基于并行优先级任务树的多核调度方法研究

黄姝娟[1,2]，朱怡安[1,2]

(1. 西北工业大学 计算机学院，陕西 西安 710072；2. 西北工业大学 软件与微电子学院，陕西 西安 710072)

摘 要：文章针对具有复杂依赖关系的实时周期任务难以调度的问题，提出了一种模型转化方法，该方法将具有复杂关系的实时结点任务图转化为并行优先级任务树，然后根据模型中定义的优先关系并结合 EDF (Earliest Deadline First) 算法，将任务调度到相应的核上去执行。仿真试验表明，该方法比核与任务相对应的分配方法不仅可以提高 CPU 利用率而且大大减少了核间通信的开销。

关 键 词：并行优先级任务树；有向无环图；多核；实时任务

中图分类号：TP301 **文献标识码**：A **文章编号**：1000-2758(2012)05-0652-05

随着多核技术在嵌入式领域的快速发展，研究片上多处理器实时调度问题就成为了热点[1]。目前大部分片上多处理器任务调度方法主要有 2 种模型[2]：①基于任务间相互独立的调度模型[3]，首先由 Liu 和 Layland[4] 提出。该模型虽然是很多实时任务模型的基础，并扩展至多处理器环境下用于任务调度的可行性分析与算法设计，却未考虑任务间的数据依赖关系、任务迁移以及同步通信等方面的内容，从而有一定的局限。②考虑任务之间具有约束关系的有向无环任务图(Directed Acyclic Graph，DAG) 模型[5]。在该模型中，虽然任务之间存在制约关系，却没有涉及任务周期性问题。本文考虑具有复杂依赖关系的实时周期任务图模型在多核平台下的调度问题，利用模型转换的思路将实时周期任务图转化为一种并行优先级任务树(Parallel Priority Task Tree，PPTT) 模型，然后在分析任务之间依赖关系的基础上定义优先级别，最后结合 EDF 算法进行调度。

1 调度算法模型

1.1 任务图型

任务图模型是多处理器调度任务调度的模型之一，通常采用 DAG 图来表示。具体描述如下：

一个任务图可以用一个四元组 $G = (V, E, C, T)$ 来表示，其中：$V = \{n_i \mid n_i$ 是任务结点，$i = 1, 2, 3, \cdots, |V|, |V|$ 表示任务结点个数$\}$；$E = \{e_{ij} \mid e_{ij}$ 表示任务 n_i 到 n_j 的边；$|E|$ 表示边的条数$\}$；$C = \{c_{ij} \mid c_{ij}$ 表示任务 n_i 到 n_j 的通信时间开销，n_i 称为 n_j 的前驱任务$\}$；$T = \{t_i \mid t_i$ 表示任务 n_i 的执行开销，$i = 1, 2, 3, \cdots, |v|\}$。

同时，用 $\mathrm{pred}(n_i) = \{n_j \mid e_{ji} \in E\}$ 表示 n_i 的前驱结点集，用 $\mathrm{indegree}(n_i) = |\mathrm{pred}(n_i)|$ 表示 n_i 的入度。如果入度 $\mathrm{indegree}(n_i) = 0$，则 n_i 为开始任务。

用 $\mathrm{Succ}(n_i) = \{n_j \mid e_{ij} \in E\}$ 表示 n_i 的后继结点集，用 $\mathrm{outdegree}(n_i) = |\mathrm{succ}(n_i)|$ 表示 n_i 的出度。如果出度 $\mathrm{outdegree}(n_i) = 0$，则 n_i 为结束任务。

1.2 并行优先级任务树模型

将并行优先级任务树记为 $\mathrm{Tree} = (D, R)$，其中，D 是结点的集合，R 是结点间关系的集合，T 满足：① 有且仅有一个特定的称为根的结点；② 当 $n > 1$ 时，其余结点可分为 $m(m > 0)$ 个互不相交的有限集 $T_1, T_2, \cdots, T_m$，其中每一个集合本身又是一棵树，并且称为根的子树。对于任何一棵树而言，每个结点都有层次，根为第 1 层，根的孩子为第 2 层。若某个结点在第 l 层其孩子就在第 $l + 1$ 层，树的深度为树中结点的最大层次。

收稿日期：2011-10-20 基金项目：航空科学基金(20100753022)、航天科技创新基金(2011XR160001) 及西北工业大学基础研究基金(JC201102837) 资助

作者简介：黄姝娟(1975—)，女，西北工业大学讲师，主要从事嵌入式与普适计算研究。

定义 1　对于任意一棵树 T,除了叶子结点外,所有结点都只有一个孩子结点的树称为单行树 ST(Simple Tree) 即单行树是一棵每个父结点只有一个孩子结点的树。

定义 2　由 m 棵单行树组成的森林称为单行树森林(Simple Tree Forest,STF) 。用 dim(STF) $= m$ 来表示。如图 1 是由 4 棵单行树组成的森林。STF $= \{T_1, T_2, T_3, T_4\}$; dim(STF) $= 4$;

ST 具有如下特征: ST 每层只有一个结点。结点个数等于层数。

1.3　从任务图模型向并行优先级任务树模型的转化

为了使 DAG 图转化为并行优先级任务树模型,需定义 DAG 图中的结点之间的关系。

定义 3　执行顺序关系——对于 DAG 图中任意 2 个不同任务的结点 A 和 B,如果 A 存在于 B 的前驱结点集合中,则称 A 和 B 有先后执行顺序关系,即 A 必须在 B 开始之前执行完。

定义 4　复制关系 —— 对于 DAG 图中任何一个 fork 任务结点 A,假设 A 的后继结点有 n 个,则可以通过将 A 结点复制 $n-1$ 次,形成 $A_1, \cdots, A_{n-1}, A$, n 个结点分别对应不同的 n 个后继结点,从而消除 fork 结点。此时,这 n 个结点 A_1、$A_2 \cdots A_{n-1}$, A 之间存在复制关系。

定义 5　合并关系 —— 对于 DAG 图中任何一个 join 任务结点 A,如果 A 的前驱结点有 m 个,则可将该结点拆分为不相同的结点 A_1、$A_2 \cdots A_m$ 分别对应不同的前驱结点。此时称这 m 个结点 A_1、$A_2 \cdots A_m$ 之间存在合并关系。

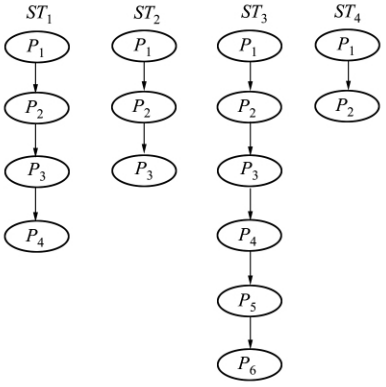

图 1　由 4 棵单行树组成的森林

定义 6　独立关系 —— 如果 A 和 B 结点之间不存在上述 3 种关系,则称此 2 个结点相互独立。

下面定义并行优先级任务树之间的关系。

定义 7　在 STF 中只有处于同一层上的结点之间具有复制关系和合并关系。将具有复制关系和合并关系的不同 ST 树称为具有复制关联关系或合并关联关系的单行树。

定义 8　如果 k 棵单行树既没有合并关联关系也没有复制关联关系,则称这 k 棵树之间具有相互独立关系。具有相互独立关系的不同树上的结点均相互独立。

定义 9　并行优先级任务树(PPTT——Parallel Priority Task Tree) 模型是一个二元组 PPTT $= (\boldsymbol{D}, \boldsymbol{R})$ 。其中 $\boldsymbol{D}$ 是实时周期任务的集合。$\boldsymbol{D} = \{T_1, T_2, \cdots, T_n\}$; $T_i = (e_i, d_i)$; $\boldsymbol{R}$ 用来定义该树中任意 2 个结点之间存在的关系,包括执行顺序关系、复制关系、合并关系和独立关系。

下面说明从任务图向并行优先级任务树的转化算法如下:

第 1 步　查找图中所有入度为 0 的结点放入树中第 1 层作为根的集合。

第 2 步　第 $k > 1$ 层集合的形成: 找到一个不在 $k-1$ 层中的结点,如果其全部前驱结点都在前 $k-1$ 层集合中,则该结点为第 k 层结点。

第 3 步　如果该结点的父结点已有其他孩子结点关联,则将父结点在自己所在的层中进行复制后与本结点建立父子执行顺序关系。并将复制结点进行标记; 如果没有,则直接建立父子执行顺序关系。

第 4 步　如果该结点对应有 m 个父结点,则将该结点在本层中拆分为不同的 m 个子结点,分别对应 m 个父结点。并将拆分结点标记为合并结点。

第 5 步　重复第 2 步至第 4 步直到所有结点都被放置到相应的层中,并最终形成一个单行树森林。而对单行树森林中的所有结点之间存在的关系进行界定就形成并行优先级任务树。

如图 2 所示的任务图 a) ,根据上述法则可以转化为单行树森林如图 2b) 所示。

定理　任何一个 DAG 任务图都可以通过任务复制和任务合并的方法形成惟一一棵单行树森林。

证明: 对于任意一个 DAG 任务图,则该图中的任何一个结点,根据上述方法在构建并行优先级任务树的过程中可以惟一确定该结点的层数。因为如果一个结点存在于不同的 2 个层则说明该结点的父

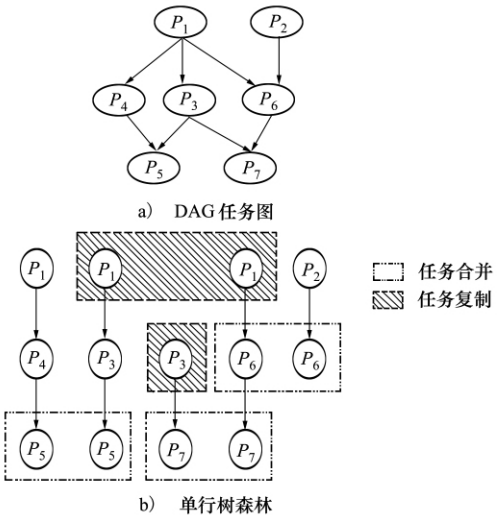

图2 任务图转化为单行树森林

结点就是本身,就会构成环路,这与DAG图相矛盾。又因为该接点的入度以及出度都是固定的,因而该结点需要复制或合并的个数也是固定的,再由于单行数森林中的单行树之间都是不存在先后顺序的问题,因此只存在惟一一种形式的单行数森林。

2 优先关系确定

为了研究并行优先级任务树中结点之间的优先关系,假设一个单行树森林STF,$\dim(\mathrm{STF}) = H$。首先定义一些符号:T^h表示第$h(1 \leqslant h \leqslant H)$棵单行树。$T_l^h$表示第$h$棵单行树上第$l(1 \leqslant l \leqslant L)$层上的结点,该结点为一个实时周期任务,周期为$p_l^h$,执行时间为$e_l^h$,即$T_l^h = (e_l^h, p_l^h)$。$T_l^h$的利用率为$U_l^h = e_l^h / p_l^h$。$T^h$树任务利用率为$U^h = \sum_{i=1}^{L} U_i^h$;系统STF总利用率为$U_{\mathrm{sum}} = \sum_{i=1}^{H} U^i$。由于每个任务被周期性地执行和释放,则称每次执行为一次工作(job)。$T_{l,j}^h$表示T_l^h的第j次job,该job的释放时间为$r_{l,j}^h$,相对死限为$d_{l,j}^h$,则每个job的绝对死限为$d_{l,j}^h = r_{l,j}^h + p_l^h$。由于job序列执行要求有先后顺序,即当$j > 1$时,第$j$次job的执行必须在$j-1$次job都释放后才能开始。同样,当$l > 1$时,$T_{l,j}^h$不能执行直到$T_{l-1,j}^h$以前的状态都执行完成。为避免混淆,定义2种前驱优先关系:定义L(Left)型前驱结点为$T_{l,i}^h$其中$i < j$。定义U(Upper)型前驱结点为$T_{k,i}^h$,其中$k < l$。任意一个job $T_{l,j}^h$能够开始执行,如果它的U前驱和L前驱都完成而且该项工作被释放。

下面就STF上不同关系的树与树之间、结点之间的优先关系(用"$\prec$"表示)进行讨论。

第1种情况 对于同一棵ST树上不同结点,可以定义这些结点的job的优先关系为上层结点的第j次job优先于下层结点的第j次job,即$T_{k,i}^h \prec T_{l,j}^h$其中$k < l$。同一结点的不同job,前者优先于后者即$T_{l,i}^h \prec T_{l,j}^h$其中$i < j$。

第2种情况 对于不同ST树上的同一层次的结点,可以存在3种关系:独立关系、合并关系和复制关系。如果这两个结点是合并关系,则其中一个结点需要与另一个结点同步,其中的一个结点的U前驱也将成为另外一个结点的U前驱。为了区分这两种不同的前驱,把将其他合并结点的前驱称为该结点的CU(Cross Upper交叉上型)前驱,该结点的每次job必须等待所有U前驱和CU前驱的同次job执行完毕后才能执行。如果在不同的核上执行U前驱和CU前驱,则需要同步通信开销。如果这两个结点是复制关系或独立关系,则可以并行执行,根据第3种情况的独立关系结点定义优先关系。

第3种情况 对于不同ST树上的不同层次的结点,只存在2种关系:CU前驱关系和独立关系。如果2个结点中一个结点是另一个结点的CU前驱,则CU前驱结点的第j次job优先于本结点的第j次job。即$T_{i,j}^w \prec T_{a,b}^c$,其中$(i < a) \wedge (j < b)$且$T_{i,j}^w$是$T_{a,b}^c$的CU前驱。对于任何2个具有相互独立关系的结点$T_{i,j}^w$和$T_{a,b}^c$,则按照EDF算法来定义优先关系,如果这两个结点为开始结点则优先顺序是:如果$d_{i,j}^w < d_{a,b}^c$,那么$T_{i,j}^w \prec T_{a,b}^c$。若死限相等,对于同一层的结点,如果$(d_{i,j}^w = d_{a,b}^c) \wedge (w < c) \wedge (i = a)$则$T_{i,j}^w \prec T_{a,b}^c$;对于不同层的结点如果$(d_{i,j}^w = d_{a,b}^c) \wedge (i < a)$则$T_{i,j}^w \prec T_{a,b}^c$。

3 处理器预分配算法

本文假定在进行处理器预分配算法时,忽略由于任务复制带来的开销。在分配核时,尽量将具有合并关系的结点任务所在的单行树分配到同一个核上执行,以减少同步通信开销。假设处理器核数和任务数相同情况下,处理器按照如下原则进行预分配:

1) 将 STF 中的每棵 ST 树分别分配到不同的核上去执行。

2) 为了依次减少合并次数,将具有合并关系的结点尽量分配到同一核上执行。

3) 在调度时按照优先执行关系进行调度。

系统流程图如图 3 所示。

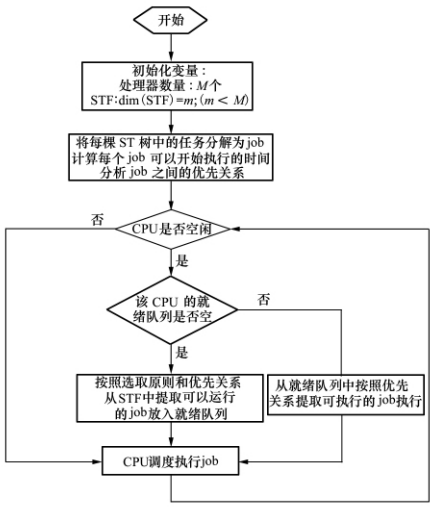

图 3　系统调度主流程图

4　仿真实验与分析

此次实验,采用了如图 2 所示的任务图。实验方法采用上述 2 种调度方法模拟分配实时周期任务到 7 个核上去执行,其中每个任务结点的实时周期特征采用随机数,但要满足以下 2 个约束条件: ①每个任务的最大执行时间 e 要小于或等于死限 d; ②若一个任务有前驱,则该任务的死限要大于其全部的前驱死限值。根据多组实验得出 CPU 利用率的平均值如图 4a) 所示。从图中可以看出采用 PPTT 模型分配使得 CPU 利用率得到提高并且可以减少系统所使用核的数量。另外,在 50 个时间片内所得出的核间通信次数如图 4b) 所示,可以看出采用 PPTT 模型分配的核间通信次数较核与任务对应分配的方法所需要的通信次数要少得多。

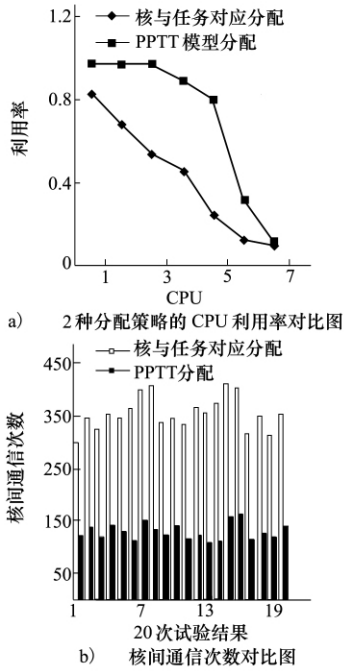

图 4　2 种算法结果对比图

5　结　论

本文针对具有复杂关系的实时周期任务在多核平台下的调度问题,提出了一种模型转化的方法,将复杂的带有实时性结点的 DAG 图转化为并行优先级任务树模型进行调度。文中首先描述了并行优先级任务树的模型,接着描述了带有实时性任务结点的 DAG 图向并行优先级任务树的转化方法,然后在此基础上,定义了相应的优先关系,最后结合 EDF 算法将其映射到具体的核上去执行。实验结果说明,该方法在对核的利用率方面比任务与核相对应的分配方法要高且大幅度减少了核间通信次数。

参考文献:

[1] Cong L, Anderson J H. Supporting Pipelines in Soft Real-Time Multiprocessor Systems. ECRTS'09. 21st Euromicro Conference on, Real-Time Systems, 2009, 269-278

[2] 李仁发等. 多处理器片上系统任务调度研究进展评述. 计算机研究与发展, 2008, 45(9): 1620-1629
Li Renfa, et al. A Survey of Task Scheduling Research Progress on Multiprocessor System-on-Chip. Journal of Computer Research and Development, 2008, 45(9): 1620-1629 (in Chinese)

[3] Mills A F, Anderson J H. A Stochastic Framework for Multiprocessor Soft Real-Time Scheduling. Real-Time and Embedded

Technology and Applications Symposium (RTAS), 2010, 10: 311-320

[4] Liu C L, Layland J W. Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment. Journal of the Association for Computing Machinery, 1973, 20: 46-61

[5] Bozdag D, et al. Compaction of Schedules and a Two-Stage Approach for Duplication-Based DAG Scheduling. IEEE Trans on Parallel and Distributed Systems, 2009, 20: 857-871

Improving Multi-Core Scheduling Method through Using PPTT (Parallel Priority Task Tree) Model

Huang Shujuan[1,2], Zhu Yi´an[1,2]

(1. Department of Computer Science and Engineering, Northwestern Polytechnical University, Xi´an 710072 China
2. Department of Software Engineering, Northwestern Polytechnical University, Xi´an 710072 China)

Abstract: Sections 1 through 4 of the full paper explain and evaluate the improvement mentioned in the title. The core of our thinking and that of sections 1 through 4 consists of: (1) past research papers on the multi-core scheduling model only focus on the independent tasks or the dependent tasks that are not real-time tasks; we propose a new scheduling model PPTT which can study the mutually dependent real-time tasks; it can ensure the satisfaction of the dependence relationship between any two tasks and can work together with the EDF(Earliest Deadline First) algorithm to schedule these real-time tasks running on the corresponding cores; (2) section 1 is entitled "Scheduling Algorithm Based on PPTT Mathematical Model"; (3) section 2 is entitled "Establishing Priority Relationship"; (4) section 3 in entitled "Allocating Algorithm of Multiprocessor"; (5) simulation results, presented in Fig. 4, and their analysis show preliminarily that the model and algorithm presented in this paper are indeed effective and the core has indeed a higher utilization.

Key words: algorithms, computer simulation, control, design, efficiency, embedded systems, flowcharting, mathematical models, multiprocessing systems, multitasking, real time systems, resource allocation, scheduling, synchronization; directed acyclic graph (DAG), multi-core, parallel priority task tree (PPTT)