systems (FMSs) consist of a number of workstations (machines) capable of performing a set of operations, and of a Material Handling System (MHS) that carries the parts (jobs) among the workstations. Machines and transport units are under the control of a computer or of a network of computers. FMSs can simultaneously process medium-sized volumes of various part types, combining efficiency in using available resources and rapidity in responding to changes on the marketplace. However, due to the concurrence of various jobs competing for limited resources, the management and the operation of automated production systems bear some control problems such as blocking, conflicts and deadlocks. Specifically, deadlocks cause heavy damages because they stall the system activities and prevent parts from flowing. Namely, when a deadlock occurs, transferring some jobs from one resource to another is impossible because each such job is waiting for a resource that is held by parts in the same set. Hence if deadlocks can arise in a manufacturing system it is necessary to characterize these situations and to provide suitable remedies for them.

Approaches to address the problem of deadlocks in FMSs can be classified as i) prevention methods, ii) detection and recovery strategies and iii) avoidance policies. The simplest means of preventing deadlock in automated manufacturing systems is to outlaw circular waits among concurring jobs at the design stage. In other words, the layout must allow all the jobs to flow the same direction or, in any case, must make jobs in production belong only to types following a unidirectional route. However, these approaches unnecessarily limit the manufacturing flexibility by reducing the variety of the part-mix the system could produce and by making choices of alternative routing impossible. Detection and recovery

approaches use a monitoring mechanism for detecting the deadlock occurrence, and a resolution procedure for appropriately preempting some deadlocked resources. Obviously detection and recovery approaches require a continuous monitoring for deadlocks and lead to a reduction in productivity due to the recovery process. Finally, deadlock avoidance schemes in manufacturing processes prevent circular waits to occur by proper operational control of the part flow. This paper falls just into this category.

Many methods for the synthesis of deadlock avoidance controllers use Petri Nets (PNs) as a formalism to describe FMSs and to develop appropriate deadlock avoidance policies. Namely PNs, as a graphical and mathematical tool, provide a uniform environment for analysis of agile and flexible production systems [3,15,20]. In one of the first papers dealing with deadlock in manufacturing systems, Banaszak and Krogh [1] observe that avoidance methods are the most appropriate to face deadlock phenomena. These authors also stress that classical avoidance methods developed by the computer science community are "unduly conservative", since they ignore available information about the organization of the manufacturing processes. Banaszak and Krogh develop an algorithm for a class of PN models formed by a set of sequential processes. The algorithm ensures that the production progress is always allowed by controlling the input of new tokens in a model "zone". For a general class of PN models, Viswanadham et al [16] propose a methodology for establishing both prevention and avoidance control policies. The first one uses the net reachability graph, while the second one develops a look-ahead procedure that searches for deadlocks by simulating the system dynamics over a given number of steps. Since the avoidance policy does not prevent deadlock from occurring, the authors combine this policy with a deadlock recovery method. Hsieh and Chang [11] propose a deadlock avoidance controller for a class of PNs that belong to a larger class than the nets of Banaszak and Krogh. However this class is not as general as the category of PNs considered by Viswanadham et al. More recently, Ezpeleta et al [4] have considered a generalization of the PN model used by Banaszak and Krogh. The new models, called by the authors "Systems of Simple Sequential Processes with Resources", present nice properties characterizing the liveness in terms of structural PN items (siphons). The intensive use of information on the PN structure is one of the main interesting characteristics of this paper. Also Xing et al [18] adopt a PN model that is similar to the nets considered in [1] and emphasizes the concept of deadlock structure for constructing a deadlock avoidance policy. Finally Reveliotis and Ferreira [14] suggest a different approach for developing provably deadlock-free policies. They provide a new characterization of the system state safety by modelling an Automated Manufacturing Cell as a Finite State Automaton.

As an alternative to PN models, Wysk et al [17] introduce a digraph representation of the jobs/resources interactions. The authors present an algorithm for deadlock detection, used in conjunction with resolution procedures. Still following a digraph approach, Cho et al [2] propose several detection methods for part flow deadlocks as well as for particular situations foreboding deadlock conditions and named impending part-flow deadlocks. These procedures are based on the characterization of special types of circuits named "bounded circuits" and are completed by a deadlock resolution methodology. The paper of Kumaran et al [12] improves the methodology of Wysk et al by determining deadlock dynamically on the basis of a dynamically updated digraph. The authors also introduce a deadlock avoidance method that, however, may be computationally complex, if the number of parts/workstations is large.

Recently, Fanti et al [8] have improved the knowledge of deadlock phenomena in flexible production systems by showing that deadlocks can be easily identified by the occurrence of particular figures in two digraphs, associated with the discrete-event dynamical system (DEDS) modelling the manufacturing plant. The first digraph, named Working Procedure Digraph, defines once for all the potential interactions among the whole production mix and the resource sequences required by each part in the mix. The second digraph, named Transition Digraph, describes the current interactions between jobs and resources for each system state. It indicates both the resources currently held by jobs in process and the resources required by the same parts in the near future. Using such digraphs, the authors derive necessary and sufficient conditions for deadlock occurrence and for the identification of critical situations named Second Level Deadlocks (SLD) that are not circular waits, even if they necessarily evolve to deadlocks in the near future. Such conditions also allow the authors to formulate effective control laws (restriction policies) for deadlock avoidance [6-8]. The approach is a general one even if it has

a limit on the fact that it mainly deals with deadlocks in single resource systems. Namely, it easily takes into account multiple items of the same resource type in some well defined cases only.

In this paper the authors generalize the results of a companion work [8], rigorously characterizing deadlocks in multiple-resource systems. Once again the Working Procedure Digraph and the Transition Digraph turn out to be very effective in deriving the necessary and sufficient conditions for deadlock occurrence and for the characterization of the so called Second Level Deadlocks. The digraph theory provides the theoretic framework allowing a clear statement of the results. If the cycle is the main concept underlying the identification of deadlocks in single resource systems, so strong components with particular characteristics provide the theoretical framework for characterizing deadlocks in multiple-resource plants. On this basis, the authors develop control policies that allow concurrent production events leading to high resource utilization and avoiding deadlocks.

The organization of this paper is the following. In Section II we establish notations and basic definitions. In particular we describe the peculiarities of the DEDS modelling the manufacturing system. Moreover we introduce the two digraphs, the Working Procedure Digraph and the Transition Digraph, that are the main tools to state concepts and results of the paper. Finally, in order to clarify notations and nomenclature, we summarize some digraph properties widely used in the following. In Section III we use some particular figures of the Transition Digraph to state a necessary and sufficient condition for a deadlock occurrence and illustrate how to use such a condition for developing deadlock avoidance policies. In Section IV we analyze some particular situations named Second Level Deadlocks that are not actual deadlocks but inevitably give rise to a deadlock in the immediate future. The characterization of Second Level Deadlocks is the key to state the control policies in Section V. Such policies differ in complexity and in the degree of restriction that they impose on the free assignment of the system resources to jobs. Usually, higher degrees of freedom in resource allocation lead to better performance of the production process. Section VI discusses the details of the computation complexity in implementing the proposed control policies. The discussion shows that each policy involves two distinct computation levels. The first one concerns the off-line computations and refers to the algorithms executed only once, before the mix production start-up. On the contrary, the second level concerns the computations performed on-line and in real time. Finally, Section VII draws some concluding remarks. With the purpose of improving the readability of the paper, the most complex proofs are reported in two Appendices.

2. Notations and Basic Definitions

This Section extends the notations already used in the companion paper [8] and introduces new concepts necessary to deal with systems containing multiple capacity resources.

A.The Model

We consider a production system S containing multiple capacity resources (e.g. multiple-slot buffers, pools of identical machines, AGV systems with several trucks, etc.). For the convenience of analysis, we include a fictitious resource r_R the jobs acquire as they leave the system, in the resource set $R=\{r_i, i=1,2,\ldots,R\}$. According to this notation, $C(r_i)$ indicates the capacity of r_i, i.e. the maximum number of jobs that can contemporaneously hold such a resource. Thus $C(r_i)$ is a finite positive integer for $i=1,\ldots,R-1$ and is infinite for $i=R$, i.e. $C(r_R)=\infty$. Finally, we assume that processing each job requires a sequence of resources, named working procedure. Thus, if J is the set of jobs we have to produce, $W=\{\mathbf{w}\}$ denotes the set of all the working procedures, necessary to process all the jobs from J. Obviously r_R is the terminal resource of each $\mathbf{w}\in W$.

We describe the behavior of S as a Discrete Event Dynamical System (DEDS) [5,19], whose state, $\mathbf{q}$, contains the following information on the operating conditions: set $J_\mathbf{q}$ of jobs in process, the resources currently held by each job from $J_\mathbf{q}$, the working procedures associated with such jobs, and, finally, the residual working procedures, i.e. the resources necessary for each $j\in J_\mathbf{q}$ to complete its processing. So, concerning the current state $\mathbf{q}$, HR(j) denotes the resource currently held by $j\in J_\mathbf{q}$, SR(j) and TR(j) identify respectively the second and the third resources of the residual working procedure pertaining to j and, finally,

WP(j) indicates the working procedure of such a job. Note that SR(j) is defined in all the cases. Namely, by assumption, a job leaving the system accedes to r_R, but, at the same time, it is removed from set J_q. On the contrary, TR(j) is defined only if SR(j)$\neq r_R$. We denote the set of all system states by Q.

The system state evolves according to the asynchronous occurrence of certain discrete events. All the events that involve jobs releasing or acquiring resources are relevant in this context. Therefore the DEDS model must encompass the following two types of events:

is the working procedure this job has to follow;

(b) a job progresses from a resource to another one, or it leaves the system (2-type event). This event is specified by a job $j\in J_q$ progressing from HR(j) to SR(j), where $q\in Q$ indicates the current state of S.

B. Two Useful Digraphs

Now we introduce two digraphs: the former, $D_W=(N,E_W)$, named the Working Procedure Digraph, shows the specific order in which the resources appear in all the working procedures.

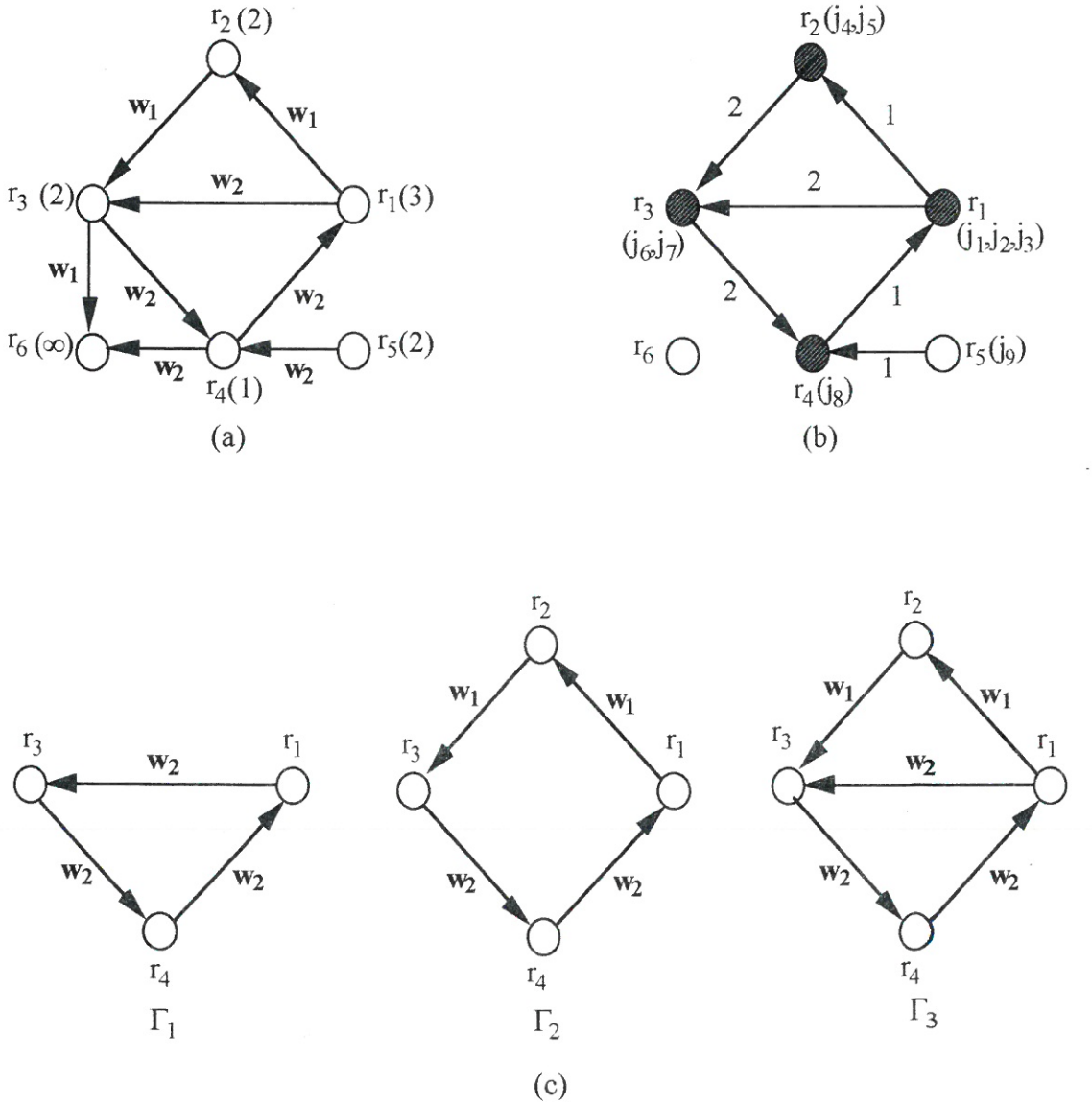

Figure 1. Example 1: (a) Digraph D_W; (b) Digraph $D_{Tr}(q)$; (c) Strong subdigraphs of D_W

(a) a new job enters the system (1-type event). This event is identified by a pair (j,w), where $j\in J$ is the job entering S and $w\in W$

The vertex set $N=\{r_i\}$ denotes the resource set, with the same symbol r_i indicating a generic

vertex or a resource. In geometric diagrams a small cycle (dot) represents a node and a bracketed integer indicates the capacity of the corresponding resource (see Figure 1(a)). Moreover the edge $e_{im}=(r_i,r_m)$, directed from r_i to r_m, belongs to $E_W \subset N \times N$ iff r_m immediately follows r_i in some $\mathbf{w} \in W$. Commonly, e_{im} is said to be "incident out r_i" and "incident into r_m". Furthermore, the label of each edge from E_W denotes the working procedures generating it and the set $E_{\mathbf{w}} \subset E_W$ indicates the edge set labelled by **w**.

The Working Procedure Digraph depends on the mix characteristics only and so it does not vary as the system state **q** changes. We introduce a second digraph, $D_{Tr}(\mathbf{q})=[N,E_{Tr}(\mathbf{q})]$ named Transition Digraph that, on the contrary, depends on the current state. More precisely, while the vertex set still coincides with the resource set and is fixed, the edge set changes as **q** is updated. In any case, $D_{Tr}(\mathbf{q})$ is a subdigraph of D_W because $E_{Tr}(\mathbf{q}) \subset E_W$. In particular, an edge e_{im} from E_W is in $E_{Tr}(\mathbf{q})$ iff a job $j \in J_{\mathbf{q}}$ holds r_i in the state **q** and requires r_m as a next resource. Therefore an edge e_{im} from $E_{Tr}(\mathbf{q})$ is also named "transition e_{im}". In contrast with the Working Procedure Digraph, the Transition Digraph describes the current interactions between jobs and resources by indicating both the resources currently held by jobs from $J_{\mathbf{q}}$ and the resources required by the same jobs at the next step of their working procedures.

Let us note that, if the capacity of resource r_i is greater than one, a single edge $e_{im} \in E_{Tr}(\mathbf{q})$ may represent more jobs detaining r_i and requiring r_m. To specify this situation, we associate the following weight with each edge $e_{im} \in E_{Tr}(\mathbf{q})$

$$a_{\mathbf{q}}(e_{im}) = \mathrm{Card}(\{j \in J_{\mathbf{q}} : HR(j)=r_i \text{ and } SR(j)=r_m\}) \quad (1)$$

where Card(.) stands for "cardinality of ...". Thus $a_{\mathbf{q}}(e_{im})$ yields the number of jobs that in the state **q** hold r_i and request r_m as a next resource.

Now, according to Harary et al [10], we define Outdegree Value of a node r_i the sum of weights of the edges incident out r_i in $D_{Tr}(\mathbf{q})$, i.e.

$$OV_{\mathbf{q}}(r_i) = \sum_{m=1}^{R} a_{\mathbf{q}}(e_{im}) \quad (2)$$

where we consider $a_{\mathbf{q}}(e_{im})=0$ if $e_{im} \notin E_{Tr}(\mathbf{q})$. Hence, with reference to a generic state **q**, the Outdegree Value of a vertex indicates the number of jobs currently using the corresponding resource. In particular, if $OV_{\mathbf{q}}(r_i)=0$ then r_i is *empty;* if $0<OV_{\mathbf{q}}(r_i)<C(r_i)$ then r_i is *idle*; finally, if $OV_{\mathbf{q}}(r_i)=C(r_i)$ then r_i is *busy*. Obviously, a job $j \in J_{\mathbf{q}}$ holding r_i and requiring r_m as a next resource, is *blocked* iff $OV_{\mathbf{q}}(r_m)=C(r_m)$. In this case, also the transition $e_{im} \in E_{Tr}(\mathbf{q})$ is said to be *blocked*. On the contrary, if $OV_{\mathbf{q}}(r_m)<C(r_m)$ then job j is *unblocked* and the transition $e_{im} \in E_{Tr}(\mathbf{q})$ is *feasible*. In the sequel $J_{\mathbf{q},b}$ and $J_{\mathbf{q},u}$ indicate the subsets of $J_{\mathbf{q}}$ collecting *blocked* and *unblocked* jobs, respectively. Moreover $E_F(\mathbf{q})$ denotes the set of *feasible* transitions in the state **q**. Diagrams representing Transition Digraphs use black (white) circles to indicate *busy* (*idle* or *empty*) nodes and specify jobs holding each resource. Edges are labelled with the corresponding weights (see Figure 1(b)).

For any $E^* \subset E_{Tr}(\mathbf{q})$ and $N^* \subset N$, throughout the paper the following notations are used:

$$a_{\mathbf{q}}(E^*) = \sum_{e_{im} \in E^*} a_{\mathbf{q}}(e_{im}) \quad (3a)$$

$$C(N^*) = \sum_{r_i \in N^*} C(r_i) \quad (3b)$$

$$OV_{\mathbf{q}}(N^*) = \sum_{r_i \in N^*} OV_{\mathbf{q}}(r_i) \quad (3c)$$

Before continuing, we shall describe how to update $D_{Tr}(\mathbf{q})$ on the occurrence of an 1-type or a 2-type event. Suppose S in state **q**. For a job $j \in J$ entering S to receive service according to $\mathbf{w} \in W$, the first resource in such a working procedure must expressly be *idle* or *empty*. On

the occurrence of the 1-type event, S makes the transition from $\mathbf{q}$ to a new state $\mathbf{q}'$ where $D_{Tr}(\mathbf{q},\mathbf{w})=[N,E_{Tr}(\mathbf{q},\mathbf{w})]$ indicate the corresponding Transition Digraph, i.e. $D_{Tr}(\mathbf{q}')$. If r_m and r_p are respectively the first and the second resource in $\mathbf{w}$, then the edge set $E_{Tr}(\mathbf{q},\mathbf{w})$ equals $E_{Tr}(\mathbf{q})\cup\{e_{mp}\}$. Clearly it holds: $OV_{\mathbf{q}'}(r_m)=OV_{\mathbf{q}}(r_m)+1$.

Analogously, let $j\in J_{\mathbf{q},u}$ with $r_i=HR(j)$ and $r_m=SR(j)$. By definition of set $J_{\mathbf{q},u}$, r_m is *idle* or *empty* in the state $\mathbf{q}$, so that the transition taking j from r_i to r_m may occur. This 2-type event updates the state from $\mathbf{q}$ to $\mathbf{q}'$. So, denoting by $D_{Tr}(\mathbf{q},j)=[N,E_{Tr}(\mathbf{q},j)]$ the Transition Digraph associated with $\mathbf{q}'$ and putting $r_p=TR(j)$, the edge set $E_{Tr}(\mathbf{q},j)$ results from the following operations executed on $E_{Tr}(\mathbf{q})$:

1. updating $a_{\mathbf{q}'}(e_{im})=a_{\mathbf{q}}(e_{im})-1$ and, if $a_{\mathbf{q}}(e_{im})=1$, removing the edge e_{im} from $E_{Tr}(\mathbf{q})$;

2. putting $a_{\mathbf{q}'}(e_{mp})=a_{\mathbf{q}}(e_{mp})+1$ and, if $a_{\mathbf{q}}(e_{mp})=0$, adding the edge e_{mp} to $E_{Tr}(\mathbf{q})$. Clearly, this operation takes place only if $SR(j)\neq r_R$.

According to operations [1] and [2], r_i gets *empty* (if $OV_{\mathbf{q}}(r_i)=1$) or *idle* (if $OV_{\mathbf{q}}(r_i)>1$), while r_m becomes *busy* (if $OV_{\mathbf{q}}(r_m)=C(r_m)-1$) or *idle* (if $(OV_{\mathbf{q}}(r_m)<C(r_m)-1$). Of course, all the remaining nodes keep the *busy/idle/empty* condition they had in $D_{Tr}(\mathbf{q})$ unchanged.

C. Digraph Properties

Now we recall some digraph properties useful in the sequel [9,10].

Let us consider a generic digraph $D=(N,E)$ where N is the vertex set and E is the edge set. A *walk* in D is an alternating sequence of vertices and arcs, e.g. $r_1, e_{12}, r_2, e_{23},\ldots, r_m$. The *length* of such a walk is the number of occurrences of edges in it. For completeness we call trivial a zero-length walk. A *path* is a walk in which all vertices are distinct; a *cycle* is a nontrivial walk with all nodes but the first and the last distinct[9]. A *selfloop* is an one-edge cycle.

If there is a path from r_i to r_m, then r_m is said to be *reachable* from r_i. Moreover, if r_m is *reachable* from r_i and r_i is *reachable* from r_m, then r_i and r_m are said to be *mutually reachable*.

Given two subdigraphs $D_1=(N_1,E_1)$ and $D_2=(N_2,E_2)$ of D, the union of D_1 and D_2 is defined as the subdigraph $D_1\cup D_2=(N_1\cup N_2,E_1\cup E_2)$. A subdigraph of D is *strong* iff every two vertices are mutually reachable. We note that a subdigraph of D consisting of exactly one vertex is strong. Obviously, the only strong acyclic subdigraph is the one consisting of exactly one node. A digraph with just one node is called trivial.

Finally a *strong component* of a digraph is a maximal strong subdigraph, i.e. no larger strong subdigraph (with more nodes or edges) contains it. Each node belongs exactly to one strong component; each edge is contained in at most one strong component. Furthermore, an edge lies in one strong component iff it is in a cycle [10].

By construction, both digraphs D_W and $D_{Tr}(\mathbf{q})$ contain no selfloop. Moreover in the following we use the expression "strong subdigraphs", "cycles" and "strong components" referring to nontrivial subgraphs only, i.e. containing more than one vertex. Only such nontrivial figures, indeed, play an essential role in stating the results of this paper.

Now let $\gamma=(N_\gamma,E_\gamma)$ be a cycle of D_W and let $\Gamma=(N_\Gamma,E_\Gamma)$ be a strong subdigraph of D_W. Using notation (3b), we define *Cycle Capacity* of γ and *Strong Subdigraph Capacity* of Γ the integers $C(N_\gamma)$ and $C(N_\Gamma)$, respectively .

3. Necessary and Sufficient Conditions for Deadlock

Deadlock is a situation in which a set of parts is in "circular wait" condition. This means that pieces in such a set request indefinitely resources held by other parts in the same set. To make these concepts more precise, this Section introduces a formal definition of deadlock state for a system S with non-unitary capacity resources.

Definition 1 We say that $\mathbf{q}\in Q$ is a deadlock state for S if there exist two non-empty subsets

$J_D \subset J_q$ and $R_D \subset R$, satisfying the following properties:

D1a) J_D is the maximal subset of J_q such that $HR(J_D)=R_D$

D1b) $SR(J_D) \subset R_D$;

D1c) all the resources of $SR(J_D)$ are *busy*.

We specify that in the above definition and in the sequel the symbol $A \subset B$ means that set A is contained by or equal to B. By Definition 1, each job in J_D remains blocked because it is indefinitely waiting for a *busy* resource held by other jobs in J_D.

A deadlock state is associated with particular figures in the Transition Digraph. To clarify this point we introduce the following definition.

Definition 2. Let $\sigma=(N_\sigma, E_\sigma)$ be a strong component of $D_{Tr}(\mathbf{q})$. We call σ a *"Maximal-weight Zero-outdegree Strong Component"* in $D_{Tr}(\mathbf{q})$ (MZSC for brevity) if the following properties hold true:

D2a) *Maximal-weight:* all the resources from N_σ are *busy*: $OV_{\mathbf{q}}(N_\sigma)=C(N_\sigma)$; i.e. the number of jobs holding resources from N_σ is the largest one can achieve and equals the whole capacity of N_σ;

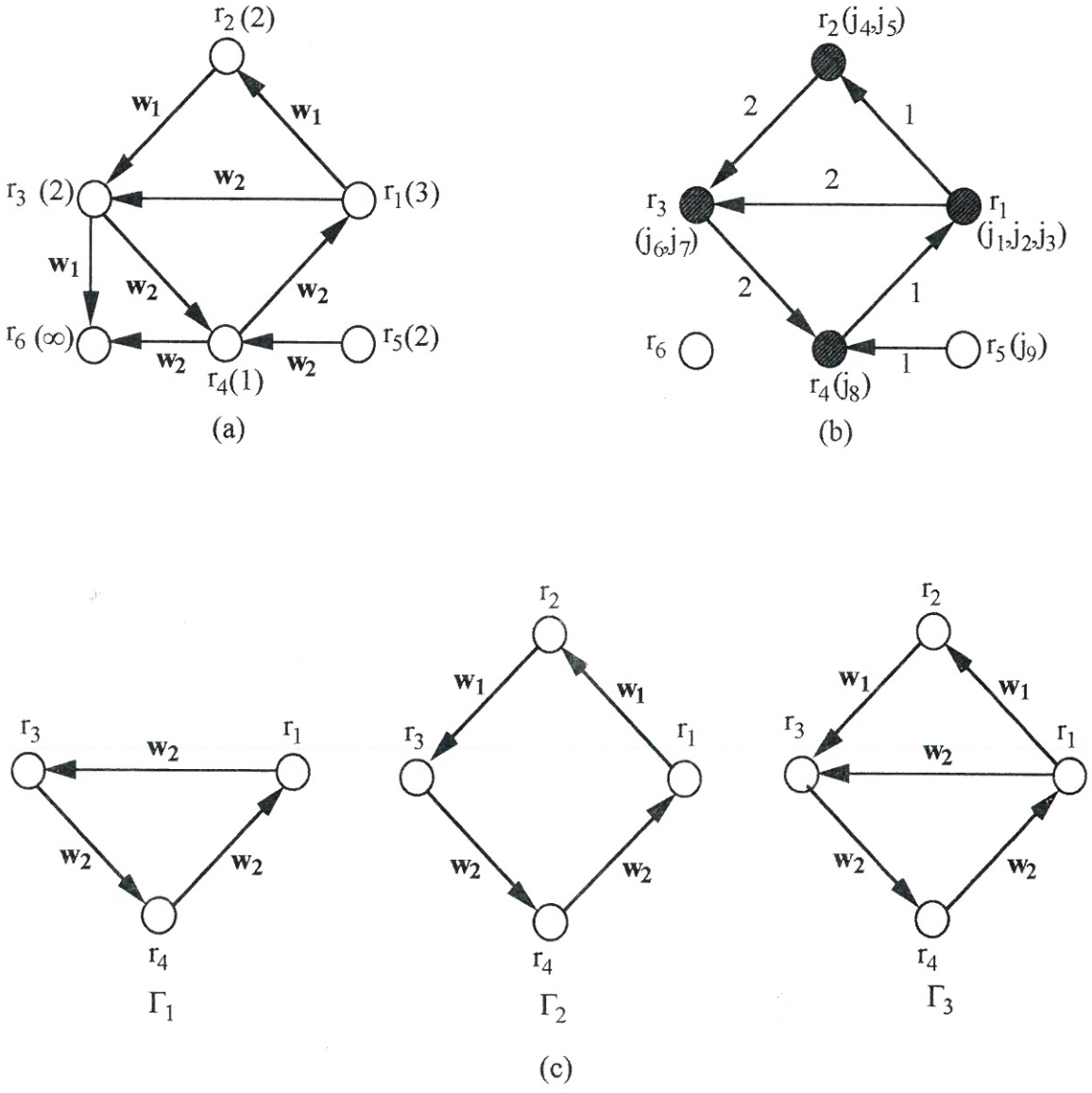

Figure 2. Example 2: (a) Digraph D_W; (b) Digraph $D_{Tr}(q)$; (c) Strong subdigraphs of D_W

D2b) *Zero-outdegree:* all the edges of $D_{Tr}(\mathbf{q})$ outgoing from vertices of N_σ belong to E_σ, i.e. the elements from N_σ are the only vertices in $D_{Tr}(\mathbf{q})$ reachable from vertices in N_σ.

The following theorem proved in [6] relates the deadlock to the presence of an MZSC in $D_{Tr}(\mathbf{q})$.

Theorem 1. $\mathbf{q}$ is a deadlock state for S iff there exists at least one MZSC in $D_{Tr}(\mathbf{q})$.

Since in the following we use digraphs to exhibit deadlock conditions, we indifferently say that "$\mathbf{q}$ is a deadlock state" or that "$D_{Tr}(\mathbf{q})$ is in deadlock condition". Moreover let $\Gamma=(N_\Gamma,E_\Gamma)$ be a strong subdigraph of D_W. We say that Γ is in deadlock condition in state $\mathbf{q}$ if it is an MZSC of $D_{Tr}(\mathbf{q})$.

Remark 1: The proof of Theorem 1 [6] shows that, if σ is an MZSC of $D_{Tr}(\mathbf{q})$, then the complete set J_D of jobs holding resources from N_σ and the set of resources $R_D=N_\sigma$ enjoy the properties of Definition 1.

The following example illustrates Theorem 1 and previous notations.

Example 1: Let us consider a six-resource system S (R=6) where non-fictitious resources have capacities $C(r_1)=3$, $C(r_2)=C(r_3)=C(r_5)=2$ and $C(r_4)=1$. The system produces a job mix J according to the working procedures $\mathbf{w_1}=(r_1, r_2, r_3, r_6)$ and $\mathbf{w_2}=(r_5, r_4, r_1, r_3, r_4, r_6)$. Figure1(a) shows the corresponding Working Procedure Digraph that contains three strong subdigraphs $\Gamma_1=(\{r_1,r_3,r_4\}, \{e_{13},e_{34},e_{41}\})$, $\Gamma_2=(\{r_1,r_2,r_3,r_4\}, \{e_{12},e_{23},e_{34},e_{41}\})$ and $\Gamma_3=(\{r_1,r_2,r_3,r_4\}, \{e_{12},e_{23},e_{34},e_{41},e_{13}\})$, exhibited by Fig. 1(c).

Let S be in a state $\mathbf{q}$, with: $J_\mathbf{q}=\{j_i: i=1, 2,\ldots, 9\}$, $HR(j_1)=HR(j_2)=HR(j_3)=r_1$ $SR(j_1)=r_2$, $SR(j_2)=SR(j_3)=r_3$, $HR(j_4)=HR(j_5)=r_2$, $SR(j_4)=SR(j_5)=r_3$, $HR(j_6)=HR(j_7)=r_3$, $SR(j_6)=SR(j_7)=r_4$, $HR(j_8)=r_4$, $SR(j_8)=r_1$, $HR(j_9)=r_5$, $SR(j_9)=r_4$. The corresponding Transition Digraph is shown by Figure 1(b). In particular, the strong subdigraph Γ_3 is an MZSC in $D_{Tr}(\mathbf{q})$. Namely, putting $N_{\Gamma 3}=\{r_1,r_2,r_3,r_4\}$ and $E_{\Gamma 3}=\{e_{12},e_{23},e_{34},e_{41},e_{13}\}$ we observe that $OV_\mathbf{q}(N_{\Gamma 3})=C(N_{\Gamma 3})=8$ and that each edge incident out vertices in $N_{\Gamma 3}$ is in $E_{\Gamma 3}$. Thus according to Theorem 1, $\mathbf{q}$ is a deadlock state. Indeed, it is easy to verify that jobs j_i for i=1, 2,..., 8 are permanently blocked, in circular wait condition. On the contrary, Γ_1 and Γ_2 are not MZSC, since they are not strong components in $D_{Tr}(\mathbf{q})$ (i.e. they are not maximal).

Now let Γ be any strong subdigraph of D_W. Then we define

$$C_0 = \min_{\Gamma \subset D_W} C(N_\Gamma) \tag{5}$$

where we put $C_0 = \infty$ if D_W is acyclic. Obviously, the minimum can be computed over the cycles of D_W only. Furthermore, for any $\Gamma \subset D_W$ and $\mathbf{q}\in Q$, let us introduce the following job subset:

$$J_{\mathbf{q},\Gamma} = \{j\in J_\mathbf{q} : \text{the set } E_\Gamma \cap E_\mathbf{w} \text{ is not empty, with } \mathbf{w}=WP(j)\} \tag{6}$$

In other words, $J_{\mathbf{q},\Gamma}$ collects all the jobs in process according to the working procedures involving edges of Γ.

By Theorem 1, if $\mathbf{q}$ is a deadlock state, $D_{Tr}(\mathbf{q})$ contains an MZSC and, then, a strong subdigraph (say Γ) with *busy* vertices. This determines that subset $J_{\mathbf{q},\Gamma}$ contains at least $C(N_\Gamma)$ jobs. So the following corollary holds.

Corollary 1: The necessary condition for $\mathbf{q}$ to be a deadlock state is that D_W contains at least one strong subdigraph Γ such that $J_{\mathbf{q},\Gamma}$ satisfies the following condition

$$Card(J_{\mathbf{q},\Gamma}) \geq C(N_\Gamma) \tag{7}$$

Remark 2: Because of Corollary 1, we can immediately state that

$$Card(J_\mathbf{q}) \geq C_0 \tag{8}$$

is a necessary condition for $\mathbf{q}$ to be a deadlock state.

Before closing this Section, we observe that in

systems with all unit-capacity resources the cycles are the only MZSCs possibly occurring in $D_{Tr}(\mathbf{q})$. So Theorem 1 reduces to the Theorem 1 of [8]. Analogously, the result stated in Remark 2 is equivalent to Corollary 2 of [8].

4. Second Level Deadlock

To define deadlock avoidance policies, we have to focus on a situation that is not an actual deadlock but inevitably gives rise to a deadlock in the immediate future. This situation, called Second Level Deadlock (SLD), is defined as follows:

Definition 3: Let $\mathbf{q}$ be not a deadlock state for S. We say $\mathbf{q}$ is an SLD state for S, if there exist two non empty subsets $J_S \subset J_{\mathbf{q}}$ and $R_S \subset R$ satisfying the following properties:

D3a) $HR(J_S) \subset R_S$, $SR(J_S) \subset R_S$;

D3b) If $j \in (J_{\mathbf{q}} - J_S)$ then $HR(j) \notin R_S$, i.e. set J_S collects all the jobs holding resources from R_S;

D3c) for each job $j \in J_S \cap J_{\mathbf{q},u}$, the transition releasing HR(j) to hold SR(j) leads to a deadlock state for which $J_D \subset J_S$.

In the sequel, we simply denote the set of *feasible* transitions in $D_{Tr}(\mathbf{q})$ defined by D3c), by $E_S = \{e_{im} \in E_F(\mathbf{q})$: $r_i = HR(j)$, $r_m = SR(j)$ for some $j \in J_S \cap J_{\mathbf{q},u}\}$.

Definition 3 identifies a critical situation in which each job of J_S is either blocked, because it requests access to a *busy* resource held by other parts from J_S, or determines a deadlock on acquiring the next resource in its residual working procedure. The following example clarifies the idea of SLD.

Example 2: Consider again a six-resource system as in Example 1, processing jobs according to the following working procedures: $\mathbf{w_1}=(r_5, r_4, r_3, r_6)$ and $\mathbf{w_2}=(r_3, r_2, r_4, r_1, r_5, r_4, r_6)$. Figure 2(a) shows the corresponding Working Procedure Digraph that contains three strong subdigraphs: $\Gamma_1=(\{r_1,r_5,r_4\}, \{e_{15},e_{54},e_{41}\})$, $\Gamma_2=(\{r_3,r_2,r_4\}, \{e_{32},e_{24},e_{43}\})$, $\Gamma_3=(\{r_1,r_5,r_4,r_3,r_2\}, \{e_{15},e_{54},e_{41},e_{43}, e_{32},e_{24}\})$ exhibited by Figure 2(c).

Moreover let the system be in a state $\mathbf{q}$, with: $J_{\mathbf{q}}=\{j_i$: i=1, 2,..., 9$\}$, $WP(j_i)=\mathbf{w_2}$ for i=1,...,7, $WP(j_8)=WP(j_9)=\mathbf{w_1}$, $HR(j_1)=HR(j_2)=HR(j_3)=r_1$ $SR(j_1)=SR(j_2)=SR(j_3)=r_5$, $HR(j_4)=HR(j_5)=r_3$, $SR(j_4)=SR(j_5)=r_2$, $HR(j_6)=HR(j_7)=r_2$, $SR(j_6)=SR(j_7)=r_4$, $HR(j_8)=HR(j_9)=r_5$, $SR(j_8)=SR(j_9)=r_4$. To make the ideas more clear, Figure 2(b) shows the Transition Digraph $D_{Tr}(\mathbf{q})$ in full lines while dashed lines represent the second transitions in the residual working procedures of jobs from $J_{\mathbf{q}}$. Though they are unblocked, progressing to their next destination, each of the jobs j_6 and j_7 (j_8 and j_9) leads to a deadlock condition involving the strong subdigraph Γ_1 (Γ_2). Note that Γ_1 (Γ_2) is an MZSC of $D_{Tr}(\mathbf{q},j_6)$ and $D_{Tr}(\mathbf{q},j_7)$ ($D_{Tr}(\mathbf{q},j_8)$ and $D_{Tr}(\mathbf{q},j_9)$).

In Definition 3, $\mathbf{q}$ represents the state just "one step before" the deadlock occurs. Namely, for any $j \in J_S \cap J_{\mathbf{q},u}$ the transition releasing $r_i=HR(j)$ to acquire $r_m=SR(j)$ leads to a digraph $D_{Tr}(\mathbf{q},j)$ that is in deadlock condition. By Theorem 1, $D_{Tr}(\mathbf{q},j)$ contains an MZSC, while $D_{Tr}(\mathbf{q})$ does not. Clearly, edges and vertices of this MZSC compose a corresponding strong subdigraph in D_W. At this point some questions arise. Consider the strong subdigraphs of D_W deadlocked by distinct jobs in $J_S \cap J_{\mathbf{q},u}$. How are they related to each other? How are the *busy/idle* resources distributed among such digraphs when the system is in a SLD state? The following theorem answers these questions and clarifies the entire subject.

Theorem 2. Let $\mathbf{q}$ be not a deadlock state for S. Necessary and sufficient condition for $\mathbf{q}$ to be an SLD state is that there exists a set $H=\{\Gamma_1,...,\Gamma_\mu,...,\Gamma_H\}$ of strong subdigraphs of D_W such that, putting

$$N_H = \bigcup_{\mu=1}^{H} N_{\Gamma_\mu} \tag{9a}$$

$$E_H = \bigcup_{\mu=1}^{H} E_{\Gamma_\mu} \tag{9b}$$

the following conditions hold in state $\mathbf{q}$:

Th2a) there exists only one *empty* resource (say r_m) in N_H and for this resource it holds $C(r_m)=1$ and $r_m \in N_{\Gamma\mu}$ for each $\mu \in \{1,2,\ldots,H\}$. All the remaining resources from N_H are *busy*;

Th2b) for each $\Gamma \in H$, there exists a job $j \in J_{q,u}$ such that Γ is in deadlock condition in $D_{Tr}(q,j)$;

Th2c) for each job $j \in J_{q,u}$ such that the *feasible* transition e_{im} (with r_i=HR(j) and r_m=SR(j)) is in set E_H, there exists a strong subdigraph $\Gamma \in H$ in deadlock condition in $D_{Tr}(q,j)$.

Proof: see Appendix 1

Though it fully characterizes the SLD, Theorem 2 is not fit for stating deadlock avoidance policies because of its complexity. However this result is necessary to establish the following Theorem 3, that provides an efficient starting point for developing appropriate policies. In particular, Theorem 3 gives a necessary condition for an SLD occurrence, based on the detection of all the potential SLDs. This result uses the Working Procedure Digraph only and is related to some cycles of D_W . One must not be surprised that some cycles of D_W, exhibiting particular characteristics, play a particular role in the detection of all the potential SLD conditions. Namely, the strong subdigraphs, that are the foundation of Theorem 2, may be viewed as the union of cycles. On the other hand, a cycle represents the simplest figure of any (non trivial) strong subdigraph. In Appendix 2, the logical path leading to the proof of Theorem 3 clearly focuses on the role of cycles of D_W.

Now some preliminary definitions are to introduce Theorem 3. As in [8], we collapse each cycle of D_W into a vertex of a new digraph called Second Level Digraph, defined as follows.

Definition 4: Let $\{\gamma_1, \gamma_2, \ldots, \gamma_K\}$ be the complete set of all the cycles of D_W. We define the Second Level Digraph $D^2{}_W=(N^2,E^2{}_W)$, by associating a vertex $n^2{}_s$ with each cycle γ_s of D_W, so that $N^2=\{n^2{}_1,n^2{}_2,\ldots,n^2{}_K\}$. Moreover, for $s,t \in \{1,2,\ldots,K\}$, the edge $e^2{}_{st}=(n^2{}_s,n^2{}_t)$ belongs to $E^2{}_W$, iff:

D4a) γ_s and γ_t have only one vertex in common (say r_m) with $C(r_m)=1$;

D4b) there exists a working procedure $w \in W$, containing vertices r_i, r_m and r_p in a strict order of succession, with $e_{im} \in E_{\gamma_s}$ and $e_{mp} \in E_{\gamma_t}$.

The Second Level Digraph may contain cycles. These are closely related to the Second Level Deadlock and hence they are relevant to state deadlock avoidance policies. The following definition completes the tools necessary for this aim.

Definition 5: Let $\gamma^2=(N^2{}_\gamma,E^2{}_\gamma)$ be a cycle of $D^2{}_W$ (Second Level Cycle) and let $\{\gamma_1, \gamma_2, \ldots, \gamma_P\}$ be the set of cycles in D_W corresponding to the vertices in $N^2{}_\gamma$. The Cycle Capacity, $C(N^2{}_\gamma)$, of the second level cycle γ^2 equals the maximum number of jobs that the cycles corresponding to the vertices in $N^2{}_\gamma$ can hold:

$$C(N^2_\gamma) = C\left(\bigcup_{s=1}^{P} N_{\gamma_s}\right) \quad (10)$$

Finally, we define a particular subset of second level cycles as follows:

$\Gamma^2 = \{\gamma^2$ of $D^2{}_W$: there exists a vertex $r_m \in N$

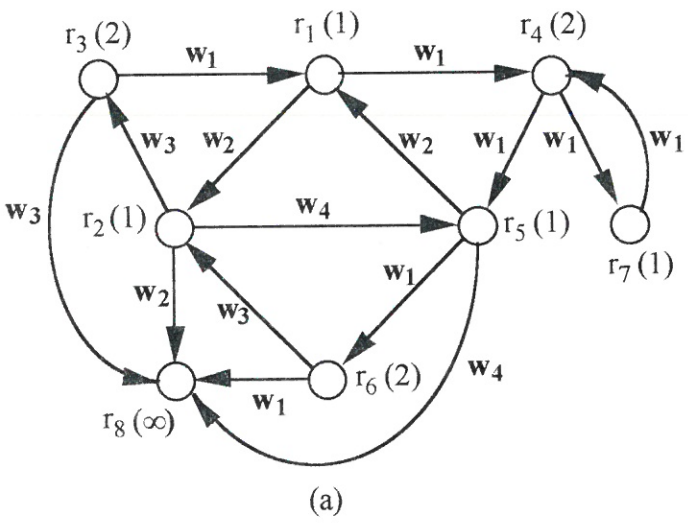

Figure 3. Example 3: (a) Digraph D_W

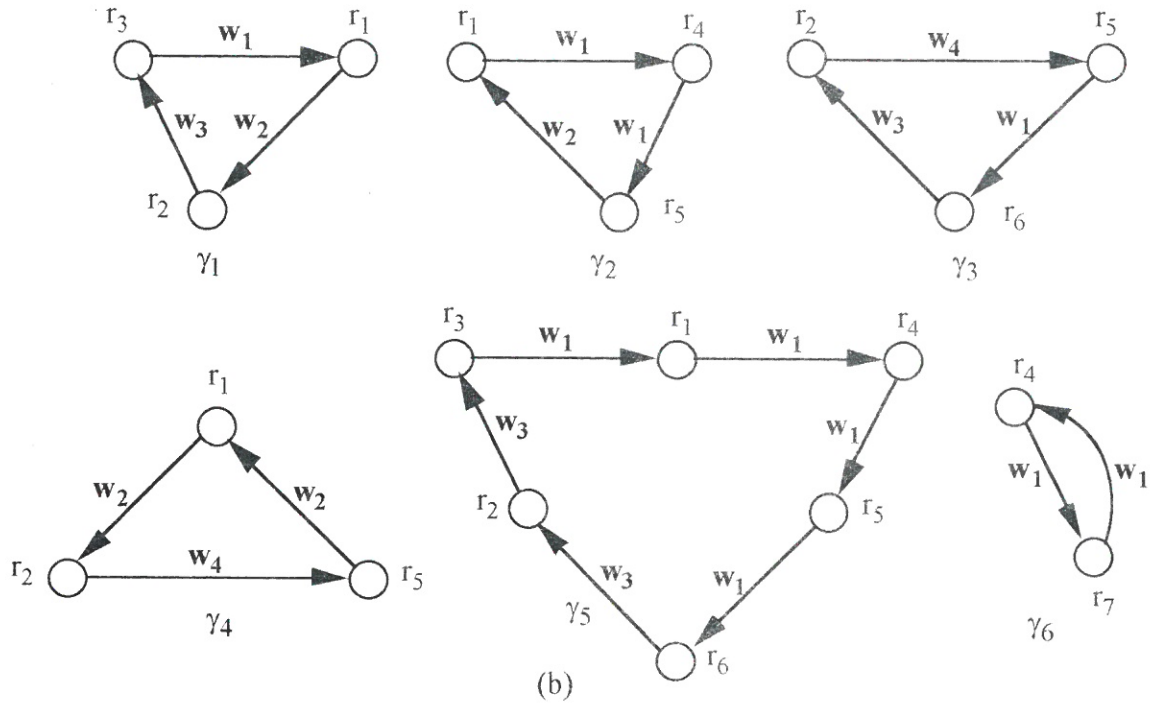

Figure 3. Example 3: (b) Cycles of D_W

such that $C(r_m)=1$ and $N_{\gamma_s} \cap N_{\gamma_t} = \{r_m\}$

for any $s,t \in \{1,2,\ldots,P\}$ with $s \neq t$} (11)

In other words a second level cycle is in Γ^2 iff all the (first level) cycles corresponding to its vertices share one and only one vertex of D_W. Such a vertex that we name *center* of γ^2, must have unit capacity. The following example illustrates Definitions 4 and 5.

Example 3: Consider an eight-resource system (R=8) with capacities: $C(r_1)=C(r_2)=C(r_5)=C(r_7)=1$, $C(r_3)=C(r_4)=C(r_6)=2$, $C(r_8)=\infty$ and four working procedures: $\mathbf{w_1}=(r_3, r_1, r_4, r_7, r_4, r_5, r_6, r_8)$, $\mathbf{w_2}=(r_5, r_1, r_2, r_8)$, $\mathbf{w_3}=(r_6, r_2, r_3, r_8)$, $\mathbf{w_4}=(r_2, r_5, r_8)$. Figure 3(a) shows the corresponding digraph D_W which contains six cycles: $\gamma_1=(\{r_1,r_2,r_3\}, \{e_{12},e_{23},e_{31}\})$, $\gamma_2=(\{r_1,r_4,r_5\}, \{e_{14},e_{45},e_{51}\})$, $\gamma_3=(\{r_2,r_5,r_6\}, \{e_{25},e_{56},e_{62}\})$, $\gamma_4=(\{r_1,r_2,r_5\}, \{e_{12},e_{25},e_{51}\})$, $\gamma_5=(\{r_1,r_4,r_5,r_6,r_2,r_3\}, \{e_{14},e_{45},e_{56},e_{62},e_{23},e_{31}\})$ and $\gamma_6=(\{r_4,r_7\}, \{e_{47},e_{74}\})$ shown by Figure 3(b). Accordingly, the Second Level Digraph has six vertices: $N^2=\{n^2_1,n^2_2,n^2_3,n^2_4,n^2_5,n^2_6\}$. As to the edge set E^2_W, we first observe that γ_1, γ_2, γ_3, and γ_5 share more than one node with γ_4 and that γ_6 and γ_4 have no vertex in common. Cycles γ_5 and γ_6 (as well as γ_2 and γ_6) share vertex r_4 only: however r_4 is not a unit-capacity resource. Moreover none of the cycles γ_s for s=1,2,3,4 (for s=1,3,4) has just one vertex in common with γ_5 (with γ_6). Thus n^2_4, n^2_5 and n^2_6 are isolated vertices in D^2_W. On the other hand γ_1 and γ_2 have only the vertex r_1 in common, with $C(r_1)=1$. Moreover the working

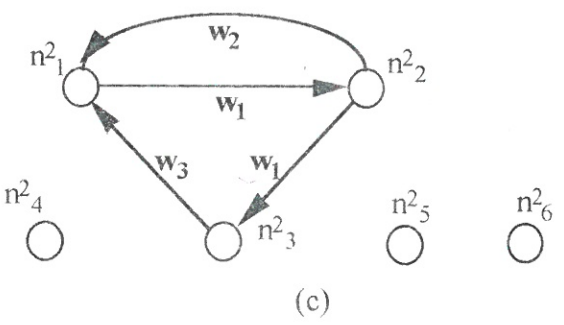

Figure 3. Example 3: (c) Digraph D^2_W

procedure $\mathbf{w_1}$ contains vertices r_3, r_1 and r_2 in a strict order of succession, with $e_{31} \in E_{\gamma 1}$ and $e_{12} \in E_{\gamma 2}$, and $\mathbf{w_2}$ contains the subsequence r_5, r_1, r_2, with $e_{51} \in E_{\gamma 2}$ and $e_{12} \in E_{\gamma 1}$. Thus both e^2_{12} and e^2_{21} are in E^2_W. Analogous considerations lead to $E^2_W = \{e^2_{12}, e^2_{21}, e^2_{23}, e^2_{31}\}$ and to the Second Level Digraph shown in Figure 3(c). This contains two Second Level Cycles $\gamma^2_1 = (\{n^2_1, n^2_2\}, \{e^2_{12}, e^2_{21}\})$ and $\gamma^2_2 = (\{n^2_1, n^2_2, n^2_3\}, \{e^2_{21}, e^2_{23}, e^2_{31}\})$. However only γ^2_1 is in Γ^2. Indeed, while γ_1 and γ_2 share only one vertex ($N_{\gamma 1} \cap N_{\gamma 2} = \{r_1\}$), for cycles γ_1, γ_2 and γ_3 we get: $N_{\gamma 1} \cap N_{\gamma 2} = \{r_1\} \neq N_{\gamma 1} \cap N_{\gamma 3} = \{r_2\} \neq N_{\gamma 2} \cap N_{\gamma 3} = \{r_5\}$•

The following result establishes that only Second Level Cycles in Γ^2 may be responsible for Second Level Deadlocks.

Theorem 3. If $\mathbf{q}$ is an SLD state for S, then there exists a second level cycle $\gamma^2 \in \Gamma^2$ such that

$$\text{Card}(J_{\mathbf{q}}) \geq [C(N^2_{\gamma}) - 1] \qquad (12)$$

Proof: see Appendix 2

5. Deadlock Avoidance Techniques

In this Section we propose some real-time policies (Restriction Policies) that, on the basis of the current state, rule the resource allocation to avoid deadlock. More precisely, the control action consists in inhibiting some selected events, i.e. in keeping them from occurring. For example, an 1-type event (j,**w**) is *inhibited* if the Restriction Policy prevents $j \in J$ from entering S to be processed according to the working procedure **w**. Analogously, the 2-type event $j \in J_{\mathbf{q}}$ is inhibited if the control does not allow j to release HR(j) for taking SR(j). Noninhibited events are said to be *enabled.*

To formally state the Restriction Policies we define the sets:

$X_1 = \{(\mathbf{q},\mathbf{w}) \in Q \times W$: the first resource of **w** is *idle* or *empty* in state $\mathbf{q}\}$

$X_2 = \{(\mathbf{q},j) \in Q \times J : j \in J_{\mathbf{q},u}\}$

So the Control Rule for 1-type events and 2-type events are respectively defined as:

$$f_1 : X_1 \rightarrow \{0,1\} \qquad (13)$$

$$f_2 : X_2 \rightarrow \{0,1\} \qquad (14)$$

With such notations, $f_1(\mathbf{q},\mathbf{w})=0$ ($f_1(\mathbf{q},\mathbf{w})=1$) means that, for S in state $\mathbf{q}$, any 1-type event involving the working procedure **w** is inhibited (enabled). Analogously, $f_2(\mathbf{q},j)=0$ ($f_2(\mathbf{q},j)=1$) indicates that, for S in state $\mathbf{q}$, the 2-type event representing the transition of j from HR(j) to SR(j) is inhibited (enabled). We name Restriction Policy a pair (f_1,f_2).

As already remarked in other papers [1,8], an unsuited choice of the Restriction Policy could lead to a situation known as Restricted Deadlock (RD) that is similar to a deadlock. In an RD the jobs in a set $J_R \subset J_{\mathbf{q}}$ remain in circular wait indefinitely, because some of them are blocked by other jobs from J_R, while the transitions of the remaining jobs are inhibited by the Restriction Policy. Moreover such transitions keep on remaining inhibited, if no job in J_R releases the resource it currently holds. The next definition formalizes the condition for an RD occurrence.

Definition 6. Let $\mathbf{q} \in Q$ be not a deadlock state for S and let (f_1,f_2) be a Restriction Policy. Then $\mathbf{q}$ is an RD state for S under (f_1,f_2), if there exists a subset $J_R \subset J_{\mathbf{q}}$, such that:

D6a) all the jobs holding resources from $SR(J_R \cap J_{\mathbf{q},b})$ are in J_R;

D6b) set $J_R \cap J_{\mathbf{q},u}$ is not empty and for each job $j \in J_R \cap J_{\mathbf{q},u}$, f_2 inhibits the corresponding *feasible* transition from HR(j) to SR(j). Moreover, such transition keeps on remaining inhibited if no job in J_R releases the resource that it currently holds.

Although Definition 6 is slightly different from the analogous Definition 7 given in [8], it characterizes just the same situation. On the other hand, the formulation given here is more general and adequate for systems with multiple capacity resources.

Remark 3: As already underlined in the

companion paper [8], only the Control Rule f_2 may be responsible for an RD. Restricted deadlock, indeed, is independent of f_1. Of course, if $f_2(\mathbf{q},e_{im})=1$ for every $(\mathbf{q},e_{im})\in X_2$ (i.e. f_2 does not inhibit any transition), there exists no RD state for S.

A Restriction Policy is effective only if it avoids both deadlock and RD, according to the following definition.

Definition 7. Let (f_1,f_2) be a Restriction Policy. Moreover, assume that, for every $\mathbf{q_0}\in Q_0$ with $Q_0\subset Q$, any $\mathbf{q}\in Q$ reachable from $\mathbf{q_0}$ under (f_1,f_2) is neither a deadlock nor an RD state. In this case, we say that S, starting at Q_0 under the Restriction Policy (f_1,f_2), is *deadlock-free*.

Here we propose a first Restriction Policy that is very simple and evident, since it establishes an upper bound for the number of jobs in progress.

Restriction Policy 1 (RP1):

$f_1(\mathbf{q},\mathbf{w})=1$ if $\mathrm{Card}(J_{\mathbf{q}})<(C_0-1)$
$f_1(\mathbf{q},\mathbf{w})=0$ otherwise

$f_2(\mathbf{q},j)=1$ for any $(\mathbf{q},j)\in X_2$.

The following proposition shows that RP1 is suitable for deadlock avoidance.

Proposition 1: Let $Q_0=\{\mathbf{q_0}\in Q : \mathrm{Card}(J_{\mathbf{q_0}})<C_0\}$. Then system S, starting at Q_0 under RP1, is deadlock-free.

Proof

By the definition of Q_0 and of Control Rule f_1, we get $\mathrm{Card}(J_{\mathbf{q}})<C_0$ in any state $\mathbf{q}$ reachable from $\mathbf{q_0}$. Therefore, according to Remark 2, S may not end in deadlock. Finally, by Remark 3, $\mathbf{q}$ may not be an RD state for S under RP1. •

To introduce the second Restriction Policy we need further notations. In particular, for any $\mathbf{w}\in W$, we define the following set of strong subdigraphs of D_W:

$$H_{\mathbf{w}}=\{\Gamma \text{ of } D_W : \text{the set } E_\Gamma\cap E_{\mathbf{w}} \text{ is not empty}\} \quad (15)$$

So set $H_{\mathbf{w}}$ contains the strong subdigraphs of D_W having at least one edge labelled by $\mathbf{w}$. With the above notations and the subset defined by (6), we introduce the second Restriction Policy.

Restriction Policy 2 (RP2):

$f_1(\mathbf{q},\mathbf{w})=1$ if
$\mathrm{Card}(J_{\mathbf{q},\Gamma})<[C(N_\Gamma)-1]$ for any $\Gamma\in H_{\mathbf{w}}$;
$f_1(\mathbf{q},\mathbf{w})=0$ otherwise

$f_2(\mathbf{q},j)=1$ for any $(\mathbf{q},j)\in X_2$.

The constraints introduced by the Restriction Policy 2 on the work in progress are less strict than the bounds imposed by RP1. Moreover, according to the following proposition, RP2 avoids deadlock and RD.

Proposition 2: Let $Q_0=\{\mathbf{q_0}\in Q : \mathrm{Card}(J_{\mathbf{q_0},\Gamma})<C(N_\Gamma)$ for each Γ of $D_W\}$. Then system S, starting at Q_0 under RP2 is deadlock-free.

Proof

Owing to the condition on $\mathbf{q_0}$ and to the application of RP2, we have $\mathrm{Card}(J_{\mathbf{q},\Gamma})<C(N_\Gamma)$ for each strong subdigraph Γ of D_W and for any state $\mathbf{q}$ reachable from $\mathbf{q_0}$. Hence by Corollary 1, S may not end in deadlock. In addition, by Remark 3, $\mathbf{q}$ may not be an RD state for S under RP2.

Next we consider the following Restriction Policy.

Restriction Policy 3 (RP3):

$f_1(\mathbf{q},\mathbf{w})=1$ if $D_{Tr}(\mathbf{q},\mathbf{w})$ does not contain any MZSC
$f_1(\mathbf{q},\mathbf{w})=0$ otherwise

$f_2(\mathbf{q},j)=1$ if $D_{Tr}(\mathbf{q},j)$ does not contain any MZSC
$f_2(\mathbf{q},j)=0$ otherwise

RP3 controls both 1-type and 2-type events to avoid deadlocks. However, such a policy might not avoid RD. The next proposition shows that RD and SLD are strictly related if system S is under RP3.

Proposition 3: Let f_2 be defined according to RP3. For any f_1, $\mathbf{q}\in Q$ is an RD state for S under (f_1,f_2) iff it is an SLD state.

Proof

Only if part

Let $\mathbf{q}$ be an RD state for S under (f_1,f_2) and $J_R\subset J_{\mathbf{q}}$ be the job set satisfying D6a) and D6b). In particular, D6b) states that for each job $j\in J_R\cap J_{\mathbf{q},u}$, it holds $f_2(\mathbf{q},j)=0$, i.e. $D_{Tr}(\mathbf{q},j)$ contains an MZSC, say $\sigma=(N_\sigma, E_\sigma)$. Let J_D indicate the job set involved in the corresponding deadlock.

We first show that $J_D\subset J_R$. Namely, with the system in state $\mathbf{q}$, if any job $j'\in J_D-\{j\}$ released the resource HR(j'), the event specified by j would not cause a deadlock and would be enabled by f_2. Since, by D6b), the transition of j from HR(j) to SR(j) keeps on remaining inhibited if no job in J_R releases the resource it currently holds, we get $j'\in J_R$, proving $J_D\subset J_R$.

Now note that $J_D-\{j\}$ represents the set of jobs holding resources of N_σ in state $\mathbf{q}$ and that the definition of MZSC leads to $SR(J_D-\{j\})\subset N_\sigma$. So, applying the previous arguments to any job in $J_R\cap J_{\mathbf{q},u}$ leads to K (not necessarily distinct) MZSCs: $\sigma_1,\sigma_2,\ldots,\sigma_K$, where $K=card(J_R\cap J_{\mathbf{q},u})$. So considering the vertex set of these MZSCs:

$$R_S=\bigcup_{i=1}^{K} N_{\sigma_i}$$

define J_S as the set of all jobs holding resources from R_S in state $\mathbf{q}$. In this way D3a) and D3b) hold. Moreover, according to the previous considerations, $J_S\subset J_R$ so that for each job $j\in J_S\cap J_{\mathbf{q},u}$ it holds $f_2(\mathbf{q},j)=0$. This means that the transition of j from HR(j) to SR(j) leads to a deadlock, with $J_D\subset J_S$. Thus J_S enjoys D3c) too, proving the implication.

If part

If $\mathbf{q}$ is an SLD state, putting $J_R=J_S$, it is easy to prove that under RP3 conditions D6a) and D6b) hold.

Note that if an RD takes place under RP3, such a condition forebodes a classic deadlock that would inevitably begin, in the absence of any Restriction Policy. Example 4 makes this point clearer.

Example 4: Let the system introduced by Example 2 be in the state described by Figure 2(b). Even if j_6, j_7, j_8 and j_9 are unblocked, RP3 inhibits such jobs indefinitely. Namely events j_6 and j_7 (j_8 and j_9) lead to a deadlock condition involving the strong subdigraph Γ_1 (Γ_2), as defined in Example 2. Thus $\mathbf{q}$ is a Restricted Deadlock state under RP3.

Clearly, the prevention of an SLD occurrence guarantees that RP3 does not lead to any RD. Hence using the control rule f_1 to falsify one or more of the necessary conditions for an SLD occurrence is an effective means to avoid RD. On this basis, putting

$$C^2{}_0=\min_{\Gamma^2} C(N^2{}_\gamma) \qquad (16)$$

(with $C^2{}_0=\infty$ if Γ^2 is empty) we can define a further Restriction Policy.

Restriction Policy 4:(RP4):

$f_1(\mathbf{q},\mathbf{w})=1$ if $D_{Tr}(\mathbf{q},\mathbf{w})$ does not contain any MZSC and $Card(J_{\mathbf{q}})<(C^2{}_0-2)$;

$f_1(\mathbf{q},\mathbf{w})=0$ otherwise;

$f_2(\mathbf{q},j)=1$ if $D_{Tr}(\mathbf{q},j)$ does not contain any MZSC;

$f_2(\mathbf{q},j)=0$ otherwise

The following proposition proves that RP4 is a deadlock avoidance policy.

Proposition 4: Let $Q_0=\{\mathbf{q_0}\in Q : Card(J_{\mathbf{q_0}})<(C^2{}_0-1)$ and $\mathbf{q_0}$ is not a deadlock state$\}$. Then system S, starting from Q_0 under RP4, is deadlock-free.

Proof

Let $\mathbf{q}\in Q$ be any state reachable from $\mathbf{q_0}\in Q_0$

under RP4. Such a policy prevents **q** from being a deadlock state. In addition, Control Rule f_1 leads to $\text{Card}(J_{\mathbf{q}})<(C^2{}_0-1)$, so that $\text{Card}(J_{\mathbf{q}})<[C(\gamma^2)-1]$ for any $\gamma^2\in\Gamma^2$. Hence, by Theorem 3 and Proposition 3, **q** is not an RD state for S under RP4. This completes the proof. •

Before closing this Section we note that if Γ^2 is empty, the Restriction Policies 3 and 4 coincide.

6. Computational Complexity

In order to discuss the computational complexity of the proposed Restriction Policies, we distinguish the off-line costs from the on-line ones. Off-line costs concern algorithms that are employed only once, before the proper real-time control. On the contrary, on-line costs characterize real-time computations.

To use RP1 one should determine off-line the minimum capacity C_0 on all the cycles from D_W. Such a value can be easily computed by searching the (non trivial) shortest walk starting from each vertex in D_W and ending in the same vertex. In this procedure for any edge $e_{im}\in E_W$, the weight to consider is just the capacity $C(r_i)$ of the first vertex r_i. Such an algorithm requires $O\{[\text{Card}(N)]^3\}$ operations [13]. Finally, the on-line costs of RP1 are due to the real-time updating of $\text{Card}(J_{\mathbf{q}})$ only.

To apply RP2 we must determine off-line all the strong subdigraphs from D_W with their capacities, and build up a relationship between working procedures and strong digraphs, for identifying sets $H_{\mathbf{w}}$ (see (6)). Now any strong subdigraph is the union of cycles sharing some vertices and, in case, some edges. Thus the first step lies in generating all the cycles of D_W. This requires $O\{[\text{Card}(N)+\text{Card}(E_W)](c_1+1)\}$ operations, where c_1 is the number of cycles in D_W [13]. As a second step, one must determine the strong subdigraphs consisting of two cycles, by finding all the pairs of cycles sharing at least one vertex. The same way, by comparing cycles and strong subdigraphs consisting of two cycles, one can determine the three-cycle strong subdigraphs, and so on. Clearly the number of comparisons among cycles is exponential in c_1, i.e. $(2^{c1}-1)$. Thus finding all the strong subdigraphs requires $O[\text{Card}(N)\,(2^{c1}-1)]$ operations. A similar complexity characterizes the determination of sets $H_{\mathbf{w}}$, i.e. $O[\text{Card}(W)\ \text{Card}(E_W)\ (2^{c1}-1)]$. Finally, the on-line algorithm of RP2 needs the updating of $\text{Card}(J_{\mathbf{q},\Gamma})$ only, for each strong subdigraph of D_W.

RP3 demands no off-line computations while the on-line algorithm is in two steps. The first one transforms $D_{Tr}(\mathbf{q})$ into $D_{Tr}(\mathbf{q}')$, where $\mathbf{q}'$ indicates the state the system will reach from **q** as a consequence of the considered 1-type or 2-type event. The second step searches for MZSCs in $D_{Tr}(\mathbf{q}')$. To detect an MZSC the following procedure [6] can be used. Let j be the job entering the system or requiring a transition from one resource to another one. Moreover let r_m be the resource that j has to take next. Now $D_{Tr}(\mathbf{q}')$ contains an MZSC iff all the vertices reachable from r_m are *busy*. Hence the core of the detection algorithm is a depth-first search, starting from r_m and consisting of $O\{\text{Card}[E_{Tr}(\mathbf{q}')]\}$ operations.

Implementing RP4 requires off-line detection of the cycles from D_W so that to generate the digraph $D^2{}_W$. Moreover it is necessary to find the cycles from $D^2{}_W$ with their capacities, the subset Γ^2 and the minimum capacity $C^2{}_0$. Building $D^2{}_W$ requires $O[(c_1)^2 L]$ operations, where L indicates the sum of the lengths of all the working procedures (i.e. the sum of resources appearing in all the working procedures, counting repetitions). Furthermore, generating the cycles of $D^2{}_W$ and set Γ^2 needs $O\{[c_1+\text{Card}(E^2{}_W)](c_2+1)\}$ and $O[c_1\,c_2\,\text{Card}(N)]$ operations respectively, where c_2 indicates the number of Second Level Cycles. Finally, RP4 requires the same on-line computations as RP3 and, in addition, on-line updating of $\text{Card}(J_{\mathbf{q}})$.

The preceding consideration shows that the off-line complexity of RP1 is polynomial while RP2 needs exponential-time off-line algorithms. Thus RP2 will be suitable if there is a small number of cycles in D_W. Nevertheless, this is the case in many practical situations. The off-line computational cost of RP4 depends

on the number of cycles in D_W and $D^2{}_W$. On the other hand, as they have small on-line computational costs, all the policies are suitable for real-time control applications.

Finally, RP3 requires low costs and appears to be the least restrictive policy one can use. Indeed it inhibits only events just leading to a deadlock state at the next step. Nevertheless, it is deadlock-free if Γ^2 is empty. Production systems often satisfy this condition, particularly if they have multiple capacity resources. Namely, by definition (11), each second level cycle from Γ^2 must have a unit-capacity centre. So, only unit-capacity resources are relevant, as possible centres, to detect the second level cycles that might be in Γ^2. For example, if the capacity of each resource in the system is more than one, Γ^2 is certainly empty. Furthermore, if the number of edges incident out (or into) a unit-capacity vertex is one, such a vertex cannot be the centre of a second level cycle from Γ^2. Namely, any pair of cycles containing that vertex must share another resource too, in contradiction with (11). Obviously, if a unit-capacity vertex has zero outdegree (or indegree), no cycle of D_W contains it. So, also in this case, the vertex cannot be the centre of a second level cycle.

7. Concluding Remarks

In the previous Sections we propose some control policies for deadlock avoidance in flexible production systems. The methodological framework is the discrete-event representation of the manufacturing process. We use the Working Procedure Digraph for representing all the possible interactions between jobs and resources, and the Transition Digraph for considering only their current relations (occupations and requests). All the material developed in this work represents an extension to systems with multiple capacity resources of the analysis already performed in a companion paper (for devices with unit-capacity resources). Although this generalization involves more difficult logical paths, it leads to restriction policies with a complexity comparable to the implementation effort of the policies designed for systems with unit-capacity resources. In particular, RP1 and RP3 have small off-line costs. On the other hand, the off-line computations pertaining to RP2 and RP4 have exponential worst-case complexity, because the number of cycles in a digraph can be exponential in the number of vertices. However, in many practical cases such computations can be executed in a reasonably short time. Moreover, they are carried out only once, before the proper on-line control.

All the Restriction Policies require small on-line computational costs, so that they are suitable for real-time implementation.

Restriction Policy RP3 exhibits some remarkable peculiarities: it is very easy to apply and it is the least restrictive policy one can implement. Namely, RP3 inhibits only the events immediately leading to a deadlock. Its use however, is limited to systems where set Γ^2 is empty. Only in this case, indeed, such a policy is deadlock-free. The importance of RP3 is revealed by the fact that set Γ^2 is often just empty. Moreover, as remarked at the end of the previous Section, systems with multiple capacity resources are more likely to enjoy such a condition than devices with unit-capacity resources.

Appendix 1

Proving Theorem 2 requires some preparation that leads to three lemmas. Lemma 1 concerns the details of the mechanism leading a strong subdigraph of D_W to deadlock. In particular, it focuses on the distribution of *busy* and *idle* resources inside the strong subdigraph one step before the deadlock occurs and characterizes the *feasible* transition leading to the deadlock. Lemma 2 establishes that the strong subdigraph deadlocked by a given transition is unique. Finally, Lemma 3 uses the previous results to derive some important relationships concerning strong subdigraphs of D_W deadlocked by jobs in set $J_S \cap J_{\mathbf{q},u}$ defined by D3c).

Lemma 1: Let a strong subdigraph $\Gamma=(N_\Gamma,E_\Gamma)$ of D_W be not in deadlock condition in $D_{Tr}(\mathbf{q})$ and let $j \in J_{\mathbf{q},u}$, progressing from $r_i=HR(j)$ to $r_m=SR(j)$, cause Γ to end in deadlock in $D_{Tr}(\mathbf{q},j)$. Then, with reference to state $\mathbf{q}$, Γ satisfies the following properties:

L1a) all the resources from N_Γ are *busy* but one, that is *idle* or *empty*;

L1b) the *idle* or *empty* resource in N_Γ is just r_m with $r_m \neq r_R$; moreover $r_i \notin N_\Gamma$,

$OV_{\mathbf{q}}(r_m)=C(r_m)-1$ and $OV_{\mathbf{q}}(r_k)=C(r_k)$ for each $r_k \in N_\Gamma$ with $r_k \neq r_m$;

L1c) denoting by E_k the subset of E_Γ collecting edges incident out $r_k \in N_\Gamma$, it holds:

$$a_{\mathbf{q}}(E_k) = C(r_k) \quad \text{if } r_k \neq r_m \tag{21a}$$

and

$$a_{\mathbf{q}}(E_m) = C(r_m)-1 \tag{21b}$$

L1d) as to the digraph $D_{Tr}(\mathbf{q})$, r_m is reachable from each vertex of set N_Γ by a path containing edges of E_Γ only. Moreover, each vertex reachable from r_m is in N_Γ. Finally, edges from $E_\Gamma \cap E_{Tr}(\mathbf{q})$ represent *blocked* transitions, but edges incident into r_m describing *feasible* transitions.

Proof

L1a) is true.

At the outset we emphasize that, since it is not in deadlock condition in $D_{Tr}(\mathbf{q})$, Γ is not an MZSC of $D_{Tr}(\mathbf{q})$ (see Theorem 1). Moreover, since transition e_{im} is *feasible*, r_m is not *busy* in state $\mathbf{q}$. Having this in mind, we return to the proof of L1a).

By contradiction, assume L1a) false and consider two cases.

Case 1: All the resources from N_Γ are *busy* in state $\mathbf{q}$.

This implies $r_m \notin N_\Gamma$ so that edges of D_W incident out or into r_m are not in E_Γ. Hence, the occurrence of the transition of j from r_i to r_m does not modify edges from E_Γ (and their weights) in $D_{Tr}(\mathbf{q})$ (see operations 1) and 2) in Section II). So the subdigraph $\Gamma=(N_\Gamma,E_\Gamma)$ of D_W, that is an MZSC of $D_{Tr}(\mathbf{q},j)$, must also be an MZSC of $D_{Tr}(\mathbf{q})$ and, by Theorem 1, $D_{Tr}(\mathbf{q})$ must be in deadlock condition. In conclusion a contradiction is reached.

Case 2: At least two resources from N_Γ are *idle* or *empty* in state $\mathbf{q}$.

Since two idle or empty resources may not become *busy* as a result of a single transition, there exists no job $j \in J_{\mathbf{q},u}$ making Γ an MZSC of $D_{Tr}(\mathbf{q},j)$: a contradiction.

L1b) is true.

Since all the resources of an MZSC are *busy*, by L1a) the occurrence of the *feasible* transition e_{im} must increase the number of *busy* resources of N_Γ. Such a number can get higher only if e_{im} corresponds to a job leaving $r_i \notin N_\Gamma$ to occupy $r_m \in N_\Gamma$. So, it must hold $OV_{\mathbf{q}}(r_m)=C(r_m)-1$ and $OV_{\mathbf{q}}(r_k)=C(r_k)$ for each $r_k \in N_\Gamma$ with $r_k \neq r_m$. Finally, since the outdegree of r_R is always zero, no strong subdigraph in D_W may contain the fictitious resource. Hence we get: $r_m \neq r_R$.

L1c) is true.

Let $\mathbf{q}'$ be the deadlock state reached from $\mathbf{q}$ as a consequence of the transition of j from r_i to r_m. By assumption, Γ in an MZSC in $D_{Tr}(\mathbf{q},j)$. Hence, Γ has to meet the requirements D2a) and D2b) which yield $a_{\mathbf{q}'}(E_k)=C(r_k)$ for each $r_k \in N_\Gamma$.

To show (21) we have to consider the condition of the resources before the transition of j from r_i to r_m. By L1b) it holds $SR(j)=r_m \neq r_R$ so that it makes sense to define TR(j). So, putting $r_p=TR(j)$, condition D2b) of Definition 2 yields: $r_p \in N_\Gamma$. Moreover, according to the transformation mechanism leading from $D_{Tr}(\mathbf{q})$ to $D_{Tr}(\mathbf{q},j)$, the weights of edges from Γ do not change but for e_{mp} that becomes $a_{\mathbf{q}'}(e_{mp})=a_{\mathbf{q}}(e_{mp})+1$. Hence (21a) and (21b) follow.

L1d) is true.

Since it is an MZSC of $D_{Tr}(\mathbf{q},j)$, $\Gamma=(N_\Gamma, E_\Gamma)$ is a strongly connected digraph. In other words, two distinct nodes of N_Γ are mutually reachable. According to the transformation mechanism leading from $D_{Tr}(\mathbf{q})$ to $D_{Tr}(\mathbf{q},j)$, set $E_\Gamma \cap E_{Tr}(\mathbf{q},j)$ differs from $E_\Gamma \cap E_{Tr}(\mathbf{q})$ in the fact that it contains one more edge at the most which starts from r_m (say e_{mp} with $r_p=TR(j)$ and $r_p \in N_\Gamma$). It turns out that r_m is reachable from any vertex in N_Γ by a path consisting of

edges from $E_\Gamma \cap E_{Tr}(\mathbf{q})$ and that each vertex reachable from r_m belongs to N_Γ. Obviously, since all the vertices from N_Γ but r_m are *busy*, all the edges from $E_\Gamma \cap E_{Tr}(\mathbf{q})$ represent *blocked* transitions but the edges incident into r_m, representing *feasible* transitions. Hence the proof.

The following Remarks further enlighten the properties of a strong subdigraph Γ satisfying Lemma 1.

Remark A1: From L1b) and L1c) it follows that each edge of $D_{Tr}(\mathbf{q})$ incident out a vertex of N_Γ is in E_Γ. Thus such an edge is incident into a node still belonging to N_Γ.

Remark A2: By L1d) it follows that there exists at least an edge from $E_\Gamma \cap E_{Tr}(\mathbf{q})$ ending in r_m and representing a *feasible* transition. For example e_{24} (e_{54}) is a feasible transition of $E_{\Gamma_1} \cap E_{Tr}(\mathbf{q})$ (of $E_{\Gamma_2} \cap E_{Tr}(\mathbf{q})$) in the SLD state of Figure 2(b).

Remark A3: Let $\gamma=(N_\gamma,E_\gamma)$ be a cycle from Γ containing the *idle* (or *empty*) resource r_m. From the proof of L1d) it follows that set $E_\gamma \cap E_{Tr}(\mathbf{q})$ contains all the edges of cycle γ but the edge incident out r_m at most. This means that only if $C(r_m)=1$ there is no edge from $E_{Tr}(\mathbf{q})$ incident out r_m.

Now one may ask whether there exist more strong subdigraphs of D_W deadlocked by the same job transition. The next lemma answers this question.

Lemma 2: Let $\mathbf{q}$ be not a deadlock state and let $j \in J_{\mathbf{q},u}$. If Γ is a strong subdigraph of D_W in deadlock condition in $D_{Tr}(\mathbf{q},j)$, then it is the only MZSC of $D_{Tr}(\mathbf{q},j)$.

Proof

By contradiction, assume that $D_{Tr}(\mathbf{q},j)$ contains another MZSC, say $\Gamma_1=(N_{\Gamma 1},E_{\Gamma 1})$ with $\Gamma \neq \Gamma_1$. Putting $r_i=HR(j)$ and $r_m=SR(j)$, by L1b) we get $r_m \in N_\Gamma \cap N_{\Gamma 1}$. Since two different strong components may not have a node in common, it follows $\Gamma=\Gamma_1$, a contradiction. •

Now let $\mathbf{q}$ be an SLD state for S with J_S and R_S being the corresponding job and resource sets introduced by Definition 3. Moreover assume that such sets are *minimal*, i.e. no subsets $R_{S1} \subset R_S$ and $J_{S1} \subset J_S$ exist (with $R_{S1} \neq R_S$) satisfying Definition 3. By D3c) it follows that for each $j \in J_S \cap J_{\mathbf{q},u}$ there exists a strong subdigraph of D_W in deadlock condition in $D_{Tr}(\mathbf{q},j)$. So, if set H collects all the strong subdigraphs of D_W deadlocked by the jobs in $J_S \cap J_{\mathbf{q},u}$, the following result holds.

Lemma 3: The strong subdigraphs in set $H=\{\Gamma_1,\ldots,\Gamma_\mu,\ldots,\Gamma_H\}$ satisfy the following conditions:

L3a) for each $\Gamma \in H$, there exists at least one job $j \in J_S \cap J_{\mathbf{q},u}$ such that Γ is in deadlock condition in $D_{Tr}(\mathbf{q},j)$;

L3b) for each $j \in J_S \cap J_{\mathbf{q},u}$ there exists $\Gamma \in H$ that is in deadlock condition in $D_{Tr}(\mathbf{q},j)$;

L3c) all the vertices from N_H are in R_S;

L3d) all the *feasible* transitions from E_H are in E_S;

L3e) in the state $\mathbf{q}$ there exists only one *empty* resource in N_H (say r_m). For such a resource, $C(r_m)=1$ and $r_m \in N_{\Gamma\mu}$ for each $\mu \in \{1,2,\ldots,H\}$. All the remaining resources from N_H are *busy*.

Proof

L3a) and L3b) are true.

This proof is a straightforward consequence of the construction of H.

L3c) is true.

Let $\Gamma \in H$ and $r_k \in N_\Gamma$. By L3a), there exists $j \in J_S \cap J_{\mathbf{q},u}$ such that Γ is in deadlock condition in $D_{Tr}(\mathbf{q},j)$. Let $\mathbf{q}'$ indicate the corresponding deadlock state. The sets $R_D=N_\Gamma$ and J_D, where J_D collects all the jobs that in state $\mathbf{q}'$ hold resources from N_Γ, satisfy Definition 1 (see Remark 1). So let j_0 denote a job from J_D such that $HR(j_0)=r_k$ in $\mathbf{q}'$. Since by D3c) $J_D \subset J_S$, j_0

is also in J_S and two cases are in order. If j_0=j, we get r_k=SR(j_0)∈SR(J_S) in state **q**, so that D3a) implies $r_k \in R_S$. If, on the other hand, $j_0 \neq$j we get r_k=HR(j_0)∈HR(J_S) in state **q**, so that D3a) yields $r_k \in R_S$ again, thus completing the proof.

L3d) is true.

Let $\Gamma \in H$ and $e_{im} \in E_F(\mathbf{q}) \cap E_\Gamma$. By L3c) it holds $r_i, r_m \in R_S$. Moreover, since e_{im} is *feasible*, there exists $j \in J_{\mathbf{q},u}$ such that HR(j)=r_i and SR(j)=r_m. Since r_i is in R_S, by D3b) $j \in J_S$. Therefore, by definition of E_S, we conclude $e_{im} \in E_S$.

L3e) is true.

By L3c) $N_H \subset R_S$. Moreover by L3a), in state **q** any $\Gamma \in H$ satisfies Lemma 1, so that there exists at least an *idle* or *empty* resource in N_H (say $r_m \in N_\Gamma$). In particular, if C(r_m)=1 the resource r_m is certainly *empty* in state **q**. We now show that this is just the case.

We prove that the assumption C(r_m)>1 leads to a contradiction. Since L1b) yields $OV_{\mathbf{q}}(r_m)$=C(r_m)-1, $D_{Tr}(\mathbf{q})$ certainly contains an edge (say e_{mk}) incident out r_m. Moreover, by L1d), $r_k \in N_\Gamma$ and r_m is reachable from each vertex in N_Γ. Consequently, there exists a path p in $D_{Tr}(\mathbf{q})$ from r_k to r_m, with all its edges in E_Γ and ending with a *feasible* transition, say $e_{im} \in E_\Gamma$. Considering edge e_{mk} followed by path p, in which the last edge e_{im} is dropped, we realize that r_i is reachable from r_m. Again L1d) certainly implies $r_i \in N_\Gamma$. However it is just this fact that leads to a contradiction. Namely, by L3d) it holds $e_{im} \in E_S$. Thus a job $j \in J_S \cap J_{\mathbf{q},u}$ can be found such that HR(j)=r_i and SR(j)=r_m and by L3b) there exists $\Gamma_1 \in H$ in deadlock condition in $D_{Tr}(\mathbf{q},j)$. Taking into account L1b), we get $r_i \notin N_{\Gamma_1}$ and $r_m \in N_{\Gamma_1}$. However, as r_i is reachable from r_m, L1d) yields $r_i \in N_{\Gamma_1}$. This is indeed the just mentioned contradiction.

At this point, we show that, in state **q**, r_m is the only *empty* resource from N_H. Assume the contrary, i.e. there exists another *empty* resource $r_\ell \in N_H$, with $r_\ell \neq r_m$. Let H_1 collect all the strong subdigraphs of H containing r_m and let N_{H_1} and E_{H_1} be the sets of all the vertices and edges in such strong subdigraphs. By L1a), r_m is the only *empty* resource in N_{H_1}, so that $r_\ell \notin N_{H_1}$. If we put $R_{S_1}=N_{H_1}$ and $J_{S_1}=\{j \in J_{\mathbf{q}}: HR(j) \in R_{S_1}\}$, by construction the sets R_{S_1} and J_{S_1} satisfy D3b) and the condition $HR(J_{S_1}) \subset R_{S_1}$. Moreover, as by Remark A1 each edge from $E_{Tr}(\mathbf{q})$ incident out a vertex of N_{H_1} still ends in N_{H_1}, it holds $SR(J_{S_1}) \subset R_{S_1}$. Hence D3a) is also verified. It is finally easy to show that R_{S_1} and J_{S_1} satisfy D3c) too. Namely, given $j \in J_{S_1} \cap J_{\mathbf{q},u}$, let $\Gamma \in H$ be the strong subdigraph of D_W in deadlock condition in $D_{Tr}(\mathbf{q},j)$. As $HR(j) \in N_{H_1}$, Remark A1 yields $SR(j) \in N_{H_1}$. Moreover, as all the resources in N_{H_1} but r_m are *busy* (see L1a), it follows SR(j)=r_m. Now, L1b) gives $r_m \in N_\Gamma$ so that: $N_\Gamma \subset N_{H_1}$. Consequently, for each job $j \in J_{S_1} \cap J_{\mathbf{q},u}$ the transition releasing HR(j) to hold SR(j) leads to a deadlock involving only jobs from J_{S_1}, i.e. $J_D \subset J_{S_1}$. In other words, R_{S_1} and J_{S_1} satisfy D3c) too. In conclusion, the above arguments show that there exist the proper subsets $R_{S_1} \subset R_S$ and $J_{S_1} \subset J_S$ (with $R_{S_1} \neq R_S$) satisfying Definition 3. However this means that R_S and J_S are not minimal, which is a contradiction.

At this point, to complete the proof, it is sufficient to note that by Lemma 1, each strong subdigraph from H has only one *idle* vertex while all the remaining resources are *busy*. So r_m is shared by all the strong subdigraphs from H.

With the help of the preceding lemmas and notations it is now easy to prove Theorem 2.

Proof of Theorem 2

Sufficiency

The proof is by construction. To begin with, let R_S be the set of resources corresponding to the vertices of the strong subdigraphs in H:

$$R_S = N_H \qquad (22)$$

and let J_S be the complete set of jobs holding resources in R_S:

$$J_S = \{j \in J_{\mathbf{q}} : HR(j) \in R_S\} \quad (23)$$

Definition (23) implies $HR(J_S) \subset R_S$. Furthermore, by Remark A1, for any strong subdigraph $\Gamma \in H$, each edge of $D_{Tr}(\mathbf{q})$ outgoing from a vertex in N_Γ ends in a node still belonging to N_Γ. This means that $SR(J_S) \subset R_S$. Thus sets R_S and J_S enjoy D3a). Moreover, by (23), they satisfy D3b) too.

Finally, by Remark A1, the feasible transitions of $D_{Tr}(\mathbf{q})$ corresponding to jobs in $J_S \cap J_{\mathbf{q},u}$ are in E_H. Hence, by Th2c), for each $j \in J_S \cap J_{\mathbf{q},u}$ there exists a strong subdigraph $\Gamma \in H$ in deadlock condition in $D_{Tr}(\mathbf{q},j)$. Since in this situation all the resources in N_Γ are held by jobs from J_S, we get $J_D \subset J_S$, as D3c) requires.

Necessity

If we assume that $\mathbf{q}$ is an SLD state, there exists a set H of strong subdigraphs enjoying Lemma 3. So, Th2a) and Th2b) follow from L3e) and L3a), respectively.

To prove Th2c), let $j \in J_{\mathbf{q},u}$ be such that $e_{im} \in E_H \cap E_F(\mathbf{q})$,with r_i=HR(j) and r_m=SR(j). Obviously we get $r_i \in N_H$ and, by L3c), $r_i \in R_S$. Moreover, by D3b), $j \in J_S$ and, according to L3b), condition Th2c) follows. •

Appendix 2

Proof of Theorem 3

Proving this result requires four steps.

The first Step introduces a new digraph to exhibit a relationship among the strong subdigraphs in set H of Theorem 2.

At the second Step we select a subset $H_1 \subset H$ whose elements are in a cyclic relationship: namely, any strong subdigraph from H_1 contains a feasible transition that deadlocks another strong subdigraph in the same set.

At Step 3, we select a particular cycle in each strong subdigraph from H_1. Moreover we prove that all the cycles and any two cycles chosen in that way share only one vertex.

Finally, Step 4 shows that the cycles determined at Step 3 enjoy a particular property: they correspond to the vertices of a second level cycle from Γ^2.

Step 1

If $\mathbf{q}$ is an SLD state, D_W contains a set H of strong subdigraphs satisfying Theorem 2. Now, starting with H, we build a new digraph $D_H(\mathbf{q})=[V_H,L_H(\mathbf{q})]$, by associating a vertex $v_\mu \in V_H$ to each strong subdigraph $\Gamma_\mu \in H$ and by considering the edge $\ell_{\mu\nu}=(v_\mu,v_\nu)$ in $L_H(\mathbf{q})$ iff there exists $j \in J_{\mathbf{q},u}$ such that, putting r_i=HR(j) and r_m=SR(j), the feasible transition e_{im} belongs to E_{Γ_μ} and Γ_ν is in deadlock condition in $D_{Tr}(\mathbf{q},j)$. By Remark A2 and Th2c), the outdegree of each vertex from V_H is greater than zero. This ensures that digraph $D_H(\mathbf{q})$ contains at least one cycle [10]. Now, from among the cycles of $D_H(\mathbf{q})$ we choose one of minimum length (say γ_{H1}). Cycle γ_{H_1} cannot be a selfloop, because L1b) excludes this possibility. Moreover, in $D_H(\mathbf{q})$ any two nodes of γ_{H_1} can be only connected by edges belonging to γ_{H1}. Otherwise, γ_{H_1} would not be a minimum length cycle.

Step 2

Let $H_1 \subset H$ be the set of strong subdigraphs corresponding to vertices of γ_{H_1}. By Lemma 3 there is only one *empty* resource in $D_{Tr}(\mathbf{q})$ (i.e. r_m) shared by all the strong subdigraphs from H. Moreover let $\ell_{\mu\nu}$ be an edge of the cycle γ_{H_1}. By construction this means that there exists $j \in J_{\mathbf{q},u}$ such that, putting r_i=HR(j), it holds: $e_{im} \in E_F(\mathbf{q}) \cap E_{\Gamma_\mu}$ and Γ_ν is in deadlock condition in $D_{Tr}(\mathbf{q},j)$. In the sequel we also write $e_\mu=e_{im}$ to exhibit that e_{im} belongs to E_{Γ_μ} (i.e. to the strong subdigraph Γ_μ). By using this notation for each edge of γ_{H_1}, we can associate a *feasible* transition of $D_{Tr}(\mathbf{q})$ to any element from H_1.

that immediately follows v_μ in the cycle γ_{H_1}. We state that for each $\Gamma_\rho \in H_1$ with $\Gamma_\rho \neq \Gamma_\mu$ it holds $e_\mu \notin E_{\Gamma_\rho}$. To prove this, assume the contrary, i.e. $e_\mu \in E_{\Gamma_\rho}$. Thus, there exists $j \in J_{q,u}$ that deadlocks Γ_ν by executing transition e_μ. Hence, by construction, $L_H(\mathbf{q})$ contains both the edges $\ell_{\mu\nu}$ and $\ell_{\rho\nu}$, with $v_\mu \neq v_\rho$ (see Figure 4). Obviously, since $\ell_{\mu\nu}$ is an edge of cycle γ_{H_1}, $\ell_{\rho\nu}$ does not belong to such a cycle. But $\ell_{\rho\nu}$ joins two vertices of γ_{H_1}: a

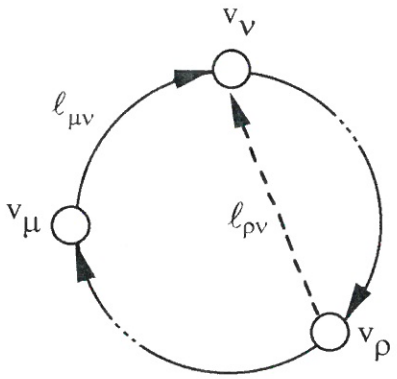

Figure 4. Subdigraph of $D_H(q)$ Showing Cycle γ_{H_1} and Edge $\ell_{\rho\nu}$

contradiction, because γ_{H_1} is a cycle of minimum length in $D_H(\mathbf{q})$.

Step 3

For any $\Gamma_\mu \in H_1$ let us consider a cycle $\gamma_\mu = (N_{\gamma_\mu}, E_{\gamma_\mu}) \subset \Gamma_\mu$ of D_W, such that $e_\mu \in E_{\gamma_\mu}$. Such a cycle certainly exists because Γ_μ is a strong subdigraph of D_W. Moreover, as e_μ is a *feasible* transition in $D_{Tr}(\mathbf{q})$, its ending vertex is the only *empty* resource of the strong subdigraphs from H: thus $r_m \in N_{\gamma_\mu}$. At this Step we prove that for each $\Gamma_\mu, \Gamma_\rho \in H_1$ with $\Gamma_\mu \neq \Gamma_\rho$, it holds $N_{\gamma_\mu} \cap N_{\gamma_\rho} = \{r_m\}$.

Proceeding by contradiction, suppose that there exists a vertex $r_k \neq r_m$ such that $r_k \in N_{\gamma_\mu} \cap N_{\gamma_\rho}$. By L1a), r_k is *busy* in state $\mathbf{q}$. Moreover, by Remark A3, there exists a path from r_k to r_m in $D_{Tr}(\mathbf{q})$, constituted by edges of E_{γ_μ}. Obviously, such a path ends with the *feasible* transition e_μ. Since $r_k \in N_{\gamma_\mu} \cap N_{\gamma_\rho}$, by Remark A1 it holds $e_\mu \in E_{\Gamma_\mu} \cap E_{\Gamma_\rho}$, that contradicts the statement proven at Step 2.

Step 4

Let γ_μ and γ_ν be two first level cycles defined according to Step 3 and corresponding to a pair $\Gamma_\mu, \Gamma_\nu \in H_1$. Moreover let n^2_μ and n^2_ν be the nodes of the Second Level Digraph D^2_W representing γ_μ and γ_ν.

We now show that if γ_{H_1} contains the edge $\ell_{\mu\nu}$, then there exists the edge $e^2_{\mu\nu}$ in D^2_W. First of all let us note that, as proven at Step 3, γ_μ and γ_ν satisfy D4a). Secondly if $\ell_{\mu\nu} \in L_H(\mathbf{q})$ then there exists a job $j \in J_{q,u}$ that, by executing transition e_μ, deadlocks the strong subdigraph Γ_ν. By Remark A3 and L3e), set $E_{Tr}(\mathbf{q}) \cap E_{\gamma_\nu}$ contains all the edges of E_{γ_ν}, but the edge starting from the *empty* resource r_m. Moreover, it holds $E_{Tr}(\mathbf{q},j) \cap E_{\gamma_\nu} = E_{\gamma_\nu}$. Putting $r_i = HR(j)$ and $r_p = TR(j)$ and noting that $SR(j) = r_m$, it is easy to realize that $e_{mp} \in E_{\gamma_\nu}$ and $e_\mu = e_{im} \in E_{\gamma_\mu}$. Hence cycles γ_μ and γ_ν satisfy D4b) too, so that $e^2_{\mu\nu} \in E^2_W$.

From this proof, it immediately follows that corresponding to the cycle γ_{H1} there exists a cycle γ^2 in D^2_W and, by Step 3, such a cycle belongs to Γ^2. Finally, since in state $\mathbf{q}$ all the vertices of the first level cycles corresponding to the nodes of γ^2 are busy but r_m, we get:

$$Card(J_q) \geq [C(N^2_\gamma) - 1]$$

REFERENCES

1. BANASZAK, Z.A. and KROGH, B.H., **Deadlock Avoidance in Flexible Manufacturing Systems with Concurrently Competing Process Flows**, IEEE TRANS. ON ROBOTICS AND AUTOMATION, Vol. 6, No. 6, December 1990, pp. 724-734.
2. CHO, H., KUMARAN, T.K. and WYSK, R.A., **Graph-Theoretic Deadlock Detection and Resolution for Flexible Manufacturing Systems,** IEEE TRANS. ON ROBOTICS AND

AUTOMATION, Vol. 11, No.3, June 1995, pp. 413-421.

3. DESROCHERS, A.A. and AAL-JAAR, R.Y., **Applications of Petri Nets in Manufacturing Systems: Modeling, Control and Performance Analysis**, IEEE PRESS, New York, 1955.
4. EZPELETA, J., COLOM, J.M. and MARTINEZ, J., **A Petri Net Based Deadlock Prevention Policy for Flexible Manufacturing Systems**, IEEE TRANS. ON ROBOTICS AND AUTOMATION, Vol. 11, No. 2, April 1995, pp. 173-184.
5. FANTI, M.P., MAIONE, B., PISCITELLI, G. and TURCHIANO, B., **System Approach to A Design Generic Software for Real-time Control of Flexible Manufacturing System**, IEEE TRANS. ON SYSTEMS, MAN AND CYBERNETICS - PART A: SYSTEMS AND HUMANS, Vol. 26, No. 2, March 1996, pp. 190-202.
6. FANTI, M.P., MAIONE, B. and TURCHIANO, B., **Digraph-Theoretic Approach for Deadlock Detection and Recovery in Flexible Production Systems**, STUDIES IN INFORMATICS AND CONTROL, Vol. 5, No. 4, 1996, pp. 373-383.
7. FANTI, M.P., MAIONE, B., MASCOLO, S. and TURCHIANO, B., **Low-cost Deadlock Avoidance Policies for Flexible Production Systems**, INT. J. ON MODELLING AND SIMULATION, June 1997.
8. FANTI, M.P., MAIONE, B., MASCOLO, S. and TURCHIANO, B., **Event Based Feedback Control for Deadlock Avoidance in Flexible Manufacturing Systems**, IEEE TRANS. ON ROBOTICS AND AUTOMATION, Vol. 13, No. 3, June 1997.
9. HARARY, F., **Graph Theory** , ADDISON-WESLEY PUBLISHING COMPANY, INC., USA, 1971.
10. HARARY, F., NORMAN, R.Z. and CARTWRIGHT, D., **Structural Models: An Introduction to the Theory of Directed Graphs**, JOHN WILEY & SONS, INC., New York, 1965.
11. HSIEH, F. and CHANG, S., **Dispatching-Driven Deadlock Avoidance Controller Synthesis for Flexible Manufacturing System**, IEEE TRANS. ON ROBOTICS AND AUTOMATION, Vol. 10, No. 2, April 1994, pp. 196-209.
12. KUMARAN, T.K., CHANG, W., CHO, H. and WYSK, R.A., **A Structured Approach to Deadlock Detection, Avoidance and Resolution in Flexible Manufacturing Systems**, INT. J. PROD. RES., Vol. 32, No. 10, 1994, pp. 2361-2379.
13. REINGOLD, E.M., NIEVERGELT, J. and DEO, N., **Combinatorial Algorithms**, PRENTICE HALL, Englewood Cliffs, NJ, 1977.
14. REVELIOTIS, S.A. and FERREIRA, P.M., **Deadlock Avoidance Policies for Automated Manufacturing Cells**, IEEE TRANS. ON ROBOTICS AND AUTOMATION, Vol. 12, 1996, pp. 845-857.
15. VISWANADHAM, N. and NARAHARI, V., **Performance Modeling of Automated Manufacturing Systems,** PRENTICE HALL, Englewood Cliffs, NJ, 1992
16. VISWANADHAM , N., NARAHARI, Y. and JOHNSON, T.L., **Deadlock Prevention and Deadlock Avoidance in Flexible Manufacturing Systems Using Petri Net Models**, IEEE TRANS. ON ROBOTICS AND AUTOMATION, Vol. 6, No. 6, December 1990, pp. 713-723.
17. WYSK, R.A., YANG, N.S. and JOSHI, **Detection of Deadlocks in Flexible Manufacturing Cells**, IEEE TRANS. ROBOT. AUTOMAT., Vol. 7, 1991, pp. 853-859.
18. XING, K.Y., HU, B.S. and CHEN, H.X., **Deadlock Avoidance Policy for Petri Net Modeling of Flexible Manufacturing Systems with Shared Resources,** IEEE TRANS. ON AUT. AND CONTROL, Vol. 41, No. 2, February 1996, pp. 289-295.
19. ZEIGLER, **Multifacetted Modelling and Discrete Event Simulation**, ACADEMIC PRESS INC., London, 1984.
20. ZHOU, M. and DI CESARE, F., **Petri Net Synthesis for Discrete Event Control of Manufacturing Systems**, KLUWER ACADEMIC PUBLISHERS, Boston, 1993.