第35卷 第17期 计 算 机 工 程 2009年9月
Vol.35 No.17 Computer Engineering September 2009
·软件技术与数据库· 文章编号：1000—3428(2009)17—0055—03 文献标识码：A 中图分类号：TP393

基于网格资源预测的任务优先级调度算法

刘洪伟[1]，于 炯[1,2]，田国忠[1,3]，龚红翠[1]
(1. 新疆大学信息科学与工程学院，乌鲁木齐 830046；2. 北京理工大学计算机学院，北京 100081；
3. 新疆工业高等专科学校计算机工程系，乌鲁木齐 830091)

摘 要：根据网格工作流中任务的依赖关系和截止时间，以及资源的有效度和MIPS(每秒百万条指令)，提出基于网格资源预测的任务优先级调度算法。把网格任务工作流抽象为有向无环图，找到该工作流的关键路径，计算每个任务的最迟开始执行时间，作为任务的优先级。在算法中考虑用户的要求和资源的类型，以及任务调度失败后重新分配的问题。实验验证了该算法的有效性。

关键词：工作流；网格；资源状态；优先级

Task Priority Schedule Algorithm Based on Grid Resource Forecast

LIU Hong-wei[1], YU Jiong[1,2], TIAN Guo-zhong[1,3], GONG Hong-cui[1]
(1. School of Information Science and Engineering, Xinjiang University, Urumqi 830046; 2. School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081; 3. Dept. of Computer Engineering, Xinjiang Polytechnic College, Urumqi 830091)

【Abstract】According to the tasks' dependence and deadline of grid workflow, effective degrees and MIPS of the grid resources, the task priority schedule algorithm based on grid resource forecast is presented. The algorithm uses DAG to find the critical path, obtains the deadline of every task and computes their PRI. The algorithm considers the request of user, the type of resources and re-schedule of failed tasks. Experimental result shows that the algorithm is effective.

【Key words】workflow; grid; resource state; priority

1 概述

网格工作流[1]调度算法作为网格工作流技术的重要组成部分，目前已成为网格领域的研究热点。网格工作流与传统的工作流不同，它运行于异构、分布的网格环境中，网格中各个资源除了对外开放自己的计算和服务能力，还要运行本地任务。在网格服务组成的工作流中，服务与资源间的映射关系组成网格服务的调度方案，而这种调度往往是NP完全问题。

用户在网格中使用工作流技术，可以根据自己的工作流程把简单、独立的网格任务组合起来，实现大规模、复杂的业务层网格应用。已有的网格工作流系统大多是根据高能物理或者生物信息学等专用网格系统中的工作流程复用等需求而设计的[2]。它们更偏重于工作流程建模，而其用户比较单纯，所以，没有划分用户和任务的优先级别，相应的调度策略也无优先性。对于用户群体比较复杂的一般网格而言，由于资源的有限性，级别较高的用户提交的紧迫性任务需要一定的机制来保证优先执行。

目前已在实验平台试验或在实际系统采用的基于优先级的动态调度算法有很多，最基本的有：最短截止期优先(Earliest Deadline First, EDF)[3]，最短处理时间优先(Shortest Processing Time First, SPTF)[3]，最小空闲时间优先(Least Slack First, LSF)和最高价值优先(Highest Value First, HVF)等。它们都只考虑了任务在某一方面的特征，如截止期、处理时间、空闲时间、价值等，而忽略了其他方面的特征，没有充分考虑任务完成所需要的诸多因素，特别是在网格工作流中资源的动态性。

本文引入资源预测[4]技术，全面了解资源的有效度和MIPS；尝试基于资源预测的调度研究，选择最适合任务运行的资源。文献[5]的资源预测模型算法是值得关注的，它对资源的进行等时间取样进行状态的取样，估算状态转移矩阵中的转移概率，运用连续时间Markov链计算“闲状态”的有效度，将有效度值较大的资源分配给即将执行的任务。它要在24 h内确定48次“忙”、“闲”、“停”状态的转移概率，得到48个转移矩阵来计算其空闲状态的有效度，过于复杂繁琐。还有一类预测资源的模型，即算术平均运算函数[4]，实际上这类预测函数已经出现，就是使用平均值作为预测值的预测算法(Mean-based Method)。基于这种算术平均运算函数的有均值算法(Running Average Method)、滑动窗口均值算法(Sliding Window Average Method)、LAST算法等。

这些算法考虑了资源的状态，但是没有预测资源的整体性能。本文提出了一种自适应调整算法来预测资源有效度和性能信息。

针对PESSIMIST类型的网格工作流系统，结合优先级分配算法和基于资源预测类算法的优点。将用户定义的具体网格工作流抽象为DAG图，首先在DAG图中找到其关键路径[6]，根据关键路径和用户的类型来计算任务的优先级，然后根据任务节点的预测执行时间计算和比较若干个候选资源

基金项目：国家自然科学基金资助项目(60563002)；教育部春晖计划基金资助项目(Z2005-1-65009)；新疆自治区高校科研基金资助重点项目(XJEDU2004I03)

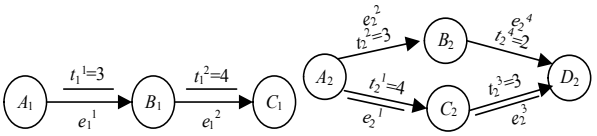
作者简介：刘洪伟(1980－)，男，硕士研究生，主研方向：分布式计算，网格计算；于 炯，教授；田国忠，讲师、硕士；龚红翠，硕士研究生

收稿日期：2009-05-09 **E-mail：**yujiong@xju.edu.cn

一段时间后性能信息和处于“闲状态”概率的大小，确定任务所要绑定的资源的新型调度算法。

2 网格工作流中任务优先级生成算法

网格工作流的任务结构可为 2 种：DAG(有向无环图)和 Non-DAG(非有向无环图)。针对 DAG 结构类型的网格工作流，通过图 1 中 2 个不同用户提交工作流 DAG 图来讨论本文的任务调度算法。



(a)工作流W_1 (b)工作流W_2

图 1 工作流实例

图 1(a)是一个具体的网格工作流实例，由有序的工作集合组成，它可抽象成 DAG 图 $G(V,E,W)$。其中，$V(G_1)=\{VA_1,VB_1,VC_1\}$是 G_1 的顶点集合，表示工作流的所有工作集合；$E(G_1)=\{e_1^1,e_1^2\}$, $e_1^1=<VA_1,VB_1>$, $e_1^2=<VB_1,VC_1>$，表示工作的执行活动，即任务。有向边上的权值集合 $W(G_1)=\{t_1^1,t_1^2\}$表示每一个任务预测的执行时间，如 $t_1^1=3$ 表示任务 e_1^1 的预测执行时间为 3 个时间单位。$V(G_2)=\{VA_2,VB_2,VC_2,VD_2\}$是 G_2 的顶点集合，表示工作流的所有工作集合，$E(G_2)=\{e_2^1,e_2^2,e_2^3,e_2^4\}$, $e_2^1=<VA_2,VB_2>$, $e_2^2=<VA_2,VC_2>$, $e_2^3=<VB_2,VD_2>$, $e_2^4=<VC_2,VD_2>$。权值集合 $W(G_2)=\{t_2^1,t_2^2,t_2^3,t_2^4\}$。

每个工作流是否在截至时间期限内完成，取决于关键路径(如图 1(b)中从起始顶点 VA_2 到结束顶点 VD_2 之间执行时间路径最长的路径)上的任务能否在各自的截至时间期限内完成。在 DAG 图中找到其关键路径(critical path)，这一算法很成熟，本文略去。

很容易找到关键路径为 $VA_2 \rightarrow VC_2 \rightarrow VD_2$，在这条路径上有 2 个关键任务要执行，分别是 $e_2^2=<VA_2,VC_2>$, $e_2^4=<VC_2,VD_2>$，在图中以复线标出。$First(t_i^j)$表示第 i 个工作流，第 j 个任务的最早能执行时间；$Last(t_i^j)$表示第 i 个工作流，第 j 个任务的最晚执行时间；$Start(t_i^j)$表示第 i 个工作流，第 j 个任务的实际开始执行时间；$Span(t_i^j)$表示第 i 个工作流，第 j 个任务的实际开始执行时间和最早执行时间之间的跨度，跨度越小表明任务等待时间越短；$PRI(e_i^j)$表示第 i 个工作流，第 j 个任务的优先级值。

$$Span(t_i^j)=Start(t_i^j)-First(t_i^j) \quad (1)$$

$$PRI(e_i^j)=Last(t_i^j)-T_C(T_C \text{ 为当前时间}) \quad (2)$$

优先级生成算法步骤如下：

步骤 1 利用关键路径推算出每个工作流的大约执行时间。

步骤 2 根据用户提交工作流的时间和用户的需求，计算出工作流的最后截止时间。

步骤 3 根据步骤 2 得出的工作流截止时间，计算本工作流中每个任务 E 的最迟开始执行时间 $Last(t_i^j)$和最早开始执行时间 $First(t_i^j)$。

步骤 4 利用步骤 3 算出的最迟开始执行时间，应用式(2)为任务赋予相应的优先级，任务越紧迫，优先级值越小。

步骤 5 按优先级值从小到大进行排序，如果优先级值相同，判断是否为关键路径上的任务。如果是，则排到相同优先级值的靠前位置；否则，再按照用户类型排序。

为了解决“饿死”(任务优先级较低，在截止时间之前得不到资源)的问题，任务的优先级值随着时间的消耗，进一步减少。将低优先级任务提升为高优先级，再根据上面的定义规则进行处理。

当某一任务的优先级值 PRI 减少到小于 0 时，查找出该任务所在的工作流中的全部任务并从队列中删除，返回该工作流调度失败信息。

3 资源预测的自适应调整算法

资源预测是以计算机资源为对象，将资源随着时间的顺序所得的观察值定义为时间序列，通过分析该时间序列，采用数理统计的方法对下一个时间段内的可能资源情况进行估计和评测。

本文所讲的计算机资源是指内存、CPU、硬盘以及网络，对于任务处理有重要影响的计算资源，和这些资源处于空闲状态的概率，它们在很大程度上决定了用户所提交任务在网格系统内的处理速度。现有的不同资源预测算法分别对以上预测形式都有体现，在文中使用的是资源预测自适应调整算法。

调整算法的思想是，根据对 t 时刻所预测的资源的有效度和 MIPS 值，对 t 时刻的实测值进行分析比较，进行适当的调整后作为 t+1 时刻的预测值。其数学表达如下：

$$ADAPT_f(t+1)=ADAPT_f(t)+\alpha\cdot(value(t)-ADAPT_f(t)) \quad (3)$$

其中，$ADAPT_f(t)$是指预测函数对 t 时刻的预测值；$Value(t)$是指 t 时刻的实测值；α 是调整因子，取值为 0~1 之间。在现有的算法中 α 的取值有 5%, 10%, 15%, 30%, 40%, 50%, 75%等。可以看出，这种算法的优点是，能够在不断的预测过程中自我调整，使得预测总是在正确的轨道上。但是其不足之处也很明显，就是其调整幅度只能够取一个阈值 α，固定的调整幅度不一定就是符合实际差值的尺度。

$$err_f(t)=value(t)-prediction_f(t-1)$$

它表示在 t 时刻所测得的实际数值和先前由函数 f 预测到的数值之间的差值，这实际上是一个评价标准，它的值越小说明预测越精确。而误差的大小是相对的，同一个误差值对于不同的实测值其意义是不同的。MPE(Mean Percentage Error)给出了一个很好的衡量标准。它以一段时间内误差值相对于实测值的比例(反映其对预测的影响程度)的平均值作为预测的精确度衡量标准。

$$MPE_f(t)=\frac{1}{t+1}\sum_{i=0}^{t}|err_f(i)|/value(i)$$

自适应调整算法的基本原理是：每次在 t 时刻被传送过来的数据都使用上文提到的 α 因子值进行计算并将结果 $predict_i(t+1)$保存下来，等到下一个时间 $t+1$ 将采集到的数值 $value$ 与相对应的预测值 $predict_i(t+1)$通过 MPE 进行比较，取其中预测最精确的作为下次的预测因子。在本文中的预测值包括 MIPS 和资源的闲、忙、停 3 个状态中处于空闲状态的概率。

4 算法描述

算法描述如下：

步骤 1 对用户提交的工作流，由工作流调度器按照上文中提到任务优先级生成算法，为其中的每个 P_i^j 计算优先级向量 $PRI(e_i^j)$。

步骤 2 为 W 建立临时工作队列 Q，将 W 中的每个 P_i^j 按照优先级向量 $PRI(e_i^j)$中的优先级值从小到大插入到 Q 中。

步骤 3 对任务队列中等待的所有应用程序用式(2)重新

计算优先级，并且更新每个任务的最早开始时间和最迟开始时间。如果任务的优先级值小于 0，查找出该任务所在的工作流中的全部任务并从队列中删除，返回该工作流调度失败信息。

步骤 4 如果 Q 中仍有任务，则将 Q 中具有最高优先级的任务取出，把它调度到由自适应资源预测算法中性能最好的资源上，并把任务进行标识。

步骤 5 等待：

(1)来自网格工作流中的任务完成事件，将该任务从 Q 中删除；

(2)任务调度失败或者被中止事件；

(3)新的工作流到达事件。

如果(1)或(2)发生，转步骤 3；如果(3)发生，转步骤 1。否则继续等待。

5 实验

在仿真实验中，分别对网格环境下的任务序列和资源序列进行模拟。设定了 2 组不同类型的包含时间期限的工作流队列，如图 1 所示。对于每一种情况，分别比较基于网格资源预测的任务优先级调度算法和 Min-Min 算法。在 Gridsim 仿真环境中用第 2 节中的算法计算 W_1 和 W_2 中任务初始化时的优先级向量，结果如表 1 所示。

表 1 W_1 和 W_2 中任务的优先级向量

任务	优先级向量
E_1^1	3,1
E_1^2	6,1
E_2^1	5,1
E_2^2	7,0
E_2^3	9,1
E_2^4	10,0

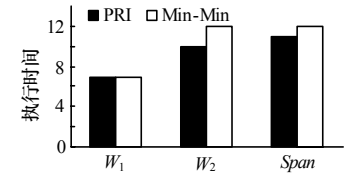
为了考察文中的调度算法在不同网格资源状况下的性能，进行如下实验：在 Gridsim 中设置 2 个可用资源。其性能比分别为 1:1 和 1:0.8；W_1 和 W_2 中每个应用程序的运行时间如图 1 所示，分别按照基于网格资源预测的任务优先级调度算法和非优先级的 Min-Min 算法，在 2 种性能比之下对这 2 个工作流进行调度。

由图 2(a)可以看出，当网格中资源性能相同时基于网格资源预测的任务优先级调度算法可以保证高优先级的任务优先完成。2 个工作流的总体执行时间比 Min-Min 节约了 11%。Min-Min 算法由于要等待前一个任务的执行结果导致了工作流的总体执行时间增加。

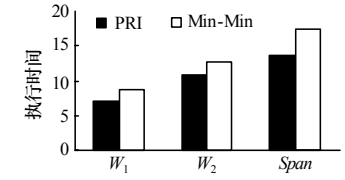
由图 2(b)可以看出，当网格中资源性能有差异时基于网格资源预测的任务优先级调度算法可以保证优先级高的任务使用优势资源，且关键路径上的节点可以在截至期限内完成，保证了工作流的总体进度，执行时间比 Min-Min 节约了 17.5%。

从上面的 2 个实验结果可以看到，基于网格资源预测的任务优先级调度算法任务执行的总体时间跨度要小于 Min-Min 算法。

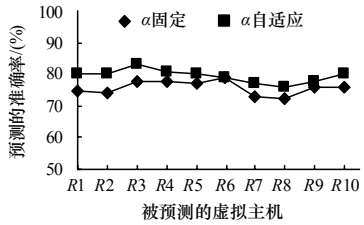
在资源预测中对 10 个资源进行设置，α 调整因子固定时和 α 自适应调整时进行了比较，重复 100 次求平均值得到的实验结果如图 2(c)所示。从中可以看出，自适应资源调整算法能对资源进行更好的预测。



(a)资源性能 1：1 情况比较



(b)资源性能 1：0.8 情况比较



(c)资源预测算法比较

图 2 工作流调度结果和资源预测比较

6 结束语

本文把网格中的工作流抽象为 DAG 图，根据每个工作流上任务的执行时间找到关键路径，利用关键路径来计算每个任务的优先级，把优先级高的任务分配到由资源自适应预测算法的优势资源上。实验结果表明，基于网格资源预测的任务优先级调度算法可以保证优先级高的任务在截至时间内完成，利用时间较短，又能充分使用网格中的资源，具有较好的性能，并且资源预测也有所提高。

参考文献

[1] 余 波, 周龙骧, 钟锡昌, 等. 网格工作流技术综述[J]. 计算机工程, 2006, 32(2): 4-8.

[2] Graham G E, Evans D, Bertram I. McRunjob: A High Energy Physics Workflow Planner for Grid[C]//Proc. of Conference on Computing in High Energy and Nuclear Physics. La Jolla, USA: [s. n.], 2003.

[3] 刘 怀, 胡继峰. 实时系统的多任务调度[J]. 计算机工程, 2002, 28(3): 43-44.

[4] 赵美平, 李胜利. 基于预测的集群作业资源管理研究[D]. 武汉: 华中科技大学, 2004.

[5] Byuna E J, Choia S J, Baikb M S. MJSA Markov Job Scheduler Based on Availability in Desktop Grid[J]. Future Generation Computer Systems, 2007, 23(4): 616-622.

[6] Yu Jia, Buyya R, Tham C K. A Cost-based Scheduling of Scientific Workflow Applications on Utility Grids[C]//Proc. of the 1st International Conference on e-Science and Grid Computing. Melbourne, Australia: [s. n.], 2005.

编辑 顾逸斐