采用遗传-退火算法的网格依赖任务可信调度

王洪峰[1] 朱 海[1,2]

(周口师范学院计算机科学与技术学院 周口 466000)[1] (西安电子科技大学计算机学院 西安 710071)[2]

摘 要 针对异构网格环境下的依赖任务调度问题面临的安全性挑战,综合考虑网格资源节点的固有安全性和行为安全性,构建一个网格资源节点身份可靠性度量函数和行为表现信誉度评估策略;同时为了确立任务安全需求与资源节点安全属性之间的隶属关系,定义了安全效益隶属度函数,从而建立了一个网格任务调度的安全可信模型。以此为基础,定义任务需求表示模型和网格资源拓扑模型,提出一种安全可信的网格任务调度新模型。为求解该模型,在遗传算法的基础上,设计新的进化算子即改进的交叉算子、内部交叉算子及作为变异的迁移算子,同时引入模拟退火算法增加搜索精度,从而提出了一种新的遗传-退火算法。仿真实验表明,在相同条件下,该算法比同类算法在调度长度、安全可信值及收敛性等方面具有更好的综合性能。

关键词 网格计算,任务调度,安全可信模型,进化算子,遗传-退火算法

中图法分类号 TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.6.056

Trusted Scheduling of Dependent Tasks Using Genetic-annealing Algorithm under Grid Environment

WANG Hong-feng[1] ZHU Hai[1,2]

(Department of Computer Science and Technology, Zhoukou Normal University, Zhoukou 466000, China)[1]

(School of Computer Science and Technology, Xidian University, Xi'an 710071, China)[2]

Abstract Given the security challenges faced for the dependent task-scheduling problem under heterogeneous grid environment, considering grid nodes' inherent security and behavioral security, we built a function to measure each node's identical reliability and a strategy to assess its behavioral credibility. Meanwhile, in order to establish the affiliation between the security requirement and security attributes of each task, an affiliation function of security benefits was defined. Thus, a security trusted task-scheduling model under grid environment was built in this paper. On this basis, with the task's requirement model presented and grid resources' topology model introduced, a new security trusted grid task-scheduling model was proposed. To solve this model by using genetic algorithm, we designed several new genetic operators, including improved crossover operator, crossover operator within each individual and migration operator which is taken as mutation operator, and at the same time, with a view to increase search precision, the simulated annealing algorithm was introduced, by which we further proposed a new genetic-annealing algorithm. The simulation results show that compared with similar algorithms under the same conditions, the proposed algorithm has a better overall performance in terms of scheduling length, security trusted value, convergence and other aspects.

Keywords Grid computing, Task-scheduling, Security trusted model, Evolution operator, Genetic-annealing algorithm

网格将互联网中的计算资源、存储资源、信息资源、知识资源等多种资源组织在一个统一的框架下[1],为各种复杂的应用任务提供服务资源,因此应用任务在各种资源间的有效调度就成为网格应用系统获得高性能的关键因素之一。然而,网格环境的异构性、分布性、开放性、不确定性及动态性等特征,对传统的调度策略提出了新的挑战。任务在网格平台调度执行时,除了满足传统调度策略中的时间性能需求外,还必须面临网格环境的开放特性可能造成敏感数据在执行或传输过程中被泄密、篡改或冒名顶替等风险[2,3];同时由于网格资源节点的行为属性失效[3],任务需要重新调度或延迟执行等。近年来有不少学者开始关注此问题,如文献[4]对同构网格环境下的实时任务调度应用融入了安全因素,但其没有考虑网格节点异构性和不确定性等因素;文献[5]针对网格资源节点的不确定性,参考社会人际关系信任模型建立了网格节点信任推荐机制,有效提高了任务在信任方面的服务质量需求,但是对网格资源节点本身的固有安全性欠缺考虑;文献[6]对网格资源管理中信任机制和调度机制分离的缺陷提出了一种解决方案,但是对获得的网格资源信任值并没有给出

到稿日期:2014-06-22 返修日期:2014-09-30 本文受国家自然科学基金资助项目(61103143),中国博士后科学基金资助项目(2012M512008),河南省科技厅科技发展计划重点科技攻关项目(142102110152),河南省高校科技创新人才支持计划项目(2012HASTIT032),河南省教育厅科学技术研究重点项目指导计划基础前沿项目(14B520057)资助。

王洪峰(1981—),男,硕士,讲师,主要研究方向为分布式并行计算与智能优化算法,E-mail:40577025@qq.com;朱 海(1978—),男,博士,副教授,主要研究方向为新型网格计算、任务调度及人工智能,E-mail:zhu_sea@163.com。

量化形式，在网格具体应用中无法实施。针对相关研究工作存在的不足，本文提出了一个网格任务调度的安全可信模型。该模型综合考虑了异构网格环境下资源节点的固有安全性和行为可靠性，根据资源节点本身所采用的加密算法、Hash 函数及身份认证技术类型构造了资源节点的身份可靠性度量函数，同时根据资源节点历史行为表现构建了节点的信誉度动态评估策略，并且定义安全效益隶属度函数使任务安全需求和资源节点安全属性间的关系得以确立。

当前网格任务调度的研究大多针对独立任务或元任务的特殊形式[7,8]，忽视了任务间的数据关联与优先约束关系，不能反映应用任务间的实际特征。具有依赖关系的任务调度在传统调度策略中的研究由来已久[9]，但通常较少考虑任务间的通信关系以及资源节点的异构性带来的链路竞争等影响。典型依赖任务调度模型大都是建立在图的基础之上[10,11]，也就是任务图优先模型。根据任务图的基本信息与处理单元本身及其拓扑结构的基本信息是否在应用任务调度前获得、已经调度好的任务是否可以迁移等因素，将任务调度分为静态调度和动态调度两大类。本文对具有依赖关系的静态网格任务调度问题给出了任务需求模型和资源拓扑模型的定义，然后基于提出的安全可信模型将不同网格任务的时间性能和安全性能等需求和网格资源服务水平结合起来，使网格任务调度在满足任务间依赖关系的约束下将任务调度长度和安全效益值达到最优，从而建立了一个网格依赖任务可信调度的优化新模型。

把复杂的应用任务合理调度到网格资源节点上使其有效执行，已被证明是一个 NP 难完全问题。在已往研究中，产生了许多调度算法，如 Min-min 算法、Max-min 算法、DLS 算法、遗传算法、模拟退火算法及粒子群算法等[12-14]。如 Vincenzo 介绍了一种基于遗传算法的资源调度算法，其目的是尽可能地提高资源的使用率和吞吐量[15]。另外，Abrahanm 等介绍了模拟退火算法等进化算法在网格资源调度中的应用[15]。但他们没有考虑任务之间的依赖关系。本文针对具有依赖关系的任务调度问题，提出了一种新的遗传-退火算法。该算法在列表编码的基础上，首先对初始解群的构造给出了一种相对均衡的生成方法，然后设计了新的遗传算子即改进的交叉算子、内部交叉算子及作为变异的迁移算子，同时引入了模拟退火算法增加算法的搜索精度，最后对算法进行了仿真实验，其结果表明该算法具有较好的综合性能。

1 安全可信模型

网格调度系统中安全可信性主要包括网格资源节点的身份可靠性和行为可信性，前者主要衡量网格节点对应用任务数据的保密性、完整性和真实性等固有安全计算服务保障程度，后者则为网格资源节点的外部行为表现的一种反映。

1.1 资源节点的身份可靠性度量函数

网格资源节点的身份可靠性的确保主要是由其采用的算法或方法决定：如保密性一般主要依赖不同加密算法（如 IDEA、DES、RC5、Blowfish 和 RC4 等）实现，预防任务在网格环境中执行或传输时暴露给未经授权的其它用户或进程，避免信息泄露；完整性主要通过不同的 Hash 函数（如 MD4、RIPEMD、SHA-1 和 Tiger 等）来实现，确保任务在网格平台下执行时不会被非法的用户修改或篡改；真实性使用不同的数字签名技术（HMAC-MD5、HMAC-SHA-1 和 CBC-MAC-AES 等）对通讯实体身份的真实性进行鉴别，保证所处理的任务是由合法的用户提交的。下面以保密性为例，说明如何合理设计网格资源节点的保密性安全水平值。假定在网格环境下所有资源节点共采用了 8 个加密算法[17]，如表 1 所列。将性能最低的 IDEA 加密算法对应的保密性安全水平值假定为 1，则其它任一网格资源节点 r_u 上加密算法的安全水平值根据其性能可由式(1)计算获得：

$$SL^e_{r_u}=13.5/\mu^e_{r_u}(k),1\leqslant k\leqslant 8 \tag{1}$$

表 1 加密算法及其安全水平值

加密算法	算法性能(kB/ms)	安全水平值
SEAL	168.75	0.08
RC4	96.43	0.14
Blowfish	37.50	0.36
Knufu/Khafre	33.75	0.40
RC5	29.35	0.46
Rijndael	21.09	0.64
DES	15.00	0.90
IDEA	13.50	1.00

这样，根据加密算法的不同性能将其安全值设定为 0.08 到 1 之间的值。可以看出，加密算法的保密性安全水平值与它的性能成反比例关系，安全水平值越低其性能一般越高。这也符合一般逻辑思维，在同等条件下，没有人会选一个安全水平低且其性能也低的加密机制。

同理，网格资源节点 r_u 根据其采用的 Hash 函数和数字签名技术类型，得到其完整性和真实性水平值分别为 $SL^g_{r_u}$ 和 $SL^a_{r_u}$。

定义 1（节点的身份可靠性） 节点的身份可靠性指其上执行任务时所能够提供的保密性、完整性和真实性等安全服务保障能力。对任一网格资源节点 r_u，若只考虑保密性、完整性和真实性等安全性能，则其身份可靠性安全水平值可用下式度量：

$$\begin{aligned}&SL_{r_u}=\omega_1 SL^e_{r_u}+\omega_2 SL^g_{r_u}+\omega_3 SL^a_{r_u}\\&\text{s.t.}\sum_{i=1}^{3}\omega_i=1,\omega_i\geqslant 0\\&0\leqslant SL^k\leqslant 1,k\in\{e,g,a\}\end{aligned} \tag{2}$$

式中，ω_i 表示不同安全服务的权重，权重越大表示其相应的安全服务越重要。

1.2 资源节点的信誉度评估策略

在异构网格环境调度系统中，任务调度器根据网格资源节点的历史行为表现（包括最近成功执行任务的时间、先前任务执行获得的评价和节点资源累计使用率等）来评估节点的行为可信度。每一个节点都有其相应的信誉度，其值的高低主要表征节点在过去执行任务的行为表现，反映节点的可信性程度。

定义 2（节点的信誉度） 节点的信誉度主要表征资源节点的历史行为表现特征，定义为一个随时间变化的函数，除了与网格资源节点的累计使用率有关，还与节点在累计执行任务获得的信任评价相关。资源节点 r_u 在网格系统中的信誉度评估策略为：

$$\begin{aligned}&RL_{r_u}=\alpha\times e^{-TB_{r_u}}+\beta\times RS_{r_u}+\gamma\times RU_{r_u}\\&\text{s.t.}\ 0<\alpha,\beta,\gamma<1\\&\alpha+\beta+\gamma=1\end{aligned} \tag{3}$$

式中，α 为时间衰减权重系数，β 为网格资源节点 r_u 累计执行调度任务获得的信任评价权重系数，γ 为资源累计使用率权重系数；TB_{r_u} 为网格节点 r_u 最近一次成功执行任务到目前为

止的时间间隔，$TB_{r_u}>0$；RU_{r_u} 为节点 r_u 累计使用率，可用节点 r_u 累计执行任务时间与累计活跃时间的比值来表示；RS_{r_u} 为节点累计执行任务获得的信任评价，包含 3 个因子即时间、身份可靠性安全值和行为信誉度，其计算方法为：

$$RS_{r_u}=\chi\times\frac{\sum_i(c_{io}-c_i)}{\sum_i c_{io}}+\delta\times\frac{\sum_i(s_i-s_{io})}{\sum_i s_{io}}+\eta\times\frac{\sum_i(r_{io}-r_i)}{\sum_i r_{io}} \tag{4}$$

在式(4)中，系数 χ、δ 和 η 分别为时间、身份可靠性和信誉度对节点 r_u 信任评价值的贡献率，$0<\chi,\delta,\eta<1$ 且 $\chi+\delta+\eta=1$。其中 c_{io} 为预估计完成时间，c_i 为实际花费时间；s_{io} 为资源节点 r_u 声明的安全值，s_i 为实际获得的安全值；同样，r_{io} 为资源节点 r_u 调度时声称的信誉度值，r_i 为实际获得的信誉值。显然，当任务 t_i 在分配节点 r_u 上执行时，若花费的时间越短，获得的安全值和信誉度值越高，则其对节点 r_u 的评价就越高，这样资源节点 r_u 的获得信任评价就越好，对最终信誉度的贡献就越大。

1.3 安全可信模型

网格环境下任务调度除了追求时间性能指标外，需要尽可能将任务分配调度到安全可信属性高的网格资源节点执行。不同的网格任务在调度过程中对资源节点身份可靠性及行为可信性等安全值的需求一般不同，根据其需求类型可将任务的安全需求分为强安全需求关系和弱安全需求关系。强安全需求关系指任务调度的安全需求必须得到满足，否则此任务丢弃；弱安全关系指在分配调度时尽可能满足任务的安全需求，否则可降低任务请求者的安全需求，但其安全效益值随之下降。本文网格任务调度安全可信模型的研究属于后者，任务对资源节点的安全属性需求满意度通过隶属度函数获得的安全效益函数值进行量化。

定义 3(身份可靠性隶属度) 身份可靠性隶属度表征网格任务对资源节点的保密性、完整性及真实性等固有安全水平的满意程度。网格任务 t_i 对资源节点 r_u 的身份可靠性满意度可由下式定义的隶属度函数获得。

$$SV(t_i,r_u)=\begin{cases}1, & \text{if } TS_{t_i}\leqslant SL_{r_u}\\ 1-\dfrac{SL_{r_u}-TS_{t_i}}{max\_level-SL_{r_u}}, & \text{else}\end{cases} \tag{5}$$

式中，TS_{t_i} 为网格任务 t_i 的可靠性安全需求，SL_{r_u} 为网格资源节点 r_u 的身份可靠性安全水平值。

定义 4(行为可信性隶属度) 行为可信性隶属度表征网格任务对资源节点的历史行为表现特征的满意程度。网格任务 t_i 对资源节点 r_u 的行为可信性满意度可参照文献[6]定义其隶属度函数为：

$$RV(t_i,r_u)=\begin{cases}1, & \text{if } TR_{t_i}\leqslant RS_{r_u}\\ 1-\dfrac{\exp(-(RS_{r_u}+1-TR_{t_i}))}{\exp(-(1-TR_{t_i}))}, & \text{else}\end{cases} \tag{6}$$

式中，TR_{t_i} 为网格任务 t_i 的对资源节点行为可信性的安全需求，RS_{r_u} 为网格资源节点 r_u 的外部行为信誉度安全水平值。

定义 5(安全可信满意度) 网格任务调度安全可信满意度指网格任务对向其提供服务资源节点的安全可靠属性满意程度，其值由隶属度函数获得的综合安全效益值表征。网格任务 t_i 被分配到资源节点 r_u 上执行，获得的综合安全可信效益值定义为：

$$Dep(t_i,r_u)=\omega_1 SV(t_i,r_u)+\omega_2 RV(t_i,r_u)$$
$$\text{s. t. } 0\leqslant\omega_1,\omega_2\leqslant 1 \tag{7}$$
$$\omega_1+\omega_2=1$$

式中，ω_1，ω_2 分别代表任务 t_i 在节点 r_u 上获得的身份可靠性和行为可信性满意度的权重，$SV(t_i,r_u)$ 和 $RV(t_i,r_u)$ 的值可分别由式(5)和式(6)获得。

定义 6(安全可信模型) 网格任务调度的安全可信模型定义为一个四元组(R,T,A,D)，其中 R 代表网格资源节点集合，即服务提供者；T 代表应用任务集合，即服务需求者，A 代表安全属性集合，即包括任务调度的身份可靠性和行为可信性，D 代表任务调度的安全可信满意度，即包括身份可靠性隶属度函数和行为可信性隶属度函数。

在网格安全可信模型中，综合考虑了网格资源节点的固有安全性和行为安全性，采用安全效益函数使任务安全需求和资源节点安全属性间的关系得以确立，将调度机制和安全机制较好地融合起来。下面以此模型为基础，首先定义了网格任务需求表示模型和资源拓扑结构模型，然后提出了网格依赖任务可信调度的优化模型。

2 可信依赖任务调度模型

2.1 任务需求模型

为了便于对网格应用任务需求进行描述，采用任务图的形式建立应用任务的需求模型。任务图通常为有向无环图(DAG)，是根据网格应用的数据处理流程建立的应用任务模型。图的顶点表示网格应用的子任务，边表示任务之间的约束关系。顶点的权值表示网格应用子任务的计算量和安全需求，而边的权值为节点间的通信数据传输量和数据在网格环境传输时的安全需求。通常一个网格应用往往需要由多个子任务协同完成，其任务图如图 1 所示。

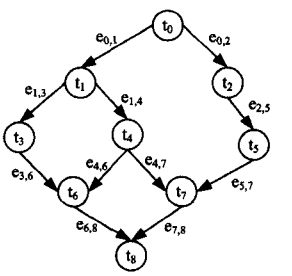


图 1 网格任务图模型

在网格任务图中，将没有父节点的任务节点称为入口节点，没有孩子节点的任务节点称为出口节点。如果任务图中有多个入口(或出口)节点，则可以将它们用权重为 0 的边连到一个伪入口(或出口)节点上。在没有收到所有父节点发来的消息之前，任务节点不能开始执行。

定义 7(任务需求表示模型) 任务需求表示模型可由一个四元组 TG 表示：$TG=(V_T,E_T,W_T,U_T)$。其中，$V_T=\{t_0,t_1,\cdots,t_{n-1}\}$ 是任务图的顶点集合，t_i 表示第 i 个子任务，$|V_T|=n$ 表示网格应用中子任务的个数；$E_T=\{e_{t_i,t_j}\mid t_i\in V_T,t_j\in V_T \text{ and } i\neq j\}$ 为任务图边的集合，表示任务之间依赖关系集合，e_{t_i,t_j} 表示任务 t_i 和 t_j 之间具有依赖关系；$W_T=\{w_{t_i}\mid t_i\in V_T\}$ 为任务图顶点权值集合，$w_{t_i}=(a_{t_i},s_{t_i})$ 为一个二元组，表示顶点 t_i 的权值，a_{t_i} 和 s_{t_i} 分别表示任务 t_i 的计算量和安全可

信需求；U_T 为任务图边权值集合 $U_T=\{u_{t_{i,j}} \mid e_{t_{i,j}} \in E_T\}$，$u_{t_{i,j}}=(a_{t_{i,j}}, s_{t_{i,j}})$为一个二元组表示边 $e_{t_{i,j}}$ 的权值，$a_{t_{i,j}}$ 和 $s_{t_{i,j}}$ 分别表示任务 t_i 和 t_j 之间通信量和安全可靠性需求。

2.2 网格资源模型

任务调度的网格资源系统由具有一定拓扑结构的网格节点组成，其中不同的网格节点拥有不同的执行能力和安全水平，同时节点之间的链路也具有不同的单位通信能力和安全可靠性。

根据网格资源的拓扑结构可以用资源图的形式建立网格资源的拓扑结构模型，对资源之间的连接状况进行描述。资源图类似于任务需求模型中的任务图(如图 2 所示)，主要用于描述资源节点性能和资源间的通信关系。资源图的顶点和边分别表示网格资源节点本身和资源节点之间的链路，顶点的权值表示网格资源节点的计算能力和安全水平，而边的权值为节点间的单位通信能力和链路安全服务水平。

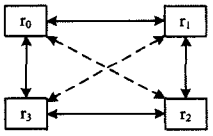


图 2 网格资源(环形)拓扑结构图模型

定义 8(网格资源表示模型) 网格资源系统拓扑结构图可以用一个四元组 RG 表示：$RG=(V_R, E_R, W_R, U_R)$。其中，$V_R=\{r_0, r_1, \cdots, r_{m-1}\}$为资源图的顶点集合，$r_u$ 表示第 u 个资源节点，$|V_R|=m$ 表示网格资源图中资源的数目；$E_R=\{e_{r_u,v} \mid r_u \in V_R, r_v \in V_R \text{ and } u \neq v\}$为网格资源图边的集合，表示资源之间具有通信关系的集合，$e_{r_u,v}$ 表示网格资源 r_u 和 r_v 之间的通信关系；$W_R=\{w_{r_u} \mid r_u \in V_R\}$为资源图顶点权值集合，$w_{r_u}=(a_{r_u}, s_{r_u})$为一个二元组，表示资源节点 r_u 的权值，a_{r_u} 和 s_{r_u} 分别表示资源 r_u 的单位时间处理能力和安全可信水平；$U_R=\{u_{r_{u,v}} \mid e_{r_{u,v}} \in E_R\}$为资源图边权值集合，$u_{r_{u,v}}=(a_{r_{u,v}}, s_{r_{u,v}})$为一个二元组，表示边 $e_{r_{u,v}}$ 的权值，$a_{r_{u,v}}$ 和 $s_{t_{i,j}}$ 分别表示资源节点 r_u 和 r_v 之间的单位通信能力和链路安全服务水平。

2.3 可信依赖任务调度优化模型

在网格任务图和资源拓扑结构图表示模型基础上，网格任务分配调度策略由任务-资源映射图表示，其中图的顶点表示任务-资源的映射对，顶点权值为任务的计算量、安全可信需求以及相应资源节点的计算能力和安全可信水平；边表示任务之间的约束及通信关系，边权值为具有约束关系相邻任务间的通信量、通信安全需求及对应资源节点间的单位通信能力和链路安全性。图 3 所示为图 1 的任务图在图 2 网格资源拓扑结构中的一个映射调度方案。

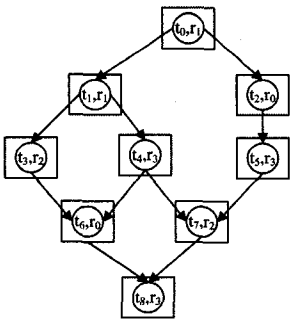


图 3 网格任务-资源分配图模型

定义 9(网格任务-资源分配图表示模型) 网格任务资源分配映射图可以用一个四元组 $T\text{-}RAG$ 表示：$T\text{-}RAG=(V_L, E_L, W_L, U_L)$。其中，$V_L=\{l_0, l_1, \cdots, l_{n-1}\}$为任务-资源分配图顶点集合，$l_x=\{(t_i, r_u) \mid 0 \leqslant i < n, 0 \leqslant u < m, r_0, r_1, \cdots, r_{m-1}\}$，$(t_i, r_u)$表示任务 t_i 映射到网格资源节点 r_u 上执行；$E_L=\{e_{l_x,l_y} \mid l_x=(t_i, r_u), l_y=(t_j, r_v) \text{ and } x \neq y\}$为任务-资源分配图边集合，表示资源之间具有通信关系的集合，e_{l_x,l_y} 表示网格资源 r_u 上的任务 t_i 和资源 r_v 上的任务 t_j 之间的通信关系；$W_L=(W_T, W_R)$为任务-资源分配图顶点权值集合，为一个二元组，W_T 和 W_R 分别为任务图和资源拓扑结构图中相应顶点的权值；$U_L=(U_T, U_R)$为任务-资源分配图边权值集合，也为一个二元组，表示图中顶点 $l_x=(t_i, r_u)$和顶点 $l_y=(t_j, r_v)$连接边上的权值集合。

定义 10(任务调度系统可信安全效益值) 网格任务调度系统获得的可信安全效益值表征网格应用在资源节点上调度的可信满意程度，可由下式获得：

$$S(G_K)=\frac{\sum_{i=0}^{n-1}\sum_{u=0}^{m-1} x_{iu} Dep(t_i, r_u)}{N} \tag{8}$$

其中，$Dep(t_i, r_u)$可由式(7)得到，为任务 t_i 在网格节点 r_u 上获得安全可信效益值；x_{iu} 为决策变量，其定义如下：

$$x_{iu}=\begin{cases}1, & \text{任务 } t_i \text{ 分配到节 } r_u \text{ 上执行}\\ 0, & \text{否则}\end{cases} \tag{9}$$

定义 11(任务调度长度) 网格任务调度长度为任务-资源分配图模型中从入口节点到出口节点的最大路径长度。

$$\begin{aligned}L(G_K)&=\max\{L(P_S)\}\\&=\max\{\sum_{V_S \in V_L} L(V_S)+\sum_{E_S \in E_L} L(E_S)\}\end{aligned} \tag{10}$$

式中，$L(P_S)$表示图 T-RAG(简称图 G)中任意某路径 P_S 从入口节点 S_e 到出口节点 S_o 的时间花费；$L(E_S)$为路径 P_S 中间的数据通信时间，若任务 t_i, t_j 分别分配到资源 r_u 和 r_v，则有 $L(E_S)=c(t_i, t_j)/d(r_u, r_v)$，$c(t_i, t_j)$为任务 t_i 和 t_j 之间的数据传输量，$d(r_u, r_v)$为资源 r_u 和 r_v 之间的通信能力；$L(V_S)$为路径 P_S 中某一节点的时间花费，若任务 t_i 分配到网格资源节点 r_u 上，则有 $L(V_S)=a(t_i)/b(r_u)+ST(t_i, r_u)$，$ST(t_i, r_u)$为任务 t_i 在网格节点 r_u 上获得安全可信效益值的时间花费，可由下式得到：

$$ST(t_i, r_u)=\sum_{k \in \{a,e,g\}} ST(SV^k(t_i, r_u)) \tag{11}$$

其中，式(11)的计算方法可查阅文献[18]。

定义 12(可信调度优化模型) 网格任务的可信调度模型由需求模型、网格资源拓扑结构模型、安全可信模型及分配调度方案组成，可表示为一个四元组(T, R, S, MAP)，其中 MAP 为分配调度方案集合，即 $MAP=\{map_1, map_2, \cdots, map_n\}$。分配调度方案为二元组 $map=(a, \varphi)$，其中 a 表示将网格应用任务 T 中 n 个任务映射到资源节点 R 上的分配方案(可由网格-任务资源分配图表示)，φ 表示单个网格节点 r_u 上的调度方案。则异构网格系统下任务调度的目标就是寻找任务需求模型与资源拓扑结构模型之间的映射调度方案，在满足任务间依赖关系的约束下，使网格任务的调度长度和获得的安全可信效益达到最优。其数学模型表示如下：

$$\begin{aligned}&\min(Z)=\omega_1 L(G_K)+\omega_2(1-S(G_K))\\&\text{s. t.}\sum_{u=0}^{m-1} x_{iu}=1, i=0, 1\cdots, n-1\\&f(t_j, r_v)-f(t_i, r_u) \geqslant \frac{c(t_i, t_j)}{d(r_u, r_v)}+\frac{a(t_j)}{b(r_v)}+ST(t_j, r_v)\end{aligned} \tag{12}$$

式中，$Z=\{z_1,z_2,\cdots,z_k,\cdots\}$，$z_k$ 为第 k 个分配调度方案；$L(G_K)$和 $S(G_K)$为网格任务的调度长度和安全满意度，可分别由式(8)和式(10)获得；$0\leqslant\omega_1,\omega_2\leqslant1$，$\omega_1+\omega_2=1$，根据任务特性及要求不同，可以通过调整权重 $\omega_k(k=1,2)$比例对任意目标进行重新优化和满足不同网格用户的倾向性需求；第一个约束条件保证每个网格任务有且仅分配到一个资源节点上完成，第二个约束条件满足任务执行时的依赖关系。

3 遗传-退火算法设计

遗传算法具有天生的并行性和全局寻优等特性。但由于优化过程中种群缺乏多样性等问题容易导致过早收敛，使得搜索效率降低，因此很难保证找到问题在可行域上的最优解，而可能只是局部最优解，这一现象俗称“早熟”。“早熟”现象的产生与遗传算法所使用的子代选择遗传方法有关。以经典的轮盘赌选择方法为例，应用这一方法时，在新种群中产生的染色体数目与前一代中染色体的适应度大小成正比，这就造成局部有相对优势的个体数目急剧增加，充斥新种群。通过交叉操作和变异操作产生的新的适应度更好的个体由于数量不多，优势不明显，不能在后代遗传操作中得到很好的推广，发生“早熟”现象。在遗传算法的后期，适应度逐步趋于一致，搜索速度减慢甚至停滞不前。所以遗传算法能否搜索到全局最优解，在很大程度上依赖于初始种群的个体分布情况以及遗传操作中的交叉变异算子。

相比于遗传算法，模拟退火算法已被证明是一种以概率1收敛于全局最优解的启发式算法，其收敛值与初始值无关，具有很好的全局收敛性。但在采用基本模拟退火算法求解全局规模较大的问题时，需要较高的初始温度、较慢的降温速率、较低的终止温度，以及各温度下足够多次的抽样，这将直接导致算法的收敛时间较长，效率低下。

针对网格任务调度 NP 难问题，采用遗传算法实现时，其种群规模有限，收敛效率高，但容易“早熟”；模拟退火算法虽具有优秀的全局收敛性，但为获得优秀解，算法的收敛时间较长，效率低下。鉴于两种算法的优缺点互补性非常高，本文在对基本遗传算法的交叉和变异算子进行改进的基础上，引入模拟退火算法增加种群个体的多样性，以获得尽可能接近问题最优解的 Pareto 最优解。

3.1 初始种群生成

初始解群的生成方法是，将应用任务图(记为 DAG 图)中的所有结点集合按高度值划分成 $h'+1$ 个子集 $DAG(i)$，$0\leqslant i\leqslant h$，h 是 DAG 图中最大的高度值。随机地将子集 $DAG(i)$的所有任务指派到 m 个网格节点上。然后，将所有指派到同一网格节点上的任务按高度值作升序排列，得到一个合法的调度。只要重复执行这个过程，就能得到一定群体规模的初始解群。

$DAG(i)$中的所有任务是可以并行执行的，如果能将它们相对均匀地指派到不同的处理机上，则能提高整个任务的并行性，从而使调度完成时间更短的。不妨设$|DAG(i)|=p'$，则处理机 p_i 上被指派的任务个数 q_i 为

$$q_i=\mu+\delta,1\leqslant i\leqslant m \tag{13}$$

其中，$\mu=\lfloor p'/m\rfloor$为平均任务个数，δ 为随机项。如可定义 $\delta=((random(2\times k+1)-k))/2$，$k$ 为常数(通常令 $k=\mu$)，函数 $random(x)$生成一个 0 到$(x-1)$的随机整数，即 $q_i\in[\mu-\mu/2,\mu+\mu/2]$，显然应有$\sum_{i=1}^{m}q_i=p'$。

3.2 遗传算子设计

3.2.1 改进的交叉算子

遗传算法以交叉算子为主要的搜索算子。在任务分配与调度问题中，由于受限于各任务间的约束依赖关系，对于两个合法的解，通过常规遗传算法的标准杂交算子形成的新解可能是非法的。本文称文献[19]的交叉算子为标准交叉算子(SCX)。对于给定的两个调度 S_1 和 S_2，SCX 操作过程为：(1)随机生成一个数 n'，$0\leqslant n'\leqslant h'$，在 S_1 和 S_2 的所有任务列表 $L_i(1\leqslant i\leqslant m)$中选择杂交点，以便能将每个任务列表分成两部分。杂交点的选择要满足两个条件，1)所有紧跟杂交点后面的任务的高度值与 n'不等，2)紧接杂交点前的任务的高度值都与 n'相等。(2)交换对应任务列表 $L_i(1\leqslant i\leqslant m)$上的后半部分任务，生成新的两个解 S_1'和 S_2'，如图 4 所示。文献[19]证明了合法解通过 SCX 交叉算子生成的新解也是合法的。

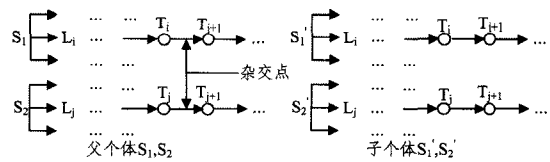


图 4 交叉算子 SCX 或改进的杂交算子 MCX

SCX 对杂交点选择时，第 2 个条件有可能得不到满足。为了有效地搜索解空间，对杂交点选择的条件进行调整后，得到改进的杂交算子(improved crossover operator，MCX)。MCX 的操作过程与 SCX 相似，只是杂交点的选择要满足两个条件，1)所有紧跟杂交点后面的任务的高度值与 n'不等，2)紧接杂交点前的任务的高度值小于或等于 n'。由于对杂交点选择的条件修改不会破坏原来任务高度值的升序排列，因此生成的新调度也是合法的。

MCX 对 SCX 中杂交点选择的条件进行适当的放松，这样杂交操作发生的概率相对 SCX 来说更大，从而能进行更有效的搜索，即基于给定的两个点，在其周围的搜索空间中，MCX 能比 SCX 搜索更多的点。设 $MCX(S_1,S_2)$表示 MCX 基于给定点 S_1 和 S_2 可能解搜索空间的点数，$SCX(S_1,S_2)$表示 SCX 基于给定点 S_1 和 S_2 可能搜索解空间的点数，则有 $MCX(S_1,S_2)\geqslant SCX(S_1,S_2)$。

3.2.2 内部交叉算子

标准交叉算子是从两个已知的解中产生出新解。在分配与调度问题中，基于同一调度内的两个任务相互交换，称为内部杂交算子(internal crossover，NCX)。NCX 算子的操作过程如下：(1)生成一个任意的随机数 n'，$0\leqslant n'\leqslant h'$，再从一个调度 S_1 中随机选择两个任务列表 L_i 和 L_j，根据 n'在 L_i 和 L_j 中确定杂交点；(2)将 L_i 和 L_j 中杂交点右半部分相互交换，生成一个新的调度 S_1'。

定理 1 如果杂交点的选择满足如下条件：(1)紧跟杂交点后面的任务 T_i，$h(T_i)\leqslant n'$；(2)紧邻接杂交点前面的任务 T_i，$h(T_i)\leqslant n'$，杂交后生成的调度是合法的调度。

证明：由于内部杂交算子只发生在一个调度内，显然，各个子任务在调度内出现的唯一性可以得到保证。现在只需要

证明在杂交操作后，各个子任务的依赖关系不会被破坏。如图5所示，杂交前，对 S_1 而言，$h(T_i)\leqslant n'\leqslant h(T_{i+1})$，$h(T_j)\leqslant n'\leqslant h(T_{j+1})$。杂交完成以后，对 S_1' 而言，$h(T_i)\leqslant n'\leqslant h(T_{i+1})$，$h(T_j)\leqslant n'\leqslant h(T_{j+1})$，这种优先关系没有改变，满足高度值的升序排列，因而是合法的调度。证毕。

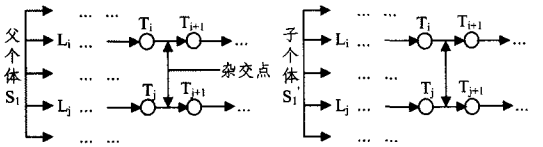


图5 内部交叉算子

本文任务分配与调度算法中，将NCX作为主要的搜索算子。这样做的主要原因有两点：首先，对于SCX算子，当两个分配调度 S_1 及 S_2 一样时，SCX算子和MCX算子都不能产生新的解，但是NCX算子可能产生新解；其次，在某些情况下，NCX算子能产生SCX算子或MCX所不能产生的新解。NCX具有SCX与MCX所不具备的局部爬山功能。

考虑到图5所示的情况，对SCX和MCX而言，必有 $h(T_i)<h(T_{i+1})$，且 $h(T_j)<h(T_{j+1})$。反之，如果 $h(T_i)=h(T_{i+1})$，或 $h(T_j)=h(T_{j+1})$，那么，图中所给的就不可能是杂交点。由于NCX算子中对杂交点的选择条件较弱，即使 $h(T_i)=h(T_{i+1})$，或 $h(T_j)=h(T_{j+1})$，任务的交换仍然可以完成。

3.2.3 迁移算子

遗传算法中的变异操作是一种辅助算子，在解群局部收敛时，通过变异算子的突变作用来保持解群一定的多样性。在分配调度问题中，变异操作就是在一个调度内，随机交换两个具有相同高度值的子任务。显然，这种交换是合法的。除了这种子任务的交换，期望有一种更有效的变异算子，在一个调度内，允许一个子任务从一个处理机上迁移到另一个处理机上。本文称这种操作为迁移(migration)。迁移操作的过程为：

(1)生成一个任意的随机数 n'，$0\leqslant n'\leqslant h'$；

(2)计算调度S中每个任务列表的高度值等于 n' 的子任务数目 q_i'，$1\leqslant i\leqslant m$；

(3)从最大 $q_i'(1\leqslant i\leqslant m)$ 所在的任务列表中随机选择一个子任务 $T_k(h(T_k)=n')$，将 T_k 迁移到最小 $q_i'(1\leqslant i\leqslant m)$ 所在的任务列表中。

一旦确定了要迁移的子任务，下一步就是确定将该要迁移的子任务插入到另一任务列表的什么位置上，以确保迁移后的调度是合法的。有如下的定理：

定理2 迁移操作中，设紧邻插入点的前面和后面的子任务分别为 T_{i_j} 和 $T_{i_{j+1}}$，如果 $h(T_{i_j})\leqslant n'$ 且 $n'\leqslant h(T_{i_j}+1)$，则子任务 $T_k(h(T_k)=n')$ 迁移进来后，整个调度仍是合法调度。

该定理的证明比较简单，从略。

引入迁移算子至少有3方面优点：第一，迁移算子能间接地起到交换两个子任务的作用，包括在一个调度内的同一列表中交换两个等高度值的子任务；第二，多个串行执行的子任务通过迁移操作后，有可能获得并行执行的效果，能充分利用多处理机系统的并行性；第三，当NCX与迁移算子相互结合后，算法的搜索能力可能是充分的。即通过NCX与迁移，如能有适当的初始解群和控制参数，则算法有能力搜索到解空间的每个点。

3.3 模拟退火操作

模拟退火算法(Simulate Anneal Arithmetic)来源于固体退火原理，最早由Metropolis等于1953年提出，1983年由Kirkpatrick等将其应用于组合优化领域，其核心是在进化过程中根据Metropolis准则接受新解。其具体思想如下：

Step1 设定退火初始温度 T(需要合理选择)、初始解 S(任选)、等温循环代数 Gen、退火速率 K 等算法运行参数。

Step2 内循环。从 S 的邻域中随机选一个解 S'，计算 S 和 S' 对应的目标函数值 $E(S)$ 和 $E(S')$，如 S' 对应目标函数值较小，则接受 S' 为新解；否则以概率 $\exp(-(E(s')-E(s))/T)$ 接受 S' 为新解。若不满足内循环终止条件，重复本步骤。

Step3 外循环。降温操作 $T_{new}=K*T_{old}$。不满足外循环停止条件，则转Step2；否则算法结束，输出最优解。

3.4 遗传退火算法流程

本文涉及算法主要是融合遗传算法和模拟退火算法进行实现：采用遗传算法产生初始种群；针对每一代种群，首先采用MCX和NCX交叉算子作用于种群；然后将迁移算子作用于种群，最后对种群中的所有个体进行模拟退火操作，并构造子种群；重复此操作，直至满足终止条件得到最优解。算法流程的具体描述如下：

步骤1 设定遗传算法的最大迭代代数 $gmax$、种群规模 $popsize$、交叉概率 $crossfactor$、变异概率 $mutationfactor$，以及退火算法的初始温度 T、退火速率 K 和等温持续代数 $gensimu(gmax\gg gensimu)$。

步骤2 按3.1节所述算法进行遗传种群初始化。

步骤3 将MCX和NCX交叉算子以概率 $crossfactor$ 作用于种群。

步骤4 将迁移算子以概率 $mutationfactor$ 作用于交叉后种群。

步骤5 对变异后种群中的每个个体进行退火操作。退火算法采用“逆转两端”法生成新个体，即随机生成两个位置 L 和 $M(L<M)$，然后将 M-n 和 1-L 位置上的方案互换。再计算新个体相对于旧个体的适应度增量值 ΔE，并以概率 $\min\{1,\exp(-\Delta E/T)\}$ 接受新个体，从而产生新一代种群。

步骤6 判断解是否稳定，迭代次数是否达到 $gmax$，两者满足任何一个，执行步骤8，否则执行步骤7。

步骤7 若满足温度下降条件($gmax\%gensimu=0$)，则更新退火温度即 $T=K\times T$；不论是否满足温度下降条件都返回步骤3继续执行。

步骤8 从当前种群中选择适应度最好的个体作为目标问题的全局最优解并返回，算法结束。

3.5 算法时间复杂度分析

设 m 为网格节点数目，n 为依赖任务DAG图中的任务数，h 为DAG图的深度，遗传算法种群个数为 p，迭代次数为 q，模拟退火算法的迭代次数 r。本算法的时间复杂度计算主要包括种群初始化和遗传过程两个部分。种群初始化的时间复杂度为 $O(np)$；由于每生成1个满足条件交叉点的数量级为 h，遗传过程中交叉和变异操作的复杂度均为 $O(pmh)$，模

拟退火过程的时间复杂度为 $O(np+p^2r)$，故遗传过程的时间复杂度为 $O(pqmh+npq+p^2qr)$。由于种群初始化的时间复杂度远小于遗传过程的时间复杂度，因此本算法的时间复杂度为 $O(pqmh+npq+p^2qr)$。

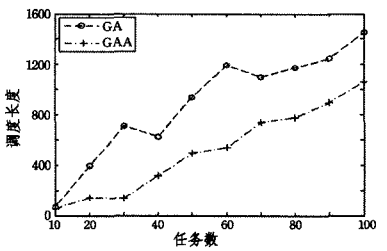
4 仿真实验与结果分析

为了验证本文提出的 GAA 算法的有效性，分别从算法的收敛性能和任务调度的调度长度和安全性等性能指标进行了仿真实验，并将其与遗传算法 GA[20] 进行了对比分析。

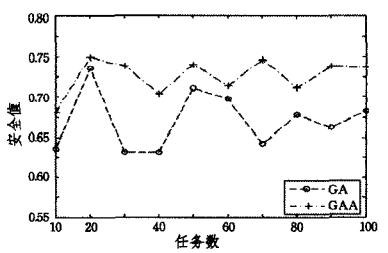
基于 C++语言设计开发了一个由组件 GridGenerator 和组件 TaskGenerator 构成的网格任务调度仿真器进行模拟。其中组件 GridGenerator 负责模拟网格环境，能随机产生具有不同性能的网格节点和不同通信能力的节点间链路；组件 TaskGenerator 主要生成不同大小具有约束关系的子任务及相应的安全可靠性需求。另外，对在一个真实网格环境下的具体应用进行了仿真实验，进一步验证了本文所提算法的有效性。

实验 1　性能指标比较

(i)$m=20$ 个网格节点，任务数 n 变化(n 为 10～100 个子任务)的情况下，对本文 GAA 算法和标准 GA 算法各运行 20 次的平均性能指标进行比较，如图 6 所示。



(a)任务调度长度比较

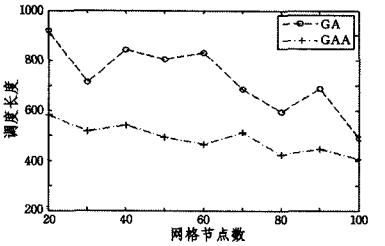


(b)任务可信值比较

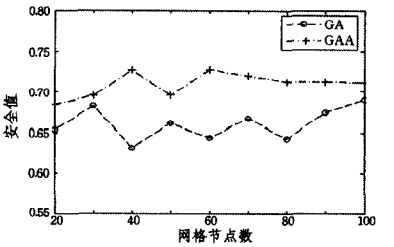
图 6　网格环境不变，任务数变化下的性能比较

网格环境不变的情况下，从图 6(a)可以看出，随着任务数的增加，任务调度长度都有所增加，其 GAA 算法明显优于遗传算法 GA，特别是随着问题规模的变大其优势更明显。从图 6(b)看出，随着任务数增加，任务对在网格节点及链路上获得较优可信值的竞争更激烈，致使任务平均信誉值都有所降低，其中在同等条件下本文提出的 GAA 算法明显占优。

(ii)任务数固定($n=60$)，网格环境节点数变化的情况下，对不同性能指标进行比较，结果如图 7 所示。从图 7(a)可以看出，随着节点数的扩充，任务调度长度都有所缩短，这主要是因为任务在调度时可以选择性能更好的网格节点执行，从而缩短了任务调度长度。从图 7(b)可以看出，网格节点数扩充时任务获得的安全值有所提升，其中遗传退火算法 GAA 在同等条件下获得的值大。



(a)任务调度长度比较



(b)任务安全值比较

图 7　任务数不变，网格环境变化下的性能比较

实验 2　收敛性比较

为了比较算法的收敛特性，图 8 给出了本文 GAA 算法和 GA 算法在 200 次迭代过程中的调度长度变化曲线。

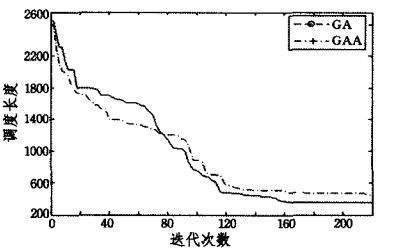
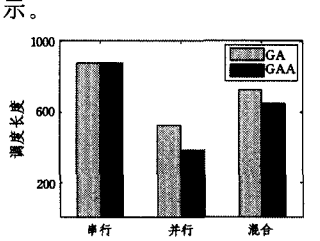


图 8　GAA 与 GA 收敛性的比较

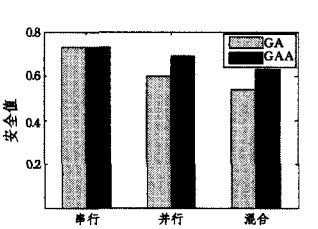
从图 8 中可以看出，在算法的初始运行阶段，GA 算法的调度长度值优于遗传退火算法 GAA，但随着进化代数的增加 GAA 的优势得以体现，其收敛速度虽然和 GA 差不多，但其值明显优于 GA。

实验 3　一个真实网格环境下的仿真实验

为了进一步验证本文所提算法的有效性，分别针对串行工作流、并行工作流、混合型 3 种典型依赖应用的 DAG 图：最大并行度为 1 且深度为 30 的流水线串行应用作业、最大并行度为 6 的 7×7 矩阵的高斯消去法和最大并行度为 7 的分子动力学代码[21]，进行了实验。在由 4 台 IBM Systemx3850 X5(每台 8 颗 CPU，每颗 CPU 8 核心，内存 256GB)搭建的集群上构建了 1 个由 60 个虚拟机组成的网格环境。网格节点的计算能力配置参数如表 2 所列，每个节点类别内部的虚拟网格节点安全值分布于{0.2，0.4，0.6，0.8，1.0}。实验主要从调度长度和任务安全值方面进行了仿真，实验结果如图 9 所示。



(a)3 种应用类型任务的调度长度比较



(b)3 种应用任务的可信值比较

图 9　真实网格环境下 3 种具体应用的性能比较

表 2　网格环境配置参数表

节点类别名称	配置参数	节点个数
COM1	2 个 CPU,4GB 内存	20
COM2	1 个 CPU,2GB 内存	20
COM3	4 个 CPU,8GB 内存	20

从图 9(a)可以看出,在总体任务量一定的情况下,串行类型任务调度长度最长,而并行工作流类任务调度长度最短;针对并行类和混合类应用,本算法相对 GA 算法明显有效,混合类性能提升约 11%,并行类性能提升约 18%。这主要是因为应用的并行度越高,GAA 优势越明显,调度长度越短。从图 9(b)看出,针对并行工作流和混合类应用,在同等条件下本文提出的 GAA 明显占优。这主要因为在遗传算法全局寻优能力较强的情况下,本文算法经过模拟退火进一步局部优化,在同等条件下能找到安全值更大的解。

由以上仿真实验可以得出:遗传算法种群的规模有限、收敛效率高,缺点是易"早熟";而模拟退火算法具有很好的全局收敛性,缺点是初始温度高、收敛效率低。将退火算法添加到遗传算法当中,使两种算法优缺点互补性高,得到更好的收敛速度和全局收敛性,实现了两种算法互补优缺、扬长避短的效果。

结束语　本文结合网格任务调度问题的具体特点,根据网格节点的行为表现,构建了网格节点的动态信誉度评估策略,并针对网格依赖任务提出其相应的可信调度模型,一定程度上弥补了相关研究工作的不足。在提出了网格依赖任务可信调度优化模型的基础上,将遗传算法和模拟退火算法相结合,提出了一种新的遗传退火算法。最后对算法进行了仿真实验,其结果表明该算法具有较好的综合性能。

参 考 文 献

[1] L Shin-Yeu, H Shih-Cheng, L Cheng-Zih. Expanding service capacities and increasing service reliabilities for the grid-based utility computing[J]. IEEE Transactions on Systems, man, and cybernetics-Part A, 2011, 41(1): 149-160

[2] Sarkar V, Khaparde S A. Improving Demand Response and Bid-Consistency of Price Outcome in the Security-Constrained Dispatch Scheduling[J]. IEEE Transactions on Power Systems, 2013, 28(8): 2433-2445

[3] Song S S, Hwang Kai, Kwok Yu-Kwong. Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling[J]. IEEE Transactions on Computers, 2006, 55(6): 703-719

[4] Gong Chen, Wang Xiao-dong, Xu Wei-qiang, et al. Distributed Real-Time Energy Scheduling in Smart Grid Stochastic Model and Fast Optimization[J]. IEEE Transactions on Smart Grid, 2013, 4(9): 1476-1489

[5] Yuan L L, Zeng G S, Jiang L L, et al. Dynamic level scheduling based on trust model in grid computing[J]. Chinese Journal of Computers, 2006, 29(7): 1217-1224

[6] Zhang W Z, Liu X R, Hu M Z, et al. Trust-driven job scheduling heuristics for computing grid[J]. Journal on Communication, 2006, 27(2): 73-79

[7] Braun T D, Slegel H J, Becj N. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems[J]. Journal of Parallel and Distributed Computing, 2001, 61(6): 810-837

[8] Xiao P, Hu Z G. Co-Scheduling Model for Independent Tasks with Deadline Constraint in Computational Grid[J]. Acta Electronic Sinica, 2011, 39(8): 1852-1857

[9] 马艳,龚斌,邹立达. 网格环境下基于复制的能耗有效依赖任务调度研究[J]. 计算机研究与发展,2013,50(2):420-429
Ma Yan, Gong Bin, Zou Li-da. Duplication Based Energy-Efficient Scheduling for Dependent Tasks in Grid Environment[J]. Journal of Computer Research and Development, 2013, 50(2): 420-429

[10] 阎朝坤,胡志刚,李玺,等. 面向可靠性-费用优化的网格任务调度模型及算法研究[J]. 计算机科学,2013,40(5):136-141
Yan Chao-kun, Hu Zhi-gang, Li Xi, et al. Reliability-Cost Optimization Scheduling Model and Algorithm in Grid[J]. Journal of Computer Science, 2013, 40(3): 136-141

[11] Du X L, Jiang C J, Xu G R. A grid DAG scheduling algorithm based on fuzzy clustering[J]. Journal of Software, 2006, 17(11): 2277-2288

[12] Xu Jin, Lam A Y S, Li V O K, et al. Chemical Reaction Optimization for Task Scheduling in Grid Computing[J]. IEEE Transactions on Parallel and Distributed Systems, 2011, 22(10): 1624-1631

[13] Sun W F, Qin Z Q, Li M C, et al. QIACO: An Algorithm for Grid Task Scheduling of Multiple QoS Dimensions[J]. Acta Electronic Sinica, 2011, 39(5): 1115-1120

[14] He X, Sun X, Laszewski G V. QoS guided min-min heuristic for grid task scheduling[J]. Journal of Computer Science and Technology, 2003, 18(4): 442-451

[15] Martino V D, Mililotti M. Scheduling in a grid computing environment using genetic algorihms[C]// The 16th Int'l Parallel and Distributed Processing Symp, 2002. USA: IEEE Press, 2002

[16] Abraham A, Buyya R, Nath B. Nature's Heuristics for Scheduling Jobs on Computational Grids[C]// The 8th Int'l Conference on Advanced Computing and Communication, 2000. India: IEEE Press, 2000

[17] Xie T, Qin X. Scheduling security-critical real-time applications on clusters[J]. IEEE Transactions on Computers, 2006, 55(7): 864-879

[18] Zhu H, Wang Y P. Grid Dependent Tasks Security Scheduling Model and DPSO Algorithm[J]. Journal of Networks, 2011, 6(6): 850-857

[19] Hou E S H, Ansari N. Genetic algorithm for multi-processor scheduling[J]. IEEE Transactions on Parallel and Distributed Systems, 1994, 5(2): 113-120

[20] Ricardo C C, Afonso F, Pascal R. Scheduling multiprocessor tasks with genetic algorithm[J]. IEEE Transactions on Parallel and Distributed Systems, 1999, 10(8): 825-837

[21] 徐雨明,朱宁波,欧阳艾嘉,等. 异构系统中 DAG 任务调度的双螺旋结构遗传算法[J]. 计算机研究与发展,2014,51(6):1240-1252
Xu Yu-ming, Zhu Ning-bo, Ouyang Ai-jia, et al. A Double-Helix Structure Genetic Algorithm for Task Scheduling on Heterogeneous Computing Systems[J]. Journal of Computer Research and Development, 2014, 51(6): 1240-1252