基于信息中心策略的 P2P 资源管理与调度模型

霍　英[1,2]　李　登[2]　陈志刚[2]
[1](韶关学院信息工程学院,广东韶关 512005)
[2](中南大学信息科学与工程学院,长沙 410083)
E-mail:huoying@tom.com

摘　要　采用 P2P 方法构建资源管理和任务调度模型是目前网格研究的前沿和热点领域，但目前的 P2P 研究多集中在静态文件的搜索和调度领域。文章分析了当前主流 P2P 资源发现算法的优缺点,针对系统开销的根源,从全新的角度提出了 P2P 系统信息交互的分类方法,并在此基础上提出了一种基于广义资源和任务概念的区域自治的 P2P 模型,其信息中心的选择算法具有更好的合理性。最后通过理论分析证明该模型能较大地提高系统的运行能力,并能够兼顾效率与容错性。

关键词　对等计算　资源管理　任务调度　信息中心　网格

文章编号 1002-8331-(2006)19-0119-04　**文献标识码** A　**中图分类号** TP31

A Model of P2P Resource Management and Task Scheduling Based on Info-Center Policy

Huo Ying[1,2]　Li Deng[2]　Chen Zhigang[2]
[1](College of Information Engineering,Shaoguan University,Shaoguan,Guangdong 512005)
[2](College of Information Science and Engineering,Central South University,Changsha 410083)

Abstract: Building resource management and task scheduling models based on P2P structure is the frontier and focus in grid research field.However, current researches concentrate on algorithms of searching and scheduling about static files.In this paper,we present an overview of several search methods for P2P networks,and analyze the advantage and disadvantage of each search method.To solve the factors of extra spending,we propose a classification method of information exchange in P2P system from a whole new point of view.Then,based on concept of broad sense resource and mission,we present a locality-aware P2P model whose select algorithm about information center is more rational.At last, theory analysis is proposed.It proves that the model can improve performance ability of system,and it also gives attention to both efficiency and fault-tolerance.

Keywords: P2P,resource management,task scheduling,information center,grid

1　引言

目前大部分 Grid 系统采用集中式资源管理模式，较少考虑大规模网络环境下的可扩展性、容错性、动态性和不确定性。P2P 技术则是一种在高度动态的大规模网络环境下实现资源共享的技术,因此,使用 P2P 方法来进行资源管理和调度是网格研究中在该领域研究的新方向。然而目前多数 P2P 网络结点任务管理和调度算法的研究严格说来还是基于受限制的局域网的分布式系统[1,2],用模拟实验的方式或者理论推导的方式验证了算法的正确性,在理论研究上有一定的借鉴意义,但往往对于算法中的结点情况、所提出的任务提出了许多限制或者将实验环境理想化,特别是一些基于预测的调度算法[3],随着用户群体和计算类型的变化,难以找到一个普适性的负载行为预测和评估方法,因此,针对 P2P 系统的随机性和总体的不可预测性,面向普适计算的预测算法的有效性有待推敲。

本文首先阐述了资源管理和任务调度算法中产生额外开销的三个因素,然后在比较了目前存在的几种主流的资源发现算法之后，从一个全新的角度提出了资源信息交互的分类方法。并在此方法的基础上提出了基于信息中心策略的对等网资源管理和调度策略,最后对其合理性和优越性进行了理论分析。

2　资源管理和任务调度算法产生额外开销的因素

目前对“资源”有狭义和广义的两种解释,狭义的可以解释成某种物理实体,如 CPU、网络或存储资源等,广义的解释为在网络化环境中可被共享和利用的任何能力,包括物理资源(如计算力、科学仪器等)和逻辑资源(网络带宽、软件、应用服务等),本文采用资源的广义解释。任务也是广义的,除了计算任务,文件也属于任务,文中一律不加区别称为任务。

一般来说，资源管理和任务调度算法包含三个基本步骤,这三个过程均会产生额外开销:

(1)各结点间状态信息的收集

基金项目:国家自然科学基金资助项目(编号:60573127);国家教育部博士点基金资助项目(编号:20040533036)
作者简介:霍英(1975-),女,讲师,博士生,研究方向为对等网计算,网格计算。李登(1978-),男,博士生,研究方向为网格计算,对等网计算。陈志刚(1964-),男,教授,博导,研究方向为网格计算与分布式处理。

资源管理和任务调度中,任务要转移,就必须要在交互的结点间进行信息的传送,多数算法中各结点必须采集相邻结点的状态信息。

(2)根据所收集的状态信息进行决策

各结点根据自己和他结点的结点信息决定何时向何处怎样来发送或接受任务。

(3)任务的移动

调度本身就是任务的迁移过程,资源调度中常常出现的一个问题是:迁移的任务,在各结点间被不断地迁移而得不到执行,这种现象称为任务迁移的"颠簸"问题。

3 资源管理和任务调度算法的分类

P2P 系统中的资源搜索和定位算法主要分为三类:

(1)集中索引算法,以 Napster[4]系统为代表,采用了集中式的目录服务器机制,目录服务器中集中存放了各个结点的地址信息和所保存数据的信息。

(2)非结构化算法,代表系统为 Gnutella[5],采用洪泛或类洪泛算法,每一个用户消息都将被广播给与该用户直接相连的若干其他用户,这些用户收到消息后,也同样地将消息广播给各自连接的用户,依此类推,直到请求被应答或消息的 TTL 值减少为 0。

(3)结构化算法,典型代表包括 Tapestry[6],Pastry[7],CAN[8]和 Chord[9],它们采用了一种分布式 Hash 表(DHT)的数据结构。DHT 以〈key,data〉为描述数据的基本单元,〈key,data〉在网络上的安放位置决定了搜索效率。每个文件必须安放到预先规定好的 key 空间的某个位置上,这样才能保证搜索步数处于 $O(\log N)$数量级。

本文针对任何资源发现算法都必须涉及的额外开销的三个因素提出了一种新的分类标准。考虑到无论任何算法都必须涉及任务及其相关情况(即任务信息),所以我们将目前的算法归纳为任务信息无关调度策略和任务信息相关调度策略。

3.1 任务信息无关调度策略

任务信息无关调度策略在各结点间不进行任务信息的交换,当有转移(包括移出和移入)任务的需要时,就不断地打扰相邻结点,以满足其需求。典型的算法就是 Gnutella 系统采用的算法和集中式系统。

任务信息无关调度策略避免了频繁的信息交换,消除了任务信息交换带来的额外开销,但是带来了更大的决策和迁移中的额外开销。在结点有需要时,必须不断打扰相邻结点,给其他结点来负担,而有需要的结点如果处于超载的情况,对任务移出的不断请求则更加重了其负担。而集中式更是将其他结点的工作转嫁给了一台专用机,既限制了系统规模,又使一台机器不能参与任务执行,造成了资源浪费。

3.2 任务信息相关调度策略

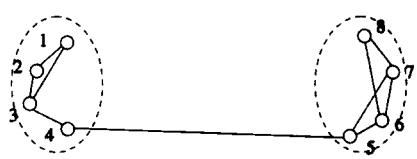
任务信息相关调度策略在各结点间根据不同的触发机制(周期性或状态变化驱动)需要进行任务信息的交换,各结点必须掌握并保留相邻结点的状态信息,这种信息报告(或询问)可以采用 Gnutella-like[10]的方式,即逐个通知,或是以广播的形式同时通知。目前的 Tapestry,Pastry,CAN 和 Chord 等采用结构化资源发现算法的系统也都是任务信息相关的。

这种方式的优点是信息较详细、准确,不需要被有任务请求的结点打扰,但是每隔一段时间(或者周期性,或者相邻结点状态发生变化时),就要被打扰;另外,自己状态的变化信息要通知多个相邻结点,加重了工作负担。特别在某一结点的相邻结点依次持续发生状态变化时 (该策略信息收集阶段最坏情况),该结点将不间断地受到打扰。

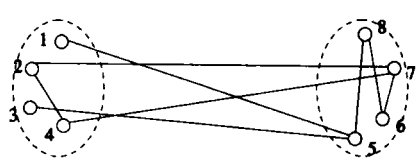
3.3 基于信息中心的调度策略

基于信息中心的调度策略结合了上述两种策略的优点而避免了其缺点。采用这种策略的 P2P 网络中,每一结点都属于一个结点自治域,该自治域中有一个信息中心存放其他结点的状态信息,只有当结点状态发生变化时,才向信息中心汇报状态信息,减少网络竞争。信息汇报采用状态动态变化触发机制,结点在负载变为超载(即有崩溃的危险)时,超载任务放入本结点的待命队列,并在信息中心登记,信息中心统计超载登记情况,结点变为轻载结点时,汇报负载情况时查看信息登记表,发现结点集合中有超载任务时,根据信息中心登记的情况从超载结点调入超载任务。除信息中心外,其他结点都不用具备记录邻接结点负载状态信息及相互间任务传递情况的属性,这是比结构化模型更优越的地方。

采用区域自治的 P2P 模型与一般 P2P 模型的区别如图 1 所示。图 1(a)中结点的交互首先尽量发生在同一个自治域之间,当本自治域无法满足需求时再通过访问更高层的信息中心获取其他域的信息,进行信息交换和任务处理,这样可以大大提高系统效率。



(a)区域自治的 P2P 模型



(b)一般的 P2P 模型

图 1 采用区域自治的 P2P 模型与一般的 P2P 模型的对比

4 基于信息中心策略的资源管理和任务调度

本研究提出的是资源管理和任务调度相分离的层次模型,通过资源管理层和任务调度层的划分,采用资源信息交互与任务调度相分离的方法,以解决 P2P 系统的分布、动态和自治性等问题。将信息管理从调度中分离的最大优点是:为便于 P2P 范围内资源发现而使用的资源信息聚合方法可采用松散的一致性模型, 这进一步减少了收集和搜索有关分布大资源集[11,12]信息的代价。借鉴网格研究中虚拟组织[13]的概念,我们突出强化资源与应用任务间的松耦合和动态绑定关系,使资源可以加入到不同的虚拟组织以支持细粒度的资源共享。

资源管理层与任务调度层是相辅相成的关系,例如任务调度层的状态发生变化,触发资源管理层的信息汇报机制,另一

方面,任务调度基于资源管理层汇聚的资源信息和结点状态信息。

4.1 信息中心的选择

由于 P2P 的特性,使用专门的服务器作为中央控制器不经济,容错性也不好。基于小世界理论和幂规律的 P2P 模型[14]和一种有效均衡的自治域模型[15]都采用了簇中心结点的概念,选取网络中连接结点最多的结点作为中心结点,两者都很好地应用了自组织网络的特性,大大提高了系统的效率。然而没有足够的证据表明依照小世界理论和幂规律选取的结点是网络中最强壮的结点。因而对于强调容错性的 P2P 网络而言,以聚集度选取中心结点的算法有较大的改进空间。文献[1]给出了两个公式以计算资源的可用性,在选择备份资源时只要备份资源的可用性不低于资源集合中可用性最低的资源就可以,这可以增加备份资源的选择范围。在理想的情况下这种算法显然是最优的,然而在实际情况中,P2P 网络中的结点失效率很难预测到。

本文中提出一种 Source Rank 机制,自治域中的结点 P_i 通过在线排队函数 SR 对于本域其他结点的任务完成历史进行排序。选取历史记录最良好的结点作为自治域的中心结点,选取记录次之的数个结点作为备份结点。初始化情况下,为了计算每个结点的历史记录等级 L_i,P_i 查询域内所有相邻结点的历史记录 h,并使用以下公式计算 $SR_{P_i}(L_i,h)$:

$$SR_{P_i}(L_i,h)=Dsim(h_j,h)^{\alpha}*N(L_i,h_j)$$

其中 $Dsim$ 是一个 sim 函数,它所反映的是结点 P_j 与 P_i 在物理位置上的向量关系,物理上越接近的结点 $Dsim$ 越大,我们假定,物理上相近的任务,在网络延时等方面耗费的开销比较远的结点间交互更加可靠与高效。$N(L_i,h_j)$ 反映了结点 P_i 与 P_j 间的交互次数,在自治域中我们总是倾向于选取与更多结点发生信息交互更多的结点作为信息中心,及在充分考虑容错性的同时兼顾效率。另外,我们增加了一个参数 α,以增加物理上相近的结点的权重,尽量避免网络拥塞等方面的影响。假如 α 值较大,则在物理上接近的结点($Dsim$ 最大)占优势。由上可知,信息中心策略的提出使系统具备较强的区域自治性,能较好地实现区域优化。

4.2 任务处理算法

我们用符号 S,S'和 S''来表示调度进程,用 A_i 来表示应用进程,其中 $1\leqslant i\leqslant n$(n 为一个有限整数,叫作某一结点的应用进程总数,其值由结点机性能决定),由于每个结点上的调度进程是唯一的,所以为了简单起见,我们也可直接用调度进程标识符来表示它所在的结点号,即调度进程 S 所在的结点也可称为结点 S。与调度进程 S 相关的表示还有:

(1)U_s:表示结点 S 上的应用进程集;

(2)W_s:表示结点 S 上还未计算执行的任务集。

定义 1 若 U_s 中每个应用进程的 busy 函数值为 true,即繁忙进程总数 $n_{busy}=n$,那么称结点 S 处于超载状态,结点状态标记为 l_s=high。

定义 2 若 U_s 中 busy 函数值为 false 的应用进程数($n-n_{busy}$)$<n*p$(p 称为轻载系数,$0\leqslant p\leqslant 1/2$,其值由结点机性能决定),那么称结点 S 处于轻载状态,结点状态标记为 l_s=low。

定义 3 若 U_s 中一部分应用进程的 busy 函数值为 true 且 $n*p\leqslant n_{busy}<n$,另一部分应用进程的 busy 函数值为 false,那么称结点 S 的负载状态为正常运行状态,结点状态标记为 l_s=normal。

定义 4 若对于某一有若干结点的自治域$\{S\}$,$\forall S'\in\{S\}$,$l_{s'}$=high,则称自治域$\{S\}$处于集合超载状态,记为 $L_{\{S\}}=busy$。

定义 5 对于某一有若干结点的自治域 $\{S\}$,$\exists S'\in\{S\}$,$l_{s'}$=low,对于 $S''\in\{S\}$,有$\{S''\}=\{S\}-\{S'\}$,$\forall S''\in\{S\}$,$l_{s''}$=high 且 $\sum l_{s'}>>\sum l_{s''}$,则称自治域$\{S\}$处于集合轻载状态,记为 $L_{\{S\}}$=free。

在以下的算法中,统一使用 *send-to*(*,*)和 *receive-from*(*,*)表示任务和结点状态信息的传递,将任务和结点状态信息传递分为 send-to 和 receive-from 是为了区分任务的启动者是本结点还是其他结点。这两者的后一个参数都表示传递的是结点状态信息还是任务,在算法中用 m,i 分别表示任务和结点状态信息。*send-to*(*,*)中的前一个参数表示结点状态信息或任务传递的目的进程,源进程即使用 send-to 的结点,而 *receive-from*(*,*)的前一个参数表示结点状态信息和任务传递的源进程,使用 receive-from 的结点即是目的进程。*set*(*)用来在信息中心设置结点状态,繁忙则用 busy,空闲用 free。

调度进程 S 对任务的调度分集合轻载和集合超载两种情况进行。以集合超载为例,有新任务到达时,则 S 对该任务的主要处理算法为:

```
(1)任给 A_i∈U_s,if(busy(A_i)==true)goto(4)
(2)if(∃A_i∈U_s,busy(A_i)==false)
  {  send-to(A_i,m);
     busy(A_i)=true;
     任给 A_i∈U_s,if(busy(A_i)==true)
       { if(l_s==normal){l_s=high;set(busy)};
         if(l_s==high) end;
       }
  }
  if(结点 S 符合定义 2)
  {  if(l_s==normal){l_s=low;set(free)}
     if(l_s==low)
     {if(W_s≠φ)
        {receive-from(W_s,m);W_s=W_s-{m};}
      if(W_s==φ) goto(3);}
  }
  if(结点 S 符合定义 3)
  {  if(l_s==high or l_s==low)
        {l_s=normal;set(normal)}
     if(l_s==normal)end;
  }
  (3)调度进程 S 访问信息中心负载状态记录,if(∃S'∈{S},l_s'==high){receive-from(S',m);end;}
  (4)W_s=W_s∪{m};end;
```

上述算法区分集合超载和集合轻载,超载时由轻载结点进行任务的调度,减轻了超载结点的负担,集合轻载时由少数的超载结点进行任务的调度分配,避免了众多轻载结点出现争抢任务的情况出现。

各结点不是在状态发生变化时向信息中心汇报,而是采用阈值触发机制,当负载状况超过限制时进行,这样进一步减少了信息中心和结点本身的工作量。而且,超载的任务不是立刻

发送出去,而是在得知确有受理结点时才发送出去。最大限度地减少了任务的“颠簸”,并且能有效保证系统中各结点工作在较安全范围,减少了结点崩溃的可能性,具有较好的容错性。

4.3 性能分析

当不引入信息中心算法时,设应用程序包含 n 个独立任务,其粒度分别为 $g_i, i\in\{1,\cdots,n\}$。并行系统包括 m 个单 CPU 工作站结点,其计算能力分别为 $c_j, j\in\{1,\cdots,m\}$。

结点 j 单独执行任务 i 的响应时间为:

$$t_j(g_i)=\frac{g_i}{c_j}$$

设结点 j 运行分配到的 n_j 个任务的总时间为 T_j,则并行加速比 s 就是在最快的结点上运行整个应用程序所需时间与并行执行时最晚结束的结点运行时间的比值:

$$s=\frac{\min\limits_{j\in\{1,\cdots,m\}}\sum\limits_{i\in\{1,\cdots,n\}}t_j(g_i)}{\max\limits_{j\in\{1,\cdots,m\}}(T_j)}$$

其中分子部分是一个常数,设 c 为在最坏的情况下,即并行计算中分配在最慢结点上的任务粒度远远大于其他任务时,可得 $s\approx\frac{c_{slow}}{l\cdot c_{fast}}$,其中 c_{fast} 和 c_{slow} 分别为最快和最慢结点的计算能力,l 为分配在最慢结点上的任务粒度与应用程序总粒度之比。可以看出,当系统各结点计算能力相差较大且任务粒度不均时,系统可能获得远远小于 1 的最差加速比,也就是说,并行时的性能不如在最快的一个结点上运行的性能。

下面再看一下采用本文算法时的情况。首先可以证明,最早结束的结点 a 的运行时间 T_a 与最晚结束的结点 b 的运行时间 T_b 之差 $\Delta\leqslant t_{slow}(g_{max})$,其中 g_{max} 是应用程序中最大的任务的粒度,$t_{slow}(g_{max})$是最慢的结点运行 g_{max} 的时间。

证明:用反证法。假设 $\Delta>t_{slow}(g_{max})$,则当结点 a 运行结束时,结点 b 正在运行的不可能是最后一个任务(也就是说,队列 Q 中还有未开始运行的任务),这是因为任何任务 x 的执行时间 $t_b(g_x)$小于等于 $t_{slow}(g_{max})$。根据算法,这时,结点 a 将向结点 b 申请运行任务。这就意味着结点 a 并未运行结束,与前提矛盾。

上面的结论保证了各结点的运行时间不会相差太大,从而实现了负载平衡。

另外,从加速比来看,由于最快的结点上运行的任务集是总任务集的子集,故运行时间不会超过常数 c。结合上面结论可得:

$$s>\frac{c}{c+t_{slow}(g_{max})}\approx 1$$

即在最坏的情况下,并行运算的时间与在最快的一个结点上串行运行相当。综上所述,采用信息中心策略的 P2P 资源管理和任务调度模型在保证良好容错性的同时,具有较高的效率。

5 结论和进一步的工作

本文主要在研究当前的 P2P 系统资源发现算法的基础上,针对 P2P 系统中产生额外开销的因素提出了一种新的信息交互机制的分类方法,并且通过 Source Rank 思想的提出阐述了一种新的信息中心结点的选取算法。最后在以上基础上提出了一种基于信息中心策略的兼顾效率与容错性的区域自治的 P2P 资源管理和任务调度模型。文中的理论分析证明这个模型是动态负载均衡的,有一定的先进性。文章从框架上解决了 P2P 系统中信息交互和任务调度的工作,接下来的工作将着重基于异构性考虑提出细粒度的自治域划分方法。并在此基础上提出一个 P2P 系统的原型系统,在进一步的工作中还将充分考虑资源间的通信特性,把应用的可用性要求和性能要求结合起来考虑。(收稿日期:2006 年 3 月)

参考文献

1.李春江,杨学军,肖侬.基于资源聚集的计算网格备份资源选择算法[J].计算机学报,2004;27(8):1137~1142

2.金海,陈刚,赵美平.容错计算网格作业调度模型的研究[J].计算机研究与发展,2004;41(8):1382~1388

3.L Yang,I Foster,J Schopf.Homeostatic and Tendency-based CPU Load Predictions[C].In:Proceedings of IPDPS 2003,2003-04

4.Napster.http://www.napster.com,2003

5.Gnutella.http://gnutella.wego.com,2003

6.B Kubiatowicz J,Joseph A.Tapestry:An infrastructure for fault-tolerant wide-area location and routing[R].Technical Report,UCB/CSD-01-1141,Computer Science Division,U C Berkeley,2001

7.Rowstron A,Druschel P.Pastry:scalable,distributed object location and routing for large-scale peer-to-peer systems[C].In:Guerraoui R ed.Proc of the Middleware,Heidelberg:Springer-Verlag,2001:329~350

8.Ratnasamy S,Francis P,Handley M et al.A scalable content-addressable network[C].In:Govindan R ed.Proc of the ACM SIGCOMM 2001,ACM Press,2001:161~172

9.Stoica I,Morris R,Karger D et al.Chord:A scalable peer-to-peer lookup service for Internet applications[C].In:Govindan ed.Proc of the ACM SIGCOMM 2001,ACM Press,2001:149~160

10.Yatin Chawathe, Sylvia Ratnasamy,Lee Breslau.Making gnutella-like P2P systems scalable[C].In:Proceedings of the 2003 Conference on Applications,Technologies,Architectures,and Protocols for Computer Communications,2003-08

11.Czajkowski K,Foster I,Kesselman C.Co-allocation services for computational Grids[C].In:8th IEEE International Symposium on High Performance Distributed Computing,IEEE Computer Society Press,Los Alamitos,CA,1999

12.Fitzgerald S,Foster I,Kesselman C et al.A directory service for configuring high-performance distributed computations[C].In:6th IEEE Symposium on High-Performance Distributed Computing,IEEE Computer Society Press,Los Alamitos,CA,1997:365~375

13.Foster I,Kesselman C,Tuecke S.The anatomy of the grid:Enabling scalable virtual organizations[J].Int'l Journal of Supercomputer Applications,2001;15(3):1~10

14.黄道颖,黄建华,庄雷等.基于主动网络的分布式 P2P 网络模型[J].软件学报,2004;15(7):1081~1089

15.陈志刚,刘安丰,熊策等.一种有效负载均衡的网格 Web 服务结构模型[J].计算机学报,2005;28(4)

16.Kunz T.The influence of different workload descriptions on a heuristic load balancing scheme[J].IEEE Transactions on Software Engineering,1991;17(7):725~730