基于多 QoS 需求驱动的网格资源调度研究*

莫 赞，谢 娜，贾功祥，赵 洁
（广东工业大学 管理学院，广州 510520）

摘 要：为解决网格用户多 QoS 需求的资源调度问题，引入了满意度函数模型和经典 Min-Min 算法。将众多网格 QoS 分为性能和信任两类，选取性能 QoS 中的优先级、时效性、精度性和信任 QoS 中的安全性、可靠性共五个指标，分别构建每一维 QoS 参数的满意度函数模型并形成 QoS 综合满意度函数模型，由此设计多 QoS 约束的网格资源调度（Q-Min-Min）算法，以期将 Min-Min 算法中按照期待执行时间（ETC）进行调度改为按照服务质量综合满意度（QSM）进行调度。仿真实验表明，改进的 Q-Min-Min 算法在任务的跨度和成本两项性能指标上均比 Min-Min 算法更具优势，取得了较为理想的结果，证明了基于多 QoS 需求驱动的网格资源调度的有效性。

关键词：网格；网格 QoS；资源调度；满意度函数模型；GridSim 仿真
中图分类号：TP393.04　　**文献标志码**：A　　**文章编号**：1001-3695(2012)10-3904-04
doi:10.3969/j.issn.1001-3695.2012.10.080

Research on grid resource scheduling for multi-QoS demands

MO Zan, XIE Na, JIA Gong-xiang, ZHAO Jie
(*School of Management*, *Guangdong University of Technology*, *Guangzhou* 510520, *China*)

Abstract: To solve the resource scheduling problem for multi-QoS demands, this paper adopted satisfaction function model and classical Min-Min algorithm. Firstly, it classified the numerous QoS parameters into two types which were performance QoS and trust QoS. Based on the priority, timeliness, precision of performance QoS and the security, reliability of trust QoS, it constructed five QoS satisfaction function models and formulated the overall satisfaction function model. Then it designed a multi-QoS constrained grid resource scheduling (Q-Min-Min) algorithm to compare with the classical Min-Min algorithm, in which the QSM replaced the ETC matrix. The simulation results show that Q-Min-Min algorithm gives more advantages over Min-Min algorithm in makespan and cost indexes, and prove the validity of grid resource scheduling based on multi-QoS demands.
Key words: grid; grid QoS; resource scheduling; satisfaction function model; GridSim simulation

0 引言

网格是一种能够把分布在网络上数以亿计的计算、存储、数据、信息、知识等资源结合起来，形成一个虚拟的逻辑整体，并将这些资源转换成一种随时随地可得、可靠、经济的计算能力的新技术和管理方法。在网格技术研究中，网格资源调度尤为重要，它根据网格节点的计算性能和资源状况，保障网格用户提交的任务以合理的方式分配到相应的网格资源上执行，同时满足网格用户的服务质量需求。

资源调度问题是一个 NP 完全问题，研究表明此类问题往往难以得到最优解，甚至根本就不存在最优解。因此本文引入满意度函数的概念，以讨论多 QoS 需求驱动情况下，寻找资源调度的满意解。

在国外已有的网格 QoS 的研究中，Lee 等人[1]提出了容错服务，以满足网格计算中的 QoS 要求，并提出和设计了故障检测服务和故障管理服务的算法。Doulamis 等人[2]在网格环境下，提出一个结合公平调度算法的非线性任务工作量预测机制，以解决网格中的任务分配和资源管理问题。Kyriazis 等人[3]提出可以绘制网格提供服务流程的一个新算法，同时该算法可以根据网格用户定义的参数和首选项提供 peer-to-peer 的服务质量。Castillo 等人[4]使用计算几何的概念来解决满足服务质量时资源提前预留机制中产生的资源碎片，并制定了一套调度策略。

国内方面，王嫚等人[5]提出了一种基于资源预留和选择性 QoS 的动态网格资源分配策略，实验表明可以获得近似最优的资源分配方案。张伟哲等人[6]提出了基于信任关系的网格服务调度算法，该算法能够兼顾性能 QoS 和信任 QoS 的两方面的要求，实验验证了该算法的有效性。赵建峰等人[7]采用克隆选择算法以解决多 QoS 约束下的工作流调度难题，提出克隆选择算法，与多种算法比较后发现该算法能进行最优调度。孙伟峰等人[8]以带有 QoS 约束的任务为研究对象，结合改进的蚁群算法，提出了一种基于蚁群算法的多 QoS 约束的 QIACO 算法，该算法将 QoS 约束转换成效用。王磊等人[9]提出一种基于 QoS 的协作型任务调度遗传算法，将任务完成时间、价格和可靠性三个 QoS 参数引入遗传算法，实现了网格环境下协作型任务调度对服务质量的优化，并保证了协作型任务

收稿日期：2012-03-15；**修回日期**：2012-04-30　**基金项目**：国家自然科学基金资助项目(71171062)；十二五国家科技支撑计划资助项目(2011BAD13B11)；广东省自然科学博士启动基金资助项目(S2011040004285)；广东工业大学教学质量工程基金资助项目(402102283)

作者简介：莫赞(1962-)，男(壮族)，广西来宾人，教授，博士，主要研究方向为电子商务、管理信息系统(mozan@gdut.edu.cn)；谢娜(1986-)，女，硕士，主要研究方向为电子商务、管理信息系统；贾功祥(1986-)，男，硕士，主要研究方向为决策理论与方法；赵洁(1979-)，女，副教授，博士，主要研究方向为信息系统、电子商务.

之间的数据依赖。

从这些文献中可以发现,研究者考虑到了多 QoS 约束的问题,但是对于众多的 QoS 没有进行细分,特别是信任和性能两种 QoS 约束指标的讨论[10]。同时文献定义了信任、公平、容错概念,并将其纳入 QoS 约束中,但是没有兼顾到性能 QoS 的需求[2~4,6]。虽然文献考虑到了用户的 QoS 需求问题,但是考虑到的 QoS 指标较少,需要扩充更多的指标以满足网格中大规模用户的实际情形[5,9]。还有的文献与现代优化算法,如蚁群算法、克隆选择算法相结合,将多个 QoS 聚合成综合 QoS 约束[7,8]。网格作为一种技术和管理方法,需要提供和解决大规模的网格用户 QoS 需求问题,但是网格中的规模性用户和资源等多个实体对管理机制、安全策略、费用的 QoS 目标都不尽相同,甚至相互抵触,因此不仅需要考虑到更多的细分的 QoS 约束指标,同时还需设计出能够使得这一抵触可以消解或调和的算法。

1　网格 QoS 的分类和满意度函数模型

1.1　网格 QoS 的分类

基于多 QoS 需求驱动的网格资源调度研究的首要问题是对用户 QoS 进行分类,这样才能为进一步参数量化和满意度函数模型的构建作准备。网格环境下的 QoS 参数有很多,不同的标准将获得不同的分类结果。

Sabata 等人[11]将分布式系统中的 QoS 描述分为度量标准(metrics)和策略(polices)两类,即将 QoS 参数分为 QoS 度量和 QoS 策略两组。QoS 度量用于量化可计量的 QoS 参数,进一步分为时效性、精密性、准确性、安全性、应用的相对重要性等;QoS 策略描述行为规范,包括管理策略、服务级别。

结合 Sabata 等人对的 QoS 的分类及本文的研究内容,本文主要选取性能 QoS 中的优先级(priority)、时效性(timeliness)、精度性(precision)和信任 QoS 中的安全性(security)、可靠性(reliability),并将服务级别定义为三类服务级别:

a)刚性级(hard)。具有这种级别的 QoS 需求必须得到满足。即任务获得的网格资源属性值必须高于任务本身请求值,如果不存在满足条件的资源,则此请求将被放弃。

b)弹性级(soft)。任务获得的网格资源属性值如果高于任务本身请求值则获得最大效益;如果请求条件不被满足,任务的执行仍然有效,但其满意度相应降低。

c)尽力级(try)。任务获得的网格资源属性值与任务本身请求值的大小结果对任务的执行没有影响,但该类约束被满足时,也可获得相应的满意度。

1.2　满意度函数模型

网格环境下多 QoS 需求驱动的资源调度问题,属于多目标优化问题,但是要找到一个能够兼顾五个 QoS 约束优化目标的最优解往往难以实现,在实际应用中,常常是去寻找次优解即满意解。求满意解首先需要构建评价函数,从而将多 QoS 约束转换为能够综合这些 QoS 的单目标优化问题,利用传统的单目标优化问题的求解方法进行求解。

因此选取满意度函数,同时考虑到不同用户的满意偏好不同,定义如下能够反映用户满意度变化的满意度函数模型。

1.2.1　性能 QoS 满意度函数模型

1)优先级满意度函数

定义网格资源当前等待任务队列的计算规模 $T_{\text{wait}}(j)$,任务自身的计算规模 $T_{\text{self}}(i)$,单个网格资源的最大处理规模 $R_{c\text{-max}}(j)$。结合满意优化理论可得到在优先级上的满意度函数:

$$U_{\text{prio}(i,j)}=\begin{cases}1-\dfrac{T_{\text{self}}(i)+T_{\text{wait}}(j)}{R_{c\text{-max}}(j)} & T_{\text{self}}(i)+T_{\text{wait}}(j)\leqslant R_{c\text{-max}}(j)\\ 0 & T_{\text{self}}(i)+T_{\text{wait}}(j)>R_{c\text{-max}}(j)\end{cases}$$

其中,$1\leqslant i\leqslant m$,$1\leqslant j\leqslant n$。在任务调度中,使得

$$U_{\text{prio}}=\{\max U_{\text{prio}}(1,j)+\cdots+\max U_{\text{prio}}(m,j),j=1,2,\cdots,n\}$$

2)时效性满意度函数

时效性满意度函数即网格用户提交某一个任务 $t_i(t_i\in T,i=1,2,\cdots,M)$,该任务被选择调度到网格资源 $r_j(r_j\in R,i=1,2,\cdots,N)$ 上时从执行运行效率上获得满意程度,由运行时间满意度函数与计算规模满意度函数共同确定。

a)定义任务的最迟完成时间 $\text{timel}_{\max}(i)$、任务分配到资源上的期待执行时间 $\text{timel}_{\text{expect}}(i,j)$,则运行时间满意度函数分别定义为

刚性级:

$$U_{\text{makespan-hard}}(i,j)=\begin{cases}1 & \text{timel}_{\text{expect}}(i,j)\leqslant\text{timel}_{\max}(i)\\ 0 & \text{else}\end{cases}$$

软性级:

$$U_{\text{makespan-soft}}(i,j)=\begin{cases}1 & \text{timel}_{\text{expect}}(i,j)\leqslant\text{timel}_{\max}(i)\\ \text{power}(a,\text{timel}_{\max}(i)-\text{timel}_{\text{expect}}(i)) & \text{else}\end{cases}$$

其中 power( )表示指数函数,$a>1$ 的常数。

尽力级:

$$U_{\text{makespan-try}}(i,j)=\text{power}(a,-\text{timel}_{\text{excepl}}(i,j))$$

b)获得任务在时效性上的满意度时,需要同时考虑到该任务的计算规模和相应资源的处理能力。定义任务 $t_i(t_i\in T,i=1,2,\cdots,M)$ 自身的计算规模 $T_{\text{self}}(i)$,网格资源 $r_j(r_j\in R,j=1,2,\cdots,N)$ 的最大处理规模 $R_{c\text{-max}}(j)$,得到在计算规模上的满意度函数为

$$U_{cr}=\begin{cases}\text{power}[a,(T_{\text{self}}(i)-R_{c\text{-max}}(j))] & T_{\text{self}}(i)\leqslant R_{c\text{-max}}(j)\\ 0 & \text{else}\end{cases}$$

采用加权的方法得到时效性的综合满意度函数为

$$U_{\text{timel-}k}(i,j)=\begin{cases}\omega_1\times U_{\text{makespan-}k}(i,j)+\omega_2\times U_{cr}\\ \omega_1+\omega_2=1\end{cases}$$

其中,k 分别为 hard、soft、try。

根据 GridSim 的实验手册[12]和前人的研究[6,13~15]讨论,选取 $\omega_1=0.8$,$\omega_2=0.2$。

3)精度性满意度函数

精度性满意度函数表征将一个任务 $t_i(t_i\in T,i=1,2,\cdots,M)$ 分配到资源 $r_j(r_j\in R,j=1,2,\cdots,N)$ 上时,资源能够保障任务精度 QoS 需求的程度而给网格资源用户带来的满足程度。定义任务 t_i 的最小精度 QoS 需求 $T_{t\text{-precision}}(i)$,网格资源 r_j 提供的精度 QoS 保证 $T_{r\text{-precision}}(j)$,则精度性的刚性级、软性级、尽力级满意度函数分别为

$$U_{\text{precision-hard}}(i,j)=\begin{cases}1 & T_{t\text{-precision}}(i)\leqslant T_{r\text{-precision}}(j)\\ 0 & \text{else}\end{cases}$$

$$U_{\text{precision-soft}}(i,j)=\begin{cases}1 & T_{t\text{-precision}}(i)\leqslant T_{r\text{-precision}}(j)\\ 1-\dfrac{T_{t\text{-precision}}(i)-T_{r\text{-precision}}(j)}{T_{t\text{-precision}}(i)} & \text{else}\end{cases}$$

$$U_{\text{precision-try}}(i,j)=\begin{cases}\dfrac{T_{r\text{-precision}}(j)}{T_{t\text{-precision}}(i)} & T_{r\text{-precision}}(j)<T_{t\text{-precision}}(i)\\ 1 & \text{else}\end{cases}$$

1.2.2 信任 QoS 满意度函数模型

1)安全性满意度函数

服务级别分别为硬性级、软性级和尽力级时,任务 t_i 分配到资源 r_j 上第 k 个执行时安全性效益函数在定义为

$$U_{\text{sec-hard}}(i,j)=\begin{cases}1 & T_{t\text{-safe}}(i)\leqslant T_{r\text{-safe}}(j)\\ 0 & \text{else}\end{cases}$$

$$U_{\text{sec-soft}}(i,j)=\begin{cases}1 & T_{t\text{-safe}}(i)\leqslant T_{r\text{-safe}}(j)\\ 1-\dfrac{T_{t\text{-safe}}(i)-T_{r\text{-safe}}(j)}{T_{t\text{-safe}}(i)} & \text{else}\end{cases}$$

$$U_{\text{sec-try}}(i,j)=\begin{cases}\dfrac{T_{r\text{-safe}}(j)}{T_{t\text{-safe}}(i)} & T_{r\text{-safe}}(j)<T_{t\text{-safe}}(i)\\ 1 & \text{else}\end{cases}$$

其中:$T_{t\text{-safe}}(i)$、$T_{r\text{-safe}}(j)$分别表示任务 t_i 安全性需求及资源 r_j 提供的安全性水平。

2)可靠性满意度函数

可靠性效益函数表征将一个任务 t_i 分配到资源 r_j 上第 k 个执行时,在可靠性 QoS 上获得的满意度。设 Rt_i 为 t_i 的最小成功完成概率,任务 t_i 在资源 r_j 运行获得的可靠性为 $Rr_{i,j}$,即

$$Rr_{i,j}=\exp\{-CT_{i,j}\times Fr_j\}$$

其中:$CT_{i,j}$为任务 t_i 在资源 r_j 上期望执行时间,Fr_j 为资源 r_j 的固有失效[16]。

分别定义可靠性函数为

刚性级:

$$U_{\text{reli-hard}}(i,j)=\begin{cases}1 & Rt_i\leqslant Rr_{i,j}\\ 0 & \text{else}\end{cases}$$

软性级:

$$U_{\text{reli-soft}}(i,j)=\begin{cases}1 & Rt_i\leqslant Rr_{i,j}\\ \dfrac{\text{power}(a,(Rt_i-1))-\text{power}(a,-(Rr_{i,j}+1-Rt_i))}{\text{power}(a,-(1-Rt_i))-\text{power}(a,-1)} & \text{else}\end{cases}$$

尽力级:

$$U_{\text{reli-try}}(i,j)=\frac{1-\text{power}(a,-Rr_{i,j})}{1-\text{power}(a,-1)}$$

1.3 QoS 综合满意度模型

采用加权形式的定义综合满意度函数,即

$$f(U)=f(U_{\text{prio}},U_{\text{timel}},U_{\text{precision}},U_{\text{sec}},U_{\text{reli}})=\sum_{1}^{di}\overline{\omega}_iU_{(\text{prio,timel,precision,sec,reli})}$$

其中,$\overline{\omega}_i$ 分别表示为综合满意度函数性能 QoS 和信任 QoS 满意度函数上的权重,满足$\sum_{1}^{di}\overline{\omega}_i=1$。

2 算法设计

设计改进的 Q-Min-Min 算法核心代码如下:

```
public void achieveij( ) {
//略去变量定义的代码
for( i =0;i < QSM. length;i ++ ) {
  for( j =0;j < QSM[ i]. length;j ++ ) {
    if( QSM[ i][ j] ==0) continue;
    if( min ==0.0 || QSM[ i][ j] < min) {
      min = QSM[ i][ j];
      I = i;
      J = j;
    }
  }
}
for( i =0;i < QSM. length;i ++ ) {
  for( j =0;j < QSM[ i]. length;j ++ ) {
    if( QSM[ i][ j] ==0) continue;
    if( j == J) {
  QSM[ i][ j] += min;
    }
    if( i == I) {
    QSM[ i][ j] =0;
    }
  }
}
//检测数组里如果有未调度的任务,就递归再去处理一次
  for( i =0;i < QSM. length;i ++ ) {
    for( j =0;j < QSM[ i]. length;j ++ ) {
      if( QSM[ i][ j] ! = 0) {
        achieveij( );
        break;
      }
    }
  }
  Gridlet gridlet;
  gridlet = ( Gridlet) this. list_. get( I);
  super. gridletSubmit( gridlet, resourceID[ J],0.0,false);
//提交任务,实现按照 Q-Min-Min 算法进行资源调度
//省去非核心代码部分
}
```

3 仿真实验及结果分析

3.1 实验参数设置

在 GridSim 仿真平台中进行仿真,结合 QoS 满意度函数模型,首先需要设置如下实验参数。$T_{\text{wait}}(j)$、$T_{\text{self}}(i)$分别在[0,10^5]、[10^1,10^5]上随机生成,$R_{c\text{-max}}(j)$在[10^1,10^5]上随机生成。设置变量 $V_Q(1\leqslant V_Q\leqslant 4)$并生成随机数 $r_1\in[0,1]$以控制任务与资源间的服务级别。若 $r_1<0.25V_Q$,则为硬性级服务级别;若 $r_1<0.5V_Q$,则为弹性级服务级别;否则为尽力级服务级别。最迟完成时间 $\text{timel}_{\max}(i)$由公式 $\text{timel}_{\max}(i)=(0.25+V_T\times r_i)\times(R/T)^{1/2}\times R_{\text{task}}(i)$生成,其中 V_T 为控制 $\text{timel}_{\max}(i)$生成的变量,r_i 为在[0,1]上均匀分布的随机实数。期待执行时间 $\text{timel}_{\text{expect}}(i,j)$由 ETC 矩阵产生,ETC 矩阵通过基于范围的 ETC 矩阵产生方法或基于变异系数的 ETC 矩阵产生方法[17]两种方法获得,本文采用第一种方法。$T_{t\text{-precision}}(i)$、$T_{r\text{-precision}}(j)$均在{16,32,64,128,256}中随机生成。任务 t_i 在资源 r_j 运行获得的可靠性为 $Rr_{i,j}$,$Rr_{i,j}$由公式 $Rr_{i,j}=\exp\{-CT_{i,j}\times Fr_j\}$生成,其中 $CT_{i,j}$为任务 t_i 在资源 r_j 上的运行时间,Fr_j 为资源资源 r_j 的固有失效率[16]。任务 t_i 的最小成功完成概率 Rt_i 由公式 $Rt_i=[V_R+(1-V_R)\times r_i]\exp((R/T)^{1/2}\times 10^{-4}\times R_{\text{task}})$生成,其中 V_R 为控制 Rt_i 生成的变量,之所以设置 V_R 控制变量是因为生成 $Rr_{i,j}$时其值范围为(0,1),所以必须保证 Rt_i 的随机生成值在相应的(0,1)范围内;Fr_j 为资源资源 r_j 的固有失效率,Fr_j 在[0.0001,0.0015]上随机产生。$T_{t\text{-safe}}(i)$、$T_{r\text{-safe}}(j)$安全需求级别在{1,3,5,7}中随机生成。

3.2 仿真实验及结果分析

3.2.1 仿真对比实验 1

系统随机产生两个资源 R_1、R_2 和四个任务 $T_1\sim T_4$。运行结果如表 1 所示。

表 1 仿真对比实验 1 的实验结果

算法	R_1	R_2	成本	运行时间
Min-Min	T_2、T_4	T_1、T_3	141.144	266.528
Q-Min-Min	T_2、T_3	T_1、T_4	132.206	254.909

从表 1 中可知:

a)改进后的 Q-Min-Min 算法与经典 Min-Min 算法的调度结果不同。Q-Min-Min 算法的调度结果是将 T_2、T_3 调度到 R_1 上执行,将 T_1、T_4 调度到 R_2 上执行,而 Min-Min 算法的调度结果是将 T_2、T_4 调度到 R_1 上执行,将 T_1、T_3 调度到 R_2 上执行;

b)Q-Min-Min 算法调度执行的平均总成本为 132.206,经典 Min-Min 算法调度执行的平均总成本为 141.144,可见改进后的算法对于成本有较好的节约,节约率为 6.332%。

c)改进的 Q-Min-Min、经典 Min-Min 算法调度执行的 makespan 分别为 254.909、266.528,改进后的算法运行时间节约率为 4.360%。

3.2.2　仿真对比实验 2

设置资源数为 10,任务数分别为 20、40、60、80、100、120、140、160、180、200,依次递增。最后的实验结果统计如表 2 所示。

表 2　仿真对比实验 2 的实验结果

任务数	资源数	Q-Min-Min		Min-Min	
		总成本	运行时间	总成本	运行时间
20	10	449.777	1 300.566	508.503	1 712.021
40	10	736.440	3 275.320	836.492	3 431.323
60	10	1 130.480	4 300.867	1 292.200	5 041.973
80	10	1 632.040	6 451.133	1 769.962	6 670.027
100	10	2 107.292	8 229.871	2 339.758	8 370.439
120	10	2 397.407	9 658.656	2 720.005	10 042.708
140	10	2 800.682	10 600.774	3 026.763	11 626.961
160	10	3 296.060	11 802.639	3 352.678	13 273.927
180	10	3 728.645	13 505.722	3 761.084	14 827.415
200	10	4 073.377	16 183.792	4 190.613	16 678.111

根据表 2 作出图 1 和 2,以进一步清晰地比较两种算法的平均总成本和运行时间,分析两种算法的优劣势和改进状况。从图中可知:a)当资源数量不变,任务数量增加时,经典的 Min-Min 算法和改进的 Q-Min-Min 算法的运行时间和成本都随之增加,与理论相一致,充分说明了本模型和算法的正确性;b)改进的 Q-Min-Min 算法无论在运行时间即跨度还是总成本方面都比 Min-Min 算法表现更好,这表明本文考虑了 QoS 约束的算法是有效的,能够获得比经典算法更好的运行结果。

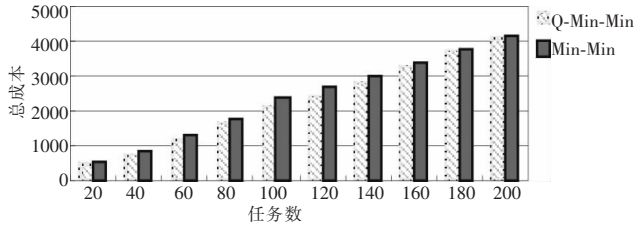

图 1　仿真对比实验 2 的总成本统计

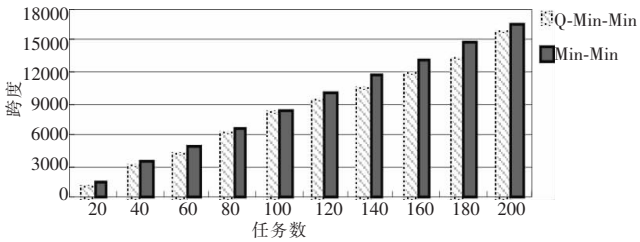

图 2　仿真对比实验 2 的跨度统计

3.2.3　仿真对比实验 3

设置任务数为 300,资源数分别为 20、40、60、80、100,依次递增。最后的实验结果统计如表 3 所示。

表 3　仿真对比实验 3 的实验结果

任务数	资源数	Q-Min-Min		Min-Min	
		总成本	运行时间	总成本	运行时间
300	20	6 531.205	36 433.348	6 568.705	36 839.328
300	40	6 156.98	23 721.131	6 297.238	25 504.038
300	60	6 103.298	24 391.930	6 402.626	24 840.307
300	80	6 352.196	24 921.531	6 469.859	25 044.899
300	100	6 324.68	24 411.062	6 342.765	25 076.547

根据表 3 作出图 3 和 4,从图中可以得出,当任务数固定时,随着资源数的增加,经典 Min-Min 和改进的 Q-Min-Min 算法的成本和运行时间都发生相应的变化,大致呈现先递减后增加的趋势。其原因在于,任务调度和资源的匹配是一个相互选择的动态过程:当资源较少时,增加资源可以较好地节约运行时间和成本;但当资源增加到一定程度时,反而增加了任务和资源匹配的过程,所以成本和运行时间不是随着资源数的增加而相应递减。

另一方面,改进的 Q-Min-Min 调度算法在总成本、跨度上比经典算法更具优势,且总成本的节约率相对于跨度的节约率更好,这是因为考虑 QoS 时就增加了任务调度到资源上的约束,调度算法在调度任务时不可避免地增加了运行时间,即对跨度的改进不明显。

实验结果依然可以得出这样的结论:改进的 Q-Min-Min 算法在 makespan、总成本方面都要优于经典的 Min-Min 算法。这说明基于多 QoS 需求驱动的网格资源调度能获得更好的结果,它能结合网格用户不同 QoS 需求的变化,及时调整资源调度策略,更有效地利用了系统资源,从而不仅完成了 m 个任务在 n 个资源上的映射,更重要的是保障了用户多 QoS 需求。

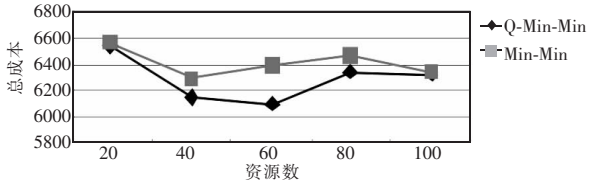

图 3　仿真对比实验 3 的总成本统计

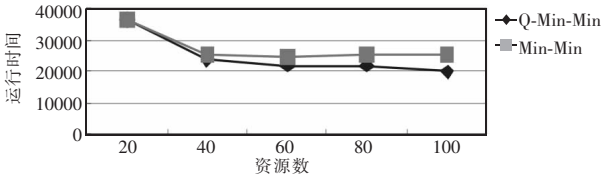

图 4　仿真对比实验 3 的跨度统计

4　结束语

基于用户 QoS 需求驱动的网格资源调度是网格技术研究的热点问题,是解决网格提供个性化服务的重要方法,而网格资源调度算法的研究又是其核心内容。仿真实验结果表明,结合满意度函数模型的多 QoS 需求驱动的网格资源调度(改进的 Q-Min-Min)算法在运行时间和成本方面能够获得比经典 Min-Min 算法更好的结果。这说明在网格环境中应用 Q-Min-Min 算法不仅能够满足网格用户的个性化需求,还可以更有效地共享系统资源,实现网格的长远发展。

参考文献:

[1] LEE H M, CHUNG K S, JIN S H, *et al*. A fault tolerance service for QoS in grid computing[C]//Proc of International Conference on Computational Science. Berlin: Springer-Verlag, 2003: 98-109.

(下转第 3925 页)

(上接第 3907 页)

[2] DOULAMIS N, DOULAMISN A, LITKE A, *et al*. Adjusted fair scheduling and non-linear workload prediction for QoS guarantees in grid computing[J]. Computer Communications, 2007, 30(3): 499-515.

[3] KYRIAZIS D, TSERPES K, MENYCHTAS A, *et al*. An innovative workflow mapping mechanism for Grids in the frame of quality of service[J]. Future Generation Computer Systems, 2008, 24(6): 498-511.

[4] CASTILLO C, ROUSKAS G N, HARFOUSH K. Online algorithms for advance resource reservations[J]. Journal of Parallel and Distributed Computing, 2011, 71(7): 963-973.

[5] 王嫚,徐惠民. 基于 QoS 的动态网格资源分配策略研究[J]. 微电子学与计算机,2005,22(12): 65-69.

[6] 张伟哲,方滨兴,胡铭曾,等. 基于信任 QoS 增强的网格服务调度算法[J]. 计算机学报,2006,29(7):1157-1166.

[7] 赵建峰,曾文华,刘敏,等. 多 QoS 约束下网格工作流调度的克隆选择算法[J]. 模式识别与人工智能,2011,24(5):713-722.

[8] 孙伟峰,覃振权,李明楚,等. QIACO:一种多 QoS 约束网格任务调度算法[J]. 电子学报,2011,39(5):1115-1120.

[9] 王磊,夏阳,史强,等. 网格环境下基于 QoS 的协作型任务调度算法研究[J]. 小型微型计算机系统,2011,32(8):1643-1646.

[10] DOGAN A, ÖZGÜNER F. Scheduling independent tasks with QoS requirements in grid computing with time-varying resource prices[C]// Proc of the 3rd International Workshop on Grid Computing. London: Springer-Verlag, 2002: 58-69.

[11] SABATA B, CHATTERJEE S, DAVIS M, *et al*. Taxonomy for QoS specifications[C]//Proc of the 3rd International Workshop on Object-Oriented Real-Time Dependable Systems. Washington DC: IEEE Computer Society, 1997: 100.

[12] University of Melbourne. The cloud computing and distributed systems (CLOUDS) laboratory[EB/OL]. http://www.cloudbus.org/gridsim/.

[13] REDDY S R. Market economy based resource allocation in grids[D]. Kharagpur, India: Indian Institute of Technology, 2006.

[14] YU Jia, BUYYA R, THAM C K, *et al*. A novel architecture for realizing grid workflow using tuple spaces[C]//Proc of the 5th IEEE/ACM International Workshop on Grid Computing. Washington DC: IEEE Computer Society, 2004: 119-128.

[15] AFZAL A, DARLINGTON J, McGOUGH A. QoS-constrained stochastic workflow scheduling in enterprise and scientific grids[C]//Proc of the 7th IEEE/ACM International Workshop on Grid Computing. Washington DC: IEEE Computer Society, 2006: 1-8.

[16] 张伟哲,刘欣然,云晓春,等. 信任驱动的网格作业调度算法[J]. 通信学报, 2006, 27(2): 73-79.

[17] ALI S, SIEGEL H J, MAHESWARAN M, *et al*. Task execution time modeling for heterogeneous computing systems[C]//Proc of the 9th Heterogeneous Computing Workshop. Washington DC: IEEE Computer Society, 2000: 185.