Journal of System Simulation

Volume 30 | Issue 11 Article 1

1-4-2019

Event Triggered Optimization Method for Dynamic Task Decomposition Mode in Cloud Fusion

Wang Yan
College of the Internet of Things, Jiangnan University, Wuxi 214122, China;

Lijun Cheng
College of the Internet of Things, Jiangnan University, Wuxi 214122, China;

Event Triggered Optimization Method for Dynamic Task Decomposition Mode in Cloud Fusion

Abstract

Abstract: Aiming at the characteristics of randomness, dynamics, diversity and uncertainty of arrival time in the cloud fusion mode, *two triggered tasks of the new tasks randomly reaching the cloud platform and the non-executable sub-tasks* are considered. *To improve the sub-tasks correlation degree and the workload balance of each group, the coupling degree of the sub-tasks is decreased, and an event-triggered mechanism (ETM) based optimization method for dynamic task decomposition mode is proposed*. On the basis of the unified tasks description, *the optimal flow of dynamic task decomposition is designed to determine the sub-tasks information flow. A multi-objective optimal model of the dynamical task decomposition based on groups is established. An improved adaptive genetic algorithm is proposed to solve the problem*. The simulation results show that the proposed method can achieve the balance of cloud-fusion task-resource allocation, improve the efficiency and balance of task assignment; the presented optimization algorithm also obtains better precision and convergence performance than traditional methods.

Keywords

cloud fusion, event-triggered, dynamic task decomposition, task grouping, adaptive genetic algorithm

第 30 卷第 11 期 系统仿真学报© Vol. 30 No. 11
2018 年 11 月 Journal of System Simulation Nov., 2018

基于事件驱动的云端动态任务分解模式优化方法

王艳，程丽军
(江南大学 物联网工程学院，江苏 无锡 214122)

摘要：针对云端融合模式下任务具有随机性、动态性、多样性、到达时刻不确定等特征，考虑*随机到达云平台的新任务、不可执行子任务两类触发事件，提出一种基于事件驱动机制(Event-triggered mechanism, ETM)的动态任务分解模式优化方法，提高组内子任务相关度、每组任务工作量均衡度、降低组间子任务耦合度。在给出任务统一描述方法的基础上，设计基于 ETM 的动态任务分解优化流程，确定子任务信息流向关系，建立基于分组的动态任务分解多目标优化模型，并提出一种改进自适应遗传算法进行求解。*仿真表明，提出的方法能够实现云端融合任务-资源分配的均衡性与高效性，且优化算法在寻优精度和收敛性能方面具有优势。

关键词：云端融合；事件驱动，动态任务分解；任务分组；自适应遗传算法
中图分类号：TP391.9 文献标识码：A 文章编号：1004-731X (2018) 11-4029-14
DOI: 10.16182/j.issn1004731x.joss.201811001

Event Triggered Optimization Method for Dynamic Task Decomposition Mode in Cloud Fusion

Wang Yan, Cheng Lijun

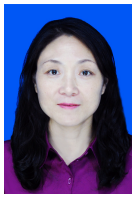
(College of the Internet of Things, Jiangnan University, Wuxi 214122, China)

Abstract: Aiming at the characteristics of randomness, dynamics, diversity and uncertainty of arrival time in the cloud fusion mode, *two triggered tasks of the new tasks randomly reaching the cloud platform and the non-executable sub-tasks* are considered. *To improve the sub-tasks correlation degree and the workload balance of each group, the coupling degree of the sub-tasks is decreased, and an event-triggered mechanism (ETM) based optimization method for dynamic task decomposition mode is proposed.* On the basis of the unified tasks description, *the optimal flow of dynamic task decomposition is designed to determine the sub-tasks information flow. A multi-objective optimal model of the dynamical task decomposition based on groups is established. An improved adaptive genetic algorithm is proposed to solve the problem.* The simulation results show that the proposed method can achieve the balance of cloud-fusion task-resource allocation, improve the efficiency and balance of task assignment; the presented optimization algorithm also obtains better precision and convergence performance than traditional methods.

Keywords: cloud fusion; event-triggered; dynamic task decomposition; task grouping; adaptive genetic algorithm

引言

在制造生产活动中，用户接到任务订单后，首先会根据客户需求，并结合自身的生产能力以及现存资源情况对订单进行分析，制定相应的生产计划，再将生产计划进一步细分为可执行的生产任务。在此过程中，确定企业内部能够独立完成的任务即本地任务，以及需要其他企业或资源协助才能完成的任务，即外协任务。传统生产模式下，对于需要进行外协的任务，任务的执行通常依赖且局限



收稿日期：2018-09-27 修回日期：2018-10-25;
基金项目：江苏省杰出青年基金(BK20160001)，江苏省高等学校优秀科技创新团队项目；
作者简介：王艳(1978-)，女，江苏盐城，博士后，教授，研究方为工业互联网智能优化制造。

于相同地域的、具备完成该任务所需生产能力和资源的企业。而在云端融合模式下，企业可通过云端融合平台终端发布相关任务请求，云平台根据任务的相关描述信息确定任务基本类型，从而对任务进一步细分，以为其匹配更优质的资源服务。云端融合平台在对云任务和云资源进行统一描述的基础上，对云任务进行分解，再根据服务需求方的任务请求信息，通过语义搜索等手段为其匹配合适的云资源服务，实现对任务和资源的统一调度。

云任务分解环节作为实现云任务-资源精准匹配的前提，是云端融合过程的重要环节，同时是实现资源高效共享的关键，引起了国内外很多学者对其机制进行研究。文献[1]运用极大-加代数方法，基于瓶颈子任务对子过程细分模式进行研究，运用 Petri 网实现瓶颈子过程的并联控制，并给出任务完成时间最短的情况下子过程的分割数目计算方法。文献[2]针对云任务分解与资源匹配环节脱节的问题，提出一种基于分层任务网络的方法对云任务分解过程进行研究；文献[3]在任务分解的基础上，分析了不同调度算法对任务分解的影响；文献[4]在任务分解的基础上研究了基于工作负载的任务调度方法对云系统性能的影响，并且在任务分解的基础上，建立了任务调度优化的多目标优化模型；文献[5]同样在任务分解的基础上，考虑云用户的个性化需求对云任务调度过程进行研究。然而，这些研究并没有对所涉及的云任务分解过程进行具体分析。

此外，云端融合模式下服务需求方提交的云任务在不同时刻随机到达云平台，具有很强的动态性特征，而上述研究文献均没有考虑云任务的动态性特点。文献[6]虽研究了基于任务分解的云服务多路径动态组合问题，但没有具体分析如何实现动态任务的分解过程；文献[7]针对云制造环境下面临信息的实时性问题，提出一种基于博弈论的柔性作业车间调度动态优化模型，而此问题同样是基于动态任务分解基础之上，但文献并没有研究怎样实现动态任务分解过程；文献[8]为解决动态云环境中任务调度问题，提出一种事件触发的动态任务调度方法，虽未对动态任务分解过程进行分析，其事件触发的思路值得借鉴。

在模型寻优方面，国内外也发展了很多相关算法。遗传算法是受生物进化过程中“优胜劣汰”规则的启发，在 20 世纪 60 年代，Michigan 大学的 John Holland 就提出了遗传算法的思想。与其他诸如蚁群算法[9]、粒子群算法[10]等寻优算法不同，遗传算法通用性强，能解决各种组合优化问题的寻优，并且在全局搜索方面表现出很好的收敛性[11-12]。

基于此，本文考虑任务的动态、随机等特征，以云端动态任务分解过程为对象，首先给出任务分解问题的统一描述；其次，以提高任务分配的均衡性和效率性为目标，设计基于事件驱动机制进行任务初步分解，并对子任务可执行性进行判断；进而，考虑分解子任务间信息交互关系，基于分组思想建立任务分解模式优化模型，并提出一种改进的自适应遗传算法进行模型求解；最后通过仿真实例验证了所提方法在提高任务-资源分配均衡性和高效性方面的有效性。

1 云端动态任务分解问题

1.1 云端动态任务统一描述

云端动态任务统一描述是任务分解和进行资源匹配的前提。设某一时刻共有 ρ 个需要外协的任务 $\{T^1,T^2,\cdots,T^\rho\}$ 到达平台，任务 $T^i(i=1,2,\cdots,\rho)$ 可由五元组描述[13]，将其统一形式化描述为：

$$T^i=\{GenInf,FunInf,ConsInf,Subtasks,StaInf\} \quad (1)$$

式中：*GenInf* 表示任务基本信息；*FunInf* 表示任务功能信息；*ConsInf* 表示任务约束信息；*Subtasks* 表示子任务集合；*StaInf* 表示任务当前状态。下面针对任务各部分具体描述信息进行具体阐述。

(1) 任务基本信息 *GenInf*

$$GenInf=\{TaskID,Name,Type,ServiceInf\} \quad (2)$$

式中：*TaskID* 表示任务编号，具有唯一性；*Name* 表示任务名称；*Type* 表示任务类型；*ServiceInf* 主

DOI: 10.16182/j.issn1004731x.joss.201811001

要提供任务提供方以及任务执行方的企业名称、联系方式等信息，任务基本信息由服务需求方和服务提供方共同写入。

(2) 任务功能信息 *FunInf*

任务功能信息描述的是任务需达成的目标。不同的任务其功能描述信息不尽相同，由服务需求方写入任务功能信息。

(3) 任务约束信息 *ConsInf*

$$ConsInf=\{Time,Quality,Cost,Service\} \tag{3}$$

式中：*Time* 表示任务的时间约束；*Quality* 表示任务的质量约束；*Cost* 表示任务的成本约束，主要包括任务执行过程中的通信成本和执行成本；*Service* 表示对服务提供方的服务约束，如服务延时率、完成率、纠纷率以及服务商的信誉度等指标应达到服务需求方的相应标准，由服务需求方写入任务约束信息。

(4) 子任务集合 *Subtasks*

$$Subtasks=\{T_1^i,T_2^i,\cdots,T_n^i\} \tag{4}$$

式中：$T_j^i(i=1,2,\cdots,\rho;\ j=1,2,\cdots n)$ 表示第 i 个任务的第 j 个子任务，当任务成功分解后由云端融合平台运营方写入该项信息。

(5) 任务当前状态 *StaInf*

$$StaInf\in\{(Waiting)\cup(\text{Execution})\cup(Failure)\cup(Finished)\} \tag{5}$$

其中，*Waiting* 表示任务处于等待状态；*Execution* 表示任务处于正在执行状态；*Failure* 表示任务由于故障处于异常状态，此时服务需求方可根据状态恢复正常需要等待的时间确定是否继续等待或是终止任务；*Finished* 表示任务处于已完成状态。用户可通过任务的当前状态信息对任务进行监控以及执行相应操作，确保任务的顺利执行。由于云端融合平台上接收的都是需要外协的任务，因此，任务当前状态信息根据任务所需服务的不同，由平台运营方或者服务需求方进行写入。

1.2 云端动态任务分解模式优化流程

云端动态任务分解模式优化的目的是基于事件驱动机制(ETM)对某时刻到达云端融合平台的 ρ 个任务进行动态分解，再对分解子任务进行分组，使得组内任务相关度最大，组间任务耦合度最小，同时各组任务工作量均衡度统一，以寻求最优的任务分组方案，方便后续平台统一为任务匹配相应资源服务，提高任务执行的实时性。动态任务分解模式优化流程如图 1 所示。

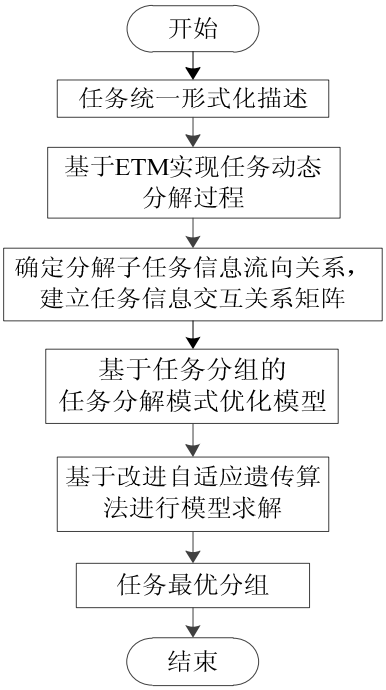


图 1 云端融合动态任务分解模式优化流程图

Fig. 1 Cloud fusion dynamic task decomposition mode optimization flow chart

结合图 1 所示的动态任务分解模式优化流程，其分解步骤具体如下：

Step 1: 对任务及分解后的子任务进行统一形式化描述；

Step 2: 基于 ETM 实现云端融合模式下的动态任务分解过程；

Step 3: 根据分解后子任务之间的信息流向关系，构建任务信息交互关系矩阵；

Step 4: 构建基于子任务分组的动态任务分解模式优化模型；

Step 5: 基于改进自适应遗传算法对动态任务分解模式优化模型进行求解；

Step 6: 确定任务最优分组。

2 基于 ETM 的动态任务分解过程

云端融合模式下，云用户跨越地理位置的限制分布于各地，体现出“分散资源集中使用，集中资源分散服务”的思想[13]。同一时间不同用户提交的任务需求各不相同，不同时间相同用户提交的任务也不尽相同。因此，云平台在同一时刻接收到的任务具有随机性兼具多样性特征，且平台中的任务时刻处于动态变化之中。对于云端融合模式下任务分解机制的研究不同于传统模式，需要着重研究云任务的动态性特征。

2.1 ETM 过程

为探究云端融合模式下动态任务分解问题，本文提出基于事件驱动(ETM)的动态任务分解模式优化方法。所设计的事件驱动机制触发事件包括两种类型事件，一种是新任务随机达云端融合平台事件，即当有新任务提交到云平台时，触发任务分解机制。一种是不可执行子任务事件，即当判定子任务不可执行时，为确保子任务的可执行性，触发任务分解机制对子任务实施进一步分解。结合图 2 对 ETM 过程进行具体分析。

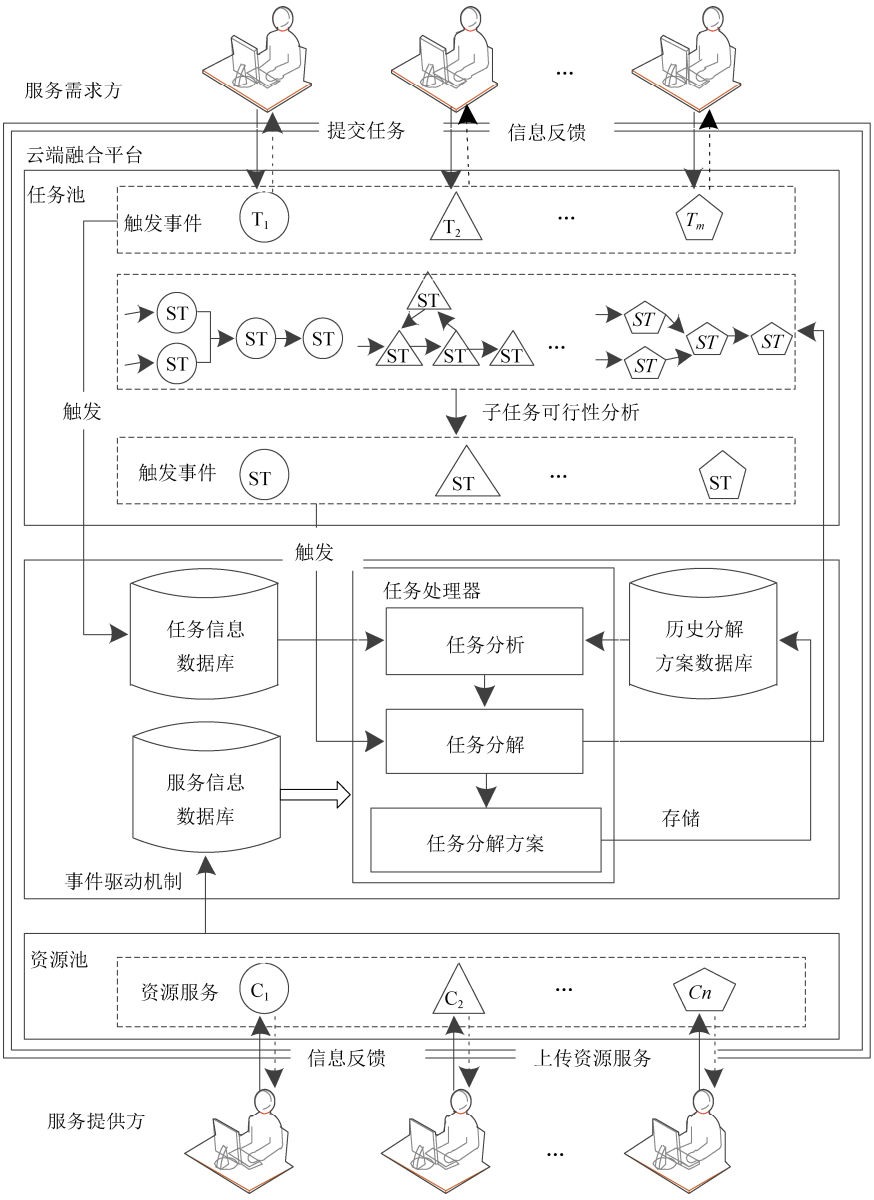


图 2 云端融合模式下基于 ETM 的动态任务分解过程

Fig. 2 ETM-based dynamic task decomposition process in cloud fusion mode

https://dc-china-simulation.researchcommons.org/journal/vol30/iss11/1
DOI: 10.16182/j.issn1004731x.joss.201811001

由图 2 可知，当有用户提交任务到达云端融合平台时，触发事件驱动机制，事件驱动机制调用任务信息数据库中的任务描述信息，同时结合服务信息库中的资源描述信息，对任务进行分析，初步确定任务类型以及是否有可匹配资源，从而确定到达任务是否满足任务可分解条件。在满足任务可分解条件的前提下，为提高任务分解的实时性，事件驱动机制调用任务历史分解方案数据库，将当前待分解任务描述信息与历史分解方案的描述信息进行匹配。若匹配到符合用户需求的历史分解方案，则直接应用历史分解方案对到达任务进行分解。否则根据任务描述信息对到达任务进行初步分解，并对分解子任务的可执行性进行判断，不可执行子任务事件将触发事件驱动机制对子任务进行再分解。当所有子任务都满足可执行性条件，则视为对当前到达任务成功地进行了分解，并将形成的分解方案存储到任务历史分解方案数据库中，供下一时段的到达任务触发事件驱动机制时调用。

2.2 子任务可执行性分析

云端融合模式下，对任务进行分解的目的是将任务分解成若干个可执行的子任务，从而降低任务复杂度，提高任务与资源的匹配效率。如果对任务进行分解后，得到的子任务不可执行，则失去了任务分解的意义。因此，可执行性是判断是否对子任务进一步进行分解的重要标准。

设 t_0 时刻到达云端融合平台的任务 T，将其基于 ETM 分解后得到的子任务集为 $\{T_1, T_2, \cdots, T_n\}$，将子任务 $T_j(j=1,2,\cdots n)$ 描述信息中的基本信息、功能信息和约束信息与云端融合平台中资源服务的相应信息进行匹配，得到一系列候选资源集合 $\{R_1^j, R_2^j, \cdots, R_k^j\}$，$R_h^j(h=1,2,\cdots,k)$ 表示任务 T 的第 j 个子任务对应的候选资源 h，k 为候选资源的个数。从以下 5 个约束指标评价子任务可执行性：

(1) 时间一致性约束

设于 t_0 时刻到达云端融合平台的云任务 T，触发事件驱动机制后，于 t_1 时刻对进行任务分解，则从任务到达云平台到得到分解的等待时间为 $\Delta t = t_1 - t_0$。则分解后得到的各子任务预计完成时间 t_j 与等待时间 Δt 之和小于服务需求方的交货时间 $T_{\max}$。时间一致性约束公式表达为

$$\Delta t + \sum_{j=1}^{n} t_j \leqslant T_{\max} \tag{6}$$

(2) 成本一致性约束

云任务分解后得到的各子任务预计执行成本 c_j 之和小于服务需求方的期望最大成本 $C_{\max}$，即

$$\sum_{j=1}^{n} c_j \leqslant C_{\max} \tag{7}$$

(3) 质量一致性约束

云任务分解得到的各子任务预计完成质量 q_j 大于服务需求方期望的最低可接受质量 Q_{mim}，即

$$q_j \geqslant Q_{\min} \tag{8}$$

(4) 资源服务评价约束

资源服务评价 F_j 包括服务延时率 D_j、完成率 F_j、纠纷率 B_j 以及服务商的信誉度 G_j，必须满足服务需求方的最小期望服务评价，即

$$F_j \geqslant F_{\min} \tag{9}$$

(5) 候选资源服务数量约束

为确保任务的顺利执行，通常在满足服务需求方基本要求的基础上，为每个子任务匹配一系列具有相同功能的候选资源，平台在候选资源集中为任务匹配最优的资源服务。设每个子任务对应 $h = N(j)$ 个候选资源，满足

$$N_j \geqslant N_{\min} \tag{10}$$

式中：$N_{\min}$ 表示要求的候选资源的最少数量。

此外，根据约束评价的重要性分配相应的权重 $\omega = \{\omega_1, \omega_2, \cdots, \omega_L\}$，$L$ 为评价指标的个数，$\omega_l(l=1,2,\cdots,L)$ 为第 l 个指标对应的权重，其数值由模糊语言变量确定[14]，对应数值见表 1。则可确定子任务 T_j 对候选资源 R_h^j 的满意度 S_{jh}^i 为

$$S_h^j = \frac{\sum_{l=1}^{L} \omega_l F_l}{\sum_{l=1}^{L} \omega_l} \tag{11}$$

式中：$F_l = \{0,1\}$，表示资源约束属性是否满足任务约束属性，其值取 0 时表示资源约束属性不能满足任务约束属性，其值取 1 时表示满足[14]。则最终

子任务可执行性 E_j 表示为

$$E_j = \max(S_1^j, S_2^j, \cdots, S_h^j) \tag{12}$$

设定可执行阈值 ξ，当 $E_j \geqslant \xi$ 时，则认为子任务是可执行的。反之，子任务不可执行，触发事件驱动机制对子任务进行再分解。

表 1 ω_l 取值对应的模糊语言变量含义

Tab. 1 Meaning of fuzzy language variables corresponding to the value of ω_l

语言评估	非常重要	很重要	重要	一般	不重要
ω_l	0.8–1.0	0.6–0.8	0.4–0.6	0.2–0.4	0.0–0.2

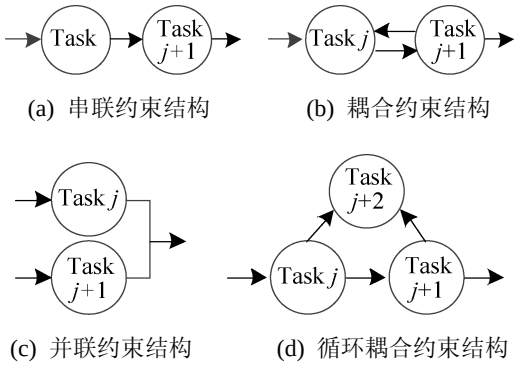
3 基于分组的动态任务分解模式优化模型

为降低分解后子任务间的信息交互程度，提高任务与资源的匹配效率和精度，对分解后的子任务进行分组，尽量使信息交互程度高的子任务归到相同的组别中，同时使得每组任务工作量均衡。因此，任务分组总的目标是达到组内任务相关度尽可能高、组间任务耦合度尽可能低，同时每组任务工作量均衡。

3.1 子任务间信息交互关系描述

无论是简单任务还是复杂任务，都可以借鉴程序设计中语句结构的思想，将复合的任务简单化。在程序设计中，一个良好的程序，无论多么复杂，都可由 3 种基本结构组成，即顺序结构、选择结构以及循环结构。类似地，任何一个任务，无论多么复杂，都可以由 4 种基本约束结构的任务构成：串联约束结构，并行约束结构，耦合约束结构以及循环耦合约束结构，4 种基本约束结构示意图见图 3。

由图 3 可知，任务间存在两种信息交互关系，一种是信息由前序子任务传递到后序子任务，即前序子任务对后序子任务的影响记作 $A=\{a_{ij}\}_{n\times n}$，其中，$0 \leqslant a_{ij} \leqslant 1$ 且 $a_{ii}=0$ 为前序子任务对后序子任务的信息影响程度的数值表示，由模糊语言变量方法确定，具体取值情况见表 2。第二种是信息由后序子任务传递到前序子任务，即后序子任务对前序子任务的信息反馈。设矩阵 $B=\{b_{ji}\}_{n\times n}$ 表示第二种信息交互关系，其中，$0 \leqslant b_{ji} \leqslant 1$ 且 $b_{ii}=0$ 为后序子任务对前序子任务的信息反馈大小的数值表示，确定方法同 a_{ij}。这两种信息交互关系组成如图 3 所示的任务 4 种基本约束结构。



(a) 串联约束结构 (b) 耦合约束结构

(c) 并联约束结构 (d) 循环耦合约束结构

图 3 任务基本约束结构类型

Fig. 3 Task basic constraint structure type

表 2 a_{ij} 取值对应的模糊语言变量含义

Tab. 2 Meaning of fuzzy language variables corresponding to the value of a_{ij}

语言评估	很强	强	一般	弱	无
a_{ij}	0.8–1.0	0.6–0.8	0.4–0.6	0.2–0.4	0.0–0.2

对于串行约束结构和并行约束结构，任务间只存在第一种信息交互关系。对于耦合约束结构和循环耦合约束结构，任务间同时存在两种信息交互关系[14]。设矩阵 $\boldsymbol{C}=\{c_{ij}\}_{n\times n}$，当任务约束结构为串行或并行约束结构时为

$$c_{ij} = \begin{cases} a_{ij} \\ b_{ji} \end{cases} \tag{13}$$

当任务约束结构为耦合或循环耦合约束结构时为

$$c_{ij} = (a_{ij} b_{ji})^{1/\tau} \tag{14}$$

其中，τ 表示发生耦合的子任务数目。则最终得到子任务间信息交互关系矩阵 $\boldsymbol{C}=\{c_{ij}\}_{n\times n}$ 为

$$\boldsymbol{C} = \begin{bmatrix} c_{11} & & & & \\ \vdots & & & & \\ c_{i1} & \cdots & c_{ii} & & \\ \vdots & \vdots & \vdots & \ddots & \\ c_{n1} & \cdots & c_{nj} & \cdots & c_{nn} \end{bmatrix} \tag{15}$$

https://dc-china-simulation.researchcommons.org/journal/vol30/iss11/1
DOI: 10.16182/j.issn1004731x.joss.201811001

3.2 任务分解模式优化模型

设 $TG=\{tg_{ik}\}_{n\times m}$ 表示子任务的分组矩阵，其中，n 表示子任务数，$m(m\leqslant n)$ 表示组别数。$tg_{ik}=1$ 表示子任务 i 属于第 k 个任务组，$tg_{ik}=0$ 表示第 k 个任务组不包含子任务 i。且一个任务只能属于其中的一个任务组，即 $\sum_{k=1}^{m}tg_{ik}=1$。

3.2.1 目标函数

(1) 组内子任务相关度

设第 k 个任务组中共包含 q_k 个子任务，则组内任务相关度为[15]

$$D_k=\frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}c_{ij}tg_{ik}tg_{jk}}{c_{q_k}^2} \tag{16}$$

$$c_{q_k}^2=\frac{q_k!}{2(q_k-2)!} \tag{17}$$

其中，c_{ij} 表示第 i 个子任务与第 j 个子任务的综合信息交互程度，$c_{q_k}^2$ 表示 q_k 个任务的组合。则 m 个组的子任务平均相关度可表示为

$$f_1=\max(D_k^m)=\max(\sum_{k=1}^{m}D_k/m)$$
$$\max(\sum_{k=1}^{m}(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}c_{ij}tg_{ik}tg_{jk}/c_{q_k}^2)/m) \tag{18}$$

(2) 组间子任务耦合度

设有两个不同的任务组，第 k 个任务组中共包含 q_k 个子任务，第 l 个任务组共包含 q_l 个子任务，则组间子任务的耦合度表示为[15]

$$D_{kl}=\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}tg_{ik}tg_{jl}}{c_{(q_k+q_l)}^2} \tag{19}$$

$$c_{(q_k+q_l)}^2=\frac{(q_k+q_l)!}{2((q_k+q_l)-2)!} \tag{20}$$

则 m 个组的组间子任务平均耦合度可表示为

$$D_{kl}^m=\sum_{k=1}^{m-1}\sum_{l=k+1}^{m}D_{kl}/c_m^2=$$
$$\frac{\sum_{k=1}^{m-1}\sum_{l=k+1}^{m}\left(\sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}tg_{ik}tg_{jl}/c_{(q_k+q_l)}^2\right)}{c_m^2} \tag{21}$$

即 $f_2=\min(D_{kl}^m)$ (22)

(3) 每组任务均衡度

设分解后 n 个子任务对应的工作量矩阵为 $EG=\{eg_{ik}\}_{n\times m}$，$eg_{ik}$ 表示第 k 个任务组中第 i 个子任务的工作量，则 m 个组的子任务平均工作量 E_a 为

$$E_a=\sum_{k=1}^{m}\sum_{i=1}^{n}eg_{ik}/m \tag{23}$$

则 m 个组的任务均衡度计算公式如式(24)所示，即当各组子任务的工作量与平均工作量 E_a 的偏差越小时，各组子任务工作量越均衡。

$$E_a^m=\sum_{k=1}^{m}(\sum_{i=1}^{n}tg_{ik}\cdot eg_{ik}-E_a)^2 \tag{24}$$

即 $f_3=\min(E_a^m)$ (25)

综上，将多目标优化函数转换为单目标优化函数，设定权值 δ_1、δ_2、δ_3，满足 $\delta_1+\delta_2+\delta_3=1$，为保证各目标函数的公平性，取 $\delta_1=\delta_2=\delta_3=1/3$，转换后的单目标函数表述为

$$f=\delta_1\max f_1+\delta_2(1-\min f_2)+\delta_3(1/(\min f_3)) \tag{26}$$

3.2.2 约束条件

(1) 可执行性约束

约束条件主要包括时间一致性约束、成本一致性约束、质量一致性约束、资源服务评价约束以及候选资源服务数量约束，这 5 种约束条件在 2.2 节中已经归为任务可执行性约束。由式(12)可得 m 组子任务平均可执行性为

$$E_m=\left(\frac{(\sum_{j=1}^{q_k}(\max(S_1^j,S_2^j,\cdots,S_h^j)))/q_k}{m}\right) \tag{27}$$

且满足条件

$$E_m\geqslant\xi \tag{28}$$

(2) 其他约束条件

其他约束条件体现在任务分组过程中。一个任务只能属于其中的一个任务组，即

$$\sum_{k=1}^{m}tg_{ik}=1 \tag{29}$$

且有 $\sum_{i=1}^{n}tg_{ik}=q_k$ (30)

其中，$tg_{ik}=1$ 表示子任务 i 属于第 k 个任务组；

$tg_{ik}=0$ 表示第 k 个任务组不包含子任务； q_k 表示第 k 个任务组中共包含 q_k 个子任务。

3.3 基于改进自适应遗传算法的模型求解

受生物进化过程中“物竞天择，适者生存”自然法则的启发，Michigan 大学的 John Holland 在 20 世纪 60 年代就提出了遗传算法的思想。遗传算法将优化目标函数转化为适应度函数，将优化问题的参数空间通过编码操作转换为编码空间，以适应度函数为评价标准，对群体进行选择、交叉和变异操作，实现新一代群体的不断迭代过程，从而建立问题的最优解。但传统遗传算法的交叉概率 P_c 和变异概率 P_m 固定不变，意味着种群中无论是优良个体还是较差个体都以固定概率进行交叉变异操作，导致寻优精度和速率降低。而实际应用中，通常希望优良个体的交叉变异概率尽量低而较差个体的交叉变异概率尽量高，以使群体中优良基因得以尽量保存的同时，产生更多的优良基因。由此，自适应遗传算法应运而生。

Srinivas 和 Patnaik[16]根据每代种群的个体适应度值的改变动态地改变 P_c 和 P_m，真正意义上实现了遗传算法参数能够自适应调整，但该算法在保护种群中优势个体的同时，使得算法的收敛速度过快，导致算法容易陷入局部最优，且算法通过个体适应度值的改变控制 P_c 和 P_m，每次迭代都需要计算每个个体的 P_c 和 P_m，影响算法的运行效率。对此，Zhang 等[15]提出一种改进自适应遗传算法，对 P_c 和 P_m 进行改进。该算法虽然改善了收敛速度过快的问题，但每次迭代仍需计算每个个体的 P_c 和 P_m，算法的执行效率依然较低，不适用于实际操作过程。此外，算法在迭代过程中， P_c 和 P_m 的值可能超过 1。基于此，本文根据文献[17]，针对以上两种算法的不足，提出一种改进的自适应算法(Improved adaptive algorithm, IAGA)，对 P_c 和 P_m 进行进一步改进。

3.3.1 自适应交叉概率调整策略

交叉算子主要控制种群中新个体的产生机制，交叉概率过大，不利于保存优良个体的基因，从而破坏种群的优良遗传模式。而交叉概率过小则会造成新个体进化速度过慢。

纵观整个进化过程，在进化前期，应适当增大 P_c，旨在提高算法的全局搜索能力，加快种群更新速度；在进化后期，随着算法迭代次数的增加，应适当减小 P_c，从而有利于种群逐步趋于稳定，保存优良个体的基因。而在进化过程中，对于适应度较高的优良个体，其 P_c 值应随迭代次数的增加而减小，确保优良基因不被交叉操作而破坏；对于适应度值低的较差个体， P_c 取值应随着迭代次数的增加不断增大，以促使基因不断优化。从种群进化代数与适应度两方面出发，所述自适应交叉概率调整策略表述为

$$P_{ci}=\begin{cases}P_{c\max}-\dfrac{\lambda g}{G}\cdot(P_{c\max}-P_{c\min})\left(\dfrac{f_i-\bar{f}}{f_{\max}-\bar{f}}\right), & f_i\geqslant\bar{f}\\ \dfrac{\lambda g}{G}\cdot P_{c\max}, & f_i\geqslant\bar{f}\end{cases} \tag{31}$$

式中：P_{ci} 为个体 i 的交叉概率；λ 为进化代数乘积因子调节参数；G 为算法最大迭代次数；g 为当前迭代次数； $P_{c\max}$ 为预设最大交叉概率，其取值情况与进化整体进度有关，如式(32)所示；$P_{c\min}=0.6$ 为预设最小交叉概率； f_i 表示个体 i 的适应度； $f_{\max}$ 表示当前种群中个体的最大适应度； $\bar{f}$ 表示当前种群的平均适应度。

$$P_{c\max}=\begin{cases}0.95, & g\leqslant G/4\\ 0.85, & G/4<g\leqslant 3G/4\\ 0.75, & 3G/4<g\leqslant G\end{cases} \tag{32}$$

3.3.2 自适应变异概率调整策略

变异算子主要控制种群的多样性机制，防止算法发生早熟的情况。变异概率过大，将导致算法的搜索过程近乎随机搜索状态，同样破坏种群的良性遗传机制；而变异概率过小，种群中难以通过变异操作产生新个体，即无法维持种群多样性，从而易使算法陷入局部最优。

纵观整个进化过程，在进化前期，应适当减小 P_m，尽可能降低种群个体发生变异，抑制算法发生早熟；在进化后期，随着算法迭代次数的增加，应适当增加 P_m，增加种群多样性，以扩大种群搜索范围，防止算法陷入局部最优。而在种群进化过程中，对于适应度高的优良个体，其 P_m 值应随迭代次数的增加而减小，确保优良基因不被变异操作而破坏；对于适应度值低的较差个体，P_m 取值应随着迭代次数的增加不断增大，以促使基因通过变异不断优化。从种群进化代数与适应度出发，所述自适应变异概率调整策略表述为

$$P_{mi}=\begin{cases}P_{m\min}+\dfrac{\lambda g}{G}\cdot(P_{m\max}-P_{m\min})(\dfrac{f_i-\overline{f}}{f_{\max}-\overline{f}}), & f_i\geqslant\overline{f}\\ \dfrac{\lambda g}{G}\cdot P_{m\min}, & f_i\geqslant\overline{f}\end{cases} \tag{33}$$

式中：P_{mi} 为个体 i 的变异概率；λ 为进化代数乘积因子调节参数；G 为算法最大迭代次数；g 为当前迭代次数；$P_{m\min}$ 为预设最小变异概率，其取值情况与进化整体进度有关，如式(34)所示；$P_{m\max}=0.1$ 为预设最大变异概率；f_i 表示个体 i 的适应度；$f_{\max}$ 表示当前种群中个体的最大适应度；$\overline{f}$ 表示当前种群的平均适应度。

$$P_{m\min}=\begin{cases}0.001, & g\leq G/4\\ 0.002, & G/4<g\leq 3G/4\\ 0.003, & 3G/4<g\leq G\end{cases} \tag{34}$$

4 算例分析

4.1 改进自适应遗传算法的求解性能测试

为证明本文所提 IAGA 算法有效，实验选取 2 类算法常用的典型测试函数如式(35)~(36)所示：

$$f_1(x,y)=x^2+y^2;\ x,y\in[-5.12,5.12] \tag{35}$$

$$f_2(x,y)=(1-x)^2+106(y-x^2)^2;\quad x,y\in[-2,2] \tag{36}$$

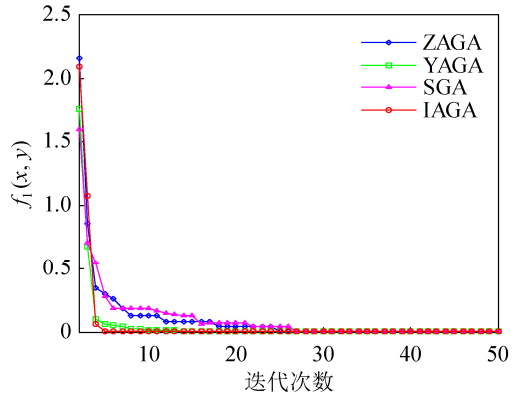
式(35)为平方和函数，该函数较为简单，函数在 $(0,0)$ 处出现全局最小值；式(36)为 Rosenbrock 函数，该函数虽为单峰值函数，在 $(1,1)$ 处出现全局最小值，但由于函数在 $y=x^2$ 处呈现狭长的深谷状，在寻优过程中极易陷入局部最优解，因而具有一定的代表性。

实验采用 3 种相关遗传算法为对比算法，分别包括 Zhang 等[15]提出的改进自适应遗传算法记为 ZAGA、于莹莹等[17]提出的改进自适应遗传算法记为 YAGA 以及标准遗传算法记为 SGA，在 matlab 2014a 实验平台上进行编程，实现与本文提出的改进自适应遗传算法(IAGA)寻优性能的对比。四种算法的相关实验参数设置如表 3 所示。

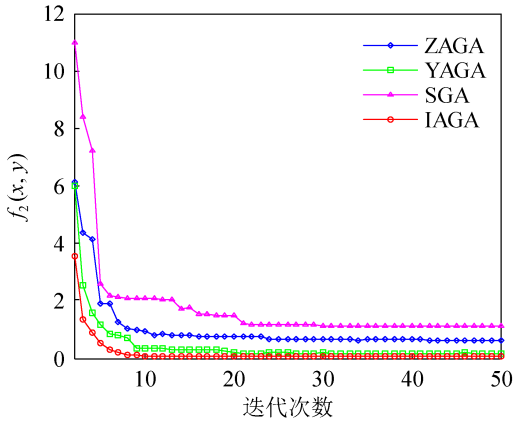
实验过程中，设置每种算法迭代 50 次，种群大小设置为 20。绘制不同算法与所述 IAGA 算法性能实验结果对比如图 4 所示，并记录每种算法运行 10 次平均运行时间 $\overline{t}$、达到最优值的迭代次数 N 以及适应度平均值 $\overline{f}$、最大值 $f_{\max}$、最小值 $f_{\min}$ 结果如表 4 所示。

表 3 ZAGA、YAGA、SGA 与 IAGA4 种算法相关实验参数设置
Tab. 3 Related experiment parameter settings of ZAGA, YAGA, SGA and IAGA

算法		ZAGA	YAGA	SGA	IAGA
参数设置	$P_{c\max}$	0.95	0.9 ($g\leqslant G/4$) 0.8 ($G/4<g\leqslant 3G/4$) 0.7 ($3G/4<g\leqslant G$)	$P_c=0.9$	0.95 ($g\leqslant G/4$) 0.85 ($G/4<g\leqslant 3G/4$) 0.75 ($3G/4<g\leqslant G$)
	$P_{c\min}$	0.75	0.6		0.65
	$P_{m\max}$	0.1	0.005	$P_m=0.1$	0.1
	$P_{m\min}$	0.01	0.001 ($g\leqslant G/4$) 0.002 ($G/4<g\leqslant 3G/4$) 0.003 ($3G/4<g\leqslant G$)		0.01 ($g\leqslant G/4$) 0.02 ($G/4<g\leqslant 3G/4$) 0.03 ($3G/4<g\leqslant G$)



(a) 平方和测试函数 f_1



(b) Rosenbrock 测试函数 f_2

图 4 4 种算法性能对比

Fig. 4 Performance comparison of four algorithms

表 4 算法平均运行时间、迭代次数及适应度对比

Tab. 4 Comparison of algorithm average running times, iterations, and fitness values

f	算法	$\bar{f}$	$f_{\max}$	$f_{\min}$	$\bar{t}$ / s	N
f_1	ZAGA	0.000 2	0.000 2	0.000 2	7.890	24
	YAGA	0.000 2	0.000 2	0.000 2	7.650	16
	SGA	0.000 2	0.000 2	0.000 2	7.100	26
	IAGA	0.000 2	0.000 2	0.000 2	7.320	11
f_2	ZAGA	0.932 0	3.655 0	0.000 4	5.412	23
	YAGA	0.090 0	0.545 0	0.000 4	5.369	31
	SGA	1.124 0	2.913 0	0.135 0	5.323	30
	IAGA	0.066 0	0.259 0	0.000 4	5.356	15

由图 4(a)和表 4 可知，测试函数 f_1 比较简单，4 种算法的仿真效果相近，但从算法达到最优值的迭代次数和运行时间来看，IAGA 算法仍占优势。

由图 4(b)和表 4 可知，由于测试函数 f_2 较为复杂且呈现病态特性，4 种算法表现出不同的性能。其中，IAGA 算法性能明显优于其他 3 种算法。从寻优精度来看，IAGA 算法的寻优精度比 ZAGA 和 SGA 算法提高 90%以上，比 YAGA 算法提高 26.7%；从适应度最值来看，IAGA 算法的稳定性优于其他算法；从算法运行时间来看，IAGA 算法的运行时间只稍逊于 SGA 算法，但算法收敛性能均较其他算法好，在算法进化到 15 代时即达到了函数的最优值。这是由于与 ZAGA 和 SGA 算法相比，IAGA 算法引入了与算法进化代数密切相关的交叉概率与变异概率，在迭代过程中容易跳出局部最优解，算法表现出较好的收敛性能；与 YAGA 算法相比，虽然两者都引入了进化代数，但 IAGA 算法将进化代数作为乘积算子引入交叉和变异概率中，提高了算法的稳定性和寻优精度。

4.2 云任务分解模式优化方法有效性验证

为简化实验设置，在文献[15]中关于某品牌手机结构设计任务实验数据的基础上对云任务分解模式优化方法的有效性进行验证。

(1) 基于 ETM 的动态任务分解

设企业将某品牌手机结构设计任务发布到云端融合平台，并完成任务各项描述信息的录入，平台检测到新任务到达事件，立即触发 ETM，ETM 调用任务信息数据库中的任务描述信息，同时结合服务信息库中的资源描述信息，对任务可分解性进行分析，调取任务历史分解方案数据库，将当前待分解任务描述信息与历史分解方案进行匹配，最终将该任务分解为形状设计、观造型设计等 14 个子任务记 $T=\{T_1,T_2,\cdots,T_{14}\}$。

(2) 确定子任务可执行性

设定可执行阈值 $\xi=0.75$，基于任务描述的基本信息 *GenInf*、功能信息 *FunInf* 和约束信息 *ConsInf*，根据式(6)~(12)确定子任务的可执行性矩阵 $\boldsymbol{E}_j$ 见表 5，由表 5 可知，各子任务都是可执行的，无需对子任务进行再分解。

(3) 确定子任务信息交互关系矩阵

确定子任务可执行性后，即可对子任务的约束结构类型进行判定，同时，根据子任务间的约束结

构类型，参照表 2，评估前序子任务对后序子任务的影响矩阵 $\boldsymbol{A}=\{a_{ij}\}_{n\times n}$ 以及后序子任务对前序子任务的信息反馈矩阵 $\boldsymbol{B}=\{\mathrm{b}_{ji}\}_{n\times n}$，再由式(13)~(14)，计算最终子任务间的信息交互关系矩阵 $\boldsymbol{C}=\{c_{ij}\}_{n\times n}$ 如表 6 所示。

(4) 确定子任务工作量矩阵

子任务的工作量由企业工程师进行评估，各子任务的工作量见表 7。

(5) 子任务分组模型求解

根据子任务可执行性矩阵 $\boldsymbol{E}_j$、子任务间的信息交互关系矩阵 $\boldsymbol{C}=\{c_{ij}\}_{n\times n}$ 和子任务工作量矩阵 $\boldsymbol{EG}=\{eg_{ik}\}_{n\times m}$，以组内子任务相关度、组间子任务耦合度和每组任务均衡度为优化目标，由式(16)~(25)构建多目标优化模型，并根据式(25)将其转化成单目标优化模型。采用 IAGA 算法，在 matlab 2014a 实验平台上进行编程(实验相关参数设置同表 3)，实现对所构建任务分解模式优化模型的求解，将求解结果导入 Origin 2017 数据分析软件中，对 3 个目标函数进行可视化验证。为更直观地呈现目标函数的验证结果，绘制双纵坐标图见图 5。图 5 中，横坐标代表算法迭代次数，左纵坐标对应目标函数包括组内子任务相关度与组间子任务耦合度，右纵坐标对应目标函数为任务均衡度。由图 5 可知，算法进化到 126 代时目标函数达到最优，此时子任务分为 5 组，组内子任务相关度达到 0.69，组间子任务耦合度为 0.13，任务均衡度为 2，最终任务分组结果如表 8 所示。

表 5 子任务可执行性矩阵 $\boldsymbol{E}_j$
Tab. 5 Executability matrix $\boldsymbol{E}_j$ of subtasks

任务	可执行性	任务	可执行性	任务	可执行性
T_1	0.84	T_6	0.93	T_{11}	0.95
T_2	0.91	T_7	0.92	T_{12}	0.83
T_3	0.81	T_8	0.83	T_{13}	0.89
T_4	0.85	T_9	0.79	T_{14}	0.82
T_5	0.89	T_{10}	0.78		

表 6 子任务间信息交互关系矩阵 $\boldsymbol{C}=\{c_{ij}\}_{n\times n}$
Tab. 6 Information interaction matrix $\boldsymbol{C}=\{c_{ij}\}_{n\times n}$ between subtasks

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}
t_1	0.00	0.69	0.45	0.60	0.49	0.35	0.40	0.30	0.30	0.00	0.00	0.00	0.00	0.00
t_2		0.00	0.45	0.60	0.42	0.35	0.35	0.30	0.30	0.00	0.00	0.00	0.00	0.00
t_3			0.00	0.70	0.65	0.00	0.57	0.57	0.30	0.00	0.30	0.00	0.00	0.00
t_4				0.00	0.60	0.30	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00
t_5					0.00	0.75	0.40	0.30	0.30	0.30	0.30	0.30	0.30	0.30
t_6						0.00	0.50	0.50	0.20	0.30	0.30	0.30	0.30	0.30
t_7							0.00	0.69	0.75	0.35	0.30	0.20	0.30	0.20
t_8								0.00	0.70	0.60	0.60	0.50	0.50	0.50
t_9									0.00	0.60	0.60	0.50	0.30	0.30
t_{10}										0.00	0.69	0.80	0.50	0.50
t_{11}											0.00	0.80	0.42	0.50
t_{12}												0.00	0.46	0.42
t_{13}													0.00	0.80
t_{14}														0.00

表 7 子任务工作量矩阵 $\boldsymbol{EG}=\{eg_{ik}\}_{n\times m}$
Tab. 7 Workload matrix $\boldsymbol{EG}=\{eg_{ik}\}_{n\times m}$ of subtasks

任务	工作量	任务	工作量	任务	工作量	任务	工作量	任务	工作量	任务	工作量
T_1	20	T_6	8	T_{11}	12	T_4	12	T_9	7	T_{14}	20
T_2	15	T_7	6	T_{12}	16	T_5	5	T_{10}	13		
T_3	6	T_8	18	T_{13}	17						

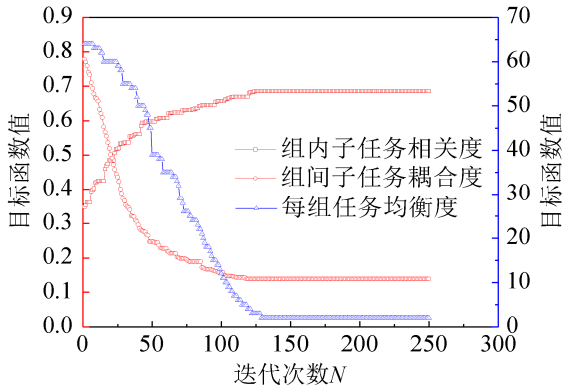


图 5 目标函数的可视化验证
Fig. 5 Visual verification of objective function

表 8 子任务分组结果
Tab. 8 Subtasks grouping results

组别	任务
1	$\{T_1、T_2\}$
2	$\{T_3、T_6、T_{14}\}$
3	$\{T_4、T_5、T_9、T_{11}\}$
4	$\{T_8、T_{13}\}$
5	$\{T_7、T_{10}、T_{12}\}$

4.3 模型对比

文献[15]提出一种静态任务分解模式，并以任务相关度和任务耦合度为优化目标对任务分解模式进行优化。本文针对云端融合平台中任务时刻处于动态性变化的特征，基于 ETM 提出一种动态云任务分解模式，以动态任务可执行性为云任务分解的标准，任务的每一次细分都综合考虑了实际资源、服务等环境的变化情况；在优化目标上，不仅考虑任务相关度、任务耦合度，同时考虑任务均衡度，并通过性能更优的 IAGA 算法对任务分解过程进行优化。

考虑任务均衡度的情况，在相同的实验条件下，文献[15]最终的任务分组情况见表 9，当目标函数达到最优时，记录不同分解模式的各指标数值见表 10，其中，表 10 的第 4 列表示本文结果相对于文献[15]各指标值提高的百分比。绘制不同分解模式组内任务相关度、组间任务耦合度以及每组任务均衡度的仿真结果对比图见图 6~8。

表 9 子任务分组结果[15]
Tab. 9 Subtasks grouping results[15]

组别	任务
1	$\{T_1、T_2\}$
2	$\{T_3、T_4、T_{13}\}$
3	$\{T_7、T_8、T_{11}\}$
4	$\{T_5、T_6、T_{14}\}$
5	$\{T_9、T_{10}、T_{12}\}$

表 10 不同分解模式结果对比
Tab. 10 Comparison of different decomposition mode

指标名称	文献[15]	本文	提高/%
组内任务相关度	0.65	0.69	6.15
组间任务耦合度	0.19	0.13	31.58
任务均衡度	6	2	66.67
运行时间	61.28s	56.36s	8.03
迭代次数	165	126	23.64

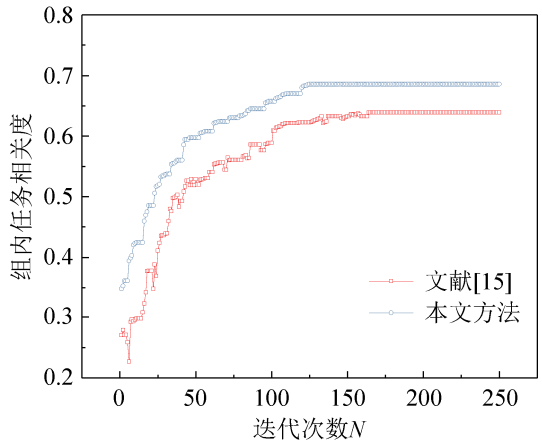


图 6 组内任务相关度仿真结果对比
Fig. 6 Comparison of group task correlation results

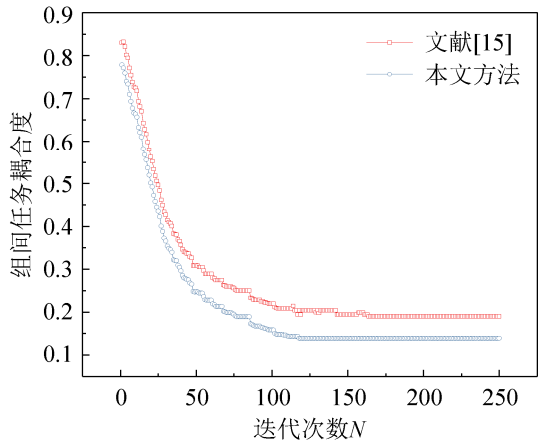


图 7 组间任务耦合度仿真结果对比
Fig. 7 Comparison of task coupling degree between groups

DOI: 10.16182/j.issn1004731x.joss.201811001

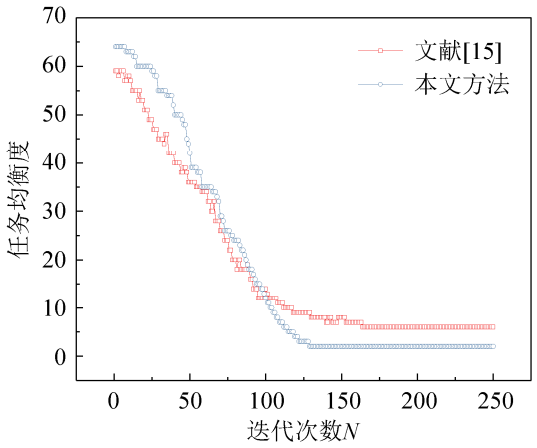


图 8 每组任务均衡度仿真结果对比
Fig. 8 Comparison of task balance of each group

基于第 3 节的分析可知，组内任务相关度越高、组间任务耦合度和任务均衡度越低，目标函数值越大，即任务分组效果越好。表 10 与图 6~8 的结果显示：本文各指标精度高于文献[15]，尤其是任务均衡度指标比文献[15]提高 66.67%，这对于存在海量任务的云端融合平台来说，将大大减轻任务处理器的负担，提高任务的执行效率；此外，本文所提出的动态任务分解模式较文献[15]的静态任务分解模式的运行时间短，能更好满足云端任务处理的实时性高的要求。相较之下，本文所提出的动态云任务分解模式优势明显。

5 结论

本文研究了云端融合过程中动态任务分解模式优化问题，针对云任务的动态性特征，通过引入事件触发机制，解决了随机到达云端融合平台任务的动态分解问题，在此基础上根据组内任务相关度最大、组间任务耦合度最小以及每组任务工作量均衡的原则，对分解后的子任务进行分组建模，并通过改进自适应遗传算法经算例编程加以实现，得出以下结论：

(1) 基于 ETM 的动态任务分解方法能够有效处理随机到达云端融合平台的任务分解问题，为云端融合模式下的实时化、智能化事件处理机制提供了一个很好的思路；

(2) 建立的任务分组模型不仅考虑了组内任务相关度和组间任务耦合度相关因素，还考虑了每组任务均衡性，有效降低了任务间的信息交互频率和复杂度。此外，理论上同一分组内的任务对资源服务具有相似性需求，为后续研究云任务-资源匹配及云任务-资源调度问题做了很好的铺垫；

(3) 设计的一种改进自适应遗传算法，引入与进化代数密切相关的交叉算子和变异算子，能够有效提高算法的寻优精度和收敛性能，避免算法发生早熟情况，同时算法稳定性得到有效提高。

当然，本文虽然提出了基于 ETM 的动态任务分解模式优化方法，只是在理论上对其进行了分析，其实际机理非常复杂，也是我们下一步需要进行细化研究的方向。

参考文献：

[1] 王彩璐, 陶跃钢, 杨鹏, 等. 云控制系统并行任务分配优化算法与并联控制[J]. 自动化学报, 2017, 43(11): 1973-1983.
Wang Cailu, Tao Yuegang, Yang Peng, et al. Parallel Task Assignment Optimization Algorithm and Parallel Control for Cloud Control Systems [J]. Acta Automatica Sinica, 2017, 43(11): 1973-1983.

[2] 刘明周, 王强, 凌琳. 基于分层任务网络的云制造任务分解方法[J]. 中国机械工程, 2017, 28(8): 924-930.
Liu Mingzhou, Wang Qiang, Ling Lin. Cloud Manufacturing Task Decomposition Method Based on HTN[J]. China Mechanical Engineering, 2017, 28(8): 924-930.

[3] Sanjaya K, Prasanta K. Normalization-Based Task Scheduling Algorithms for Heterogeneous Multi-Cloud Environment[J]. Information Systems Frontiers (S1387-3326), 2018, 20(2): 373-399.

[4] Liu A J, Pfund M, Fowler J. Scheduling optimization of task allocation in integrated manufacturing system based on task decomposition [J]. Journal of Systems Engineering and Electronics(S1004-4132), 2016, 27(2): 422-433.

[5] Zhou L, Zhang L, Zhao C, et al. Diverse task scheduling for individualized requirements in cloud manufacturing [J]. Enterprise Information Systems(S1751-7575), 2017, 12(3): 300-318.

[6] Liu Z, Song C, Chu D, et al. An Approach for Multipath

Cloud Manufacturing Services Dynamic Composition [J]. International Journal of Intelligent Systems (S1098-111X), 2017, 32(4): 371-393.

[7] Zhang Y, Wang J, Liu S, et al. Game Theory Based Real-Time Shop Floor Scheduling Strategy and Method for Cloud Manufacturing [J]. International Journal of Intelligent Systems(S1098-111X), 2017, 32(4): 437-463.

[8] Zhou L F, Zhang L, Sarker B R, et al. An event-triggered dynamic scheduling method for randomly arriving tasks in cloud manufacturing[J]. International Journal of Computer Integrated Manufacturing(S0951-192X), 2017, 31(3):1-16.

[9] Cao Y, Wang S, Kang L, et al. A TQCS-based service selection and scheduling strategy in cloud manufacturing [J]. International Journal of Advanced Manufacturing Technology(S0268-3768), 2016, 82(1/2/3/4): 235-251.

[10] Moslehi G, Mahnam M. A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search [J]. International Journal of Production Economics (S0925-5273), 2011, 129(1): 14-22.

[11] Jin H, Yao X, Chen Y. Correlation-aware QoS modeling and manufacturing cloud service composition [J]. Journal of Intelligent Manufacturing(S0956-5515), 2017, 28(8): 1947-1960.

[12] Que Y, Zhong W, Chen H, et al. Improved adaptive immune genetic algorithm for optimal QoS-aware service composition selection in cloud manufacturing [J]. International Journal of Advanced Manufacturing Technology(S0268-3768), 2018, 96(10): 4455-4465.

[13] 杨财. 云制造环境下的企业生产管理模式研究[D]. 杭州: 浙江大学, 2014.
Yang Cai. Rearch on Enterprise Production Management Mode in Cloud Manufacturing Environment[D]. Hangzhou: Zhejiang University, 2014.

[14] 侯亮, 陈峰, 温志嘉. 跨企业产品协同开发中的设计任务分解与分配[J]. 浙江大学学报(工学版), 2007, 41(12): 1976-1981.
Hou Liang, Chen Feng, Wen Zhijia. Design task decomposition and allocation for inter-firm product collaborative development[J]. Journal of Zhejiang University (Engineering Science), 2007, 41(12): 1976-1981.

[15] Zhang X F, Yang Y, Bao B. Task Decomposition and Grouping for Customer Collaboration in Product Development[J]. Journal of Intelligent Systems, 2016, 25(3): 361-375.

[16] Srinivas M, Patnaik L M. Adaptive probabilities of crossover and mutation in genetic algorithms[J]. IEEE Transactions on Systems Man & Cybernetics (S2168-2909), 2002, 24(4): 656-667.

[17] 于莹莹, 陈燕, 李桃迎. 改进的遗传算法求解旅行商问题[J]. 控制与决策, 2014, 29(8):1483-1488.
Yu Yingying, Chen Yan, Li Taoying. Improved genetic algorithm for solving TSP[J]. Control and Decision, 2014, 29(8): 1483-1488.

《系统仿真学报》荣获“2017 中国国际影响力优秀学术期刊”证书

由中国学术期刊（光盘版）电子杂志社与清华大学图书馆联合成立的中国学术文献国际评价研究中心，发布了 2017 版《中国学术期刊国际引证年报》，《系统仿真学报》荣获“2017 中国国际影响力优秀学术期刊”。

《年报》（2017 版）采用的统计源期刊为 20192 种，涵盖 WoS 收录的 SCI 期刊 8874 种、SSCI 和 A&HCI 期刊 4645 种，ESCI 期刊 5578 种；增补期刊 1762 种。参照中外文学术期刊总被引频次、影响因子、半衰期等各项国际引证指标，计算期刊影响力指数(CI)，对国内 6210 种学术期刊排序，遴选了人文社科、自然科学与工程技术两个类别的 TOP10%为国际影响力品牌学术期刊。TOP5%以内的期刊为“最具国际影响力学术期刊”、TOP5-10%之间的为“国际影响力优秀学术期刊”。

https://dc-china-simulation.researchcommons.org/journal/vol30/iss11/1
DOI: 10.16182/j.issn1004731x.joss.201811001