Comput Syst Sci & Eng (2019) 4: 171–178
© 2019 CRL Publishing Ltd

International Journal of
Computer Systems
Science & Engineering

An Upper Bound of Task Loads in a Deadline-d all Busy Period for Multiprocessor Global EDF Real-Time Systems

Fengxiang Zhang*

College of Computer and Information Science, Southwest University, China

This paper addresses a number of mathematical issues related to multiprocessor global EDF platforms. We present a deadline-d all busy period and backward interference which are important concepts for multiprocessor EDF systems, and some general schedulability conditions for any studied job are proposed. We formally prove that at most m-1 different tasks' jobs could contribute their execution time to an interval starting with a P_{busy-d}, and we propose an approach for computing an exact upper bound of the total deadline-d task load in a given interval. Therefore, the proposed results are important foundations for constructing exact schedulability analyses of global EDF scheduling systems.

Keywords: Real-time systems; Multiprocessor scheduling; EDF algorithm; Schedulability analysis

1. INTRODUCTION

Multiprocessor scheduling is known to be an NP-hard optimization problem. Multiprocessor global scheduling where the jobs are allowed to migrate from one processor to another is a much more difficult problem than uniprocessor scheduling. EDF is the most widely studied fixed-job priority scheduling algorithm on multiprocessor platforms. The pessimism of existing global EDF scheduling analysis has a strong impact on the schedulability of small or medium-sized platforms [1], and this problem remains one of the key open questions in the field today [2].

In a real-time system, any job must have sufficient processor time to complete its execution within a given time interval. However, this is not always true when some more urgent jobs occupy all the processors in this interval to prevent a given job getting its required execution time. In such a case, we say that this job and the system are unschedulable. One important role of scheduling analysis of real-time systems is to guarantee that all jobs in a system are schedulable.

In a single processor platform, a processor busy period in which there are always jobs with urgent deadlines executing or waiting to execute is an important foundation for schedulability analysis. However, this idea is not directly applicable to multiprocessor systems since any job with deadline d can always execute as soon as possible when there are no ready jobs with $d_i \leq d$ on any one processor.

In order to study any given job's schedulability on multiprocessor platforms, we present a deadline-d all busy period P_{busy-d} and a backward interference P_{busy-d} which are two important concepts for multiprocessor systems. Based on the definitions of P_{busy-d} and the backward interference P_{busy-d}, two general schedulability conditions for any given job are proposed for EDF global scheduling.

For any particular job, the amount of execution time required by the more urgent jobs within a studied problem time interval strongly influence the given job's schedulability. A number of

*Email: zhangfx@swu.edu.cn

papers have considered this issue for EDF global scheduling (e.g. [3–8]), most of these approaches consider only the interval between the arrival time and the deadline of a studied job (i.e. relative deadline of this job), they consider a worst-case scenario in a small length of time interval thus leading to pessimistic schedulability analysis. Baurah [5] considered an interval starting with a time point such that at least one processor is idle, and he also concluded that the number of the carry-in jobs contributing to this interval could be bounded to $m-1$. However, a formal proof of this property was not provided.

An exact upper bound of the total task load in a studied interval is essential for constructing a schedulability analysis with less pessimism . Therefore, the proposed results provide some theoretical foundations for constructing exact schedulability analysis for multiprocessor EDF systems. The remainder of this paper is organized as follows. In Section 2, the system model used by this paper is described. Several important definitions are presented in Section 3 and, based on these definitions, a general schedulability analysis is proposed. In Section 4, we provide a formal proof that at most $m-1$ different tasks' jobs can have their loads contributions to a P_{busy-d}. In Section 5, a computation of the total deadline-d load of a time interval starting with a backward interference P_{busy-d} is presented. A summary is given in Section 6.

2. SYSTEM MODEL

A real-time system model comprises a set of independent real-time tasks $\{\tau_1, \tau_2, \ldots, \tau_n\}$, and each task consists of either an infinite or finite stream of jobs which must be completed before their absolute deadlines. Each job comprising the same task requires the same worst-case execution time to complete its execution on any processor. Hence any task τ_i (where $i = 1, 2, \ldots, n$) is characterized by a worst-case execution C_i, which must be completed before its relative deadline D_i, and its jobs' minimum inter-arrival time T_i.

We denote j_k to be any given job of a task τ_i. If a job j_k arriving at time t_a, then it must be completed before its absolute deadline at $d = t_a + D_k$, otherwise j_k and the system are unschedulable.

Each job in the system has a priority. The higher priority jobs of j_k are defined to be the jobs that are more urgent than j_k (in the case of a fixed job priority scheme such as EDF) or to be the jobs with priorities higher than j_k (in the case of a fixed task priority scheme).

This paper focuses on the scheduling of real-time tasks in a symmetric multiprocessor system. Let m be the number of processors with unitary capacity in a system. At any time, if there is any idle processor, an arrived ready job with the highest priority is chosen for immediate execution. If there is no idle processor, the arrival of a higher priority job could preempt a lowest priority job's execution at any time. Any job is allowed to migrate from one processor to another one; when a job is returned for execution, it is not necessary to execute it on the same processor.

3. GENERAL SCHEDULABILITY CONDITIONS

This section proposes general schedulability conditions for any given studied job. At the beginning of this section, several important terminologies are introduced for the analysis.

3.1 Definitions and Terminologies

In a single processor platform, a processor busy period is an important concept for schedulability analysis. In a global multiprocessor scheduling system, the busy period is not directly applicable since any job with a deadline equal to d can always be executed as soon as possible when there are no ready jobs with $d_i \leq d$ on any processor. In order to study any given job's schedulability on multiprocessor systems, the following definition is presented.

Definition 1 (deadline-d all busy period) A deadline-d all busy period, denoted by P_{busy-d}, is a consecutive time interval in which there are always ready jobs with absolute deadlines $d_i \leq d$ executing or waiting to execute on each processor.

Note that the character d in Definition 1 could be any specified time point; hence, it could be substituted with ANY time value and notation.

It is obvious that only a deadline-d all busy period could stop the execution of a job with a deadline equal to d and cause this job to miss its deadline.

In a given time interval, if there exists any time point such that any processor is executing a job with $d_k > d$ or it is idle, this interval is not a P_{busy-d}.

Definition 2 (start and end edges of a P_{busy-d}) The start edge of a P_{busy-d} denoted by t_{start}^{edge} is a time point such that $(t_{start}^{edge} - \varepsilon, t_{start}^{edge})$ is not a P_{busy-d}; and the end edge of a P_{busy-d} denoted by t_{end}^{edge} is a time point such that $(t_{end}^{edge}, t_{end}^{edge} + \varepsilon)$ is not a P_{busy-d}; where ε is an is an infinitesimally small positive number.

According to Definition 2, at a time point that is infinitesimally earlier than t_{start}^{edge} or that is infinitesimally posterior to t_{start}^{edge}, there must be at least one processor on which there are no ready jobs with $d_i < d$.

Definition 3 The total length of all the P_{busy-d}s in a time interval $[t_s, d]$ is denoted by $L_{busy-d}^{tot}[t_s, d]$.

When we study the schedulability of a given job j_k with deadline equals d, we have to separate j_k from all other jobs, and consider how much processor time that j_k can have in an interval; hence, the following definition is presented.

Definition 4 The total length of all the P_{busy-d}s without j_k's execution in an interval $[t_s, d]$, denoted by $L_{busy-d}^{tot-j_k}[t_s, d]$, represents the length summation of the intervals in which j_k with a deadline d may not get executed in this interval.

When a job j_k arriving at time t_s with a deadline equals d, the value of $L_{busy-d}^{tot-j_k}[t_s, d]$ determines if j_k can get sufficient processor time in $[t_s, d]$. However, the value of $L_{busy-d}^{tot-j_k}[t_s, d]$ is not only affected by the P_{busy-d}s in $[t_s, d]$, but also affected by a special P_{busy-d} starting before t_s.

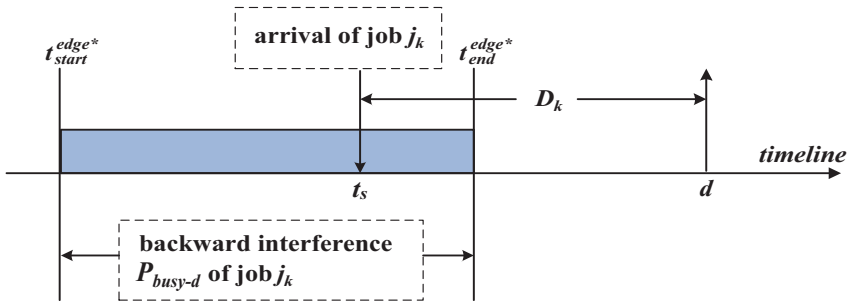

Figure 1 Backward interference P_{busy-d} of a job.

Definition 5 (backward interference P_{busy-d} of a job) When a job j_k arrives at t_s with an absolute deadline $d = t_s + D_k$, then the j_k's backward interference P_{busy-d} is a consecutive P_{busy-d} having a start edge t_{start}^{edge*} before or at t_s and an end edge t_{end}^{edge*} after t_s if such a P_{busy-d} exists, as shown in Figure 1.

We can see that the value of $L_{busy-d}^{tot-j_k}[t_s, d]$ is determined by two parts:

(1) the remainder of backward interference P_{busy-d} of job j_k after t_s; and

(2) the length summation of the P_{busy-d}s having their start edges after or at t_s.

Definition 6 The total deadline-d task load EXCEPT j_k's contribution in a time interval $[t_s, d]$ is denoted by $Load^{tot-j_k}[t_s, d]$, which represents the amount of execution time required by all the jobs having both their arrival times and their deadlines in $[t_s, d]$, plus the amount of unfinished jobs arriving before t_s and having their deadlines before or at d.

3.2 General Schedulability Conditions

As discussed in Section 3.1, for a job j_k arriving at t_s with a deadline d and a worst-case execution time C_k, whether j_k can complete its execution time C_k depends on the value of $L_{busy-d}^{tot-j_k}[t_s, d]$. It is obvious that the following conclusion holds.

Theorem 1 *Any given job j_k arriving at t_s with a deadline d and a worst-case execution time C_k will be schedulable iff*

$$L_{busy-d}^{tot-j_k}[t_s, d] \leq d - t_s - C_k,$$

where $L_{busy-d}^{tot-j_k}[t_s, d]$ is given by Definition 4.

Proof: According to the global EDF scheduling policy, j_k could not get execution only in a deadline-d all busy period, and j_k gets execution immediately in all other cases, therefore, the conclusion has been proven.

However, the length of $[t_s, d]$ equals D_k which is a relatively small number. If we consider all the worst-case situations for j_k's schedulability only in this small interval, this will lead to a pessimistic analysis, as evident in past researches.

To construct a worst-case tasks' arrival pattern and to decrease inaccuracy, we can extend the period $[t_s, d]$ to a longer interval. Denote t_{start}^{edge*} to be the start edge of j_k's backward interference P_{busy-d}, then the studied interval becomes $[t_{start}^{edge*}, d]$.

Theorem 2 *Let t_s be the arrival time of a particular job j_k having a deadline d, and t_{start}^{edge*} be the start edge of j_k's backward interference P_{busy-d} if it exists (as shown in Figure 1). Then j_k is schedulable if and only if*

$$L_{busy-d}^{tot-j_k}[t_{start}^{edge*}, d] \leq d - t_{start}^{edge*} - C_k,$$

where $L_{busy-d}^{tot-j_k}[t_s, d]$ is given by Definition 4.

Proof: Since $[t_{start}^{edge*}, t_s]$ is a deadline-d all busy period and t_s is the arrival time of j_k,

$$L_{busy-d}^{tot-j_k}[t_{start}^{edge*}, t_s] = L_{busy-d}^{tot}[t_{start}^{edge*}, t_s] = t_s - t_{start}^{edge*},$$

therefore, we have:

$$\begin{aligned} L_{busy-d}^{tot-j_k}[t_{start}^{edge*}, d] &= L_{busy-d}^{tot-j_k}[t_s, d] + L_{busy-d}^{tot-j_k}[t_{start}^{edge*}, t_s] \\ &= L_{busy-d}^{tot-j_k}[t_s, d] + t_s - t_{start}^{edge*}. \end{aligned} \quad (1)$$

Since j_k is schedulable if and only if

$$L_{busy-d}^{tot-j_k}[t_s, d] \leq d - t_s - C_k, \quad (2)$$

then by combing equation (1) and inequality (2), the theorem is proven.

By Theorem 2, the schedulability of j_k is determined by the value of $L_{busy-d}^{tot-j_k}[t_{start}^{edge*}, d]$. The following theorem concludes that the maximum possible value of $L_{busy-d}^{tot-j_k}[t_{start}^{edge*}, d]$ is determined by the value of $Load^{tot-j_k}[t_s, d]$ which is given by Definition 6.

Theorem 3 [9] *Let $MaxL_{busy-d}^{tot-j_k}[t_{start}^{edge*}, d]$ be the maximum possible value of $L_{busy-d}^{tot-j_k}[t_{start}^{edge*}, d]$, and let $Load^{tot-j_k}[t_{start}^{edge*}, d]$ be the total deadline-d task load EXCEPT j_k's execution in the interval $[t_{start}^{edge*}, d]$. When $Load^{tot-j_k}[t_{start}^{edge*}, d]$ is a fixed value, we have*

$$MaxL_{busy-d}^{tot-j_k}[t_{start}^{edge*}, d] = \frac{1}{m} Load^{tot-j_k}[t_{start}^{edge*}, d].$$

From the discussions of Theorem 2 and Theorem 3, the schedulability of any given job j_k's schedulability is determined by the value of $Load^{tot-j_k}[t_{start}^{edge*}, d]$. Therefore, an exact upper bound of $Load^{tot-j_k}[t_{start}^{edge*}, d]$ is important for constructing an exact EDF global schedulability analysis.

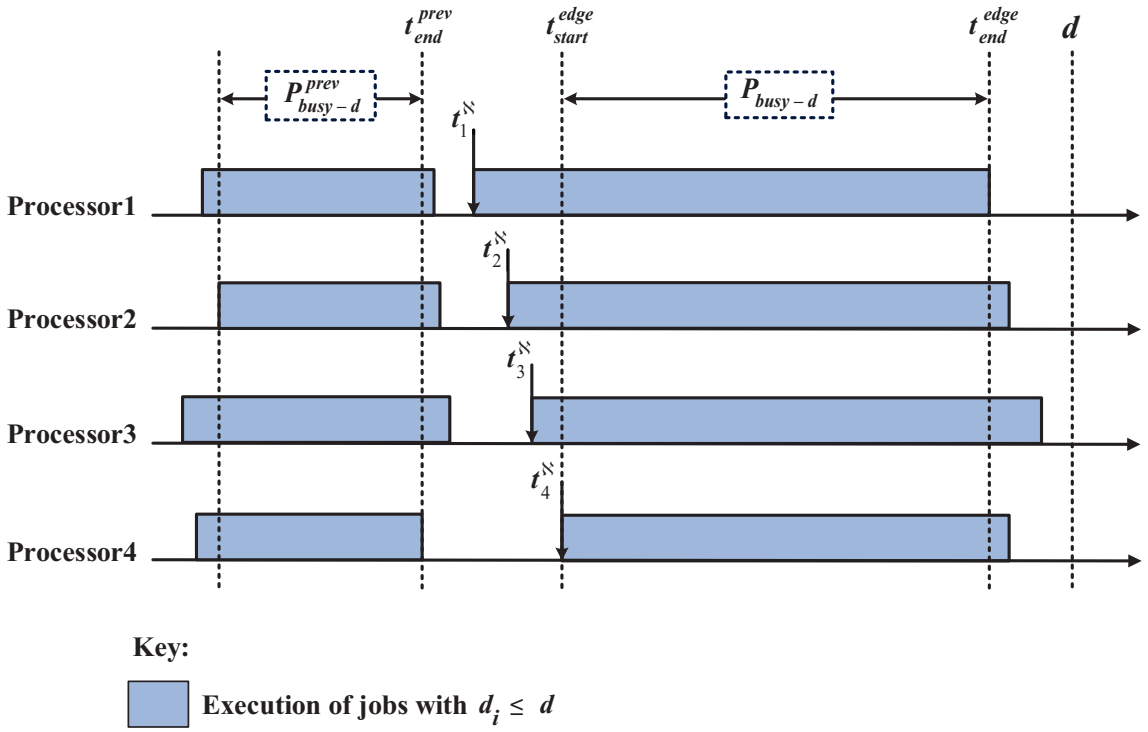

Figure 2 An illustrative example of Case one of Lemma 2's proof when there are four processors in a system.

4. BOUNDING THE VALUE OF DEADLINE-D TASK LOADS

According to Definition 6, any jobs with $d_i \leq d$ arriving before any given time point t_s could contribute their loads to the value of $Load^{tot-j_k}[t_s, d]$ if they have not completed by t_s, and on each processor, there could be more than one such job being executed or waiting to be executed before t_s. This assumption produces too much deadline-d task load in a particular problem interval, and thus leads to a pessimistic schedulability. The following lemmas are needed to bound the value of $Load^{tot-j_k}[t_{start}^{edge*}, d]$.

Lemma 1 Let P_{busy-d}^{prev} be the closest deadline-d all busy period before any given P_{busy-d}, and denote t_{end}^{prev} to be the end edge of P_{busy-d}^{prev}, as shown in Figure 3. Then at any time during the interval $(t_{end}^{prev}, t_{start}^{edge})$, there are no ready jobs with $d_i \leq d$ waiting to execute in the system.

Proof: Since there are no other deadline-d busy periods between P_{busy-d}^{prev} and P_{busy-d}, at any time in the interval $(t_{end}^{prev}, t_{start}^{edge})$, there is at least one processor which is idle or is executing some job with $d_i > d$, therefore, any job with $d_i \leq d$ arrives in $(t_{end}^{prev}, t_m^{\aleph})$ can be executed immediately and there are no ready jobs waiting to execute.

Lemma 2 On any given processor P_κ, let $t_\kappa^{\aleph}$ be the last time before or at t_{start}^{edge} of a P_{busy-d} such that there are no jobs with $d_i \leq d$,where $\kappa \in \{1, 2, \ldots, m\}$. Denote t_{end}^{prev} to be the end edge of P_{busy-d}^{prev} which is the closest deadline-d all busy period before P_{busy-d}. As illustrated in Figure 2 and Figure 3. Then on processor P_κ:

If $t_{end}^{prev} < t_\kappa^{\aleph} < t_{start}^{edge}$, there is only one job with $d_i \leq d$ executing in the time interval $(t_\kappa^{\aleph}, t_{start}^{edge})$.

If $t_\kappa^{\aleph} \leq t_{end}^{prev}$, there is only one job with $d_i \leq d$ executing in the interval $(t_{end}^{prev}, t_{start}^{edge})$.

Proof: There are two cases to consider:

Case one: for $\forall\kappa \in \{1, 2, \ldots, m\}$, $t_{end}^{prev} < t_\kappa^{\aleph} \leq t_{start}^{edge}$, as illustrated in Figure 2. Then by Lemma 1, any job with $d_i \leq d$ arriving in $(t_{end}^{prev}, t_{start}^{edge})$ can get executed immediately; hence, no job in $(t_{end}^{prev}, t_{start}^{edge})$ will be preempted until it completes or extends its execution to t_{start}^{edge}. Therefore, for any $\kappa \in \{1, \ldots, m\}$ and $t_\kappa^{\aleph} \leq t_{start}^{edge}$: $t_\kappa^{\aleph}$ is the arrival time of a job $j_\kappa^{\aleph}$ with $d_i \leq d$, $j_\kappa^{\aleph}$ will not finish before t_{start}^{edge} (or else $t_\kappa^{\aleph}$ will not be the last time before t_{start}^{edge} such that there are no ready jobs with $d_i \leq d$ on P_κ), and $j_\kappa^{\aleph}$ is the only job executing in $(t_\kappa^{\aleph}, t_{start}^{edge})$ on P_κ.

Case two: there exists $\kappa \in \{1, \ldots, m\}$ such that $t_\kappa^{\aleph} \leq t_{end}^{prev}$, as illustrated in Figure 3, then there are always ready jobs with $d_i \leq d$ in $(t_\kappa^{\aleph}, t_{start}^{edge})$ on processor P_κ. There must be a job $j_\kappa^{\aleph}$ with $d_i \leq d$ arriving before or at t_{end}^{prev} and executing in $(t_{end}^{prev}, t_{start}^{edge})$, the same as for case one, $j_\kappa^{\aleph}$ will not finish or be preempted before t_{start}^{edge}, and $j_\kappa^{\aleph}$ is the only one job with $d_i \leq d$ executing in $(t_{end}^{prev}, t_{start}^{edge})$ on P_κ.

The deadline-d task load of an interval starting with a P_{busy-d} can be bounded by the following theorems.

Theorem 4 *On each processor, at most one task arriving before t_{start}^{edge} of a P_{busy-d} could contribute its load to the deadline-d task load of $[t_{start}^{edge}, t_\psi]$, where $\forall t_\psi \in (t_{start}^{edge}, d]$.*

Proof: Let $t_i^{\aleph}$ be the last time at or before t_{start}^{edge} such that there are no ready jobs with $d_i \leq d$ on a given processor P_i, and t_{end}^{prev}

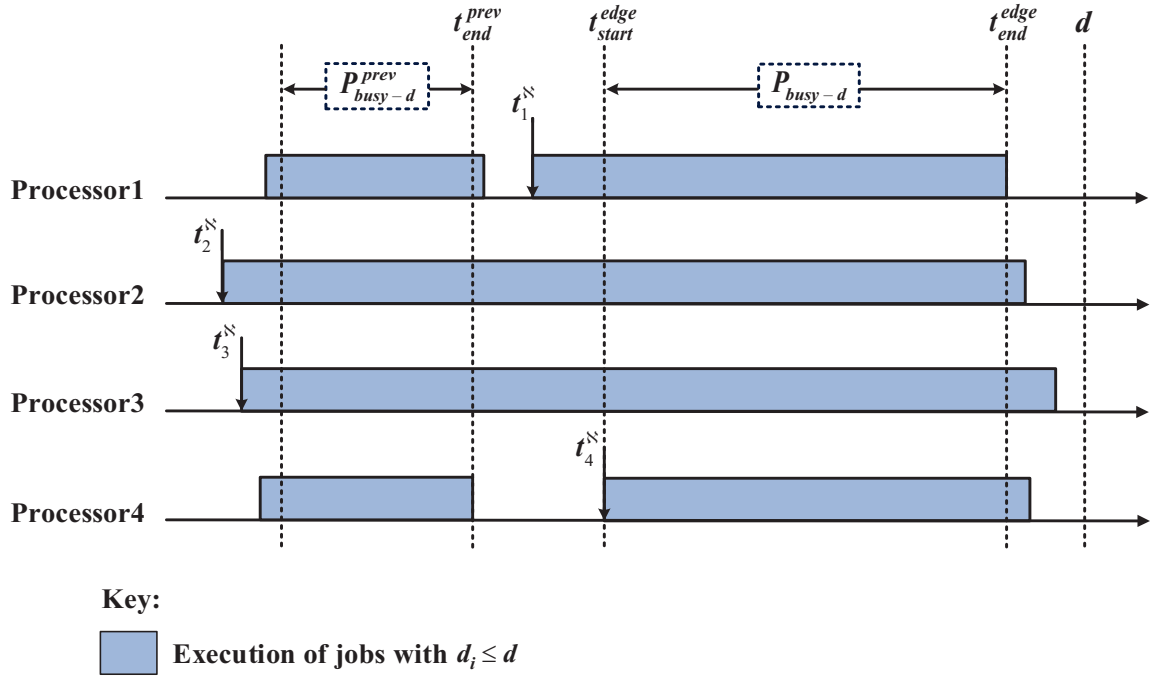

Figure 3 An illustrative example of Case two of Lemma 2's proof when there are four processors in a system.

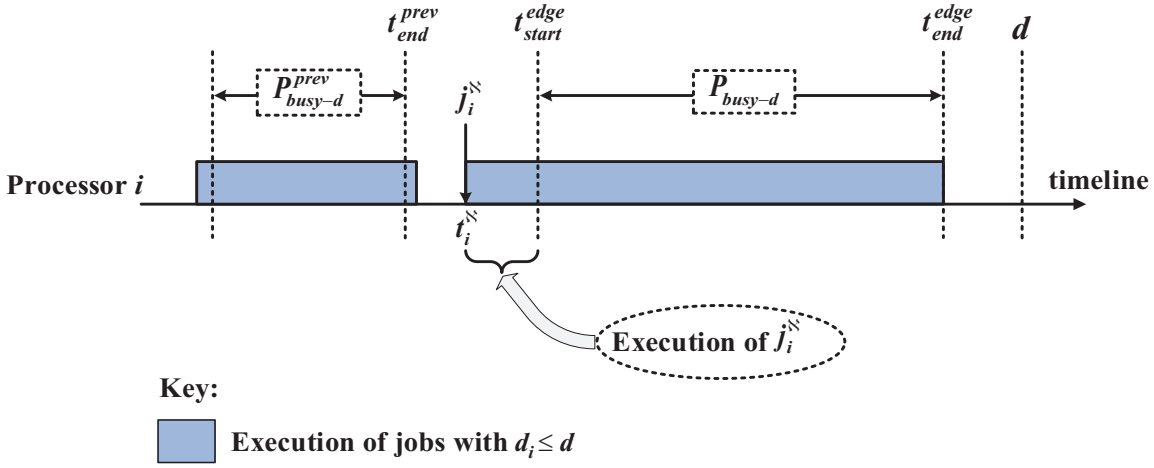

Figure 4 Case when $t_{end}^{prev} < t_i^{\aleph} < t_{start}^{edge}$ in Theorem 4's proof.

be the end edge of the closest deadline-d all busy period before t_{start}^{edge}. There are two cases.

If $t_{end}^{prev} < t_i^{\aleph} < t_{start}^{edge}$ (as shown in Figure 4): by Lemma 2, there is only one job $j_i^{\aleph}$ executing in $(t_i^{\aleph}, t_{start}^{edge})$ on processor P_i, and $t_i^{\aleph}$ must be the arrival time of $j_i^{\aleph}$, from Lemma 1, there are no ready jobs waiting to execute; therefore, $j_i^{\aleph}$ is the only job which arrives before t_{start}^{edge} and contributes its execution time to the deadline-d load of $[t_{start}^{edge}, t_\psi]$.

If $t_i^{\aleph} \le t_{end}^{prev}$: from Lemma 2's discussions, there must be a job $j_i^{\Re}$ arrives before or at t_{end}^{prev} and extends its execution to t_{start}^{edge}, and $j_i^{\Re}$ is the only job executes during $(t_{end}^{prev}, t_{start}^{edge})$ on processor P_i, since there are also no ready jobs with $d_i \le d$ waiting to execute during $(t_{end}^{prev}, t_{start}^{edge})$, $j_i^{\Re}$ is the only job contributing to the dead-line-d load of $[t_{start}^{edge}, t_\psi]$.

Theorem 5 *At most $m-1$ different tasks' jobs arriving before t_{start}^{edge} of a P_{busy-d} could contribute their task loads to the total deadline-d task load of $[t_{start}^{edge}, t_\psi]$, where $\forall t_\psi \in (t_{start}^{edge}, d]$.*

Proof: By Definition 2, t_{start}^{edge} must be the arrival time of a job with $d_i \le d$ on a given processor P_κ, and any other job arriving before t_{start}^{edge} on P_κ must have completed before t_{start}^{edge}, hence no job arriving before t_{start}^{edge} on P_κ contributes its execution to $[t_{start}^{edge}, t_\psi]$. Theorem 4 has thus been proven.

Corollary 1 Let t_{start}^{edge*} to be the start edge of a studied job j_k's backward interference P_{busy-d}, then at most $m-1$ tasks' jobs arriving before t_{start}^{edge*} could contribute their task loads to $Load^{tot-j_k}[t_{start}^{edge*}, d]$, where $Load^{tot-j_k}[t_{start}^{edge*}, d]$ is given by definition 4.

Proof: Since t_{start}^{edge*} is a start edge of a P_{busy-d} and $d \in (t_{start}^{edge}, \mathrm{d}]$, this result is obtained directly from Theorem 5's conclusion.

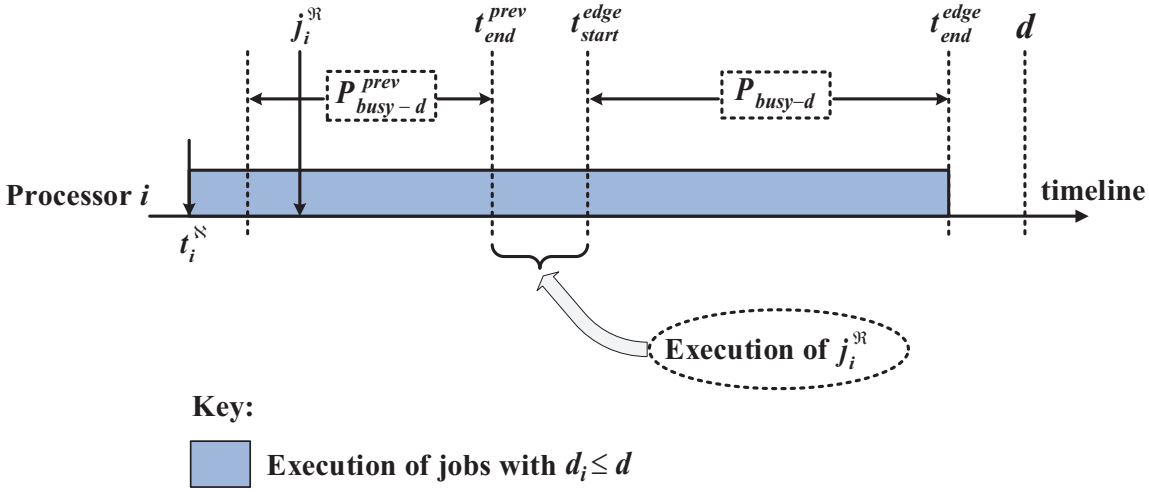

Figure 5 Case when $t_i^{\aleph} \leq t_{end}^{prev}$ in Theorem 4's proof.

5. COMPUTATIONS TO THE MAXIMUM VALUE OF $Load^{tot-j_k}[t_{start}^{edge*}, d]$

This section calculates maximum value of $Load^{tot-j_k}[t_{start}^{edge*}, d]$. There are two categories of the jobs contributing to the value of $Load^{tot-j_k}[t_{start}^{edge*}, d]$ to consider, they are

(A) The jobs with both their arrival times and deadlines in $[t_{start}^{edge*}, d]$.

(B) The jobs with their arrival times before t_{start}^{edge*} and deadlines in $[t_{start}^{edge*}, d]$.

In this section, we assume that all the jobs in Category (B) can complete their executions before their deadlines, or else the deadline-d task load in $[t_{start}^{edge*}, d]$ cannot be bounded.

According to Corollary 1's conclusion, the number of the jobs in Category (B) is bounded by $m-1$, and they are from different tasks. Therefore, we only need to calculate $m-1$ different tasks' jobs with the greatest contributions under a given tasks' arrival pattern.

However, we still do not know the worst-case tasks' arrival pattern leading to the maximum deadline-d load to the interval $[t_{start}^{edge*}, d]$, the following lemma proves a worst-case arrival pattern ofany given task τ_i in an interval.

Lemma 3 For any task τ_k, it contributes the maximum possible deadline-d load in any given fixed length of interval $[t_s, d]$ under the arrival pattern when:

(1) one of τ_i's absolute deadlines is coincident with d; and

(2) τ_i's arrival always keep the maximum rate.

Proof: Suppose $j_{i,\kappa}$ is one of task τ_i's jobs, and $j_{i,\kappa}$'s deadline is coincident with d. On the timeline, there are two cases to consider:

1) If we move 'right' all τ_i's jobs to let them arrive later and at their minimum inter-arrival time, then $j_{i,\kappa}$'s absolute deadline will be greater than d; hence, the deadline-d load of $[t_s, d]$ will be decreased by C_i. Although the unfinished execution time of τ_i's last job before t_s may be increased by C_i^{Δ}, obviously $C_i^{\Delta} \leq C_i$.

2) If we move 'left' all τ_i's jobs to let them arrive earlier, this could decrease the unfinished execution time of τ_i's last job before t_s, therefore, the deadline-d load in $[t_s, d]$ could only be decreased.

The discussions above support our conclusion.

The value of $Load^{tot-j_k}[t_{start}^{edge*}, d]$ takes the maximum when all tasks in the system satisfy Lemma 3's pattern. Therefore, for any given task τ_i, when its deadline is coincident with d, let job_{prev}^i be the last job of τ_i arriving at or before t_{start}^{edge*}, and let t_{arri}^i be the arrival time of job_{prev}^i, as shown in Figure 6. If t_{arri}^i is also coincident with t_{start}^{edge*} (i.e. the relative deadline of τ_i is an integral multiple of $d - t_{start}^{edge*}$), then $t_{arri}^i = t_{start}^{edge*}$.

Let $L_{prev}(\tau_i)$ be the distance between t_{arri}^i and t_{start}^{edge*}, we have

$$L_{prev}(\tau_i) = \left\lceil \frac{d - t_{start}^{edge*}}{T_i} \right\rceil T_i - \left(d - t_{start}^{edge*}\right), \qquad (3)$$

when $L_{prev} \neq 0$, since job_{prev}^i completes its execution before its deadline at $t_{arri}^i + D_i$, the unfinished execution time of job_{prev}^i at time t_{start}^{edge*} is at most

$$\min\left(D_i - L_{prev}(\tau_i), C_i\right),$$

and this value must be positive, therefore, when $L_{prev} \neq 0$, the maximum deadline-d load contributed by job_{prev}^i of τ_i is

$$Prev(\tau_i) = \max\left(\min\left(D_i - L_{prev}(\tau_i), C_i\right), 0\right).$$

It is obvious that when $L_{prev} = 0$, $Prev(\tau_i) = 0$, therefore, we have

$$Prev(\tau_i) = \begin{cases} max\left(min\left(D_i - L_{prev}(\tau_i), C_i\right), 0\right), & when\ L_{\text{prev}}(\tau_i) \neq 0 \\ 0, & when\ L_{prev}(\tau_i) = 0 \end{cases} \qquad (4)$$

where $L_{prev}(\tau_i)$ is given by equation (3).

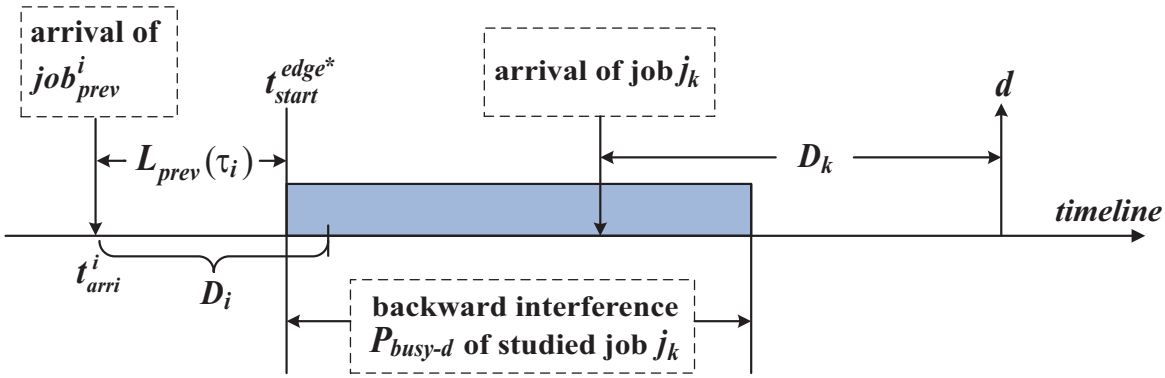

Figure 6 Computations to the maximum value of $Load^{tot-j_k}[t^{edge*}_{start}, d]$.

As there are at most $m-1$ different tasks' jobs in Category (B), only the maximum $m-1$ tasks need to be considered. Let $xi \in \{1, 2, \ldots, n\}$ and

$$Prev(\tau_{x1}) \leq Prev(\tau_{x2}) \leq \ldots \leq Prev(\tau_{xn}). \quad (5)$$

Denote $Extf[t^{edge*}_{start}, d]$ to be the maximum value of the load produced by the carry-in jobs of Category (B) in$[t^{edge*}_{start}, d]$, then $Extf[t^{edge*}_{start}, d]$ is calculated by

$$Extf[t^{edge*}_{start}, d] = \sum_{i=1}^{m-1} Prev(\tau_{xi}) \quad (6)$$

where $Prev(\tau_{xi})$ is defined by equation (4) and inequality (5).

The amount of the deadline-d load produced by the jobs in Category (A) can be calculated using the traditional processor demand bound function [10] for uniprocessor systems

$$Dbf[t^{edge*}_{start}, d] = \sum_{i=1}^{n} \max\left(1 + \left\lfloor \frac{d - t^{edge*}_{start} - D_i}{T_i} \right\rfloor, 0\right) C_i. \quad (7)$$

Therefore, the total deadline-d task load of $[t^{edge*}_{start}, d]$ is bounded by

$$Load^{tot}[t^{edge*}_{start}, d] = Extf[t^{edge*}_{start}, d] + Dbf[t^{edge*}_{start}, d],$$

and let a given job j_k's execution time be subtracted from $Load^{tot}[t^{edge*}_{start}, d]$, then the maximum value of $Load^{tot-j_k}[t^{edge*}_{start}, d]$ is calculated by

$$Load^{tot-j_k}[t^{edge*}_{start}, d] = Extf[t^{edge*}_{start}, d] + Dbf[t^{edge*}_{start}, d] - C_k,$$

where $Extf[t^{edge*}_{start}, d]$ is given by equation (6), and $Dbf[t^{edge*}_{start}, d]$ is given by equation (7).

6. CONCLUSION

In this paper, we address a number of problems for schedulability analysis of multiprocessor EDF global scheduling. The deadline-d all busy period P_{busy-d} and the backward interference P_{busy-d} are important concepts firstly presented for analyzing a particular job's schedulability, and some general conditions for schedulability are presented based on these concepts. We consider an interval starting with a backward interference P_{busy-d} for a job's schedulability; we formally prove that the total deadline-d task load in such an interval can be bounded to $m-1$ different tasks' carry-in jobs; we present the relationship between the maximum value of the total length of all the P_{busy-d}s and the total deadline-d task load in a particular interval.

We provide approaches for calculating an exact upper bound of the total deadline-d task load in a time interval starting with a backward interference P_{busy-d}; this value determines the maximum executable time of a studied job with a deadline d in this interval. Therefore, the proposed results are important theoretical foundations that can be used to build exact schedulability analyses for multiprocessor EDF global scheduling platforms.

7. ACKNOWLEDGEMENTS

This work is supported by the Fundamental Research Funds for the Central Universities under the grant XDJK2019B025.

REFERENCES

1. R.I. Davis and A. Burns, "A Survey of Hard Real-Time Scheduling for Multiprocessor Systems", *ACM Computing Surveys*. 43(4): Article 35, 2011.
2. T.P. Baker and S.K. Sanjoy, "Schedulabiliy Analysis of Multiprocessor Sporadic Task Systems", in *In Handbook of Real-Time and Embedded Systems*. 2007, Chapman & Hall/CRC.
3. T.P. Baker, "Multiprocessor EDF and Deadline Monotonic Schedulability Analysis", *Proceedings 24th IEEE Real-Time Systems Symposium*, pp. 120–129, 2003.
4. T.P. Baker, "An Analysis of EDF Scheduling on a Multiprocessor", *Real-time Systems*. 32(1–2): 49–71, 2006.
5. S.K. Baruah, "Techniques for Multiprocessor Global Schedulability Analysis", *Proceedings 28th IEEE Real-Time Systems Symposium*, pp. 119–128, 2007.
6. M. Bertogna, M. Cirinei, and G. Lipari, "Improved Schedulability Analysis of EDF on Multiprocessor Platforms", *Proceedings of the 17th Euromicro Conference on Real-Time Systems*, pp. 209–218, 2005.

7. M. Bertogna, M. Cirinei, and G. Lipari, “Schedulability Analysis of Global Scheduling Algorithms on Multiprocessor Platforms”, *IEEE Transactions on Parallel and Distributed Systems*. 20(4): 553–566, 2008.
8. N. Guan, M. Stigge, W. Yi, and G. Yu, “New Reponse Time Bounds for Fixed Priority Multiprocessor Scheduling”, *Proceedings 30th IEEE Real-Time Systems Symposium*, pp. 387–397, 2009.
9. F. Zhang and A. Burns, “A Worst-Case Pattern of Task Load Allocation and Execution for Multiprocessor Global Real-Time Scheduling”, *International Journal of Simulation – Systems, Science & Technology*, Vol. 17, Issue 17, Pages 9.1–9.4, 2016.
10. S.K. Baruah, A.K. Mok, and L.E. Rosier, “Preemptively Scheduling Hard-Real-Time Sporadic Tasks on One Processor”, *Proceedings 11th IEEE Real-Time System Symposium*, pp. 182–190, 1990.

Fengxiang Zhang received his Ph.D. degree in Computer Science from the University of York (UK) in 2009. He is currently an associate professor atthe College of Computer and Information Science, Southwest University, China. His main research interests are in the areas of scheduling analysis of real-time and embedded systems.