01 Aug 1993

Formal Model and Specification of Deadlock

Pei-yu Li

Bruce M. McMillin
Missouri University of Science and Technology, ff@mst.edu

Formal Model and Specification of Deadlock

Pei-yu Li and Bruce McMillin

August 1993

CSC-93-31

Abstract

In this paper, we present a formal model of deadlock in a distributed system and develop the deadlock specification in terms of time-dependent predicates. Primitive activities of processes in the distributed system are specified by the predicates so that system behaviors can be described by logic operations. With the formal model, we have an insight into the definition of deadlock in local views. A rigorous proof to show the equivalence of local-time and global-time deadlock specifications is presented. The local-time deadlock specification, which defines the timing of dependence between deadlocked processes, will be useful in the correctness verification of distributed deadlock detection/resolution algorithms.

Department of Computer Science
University of Missouri-Rolla
Rolla, Missouri 65401

1 Introduction

A distributed system consists of a set of processors connected by bidirectional communication links, with processes and resources resident on each processor. Processes and resources communicate with each other via message passing. Deadlock detection/resolution is an important problem in a distributed system, and much attention has been devoted to it in the past few years. A large number of distributed deadlock detection/resolution algorithms were proposed, and many of them have been found to be incorrect [1, 2, 3, 4].

Very few sophisticated formal methods for the correctness proof of deadlock detection algorithms are existing because the formal correctness proof is difficult [5]. To formally show the algorithm correctness by the time-dependent proof technique, a formal deadlock model which specifies the timing of dependence between deadlocked processes is needed.

In this paper, we present a formal deadlock model with global and local clocks, and develop the deadlock specification by using time-dependent predicates. With this model, we have an insight into the definition of deadlock in local views. We also give a rigorous proof to show the equivalence of local-time and global-time specifications.

2 Formal model and specification of deadlock

Consider a distributed system in which each processor has a local clock. These clocks never stop and measure time in discrete units (*ticks*) independently, i.e., clocks are incremented one tick at a time.

Each resource is maintained by a resource manager which has the exclusive right to operate on a resource. If a process wants to access a resource, it must send a request message to the resource manager which manages the resource. The resource manager itself is a process, therefore the resource in this paper is considered as a kind of process. Each time a resource is exclusively granted to only one process. A process acquires a resource by receiving a grant message from the resource and releases a resource by sending a release message to the resource. A process can request a resource, only once, for execution and cannot proceed (e.g., doing computation, sending request messages, releasing acquired resources, etc.) until it has acquired the requested resource. Spontaneous aborts of processes are not permitted. Deadlock is a situation in which a set of processes are in a simultaneous wait state, each waiting for one of the others to release its acquired resources before proceeding. The following definition presents this formally and explicitly. Note that addition and subtraction in subscripts throughout this paper are modulo n, $n \geq 2$.

Definition 1 A set of processes $\mathcal{P} = \{p_0, p_1, \ldots, p_{n-1}\}$ are deadlocked over a set of resources $\mathcal{R} = \{r_0, r_1, \ldots, r_{n-1}\}$ at time λ *if and only if* for all i, $0 \leq i \leq n-1$, process p_i wants to access resource r_i and resource r_i is held by process p_{i+1}, simultaneously at time λ.

Problems in a distributed system can be specified by time-dependent predicates which formulate the characteristics of elements in the system. The following four predicates are defined to capture the primitive activities of processes and then construct our global-time and local-time specifications for the distributed deadlock.

$$R(p,r,t) = \begin{cases} 1 & \text{if } p \text{ requests } r \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

$$A(p,r,t) = \begin{cases} 1 & \text{if } p \text{ acquires } r \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

$$L(p,r,t) = \begin{cases} 1 & \text{if } p \text{ releases } r \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

$$G(p,r,t) = \begin{cases} 1 & \text{if } r \text{ sends a grant message to } p \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

In general, we will use Greek letters (e.g., $\alpha, \alpha_i, \alpha_i'$, etc.) to refer to global times, and English letters (e.g., a, a_i, a_i', etc.) to refer to local times.

2.1 Global-time specification

Definition 1 captures current definitions of a deadlock in terms of a wait-for graph (WFG) or resource allocation graph (RAG). However, the simultaneity implicitly assumes the existence of a global real-time clock, since the nature of λ is related to real-time. To formalize these simultaneous holding and waiting of deadlocked processes, the *global-time specification* for deadlock, D_G, is defined as below.

D_G: $\exists\lambda,\ \forall i,\ 0 \le i \le n-1$

$$(\exists\alpha_i, \beta_i,\ \forall\alpha_i', \beta_i',\ \alpha_i \le \alpha_i' \le \lambda,\ \beta_i \le \beta_i' \le \lambda$$

$$A(p_i, r_{i-1}, \alpha_i) \wedge \neg L(p_i, r_{i-1}, \alpha_i') \wedge$$
$$R(p_i, r_i, \beta_i) \wedge \neg A(p_i, r_i, \beta_i'))$$

2.2 Local-time specification

As we have seen, the global-time specification refers to real-time. In a distributed system which has no global clock, everything that can be specified is based on locally observable process activities. The *local-time specification* for deadlock, D_L, is presented in terms of local time-dependent predicates. Time t in predicate $G(p, r, t)$ is associated with the local clock of the processor where resource r is resident, while in predicates $R(p, r, t)$, $A(p, r, t)$ and $L(p, r, t)$, time t is associated with the local clock of the processor where process p resides. More specifically, $D_L(1)$ is for process p_i and $D_L(2)$ for resource r_i, where $0 \le i \le n-1$.

D_L: $\forall i,\ 0 \le i \le n-1$

(1) $\exists a_i, b_i, t_i,\ \forall a_i', b_i',\ a_i \le a_i' \le t_i,\ b_i \le b_i' \le t_i$

$$A(p_i, r_{i-1}, a_i) \wedge \neg L(p_i, r_{i-1}, a_i') \wedge$$
$$R(p_i, r_i, b_i) \wedge \neg A(p_i, r_i, b_i') \wedge$$

(2) $\exists c_i,\ \forall c_i' \le c_i,\ G(p_{i+1}, r_i, c_i) \wedge \neg G(p_i, r_i, c_i')$

Figure 1 shows an example of time-space diagram, which explains the necessity of $D_L(2)$. In this diagram, a solid arrow represents a process sends a request message to a resource, a

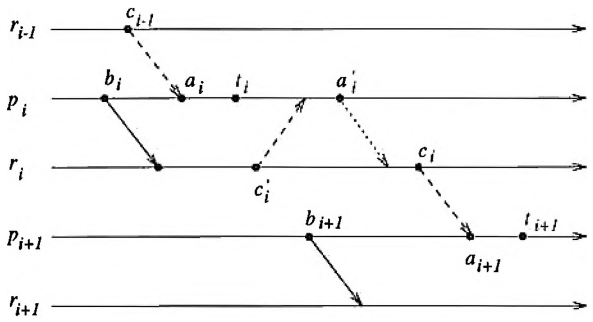

Figure 1: An example of time-space diagram in terms of local times. Process p_i was blocked at time t_i, but acquires resource r_i after t_i and then releases r_i at a_i'. In this situation, p_i is no longer blocked, but $D_L(1)$ is still satisfied.

dashed arrow represents a resource sends a grant message to a process, and a dotted arrow represents a process sends a release message to a resource. The equivalence of local-time and global-time specifications will be proved in the next section.

3 Equivalence of global and local specifications

To relate the local specification with the global specification, the deadlock model assumes the existence of a global clock. Since we are accustomed to using global time in our thinking, the use of global time can simplify the description of problems in a distributed system. Every local activity of processes in the system is associated with a point in the global time, that is, there are two different views, local and global views, corresponding to each activity of processes in the system. To express this more specifically, clock transformation functions are defined to facilitate transformation between arbitrary local-time coordinate and the global-time coordinate. Without loss of generality, assume processes in the deadlock model are resident on different processors, that is, each process refers to its own local clocks. Let $C'_q(t)$ be the tick rate at which local clock for q is running at local time t, $q \in \mathcal{Q} = \mathcal{P} \cup \mathcal{R}$, $\mathbf{N}$ the set of non-negative integers and $\mathbb{R}$ the set of non-negative real numbers.

Definition 2 Let $C_q : \mathbf{N} \xrightarrow{1-1} \mathbb{R}$ be a function from local-time values to global-time values

$$C_q(t) = \sum_{i=0}^{t-1} C'_q(i), \quad C_q(0) = 0, \quad \mathrm{Rng}(C_q) = \mathcal{T}_q \subset \mathbb{R}.$$

$C_q^{-1} : \mathcal{T}_q \xrightarrow{1-1} \mathbf{N}$ is a function from global-time values to local-time values.

3.1 Axioms

The following axioms are constructed to formalize all necessary operational assumptions on which the proof of our main theory is based. Note that all the time variables are non-negative, therefore, $a \leq b$ is equivalent to $0 \leq a \leq b$ and $\alpha \leq \beta$ is equivalent to $0 \leq \alpha \leq \beta$.

• **View axioms:**

A1. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall a \in \mathbf{N},\ (E(p,r,a) \Rightarrow E(p,r,C_q(a)))$
$\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha \in \mathbb{R},\ (E(p,r,\alpha) \Rightarrow E(p,r,C_q^{-1}(\alpha)))$
where $E = A, L, R$ or G

$$q = \begin{cases} r & \text{if } E = G \\ p & \text{otherwise} \end{cases}$$

There is a global-time activity corresponding to a local-time activity, and vice versa.

A2. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall a, b \in \mathbf{N},\ (\forall c, a \leq c \leq b, E(p,r,c)$
$\Rightarrow \forall \gamma, C_q(a) \leq \gamma \leq C_q(b), E(p,r,\gamma))$

$$\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha, \beta \in \mathbb{R},\ (\forall \gamma, \alpha \leq \gamma \leq \beta, E(p, r, \gamma) \Rightarrow \forall c, C_q^{-1}(\alpha) \leq c \leq C_q^{-1}(\beta), E(p, r, c))$$

where $E = \neg A, \neg L, \neg R$ or $\neg G$

$$q = \begin{cases} r & \text{if } E = \neg G \\ p & \text{otherwise} \end{cases}$$

There is a global view corresponding to a local view, and vice versa.

- **Event ordering axioms:**

A3. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall a, b \in \mathbf{N},\ (G(p, r, a) \wedge A(p, r, b) \Rightarrow C_r(a) < C_p(b))$

The message receiver receives a message after the message sender sends it.

A4. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha \in \mathbb{R},\ (A(p, r, \alpha) \Rightarrow \exists \beta,\ \beta \prec \alpha \wedge G(p, r, \beta))$

$\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha, \beta \in \mathbb{R},\ (\forall \gamma, \alpha \leq \gamma \leq \beta, \neg G(p, r, \gamma) \Rightarrow \forall \gamma, \alpha \leq \gamma \leq \beta, \neg A(p, r, \gamma))$

A process acquires a resource after the resource sends a grant message to it.

- **Operation axioms:**

A5. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha \in \mathbb{R},\ (R(p, r, \alpha) \Rightarrow \forall \beta \leq \alpha, \neg G(p, r, \beta) \Rightarrow \forall \beta \leq \alpha, \neg A(p, r, \beta))$

A process never requests the resource it is holding.

A6. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha \in \mathbb{R},\ (\forall \beta \leq \alpha, \neg A(p, r, \beta) \Rightarrow \forall \beta \leq \alpha, \neg L(p, r, \beta))$

A process only releases the resource it is holding.

A7. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha \in \mathbb{R},\ (\exists p', p' \neq p \wedge G(p', r, \alpha) \wedge \forall \beta \leq \alpha, \neg L(p, r, \beta) \Rightarrow \forall \beta \leq \alpha, \neg G(p, r, \beta))$

If a resource has already been granted to a process, it must be released before it is granted to another process. A resource is exclusively accessed by only one process at one time.

A8. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha, \beta \in \mathbb{R},\ (G(p, r, \alpha) \wedge A(p, r, \beta) \Rightarrow \forall \gamma, \alpha \leq \gamma \leq \beta, \neg L(p, r, \gamma))$

During the period of grant message delivery, the resource is not yet held by the granted process and there is no release operation for the resource from the granted process.

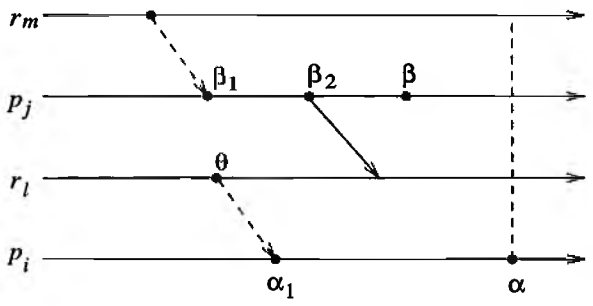

Figure 2: Global time-space diagram for Lemma 1.

A9. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha, \beta \in \mathbb{R},\ (G(p,r,\alpha) \wedge \forall \gamma, \alpha \leq \gamma \leq \beta, \neg L(p,r,\gamma)$
$\Rightarrow \forall p' \neq p \wedge \forall \gamma, \alpha \leq \gamma \leq \beta, \neg G(p',r,\gamma))$

A resource will not send a grant message to any other process while being held by one process, that is, a resource is exclusively accessed by only one process at one time.

A10. $\forall p \in \mathcal{P},\ \forall r \in \mathcal{R},\ \forall \alpha, \beta \in \mathbb{R},\ (\exists r',\ r' \neq r \wedge R(p,r',\alpha)\ \wedge \forall \theta, \alpha \leq \theta \leq \beta, \neg A(p,r',\theta) \wedge$
$\exists \gamma, \gamma \leq \beta \wedge A(p,r,\gamma) \Rightarrow \forall \theta, \alpha \leq \theta \leq \beta, \neg L(p,r,\theta))$

A process can not release the resource it is holding while waiting for another resource.

3.2 Proof of equivalence

Lemma 1 $\forall p_i, p_j \in \mathcal{P},\ \forall r_l, r_m \in \mathcal{R},\ \forall \alpha, \beta, \beta_1, \beta_2 \in \mathbb{R}$

$(\beta_1 \leq \beta \wedge \beta_2 \leq \beta \wedge \beta \leq \alpha$ (I)

$\wedge\ \exists \alpha_1, \forall \alpha', \alpha_1 \leq \alpha' \leq \alpha, A(p_i, r_l, \alpha_1) \wedge \neg L(p_i, r_l, \alpha')$ (II)

$\wedge\ \forall \beta', \beta_1 \leq \beta' \leq \beta, A(p_j, r_m, \beta_1) \wedge \neg L(p_j, r_m, \beta')$ (III)

$\wedge\ \forall \beta', \beta_2 \leq \beta' \leq \beta, R(p_j, r_l, \beta_2) \wedge \neg A(p_j, r_l, \beta')$ (IV)

$\wedge\ \exists \theta, \forall \theta' \leq \theta, G(p_i, r_l, \theta) \wedge \neg G(p_j, r_l, \theta')$ (V)

$\Longrightarrow \forall \beta', \beta_1 \leq \beta' \leq \alpha, A(p_j, r_m, \beta_1) \wedge \neg L(p_j, r_m, \beta') \wedge$
$\forall \beta', \beta_2 \leq \beta' \leq \alpha, R(p_j, r_l, \beta_2) \wedge \neg A(p_j, r_l, \beta'))$

Proof:

Figure 2 shows the time-space diagram for p_i, p_j, r_l and r_m, which will help the reader follow the proof more easily.

(a) (I) $\wedge$ (IV) $\Longrightarrow \beta \leq \alpha \wedge \beta_2 \leq \beta \wedge R(p_j, r_l, \beta_2) \Longrightarrow \beta_2 \leq \alpha \wedge R(p_j, r_l, \beta_2)$ (VI)

(V) $\Longrightarrow \exists \theta,\ \forall \theta' \leq \theta,\ \neg G(p_j, r_l, \theta') \overset{A5}{\Longrightarrow} \forall \theta' \leq \theta,\ \neg A(p_j, r_l, \theta')$ (VII)

(II) $\wedge$ (V)
$\Longrightarrow \exists\theta,\ G(p_i, r_l, \theta) \wedge \exists\alpha_1,\ \forall\alpha', \alpha_1 \leq \alpha' \leq \alpha,\ A(p_i, r_l, \alpha_1) \wedge \neg L(p_i, r_l, \alpha')$
$\stackrel{A8}{\Longrightarrow} \exists\theta,\ G(p_i, r_l, \theta) \wedge \forall\alpha', \theta \leq \alpha' \leq \alpha_1,\ \neg L(p_i, r_l, \alpha') \wedge \forall\alpha', \alpha_1 \leq \alpha' \leq \alpha,\ \neg L(p_i, r_l, \alpha')$
$\Longrightarrow \exists\theta,\ G(p_i, r_l, \theta) \wedge \forall\alpha', \theta \leq \alpha' \leq \alpha,\ \neg L(p_i, r_l, \alpha')$
$\stackrel{A9}{\Longrightarrow} \forall\alpha', \theta \leq \alpha' \leq \alpha,\ \neg G(p_j, r_l, \alpha')$
$\stackrel{A4}{\Longrightarrow} \forall\alpha', \theta \leq \alpha' \leq \alpha,\ \neg A(p_j, r_l, \alpha')$ (VIII)

(II) $\wedge$ (V) $\Longrightarrow$ (VII) $\wedge$ (VIII) $\Longrightarrow \forall\theta' \leq \alpha,\ \neg A(p_j, r_l, \theta')$ (IX)

(I) $\wedge$ (II) $\wedge$ (IV) $\wedge$ (V)
$\Longrightarrow$ (VI) $\wedge$ (IX)
$\Longrightarrow \forall\theta', \beta_2 \leq \theta' \leq \alpha,\ R(p_j, r_l, \beta_2) \wedge \neg A(p_j, r_l, \theta')$ (X)

(b) (I) $\wedge$ (III)
$\Longrightarrow \beta \leq \alpha \wedge \beta_1 \leq \beta \wedge A(p_j, r_m, \beta_1)$
$\Longrightarrow \beta_1 \leq \alpha \wedge A(p_j, r_m, \beta_1)$ (XI)

(I) $\wedge$ (II) $\wedge$ (III) $\wedge$ (IV) $\wedge$ (V)
$\Longrightarrow$ (X) $\wedge$ (XI)
$\stackrel{A10}{\Longrightarrow} \forall\beta',\ \beta_2 \leq \beta' \leq \alpha,\ \neg L(p_j, r_m, \beta')$ (XII)

(I) $\wedge$ (III) $\wedge$ (XII)
$\Longrightarrow \forall\beta', \beta_1 \leq \beta' \leq \beta, \neg L(p_j, r_m, \beta') \wedge \beta_2 \leq \beta \wedge \forall\beta',\ \beta_2 \leq \beta' \leq \alpha,\ \neg L(p_j, r_m, \beta')$
$\Longrightarrow \forall\beta', \beta_1 \leq \beta' \leq \alpha, \neg L(p_j, r_m, \beta')$ (VIII)

(I) $\wedge$ (II) $\wedge$ (III) $\wedge$ (IV) $\wedge$ (V)
$\Longrightarrow$ (XI) $\wedge$ (XIII)
$\Longrightarrow \forall\beta', \beta_1 \leq \beta' \leq \alpha,\ A(p_j, R_M, \beta_1) \wedge \neg L(p_j, r_m, \beta')$

By (a) and (b),

(I) $\wedge$ (II) $\wedge$ (III) $\wedge$ (IV) $\wedge$ (V)
$\Longrightarrow \forall\beta', \beta_1 \leq \beta' \leq \alpha, A(p_j, r_m, \beta_1) \wedge \neg L(p_j, r_m, \beta') \wedge$
$\forall\beta', \beta_2 \leq \beta' \leq \alpha, R(p_j, r_l, \beta_2) \wedge \neg A(p_j, r_l, \beta')$ $\square$

Lemma 1 simply describes that, in the global view, if process p_j is waiting on resource r_l while holding resources at time β and process p_i is holding r_l through time α ($\alpha \geq \beta$), then process p_j remains being waiting and holding through time α.

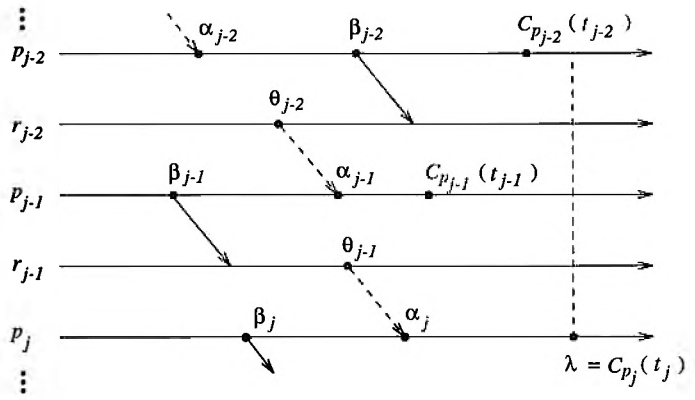

Figure 3: Global time-space diagram for the local specification.

Theorem 1 $D_L \Longleftrightarrow D_G$.

Proof:

(a) $D_L \Longrightarrow D_G$

$D_L \overset{A1,2}{\Longrightarrow} \forall i, 0 \leq i \leq n-1$

(1) $\exists \alpha_i = C_{p_i}(a_i) \wedge \exists \beta_i = C_{p_i}(b_i) \wedge \forall \alpha_i', \beta_i',\ \alpha_i \leq \alpha_i' \leq C_{p_i}(t_i),\ \beta_i \leq \beta_i' \leq C_{p_i}(t_i)$

$A(p_i, r_{i-1}, \alpha_i) \wedge \neg L(p_i, r_{i-1}, \alpha_i') \wedge$

$R(p_i, r_i, \beta_i) \wedge \neg A(p_i, r_i, \beta_i') \wedge$

(2) $\exists \theta_i = C_{r_i}(c_i) \wedge \forall \theta_i' \leq \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \neg G(p_i, r_i, \theta_i')$

Above transformation obtains the global views for deadlocked processes from the local specification, D_L. Figure 3 shows a time-space diagram for these global views. Let $\lambda = max\{C_{p_i}(t_i) \mid 0 \leq i \leq n-1\} = C_{p_j}(t_j)$. Our aim is to show, by induction, that these global views satisfy the global specification, D_G.

(i) Since $C_{p_j}(t_j) = \lambda$, we have

$\exists \alpha_j, \beta_j, \forall \alpha_j', \beta_j',\ \alpha_j \leq \alpha_j' \leq \lambda,\ \beta_j \leq \beta_j' \leq \lambda$

$A(p_j, r_{j-1}, \alpha_j) \wedge \neg L(p_j, r_{j-1}, \alpha_j') \wedge$

$R(p_j, r_j, \beta_j) \wedge \neg A(p_j, r_j, \beta_j')$

(ii) Suppose the global view for process p_k $(0 \leq k \leq n-1)$ satisfies the global specification for p_k, i.e.,

$\exists \alpha_k, \beta_k, \forall \alpha_k', \beta_k',\ \alpha_k \leq \alpha_k' \leq \lambda,\ \beta_k \leq \beta_k' \leq \lambda$

$A(p_k, r_{k-1}, \alpha_k) \wedge \neg L(p_k, r_{k-1}, \alpha_k') \wedge$

$R(p_k, r_k, \beta_k) \wedge \neg A(p_k, r_k, \beta_k')$

Then we must show that the global view for process p_{k-1} also satisfies the global specification for p_{k-1}. Consider the global views for p_k, p_{k-1} and r_{k-1}.

$$\exists \alpha_k,\ \forall \alpha_k^{'},\ \alpha_k \leq \alpha_k^{'} \leq \lambda,\ A(p_k, r_{k-1}, \alpha_k) \wedge \neg L(p_k, r_{k-1}, \alpha_k^{'}) \wedge$$

<by the hypothesis of induction>

$$\exists \alpha_{k-1}, \beta_{k-1},\ \forall \alpha_{k-1}^{'}, \beta_{k-1}^{'},\ \alpha_{k-1} \leq \alpha_{k-1}^{'} \leq C_{p_{k-1}}(t_{k-1}),\ \beta_{k-1} \leq \beta_{k-1}^{'} \leq C_{p_{k-1}}(t_{k-1})$$
$$A(p_{k-1}, r_{k-2}, \alpha_{k-1}) \wedge \neg L(p_{k-1}, r_{k-2}, \alpha_{k-1}^{'}) \wedge$$
$$R(p_{k-1}, r_{k-1}, \beta_{k-1}) \wedge \neg A(p_{k-1}, r_{k-1}, \beta_{k-1}^{'}) \wedge$$
$$\exists \theta_{k-1},\ \forall \theta_{k-1}^{'} \leq \theta_{k-1},\ G(p_k, r_{k-1}, \theta_{k-1}^{'}) \wedge \neg G(p_{k-1}, r_{k-1}, \theta_{k-1}^{'})$$
$$\stackrel{Lemma1}{\Longrightarrow} \exists \alpha_{k-1}, \beta_{k-1},\ \forall \alpha_{k-1}^{'}, \beta_{k-1}^{'},\ \alpha_{k-1} \leq \alpha_{k-1}^{'} \leq \lambda,\ \beta_{k-1} \leq \beta_{k-1}^{'} \leq \lambda$$
$$A(p_{k-1}, r_{k-2}, \alpha_{k-1}) \wedge \neg L(p_{k-1}, r_{k-2}, \alpha_{k-1}^{'}) \wedge$$
$$R(p_{k-1}, r_{k-1}, \beta_{k-1}) \wedge \neg A(p_{k-1}, r_{k-1}, \beta_{k-1}^{'})$$

By lemma 1, we show that the global view for process p_{k-1} satisfies the global specification for p_{k-1}. We have a sequence of processes, $p_j, p_{j-1}, \ldots, p_0, p_{n-1}, \ldots, p_{j+1}$. By induction, the global views for processes p_i, $0 \leq i \leq n-1$, satisfy D_G.

(b) $D_G \Longrightarrow D_L$

(1) $D_G \stackrel{A1,2}{\Longrightarrow} \forall i, 0 \leq i \leq n-1$

$$(\exists a_i = C_{p_i}^{-1}(\alpha_i) \wedge \exists b_i = C_{p_i}^{-1}(\beta_i) \wedge \forall a_i^{'}, b_i^{'},\ a_i \leq a_i^{'} \leq C_{p_i}^{-1}(\lambda),\ b_i \leq b_i^{'} \leq C_{p_i}^{-1}(\lambda)$$
$$A(p_i, r_{i-1}, a_i) \wedge \neg L(p_i, r_{i-1}, a_i^{'}) \wedge$$
$$R(p_i, r_i, b_i) \wedge \neg A(p_i, r_i, b_i^{'}))$$

$$\Longrightarrow \forall i, 0 \leq i \leq n-1$$
$$(\exists t_i \leq C_{p_i}^{-1}(\lambda) \wedge \exists a_i, b_i,\ \forall a_i^{'}, b_i^{'},\ a_i \leq a_i^{'} \leq t_i,\ b_i \leq b_i^{'} \leq t_i$$
$$A(p_i, r_{i-1}, a_i) \wedge \neg L(p_i, r_{i-1}, a_i^{'}) \wedge$$
$$R(p_i, r_i, b_i) \wedge \neg A(p_i, r_i, b_i^{'}))$$

$$\Longrightarrow \forall i, 0 \leq i \leq n-1$$
$$(\exists a_i, b_i, t_i,\ \forall a_i^{'}, b_i^{'},\ a_i \leq a_i^{'} \leq t_i,\ b_i \leq b_i^{'} \leq t_i$$
$$A(p_i, r_{i-1}, a_i) \wedge \neg L(p_i, r_{i-1}, a_i^{'}) \wedge$$
$$R(p_i, r_i, b_i) \wedge \neg A(p_i, r_i, b_i^{'}))$$

$$\Longrightarrow D_L(1)$$

(2) To prove $D_L(2)$, Consider the global specifications for p_i and p_{i+1}, $0 \leq i \leq n-1$.

$$\exists \alpha_{i+1},\ \forall \alpha_{i+1}^{'},\ \alpha_{i+1} \leq \alpha_{i+1}^{'} \leq \lambda,\ A(p_{i+1}, r_i, \alpha_{i+1}) \wedge \neg L(p_{i+1}, r_i, \alpha_{i+1}^{'})$$

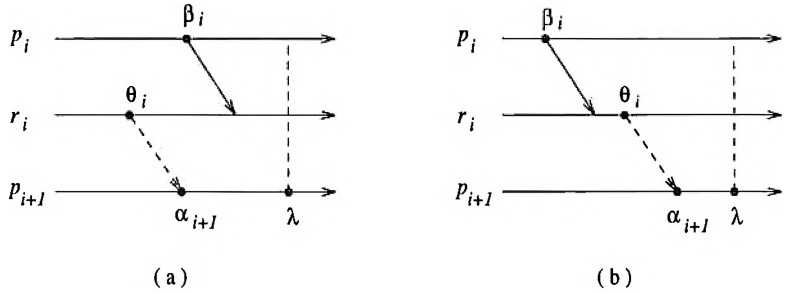

Figure 4: Global time-space diagram for two different cases.

$$\Longrightarrow \exists \alpha_{i+1} \leq \lambda \wedge A(p_{i+1}, r_i, \alpha_{i+1})$$
$$\stackrel{A4}{\Longrightarrow} \exists \theta_i,\ \theta_i < \alpha_{i+1} \leq \lambda \wedge G(p_{i+1}, r_i, \theta_i) \quad \text{(I)}$$

$$\exists \beta_i,\ \forall \beta_i', \beta_i \leq \beta_i' \leq \lambda,\ R(p_i, r_i, \beta_i) \wedge \neg A(p_i, r_i, \beta_i') \quad \text{(II)}$$

p_i may request r_i before or after r_i is granted to p_{i+1}, the following two cases are considered:

Case 1: $\theta_i \leq \beta_i$ (see Figure 4 (a))

$$\begin{aligned}
\text{(I)} \wedge \text{(II)} &\Longrightarrow \exists \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \exists \beta_i\ (\theta_i \leq \beta_i \wedge R(p_i, r_i, \beta_i)) \\
&\stackrel{A5}{\Longrightarrow} \exists \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \exists \beta_i\ (\theta_i \leq \beta_i \wedge \forall \theta_i' \leq \beta_i, \neg G(p_i, r_i, \theta_i')) \\
&\Longrightarrow \exists \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \forall \theta_i' \leq \theta_i, \neg G(p_i, r_i, \theta_i')
\end{aligned}$$

Case 2: $\theta_i > \beta_i$ (see Figure 4 (b))

$$\begin{aligned}
\text{(I)} \wedge \text{(II)} &\Longrightarrow \exists \theta_i, G(p_{i+1}, r_i, \theta_i) \wedge \exists \beta_i, R(p_i, r_i, \beta_i) \wedge \forall \beta_i', \beta_i \leq \beta_i' \leq \lambda, \neg A(p_i, r_i, \beta_i') \wedge \\
&\qquad \beta_i < \theta_i < \lambda \\
&\Longrightarrow \exists \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \exists \beta_i,\ R(p_i, r_i, \beta_i) \wedge \forall \beta_i', \beta_i \leq \beta_i' \leq \theta_i, \neg A(p_i, r_i, \beta_i') \\
&\stackrel{A5}{\Longrightarrow} \exists \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \forall \beta_i' \leq \beta_i, \neg A(p_i, r_i, \beta_i') \wedge \forall \beta_i', \beta_i \leq \beta_i' \leq \theta_i, \neg A(p_i, r_i, \beta_i') \\
&\Longrightarrow \exists \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \forall \beta_i' \leq \theta_i, \neg A(p_i, r_i, \beta_i') \\
&\stackrel{A6}{\Longrightarrow} \exists \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \forall \beta_i' \leq \theta_i, \neg L(p_i, r_i, \beta_i') \\
&\stackrel{A7}{\Longrightarrow} \exists \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \forall \theta_i' \leq \theta_i, \neg G(p_i, r_i, \theta_i')
\end{aligned}$$

By above cases,

$$\begin{aligned}
\text{(I)} \wedge \text{(II)} &\Longrightarrow \exists \theta_i,\ G(p_{i+1}, r_i, \theta_i) \wedge \forall \theta_i' \leq \theta_i, \neg G(p_i, r_i, \theta_i') \\
&\stackrel{A1,2}{\Longrightarrow} \exists c_i = C_{r_i}^{-1}(\theta_i) \wedge G(p_{i+1}, r_i, c_i) \wedge \forall c_i' \leq c_i,\ \neg G(p_i, r_i, c_i') \\
&\Longrightarrow \exists c_i, \forall c_i' \leq c_i,\ G(p_{i+1}, r_i, c_i) \wedge \neg G(p_i, r_i, c_i')
\end{aligned}$$

The local specification for r_i is derived from global specifications for p_i and p_{i+1}. Since $0 \leq i \leq n-1$, we have $D_L(2)$. □

4 Future work

We have presented a formal model of deadlock and developed the deadlock specification. The work in this paper is motivated by the desire to formally verify the correctness of distributed deadlock detection/resolution algorithms. To do this, we may combine the local-time specification of deadlock with the proof technique in [6], and apply this combination to the correctness proof of algorithms.

References

[1] D. A. Menasce and R. R. Muntz, "Locking and deadlock detection in distributed data bases," *IEEE Trans. Software Eng.*, vol. SE-5, pp. 195–202, May 1979.

[2] R. Obermarck, "Distributed deadlock detection algorithm," *ACM Trans. Database Syst.*, vol. 7, pp. 187–208, June 1982.

[3] M. K. Sinha and N. Natarajan, "A priority based distributed deadlock detection algorithm," *IEEE Trans. Software Eng.*, vol. SE-11, pp. 67–80, Jan. 1985.

[4] A. N. Choudhary, W. H. Kohler, J. A. Stankovic, and D. Towsley, "A modified priority based probe algorithm for distributed deadlock detection and resolution," *IEEE Trans. Software Eng.*, vol. SE-15, pp. 10–17, Jan. 1989.

[5] M. Singhal, "Deadlock detection in distributed systems," *IEEE Computer*, vol. 22, pp. 37–48, Nov. 1989.

[6] G. M. Levin and D. Gries, "A proof technique for communicating sequential processes," *Acta Informatica*, vol. 15, pp. 281–302, 1981.