计算机科学
COMPUTER SCIENCE

一种基于产品复用模型的高效遥感共性产品生产算法

左宪禹, 周小虎, 周黎明, 谢毅, 刘成

引用本文

左宪禹, 周小虎, 周黎明, 谢毅, 刘成. 一种基于产品复用模型的高效遥感共性产品生产算法[J]. 计算机科学, 2025, 52(6): 316-323.

ZUO Xianyu, ZHOU Xiaohu, ZHOU Liming, XIE Yi, LIU Cheng. Efficient Remote Sensing Common Product Production Algorithm Based on Product Reuse Model [J]. Computer Science, 2025, 52(6): 316-323.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.240300019

一种基于产品复用模型的高效遥感共性产品生产算法

左宪禹　周小虎　周黎明　谢　毅　刘　成
河南省大数据分析与处理重点实验室(河南大学)　河南 开封 475000
河南大学计算机与信息工程学院　河南 开封 475000
(xianyu_zuo@henu.edu.cn)

摘　要　随着各行业对遥感共性产品需求的不断增加,高性能遥感产品生产系统的应用范围不断扩大。优秀的任务调度算法作为该系统的关键部件,能显著提高生产效率。然而,在遥感共性产品的生产过程中面临特有的挑战,如果大量的工作流在短时间内被提交生产,这些工作流在处理中存在重复计算和数据处理的问题,且生成共性产品所需的数据量往往较大,流程处理时间长,很容易导致资源浪费和生产效率下降。为了解决这一问题,提出一种基于产品复用模型的任务划分策略。该策略着眼于优化工作流处理,首先将用户提交的工作流按照任务重复度打包成流程包,把带有重复任务的流程分配到同一个计算节点,旨在减少节点间的数据传输时间;然后引入一种产品复用模型,允许不同的处理流程复用已获得的产品结果,减少重复性计算和数据处理,从而提高生产效率,满足共性产品生产的高效化需求。为了验证所提算法的有效性,将所提算法和传统算法FCFS,SJF分别在CloudSim仿真模拟器中进行模拟实验。结果表明,所提调度算法任务的总完成时间和任务的平均响应时间均显著低于对比算法,展现出了更为优秀的性能。

关键词:高性能计算;共性遥感产品;产品复用;任务划分策略;CloudSim
中图分类号　TP302

Efficient Remote Sensing Common Product Production Algorithm Based on Product Reuse Model

ZUO Xianyu,ZHOU Xiaohu,ZHOU Liming,XIE Yi and LIU Cheng
Henan Key Laboratory of Big Data Analysis and Processing,Henan University,Kaifeng,Henan 475000,China
School of Computer and Information Engineering,Henan University,Kaifeng,Henan 475000,China

Abstract　With the increasing demand for remote sensing common products in various industries,the application of high-performance remote sensing product production system is increasing. As a key component of the system,excellent task scheduling algorithm can significantly improve its production efficiency. However,there are unique challenges in the production process of remote sensing generic products. If a large number of workflows are submitted for production in a short time,there are problems of double calculation and data processing in the processing of these workflows,and the amount of data required to generate generic products is often large,and the process processing time is long,which easily leads to resource waste and production efficiency decline. In order to solve this problem,this paper proposes a task division strategy based on product reuse model,which focuses on optimizing workflow processing. Firstly,workflow submitted by users is packaged into a process package according to task repetition,and processes with repetitive tasks are assigned to the same computing node to reduce the data transmission time between nodes. Then,a product reuse model is introduced to allow different processing processes to reuse the obtained product results,reduce repetitive calculation and data processing,so as to improve production efficiency and meet the high efficiency needs of common product production. In order to verify the effectiveness of the proposed algorithm,the proposed algorithm and other traditional algorithms FCFS and SJF are simulated in the CloudSim simulation simulator respectively. The results show that the proposed scheduling algorithm has significantly lower total task completion time and average task response time than the other two

到稿日期:2024-03-03　返修日期:2024-08-17
基金项目:国家重点研发计划国际合作专项(2019YFE0126600);河南省科技重大专项(201400210300);河南省高校科技创新团队(24IRTSTHN021);河南省科技厅科技攻关项目(232102210009);河南省科技攻关(242102240021);2024河南省研究生联合培养基地项目(YJS2024JD30)
This work was supported by the National Key Research and Development Program for International Cooperation(2019YFE0126600),Henan Province Major Science and Technology Project(201400210300),Henan University Science and Technology Innovation Team(24IRTSTHN021),Henan Provincial Department of Science and Technology Research Project(232102210009),Henan Province Science and Technology Research(242102240021) and Postgraduate Education Reform and Quality Improvement Project of Henan Province(YJS2024JD30).
通信作者:刘成(liucheng@henu.edu.cn)

algorithms，showing better performance.

Keywords High performance computing，Generic Remote sensing products，Product reuse，Task divison strategy，CloudSim

1 引言

近年来，随着遥感影像数据量的快速增长以及遥感影像在各个领域应用需求的增加[1]，高分辨率的遥感共性产品在实际应用中占据了重要地位，其能提供丰富的地表信息，支持城市规划、环境检测、资源管理、智慧农业等多个领域的决策、应用[2]。在遥感共性产品的生产过程中，产品生产流程的制定是通过可视化的工作流来指导算法的调用次序，并最终生产出相应的共性产品[3]。然而，大量的用户会在短时间内提交多个工作流，而这些工作流又可能存在重复计算和数据处理的问题，因此极易出现资源浪费和处理效率低下的情况。

随着各行业应用需求的增加，人们对遥感共性产品的生产效率也有更高的需求。高性能集群计算[4]因具有响应时间短、吞吐率高、价格相对低廉等显著优势，而被广泛应用于遥感数据的处理中。其核心是集群系统的任务调度算法。算法的优劣直接影响到整个集群系统的数据处理效率。

任务调度算法可以大致分为静态调度和动态调度[5]。静态调度能够事先获取更多任务信息（如任务之间的关联性、任务在服务器上的运行时间等），具有易实现、效率高、成本小等特征。动态调度是根据服务器和网络负载情况来随时调整算法，具有较高的灵活度和适应能力，在并行计算中应用较多，但是也存在难以解决的实际问题，并且实际应用较少。传统的调度算法有先来先服务算法（FCFS）[6]、短作业优先（SJF）[7]、多级反馈队列算法（RRMF）[8]、高优先权调度算法（HPS）[9]和轮转调度算法（RR）[10]等。而针对遥感任务调度算法，Shi 等提出了一种基于集群调度系统 Open PBS 的双级调度策略，该策略通过自动选择最优资源数和使用基本作业调度策略，提高了系统资源的利用率和吞吐率，但是存在调度占用时间过长等缺陷[11]。Wu 提出了一种考虑负载均衡的两级任务调度模型，首先系统根据各个计算节点当前的负载情况对遥感算法任务进行第一级调度，然后在计算节点内部运用 FCFS 的调度方法实现二级调度[12]。Zheng 等为了解决在实现大规模遥感数据产品的快速生产中所面临的效率低、生产耗时长、计算资源利用率低等问题，设计了一个以集群并行计算技术为基础的支持多任务的遥感产品生产线架构，同时采用高效的任务调度策略，来进一步提高计算资源的利用率[13]。然而，上述方法中采用的经典调度算法没有针对遥感任务进行优化，也未考虑遥感任务间的数据依赖关系和工作流间存在的重复计算、数据处理的情形。

为解决上述问题，结合国家民用空间基础设施中长期发展规划（2015－2025 年）子课题项目中所使用的遥感算法流程处理平台，本文提出了一种基于产品复用模型的任务划分策略 RTPS（Task Partitioning Strategy Based on Repetitive Tasks）。该策略首先将用户提交的工作流按照任务重复度打包成流程包，将带有重复任务的流程分配到同一个计算节点上来减少数据在节点间的传输时间；然后利用提出的产品复用模型，使不同处理流程可以复用已得到的产品结果，减少重复性计算和数据处理，从而提高生产效率，满足共性产品生产的高效化需求。

2 问题描述

为了解决在该系统的生产过程中不同工作流流程之间存在的重复计算和数据处理的问题，本章将先介绍有关工作流的相关概念，具体如表 1 所列。

表 1 工作流相关概念

Table 1 Workflow-related concepts

概念	解释
τ_i	工作流中的任务 $i\in\{en,com,out\}$，τ_{en} 表示工作流中原始数据的准备；τ_{com} 表示工作流中运行的遥感算法，com 代指执行算法的名字；τ_{out} 表示工作流中产品的存储
$\tau_i(oD)$	任务 τ_i 的出度，其中 $i\in\{en,com,out\}$
$\tau_i(iD)$	任务 τ_i 的入度，其中 $i\in\{en,com,out\}$
$e_{i,j}$	任务 τ_i 和任务 τ_j 的依赖关系（边），表示任务间的先后执行顺序，即只有任务 τ_i 执行完成后，才能够执行任务 τ_j
E_i	任务间依赖关系（边）$e_{i,j}$ 的集合
$\mu_{i,j}$	边 $e_{i,j}$ 上携带的数据量
M_i	边 $e_{i,j}$ 上携带的数据量 $\mu_{i,j}$ 的集合

2.1 工作流模型

在生产系统中，大量的用户会提交生产流程来进行遥感产品的生产。将这些工作流的集合定义为：

$$W=\{w_1,w_2,\cdots,w_n\} \tag{1}$$

其中，w_i 定义为一个工作流的结构模型，如图 1 所示。

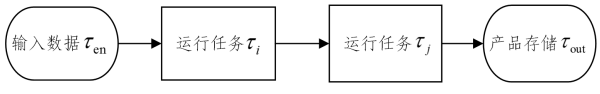


图 1 工作流模型

Fig. 1 Workflow model

该模型是由一个或者多个任务相互依赖组成，τ_{en} 表示原始数据的准备；τ_{out} 表示产品的存储，如文件等；其余表示遥感算法的执行。由于任务之间存在依赖关系，因此将 w_i 建模为有向无环图（Directed Acyclic Graph，DAG），表示为：

$$w_i=(V_i,E_i,M_i) \tag{2}$$

其中，V_i 为工作流 w_i 的任务集合，由 $V_i=\{\tau_1,\tau_2,\tau_3,\cdots,\tau_n\}$ 给出。集合中任意任务 τ_i 由唯一标识符表示，τ_1 表示原始数据的准备 τ_{en}，τ_n 表示产品的存储 τ_{out}。

本系统所设计的计算集群由 m 个非抢占式计算节点组成，$C=\{c_1,c_2,\cdots,c_m\}$，具有依赖关系的任务 τ_j 与任务 τ_i 之间的数据传输时间可以表示为 $trans(\tau_j,\tau_i)$：

$$trans(\tau_j,\tau_i)=\begin{cases}\dfrac{\mu_{j,i}}{\beta_{\langle j,i\rangle}}, & c(\tau_j)\neq c(\tau_i) \text{ and } j,i\in\{2,\cdots,n-1\}\\ 0, & \text{otherwise}\end{cases} \tag{3}$$

其中，$\beta_{\langle j,i\rangle}$ 为计算节点 c_j 和 c_i 所在的物理机[14]之间的带宽参数，$c(\tau_j)$ 为任务 τ_j 执行所在的计算节点。由式（3）可知，当相互依赖的两个任务在同一个计算节点时，它们之间的数据传输时间可以忽略不计。结合上述信息，任务 τ_i 的开始时间定义如下：

$$ST(\tau_i)=\max\{FT(\tau_j)+trans(\tau_j,\tau_i),Ava(c(\tau_i))\} \tag{4}$$

其中，$Ava(c(\tau_i))$为计算节点 $c(\tau_i)$准备执行任务τ_i的最早时间；$FT(\tau_j)$表示其直接前驱任务τ_j的完成时间，如果τ_j为任务τ_{en}，则标记 $FT(\tau_j)=ST(\tau_j)$。由此可知任务τ_i的完成时间为：

$$FT(\tau_i)=ST(\tau_i)+ET(\tau_i,c(\tau_i)) \tag{5}$$

其中，$ET(\tau_i,c(\tau_i))$表示任务τ_i在计算节点$c(\tau_i)$上的执行时间，如果任务τ_i为τ_{out}，则标记 $FT(\tau_i)=ST(\tau_i)$。由于本系统集群中的计算节点所在的物理机的配置相同，因此可以认为同一个任务τ_i在不同的计算节点上的执行时间是近似相等的[15]。一个工作流G_i的总完成时间表示为：

$$TG_i=\max\{FT(\tau_i)\mid\tau_i\in V\} \tag{6}$$

对于这类工作流，应用 W 集合中所有的工作流完成的总时间为：

$$TW_{total}=\max\{TG_i\mid i\in m\} \tag{7}$$

基于上述定义，本文定义的遥感共性产品生产的应用程序的完成时间的优化调度问题，可形式化表示为：

$$\min\ \max\{FT(\tau_i)\mid\tau_i\in V\} \tag{8}$$

2.2 工作流在具体生产中的应用

首先建立任务之间的依赖关系，并标记$\tau_{en}(oD)$为 1，$\tau_{en}(iD)$为 0，$\tau_{out}(oD)$为 0，$\tau_{out}(iD)$为 1。其余中间算法的$\tau_{com}(iD)$和$\tau_{com}(oD)$都为 1。

单个工作流在生产过程中，主要经过了以下 3 个步骤。

1）首先找到入度为 0 的任务，如果该任务是τ_{en}，则不进行算法的调用生产，然后在依赖关系中找到它的直接后继τ_{com}，并且使该$\tau_{com}(iD)$减 1。

2）再次找到入度为 0 的任务，如果任务是τ_{com}就进行算法的调用生产，然后在依赖关系中找到它的直接后继任务，无论直接后继任务是τ_{com}还是τ_{out}，其任务的入度都进行减 1 操作。

3）重复步骤 2），直至找到某个τ_{com}的直接后继任务是τ_{out}，则不做处理，然后进行最终产品的保存，最后因找不到流程中入度为 0 的任务而结束该工作流。

例如，要处理一个 1A 级数据，根据其去生产某一地区的表观辐亮度产品，构造的工作流如图 2 所示。

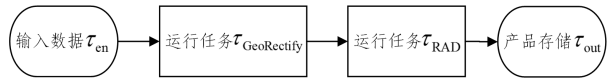


图 2 某地区生产表观辐亮度产品的流程

Fig. 2 Production process of apparent radiance products in a specific region

首先，如果入度为 0 的任务是τ_{en}，则只做算法的数据准备工作，然后找到它的直接后继任务$\tau_{GeoRectify}$，对$\tau_{GeoRectify}(iD)$进行减 1 操作；其次，寻找下一个入度为 0 的任务$\tau_{GeoRectify}$，进行算法调用生产，之后根据依赖关系找到$\tau_{GeoRectify}$的直接后继任务τ_{RAD}并把$\tau_{RAD}(iD)$进行减 1 操作；然后，寻找入度为 0 的任务τ_{RAD}进行算法调用生产，之后不断重复之前的处理，直到找到τ_{out}任务，结束该工作流的生产。

3 基于产品复用模型的任务划分策略

3.1 RTPS 策略的整体架构

基于上述的工作流模型，本章提出一种基于产品复用模型的任务划分策略（RTPS）。首先，将用户提交的大量的工作流按照任务重复度打包成流程包；然后，按照各个流程包的实际调用任务时间，基于负载均衡策略分配流程包和剩余的不相似的零散工作流到各个计算节点上；最后，利用后续提出的产品复用模型来减少重复任务调用，以达到减少工作流总体完成时间的目的。RTPS 策略的架构如图 3 所示。

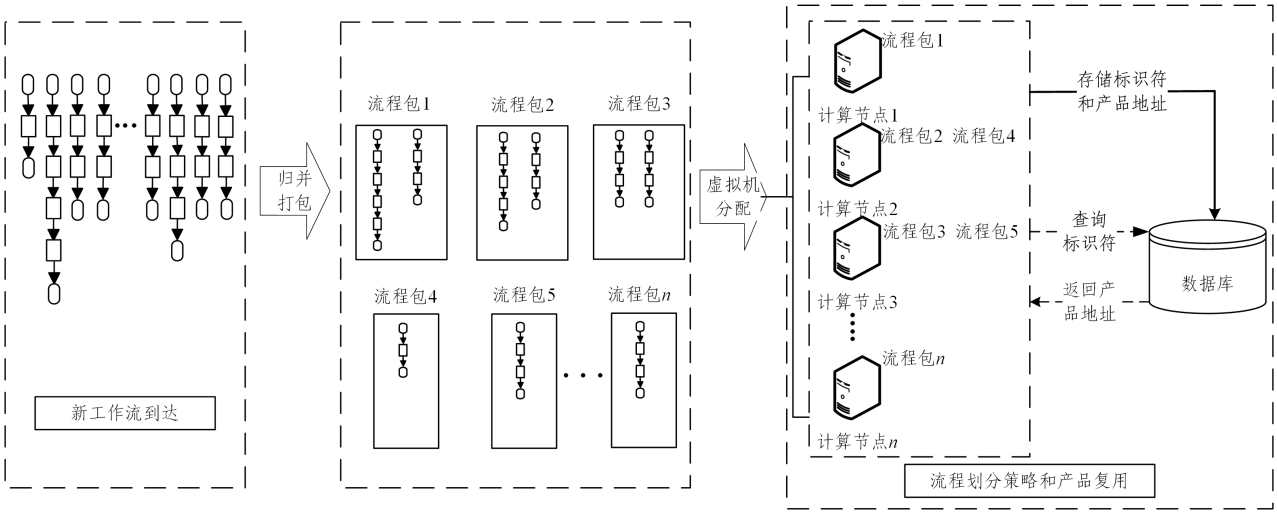


图 3 RTPS 策略架构

Fig. 3 RTPS strategy architecture

3.2 产品复用模型

结合前述工作流模型，本节给出基于工作流的产品复用模型，以更好、更高效地进行产品生产。具体的产品复用流程如图 4 所示。

由图 4 可知，产品复用算法，就是在生产过程中，把生产的数据和算法相关信息组成的一个 XML 文件经过 MD5 加密后形成的哈希值字符串作为标识符，并将其产品地址保存到数据库中，然后在再次生产过程中直接调用标识符所对应的产品地址即可。鉴于当前共性遥感产品生产算法的种类繁多[16]，并且一般由不同的组织进行开发，为了保证在产品复用的生产过程中调用算法和所需数据的信息作为唯一标识文件被保存下来，需要制定统一的文件规范，为所有算法在选择

是否复用之前提供参照标准。该文件类型为可扩展语言XML，具有良好的可扩展性和严格的语法规范，便于实现异构信息的交换，符合该调度算法的实现场景[17]。此处将该文件标识为 Identifier.xml，用于记录不同算法任务生产的重要特征，为后续算法的生产提供支持。后续算法只需根据其生成的哈希值字符串进行对照，即可保证算法调用的独立性。

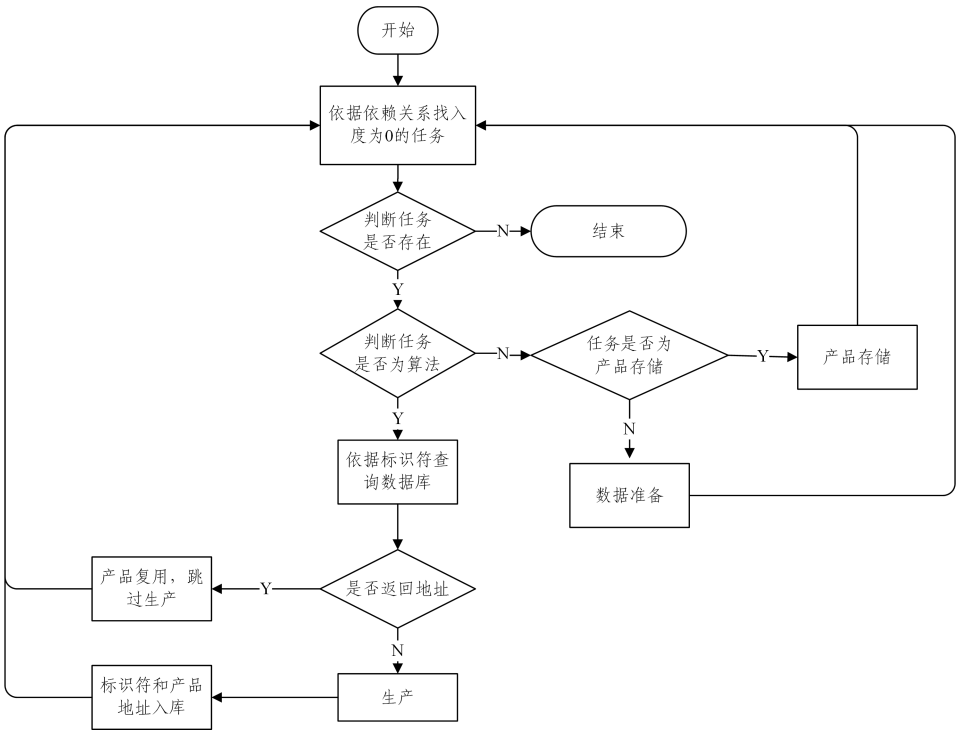


图 4 产品复用流程图

Fig. 4 Product reuse flowchart

Identifier.xml 标签的详细信息定义由表 2 给出，其制作方法如下：

1）选取算法的中英文名、版本号、算法类别、生产的产品类别、算法 EXE 名字以及文件型和字符型的输入参数分别放入 xml 对应的属性中。这是因为仅依靠算法中英文名不足以判断一个算法是否已经生产过，还需要其他辅助的属性来保证生产算法和数据的唯一性。

2）对于输入参数，首先对参数类型进行排序，如果有多个参数类型相同，则按参数类型中的参数值进行排序，以保证顺序的唯一性，进而确保生成的哈希值字符串的唯一性。

表 2 Indentifier.xml 标签的详细信息

Table 2 Detailed information of Indentifier.xml tags

名称	数据类型	说明	示例
AlgorithmName	字符串	算法组件英文名称	LAI
DllName	字符串	算法组件对应的动态链接库文件名，与算法组件英文名称一致	LAI.exe 或者 LAI.dll
ChsName	字符串	算法组件中文名称	叶面积指数
ProductionClass	字符串	算法组件分类	光学卫星 3-5 级产品生产模型
Version	字符串	算法组件版本号（主版本号.次版本号）	2.1
AlgorithmClass	字符串	算法类别	几何类产品_几何精校正产品处理
Parameter		参数定义	可以没有
ParaName	字符串	参数英文名称	tar
ParaChsName	字符串	参数中文名称	输入图片
Description	字符串	参数说明	输入图片
ParaType	字符串	参数类型，枚举值，string 或 int 或 long 或 float 或 double 或 tar.gz	tar.gz
DataType	字符串	数据类型	tar.gz
ParaSource	字符串	参数值来源，枚举值，Man：人工界面输入；Cal：系统自动计算	Man
ParaValue	字符串	参数默认值	0，1，2，3
MaxValue	字符串	数据取值范围上限	输出参数用，仅用于检查图像质量
MinValue	字符串	数据取值范围下限	输出参数用，仅用于检查图像质量
OptionValue	字符串	可选值说明，提示用户如何输入参数值	输入参数用

XML 文件格式如下：

〈? xmlversion＝“1.0”encoding＝“gb2312”? 〉

〈Root〉

〈AlgorithmName〉GeoRectify_S〈/AlgorithmName〉

〈DllName〉GeoRectify.exe〈/DllName〉

〈ChsName 正射产品〈/ChsName〉

〈ProductionClass〉光学卫星 3-5 级产品生产模型〈/AlgorithmDesc〉

〈Version〉1.0〈/Version〉

```
〈AlgorithmClass〉几何类产品_几何精校正产品处理〈/AlgorithmClass〉
〈Inputs ParameterNum="2"〉
〈Parameter〉
〈ParaName〉tar〈/ParaName〉
〈ParaChsName〉输入图片〈/ParaChsName〉
〈Description〉输入图片〈/Description〉
〈ParaType〉tar.gz〈/ParaType〉
〈DataType〉tar.gz〈/DataType〉
〈ParaSource〉Man〈/ParaSource〉〈ParaValue〉\\115.156.212.110\Node\\ftproot\Production\PL_20211219105236_0125-0\Input\ZY302_TMS_E113.3_N29.1_20210818_L1A0000881420.tar.gz〈/ParaValue〉
〈MaxValue〉DEFAULT〈/MaxValue〉
〈MinValue〉DEFAULT〈/MinValue〉
〈OptionValue〉DEFAULT〈/OptionValue〉
〈/Parameter〉
〈Parameter〉
〈ParaName〉InputChannel〈/ParaName〉
〈ParaChsName〉输入通道〈/ParaChsName〉
〈Description〉英文逗号分隔的4元数组〈/Description〉
〈ParaType〉string〈/ParaType〉
〈DataType〉string〈/DataType〉
〈ParaSource〉Man〈/ParaSource〉
〈ParaValue〉0,1,2,3〈/ParaValue〉
〈MaxValue〉DEFAULT〈/MaxValue〉
〈MinValue〉DEFAULT〈/MinValue〉
〈OptionValue〉DEFAULT〈/OptionValue〉
〈/Parameter〉
〈/Inputs〉
〈/Root〉
```

产品复用算法伪代码如算法1所示。

算法1 产品复用算法

输入：w_i，算法对应的 Identifier.xml，Runtime 交互文件

输出：已标记完成的任务集合 w_i，更新后的 Identifier.xml，更新后的 Runtime 交互文件

```
While(在w_i中选一个未完成且入度为0的任务)do
if(该任务为τ_en ‖ 该任务为τ_out)
if(该任务为τ_en)
进行数据准备后，标记τ_en完成
根据依赖关系找到直接后继任务τ_i，τ_i(iD)-1
else
    进行数据存储后，标记τ_out完成
else
查询当前任务执行算法对应的 Identifier.xml 经过 MD5 加密的标识符是否存在
if(存在)
更改该算法对应的 Runtime 交互文件
根据依赖关系找到直接后继任务，将其入度进行减1操作，并标记任务完成
else
确定执行当前任务所在的算法
记录当前算法对应的 Identifier.xml 经过 MD5 加密过的标识符和产品地址
根据依赖关系找到直接后继任务τ_i，τ_i(iD)-1
```

算法1主要是针对工作流任务，不断选取待完成且入度为0的任务并进行判定操作。其中，在选取的任务是 τ_{en} 或者 τ_{out} 的情况下，如果选取的任务是 τ_{en}，则进行数据准备，在数据准备后，标记 τ_{en} 完成；反之，如果是 τ_{out}，就对数据进行存储，标记任务完成。在选取的任务是 τ_{com} 的情况下，需要查找该任务执行算法对应的标识符是否存在。如果存在地址，则表明该数据已经被该算法生产过，可以直接跳过该算法的生产，根据数据库中存在的地址，拉取到本地工作空间即任务文件夹里，同时更改对应的 Runtime 文件中产品的输出路径为现在的产品在工作空间的地址路径，然后找到该任务的直接后续任务，将其入度减1；如果不存在地址，则需要执行当前任务，记录当前任务算法所对应的 Identifier.xml 经过 MD5 加密过的标识符和产品地址，再根据依赖关系找到其直接后继任务，并将其入度进行减1操作，然后选取待完成且入度为0的任务。如此整个循环结束时，则对所有的算法调用生产的结果和算法本身的信息进行共享保存。

下面再通过一个例子具体说明。以与之前相同的1A级产品为例，当图2流程已经生产，则根据其生产地表反射率产品的工作流如图5所示。

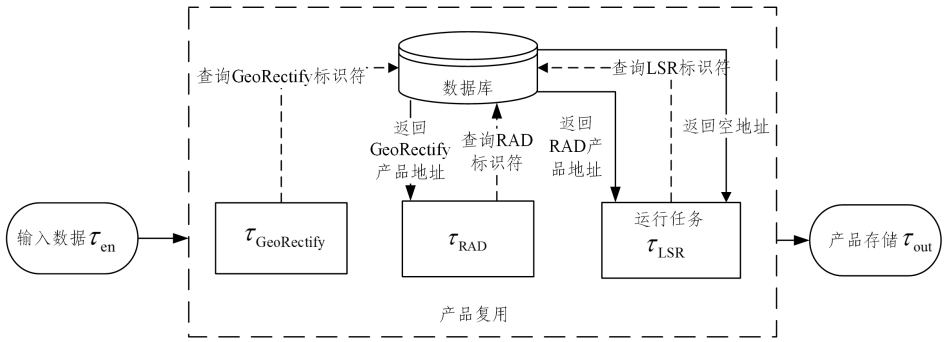


图5 LSR生产工作流

Fig. 5 LSR production workflow

3.3 RTPS 算法

3.3.1 工作流的归并打包阶段

由于项目处理的遥感产品生产的工作流具有顺序特性，例如在同一地区、同一卫星、同一分辨率下，根据该地区所表示的1A级数据来进行表观辐亮度产品生产时，需要经过几何校正、表观辐亮度的特定顺序流程的处理；在进行该地区的

地表反射率的产品生产时，就需要经过几何校正、表观辐亮度、地表反射率的流程处理。因此，在共性产品的生产过程中，在不同的用户提交的多个工作流中，容易出现不同的工作流之间有重复任务生产的情况，如果把重复度较高的工作流分配到不同的计算节点上，会产生数据传输时间的消耗。为了使总体工作流的完成时间最短，借鉴 Dekel Tsur 提出的打包思路[18]，本小节提出一种归并打包方法，将重复度高的任务打包在一块。具体如算法 2 所示。

算法 2 归并打包算法

输入：相似的工作流的包 similarGroup

输出：相似组 similarGroup，包含分好之后的相似的包

```
1. 初始化一个相似 similarGroup，用于存放含有相似的工作流的包
2. for w_i in W do
3. /* 设置一个标志位 foundGroup，并默认设置为 false */
4.         foundGroup=false
5. /* 遍历相似组 similarGroup 中的每个包 Bag */
6.          for Bag in similarGroup do
7. /* 选取包的第一个工作流 w_j */
8.              w_j=Bag[0]
9.          if(w_i的 τ2的标识符及直接后继算法名==w_j的 τ2的标识
            符及直接后继算法名)
10. /* 如果一致，将 w_i添加到 Bag 并将 foundGroup 设为 true */
11.              Bag.add(w_i)
12.              foundGroup=true
13. /* 如果没有找到，创建新的包，将 w_i添加到新的包中并将新包添
    加到 similarGroup 中 */
14.          End if
15.          if(!foundGroup)
16.            初始化一个新的包 NewBag
17.            similarGroup.add(NewBag)
18.          End if
19.        End for
20.    End for
```

工作流的归并打包就是将每个工作流与当前的所有流程包进行重复度对比，如果存在一个流程包中的第一个工作流的第一个任务算法对应的标识符和其直接后继任务的算法名字与其遍历的工作流相同，则将该工作流添加到此流程包中；反之，则为该工作流创建新的流程包，再将下一个工作流与此时的所有流程包进行重复度对比，直至所有工作流对比处理完成。

3.3.2 工作流的划分阶段

打包完成之后，在考虑重复任务的情况下，需要统计每个包实际需要花多少时间来使整个包全部生产完成，在给任务分配计算节点的过程中，先分配工作流包，在此过程中要注意并行性，不能因为需要减少通信开销，而把大量的相似度高的任务放在同一个包中。

为了权衡并行性与包的大小，定义两个阈值 ρ 和 ω（$\rho<\omega$）来控制包中流程的总花费时间。若包中预计执行流程花费时间小于 ρ，则优先向该包中添加其他重复度低的流程；若包中流程预计花费时间大于 ω，则在分配该包时，将该包中花费时间多的流程取出放到其他执行其所在计算节点流程花费时间少的计算节点上，直至要分配的包需执行工作流所花费的时间小于或者等于 ω。

$$\rho=\frac{FFT}{VN}\times\frac{1}{3} \tag{9}$$

$$\omega=\frac{FFT}{VN}\times 2 \tag{10}$$

其中，FFT 为总调用工作流所用时间，VN 为处理器的个数。最终的结果是将执行时间长、重复度低或者非重复的工作流任务优先分配给那些完成包内算法时间较短的工作流包所在的计算节点。相反，对于那些执行所有包内算法时间较长的工作流包所在的计算节点，会减少分配给它的任务，甚至可能不给它分配执行时间短、重复度低或者非重复的工作流任务，以达到节点间的负载均衡。

4 仿真及分析

CloudSim[19-20]是由澳大利亚墨尔本大学网格实验室和 Gridbus 项目推出的用于云计算仿真的软件，其支持大型云计算基础设施的建模与仿真。CloudSim 可以提供模拟的数据中心，通过对资源、任务、调度机制、资源分配进行仿真实现，建立网络资源，可以实现对云计算资源控制及任务调度的有效验证和改进。为验证本文提出的算法的有效性，通过扩展 CloudSim 仿真平台，增加或者修改 DatacenterBroker 以及 CloudletSchedulerSpaceShared 类中的方法实现任务调度算法，通过重新编译、打包实现了算法仿真。

4.1 仿真设置

1）环境配置

硬件环境：Intel(R) Core(TM) i5-7500 CPU @ 3.40 GHz 处理器，16 GB 内存，256 GB 固态硬盘，1 TB 机械硬盘存储。

软件环境：Winodws 10 操作系统，开发平台为 Eclipse-jee-juno-win32-x86_64，仿真工具为 cloudsim-3.0.3，Java 开发环境为 JDK1.8.0。

2）数据中心配置

任务参数：任务长度设置为 2 500～3 000 MIP，期望宽带设置为 1 000 MB/s。

工作流设置：模拟不同卫星和传感器下的 6 类 24 种遥感产品的工作流。用户提交的单个工作流任务数为 1～5。

资源参数：CPU 核心数设置为 1～20，宽带为 1 000 MB/s，虚拟机计算能力为 1 000 MIPS，虚拟机的磁盘映像大小为 10 000 MB，虚拟机内存大小为 512 MB，虚拟机的核心处理数为 1，虚拟机内部的传输时间忽略不计，虚拟机之间的文件传输设置为 250～300 MB/s，主机的内存容量大小为 2 048 MB，主机的存储容量为 1 000 000 Byte，主机的带宽为 10 000 MB/s。

假设任务和资源之间满足以下条件：(1)用户提交的工作流中，任务之间有依赖关系，工作流之间在运行工作流时，若产生可复用的中间产品，则可以在工作流之间共享化，没有产生可复用的中间产品时，工作流之间可以看作相互独立；(2)资源是独占的，而非共有的，即只有在目标任务完成或者失效时，资源方可被重新分配。

3)方案设置

以任务完成时间、加速比、任务的平均响应时间作为性能指标,分别在 CloudSim 上仿真实现 RTPS 算法、先来先服务算法以及短作业优先算法来进行对比实验。考虑到是为了验证任务数量对调度策略产生影响,将任务数量作为实验变量。实验过程中设置 5 台虚拟机,并创建 10 个测试组,分别包含 100,200,300,…,1 000 个工作流,每一个工作流中的任务和通信代价参数随机设置。

4.2 结果分析

从图 6 可以看出,当任务工作流较少时,因为调度时间长、可复用的中间产品少等原因,RTPS 算法与其他调度算法所用的时间差别不大;但是随着任务数量的增加,RTPS 算法所用时间均优于对比算法,而且由于其产品复用和任务划分策略,所用的总任务的完成时间增长率也在慢慢降低,展现出的性能优势也更加明显。

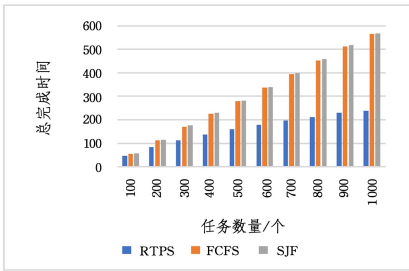


图 6 任务总完成时间

Fig. 6 Total task completion time

从图 7 和图 8 可知,RTPS 算法在平均任务响应度方面均优于 FCFS 和 SJF 算法,但是在加速比方面的性能略低,原因可能是在工作流的打包和后期调整时对虚拟机之间的负载平衡考虑得不够充分。

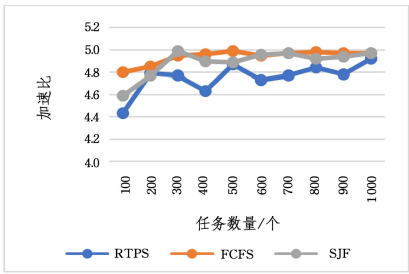


图 7 任务的加速比

Fig. 7 Task speedup ratio

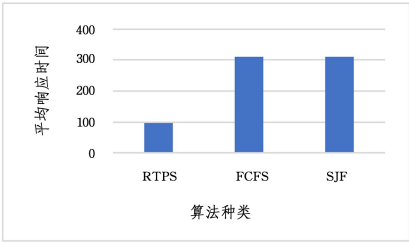


图 8 任务的平均响应时间

Fig. 8 Average task response time

结束语 针对目前遥感共性产品的生产过程中,工作流之间存在因重复计算、数据处理而导致资源浪费和处理效率低下的问题,本文提出了一种基于产品复用模型的任务划分策略。通过在 CloudSim 仿真环境中的应用与对比,本策略展现出了比先来先服务算法、短作业优先算法更为优秀的性能,表现在任务的总完成时间和任务的平均响应时间的显著提升。但是,本文所提策略在对虚拟机分配任务的过程中存在一定程度的不均衡,导致并行效果不佳。因此,下一步的研究重点是如何在工作流划分过程中对流程包进行优化设置,以实现任务分配的均衡,提升并行处理的效率。

参 考 文 献

[1] CHI M, PLAZA A, BENEDIKTSSON J A, et al. Big data for remote sensing: Challenges and opportunities[C]// Proceedings of the IEEE. 2016: 2207-2219.

[2] TONG X D. China's high resolution earth observation system construction of major projects progress [J]. Journal of Remote Sensing, 2016, 29(6): 927-933.

[3] ZHOU B, LI J G, WU G F, et al. A Visual Dataflow Model for Production of Remote Sensing Products[J]. Journal of Henan University(Natural Science Edition), 2013, 43(1): 74-78.

[4] FAN Y, LI B, FAVORITE D, et al. Dras: Deep reinforcement learning for cluster scheduling in high performance computing [J]. IEEE Transactions on Parallel and Distributed Systems, 2022, 33(12): 4903-4917.

[5] WANG X, LI N, GONG G, et al. Load-balancing scheduling of simulation tasks based on a static-dynamic hybrid algorithm[J]. Journal of Simulation, 2022, 16(2): 182-193.

[6] HOFRI M. Disk scheduling: FCFS vs. SSTF revisited[J]. Communications of the ACM, 1980, 23(11): 645-653.

[7] ALWORAFI M A, DHARI A, AL-HASHMI A A, et al. An improved SJF scheduling algorithm in cloud computing environment[C]// 2016 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT). IEEE, 2016: 208-212.

[8] QIU X C, ZANG L, YANG D, et al. Multilevel feedback queue Scheduling Algorithm based on Process execution time [J]. Science Technology and Engineering, 2015, 15(1): 78-83.

[9] ANDERSSON B, BARUAH S, JONSSON J. Static-priority scheduling on multiprocessors[C]// Proceedings 22nd IEEE Real-Time Systems Symposium(RTSS 2001). IEEE, 2001: 193-202.

[10] MIREGURI K, GU Y J. Simulation of LoadBalancing Time Slice Rotation Algorithms in Embedded Operating System[J]. Computer Simulation, 2019, 36(11): 247-250.

[11] SHI Y L, SHEN W M, XIONG W C et al. Research on job schedule and managementsystem for remote sensing data processing with cluster[J]. Computer Engineering and Applications, 2012, 48(25): 77-82.

[12] WU H H. Research and application of task scheduling model in massive remote sensing image common product generation [D].

Zhengzhou:Henan University,2023.

[13] ZHENG F B,ZHANG Z,YU T,et al. Architecture of Remote Sensing Producing Line for Supporting[J]. Computer Science,2012,39(S3):181-184,190.

[14] RAJAVEL R,MALA T. Achieving service level agreement in cloud environment using job prioritization in hierarchical scheduling[C]// Proceedings of the International Conference on Information Systems Design and Intelligent Applications 2012 (INDIA 2012) held in Visakhapatnam, India, January 2012. Springer Berlin Heidelberg,2012:547-554.

[15] QIU Y,JIANG C,WANG Y,et al. Energy aware virtual machine scheduling in data centers[J]. Energies,2019,12(4):646.

[16] LIU Q H,WEN J G,ZHOU X,et al. High resolution remote sensing common product generation and authenticity test technology system[J]. Journal of Remote Sensing,2023,27(3):544-562.

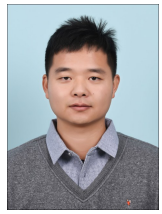
[17] ZHAO J P,CHEN D H,LI H,et al. A Dynamic Integration Framework of GIS System for Remote Sensing Algorithms[J]. Computer Measurement and Control, 2018, 26(7): 186-189, 194.

[18] TSUR D. Faster deterministic algorithms for Co-path Packing and Co-path/cycle Packing[J]. Journal of Combinatorial Optimization,2022,44(5):3701-3710.

[19] HEILIG L,RAJKUMAR B,STEFAN V. Location-aware brokering for consumers in multi-cloud computing environments [J]. Journal of Network and Computer Applications, 2017, 95(10):79-93.

[20] LIU P. Cloud Computing. 2nd Edition [M]. Publishing House of Electronics Industry,2011.

ZUO Xianyu, born in 1979, Ph.D, professor, Ph.D supervisor, is a member of CCF(No. G4801M). His main research interests include parallel computing and remote sensing big data processing.



LIU Cheng, born in 1989, Ph.D, lecturer, is a member of CCF(No. l7262M). His main research interests include pattern recognition and image segmentation.

(责任编辑:何杨)