一种网格资源优化分配方案

林东岱,姜中华
(中国科学院软件研究所信息安全国家重点实验室,北京 100080)

摘 要: 本文提出了基于线性规划的网格异构资源分配问题的建模和求解方法.该方案综合考虑了资源分配问题的资源共享、作业优先级、作业对多种资源的依赖以及算法自身的策略等多种因素和约束条件.然后提出了网格环境下对独立作业进行网格资源分配的网格服务架构.实验表明基于线性规划的资源分配方法在速度和精确性两方面都是有效的,并且能保持高作业吞吐量.基于网格服务的架构也使该系统具有可扩展性和可伸缩性.

关键词: 资源分配; 网格服务; 作业; 优化
中图分类号: TP309 **文献标识码**: A **文章编号**: 0372-2112 (2008) 05-0875-05

A Grid Resource Optimal Allocation Scheme

LIN Dong-dai, JIANG Zhong-hua
(State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing 100080, China)

Abstract: Firstly, a linear programming based method is presented for modeling and solving the resource allocating problem in grid environments with heterogeneous resources. The approach discussed here regards resource sharing, job priorities, dependencies on multiple resource types, and algorithm specific policies. A grid service style architecture is then put forward for allocating of independent jobs with resources in a grid environment and a prototype implementation is described. The performance results show that linear programming based approach for resource allocating is efficient in speed and accuracy and can satisfy high job throughput rates. Also, the grid service style architecture makes the system scalable and extendable.

Key words: resource allocation; grid service; job; optimization

1 引言

随着网格环境越来越流行,资源和作业的规模和多样性也随之增加.由于网格的跨组织的特点,网格资源不受集中控制,它们可以随时加入和离开网格.同时,网格环境下作业提交与分配比专用并行和机群环境更具不可预测性.因此网格环境下的资源与作业的动态分配在网格管理中起着重要作用.资源分配机制必须考虑到资源需求和资源本身的动态因素,同时也必须考虑系统级资源优化策略.若同时将吞吐量、负载平衡和资源使用代价作为系统级优化目标,则将多个异构资源分配给独立作业是一个 NP 完全(NPC)问题.因此,迫切需要一种能适应资源和作业需求动态变化的网格资源分配算法.

作业调度和资源分配是分布式/并行计算的研究热点[2].但已有的研究工作大多只考虑一种类型资源(处理器),且忽略处理器间的差异(即同构的),没有考虑异构资源共享.目前资源匹配的方法有代价优先、时间优先或者优先级优先等,这些方法都存在一些不足.Maheswaran[8]研究将作业映射到异构资源的问题,比较了几种启发式分配的性能,但仅限于一种类型资源,没有考虑资源同时被多个作业共享的情况.Raman[1,9]和 Liu[6,7]提出了解决资源分配问题的建模方法和算法,工作集中在为一个具有复杂资源协作需求的一个作业发现优化资源.由于没有同时考虑多个作业,从而难以达到全局优化的目的.Kumar[5]对具有随机失败情况的高可用服务的资源分配进行了建模.而这些高可用服务依赖多种类型的资源,而且需要始终满足这些条件.然而该模型没有考虑作业到达时刻和作业需求的动态性.Wolsey[12]阐述了使用整数规划技术解决在一种特定类型资源(如计算资源)的作业调度问题的优势.然而该方法并不适合网络环境,原因是它不能解决如下挑战:(1)网格中的作业和资源具有异构性.(2)作业可能连续到达,资源可能争用.(3)一个网格作业的运行往往需要多个资源.仅当满足作业最低要求的资源都找到时,一个分配才是成功的.(4)网格中的资源通常被并发运行的多个作业所共享,而这些作业的提交和完成是异步的.

本文提出了一种由异构资源组成的网格环境的资源分配方法,本质是将资源分配问题建模为一个线性规划问题,进而求解优化分配方案.该方法能处理作业的资源需求、资源的限制和作业的优先级策略,适用于作业动态异步到达情况.本文也提出了实现该方法的资源

收稿日期:2006-06-15;修回日期:2007-06-06
基金项目:国家 973 重点基础研究发展规划(No. 2004CB318004);国家自然科学基金(No. 60673069);国家 863 高技术研究发展计划(No. 2007AA01Z447)

分配系统的架构,并进行了性能分析,它能适用于大规模网格计算环境.

2 资源调度模型

线性规划是解决如下特定类型问题的一种技术:使一个目标函数取最大值,且满足一些限制条件,其中目标函数和限制条件都是线性表达式.下面我们提出基于线性规划技术的资源分配模型,并讨论了这些模型在解决资源分配问题时是如何克服网格环境中特定挑战的.

假定网格环境下作业集合中的作业相互独立,且作业到达顺序未知.一个作业由作业请求、一个可执行文件、某些输入数据(或者是数据的引用)以及一个输出四个部分组成.这个请求规定了运行这个作业的需求和首选条件.每个作业有一个优先级.为描述方便,定义作业为作业请求和可执行文件的绑定.

2.1 基本模型

一个线性规划模型包括输入参数、变量、对变量取值的一个限制集合和一个目标函数.该模型的目标是求每个变量的值,使其满足所有规定的限制条件,而且使目标函数取最大值或最小值.

将资源分配问题建模成一个线性规划问题,其模型由如下四部分组成:

(1) 输入参数(J, R, T, P, N, U):

$J=\{j_1, j_2, \cdots, j_m\}$ 表示需要分配的作业集合;

$R=\{r_1, r_2, \cdots, r_n\}$ 表示可用的网格资源集合;

$P=\{p_1, p_2, \cdots, p_m\}$ 表示作业的优先级;

$T=\{t_1, t_2, \cdots, t_k\}$ 表示资源类型集合,每个资源 r 的类型是集合 T 的一个子集;

$N=\{n_{(t,r)} \mid r\in R \wedge t\in T\}$ 表示资源 r 类型 t 的能力,本文将能力数字化,如计算机 CPU 利用率为 33 %,CPU 主频为 3 GHz 等.

$U=\{n_{(j,t)} \mid j\in J \wedge t\in T\}$ 表示资源需求集合. $u_{(j,t)}$ 表示作业 j 对类型 t 的资源的需求量.

(2) 输入变量集合($X \cap Z$):

$X=\{x_{(j,r)} \mid j\in J \wedge r\in R \wedge x_{(j,r)}\in Z_2\}$,如果作业 j 与资源 r 进行了分配,则变量 $x_{(j,r)}=1$,否则取值 0.

$Z=\{z_j \mid j\in J \wedge z_j\in Z_2\}$,如果作业 j 需要的所有资源都分配成功,则变量 $z_j=1$,否则取值 0.

(3) 最大化/最小化 f:

目标函数 f 是优化目标,如成功分配的作业数.需要针对优化目标,定义特定的目标函数.

(4) 满足的限制条件(CR, CJ, CM):

$CR=\{CR_r \mid r\in R\}$,其中 CR_r 是资源 r 的能力限制.

$CJ=\{CJ_j \mid j\in J\}$,其中 CJ_j 表示作业 j 对它需求的多个资源的限制条件,被称为完全分配限制.

$CM=\{CM_r \mid r\in R\}$,其中 CM_r 是资源 r 对作业的需求.

给定资源 R 和作业集合 J,该模型的目标是找到满足目标函数 f 条件的最佳资源分配方案.通过该模型,本文刻画了带优先级的作业、资源的总能力以及可用能力.同一资源可能有多个能力限制,如一台机器限制最多 6 个并发作业,并且内存不超过 1 GB. $n_{(r,t)}$ 用来刻画资源类型的能力最大值,$u(j,t)$ 表示作业 j 对类型 t 的资源消耗.一组变量 $X\cup Z$ 表示一个分配方案.变量 $x_{(j,r)}$ 表示作业 j 与资源 r 进行了分配,z_j 表示作业 j 是否分配了需要的所有资源.通过本模型,可将资源分配问题转化为求一组变量的值,使目标函数 f 取最大值或最小值.模型中的限制条件对可行方案进行了限制.由于存在多个可行方案,目标函数 f 被用来度量可行方案的质量.

本文的模型可用于表达广泛的资源分配问题.如果将作业和资源参数实例化,方案就成为一个特定的分配问题.

2.2 依赖的初始化

一个作业可能依赖多个资源,每一个资源都有特定的特征以及运行该作业的可用能力.把作业对资源的需求称为依赖.这样一个作业可能有多个依赖.

许多资源可能同时满足一个依赖条件,把这些资源定义为该依赖的等价集合.为方便起见,将作业 j 的依赖 d 的等价集合表示为 $E_{(j,d)}$.在等价集合中的资源是分配该依赖的候选分配.

在线性规划中,首先根据作业的需求将可用的资源 r 划分成等价类.然而,对于一个作业集合,由于需要满足完全分配限制、资源的能力限制和目标函数限制,因此不能简单的将资源随机分配给作业.

2.3 完全分配限制

如果作业请求多个资源,完全匹配限制必须保证作业需要的所有资源都得到满足,否则就不进行分配任何资源.对 $\forall j\in J$,完全分配限制被表达为一个线性方程组:

$$\sum_{r\in E(j,d)} x_{(j,r)} = z_j \qquad (1)$$

注意:如果作业 j 所有需要的资源得到匹配,则 $z_j=1$,否则 $z_j=0$.如果资源 r 与作业 j 匹配,则 $x_{(j,r)}=1$,否则 $x_{(j,r)}=0$.在式(1)的左边,等价类 $E_{(j,d)}$ 上资源对应的 $x_{(j,r)}$ 和等于为依赖 d 匹配的个数.该作业被匹配到所有需要的资源之后,确保对作业 j 的每一个依赖的匹配是唯一的;只要有一个依赖不能被满足,则不匹配任何依赖.

2.4 资源能力限制

资源能力限制确保分配所有作业的资源的总和不超过整个可用资源的能力总和.给定资源 r 和作业 j,如

果 j 和 r 匹配成功,则作业 j 将消耗资源 r 的类型 t 的能力为 $u_{(j,t)}$,否则为 0. 这样一个作业在一个资源上的能力 t 消耗可被表达成线性表达式 $u_{(j,t)} * x_{(j,r)}$. 资源的能力限制可表达成如下的不等式组:

$$\sum_{j\in J} u_{(j,t)} * x_{(j,t)} \leq n_{(r,t)}$$

其中 $r \in R, t \in T$ 为资源 r 的能力. 这个不等式组表达了在所有资源上的能力限制.

2.5　目标函数

目标函数刻画了有多种可行解存在的情况下匹配方案的质量. 匹配算法用目标函数 f 选择最优方案. 本文建模了如下四种目标函数.

(1) 吞吐量(Throughput)

该目标函数定义了一次分配过程中获得资源的作业数. 当该目标函数取最大值时,分配过程将找到一个匹配尽可能多作业的方案. 线性表达式 $f = \sum_{j\in J} z_j$ 来反映吞吐量. 考虑到作业的优先级 p_j,目标函数 f 表达为 $\sum_{j\in J} p_j z_j$. 这样优先级高的作业有更多的调度机会.

(2) 资源代价(Cost)

该目标函数定义为分配成功后所使用资源的数量. 在某些情况下,用户可能希望在尽可能少的资源上运行一批作业. 例如资源所有者对资源收费,希望将这些作业限制在一个最小的资源集合上,以降低费用. 当目标函数取最小值,则找到使用尽可能少资源的一个分配.

为了计算使用资源的数量,对 $\forall r \in R$,定义一新变量 $s_r \in Z_2$,且定义如下两个不等式:

$$\sum_{j\in J} x(j,r) \geq s_r \tag{2}$$

以及

$$\sum_{j\in J} x_{(j,r)} \leq s_r * |J| \tag{3}$$

其中 $|J|$ 表示参与匹配的作业数.

若资源 r 没有分配给任何作业,则该不等式保证 $s_r = 0$,否则 $s_r = 1$. 对于资源 r, $\sum_{j\in J} x(j,r)$ 等于分配给该资源的作业数. 如果没有作业分配到该资源,则其值为 0. 在这种情况下,基于不等式(2),有 $s_r = 0$,原因是如果使不等式成立,必须满足 $s_r \leq 0$. 相反,如果资源 r 分配给某些作业,则 $\sum_{j\in J} x(j,r) \geq 1$. 基于不等式(3), $s_r \geq 0$ 满足第 2 个条件,所以有 $s_r = 1$. 被分配的资源数量表示为如下的线性表达式:

$$f = \sum_{r\in R} s_r$$

(3) 负载平衡(Load Balance)

该目标函数 f 的含义为负载最重的资源上空闲能力的百分比. 使该目标函数取最大值,这个匹配过程将作业从负载最重的资源移动到负载较轻的资源上,从而达到某种程度的负载平衡.

对 $\forall r \in R$,定义一个新变量 $g_{(r,t)} \in [0,1]$ 表示在资源 r 上能力类型 t 已使用的百分比. 我们修改 2.4 节的资源能力限制表达式,以便计算变量 $g_{(r,t)}$ 的值.

对 $\forall r \in R, t \in T$,有

$$\sum_{j\in J} u_{(j,r)} * x_{(r,t)} = n_{(r,t)} * g_{(r,t)}, g_{(r,t)} \geq 0, g_{(r,t)} \leq 1$$

成立. 等式左边是所有作业消耗的能力总和. 因此 $g_{(r,t)}$ 是资源使用的百分比.

对 $\forall r \in R, t \in T$,下列不等式成立:

$$f \leq 1 - g_{(r,t)}$$

不等式的右边表示资源 r 的能力类型 t 空闲百分比. 由于负载最重的资源的空闲能力最少,所以 f 不大于右边的值.

(4) 资源的首选等级(Rank)

该目标函数定义了所有被匹配资源的等级总和. 作业可以提供一些选择满足基本条件的等级,等级越高越可能被匹配. 例如一个作业更喜欢 CPU 速度更高的机器,则 CPU 速度就是划分等级的标准. 我们可以通过如下的线性表达式来表达这个目标函数.

$$f = \sum_{\substack{j\in J \\ r\in R}} q_{(j,r)} * x_{(j,r)}$$

其中 $q_{(j,r)}$ 是作业 j 给资源 r 的首选等级. 如果使目标函数取最大值,能找到包含尽可能多的用户喜欢的资源匹配方案.

用户可以配置分配过程,选择这四个目标函数之一,也可以修改已有目标函数,或创建特有目标函数来适应特定需求,从而寻求优化分配方案.

3　资源分配系统

本节讨论我们提出的资源分配系统. 系统首先接受一批作业,然后将资源分配问题转化成线性规划问题,最后生成一个基于给定目标函数 f 的优化分配方案. 系统构架如图 1 所示.

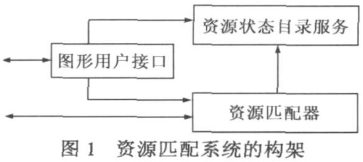

图 1　资源匹配系统的构架

系统由三个模块组成,图形用户接口用 Web 方式实现,另外两个模块用网格服务实现.

(1) 资源状态目录服务

资源状态目录服务存储所有资源的软实时状态. 实现如图 2 所示. 它使用 LDAP 目录存储资源的最新状态,并且按资源类型组织资源. 该服务提供了一个访问接口,允许客户查询资源的状态,也允许管理员管理资源元数据,例如定义新的资源类型,加入和删除资源等.

(2) 资源匹配器

资源匹配器是分配系统的决策部分. 它的结构如图 3 显示. 功能如下:(a) 它通过网格服务接口接收匹配请求;(b) 查询资源状态目录服务,获得当前的资源状态;(c) 根据实际的作业和资源数据实例化第 2 节提出的模型;(d) 提交该模型到一个线性规划求解器,以便获得该模型中变量的值;(e) 映射这些变量的值到一个匹配方案;(f) 将匹配结果返回给客户端.

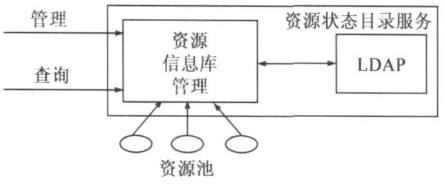

图 2 资源状态目录服务的结构

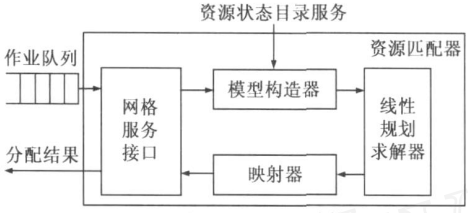

图 3 资源分配器的结构

在实现中,使用了标准的建模语言 AMPL[3] 表示第 2 节提出的模型,并且使用了功能强大的线性规划求解器 CPLEX(www.ilog.com). 由于求解器的输出是 0/1 变量,使用映射器将结果变量变成更有意义的值. 例如,如果 $x_{(j,r)}=1$,则可将其变换为资源 r 与作业 j 匹配成功.

(3) 图形用户接口

GUI 接口提供了 Web 接口以便用户提交作业、在资源分配器中配置目标函数并且管理资源状态信息库. 由于资源状态目录服务和资源分配服务都是网格服务,因此很容易与其他网格服务集成. 而且体系结构也不限于现在的实现. 例如,对资源状态目录服务进行简单的包装,就可将资源状态目录服务替换成其他信息系统如 MDS[10] 等.

4 性能分析

为了验证体系结构和设计的有效性,本节讨论该原型实现,并给出性能结果.

资源状态服务和资源分配器作为网格服务被部署在 Globus Toolkit 4[11] 上,而 GT4 本身被安装在 Fedora Core Linux 4 (FC4) 上. 图形用户接户口用于任务管理和作业提交目的使用 JSP 实现,并部署在同一台机器的 Tomcat 5 上. 用户可使用 Web 浏览器提交任务请求,也可直接与资源分配器交互. 用作信息库的 LDAP 目录使用 OpenLDAP 实现,安装在另一台 FC4 上. 在实验中,目标函数被配置为吞吐量,客户是一个运行在支持 JDK1.4 系统上的 Java 程序.

实验定义了四种资源类型:(1) 计算服务器,属性包括网络位置、操作系统版本号,CPU 类型,CPU 速度,最大和当前可用计算能力;(2) 文件系统,属性包括网络位置、类型、大小和当前可用空间;(3) 数据库,属性包括网络位置、类型、最大和当前可用的连接数;(4) 网络子系统,属性包括网络域名、类型、最大和当前可用带宽. 实验定义了 5 个计算服务器、5 个文件系统、4 个数据库和 5 个网络子系统. 属于同一类型资源的最大和可用的能力各不相同.

一个独立的客户程序来生成负载和提交作业,客户生成一批作业,并提交至作业分配器,然后接收分配结果. 不断重复该过程,该负载生成器被配置用于生成随机需求的作业,但这些作业具有如下限制:每一个作业依赖一个计算机系统,而对其他类型的资源(如文件系统、网络和数据库) 有 0 个或多个依赖. 对一个依赖,其能力需求在一个特定范围是随机决定的.

实验每次改变提交的作业数量,并且客户记录从提交一批作业开始到返回分配结果的时间间隔. 这种间隔被称为作业分配器的响应时间(ART). 对于给定作业数 m,实验重复 20 次,响应时间取 20 次匹配时间的平均值. 表 1 显示了作业数分别为 200、300 和 400 的作业组的平均响应时间. 注意,由于这些作业是一些随机的作业请求,其结果负载是由各种不同需求的作业组合产生的,结果在不同的作业组中呈现随机性. 据此,给定作业组的平均响应时间应该能看作资源匹配器在给定资源下匹配 m 个作业时的期望性能的估计值.

结果显示系统能在几十秒钟内完成数百个作业的匹配. 服务时间分成 3 部分:(1) 网格服务,它是花费在客户和网格服务之间的通信上的时间;(2) 求解,花费在求解资源匹配的混合整数规划模型的时间;(3) 目录查询,花费在查询资源状态目录获得资源状态所花的时间.

表 1 匹配 m 个作业花费的时间(ms)

作业数量	总时间	网格服务	求解	目录查询
200	18372	5293	1200	11881
300	24434	5689	1319	17426
400	31843	5948	1378	24517

结果显示,大多时间花费在网格服务和目录查询上. 对包含不同作业的作业组,通信时间几乎相同,原因是作业提交都使用了同一网格服务调用,仅参数不同. 其次目录查询时间随作业数量呈线性增长,原因是实现中对作业的每个依赖,资源分配器都进行一次查询,其性能可通过在资源分配器中缓存查询结果改进,减少查询次数. 求解优化分配方案的过程花费不超过 2 秒,结果表明用线性规划建模和求解网格资源分配问题从时

间角度来说是有效的.

5 结论

本文提出将网格资源分配建模成一个线性规划问题,并且可以用相应的算法求解.讨论了建模资源的依赖关系、能力需求和限制、资源的首选等级和作业优先级.同时提出了四种全局目标函数:最大化吞吐量、最大化以优先级为权的吞吐量、最小化资源代价和在资源集合上的负载平衡.本文也提出了相应的资源分配构架,实现了相应的作业调度和资源管理系统.系统适合由相互独立作业的资源分配.允许系统管理员动态配置全局目标,适合典型的大规模网格计算环境.系统行为具有动态性,因此绝对的优化方案是不必要的.只要不偏离预定优化目标太远,则被认为是可接受的.这种系统更重要的一个评价标准应该是其可扩展性,即随着资源的数量和作业的到达率提高,系统不应该成为一个瓶颈.实验表明,本文提出的核心资源匹配算法从速度和精确性两方面来说都是有效的.

本文提出的资源分配问题考虑到多种因素和策略,从这个意义上说,我们考虑解决的是通用的资源分配问题.然而,该方案还存在一些限制,作业是相互独立的;每个作业对每种资源只有一个依赖,这排除了需要多重资源的并行作业情况;作业是非抢占式的,同时也不考虑公平性.依赖于当前的负载和资源的可用性,资源分配器可能不分配那些即使是空闲的资源,而且无限期的继续下去,这可通过作业调度的高级预留和回填技术解决.

参考文献:

[1] Raman R, Livny M, Solomon M. Matchmaking: distributed resource management for high throughput computing[A]. Proc of the 7th IEEE International Symposium on High Performance Distributed Computing[C]. San Francisco: IEEE Computer Society, 1998. 140 - 146.

[2] Feitelson D, Rudolph L, Schwiegelshohn U. Parallel job scheduling-a status report[A]. Proc of the 10th Workshop on Job Scheduling Strategies for Parallel Processing, LNCS 3277 [C]. Heidelberg: Springer-Verlag, 2005. 1 - 16.

[3] Fourer R, Gay D, Kernighan B. AMPL: A Modeling Language for Mathematical Programming[M]. 2nd Ed, California: Thomson Brooks/ Cole Press, 2003.

[4] Ibarra O, Kim C. Heuristic algorithm for scheduling independent tasks on nonidentical processors[J]. Journal of the ACM, 1977, 24(2): 280 - 289.

[5] Kumar V, Naik V. Modeling the global optimization problem in highly available distributed environments[A]. Proc. of the 4th Applied Mathematical Programming and Modeling Conference (APMOD) 2000[C]. London: Brunel University, 2000. 17 - 24.

[6] Liu C, Yang L, Foster I, Angulo D. Design and evaluation of a resource selection framework for grid applications[A]. Proc of the 11th IEEE International Symposium on High Performance Distributed Computing[C]. San Francisco: IEEE Computer Society, 2002. 63 - 81.

[7] Liu C, Foster I. A Constraint Language Approach to Grid Resource Selection[R]. Chicago: University of Chicago, 2003.

[8] Maheswaran M, et al. Dynamic matching and scheduling of a class of independent tasks onto heterogeneous computing systems[A]. Proc. of the 8th Heterogeneous Computing Workshop [C]. San Francisco: IEEE Computer Society, 1999. 30 - 44.

[9] Raman R, Livny M, Solomon M. Policy driven heterogeneous resource co-allocation with gangmatching[A]. Proc of the 12th IEEE International Symposium on High Performance Distributed Computing[C]. Washington: IEEE Computer Society, 2003. 80 - 89.

[10] Czajkowski K, Fitzgerald S, Foster I, Kesselman C. Grid information services for distributed resource sharing[A]. Proc of the 10th IEEE International Symposium on High Performance Distributed Computing [C]. San Francisco: IEEE Computer Society, 2001. 81 - 94.

[11] Foster I. Globus toolkit version 4. software for service-oriented systems[A]. Proc of International Conference on Network and Parallel Computing (NPC '05) [C]. Heidelberg: Springer-Verlag, 2005. 2 - 13.

[12] Wolsey L A. Mixed integer programming formulations for production planning and scheduling problems[A]. Proc of the 12th International Symposium on Mathematical Programming [C]. Boston: MIT Press, 1985.

作者简介:

林东岱　男,1964 年生于山东冠县,中国科学院软件研究所研究员,博士生导师.主要从事密码理论、安全协议、网格计算、符号计算与软件设计方面的研究工作.
E-mail: ddlin@is.iscas.ac.cn

姜中华　男,1972 年生于湖北省襄樊,博士.主要研究领域为密码理论与技术、网格计算.
E-mail: jzh@is.iscas.ac.cn