# 嵌入式实时操作系统的容错调度算法研究*

臧义华, 闫保中, 马龙华

(哈尔滨工程大学自动化学院, 哈尔滨　150001)

[摘要]　基于时间冗余的实时系统调度算法, 要求一定的时间冗余。采用非精确计算技术, 将实时任务分为强制执行部分和可选执行部分, 将可选执行部分的处理部分作为强制执行部分的容错操作预留处理时间。非精确容错最早时限优先调度算法, 通过降低为容错操作专门预留的空闲处理器时间, 增加了可用于实时任务正常运行的处理器时间, 提高系统的任务吞吐量和资源利用率。

[关键词]　容错; 非精确计算; 嵌入式实时操作系统; 最早时限优先调度算法

[中图分类号]　TP302.8　　[文献标志码]　A

# The Fault-tolerant Scheduling Algorithmic Research of Embedded Real-time Operating System

ZANG Yi-hua, YAN Bao-zhong, MA Long-hua

(College of Automation, Harbin Engineering University, Harbin 150001, China)

**Abstract**: Fault-tolerant real-time scheduling algorithmic though time redundancy must have some time redundancy. The imprecise computation technology divides the real time tasks into compulsive and optional execution parts, so the time of optional execute part can turn into the obligate time of the compulsive part. The algorithm makes use of the optional execution parts time reducing the obligate time, increasing the processor time on the rails and improving tasks throughput and resource utilization.

**Key words**: fault-tolerant; imprecise computation; embedded real-time operating system; pioneer time limit first algorithm

## 1　引言

嵌入式实时系统 VxWorks 是军方指定的操作系统。现在, 越来越多的安全关键系统已经从过去依赖于人工或高可靠的模拟系统转向依靠实时计算机系统。在这种情况下, 可靠性就成为整个应用系统可靠性的决定因素。嵌入式系统的可用资源有限, 所以有必要研究冗余资源较少的容错调度算法。

## 2　容错实时调度

时间冗余为实时系统容错提供了物质基础, 若不能正确管理这些资源, 则时间冗余难以达到预期效果[1]。如图 1(a)中所示出错任务(或其替代任务)必须在出错任务的时限到来前获得足够的冗余时间完成再次运行(或运行)。如果以上情况得不到满足, 尽管预留的冗余时间大于出错任务所需的执行时间, 但容错操作将超过出错任务的时限, 任务最终的运行结果仍然是错误的[3]。

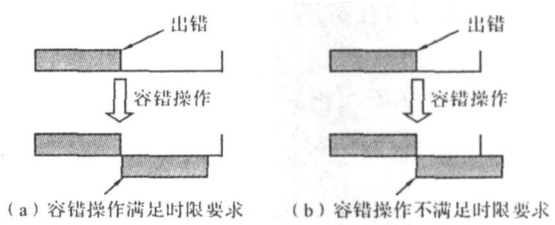


图 1　成功的时间冗余与失败的时间冗余

因此, 在改变基本算法的可调度条件的同时, 修改它的资源分配策略也是必要的。基本算法按任务属性确定任务的优先级, 而决定任务的执行顺序。容错实时调度算法在确定正常情况下任务执行序列的同时, 还需要在任务的正常执

* 收稿日期: 2006-08-25;　　修回日期: 2006-12-25
作者简介: 臧义华(1980-), 女, 辽宁抚顺人, 硕士研究生, 研究方向: 导航制导与控制。

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

行序列中合理地分布空闲时间供任务出错后的容错操作使用。

实时系统的处理器资源由实时调度算法管理。普通的实时调度算法只考虑任务正常工作时的处理器时间和时限需求。容错实时调度算法,在拥有一定量空闲的处理器资源的前提下,不仅要满足任务正常运行时的处理器时间和时限需求,还必须满足 $n$ 次容错操作的处理器时间和时限需求,且容错操作不影响其他任务的正常运行。

## 3 单处理器动态硬实时调度算法

Liu 和 Layland 提出的时限驱动调度算法,也称为最早时限优先调度(EDF)。这是一种动态可抢占优先级实时调度,实时任务集的模型可以表述如下:存在任务集 $S(\tau_1, \tau_2, \cdots, \tau_n)$,所有实时任务必须满足以下限制条件:

(1) 所有实时任务均为周期任务,且周期大于或等于时限;

(2) 所有实时任务必须在其时限到来前结束;

(3) 所有实时任务相互独立;

(4) 所有实时任务均具有恒定的运行时间。

任务按周期由小到大排列为 $T_1 \leqslant T_2 \leqslant \cdots \leqslant T_n$。EDF 以任务的时限与当前时刻的距离确定任务优先级(将这一距离称为时限距离),距离越近,优先级越高。因此,EDF 总是选择当前最迫切需要完成的任务获得处理器。Liu 和 Layland 证明如下定理:

定理 1 对于由 $n$ 个周期任务组成实时任务集,当且仅当:

$$\left(\frac{C_1}{T_1}\right)+\left(\frac{C_2}{T_2}\right)+\cdots+\left(\frac{C_n}{T_n}\right)\leqslant 1$$

该任务集能够由 EDF 调度。其中,$C$ 为最坏执行时间。

## 4 基本容错

### 4.1 基本 EDF 容错

对于满足调度条件的实时任务集,EDF 算法按任务的优先级分配处理器资源,保证所有任务均满足时限要求[2]。显然,若调度算法能够保证实时任务 $\tau_i$ 的第 $k$ 个实例在 $(k+1)T_i - C_i$ 前结束,则该实例能够在不影响其他任务执行的前提下再实现一次重新执行。

定理 2 对于由 $n$ 个任务周期组成实时任务集 $S$,当且仅当:

$$\left(\frac{C_1}{(T_1-C_1)}\right)+\left(\frac{C_2}{(T_2-C_2)}\right)+\cdots+\left(\frac{C_n}{(T_n-C_n)}\right)\leqslant 1$$

实时任务集 $S$ 可由基本的容错 EDF,且每个任务实例均能在不影响其他任务正常运行的前提下,实现一次容错操作。显然,基本容错 EDF 适用于资源利用率低、任务出错概率高的实时任务集调度。

### 4.2 容错 EDF

与基本的容错 EDF 不同,容错 EDF(以下简称 FT-EDF)的冗余处理器资源由所有任务共享,适用于资源利用率高,出错概率低的实时应用。容错 EDF 的设计目标是,在 $T_n + T_{n-1}$ 内,冗余的处理器资源保证一次容错操作,且所有任务均满足时限要求。因此,在两个连续的最大周期边界 $[kT_n, (k+1)T_n]$ 内,冗余的处理器资源为 $\max(C_i), i \in S$。

容错 EDF 按照任务的时限先后决定任务的优先级,时限近者优先级高。系统运行时,若当前执行的任务 $\tau_i$ 出错,调度器调度该任务再次运行。在容错处理的过程中,若新就绪的任务 $\tau$ 的优先级高于 $\tau_i$,则 $\tau$ 抢占 $\tau_i$。容错 EDF 的设计目标是,$T_n + T_{n-1}$ 内,冗余的处理器资源保证一次容错操作,且所有任务均满足时限要求,其可调度条件由以下定理表述。

定理 3 若实时任务集 $S$ 满足:

$$F_{\text{FT-EDF}}(S)=\left(\frac{\sum_{i=1}^{n-1}\left(\left[\frac{T_n}{T_i}\right]\cdot(C_i)\right)+\max_{i\in s}(C_i)+C_n}{T_n}\right)\leqslant 1$$

则必可由 FT-EDF 调度。其中,$F_{\text{FT-EDF}}(S)$ 称为 FT-EDF 算法的可调度性估计函数。

定理 4 实时任务集 $S$ 可由容错 EDF 调度,当相邻两个错误发生的时间间隔大于 $T_n + T_{n-1}$,则预留的空闲处理器资源可以满足任何任务的

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

容错要求。

容错 EDF 通过调度器实现任务间空闲资源的共享和动态分配，调度算法较为简单，同时保证系统正常工作时具有较高的资源利用率。FT-EDF 适用于高可靠、低故障率的实时系统。

## 5　基于非精确计算的 FT-EDF

尽管可选执行部分拥有的处理器资源不能完全满足容错操作的时间需求，但通过降低为容错操作专门预留的空闲处理器时间的数量，增加了可用于实时任务正常运行的处理器时间，提高系统的任务吞吐量。这一改进算法被称为非精确容错 EDF 调度(IC-FT-EDF)。

### 5.1　系统模型

基于非精确计算的容错 EDF 的实时任务集的模型可以表述如下：存在任务集，$S(\tau_1, \tau_2, \cdots, \tau_n)$，所有硬实时任务必须满足以下限制条件：

(1) 所有实时任务均为周期任务，且周期大于或等于时限；

(2) 所有实时任务必须在其时限到来前结束；

(3) 所有实时任务相互独立；

(4) 所有实时任务均具有恒定的运行时间。

任务按周期由小到大排列为 $T_1 \leqslant T_2 \leqslant \cdots \leqslant T_n$。任务 $\tau_i$ 的第 $k$ 个实例在 $k\tau_i$ 时就绪，在 $(k+1)\tau_i$ 时必须结束。基于非精确计算的容错 EDF 以任务的时限与当前时刻的距离确定任务优先级，距离越近，优先级越高。IC-FT-EDF 将任务 $\tau_i$ 分为强制执行部分 $M_i$ 和可选执行部分 $O_i$，$m_i$ 和 $o_i$ 分别为 $M_i$ 和 $O_i$ 的执行时间，$C_i = m_i + o_i$，$O_i$ 只能在 $M_i$ 完成后开始。$O_i$ 的执行按 0/1 限制决定，即只能选择完全执行和完全放弃两种方式。

任务的错误检测在 $M_i$ 和 $O_i$ 运行结束时进行，其开销分别计入 $m_i$ 和 $o_i$。如果错误出现在 $M_i$ 执行阶段，则任务将重新执行。如果错误出现在 $O_i$ 执行阶段，因为 $M_i$ 已经满足系统对 $\tau_i$ 的最基本需要，为避免重新执行 $O_i$ 而影响其他任务，$O_i$ 执行结果将被放弃。

### 5.2　可调度条件及算法正确性

基于对 IC-FT-EDF 容错机制的分析可以看出，可调度条件是该算法的关键所在，IC-FT-EDF 的容错目标是时间冗余策略和调度策略，在规定时间内保证规定次数的容错操作满足时限[4]。文中将仅考虑在规定时间内单个暂态错误的容错操作满足时限要求。给出错误发生的最小时间间隔，即若实时任务集满足 IC-FT-EDF 的可调度条件，且任意两个错误发生的时间间隔大于某一常数，则预留的空闲处理器资源可以满足任何任务的容错要求。

若实时任务集 $S$ 满足：

$$F_{\text{IC-FT-EDF}}(s) = \left( \frac{\sum_{i=1}^{i-1} \left( \left[ \frac{T_n}{T_i} \right] \cdot (M_i + O_i) \right) + \max_{i \in s} (M_i - O_i) + (M_n + O_n)}{T_n} \right) \leqslant 1$$

则 $S$ 必可由 IC-FT-EDF 调度。其中，$F_{\text{IC-FT-EDF}}(S)$ 称为 IC-FT-EDF 算法的可调度性估计函数。

对于单处理器实时系统，有以下两个结论：

(1) 从临界时刻(所有任务同时就绪) 开始，调度器只要能满足每个任务的第一个时限要求，则能够满足其以后的所有时限要求。

(2) 同时，EDF 算法在任务运行超出时限前不可能存在空闲时间，所以只要任务集的资源利用率小于等于 1，则必能由 EDF 调度。

因此，为满足条件 1，只需要考虑优先级高于出错任务的任务对出错任务的影响。从临界时刻起，出错任务 $\tau_r$ 完成再次运行后，处理器运行时间与 $T_r$ 的比值为：

$$\frac{\sum_{i=1}^{r-1} \left( \left[ \frac{T_r}{T_i} \right] \cdot (M_i + O_i) \right) + 2M_r}{T_r}$$

因此，任务集 $S$ 若满足以下条件，则必满足条件 1。

$$\max_{r \in s} \left( \frac{\sum_{i=1}^{r-1} \left( \left[ \frac{T_r}{T_i} \right] \cdot (M_i + O_i) \right) + 2M_r}{T_r} \right) \leqslant 1$$

同理，为满足条件 2，只需考虑在比当前任务优先级高的任务中，执行时间最长的任务进行过容错操作。从临界时刻起，当前任务 $\tau_i$ 完成后，处理器运行时间与 $T_i$ 的比值为：

$$\frac{\sum_{k=1}^{i-1} \left( \left[ \frac{T_i}{T_k} \right] \cdot (M_k + O_k) \right) + \max_{m \in [1, i-1]} (M_m - O_m) + (M_i + O_i)}{T_i}$$

因此，任务集 $S$ 若满足以下条件，则必满足条件 2。

$$\max_{i \in s} \left( \frac{\sum_{k=1}^{i-1} \left( \left[ \frac{T_i}{T_k} \right] \cdot (M_k + O_k) \right) + \max_{m \in [1, i-1]} (M_m - O_m) + (M_i + O_i)}{T_i} \right) \leqslant 1$$

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

取表达式的最大值。得:

$$F_{IC-FT-EDF}(S) = \left( \frac{\sum_{i=1}^{n-1}\left(\left[\frac{T_n}{T_i}\right]\cdot(M_i+O_i)\right)+\max_{i\in s}(M_i-O_i)+(M_n+O_n)}{T_n} \right) \leqslant 1$$

则,实时任务集 $S$ 满足上式,且必可由 IC-FT-EDF 调度。

实时任务集 $S$ 可由 IC-FT-EDF 调度,当相邻两个错误发生的时间间隔大于 $T_n + T_{n-1}$,则预留的空闲处理器资源可以满足任何任务的容错要求。

证明:假设在 $T_i$ 范围内,任务 $\tau_i$ 无法实现容错操作。由于实时任务集 $S$ 可由 IC-FT-EDF 调度,$\tau_i$ 可以实现容错操作,除非其所需的处理器资源已被 $\tau_j$ 执行容错操作而占用。因此,$T_i$ 必定与 $T_j$ 相交,则两个相邻错误发生的时间间隔不大于 $T_i + T_j$。

因为 $T_i + T_j \leqslant T_i + T_{n-1}$,这与命题中要求两个相邻错误发生的时间间隔大于 $T_{n-1} + T_n$ 矛盾。所以,实时任务集 $S$ 可由 IC-FT-EDF 调度,当相邻两个错误发生的时间间隔大于 $T_{n-1} + T_n$,则预留的空闲处理器资源可以满足任何任务的容错要求。

由此可知,只要实时任务集可由 IC-FT-EDF 调度,当相邻两个错误发生的时间间隔大于 $T_{n-1} + T_n$,出错任务的容错操作总能满足任务的时限要求,且不影响其他任务的正常运行。实际上,上述结论是基于任务的最坏执行时间确定的。在多数情况下,IC-FT-EDF 在 $T_{n-1} + T_n$ 范围内能够支持的容错操作的次数大于 1。

## 6 结论

IC-FT-EDF 和 FT-EDF 采用同样的基本调度策略和恢复策略,只有资源冗余策略不同。二者最大区别在于,IC-FT-EDF 通过分解任务,将 FT-EDF 完全依靠处理器冗余时间实现容错改变为依靠处理器冗余时间和任务本身的可选执行部分结合以实现容错。

调度算法的可调度条件直接决定其适用范围。因此,可以通过比较 IC-FT-EDF 与 FT-EDF 的可调度条件,以确定其适用范围间的关系。对于可以采用非精确计算的实时任务集,IC-FT-EDF 的适用范围大于 FT-EDF。对于 IC-FT-EDF 而言,若部分任务不能采用非精确执行,即 $m_i = C_i, O_i = 0$。在这种情况下,IC-FT-EDF 与 FT-EDF 的可调度性估计函数的值可能相同。对于任务集中任务必须实现精确执行的情况,IC-FT-EDF 退化为 FT-EDF。

为采用 IC-FT-EDF,必须精心地设计任务,以便在减少处理器资源冗余需求的同时,保证满足任务用户对任务的最低性能要求。另外,每个任务将执行两次错误检测,这将产生新的开销。

以一个实例直观地比较 IC-FT-EDF 和 FT-EDF。设任务集 $S$ 参数如下:

$\tau_1$: $C_1 = 0.8$, $m_1 = 0.6$, $o_1 = 0.2$, $T_1 = 4.0$

$\tau_2$: $C_2 = 1.5$, $m_2 = 1.0$, $o_2 = 0.5$, $T_2 = 5.0$

$\tau_3$: $C_3 = 2.5$, $m_3 = 1.8$, $o_3 = 0.7$, $T_3 = 9$

$\tau_4$: $C_4 = 1.0$, $m_4 = 0.7$, $o_4 = 0.3$, $T_4 = 10$

根据上述参数,FT-EDF 和 IC-FT-EDF 的可调度性估计函数分别为:

$F_{FT-EDF}(S) = 1.13 > 1$

$F_{IC-FT-EDF}(S) = 0.987 < 1$

所以,FT-EDF 无法调度该任务集,而 IC-FT-EDF 则可以。通过上述范例可以看出,对于适用于非精确计算的实时任务集,即完成部分操作就能够满足系统最基本要求,IC-FT-EDF 具有比 FT-EDF 更为广泛的适用范围,而嵌入式系统资源很有限,因此比较适用此算法,可以提高资源的利用率。

[参 考 文 献]

[1] A Burns, A Welling. Real-time system and programming languages[M]. Published by Addison Wesley, 1996.

[2] Anderson T, P A Lee. Fault tolerance principles and practice[M]. Prentice Hall, Engle Wood Cliff, NJ, 1981.

[3] Y Oh. The design and analysis of scheduling algorithm for real time computer systems[D]. University of Virginia, 1994.

[4] Lin K J, Natarajan S, Liu J W S. Imprecise results: utilizing partial computations in real-time systems[M]. Real Time Systems. Dec, 1987.

[5] P Alvarez, H Aydin, D Mossé. Scheduling optional computations in fault tolerant real-time systems[A]. Proceedings of the 8th International Conference on Real Time Computing Systems and Applications[C].

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net